Projeto 1 de Cálculo Numérico

Gustavo Nascimento Soares
217530

24 de Setembro de 2018

1 Introdução

A *Equação de Butler-Volmer* é usada na modelagem de processos eletroquímicos relacionando a densidade de corrente com o potencial em um eletrodo:

$$f(x) = e^{\alpha x} - e^{(\alpha-1)x} - \beta$$

onde α e β são constantes características do sistema e e é o número de Euler.

Não é possível encontrar uma solução analítica para essa equação. Assim, se faz necessário o uso de métodos numéricos para encontrar o(s) valor(es) da(s) raiz(es) dela.

2 Metodologia

Para avaliar a raiz da *Equação de Butler-Volmer* nas constantes definidas pelo problema - $\alpha = 0,2$ e $\beta = 2$ - , foi construído um script para o software MATLAB®. O script usa os métodos numéricos da **bissecção** (MB) e de **Newton-Raphson** (MNR) como descritos em [1].

2.1 Método da Bissecção

São tomados dois limites a e b de um intervalo tal que $f(a) = -f(b)$. Com isso, é possível garantir que existe pelo menos um valor $a \leq r \leq b$ tal que $f(r) = 0$. A partir disso, o valor de $f(x)$ é avaliado, a cada iteração, no valor médio de a e b. Com base nesse valor, o limite inferior ou superior é alterado para diminuir o intervalo e se aproximar da raiz da função dentro desse intervalo. Esse processo é repetido até que os critérios de parada sejam atendidos.

2.2 Método de Newton-Raphson

Esse método é derivado do método do ponto fixo, em que para uma função $\phi(x)$, $\exists\epsilon(\phi(\epsilon) = \epsilon)$. O que o MNR faz é encontrar a função de iteração que acelera a convergência do método. Dessa maneira, o algoritmo pode ser aplicado usando a equação:

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

onde k é o índice de iteração. Com isso, é preciso fazer um chute para x_0 para que seja iniciada a execução do método. Para garantir a convergência rápida do método, é importante que seja escolhido um valor próximo o suficiente da raiz exata.

2.3 Resolução

Primeiramente, foi obtida uma representação gráfica do comportamento da curva para o intervalo de x de -5 a 5. Com essa representação, é possível concluir que existe uma única raiz dentro desse intervalo entre $x = 3$ e $x = 4$.

Com esse resultado, os limites inferior e superior iniciais para a implementação do MB foram definidos exatamente em $x = 3$ e $x = 4$. Além disso, o chute inicial para o MNR foi definido como o ponto médio desse intervalo.

Posteriormente, foram alterados os intervalos e chutes iniciais dos métodos para investigação do comportamento dos algoritmos em diferentes condições iniciais.

2.4 Critérios de parada

Para ambos os métodos, foi definido um erro máximo $\epsilon = 1e-12$. Para o MB, o erro foi determinado pelo tamanho do intervalo entre os limites inferior e superior $(b-a)$. Para o MNR, o erro foi a distância entre o valor da função obtido no x atual e 0.

Como o MATLAB® usa o padrão IEEE® 745 de dupla precisão, o expoente do erro escolhido não chega nem a metade do expoente limite (38) dessa representação de números de ponto flutuante. Com isso, é possível ter razoável confiança que não haverá *overflow* ou *underflow* nem erros por arredondamento em algoritmos significativos.

Definiu-se também um número máximo de iterações. Isso garante que o programa não execute indefinidamente caso haja problemas na resolução.

2.5 Código-Fonte

Segue abaixo o código-fonte usado no script juntamente com os devidos comentários para cada seção:

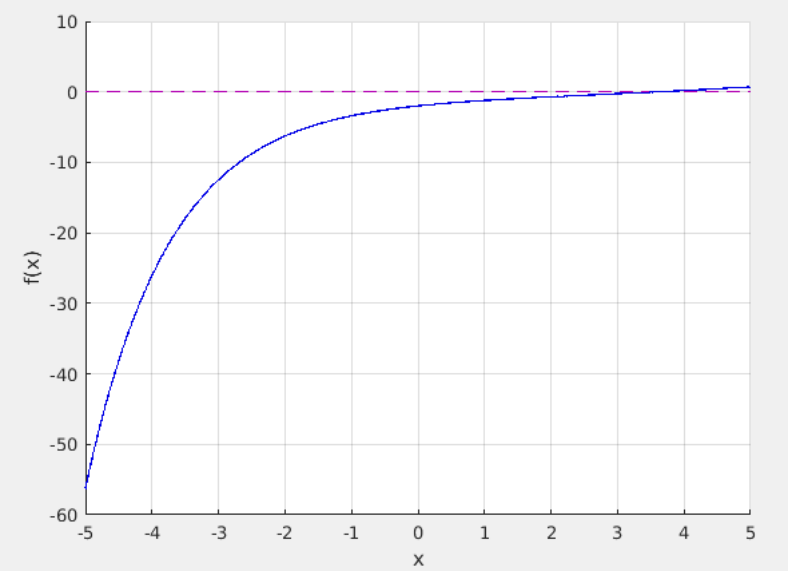
```
% Limpeza do workspace
clear all
clc

% Atribuição das constantes
EPSILON = 1e-12 % Erro aceitável
ALPHA = 0.2     % Constante da equação
BETA = 2        % Constante da equação

% Estabelecimento do intervalo em que a raiz se encontra
x = -5:0.01:5   % Intervalo de x para investigação da curva
y = exp(ALPHA*x) - exp((ALPHA-1)*x) - BETA      % Atribuição de x na eq.: f(x) = y
plot(x,y)

% Método da Bissecção
a = 3   % Valor mínimo do intervalo
b = 4   % Valor máximo do intervalo
[x_bissec, n_bissec] = bissecBV(a,b, ALPHA, BETA, EPSILON)

% Método de Newton
x_newt = (3+4)/2
n_max = 10
[x_newt, n_newt] = newtonBV(x_newt, ALPHA, BETA, EPSILON, n_max)

% Implementação das funções de resolução

% Método da Bissecção
function [r, n] = bissecBV(a, b, ALPHA, BETA, EPSILON)
 n = 0  % Número de iterações
 n_max = ceil((log10(b-a)-log10(EPSILON))/(log10(2)))    % Número máximo de iterações

 while b - a >= EPSILON & n < n_max

  y_k = exp(ALPHA*((a+b)/2)) - exp((ALPHA-1)*((a+b)/2)) - BETA

  if y_k > 0
   b = (a+b)/2
   n = n + 1
  elseif y_k < 0
   a = (a+b)/2
   n = n + 1
  else
   break
```

```
  end
 end
  r = (a+b)/2
end

% Método de Newton
function [r, n] = newtonBV(x0, ALPHA, BETA, EPSILON, n_max)
 n = 0

 while abs(exp(ALPHA*x0) - exp((ALPHA-1)*x0) - BETA) >= EPSILON & n < n_max

  x0 = x0 - (exp(ALPHA*x0) - exp((ALPHA-1)*x0) - BETA)/(exp((ALPHA-1)*x0)*(ALPHA*
(exp(x0)-1)+1))
  n = n + 1

 end
 r = x0
end
```

3 Resultados

O gráfico abaixo traz a curva da equação modelo para o intervalo investigado:

Gráfico 01: Curva da Equação de Butler-Volmer



Os valores encontrados para a raiz aproximada da *Equação de Butler-Volmer* com $\alpha = 0,2$ e $\beta = 2$ estão nas Tabela 01 e Tabela 02.

Tabela 01: Método da Bissecção

a	b	$\overline{x}$	iterações
3	4	3,603732449185	40
0	4	3,603732449186	40
-5	5	3,603732449186	44
-10	10	3,603732449186	45

Tabela 02: Método de Newton-Raphson

x_0	$\overline{x}$	iterações
3,5	3,603732449186	3
0	3,603732449186	5
2	3,603732449186	4
10	3,603732449186	6

4 Discussão

Ambos os métodos conseguem aferir com precisão e exatidão o valor aproximado da raiz mesmo em condições iniciais bem distintas. É posssível concluir isso a partir da variação entre os resultados obtidos para os diferentes cenários testados. Neles, há divergência nos valores apenas na décima segunda casa decimal. Divergência essa ocorrida devido ao valor definido para o erro ϵ.

Foi constatada a hipótese inicial de que não haveria problemas com *overflow* ou *underflow* ou erros por arredondamento em algoritmos significativos por decorrência da distância do valor do erro definido ao limite de precisão permitido pelo *software* utilizado.

É notável a eficiência computacional que o MNR tem sobre o MB. O número de iterações necessárias para o MNR é consideravelmente menor em todos os cenários investigados. Além disso, o programa executor do algoritmo levará menos tempo por iteração do MNR com relação ao MB porque aquele necessita notadamente de menos instruções do que este, apesar de exigir instruções mais complexas.

5 Referências

[1] RUGGIERO, M., LOPES, V. Cálculo Numérico - *Aspectos Teóricos e Computacionais*. Segunda Edição. Pearson: 1997.