# Matemática Computacional
# MEBiol, MEBiom e MEFT - Aula 2

Ana Leonor Silvestre

*Instituto Superior Técnico, $1^{\underline{o}}$ Semestre, 2020/2021*

# Matemática Computacional

O objectivo principal é o estudo dos métodos numéricos básicos do cálculo científico, incluindo a análise teórica das suas propriedades de convergência, a implementação computacional dos algoritmos correspondentes e a sua aplicação a exemplos da engenharia, da física, da economia, etc.

A análise dos métodos numéricos centra-se no estudo de condições que garantem a sua convergência e na dedução de fórmulas de majoração dos erros.

# Cap. 1 Conceitos Básicos do Cálculo Científico

Exemplo: Cálculo em sistemas de ponto flutuante

$$f(x) := \frac{1}{x} - \frac{1}{x+1} = \frac{1}{x(x+1)} \quad (x \in \mathbb{R}\backslash\{-1, 0\})$$

No Matlab R2015b (IEEE-754 com precisão dupla):

» format long

» $1/10^{20} - 1/(10^{20}+1)$

ans = 0

» $1/((10^{20}+1) * 10^{20})$

ans = 9.999999999999999e-41 $\quad (\approx 10^{-40})$

O resultado ans = 0 tem erro de 100%.

Como se explica este resultado?

# Cap. 1 Conceitos Básicos do Cálculo Científico

## Conteúdo do Capítulo 1

Principais fontes de erros em Cálculo Científico.
Representação de números no computador, sistemas de ponto flutuante. Arredondamentos, erros de arredondamento, unidade de arredondamento (machine epsilon).
Cálculo em sistemas de ponto flutuante.
Teoria linear de erros. Propagação de erros em funções e algoritmos.
Condicionamento, estabilidade algorítmica.

# Sumário da Aula 2

Principais fontes de erros em Cálculo Científico.
Erro absoluto, erro relativo e percentagem de erro.
Representação dos números no computador. Sistemas de ponto flutuante. Underflow, overflow.
Arredondamento por corte e simétrico. Erros de arredondamento por corte e simétrico. Unidade de arredondamento (machine epsilon).
O padrão IEEE 754 de precisão simples e precisão dupla.

# Erros em Cálculo numérico

- Erros do modelo matemático
- Erros dos dados:
  serão tidos em conta apenas na propagação de erros em funções e algoritmos;
- Erros computacionais:
  os computadores têm memória limitada, o que não permite representar números e efetuar cálculos com precisão infinita;
- Erros do método numérico:
  devem-se ao uso de fórmulas que apenas dão valores aproximados para a solução do problema matemático em causa, por exemplo, truncatura de séries e utilização de polinómios de Taylor para aproximar funções.

# Notação para erros

- $x \approx \tilde{x}$: $\tilde{x}$ valor aproximado de $x$
- erro de $\tilde{x}$: $e_{\tilde{x}} = x - \tilde{x}$
- erro absoluto de $\tilde{x}$: $|e_{\tilde{x}}| = |x - \tilde{x}|$

  $\delta_{\tilde{x}} := \dfrac{e_{\tilde{x}}}{x}\ (x \neq 0)$
- erro relativo de $\tilde{x}$: $|\delta_{\tilde{x}}| = \dfrac{|e_{\tilde{x}}|}{|x|}$
- erro relativo percentual de $\tilde{x}$: $100\%|\delta_{\tilde{x}}|$

Estas noções podem ser generalizadas a vetores e matrizes. Isto será feito no capítulo dedicado à resolução de sistemas de equações.

# Exemplo

Um problema com microprocessadores no Pentium provocou uma falha na divisão de números em ponto flutuante.

Ao dividir 4195835.0 por 3145727.0, o resultado apresentado era 1.33374 ao invés de 1.33382.

$$x = 1.33382, \qquad \tilde{x} = 1.33374$$

$$|e_{\tilde{x}}| = |x - \tilde{x}| = 0.00008$$

$$|\delta_{\tilde{x}}| = \frac{|e_{\tilde{x}}|}{|x|} = \frac{0.00008}{1.33382} = 0.0000599..., \qquad 100\%|\delta_{\tilde{x}}| \approx 0.006\%$$

# Exemplo

$$f(x) := \frac{1}{x} - \frac{1}{x+1} = \frac{1}{x(x+1)} \quad (x \in \mathbb{R}\backslash\{-1, 0\})$$

O cálculo de

$$y := f(10^{20}) \quad (\neq 0)$$

no Matlab R2015b (IEEE-754 com precisão dupla) forneceu:

» format long

» $1/10^{20} - 1/(10^{20}+1)$
ans = 0

Neste caso, $\tilde{y} = 0$ e o erro relativo percentual de $\tilde{y}$ é

$$100\%|\delta_{\tilde{y}}| = 100\% \left|\frac{e_{\tilde{y}}}{y}\right| = 100\% \frac{|f(10^{20}) - 0|}{|f(10^{20})|}.$$

Portanto, o resultado ans = 0 tem erro de 100%.

# Representação de números no computador

Sejam $\beta \in \mathbb{N} \setminus \{1\}$, $n \in \mathbb{N}$, $t_{min}, t_{max} \in \mathbb{Z}$.

Designa-se por sistema de ponto flutuante na base $\beta$, com $n$ dígitos na mantissa $(0.a_1...a_n)_\beta$ (onde $a_i \in \{0, ..., \beta - 1\}$, $i = 1, ..., n$) e expoente $t$ variando entre $t_{min}$ e $t_{max}$ o conjunto

$$\begin{aligned}\mathbb{F} = \mathbb{F}(\beta, n, t_{min}, t_{tmax}) = \\ \{0\} \cup \{x \in \mathbb{Q} : \ x = (-1)^s (0.a_1 a_2 ... a_n)_\beta \times \beta^t, \\ a_1 \neq 0, \ s \in \{0, 1\}, \ t \in \mathbb{Z} \cap [t_{min}, t_{max}]\}.\end{aligned}$$

A mantissa pode ser escrita como

$$\begin{aligned}& (0.a_1 a_2 ... a_n)_\beta \\ = \ & a_1 (0.10...0)_\beta + a_2 (0.010...0)_\beta + ... + a_n (0.0...01)_\beta \\ = \ & a_1 \beta^{-1} + a_2 \beta^{-2} + ... + a_n \beta^{-n}.\end{aligned}$$

# Exercício:

Seja $\mathbb{F} := \mathbb{F}(\beta, n, t_{min}, t_{max})$ um sistema de ponto flutuante.

(i) Qual é o menor número positivo pertencente ao sistema $\mathbb{F}$?

(ii) E o maior número positivo?

(iii) Quanto é $\#\mathbb{F}$?

# Resposta:

$$x \in \mathbb{F} \setminus \{0\} : \; x = (-1)^s (0.a_1 a_2 ... a_n)_\beta \times \beta^t,$$
$$a_1 \neq 0, \quad s \in \{0, 1\}, \quad t \in \mathbb{Z} \cap [t_{min}, t_{max}]$$

(i) Menor número positivo é

$$\begin{aligned} x_{min} &= (0.10...0)_\beta \times \beta^{t_{min}} \\ &= \beta^{-1} \times \beta^{t_{min}} = \beta^{t_{min}-1} \end{aligned}$$

(ii) Maior número positivo é

$$\begin{aligned} x_{max} &= 0.(\beta-1)...(\beta-1))_\beta \times \beta^{t_{max}} \\ &= (1 - (0.0...01)_\beta) \times \beta^{t_{max}} \\ &= (1 - \beta^{-n}) \times \beta^{t_{max}} = \beta^{t_{max}} - \beta^{t_{max}-n} \end{aligned}$$

# Resposta:

(iii) Vejamos quantos números positivos pertencem a $\mathbb{F}$.

$$x = (0.a_1 a_2 ... a_n)_\beta \times \beta^t,$$
$$a_1 \neq 0, \quad s \in \{0, 1\}, \quad t \in \mathbb{Z} \cap [t_{min}, t_{max}]$$

- o dígito $a_1$ pode tomar $\beta - 1$ valores
- os dígitos $a_2, ..., a_n$ podem tomar $\beta$ valores cada um
- o expoente $t$ pode tomar $t_{max} - t_{min} + 1$ valores.

Assim, existem

$$(\beta - 1) \times \beta^{n-1} \times (t_{max} - t_{min} + 1)$$

números positivos em $\mathbb{F}$.

Portanto,

$$\#\mathbb{F} = 1 + 2 \times (\beta - 1) \times \beta^{n-1} \times (t_{max} - t_{min} + 1).$$

# Arredondamentos

**Arredondamento por corte ou truncatura:**
$\mathrm{fl}_c(x)$ é o número de $\mathbb{F}$ mais perto de $x$ que está entre $0$ e $x$

**Arredondamento simétrico:**
$\mathrm{fl}_s(x)$ é o número de $\mathbb{F}$ mais perto de $x$

**Exemplo:**
Representar $\pi$ num sistema de ponto flutuante $\mathbb{F}(10, 7)$

$$\pi = 3.1415926535... = 0.31415926535... \times 10^1$$

$$\mathrm{fl}_c(\pi) = 0.3141592 \times 10^1$$

$$\mathrm{fl}_s(\pi) = 0.3141593 \times 10^1 = \mathrm{fl}((0.3141592 + 0.0000001) \times 10^1)$$

# Arredondamentos, *underflow* e *overflow*

Um sistema de ponto flutuante $\mathbb{F}$ é um conjunto finito, onde apenas é possível obter resultados com precisão finita.

Questão: Dado um número genérico $x \in \mathbb{R}$, se $x \notin \mathbb{F}$, como é feita a sua representação, fl($x$), no sistema $\mathbb{F}$?

- Se $0 < |x| < x_{min}$ então não há precisão suficiente para representar $x$ em $\mathbb{F}$.
  Diz-se que $x$ está no nível de Underflow e será fl($x$)$= 0$
  (ocorre underflow quando números próximos de 0 são arredondados para 0).
- Se $|x| > x_{max}$ então $x$ está na zona de Overflow e eventualmente é apresentado INF como output.

# Arredondamentos, *underflow* e *overflow*

Para $x \in [-x_{max}, -x_{min}] \cup [x_{min}, x_{max}]$, partindo da representação

$$x = (-1)^s (0.a_1 a_2 ... a_n a_{n+1} ...)_\beta \times \beta^t,\ a_1 \neq 0,$$

define-se

- Arredondamento por corte ou truncatura

$$\mathrm{fl}_c(x) = (-1)^s (0.a_1 a_2 ... a_n)_\beta \times \beta^t$$

- Arredondamento simétrico (quando $\beta$ é par)

$$\mathrm{fl}_s(x) = \begin{cases} (-1)^s (0.a_1 a_2 ... a_n)_\beta \times \beta^t,\ a_{n+1} < \frac{\beta}{2}, \\ \mathrm{fl}\left((-1)^s \left((0.a_1 a_2 ... a_n)_\beta + \beta^{-n}\right) \times \beta^t\right),\ \frac{\beta}{2} \leq a_{n+1}, \end{cases}$$

onde o último $\mathrm{fl}$ significa apenas uma eventual normalização.

# Exercício:

1. Num sistema $\mathbb{F}(10, 4)$, as representações de $x = -0.0013296$ são

$$\text{fl}_c(x) = \text{fl}_c(-0.13296 \times 10^{-2}) = -0.1329 \times 10^{-2}$$

$$\text{fl}_s(x) = \text{fl}(-(0.1329 + 0.0001) \times 10^{-2}) = -0.1330 \times 10^{-2}$$

2. Num sistema $\mathbb{F}(10, 4)$, as representações de $y = 0.9999601$ são

$$\text{fl}_c(y) = \text{fl}_c(0.9999601 \times 10^{0}) = 0.9999 \times 10^{0}$$

$$\begin{aligned} \text{fl}_s(x) &= \text{fl}((0.9999 + 0.0001) \times 10^{0}) \\ &= \text{fl}(1.000 \times 10^{0}) = 0.1000 \times 10^{1} \end{aligned}$$

# Erros de arredondamento

## Teorema

Para qualquer $x \in \mathbb{R}$ com representação em $\mathbb{F}$, tem-se

$$|e_{\mathrm{fl}(x)}| \leq \beta^{t-n}, \quad |\delta_{\mathrm{fl}(x)}| \leq \beta^{1-n}, \text{ em arredondamento por corte,}$$

$$|e_{\mathrm{fl}(x)}| \leq \frac{1}{2}\beta^{t-n}, \quad |\delta_{\mathrm{fl}(x)}| \leq \frac{1}{2}\beta^{1-n}, \text{ em arredondamento simétrico.}$$

**Nota:** A quantidade $\beta^{1-n}$ que permite controlar os erros relativos de arredondamento depende apenas do sistema de ponto flutuante.

# Demonstração:

Para o erro absoluto de arredondamento de um número real

$$x = (-1)^s (0.a_1 a_2 ... a_n a_{n+1} ...)_\beta \times \beta^t$$

tem-se

$$|e_{\mathrm{fl}_c(x)}| = |x - \mathrm{fl}_c(x)| = (0.0...0 a_{n+1} ...)_\beta \times \beta^t$$

$$= (0.a_{n+1} ...)_\beta \times \beta^{t-n} \leq \beta^{t-n}$$

e

$$|e_{\mathrm{fl}_s(x)}| = |x - \mathrm{fl}_s(x)| \leq \frac{1}{2} \beta^{t-n}.$$

As desigualdades para os erros relativos $|\delta_{\mathrm{fl}(x)}| = \frac{|e_{\mathrm{fl}(x)}|}{|x|}$ decorrem das majorações efetuadas para os erros absolutos e de

$$|x| \geq (0.10...0...)_\beta \times \beta^t = \beta^{t-1} \Rightarrow \frac{1}{|x|} \leq \beta^{1-t}.$$

# O que é o *machine epsilon*?

A unidade de arredondamento do sistema de ponto flutuante $\mathbb{F}(\beta, n, t_{min}, t_{max})$ ou machine epsilon de $\mathbb{F}(\beta, n, t_{min}, t_{max})$ é

$$\epsilon_M = \beta^{1-n}, \text{ quando é utilizado arredondamento por corte,}$$

ou

$$\epsilon_M = \frac{1}{2}\beta^{1-n}, \text{ quando é utilizado arredondamento simétrico.}$$

A $\epsilon_M$ também se chama precisão do sistema de ponto flutuante, precisão da máquina ou epsilon da máquina.

Em geral tem-se

$$0 < x_{min} \ll \epsilon_M \ll x_{max}.$$

# Os padrões IEEE = Institute of Electrical and Electronics Engineers

Em 1985, o *IEEE Standards Board* e o *American National Standards Institute* adotaram o padrão ANSI/IEEE 754-1985 para a aritmética binária de ponto flutuante. Foi o culminar de quase uma década de trabalho realizado por um grupo interdisciplinar de 92 pessoas, incluindo engenheiros, matemáticos, cientistas da computação, académicos, fabricantes de computadores e empresas de microprocessadores.

Isto significa: existe um modelo independente da máquina para a realização da aritmética de ponto flutuante.

Precisão simples - 32 bits (4 bytes) para a representação numérica: 1 bit para o sinal, 8 bits para o expoente e 23 bits para a mantissa.

Precisão dupla - 64 bits (8 bytes) para a representação numérica: 1 bit para o sinal, 11 bits para o expoente e 52 bits para a mantissa.

# Atualizações/Evolução do padrão IEEE

1985: é adotado o padrão ANSI/IEEE Std 754-1985;

1989: dá-se o reconhecimento internacional como IEC 559;

2008: surge uma versão atualizada, denominada IEEE 754-2008;

2011: o padrão internacional ISO/IEC/IEEE 60559:2011 foi aprovado para utilização;

2019: surge revisão menor IEEE 754-2019.

# Os padrões IEEE

No padrão IEEE 754, define-se a representação dos números de ponto flutuante diferentes de zero por

$$x = (-1)^s (d_1.d_2...d_n)_2 \times 2^t,\ d_1 \neq 0,\ s \in \{0,1\}, t \in \mathbb{Z} \cap [t_{max}, t_{min}].$$

A condição de normalização $d_1 \neq 0$ neste caso significa simplesmente $d_1 = 1$ e podemos considerar a representação

$$x = (-1)^s (1 + 0.a_1...a_N)_2 \times 2^t,\ s \in \{0,1\}, t \in \mathbb{Z} \cap [t_{max}, t_{min}],$$

onde se devem tomar os valores para $N = n - 1$, $t_{max}$ e $t_{min}$ seguintes

| Sistema | $N$ | $t_{min}$ | $t_{max}$ |
|---|---|---|---|
| IEEE precisão simples | 23 | $-126$ | 127 |
| IEEE precisão dupla | 52 | $-1022$ | 1023 |

# Os padrões IEEE

| Sistema | $N$ | $t_{min}$ | $t_{max}$ |
|---|---|---|---|
| IEEE precisão simples | 23 | $-126$ | 127 |
| IEEE precisão dupla | 52 | $-1022$ | 1023 |

| Sistema | $x_{min}$ | $x_{max}$ |
|---|---|---|
| IEEE p. simples | $2^{-126}$ | $(1.1...1)_2 \times 2^{127}$ |
| IEEE p. dupla | $2^{-1022}$ | $(1.1... ...1)_2 \times 2^{1023}$ |

| Sistema | $\epsilon_M$ |
|---|---|
| IEEE p. simples | $2^{-23} \approx 0.2 \times 10^{-6}$ |
| IEEE p. dupla | $2^{-52} \approx 0.2 \times 10^{-15}$ |

# Exercício no Matlab (padrão IEEE com precisão dupla):

Quanto é $x_{min}$?

```
>>  realmin
ans = 2.2251 x 10^(-308)
```

Quanto é $x_{max}$?

```
>> realmax
ans = 1.7977 x 10^308
```

Quanto é $\epsilon_M$?

```
>> eps
ans = 2.2204 x 10^(-16)
```