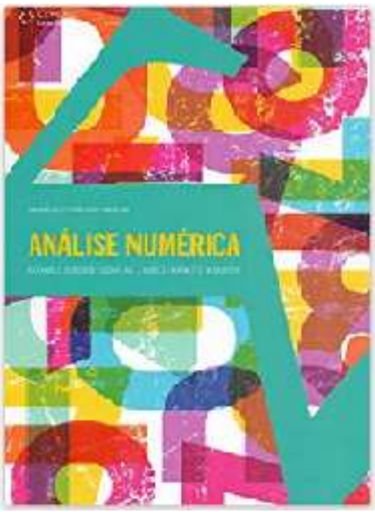
**FACULDADE DE COMPUTAÇÃO E INFORMÁTICA**
**BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO**
**ANÁLISE NUMÉRICA – Aula 02 – 2º SEMESTRE/2019**
**PROF. Jamil Kalil Naufal Júnior**

## TEORIA: ALGORITMOS NUMÉRICOS

Nossos **objetivos** nesta aula são:

- Conhecer o conceito de algoritmo numérico.
- Conhecer os conceitos de complexidade e convergência de algoritmos numéricos.
- Praticar com cálculo de complexidade e convergência de algoritmos numéricos.




Para esta semana, usamos como referência as **Seções 1.3 (Algoritmos e Convergência) e 1.4 (Software Numérico)** do nosso livro da referência básica:

BURDEN, R.L., FAIRES, J.D. **Análise Numérica**. 10.ed. São Paulo: Cengage Learning, 2017.

*Não deixem de ler estas seções depois desta aula!*

---

## ALGORITMOS NUMÉRICOS E COMPLEXIDADE

- **Algoritmos (ou métodos) numéricos** são esquemas que, normalmente, buscam por **soluções numéricas exatas ou aproximadas de problemas**.

- O estudo destes algoritmos é, geralmente, realizado por duas disciplinas: **Cálculo Numérico** e **Análise Numérica**. Em **Cálculo Numérico**, a preocupação essencial é o desenvolvimento do próprio algoritmo, sem preocupações com complexidade e convergência. Em Análise Numérica, além da **preocupação com o algoritmo**, as questões de **quão eficiente é o algoritmo** em termos de **complexidade** e **convergência** tornam-se importantes.

- A **descrição de um algoritmo numérico** pode ser feita, por exemplo, através de **pseudocódigo**. Geralmente, um algoritmo numérico possui uma ou mais entradas e uma ou mais saídas. O **cálculo da complexidade do algoritmo** é feito, normalmente, através de **análise assintótica**, com **cota superior** $\boldsymbol{O}(\cdot)$ e, se necessário, com **cota inferior** $\boldsymbol{\Omega}(\cdot)$.

## EXERCÍCIO TUTORIADO

1. Descreva, em pseudocódigo, um algoritmo para calcular a soma abaixo. Calcule a complexidade com cota superior deste algoritmo.

$$\sum_{i=1}^{N} x_i = x_1 + x_2 + \cdots + x_N$$

<u>SOLUÇÃO</u>

ENTRADA $\quad N, x_1, x_2, \ldots, x_n.$

SAÍDA $\quad \text{SOMA} = \sum_{i=1}^{N} x_i.$

Passo 1 Faça SOMA = 0. *(Inicializar o acumulador.)*

Passo 2 Para i = 1, 2, ..., N execute

Faça SOMA = SOMA + $x_i$. *(Adicionar o próximo termo.)*

Passo 3 SAÍDA (SOMA);

Pare.

2. Descreva, em pseudocódigo, um algoritmo para calcular uma aproximação da função

$$f(x) = \ln(1+x)$$

utilizando a expansão em Série de Taylor truncada mostrada a seguir. Calcule a complexidade com cota superior deste algoritmo.

$$\ln\ (1+x)\ \approx \sum_{i=1}^{N} \frac{(-1)^{i+1}x^i}{i} = x - \frac{x^2}{2} + \frac{x^3}{3} - \cdots + \frac{(-1)^{N+1}x^N}{N}$$

Ou, em torno de $x_0 = 1$

$$P_n\ (x)\ \approx \sum_{i=1}^{N} \frac{(-1)^{i+1}(x-1)^i}{i} = (x-1) - \frac{(x-1)^2}{2} + \frac{(x-1)^3}{3} - \cdots + \frac{(-1)^{N+1}(x-1)^N}{N}$$

- Para x = 1,5 obtemos ln(1,5) = 0,40546511 com 8 casas decimais.
- Queremos calcular o mínimo valor de N para $|\ln(1,5) - P_N(1,5)| < 10^{-5}$ sem usar o resto do polinômio de Taylor.
- Do Cálculo sabemos que se $\sum_{n=1}^{\infty} a_n$ é uma série alternada com limite A, cujos termos decrescem em módulo, então A e a n-ésima soma parcial $A_N = \sum_{n=1}^{N} a_n$ diferem por menos do que o módulo do termo (N + 1), isto é:

$$|A - A_N| \le |a_{n+1}|$$

- O algoritmo abaixo usa esse limitante:

ALGORITMO PARA O LOGARITMO NATURAL EXPANDIDO EM SÉRIE DE TAYLOR

```
ENTRADA     valor x, tolerância TOL, número máximo de iterações M.
SAÍDA       Grau N do polinômio ou mensagem de erro.

Passo 1     Faça N = 1;
            y = x – 1;
            SOMA = 0;
            Potência = y;
            Termo = y;
            Sinal = -1.      (Usado para implementar a alternância de sinal.)

Passo 2     Enquanto N ≤ M execute passos 3 a 5.
    Passo 3     Faça Sinal = - Sinal;              (Alterna o sinal.)
                SOMA = SOMA + Sinal.Termo;         (Acumula termo.)
                Potência = Potência.y;
                Termo = Potência/(N + 1).          (Calcula o próximo termo.)
    Passo 4     Se |Termo| < TOL  então            (Teste de precisão.)
                    SAÍDA (N);
                    PARE.                          (O procedimento foi bem sucedido.)
Passo 5     Faça N = N + 1.                        ( Preparar para a próxima iteração.)
Passo 6     SAÍDA ('O método falhou')              (O procedimento foi mal sucedido.)
            PARE.
```

- A entrada do nosso problema é x = 1,5, TOL = $10^{-5}$ e talvez M = 15. Essa escolha de M fornece um limitante superior para a quantidade de cálculos que estamos dispostos a efetua, reconhecendo que é possível que o algoritmo falhe se esse limitante for ultrapassado. Se a saída será um valor de N ou uma mensagem de erro, depende da precisão do dispositivo de cálculo.

## CONVERGÊNCIA DE ALGORITMOS NUMÉRICOS

- Além do aspecto da própria complexidade do algoritmo numérico, saber o **quão rápido ele se aproxima (converge) de (para) um valor desejado** é uma **medida** importante associada ao algoritmo.

- Geralmente, os algoritmos numéricos geram valores **intermediários nas aproximações**, que podem ser vistos como **sequências de números reais**.

- **Definição:** Suponha que se saiba que a sequencia $\{\boldsymbol{\beta_n}\}_{\boldsymbol{n=1}}^{\infty}$ **convirja para zero** e que $\{\boldsymbol{\alpha_n}\}_{\boldsymbol{n=1}}^{\infty}$ **convirja para um número real $\boldsymbol{\alpha}$**. Se existir uma constante $K$ positiva tal que

  $$|\boldsymbol{\alpha_n} - \boldsymbol{\alpha}| \leq \boldsymbol{K}|\boldsymbol{\beta_n}|$$

  dizemos que a sequência $\{\alpha_n\}_{n=1}^{\infty}$ converge para $\alpha$ com **taxa de convergência** $O(\beta_n)$. Normalmente, indica-se esta convergência por $\boldsymbol{\alpha_n} = \boldsymbol{\alpha} + \boldsymbol{O}(\boldsymbol{\beta_n})$

- Na prática, normalmente se utiliza $\boldsymbol{\beta_n} = \frac{\boldsymbol{1}}{\boldsymbol{n^p}}$ , para algum **número real $\boldsymbol{p} > \boldsymbol{0}$**. Assim, no cálculo de taxa de convergência de um algoritmo numérico, estamos interessados no **maior valor de $\boldsymbol{p}$** com $\boldsymbol{\alpha_n} = \boldsymbol{\alpha} + \boldsymbol{O}\left(\frac{\boldsymbol{1}}{\boldsymbol{n^p}}\right)$.

## EXERCÍCIO COM DISCUSSÃO EM DUPLAS

3. Considere dois algoritmos numéricos $\alpha_n$ e $\hat{\alpha}_n$, cujas sequências de aproximação sejam dadas por:

$$\alpha_n = \frac{n+1}{n^2} \quad e \quad \hat{\alpha}_n = \frac{n+3}{n^3}\ , para\ n \geq 1$$

   (a) Calcule o limite destas duas sequências e mostre qual o valor para onde elas convergem.

   (a) Embora tanto $\lim_{n\to\infty} \alpha_n = 0$ quanto $\lim_{n\to\infty} \hat{\alpha}_n = 0$, a sequência $\{\hat{\alpha}_n\}$ converge para esse limite muito mais rapidamente que a sequência $\{\alpha_n\}$.

(b) Preencha a tabela com os valores abaixo e verifique qual destes dois algoritmos parece convergir mais rápido para 0.

| $n$ | $n=1$ | $n=2$ | $n=3$ | $n=4$ | $n=5$ | $n=6$ | $n=7$ |
|---|---|---|---|---|---|---|---|
| $\alpha_n$ | | | | | | | |
| $\hat{\alpha}_n$ | | | | | | | |

| Algoritmo | $n=1$ | $n=2$ | $n=3$ | $n=4$ | $n=5$ | $n=6$ | $n=7$ |
|---|---|---|---|---|---|---|---|
| $\alpha_n$ | 2,00000 | 0,75000 | 0,44444 | 0,31250 | 0,24000 | 0,19444 | 0,16327 |
| $\hat{\alpha}_n$ | 4,00000 | 0,62500 | 0,22222 | 0,10938 | 0,064000 | 0,041667 | 0,029155 |

## EXERCÍCIO TUTORIADO

4. Calcule as taxas de convergência dos dois algoritmos anteriores e mostre qual deles converge mais rápido.

$$\alpha_n = \frac{n+1}{n^2} \quad e \quad \hat{\alpha}_n = \frac{n+3}{n^3}\ , para\ n \geq 1$$

(c) Resolvendo primeiro para $\alpha_n$

Queremos verificar se existe um K > 0, tal que $|\alpha_n - \alpha| \leq K\beta_n$.

$$\lim_{n\to\infty} \alpha_n = \frac{n+1}{n^2} = \frac{n}{n^2} = \frac{1}{n} = 0$$

Definimos a sequência $\{\beta_n\} = \frac{1}{n}$. Então:

$$|\alpha_n - 0| = \frac{n+1}{n^2} \leq \frac{n+n}{n^2} = \frac{2n}{n^2} = 2\frac{1}{n} = 2\beta_n$$

Logo, K = 2. Como conseguimos um K > 0 que satisfaça a inequação acima, podemos dizer que a taxa de convergência de $\{\alpha_n\}$ a zero é similar à taxa de convergência de {1/n} a zero, ou seja:

$$\alpha_n = 0 + O(\frac{1}{n})$$

Resolvendo para $\hat{\alpha}_n$:

Queremos verificar se um K > 0, existe tal que $|\hat{\alpha}_n - \hat{\alpha}| \leq K\hat{\beta}_n$

$$\lim_{n\to\infty} \hat{\alpha}_n = \frac{n+3}{n^3} = \frac{n}{n^3} = \frac{1}{n^2} = 0$$

Definimos a sequência $\widehat{\{\beta_n\}} = \frac{1}{n^2}$. Então:

$$|\hat{\alpha}_n - 0| = \frac{n+3}{n^3} \leq \frac{n+3n}{n^3} = \frac{4n}{n^3} = 4\frac{1}{n^2} = 4\hat{\beta}_n$$

Logo, K = 4. Como conseguimos um K > 0 que satisfaça a inequação acima, podemos dizer que a taxa de convergência de $\{\hat{\alpha}_n\}$ a zero é similar à taxa de convergência de {1/n²} a zero, ou seja:

$$\hat{\alpha}_n = 0 + O(\frac{1}{n^2})$$

1. Descreva, em pseudocódigo, um algoritmo para calcular uma aproximação da função

$$f(x) = e^x$$

utilizando a expansão em Série de Taylor truncada mostrada abaixo. Calcule a complexidade com cota superior deste algoritmo.

$$e^x \approx \sum_{i=0}^{N} \frac{x^i}{i!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} - \cdots + \frac{x^N}{N!}$$

2. Utilizando o seu algoritmo, calcule aproximações para $e^3$ com N=5 e N=10, utilizando aritmética de arredondamento com 4 casas decimais. Para qual valor de N (5 ou 10), a aproximação pareceu ser mais precisa ? Por quê ?

3. Determine a taxa de convergência de cada uma das sequencias abaixo:

   a. $\alpha_n = sen\ \frac{1}{n}\ , para\ n \geq 1$

   b. $\alpha_n = \frac{sen\ n}{n}\ , para\ n \geq 1$

   c. $\alpha_n = \frac{1-e^n}{n}\ , para\ n \geq 1$