# Métodos iterativos

Vicente Helano
UFCA | Centro de Ciências e Tecnologia

$$x = G(x) \Leftarrow \overbrace{Ax - b}^{F(x)} = 0$$

$$\overbrace{Ax = b}$$

# Método de Jacobi

$$\begin{array}{ccccccc} 9x_1 & + & x_2 & + & x_3 & = & 1 \\ 2x_1 & + & 10x_2 & + & 3x_3 & = & 2 \\ 3x_1 & + & 4x_2 & + & 11x_3 & = & 3 \end{array}$$

$$x_1 = \frac{1}{9}(1 - x_2 - x_3)$$

$$x_2 = \frac{1}{10}(2 - 2x_1 - 3x_3)$$

$$x_3 = \frac{1}{11}(3 - 3x_1 - 4x_2)$$

$$\Rightarrow \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 & -1/9 & -1/9 \\ -1/5 & 0 & -3/10 \\ -3/11 & -4/11 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 1/9 \\ 1/5 \\ 3/11 \end{bmatrix}$$

# JACOBI

$$x_1^{(k+1)} = \frac{1}{9}\left(1 - x_2^{(k)} - x_3^{(k)}\right)$$

$$x_2^{(k+1)} = \frac{1}{10}\left(2 - 2x_1^{(k)} - 3x_3^{(k)}\right)$$

$$x_3^{(k+1)} = \frac{1}{11}\left(3 - 3x_1^{(k)} - 4x_2^{(k)}\right)$$

$$\Rightarrow \begin{array}{cccccc} x^{(k+1)} & & M & x^{(k)} & & c \\ \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_{(k+1)} & = & \begin{bmatrix} 0 & -1/9 & -1/9 \\ -1/5 & 0 & -3/10 \\ -3/11 & -4/11 & 0 \end{bmatrix} & \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_{(k)} & + & \begin{bmatrix} 1/9 \\ 1/5 \\ 3/11 \end{bmatrix} \end{array}$$

$$\boxed{x^{(k+1)} = \underbrace{Mx^{(k)} + c}_{G(x)}}$$

$$\boxed{x = G(x)}$$

AUTOVALOR DE M:

RAÍZES DO POLINÔMIO CARACTERÍSTICO:

$$p(\lambda) = \det(M - \lambda I)$$

AUTOVALOR DE M: (EIGENVALUE)

EIGENVECTOR

RAÍZES DO POLINÔMIO CARACTERÍSTICO:

$$\begin{array}{cccc} x^{(k+1)} & M & x^{(k)} & c \\ \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}^{(k+1)} = & \begin{bmatrix} 0 & -1/9 & -1/9 \\ -1/5 & 0 & -3/10 \\ -3/11 & -4/11 & 0 \end{bmatrix} & \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}^{(k)} + & \begin{bmatrix} 1/9 \\ 1/5 \\ 3/11 \end{bmatrix} \end{array}$$

$$\boxed{x^{(k+1)} = \underbrace{Mx^{(k)} + c}_{G(x)}}$$

$$p(\lambda) = \det(M - \lambda I)$$

$$= \det\left(\begin{bmatrix} 0 & -1/9 & -1/9 \\ -1/5 & 0 & -3/10 \\ -3/11 & -4/11 & 0 \end{bmatrix} - \begin{bmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{bmatrix}\right)$$

$$= \det\left(\begin{bmatrix} -\lambda & -1/9 & -1/9 \\ -1/5 & -\lambda & -3/10 \\ -3/11 & -4/11 & -\lambda \end{bmatrix}\right)$$

$x^{(0)} = (0,0,0)$ (VALOR ARBITRADO INICIALMENTE – CHUTE INICIAL)

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}^{(1)} = \underbrace{\begin{bmatrix} 0 & -1/9 & -1/9 \\ -1/5 & 0 & -3/10 \\ -3/11 & -4/11 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}^{(0)}}_{(0,0,0)} + \begin{bmatrix} 1/9 \\ 1/5 \\ 3/11 \end{bmatrix} = \begin{bmatrix} 1/9 \\ 1/5 \\ 3/11 \end{bmatrix} \approx \begin{bmatrix} 0{,}11 \\ 0{,}2 \\ 0{,}27 \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}^{(2)} = \begin{bmatrix} 0 & -1/9 & -1/9 \\ -1/5 & 0 & -3/10 \\ -3/11 & -4/11 & 0 \end{bmatrix} \begin{bmatrix} 1/9 \\ 1/5 \\ 3/11 \end{bmatrix}^{(1)} + \begin{bmatrix} 1/9 \\ 1/5 \\ 3/11 \end{bmatrix}$$

# Método de Jacobi

Para um sistema $n \times n$, teremos:

$$x_1^{(k+1)} = \frac{1}{a_{11}}\left(-a_{12}x_2^{(k)} - a_{13}x_3^{(k)} - \cdots - a_{1n}x_n^{(k)} + b_1\right)$$

$$x_2^{(k+1)} = \frac{1}{a_{22}}\left(-a_{21}x_1^{(k)} - a_{23}x_3^{(k)} - \cdots - a_{2n}x_n^{(k)} + b_2\right)$$

$$\vdots$$

$$x_n^{(k+1)} = \frac{1}{a_{nn}}\left(-a_{n1}x_1^{(k)} - a_{n2}x_2^{(k)} - \cdots - a_{n,(n-1)}x_{n-1}^{(k)} + b_n\right)$$

# Método de Jacobi

De uma forma compacta,

$$\begin{cases} x_1^{(k+1)} = \dfrac{1}{a_{11}}\left(\underbrace{-a_{12}x_2^{(k)} - a_{13}x_3^{(k)} - \cdots - a_{1n}x_n^{(k)}} + b_1\right) \\ x_2^{(k+1)} = \dfrac{1}{a_{22}}\left(-a_{21}x_1^{(k)} - a_{23}x_3^{(k)} - \cdots - a_{2n}x_n^{(k)} + b_2\right) \\ \vdots \\ x_n^{(k+1)} = \dfrac{1}{a_{nn}}\left(-a_{n1}x_1^{(k)} - a_{n2}x_2^{(k)} - \cdots - a_{n,(n-1)}x_{n-1}^{(k)} + b_n\right) \end{cases}$$

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

# Critério de parada

NORMA "COMPRIMENTO"

$$|x_{k+1} - x_k| < \varepsilon \cdot |x_{k+1}|$$

$$\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\| < \varepsilon \cdot \max\left\{\|\mathbf{x}^{(k+1)}\|, 1\right\}$$

Usaremos a `linalg.norm` por conveniência. Por exemplo,

$$\|x\| = \sqrt{|x_1|^2 + \dots + |x_n|^2}$$

In [1]:
```
import numpy as np
from numpy import linalg as la

tol = 1e-6
x0 = np.array([[1],[2],[3]])
x = np.array([[4],[5],[6]])

la.norm(x-x0) < tol*max([la.norm(x),1.0])
```

Out[1]:
```
False
```

# Implementação

CHUTE

MÁX. DE ITERAÇÕES

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\underbrace{\left[-\sum_{\substack{j=1\\ j\neq i}}^{n}\left(a_{ij}x_j^{(k)}\right)+b_i\right]}_{\text{SOMA}}$$

In [2]:

```
def jacobi(A,b,x0,N,tol):
    m,n = A.shape
    x = np.zeros((n,1))
    k = 1
    while (k <= N):
        for i in range(n):
            soma = b[i]
            for j in range(n):
                if (i != j):
                    soma = soma - A[i,j]*x0[j]
            x[i] = soma/A[i,i]

        if la.norm(x-x0) < tol*max([la.norm(x),1]):
            return x,k

        k = k + 1
        for i in range(n):
            x0[i] = x[i]

    print("Número máximo de iterações foi excedido")
    return x,k
```

$\text{tol} \to \varepsilon$

$k = 1 \to$ Nº DA ITERAÇÃO

VETOR SOLUÇÃO $x^{(k+1)}$

SOMA

$x^{(k)}$

CONVERGIU

NÃO CONVERGIU

# Implementação

In [2]:
```python
def jacobi(A,b,x0,N,tol):
    m,n = A.shape
    x = np.zeros((n,1))
    k = 1
    while (k <= N):
        for i in range(n):
            soma = b[i]
            for j in range(n):
                if (i != j):
                    soma = soma - A[i,j]*x0[j]
            x[i] = soma/A[i,i]

        if la.norm(x-x0) < tol*max([la.norm(x),1]):
            return x,k

        k = k + 1
        for i in range(n):
            x0[i] = x[i]

    print("Número máximo de iterações foi excedido")
    return x,k
```

$\rightarrow O(n)$ REPETIÇÕES

$\rightarrow O(n)$ REPETIÇÕES

$$\begin{array}{r} \overbrace{2 \cdot O(n) \cdot O(n)}^{O(n^2)} \\ + \; 1 \cdot O(n) \\ \hline O(n^2) \end{array}$$

$\Downarrow$

TOTAL $O(N \cdot n^2)$

$\xrightarrow[N << n]{n} O(n^2)$

Quantos flops realiza o método de Jacobi?

# Exemplo

$$\begin{array}{ccccccc} 9x_1 & + & x_2 & + & x_3 & = & 1 \\ 2x_1 & + & 10x_2 & + & 3x_3 & = & 2 \\ 3x_1 & + & 4x_2 & + & 11x_3 & = & 3 \end{array}$$

In [3]:
```
# Teste de Jacobi
A = np.array([[9.0,  1.0,  1.0],
              [2.0, 10.0,  3.0],
              [3.0,  4.0, 11.0]])
b = np.array([[1.0],[2.0],[3.0]])
x0 = np.array([[0.0],[0.0],[0.0]])

jacobi(A,b,x0,100,1e-8)
```

Out[3]:
```
(array([[0.07438017],
        [0.12278631],
        [0.20779221]]),
 23)
```

# Método de Gauss-Seidel

$$
\begin{aligned}
x_1^{(k+1)} &= \frac{1}{a_{11}}\left(-a_{12}x_2^{(k)} - a_{13}x_3^{(k)} - \cdots - a_{1n}x_n^{(k)} + b_1\right)\\
x_2^{(k+1)} &= \frac{1}{a_{22}}\left(-a_{21}x_1^{\overset{k}{\cancel{(k+1)}}} - a_{23}x_3^{(k)} - \cdots - a_{2n}x_n^{(k)} + b_2\right)\\
&\vdots\\
x_n^{(k+1)} &= \frac{1}{a_{nn}}\left(-a_{n1}x_1^{\overset{k}{\cancel{(k+1)}}} - a_{n2}x_2^{\overset{k}{\cancel{(k+1)}}} - \cdots - a_{n,(n-1)}x_{n-1}^{\overset{k}{\cancel{(k+1)}}} + b_n\right)
\end{aligned}
$$

# MODELO COMPUTACIONAL SEQUENCIAL (UNIPROCESSADOR)

$$x_1^{(k+1)} = \frac{1}{9}\left(1 - x_2^{(k)} - x_3^{(k)}\right)$$

$$x_2^{(k+1)} = \frac{1}{10}\left(2 - 2x_1^{(k+1)} - 3x_3^{(k)}\right)$$

$$x_3^{(k+1)} = \frac{1}{11}\left(3 - 3x_1^{(k+1)} - 4x_2^{(k+1)}\right)$$

$$x_1^{(1)} = G_1\left(x_2^{(0)}, x_3^{(0)}\right)$$

$$x_2^{(1)} = G_2\left(x_1^{(1)}, x_3^{(0)}\right)$$

$$x_3^{(2)} = G_3\left(x_1^{(1)}, x_2^{(1)}\right)$$

$$\begin{cases} x_1^{(k+1)} = \frac{1}{9}\left(1 - x_2^{(k)} - x_3^{(k)}\right) \\ x_2^{(k+1)} = \frac{1}{10}\left(2 - 2x_1^{(k+1)} - 3x_3^{(k)}\right) \\ x_3^{(k+1)} = \frac{1}{11}\left(3 - 3x_1^{(k+1)} - 4x_2^{(k+1)}\right) \end{cases}$$

2ª EQUAÇÃO

$$x_2^{(k+1)} = \frac{1}{10}\left(2 - 2\overbrace{\left[\frac{1}{9}\left(1 - x_2^{(k)} - x_3^{(k)}\right)\right]}^{x_1^{(k+1)}} - 3x_3^{(k)}\right)$$

$$= \frac{1}{10}\left[2 - \frac{2}{9} + \frac{2}{9}x_2^{(k)} + \frac{2}{9}x_3^{(k)} - 3x_3^{(k)}\right]$$

$$= \frac{1}{10}\left(\frac{16}{9} + \frac{2}{9}x_2^{(k)} - \frac{25}{9}x_3^{(k)}\right) = \frac{8}{45} + \frac{1}{45}x_2^{(k)} - \frac{5}{18}x_3^{(k)}$$

$$\begin{cases} x_1^{(k+1)} = \frac{1}{9}\left(1 - x_2^{(k)} - x_3^{(k)}\right) \\ x_2^{(k+1)} = \frac{8}{45} + \frac{1}{45}x_2^{(k)} - \frac{5}{18}x_3^{(k)} \\ x_3^{(k+1)} = \frac{1}{11}\left(3 - 3x_1^{(k+1)} - 4x_2^{(k+1)}\right) \end{cases}$$

3ª EQUAÇÃO

$$x_3^{(k+1)} = \frac{1}{11}\left[3 - \cancel{3}^{1}\,\frac{1}{\cancel{9}_{3}}\left(1 - x_2^{(k)} - x_3^{(k)}\right) - 4\left(\frac{8}{45} + \frac{1}{45}x_2^{(k)} - \frac{5}{18}x_3^{(k)}\right)\right]$$

$$= \frac{1}{11}\left[3 - \frac{1}{3} + \frac{1}{3}x_2^{(k)} + \frac{1}{3}x_3^{(k)} - \frac{32}{45} - \frac{4}{45}x_2^{(k)} + \frac{10}{9}x_3^{(k)}\right]$$

$$= \frac{1}{11}\left[\frac{135 - 15 - 32}{45} + \ldots\right]$$ (P/ CASA)

## DESENVOLVIMENTO MATRICIAL:

$$9x_1^{(k+1)} = \frac{1}{\cancel{9}}\left(1 - x_2^{(k)} - x_3^{(k)}\right)$$

$$10x_2^{(k+1)} = \frac{1}{\cancel{10}}\left(2 - 2x_1^{(k+1)} - 3x_3^{(k)}\right)$$

$$11x_3^{(k+1)} = \frac{1}{\cancel{11}}\left(3 - 3x_1^{(k+1)} - 4x_2^{(k+1)}\right)$$

$$\begin{bmatrix} 9 & 0 & 0 \\ 2 & 10 & 0 \\ 3 & 4 & 11 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_{(k+1)} = \begin{bmatrix} 0 & -1 & -1 \\ 0 & 0 & -3 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_{(k)} + \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

$$\begin{aligned} 9x_1 &+ x_2 &+ x_3 &= 1 \\ 2x_1 &+ 10x_2 &+ 3x_3 &= 2 \\ 3x_1 &+ 4x_2 &+ 11x_3 &= 3 \end{aligned} \Rightarrow \begin{bmatrix} 9 & 1 & 1 \\ 2 & 10 & 3 \\ 3 & 4 & 11 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = b$$

$$\underbrace{\begin{bmatrix} 9 & 0 & 0 \\ 2 & 10 & 0 \\ 3 & 4 & 11 \end{bmatrix}}_{(L+D)} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}^{(k+1)} = \underbrace{\begin{bmatrix} 0 & -1 & -1 \\ 0 & 0 & -3 \\ 0 & 0 & 0 \end{bmatrix}}_{} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}^{(k)} + \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

$$x^{(k+1)} = -Ux^{(k)} + b$$

$$x^{(k+1)} = \underbrace{-(L+D)^{-1}U}_{} x^{(k)} + (L+D)^{-1} \cdot b$$

↳ MATRIZ DE GAUSS-SEIDEL

# Implementação

In [4]:
```python
def gauss_seidel(A,b,x0,N,tol):
    m,n = A.shape
    x = np.zeros((n,1))
    k = 1
    while (k <= N):
        for i in range(n):
            soma = b[i]
            # contribuição de x_k
            for j in range(i+1,n):
                soma = soma - A[i,j]*x0[j]

            # contribuição de x_{k+1}
            for j in range(0,i):
                soma = soma - A[i,j]*x[j]

            x[i] = soma/A[i,i]

        if la.norm(x-x0) < tol*max([la.norm(x),1]):
            return x,k

        k = k + 1
        for i in range(n):
            x0[i] = x[i]

    print("Número máximo de iterações foi excedido")
    return x,k
```

NOVO

$x^{(k)}$

$$\begin{cases} x_1^{(k+1)} = \frac{1}{9}\left(1 - x_2^{(k)} - x_3^{(k)}\right) \\ x_2^{(k+1)} = \frac{1}{10}\left(2 - 2x_1^{(k+1)} - 3x_3^{(k)}\right) \\ x_3^{(k+1)} = \frac{1}{11}\left(3 - 3x_1^{(k+1)} - 4x_2^{(k+1)}\right) \end{cases}$$

$$x_1^{(k+1)} \cdots x_{i-1}^{(k+1)} \quad x_i^{(k+1)} \quad x_{i+1}^{(k)} \cdots x_n^{(k)}$$

ATENTION

# Exemplo

In [5]:
```
# Teste de Gauss-Seidel
A = np.array([[9.0,  1.0,  1.0],
              [2.0, 10.0,  3.0],
              [3.0,  4.0, 11.0]])
b = np.array([[1.0],[2.0],[3.0]])
x0 = np.array([[0.0],[0.0],[0.0]])

gauss_seidel(A,b,x0,100,1e-8)
```

Out[5]:
```
(array([[0.07438017],
        [0.1227863 ],
        [0.20779221]]),
 9)
```

# Técnica de particionamento

Considere um sistema $\mathbf{Ax} = \mathbf{b}$ e escreva

$$\mathbf{A} = \mathbf{D} + \mathbf{L} + \mathbf{U}$$

- $\mathbf{D} = [\, a_{ii} \,], i = 1, 2, \ldots, n$
- $\mathbf{L} = [\, a_{ij} \,], i, j = 1, 2, \ldots, n, i > j$
- $\mathbf{U} = [\, a_{ij} \,], i, j = 1, 2, \ldots, n, i < j$

D U

$$\begin{bmatrix} 9 & 1 & 1 \\ 2 & 10 & 3 \\ 3 & 4 & 11 \end{bmatrix}$$

L

$$T_g = \underline{\underline{-(L+D)^{-1}U}}$$

$$(L+D) = \begin{bmatrix} 9 & 0 & 0 \\ 2 & 10 & 0 \\ 3 & 4 & 11 \end{bmatrix} \qquad -U = \begin{bmatrix} 0 & -1 & -1 \\ 0 & 0 & -3 \\ 0 & 0 & 0 \end{bmatrix}$$

NÃO É FATORAÇÃO $A = LU$

# Matriz de Jacobi

$$
\begin{aligned}
\mathbf{A}\mathbf{x} &= (\mathbf{D}+\mathbf{L}+\mathbf{U})\,\mathbf{x} = \mathbf{b} \\
\mathbf{D}\mathbf{x} &= -\,(\mathbf{L}+\mathbf{U})\,\mathbf{x} + \mathbf{b} \\
\mathbf{x}^{(k+1)} &= \underbrace{-\mathbf{D}^{-1}\,(\mathbf{L}+\mathbf{U})}\,\mathbf{x}^{(k)} + \underbrace{\mathbf{D}^{-1}\mathbf{b}} \\
&= \mathbf{T}_j\mathbf{x}^{(k)} + \mathbf{c}
\end{aligned}
$$

# Matriz de Jacobi

$$\begin{aligned} 9x_1 &+ & x_2 &+ & x_3 &= & 1 \\ 2x_1 &+ & 10x_2 &+ & 3x_3 &= & 2 \\ 3x_1 &+ & 4x_2 &+ & 11x_3 &= & 3 \end{aligned}$$

# Matriz de Jacobi

$$
\begin{array}{rcrcrcl}
9x_1 & + & x_2 & + & x_3 & = & \\
2x_1 & + & 10x_2 & + & 3x_3 & = & \\
3x_1 & + & 4x_2 & + & 11x_3 & = &
\end{array}
\overset{b}{\boxed{\begin{array}{c} 1 \\ 2 \\ 3 \end{array}}}
$$

$$
\mathbf{T}_j = \begin{bmatrix} 0 & -\frac{1}{9} & -\frac{1}{9} \\ -\frac{2}{10} & 0 & -\frac{3}{10} \\ -\frac{3}{11} & -\frac{4}{11} & 0 \end{bmatrix}
\rightsquigarrow x^{(k+1)} = T_j x^{(k)} + \underset{\displaystyle D^{-1}b}{\underset{\downarrow}{c}}
$$

# Matriz de Jacobi

$$
\begin{array}{rcrcrcl}
9x_1 & + & x_2 & + & x_3 & = & 1 \\
2x_1 & + & 10x_2 & + & 3x_3 & = & 2 \\
3x_1 & + & 4x_2 & + & 11x_3 & = & 3
\end{array}
$$

$$
\mathbf{T}_j = \begin{bmatrix} 0 & -\frac{1}{9} & -\frac{1}{9} \\ -\frac{2}{10} & 0 & -\frac{3}{10} \\ -\frac{3}{11} & -\frac{4}{11} & 0 \end{bmatrix}
$$

```
In [6]: A = np.array([[9.0,  1.0,  1.0],
                      [2.0, 10.0,  3.0],
                      [3.0,  4.0, 11.0]])
        D = np.diag(np.diag(A))
        L = np.tril(A,-1)
        U = np.triu(A,1)
        Tj = -np.dot(la.inv(D),(L+U))
        Tj
```

$T_j = -D^{-1}(L+U)$

```
Out[6]: array([[-0.        , -0.11111111, -0.11111111],
               [-0.2       , -0.        , -0.3       ],
               [-0.27272727, -0.36363636, -0.        ]])
```

# Matriz de Gauss-Seidel

$$
\begin{aligned}
\mathbf{A}\mathbf{x} &= (\mathbf{D}+\mathbf{L}+\mathbf{U})\,\mathbf{x} = \mathbf{b} \\
(\mathbf{D}+\mathbf{L})\,\mathbf{x} &= -\mathbf{U}\mathbf{x} + \mathbf{b} \\
\mathbf{x}^{(k+1)} &= \underbrace{-(\mathbf{D}+\mathbf{L})^{-1}\mathbf{U}}\mathbf{x}^{(k)} + \underbrace{(\mathbf{D}+\mathbf{L})^{-1}\mathbf{b}} \\
&= \mathbf{T}_g\mathbf{x}^{(k)} + \mathbf{c}
\end{aligned}
$$

# Matriz de Gauss-Seidel

$$\begin{array}{rcrcrcl}
9x_1 & + & x_2 & + & x_3 & = & 1 \\
2x_1 & + & 10x_2 & + & 3x_3 & = & 2 \\
3x_1 & + & 4x_2 & + & 11x_3 & = & 3
\end{array}$$

# Matriz de Gauss-Seidel

$$\begin{array}{rcrcrcl} 9x_1 & + & x_2 & + & x_3 & = & 1 \\ 2x_1 & + & 10x_2 & + & 3x_3 & = & 2 \\ 3x_1 & + & 4x_2 & + & 11x_3 & = & 3 \end{array}$$

$$\mathbf{T}_g = \begin{bmatrix} 0 & -\frac{1}{9} & -\frac{1}{9} \\ 0 & \frac{1}{45} & -\frac{5}{18} \\ 0 & \frac{1}{45} & \frac{13}{99} \end{bmatrix} = -(L+D)^{-1}U$$

# Matriz de Gauss-Seidel

$$
\begin{array}{ccccccc}
9x_1 & + & x_2 & + & x_3 & = & 1 \\
2x_1 & + & 10x_2 & + & 3x_3 & = & 2 \\
3x_1 & + & 4x_2 & + & 11x_3 & = & 3
\end{array}
$$

$$
\mathbf{T}_g = \begin{bmatrix} 0 & -\frac{1}{9} & -\frac{1}{9} \\ 0 & \frac{1}{45} & -\frac{5}{18} \\ 0 & \frac{1}{45} & \frac{13}{99} \end{bmatrix}
$$

In [7]:
```
A = np.array([[9.0,  1.0,  1.0],
              [2.0, 10.0,  3.0],
              [3.0,  4.0, 11.0]])
D = np.diag(np.diag(A))
L = np.tril(A,-1)
U = np.triu(A,1)
Tg = -np.dot(la.inv(D+L),U)
Tg
```

Out[7]:
```
array([[-0.        , -0.11111111, -0.11111111],
       [-0.        ,  0.02222222, -0.27777778],
       [-0.        ,  0.02222222,  0.13131313]])
```

## Critério de convergência (NECESSÁRIO E SUFICIENTE)

**Teorema.** Um método iterativo da forma

$\rightarrow Ax = b$

$$\boxed{\mathbf{x}^{(k+1)} = \mathbf{M}\mathbf{x}^{(k)} + \mathbf{c}} \qquad k = 0, 1, \ldots$$

converge para qualquer valor inicial $\mathbf{x}^{(0)}$ se, e somente se,

$$\rho(\mathbf{M}) = \max_{1 \le i \le n} |\lambda_i| < 1$$

$\hookrightarrow$ RAIO ESPECTRAL DE M

# Critério de convergência

$$
\begin{array}{rcrcrcl}
9x_1 & + & x_2 & + & x_3 & = & 1 \\
2x_1 & + & 10x_2 & + & 3x_3 & = & 2 \\
3x_1 & + & 4x_2 & + & 11x_3 & = & 3
\end{array}
$$

# Critério de convergência

$$
\begin{array}{rcrcrcl}
9x_1 & + & x_2 & + & x_3 & = & 1 \\
2x_1 & + & 10x_2 & + & 3x_3 & = & 2 \\
3x_1 & + & 4x_2 & + & 11x_3 & = & 3
\end{array}
$$

$$
\mathbf{T}_j = \begin{bmatrix} 0 & -\frac{1}{9} & -\frac{1}{9} \\ -\frac{2}{10} & 0 & -\frac{3}{10} \\ -\frac{3}{11} & -\frac{4}{11} & 0 \end{bmatrix}
$$

# Critério de convergência

$$\begin{array}{rcrcrcl} 9x_1 & + & x_2 & + & x_3 & = & 1 \\ 2x_1 & + & 10x_2 & + & 3x_3 & = & 2 \\ 3x_1 & + & 4x_2 & + & 11x_3 & = & 3 \end{array}$$

$$\mathbf{T}_j = \begin{bmatrix} 0 & -\frac{1}{9} & -\frac{1}{9} \\ -\frac{2}{10} & 0 & -\frac{3}{10} \\ -\frac{3}{11} & -\frac{4}{11} & 0 \end{bmatrix}$$

In [8]:
```
abs(la.eigvals(Tj))
```

Out[8]:
```
array([0.44722715, 0.11587747, 0.33134968])
```

$\rho(T_j)$

# Critério de convergência

$$\begin{array}{rcrcrcl}
9x_1 & + & x_2 & + & x_3 & = & 1 \\
2x_1 & + & 10x_2 & + & 3x_3 & = & 2 \\
3x_1 & + & 4x_2 & + & 11x_3 & = & 3
\end{array}$$

# Critério de convergência

$$\begin{aligned} 9x_1 &+ x_2 &+ x_3 &= 1 \\ 2x_1 &+ 10x_2 &+ 3x_3 &= 2 \\ 3x_1 &+ 4x_2 &+ 11x_3 &= 3 \end{aligned}$$

$$\mathbf{T}_g = \begin{bmatrix} 0 & -\frac{1}{9} & -\frac{1}{9} \\ 0 & \frac{1}{45} & -\frac{5}{18} \\ 0 & \frac{1}{45} & \frac{13}{99} \end{bmatrix}$$

# Critério de convergência

$$
\begin{array}{rcrcrcr}
9x_1 & + & x_2 & + & x_3 & = & 1 \\
2x_1 & + & 10x_2 & + & 3x_3 & = & 2 \\
3x_1 & + & 4x_2 & + & 11x_3 & = & 3
\end{array}
$$

$$
\mathbf{T}_g = \begin{bmatrix} 0 & -\frac{1}{9} & -\frac{1}{9} \\ 0 & \frac{1}{45} & -\frac{5}{18} \\ 0 & \frac{1}{45} & \frac{13}{99} \end{bmatrix}
$$

```
In [9]: abs(la.eigvals(Tg))

Out[9]: array([0.        , 0.09534626, 0.09534626])
```

$\rho(T_g)$

# Velocidade de convergência

A velocidade de convergência de um método iterativo depende do raio espectral da matriz do método. Embora não tenhamos um resultado para sistemas em geral, em alguns casos é possível prever qual deles convergirá mais rápido.

# Velocidade de convergência

$$\begin{array}{rcrcrcl} 9x_1 & + & x_2 & + & x_3 & = & 1 \\ 2x_1 & + & 10x_2 & + & 3x_3 & = & 2 \\ 3x_1 & + & 4x_2 & + & 11x_3 & = & 3 \end{array}$$

In [11]:
```
abs(la.eigvals(Tj)).max()
```
Out[11]: 0.4472271510919682

In [12]:
```
abs(la.eigvals(Tg)).max()
```
Out[12]: 0.09534625892455925

# Velocidade de convergência

$$
\begin{array}{rcrcrcl}
7x_1 & + & 6x_2 & + & 9x_3 & = & 2 \\
4x_1 & + & 5x_2 & - & 4x_3 & = & 1 \\
-7x_1 & - & 3x_2 & + & 8x_3 & = & 3
\end{array}
$$

In [13]:
```
A = np.array([[ 7.0, 6.0, 9.0],
              [ 4.0, 5.0,-4.0],
              [-7.0,-3.0, 8.0]])
D = np.diag(np.diag(A))
L = np.tril(A,-1)
U = np.triu(A,1)
Tj  = -np.dot(la.inv(D),(L+U))
Tg  = -np.dot(la.inv(D+L),U)
```

In [14]:
```
abs(la.eigvals(Tj)).max()
```

Out[14]: 0.641132809955697

In [15]:
```
abs(la.eigvals(Tg)).max()
```

Out[15]: 0.7745966692414834

## Saiba mais

- Estas anotações foram baseadas na Seção 7.3 de nosso livro-texto: Burden, R. L.; Faires, D.; Burden, A. M., **Análise Numérica**, 3ª ed., Cengage Learning: São Paulo, 2015.

- Se houver tempo, assista aos vídeos da professora Emanuele da UFC aqui. O material dela é excelente!

- Para aprender sobre a notação assintótica utilizada na representação da quantidade de flops ("Oh-zão" ou Oh-grande), recomendo assistir este vídeo da professora Carla (UFABC) ou ler este material do professor Paulo Feofiloff (IME/USP).

Vicente Helano
UFCA | Centro de Ciências e Tecnologia