CL24A-COM4A
CÁLCULO NUMÉRICO

Aula 02 - Tipos de Erros

Prof. Rafael G. Mantovani
Profa. Tamara A. Baldo

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Apucarana - PR, Brasil

Universidade Tecnológica Federal do Paraná (UTFPR)
Engenharia de Computação

Roteiro

1. Contextualização
2. Revisão
3. Definições de Erro
4. Algoritmo Iterativo básico
5. Tipos de erro
6. Referências

Roteiro

1. Contextualização
2. Revisão
3. Definições de Erro
4. Algoritmo Iterativo básico
5. Tipos de erro
6. Referências

Erros

Problemas reais

Erros

Problemas reais

- **raras** exceções teremos **soluções** analíticas (**exatas**)
- trabalhamos com: aproximações ou estimativa dos erros

Erros

Problemas reais

Métodos numéricos

Erros

Problemas reais

Aproximações

Métodos numéricos

Erros

Aproximações

Erros

Problemas reais

Métodos numéricos

Erros

Problemas reais

Aproximações

Erros

erros de arredondamento

erros de truncamento

incerteza dos dados

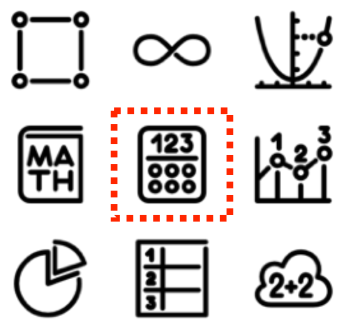
Métodos numéricos

Roteiro

1. Contextualização
2. **Revisão**
3. Definições de Erro
4. Algoritmo Iterativo básico
5. Tipos de erro
6. Referências

Algarismos significativos

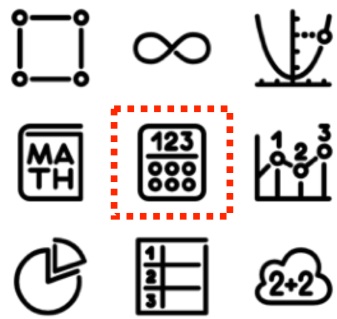
Problemas reais



Métodos numéricos

Algarismos significativos

Problemas reais



Métodos numéricos

solução
(x)

Algarismos significativos

Problemas reai

- solução representada por um número finito de algarismos
- **confiabilidade** do método
- notação científica (base 10), ponto flutuante

solução (x)

Acurácia e Precisão

- Dois conceitos:

acurácia

precisão

Acurácia e Precisão

- Dois conceitos:

acurácia	quão próximo o valor calculado está do valor verdadeiro
precisão	quão próximos os valores calculados estão uns dos doutros

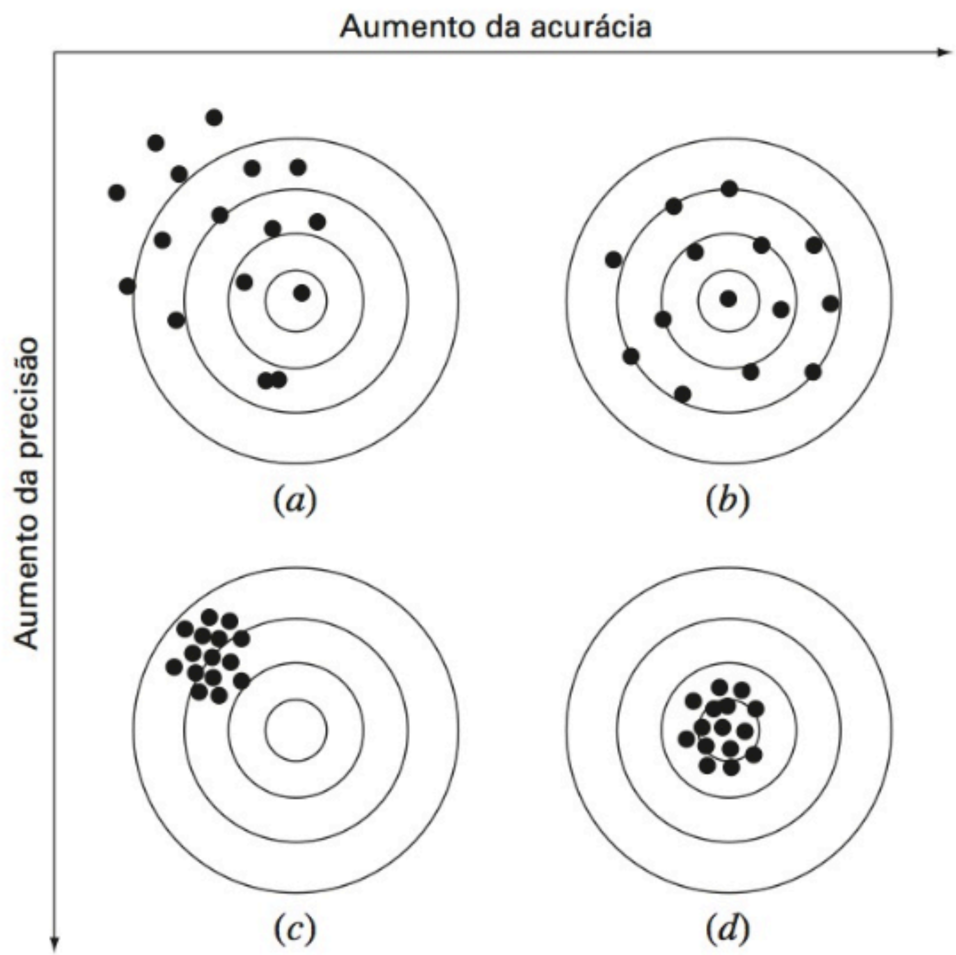
Acurácia e Precisão

- Dois conceitos:

acurácia	quão próximo o valor calculado está do valor verdadeiro
viés	desvio da verdade

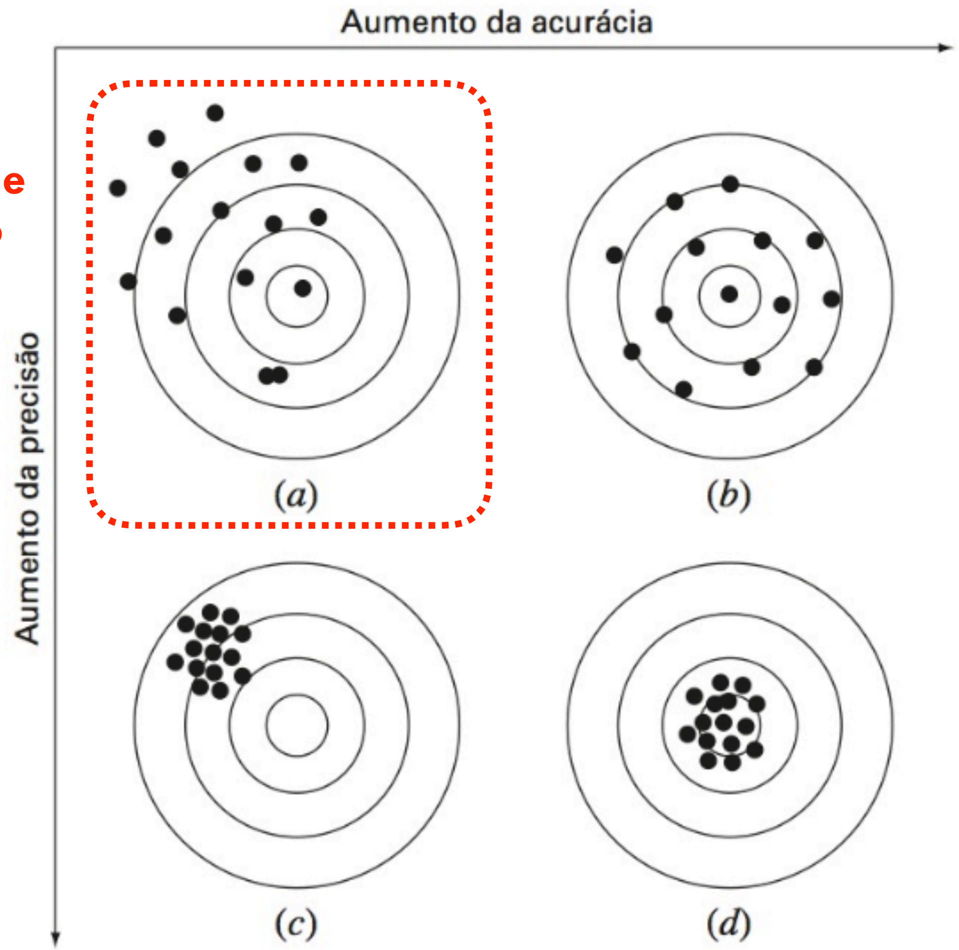
precisão	quão próximos os valores calculados estão uns dos doutros
imprecisão	espalhamento das soluções

Acurácia e Precisão

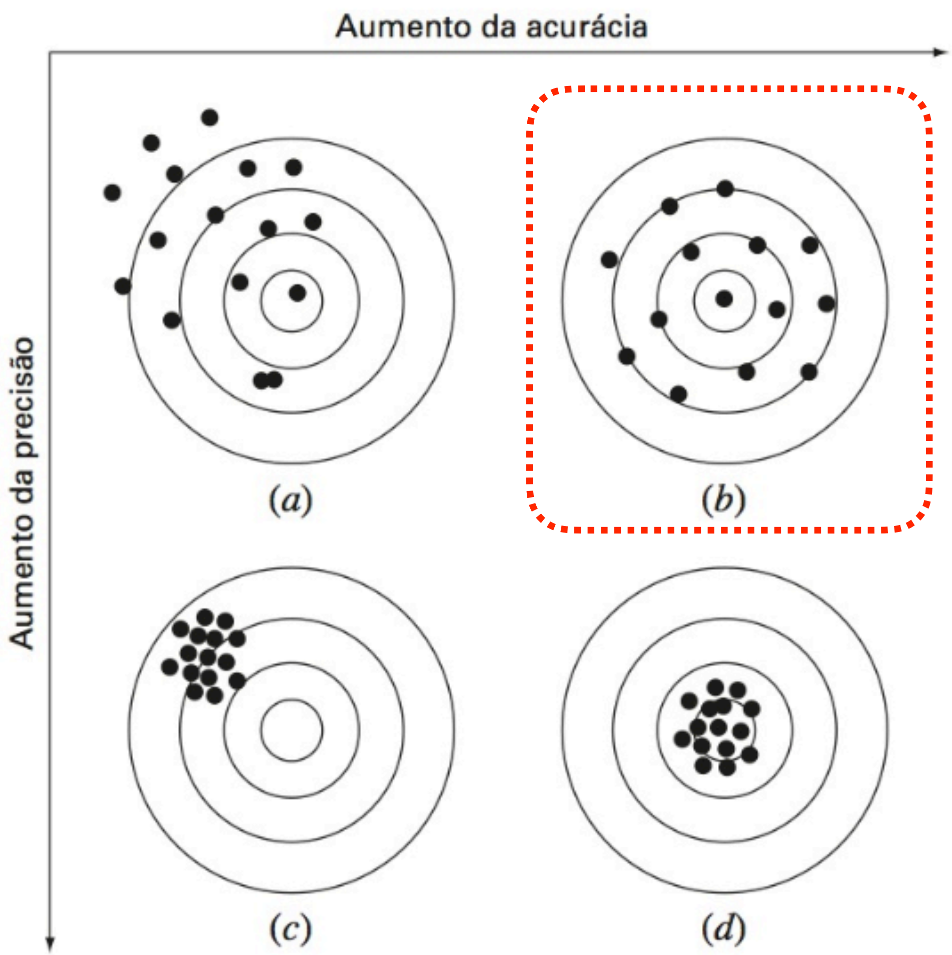


Acurácia e Precisão

Inacurado e impreciso

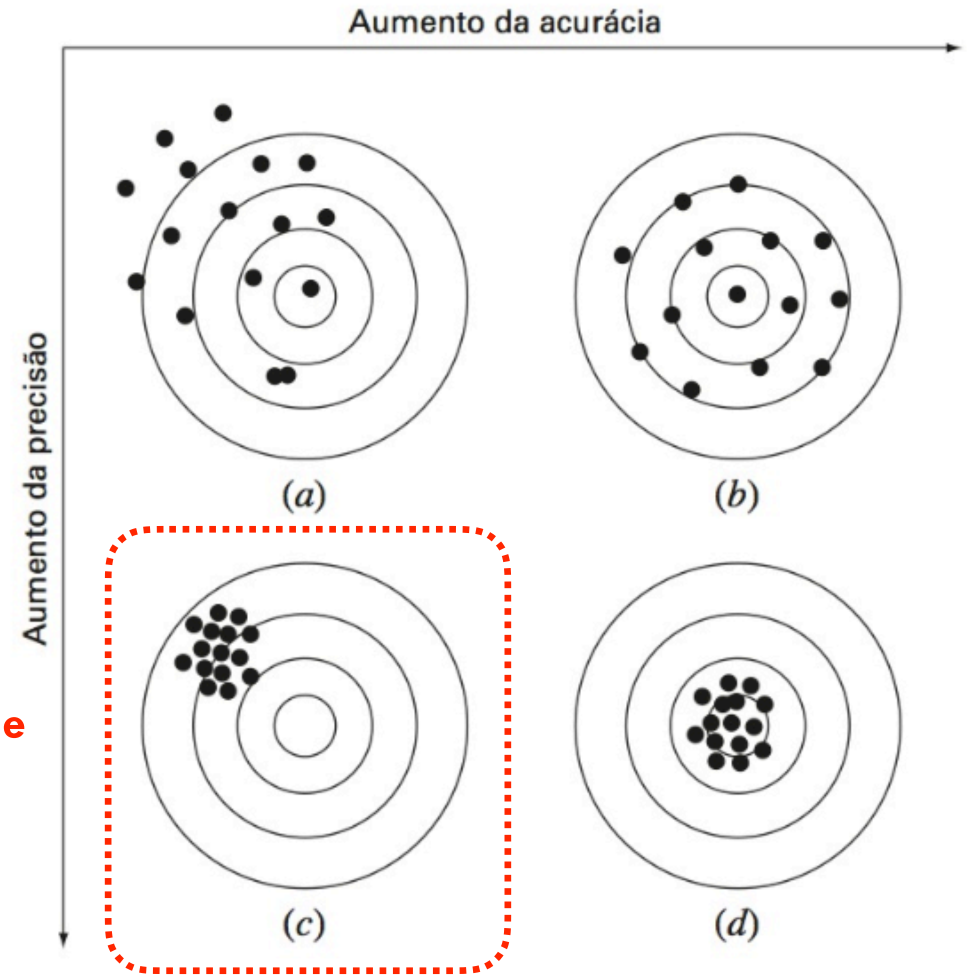


Acurácia e Precisão



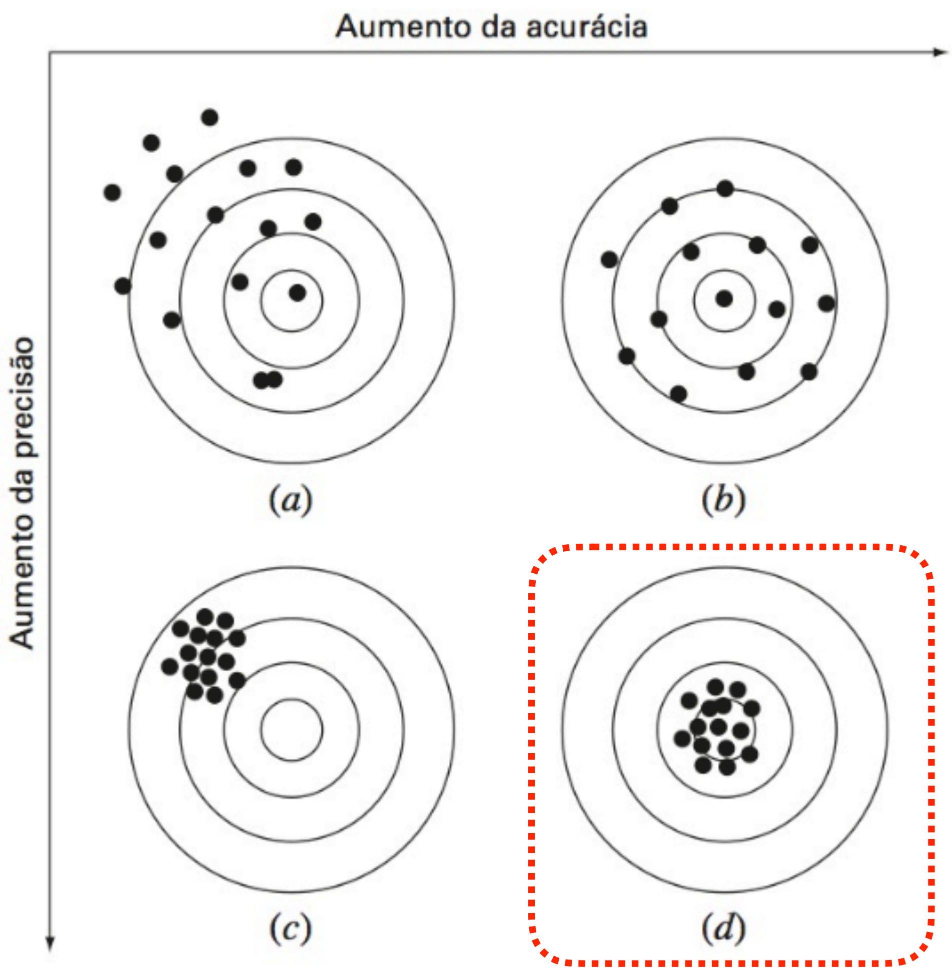
Acurado e impreciso

Acurácia e Precisão



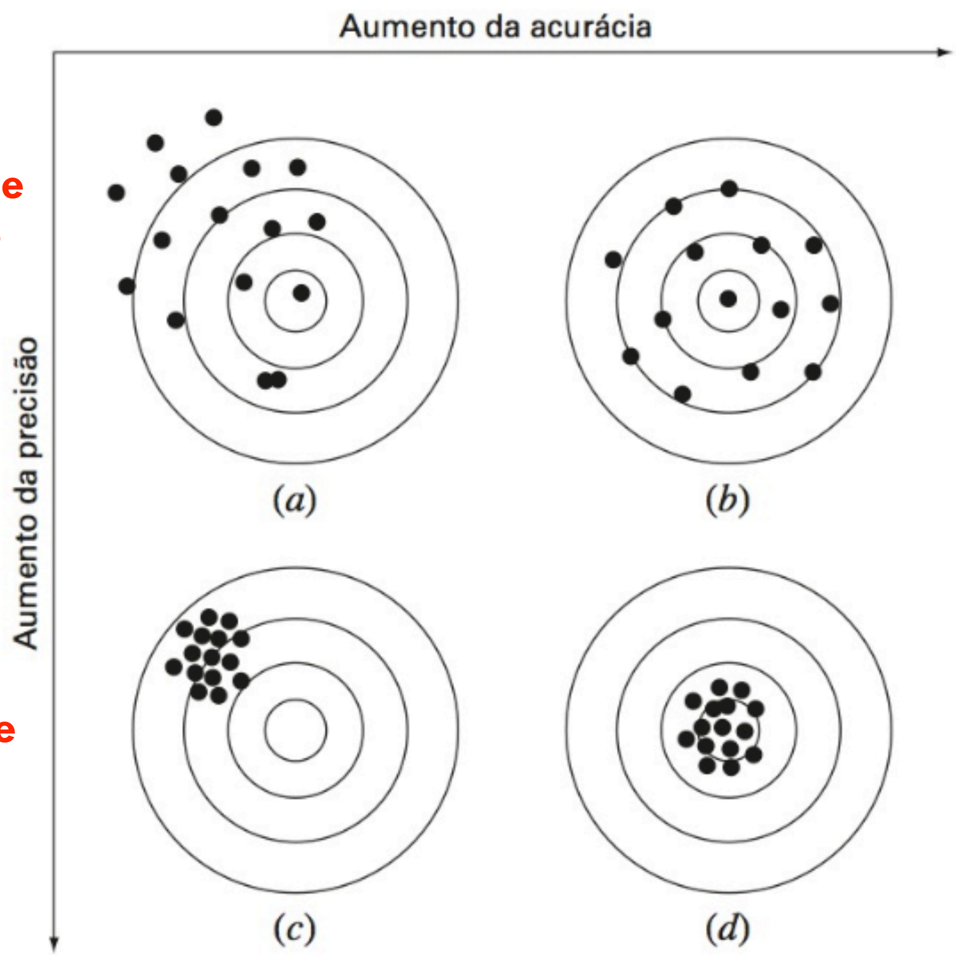
Inacurado e preciso

Acurácia e Precisão



Acurado e preciso

Acurácia e Precisão



Inacurado e impreciso

Acurado e impreciso

Inacurado e preciso

Acurado e preciso

Roteiro

1. Contextualização
2. Revisão
3. **Definições de Erro**
4. Algoritmo Iterativo básico
5. Tipos de erro
6. Referências

Definições de erro

valor verdadeiro = aproximação + **erro** (1)

Definições de erro

valor verdadeiro = aproximação + **erro** (1)

erro = valor verdadeiro - aproximação (2)

Definições de erro

valor verdadeiro = aproximação + **erro** (1)

erro = valor verdadeiro - aproximação (2)

[Erro exato] $\mathbf{E_t}$ = valor verdadeiro - aproximação (3)

Definições de erro

valor verdadeiro = aproximação + **erro** (1)

erro = valor verdadeiro - aproximação (2)

[Erro exato] E_t = valor verdadeiro - aproximação (3)

- **Problema:** considera a grandeza do problema

Definições de erro

valor verdadeiro = aproximação + **erro** (1)

erro = valor verdadeiro - aproximação (2)

[Erro exato] $\mathbf{E_t}$ = valor verdadeiro - aproximação (3)

[Erro relativo] $\mathbf{e_t}$ = erro verdadeiro / valor verdadeiro (4)

Definições de erro

valor verdadeiro = aproximação + **erro** (1)

erro = valor verdadeiro - aproximação (2)

[Erro exato] $\mathbf{E_t}$ = valor verdadeiro - aproximação (3)

[Erro relativo] $\mathbf{e_t}$ = erro verdadeiro / valor verdadeiro (4)

$\mathbf{e_t}$ = E_t / valor verdadeiro (5)

Definições de erro

valor verdadeiro = aproximação + **erro** (1)

erro = valor verdadeiro - aproximação (2)

[Erro exato] $\mathbf{E_t}$ = valor verdadeiro - aproximação (3)

[Erro relativo] $\mathbf{e_t}$ = erro verdadeiro / valor verdadeiro (4)

$\mathbf{e_t}$ = E_t / valor verdadeiro (5)

$\mathbf{e_t}$ = (E_t / valor verdadeiro) * 100 (6)

Exemplo 1

- Ponte possui:
  - uma estimativa x = 9.999 cm
  - valor verdadeiro = 10000 cm
- Rebite possui:
  - uma estimativa x = 9 cm
  - valor verdadeiro = 10 cm

- a) erro verdadeiro (**E_t**)?
- b) erro relativo percentual (**e_t**)?

Exemplo 1

```python
def erroVerdadeiro(val, x):
    Et = val - x
    return(Et)

def erroRelativo(val, Et):
    et = (Et/val) * 100
    return(et)
```

Exemplo 1

```
def erroVerdadeiro(val, x):
    Et = val - x
    return(Et)

def erroRelativo(val, Et):
    et = (Et/val) * 100
    return(et)

# ponte -> val = 10000, x = 9999
E_ponte = erroVerdadeiro(val = 10000, x = 9999)
e_ponte = erroRelativo(val = 10000, Et = E_ponte)
print("E_t = ", E_ponte, ", et = ", e_ponte, "%")

# rebite -> val = 10, x = 9
E_rebite = erroVerdadeiro(val = 10, x = 9)
e_rebite = erroRelativo(val = 10, Et = E_rebite)
print("E_t = ", E_rebite, ", et = ", e_rebite, “%")
```

Exemplo 1

```python
def erroVerdadeiro(val, x):
    Et = val - x
    return(Et)

def erroRelativo(val, Et):
    et = (Et/val) * 100
    return(et)

# ponte -> val = 10000, x = 9999
E_ponte = erroVerdadeiro(val = 10000, x = 9999)
e_ponte = erroRelativo(val = 10000, Et = E_ponte)
print("E_t = ", E_ponte, ", et = ", e_ponte, "%")

# rebite -> val = 10, x = 9
E_rebite = erroVerdadeiro(val = 10, x = 9)
e_rebite = erroRelativo(val = 10, Et = E_rebite)
print("E_t = ", E_rebite, ", et = ", e_rebite, "%")
```

Conclusão???

Exercícios

```
# Use os códigos do exemplo anterior para responder as seguintes
# questões?

# quais os valores de erros absolutos e relativos para a
aproximação x = 22/7 de pi?

# quais os valores de erros absolutos e relativos para a
aproximação x = 355/113 de pi?
```

Definições de erro

- Problemas reais:
  - valor verdadeiro só é conhecido ao lidar com funções de soluções analítica
  - normaliza o erro com base na melhor estimativa possível

Definições de erro

- Problemas reais:
  - valor verdadeiro só é conhecido ao lidar com funções de soluções analítica
  - normaliza o erro com base na melhor estimativa possível

[Erro relativo aproximado]

$$e_a = [(\text{solução\_atual} - \text{solução\_anterior}) / \text{solução\_atual}] * 100 \qquad (7)$$

Definições de erro

- Problemas reais:
  - valor verdadeiro só é conhecido ao lidar com funções de soluções analítica
  - normaliza o erro com base na melhor estimativa possível

[Erro relativo aproximado]

$\mathbf{e_a}$ = [(solução_atual - solução_anterior) / solução_atual] * 100 (7)

solução_atual > valor_verdadeiro → erro positivo

solução_atual < valor_verdadeiro → erro negativo

Definições de erro

- Problemas reais:
  - valor verdadeiro só é conhecido ao lidar com funções de soluções analítica
  - normaliza o erro com base na melhor estimativa possível

[Erro relativo aproximado]

e_a = | [(solução_atual - solução_anterior) / solução_atual] | * 100 (8)

solução_atual > valor_verdadeiro → erro positivo

solução_atual < valor_verdadeiro → erro negativo

Exercícios

```
# 1) defina uma função em python para computar o erro relativo
em função de um valor verdadeiro (val) e uma aproximação (x)

def erroRelativo2(val, x):
# … comandos …


# 2) defina uma função em python para computar o erro relativo
# absoluto de uma solução numérica, especifique também os
parâmetros necessários

def erroRelativoAbsoluto(…):
# … comandos …
```

Roteiro

Algoritmo iterativo básico

- Soluções numéricas:
  - aproximações sucessivas a partir de um palpite inicial
  - uso de laços de repetição
  - critérios de parada
    - número máximo de iterações
    - erro estimado aproximado < limiar

Algoritmo iterativo básico

```
function metodoIterativo(val, es, maxit)

  maxit = 1
  sol = val
  ea = 100

  do
    sol_old = sol
    sol     =  ... # calcula a solucao atual
    iter = iter + 1

    if sol != 0
      ea = abs((sol - sold_old)/sol) * 100

    if ea <= es || iter >= maxit
      exit
  end do

  return(sol)

end function
```

Algoritmo iterativo básico

```
function metodoIterativo(val, es, maxit)

  maxit = 1
  sol = val
  ea = 100

  do
    sol_old = sol
    sol      =  ... # calcula a solucao atual
    iter = iter + 1

    if sol != 0
      ea = abs((sol - sold_old)/sol) * 100

    if ea <= es || iter >= maxit
      exit
  end do

  return(sol)

end function
```

val: chute inicial

es: erro de parada

maxi: número de paradas

Algoritmo iterativo básico

```
function metodoIterativo(val, es, maxit)

  maxit = 1
  sol = val
  ea = 100

  do
    sol_old = sol
    sol     =  ... # calcula a solucao atual
    iter = iter + 1

    if sol != 0
      ea = abs((sol - sold_old)/sol) * 100

    if ea <= es || iter >= maxit
      exit
  end do

  return(sol)

end function
```

maxit: contador de iterações

sol: solução atual

ea: erro relativo aproximado

Algoritmo iterativo básico

```
function metodoIterativo(val, es, maxit)

  maxit = 1
  sol = val        sol: solução atual
  ea = 100

  do
    sol_old = sol
    sol     =  ... # calcula a solucao atual
    iter = iter + 1

    if sol != 0                                atualização do
      ea = abs((sol - sold_old)/sol) * 100     erro (iterativo)

    if ea <= es || iter >= maxit
      exit
  end do

  return(sol)

end function
```

Algoritmo iterativo básico

```
function metodoIterativo(val, es, maxit)

  maxit = 1
  sol = val
  ea = 100

  do
    sol_old = sol
    sol     =  ... # calcula a solucao atual
    iter = iter + 1

    if sol != 0
      ea = abs((sol - sold_old)/sol) * 100

    if ea <= es || iter >= maxit
      exit
  end do

  return(sol)

end function
```

sol: solução atual

verificam-se os critérios de parada do método

Exercícios

```
# 3) codifique o processo básico iterativo em python
# por enquanto, abstraia o calculo da solução atual, apenas
desenvolva o algoritmo genérico

def metodoIterativo(…):
```

Exercícios

```
# 3) codifique o processo básico iterativo em python
# por enquanto, abstraia o calculo da solução atual, apenas
desenvolva o algoritmo genérico

def metodoIterativo(…):

# 4) aproxime a função e^x por meio de uma adaptação do
# algoritmo iterativo anterior:
#
#                      e^x = sum 0,n (x^n / n!)
#
# Dentro da iteração calcule a solução atual por meio do comando
# sol = sol + x^iter /factorial(iter)

def eDeX(…):
```

Roteiro

1. Contextualização
2. Revisão
3. Definições de erro
4. Algoritmo Iterativo básico
5. **Tipos de erro**
6. Referências

Tipos de erro

Problemas reais

Aproximações

Erros

erros de arredondamento

erros de truncamento

incerteza dos dados

Métodos numéricos

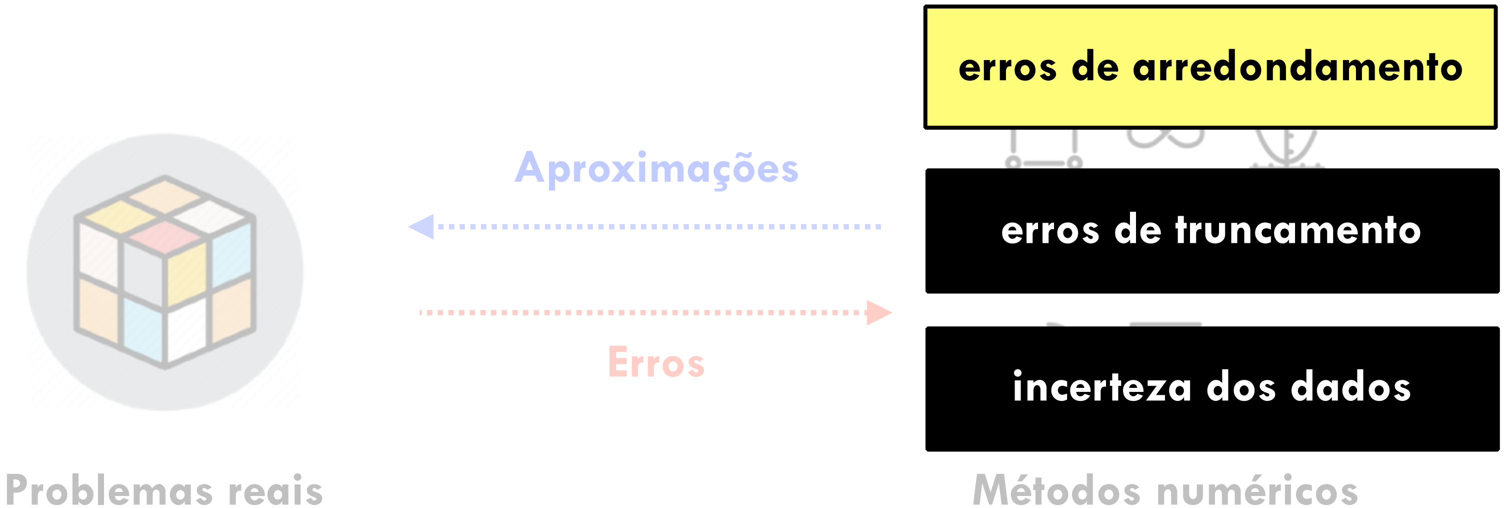
Tipos de erro

- erros de truncamento → aproximações para representar procedimentos matemáticos exatos

- erros de arredondamento → números com quantidade limitada de algarismos significativos são usados para representar números exatos

- incertezas dos dados → erros nos dados de entrada
  - medições, extrações dos dados

Tipos de erro



Erro de Arredondamento

- Quantidade fixa de algarismos significativo

- $\pi, e, \sqrt{7}$ não podem ser expressos por um número fixo de algarismos
  - não existe uma representação exata em computador
  - computador usa representação base 2, não podem representar precisamente certos números exatos na base 10

- discrepância introduzida pela omissão de algarismos significativos é chamado de erro de arredondamento.

Erro de Arredondamento

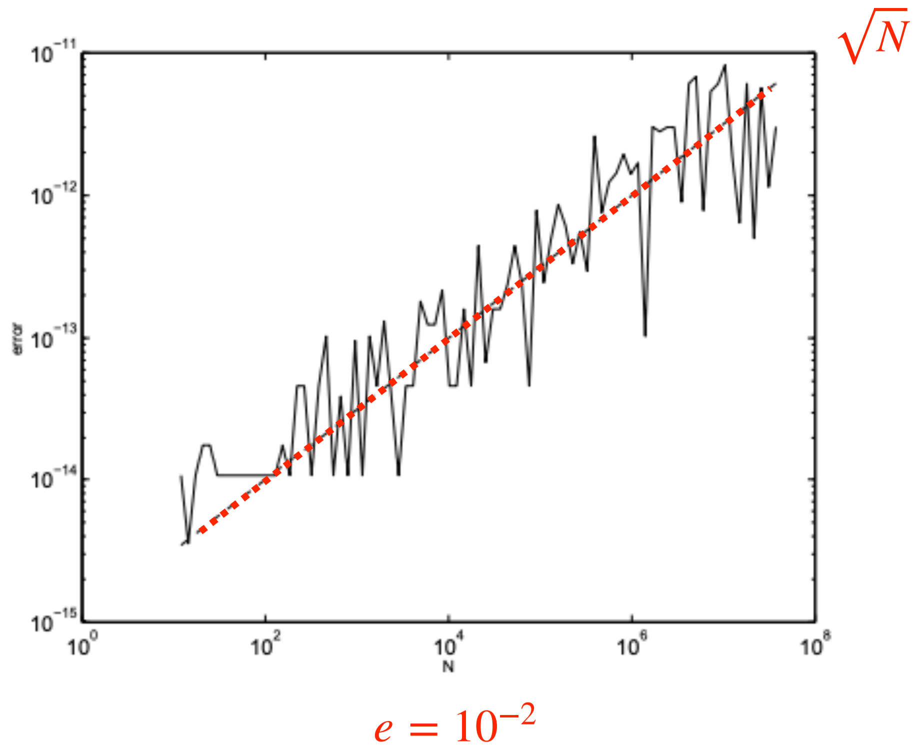


Figure 1. Roundoff error of a simple mathematical operation as a function of iteration number. The dashed line shows exact $\sqrt{N}$ behavior. The randomness is related to the randomness of roundoff error. This plot was generated using double precision arithmetic.

Erro de Arredondamento

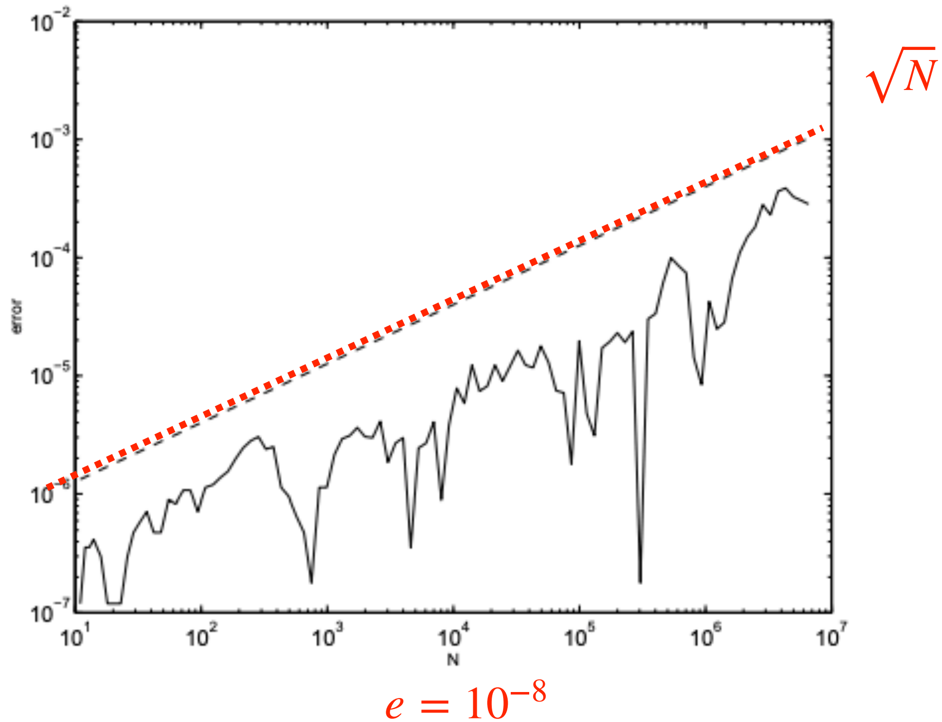
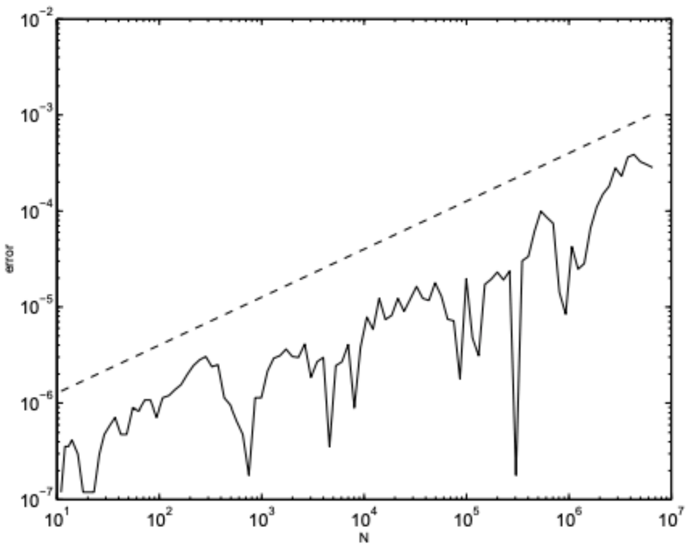
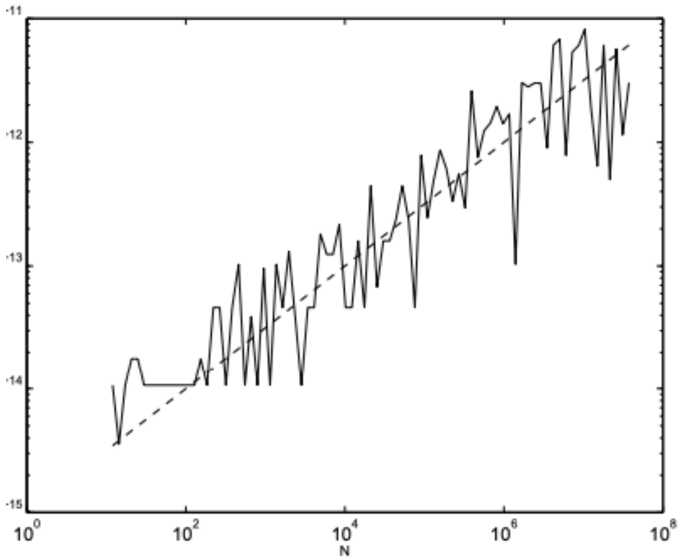


Figure 2. Roundoff error of a simple mathematical operation as a function of iteration number. The dashed line shows exact $\sqrt{N}$ behavior. The randomness is related to the randomness of roundoff error. This plot used single precision which truncates at $\epsilon = 10^{-8}$.

Erro de Arredondamento



$e = 10^{-8}$



$e = 10^{-2}$

Epsilon de máquina

- Devido ao limite de precisão da representação de números em ponto flutuante, existe um menor número representável que é maior do que 1.
  - 1 + eps
  - eps = epsilon de máquina
  - menor número que somado a 1 produz um resultado superior a 1 no sistema de numeração usado.

Epsilon de máquina

```python
# diferença de arredondamento de acordo com a representação
# escolhida

import numpy as np

print(np.finfo(float).eps)       #2.22044604925031e-16
print(np.finfo(np.float32).eps)  #1.1920929e-07
print(np.finfo(np.float64).eps)  #2.22044604925031e-16

# np.finfo().eps = The smallest representable positive number
# such that 1.0 + eps != 1.0.
# Type of eps is an appropriate floating point type.
```

Tipos de erro

Aproximações

Erros

Problemas reais

erros de arredondamento

erros de truncamento

incerteza dos dados

Métodos numéricos

Erro de Truncamento

- Ocorre pelo fato de que sempre estamos fazendo uma aproximação discreta de alguma função/problema contínuo
  - erro de discretização
  - introduzido pelas aproximações que vamos realizando, não pelo computador

Erro de Truncamento

- Ocorre pelo fato de que sempre estamos fazendo uma aproximação discreta de alguma função/problema contínuo
  - erro de discretização
  - introduzido pelas aproximações que vamos realizando, não pelo computador

- quando aproximamos um conceito matemático formado por uma sequência infinita de passos por um procedimento finito.
  - integral: limite de somas
  - numericamente → soma finita

Erro de Truncamento

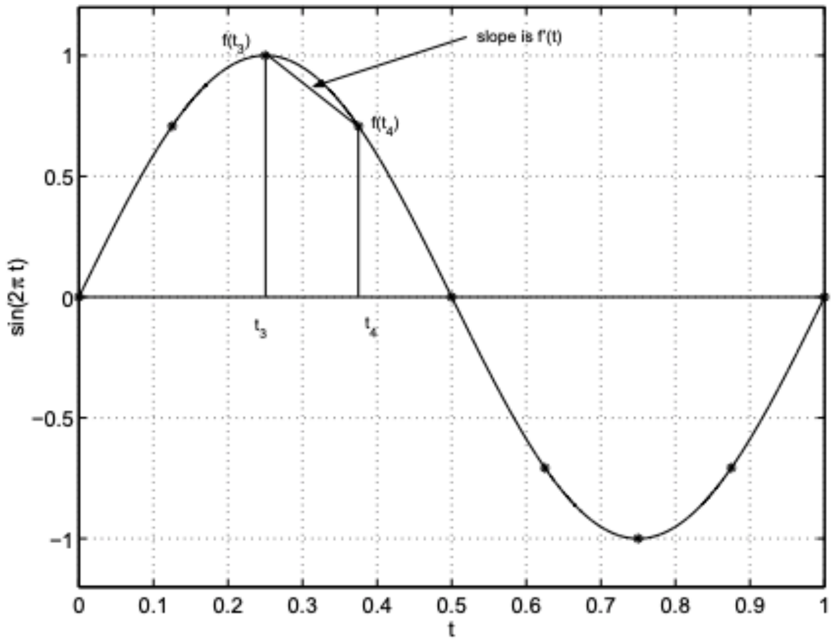


Figure 4. Graphical representation of the numerical derivative. The true function is $f(t) = sin(2\pi t)$, but we only have taken 8 samples at equally spaced intervals during one period of this sine wave. To compute the derivative, we compute the slope between adjacent sample points.

Erro de Truncamento

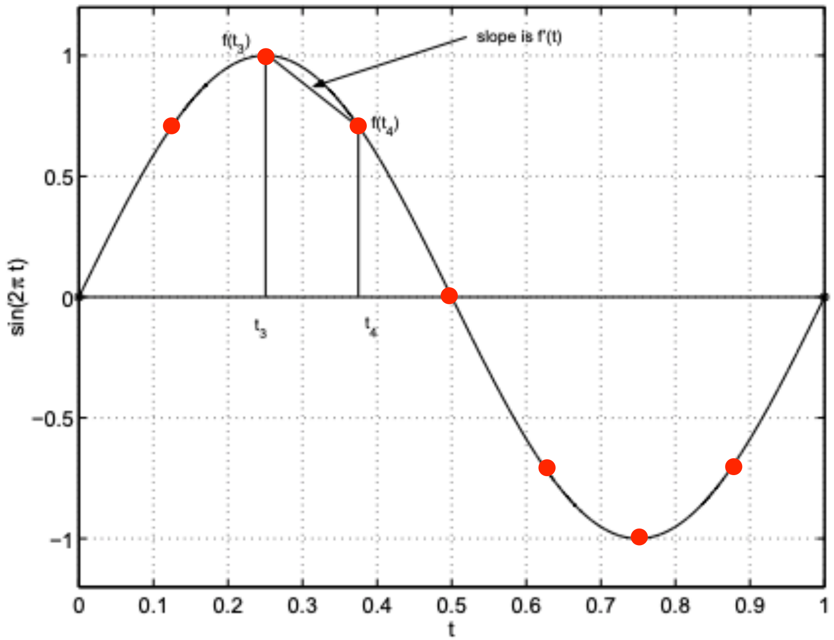


Figure 4. Graphical representation of the numerical derivative. The true function is $f(t) = sin(2\pi t)$, but we only have taken 8 samples at equally spaced intervals during one period of this sine wave. To compute the derivative, we compute the slope between adjacent sample points.

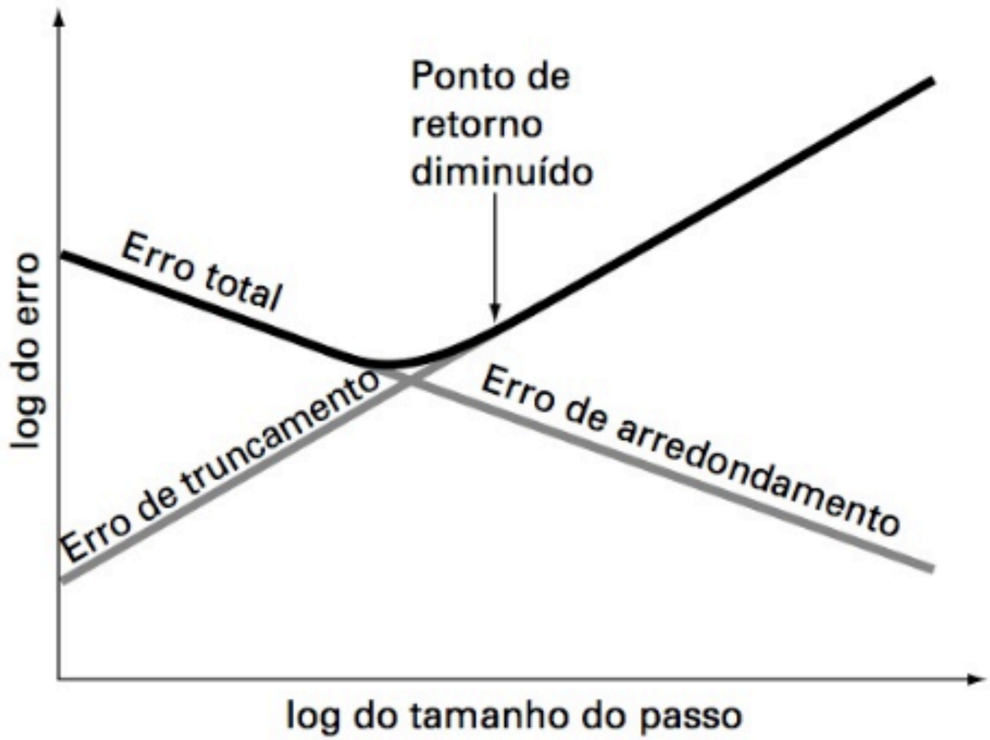
Erro Numérico Total

- soma do erro de truncamento + erro de arredondamento
    - minimizar arredondamento = aumentar algarismos
    - minimizar truncamente = diminuir o tamanho do passo

- estratégia para diminuir um tipo de erro aumenta o outro
- prática
    - computadores usam algarismos suficientes para que os erros de arredondamento não predominem

Erro Numérico Total

FIGURA 4.8
Uma descrição gráfica do balanço entre erros de arredondamento e de truncamento que ocorrem algumas vezes na utilização de um método numérico. É mostrado o ponto de retorno diminuído, no qual o erro de arredondamento começa a neutralizar as vantagens da redução do tamanho do passo.



Exercícios

```
# 5) Considere as expressões:
```

$$\frac{exp(1/u)}{1 + exp(1/u)} \qquad \frac{1}{exp(-1/u) + 1}$$

```
com u > 0. Teste cada uma delas para u = [0.1, 0.001]. Faça
incrementos de 0.001 por vez. Qual dessas expressões é mais
adequada quando u é pequeno? Porque?
```

Exercícios

```
#  Desenvolva um programa bem estruturado em python para
calcular a expansão em série de MacLaurin da função cosseno,
como descrito abaixo:
```

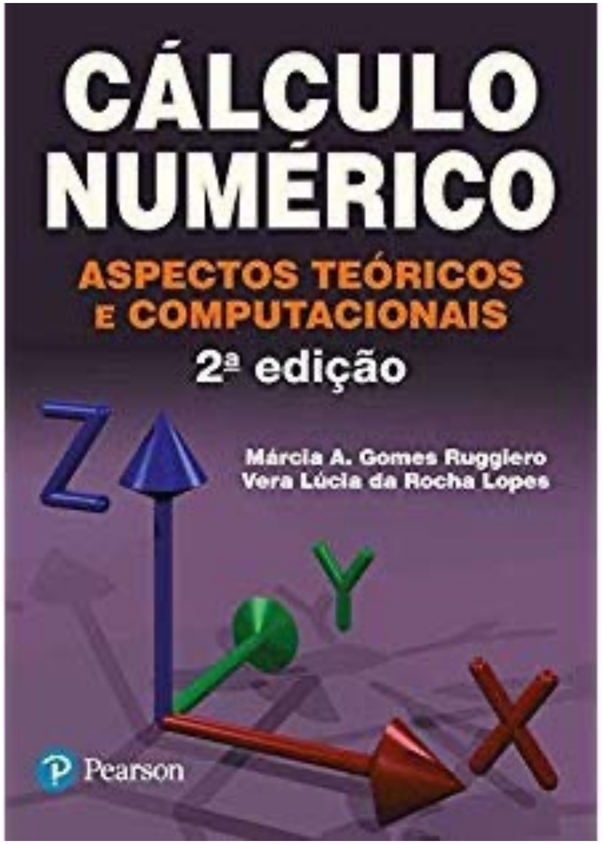
$$cos(x) = 1 - \frac{x^2}{2} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} - \ldots$$

```
A função deve ter as seguintes características:

* Iterar até que o erro relativo caia abaixo de um critério de
parada (es) ou exceda um número máximo de iterações (itmax).
Permita que o usuário especifique esses parâmetros.

* inclua valores padrão de es (=0,000001) e itmax(=100) para o
caso deles não serem especificados pelo usuário.

* devolver a estimativa de cos(x), o erro relativo aproximado, o
número de iterações e o erro relativo verdadeiro (que pode ser
calculado com base na função cosseno do pyton).
```

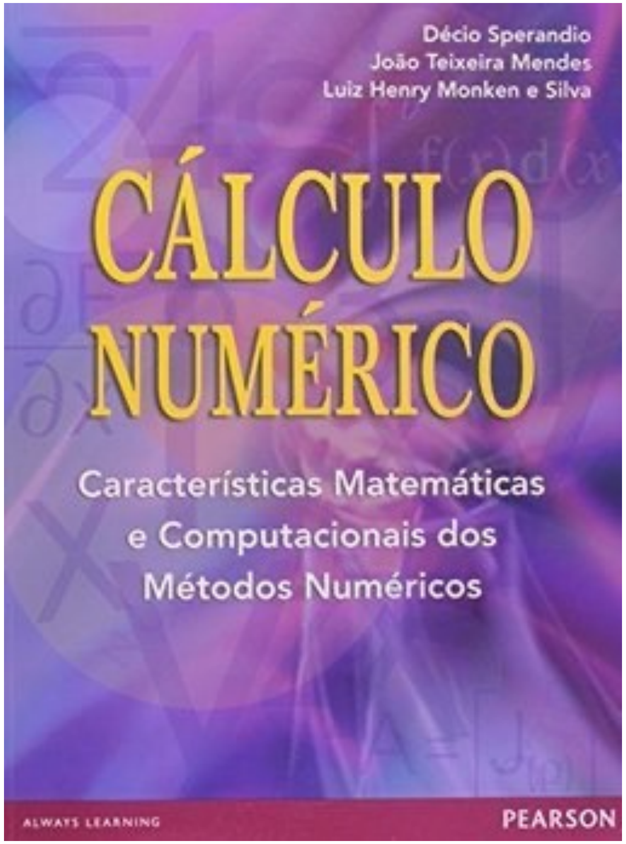
Roteiro

1. Contextualização
2. Revisão
3. Definições de erro
4. Algoritmo Iterativo básico
5. Tipos de erro
6. Referências

Referências sugeridas

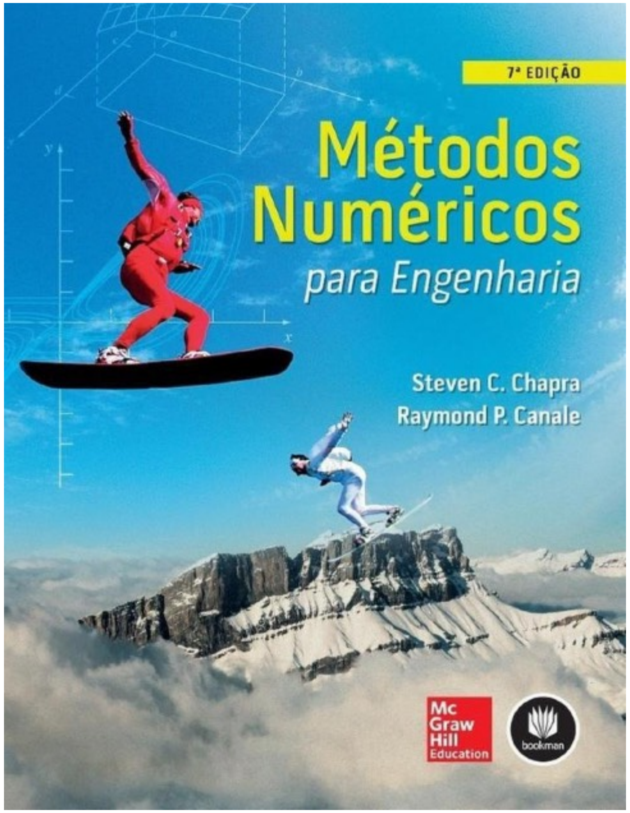


[RUGGIERO & LOPES, 1997]



[SPERANDIO et al, 2003]

Referências sugeridas



[CHAPRA & CANALE, 2016]

Perguntas?

Prof. Rafael G. **Mantovani**

rafaelmantovani@utfpr.edu.br