



Cálculo Numérico Computacional
Derivada aproximada, raízes de funções

T. Praciano-Pereira

alun@:

Univ. Estadual Vale do Acaraú

página da disciplina

Documento processado com L^AT_EX

Lista 03

tarcisio@member.ams.org

Dep. de Computação

16 de agosto de 2009

www.calculo-numerico.sobralmatematica.org

sis. op. Debian/Gnu/Linux

0.1 Introdução

Se você quiser entregar o trabalho eletronicamente, entregue usando o meu endereço eletrônico o arquivo preferencialmente em pdf. Procure o link “textos”. Siga as instruções sobre nomes de arquivos.

cnum_seu_email_03.pdf

Data da entrega da lista: dia 24 de Agosto, segunda-feira.

0.2 Orientação

Esta lista se dedica a um problema antigo, a determinação das raízes de uma função, cuja importância reside hoje nos métodos para os quais ele, o problema, serve de motivação. Dizer que o problema é antigo não significa que seja um problema resolvido, e um dos objetivos da lista é mostrar que não sabemos resolver, ainda hoje, este problema.

O problema: Queremos encontrar um valor aproximado que resolva o problema $f(x) = 0$. Os gráficos na figura (1) mostram a idéia, geometricamente. Num dos

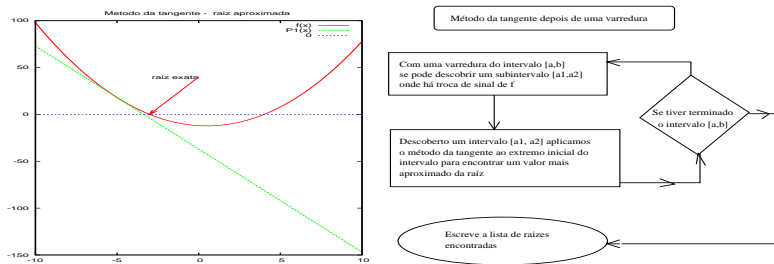


Figura 1: Raíz aproximada - métodos tangente, secante

gráficos você pode ver a reta tangente, a raiz aproximada (da reta tangente) e a raiz exata. No outro a raiz aproximada é a raiz da equação do primeiro da reta secante.

Ao longo do texto vou cometer um erro de linguagem que torna a comunicação mais simples, uma reta, uma parábola, não têm raízes, quem pode ter raiz é a equação da reta, equação da parábola, mas vou falar com frequência da raiz da reta para evitar uma frase mais longa e complicada. Na verdade não é um erro, é uma *figura literária* chamada metonímia...

A leitura básica é o segundo capítulo do meu livro (notas de aula) que se encontra na página, no link “textos”. Os programas

`raizes03.c`, `raizes_tangente.c`, `raizes_secante.c`

que se encontram no link “programas”, na página da disciplina, servem para você fazer experiências com as questões desta lista. Os programas

`exer03_03.gnuplot`, `exer03_04.gnuplot`

foram usados para elaborar questões e podem ser usados por você para entendê-las, modifique os programas e rode-os fazendo suas experiências como *material de laboratório*.

palavras chave: busca binária, derivada aproximada, métodos da tangente e da secante, raiz aproximada, recursividade.

0.3 Exercícios

1. Raíz de função

- (a) $(V)[\](F)[\]$ O teorema do valor médio nos garante que se

$$f(a) < 0 < f(b)$$

então existe um ponto $c \in [a, b]$ tal que $f(c) = 0$ e então diremos que “ c é uma raiz de f no intervalo $[a, b]$ ”.

- (b) $(V)[\](F)[\]$ O teorema do valor médio nos garante que se f for uma função contínua no intervalo $[a, b]$ e se

$$f(a) < 0 < f(b)$$

então existe um ponto $c \in [a, b]$ tal que $f(c) = 0$ e então diremos que “ c é uma raiz de f no intervalo $[a, b]$ ”.

- (c) $(V)[\](F)[\]$ Se f for uma função contínua no intervalo $[a, b]$ e se

$$f(a) < 0 < f(b)$$

então existe **um único** ponto $c \in [a, b]$ tal que $f(c) = 0$.

- (d) $(V)[\](F)[\]$ Se f for uma função contínua no intervalo $[a, b]$ e se

$$f(a) < 0 < f(b)$$

então existe um ponto $c \in [a, b]$ tal que $f(c) = 0$, mas este ponto não precisa ser único.

- (e) $(V)[](F)[]$ Se f for uma função polinomial de grau n então haverá n raízes de f em qualquer intervalo da reta, pelo Teorema Fundamental da Álgebra.
- (f) $(V)[](F)[]$ Se f for uma função polinomial de grau n então pode haver no máximo n raízes de f em um dado intervalo, pelo Teorema Fundamental da Álgebra.

2. Varredura com um programa Leia e rode o programa `raizes03.cc` que se encontra no link “programas” da página.

e selecione, em cada caso, a opção correta.

- (a) $(V)[](F)[]$ Há três funções definidas no programa, uma delas serve para ler dados numéricos pelo teclado.
- (b) $(V)[](F)[]$ Há duas funções definidas no programa, uma delas serve para ler dados pelo teclado.
- (c) $(V)[](F)[]$ Há uma função definida no programa, ela aplica a equação de um polinômio do primeiro grau.
- (d) $(V)[](F)[]$ Há uma função definida no programa, ela aplica a equação de um polinômio do segundo grau.
- (e) $(V)[](F)[]$ O programa solicita do usuário tres números e para isto usa funções definidas em alguma biblioteca.
- (f) $(V)[](F)[]$ O programa usa um intervalo predefinido no programa e aplica ao mesmo uma malha com passo predefinido.
- (g) $(V)[](F)[]$ O programa pede ao usuário que selecione um intervalo e um passo para percorrer o intervalo escolhido.

3. Varredura com programas Laboratório: o programa `raizes03.c`

está no link “programas” da página e você deve baixá-lo e rodar, ler e modificar, como laboratório para esta questão.

- (a) $(V)[](F)[]$ Num desses programas
- são lidos dois números racionais a, b (entrada de dados);
 - está definida uma função f ;
 - foi definido um *passo de malha*, na variável `delta`.
- (b) $(V)[](F)[]$ No trecho de programa abaixo, é percorrido o intervalo $[a, b]$ com passo `delta` e são impressos todos os sub-intervalos, menos o último.

```
x = a;
while(x < b)
{
cout << x << ', ' << x+delta << ']' << endl;
x = x + delta;
}
```

- (c) $(V)[](F)[]$ No trecho de programa abaixo, é percorrido o intervalo $[a, b]$ com passo `delta` e são impressos todos os sub-intervalos.

```
x = a;
while(x < b)
{
cout << x << ', ' << x+delta << ']' << endl;
x = x + delta;
}
```

- (d) $(V)[](F)[]$ No trecho de programa abaixo, é percorrido o intervalo $[a, b]$ com passo `delta` e são impressos todos os sub-intervalos, e mais um extra, externo ao intervalo $[a, b]$.

```
x = a;
while(x <= b)
{
cout << x << ', ' << x+delta << ']' << endl;
x = x + delta;
}
```

- (e) $(V)[](F)[]$ O passo *delta* é a medida dos subintervalos que tem assim, todos, a mesma medida, é uma *partição uniforme do intervalo*.
- (f) $(V)[](F)[]$ O condicional, dentro do laço, no trecho abaixo, detecta quando a função f troca de sinal em um sub-intervalo, e imprime o intervalo correspondente.

```
x = a;
while(x < b)
{
if ( f(x+delta)*f(x) <= 0 )
cout << x << ', ' << x+delta << ']' << endl;
x = x + delta;
}
```

- (g) $(V)[](F)[]$ O condicional, dentro do laço, no trecho abaixo, **pode** detectar quando a função f troca de sinal em um sub-intervalo, dependendo do passo, e então imprime o intervalo correspondente.

```
x = a;
while(x < b)
{
if ( f(x+delta)*f(x) <= 0 )
cout << x << ', ' << x+delta << ']' << endl;
x = x + delta;
}
```

- (h) $(V)[](F)[]$ O laço no trecho de programa abaixo imprime todos os sub-intervalos do intervalo $[a, b]$ onde a função f trocar de sinal

```

x = a;
while(x < b)
{
if ( f(x+delta)*f(x) <= 0 )
cout << x << ', ' << x+delta << ']' << endl;
x = x + delta;
}

```

- (i) (V)[](F)[] O laço, no trecho de programa acima, pode não detectar nenhum subintervalo em que a função f troque de sinal no intervalo $[a, b]$, mesmo que a função troque de sinal em $[a, b]$.
- (j) (V)[](F)[] Se a função trocar de sinal no intervalo $[a, b]$ é possível que o programa imprima um intervalo de comprimento δ onde esteja uma raiz da função.
- (k) (V)[](F)[] Se a função f tiver alguma raiz no intervalo $[a, b]$, **existe** um valor para o passo δ tal que o programa irá encontrar pelo menos um intervalo em que há troca de sinal e conseqüentemente uma raiz.
- (l) (V)[](F)[] Se a função não trocar de sinal, mas tiver raízes, por exemplo

$$f(x) = x^2; x \in [-3, 3]$$

se usarmos o teste " $f(x+\delta)*f(x) \leq 0$ " **pode** tanto acontecer que o programa ache um intervalo contendo uma raiz como também pode acontecer que não ache nenhum intervalo com raízes.

4. **Varredura e método da tangente** O método da tangente consiste em usar uma reta tangente ao gráfico de f próximo a um ponto onde se presume que há uma raiz e usar a raiz da reta tangente, como valor aproximado da raiz de f . A figura (1), página 1, oferece uma ilustração gráfica da idéia. O programa `exer03_03.gnuplot`, que se encontra na página, link "programas", foi usado para produzir esta questão.

- (a) (V)[](F)[] Se f for diferenciável e trocar de sinal no intervalo $[a, b]$ então a reta tangente no ponto $(a, f(a))$ terá um zero no intervalo $[a, b]$ dado por

$$y = f(a) + f'(a)(x - a) = 0; \quad (1)$$

$$x_0 = a - \frac{f(a)}{f'(a)}; \quad (2)$$

se $f'(a) \neq 0$.

- (b) (V)[](F)[] Se f for diferenciável e trocar de sinal no intervalo $[a, b]$ então a reta tangente no ponto $(a, f(a))$ poderá ter um zero no intervalo $[a, b]$ dado por

$$y = f(a) + f'(a)(x - a) = 0; \quad (3)$$

$$x_0 = a - \frac{f(a)}{f'(a)}; \quad (4)$$

se $f'(a) \neq 0$.

- (c) (V)[](F)[] As figuras produzidas pelo programa `exer03_03.gnuplot` nos conduzem a concluir que se f for diferenciável podemos obter uma aproximação do zero de f usando o zero da reta tangente no ponto $(a, f(a))$.
- (d) (V)[](F)[] As figuras produzidas pelo programa `exer03_03.gnuplot` nos conduzem a concluir que se f for diferenciável e trocar de sinal no intervalo $[a, b]$ podemos obter uma aproximação do zero de f usando o zero da reta tangente no ponto $(a, f(a))$.

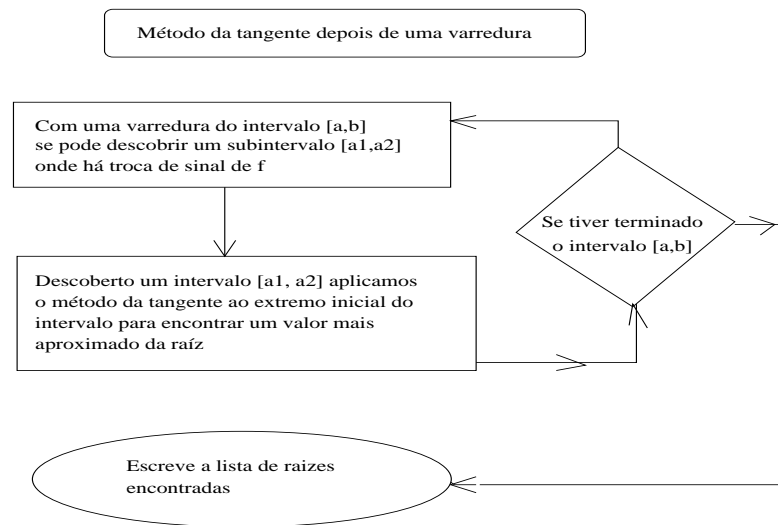


Figura 2: Fluxograma `raizes03.c`

- (e) (V)[](F)[] **Laboratório:** `exer03_03.gnuplot`

A figura (2), página 6, contém o fluxograma do programa `raizes03.c` e pode ser traduzida com o seguinte esquema:

- i. **Ponto Inicial** Percorra a malha em busca de uma troca de sinal;
 - ii. Havendo uma troca de sinal, execute a rotina `raizes_tangente()`
 - iii.
 - **Se** a malha tiver sido toda percorrida, escreva a lista das raízes encontradas,
 - **senão** retorne ao (**Ponto Inicial** 4(e)i).
- (f) (V)[](F)[] **Laboratório:** `exer03_03.gnuplot` O fluxograma da figura (2), página 6, indica que uma sucessão de retas tangentes, com seus

respectivos zeros, produz uma sucessão de números se aproximando do zero exato de f .

- (g) (V)(F) Laboratório: `exer03_03.gnuplot` O fluxograma da figura (2), página 6, indica que se f for diferenciável e trocar de sinal no intervalo $[a, b]$ uma sucessão de retas tangentes, com seus respectivos zeros, produz uma sucessão de números se aproximando do zero exato de f .