



UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO - UFRPE
UNIDADE ACADÊMICA DE BELO JARDIM – UABJ

WANDSON EMANUEL DOS SANTOS SILVA
ENGENHARIA DA COMPUTAÇÃO
COMPONENTE CURRICULAR: CÁLCULO NUMÉRICO

ZEROS DE FUNÇÕES REAIS - MÉTODOS NUMÉRICOS

SUMÁRIO

1. INTRODUÇÃO.....	3
2. MÉTODOS E PROCEDIMENTOS.....	4
2.1 FUNÇÃO.....	4
2.2 MÉTODO DA BISSECÇÃO.....	5
2.3 MÉTODO DA POSIÇÃO FALSA.....	6
2.4 MÉTODO DO PONTO FIXO.....	7
2.5 MÉTODO DE NEWTON.....	9
2.6 MÉTODO DA SECANTE.....	10
3. RESULTADOS E DISCUSSÕES.....	11
4. CONCLUSÃO.....	11



1. INTRODUÇÃO

Este relatório descreve cinco métodos para encontrar as raízes de funções matemáticas. Uma raiz é um valor de "X" que torna a função igual a zero. Encontrar raízes pode ser uma tarefa simples para algumas equações, mas pode ser difícil ou impossível de fazer à mão para outras equações. Nestes casos, é útil utilizar métodos algébricos ou computacionais.

O primeiro passo para encontrar raízes é isolá-las. Isto significa encontrar intervalos que contenham as raízes. Uma vez encontrados os intervalos, estes podem ser refinados para obter uma estimativa mais precisa das raízes.

Os cinco métodos descritos neste relatório são:

- O método da bissecção
- O método da posição falsa
- O método do ponto fixo
- O método de Newton-Raphson
- O método da secante

Cada um destes métodos tem as suas próprias vantagens e desvantagens. O método da bissecção é o mais simples, mas é também o mais lento. O método de Newton-Raphson é mais rápido, mas pode ser menos exato para certas funções. O método da secante é um bom compromisso entre velocidade e exatidão. A escolha do método a utilizar depende da equação específica que está a ser resolvida. Para equações simples, o método da bissecção pode ser suficiente. Para equações mais complexas, pode ser necessário um método mais rápido, como o método de Newton-Raphson.



2. MÉTODOS E PROCEDIMENTOS

2.1 FUNÇÃO

A função a ser trabalhada utilizando os métodos numéricos apresentados neste relatório é

$$f(x) = \sqrt{9 - x^2} - \frac{e^x}{8}$$

(Figura 1 - Função)

Para determinar as raízes desta função, foram desenvolvidos diferentes métodos utilizando a linguagem de programação Python, os quais serão comparados entre si para identificar o mais eficiente computacionalmente. O primeiro passo foi determinar os intervalos nos quais a função possui raízes reais, seguindo os procedimentos descritos no teorema mencionado no início deste relatório. Ao analisar visualmente a natureza da função, foi possível estabelecer o intervalo de valores válidos para "x", na qual a função possui seu domínio: $\text{Dom } f(x) = [-3, 3]$. Com o domínio da função determinado, foi determinado também o intervalo $[0, 3]$ com o objetivo de encontrar aquele que contém raízes reais. A tabela abaixo apresenta os resultados obtidos a partir dessas análises.

VALORES PARA F(X)	0	1	2	3
SINAL DE F(X)	+	+	+	-

Observando os valores da tabela, percebemos que a função muda de sinal no intervalo: I (2,3). Isso significa que dentro desse intervalo há um valor que zere a função, ou seja, uma raiz. Entretanto, temos que verificar se a função não alterna de sinal constantemente, assim como as funções trigonométricas. Para fazer essa verificação podemos derivar a função uma vez e verificar algumas informações...

Derivando a função:

Passos da solução

$$\frac{d}{dx} \left(\sqrt{9 - x^2} - \frac{e^x}{8} \right)$$

Aplicar a regra da soma/diferença: $(f \pm g)' = f' \pm g'$

$$= \frac{d}{dx} \left(\sqrt{9 - x^2} \right) - \frac{d}{dx} \left(\frac{e^x}{8} \right)$$

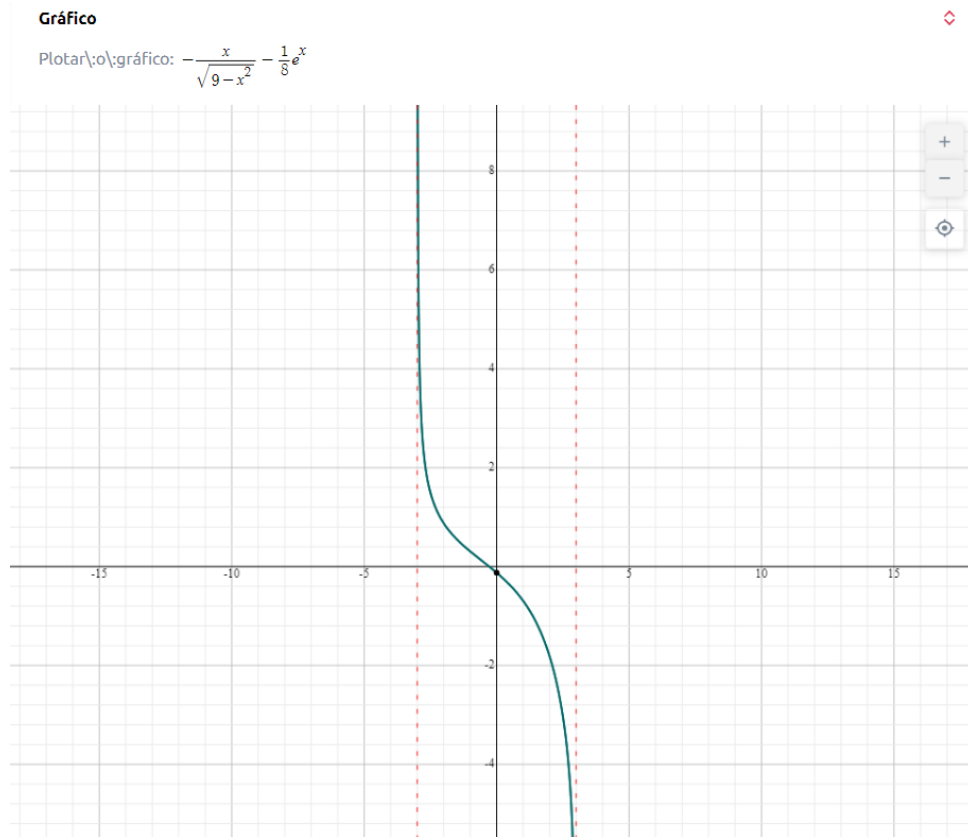
(Figura 2 - Passo 1 da Derivação)

$$\frac{d}{dx} \left(\sqrt{9 - x^2} \right) = -\frac{x}{\sqrt{9 - x^2}}$$

$$\frac{d}{dx} \left(\frac{e^x}{8} \right) = \frac{1}{8} e^x$$

$$= -\frac{x}{\sqrt{9 - x^2}} - \frac{1}{8} e^x$$

(Figura 3 - Passo 2 da Derivação)



(Figura 4 - Gráfico da função derivada)

A partir do gráfico da derivada, é possível identificar que ela é decrescente e possui apenas uma raiz no intervalo $(-3,3)$. Sendo assim, podemos prosseguir para os métodos que irão encontrar um valor aproximado para essa raiz.

2.2 MÉTODO DA BISSECÇÃO

O método da bissecção é um método de busca de raízes que consiste em dividir repetidamente um intervalo ao meio e selecionar um subintervalo contendo a raiz para processamento adicional. O método garante a existência de uma solução para $f(x) = 0$ no intervalo (a, b) desde que $f : [a, b] \rightarrow \mathbb{R}$ seja contínua e satisfaz $f(a).f(b) < 0$. Em seguida, o procedimento é repetido com o subintervalo de interesse. Após um número finito de subdivisões, o método se aproximará para a solução seguindo aproximação de uma precisão pré-estabelecida (ϵ).



2.2.1 Código em Python para cálculo da solução utilizando o método da bissecção.

```
import math

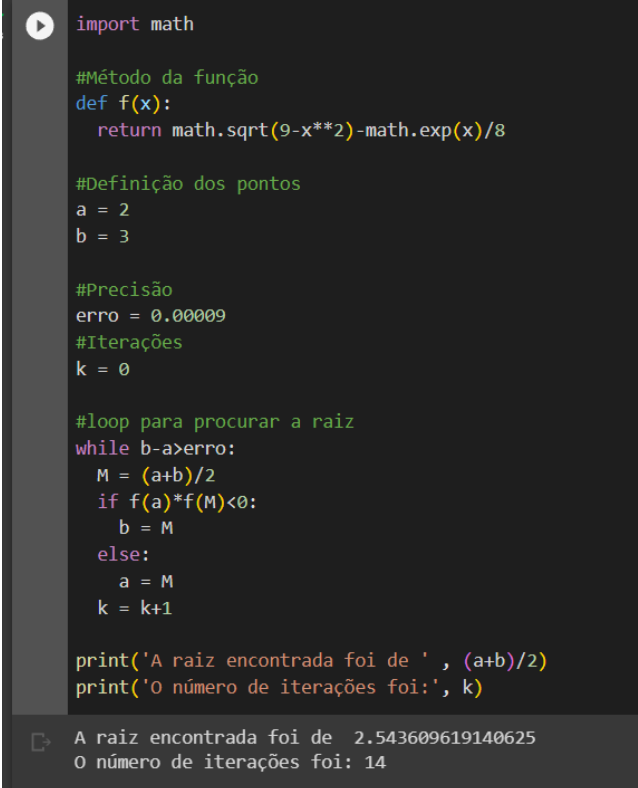
#Método da função
def f(x):
    return math.sqrt(9-x**2)-math.exp(x)/8

#Definição dos pontos
a = 2
b = 3

#Precisão
erro = 0.00009
#Iterações
k = 0

#loop para procurar a raiz
while b-a>erro:
    M = (a+b)/2
    if f(a)*f(M)<0:
        b = M
    else:
        a = M
    k = k+1

print('A raiz encontrada foi de ' , (a+b)/2)
print('O número de iterações foi:', k)
```



```
import math

#Método da função
def f(x):
    return math.sqrt(9-x**2)-math.exp(x)/8

#Definição dos pontos
a = 2
b = 3

#Precisão
erro = 0.00009
#Iterações
k = 0

#loop para procurar a raiz
while b-a>erro:
    M = (a+b)/2
    if f(a)*f(M)<0:
        b = M
    else:
        a = M
    k = k+1

print('A raiz encontrada foi de ' , (a+b)/2)
print('O número de iterações foi:', k)
```

A raiz encontrada foi de 2.543609619140625
O número de iterações foi: 14

(Figura 5 - Execução do Método da Bissecção)

A função $f(x)$ é definida na linha 4 e é utilizada para calcular o valor da função em um determinado ponto x . Os pontos a e b são definidos nas linhas 7 e 8, respectivamente, e representam o intervalo no qual a raiz da função será procurada. O erro máximo permitido é definido na linha 11.

O loop `while` nas linhas 14-21 é utilizado para procurar a raiz da função. A cada iteração do loop, o ponto médio M do intervalo $[a,b]$ é calculado na linha 15. Se o produto $f(a)*f(M)$ for negativo, isso significa que a raiz da função está no intervalo $[a,M]$, então o valor de b é atualizado para M na linha 17. Caso contrário, a raiz está no intervalo $[M,b]$, então o valor de a é atualizado para M na linha 19. O número de iterações necessárias para encontrar a raiz é armazenado na variável k . Finalmente, as linhas 23 e 24 imprimem o valor da raiz encontrada e o número de iterações necessárias para achar a raiz.



2.3 MÉTODO DA POSIÇÃO FALSA

O método da posição falsa é um método numérico utilizado para encontrar a raiz de uma função contínua no intervalo $[a,b]$. Esse método é semelhante ao método da bissecção, mas em vez de dividir o intervalo ao meio a cada iteração, ele utiliza uma interpolação linear para aproximar a raiz.

O método começa com dois pontos iniciais $(a, f(a))$ e $(b, f(b))$, onde $f(a)$ e $f(b)$ têm sinais opostos. Uma reta é traçada entre esses dois pontos e o ponto de intersecção dessa reta com o eixo x é calculado. Esse ponto de intersecção é então usado como um novo ponto para gerar uma nova reta que intersecta o eixo x . O processo é repetido até que a raiz seja encontrada com a precisão desejada.

A cada iteração, o valor de M é determinado pela média ponderada entre “ a ” e “ b ”, com pesos “ $f(a)$ ” e “ $f(b)$ ”. Isso garante que os pontos $(a, f(a))$, $(b, f(b))$ e $(M, f(M))$ sejam colineares e respeitem o critério do método.

Em resumo, o método da posição falsa é um método iterativo que utiliza interpolação linear para aproximar a raiz de uma função contínua no intervalo $[a,b]$. Ele é semelhante ao método da bissecção, mas pode convergir mais rapidamente em algumas situações.

O valor de M será determinado por:

$$\begin{vmatrix} a & f(a) \\ b & f(b) \\ M & f(M) \\ a & f(a) \end{vmatrix} = 0$$

(Figura 6 - Determinante da Matriz)

$$M = \frac{a \cdot f(b) - b \cdot f(a)}{f(b) - f(a)}$$

(Figura 7 - Fórmula para o valor da Média)

2.3.1 Código em Python para cálculo da solução utilizando o método da posição falsa.

```
import math

#método da função
def f(x):
    return math.sqrt(9-x**2)-math.exp(x)/8

#definição dos pontos
a = 2
b = 3

#precisão
erro = 0.00009
#iterações
k = 0

M = (a*f(b)-b*f(a))/(f(b)-f(a))

#loop para procurar a raiz
while abs(f(M))>erro:
    M = (a*f(b)-b*f(a))/(f(b)-f(a))
    if f(a)*f(M)<0:
        b = M
    else:
        a = M
    k = k+1

print('A raiz encontrada foi de', M)
print('O número de iterações foi:', k)
```

```
import math

#método da função
def f(x):
    return math.sqrt(9-x**2)-math.exp(x)/8

#definição dos pontos
a = 2
b = 3

#precisão
erro = 0.00009
#iterações
k = 0

M = (a*f(b)-b*f(a))/(f(b)-f(a))

#loop para procurar a raiz
while abs(f(M))>erro:
    M = (a*f(b)-b*f(a))/(f(b)-f(a))
    if f(a)*f(M)<0:
        b = M
    else:
        a = M
    k = k+1

print('A raiz encontrada foi de', M)
print('O número de iterações foi:', k)
```

A raiz encontrada foi de 2.5435726745481926
O número de iterações foi: 12

(Figura 8 - Execução do Método da Posição Falsa)



2.4 MÉTODO DO PONTO FIXO

O método do ponto fixo é um método iterativo que é utilizado para encontrar raízes de equações não lineares. Ele consiste em transformar a equação em uma equação equivalente $x = \Phi(x)$ e a partir de uma aproximação inicial gerar uma sequência de aproximações para a raiz R pela relação $x_{k+1} = \Phi(x_k)$. A função $\Phi(x)$ é tal que $f(x^*) = 0$ se e somente se $\Phi(x^*) = R$. Assim, transforma-se o problema de encontrar um zero de $f(x)$ no problema de encontrar um ponto fixo de $\Phi(x)$.

$$x_{k+1} = \varphi(x_k)$$

(Figura 9 - Relação X_k)

$$x = \sqrt{-\frac{e^{2x}}{64} + 9}$$

(Figura 10 - X isolado)

Seja x^* uma raiz da equação $f(x) = 0$, isolada num intervalo I . Seja $\varphi(x)$ uma função de iteração para a equação $f(x) = 0$. Se:

- i) $\varphi^*(x)$ e $\varphi(x)$ são contínuas em I ,
- ii) $|\varphi'(x)| \leq M < 1$ para todo $x \in I$ e
- iii) $x_0 \in I$ então a sequência $\{x_k\}$ gerada pelo processo iterativo $x_{k+1} = \varphi(x_k)$ converge para x^*

2.4.1 Código em Python para cálculo da solução utilizando o método do ponto fixo.

```
import math
```

```
#função original
```

```
def f(x):
    return math.sqrt(9-x**2)-math.exp(x)/8
```

```
#função de iteração
```

```
def phi(x):
    #math.sqrt(-math.exp(2*x)/64 + 9)
    #math.log(8) + math.log(3-x)
    return math.sqrt(-math.exp(2*x)/64 +
9)
```

```
#aproximação inicial
```

```
x_k = 1
```

```
#erro
```

```
erro = 0.00009
```

```
#contagem das iterações
```

```
k = 0
```

```
#iterações
```

```
while abs(f(x_k))>erro:
```

```
    x_k = phi(x_k)
```

```
    k += 1
```

```
print("A solução encontrada foi:", x_k)
```

```
print("O número de iterações foi:", k)
```

```
import math

#função original
def f(x):
    return math.sqrt(9-x**2)-math.exp(x)/8

#função de iteração
def phi(x):
    #math.sqrt(-math.exp(2*x)/64 + 9)
    #math.log(8) + math.log(3-x)
    return math.sqrt(-math.exp(2*x)/64 + 9)

#aproximação inicial
x_k = 1

#erro
erro = 0.00009

#contagem das iterações
k = 0

#iterações
while abs(f(x_k))>erro:
    x_k = phi(x_k)
    k += 1

print("-----PONTO FIXO-----")
print("A solução encontrada foi:", x_k)
print("O número de iterações foi:", k)
```

```
-----PONTO FIXO-----
A solução encontrada foi: 2.543557283345583
O número de iterações foi: 1596
```

(Figura 11 - Execução do Método do Ponto Fixo)



2.5 MÉTODO DE NEWTON

O método de Newton é um método iterativo para encontrar raízes de equações não lineares. Ele é considerado um dos métodos mais eficientes para encontrar raízes de equações não lineares, inclusive sendo o mesmo método que o ponto fixo, entretanto de forma mais eficiente. O primeiro passo é escolher uma aproximação inicial. Após isso, calcula-se a equação da reta tangente (por meio da derivada) da função nesse ponto e a interseção dela com o eixo X, a fim de encontrar uma melhor aproximação para a raiz. Repetindo-se o processo, cria-se um método iterativo para encontrarmos a raiz da função:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

(Figura 12 - Fórmula de iteração)

2.5.1 Código em Python para cálculo da solução utilizando o método do Newton.

```
import math

#função original
def f(x):
    return math.sqrt(9-x**2)-math.exp(x)/8

def f_linha(x):
    return (-x/math.sqrt(9-x**2))-math.exp(x)/8

#função de iteração
def phi(x):
    return x - (f(x) / f_linha(x))

#aproximação inicial
x_k = 2

#erro
erro = 0.00009

#contagem das iterações
k = 0

#iterações
while abs(f(x_k))>=erro:
    x_k = phi(x_k)
    k += 1

print("-----NEWTON-----")
print("A solução encontrada foi:", x_k)
print("O número de iterações foi:", k)
```

```
import math

#função original
def f(x):
    return math.sqrt(9-x**2)-math.exp(x)/8

def f_linha(x):
    return (-x/math.sqrt(9-x**2))-math.exp(x)/8

#função de iteração
def phi(x):
    return x - (f(x) / f_linha(x))

#aproximação inicial
x_k = 2

#erro
erro = 0.00009

#contagem das iterações
k = 0

#iterações
while abs(f(x_k))>=erro:
    x_k = phi(x_k)
    k += 1

print("-----NEWTON-----")
print("A solução encontrada foi:", x_k)
print("O número de iterações foi:", k)
```

-----NEWTON-----
A solução encontrada foi: 2.54358546180783
O número de iterações foi: 4

(Figura 13 - Execução do Método de Newton)



2.6 MÉTODO DA SECANTE

O método da secante, assim como os métodos acima citados, é um método numérico utilizado para encontrar as raízes de uma função. Ele é uma extensão do método da falsa posição, que utiliza uma reta secante para estimar a raiz da função. O método da secante não requer a derivada da função, ao contrário do método de Newton-Raphson, o que o torna mais flexível em certas situações. No entanto, o método da secante pode ser um pouco mais lento em termos de convergência em comparação com o método de Newton-Raphson.

$$f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$$

(Figura 14 - Função para o método da secante)

$$x_{k+1} = \frac{x_{k-1} \cdot f(x_k) - x_k \cdot f(x_{k-1})}{f(x_k) - f(x_{k-1})}$$

(Figura 15 - Função de iteração para Secante)

2.6.1 Código em Python para cálculo da solução utilizando o método da secante.

```
import math

#função
def f(x):
    return math.sqrt(9-x**2)-math.exp(x)/8

#aproximações inicial
x0 = 0
x1 = 3

#erro
erro = 0.00009

#contagem das iterações
k = 0

#iterações
while abs(f(x1))>erro:
    if abs(x0-x1)<erro:
        break
    x2 = (x0*f(x1)-x1*f(x0))/(f(x1)-f(x0))
    x0 = x1
    x1 = x2
    k += 1

print("-----SECANTE-----")
print("A solução encontrada foi:", x1)
print("O número de iterações foi:", k)
```

```
import math

#função
def f(x):
    return math.sqrt(9-x**2)-math.exp(x)/8

#aproximações inicial
x0 = 0
x1 = 3

#erro
erro = 0.00009

#contagem das iterações
k = 0

#iterações
while abs(f(x1))>erro:
    if abs(x0-x1)<erro:
        break
    x2 = (x0*f(x1)-x1*f(x0))/(f(x1)-f(x0))
    x0 = x1
    x1 = x2
    k += 1

print("-----SECANTE-----")
print("A solução encontrada foi:", x1)
print("O número de iterações foi:", k)
```

-----SECANTE-----
A solução encontrada foi: 2.543585093485565
O número de iterações foi: 7

(Figura 16 - Execução do Método da Secante)



3. RESULTADOS E DISCUSSÕES

Os resultados obtidos a partir dos cálculos utilizando os diferentes métodos numéricos para a determinação de zeros de funções reais estão agrupados na tabela abaixo.

	Método da Bisseção	Método da Posição Falsa	Método do Ponto Fixo	Método de Newton	Método da Secante
Dados e aproximações iniciais	$a=2$ $b=3$	$a=2$ $b=3$	$x_0 = 1$	$x_0 = 2$	$x_0 = 0$ $x_1 = 3$
Solução Encontrada	$\approx 2.543609619140625$	$\approx 2.5435726745481926$	$\approx 2.543557283345583$	≈ 2.54358546180783	$\approx 2.543585093485565$
Quantidade de Interações	14	12	1596	4	7

De acordo com os resultados da tabela acima, é possível perceber que houve uma divergência considerável em relação à quantidade de interações que cada método obteve para determinar o valor aproximado ou exato da raiz. O valor aproximado da raiz da função trabalhada neste relatório é $X \approx 2,54$, logo, para a resolução desta função, o método de Newton foi o mais eficiente, realizando apenas 4 iterações para determinar o valor aproximado da raiz. O método menos eficiente em uma visão computacional foi o método do ponto fixo, na qual teve 1596 interações até obter um valor aproximado da raiz dentro de uma precisão pré-estabelecida que foi comum a todos os métodos. É importante destacar que esses resultados podem variar de função para função, onde o método mais eficiente neste caso, poderá ser o menos eficiente de acordo com o tipo de função a ser trabalhada. Podemos considerar então, que existe uma relatividade inerente a tais métodos à medida que ocorra variações de funções a serem trabalhadas usando estes mesmos métodos numéricos.

4. CONCLUSÃO

Os métodos numéricos para a determinação de zeros de funções reais apresentados no relatório são viáveis para tal objetivo, visto que foi possível determinar aproximações válidas para o valor real da função apresentada. O método de Newton foi o mais efetivo. Como esperado, houve significativas variações em relação à quantidade de interações realizadas por cada método até se chegar a uma aproximação válida para o valor da raiz. Este resultado se dá pelo fato de que cada método apresenta uma dinâmica diferente para a realização dos cálculos, a qual reflete diretamente no número de interações, bem como no valor aproximado da raiz. Cada método tem suas vantagens e desvantagens e pode ser mais adequado dependendo do problema em questão.