# INTRODUÇÃO

## Métodos Numéricos

O Cálculo Numérico, entendido com uma coletânea de métodos numéricos, consiste de uma poderosa ferramenta que nos auxilia na obtenção de soluções numéricas, em geral aproximadas, de diversos problemas que encontramos no mundo real. Por um método numérico entendemos um conjunto de regras escritas sob a forma de uma sequencia de operações elementares (soma, adição, multiplicação e divisão) que levam a uma solução do problema. A esse conjunto de regras denominamos algoritmo.

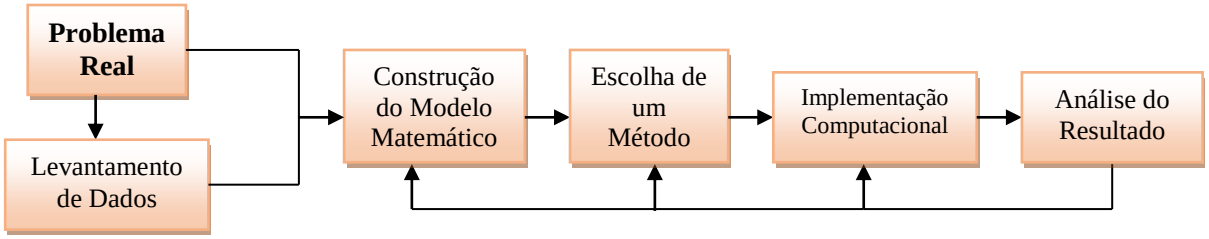
Com a popularização de computadores de baixo custo e de alta capacidade de processamento, muitas das atividades da Ciência, em particular as das Engenharias, têm feito uso cada vez mais intensivo dos métodos e técnicas computacionais na resolução de problemas reais, para os quais as soluções manuais são impraticáveis, imprecisas, ou ainda, são muito custosas em relação ao tempo de execução.

Desta forma, o uso do computador como ferramenta de trabalho de cálculo numérico requer o entendimento dos seus princípios de operação e de como eles interferem nos resultados obtidos. Geralmente, é aceito como verdade que computadores não erram e que são os usuários é que cometem enganos que levam ao mau funcionamento do computador. Na realidade, o computador, como dispositivo de cálculo numérico também "comete" erros devido às suas características intrínsecas e o papel do usuário é quantificar esses erros e encontrar formas de, se não eliminá-los, pelo menos minimizá-los.

O desenvolvimento e aplicação dos métodos numéricos estão vinculados ao desenvolvimento computacional. Portanto os fatores relevantes para a escolha de um algoritmo devem envolver os aspectos:

i. Precisão desejada;
ii. Capacidade do método em conduzir aos resultados desejados (velocidade de convergência);
iii. Esforço computacional utilizado (tempo de processamento, economia de memória necessária para a resolução).

O processo de modelagem matemática para resolver problemas reais pode ser visto pelas seguintes etapas:



A **implementação computacional** consiste em, escolhida uma linguagem computacional (Fortran, Pascal, C, C++, dentre outras), colocar o algoritmo elaborado em termos dessa linguagem.

Pode-se também escolher vários métodos numéricos para comparar os resultados obtidos e assim optar pelo uso do que apresentou melhor resultado.

**Exemplo:**

Um fabricante de plásticos produz 2 tipos de plástico: o normal e o especial. Cada tonelada de plástico normal exige 2 horas na máquina A e 5 horas na máquina B; cada tonelada de plástico especial exige 2 horas na máquina A e 3 horas na máquina B. Se a máquina A está disponível 8 horas por dia e a máquina B está disponível 15 horas por dia, quantas toneladas de cada tipo de plástico deveriam ser produzidas diariamente de maneira que as duas máquinas se mantenham totalmente ocupadas?

**(Construção do modelo matemático - Modelagem)**

$$\left.\begin{array}{l}\text{plástico normal = variável } x\\ \text{plástico especial = variável } y\end{array}\right\} \Rightarrow \begin{cases}2x + 2y = 8\\ 5x + 3y = 15\end{cases}$$

**(Escolha de um método numérico -** Substituição de variáveis)

$$\begin{array}{l}2x + 2y = 8\\ 5x + 3y = 15\end{array} \Rightarrow \begin{array}{l}2x + 2y = 8\\ -4y = -10\end{array} \Rightarrow \begin{cases} -4y = -10 \Rightarrow y = 2.5\\ 2x + 2(2.5) = 8 \Rightarrow 2x = 8 - 5 \Rightarrow x = 1.5\end{cases}$$

**(Análise dos resultados obtidos)**

Devem ser produzidas 1.5 toneladas de plástico normal (variável $x$) e 2.5 toneladas de plástico especial (variável $y$).

## Algoritmos Numéricos

São algoritmos voltados ao processamento numérico, isto é, as operações aritméticas formam o núcleo do algoritmo e seu objetivo é obter um ou mais resultados numéricos.

Um algoritmo numérico de boa qualidade deve possuir as seguintes características:

1. ***Inexistência de erro lógico***
   Previsão completa de todas as tendências do processo, levando em conta as características das operações aritméticas e dos modelos matemáticos. O algoritmo deve identificar todas as etapas do modelo.

2. ***Inexistência de erro operacional*** (*overflow/underflow*)
   O algoritmo pode falhar por violar restrições físicas da máquina, e então os erros são detectados em tempo de execução.

$$y \Rightarrow \begin{cases} |y| < \text{limite inferior (underflow)} \\ |y| > \text{limite superior (overflow)} \end{cases}$$

   Exemplo:

   Seja $z = x + iy \in C$; $x$ e $y \in \Re$. Queremos calcular $|z| = \sqrt{x^2 + y^2}$.

   Se num algoritmo implementarmos diretamente $|z| = \sqrt{x^2 + y^2}$, dependendo dos valores de $x$ e $y$ podemos ter overflow em $x^2$ ou $y^2$, embora valha $\sqrt{x^2 + y^2}$ < limite superior. Então devemos fazer:

$$|z| = \sqrt{x^2 + y^2} = \sqrt{x^2\left(1 + \frac{y^2}{x^2}\right)} = |x|\sqrt{1 + \left(\frac{y}{x}\right)^2}, \ \forall |x| > |y|$$

   E assim,

$$|z| = |x|\sqrt{1 + \left(\frac{y}{x}\right)^2}, \ \forall |x| > |y|$$

3. ***Quantidade finita de cálculos*** (critério de parada)
   Como muitos problemas numéricos são resolvidos por métodos iterativos (um método é dito iterativo quando o mesmo procedimento é repetido diversas vezes gerando uma sequencia de aproximações da solução procurada.), é necessário estabelecer um critério de parada, para que o algoritmo possa terminar após um número finito de operações. Ainda é aconselhável estabelecer um número máximo de iterações a serem executadas pelo algoritmo.

4. ***Existência de um critério de exatidão***
   Em virtude das limitações de precisão e exatidão da máquina e do método, todo resultado obtido do computador deverá enquadrar-se em um critério de exatidão fornecido de antemão.

   *resultado = valor aproximado ± limite do erro (precisão)*

5. ***Independência da máquina***
   Nos algoritmos numéricos não deve haver dados dependentes da máquina. O problema gerado pelo algoritmo deve ser executado em diferentes máquinas, visando à máxima portabilidade.

   Exemplo: *e* = *exp*(1) e não *e* = 2.71828182.

6. Com precisão infinita, os limites do erro devem convergir a zero (convergência numérica).
   Esta exigência estabelece a dependência entre a solução ideal em $\Re$ e a solução da máquina. Sem essa condição de convergência a solução da máquina não precisará necessariamente estar relacionada com a solução verdadeira.

Exemplo: Dado $a \in \Re$, calcular x = sin(a)

*Inicio*
*ler (a)*
$x \leftarrow 0 \pm 1$
*imprimir (x)*
*Fim*

Este algoritmo satisfaz todas as exigências vistas até o momento:

- Não há erro lógico nem operacional
- Os dados não dependem da máquina
- Resultado dentro dos limites de erro

Porém não há **convergência numérica**.

7. ***Eficiência***

Encontrar a solução de um problema visando obter economia de recursos envolvidos (tempo, exatidão, volume de dados de referência, dificuldades de representação, espaço de memória).

As exigências (1) e (2) dizem respeito à eficácia.

Eficácia: qualidade de produzir uma resposta correta ao problema dado.
Eficiência: eficácia + economia.

# ERROS

Na busca da solução do modelo matemático por meio de cálculo numérico, os erros surgem de várias fontes e merecem cuidado especial. Caso contrário, pode-se chegar a resultados distantes do que se esperaria ou até mesmo obter outros que não têm nenhuma relação com a solução do problema original.

As principais fontes de erros são:

**i)** Erros nos dados de entrada;
**ii)** Erros no estabelecimento do modelo matemático;
**iii)** Erros de arredondamentos durante a computação;
**iv)** Erros de truncamento;
**v)** Erros humanos e de máquinas.

O modelo matemático para o problema real deve traduzir e representar o fenômeno que está ocorrendo no mundo físico. Entretanto, nem sempre isso é fácil. Normalmente, são necessárias simplificações no modelo físico para se obter um modelo matemático que fornecerá uma solução para o problema original. As simplificações realizadas são fontes de erros, o que pode implicar a necessidade de reformulação do modelo físico e matemático. Ainda, frequentemente os dados que são analisados na busca de um modelo matemático que o represente são obtidos através de medidas experimentais, e, portanto sujeito a imprecisões.

## Erros na fase de modelagem

Ao se tentar representar um fenômeno do mundo físico por meio de um modelo matemático, raramente se tem uma descrição correta deste fenômeno. Normalmente, são necessárias várias simplificações do mundo físico para que se tenha um modelo matemático com o qual se possa trabalhar. Simplificações estas como a resistência do ar, a velocidade do vento, variações de temperatura, umidade relativa do ar e outras. Ainda, frequentemente os dados que são analisados na busca de um modelo matemático que o represente são obtidos através de medidas experimentais, e portanto, sujeito a imprecisões.

Desta forma notamos uma forte influência do modelo matemático e da precisão dos dados na confiabilidade da resposta obtida.

## Erros na fase de resolução

Na execução de determinado método numérico são utilizados, em geral, computadores, calculadoras que trabalham com uma representação finita dos números. No entanto operações que envolvam números que não possam ser representados através de um número finito de dígitos não fornecerão como resultado um valor exato. Por exemplo: Dentro de algum cálculo deve-se utilizar o valor da fração 1/3, cuja representação na forma decimal gera uma dízima periódica 0,333333...

A representação de um número depende da base escolhida ou disponível na máquina em uso, e do número máximo de dígitos usados na sua representação.

A base decimal [0..9] é a que mais empregamos no dia a dia. Já os computadores operam com base binária [0 e 1]. Assim na relação homem/máquina ocorrem as seguintes conversões:

- O usuário (homem) fornece os dados no sistema decimal;
- Todos os dados recebidos (pela máquina) são convertidos para binário;
- Os resultados obtidos (pela máquina) são convertidos para decimal e finalmente transmitidos ao usuário.

Este processo de conversão é uma fonte de **erros**.

A escolha do sistema binário se deve ao fato que internamente, na máquina, um número é representado por um sequência de pulsos elétricos que indicam 2 estados: *zero* e *um*.

## Erros de arredondamento

Os erros de arredondamento surgem de duas fontes distintas:

1. No processo de conversão de base
   Um número pode ter representação finita em uma base e não finita em outra.

2. Na representação finita de dígitos que as máquinas utilizam.

Assim, esses erros dependem de como os números são representados na máquina, e a representação, por sua vez, depende da base em que esses números são escritos e da quantidade máxima de dígitos usados nessa representação. Quanto maior o número de dígitos utilizados após o ponto, maior será a precisão.

**Exemplo:**

Arredondamento com 4 casas decimais:
2.39786
2.39783

## Erros de truncamento

É o erro inerente ao método numérico. Surge cada vez que se substitui um procedimento matemático infinito por um processo finito ou discreto. Um exemplo de erro de truncamento é o da aproximação de uma função pela série de Taylor, tais como funções trigonométricas, logarítmicas, exponenciação, etc. Nestes casos as séries possuem infinitos termos, porém é usual fixarmos a quantidade de termos que serão considerados.

Tanto erros por arredondamento como por truncamento fazem com que num processo iterativo o erro se propague, podendo conduzir a resposta obtida a um valor diferente da resposta esperada (correta).

Apesar de incorrer em erros menores, o uso do arredondamento acarreta em tempo maior de execução. Por essa razão o erro de truncamento é usado mais frequentemente.

## Erros absolutos e relativos

A partir do momento em que se calcula um resultado por aproximação, é preciso saber como estimar ou delimitar o erro cometido na aproximação. Sem isso, a aproximação obtida não tem significado. Frequentemente é possível, no cálculo numérico, estimar o erro ou até delimitá-lo, isto é, estabelecer a menor das cotas superiores para o erro. A delimitação do erro é sempre desejável, pois com ela tem-se um valor em que o erro cometido seguramente é inferior a um limite.

Para se estimar ou delimitar o erro, recorre-se a dois conceitos: erro absoluto e erro relativo. Seja $\bar{x}$ um valor aproximado para uma quantidade cujo valor exato é x. Então temos:

**i) Erro Absoluto em x:** $EA_x = |x - \bar{x}|$

**ii) Erro Relativo em x:** $ER_x = \dfrac{|x - \bar{x}|}{|x|}$

Como geralmente apenas o valor de $\bar{x}$ é conhecido obtemos um limitante superior ou uma estimativa para o erro absoluto:

$$EA_x = |x - \bar{x}| \leq \varepsilon$$

$$ER_x = \frac{|x - \bar{x}|}{|x|} \leq \varepsilon$$

**Exemplos:**

**a)** Considerando x = 231.29 e $\bar{x}$ =232.04 temos:

$EA_x =$

$ER_x =$

**b)** Considerando x = 0.5682 e $\bar{x}$ =0.5701 temos:

$EA_x$ =

$ER_x$ =

**c)** Considerando x = 12.329 e $\bar{x}$ = 12.331 temos:

$EA_x$ =

$ER_x$ =

Considerando x = 0.397682 e $\bar{x}$ = 0.396965 temos:

$EA_x$ =

$ER_x$ =

Como podemos observar nos exemplos dependendo da ordem de grandeza dos números envolvidos o erro absoluto não é suficiente para descrever a precisão de um cálculo. Por essa razão, o erro relativo é amplamente utilizado.

Assim, a seguinte expressão é utilizada para o cálculo do erro:

$$\text{Erro} = \frac{|\bar{x} - x|}{max\{|x|, 1\}}$$

## Instabilidade numérica

Um dos aspectos importantes do cálculo numérico é manter o "controle" dos erros de arredondamento e truncamento.

Dada uma sequência de operações é importante ter a noção de como o erro se propaga ao longo das operações subsequentes.

Se a propagação não é significativa dizemos que o problema é **estável numericamente**. Caso contrário, isto é, se o problema é sensível a "pequenas perturbações", por exemplo, um erro de arredondamento cometido numa determinada etapa leva ao final das operações a um resultado

absurdo se comparado ao resultado esperado, tem-se então uma situação de **instabilidade numérica**.

**Exemplo:**

Supondo-se que as operações abaixo sejam processadas em uma máquina com 4 dígitos significativos e fazendo-se:

$x_1 = 0.3491\ 10^4$

$x_2 = 0.2345\ 10^0$

temos:

$(x_2 + x_1) – x_1 = ( 0.2345\ 10^0 + 0.3491\ 10^4 ) - 0.3491\ 10^4 =$

$0.3491\ 10^4 - 0.3491\ 10^4 =$

$0.0000$

$x_2 + (x_1 – x_1) = 0.2345\ 10^0 + ( 0.3491\ 10^4 - 0.3491\ 10^4 ) =$

$0.2345\ 10^0 + 0.0000 =$

$0.2345$

Os dois resultados são diferentes, quando não deveriam ser, pois a adição é uma operação distributiva. A causa dessa diferença foi um arredondamento feito na adição ($x_2 + x_1$), cujo resultado tem 8 dígitos. Como a máquina só armazena 4 dígitos, os menos significativos foram desprezados.

Ao se utilizar máquinas de calcular deve-se estar atento a essas particularidades causadas pelo erro de arredondamento, não só na adição, mas também nas outras operações.

# SISTEMAS DE NUMERAÇÃO

## Sistemas Numéricos

**Sistemas numéricos** são sistemas de notação usados para representar quantidades abstratas denominadas números. Um sistema numérico é definido pela base que utiliza.

**A base** é o número de símbolos diferentes, ou algarismos, necessários para representar um número qualquer, dos infinitos possíveis no sistema.

Os tipos de sistemas de numeração mais usuais são: Decimal, Binário, Octal e Hexadecimal.

Em um sistema de número posicional, um número é representado por uma sequência de dígitos onde cada posição de dígito tem um peso associado.

Por exemplo, o número 3.098.323 (base 10) é a representação de:
$3*10^6 + 0*10^5 + 9*10^4 + 8*10^3 + 3*10^2 + 2*10^1 + 3*10^0$.

Assim, dado um número N na base $\beta$, $N = (a_n a_{n-1} \ldots a_2 a_1 a_0)_\beta$ podemos escrever N na forma polinomial:

$$N = a_n \beta^n + a_{n-1} \beta^{n-1} + \ldots + a_2 \beta^2 + a_1 \beta^1 + a_0 \beta^0$$

## Sistema Binário

É o sistema de numeração dos computadores utilizado internamente pelo hardware. O sistema binário, ou base 2 apresenta unicamente dois dígitos: 0,1.

### *Binário para Decimal*

Sendo binário um sistema de número posicional, o valor B de um número binário de 8 dígitos $b_7b_6b_5b_4b_3b_2b_1b_0$ é:

$$B = b_7*2^7 + b_6*2^6 + b_5*2^5 + b_4*2^4 + b_3*2^3 + b_2*2^2 + b_1*2^1 + b_0*2^0.$$

Cada dígito $b_i$ tem um peso de $2^i$. Assim o valor binário $10101010_b$ é calculado como segue:

$$10101010_2 = 1*2^7 + 0*2^6 + 1*2^5 + 0*2^4 + 1*2^3 + 0*2^2 + 1*2^1 + 0*2^0 = 170_{10}.$$

Esta é a conversão de um número binário para decimal.

Converter para a base decimal:

a) $10011001_2$ =

b) $10111_2$ =

***Decimal para Binário***

Para a conversão de decimal para binário utilizamos o processo de divisões sucessivas. Por exemplo, para obtermos o correspondente binário do número $200_{10}$, dividimos primeiramente este valor por 2 e anotamos o resto da divisão. Em seguida, dividimos novamente o quociente da operação anterior por 2 e anotamos novamente o resto da divisão. Isto é repetido até que o quociente da divisão seja 0, conforme abaixo:

| | |
|---|---|
| 200/2 = 100 | Resto 0 |
| 100/2 = 50 | Resto 0 |
| 50/2 = 25 | Resto 0 |
| 25/2 = 12 | Resto 1 |
| 12/2 = 6 | Resto 0 |
| 6/2 = 3 | Resto 0 |
| 3/2 = 1 | Resto 1 |
| 1/2 = 0 | Resto 1 |

O correspondente binário de $200_{10}$ é obtido unindo-se os restos da divisão por 2 na ordem inversa, assim $200_{10} = 11001000_2$.

Genericamente:

$$\begin{array}{l|l|l|l|l} N & 2 & & & \\ r_1 & q_1 & 2 & & \\ & r_2 & q_2 & 2 & \\ & & r_3 & q_3 & 2 \\ & & & r_4 & q_4 \text{ (0 ou 1)} \end{array} \qquad N = ( q_4\, r_4\, r_3\, r_2\, r_1 )_2$$

**Exemplo:**

a) Representar $13_{10}$ na base 2.

b) Representar $347_{10}$ na base 2

## Representação de um Número Fracionário

***Binário para Decimal***

Para convertermos um número binário fracionário em um número com base decimal, escrevemos o número em potência de 2 como segue:

$(0.110)_2 = 1*2^{-1} + 1*2^{-2} + 0*2^{-3} = (0{,}750)_{10}$

**Exemplo:**

Representar os números que estão na base binária na base decimal

a) $1011.11_2$ =

b) $1101.111_2$ =

c) 1011011.1101 =

***Decimal para Binário***

A parte inteira do número fracionário é feita como antes, e a parte fracionária segue o procedimento:

- Multiplique a parte fracionária por 2;
- Desse resultado, a parte inteira será o primeiro dígito do número na base 2;
- A parte fracionária é novamente multiplicada por 2;
- O processo é repetido até que a parte fracionária seja zero.

Exemplos:

a) Representar $(0.625)_{10}$ na base 2
0.625 * 2 = 1.25
0,25 * 2 = 0.5
0,5 * 2 = 1.0
Portanto,
$0.625_{10} = 0.101_2$

**OBS:** Nem sempre um número decimal exato possui uma representação binária exata. Este fato é a principal causa de erros de arredondamento no cálculo numérico em computadores.

a) Representar $(0,1)_{10}$ na base 2

b) Representar $(0.6)_{10}$ na base 2

c) Representar $(3.8)_{10}$ na base 2

## Aritmética de ponto flutuante – F(*β, t, m, n*)

A representação em ponto flutuante surge da necessidade de representar números reais com uma faixa maior àquela propiciada pelo ponto fixo e com o objetivo de permitir ao computador o tratamento de números extremamente grandes e extremamente pequenos.

Na sua representação utiliza-se a notação matemática:

$$N = mantissa\ (base)^{expoente}$$

A representação em ponto flutuante varia conforme o comprimento da palavra do computador (quantidade de dígitos de uma representação), a base, o sistema usado para representar a mantissa e o expoente. Assim, um número N tem sua representação em ponto flutuante de *t* dígitos feita por truncamento ou arredondamento, por exemplo, o número 10.053 num sistema de ponto flutuante de 3 dígitos para a base decimal e o intervalo dos expoentes definido em [-4..+4] seria:

- Por arredondamento: $0.101\ 10^2$
- Por truncamento: $0.100\ 10^2$

Quando o expoente do número fica fora dos limites da máquina ocorrem os erros de:

- *underflow:* expoente menor que o limite inferior
- *overflow*: expoente maior que o limite superior

**Exemplos:**

- Representar o número 12 em notação normalizada de 2 dígitos, base binária e intervalo dos expoentes definido em [-4 +4]

- Representar o número 25 em notação normalizada de 2 dígitos, base decimal e intervalo dos expoentes definido em [-2 +2]

- Número 17.85 em notação normalizada de 3 dígitos, base decimal e intervalo dos expoentes definido em [-4 +4]
  - ⇒ por arredondamento:
  - ⇒ por truncamento:

- Considere o sistema F(10, 3, 2, 2). Represente nesse sistema, os números a seguir de modo que eles estejam normalizados.

a) 0.35 =

b) -5.172 =

c) 0.0123 =

d) 5391.3=

e) 0.0003 =