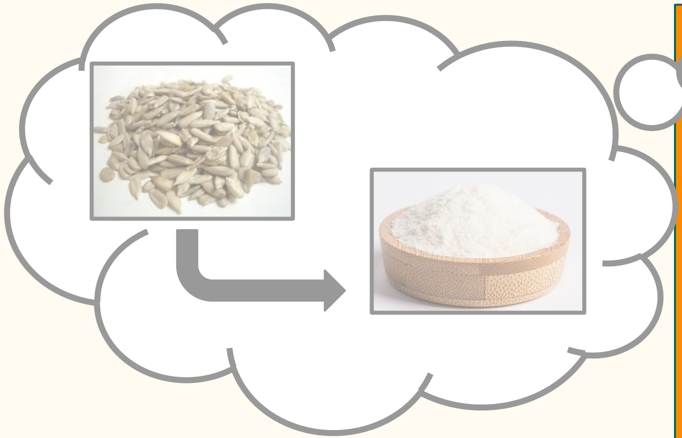
Parte 3

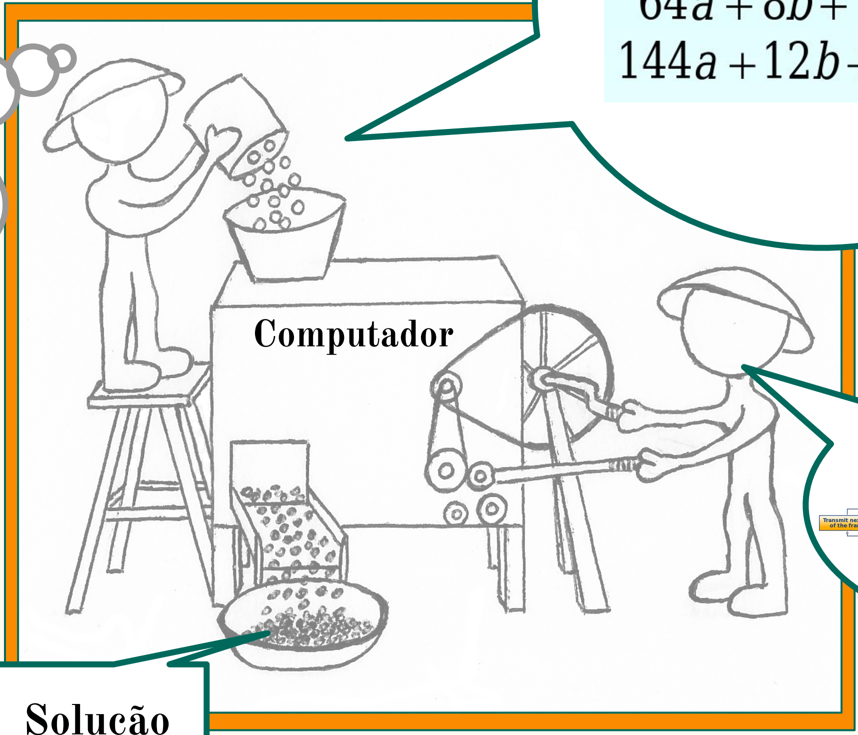
Resolução de Sistemas Lineares - 1

CI1164 - Introdução à Computação Científica

Profs. Armando Delgado e Guilherme Derenievicz
Departamento de Informática - UFPR

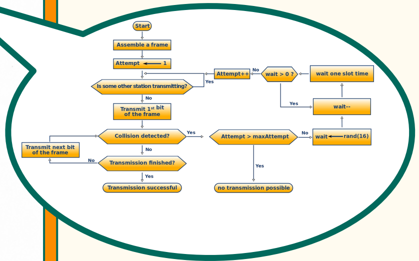
Visão geral da disciplina:





$$
\begin{aligned}
25a + 5b + c &= 106 \\
64a + 8b + c &= 177 \\
144a + 12b + c &= 600
\end{aligned}
$$

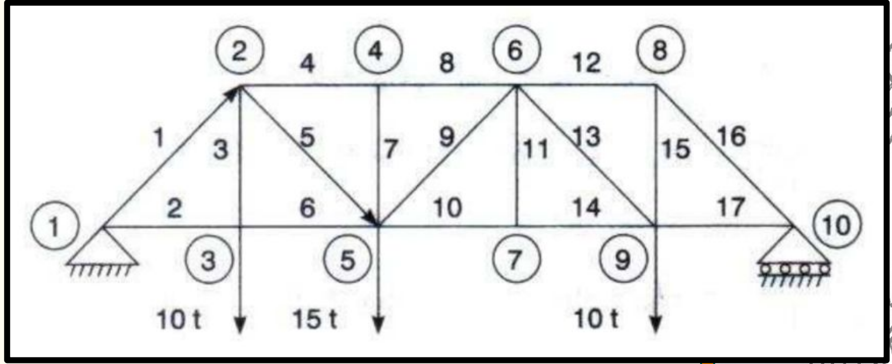
Sistemas Lineares



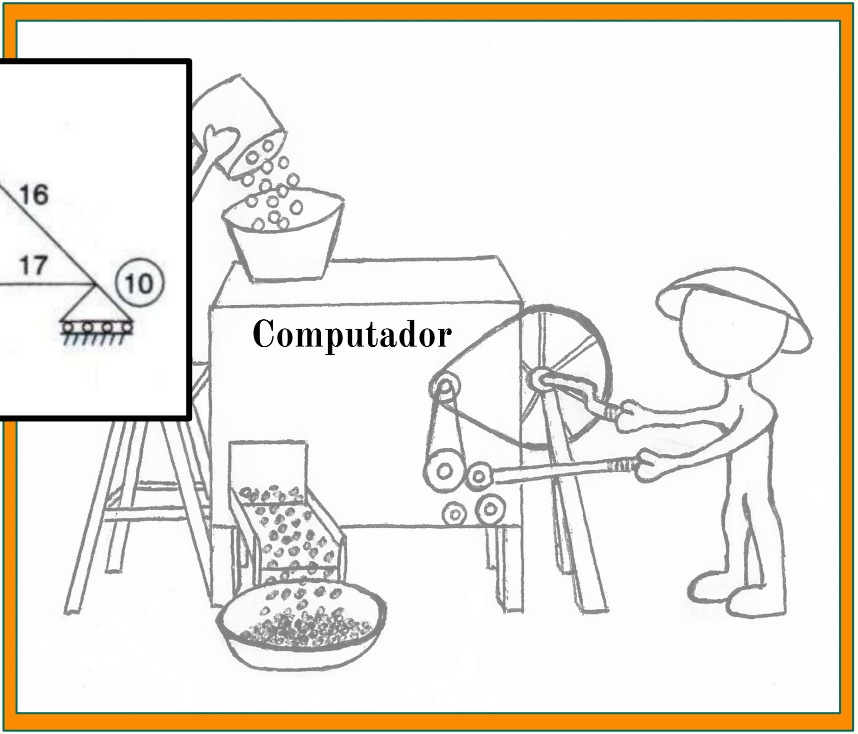
Método Numérico

Resolução de Sistemas Lineares

Fonte: Cálculo Numérico (Ruggiero & Lopes)

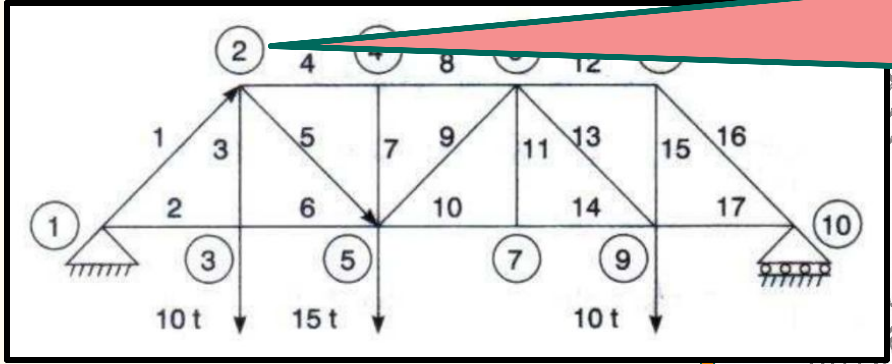


Descrição do Problema

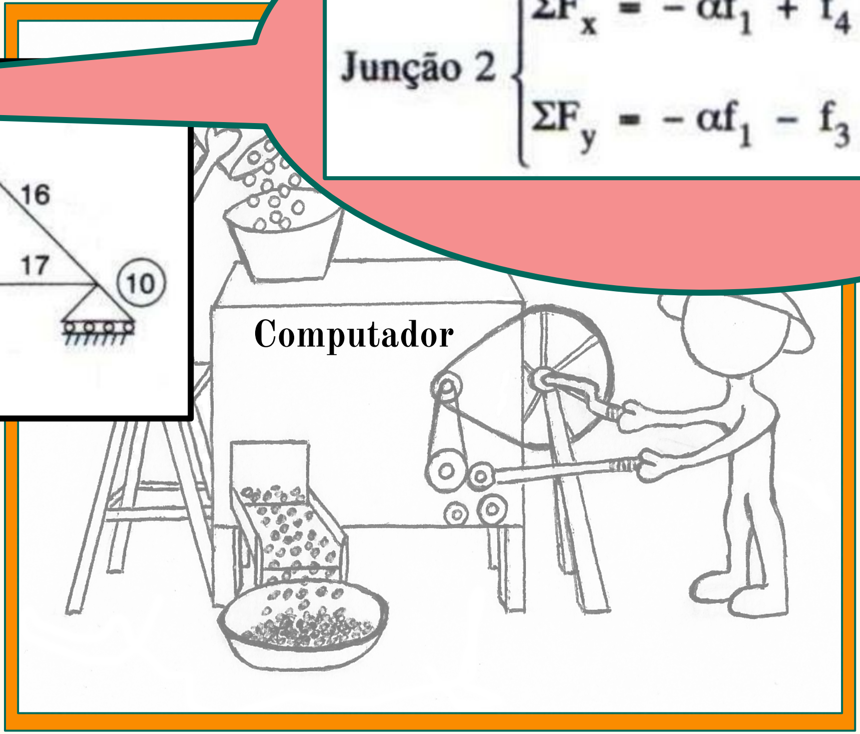


Resolução de Sistemas Lineares

Fonte: Cálculo Numérico (Ruggiero & Lopes)



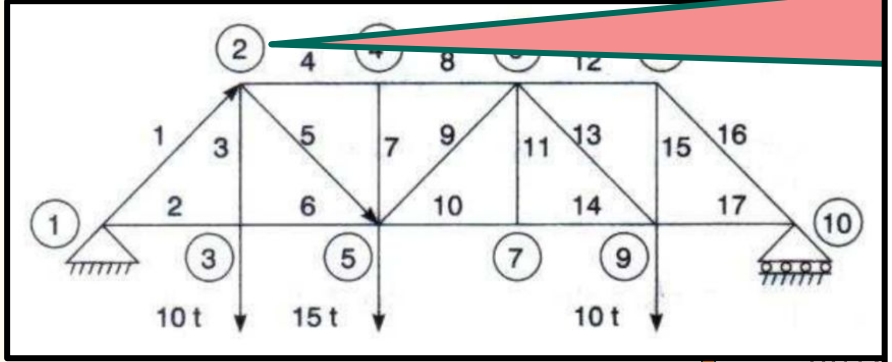
Descrição do Problema

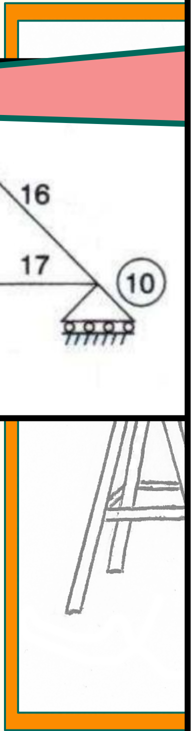


$$\text{Junção 2}\begin{cases}\Sigma F_x = -\alpha f_1 + f_4 + \alpha f_5 = 0\\ \Sigma F_y = -\alpha f_1 - f_3 - \alpha f_5 = 0\end{cases}$$

Resolução de Sistemas Lineares

Fonte: Cálculo Numérico (Ruggiero & Lopes)





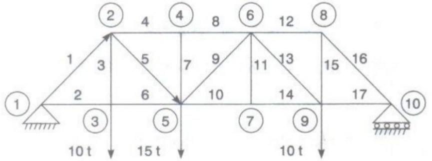
$$
\begin{cases}
-\alpha f_1 + f_4 + \alpha f_5 = 0 \\
-\alpha f_1 - f_3 - \alpha f_5 = 0 \\
- f_2 + f_6 = 0 \\
f_3 - 10 = 0 \\
- f_4 + f_8 = 0 \\
- f_7 = 0 \\
-\alpha f_5 - f_6 + \alpha f_9 + f_{10} = 0 \\
\alpha f_5 + f_7 + \alpha f_9 - 15 = 0 \\
- f_8 - \alpha f_9 + f_{12} + \alpha f_{13} = 0 \\
-\alpha f_9 - f_{11} - \alpha f_{13} = 0 \\
- f_{10} + f_{14} = 0 \\
f_{11} = 0 \\
- f_{12} + \alpha f_{16} = 0 \\
- f_{15} - \alpha f_{16} = 0 \\
-\alpha f_{13} - f_{14} + f_{17} = 0 \\
\alpha f_{13} + f_{15} - f_{10} = 0 \\
-\alpha f_{16} - f_{17} = 0
\end{cases}
$$

$\alpha f_5 = 0$

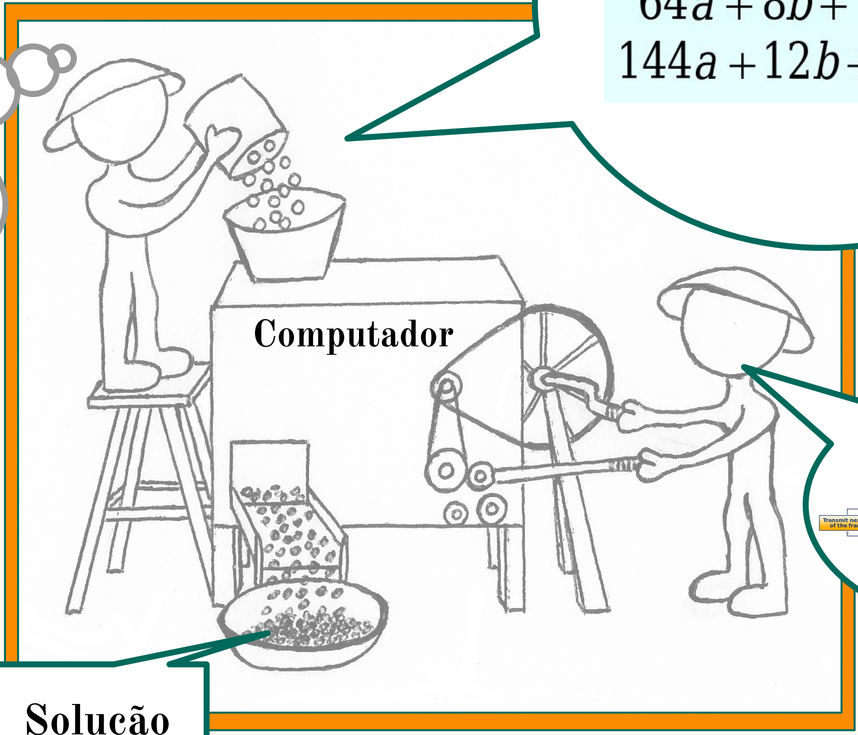
$\alpha f_5 = 0$

Descrição do Problema

Resolução de Sistemas Lineares



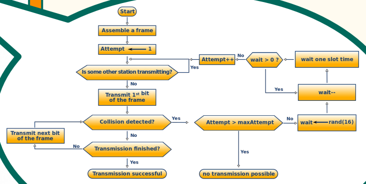
Descrição do Problema



Computador

$$
\begin{aligned}
25a + 5b + c &= 106 \\
64a + 8b + c &= 177 \\
144a + 12b + c &= 600
\end{aligned}
$$

Sistemas Lineares



Método Numérico

Solução

Sistema de Equações Lineares (Sistema Linear)

$$
\begin{cases}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \ldots + a_{1n}x_n = b_1 \\
a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \ldots + a_{2n}x_n = b_2 \\
\vdots \qquad\qquad \vdots \qquad\qquad \vdots \qquad\qquad\qquad \vdots \qquad\quad \vdots \\
a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \ldots + a_{nn}x_n = b_n
\end{cases}
$$

- $\boldsymbol{m}$ equações;
- $\boldsymbol{n}$ incógnitas;
- $\boldsymbol{a_{ij}}$ são os coeficientes;
- $\boldsymbol{b_i}$ são os termos independentes;

Sistema de Equações Lineares (Sistema Linear)

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \ldots + a_{2n}x_n = b_2 \\ \vdots \qquad\quad \vdots \qquad\quad \vdots \qquad\qquad\quad \vdots \qquad\quad \vdots \\ a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \ldots + a_{nn}x_n = b_n \end{cases}$$

- ***m*** equações;
- ***n*** incógnitas;
- $\boldsymbol{a_{ij}}$ são os coeficientes;
- $\boldsymbol{b_i}$ são os termos independentes;

$\boldsymbol{m = n}$

Sistema de Equações Lineares (Sistema Linear)

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \ldots + a_{2n}x_n = b_2 \\ \vdots \qquad \vdots \qquad \vdots \qquad \vdots \qquad \vdots \\ a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \ldots + a_{nn}x_n = b_n \end{cases}$$

Representação Matricial

$$[A] \cdot [X] = [B] \quad \text{ou}$$

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

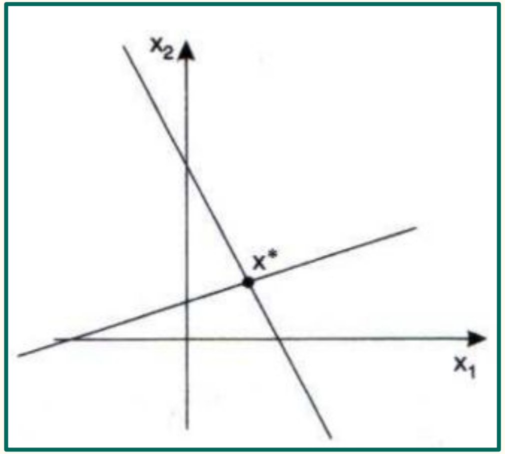
Matriz de Coeficientes — *Vetor das incógnitas ou vetor solução* — *Vetor das constantes*

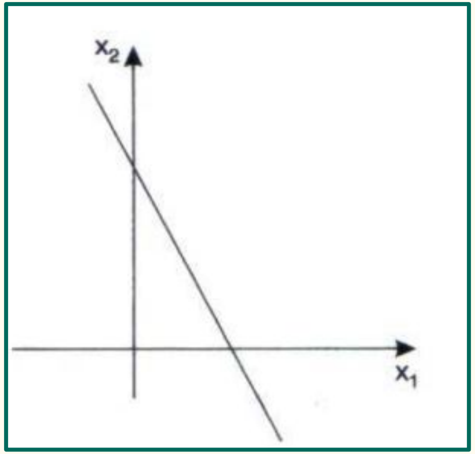
Sistema de Equações Lineares (Sistema Linear)

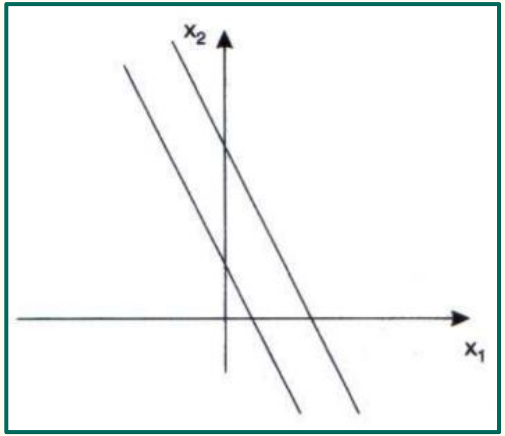
$$\begin{cases} 2x_1 + x_2 = 3 \\ x_1 - 3x_2 = -2 \end{cases}$$

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

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







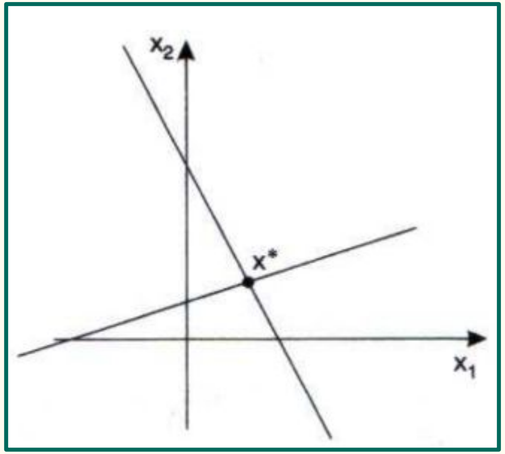
Sistema de Equações Lineares (Sistema Linear)

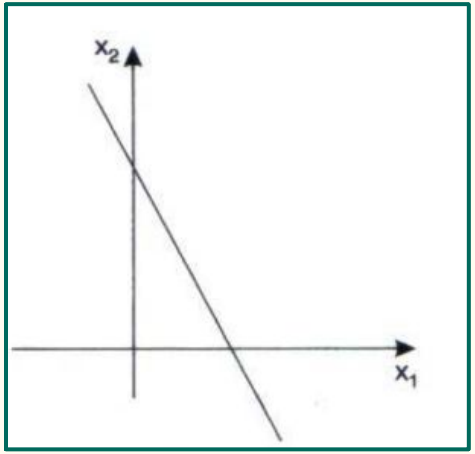
Sistema possível determinado

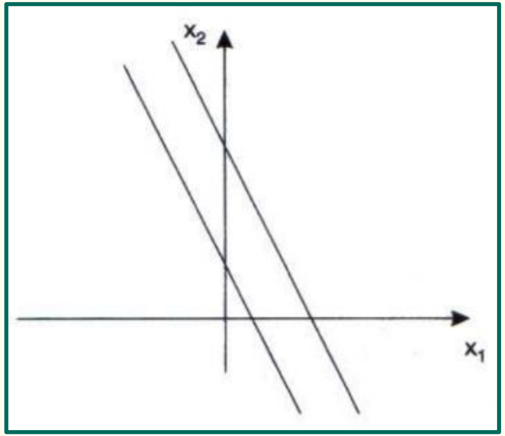
$$\begin{cases} 2x_1 + x_2 = 3 \\ x_1 - 3x_2 = -2 \end{cases}$$

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

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



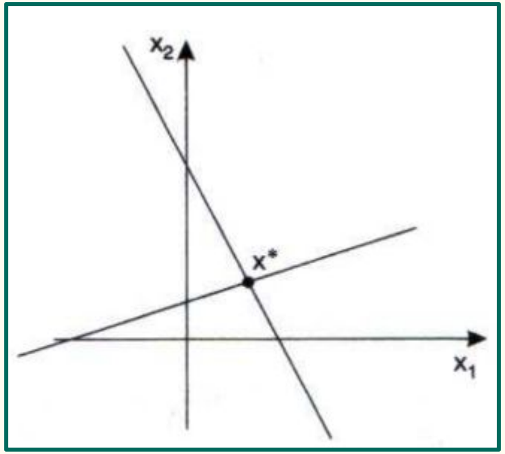




Sistema de Equações Lineares (Sistema Linear)

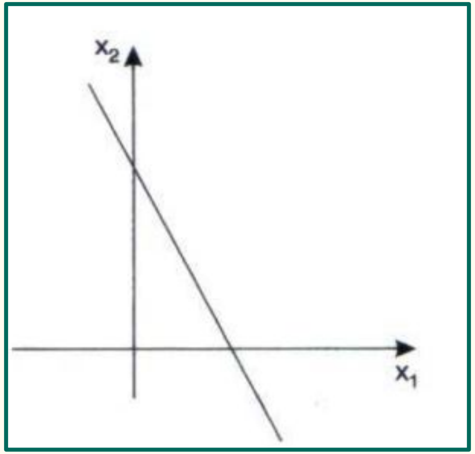
Sistema possível determinado

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

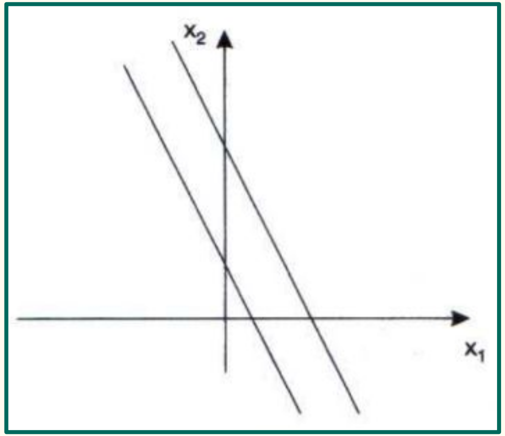


Sistema possível indeterminado

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



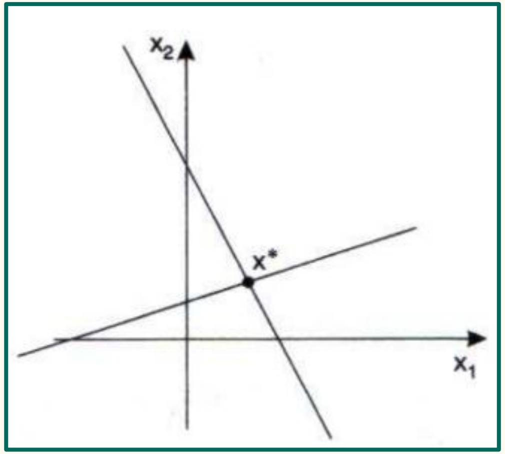
$$\begin{cases} 2x_1 + x_2 = 3 \\ 4x_1 + 2x_2 = 2 \end{cases}$$



Sistema de Equações Lineares (Sistema Linear)

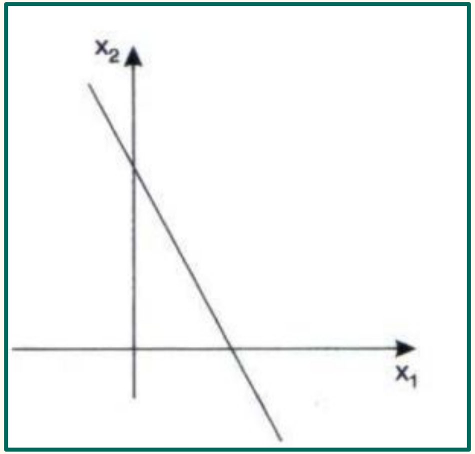
Sistema possível determinado

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



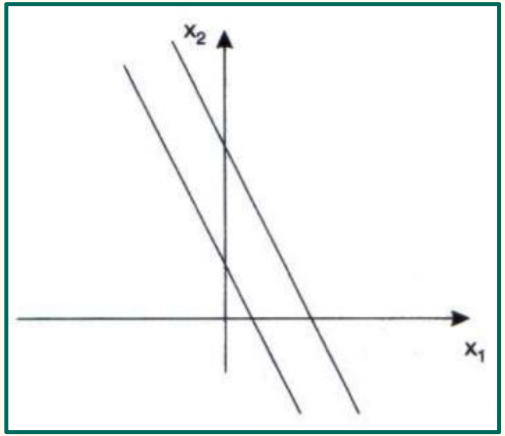
Sistema possível indeterminado

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

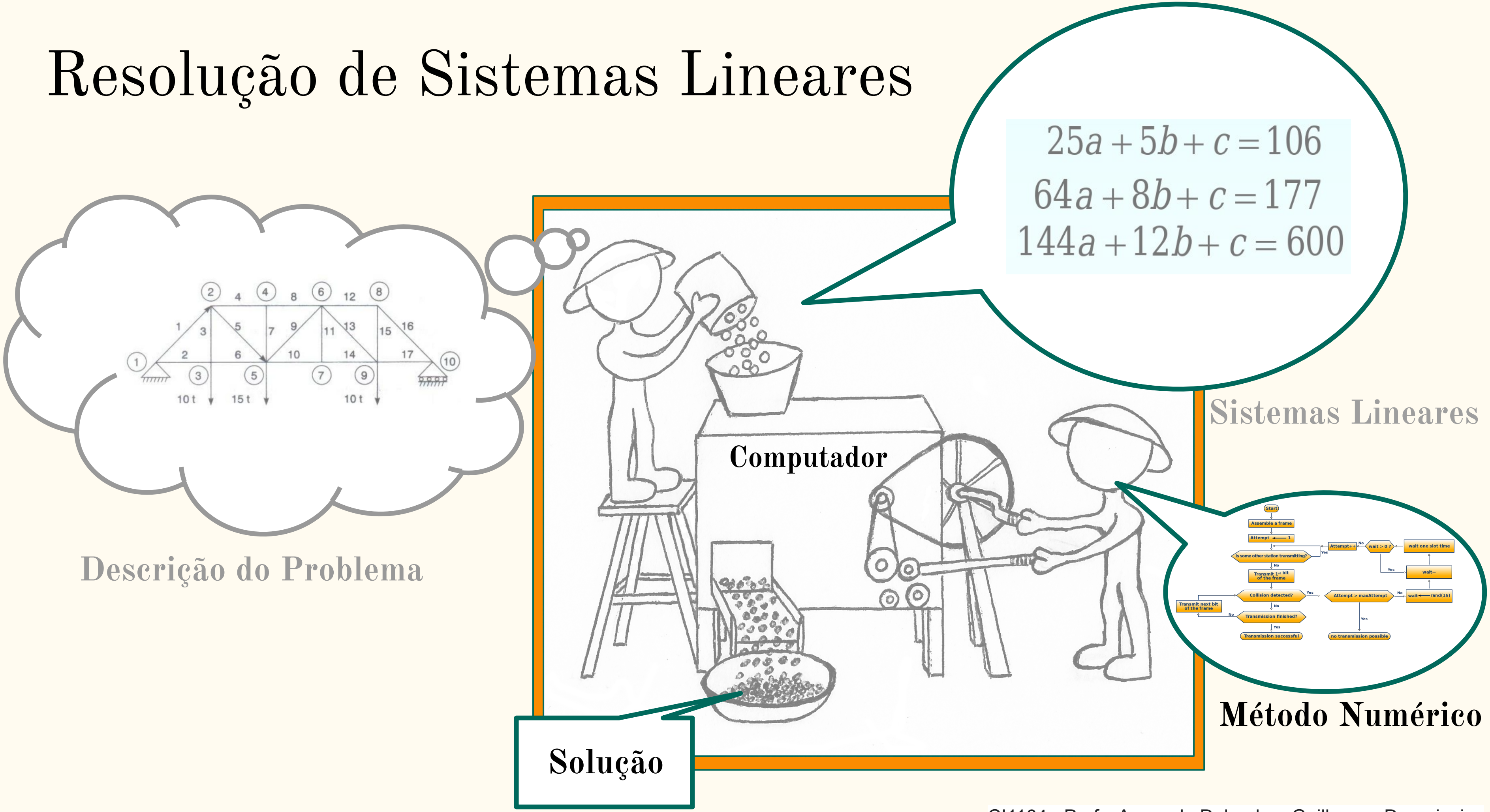


Sistema impossível

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



Resolução de Sistemas Lineares



Métodos de Resolução de Sistemas Lineares

- **Métodos Exatos ou Diretos:** permitiram a solução exata com um número finito de operações, se não fosse por erros numéricos.

- **Métodos Iterativos:** permitem uma solução aproximada através de um processo infinito convergente.

Resolução de Sistemas Triangulares

$$\begin{cases} 2x_1 + 3x_2 - x_3 = 3 \\ x_2 + 2x_3 = 1 \\ 3x_3 = -6 \end{cases}$$

Resolução de Sistemas Triangulares

$$\begin{cases} 2x_1 + 3x_2 - x_3 = 3 \\ \quad\quad\;\; x_2 + 2x_3 = 1 \\ \quad\quad\quad\quad\;\; 3x_3 = -6 \end{cases}$$

Resolução de Sistemas Triangulares

$$\begin{bmatrix} 2 & 3 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \\ -6 \end{bmatrix}$$

$$\begin{cases} 2x_1 + 3x_2 - x_3 = 3 \\ \quad\quad\;\; x_2 + 2x_3 = 1 \\ \quad\quad\quad\quad\;\; 3x_3 = -6 \end{cases}$$

Resolução de Sistemas Triangulares

$$\begin{bmatrix} 2 & 3 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \\ -6 \end{bmatrix}$$

$$\begin{cases} 2x_1 + 3x_2 - x_3 = 3 \\ \quad\quad\;\; x_2 + 2x_3 = 1 \\ \quad\quad\quad\quad\;\; 3x_3 = -6 \end{cases} \Rightarrow x_3 = -2$$

Resolução de Sistemas Triangulares

$$\begin{bmatrix} 2 & 3 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \\ -6 \end{bmatrix}$$

$$\begin{cases} 2x_1 + 3x_2 - x_3 = 3 \\ x_2 + 2x_3 = 1 \\ 3x_3 = -6 \end{cases}$$

$x_2 = 1+4 = 5$

$x_3 = -2$

Resolução de Sistemas Triangulares

$$\begin{bmatrix} 2 & 3 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \\ -6 \end{bmatrix}$$

$$\begin{cases} 2x_1 + 3x_2 - x_3 = 3 \\ \phantom{2x_1 + {}} x_2 + 2x_3 = 1 \\ \phantom{2x_1 + x_2 + {}} 3x_3 = -6 \end{cases}$$

$x_1 = ½(3-15-2) = -7$

$x_2 = 1+4 = 5$

$x_3 = -2$

Resolução de Sistemas Triangulares

$$\begin{bmatrix} 2 & 3 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \\ -6 \end{bmatrix}$$

$$\begin{cases} 2x_1 + 3x_2 - x_3 = 3 \\ \quad\quad\; x_2 + 2x_3 = 1 \\ \quad\quad\quad\quad\;\; 3x_3 = -6 \end{cases}$$

$x_1 = ½(3-15-2) = -7$

$x_2 = 1+4 = 5$

$x_3 = -2$

```
/* Seja um S.L. de ordem 'n'
*/
void retrossubs(double **A, double *b, double *x, uint n) {
    for (int i = n-1; i >= 0; --i) {
        x[i] = b[i];
        for (int j = i+1; j < n; ++j)
            x[i] -= A[i][j] * x[j];
        X[i] /= A[i][i];
    }
}
```

Resolução de Sistemas Triangulares

$$\begin{bmatrix} 2 & 3 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \\ -6 \end{bmatrix}$$

$$\begin{cases} 2x_1 + 3x_2 - x_3 = \\ \quad\quad\quad\; x_2 + 2x_3 \\ \quad\quad\quad\quad\quad\; 3x_3 = \end{cases}$$

```
/* Seja um S.L. de ordem 'n'
*/
void retrossubs(double **A, double
    for (int i = n-1; i >= 0; --i
        x[i] = b[i];
        for (int j = i+1; j < n; +
            x[i] -= A[i][j] * x[j];
        X[i] /= A[i][i];
    }
}
```

<u>Medida de Desempenho:</u>

FLOPS
(*FLoating-point Operations Per Second*)

$0 + 1 + 2 + \ldots + n\text{-}1$ subtrações
$0 + 1 + 2 + \ldots + n\text{-}1$ multiplicações
n divisões

total $= n^2 - n + n = n^2$ FLOPs

Resolução de Sistemas Triangulares

$$\begin{bmatrix} 2 & 3 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \\ -6 \end{bmatrix}$$

$$\begin{cases} 2x_1 + 3x_2 - \ldots = -7 \\ x_2 + \ldots \\ \end{cases}$$

Problema 1: cancelamento subtrativo.

$$\left|\frac{\Delta z}{z}\right| = \frac{|\Delta x| + |\Delta y|}{|x - y|}$$

```
/* Seja um S.L. de ordem 'n'
*/
void retrossubs(double **A,
    for (int i = n-1; i >= 0; --
        x[i] = b[i];
        for (int j = i+1; j < n; ++j)
            x[i] -= A[i][j] * x[j];
        X[i] /= A[i][i];
    }
}
```

Resolução de Sistemas Triangulares

$$\begin{bmatrix} 2 & 3 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \\ -6 \end{bmatrix}$$

$$\begin{cases} 2x_1 + 3x_2 - \ \ldots = -7 \\ \quad\ \ x_2 + \ldots \\ \end{cases}$$

```
/* Seja um S.L. de ordem 'n'
*/
void retrossubs(double **A,
    for (int i = n-1; i >= 0; --
        x[i] = b[i];
        for (int j = i+1; j < n; ++j)
            x[i] -= A[i][j] * x[j];
        X[i] /= A[i][i];
    }
}
```

Problema 1: cancelamento subtrativo.

Problema 2: overflow.

Resolução de Sistemas Triangulares

$$\begin{bmatrix} 2 & 3 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \\ -6 \end{bmatrix}$$

$$\begin{cases} 2x_1 + 3x_2 - \ldots = -7 \\ \quad x_2 + \ldots \\ \end{cases}$$

```
/* Seja um S.L. de ordem 'n'
*/
void retrossubs(double **A,
    for (int i = n-1; i >= 0; --
        x[i] = b[i];
        for (int j = i+1; j
            x[i] -= A[i][j] * x[j];
        X[i] /= A[i][i];
    }
}
```

Problema 1: cancelamento subtrativo.

Problema 2: overflow.

Problema 3: divisão por zero.

Resolução de Sistemas Triangulares

$$\begin{bmatrix} 0 & 3 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} 17 \\ 1 \\ -6 \end{bmatrix}$$

$$\begin{cases} 0x_1 + 3x_2 - x_3 = 17 \Rightarrow x_1 \in \mathbb{R} \\ x_2 + 2x_3 = 1 \Rightarrow x_2 = 1+4 = 5 \\ 3x_3 = -6 \Rightarrow x_3 = -2 \end{cases}$$

```
/* Seja um S.L. de ordem 'n'
*/
void retrossubs(double **A,
    for (int i = n-1; i >= 0; --
        x[i] = b[i];
        for (int j = i+1; j        ++j
            x[i] -= A[i][j] * x[j];
        X[i] /= A[i][i];
    }
}
```

Problema 2: overflow.

Problema 3: divisão por zero.

Resolução de Sistemas Triangulares

$$\begin{bmatrix} 0 & 3 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \\ -6 \end{bmatrix}$$

$$\begin{cases} 0x_1 + 3x_2 - x_3 = 3 & \Rightarrow \quad 17 \neq 3 \\ x_2 + 2x_3 = 1 & \Rightarrow \quad x_2 = 1+4 = 5 \\ 3x_3 = -6 & \Rightarrow \quad x_3 = -2 \end{cases}$$

```
/* Seja um S.L. de ordem 'n'
*/
void retrossubs(double **A,
    for (int i = n-1; i >= 0; --
        x[i] = b[i];
        for (int j = i+1; j
            x[i] -= A[i][j] * x[j];
        X[i] /= A[i][i];
    }
}
```

Problema 2: overflow.

Problema 3: divisão por zero.

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

Dado um Sistema Linear qualquer $\Rightarrow$ obter um Sistema Triangular equivalente

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

$$\begin{array}{rcc}
4x_1+3x_2+x_3=-1 & & (eq_1=eq_1\times 3) & & 12x_1+9x_2+3x_3=-3 \\
2x_1-4x_2+2x_3=3 & \Rightarrow & (eq_2=eq_2\times 2) & \Rightarrow & 4x_1-8x_2+4x_3=6 \\
-x_1+2x_2+3x_3=2 & & (eq_3=eq_3\times -4) & & 4x_1-8x_2-12x_3=-8
\end{array}$$

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

$$\begin{array}{l} 4x_1+3x_2+x_3=-1 \\ 2x_1-4x_2+2x_3=3 \\ -x_1+2x_2+3x_3=2 \end{array} \Rightarrow (eq_2=eq_2-eq_1) \Rightarrow \begin{array}{l} 4x_1+3x_2+x_3=-1 \\ -2x_1-7x_2+x_3=4 \\ -x_1+2x_2+3x_3=2 \end{array}$$

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

$$\begin{array}{l} 4x_1+3x_2+x_3=-1 \\ 2x_1-4x_2+2x_3=3 \\ -x_1+2x_2+3x_3=2 \end{array} \Rightarrow \begin{array}{c} (eq_2=eq_2-eq_1\times 2) \\ (eq_3=eq_3\times -4) \end{array} \Rightarrow \begin{array}{r} 4x_1+3x_2+x_3=-4 \\ -11x_2+3x_3=7 \\ -11x_2-13x_3=-7 \end{array}$$

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

$$\begin{array}{c} 4x_1+3x_2+x_3=-1 \\ 2x_1-4x_2+2x_3=3 \\ 6x_1-12x_2+6x_3=9 \\ -x_1+2x_2+3x_3=2 \end{array} \Rightarrow \quad eq_3=eq_2\times 3 \quad \Rightarrow \begin{array}{c} 4x_1+3x_2+x_3=-1 \\ 2x_1-4x_2+2x_3=3 \\ -x_1+2x_2+3x_3=2 \end{array}$$

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

$$\begin{aligned} 4x_1+3x_2+x_3&=-1\\ 2x_1-4x_2+2x_3&=3\\ 0x_1-0x_2+0x_3&=0\\ -x_1+2x_2+3x_3&=2 \end{aligned} \quad\Rightarrow\quad eq_3 \text{ nula} \quad\Rightarrow\quad \begin{aligned} 4x_1+3x_2+x_3&=-1\\ 2x_1-4x_2+2x_3&=3\\ -x_1+2x_2+3x_3&=2 \end{aligned}$$

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

$$\begin{matrix} 4x_1+3x_2+x_3=-1 \\ 2x_1-4x_2+2x_3=3 \\ 6x_1-x_2+3x_3=2 \\ -x_1+2x_2+3x_3=2 \end{matrix} \Rightarrow \quad eq_3=eq_2+eq_1 \quad \Rightarrow \begin{matrix} 4x_1+3x_2+x_3=-1 \\ 2x_1-4x_2+2x_3=3 \\ -x_1+2x_2+3x_3=2 \end{matrix}$$

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

$$\begin{array}{l} 4x_1+3x_2+x_3=-1 \\ 2x_1-4x_2+2x_3=3 \\ -x_1+2x_2+3x_3=2 \end{array} \Rightarrow \begin{array}{l} -x_1+2x_2+3x_3=2 \\ 2x_1-4x_2+2x_3=3 \\ 4x_1+3x_2+x_3=-1 \end{array} \Rightarrow \begin{array}{l} 3x_3+2x_2-x_1=2 \\ 2x_3-4x_2+2x_1=3 \\ x_3+3x_2+4x_1=-1 \end{array}$$

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

Dado um Sistema Linear qualquer $\Rightarrow$ obter um Sistema Triangular equivalente

$$\begin{aligned} 4x_1 + 3x_2 + x_3 &= -1 \\ 2x_1 - 4x_2 + 2x_3 &= 3 \\ -x_1 + 2x_2 + 3x_3 &= 2 \end{aligned} \quad \Rightarrow \quad \begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= b_1 \\ a_{22}x_2 + a_{23}x_3 &= b_2 \\ a_{33}x_3 &= b_3 \end{aligned}$$

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

Dado um Sistema Linear qualquer $\Rightarrow$ obter um Sistema Triangular equivalente

$$\begin{array}{l} 4x_1+3x_2+x_3=-1 \\ 2x_1-4x_2+2x_3=3 \\ -x_1+2x_2+3x_3=2 \end{array} \quad \Rightarrow \quad \begin{array}{rrrl} a_{11}x_1 & +\,a_{12}x_2 & +\,a_{13}x_3 & =b_1 \\ 1 & a_{22}x_2 & +\,a_{23}x_3 & =b_2 \\ 2 & 3 & a_{33}x_3 & =b_3 \end{array}$$

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

Dado um Sistema Linear qualquer ⇒ obter um Sistema Triangular equivalente

$$\begin{aligned} 4x_1 + 3x_2 + x_3 &= -1 \\ 2x_1 - 4x_2 + 2x_3 &= 3 \\ -x_1 + 2x_2 + 3x_3 &= 2 \end{aligned} \quad \Rightarrow \quad \begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= b_1 \\ a_{22}x_2 + a_{23}x_3 &= b_2 \\ a_{33}x_3 &= b_3 \end{aligned}$$

1 2 4 3 5 6

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

Dado um Sistema Linear qualquer $\Rightarrow$ obter um Sistema Triangular equivalente

$$\begin{aligned} 4x_1+3x_2+x_3&=-1 \\ 2x_1-4x_2+2x_3&=3 \\ -x_1+2x_2+3x_3&=2 \end{aligned} \quad \Rightarrow \quad \begin{aligned} a_{11}x_1+a_{12}x_2+a_{13}x_3&=b_1 \\ a_{22}x_2+a_{23}x_3&=b_2 \end{aligned}$$

1

$$m_{21}=\frac{a_{21}}{a_{11}}$$

$$eq_2=eq_2-m_{21}*eq_1$$

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

Dado um Sistema Linear qualquer ⇒ obter um Sistema Triangular equivalente

$$\begin{aligned} 4x_1 + 3x_2 + x_3 &= -1 \\ 2x_1 - 4x_2 + 2x_3 &= 3 \\ -x_1 + 2x_2 + 3x_3 &= 2 \end{aligned}$$

⇒

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1$$

1

$$a_{21} = a_{21} - \frac{a_{21}}{a_{11}} * a_{11}$$

$$m_{21} = \frac{a_{21}}{a_{11}}$$

$$eq_2 = eq_2 - m_{21} * eq_1$$

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

Dado um Sistema Linear qualquer $\Rightarrow$ obter um Sistema Triangular equivalente

$$\begin{aligned} 4x_1+3x_2+x_3&=-1\\ 2x_1-4x_2+2x_3&=3\\ -x_1+2x_2+3x_3&=2 \end{aligned} \quad \Rightarrow \quad \begin{aligned} a_{11}x_1+a_{12}x_2+a_{13}x_3&=b_1\\ a_{22}x_2+a_{23}x_3&=b_2 \end{aligned}$$

$$m_{31}=\frac{a_{31}}{a_{11}}$$

$$eq_3=eq_3-m_{31}*eq_1$$

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

Dado um Sistema Linear qualquer $\Rightarrow$ obter um Sistema Triangular equivalente

$$\begin{aligned}4x_1+3x_2+x_3&=-1\\2x_1-4x_2+2x_3&=3\\-x_1+2x_2+3x_3&=2\end{aligned} \quad\Rightarrow\quad \begin{aligned}a_{11}x_1+a_{12}x_2+a_{13}x_3&=b_1\\a_{22}x_2+a_{23}x_3&=b_2\end{aligned}$$

1

2

$$m_{k1}=\frac{a_{k1}}{a_{11}}$$

$$eq_k=eq_k-m_{k1}*eq_1$$

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

Dado um Sistema Linear qualquer ⟹ obter um Sistema Triangular equivalente

$$\begin{aligned} 4x_1+3x_2+x_3&=-1\\ 2x_1-4x_2+2x_3&=3\\ -x_1+2x_2+3x_3&=2 \end{aligned} \quad\Longrightarrow\quad \begin{aligned} a_{11}x_1+a_{12}x_2+a_{13}x_3&=b_1\\ a_{22}x_2+a_{23}x_3&=b_2 \end{aligned}$$

1

2

$$m_{k1}=\frac{a_{k1}}{a_{11}}$$

Pivô

$$eq_k=eq_k-m_{k1}*eq_1$$

Método da Eliminação de Gauss

Dois Sistemas Lineares são equivalentes quando possuem a mesma solução.

Dado um Sistema Linear qualquer ⟹ obter um Sistema Triangular equivalente

$$\begin{aligned} 4x_1+3x_2+x_3&=-1\\ 2x_1-4x_2+2x_3&=3\\ -x_1+2x_2+3x_3&=2 \end{aligned} \quad\Longrightarrow\quad \begin{aligned} a_{11}x_1+a_{12}x_2+a_{13}x_3&=b_1\\ a_{22}x_2+a_{23}x_3&=b_2 \end{aligned}$$

1

2

$$m_{k2}=\frac{a_{k2}}{a_{22}}$$

Pivô

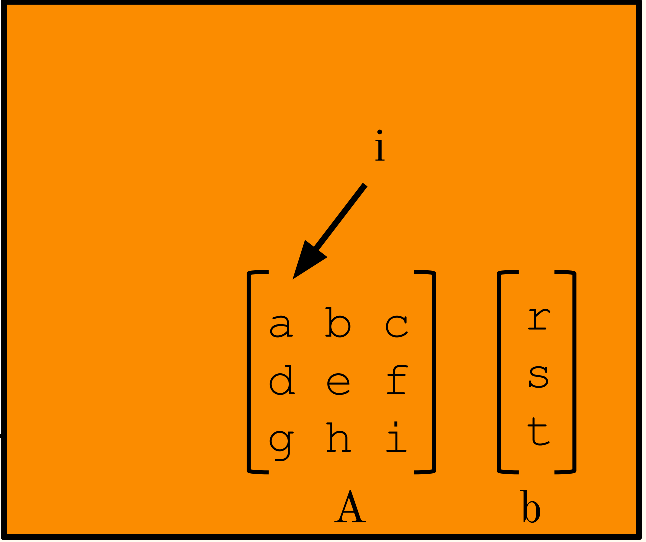
$$\boxed{eq_k=eq_k-m_{k2}*eq_2}$$

Método da Eliminação de Gauss

```c
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```

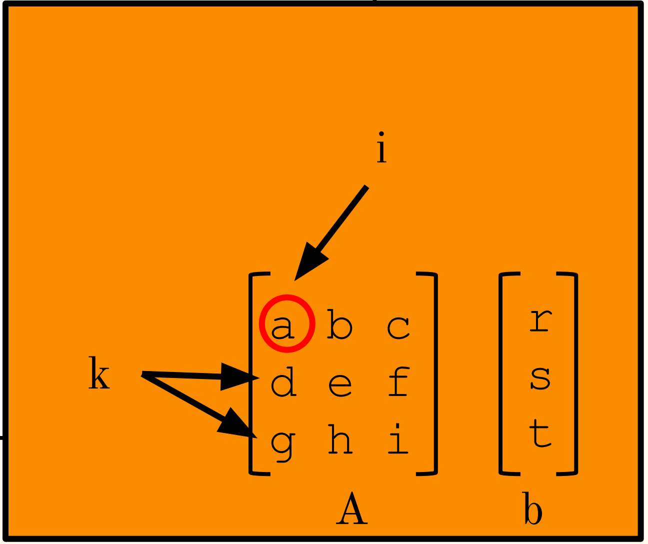
Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```



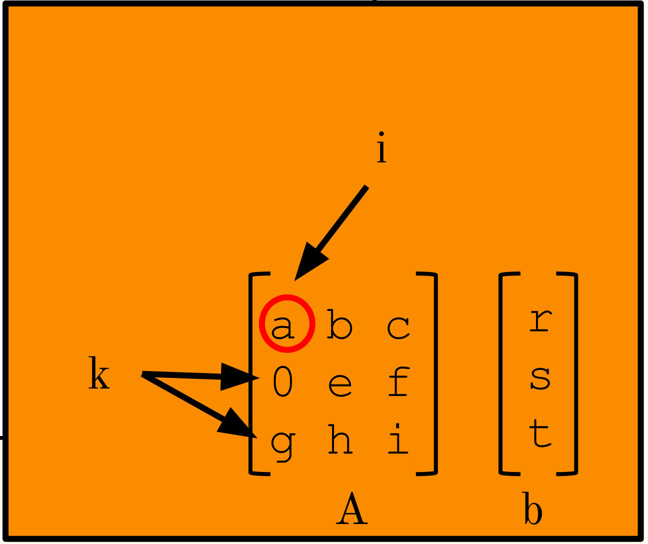
Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```



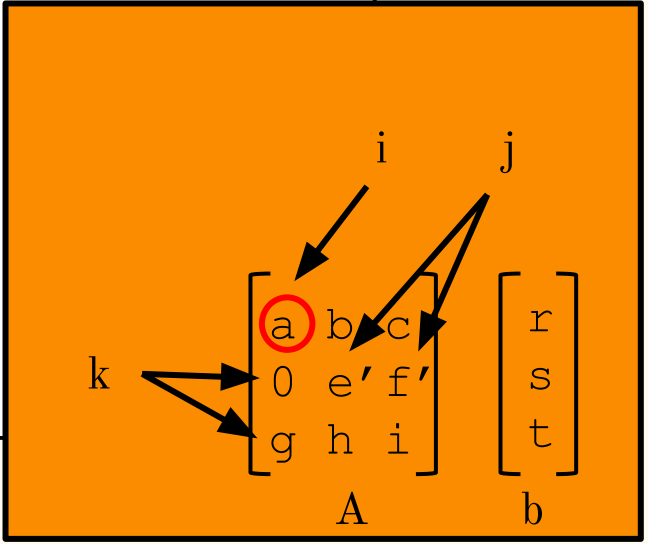
Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```



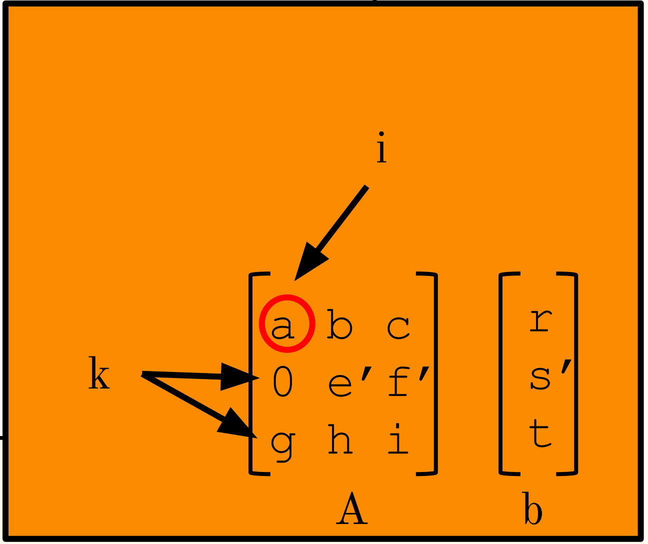
Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```



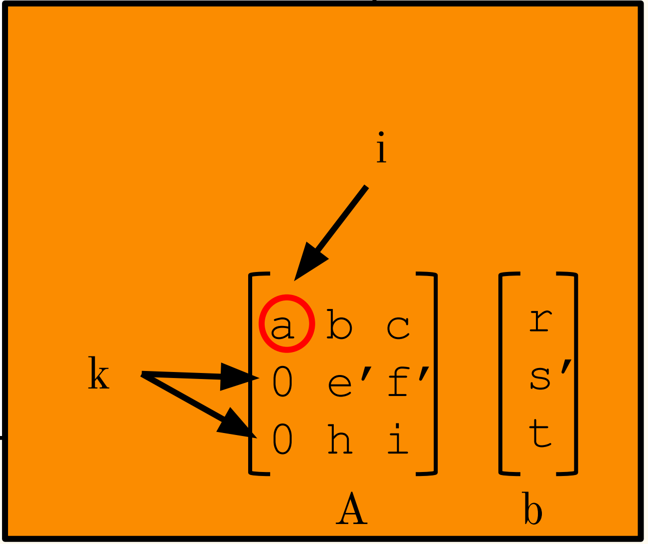
Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```



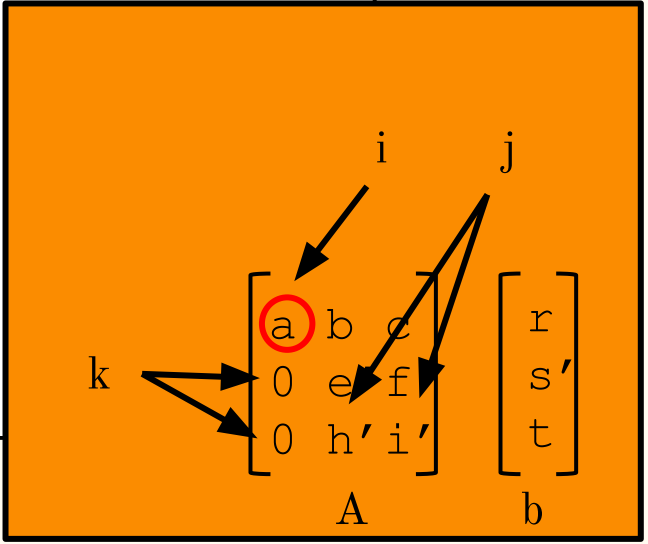
Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```



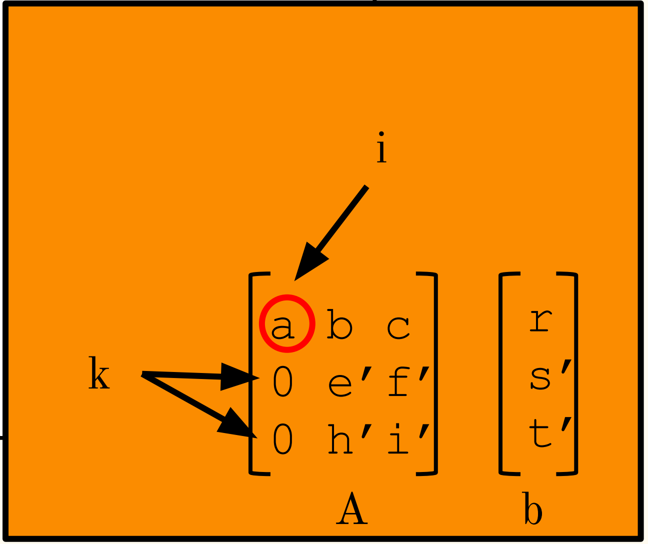
Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```



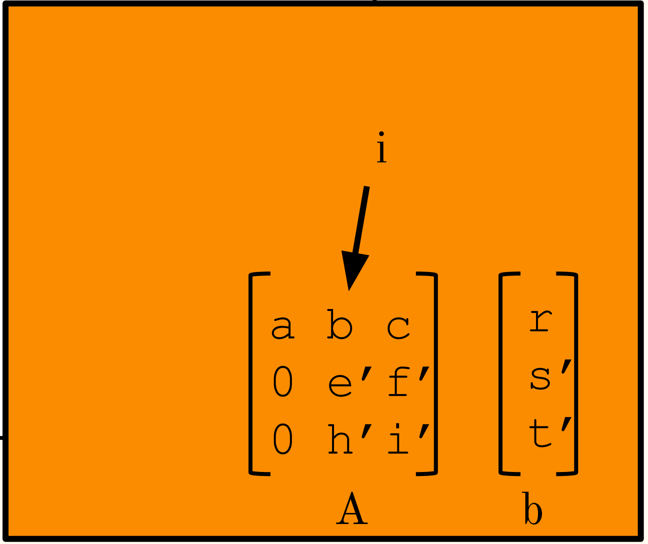
Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```



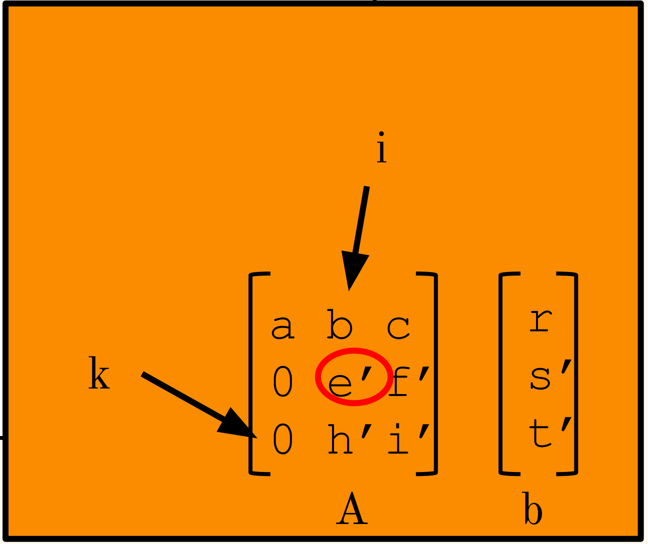
Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```



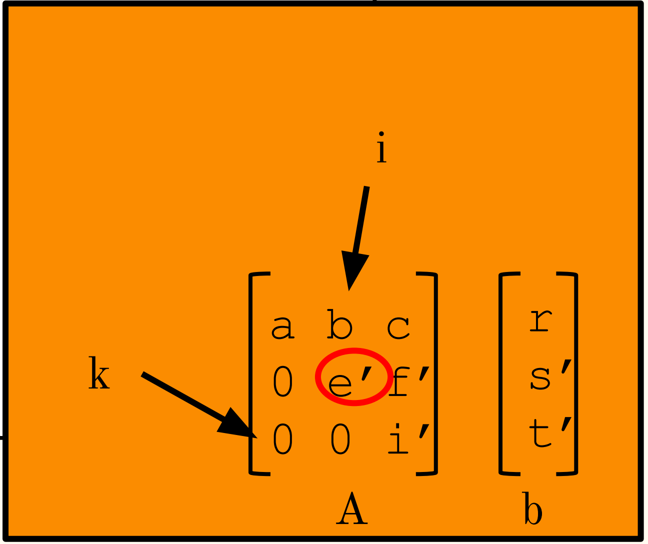
Método da Eliminação de Gauss

```c
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```



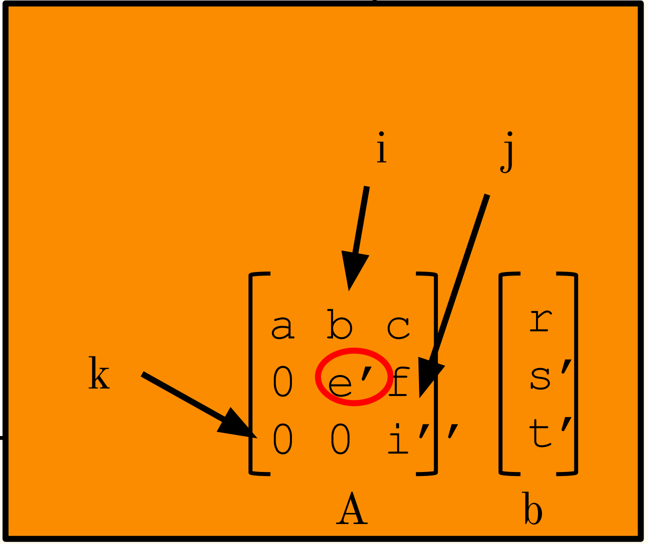
Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```



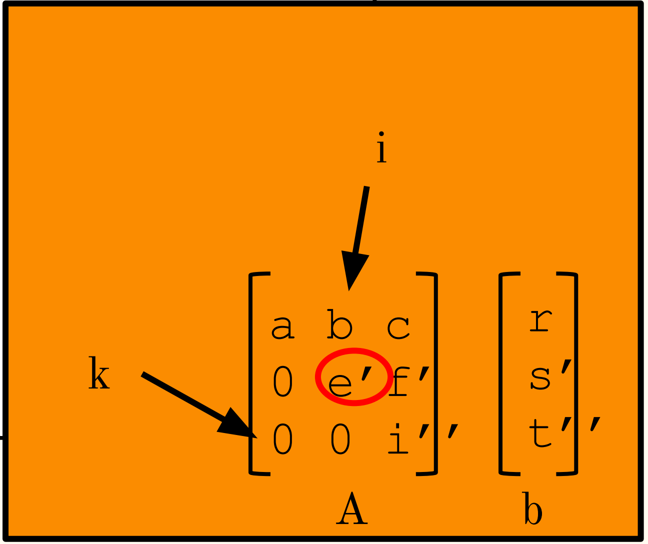
Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```



Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```



Método da Eliminação de Gauss

```c
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```

$$\begin{array}{cc} \begin{array}{ccc} & & i \\ & & \downarrow \end{array} & \\ \left[\begin{array}{ccc} a & b & c \\ 0 & e' & f' \\ 0 & 0 & i' \end{array}\right]', & \left[\begin{array}{c} r \\ s' \\ t' \end{array}\right]' \\ A & b \end{array}$$

Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
```

$$
\begin{array}{llll}
a_{11}x_1 + & a_{12}x_2 + & a_{13}x_3 = & b_1 \\
1 & a_{22}x_2 + & a_{23}x_3 = & b_2 \\
2 & 3 & a_{33}x_3 = & b_3
\end{array}
$$

A[1][0] = A[1][0] - A[0][0] * (A[1][0] / A[0][0])
A[1][0] = 0

Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
```

$$
\begin{array}{lllr}
a_{11}x_1 & +a_{12}x_2 & +a_{13}x_3 & =b_1 \\
1 & a_{22}x_2 & +a_{23}x_3 & =b_2 \\
2 & 3 & a_{33}x_3 & =b_3
\end{array}
$$

A[1][0] = A[1][0] - A[0][0] * (A[1][0] / A[0][0])
A[1][0] = 0 + ERROS

Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```

$n - (i + 1)$ subtrações
$n - (i + 1)$ multiplicações

Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```

$n - (i+1)$ subtrações
$n - (i+1)$ multiplicações

1 subtração
1 multiplicação

Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```

1 divisão

$n - (i+1)$ subtrações
$n - (i+1)$ multiplicações

1 subtração
1 multiplicação

Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```

$n - (i + 1)$

1 divisão

$n - (i + 1)$ subtrações
$n - (i + 1)$ multiplicações

1 subtração
1 multiplicação

Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```

n - (i + 1)

1 divisão

n - (i + 1) subtrações
n - (i + 1) multiplicações

1 subtração
1 multiplicação

= (n - i - 1) * (2(n - i) + 1)
= $2i^2$ + i - 4ni + $2n^2$ - n - 1

Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```

$\sum_{i=0}^{n-1}$

n - (i + 1)

1 divisão

n - (i + 1) subtrações
n - (i + 1) multiplicações

1 subtração
1 multiplicação

$= (n - i - 1) * (2(n - i) + 1)$

$= 2i^2 + i - 4ni + 2n^2 - n - 1$

Método da Eliminação de Gauss

$$\sum_{i=0}^{n-1}$$

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
     for(int k=i+1; k < n; ++k) {
       ...] / A[i][i];
       ... n; ++j)
       ...[j] * m;
```

1 divisão

$n - (i + 1)$ subtrações
$n - (i + 1)$ multiplicações

$$= (n - i - 1) * (2(n - i) + 1)$$
$$= 2i^2 + i - 4ni + 2n^2 - n - 1$$

$$= \sum_{i=0}^{n-1} (2i^2 + i - 4ni) + n(2n^2 - n - 1)$$

$$= 2\sum_{i=0}^{n-1} i^2 + \sum_{i=0}^{n-1} i - 4n\sum_{i=0}^{n-1} i + 2n^3 - n^2 - n$$

$$= \frac{2n^3}{3} + \frac{n^2}{2} - \frac{7n}{6} \quad \text{FLOPs}$$

Método da Eliminação de Gauss

Problema 1: cancelamento subtrativo.

```
n'

le **A, double *b, uint n ){
rtir da primeira */
for (int i        n; ++i) {
    for(int       k < n; ++k) {
        double    A[k][i] / A[i][i];
        A[k][i]    .0;
        for(int j=  +1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
        b[k] -= b[i] * m;
    }
}
}
```

Método da Eliminação de Gauss

```
n'

le **A, double *b, uint n ){
rtir da primeira */
for (int i        n; ++i) {
    for(int       k < n; ++k) {
        double     A[k][i] / A[i][i];
        A[k][i]    .0;
        for(int j= +1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
        b[k] -= b[i] * m;
    }
}
```

Problema 1: cancelamento subtrativo.

Problema 2: overflow.

Método da Eliminação de Gauss

```
n'

le **A, double *b, uint n ){
rtir da primeira */
for (int i        n; ++i) {
    for(int       k < n; ++k) {
        double     A[k][i] / A[i][i];
        A[k][i] =   .0;
        for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
        b[k] -= b[i] * m;
    }
}
```

Problema 1: cancelamento subtrativo.

Problema 2: overflow.

Problema 3: divisão por zero.

Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
```

ema 3: divisão por zero.

$$\begin{bmatrix} 0 & 3 & -1 \\ 1 & 1 & 2 \\ 3 & 2 & -1 \end{bmatrix} \begin{bmatrix} 4 \\ 7 \\ 2 \end{bmatrix}$$

$$\begin{cases} 3x_2 - x_3 = 4 \\ x_1 + x_2 + 2x_3 = 7 \\ 3x_1 + 2x_2 - x_3 = 2 \end{cases}$$

Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
```

ema 3: divisão por zero.

$$\begin{bmatrix} 0 & 3 & -1 \\ 1 & 1 & 2 \\ 3 & 2 & -1 \end{bmatrix} \begin{bmatrix} 4 \\ 7 \\ 2 \end{bmatrix}$$

$$\begin{cases} \quad\;\; 3x_2 - x_3 = 4 \\ x_1 + x_2 + 2x_3 = 7 \\ 3x_1 + 2x_2 - x_3 = 2 \end{cases}$$

Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
```

ema 3: divisão por zero.

$$\begin{bmatrix} 3 & 2 & -1 \\ 1 & 1 & 2 \\ 0 & 3 & -1 \end{bmatrix} \begin{bmatrix} 2 \\ 7 \\ 4 \end{bmatrix}$$

$$\begin{cases} 3x_1 + 2x_2 - x_3 = 2 \\ x_1 + x_2 + 2x_3 = 7 \\ 3x_2 - x_3 = 4 \end{cases}$$

Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
    /* para cada linha a partir da primeira */
    for (int i=0; i < n; ++i) {
        for(int k=i+1; k < n; ++k) {
            double m = A[k][i] / A[i][i];
            A[k][i] = 0.0;
            for(int j=i+1; j < n; ++j)
                A[k][j] -= A[i][j] * m;
            b[k] -= b[i] * m;
```

```
encontraMax(A, i);
```

ema 3: divisão por zero.

$$\begin{bmatrix} 3 & 2 & -1 \\ 1 & 1 & 2 \\ 0 & 3 & -1 \end{bmatrix} \begin{bmatrix} 2 \\ 7 \\ 4 \end{bmatrix}$$

$$\begin{cases} 3x_1 + 2x_2 - x_3 = 2 \\ x_1 + x_2 + 2x_3 = 7 \\ 3x_2 - x_3 = 4 \end{cases}$$

Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
```

```
encontraMax(A, i);
```

Pivoteamento Parcial

...ema 3: divisão por zero.

$$\begin{bmatrix} 3 & 2 & -1 \\ 1 & 1 & 2 \\ 0 & 3 & -1 \end{bmatrix} \begin{bmatrix} 2 \\ 7 \\ 4 \end{bmatrix}$$

$$\begin{cases} 3x_1 + 2x_2 - x_3 = 2 \\ x_1 + x_2 + 2x_3 = 7 \\ 3x_2 - x_3 = 4 \end{cases}$$

Estratégia de Pivoteamento

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      uint iPivo = encontraMax(A, i);
      if ( i != iPivo )
         trocaLinha( A, b, i, iPivo );

      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```

Pivoteamento Parcial

Estratégia de Pivoteamento

$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

Solução Exata: $x_1 = 10$

$x_2 = 1$

Estratégia de Pivoteamento

$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

Solução Exata: $x_1 = 10$
$x_2 = 1$

Considerando 4 dígitos e arredondamento (**sem** pivoteamento parcial):

Estratégia de Pivoteamento

$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

Solução Exata: $x_1 = 10$

$x_2 = 1$

Considerando 4 dígitos e arredondamento (**sem** pivoteamento parcial):

$$m = 0.423/0.004 = 105.75 \rightarrow 105.8$$

Estratégia de Pivoteamento

$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

Solução Exata: $x_1 = 10$

$x_2 = 1$

Considerando 4 dígitos e arredondamento (**sem** pivoteamento parcial):

$$m = 0.423/0.004 = 105.75 \rightarrow 105.8$$

$$\begin{bmatrix} 0.004 & 15.73 \\ 0 & (-24.72-105.8*15.73) \end{bmatrix} \begin{bmatrix} 15.77 \\ -20.49-105.8*15.77 \end{bmatrix}$$

Estratégia de Pivoteamento

$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

Solução Exata: $x_1 = 10$
$x_2 = 1$

Considerando 4 dígitos e arredondamento (**sem** pivoteamento parcial):

$m = 0.423/0.004 = 105.75 \rightarrow 105.8$

$$\begin{bmatrix} 0.004 & 15.73 \\ 0 & (-24.72-\underbrace{105.8*15.73}_{1664.234}) \end{bmatrix} \begin{bmatrix} 15.77 \\ -20.49-\underbrace{105.8*15.77}_{1668.466} \end{bmatrix}$$

Estratégia de Pivoteamento

$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

Solução Exata: $x_1 = 10$, $x_2 = 1$

Considerando 4 dígitos e arredondamento (**sem** pivoteamento parcial):

$$m = 0.423/0.004 = 105.75 \rightarrow 105.8$$

$$\left[\begin{array}{cc} 0.004 & 15.73 \\ 0 & \underbrace{(-24.72-\underbrace{105.8*15.73}_{1664.234})}_{-1688.72} \end{array}\right]\left[\begin{array}{c} 15.77 \\ \underbrace{-20.49-\underbrace{105.8*15.77}_{1668.466}}_{-1688.49} \end{array}\right]$$

Estratégia de Pivoteamento

$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

Solução Exata: $x_1 = 10$, $x_2 = 1$

Considerando 4 dígitos e arredondamento (**sem** pivot

$$\left[\begin{array}{cc} 0.004 & 15.73 \\ 0 & -1689 \end{array}\right] \left[\begin{array}{c} 15.77 \\ -1688 \end{array}\right]$$

$$m = 0.423/0.004 = 105.75 \rightarrow 105.8$$

$$\left[\begin{array}{cc} 0.004 & 15.73 \\ 0 & (\underbrace{-24.72-\underbrace{105.8*15.73}_{1664.234}}_{-1688.72}) \end{array}\right] \left[\begin{array}{c} 15.77 \\ \underbrace{-20.49-\underbrace{105.8*15.77}_{1668.466}}_{-1688.49} \end{array}\right]$$

Estratégia de Pivoteamento

$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

Solução Exata: $x_1 = 10$

$x_2 = 1$

Considerando 4 dígitos e arredondamento (**sem** pivote

$$\begin{bmatrix} 0.004 & 15.73 \\ 0 & -1689 \end{bmatrix} \begin{bmatrix} 15.77 \\ -1688 \end{bmatrix}$$

Solução: $x_2 = -1688/-1689 = 0.9994$

Estratégia de Pivoteamento

$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

Solução Exata: $x_1 = 10$

$x_2 = 1$

Considerando 4 dígitos e arredondamento (**sem** pivot

$$\begin{bmatrix} 0.004 & 15.73 \\ 0 & -1689 \end{bmatrix} \begin{bmatrix} 15.77 \\ -1688 \end{bmatrix}$$

Solução: $x_2 = -1688/-1689 = 0.9994$

$x_1 = (15.77 - 15.73*0.9994)/0.004$

$= (15.77 - 15.72)/0.004 = 12.50$

Estratégia de Pivoteamento

$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

Solução Exata: $x_1 = 10$
$x_2 = 1$

Considerando 4 dígitos e arredondamento (**sem** pivot

$$\begin{bmatrix} 0.004 & 15.73 \\ 0 & -1689 \end{bmatrix} \begin{bmatrix} 15.77 \\ -1688 \end{bmatrix}$$

Solução:

$$x_2 = -1688/-1689 = 0.9994$$

$$x_1 = (15.77 - 15.73*0.9994)/0.004$$
$$= (15.77 - 15.72)/0.004 = 12.50$$

$$ER_{x_1} = 25\%$$

Estratégia de Pivoteamento

$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

Solução Exata: $x_1 = 10$

$x_2 = 1$

Considerando 4 dígitos e arredondamento (**com** pivoteamento parcial):

Estratégia de Pivoteamento

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0.004x_1 + 15.73x_2 = 15.77 \end{cases}$$

Solução Exata: $x_1 = 10$
$x_2 = 1$

Considerando 4 dígitos e arredondamento (**com** pivoteamento parcial):

Estratégia de Pivoteamento

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0.004x_1 + 15.73x_2 = 15.77 \end{cases}$$

Solução Exata: $x_1 = 10$
$x_2 = 1$

Considerando 4 dígitos e arredondamento (**com** pivoteamento parcial):

$$m = 0.004/0.423 = 0.009456265 \rightarrow 0.009456$$

Estratégia de Pivoteamento

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0.004x_1 + 15.73x_2 = 15.77 \end{cases}$$

Solução Exata: $x_1 = 10$
$x_2 = 1$

Considerando 4 dígitos e arredondamento (**com** pivoteamento parcial):

$$m = 0.004/0.423 = 0.009456265 \rightarrow 0.009456$$

$$\left[\begin{array}{ll} 0.423 & -24.72 \\ 0 & (15.73-0.009456*(-24.72)) \end{array}\right] \left[\begin{array}{l} -20.49 \\ 15.77-0.009456*(-20.49) \end{array}\right]$$

Estratégia de Pivoteamento

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0.004x_1 + 15.73x_2 = 15.77 \end{cases}$$

Solução Exata: $x_1 = 10$

$x_2 = 1$

Considerando 4 dígitos e arredondamento (**com** pivoteamento parcial):

$$m = 0.004/0.423 = 0.009456265 \rightarrow 0.009456$$

$$\begin{bmatrix} 0.423 & -24.72 \\ 0 & (15.73 \underbrace{-0.009456*(-24.72)}_{-0.23375232}) \end{bmatrix} \begin{bmatrix} -20.49 \\ 15.77 \underbrace{-0.009456*(-20.49)}_{-0.19375344} \end{bmatrix}$$

Estratégia de Pivoteamento

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0.004x_1 + 15.73x_2 = 15.77 \end{cases}$$

Solução Exata: $x_1 = 10$, $x_2 = 1$

Considerando 4 dígitos e arredondamento (**com** pivoteamento parcial):

$$m = 0.004/0.423 = 0.009456265 \rightarrow 0.009456$$

$$\begin{bmatrix} 0.423 & -24.72 \\ 0 & (\underbrace{15.73-\underbrace{0.009456*(-24.72)}_{-0.23375232}}_{15.9638}) \end{bmatrix} \begin{bmatrix} -20.49 \\ \underbrace{15.77-\underbrace{0.009456*(-20.49)}_{-0.19375344}}_{15.9638} \end{bmatrix}$$

Estratégia de Pivoteamento

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0.004x_1 + 15.73x_2 = 15.77 \end{cases}$$

Solução Exata: $x_1 = 10$

$x_2 = 1$

$$\begin{bmatrix} 0.423 & -24.72 \\ 0 & 15.96 \end{bmatrix} \begin{bmatrix} -20.49 \\ 15.96 \end{bmatrix}$$

Considerando 4 dígitos e arredondamento (**com** pivot

$$\begin{bmatrix} 0.423 & -24.72 \\ 0 & \underbrace{(15.73 \underbrace{-0.009456*(-24.72)}_{-0.23375232})}_{15.9638} \end{bmatrix} \begin{bmatrix} -20.49 \\ \underbrace{15.77 \underbrace{-0.009456*(-20.49)}_{-0.19375344}}_{15.9638} \end{bmatrix}$$

Estratégia de Pivoteamento

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0.004x_1 + 15.73x_2 = 15.77 \end{cases}$$

Solução Exata: $x_1 = 10$

$x_2 = 1$

Considerando 4 dígitos e arredondamento (**com** pivot

$$\begin{bmatrix} 0.423 & -24.72 \\ 0 & 15.96 \end{bmatrix}\begin{bmatrix} -20.49 \\ 15.96 \end{bmatrix}$$

Solução: $x_2 = 15.96/15.96 = 1$

Estratégia de Pivoteamento

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0.004x_1 + 15.73x_2 = 15.77 \end{cases}$$

Solução Exata: $x_1 = 10$
$x_2 = 1$

Considerando 4 dígitos e arredondamento (**com** pivot

$$\begin{bmatrix} 0.423 & -24.72 \\ 0 & 15.96 \end{bmatrix} \begin{bmatrix} -20.49 \\ 15.96 \end{bmatrix}$$

Solução: $x_2 = 15.96/15.96 = 1$

$x_1 = (-20.49 + 24.72*1)/0.423$

$= 4.23/0.423 = 10$

Estratégia de Pivoteamento

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0.004x_1 + 15.73x_2 = 15.77 \end{cases}$$

Solução Exata: $x_1 = 10$

$x_2 = 1$

Considerando 4 dígitos e arredondamento (**com** pivot

$$\left[\begin{array}{cc} 0.423 & -24.72 \\ 0 & 15.96 \end{array}\right]\left[\begin{array}{c} -20.49 \\ 15.96 \end{array}\right]$$

Solução:

$x_2 = 15.96/15.96 = 1$

$x_1 = (-20.49 + 24.72*1)/0.423$

$= 4.23/0.423 = 10$

$$ER_{x_i} = 0\%$$

Estratégia de Pivoteamento

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0.004x_1 + 15.73x_2 = 15.77 \end{cases}$$

Solução Exata: $x_1 = 10$

$x_2 = 1$

Considerando

$$(d_1.d_2d_3...d_t \times 2^{e'}) \;/\; (b_1.b_2b_3...b_t \times 2^{e''})$$

$$= (d_1.d_2d_3...d_t \;/\; b_1.b_2b_3...b_t) \times 2^{e' - e''}$$

$$\begin{bmatrix} \ldots & .72 \\ \ldots & .96 \end{bmatrix} \begin{bmatrix} -20.49 \\ 15.96 \end{bmatrix}$$

Solução: $x_2 = 15.96/15.96 = 1$

$x_1 = (-20.49 + 24.72*1)/0.423$

$= 4.23/0.423 = 10$

$ER_{x_i} = 0\%$

Estratégia de Pivoteamento

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0.004x_1 + 15.73x \ldots = \ldots 15.77 \end{cases}$$

Solução Exata: $x_1 = 10$

$x_2 = 1$

$$(d_1.d_2d_3\ldots d_t \times 2^{e'}) \;/\; (b_1.b_2b_3\ldots b_t \times 2^{e''})$$

$$= (d_1.d_2d_3\ldots d_t \;/\; b_1.b_2b_3\ldots b_t) \times 2^{e' - e''} = e' + |e''|$$

Considerando

$$\ldots .72 \quad \begin{bmatrix} -20.49 \\ 15.96 \end{bmatrix}$$
$$\ldots .96$$

Solução: $x_2 = 15.96/15.96 = 1$

$x_1 = (-20.49 + 24.72*1)/0.423$

$= 4.23/0.423 = 10$

$ER_{x_i} = 0\%$

Pivoteamento Total

$$\begin{cases} 0.448x_1 + 0.832x_2 + 0.193x_3 = 1 \\ 0.421x_1 + 0.784x_2 - 0.207x_3 = 0 \\ -0.319x_1 + 0.884x_2 + 0.279x_3 = 0 \end{cases}$$

Fonte: Cálculo Numérico Computacional (Peters & Szeremeta)

Pivoteamento Total

$$\begin{cases} 0.448x_1 + 0.832x_2 + 0.193x_3 = 1 \\ 0.421x_1 + 0.784x_2 - 0.207x_3 = 0 \\ -0.319x_1 + 0.884x_2 + 0.279x_3 = 0 \end{cases}$$

$$A = \begin{bmatrix} 0.448 & 0.832 & 0.193 & \vdots & 1 \\ 0.421 & 0.784 & -0.207 & \vdots & 0 \\ -0.319 & 0.884 & 0.279 & \vdots & 0 \end{bmatrix}$$

Fonte: Cálculo Numérico Computacional (Peters & Szeremeta)

Pivoteamento Total

$$\begin{cases} 0.448x_1 + 0.832x_2 + 0.193x_3 = 1 \\ 0.421x_1 + 0.784x_2 - 0.207x_3 = 0 \\ -0.319x_1 + 0.884x_2 + 0.279x_3 = 0 \end{cases}$$

$$\begin{array}{c} A = \left[\begin{array}{ccc:c} 0.448 & 0.832 & 0.193 & 1 \\ 0.421 & 0.784 & -0.207 & 0 \\ -0.319 & \mathbf{0.884} & 0.279 & 0 \end{array}\right] \\ \quad\quad x_1 \quad\quad x_2 \quad\quad x_3 \end{array}$$

Fonte: Cálculo Numérico Computacional (Peters & Szeremeta)

Pivoteamento Total

$$\begin{cases} 0.448x_1 + 0.832x_2 + 0.193x_3 = 1 \\ 0.421x_1 + 0.784x_2 - 0.207x_3 = 0 \\ -0.319x_1 + 0.884x_2 + 0.279x_3 = 0 \end{cases}$$

$$A = \begin{array}{c} \left[\begin{array}{ccc|c} \mathbf{0.884} & -0.319 & 0.279 & 0 \\ 0.784 & 0.421 & -0.207 & 0 \\ 0.832 & 0.448 & 0.193 & 1 \end{array}\right] \\ \begin{array}{cccc} x_2 & \quad x_1 & \quad x_3 & \end{array} \end{array}$$

Fonte: Cálculo Numérico Computacional (Peters & Szeremeta)

Pivoteamento Total

$$\begin{cases} 0.448x_1 + 0.832x_2 + 0.193x_3 = 1 \\ 0.421x_1 + 0.784x_2 - 0.207x_3 = 0 \\ -0.319x_1 + 0.884x_2 + 0.279x_3 = 0 \end{cases}$$

$$\begin{array}{c} A = \left[\begin{array}{ccc|c} \mathbf{0.884} & -0.319 & 0.279 & 0 \\ 0.784 & 0.421 & -0.207 & 0 \\ 0.832 & 0.448 & 0.193 & 1 \end{array}\right] \\ \quad\quad x_2 \quad\quad x_1 \quad\quad x_3 \end{array}$$

$$\begin{array}{l} L_2 \leftarrow L_2 - (0.784 / 0.884)\, L_1 \\ L_3 \leftarrow L_3 - (0.832 / 0.884)\, L_1 \end{array}$$

Fonte: Cálculo Numérico Computacional (Peters & Szeremeta)

Pivoteamento Total

$$\begin{cases} 0.448x_1 + 0.832x_2 + 0.193x_3 = 1 \\ 0.421x_1 + 0.784x_2 - 0.207x_3 = 0 \\ -0.319x_1 + 0.884x_2 + 0.279x_3 = 0 \end{cases}$$

$$A = \begin{array}{c} \left[\begin{array}{ccc:c} 0.884 & -0.319 & 0.279 & 0 \\ 0 & 0.7039 & -0.4544 & 0 \\ 0 & 0.7482 & -0.06959 & 1 \end{array}\right] \\ \begin{array}{ccc} x_2 & \quad x_1 & \quad x_3 \end{array} \end{array}$$

Fonte: Cálculo Numérico Computacional (Peters & Szeremeta)

Pivoteamento Total

$$\begin{cases} 0.448x_1 + 0.832x_2 + 0.193x_3 = 1 \\ 0.421x_1 + 0.784x_2 - 0.207x_3 = 0 \\ -0.319x_1 + 0.884x_2 + 0.279x_3 = 0 \end{cases}$$

$$A = \begin{array}{c} \left[\begin{array}{ccccc} 0.884 & -0.319 & 0.279 & \vdots & 0 \\ 0 & 0.7039 & -0.4544 & \vdots & 0 \\ 0 & \mathbf{0.7482} & -0.06959 & \vdots & 1 \end{array}\right] \\ \begin{array}{ccc} x_2 & x_1 & x_3 \end{array} \end{array}$$

Fonte: Cálculo Numérico Computacional (Peters & Szeremeta)

Pivoteamento Total

$$\begin{cases} 0.448x_1 + 0.832x_2 + 0.193x_3 = 1 \\ 0.421x_1 + 0.784x_2 - 0.207x_3 = 0 \\ -0.319x_1 + 0.884x_2 + 0.279x_3 = 0 \end{cases}$$

$$A = \begin{array}{c} \left[\begin{array}{ccc:c} 0.884 & -0.319 & 0.279 & 0 \\ 0 & \mathbf{0.7482} & -0.06959 & 1 \\ 0 & 0.7039 & -0.4544 & 0 \end{array}\right] \\ \begin{array}{ccc} x_2 & \qquad x_1 & \qquad x_3 \end{array} \end{array}$$

Fonte: Cálculo Numérico Computacional (Peters & Szeremeta)

Pivoteamento Total

$$
\begin{cases}
0.448x_1 + 0.832x_2 + 0.193x_3 = 1 \\
0.421x_1 + 0.784x_2 - 0.207x_3 = 0 \\
-0.319x_1 + 0.884x_2 + 0.279x_3 = 0
\end{cases}
$$

$$
A = \begin{array}{c}
\left[\begin{array}{ccc:c}
0.884 & -0.319 & 0.279 & 0 \\
0 & \mathbf{0.7482} & -0.06959 & 1 \\
0 & 0.7039 & -0.4544 & 0
\end{array}\right] \\
\begin{array}{ccc} x_2 & \quad x_1 & \quad x_3 \end{array}
\end{array}
\quad \boxed{L_3 \leftarrow L_3 - (0.7039 / 0.7482) L_2}
$$

Fonte: Cálculo Numérico Computacional (Peters & Szeremeta)

Pivoteamento Total

$$\begin{cases} 0.448x_1 + 0.832x_2 + 0.193x_3 = 1 \\ 0.421x_1 + 0.784x_2 - 0.207x_3 = 0 \\ -0.319x_1 + 0.884x_2 + 0.279x_3 = 0 \end{cases}$$

$$A = \begin{array}{c} \left[\begin{array}{ccc:c} 0.884 & -0.319 & 0.279 & 0 \\ 0 & \mathbf{0.7482} & -0.06959 & 1 \\ 0 & 0 & -0.3889 & -0.9408 \end{array}\right] \\ \begin{array}{ccc} x_2 & \quad x_1 & \quad x_3 \end{array} \end{array}$$

Fonte: Cálculo Numérico Computacional (Peters & Szeremeta)

Pivoteamento Total

$$\begin{cases} 0.448x_1 + 0.832x_2 + 0.193x_3 = 1 \\ 0.421x_1 + 0.784x_2 - 0.207x_3 = 0 \\ -0.319x_1 + 0.884x_2 + 0.279x_3 = 0 \end{cases}$$

$$\begin{cases} 0.884x_2 - 0.319x_1 + 0.279x_3 = 0 \\ 0x_2 + 0.7482x_1 - 0.06959x_3 = 1 \\ 0x_2 + 0x_1 - 0.3889x_3 = -0.9408 \end{cases}$$

$$\begin{array}{cc} A = & \left[\begin{array}{ccc:c} 0.884 & -0.319 & 0.279 & 0 \\ 0 & \mathbf{0.7482} & -0.06959 & 1 \\ 0 & 0 & -0.3889 & -0.9408 \end{array}\right] \\ & \begin{array}{ccc} x_2 & x_1 & x_3 \end{array} \end{array}$$

Fonte: Cálculo Numérico Computacional (Peters & Szeremeta)

Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      uint iPivo = encontraMax(A, i);
      if ( i != iPivo )
         trocaLinha( A, b, i, iPivo );

      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```

$$\boxed{eq_k = eq_k - m_{ki} * eq_i}$$

Método da Eliminação de Gauss

```
/* Seja um S.L. de ordem 'n'
*/
void eliminacaoGauss( double **A, double *b, uint n ){
   /* para cada linha a partir da primeira */
   for (int i=0; i < n; ++i) {
      uint iPivo = encontraMax(A, i);
      if ( i != iPivo )
         trocaLinha( A, b, i, iPivo );

      for(int k=i+1; k < n; ++k) {
         double m = A[k][i] / A[i][i];
         A[k][i] = 0.0;
         for(int j=i+1; j < n; ++j)
            A[k][j] -= A[i][j] * m;
         b[k] -= b[i] * m;
      }
   }
}
```

$$\boxed{eq_k = eq_k - m_{ki} * eq_i}$$

$$\boxed{eq_k = eq_k * A[i][i] - eq_i * A[k][i]}$$

Resíduo

Sistema possível determinado

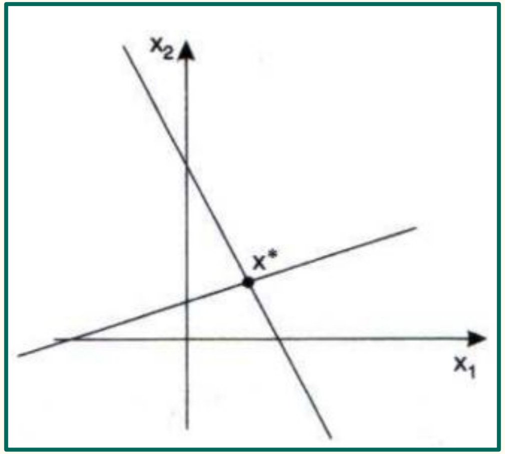
$$\begin{cases} 2x_1 + x_2 = 3 \\ x_1 - 3x_2 = -2 \end{cases}$$

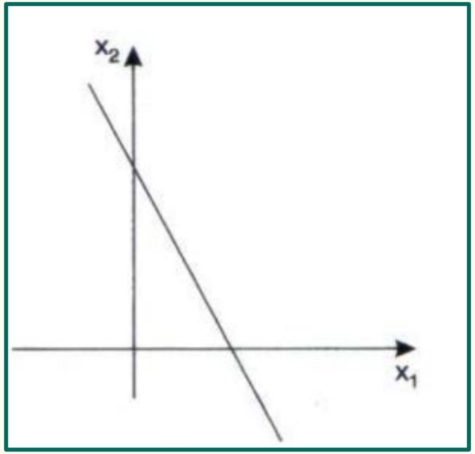
Sistema possível indeterminado

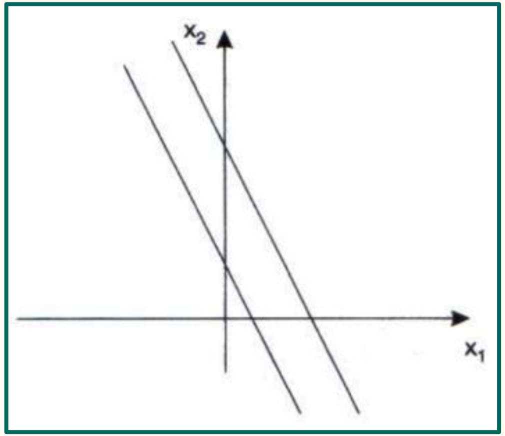
$$\begin{cases} 2x_1 + x_2 = 3 \\ 4x_1 + 2x_2 = 6 \end{cases}$$

Sistema impossível

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







Resíduo

Sistema possível determinado

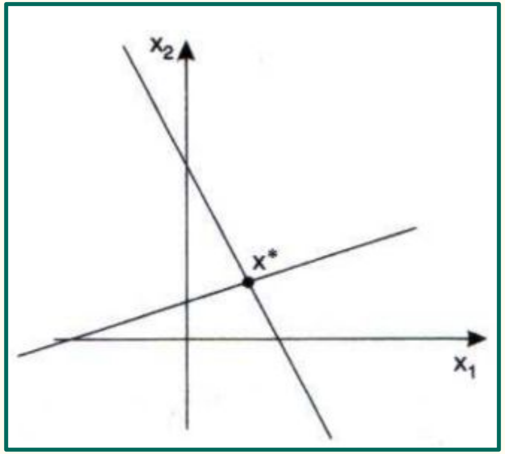
$$\begin{cases} 2x_1 + x_2 = 3 \\ x_1 - 3x_2 = -2 \end{cases}$$

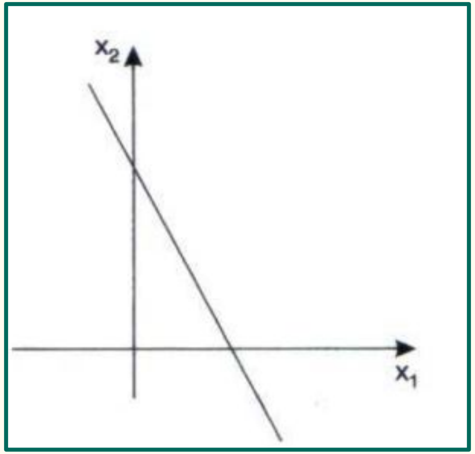
Sistema possível indeterminado

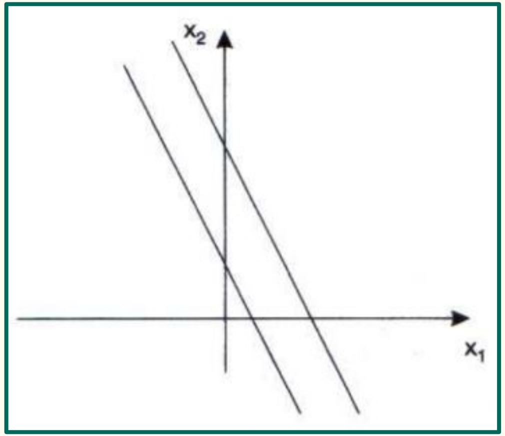
$$\left[\begin{array}{cccc} 0.448 & 0.832 & 0.193 & \vdots & 1 \\ 0 & 0.002170 & -0.3884 & \vdots & -0.9397 \\ 0 & 0 & 0 & \vdots & 0 \end{array}\right]$$

Sistema impossível

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







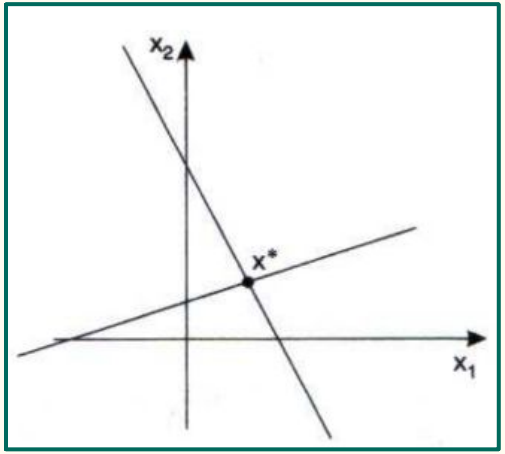
Resíduo

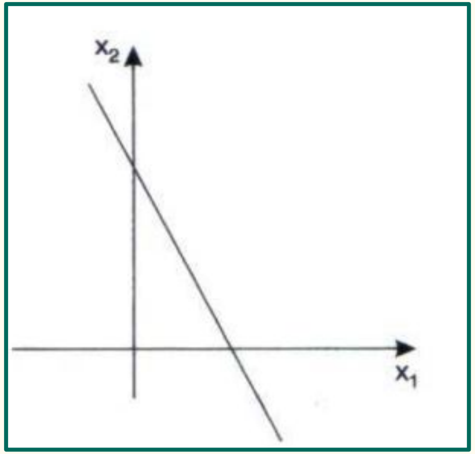
Sistema possível determinado	Sistema possível indeterminado	Sistema impossível

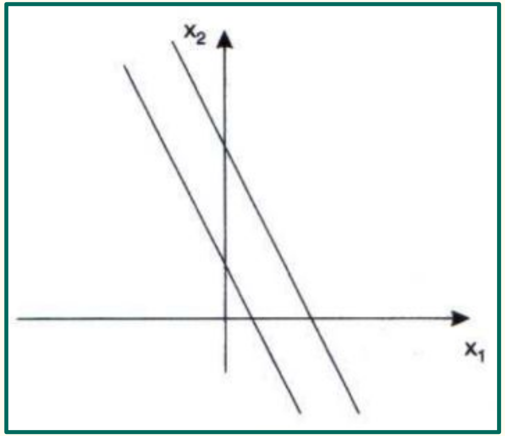
$$\begin{cases} 2x_1 + x_2 = 3 \\ x_1 - 3x_2 = -2 \end{cases}$$

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

$$\left[\begin{array}{cccc} 0.448 & 0.832 & 0.193 & \vdots & 1 \\ 0 & 0.002170 & -0.3884 & \vdots & -0.9397 \\ 0 & 0 & 0 & \vdots & 49.0603 \end{array}\right]$$



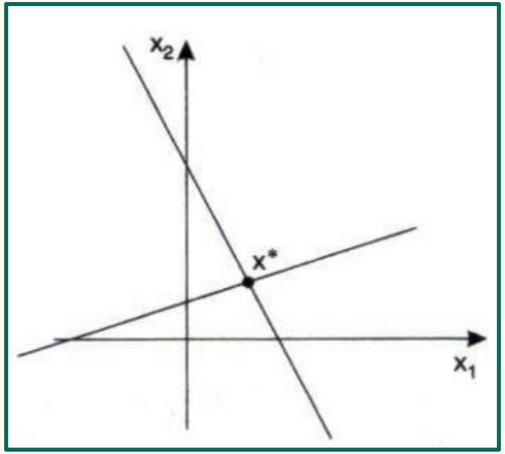




Resíduo

Sistema possível determinado

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



$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

Solução: $x_2 = 0.9994$

$x_1 = 12.50$

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0.004x_1 + 15.73x_2 = 15.77 \end{cases}$$

Solução: $x_2 = 1$

$x_1 = 10$

Resíduo

R = AX - B

$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

Solução: $x_2 = 0.9994$

$x_1 = 12.50$

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0.004x_1 + 15.73x_2 = 15.77 \end{cases}$$

Solução: $x_2 = 1$

$x_1 = 10$

Resíduo

$$R = AX - B$$

$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

$$\begin{cases} 0.004x_1 + 15.73x_2 - 15.77 = 0.000562 \\ 0.423x_1 - 24.72x_2 + 20.49 = 1.072332 \end{cases}$$

Solução: $x_2 = 0.9994$

$x_1 = 12.50$

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0.004x_1 + 15.73x_2 = 15.77 \end{cases}$$

Solução: $x_2 = 1$

$x_1 = 10$

Resíduo

R = AX - B

$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

$$\begin{cases} 0.004x_1 + 15.73x_2 - 15.77 = 0.000562 \\ 0.423x_1 - 24.72x_2 + 20.49 = 1.072332 \end{cases}$$

Solução: $x_2 = 0.9994$

$x_1 = 12.50$

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0.004x_1 + 15.73x_2 = 15.77 \end{cases}$$

$$\begin{cases} 0.423x_1 - 24.72x_2 + 20.49 = 0 \\ 0.004x_1 + 15.73x_2 - 15.77 = 0 \end{cases}$$

Solução: $x_2 = 1$

$x_1 = 10$

Sistemas Mal-condicionados

Quando a solução depende **sensivelmente** dos coeficientes.

$$\begin{cases} x_1 + x_2 = 1 \\ 99x_1 + 100x_2 = 99.5 \end{cases}$$

Solução: $x_1 = 0.5$

$x_2 = 0.5$

Sistemas Mal-condicionados

Quando a solução depende **sensivelmente** dos coeficientes.

$$\begin{cases} x_1 + x_2 = 1 \\ 99x_1 + 100x_2 = 99.5 \end{cases}$$

Solução: $x_1 = 0.5$
$x_2 = 0.5$

Variação $< 0.5\%$ $\Rightarrow$

$$\begin{cases} x_1 + x_2 = 1 \\ 99.4x_1 + 99.9x_2 = 99.2 \end{cases}$$

Solução: $x_1 = 1.4$
$x_2 = -0.4$

Matrizes de Hilbert

$$\begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{n+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \frac{1}{n+2} & \cdots & \frac{1}{2n-1} \end{pmatrix}$$

Matrizes de Hilbert

$$\begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{n+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \frac{1}{n+2} & \cdots & \frac{1}{2n-1} \end{pmatrix}$$

Exemplo:

$$\left\{ \begin{array}{ccccccc} x_1 & + & \frac{1}{2}x_2 & + & \frac{1}{3}x_3 & = & 1 \\ \frac{1}{2}x_1 & + & \frac{1}{3}x_2 & + & \frac{1}{4}x_3 & = & 1 \\ \frac{1}{3}x_1 & + & \frac{1}{4}x_2 & + & \frac{1}{5}x_3 & = & 1 \end{array} \right.$$

Matrizes de Hilbert

$$\begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{n+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \frac{1}{n+2} & \cdots & \frac{1}{2n-1} \end{pmatrix}$$

Exemplo:

$$\begin{cases} x_1 + \frac{1}{2}x_2 + \frac{1}{3}x_3 = 1 \\ \frac{1}{2}x_1 + \frac{1}{3}x_2 + \frac{1}{4}x_3 = 1 \\ \frac{1}{3}x_1 + \frac{1}{4}x_2 + \frac{1}{5}x_3 = 1 \end{cases}$$

<u>Frações Exatas</u>: $X = (3, -24, 30)$

Matrizes de Hilbert

$$\begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{n+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \frac{1}{n+2} & \cdots & \frac{1}{2n-1} \end{pmatrix}$$

Exemplo:

$$\begin{cases} x_1 & + & \frac{1}{2}x_2 & + & \frac{1}{3}x_3 & = & 1 \\ \frac{1}{2}x_1 & + & \frac{1}{3}x_2 & + & \frac{1}{4}x_3 & = & 1 \\ \frac{1}{3}x_1 & + & \frac{1}{4}x_2 & + & \frac{1}{5}x_3 & = & 1 \end{cases}$$

<u>Frações Exatas</u>: $X = (3, -24, 30)$

<u>2 algarismos significativos</u> ($1/3 = 0.33$):
$X = (0.9, -11, 17)$

Matrizes de Hilbert

$$\begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{n+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \frac{1}{n+2} & \cdots & \frac{1}{2n-1} \end{pmatrix}$$

Exemplo:

$$\begin{cases} x_1 & + & \frac{1}{2}x_2 & + & \frac{1}{3}x_3 & = & 1 \\ \frac{1}{2}x_1 & + & \frac{1}{3}x_2 & + & \frac{1}{4}x_3 & = & 1 \\ \frac{1}{3}x_1 & + & \frac{1}{4}x_2 & + & \frac{1}{5}x_3 & = & 1 \end{cases}$$

<u>Frações Exatas</u>: $X = (3, -24, 30)$

<u>2 algarismos significativos</u> ($1/3 = 0.33$):
$X = (0.9, -11, 17)$

<u>3 algarismos significativos</u> ($1/3 = 0.333$):
$X = (2.64, -21.8, 27.8)$

Refinamento

Obter uma solução x' de $Ax = b$ e depois melhorá-la.

Refinamento

Obter uma solução x' de $Ax = b$ e depois melhorá-la.

Seja x a solução verdadeira. Então, $x = x' + w$.

Refinamento

Obter uma solução x' de $Ax = b$ e depois melhorá-la.

Seja x a solução verdadeira. Então, $x = x' + w$.

$$Ax = b \quad \Rightarrow \quad A(x' + w) = b \quad \Rightarrow \quad Aw = b - Ax'$$

Refinamento

Obter uma solução x' de $Ax = b$ e depois melhorá-la.

Seja x a solução verdadeira. Então, $x = x' + w$.

$$Ax = b \quad \Rightarrow \quad A(x' + w) = b \quad \Rightarrow \quad Aw = \overbrace{b - Ax'}^{\text{Resíduo}}$$

Refinamento

Obter uma solução x' de $Ax = b$ e depois melhorá-la.

Seja x a solução verdadeira. Então, $x = x' + w$.

$$Ax = b \Longrightarrow A(x' + w) = b \Longrightarrow Aw = \overbrace{b - Ax'}^{\text{Resíduo}}$$

Algoritmo:

1 - Obter solução inicial $\mathbf{x^{(0)}}$ resolvendo $\mathbf{Ax = b}$ e inicializar $\mathbf{i = 0}$;
2 - Calcular o resíduo $\mathbf{r = b - Ax^{(i)}}$ e testar critério de parada (a);
3 - Obter $\mathbf{w}$ resolvendo $\mathbf{Aw = r}$;
4 - Obter nova solução $\mathbf{x^{(i+1)} = x^{(i)} + w}$ e testar critério de parada (b);
5 - Incrementar $\mathbf{i}$ e voltar ao passo 2;

Refinamento

Obter uma solução x' de $Ax = b$ e depois melhorá-la.

Seja x a solução verdadeira. Então, $x = x' + w$.

$$Ax = b \Longrightarrow A(x' + w) = b \Longrightarrow Aw = \overbrace{b - Ax'}^{\text{Resíduo}}$$

Algoritmo:

Precisão dupla

1 - Obter solução inicial $\mathbf{x^{(0)}}$ resolvendo $\mathbf{Ax = b}$ e inicializar $\mathbf{i = 0}$;

2 - Calcular o resíduo $\mathbf{r = b - Ax^{(i)}}$ e testar critério de parada (a);

3 - Obter $\mathbf{w}$ resolvendo $\mathbf{Aw = r}$;

4 - Obter nova solução $\mathbf{x^{(i+1)} = x^{(i)} + w}$ e testar critério de parada (b);

5 - Incrementar $\mathbf{i}$ e voltar ao passo 2;

Refinamento

Obter uma solução x' de $Ax = b$ e depois melhorá-la.

Seja x a solução verdadeira. Então, $x = x' + w$.

$$Ax = b \longrightarrow A(x' + w) = b \longrightarrow Aw = \overbrace{b - Ax'}^{\text{Resíduo}}$$

Algoritmo:

$$\|r\| = \sqrt{r_1^2 + r_2^2 + \cdots + r_n^2} < \epsilon$$

1 - Obter solução inicial $\mathbf{x^{(0)}}$ resolvendo $\mathbf{Ax = b}$ e inicializar $\mathbf{i = 0}$;
2 - Calcular o resíduo $\mathbf{r = b - Ax^{(i)}}$ e testar critério de parada (a);
3 - Obter $\mathbf{w}$ resolvendo $\mathbf{Aw = r}$;
4 - Obter nova solução $\mathbf{x^{(i+1)} = x^{(i)} + w}$ e testar critério de parada (b);
5 - Incrementar $\mathbf{i}$ e voltar ao passo 2;

Refinamento

Obter uma solução x' de $Ax = b$ e depois melhorá-la.

Seja x a solução verdadeira. Então, $x = x' + w$.

$$Ax = b \quad \Rightarrow \quad A(x' + w) = b \quad \Rightarrow \quad Aw = \overbrace{b - Ax'}^{\text{Resíduo}}$$

Algoritmo:

$max(|\hat{x}_i - \hat{x}_{i+1}|) < \epsilon$

1 - Obter solução inicial $\mathbf{x^{(0)}}$ resolvendo $\mathbf{Ax = b}$ e inicializar $\mathbf{i = 0}$;

2 - Calcular o resíduo $\mathbf{r = b - Ax^{(i)}}$ e testar critério de parada (a);

3 - Obter $\mathbf{w}$ resolvendo $\mathbf{Aw = r}$;

4 - Obter nova solução $\mathbf{x^{(i+1)} = x^{(i)} + w}$ e testar critério de parada (b);

5 - Incrementar $\mathbf{i}$ e voltar ao passo 2;

Refinamento

Obter uma solução x' de $Ax = b$ e depois melhorá-la.

Seja x a solução verdadeira. Então, $x = x' + w$.

$$Ax = b \quad \Longrightarrow \quad A(x' + w) = b \quad \Longrightarrow \quad Aw = \overbrace{b - Ax'}^{\text{Resíduo}}$$

Algoritmo:

1 - Obter solução inicial $\mathbf{x}^{(0)}$ resolvendo $\mathbf{Ax = b}$ e inicializar $\mathbf{i = 0}$;
2 - Calcular o resíduo $\mathbf{r = b - Ax^{(i)}}$ e testar critério de parada (a);
3 - Obter $\mathbf{w}$ resolvendo $\mathbf{Aw = r}$;
4 - Obter nova solução $\mathbf{x^{(i+1)} = x^{(i)} + w}$ e testar critério de parada (b);
5 - Incrementar $\mathbf{i}$ e voltar ao passo 2;

Refinamento

Obter uma solução x' de $Ax = b$ e depois melhorá-la.

Seja x a solução verdadeira. Então, $x = x' + w$.

$$Ax = b \quad \Rightarrow \quad A(x' + w) = b \quad \Rightarrow \quad Aw = \overbrace{b - Ax'}^{\text{Resíduo}}$$

Algoritmo:

Basta aplicar em **r** todas as transformações que foram aplicadas em **b**

1 - Obter solução inicial $\mathbf{x^{(0)}}$ resolvendo $\mathbf{Ax = b}$ e inicializar $\mathbf{i = 0}$;

2 - Calcular o resíduo $\mathbf{r = b - Ax^{(i)}}$ e testar critério de parada (a);

3 - Obter $\mathbf{w}$ resolvendo $\mathbf{Aw = r}$;

4 - Obter nova solução $\mathbf{x^{(i+1)} = x^{(i)} + w}$ e testar critério de parada (b);

5 - Incrementar $\mathbf{i}$ e voltar ao passo 2;

Refinamento

$$\begin{cases} 0.004x_1 + 15.73x_2 = 15.77 \\ 0.423x_1 - 24.72x_2 = -20.49 \end{cases}$$

1 - trocar linhas 1 e 2;

s melhorá-la.

$x' + w.$

$$Aw = \overbrace{b - Ax'}^{\text{Resíduo}}$$

Basta aplicar em **r** todas as transformações que foram aplicadas em **b**

$= \mathbf{b}$ e inicializar $\mathbf{i} = \mathbf{0}$;

r critério de parada (a);

testar critério de parada (b);

Refinamento

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0x_1 + 15.96x_2 = 15.96 \end{cases}$$

1 - trocar linhas 1 e 2;

2 - subtrair da linha 2 a linha 1 multiplicada por 0.009456;

s melhorá-la.

$x' + w.$

$$Aw = \overbrace{b - Ax'}^{\text{Resíduo}}$$

Basta aplicar em **r** todas as transformações que foram aplicadas em **b**

$= \mathbf{b}$ e inicializar $\mathbf{i} = \mathbf{0}$;

r critério de parada (a);

testar critério de parada (b);

Refinamento

$$\begin{cases} 0.423x_1 - 24.72x_2 = -20.49 \\ 0x_1 + 15.96x_2 = 15.96 \end{cases}$$

1 - trocar linhas 1 e 2;
2 - subtrair da linha 2 a linha 1 multiplicada por 0.009456;

$$\mathbf{r} = \begin{bmatrix} a \\ b \end{bmatrix} \Rightarrow \begin{bmatrix} b \\ a \end{bmatrix} \Rightarrow \begin{bmatrix} b \\ a - 0.009456b \end{bmatrix}$$

s melhorá-la.

$x' + w.$

$$Aw = \overbrace{b - Ax'}^{\text{Resíduo}}$$

Basta aplicar em **r** todas as transformações que foram aplicadas em **b**

$= \mathbf{b}$ e inicializar $\mathbf{i} = \mathbf{0}$;

r critério de parada (a);

testar critério de parada (b);

Referências

- Daniel Weingaertner; notas de aula da disciplina **Introdução à Computação Científica** (UFPR/DINF)

- M. Cristina C. Cunha; **Métodos Numéricos**. Editora Unicamp.

- Márcia A. G. Ruggiero e Vera L. R. Lopes; **Cálculo Numérico - Aspectos Teóricos e Computacionais**. Editora Pearson.

- Claudio H. Asano e Eduardo Colli; **Cálculo Numérico - Fundamentos e Aplicações.** Disponível em https://www.ime.usp.br/~asano/LivroNumerico/LivroNumerico.pdf

- Sérgio Peters e Julio Felipe Szeremeta; **Cálculo Numérico Computacional**. Editora UFSC. Disponível em http://sergiopeters.prof.ufsc.br/livro-calculo-numerico-computacional/

Créditos

Este documento é de autoria do Prof. Guilherme Alex Derenievicz (UFPR/DINF), para uso na disciplina Introdução à Computação Científica (CI1164).

Compartilhe este documento de acordo com a licença abaixo