UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA DE TELEINFORMÁTICA
CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

KEVEN DA SILVA GONÇALVES

TRABALHO 1 - ZERO DE FUNÇÕES

FORTALEZA

2021

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

Solucionar uma equação do tipo $f(x) = 0$ pode ser uma tarefa fácil, quando são conhecidos métodos analíticos que levem ao valor exato de x. No entanto, existem funções, como $f(x) = x\sin x + \cos x$, por exemplo, onde não podemos encontrar x, por meio de métodos analítcos. Portanto, deve-se fazer o emprego de métodos numéricos para solução destes problemas. Os princípais métodos para encontrar raízes de funções, de uma variável, são: método da bisseção, método regula falsi, método de Newton-Raphson, método da secante e método da iteração de ponto fixo. Cada um dos métodos, listados anteriormente, possui particularidades e diferentes aplicações.

O seguinte trabalho visa a implementação, estudo e análise de cada um destes métodos. Em todos os casos, o erro mínimo (e) será igual a 0.003 ($e = 0.003$). Para cada método, serão apresentados os seus respectivos fluxogramas, pseudo-códigos, descrição do método e gráficos sobre o erro relativo real versus o número da iteração. A função escolhida para ser utilizada na aplicação dos métodos é a função $f(x) = x^2 + 1$. Para todas as aplicações dos métodos, neste trabalho, o critério de parada é definido pelo erro mínimo (escolhido anteriormente). Portanto, em todos os algoritmos, somente chegaremos ao fim dos mesmos, se o valor do erro relativo real para a solução encontrada, seja menor ou igual, ao erro mínimo estabelecido.

2 DESENVOLVIMENTO

2.1 Método da Bisseção

2.1.1 Descrição do método

Este método, assim como o método Regula Falsi, é classificado como um método de confinamento. Partimos do pressuposto que a função $f(x)$ escolhida, é continua e possui solução em um dado intervalo $[a,\ b]$. Outra condição que deve ser verificada, é se $f(a) < 0$ e $f(b) > 0$ ou $f(a) > 0$ e $f(b) < 0$, tal que $f(a)f(b) < 0$. Esta condição garante que no invertalo $[a,\ b]$ existe uma solução para $f(x) = 0$. Como primeira estimativa de solução, tomamos:

$$x_{NS} = \frac{a+b}{2}$$

Onde a e b são os extremos do intervalo. Verifica-se, se o valor x_{NS} (solução numérica) é suficientemente preciso. Se não, é necessário definir um novo intervalo, menor que o anterior, que contenha a solução. A condição para definição do novo intervalo é dada pelo sinal do produto $f(a)f(b)$. Se $f(a)f(b)$ for menor que zero, então a solução pertence ao intevalo $[a,\ x_{NS}]$. Caso contrário, a solução pertencerá ao intervalo $[x_{NS},\ b]$. O cálculo de x_{NS}, e consequentemente, a definição de um novo intervalo, é feito de maneira iterativa até que x_{NS} seja suficientemente preciso.

2.1.2 Pseudo-código

Proposta de pseudo-código para o método da Bisseção.

```
/* Função f(x) = x^2 - 1
 * Raiz x = 1.
 * Onde "e" é o valor do erro máximo escolhido.
 * ERR ou Erro Relativo Real.
 * Os reais "a" e "b" indicam o inicio e o fim do intervalo,
   respectivamente, no qual
 * a solução real para f(x) = 0 está inserida.
 * xNS solução numérica.
 */

```

```
Inicio algoritmo

    Real: e, ERR, xNS, a, b
    Inteiro: n

    // Valores iniciais
    e = 0.003 ou 0.3%
    ERR = 0
    a = 0  // [a, b] = [0, 3]
    b = 3
    n = 0

    FUNÇÃO funcao(Real: x, Inteiro: ordem)
        // Tomando como f(x) = x^2 - 1
        Real: f
        Mudar(ordem):
            Caso 0:
                f = x^2 - 1
            Caso 1:
                f = 2 * x
        RETORNE f

    FUNÇÃO calculeERR(Real: xNS)
        Real: ERR
        ERR = |1 - xNS|
        RETORNE ERR

    // Considerando que f(a)f(b) < 0

    FAÇA
        xNS = (a + b) / 2
        SE (funcao(a, 0) * funcao(xNS, 0) < 0) FAÇA
            b = xNS
        SENÃO FAÇA
            a = xNS
        ERR = calculeERR(xNS)
        n = n + 1
    // O laço continuará até que ERR seja menor ou igual que o erro "
       e" escolhido.
```

```
        ENQUANTO (ERR > e)

        Exiba(xNS) // Solução numérica para f(x) = 0

    Fim algoritmo
```

2.1.3 Fluxograma

O fluxograma a seguir foi estruturado de acordo com o pseudo-código descrito anteriormente.

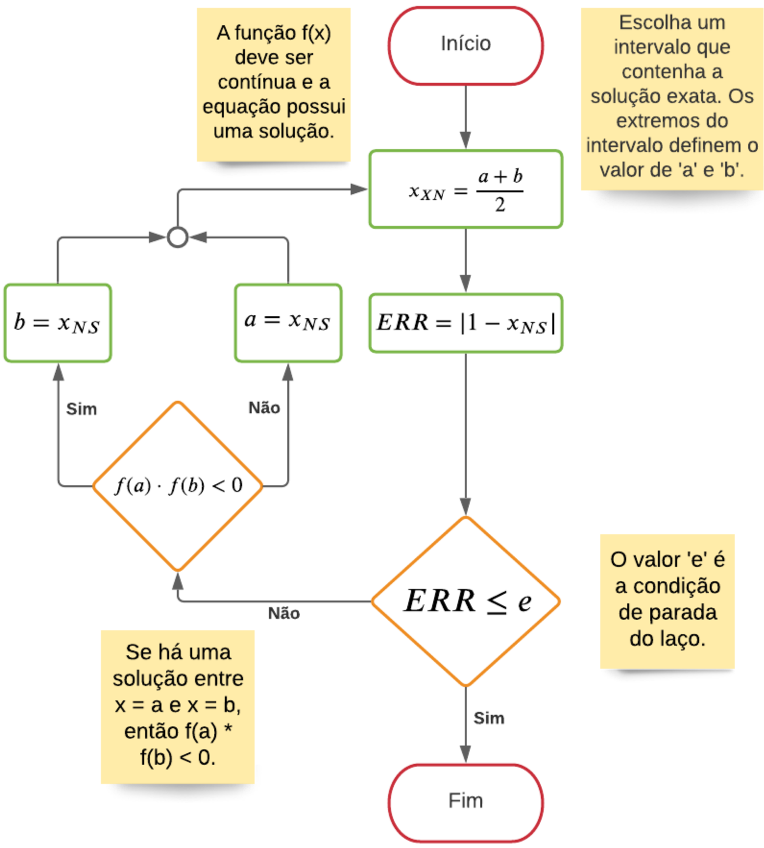

Figura 1 – Fluxograma - Método da Bisseção.

2.1.4 Erro Relativo Real versus Número da Iteração

Ao analisar o gráfico obtido, é possível verificar que na primeira iteração, o valor do ERR é muito alto. No entanto, verifica-se também uma queda brusca do erro, já nas primeiras iterações. A partir da quinta iteração o erro já começa a não variar bruscamente. É importante lembrar que as condições iniciais são fatores determinantes no resultado gerado pelo método.

(Descrição) A distribuição de pontos vermelhos no gráfico indicam os resultados obtidos na implementação do método. O número da iteração e o erro relativo real formam pares ordenados (n, ERR(n)). A curva de 'ajuste' foi utilizada para facilicitar a visualização do comportamento do erro relativo real a cada iteração.

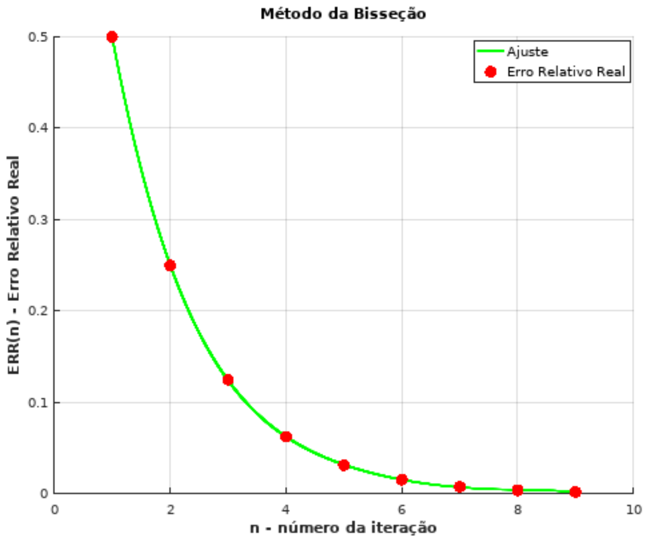

Figura 2 – Gráfico ERR vs n - Método da Bisseção.

2.2 Método Regula Falsi

2.2.1 Descrição do método

O método Regula Falsi ou método da Falsa Posição é classificado como um método de confinamento. Do mesmo modo que o método da Bisseção, a função $f(x)$ deve ser escolhida de tal forma, que seja contínua e a equação possua solução em um intervalo $[a,\ b]$. A principal diferença entre este método e o antetior, é na definição de x_{NS} (solução numérica). A solução numérica será dada por:

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

Onde os valores 'a' e 'b' são os extremos do intervalo. Esta estimativa de solução é calculada de forma iterativa até que x_{NS} seja suficientemente preciso. As condições iniciais são: $f(a)f(b) < 0$ e a solução deve pertencer ao intervalo $[a,\ b]$. A estimativa x_{NS} deve ser calculada e, caso seja insuficiente, um novo intervalo deve ser determinado. A condição para a definição do novo intervalo é dado pelo sinal do resultado de $f(a)f(x_{NS})$. Caso o resultado seja positivo, o novo intervalo será $[x_{NS},\ b]$. E, se for negativo, o novo intervalo será $[a,\ x_{NS}]$. Este processo é repetido até que a solução numérica seja suficiente.

2.2.2 Pseudo-código

Proposta de pseudo-código para o método Regula Falsi.

```
/* Função f(x) = x^2 - 1
 * Raiz x = 1.
 * Onde "e" é o valor do erro máximo escolhido.
 * ERR ou Erro Relativo Real.
 * Os reais "a" e "b" indicam o inicio e o fim do intervalo, respectivamente, no qual
 * a solução real para f(x) = 0 está inserido.
 * xNS solução numérica.
 */

Inicio algoritmo

    Real: e, ERR, xNS, a, b
    Inteiro: n

    // Valores iniciais
    e = 0.003 ou 0.3%
    ERR = 0
    a = 0  // [a, b] = [0, 3] // Mesmo intervalo do Método da Bisseção
    b = 3
    n = 0

    FUNÇÃO funcao(Real: x, Inteiro: ordem)
        // Tomando como f(x) = x^2 - 1
        Real: f
        Mudar (ordem) para:
            Caso 0:
                f = x^2 - 1
```

```
            Caso 1:
                f = 2 * x
        RETORNE f

    FUNÇÃO calculeERR(Real: xNS)
        Real: ERR
        ERR = |1 - xNS|
        RETORNE ERR

    // Considerando que f(a)f(b) < 0

    FAÇA
        xNS = (a * funcao(b, 0) - b * funcao(a, 0)) / (funcao(b, 0) - funcao(a, 0))
        SE (funcao(a, 0) * funcao(xNS, 0) < 0) FAÇA
            b = xNS
        SENÃO FAÇA
            a = xNS
        ERR = calculeERR(xNS)
        n = n + 1
    // O laço continuará até que ERR seja menor ou igual que o erro "e" escolhido.
    ENQUANTO (ERR > e)

    Exiba(xNS) // Solução numérica para f(x) = 0

Fim algoritmo
```

2.2.3 *Fluxograma*

O fluxograma a seguir foi estruturado de acordo com o pseudo-código descrito anteriormente.

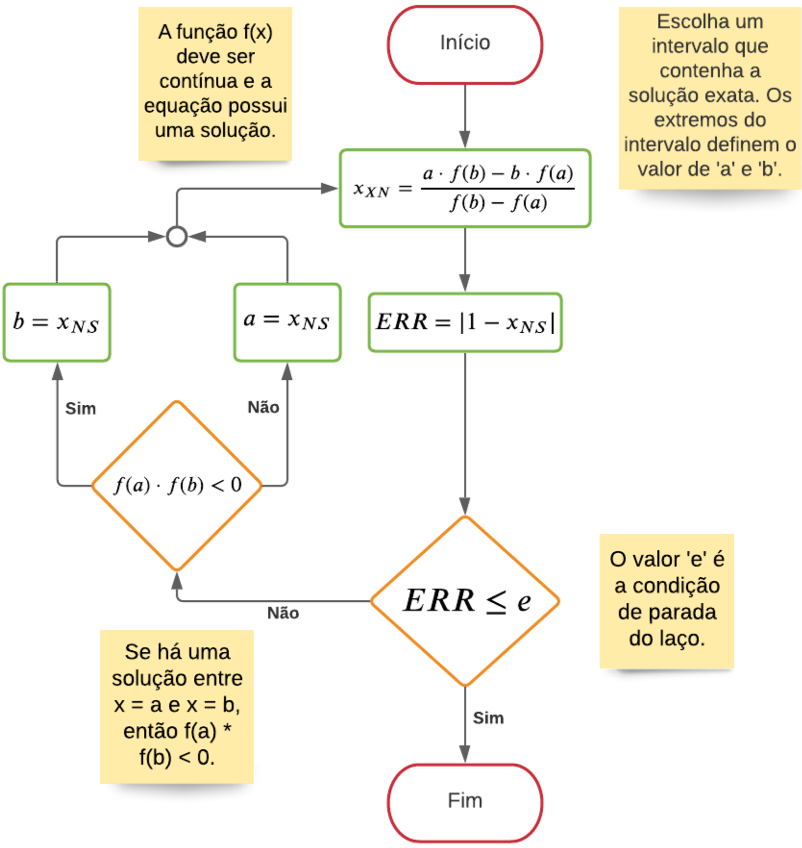

Figura 3 – Fluxograma - Método Regula Falsi.

2.2.4 *Erro Relativo Real versus Número da Iteração*

A partir do gráfico abaixo, é possível verificar que os métodos Regula Falsi e da Bisseção não só possuem procedimentos parecidos, mas também os valores do erro a cada iteração. Inicialmente, o erro apresentado é maior que o erro obtido na mesma iteração, do método da Bisseção. Apesar da divergência inicial, os gráficos que descrevem o comportamento do erro, resultante da implementação, dos dois métodos, são bem semelhantes.

(Descrição) A distribuição de pontos vermelhos no gráfico indicam os resultados obtidos na implementação do método. O número da iteração e o erro relativo real formam pares ordenados (n, ERR(n)). A curva de 'ajuste' foi utilizada para facilicitar a visualização do comportamento do erro relativo real a cada iteração.

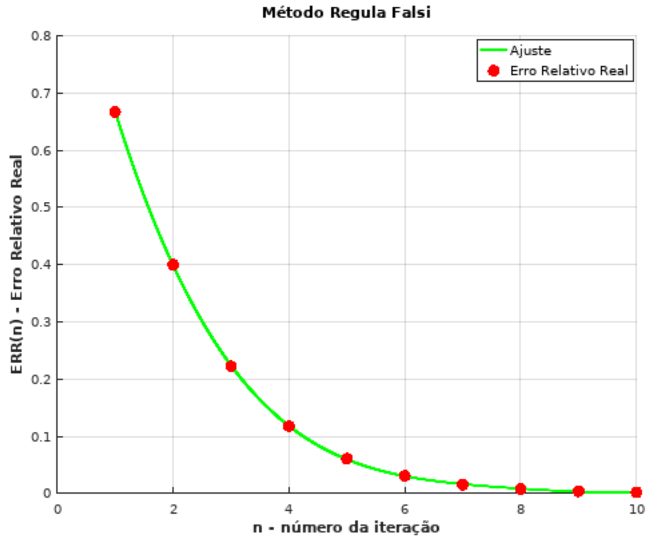

Figura 4 – Gráfico ERR vs n - Método Regula Falsi.

2.3 Método de Newton-Raphson

2.3.1 Descrição do método

Este método é classificado como aberto. Diferentemente dos métodos anteriores, este método não necessita de um intervalo, e sim de um ponto, como estimativa inicial. A função $f(x)$ escolhida deve ser contínua e diferenciável, além de possuir uma solução para $f(x) = 0$. A partir da primeira estimativa, será calculado o valor de x_{NS} (solução numérica), que é dado por:

$$x_{NS} = x_1 - \frac{f(x_1)}{f'(x_1)}$$

Onde x_1 é a estimativa inicial. O cálculo de x_{NS} é feito, até que esta solução seja suficientemente precisa.

2.3.2 Pseudo-código

Proposta de pseudo-código para o método de Newton-Raphson.

```
/* Função f(x) = x^2 - 1
 * Raiz x = 1.
 * Onde "e" é o valor do erro máximo escolhido.
```

```
 * ERR ou Erro Relativo Real.
 * O número real "a" indicam a primeira tentativa de solução.
 * xNS solução numérica.
 */

Inicio algoritmo

    Real: e, ERR, xNS, a
    Inteiro: n

    // Valores iniciais
    e = 0.003 ou 0.3%
    ERR = 0
    a = 3   // x1 = 3
    n = 0

    FUNÇÃO funcao(Real: x, Inteiro: ordem)
        // Tomando como f(x) = x^2 - 1
        Real: f
        Mudar (ordem) para:
            Caso 0:
                f = x^2 - 1
            Caso 1:
                f = 2 * x
        RETORNE f

    FUNÇÃO calculeERR(Real: xNS)
        Real: ERR
        ERR = |1 - xNS|
        RETORNE ERR

    FAÇA
        xNS = a - (funcao(a, 0) / funcao(a, 1))
        ERR = calculeERR(xNS)
        a = xNS
        n = n + 1
    // O laço continuará até que ERR seja menor ou igual que o erro "
       e" escolhido.
    ENQUANTO (ERR > e)
```

```

        Exiba(xNS) // Solução numérica para f(x) = 0

    Fim algoritmo
```

2.3.3 *Fluxograma*

O fluxograma a seguir foi estruturado de acordo com o pseudo-código descrito anteriormente.

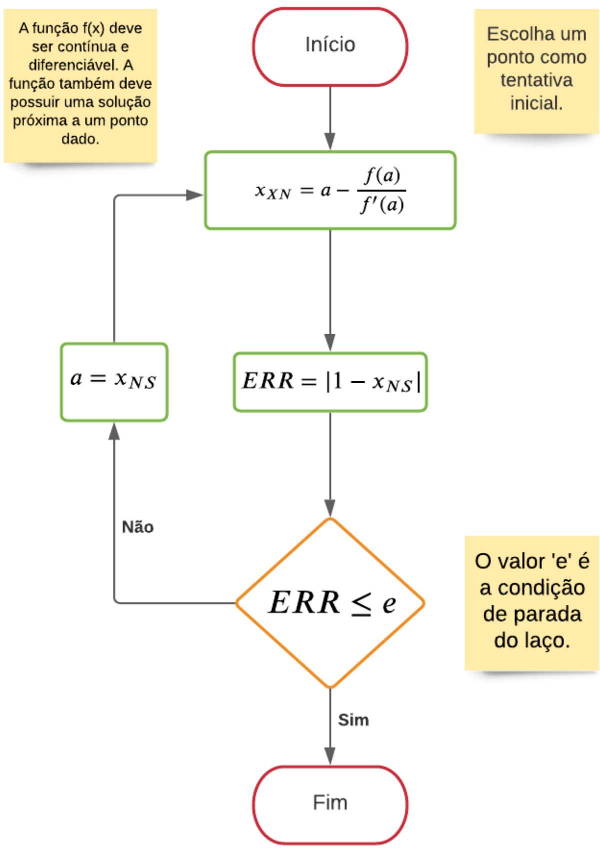

Figura 5 – Fluxograma - Método de Newton-Raphson

2.3.4 *Erro Relativo Real versus Número da Iteração*

A comparação entre os métodos anteriores com o método atual, chega até ser injusta. É claro que cada método possui uma determinada aplicação para dadas condições inicias. Mas é perceptível que este método precisou de um número muito menor de iterações para convergir.

(Descrição) A distribuição de pontos vermelhos no gráfico indicam os resultados obtidos na implementação do método. O número da iteração e o erro relativo real formam pares ordenados (n, ERR(n)). A curva de 'ajuste' foi utilizada para facilicitar a visualização do comportamento do erro relativo real a cada iteração.

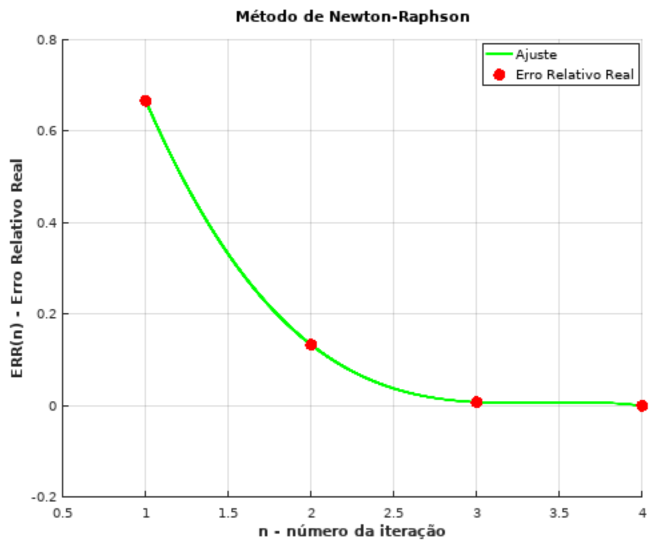

Figura 6 – Gráfico ERR vs n - Método de Newton-Raphson.

2.4 Método do Ponto Fixo

2.4.1 Descrição do método

Como o método anterior, este também é classificado como aberto. No entanto, este método é o que mais difere em relação ao procedimento para determinação de uma solução. O método é implementado, reescrevendo a equação ($f(x) = 0$) como $x = g(x)$. Onde a função $g(x)$ é definida como:

$$f(x) + x = x = g(x)$$

A função $g(x)$ ou função iterativa é utilizada para determinar a solução numérica x_{NS}. O principal desafio deste método é a determinação da função iterativa. Esta função deve ser diferenciável na vizinhança da solução. Para que a solução convirja, é necessário que a derivada de $g(x)$ possua um valor absoluto menor que 1, na vizinhança da solução (condição conhecida como Lipschitz contínua).

2.4.2 *Adendo*

A função escolhida para este trabalho foi $f(x) = x^2 + 1$. Onde teremos:

$$f(x) = 0 = x^2 + 1$$

e

$$g(x) = x = x^2 + x + 1$$

Podemos verificar que, a simples manipulação descrita anteriormente ($f(x) + x = x = g(x)$, para $f(x) = 0$), não satisfaz a condição de Lipschitz contínua. No entanto, se multiplicarmos $f(x) = 0 = x^2 + 1$ por e^x, teremos:

$$e^x \cdot (x^2 + 1) = 0$$

Verifica-se que, a equação acima, possui as seguintes soluções:

$$x^2 + 1 = 0$$

ou

$$e^x = 0$$

Onde e^x só irá convergir para zero, quando

$$\lim_{x \to -\infty} e^x = 0$$

Como nossa solução é finita, assumimos que e^x é sempre diferente de zero. Portanto, a multiplicação de $f(x) = 0 = x^2 + 1$ por e^x não alterará o resultado encontrado para a solução de $f(x) = 0$. Por fim, teremos que a função iterativa será igual a:

$$g(x) = \frac{x + e^x}{1 + xe^x}$$

2.4.3 *Pseudo-código*

Proposta de pseudo-código para o método do Ponto Fixo.

```
/* Função f(x) = x^2 - 1
 * Raiz x = 1.
 * Onde "e" é o valor do erro máximo escolhido.
 * ERR ou Erro Relativo Real.
 * O número real "a" indicam a primeira tentativa de solução.
 * xNS solução numérica.
 */

Inicio algoritmo

    Real: e, ERR, xNS, a
    Inteiro: n

    // Valores iniciais
    e = 0.003 ou 0.3%
    ERR = 0
    a = 2   // x1 = 2
    n = 0

    FUNÇÃO funcaoG(Real: x){
        Real: g
        g = (x + e^(x)) / (1 + x * e^(x));
        RETORNE g

    FUNÇÃO calculeERR(Real: xNS)
        Real: ERR
        ERR = |1 - xNS|
        RETORNE ERR

    FAÇA
        xNS = funcaoG(a)
        ERR = calculeERR(xNS)
        a = xNS
        n = n + 1
    // O laço continuará até que ERR seja menor ou igual que o erro "
```

```
            e"  escolhido.
        ENQUANTO (ERR > e)

        Exiba(xNS) // Solução numérica para f(x) = 0

    Fim algoritmo
```

2.4.4 Fluxograma

O fluxograma a seguir foi estruturado de acordo com o pseudo-código descrito anteriormente.

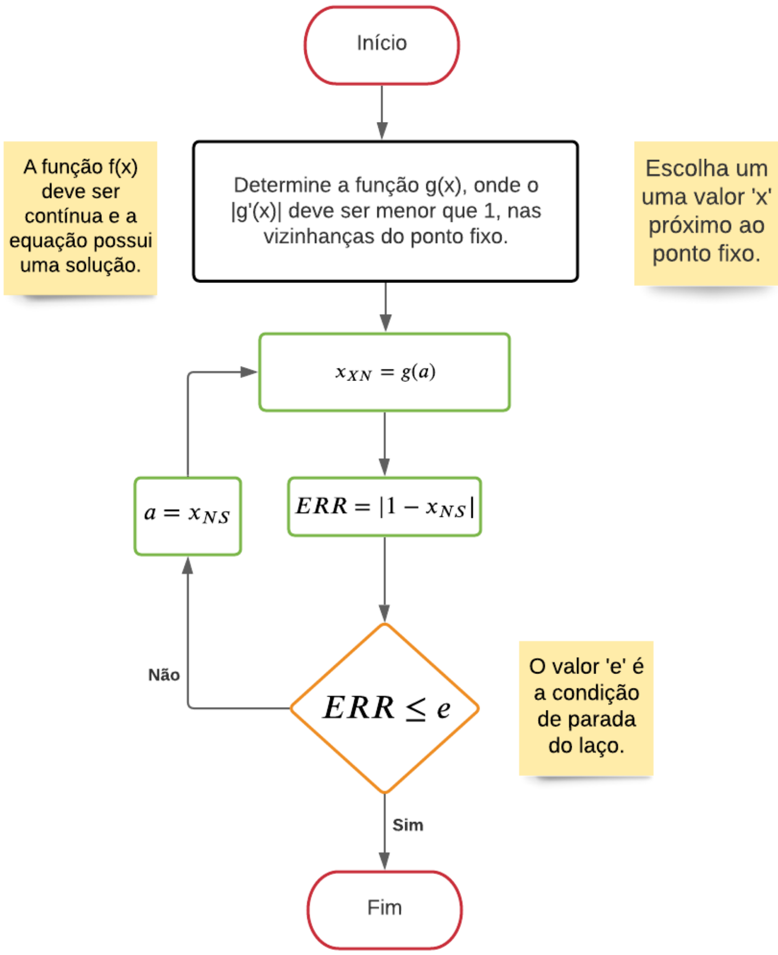

Figura 7 – Fluxograma - Método do Ponto Fixo.

2.4.5 Erro Relativo Real versus Número da Iteração

Os resultados obtidos indicam que o erro resultante deste método, a cada iteração, é bastante similar aos dois primeiros métodos. Este resultado é devido ao funcionamento do método. A cada iteração, a solução numérica (x_{NS}) realiza "saltos"sobre a solução exata. Esse "salto"é feito até que a solução encontrada apresente o erro desejado.

(Descrição) A distribuição de pontos vermelhos no gráfico indicam os resultados obtidos na implementação do método. O número da iteração e o erro relativo real formam pares ordenados (n, ERR(n)). A curva de 'ajuste' foi utilizada para facilicitar a visualização do comportamento do erro relativo real a cada iteração.

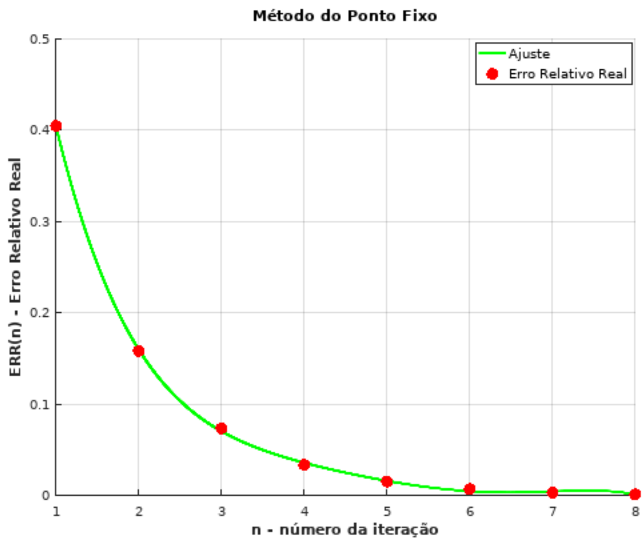

Figura 8 – Gráfico ERR vs n - Método do Ponto Fixo.

2.5 Método da Secante

2.5.1 Descrição do método

O método utiliza dois pontos na vizinhança da solução. Os dois pontos servem para definir uma linha reta (que é um reta secante ao gráfico), e o ponto onde essa reta intercepta o eixo-x definirá a nova solução estimada x_{NS}. De modo geral, a nova solução estimada a cada iteração será dada por:

$$x_{NS} = x_i - \frac{f(x_i) \cdot (x_{i-1} - x_i)}{f(x_{i-1}) - f(x_i)}$$

Onde x_i e x_{i-1} são pontos na vizinhança da solução. Estes pontos (x_i e x_{i-1}) na vizinhança da solução devem ser alterados a cada iteração. De forma que x_{i-1} receba o valor de x_i e, x_i receba o valor da solução numérica. O cálculo deve ser feito até que, o erro gerado pela solução seja menor que o erro mínimo.

2.5.2 Pseudo-código

Proposta de pseudo-código para o método da Secante.

```
/* Função f(x) = x^2 - 1
 * Raiz x = 1.
 * Onde "e" é o valor do erro máximo escolhido.
 * ERR ou Erro Relativo Real.
 * O número real "a" indicam a primeira tentativa de solução.
 * xNS solução numérica.
 */

Inicio algoritmo

    Real: e, ERR, xNS, a
    Inteiro: n

    // Valores iniciais
    e = 0.003 ou 0.3%
    ERR = 0
    a = 2 // x_i
    b = 3 // x_{i - 1}
    n = 0

    FUNÇÃO funcao(Real: x, Inteiro: ordem)
        // Tomando como f(x) = x^2 - 1
        Real: f
        Mudar (ordem) para:
            Caso 0:
                f = x^2 - 1
            Caso 1:
                f = 2 * x
        RETORNE f
```

```

    FUNÇÃO calculeERR(Real: xNS)
        Real: ERR
        ERR = |1 - xNS|
        RETORNE ERR

    FAÇA
        xNS = a - ( funcao(a, 0) * (b - a) ) / ( funcao(b, 0) -
            funcao(a, 0) )
        ERR = calculeERR(xNS)
        b = a
        a = xNS
        n = n + 1
    // O laço continuará até que ERR seja menor ou igual que o erro "
        e" escolhido.
    ENQUANTO (ERR > e)

    Exiba(xNS) // Solução numérica para f(x) = 0

Fim algoritmo
```

2.5.3 *Fluxograma*

O fluxograma a seguir foi estruturado de acordo com o pseudo-código descrito anteriormente.

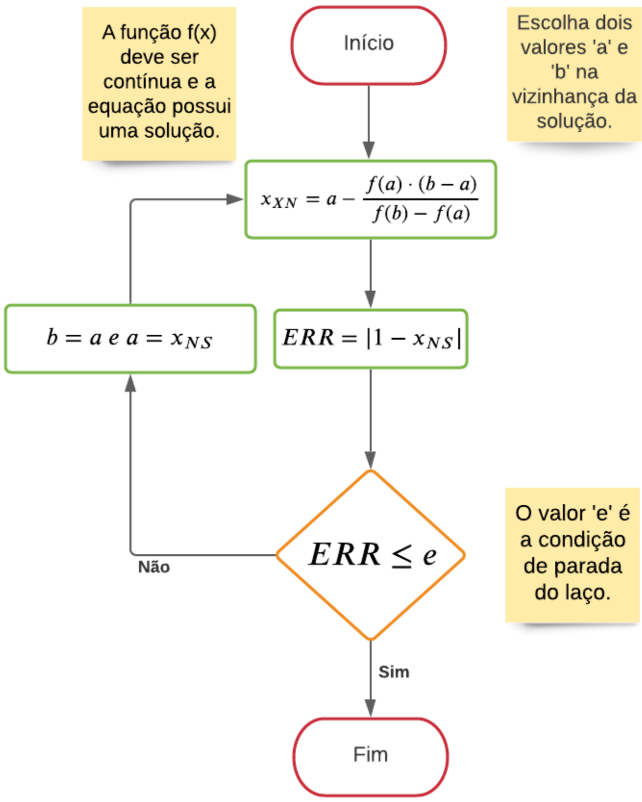

Figura 9 – Fluxograma - Método da Secante.

2.5.4 *Erro Relativo Real versus Número da Iteração*

Apesar de ser um dos métodos mais simples de se entender e implementar, este método foi um dos que demorou menos iterações para convergir. Verifica-se também uma certa similaridade entre este método e o método de Newton-Raphson, tanto no número de iterações necessárias para a solução numérica convergir, como no procedimento para a determinação de x_{NS}.

Isto acontece porque o método da Secante utiliza uma aproximação de $f'(x)$ para determinar x_{NS}. Podemos verificar isto por meio de

$$x_{NS} = x_i - \frac{f(x_i) \cdot (x_{i-1} - x_i)}{f(x_{i-1}) - f(x_i)}$$

que pode ser reescrita como:

$$x_{NS} = x_i - \frac{f(x_i)}{\frac{f(x_{i-1}) - f(x_i)}{x_{i-1} - x_i}}$$

e

$$\frac{f(x_{i-1}) - f(x_i)}{x_{i-1} - x_i} \approx f'(x_d)$$

(Descrição) A distribuição de pontos vermelhos no gráfico indicam os resultados obtidos na implementação do método. O número da iteração e o erro relativo real formam pares ordenados (n, ERR(n)). A curva de 'ajuste' foi utilizada para facilicitar a visualização do comportamento do erro relativo real a cada iteração.

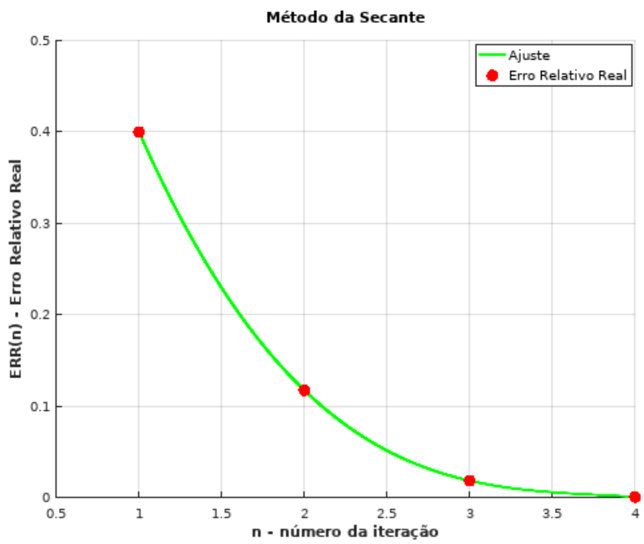

Figura 10 – Gráfico ERR vs n - Método da Secante.

3 RESULTADOS E CONCLUSÃO

Entre os métodos expostos neste trabalho, os que levaram menos iterações para convergir foram os métodos de Newton-Raphson e da Secante. Estes resultados são decorrentes da condições inicias do problema. No caso do método de Newton-Raphson, temos que a função escolhida é diferenciável e contínua. Dada a estimativa inicial $x_1 = 3$, suficientemente próxima da solução exata, o programa precisou de apenas 4 iterações para convergir. Já no caso do método da Secante, os dois pontos escolhidos inicialmente, $x_i = 2$ e $x_{i-1} = 3$, levaram o programa a convergir com o mesmo número de iterações.

O gráfico a seguir reúne informações sobre o erro relativo real a cada iteração, de cada método abordado neste trabalho.

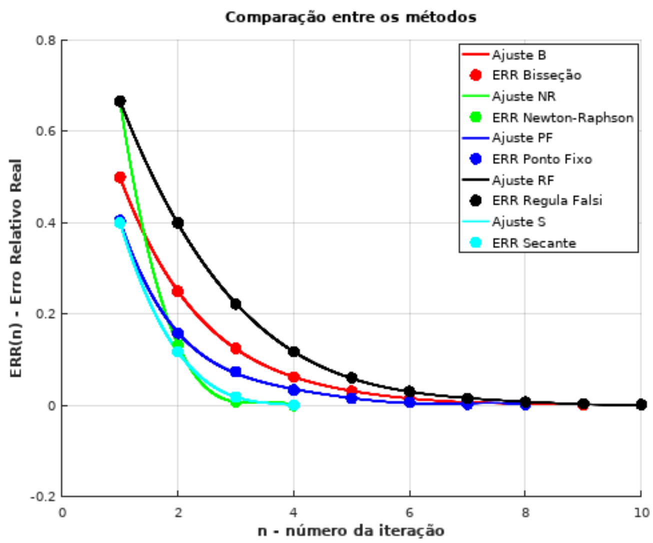

Figura 11 – Gráfico ERR vs n - Todos os métodos.

Podemos verificar que o método que apresenta maior erro a cada iteração, é o método Regula Falsi (curva e pontos na cor preta), seguido do método da Bisseção (curva e pontos na cor vermelha). Conclui-se que, para esta função, dentre os métodos de confinamento, aquele que apresentou melhores resultados foi o método da Bisseção. E, dentre os métodos abertos, o que demandou menos iterações para convergir, foi o método da Secante (curva e pontos na cor azul claro).

Ressalta-se que as condições iniciais do problema podem alterar os resultados encontrados, assim como o método que deve ser escolhido para encontrar a solução numérica desejada. Caso a função escolhida, não seja diferenciável, o método de Newton-Raphson não poderá ser aplicado. Restando apenas 4 métodos a serem utilizados. E ainda, se em um dado intervalo $[a,\ b]$, tal que a solução pertence a este intervalo, o valor de $f(a)f(b)$ seja maior que zero, os métodos Regula Falsi e da Bisseção também não poderão ser aplicados. Portanto, a escolha dos valores iniciais e o comportamento da função escolhida, podem alterar os resultados apresentados pelos métodos e, se o método consegue ou não convergir para a solução numérica desejada.