# Capítulo 4

**Objetivos Específicos de Aprendizagem**

Ao finalizar este Capítulo, você será capaz de:

- Aplicar métodos iterativos para obter a solução de um sistema não linear e avaliar a precisão do resultado obtido; e
- Utilizar os algoritmos disponibilizados.

## 4 Resolução de Sistemas de Equações não Lineares

No Capítulo 2, abordamos a solução de sistemas de equações lineares $A * X = B$ e, no Capítulo 3, tratamos da solução de equações não lineares com uma única incógnita $f(x) = 0$. Neste Capítulo, veremos que, na modelagem matemática dos fenômenos físicos, normalmente estão envolvidas relações **não lineares** entre várias incógnitas, as quais geram um sistema de equações não lineares.

Vamos iniciar o estudo das equações não lineares apresentando duas definições importantes:

Definição 1: **sistema de** $n$ **equações** não lineares com $n$ incógnitas é toda expressão do tipo:

$$\begin{cases} f_1(x_1, x_2, x_3, ..., x_n) = 0 \\ f_2(x_1, x_2, x_3, ..., x_n) = 0 \\ \quad \vdots \\ f_n(x_1, x_2, x_3, ..., x_n) = 0 \end{cases} \Rightarrow F(X) = 0$$

Em que

$$F(X) = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} \text{ e } X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Observe que temos um vetor $F$ de $n$ funções e um vetor $X$ de $n$ incógnitas. Por exemplo:

a) $$\begin{cases} f_1(x_1, x_2) = x_1 + 2x_2^3 - 3 = 0 \\ f_2(x_1, x_2) = 3x_1^2 + x_2^2 - 7 = 0 \end{cases} \Rightarrow F = \begin{bmatrix} f_1(x_1, x_2) \\ f_2(x_1, x_2) \end{bmatrix} \text{ e } X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

b) $$\begin{cases} f_1(x_1, x_2, x_3) = 3x_1 - \cos(x_1 x_3) - 0.5 = 0 \\ f_2(x_1, x_2, x_3) = x_1^3 - x_1 x_2 x_3 - 5 = 0 \\ f_3(x_1, x_2, x_3) = e^{x_1 x_2} - \ln(x_1^2 + x_2 x_3) = 0 \end{cases} \Rightarrow F = \begin{bmatrix} f_1(x_1, x_2, x_3) \\ f_2(x_1, x_2, x_3) \\ f_3(x_1, x_2, x_3) \end{bmatrix} \text{ e } X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Definição 2: **solução** de $F(X) = 0$ é todo vetor $\alpha = \{\alpha_1, \alpha_2, ..., \alpha_n\} \in \mathbb{C} / F(\alpha) = 0$.

Por exemplo:

$$\begin{cases} x_1 x_2 = 1 \\ x_1^2 + x_2^2 - 4x_1 + 2x_2 = 4 \end{cases} \Rightarrow \begin{cases} f_1(x_1, x_2) = x_1 x_2 - 1 = 0 \\ f_2(x_1, x_2) = x_1^2 + x_2^2 - 4x_1 + 2x_2 - 4 = 0 \end{cases}$$

$$\alpha = \begin{bmatrix} 0.604068 \\ 1.655442 \end{bmatrix}$$

Note a dificuldade de isolar, via métodos abstrativos, a solução $\alpha$ já a partir de $n = 2$ equações, restando, por consequência, o uso de métodos iterativos.

Aqui vamos apresentar três métodos iterativos da família newtoniana, que podem ser considerados extensões adaptadas daqueles que vimos no Capítulo 3, para resolver uma única equação a uma variável.

A solução de um sistema não linear consiste em determinar pontos no subespaço $n$ dimensional do domínio do problema que satisfaçam o conjunto de equações. Para os sistemas lineares, temos apenas três possibilidades de quantidade de soluções: **não existência**, **solução única** ou **infinitas soluções**. Já para os sistemas não lineares, podemos ter **nenhuma**, **uma**, um **número finito** ou até um **número infinito** de soluções e inexistem formas de determinar algebricamente a quantidade de soluções de um sistema $F(X) = 0$.

No **Exemplo 4.1**, vamos apresentar um sistema de duas equações não lineares com duas incógnitas, no qual geometricamente as quatro soluções são os pontos de interseção dos gráficos das funções geradoras das duas equações.

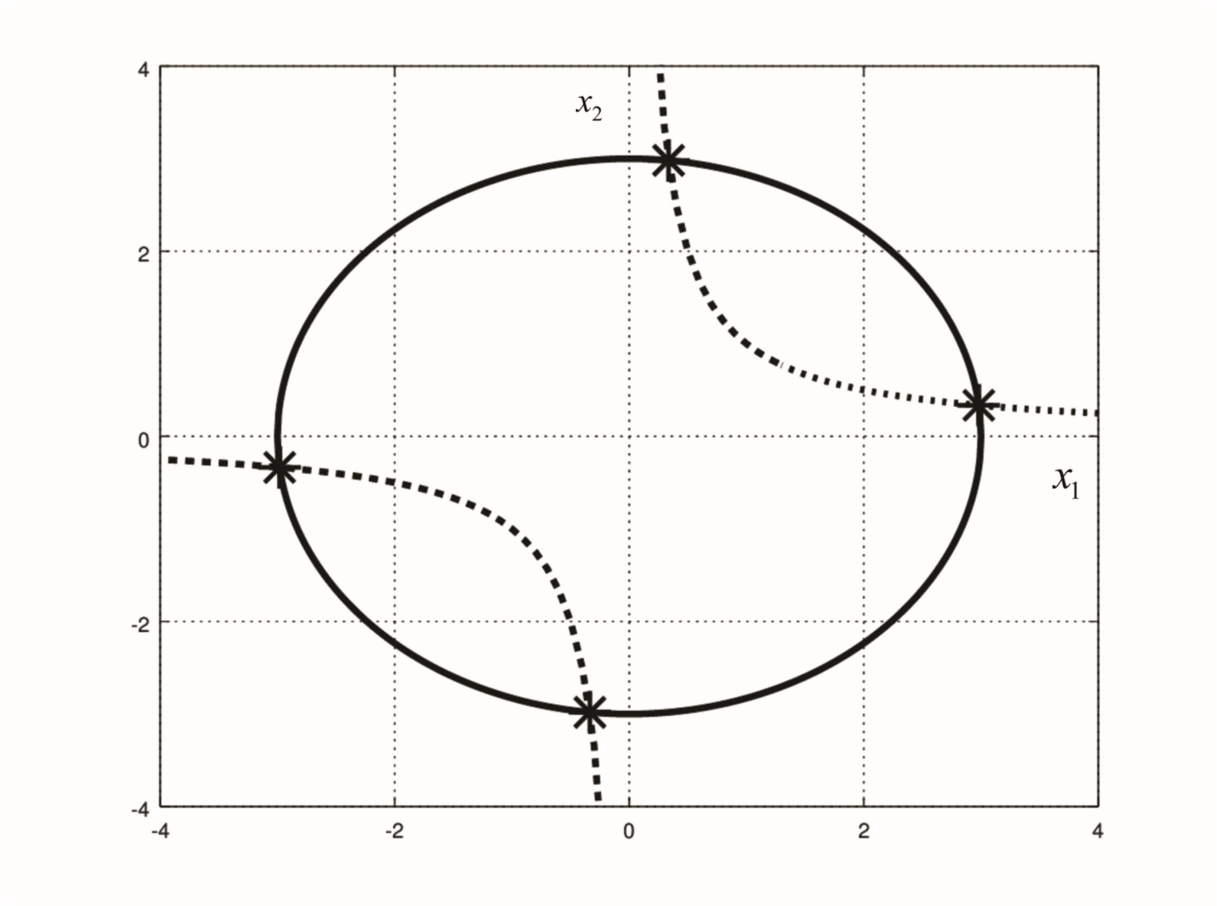
**Exemplo 4.1**: localize geometricamente as soluções de:

$$\begin{cases} x_1^2 + x_2^2 - 9 = 0 \\ x_1 x_2 - 1 = 0 \end{cases}$$

**Solução**:

Como este exemplo envolve apenas duas equações simples com duas incógnitas, basta gerar os gráficos de $x_2$ em função de $x_1$, correspondendo a uma circunferência na primeira e uma hipérbole na segunda, resultando em:

Gráfico 4.1 – Intersecção dos gráficos das funções do **Exemplo 4.1**




Fonte: Elaboração própria

Portanto, no Gráfico 4.1, temos quatro pontos de intersecção das duas funções, que são as quatro soluções do sistema dado.

A aproximação geométrica de uma solução Real ($\mathbb{R}$), ou a determinação de valores iniciais de uma solução Real, somente é possível para sistemas de duas ou três equações.

Para uma solução geral por métodos iterativos, normalmente tomamos valores iniciais baseados em conhecimentos sobre as grandezas físicas envolvidas no modelo matemático representado pelo sistema de equações. Além disso, alguns sistemas não lineares podem ter raízes Complexas ($\mathbb{C}$), sendo necessário fornecer uma solução inicial Complexa.

Nas próximas seções, vamos apresentar três métodos iterativos da família newtoniana, pois os de outras famílias necessitam de fundamentações que estão além do escopo deste livro.

## 4.1 Método de Newton

No sistema não linear $F(X)=0$,

$$\begin{cases} f_1(x_1, x_2, ....., x_n) = 0 \\ f_2(x_1, x_2, ....., x_n) = 0 \\ \quad \vdots \\ f_n(x_1, x_2, ....., x_n) = 0 \end{cases} \tag{1}$$

Tomando um vetor solução inicial $X^{(0)}$, calculamos um vetor incremento $\Delta X = X - X^{(0)}$, e obtemos o vetor solução aproximada $X = X^{(0)} + \Delta X$, em que:

$$X^{(0)} = \begin{bmatrix} x_1^{(0)} \\ x_2^{(0)} \\ \vdots \\ x_n^{(0)} \end{bmatrix}, \quad \Delta X = \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \vdots \\ \Delta x_n \end{bmatrix} = \begin{bmatrix} x_1 - x_1^{(0)} \\ x_2 - x_2^{(0)} \\ \vdots \\ x_n - x_n^{(0)} \end{bmatrix} \quad \text{e} \quad X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Aplicando a expansão em série de Taylor em cada uma das $n$ funções não lineares $f_i(x_1, x_2, ....., x_n)$ em torno do ponto inicial $X^{(0)}$, com $\Delta x_j = x_j - x_j^{(0)}$, $j = 1, \ldots, n$, analogamente à expansão em série de Taylor aplicada no método de Newton apresentado no Capítulo 3, temos:

$$\begin{cases} f_1(X) = f_1(X^{(0)}) + \dfrac{\partial f_1(X^{(0)})}{\partial x_1}\Delta x_1 + \dfrac{\partial f_1(X^{(0)})}{\partial x_2}\Delta x_2 + .... + \dfrac{\partial f_1(X^{(0)})}{\partial x_n}\Delta x_n + O(\Delta x_j)^2 \\ f_2(X) = f_2(X^{(0)}) + \dfrac{\partial f_2(X^{(0)})}{\partial x_1}\Delta x_1 + \dfrac{\partial f_2(X^{(0)})}{\partial x_2}\Delta x_2 + .... + \dfrac{\partial f_2(X^{(0)})}{\partial x_n}\Delta x_n + O(\Delta x_j)^2 \\ \quad \vdots \\ f_n(X) = f_n(X^{(0)}) + \dfrac{\partial f_n(X^{(0)})}{\partial x_1}\Delta x_1 + \dfrac{\partial f_n(X^{(0)})}{\partial x_2}\Delta x_2 + .... + \dfrac{\partial f_n(X^{(0)})}{\partial x_n}\Delta x_n + O(\Delta x_j)^2 \end{cases} \tag{2}$$

Em cada $f_i(X) = 0$, $i = 1, \ldots, n$, desprezando os termos de ordem superior $O(\Delta x_j)^2$ e reescrevendo a eq. (2) para a forma matricial, resulta o seguinte sistema linear:

$$\begin{bmatrix} \frac{\partial f_1(X^{(0)})}{\partial x_1} & \frac{\partial f_1(X^{(0)})}{\partial x_2} & \cdots & \frac{\partial f_1(X^{(0)})}{\partial x_n} \\ \frac{\partial f_2(X^{(0)})}{\partial x_1} & \frac{\partial f_2(X^{(0)})}{\partial x_2} & \cdots & \frac{\partial f_n(X^{(0)})}{\partial x_n} \\ & \vdots & & \\ \frac{\partial f_n(X^{(0)})}{\partial x_1} & \frac{\partial f_n(X^{(0)})}{\partial x_2} & \cdots & \frac{\partial f_n(X^{(0)})}{\partial x_n} \end{bmatrix} * \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \vdots \\ \Delta x_n \end{bmatrix} = - \begin{bmatrix} f_1(X^{(0)}) \\ f_2(X^{(0)}) \\ \vdots \\ f_n(X^{(0)}) \end{bmatrix} \quad (3)$$

Resolvendo por eliminação gaussiana o sistema linear dado pela eq. (3), determinamos os valores de uma aproximação para cada $\Delta x_j$ e posteriormente obtemos os novos valores das incógnitas do sistema não linear via $x_j = x_j^{(0)} + \Delta x_j$, $j = 1, \ldots, n$. Esse processo precisa ser repetido para melhorar a precisão da solução, atualizando o conjunto de valores iniciais $X^{(0)}$ pelos novos valores $X$ calculados. Com o novo $X^{(0)}$, geramos e resolvemos o sistema linear novamente e o atualizamos até que algum critério de parada seja satisfeito. Tal critério pode ser o mesmo entre aqueles utilizados na solução iterativa de sistemas lineares.

Observe que, considerando um contador de iteração $k$, o método de Newton para sistemas de equações não lineares torna-se:

$$J(X^{(k)}) * \Delta X = -F(X^{(k)}) \quad (4)$$

$$X^{(k+1)} = X^{(k)} + \Delta X \quad (5)$$

Em que

$$J(X) = \begin{bmatrix} \frac{\partial f_1(X)}{\partial x_1} & \frac{\partial f_1(X)}{\partial x_2} & \cdots & \frac{\partial f_1(X)}{\partial x_n} \\ \frac{\partial f_2(X)}{\partial x_1} & \frac{\partial f_2(X)}{\partial x_2} & \cdots & \frac{\partial f_n(X)}{\partial x_n} \\ & \vdots & & \\ \frac{\partial f_n(X)}{\partial x_1} & \frac{\partial f_n(X)}{\partial x_2} & \cdots & \frac{\partial f_n(X)}{\partial x_n} \end{bmatrix} \quad (6)$$

A matriz $J(X)$ contém todas as derivadas parciais de 1ª ordem possíveis da função $F(X)$ e é denominada de j**acobiana** dessa função.

**Exemplo 4.2**: resolva $\begin{cases} e^{x_1} + x_2 = 1 \\ x_1^2 + x_2^2 = 4 \end{cases}$ por Newton, com 3 iterações a partir de $X^{(0)} = \begin{bmatrix} +1 \\ -1 \end{bmatrix}$, e calcule o erro de truncamento da solução atingida.

**Solução**:

Temos

$$F(X) = \begin{cases} f_1(x_1, x_2) = e^{x_1} + x_2 - 1 = 0 \\ f_2(x_1, x_2) = x_1^2 + x_2^2 - 4 = 0 \end{cases}$$

Obtendo a jacobiana e o sistema de equações lineares para essas duas equações, conforme eqs. (4) e (5), temos:

$$\begin{bmatrix} \frac{\partial f_1(X)}{\partial x_1} & \frac{\partial f_1(X)}{\partial x_2} \\ \frac{\partial f_2(X)}{\partial x_1} & \frac{\partial f_2(X)}{\partial x_2} \end{bmatrix} * \begin{bmatrix} \Delta x_1 \\ \\ \Delta x_2 \end{bmatrix} = - \begin{bmatrix} f_1(X) \\ \\ f_2(X) \end{bmatrix}$$

$$\begin{bmatrix} e^{x_1} & 1 \\ 2x_1 & 2x_2 \end{bmatrix} * \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} = - \begin{bmatrix} e^{x_1} + x_2 - 1 \\ (x_1)^2 + (x_2)^2 - 4 \end{bmatrix} \qquad (*)$$

**Primeira iteração**: aplicamos $X^{(0)} = \begin{bmatrix} +1 \\ -1 \end{bmatrix}$ em (*)

$$\begin{bmatrix} e^1 & 1 \\ 2 & -2 \end{bmatrix} * \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} = - \begin{bmatrix} e-2 \\ -2 \end{bmatrix} \Rightarrow \begin{cases} \Delta x_1 = +0.075766 \\ \Delta x_2 = -0.924234 \end{cases} \Rightarrow \begin{cases} x_1^{(1)} = +1.0758 \\ x_2^{(1)} = -1.9242 \end{cases}$$

Critério de parada: $\sum_{j=1}^{n} \left| \Delta x_j \right| = 1.000000$.

**Segunda iteração**: aplicamos $X^{(1)} = \begin{bmatrix} +1.0758 \\ -1.9242 \end{bmatrix}$ em (*)

$$\begin{bmatrix} 2.9322372 & 1 \\ 2.1515314 & -3.8484686 \end{bmatrix} * \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} = - \begin{bmatrix} 0.0080029 \\ 0.8599495 \end{bmatrix} \Rightarrow \begin{cases} \Delta x_1 = -0.066295 \\ \Delta x_2 = +0.186389 \end{cases}$$

$$\begin{cases} x_1^{(2)} = +1.0095 \\ x_2^{(2)} = -1.7378 \end{cases}$$

Critério de parada: $\sum_{j=1}^{n} \left| \Delta x_j \right| = 0.2527$.

**Terceira iteração**: aplicamos $X^{(2)} = \begin{bmatrix} +1.0095 \\ -1.7378 \end{bmatrix}$ em (*)

$$\begin{bmatrix} 2.7441484 & 1 \\ 2.0189416 & -3.4756897 \end{bmatrix} * \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} = -\begin{bmatrix} 0.0080029 \\ 0.0391360 \end{bmatrix} \Rightarrow \begin{cases} \Delta x_1 = -0.0052822 \\ \Delta x_2 = +0.0081916 \end{cases}$$

$$\begin{cases} x_1^{(3)} = +1.0042 \\ x_2^{(3)} = -1.7297 \end{cases}$$

Critério de parada: $\sum_{j=1}^{n} \left| \Delta x_j \right| = 0.013474$.

Podemos obter o erro de truncamento dessa solução aproximada alcançada na terceira iteração $\begin{cases} x_1^{(3)} = \ \ 1.0042 \\ x_2^{(3)} = -1.7297 \end{cases}$, com critério de parada $\sum_{j=1}^{n} \left| \Delta x_j \right| = 0.013474$, por comparação com uma solução estimada com o dobro de iterações, conforme vimos no Capítulo 2 para soluções de sistemas lineares por métodos iterativos. Nesse exemplo, a solução calculada em variável *double* atinge critério de convergência numericamente nulo em 6 iterações, no limite da precisão envolvida:

$$\begin{cases} x_1^{(6)} = +1.00416873847466 \\ x_2^{(6)} = -1.72963728702587 \end{cases}$$

Então, os erros de truncamento de cada incógnita $x_1$ e $x_2$ são:

Erro de truncamento $X^{(3)} = \begin{bmatrix} \left| \ \ 1.0042 - (\ \ 1.00416873847466) \right| \\ \left| -1.7297 - (-1.72963728702587) \right| \end{bmatrix} = \begin{bmatrix} 3.1262e-05 \\ 6.2713e-05 \end{bmatrix}$

O erro máximo de truncamento da solução obtida com 3 iterações é $6.2713 * 10^{-05}$, enquanto o critério de parada $\sum_{j=1}^{n} \left| \Delta x_j \right|$ atingido em 3 iterações é $0.013474$, então, nesse exemplo, o critério de parada também pode ser usado como limite superior do erro de truncamento.

Nos sistemas de pequeno porte, em que esse método seja convergente, podemos rapidamente chegar à solução com precisão no limite das variáveis adotadas, ou seja, atingir a solução com 16 dígitos significativos para variável *double*, e obter erro de truncamento na mesma ordem dos erros de arredondamento.

**Exemplo 4.3**: resolva o sistema de equações não lineares

$$\begin{cases} x_1^{\,2} + x_2^2 + x_3^2 - 1 = 0 \\ 2x_1^{\,2} + x_2^2 - 4x_3 = 0 \\ 3x_1^{\,2} - 4x_2 + x_3^2 = 0 \end{cases}$$

Utilizando o método de Newton. Tome o vetor inicial $X^{(0)} = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \end{bmatrix}$ e critério de parada $\max|\Delta x_i| < 0.005$.

**Solução**:

A partir de

$$F(X) = \begin{cases} f_1(x_1, x_2, x_3) = x_1^{\,2} + x_2^2 + x_3^2 - 1 \\ f_2(x_1, x_2, x_3) = 2x_1^{\,2} + x_2^2 - 4x_3 \\ f_3(x_1, x_2, x_3) = 3x_1^{\,2} - 4x_2 + x_3^2 \end{cases}$$

obtemos a matriz jacobiana,

$$J(X) = \begin{bmatrix} 2x_1 & 2x_2 & 2x_3 \\ 4x_1 & 2x_2 & -4 \\ 6x_1 & -4 & 2x_3 \end{bmatrix}$$

Aplicando iterativamente as eqs. (4) e (5), temos:

**Primeira iteração**: para $k = 0$ e $X^{(0)} = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \end{bmatrix}$, temos que:

$$J(X^{(0)}) = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 1 & -4 \\ 3 & -4 & 1 \end{bmatrix}$$

$$F(X^{(0)}) = \begin{cases} f_1 = \ 0.5^2 + 0.5^2 + 0.5^2 - 1 = -0.25 \\ f_2 = 2 * 0.5^2 + 0.5^2 - 4 * 0.5 = -1.25 \\ f_3 = 3 * 0.5^2 - 4 * 0.5 + 0.5^2 = -1.00 \end{cases}$$

Aplicando a eq. (3), temos o sistema linear:

$$\begin{bmatrix} 1 & 1 & 1 \\ 2 & 1 & -4 \\ 3 & -4 & 1 \end{bmatrix} * \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \Delta x_3 \end{bmatrix} = - \begin{bmatrix} -0.25 \\ -1.25 \\ -1.00 \end{bmatrix}$$

Cuja solução é $\Delta X^{(0)} = \begin{bmatrix} 0.375 \\ 0 \\ -0.125 \end{bmatrix}$.

Então, os novos valores do vetor $X$ são dados por:

$$X^{(1)} = X^{(0)} + \Delta X = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \end{bmatrix} + \begin{bmatrix} 0.375 \\ 0 \\ -0.125 \end{bmatrix} = \begin{bmatrix} 0.875 \\ 0.500 \\ 0.375 \end{bmatrix} \quad \text{com} \quad \max\left|\Delta x_j\right| = 0.875.$$

**Segunda iteração**: para $k = 1$, temos:

$$X^{(2)} = \begin{bmatrix} 0.7898166 \\ 0.4966216 \\ 0.3699324 \end{bmatrix} \text{ com } \max\left|\Delta x_j\right| = 0.0851834.$$

**Terceira iteração**: para $k = 2$, temos:

$$X^{(3)} = \begin{bmatrix} 0.7852104 \\ 0.4966114 \\ 0.3699228 \end{bmatrix} \text{ com } \max\left|\Delta x_j\right| = 0.0046062.$$

O processo convergiu para $X^{(3)}$ com critério de parada $\max\left|\Delta x_j\right| < 0.005$ em 3 iterações. Mantendo o mesmo procedimento, podemos chegar a uma solução convergida no limite da precisão das variáveis *double*, em 6 iterações e $\max\left|\Delta x_j\right| = 0.0$, dada por:

$$X^{(6)} = \begin{bmatrix} -0.785196933062355 \\ -0.496611392944656 \\ -0.369922830745872 \end{bmatrix}$$

Então, os erros de truncamento de cada incógnita ($x_1$, $x_2$ e $x_3$) obtida em 3 iterações seriam:

$$\text{Erro Truncamento } X^{(3)} = \begin{bmatrix} |0.7852104 - 0.785196933062355| \\ |0.4966114 - 0.496611392944656| \\ |0.3699228 - 0.369922830745872| \end{bmatrix} = \begin{bmatrix} 1.306694e-05 \\ 1.392945e-06 \\ 2.830746e-06 \end{bmatrix}$$

O erro máximo de truncamento da solução obtida com 3 iterações é $1.306694e-05$, enquanto o critério de convergência $\max\left|\Delta x_j\right|$ atingido em 3 iterações é $4.6e-03$; portanto, também nesse exemplo, o critério de

convergência pode ser usado como limite superior do erro de truncamento. E o erro máximo de truncamento da solução exata, estimada em 6 iterações $X^{(6)}$, está na ordem dos erros de arredondamento, pois atingiu critério de convergência numericamente nulo, $\max\left|\Delta x_j\right| = 0.0$.

### 4.2 Método de Newton com Derivadas Parciais Numéricas

Além da necessidade de obter previamente uma solução inicial $X^{(0)}$, o maior problema do método Newton é ter que calcular as $n^2$ derivadas parciais para obter a matriz jacobiana $J(X)$, conforme eq. (6).

A alternativa mais simples para contornar esse problema consiste em simular as $n^2$ derivadas parciais $\frac{\partial f_i(X)}{\partial x_j}$ através de aproximações numéricas, conforme feito no método da secante do Capítulo 3.

Por definição, sabemos que a derivada exata de uma função com uma variável é:

$$f'(x) = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

Por extensão, a derivada parcial exata de uma função de várias variáveis é definida por:

$$\frac{\partial f_i(x_1,\ \ldots, x_n)}{\partial x_j} = \lim_{\Delta x_j \to 0} \frac{f_i(x_1, x_2, ..., x_j + \Delta x_j, ..., x_n) - f_i(x_1, x_2, \ldots, x_j, ..., x_n)}{\Delta x_j}$$

Se utilizarmos um $\Delta x_j$ relativamente pequeno, também podemos obter uma aproximação para as derivadas parciais:

$$\frac{\partial f_i(x_1,\ \ldots, x_n)}{\partial x_j} \cong \frac{f_i(x_1, x_2, ..., x_j + \Delta x_j, ..., x_n) - f_i(x_1, x_2, \ldots, x_j, ..., x_n)}{\Delta x_j}$$

Então, geramos uma aproximação numérica da matriz jacobiana através de:

$$J(X) \cong \begin{bmatrix} \frac{f_1(x_1+\Delta x_1, x_2, ..., x_n) - f_1(x_1, x_2, ..., x_n)}{\Delta x_1} & .... & \frac{f_1(x_1, x_2, ..., x_n+\Delta x_n) - f_1(x_1, x_2, ..., x_n)}{\Delta x_n} \\ \frac{f_2(x_1+\Delta x_1, x_2, ..., x_n) - f_2(x_1, x_2, ..., x_n)}{\Delta x_1} & .... & \frac{f_2(x_1, x_2, ..., x_n+\Delta x_n) - f_2(x_1, x_2, ..., x_n)}{\Delta x_n} \\ \vdots & & \vdots \\ \frac{f_n(x_1+\Delta x_1, x_2, ..., x_n) - f_n(x_1, x_2, ..., x_n)}{\Delta x_1} & .... & \frac{f_n(x_1, x_2, ..., x_n+\Delta x_n) - f_n(x_1, x_2, ..., x_n)}{\Delta x_n} \end{bmatrix} \quad (7)$$

O custo computacional para aproximar uma $J(X^{(k)})$ é de $(n^2+n)$ cálculos de valores das funções $f_i$ (são $n^2$ funções nos pontos incrementados e $n$ no ponto inicial). Precisamos estabelecer, ainda, um $\Delta x_j$ inicial para cada incógnita $x_j$. Neste livro, optamos por usar na matriz jacobiana o mesmo $\Delta x_j$ calculado no incremento do método de Newton. No caso de ocorrer convergência, tanto a matriz jacobiana numérica quanto a solução convergem simultaneamente com a mesma precisão.

Essa alternativa para o método de Newton, com derivadas parciais numéricas, é indicada para a resolução de sistemas em computador a fim de facilitar a entrada de dados, o que permite a generalização do seu algoritmo.

No próximo exemplo, vamos utilizar essa alternativa manualmente com objetivos unicamente didáticos.

**Exemplo 4.4**: resolva o sistema não linear $\begin{cases} x_1 * x_2 = 1 \\ x_1 - e^{x_2} = 0 \end{cases}$ pelo método de Newton utilizando derivadas parciais numéricas para aproximar a matriz jacobiana. Tome $X^{(0)} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, incremento inicial $\Delta x_j = 0.1$, para $j = 1,2$ e critério de parada $\max \left| \Delta x_j \right| < 0.00001$.

**Solução**:

Do sistema dado, temos que:

$$\begin{cases} f_1(x_1, x_2) = x_1 x_2 - 1 = 0 \\ f_2(x_1, x_2) = x_1 - e^{x_2} = 0 \end{cases}$$

**Primeira iteração**:

$X^{(0)} = \begin{bmatrix} x_1^{(0)} \\ x_2^{(0)} \end{bmatrix} = \begin{bmatrix} 1.0 \\ 1.0 \end{bmatrix}$ e incremento inicial $\Delta X = \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} = \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix}$

Determinando numericamente as derivadas da jacobiana, conforme eq. (7), resulta:

$$J(X^{(0)}) \cong \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

$$a_{11} = \frac{\partial f_1}{\partial x_1} \cong \frac{f_1(x_1^{(0)} + \Delta x_1, x_2^{(0)}) - f_1(x_1^{(0)}, x_2^{(0)})}{\Delta x_1} \cong \frac{(1.1*1-1)-(1.0*1-1)}{0.1} = 1$$

$$a_{12} = \frac{\partial f_1}{\partial x_2} \cong \frac{f_1(x_1^{(0)}, x_2^{(0)} + \Delta x_2) - f_1(x_1^{(0)}, x_2^{(0)})}{\Delta x_2} \cong \frac{(1*1.1-1)-(1*1.0-1)}{0.1} = 1$$

$$a_{21} = \frac{\partial f_2}{\partial x_1} \cong \frac{f_2(x_1^{(0)} + \Delta x_1, x_2^{(0)}) - f_2(x_1^{(0)}, x_2^{(0)})}{\Delta x_1} \cong \frac{(1.1-e^1)-(1.0-e^1)}{0.1} = 1$$

$$a_{22} = \frac{\partial f_2}{\partial x_2} \cong \frac{f_2(x_1^{(0)}, x_2^{(0)} + \Delta x_2) - f_2(x_1^{(0)}, x_2^{(0)})}{\Delta x_2} \cong \frac{(1-e^{1.1})-(1-e^{1.0})}{0.1}$$

$$a_{22} \cong -2.85884195487388$$

Aplicando iterativamente as eqs. (4) e (5), temos:

$$\begin{bmatrix} 1 & 1 \\ 1 & -2.85884195487388 \end{bmatrix} * \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} = - \begin{bmatrix} 0 \\ -1.71828182845905 \end{bmatrix}$$

$$\begin{cases} \Delta x_1 = +0.445284323264077 \\ \Delta x_2 = -0.445284323264077 \end{cases} \Rightarrow \begin{cases} x_1^{(1)} = 1.445284323264077 \\ x_2^{(1)} = 0.554715676735923 \end{cases}$$

Critério de parada: $\max \left| \Delta x_j \right| = 0.445284323264077$ .

**Segunda iteração**:

$$X^{(1)} = \begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \end{bmatrix} = \begin{bmatrix} 1.445284323264077 \\ 0.554715676735923 \end{bmatrix} \text{ e } \Delta X = \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} = \begin{bmatrix} +0.445284323264077 \\ -0.445284323264077 \end{bmatrix}$$

$$\begin{cases} x_1^{(2)} = 1.762918095884706 \\ x_2^{(2)} = 0.569994120607396 \end{cases}$$

Critério de parada: $\max \left| \Delta x_j \right| = 0.3176337726206295$ .

Repetindo esse processo iterativo até a 5ª iteração, temos:

$$\begin{cases} x_1^{(5)} = 1.763222834351872 \\ x_2^{(5)} = 0.567143290409792 \end{cases}$$

Critério de parada atingido: $\max \left| \Delta x_j \right| = 1.43228360503785e-08$ .

Você já fez o *download* do **Caderno de Algoritmos** apresentado no Capítulo 2? Caso ainda não o tenha feito, acesse o *link* < http://sergiopeters.prof.ufsc.br/livro-calculo-numerico-computacional/> para conferir o algoritmo do método de Newton no arquivo **Cap4SistemasNaolineares.m**.

Considerações importantes sobre o método de Newton com derivadas exatas e numéricas:

a) Quando resolvemos o **Exemplo 4.2** pelo método de Newton com derivadas exatas, atingimos a solução exata em 6 iterações, e se usarmos derivadas numéricas, como no **Exemplo 4.4**, necessitaremos de 7 iterações.

b) No método de Newton com derivadas numéricas não podemos convergir os incrementos $\Delta x$ até valores muito pequenos (abaixo do limite de precisão das variáveis utilizadas), por exemplo, para um valor de $x_1^{(0)} = 1$; não podemos atingir valores de $\Delta x_1$ menores do que $O(10^{-16})$, pois esse $\Delta x_1$ perde totalmente sua significação em $x_1 = x_1^{(0)} + \Delta x_1$, e o valor incrementado fica igual ao valor inicial, $x_1 = x_1^{(0)} + \Delta x_1 = x_1^{(0)}$, gerando derivada nula. Assim, usamos como critério de parada um valor mínimo para $\Delta x_j$.

c) O método de Newton com derivadas numéricas é genérico e mais fácil de ser implementado, bastando fornecer um vetor de funções, entretanto exige maior esforço computacional. Então, sempre que possível, é mais rápido usar o método de Newton fornecendo também as funções das derivadas parciais exatas. O problema desse método é o custo da quantidade de operações aritméticas a serem executadas em cada iteração, pois, para gerar a jacobiana, precisamos obter $n^2$ valores funcionais, ou $n^2 + n$, no caso da derivação numérica, e $O(2n^3/3)$ operações para solver cada sistema linear por Gauss.

A seguir, vamos apresentar um método que reduz o custo do processamento de cada iteração em relação ao método de Newton.

### 4.3 Método de Broyden

Para tentar solver um sistema não linear $F(X) = 0$, procedemos desta forma:

a) tomamos uma solução inicial $X^{(0)}$;

b) obtemos, usando a derivação numérica (ou a analítica, se possível), a matriz jacobiana $J(X^{(0)})$, a sua inversa $J^{-1}(X^{(0)})$ (conforme vimos no

Capítulo 2) e efetuamos a primeira iteração como no método de Newton via:

$$\Delta X = -J^{-1}(X^{(0)}) * F(X^{(0)}) \quad (8a)$$

$$X^{(1)} = X^{(0)} + \Delta X \quad (8b)$$

ou

$$X^{(1)} = X^{(0)} - J^{-1}(X^{(0)}) * F(X^{(0)}) \quad (8c)$$

c) para cada uma das demais iterações $k$, determinamos $X^{(k+1)}$ sem calcular a jacobiana novamente, obtendo diretamente a sua inversa $J^{-1}(X^{(k)})$, por meio da fórmula de Sherman-Morrison (BURDEN; FAIRES, 2011) LINK Este é o diferencial do método de Broyden. FIMLINK, considerando os valores iniciais obtidos no item (b) e calculando $\Delta F = F(X^{(k)}) - F(X^{(k-1)})$:

$$J^{-1}\left(X^{(k)}\right) = J^{-1}\left(X^{(k-1)}\right) + \frac{\left[\left(\Delta X - \left(J^{-1}\left(X^{(k-1)}\right)\right) * \Delta F\right) * \left(\Delta X^{T}\right)\right] * J^{-1}\left(X^{(k-1)}\right)}{\Delta X^{T} * \left[J^{-1}\left(X^{(k-1)}\right) * \Delta F\right]} \quad (9)$$

d) substituindo a eq. (9) na eq. (8c), temos

$$X^{(k+1)} = X^{(k)} - J^{-1}\left(X^{(k)}\right) * F\left(X^{(k)}\right) \quad (10)$$

e) repetimos os passos (c) e (d) até que algum critério de parada seja satisfeito. Note que, nesse método, a partir da segunda iteração, executamos $O(n^2)$ operações em cada iteração, pois efetuamos apenas adições e multiplicações de matrizes.

**Exemplo 4.5**: resolva $\begin{cases} e^{x_1} + x_2 = 1 \\ x_1^2 + x_2^2 = 4 \end{cases}$ por Broyden, com 3 iterações a partir de $X^{(0)} = \begin{bmatrix} +1 \\ -1 \end{bmatrix}$, e calcule a solução com precisão máxima.

**Solução**:

Temos $F(X) = \begin{cases} f_1(x_1, x_2) = e^{x_1} + x_2 - 1 = 0 \\ f_2(x_1, x_2) = x_1^2 + x_2^2 - 4 = 0 \end{cases}$

Aplicando a 1ª iteração do método de Newton, obtendo a jacobiana inicial e sua inversa, aplicando as eqs. (8a), (8b) e (8c) e calculando $\Delta F = F(X^{(k)}) - F(X^{(k-1)})$, temos:

$$J(X)=\begin{bmatrix}\frac{\partial f_1(X)}{\partial x_1} & \frac{\partial f_1(X)}{\partial x_2}\\ \frac{\partial f_2(X)}{\partial x_1} & \frac{\partial f_2(X)}{\partial x_2}\end{bmatrix}=\begin{bmatrix}e^{x_1} & 1\\ 2x_1 & 2x_2\end{bmatrix}$$

$$J(X^{(0)})=\begin{bmatrix}e^{1} & 1\\ 2*1 & 2(-1)\end{bmatrix}=\begin{bmatrix}e & 1\\ 2 & -2\end{bmatrix}$$

$$J^{-1}(X^{(0)})=\begin{bmatrix}0.268941421369995 & 0.13447071068499\\ 0.268941421369995 & -0.365529289315002\end{bmatrix}$$

$$F(X^{(0)})=\begin{bmatrix}0.718281828459045\\ -2.000000000000000\end{bmatrix}$$

$$\Delta X=\begin{bmatrix}-0.9242343145200196\\ 0.0757656854799806\end{bmatrix}$$

$$X^{(1)}=\begin{bmatrix}1.07576568547998\\ -1.92423431452002\end{bmatrix}$$

$$F(X^{(1)})=\begin{bmatrix}0.00800289814302335\\ 0.85994950723254249\end{bmatrix}$$

Calculamos $\Delta F=F(X^{(k)})-F(X^{(k-1)})$ para iniciar o método de Broyden:

$$\Delta F=F(X^{(1)})-F(X^{(0)})=\begin{bmatrix}-0.710278930316022\\ 2.859949507232542\end{bmatrix}$$

E, aplicando iterativamente as eqs. (9) e (10), temos:

**Primeira iteração**: $k=1$

$$J^{-1}(X^{(1)})=\begin{bmatrix}0.2921643816775817 & 0.0990520599255256\\ 0.2073926822465813 & -0.2716578247461259\end{bmatrix}$$ via eq. (9)

$$X^{(2)}=\begin{bmatrix}0.988247753569072\\ -1.692282044505352\end{bmatrix}$$ via eq. (10)

$$\sum_{j=1}^{n}\left|\Delta x_j\right|=0.319470201925577$$

**Segunda iteração**: $k=2$

$$X^{(2)}=\begin{bmatrix}1.003016915718886\\ -1.727881103667840\end{bmatrix}$$

$$\sum_{j=1}^{n}\left|\Delta x_j\right|=0.0503682213122766$$

**Terceira iteração**: $k=3$

$$X^{(3)}=\begin{bmatrix}1.004174094411670\\ -1.729629894768390\end{bmatrix}$$

$$\sum_{j=1}^{n}\left|\Delta x_j\right| = 0.00290596979335936$$

Seguindo os passos iterativos, chegamos à solução:

$$X^{(8)} = \begin{bmatrix} 1.00416873847466 \\ -1.72963728702587 \end{bmatrix} \text{ em } k = 8 \text{ iterações}$$

$$\sum_{j=1}^{n}\left|\Delta x_j\right| = 2.62331707805421e-16$$

**Exemplo 4.6**: elabore um algoritmo que resolva o sistema não linear

$$\begin{cases} x_1^{\,2} + x_2^2 + x_3^2 - 1 = 0 \\ 2x_1^{\,2} + x_2^2 - 4x_3 = 0 \\ 3x_1^{\,2} - 4x_2 + x_3^2 = 0 \end{cases}$$

utilizando os métodos de Newton e de Broyden, LINK Para saber mais sobre o método de Broyden, consulte Burden e Faires (2011) e o artigo disponível em: <https://es.wikipedia.org/wiki/M%C3%A9todo_de_Broyden>. Acesso em: 3 nov. 2016. FIM DO LINK com critério de parada $\sum_{j=1}^{n}\left|\Delta x_j\right| < 10^{-14}$, e Newton e Broyden com derivadas parciais numéricas e critério de parada $\min\left|\Delta x_i\right| < 10^{-14}$ LINK Esse critério de parada com mínimo desvio é necessário para que, na simulação das derivadas, não sejam geradas derivadas nulas, por perda de significação FIM DO LINK, partindo de um vetor inicial $X^{(0)} = \begin{bmatrix} 0.5 & 0.5 & 0.5 \end{bmatrix}^T$. Calcule o resíduo máximo das equações e compare os resultados entre os métodos aplicados.

**Solução**:

Comparação entre os métodos:

a) Método de Newton com derivadas parciais exatas:

Solução $X = [0.785196933062355,\ 0.496611392944656,\ 0.369922830745872]$

Resíduo máximo $F = 2.22044604925031e-16$

Número de iterações $k = 6$

Critério de parada atingido: $\sum_{j=1}^{n}|\Delta x_j| = 7.14758491003723e-17$

b) Método de Newton com derivadas parciais numéricas:

Solução $X = [0.785196933062355,\ 0.496611392944656,\ 0.369922830745872]$

Resíduo máximo $F = 3.13082892944294e-14$

Critério de parada atingido: $\min |\Delta x_j| = 2.40985795343054\text{e}-17$

Número de iterações $k = 6$

c) Método de Broyden com derivadas parciais exatas:

Solução $X = [0.785196933062355,\ 0.496611392944656,\ 0.369922830745872]$

Resíduo máximo $F = 2.77555756156289e-16$

Critério de parada atingido: $\sum_{j=1}^{n} |\Delta x_j| = 2.67455660204432e-15$

Número de iterações $k = 8$

d) Método de Broyden com derivadas parciais numéricas:

Solução $X = [0.785196933062355,\ 0.496611392944656,\ 0.369922830745872]$

Resíduo máximo $F = 2.77555756156289e-16$

Critério de parada atingido: $\min |\Delta x_j| = 4.03368022884459e-16$

Número de iterações $k = 8$

Os algoritmos de Newton e Broyden aplicados ao **Exemplo 4.6** estão no **Caderno de Algoritmos** no arquivo **Cap4SistemasNaolineares.m**.

## 4.3 Conclusões

Tomando como referência apenas os resultados típicos colhidos no **Exemplo 4.6**, com a aplicação dos quatro métodos abordados neste Capítulo, podemos concluir que:

a) Os métodos de Newton são os mais recomendados para sistemas não lineares de ordem baixa ou moderada. O Newton geral, quando for possível a determinação direta da jacobiana; o Newton com derivadas numéricas, quando não. Já para sistemas de ordem mais elevada, sugerimos o método de Broyden, o qual normalmente necessita de mais

iterações para a mesma precisão, porém o custo de cada iteração é compensadoramente menor.

**b)** A recomendação anterior é válida para os casos em que ocorre a convergência, pois, se os métodos em questão continuarem divergindo depois de várias tentativas de tomada de novas soluções iniciais $X^0$, indicamos a utilização de outros métodos matematicamente mais robustos e que façam uso das derivadas direcionais, como o método das **estimativas descendentes**, que encontram um **mínimo local** da função somatório dos quadrados das funções não lineares de modo que esse mínimo local corresponde à mesma solução do sistema não linear, independentemente do valor inicial adotado (BURDEN; FAIRES, 2011).

Antes de iniciar o estudo do próximo capítulo, aplique o seu conhecimento respondendo aos exercícios do **Caderno de Exercícios e Respostas** disponível para *download* no *link* < http://sergiopeters.prof.ufsc.br/exercicios-e-respostas/ >.