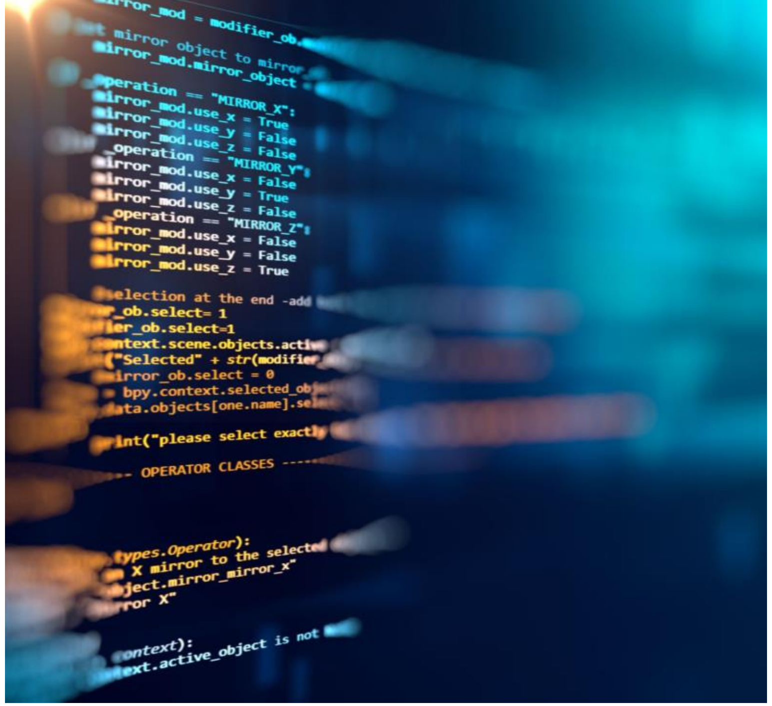
SEMANA 3



COMPUTAÇÃO CIENTÍFICA

ERROS EM REPRESENTAÇÕES NUMÉRICAS E

ARITMÉTICA EM PONTO FLUTUANTE

PARTE II

ERRO NUMÉRICO TOTAL

Computação Científica

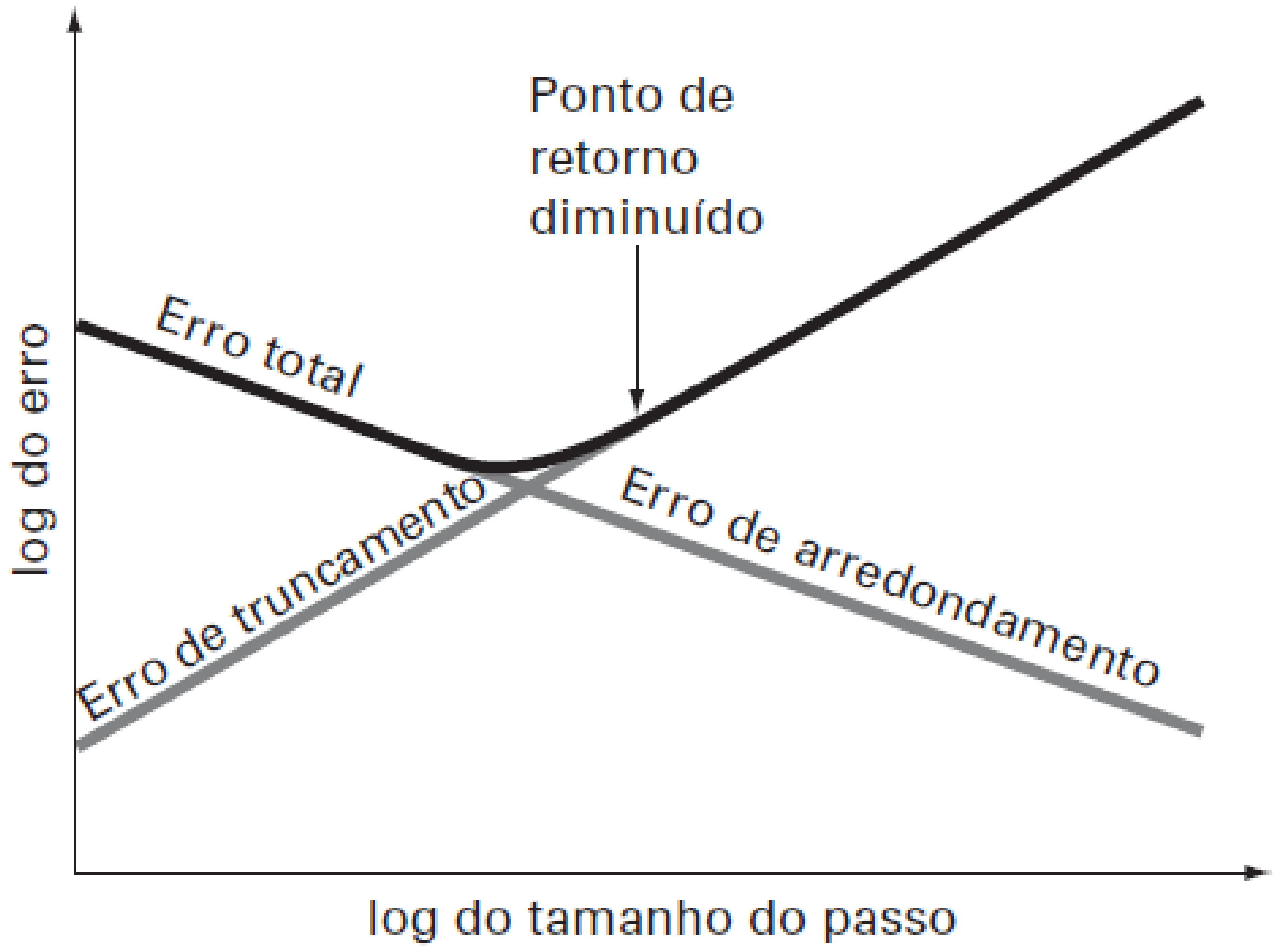
prof. Marco Villaça

ERRO NUMÉRICO TOTAL
(Erro total)

- Erro total = erro arredondamento + erro de truncamento
- **Erro de arredondamento**:
  - ✓ Minimiza-se com o aumento do número de algarismos significativos;
  - ✓ Podem aumentar por:
    - ▶ Cancelamentos na subtração;
    - ▶ Número de cálculos da análise
- **Erro de truncamento:**
  - ✓ Minimiza-se com a diminuição do passo, o que leva a aumentar o erro de arredondamento

ERRO NUMÉRICO TOTAL
(Erro total)

- O gráfico abaixo mostra que existe um passo de cálculo apropriado, chamado de ponto de retorno diminuído, onde o erro de arredon-damento começa a neutralizar as vantagens de uma redução do passo de cálculo.



ERRO NUMÉRICO TOTAL
(Erro total)

- Uma aproximação por diferença centrada para a derivada primeira pode ser escrita como:

$$f'(x_i) = \underbrace{\frac{f(x_{i+1}) - f(x_{i-1})}{2h}}_{\text{Aproximação de Primeira ordem}} - \underbrace{\frac{f^{(3)}(\xi)}{6}h^2}_{\text{erro de truncamento}}$$

- Devido ao uso de computadores digitais, os valores da função incluem o erro de arredondamento, como em:

$$f(x_{i+1}) = \tilde{f}(x_{i+1}) + e_{i+1}$$
$$f(x_{i-1}) = \tilde{f}(x_{i-1}) + e_{i-1}$$

- onde os $\tilde{\boldsymbol{f}}s$ são os valores arredondados e os $\boldsymbol{e}$s são os erros de arredondamento associados

ERRO NUMÉRICO TOTAL
(Erro total)

- Acrescentando os erros de arredondamento a equação da aproximação, resulta

$$f'(x_i) = \underbrace{\frac{\tilde{f}(x_{i+1}) - \tilde{f}(x_{i-1})}{2h}}_{\text{Aproximação de Primeira ordem}} + \underbrace{\frac{e_{i+1} - e_{i-1}}{2h}}_{\text{erro de arredondamento}} - \underbrace{\frac{f^{(3)}(\xi)}{6} h^2}_{\text{erro de truncamento}}$$

- Assumindo que:
  - O valor absoluto de cada componente do erro de arredondamento tenha um limite superior ε, o valor máximo de $e_{i+1} - e_{i-1}$ será 2ε;
  - O valor da derivada terceira tenha um valor absoluto máximo de M.

ERRO NUMÉRICO TOTAL
(Erro total)

$$f'(x_i) = \frac{\tilde{f}(x_{i+1}) - \tilde{f}(x_{i-1})}{2h} + \frac{e_{i+1} - e_{i-1}}{2h} - \frac{f^{(3)}(\xi)}{6}h^2$$

- Um limite superior do valor absoluto do erro total pode ser representado por

$$ERRO\ TOTAL = \left| f'(x_i) - \frac{\tilde{f}(x_{i+1}) - \tilde{f}(x_{i-1})}{2h} \right| \leq \boldsymbol{\frac{\varepsilon}{h} - \frac{M}{6}h^2}$$

- Derivando–se a **equação acima** e igualando a zero (ponto de mínimo), obtém um tamanho de passo ótimo (prove) :

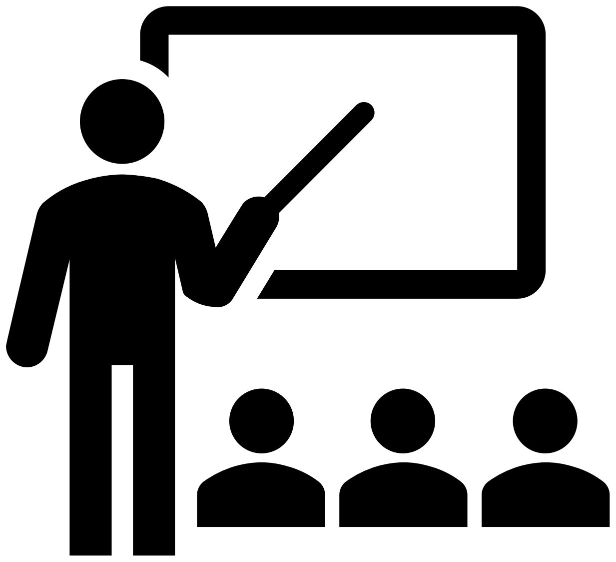
$$\boldsymbol{h_{ótimo} = \sqrt[3]{\frac{3\varepsilon}{M}}}$$

EXEMPLO

- Use uma aproximação por diferença centrada de $O(h^2)$ para estimar a derivada primeira da função a seguir em $x = 0{,}5$

$$f(x) = -0{,}1x^4 - 0{,}15x^3 - 0{,}5x^2 - 0{,}25x + 1{,}2$$

- Utilizando o Scilab, faça o cálculo iniciando com h = 2 e divida progressivamente o tamanho do passo por um fator 8 para demonstrar que o erro de arredondamento torna-se dominante à medida que o tamanho do passo é reduzido
- Com o auxílio de um gráfico *erro total x h*, relacione os resultados obtidos com a equação de $h_{otm.}$



EXEMPLO

```
clear;  j=1;
xi = input('Entre com o valor de xi: ');
h = input('Entre com o passo de cálculo inicial: ');
disp('Entre com os coef da f. polinomial entre ')
vetor = input(' no formato [a0 a1 . . . an] : ');
f = poly(vetor,'x','c’);
disp('f(x)',f);
flinha = derivat(f);
disp('f´(x)',flinha);
vreal = horner(flinha,xi);
printf("O valor real da derivada em x = %f é %f\n",xi,vreal);
printf(" tamanho do passo | diferenca finita |   erro total\n ");
```

EXEMPLO

```
while h >= 1.0D-10
    H(j)=h;
    dfdt(j) = (horner(f,xi+h)-horner(f,xi-h))/(2*h);
    e(j) = 100*abs((vreal - dfdt(j))/vreal);
    printf("%16.10f | %16.10f | %16.10f\n ",h,dfdt(j),e(j));
    h = h/8;
    j=j+1;
end
```

EXEMPLO

```
xlabel('Tamanho do passo');
ylabel('Erro total');
plot2d(H,e,style=[color('blue4')],logflag='ll');
xgrid;
f2linha = derivat(flinha);
f3linha = derivat(f2linha);
M = abs(horner(f3linha,xi));
disp('M', M,);
hotm = (3*%eps/M)^(1/3.);
disp('hotm',hotm);
```

EXEMPLO
Execução do script

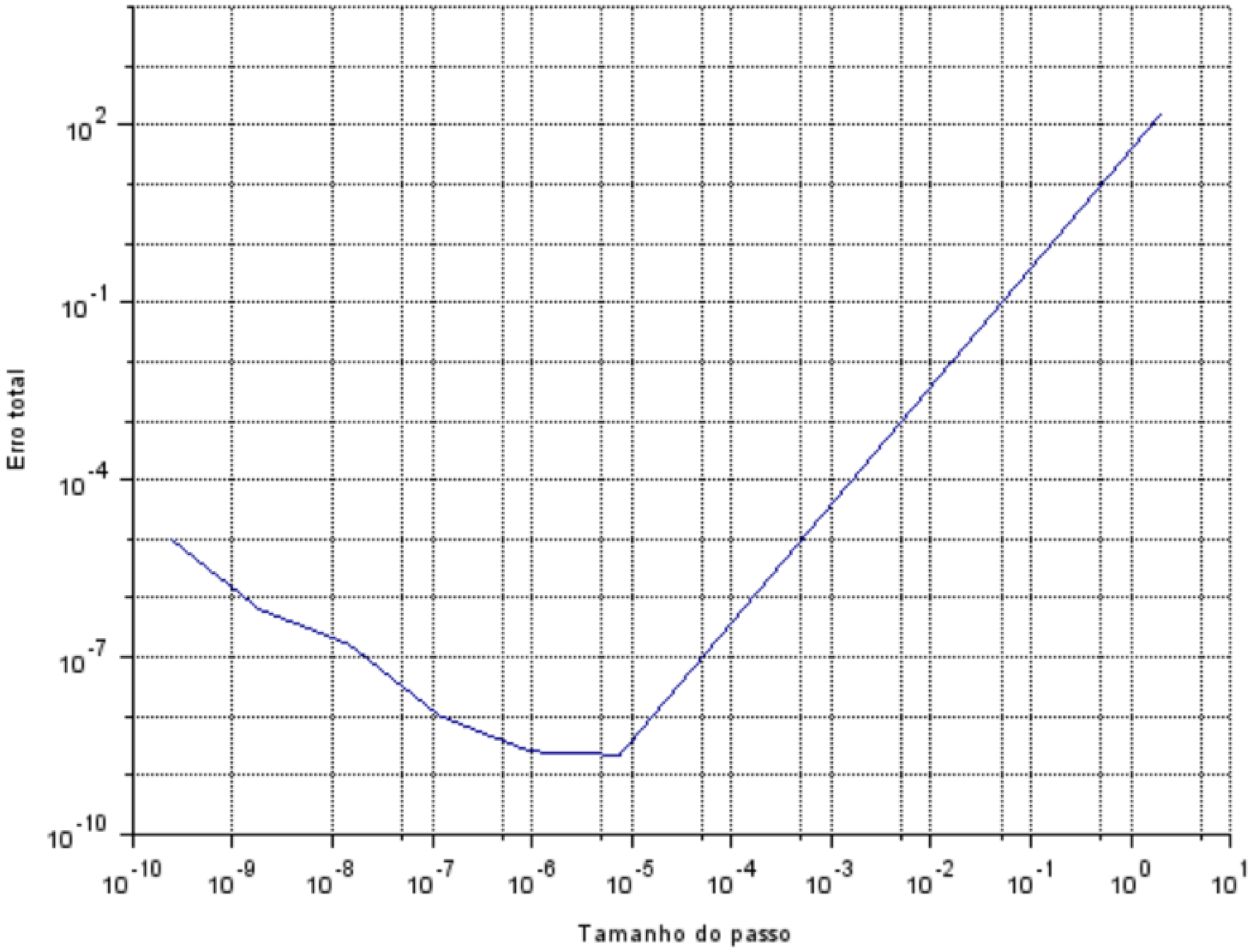
```
Entre com o valor de xi: 0.5

Entre com o passo de cálculo inicial: 2

  "Entre com os coef da f. polinomial entre "

 no formato [a0 a1 . . . an] : [1.2 -0.25 -0.5 -0.15 -0.1]

  "f(x)"

  1.2 -0.25x -0.5x² -0.15x³ -0.1x⁴

  "f´(x)"

  -0.25 -x -0.45x² -0.4x³

O valor real da derivada em x = 0.500000 é -0.912500
```

EXEMPLO
Execução do script

```
tamanho do passo | diferenca finita |   erro total
    2.0000000000 |    -2.3125000000 |   153.4246575342
    0.2500000000 |    -0.9343750000 |     2.3972602740
    0.0312500000 |    -0.9128417969 |     0.0374571918
    0.0039062500 |    -0.9125053406 |     0.0005852686
    0.0004882813 |    -0.9125000834 |     0.0000091448
    0.0000610352 |    -0.9125000013 |     0.0000001428
    0.0000076294 |    -0.9125000000 |     0.0000000022
    0.0000009537 |    -0.9125000000 |     0.0000000026
    0.0000001192 |    -0.9125000001 |     0.0000000102
    0.0000000149 |    -0.9125000015 |     0.0000001633
    0.0000000019 |    -0.9124999940 |     0.0000006532
    0.0000000002 |    -0.9124999046 |     0.0000104512
 "M"
  2.1
 "hotm"
  0.0000068
```

EXEMPLO
Execução do script

Gráfico



Outras fontes de erro (externas a computação)

- Enganos – atribuídos ao ser humano, podem ocorrer em qualquer estágio do processo de modelagem, afetando todas as outras componentes de erro.
- Erros de formulação ou inerentes ao modelo – atribuídos a modelos matemáticos incompletos, levando a resultados inapropriados
- Erros inerentes aos dados – As incertezas (ou erros) cometidas nas leituras das grandezas diretas, relacionadas com a precisão dos instrumentos de medida e com o próprio método experimental, podem ter grande repercussão no resultado final

ZEROS DE FUNÇÕES REAIS: MÉTODO DE BISSEÇÃO, NEWTON-RAPHSON E SECANTE

PARTE I

MÉTODO DA BISSECÇÃO

Computação Científica

prof. Marco Villaça

Zeros de funções reais
Apresentação do problema

- Considere o problema do bungee jumping:

  Dado um coeficiente de arraste de 0,25 kg/m, determine a massa m do saltador que leve a sua velocidade a superar uma certa velocidade após um tempo de queda. Lembrando a solução analítica para v(t):

$$v(t) = \sqrt{\frac{g \cdot m}{c_d}} \tanh\left(\sqrt{\frac{g \cdot c_d}{m}}\, t\right) \quad (m/s)$$

- Não é possível, porém, isolar m na equação acima e resolver o problema analiticamente.
- Como, então, encontrar a massa m do saltador ?

Zeros de funções reais
Apresentação do problema

- Uma abordagem possível para o problema, envolve a subtração de $v(t)$ de ambos os lados da equação, para criar uma nova função:

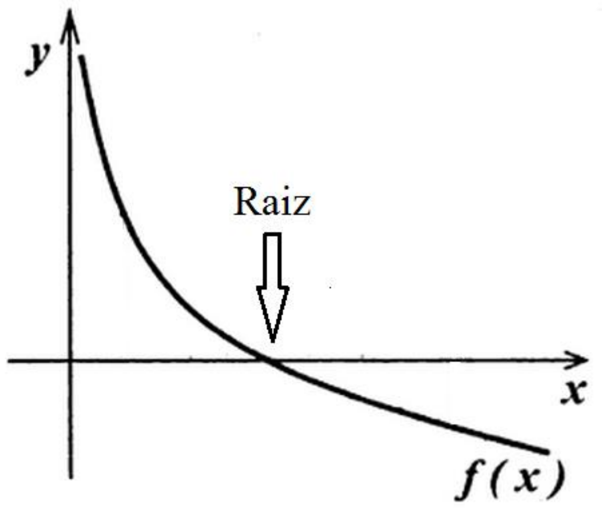
$$f(m) = \sqrt{\frac{g \cdot m}{c_d}} \tanh\left(\sqrt{\frac{g \cdot c_d}{m}}\ t\right) - v(t)$$

- O problema agora é encontrar o valor de m que torna a função igual a zero.

- Neste capítulo, estudar-se-á alguns métodos numéricos e gráficos para a determinação das raízes de equações similares a função acima.

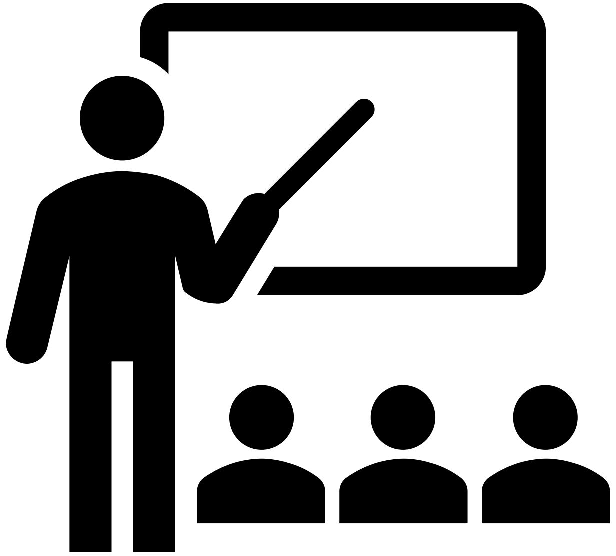
Método gráfico

- Um método simples para obter uma estimativa das raízes de uma equação $f(x) = 0$, é construir um gráfico da função e observar onde ela intercepta o eixo x, ponto que representa o valor de x em que $f(x) = 0$.



EXEMPLO 1

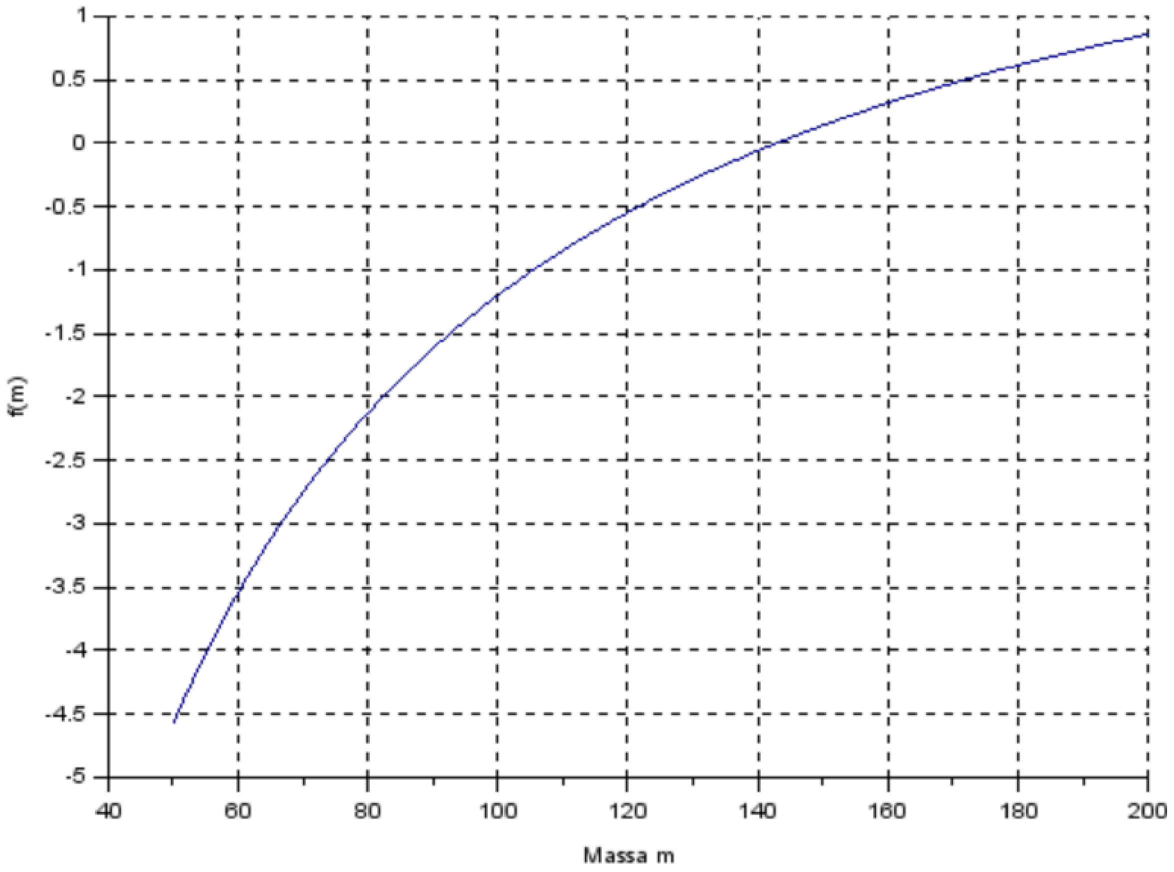
- Use a abordagem gráfica com o Scilab e faça uma estimativa da massa que deve ter um saltador de bungee jumping para que a sua velocidade ultrapasse 36 m/s após 4 s de queda.
- Considere o coeficiente de arraste 0,25 kg/m e a aceleração da gravidade g = 9,81 m/s^2.



EXEMPLO 1

```
CD = 0.25; G = 9.81; v =36; t = 4;
m = linspace(50,200,100);
fm = sqrt(G*m/CD).*tanh(sqrt(G*CD./m)*t) - v;
xlabel('Massa m');
ylabel('f(m)');
plot2d(m,fm,style=[color('blue4')]);
xgrid;
```

EXEMPLO 1
Método gráfico



- Observando o gráfico gerado no Scilab, constata-se que a função intercepta o eixo *m* entre 140 e 145 kg.
- Adotando-se, *m* = 142,5 e verificando na equação abaixo:

$$f(142{,}5) = \sqrt{\frac{g \cdot m}{c_d}} \tanh\left(\sqrt{\frac{g \cdot c_d}{m}}\; t\right) - v(t) \cong -0.005$$

Métodos intervalares e estimativas iniciais

- As interpretações gráficas são importantes pois fornecem uma "aproximação" inicial para métodos numéricos.
- Classes de métodos numéricos:
  - Intervalares:
    - ✓ São baseados em duas aproximações iniciais que delimitam a raiz;
    - ✓ Sempre funcionam, porém convergem lentamente.
  - Abertos:
    - ✓ As aproximações iniciais não precisam delimitar a raiz;
    - ✓ Podem divergir.

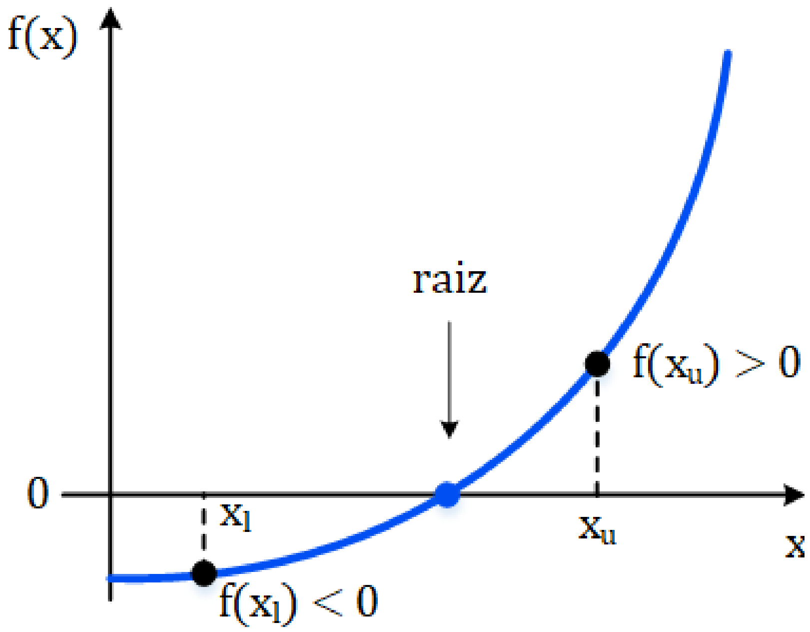
Método da bissecção

- Ao aplicar a técnica gráfica no exemplo anterior, observou-se que $f(m)$ muda de sinal em lados opostos da raiz.
- Em geral, se $f(x)$ for real e contínua em um intervalo de x_l a x_u e $f(x_l)$ e $f(x_u)$ tiverem sinais opostos, isto é,

$$f(x_l) \cdot f(x_u) < 0$$

então existe pelo menos

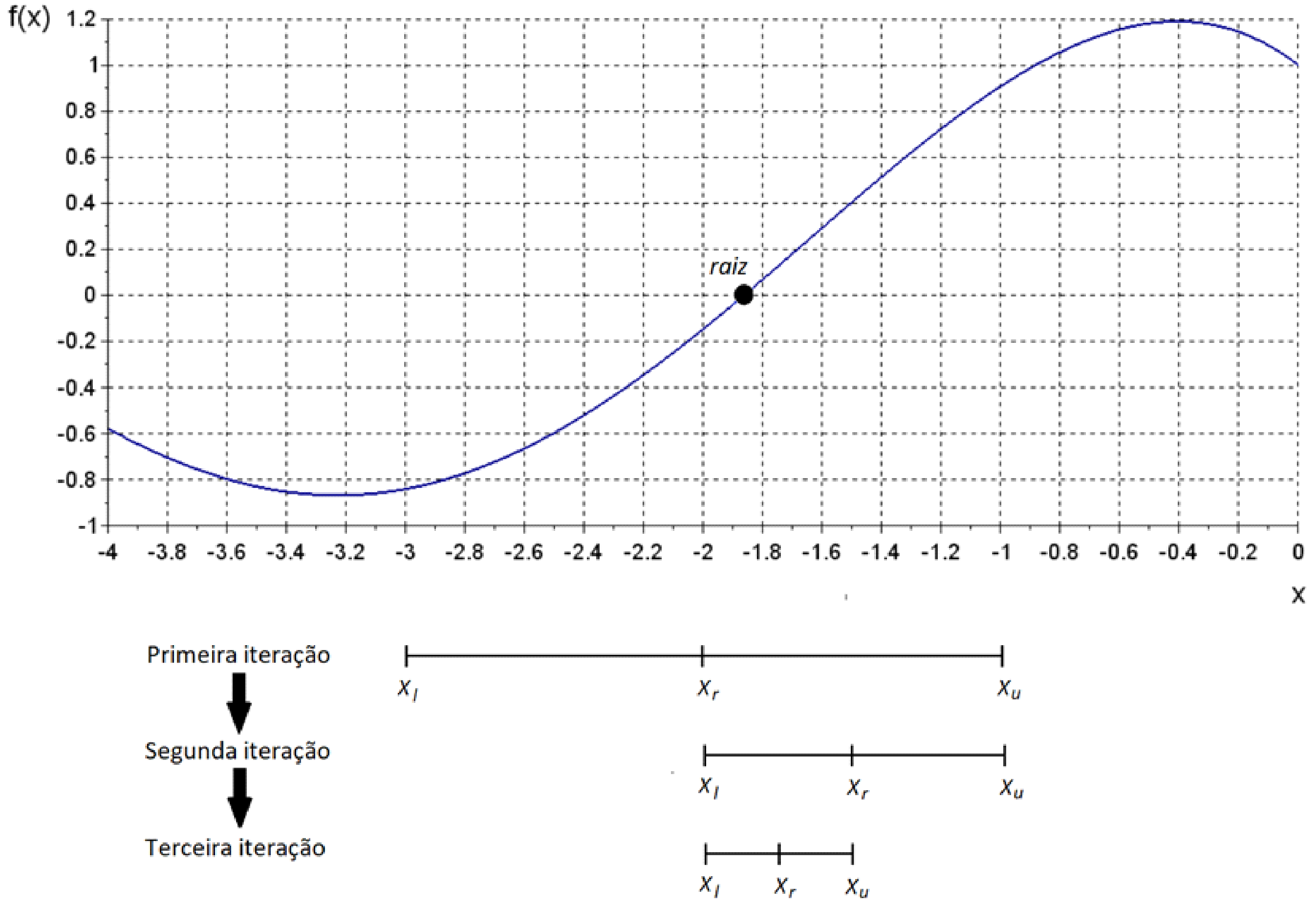
uma raiz real entre x_l a x_u.



Método da bissecção

- O método da bisseção usa como estratégia para a busca da raiz, a divisão do intervalo entre x_l e x_u em duas partes iguais para determinar o subintervalo em que f mantém a troca sinal. Esse intervalo torna-se, então o intervalo para a próxima iteração.
- O processo é repetido até que a raiz seja conhecida com a precisão desejada.

Método da bissecção
Descrição gráfica



Exercício 1

- Use o método da bisseção para obter a massa que deve ter um saltador de bungee jumping para que a sua velocidade ultrapasse 36 m/s após 4 s de queda.
- Considere o coeficiente de arraste 0,25 kg/m e a aceleração da gravidade g = 9,81 m/s².
- Pare o processo quando $E_a < 0{,}1$.
- Registre cada iteração no papel, utilizando o auxílio do console do Scilab.

$$f(m) = \sqrt{\frac{9{,}81 \cdot m}{0{,}25}} \tanh\left(\sqrt{\frac{9{,}81 \cdot 0{,}25}{m}}\, 4\right) - 36$$

Exercício – primeira iteração

- Início do procedimento ($m = x$):
- Primeira iteração:
- Com $x_l^1 = 140$ e $x_u^1 = 145$:

$$x_r^1 = \frac{x_u^1 + x_l^1}{2} = \frac{145 + 140}{2} = 142{,}5$$

- O erro absoluto aproximado vale:

$$E_a^1 = \frac{x_u^1 - x_l^1}{2} = \frac{145 - 140}{2} = 2{,}5$$

Exercício - preparativos para a segunda iteração

- Como $E_a^1 = 2{,}5 > 0{,}1$, iniciam-se os preparativos para a próxima iteração, calculando-se o valor da função no extremo inferior e no ponto médio:

$$f\left(x_l^1\right) = \sqrt{\frac{9{,}81 \cdot \mathbf{140}}{0{,}25}} \tanh\left(\sqrt{\frac{9{,}81 \cdot 0{,}25}{\mathbf{140}}}\ 4\right) - 36 = -0.0569853$$

$$f\left(x_r^1\right) = \sqrt{\frac{9{,}81 \cdot \mathbf{142,5}}{0{,}25}} \tanh\left(\sqrt{\frac{9{,}81 \cdot 0{,}25}{\mathbf{142,5}}}\ 4\right) - 36 = -0.0048683$$

$$f(x_l^1) = -0.0569853$$

$$f(x_r^1) = -0.0048683$$

Exercício – primeira iteração

- Como

$$f(x_l^1) \cdot f(x_r^1) > 0$$

a troca de sinal ocorre no subintervalo entre o ponto médio e o extremo superior, assim redefine-se a extremidade inferior com o valor do ponto médio. Assim,

$$x_u^2 = x_u^1 = 145$$

$$x_l^2 = x_r^1 = 142{,}5$$

- Resposta: ao final de 6 iterações, você deve encontrar:

$$\mathrm{m} = x_r^6 = 142{,}73438\,;\; x_l^6 = 142{,}65625;\; x_u^6 = 142{,}8125$$

Método da bissecção
Critério de parada

- Um critério normalmente definido para parar o método é estimar o erro como a diferença entre as aproximações prévia e atual e compará-lo com uma tolerância pré especificada:

$$|\varepsilon_a| = \left|\frac{x_r^{novo} - x_r^{velho}}{x_r^{novo}}\right| \times 100\ \% < \epsilon_s$$

onde x_r^{novo} é a raiz da iteração atual e x_r^{velho} é a raiz da iteração anterior.

Método da bissecção
Considerações

- Para efeito de análise, emprega-se método da bisseção para resolver o problema do Exercício 1, com:

$$x_l^1 = 50 \qquad x_u^1 = 200 \qquad \varepsilon_s = 0{,}5\%$$

sabendo que o valor exato da raiz é 142,7376

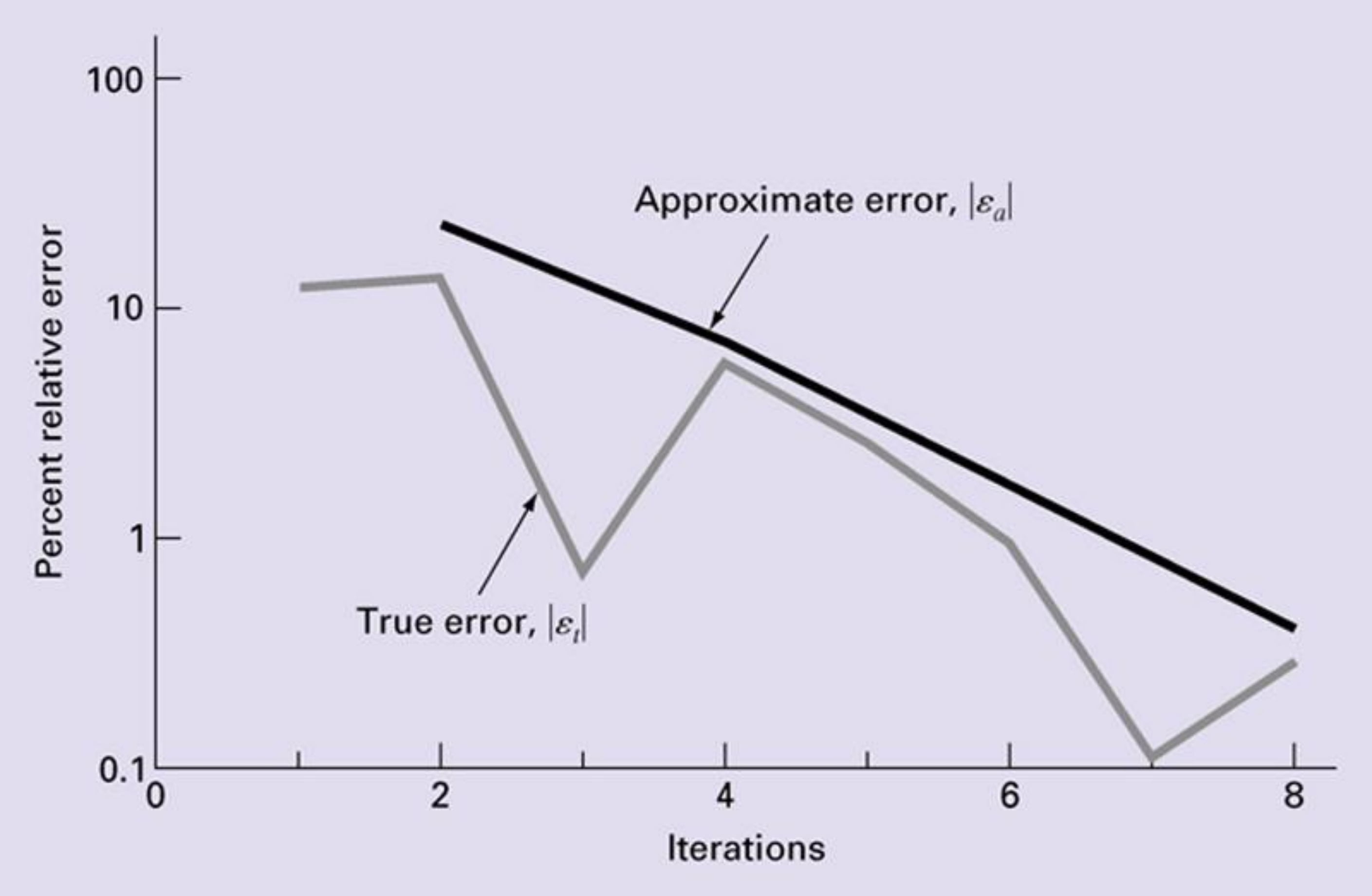
- Nas páginas seguintes apresenta-se:
  - ✓ Uma tabela com os valores de x_l, x_u, x_r, e dos erros relativos percentual aproximado $|\varepsilon_a|$e verdadeiro $|\varepsilon_t|$
  - ✓ Um gráfico comparando os erros aproximado e verdadeiro

Método da bissecção
Considerações

- Na tabela abaixo, verifica-se que o erro fica abaixo de $\varepsilon_s = 0{,}5\%$ após 8 iterações.

Iteration	x_l	x_u	x_r	$\lvert\varepsilon_a\rvert$ (%)	$\lvert\varepsilon_t\rvert$ (%)
1	50	200	125		12.43
2	125	200	162.5	23.08	13.85
3	125	162.5	143.75	13.04	0.71
4	125	143.75	134.375	6.98	5.86
5	134.375	143.75	139.0625	3.37	2.58
6	139.0625	143.75	141.4063	1.66	0.93
7	141.4063	143.75	142.5781	0.82	0.11
8	142.5781	143.75	143.1641	0.41	0.30

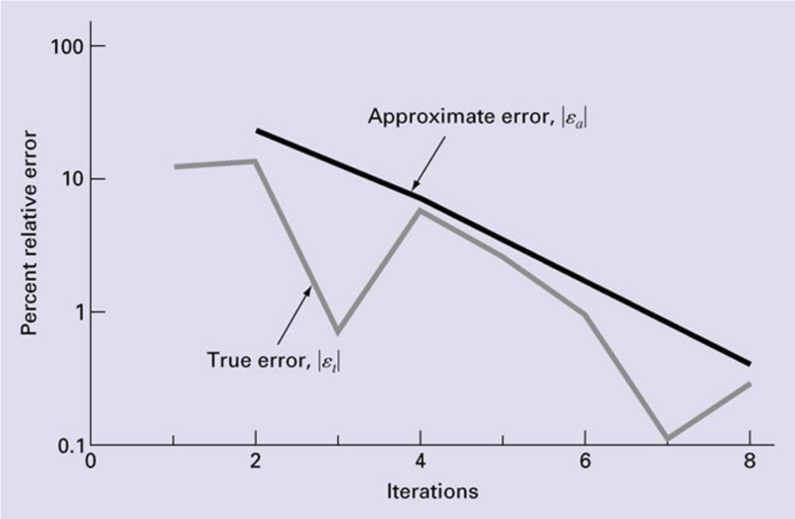
Método da bissecção Considerações

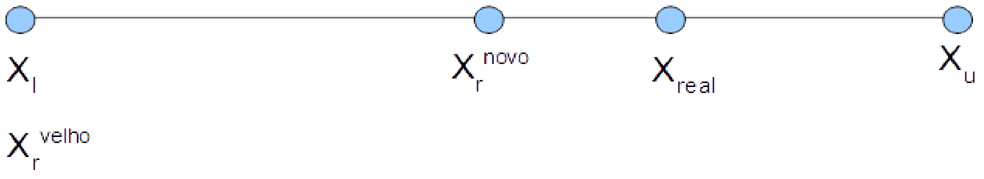


- Erros aproximado e verdadeiro em função do número de iterações Fonte: CHAPRA (2012, p. 135).

Método da bissecção
Considerações

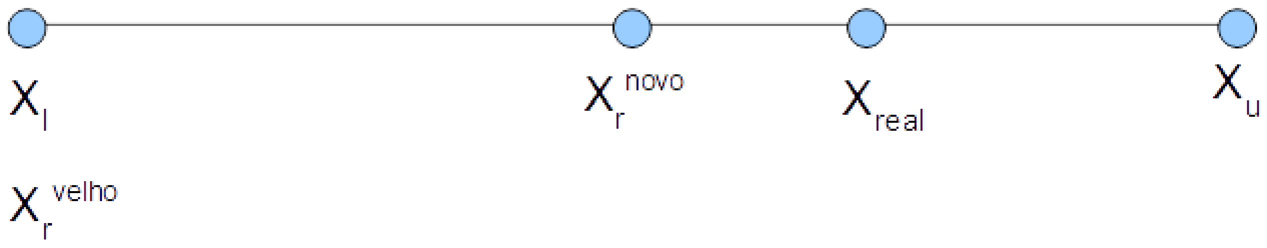
- Do gráfico, observa-se que os erros verdadeiro e aproximado:
  - ✓ Estão bem distantes quando a raiz exata está no centro do intervalo;
  - ✓ Estão bem próximos quando a raiz exata está próxima as extremidades do intervalo
- Ainda, como $|\varepsilon_a|$ sempre é maior que $|\varepsilon_t|$, quando $|\varepsilon_a|$ torna-se menor que ε_s , pode se ter a certeza que a raiz é conhecida e tão exata quanto o nível aceitável pré-especificado





Método da bissecção Considerações

- No método da bisseção, $|\varepsilon_a|$ será sempre maior que $|\varepsilon_t|$, por que cada vez que uma raiz é localizada usando a bisseção, sabe-se que a raiz verdadeira está em algum ponto do intervalo de comprimento $\Delta x = x_u - x_l$, ou a menos de $\Delta x/2$ da nossa estimativa.



- Ou seja, a expressão de erro aproximado fornece um limitante para o erro verdadeiro.

Método da bissecção
Considerações

- Além da simplicidade da análise do erro, outra vantagem do método é que o número de iterações necessária para chegar a um erro absoluto aproximado pode ser calculado a priori:
  - Após a primeira iteração o erro absoluto é

$$E_a^1 = \frac{x_u^1 - x_l^1}{2} = \frac{\Delta x^1}{2}$$

  onde o sobrescrito indica a iteração.
  - Como cada iteração divide o erro por 2, o erro está relacionado ao número de iterações n por:

$$E_a^2 = \frac{\Delta x^1}{2^n}$$

Método da bissecção
Considerações

- Substituindo E_a^n pelo erro desejado $E_{a,d}$, obtém-se

$$E_{a,d} = \frac{\Delta x^1}{2^n}$$

- Equação que resolvida para n resulta:

$$n = \frac{\log_{10}\left(\Delta x^1 / E_{a,d}\right)}{\log_{10} 2} \cong 3{,}32 \log_{10}\left(\Delta x^1 / E_{a,d}\right)$$

ou seja, conhecido um erro absoluto aceitável, a equação acima fornecerá o número de iterações necessárias

$m = x_r^6 = 142{,}73438$; $x_l^6 = 142{,}65625$; $x_u^6 = 142{,}8125$

$n \cong 3{,}32 \log_{10}\left(\Delta x^1 / E_{a,d}\right)$

Método da bissecção Considerações

- Por exemplo, quando no Exercício 1, após 6 iterações, foi alcançada a condição de parada estipulada ($E_a < 0{,}1$) , o erro absoluto era

$$E_a^6 = \frac{\Delta x^6}{2} = \frac{142{,}8125 - 142{,}65625}{2} \cong 0{,}07812 < E_{a,d}$$

- Substituindo o valor de $E_{a,d}$ na equação que calcula o número de iterações n necessárias, resulta:

$$n \cong 3{,}32 \log_{10}(5/0{,}1) = 5{,}64 \to 6$$

Função Scilab

Método da Bissecção

Função Scilab

```
function [raiz, i]=bissecao(funcao, xl, xu, es)
// Cálculo das raizes pelo processo da bisseção
// function [raiz,iter]=bissecao(funcao, xl, xu, es)
// onde raiz é a raiz procurada de funcao
//      i é o num de iterações  para o erro especificado
//      funcao é a função de entrada literal em x
//      xl é o limite inferior do intervalo de busca
//      xu é o limite superior do intervalo de busca
//      es é o criterio de parada que é opcional
// Exemplo de chamada:
// exec('path\ bissecao.sci',-1)
// fun = 'log(x) + x'
// [raiz,iter]=bissecao(fun, 0.1, 2, 0.1)
```

Função Scilab

```
// preparação
x = xu;     fu = evstr(funcao);
x = xl;     fl = evstr(funcao);
if (fu*fl >= 0) then
   error('Nenhuma raiz no intervalo dado');
end
i = 0; ea=100; xr_novo = xl;
// se es nao foi estabelecido usa 0.0001%
if argn(2) < 4 then
      es = 0.0001;
end
```

Função Scilab

```
// inicio do processo iterativo
   printf("Iter\tErro aprox.%%\tRaiz\t\txl\t\txu\n");
   while ea > es do
      xr_velho = xr_novo;
      xr_novo = (xu + xl)/2;
      if xr_novo ~=0 then // xr_novo não pode ser zero
         ea = abs((xr_novo - xr_velho)/xr_novo)*100;
      end
      i=i+1;
      printf("%d\t%f\t%f\t%f\t%f\n",i,ea,xr_novo,xl,xu);
      x=xr_novo;        fr = evstr(funcao);
      x = xl;           fl = evstr(funcao);
      if(fl*fr < 0) then
         xu = xr_novo;
      elseif(fl*fr > 0) then
         xl = xr_novo;
      else
            break;
      end
   end
   raiz = xr_novo;
endfunction
```

- Use a método da bisseção para estimar **manualmente** as raízes de

$$f(x) = \cos(x) - x \cdot e^{x}$$

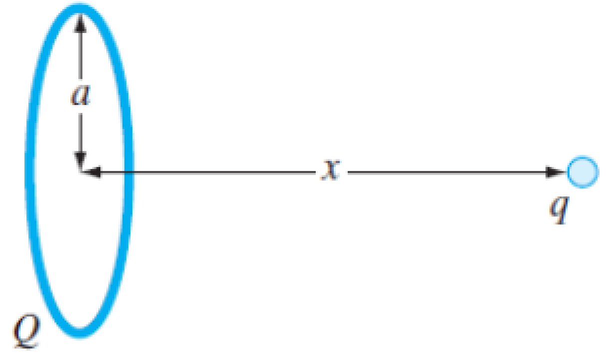
no intervalo $-2 \leq x \leq 0$, utilizando com critério de parada $|\varepsilon_a| < 1\ \%$.

- Confira o resultado utilizando a função Scilab apresentada.

Exercício 2

- Utilize a função Scilab com $\varepsilon_s = 0,0001\ \%$ para resolver o seguinte problema:
- Uma carga total *Q* está uniformemente distribuída ao redor de um condutor circular de raio *a*. Uma carga *q* está localizada a uma distância *x* do centro do anel. A Força exercida na carga pelo anel é dada por:

$$F = \frac{1}{4\pi\varepsilon_0} \cdot \frac{qQx}{(x^2 + a^2)^{3/2}}$$

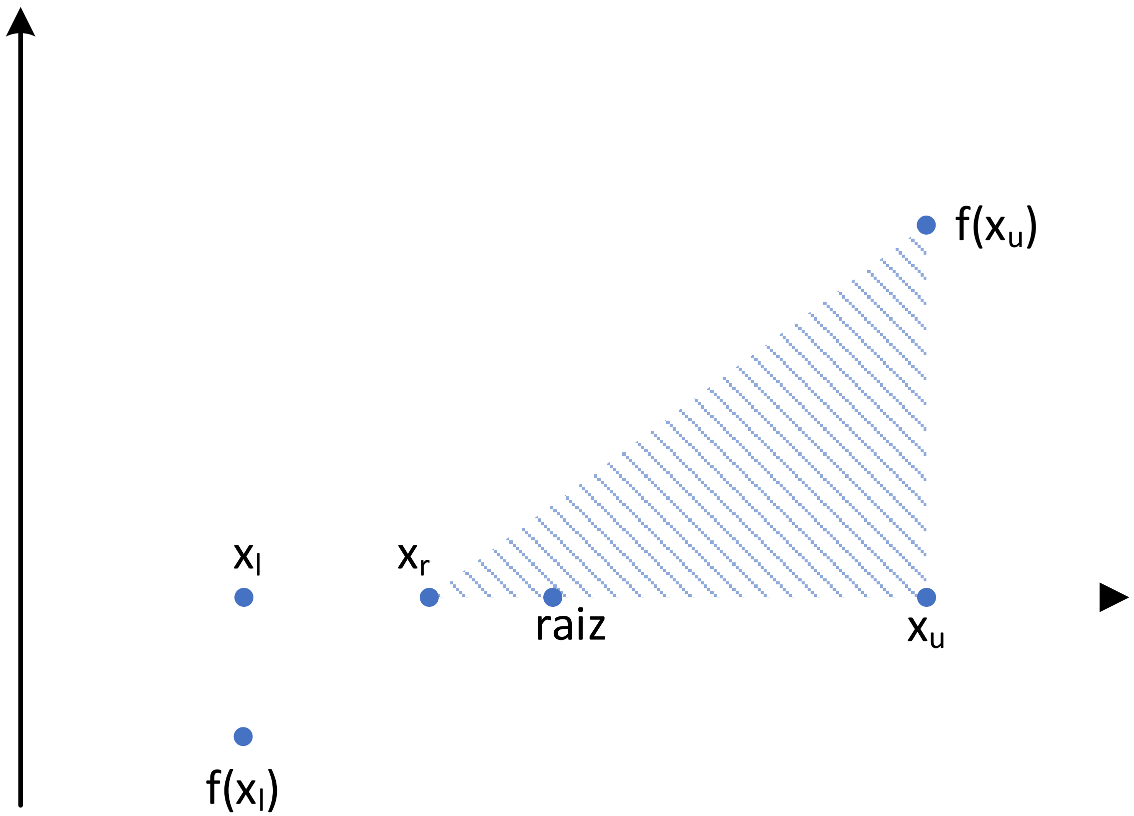


onde $\varepsilon_0 = 8{,}9 \times 10^{-12}\ C^2/Nm^2$. Encontre a distância x onde a força é 1,25 N se q e Q são 2×10^{-5} C para um anel de raio 0,85 m.

Encontrar o intervalo de busca graficamente.

Exercício 3

Exercício 4



- A partir da função Scilab do Método da Bissecção, criar uma função Scilab que implemente o método da Falsa Posição e utilizá-la para resolver o exercício 3.

$$\frac{f(x_u)-f(x_l)}{x_u-x_l} = \frac{f(x_u)}{x_u-x_r}$$

$$x_u - x_r = \frac{f(x_u)(x_u-x_l)}{f(x_u)-f(x_l)}$$

$$\boldsymbol{x_r = x_u - \frac{f(x_u)(x_u-x_l)}{f(x_u)-f(x_l)}}$$

Bibliografia e crédito das figuras

CHAPRA, Steven. **Applied numerical methods with MATHLAB for engineers and scientists.** McGrawHill, 2012.

CHAPRA, Steven e CANALE, Raymond. **Numerical methods for engineers.** McGrawHill, 2010.