I. Erros, condicionamento e estabilidade numérica

- O conjunto $\mathbb{R}$ dos números reais é infinito, contínuo e ilimitado. O subconjunto $\mathscr{F}$ dos números que se podem representar exatamente num computador digital é finito, discreto e limitado.

- Seja $[x_- , x_+]$ um intervalo real onde x_+ é o **sucessor** de x_- em $\mathscr{F}$; um número x de $]x_- , x_+[$ é representado por x_- ou x_+ [nota: usaremos $fl(x)$ para representar o valor arredondado de x].

- $|x - fl(x)|$ erro **absoluto** devido ao arredondamento

- $\frac{|x-fl(x)|}{|x|}$ erro **relativo** devido ao arredondamento [x não nulo]

- Os erros de arredondamento podem ter efeitos catastróficos.

Erros catastróficos: exemplo

- No dia 25 de Fevereiro de 1991, durante o confito que resultou da invasão do Koweit por tropas iraquianas (Guerra do Golfo), uma bateria de misseis norte-americanos Patriot cuja missão era a de proteger uma instalação militar em Daharan, na Arábia Saudita, falhou a intercepção de um míssil Scud, lançado pelas forças militares iraquianas. O Scud atingiu um aquartelamento das tropas americanas provocando 28 mortos e um número elevado de feridos.

- A origem do problema foi a propagação de um erro de arredondamento no computador da bateria dos misseis Patriot.

- O sistema de controle do Patriot usava como unidade de tempo a décima parte do segundo; o número 0.1 não tem representação binária exata. O Patriot tinha registos de ponto fixo com 24 bits:

$$\mathbf{fl(0.1)} = \mathbf{0.00011001100110011001100}\ 1100\ldots$$

Erro de arredondamento (absoluto)

$$|\mathbf{0.1} - \boldsymbol{fl}(\mathbf{0.1})| = \mathbf{2^{-24}} + \mathbf{2^{-25}} + \mathbf{2^{-28}} \ldots \backsimeq \mathbf{9.5} * \mathbf{10^{-8}}$$

Continuação do exemplo

- É este pequeno erro $\mathbf{9.5} * \mathbf{10^{-8}}$ que se vai propagar e causar o problema. A bateria estava ligada há cerca de 100 horas.

- Unidades (em décimos de segundo) de tempo: 3 600 000 (=100 *60 *60 *10)

- 3 600 000 * $\mathbf{9.5} * \mathbf{10^{-8}}$ segundos $\backsimeq$ 0.34 segundos

- O Scud tinha uma velocidade de 1.676 metros por segundo e portanto percorria mais de 500 metros num intervalo de tempo de 0.34 segundos. Este facto fez com que o Patriot falhasse a interceção do míssil iraquiano.

Mais exemplos de erros de arredondamento

- Em aritmética exacta, as condições **x > 0** e 1 + **x > 1** são equivalentes. Verifica que no Matlab

  ```
  >> 2^-53 > 0
  ```

  dá o valor lógico 1 (a proposição é verdadeira) mas

  ```
  >> 1 + 2^-53 > 1
  ```

  dá o valor lógico 0 (a proposição é falsa).

Mais exemplos de erros de arredondamento

- O Matlab não consegue calcular o valor exato da soma de 10 parcelas todas iguais a 0.1

>> x = 0.1 *ones(10, 1)

[vector com 10 entradas todas iguais 0.1; em geral, >> ones(m; n) produz uma matriz com m linhas e n colunas e entradas todas iguais à unidade].

>> sum(x)   % soma todas as entrdas do vetor x

produz o resultado **1.0000 c**om um erro que é dado por

>> ans-1

aproximadamente igual a **-1.1102e-16**

Mais exemplos de erros de arredondamento

- A adição e multiplicação de números reais são associativas e a multiplicação é distributiva em relação à adição mas, no Matlab, com

  `>> x = 0.1; y = 0.3; z = 0.7;`

  as condições

  `(x + z) + y == x + (z + y)`

  `(x *y) *z == x *(y *z)`

  `x *(y + z) == (x *y) + (x *z)`

  produzem no Matlab o valor lógico 0 (falso). Verifica.

Cálculo numérico *versus* cálculo simbólico

- *Um sistema de cálculo algébrico (simbólico) representa os números racionais na forma de um* quociente de dois inteiros e opera com eles usando as regras aritméticas apropriadas. O preço que se paga por isto é que o sistema usa mais memória para a representação dos números e a aritmética é "mais pesada".
- Num sistema de cálculo simbólico obtém-se

$$\sum_{n=1}^{100} \frac{1}{n} = \frac{14\,466\,636\,279\,520\,351\,160\,221\,518\,043\,104\,131\,447\,711}{2788\,815\,009\,188\,499\,086\,581\,352\,357\,412\,492\,142\,272}$$

e a soma dos inversos aritméticos dos 200 primeiros números inteiros positivos produz um numerador e denominador com 90 algarismos.

- Cálculo numérico e cálculo simbólico são ferramentas, ambas com vantagens e inconvenientes, que se complementam. O cálculo numérico continua a ser mais usado na computação científica mas existem códigos híbridos [numérico+simbólico].

Cálculo numérico *versus* cálculo algébrico

No Matlab existe a Toolbox de cálculo simbólico.

Exemplo 1

```
>> sym(1/2)+sym(1/3)
 ans = 5/6
```

Exemplo 2

```
>> x=sym(1./(1:100))
x = [1, 1/2, 1/3, 1/4, 1/5, 1/6,…, 1/100]

>> sum(x)
ans=14466636279520351160221518043104131447711/2788815009188499086581352357412492142
272

>> double(ans)
ans = 5.187377517639621

>> sum(1./(1:100))
ans = 5.187377517639621
```

O ponto flutuante

- Na representação de números em **ponto flutuante**, um número x representa-se por

$$x = \pm \mathrm{m} * \beta^{e} \qquad (1)$$

onde *m* é a mantissa e *e* (inteiro positivo ou negativo) é o expoente.

- Valores típicos de β são 2 (sistema binário), 8 (sistema octal), 10 (decimal) e 16 (hexadecimal).

- Para o mesmo número *x* há muitas representações da forma (1). Por exemplo,

$$x = 0.2335 = 0.02335 * 10^{1} = 23.35 * 10^{-2} = \cdots$$

Obviamente, para adicionar dois números há que garantir que as representações têm o mesmo expoente e, neste caso, adicionam-se as respetivas mantissas.

- Exemplos:

$(1.451 * 10^{0}) + (2.70145 * 10^{-3}) = (1.451 * 10^{0}) + (0.00270145 * 10^{0})$=$1.45370145 * 10^{0}$

$(1.00101 * 2^{0}) + (1.1001 * 2^{-4}) = (1.00101 * 2^{0}) + (0.00011001 * 2^{0})$=$1.01000001 * 2^{0}$

A norma IEEE754

- Os fabricantes de computadores têm adotado sistemas que diferem em muitos aspetos (base do sistema, número de dígitos na mantissa e no expoente, regras de arredondamento, etc.)
- Para os mesmos cálculos, diferentes computadores (ou máquinas de calcular) podem produzir resultados que não são exatamente iguais (podem até ser muito diferentes).
- No passado foi feito um esforço de uniformização que culminou com a publicação em 1985, da chamada "norma IEEE 754"para o sistema binário cujas especificidades apresentamos resumidamente.

formato simples (32 bits):

sinal (1 bit), expoente (8 bits), mantissa (23 bits)

formato duplo (64 bits):

sinal (1 bit), expoente (11 bits), mantissa (52 bits)

A norma IEEE 754

- Representação normalizada: o primeiro bit da mantissa é igual a 1

$$\pm(\mathbf{1}.b_{-1}b_{-2}\ldots b_{-52})*\ 2^{e}$$

- Exemplo

$$(0.1)_{10}=(0.00011001100110011001100\ 1100\ldots\ )_2$$

tem a representação normalizada (com 32 bits)

$$(\mathbf{1.10011001100110011001100}\ )*\ \mathbf{2^{-4}}$$

O bit à esquerda do ponto é o **bit implícito** (perfaz o total de 24 bits na mantissa).

A norma IEEE754 (sobre o **eps**ilon)

No formato duplo, o sucessor do número 1

$$1=(\mathbf{1}.0\ldots00) * 2^0$$

(todos os bits iguais a 0, exceto o bit implícito) é o número que tem a representação

$$(\mathbf{1}.0\ldots01) * 2^0 = 1 + 2^{-52}$$

Esta distância do número 1 ao seu sucessor é de grande importância, como veremos mais adiante. É uma das constantes definidas no Matlab

```
>> eps
ans =
   2.2204e-16

>> eps==2^-52

ans = logical  1
```

Percorrendo $\mathscr{F}$

- Se o sucessor de 1 é 1+eps, também o sucessor de 1+eps é 1+2*eps. Afinal, em geral, para obter o sucessor de um número de positivo de $\mathscr{F}$ adiciona-se uma unidade na última posição da mantissa.
- Será que cada número dista do seu sucessor esta quantidade eps?
- Isso só é verdade para os números com expoente *e=0*, isto é, números de $\mathscr{F}$ entre 1 e 2. Por exemplo, o successor de

$$2=(\mathbf{1}.0\ldots00) * 2^1$$

(nota: observe-se que todas as potências de 2 têm a mesma mantissa)

é $$(\mathbf{1}.0\ldots01) * 2^1 = (1+\mathrm{eps}) * 2 = 2 + 2\mathrm{eps}$$

- A distância entre um número x_- e o seu successor x_+ depende do expoente (*e*) de x_-. Se

$$x_- = \pm(\mathbf{1}.b_{-1}b_{-2}\ldots b_{-52}) * 2^e$$

é $$|x_+ - x_-| = 2^{e-52}$$

Majoração do erro absoluto de arredondamento

- Se $x_{-} < x < x_{+}$

resulta da expressão anterior que

$$|x - fl(x)| < 2^{e-52} \qquad (2)$$

- 2^{e-52} é o majorante do erro absoluto devido ao arredondamento
- Para números grandes este erro absoluto pode ser grande;

Exemplo: se x tem expoente *e=53,* de (2) resulta

$$|x - fl(x)| < 2$$

o que mostra que o erro absoluto pode ser quase igual a 2 (e ainda maior do que isto para expoentes maiores).

[nota: este exemplo também mostra que há muitos números inteiros que não pertencem a $\mathscr{F}$]

Majoração do erro relativo de arredondamento

- Com $x = \pm(\mathbf{1}.b_{-1}b_{-2}\ldots b_{-52}) * 2^{e}$

  resulta de (2) que $\frac{|x - fl(x)|}{|x|} < \frac{2^{e-52}}{|x|} < 2^{-52}$ (por ser $|x| > 2^{e}$)

- Tem-se

$$\frac{|x - fl(x)|}{|x|} < \text{eps} \qquad (3)$$

isto é, o erro relativo é majorado por eps.

- Ao contrário do erro absoluto, o erro relativo não depende da grandeza do número que é arredondado. Por outras palavras, o sistema de ponto flutuante representa números grandes e pequenos com a mesma precisão.

A norma IEEE754 (expoentes)

- No formato simples, os oito bits reservados para o expoente permitem obter

$$\mathbf{2^8} = 256$$

números positivos diferentes, desde 0 até 255.

- E para representar expoentes negativos (para números inferiores à unidade, em valor absoluto)?

- Não se usa um bit reservado para o sinal do expoente.

- *bias exponent* - expoente enviesado: para obter o verdadeiro expoente, o hardware subtrai 127.

- As representações 00000000 e 11111111 são usadas para situações especiais, número desnormalizado e overflow, resp.

- Formato simples $\mathbf{-126 \leq e\ (inteiro) \leq 127}$

- Formato duplo $\mathbf{-1022 \leq e\ (\mathit{inteiro}) \leq 1023}$

O maior número representável na norma IEEE 754

$$(\mathbf{1.1111111111111111111111111111111111111111111111111111})* \mathbf{2^{1023}}$$

$$= \quad \mathbf{2^{1023}} + \mathbf{2^{1022}} + \ldots + \mathbf{2^{971}}$$

```
>> realmax
ans =

  1.7977e+308
```

```
>> x = 0; for k = 971 : 1023; x = x + 2^k; end; x == realmax

ans =

     1
```

Overflow

Uma expressão para calcular realmax

- $\boldsymbol{m} = \quad \mathbf{1} + \mathbf{2^{-1}} + \mathbf{2^{-2}} + \ldots + \mathbf{2^{-51}} + \mathbf{2^{-52}}$

  2m = 2+1$+ \mathbf{2^{-1}}$ **+ ... +** $\mathbf{2^{-51}}$

  m = 2-$\mathbf{2^{-52}}$
- realmax=$(\mathbf{2} - \mathbf{2^{-52}})$*$\mathbf{2^{1023}}$=$\mathbf{2^{1024}}$-$\mathbf{2^{971}}$

```
>> 2^1024-2^971==realmax
ans = logical 0
```

```
>> (2-2^-52)*2^1023==realmax
ans =  logical  1
```

```
 >> 2^1024
ans = Inf
```

O realmin

O menor número normalizado é $(\mathbf{1.0 \ldots 0}\ )*\mathbf{2^{-1022}}$

```
 >> 2^-1022==realmin

ans = logical 1



>> realmin

ans =

  2.2251e-308
```

Underflow gradual

O Matlab representa números cujo valor absoluto é menor do que realmin

>> 2^-1023/2^-1024

ans =2

$2^{-1023} = (0.10 \ldots 00) * 2^{-1022}$

$2^{-1024} = (0.010 \ldots 00) * 2^{-1022}$

…

$2^{-1074} = (0.000 \ldots 01) * 2^{-1022}$

>> 2^-1075

ans =0

Nota: números menores do que realmin são representados com menor precisão (são armazenados menos bits significativos)

Arredondamento para o mais próximo

Para o mais próximo (‘default’ no Matlab)

x=1; x+eps/3 ==x

ans = logical 1

>> x=1/2; x+eps/3==x

ans = logical 0

>> x=1; x+eps/2==x

ans = logical 1

Quatro modos de arredondamento

Para a direita

>> system_dependent('setround',‘Inf')

Para a esquerda

>> system_dependent('setround',‘-Inf')

Na direção de zero (corte)

>> system_dependent('setround',‘0')

Para o mais próximo (´default´)

>> system_dependent('setround','nearest')

Soma com diferentes arredondamentos (1)

```
% esta script calcula o valor da soma de n números gerados
% aleatoriamente entre -1 e 1, usando os quatro modos de
% arredondamento previstos na norma IEEE

x=2*(rand(n,1)-0.5);
% rand(n,1) gera um vetor coluna com n entradas entre 0 e 1
system_dependent('setround',-Inf)
sEsq=sum(x)
% valor da soma calculada com arredondamento para -Inf
system_dependent('setround',0)
sZero=sum(x)
% valor da soma calculada com arredondamento para 0
system_dependent('setround',0.5)
sPro=sum(x)
% valor da soma calculada com arredondamento para o mais próximo
system_dependent('setround',Inf)
sDir=sum(x)
% valor da soma calculada com arredondamento para +Inf
```

Soma com diferentes arredondamentos (2)

\>> **n=1000; format long; quatro_somas**

O resultado exato está no intervalo **[sEsq, sDir]**

```
>> format long, n=100; quatro_somas
...

>> sDir-sEsq

ans =

   2.131628207280301e-14
```

Quando se usa o arredondamento para o mais próximo, o erro cresce com o número n de parcelas mas de forma moderada

Algarismos significativos

Nas representações seguintes o número π é aproximado com o mesmo número de algarismos significativos (neste caso, com 3 algarismos significativos)

$$3.14 \ * 10^{0}$$

$$314 \ * 10^{-2}$$

$$0.00314 \ * 10^{3}$$

Enfatiza-se que na última representação <u>os zeros à direita do ponto decimal não são algarismos significativos</u>.

- Na representação normalizada (norma IEEE 754)

$$\pm(\mathbf{1}.b_{-1}b_{-2} \dots b_{-52})* \ 2^{e}$$

os números têm todos 53 bits significativos independentemente da sua grandeza (expressa pelo expoente e)

Algarismos significativos corretos

Se

$$\bar{x} = (0.d_{-1}d_{-2}\dots d_{-t})*10^{e}, \quad d_{-1} \in \{1,2,\dots,9\}$$

(mantissa com t algarismos) então $|x-\bar{x}| < 10^{e-t}$, qualquer que seja o modo de arredondamento. No **arredondamento para o mais próximo**

$$|x-\bar{x}| \le \frac{10^{e-t}}{2}$$

e o majorante para o erro relativo é, por ser $|\boldsymbol{x}| \ge 10^{e-1}$,

$$\frac{|\boldsymbol{x}-\bar{x}|}{|\boldsymbol{x}|} \le \frac{\mathbf{1}}{\mathbf{2}}\frac{10^{e-t}}{10^{e-1}} = \frac{\mathbf{1}}{\mathbf{2}}10^{-t+1} = 5*10^{-t}$$

Dizemos neste caso que $\bar{x}$ aproxima x com t algarismos significativos corretos

Exemplo no Matlab

$$\boldsymbol{\pi} = \mathbf{3.1415926535} \ldots$$

Com **t** algarismos significativos corretos:

- t=3, $\bar{\boldsymbol{\pi}} = \mathbf{3.14}$

> abs((pi-3.14)/pi) dá 5.0696e-04 (< 5*10^-3)

- t=4, $\bar{\boldsymbol{\pi}} = \mathbf{3.142}$ (e não $\mathbf{3.141}$)

>> abs((pi-3.142)/pi) dá 1.2966e-04 (<5*10^-4)

- t=5, $\bar{\boldsymbol{\pi}} = \mathbf{3.1416}$ (e não $\mathbf{3.1415}$)

>> abs((pi-3.1416)/pi) dá 2.3384e-06 (<5*10^-5)

O cancelamento subtrativo (1)

Perda de algarismos significativos na subtração de números

Exemplo

Sejam $\bar{\boldsymbol{x}} = \mathbf{1.43275}$ e $\bar{\boldsymbol{y}} = \mathbf{1.43264}$ aproximações de x e y, ambas com seis algarismos corretos.

Neste caso, $\bar{\boldsymbol{x}} - \bar{\boldsymbol{y}} = \mathbf{0.00011}$ representa o valor exato de $\mathbf{x} - \boldsymbol{y}$ com apenas dois algarismos corretos

A mesma perda de algarismos ocorre se os números tiverem outros expoentes, por exemplo com

$\bar{\boldsymbol{x}} = \mathbf{1.43275{*}10^7}$

$\bar{\boldsymbol{y}} = \mathbf{1.43264{*}10^7}$

e

$$\bar{\boldsymbol{x}} - \bar{\boldsymbol{y}} = \mathbf{0.00011{*}10^7}$$

Observe-se que $\bar{x}$ e $\bar{y}$ são $O(10^7)$ e $\bar{x} - \bar{y}$ é $O(10^3)$, há perda de 4 algarismos significativos

O cancelamento subtrativo (2)

Exemplo no Matlab

A fórmula fundamental da trigonometria é $sin(x)^2 + cos(x)^2 = 1$, mas…

>> x=1e-5; 1-cos(x)^2, sin(x)^2
ans =   1.000000082740371e-10
ans =   9.999999999666671e-11

Qual destes números tem mais algarismos significativos corretos?
É o calculado por sin(x)^2, porque ocorre cancelamento subtrativo na diferença 1-cos(x)^2.

>> cos(1e-5)^2
ans =   0.999999999900000

>> 1-cos(1e-5)^2
ans =   1.000000082740371e-10

O erro relativo no cancelamento subtrativo

>> x=1e-5; abs(1-cos(x)^2-sin(x)^2)
ans = 3.247895369989818e-16

O erro absoluto é pequeno mas o erro relativo é muito maior

> ans/sin(x)^2

ans = 3.247895370098080e-06

Quanto algarismos significativos corretos tem o valor calculado de (1-cos(x)^2)?

Propagação do erro relativo

Erros relativos grandes podem gerar erros absolutos também grandes

```
>> x=1e-5; (1-cos(x)^2)/(sin(x)^2)

ans =   0.999996752104630
```

O resultado exato é 1, o erro absoluto é

```
>> abs(1-ans)

ans = 3.247895370095399e-06
```

Condicionamento de uma função

O **número de condição de uma função num ponto $\boldsymbol{x}$** mede a variação do valor $f(x)$ provocadas por pequenas alterações (perturbações) no valor do argumento x.

Exemplo

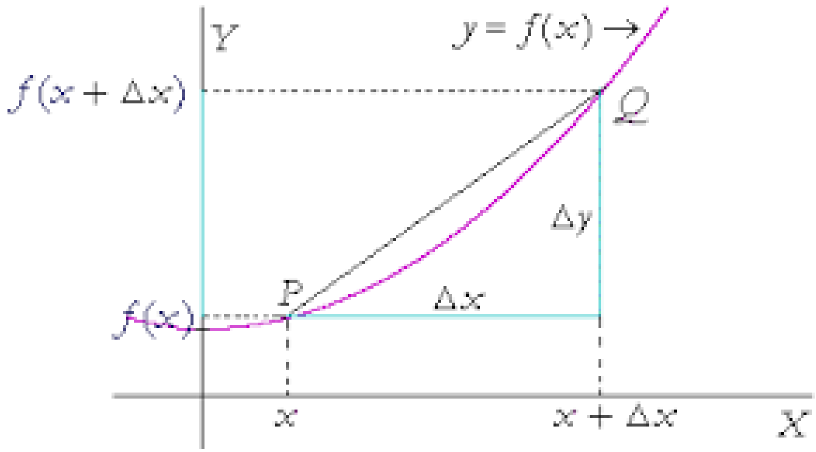
```
>> x=10; deltaX=1e-5; exp(x+deltaX)-exp(x)

ans =

   0.2203
```

Uma perturbação de $\Delta x = 10^{-5}$ no valor do argumento provocou um erro (absoluto) aproximadamente igual a $2 * 10^{-1}$, bastante maior.

O número de condição absoluto de uma função num ponto



$\Delta y = |f(x + \Delta x) - f(x)| \approx |f'(x)|.\, \Delta x$

$|f'(x)|$ é o número de condição absoluto de *f* no ponto *x*

$$f(x + \Delta x) = f(x) + f'(x).\, \Delta x + \frac{f''(x)}{2}(\Delta x)^2 + \ldots + \frac{f^{(k)}(x)}{k!}(\Delta x)^k + \cdots$$

A soma dos termos que se desprezam é o erro de TRUNCATURA

O número de condição relativo de uma função num ponto

De

$\Delta y = |f(x+\Delta x) - f(x)| \approx |f'(x)|.\, \Delta x$

$|f'(x)|$ é o número de condição absoluto de *f* no ponto *x*

com $x \neq 0$ e $f(x) \neq 0$, resulta

$\left|\frac{\Delta y}{y}\right| = \left|\frac{f(x+\Delta x)-f(x)|}{f(x)}\right| \approx \left|\frac{\boldsymbol{x.f'(x)}}{\boldsymbol{f\,(x)}}\right|.\, \left|\frac{\Delta x}{x}\right|$

$\left|\frac{x.f'(x)}{f\,(x)}\right|$ **é o número de condição relativo de *f* no ponto *x***

Exemplo: $f(x)=e^x, x = 10$, $\Delta x = 10^{-5}$

$\Delta y \approx |\boldsymbol{f'(x)}|.\, \Delta x = \boldsymbol{e^{10}} * 10^{-5} = 0.2203 \ldots$

$\left|\frac{\Delta y}{y}\right| \approx |\boldsymbol{x}|.\left|\frac{\Delta x}{x}\right| = \mathbf{10} * \ 10^{-6}$

Números de condição relativos e perda de algarimos significativos corretos

Se o número de condição relativo é da ordem de grandeza de 10^k, então há perda de k algarismos decimais no valor da função.

<u>Exemplo 1</u>

format long, x=1; deltaX=1e-5; x+deltaX, exp(x), exp(x+deltaX)

ans =

1.000010000000000 x+deltaX aproxima x com 5 algarimos corretos

ans =

2.718281828459046 número de condição relativo é |x|=1

ans =

2.718309011413245 exp(x+deltaX) aproxima exp(x) com 5 algarimos corretos

<u>Exemplo 2</u>

x=100; deltaX=1e-3; x+deltaX, exp(x), exp(x+deltaX)

ans =

1.000010000000000e+02 x+deltaX aproxima x com 5 algarimos corretos

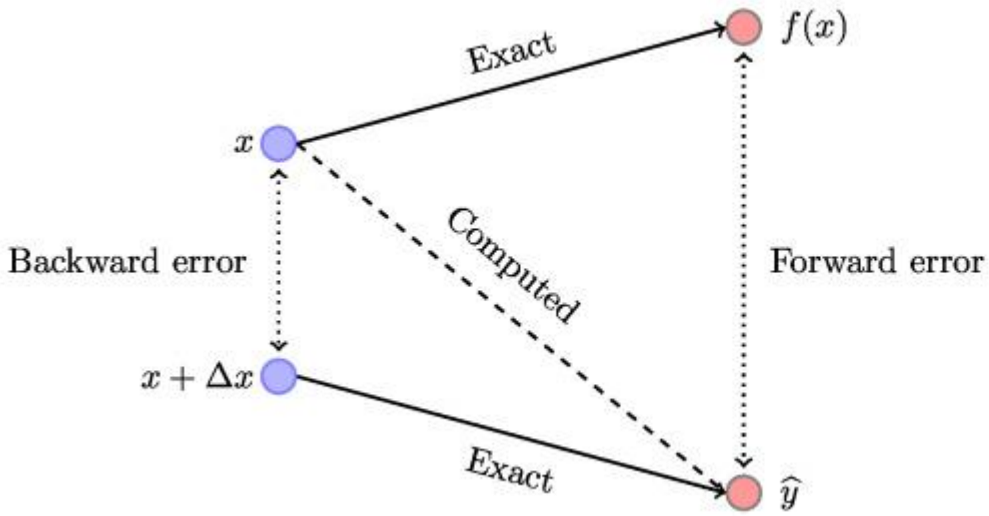
ans =

2.688117141816136e+43 número de condição é |x|=100

ans =

2.690806603464667e+43 exp(x+deltaX) aproxima exp(x) com 3 algarimos corretos

Erro direto (*forward*) e erro inverso (*backward*)



- Num algoritmo numérico (representado por f), devido aos erros de arredondamento, o valor calculado para os dados x não é o valor exato $y = f(x)$ mas uma aproximação $\hat{y}$.
- O erro direto é $\nabla y = f(x)$-$\hat{y}$.
- $\mathrm{x} + \nabla x$ são os dados perturbados a que corresponde em aritmética exata o valor calculado $\hat{y}$.
- ∇x é o erro inverso

Instabilidade numérica *versus* condicionamento

Um algoritmo (ou simplesmente uma expressão numérica) diz-se numericamente estável (backwards stable) quando $|\nabla x|$ é pequeno, isto é, quando o valor calculado $\hat{y}$ corresponde em aritmética exata a pequenas perturbações nos dados.

Se o problema for mal condicionado, $|\nabla y|$ será maior do que $|\nabla x|$.

No exemplo seguinte, a função é bem condicionada mas uma das expressões é numericamente instável porque ocorre cancelamento subtrativo.

Exemplo

```
>> x=10^12; (x+1)-sqrt(x)

    ans = 5.000038072466850e-07

>> 1/(sqrt(x+1)+sqrt(x))

    ans =4.999999999998749e-07
```

Séries de Taylor (1)

Desenvolvimento em série de potências de x de uma função f com derivadas contínuas

$$f(x) = f(0) + f'(0)\,x + \frac{f''(0)}{2}x^2 + \frac{f'''(0)}{3!}x^3 + \ldots + \frac{f^{(k)}(0)}{k!}x^k + \ldots$$

Exemplos

$$sin(x) = sin(0) + cos(0)\,x - \frac{sin(0)}{2}x^2 - \frac{cos(0)}{3!}x^3 + \frac{sin(0)}{4!}x^4 + \frac{cos(0)}{5!}x^5 + \ldots\ldots$$

$$= x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots$$

$$cos(x) = cos(0) - sin(0)\,x - \frac{cos(0)}{2}x^2 + \frac{sin(0)}{3!}x^3 + \frac{cos(0)}{4!}x^4 - \frac{sin(0)}{5!}x^5 + \ldots\ldots$$

$$= 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \ldots$$

$$\exp(x) = exp(0) + \exp(0)\,x + \frac{\exp(0)}{2}x^2 + \frac{exp(0)}{3!}x^3 + \frac{exp(0)}{4!}x^4 + \frac{exp(0)}{5!}x^5 + \ldots\ldots$$

$$= 1 + x + \frac{x^2}{2} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} + \ldots$$

Séries de Taylor (2)

Desenvolvimento em série de potências de *x-a* de uma função f com derivadas contínuas

$$f(x) = f(a) + f'(a)\,(x-a) + \frac{f''(a)}{2}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \ldots + \frac{f^{(k)}(a)}{k!}(x-a)^k + \ldots$$

Exemplo 1 ($f(x) = \sin(x)$, $a = \pi/2$)

$$sin(x) = sin(\pi/2) + cos(\pi/2)\,(x-\pi/2) - \frac{sin(\pi/2)}{2}(x-\pi/2)^2 - \frac{cos(\pi/2)}{3!}(x-\pi/2)^3 + \frac{sin(\pi/2)}{4!}(x-\pi/2)^4 + \ldots$$

$$= 1 - \frac{(x-\pi/2)^2}{2!} + \frac{(x-\pi/2)^4}{4!} - \frac{(x-\pi/2)^6}{6!} \ldots$$

Exemplo 2 ($f(x) = \log(x)$, $a = 1$)

Para o logaritmo natural (de base e) $f(x) = \log(x)$, tem-se $f'(x) = x^{-1}, f''(x) = -x^{-2}$, $f'''(x) = 2x^{-3}$, $f^{(iv)}(x) = -3 * 2x^{-4}$, $f^{(v)}(x) = 4 * 3 * 2x^{-5}, \ldots f^{(k)}(1) = (-1)^{k+1}(\mathrm{k}-1)!$

$\log(x) = (x-1) - \frac{(x-1)^2}{2} + \frac{(x-1)^3}{3} + \cdots + (-1)^{k+1}\frac{(x-1)^k}{k} + \cdots$ válida para $0 < x \leq 2$

$\log(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} + \cdots + (-1)^{k+1}\frac{x^k}{k} + \cdots$ válida para $-1 < x \leq 1$

Majoração dos erros de truncatura (1)

$$f(x) = f(a) + f'(a)\,(x-a) + \frac{f''(a)}{2}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \ldots + \frac{f^{(k)}(a)}{k!}(x-a)^k + R_k(x)$$
$$= T_k(x) + R_k(x)$$

$T_k(x)$ aproxima o valor de $f(x)$ com erro de truncatura $R_k(x)$.

Resto na forma de Lagrange

$$R_k(x) = \frac{f^{(k+1)}(\theta)}{(k+1)!}(x-a)^{k+1}$$

θ é um ponto que está entre a e x.

Se $|f^{(k+1)}(\theta)| \leq M$, para qualquer θ entre a e x, então

$$|R_k(x)| \leq \frac{M}{(k+1)!}|x-a|^{k+1}$$

Majoração dos erros de truncatura (2)

Exemplo $\mathrm{f(x)} = \sin(x),\ a = 0$:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} + R_6$$

com $R_6(x) = \frac{f^{(7)}(\theta)}{7!}x^7 = \frac{-cos\ (\theta)}{7!}x^7$

Para $x = \frac{\pi}{6}$, no Matlab

```
>> x=pi/6; T6=x-x^3/factorial(3)+x^5/factorial(5)
T6 =   0.500002132588792
```

e $|R_6\left(\frac{\pi}{6}\right)| \leq \frac{1}{7!}\left(\frac{\pi}{6}\right)^7$ por ser $|cos\ (\theta)| \leq 1$ em $[0, \frac{\pi}{6}]$

```
>> (pi/6)^7/factorial(7)

ans =2.140719769235796e-06
```

Nota: o valor exato é 0.5, o erro é 2.132588792e-06

Majoração dos erros de truncatura (3)

Numa série alternada, o erro de truncatura é inferior ao valor absoluto do primeiro termo que se despreza

$\exp(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} + \ldots$

,

com $x = -1$,

$T_4(-1) = 1 - 1 + \frac{1}{2} - \frac{1}{3!} + \frac{1}{4!}$ aproxima o valor de exp(-1) com erro de truncatura inferior a $\frac{1}{5!}$

```
>> T4=1/2-1/factorial(3)+1/factorial(4)
T4 =0.375000000000000

>> abs(exp(-1)-T4)
ans =0.007120558828558

>>1/factorial(5)
ans =0.008333333333333
```

II. Resolução de equações não-lineares

Determinar x tal que

$$f(x) = 0$$

(raízes da equação ou zeros de f)

Fórmula resolvente da equação polinomial de segundo grau $ax^2 + bx + c = 0$

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

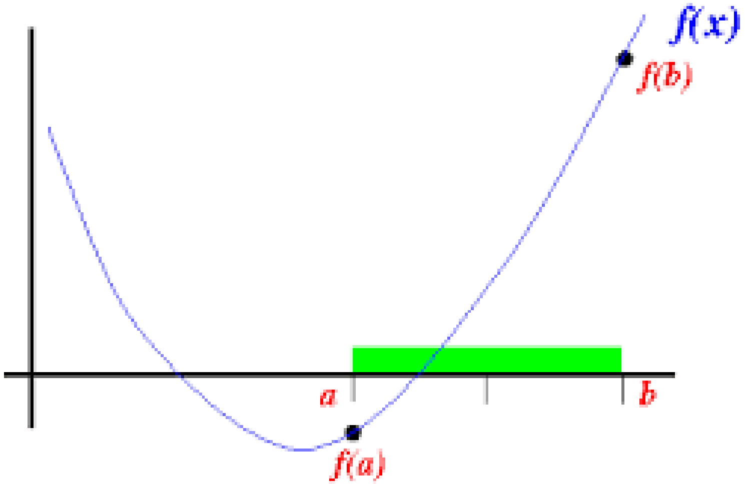
Como calcular as raízes da equação $x^5 - x^4 + 2x^3 - 3x^2 - 5x + 6 = 0$?

No Matlab

```
>> roots([1,-1,2,-3,-5,6])
ans =
  -0.0943 + 1.9135i
  -0.0943 - 1.9135i
  -1.1878 + 0.0000i
   1.3763 + 0.0000i
   1.0000 + 0.0000i
```

A importância da continuidade da função f

Se f é contínua $[a, b]$ e $f(a).f(b) < 0$, então existe pelo menos uma raiz r da equação $f(x) = 0$ entre a e b.



Método da bisseção

f é contínua $[a, b]$ e $f(a).f(b) < 0$

Em cada iteração, divide-se em duas partes iguais e escolhe-se o subintervalo em que f muda de sinal:

Calcula-se o valor de f no ponto médio $m = (a + b)/2$

Se $\mathrm{f}(m) = 0$ então $\mathrm{r} \leftarrow m$

Se $\mathrm{f}(m) * f(a) < 0$ (a raiz está entre a e m) faz-se $b \leftarrow m$.

Se $\mathrm{f}(m) * f(a) > 0$ (a raiz está entre m e b) faz-se $\mathrm{a} \leftarrow m$.

(ver figura 2.2 na p.42 do livro)

Método da bisseção. Exemplo

(ver Exemplo 2.1 do livro) No início de cada ano o cliente de um banco deposita v euros num fundo de investimento e retira ao fim do n-ésimo ano um capital de M euros. Queremos calcular a taxa de juro anual r deste investimento.

Dado que

$$M = v\sum_{k=1}^{n}(1+r)^k = v\frac{1+r}{r}[(1+r)^n - 1],$$

r é uma raiz da equação $\mathrm{f(x)} = 0$, onde

$$\mathrm{f(x)} = M - v\frac{1+x}{x}[(1+x)^n - 1].$$

Consideremos que o investidor deposita anualmente $v = 1000$ e que depois de 5 anos recebe o capital $\mathrm{M} = 6000$ euros. Qual a taxa de juro anual que lhe pagou o banco?

>> f=@(x) 6000-1000*(1+x)/x*((1+x)^5-1); fplot(f,[0.01,0.3])

Método da bisseção. Exemplo (continuação)

Vemos que f tem um zero entre $a = 0.01$ e $b = 0.3$.

No Matlab,

>> a=0.01; b=0.3; f(a), f(b)

ans =   847.9849

ans = -5.7560e+03

Primeira iteração

>> m=(a+b)/2, f(m)

m =0.1550

ans =-1.8649e+03

Porque f(a) e f(m) têm sinais contrários, a raiz está entre a=0.01 e m=0.155. Fazemos b=m e continuamos a iterar com o novo intervalo [a,b]=[0.01,0.155] cuja amplitude é metade da amplitude do intervalo inicial [0.01,0.3]

A convergência do método da bisseção (1)

Ao fim de k iterações, temos o intervalo $[a^{(k)}, b^{(k)}]$ tal que

$$b^{(k)} - a^{(k)} = \frac{b-a}{2^k}$$

O erro na aproximação

$$x^{(k)} = (a^{(k)} + b^{(k)})/2$$

é $\quad |e^{(k)}| = |x^{(k)} - r| < \frac{b-a}{2^{k+1}}$

Para garantir que $|e^{(k)}| <$tol, basta fazer k_{min} iterações onde k_{min} é o menor inteiro positivo que satisfaz a desigualdade

$$k_{min} > \log_2\left(\frac{b-a}{\mathrm{tol}}\right) - 1$$

c.a.: $\frac{b-a}{2^{k+1}} < \mathrm{tol} \Leftrightarrow \frac{b-a}{\mathrm{tol}} < 2^{k+1} \Leftrightarrow \log_2\left(\frac{b-a}{\mathrm{tol}}\right) < k+1$

A convergência do método da bisseção (2)

Voltando ao problema anterior da taxa de juro, com $a = 0.01, b = 0.3$ e $tol = 1e - 5$, de

```
>> log2((b-a)/1e-5)-1
ans =
  13.8238
```

concluímos que k_{min}=14

nota: se efetuarmos 14 iterações obtemos a aproximação

$$x^{(14)}=0.0614\ldots$$

que corresponde a uma taxa de juro anual de 6,14%

Vantagens e desvantagens do método da bisseção

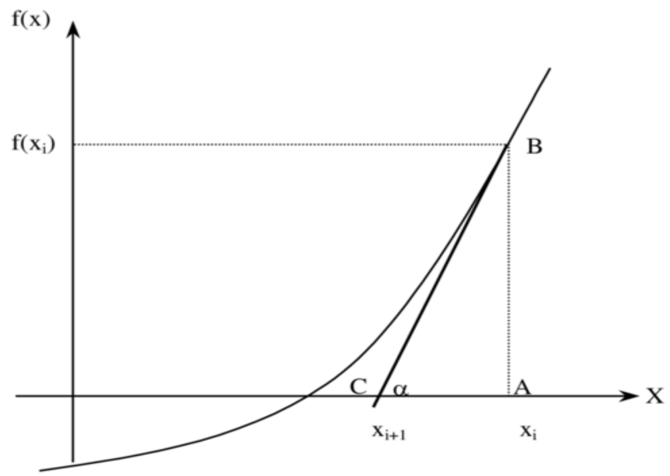
VANTAGENS:

- Convergência garantida
- A função f não precisa de ser derivável (basta que seja contínua)
- Cada iteração requer apenas o cálculo da função num ponto (o ponto médio)
- Excelente estabilidade numérica (erros no cálculo de f não afetam o resultado desde que o sinal do valor calculado esteja correto)

DESVANTAGEM (única):

- Convergência lenta (linear); cada iteração acrescenta apenas mais um bit correto à aproximação

O método de Newton-Raphson (das tangentes)



$$\tan(\alpha) = \frac{AB}{AC}$$

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

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

$x^{(i+1)}$ é a abcissa do ponto em que a reta tangente à curva no ponto de abcissa $x^{(i)}$ interseta o eixo dos xx

https://upload.wikimedia.org/wikipedia/commons/e/e0/NewtonIteration_Ani.gif

A evolução do erro de truncatura

Do desenvolvimento em série de Taylor na p. 42, com $x = r$ e $\mathrm{a} = x^{(i)}$):

$$f(r) = f\left(x^{(i)}\right) + f'\left(x^{(i)}\right)\left(r - x^{(i)}\right) + \frac{f''(\theta)}{2}\left(r - x^{(i)}\right)^2$$

e assumindo que $f'\left(x^{(i)}\right) \neq 0$, resulta

$$r - x^{(i)} = -\frac{f\left(x^{(i)}\right)}{f'(x^{(i)})} - \frac{f''(\theta)}{2f'(x^{(i)})}\left(r - x^{(i)}\right)^2$$

$$r = x^{(i)} - \frac{f\left(x^{(i)}\right)}{f'\left(x^{(i)}\right)} - \frac{f''(\theta)}{2f'\left(x^{(i)}\right)}\left(r - x^{(i)}\right)^2$$

e

$$r - x^{(i+1)} = -\frac{f''(\theta)}{2f'\left(x^{(i)}\right)}\left(r - x^{(i)}\right)^2$$

A convergência quadrática

Resulta

$$r - x^{(i+1)} = -\frac{f''(\theta)}{2f'(x^{(i)})}\left(r - x^{(i)}\right)^2$$

Conclusão: $x^{(i+1)} = x^{(i)} - \frac{f(x^{(i)})}{f'(x^{(i)})}$ aproxima o valor da raiz r com erro de truncatura proporcional ao quadrado do erro da aproximação $x^{(i)}$.

Por exemplo, Se $|r - x^{(i)}| \approx 10^{-3}$ então

$$r - x^{(i+1)} \approx -\frac{f''(\theta)}{2f'(x^{(i)})}(10^{-3})^2$$

e se $\left|\frac{f''(\theta)}{2f'(x^{(i)})}\right| \approx 1$ então $|r - x^{(i+1)}| \approx 10^{-6}$.

Para a iteração seguinte

$$r - x^{(i+2)} = -\frac{f''(\mu)}{2f'(x^{(i+1)})}\left(r - x^{(i+1)}\right)^2$$

onde μ está entre r e $x^{(i+1)}$...

Ordem de convergência

Definição: num método iterativo, se

$$\lim_{i\to+\infty} \frac{|r-x^{(i+1)}|}{|(r-x^{(i)})|^p} = C > 0$$

p é a ordem de convergência do método e C é a constante de convergência assimptótica

Para i suficientemente grande, $\left|r - x^{(i+1)}\right| \approx C.\left|r - x^{(i)}\right|^p$

No método de Newton-Raphson, se $f'(r) \neq 0$

$$\boldsymbol{\lim_{i\to+\infty} \frac{\left|r - x^{(i+1)}\right|}{\left|r - x^{(i)}\right|^2} = \left|\frac{f''(r)}{2f'(r)}\right|}$$

$p = 2$ (convergência quadrática) e C $= \left|\boldsymbol{\frac{f''(r)}{2f'(r)}}\right|$ é a constante de convergência assimptótica.

No método da bisseção é $p = 1$ (convergência linear)

Exemplo no Matlab

>> df=@(x) -1000*(-1/x^2*((1+x)^5-1)+(1+1/x)*5*(1+x)^4)

>> x= 0.3;   % aproximação inicial

>> x=x-f(x)/df(x) % 1ª iteração
x =0.118642027821101

>> x=x-f(x)/df(x) % 2ª iteração
x=**0.06**5390200813148

>> x=x-f(x)/df(x) % 3ª iteração
x=**0.0614**22972148339

>> x=x-f(x)/df(x) % 4ª iteração
x=**0.06140241**2085601

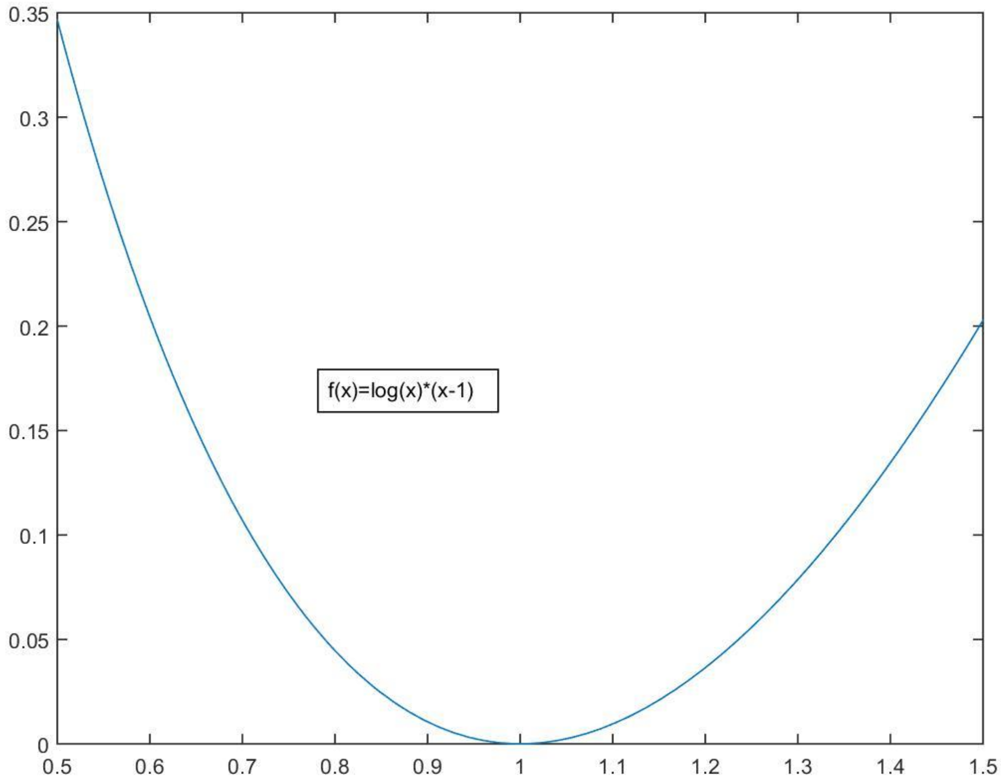
>> x=x-f(x)/df(x) % 5ª iteração
x=**0.061402411536525**

>> x=x-f(x)/df(x) % 6ª iteração
x=**0.061402411536525**

NOTA: o método da bissecção precisou de 14 iterações para produzir 0.0614…

Quando a raiz não é simples (1)

Se r tem multiplicidade superior a 1, é $f'(r) = 0$ (à medida que $x^{(i)}$ se aproxima da raiz, a reta tangente tende para o eixo dos xx). A convergência é apenas linear.



Um exemplo de aplicação

Nos primeiros modelos de computadores digitais a divisão não era efetuada por hardware mas sim por software. Assim, a divisão de a por b, implicava a multiplicação de a pelo inverso de b.

O inverso aritmético de um número $b \neq 0$ é a raiz da equação $b - \frac{1}{x} = 0$.

A fórmula iterativa $x^{(i+1)} = x^{(i)} - \frac{f\left(x^{(i)}\right)}{f'\left(x^{(i)}\right)}$

dá

$$x^{(i+1)} = x^{(i)} - \frac{b - \frac{1}{x^{(i)}}}{\frac{1}{(x^{(i)})^2}} = x^{(i)} - (b(x^{(i)})^2 - x^{(i)})$$

ou seja $x^{(i+1)} = x^{(i)} (2 - bx^{(i)})$

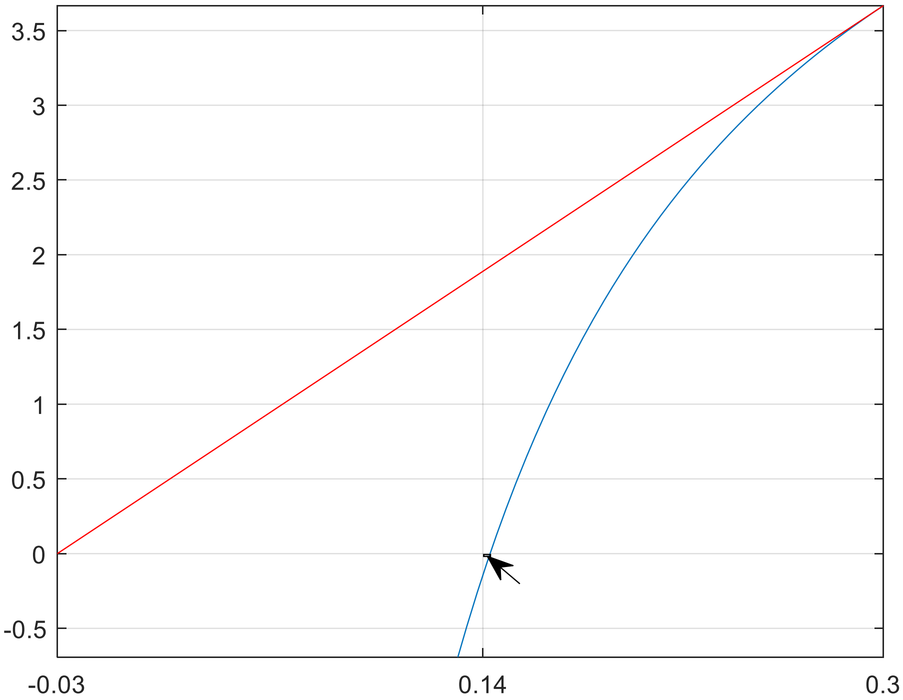
Para calcular o valor de $1/7$, por exemplo, sem usar divisão, podemos começar com $x^{(0)}=0.1$ e usar a fórmula anterior para calcular $x^{(1)}, x^{(2)},\ldots$

O método de Newton nem sempre converge

No exemplo anterior, começando com $x^{(0)}$=0.3, o método diverge.

A tangente à curva no ponto de abcissa 0.3 interseta o eixo dos xx no ponto de abcissa -0.03

```
>> b=7; x=0.3;
>> x=x*(2-b*x)
x =  -0.030000000000000
```



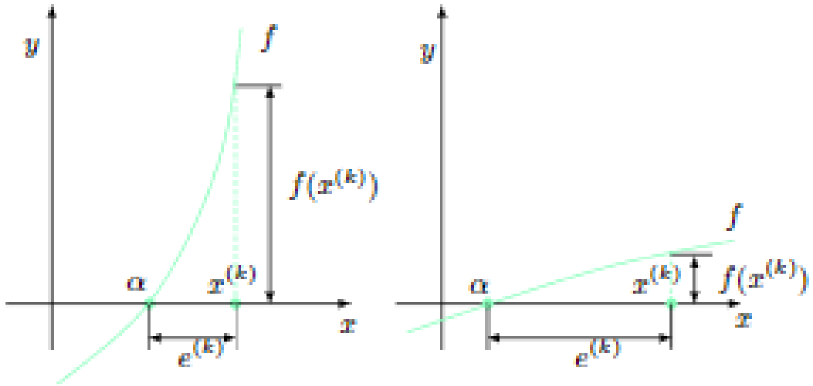
Critérios de paragem

$x^{(0)}, x^{(1)}, x^{(2)},\ldots x^{(k)}, \ldots . \to \alpha$ parar quando $|e^{(k)}| = |x^{(k)} - \alpha| < tol$?

Fixada uma tolerância tol:

- $|x^{(k)} - x^{(k-1)}| < tol$ (teste sobre o incremento)
- $|f(x^{(k)})| < tol$ (teste sobre o resíduo)

O erro $|x^{(k)} - \alpha|$ e o resíduo $|f(x^{(k)})|$ podem ser muito diferentes



O método do ponto fixo (1)

No Matlab, partindo de um qualquer valor real x, e repetindo o comando

>> x=cos(x)

a sucessão de valores converge (embora lentamente) para 0.7391

Este número diz-se um **ponto fixo** da função cosseno. É uma raiz da equação $x - \cos(x) = 0$.

Em geral, um ponto fixo de uma função φ é uma raiz da equação

$$x = \varphi(x)$$

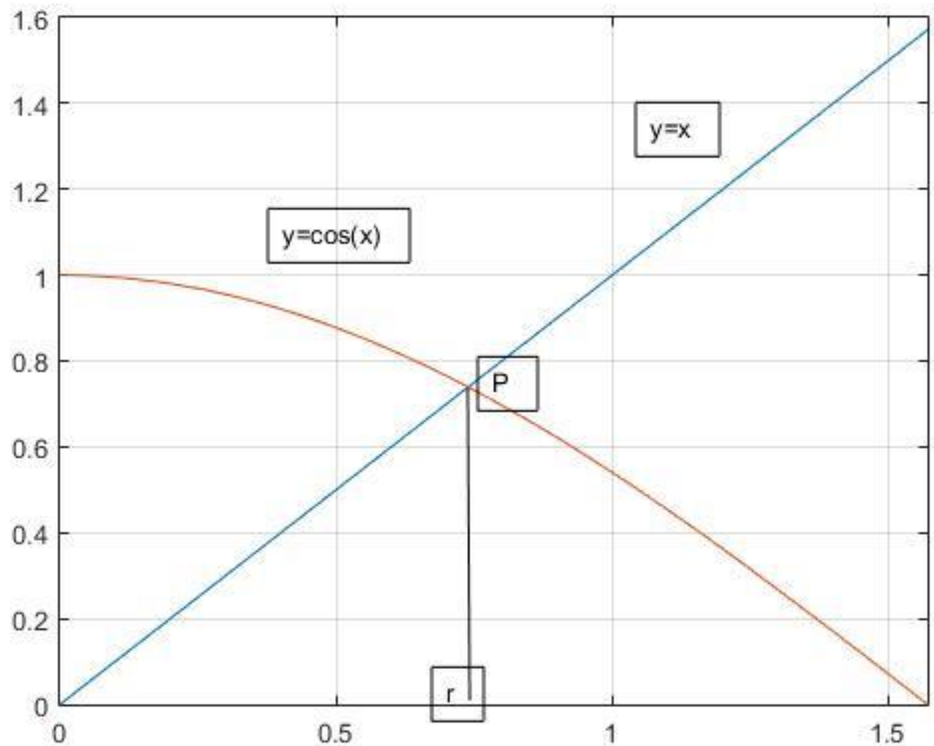
ou

$$\mathrm{f}(x) = 0$$

com $\mathrm{f}(x) = x - \varphi(x)$.

O método do ponto fixo (2)

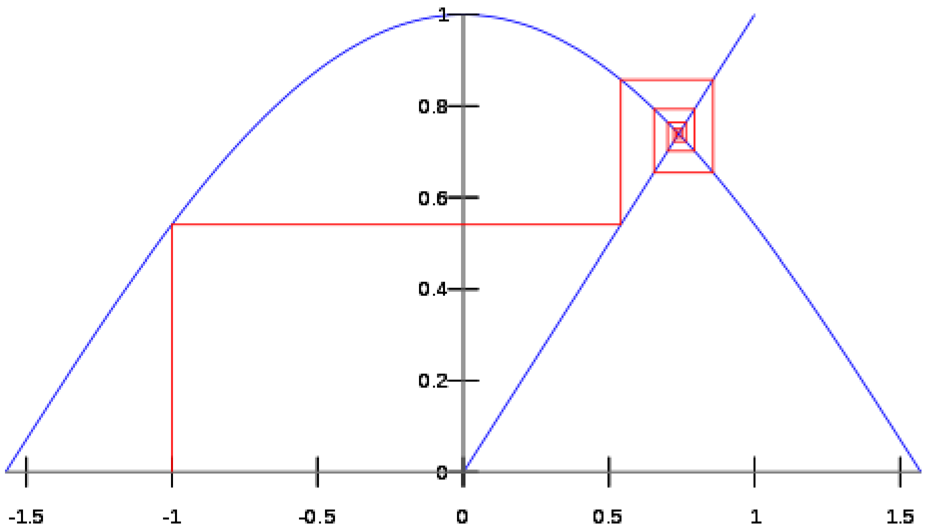
O ponto fixo de cos(x) é a abcissa do ponto P em que a curva da função cosseno interseta a reta y=x.



Interpretação geométrica das iterações do ponto fixo

O ponto fixo de cos(x) é a abcissa do ponto P em que a curva da função cosseno interseta a reta y=x.

$x^{(0)}$=-1 aproximação inicial, $x^{(1)} = \cos(-1) = 0.54\ldots$, $x^{(2)} = \cos(0.54\ldots) = 0.85\ldots$



Análise da convergência do método do ponto fixo (1)

$$x = \varphi(x)$$

A partir da aproximação inicial $x^{(0)}, x^{(1)} = \varphi(x^{(0)}), x^{(2)} = \varphi(x^{(1)}), \dots x^{(k+1)} = \varphi(x^{(k)})$

Que condições devem satisfazer a função iteradora φ e o valor inicial $x^{(0)}$ para que a sucessão seja convergente para o ponto fixo?

Teorema do valor médio de Lagrange: se φ tem derivada contínua em [a,b], então existe pelo menos um ponto θ entre a e b tal que

$$\varphi'(\theta) = \frac{\varphi(b) - \varphi(a)}{b - a}$$

Interpretação intuitiva: se $\varphi(b) - \varphi(a)$ representar a distância percorrida desde o instante a até ao instante b, então $\frac{\varphi(b)-\varphi(a)}{b-a}$ é a velocidade média nesse intervalo de tempo. O valor $\varphi'(\theta)$ é a velocidade (instantânea) no instante θ. Em pelo menos um instante, a velocidade é igual à velocidade média.

Análise da convergência do método do ponto fixo (2)

Aplicando o teorema do valor médio à **função iteradora $\boldsymbol{\varphi}$** no intervalo $[x^{(k)}, r]$: existe θ entre $x^{(k)}$ e r tal que

$$\varphi'(\theta) = \frac{\varphi\left(x^{(k)}\right) - \varphi(r)}{x^{(k)} - r}$$

resulta $$\varphi\left(x^{(k)}\right) - \varphi(r) = \varphi'(\theta)(x^{(k)} - r).$$

Uma vez que $\varphi\left(x^{(k)}\right) = x^{(k+1)}$ e $\varphi(r) = r$:

$$x^{(k+1)} - r = \varphi'(\theta)(x^{(k)} - r)$$

O erro na iteração k+1 é igual ao erro na iteração anterior multiplicado por $\varphi'(\theta)$. Se $|\varphi'(\theta)| < 1$ então $$|x^{(k+1)} - r| < |x^{(k)} - r|.$$

Se existir M $\in\,]0,1[$ tal que $|\varphi'(x)| < M$ num intervalo I centrado em r e $x^{(0)} \in$ I, então o método converge porque

$$|x^{(k)} - r| < M^k \, |x^{(0)} - r| \text{ e } \quad M^k \to 0$$

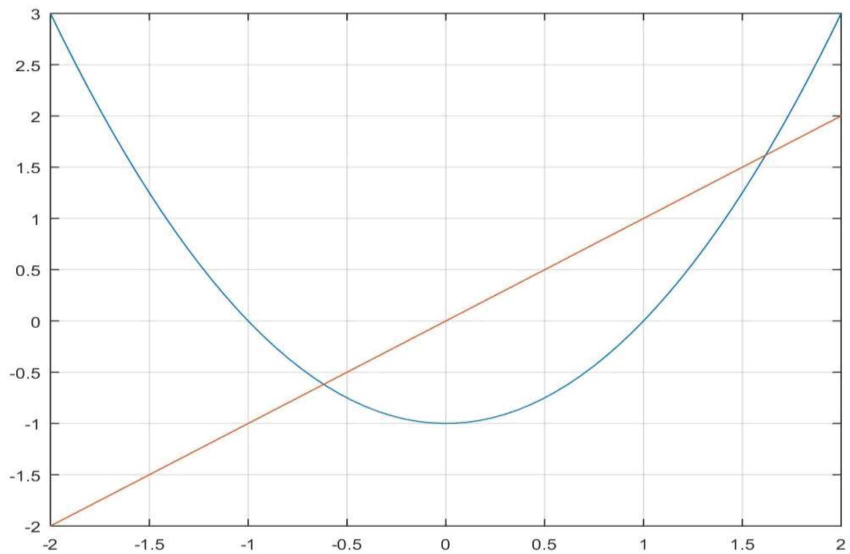
Análise da convergência do método do ponto fixo (3)

Exemplo 1: a função $\varphi(\mathrm{x}) = \cos(\mathrm{x})$ satisfaz a condição requerida:

$$\varphi'(r) = -\sin(r) = -\sin(0.7391 \ldots) \approx -0.67$$

Em qualquer intervalo $\mathrm{I} = [r - a, r + a]$ que não contenha $\pi/2$ tem-se $|\varphi'(x)| < M < 1$ e a convergência está garantida com $x^{(0)} \in I$.

Exemplo 2: $\varphi(\mathrm{x}) = x^2 - 1$ tem dois pontos fixos $r_{\pm} = \frac{1 \pm \sqrt{5}}{2}$ mas $|\varphi'(r_{\pm})| = \left|1 \pm \sqrt{5}\right| > 1$. Não há convergência.



A escolha da função iteradora (1)

Dada a equação $f(x) = 0$ há infinitas maneiras de a reescrever na forma $x = \varphi(x)$, por exemplo $x = x + f(x)$. As boas escolhas são aquelas que cumprem a condição de ser $|\varphi'(x)|$ próximo de zero numa vizinhança da raiz da equação.

Exemplo Com $x \neq 0$, a equação $\frac{1}{x} - e^{x} = 0$ pode escrever-se na forma

i. $x = e^{-x}$, $\varphi_1(x) = e^{-x} \Rightarrow \varphi'_1(x) = -e^{-x}$

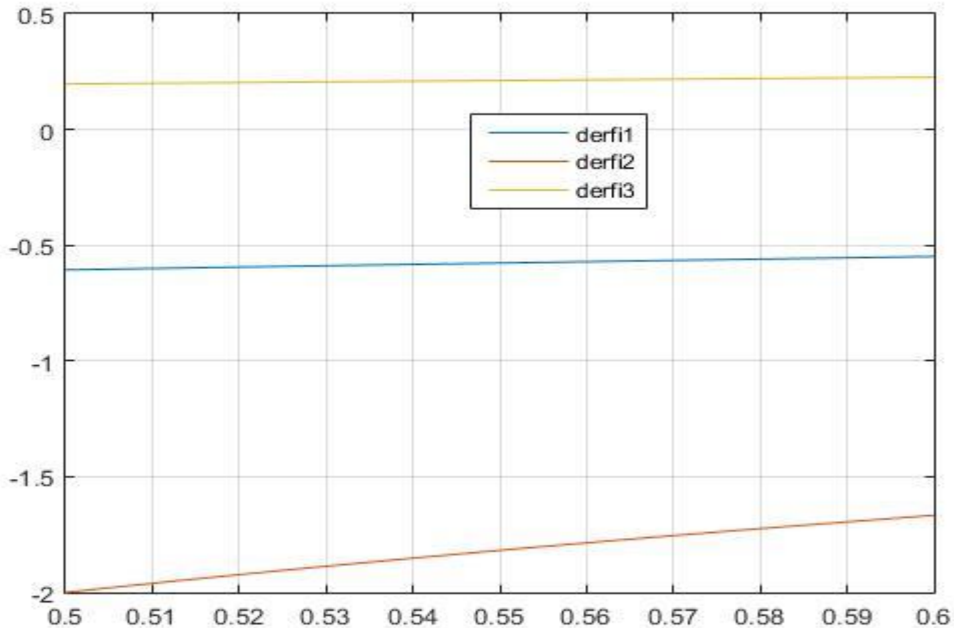
ii. $x = -\log(x)$, $\varphi_2(x) = -\log(x) \Rightarrow \varphi'_2(x) = -\frac{1}{x}$

iii. $x = \frac{x+e^{-x}}{2}$, $\varphi_3(x) = \frac{x+e^{-x}}{2}, \Rightarrow \varphi'_3(x) = \frac{1-e^{-x}}{2}$

Começando por observar que a raiz da equação está entre 0.5 e 0.6, no Matlab produzimos os gráficos de φ'_1, φ'_2 e φ'_3 no intervalo [0.5,0.6]:

```
>> fplot(@(x)[-exp(-x), -1./x, (1-exp(-x))./2],[0.5,0.6])
>> grid on, legend('derfi1','derfi2','derfi3')
```

A escolha da função iteradora (2)



Conclusões:

- As funções iteradoras φ_1 e φ_3 produzem sucessões convergentes desde que $0.5 < x^{(0)} < 0.6$ porque neste intervalo é $|\varphi'_1(x)|<1$ e é $|\varphi'_3(x)|<1$;
- φ_3 produz convergência mais rápida porque $|\varphi'_3(x)|<|\varphi'_1(x)|$;
- A sequência produzida com φ_2 diverge porque $|\varphi'_2(x)|>1$

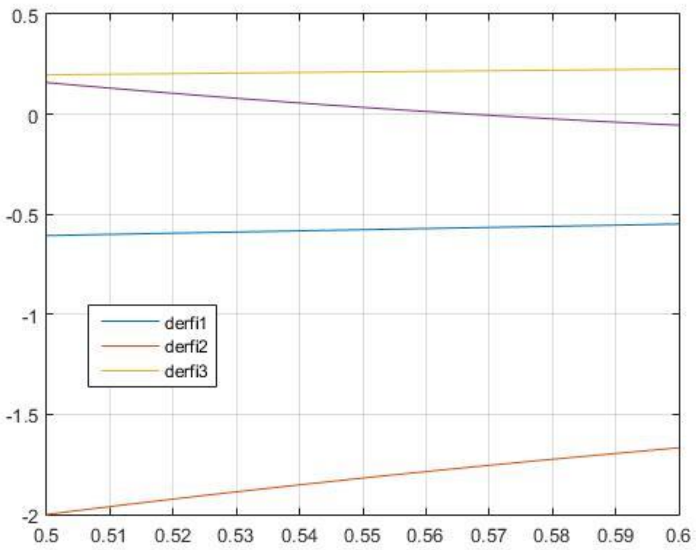
Uma escolha especial da função iteradora

No método de Newton-Raphson, a equação $f(x) = 0$ é reescrita na forma

$x = x - \frac{f(x)}{f'(x)}$ ou seja $x = \varphi(x)$ com $\varphi(x) = x - \frac{f(x)}{f'(x)}$

Tem-se $\varphi'(x) = 1 - \frac{(f'(x))^2 - f(x)f''(x)}{(f'(x))^2} = \frac{f(x)f''(x)}{(f'(x))^2}$ e $\varphi'(r) = 0$.

Na figura anterior vamos sobrepor o gráfico de $\varphi'(x)$ para esta escolha de φ :




```
>>hold on, fplot(@(x)(1./x-exp(x)).*(2./x.^3-exp(x))./(-1./x.^2-exp(x)).^2,[0.5,0.6])
```

Estimativa do erro $\left|x^{(k+1)} - r\right|$ (1)

Se pararmos a as iterações quando $\left|x^{(k+1)} - x^{(k)}\right| < \text{tol}$, podemos garantir $\left|r - x^{(k)}\right| < tol$?

Em geral, não. Tal depende dos valores de $|\varphi'(x)|$ na vizinhança da raiz r.

$$r - x^{(k)} = (r - x^{(k+1)}) + (x^{(k+1)} - x^{(k)})$$

e, uma vez que

$$r - x^{(k+1)} = \varphi'(\theta)(r - x^{(k)})$$

resulta

$$r - x^{(k)} = \varphi'(\theta)(r - x^{(k)}) + (x^{(k+1)} - x^{(k)})$$

ou seja $$r - x^{(k)} = \frac{1}{1-\varphi'(\theta)}\,(x^{(k+1)} - x^{(k)})$$

- Se $\varphi'(x) \approx 0$ numa vizinhança de r, então a diferença entre duas iteradas sucessivas dá uma boa estimativa do erro.
- $-1 < \varphi'(x) < 0 \Rightarrow \left|r - x^{(k)}\right| < |x^{(k+1)} - x^{(k)}|$
- $\varphi'(x) \approx 1 \Rightarrow \left|r - x^{(k)}\right| \gg |x^{(k+1)} - x^{(k)}|$

Estimativa do erro $\left|x^{(k+1)} - r\right|$ (2)

Exemplo: a equação $(x-r)^3 = 0$ tem a raiz r (tripla) e a fórmula iterativa

$$x^{(k+1)} = x^{(k)} - \frac{f\left(x^{(k)}\right)}{f'(x^{(k)})}$$

com $f(x) = (x-r)^3$ dá

$$x^{(k+1)} = x^{(k)} - \frac{x^{(k)} - r}{3}$$

Tem-se $x^{(k+1)} - r = \frac{2}{3}(x^{(k)} - r)$ e a convergência é linear.

A função iteradora é $\varphi(x) = x - \frac{x-r}{3}$ e a derivada é $\varphi'(x) = \frac{2}{3}$ (constante)

Resulta

$$r - x^{(k)} = \frac{1}{1-\frac{2}{3}}\ (x^{(k+1)} - x^{(k)}) = 3\ (x^{(k+1)} - x^{(k)})$$

O condicionamento das raízes (1)

No Matlab vamos calcular os coeficientes do polinómio mónico que tem os zeros 1, 1.999, 2 e 2.001.

>> p=poly([1, 1.999, 2, 2.001])

p =

1.0000 -7.0000 18.0000 -20.0000 8.0000 ,

Trata-se do polinómio $p(x) \approx x^4 - 7x^3 + 18x^2 - 20x + 8$

Vamos introduzir uma perturbação igual a 10^{-4} num dos coeficientes, por exemplo, vamos considerar o polinómio perturbado $\tilde{p}(x) \approx 1.0001x^4 - 7x^3 + 18x^2 - 20x + 8$.

Qual será o efeito desta perturbação sobre os valores dos zeros 1, 1.999, 2 e 2.001 ? Da ordem de grandeza da perturbação 10^{-4} ou maior?

>> p(1)=p(1)+1e-4; r=roots(p)

r =

2.0559 + 0.1052i

2.0559 - 0.1052i

1.8873 + 0.0000i

1.0001 + 0.0000i

O condicionamento das raízes (2)

O erro é maior do que a perturbação 10^{-4} no caso dos zeros 1.999, 2 e 2.001 mas não no caso do zero igual a 1:

```
>> err=abs(r-[2.001; 2; 1.999; 1])
err =
   0.1187
   0.1192
   0.1117
   0.0001
```

Como se explica isto? Pode mostrar-se que o número de condição (absoluto) de uma raiz r da equação $f(x) = 0$ é igual a $\frac{1}{|f'(r)|}$.

$p(x) = x^4 - 7x^3 + 18x^2 - 20x + 8$

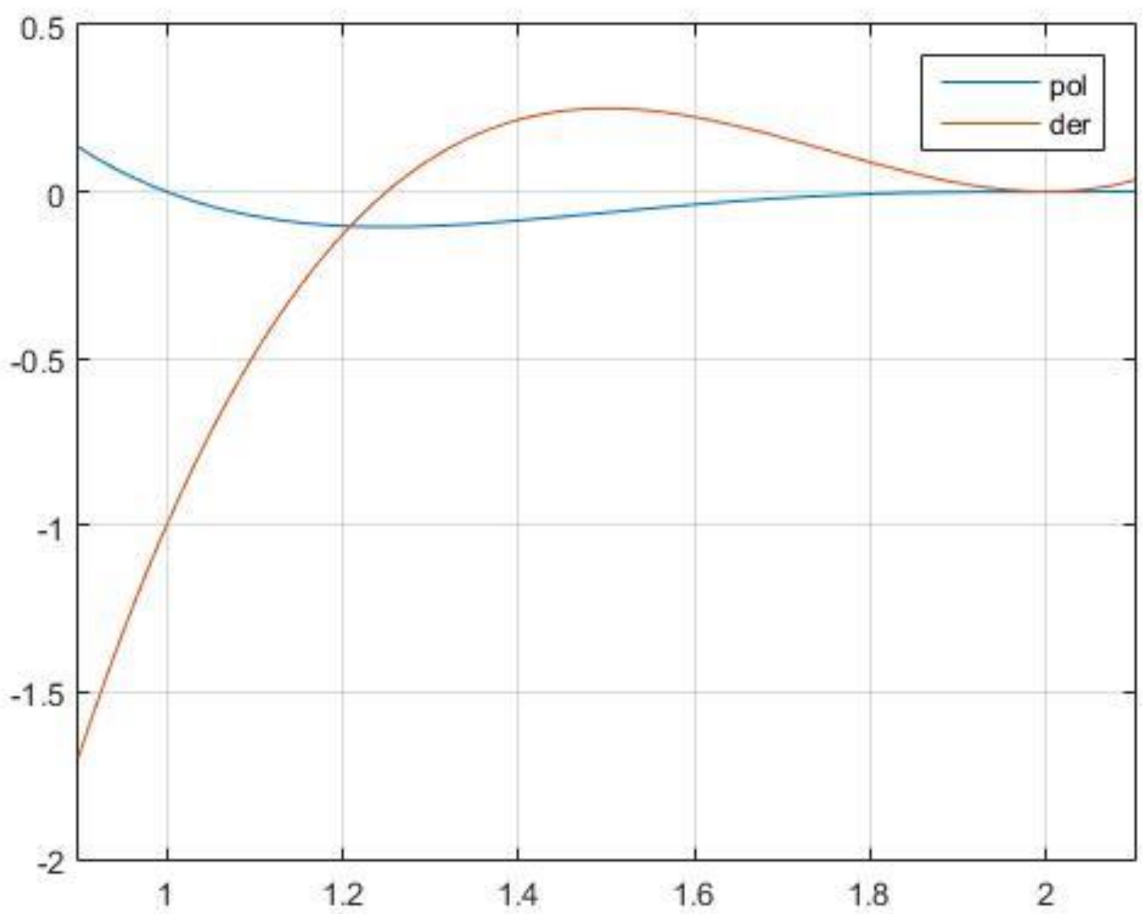
$p'(x) = 4x^3 - 21x^2 + 36x - 20$

Tem-se $p'(2.001) = 3.004e - 6$ e o erro 0.1187 é bem maior do que 10^{-4} (embora não tão grande quanto $\frac{10^{-4}}{3*10^{-6}}$).

O zero igual a 1 é bem condicionado porque $p'(1) = -1$.

O condicionamento das raízes (3)

Quanto menor for $|f'(r)|$ pior é o condicionamento da raiz r da equação $f(x)$=0.



O condicionamento das raízes (4)

No Matlab vamos calcular os coeficientes do polinómio mónico que tem o zero 2 com multiplicidade 9, isto é, $p(x) = (x-2)^9$.

>> p=poly([2 2 2 2 2 2 2 2 2])

p =

1	**-18**	**144**	**-672**	**2016**	**-4032**	**5376**	**-4608**	**2304**	**-512**

>> roots(p)

ans =

2.0689 + 0.0000i
2.0518 + 0.0449i
2.0518 - 0.0449i
2.0100 + 0.0668i
2.0100 - 0.0668i
1.9655 + 0.0566i
1.9655 - 0.0566i
1.9383 + 0.0218i
1.9383 - 0.0218i,

Se r é raiz múltipla da equação $f(x) = 0$ então $f'(r) = 0$ e a raiz é mal condicionada. O condicionamento é pior para multiplicidades maiores

,

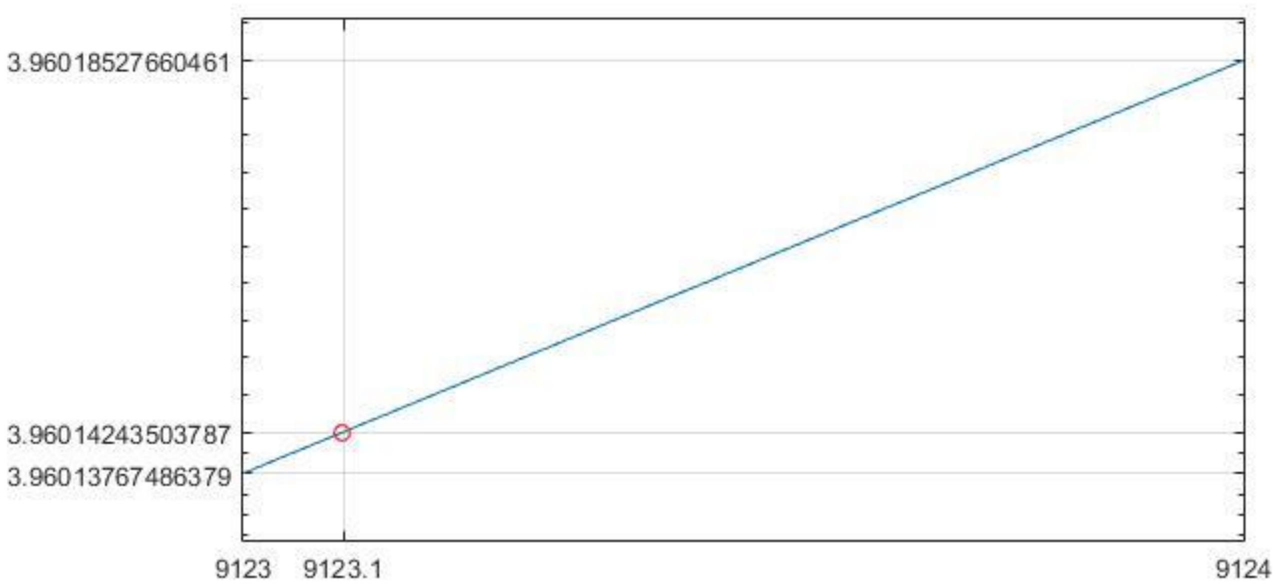
III. Interpolação polinomial de Lagrange

2		Logarithmi.			Logarithmi.
1	0	0000,00000,00000	34	1	5314,78917,04226
2	0	3010,29995,66398	35	1	5440,68044,35028
3	0	4771,21254,71966	36	1	5563,02500,76729
4	0	6020,59991,32796	37	1	5682,01724,06700
5	0	6989,70004,33602	38	1	5797,83596,61681
6	0	7781,51250,38364	39	1	5910,64607,02650
7	0	8450,98040,01426	40	1	6020,59991,32796
8	0	9030,89986,99194	41	1	6127,83856,71974
9	0	9542,42509,43932	2	1	6232,49290,39790
10	1	0000,00000,00000	43	1	6334,68455,57959

Em 1624, Henry Briggs publicou *Arithmetica Logarithmica* , uma tabela (tábua) com os logaritmos (com 14 algarismos decimais) dos números inteiros de 1 a 20 000 e de 90 001 até 100 000.

Usou, entre outros, o método da interpolação linear.

III. Interpolação polinomial de Lagrange




```
>> z=interp1([9123,9124],log10([9123,9124]),9123.1)
```

dá a ordenada no ponto de abcissa 9123.1 situado na reta que passa pelos pontos (9123,log10(9123)) e (9124,log10(9124)).

III. Interpolação polinomial de Lagrange

Existência e unicidade do polinómio interpolador de grau não superior a n

Dados (n+1) pontos, $(x_i, y_i), i = 0,1, \dots n, x_i$ distintos, existe e é único o polinómio π_n de grau menor ou igual a n, tal que

$$\pi_n(x_i) = y_i,\ i = 0,1, \dots n. \qquad (1)$$

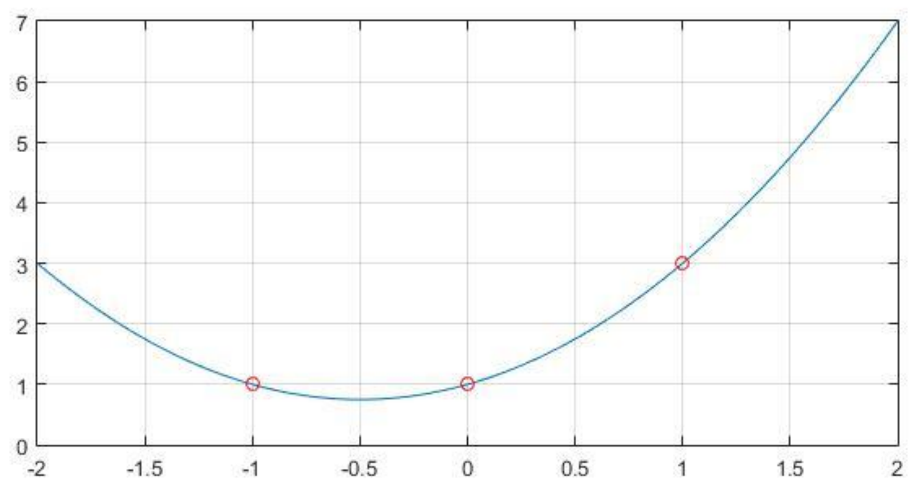
Se $y_i = f(x_i),\ \pi_n$ é o polinómio interpolador de f nos **nós** x_i. Os y_i são os **valores nodais**.

n=1: existe uma e uma só reta que passa por dois pontos

n=2: existe uma e uma só parábola que passa por 3 pontos …

Por exemplo, a parábola de equação $y = x^2 + x + 1$ passa pelos pontos (-1,1), (0,1) e (1,3)

Não existe outra parábola que passe por estes 3 pontos…



III. Interpolação polinomial de Lagrange

os coeficientes do polinómio interpolador

Com $\pi_n(x) = a_1\, x^n + a_2\, x^{n-1} + \cdots + a_n\, x + a_{n+1}$,

as relações

$$\pi_n(x_i) = y_i, i = 0,1, \dots n.$$

Os coeficientes são a solução do sistema

$$\begin{cases} a_1 x_0^n + a_2 x_0^{n-1} + \cdots + a_n x_0 + a_{n+1} = y_0 \\ a_1 x_1^n + a_2 x_1^{n-1} + \cdots + a_n x_1 + a_{n+1} = y_1 \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots . \\ a_1 x_n^n + a_2 x_n^{n-1} + \cdots a_n x_n + a_{n+1} = y_n \end{cases}$$

ou, em notação matricial,

$$\begin{bmatrix} x_0^n & \cdots & 1 \\ \vdots & \ddots & \vdots \\ x_n^n & \cdots & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_{n+1} \end{bmatrix} = \begin{bmatrix} y_0 \\ \vdots \\ y_n \end{bmatrix}$$

III. Interpolação polinomial de Lagrange

A matriz de Vandermonde

O determinante da matriz $\begin{bmatrix} x_0^n & \cdots & 1 \\ \vdots & \ddots & \vdots \\ x_n^n & \cdots & 1 \end{bmatrix}$ é diferente de zero se e só se os nós são distintos (exercício da folha 4).

Neste caso, o sistema com n+1 equações e n+1 incógnitas é possível e determinado, isto é, tem uma e uma só solução $a_1, a_2, \ldots, a_{n+1}$. Portanto, o polinómio de grau não superior a n, que satisfaz as n+1 condições (1), existe e é único.

<u>Exemplo</u>: determinar o polinómio π_2 de grau não superior a 2 que "passa" pelos 3 pontos (2,8), (3,11) e (4,14).

Os coeficientes de $\pi_2(x) = a_1 x^2 + a_2 x + a_3$ são a solução de

$$\begin{bmatrix} 4 & 2 & 1 \\ 9 & 3 & 1 \\ 16 & 4 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 8 \\ 11 \\ 14 \end{bmatrix}$$

```
>> a=vander([2,3,4])\[8;11;14]
a =
   0
   3
   2
```

Neste caso, o polinómio é $\pi_2(x) = 3x + 2$, de grau 1 (os pontos são colineares)

III. Interpolação polinomial de Lagrange

A fórmula interpoladora de Lagrange

Em geral, o que se pretende calcular é o valor do polinómio interpolador π_n em pontos dados. Para isto não é necessário determinar os coeficientes de π_n . Existem métodos mais eficientes (isto é, que requerem menos operações aritméticas).

Dados os nós $x_0, x_1, \ldots, x_n$ e os valores nodais $y_0, y_1, \ldots, y_n$, escrevemos

$$\pi_n(x)=y_0.\ L_0(x) + \ y_1.\ L_1(x)+\ldots+y_n.\ L_n(x)= \sum_{i=0}^{n} y_i . L_i(x)$$

onde

- $L_0(x_0)=1$ e $L_0\left(x_j\right)=0$ para $j \neq 0$
- $L_1(x_1)=1$ e $L_1\left(x_j\right)=0$ para $j \neq 1$
- ..
- $L_n(x_n)=1$ e $L_n\left(x_j\right)=0$ para $j \neq n$

$$L_0(x)=\frac{(x-x_1)(x-x_2)\ldots(x-x_n)}{(x_0-x_1)(x_0-x_2)\ldots(x_0-x_n)},\ldots.,\ L_i(x)=\frac{(x-x_0)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_n)}{(x_i-x_0)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)}=\frac{\prod_{j=0,j\neq i}^{n}(x-x_j)}{\prod_{j=0,j\neq i}^{n}(x_i-x_j)}$$

III. Interpolação polinomial de Lagrange

A fórmula interpoladora de Lagrange (exemplo de aplicação)

Anteriormente, determinámos o polinómio π_2 que "passa" nos pontos (2,8), (3,11) e (4,14). Sem usar a expressão encontrada, calculamos agora $\pi_2(x)$ para $x = 2.3$.

$L_0(x)=\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)}$; $L_0(2.3) = \frac{(2.3-3)(2.3-4)}{(2-3)(2-4)} = 0.595$

$L_1(x)=\frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}$; $L_1(2.3) = \frac{(2.3-2)(2.3-4)}{(3-2)(3-4)} = 0.510$

$L_2(x)=\frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}$; $L_2(2.3) = \frac{(2.3-2)(2.3-3)}{(4-2)(4-3)} =$ -0.1050

$\pi_2(2.3)$=8*0.595+11*0.510+14*(-0.1050)=8.9

Observe-se que os denominadores não dependem do ponto x em que se quer calcular π_2 (só dependem dos nós) e portanto podem calcular-se uma única vez e usar-se para o cálculo de π_2 em outros pontos x.

III. Interpolação polinomial de Lagrange

A função poLagrange

$$\pi_n(x) = \sum_{i=0}^{n} y_i \cdot \frac{\prod_{j=0, j\neq i}^{n}(x - x_j)}{\prod_{j=0, j\neq i}^{n}(x_i - x_j)}$$

```
function px=poLagrange (xi,yi,x)
% Dados os nós e respectivos valores nodais na forma dos vectores
% xi e yi, respectivamente, usa a fórmula de Lagrange para calcular
% o valor do polinómio de grau não superior a n interpolador no ponto x
n=length(xi)-1;
px=0;
for i=1:n+1             % Li(x)=num/den;
   num=1; den=1;
   for j=1:n+1
      if (j~=i)
         num=num*(x-xi(j));
         den=den*(xi(i)-xi(j));
      end
   end
   px=px+yi(i)*(num/den);
end
```

III. Interpolação polinomial de Lagrange

Interpolações da função log10 com a função poLagrange

>> format long, xi=[9,10]; px=poLagrange(xi,log10(xi),9.1) % com dois nós

px =**0.95**8818258495392

>> xi=[9,9.5,10]; px=poLagrange(xi,log10(xi),9.1) % com 3 nós

px =**0.9590**35104700299

>> log10(9.1)

ans = **0.959041392321094**

>> xi=[9123,9124]; px=poLagrange(xi, log10(xi), 9123.1) % com dois nós

px = **3.960142435**037876

>> xi=[9123, 9123.5, 9124]; px=poLagrange(xi,log10(xi),9123.1) % com 3 nós

px = **3.9601424352726**63

>> log10(9123.1)

ans =**3.960142435272670**

III. Interpolação polinomial de Lagrange

A complexidade aritmética $O(n^2)$ da fórmula de Lagrange

$$\pi_n(x)=\sum_{i=0}^{n} y_i . \frac{\prod_{j=0,j\neq i}^{n}(x-x_j)}{\prod_{j=0,j\neq i}^{n}(x_i-x_j)}$$

Para cada $i=0,\ldots,n$, o cálculo do valor do numerador requer n subtrações e n-1 multiplicações. O denominador outas tantas. O cálculo de cada $L_i(x)$, $i=0,\ldots,n$, custa então $4n-1$ operações aritméticas.

Em rigor, o total de operações no cálculo de $\pi_n(x)$ é de

$$(n+1)((4n-1)+1)+n=4n^2+5n$$

Esta não é a fórmula mais eficiente para calcular o valor de $\pi_n(x)$

III. Interpolação polinomial de Lagrange

As diferenças divididas

Dados os nós $x_0, x_1, \dots, x_n$ e os valores $y_0 = f(x_0)$, $y_1 = f(x_1)$,..., $y_n = f(x_n)$, de uma certa função f:

- Diferença dividida de primeira ordem relativa aos nós x_0 e x_1: $f[x_0\,, x_1] = \frac{f(x_0)-f(x_1)}{x_0-x_1}$
- D. dividida de primeira ordem relativa aos nós x_1 e x_2: $f[x_1\,, x_2] = \frac{f(x_1)-f(x_2)}{x_1-x_2}$
- …
- D. D. de primeira ordem relativa aos nós x_{n-1} e x_n: $f[x_{n-1}\,, x_n] = \frac{f(x_{n-1})-f(x_n)}{x_{n-1}-x_n}$

- D.D. de 2ª ordem relativa aos nós x_0, x_1 e x_2: $f[x_0\,, x_1, x_2] = \frac{f[x_0\,, x_1]-f[x_1\,, x_2]}{x_0-x_2}$
- D.D. de 2ª ordem relativa aos nós x_1, x_2 e x_3: $f[x_1\,, x_2, x_3] = \frac{f[x_1\,, x_2]-f[x_2\,, x_3]}{x_1-x_3}$
- D.D. de 3ª ordem relativa aos nós x_0, x_1, x_2 e x_3: $f[x_0, x_1\,, x_2, x_3] = \frac{f[x_0\,, x_1, x_2]-f[x_1\,, x_2, x_3]}{x_0-x_3}$

III. Interpolação polinomial de Lagrange

A tabela das diferenças divididas (exemplo)

x_0=1, x_1=2, x_2=3, x_3=4, x_4=5 *(nós)*

y_0=-5, y_1=-25, $y_2 = -47$, $y_3 = -35$, y_4=71

x	f(x)	f[,]	f[,,]	f[,,,]	f[,,,,]
1	-5				
2	-25	-20			
3	-47	-22	-1		
4	-35	12	17	6	
5	71	106	47	10	1

$f(x_0)$=-5

$f[x_0, x_1]$=-20

$f[x_0, x_1, x_2]$=-1

$f[x_0, x_1, x_2, x_3]$=6

$f[x_0, x_1, x_2, x_3, x_4]$=1

III. Interpolação polinomial de Lagrange

A fórmula interpoladora de Newton

De $f[x, x_0] = \frac{f(x)-f(x_0)}{x-x_0}$, com $x \neq x_0$, resulta

$$f(x) = f(x_0) + (x - x_0)\ f[x, x_0] \qquad (1)$$

De $f[x, x_0, x_1] = \frac{f[x, x_0]-f[x_0, x_1]}{x-x_1}$ resulta $f[x, x_0] = f[x_0, x_1] + (x - x_1)\ f[x, x_0, x_1]$

e substituindo em (1) fica

$$f(x) = \boldsymbol{f(x_0) + (x - x_0)\ f[x_0, x_1]} + (x - x_0)(x - x_1)\ f[x, x_0, x_1] \qquad (2)$$

A expressão a bold define o polinómio interpolador de f de grau não superior a 1 nos nós x_0 e x_1 (porquê?)

De $f[x, x_0, x_1, x_2] = \frac{f[x, x_0, x_1]-f[x_0, x_1, x_2]}{x-x_2}$ vem $f[x, x_0, x_1] = f[x_0, x_1, x_2] + (x - x_2)\ f[x, x_0, x_1, x_2]$

e substituindo em (2) fica

$$f(x) = \boldsymbol{f(x_0) + (x - x_0)\ f[x_0, x_1] + (x - x_0)(x - x_1)\ f[x_0, x_1, x_2]} + (x - x_0)(x - x_1)(x - x_2)\ f[x, x_0, x_1, x_2]$$

A expressão a bold define o polinómio interpolador de f de grau não superior a 2 nos nós x_0, x_1 e x_2.

III. Interpolação polinomial de Lagrange

A fórmula interpoladora de Newton

Em geral tem-se

$$f(x)=\pi_n(x) + (x-x_0)(x-x_1)\ldots(x-x_n)\, f[x, x_0, x_1, \ldots, x_n]$$

onde

$$\pi_n(x) = f(x_0)+(x-x_0)\, f[x_0, x_1]+(x-x_0)(x-x_1)\, f[x_0, x_1, x_2]+\ldots+(x-x_0)(x-x_1)\ldots(x-x_{n-1})\, f[x_0, x_1, \ldots x_n]$$

é o polinómio interpolador na forma de Newton, com diferenças divididas, e

$$(x-x_0)(x-x_1)\ldots(x-x_n)\, f[x, x_0, x_1, \ldots, x_n]$$

é o erro do polinómio interpolador

III. Interpolação polinomial de Lagrange

O erro do polinómio interpolador

O erro do polinómio interpolador pode expressar-se numa forma mais conveniente. Tem-se

$$f(x) = \pi_n(x) + (x - x_0)(x - x_1)\ldots(x - x_n)\ \frac{f^{(n+1)}(\xi_x)}{(n+1)!}$$

onde ξ_x é um ponto (indeterminado) que está no intervalo I que contem os nós e o ponto x.

Majoração do erro: se $|f^{(n+1)}(x)| \leq M$ para $x \in I$, então

$$|f(x) - \pi_n(x)| \leq (x - x_0)(x - x_1)\ldots(x - x_n)\ \frac{M}{(n+1)!}$$

O polinómio W_n (x)=$(x - x_0)(x - x_1)\ldots(x - x_n)$, cujos zeros são os nós de interpolação, é o polinómio nodal

III. Interpolação polinomial de Lagrange

Exemplos

Exemplo 1

Com os nós $x_0 = 1, x_1 = 1.2$ e os valores nodais $y_0 = \log(1)$ e $y_1 = \log(1.2)$ vamos usar interpolação linear para aproximar o valor de $\log(1.1)$. A fórmula de Lagrange dá

$$\pi_1(x) = \log(1) \times \frac{x-1.2}{1-1.2} + \log(1.2) \times \frac{x-1}{1.2-1}$$

```
>> x=1.1; log(1)*(x-1.2)/(1-1.2)+log(1.2)*(x-1)/(1.2-1)
ans = 0.0912
```

A fórmula de Newton dá $\pi_1(x) = \log(1) + \frac{\log(1)-\log(1.2)}{1-1.2}(\mathrm{x}-1)$

```
>> x=1.1; log(1)+(log(1)-log(1.2))/(1-1.2)*(x-1)
ans =  0.0912
```

III. Interpolação polinomial de Lagrange

Exemplos

Exemplo 1(cont.)

Para o erro tem-se

$\log(1.1) - \pi_1(1.1) = (1.1\text{-}1) \times (1.1\text{-}1.2) \times \frac{-1/\xi^2}{2}$ onde ξ está entre 1 e 1.2

Uma vez que $|-\frac{1}{\xi^2}| \leq \frac{1}{1^2}$, resulta

$$|\log(1.1) - \pi_1(1.1)| \leq 0.1 \times 0.1 \times \frac{1}{2} = 0.005$$

Observe-que que $\log(1.1)=0.0953\ldots$, $\pi_1(1.1)=0.0912$,

e $$|\log(1.1) - \pi_1(1.1)| = -0.0041\ldots$$

III. Interpolação polinomial de Lagrange

Exemplos

Exemplo 2

Com os nós $x = [1, 1.2, 1.3]$e os valores nodais $y = \log(x)$
vamos usar interpolação quadrática para aproximar o valor de $\log(1.1)$. A fórmula de Lagrange dá

$$\pi_2(x) = \log(1)\ \times \frac{(x-1.2)(x-1.3)}{(1-1.2)(1-1.3)} + \log(1.2) \times \frac{(x-1)(x-1.3)}{(1.2-1)(1.2-1.3)} + \log(1.3) \times \frac{(x-1)(x-1.2)}{(1.3-1)(1.3-1.2)}$$

```
>> poLagrange(x,log(x),1.1)
ans =0.0949
```

Para usar a formula de Newton, começamos por calcular a tabela das diferenças divididas

```
>> T=TabDifDiv(x,log(x))
T =
 1      0        0        0
1.2   0.1823   0.9116     0
1.3   0.2624   0.8004  -0.3706
```

```
> x=1.1;  0+0.9116*(x-1)-0.3706*(x-1)*(x-1.2)
ans = 0.0949
```

III. Interpolação polinomial de Lagrange

Exemplos

Exemplo 2 (cont.)

$\log(x) = \pi_2(x) + (x-1)(x-1.2)(\text{x-1.3})\ \frac{2/{\xi_x}^3}{3!}$

onde ξ_x está entre 1 e 1.3. Resulta que $|2/{\xi_x}^3| \leq 2$ e

$|\log(1.1) - \pi_2(1.1)| \leq (1.1-1)(1.1-1.2)(1.1\text{-}1.3)\ \frac{2}{3!}$

\>>x=1.1; (x-1)*(x-1.2)*(x-1.3)*2/6

ans = 6.6667e-04

Uma vez mais, podemos verificar que o erro é inferior a este majorante

\>>log(1.1)-0.0949

ans = 4.1018e-04

IV. Integração numérica

Teorema Fundamental do Cálculo Integral

$$\int_a^b f(x)dx = F(b) - F(a)$$

F é uma primitiva de f em $[a, b]$, isto é, $F'(x) = f(x)$ para todo $x \in [a, b]$.

Exemplo:

$$\int_0^1 e^{-x}dx = -e^{-1} - (-e^0) = 1 - e^{-1} \qquad [F(x) = -e^{-x}]$$

$$\int_0^1 e^{-x^2}dx, \qquad F = ?$$

Situações em que se usam métodos numéricos para calcular integrais:

- Ainda que f seja definida por uma expressão analítica, não existe a primitiva F ou é difícil determiná-la;
- De f apenas se conhecem os valores que a função toma nalguns pontos

IV. Integração numérica

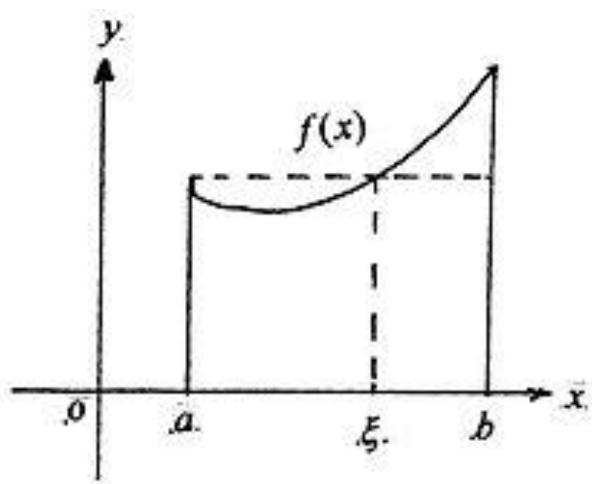
Teorema do valor médio para integrais

Se f é contínua e g é integrável e não muda de sinal em $[a, b]$, então existe $\xi \in [a, b]$ tal que

$$\int_a^b f(x)g(x)dx = f(\xi)\int_a^b g(x)dx$$

Para $g(x)$=1

$$\int_a^b f(x)dx = f(\xi)(b-a)$$



IV. Integração numérica

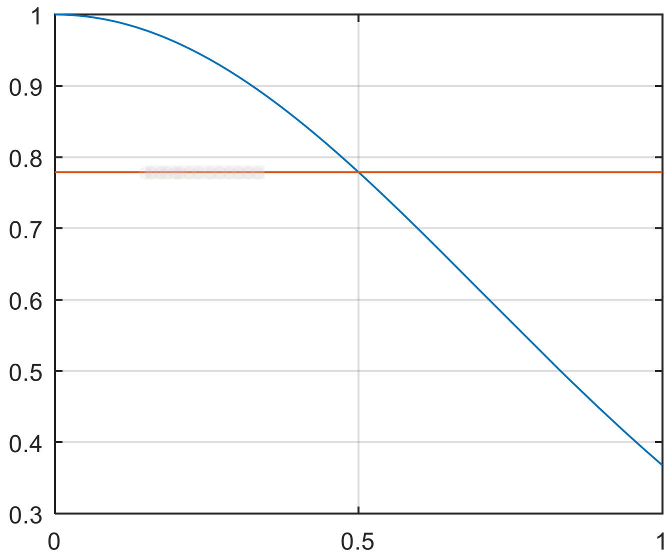
Regras de Newton-Cotes

Baseiam-se na interpolação polinomial

n=0 (regra do ponto médio): aproxima f pelo polinómio de grau 0 que a interpola no ponto médio $\frac{a+b}{2}$.

$$\int_a^b f(x)dx = \int_a^b f\left(\frac{a+b}{2}\right)dx + \text{erro de truncatura}$$

$$= (\mathrm{b}-\mathrm{a}) f\left(\frac{a+b}{2}\right) + \text{erro de truncatura}$$



IV. Integração numérica

Regras de Newton-Cotes

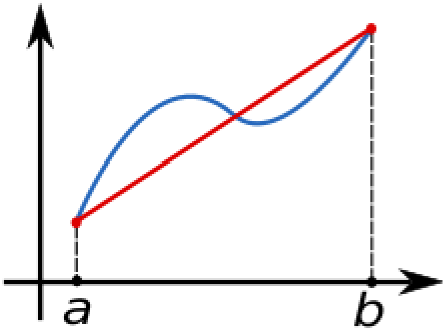
n=1 (regra trapezoidal): aproxima f pelo polinómio de grau não superior a 1 que a interpola nos extremos a e b. De

$$f(x) = f(a)\,\frac{x-b}{a-b} + f(b)\,\frac{x-a}{b-a} + (x-a)(x-b)\,\frac{f''(\xi_x)}{2}$$

resulta

$$\int_a^b f(x)dx = f(a)\int_a^b \frac{x-b}{a-b}\,dx + f(b)\int_a^b \frac{x-a}{b-a}\,dx + \frac{f''(\xi)}{2}\int_a^b (x-a)(x-b)\,dx$$

$$= h\,\frac{f(a)+f(b)}{2} - \frac{h^3}{12} f''(\xi) \quad \text{onde } h = b-a$$



A regra é de grau 1 (exata para polinómios de grau não superior a 1)

IV. Integração numérica

Regras de Newton-Cotes

regra trapezoidal composta: divide $[a, b]$ em n sub-intervalos de igual amplitude $h = \frac{b-a}{n}$ e aplica a regra em cada um dos subintervalos.

https://upload.wikimedia.org/wikipedia/commons/7/7e/Trapezium2.gif

Com $x_i = a + i.h,\ \ i = 1, \dots, n-1$

$$\int_a^b f(x)dx = \int_a^{x_1} f(x)dx + \int_{x_1}^{x_2} f(x)dx + \dots + \int_{x_{n-1}}^{b} f(x)dx$$

$$= \frac{h}{2}[f(a) + f(x_1)] + \frac{h}{2}[f(x_1) + f(x_2)] + \dots + \frac{h}{2}[f(x_{n-1}) + f(b)] - \frac{h^3}{12}\sum_{i=0}^{n-1} f''(\eta_i)$$

$$= \frac{h}{2}[f(a) + 2\sum_{i=1}^{n-1} f(x_i) + f(b)] - \frac{h^3}{12} n f''(\eta)$$

$$= \frac{h}{2}[f(a) + 2\sum_{i=1}^{n-1} f(x_i) + f(b)] - \frac{h^2}{12}(b-a) f''(\eta)$$

Se $|f''(x)| \leq M$ para $a \leq x \leq b$, então o erro de truncatura tende para zero à medida que n cresce.