Cálculo Numérico

Universidade Federal do Rio Grande do Norte
-
DIMAP

Cálculo Numérico
U1 – Representação Numérica e Erros

Antonio Carlos Gay Thomé

Cálculo Numérico

Representação Numérica

Números são usados para quantificar e sua utilização remonta eras muita antigas.

A numeração escrita nasceu, em épocas muito primitivas, a partir do desejo de manter registros de bens, com marcas ou traços em paus, pedras, etc., aplicando o princípio da correspondência biunívoca.

Os sistemas de escrita numérica mais antigos que se conhece são os dos egípcios e dos babilônios, que datam aproximadamente do ano 3500 A.C..

Cálculo Numérico

Representação Numérica

Os egípcios usavam um sistema de agrupamento simples, com base 10.

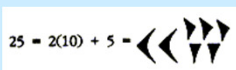
Para os egípcios, um traço vertical valia 1; o número 10 era representado por um osso de calcanhar invertido ∩; o 100 por um laço, e o 1000 por uma flor de lotus. Outros números eram escritos com a combinação desses símbolos. Assim, por exemplo 2125 se escrevia como

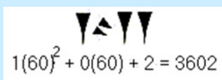
Cálculo Numérico

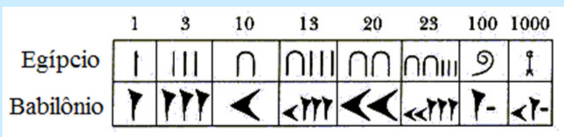
Representação Numérica

Os babilônios usavam um sistema posicional que, em alguns aspectos era semelhante ao dos egípcios.

Algumas inscrições mostram que, surpreendentemente, eles usavam não somente um sistema decimal mas também um sistema sexagesimal (isto é, base 60).







Representação Numérica

Durante toda a história do homem, assim como a palavra, o número também passou por diversas mudanças na sua forma de representação.

Os símbolos “9”, “nove”, “IX”, por exemplo, são numerais diferentes que representam a mesma quantidade, apenas escritos em idiomas e épocas distintas.

Sistema de Numeração é um sistema que representa números de uma forma consistente, dando a cada número uma representação única e possibilitando criar estruturas algébricas e realizar operações aritméticas segundo regras bem definidas.

Sistemas de Numeração

Consistem de conjuntos abstratos de símbolos e regras de formação utilizados para representar e manipular informações com valor quantitativo.

Em condições ideais, um sistema de numeração deve:

1. Representar uma grande quantidade de números úteis (ex.: todos os números inteiros, ou todos os números reais);
2. Dar a cada número representado uma única descrição (ou pelo menos uma representação padrão);
3. Refletir as estruturas algébricas e aritméticas dos números.

Sistemas de Numeração

Podem ser dos tipos: ***Posicional e Não Posicional***.

MCMXCIX ⇔ 1999

Romano

I – unidade X – dezena D – 5 dezenas
V – 5 unidades C – centena M – milhar

Decimal

$1*10^3 + 9*10^2 + 9*10^1 + 9*10^0$

Sistemas de Numeração

Nos sistemas **Não Posicionais**, os símbolos possuem apenas seus próprios valores intrínsecos (individualmente associados a quantidades específicas). São os sistemas mais antigos, como o Egípcio, o Babilônio, o Romano e outros, abandonados após o surgimento dos sistemas posicionais.

Nos sistemas **Posicionais** existe um conjunto de símbolos que é limitado e referenciado como base do sistema, e a cada símbolo da base é associado um valor intrínseco e um valor de posição. Diferentes sistemas são caracterizados em função da quantidade de símbolos que compõem a base.

Representação Posicional

Quantidade de Símbolos	Sistema
02	binário
08	octal
10	decimal
16	hexadecimal

Números Inteiros

✓ **Base decimal**

- 10 dígitos compõem a base [0,1,2, ... ,9]
- "Posição" indica potência positiva de 10
- $1011 = 1x10^3 + 0x10^2 + 1x10^1 + 1x10^0$

$1011_{/10}$

A quantidade de 10 laranjas é representada pelo número 10 (um e zero)

Representação Posicional

Sistema Decimal: Base: {0,1,2,3,4,5,6,7,8,9}

347

Valor intrínseco – 7
valor de posição – 7

Valor intrínseco – 4
valor de posição – 40

Valor intrínseco – 3
valor de posição – 300

Somando-se os valores de posição chega-se a quantidade representada pelo algarismo, 347 'laranjas' no caso presente.

Representação Posicional

✓ ***Base Binária***

- 2 dígitos compõem a base [0,1]
- "Posição" indica potência positiva de 2
- $1011 = 1x10^3 + 0x10^2 + 1x10^1 + 1x10^0$

$11_{/10}$

A quantidade de 2 laranjas é representada pelo número 10 (um e zero)

✓ ***Base Hexadecimal***

- 16 dígitos compõem a base [0 ... 9, A, B, C, D, E, F]
- "Posição" indica potência positiva de 16
- $1011 = 1x10^3 + 0x10^2 + 1x10^1 + 1x10^0$

$4113_{/10}$

A quantidade de 16 laranjas é representada pelo número 10 (um e zero)

Representação Posicional

Números Fracionários

✓ ***Base decimal***

- 10 dígitos compõem a base [0,1,2, ... ,9]
- "Posição" indica potência positiva de 10
- $1011.01 = 1x10^3 + 0x10^2 + 1x10^1 + 1x10^0 + 0x10^{-1} + 1*10^{-2}$

$1011.01_{/10}$

✓ ***Base Binária***

- 2 dígitos compõem a base [0,1]
- "Posição" indica potência positiva de 2
- $1011.01 = 1x10^3 + 0x10^2 + 1x10^1 + 1x10^0 + 0x10^{-1} + 1*10^{-2}$

$11.25_{/10}$

Representação Posicional

Números Fracionários

$4113.00390625_{/10}$

✓ ***Base Hexadecimal***

- 16 dígitos compõem a base [0 ... 9, A, B, C, D, E, F]
- "Posição" indica potência positiva de 16
- $1011.01 = 1x10^3 + 0x10^2 + 1x10^1 + 1x10^0 + 0x10^{-1} + 1*10^{-2}$

Para os humanos a base 10 é mais conveniente (10 dedos), já para os sistemas computacionais a base 2 (1 bit), base 4 (2 bits), base 8 (3 bits), base 16 (4 bits) são as mais adequadas.

Cálculo Numérico

Representação Posicional

Decimal	Binário	Octal	Hexadecimal
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
⋮	⋮	⋮	⋮

Cálculo Numérico

Conversão de Base

A conversão entre duas bases quaisquer é uma operação aritmética simples e um artifício muito utilizado pelo computador que opera unicamente na base dois (binária).

✓ **Conversão de uma base "b" qualquer para a base 10**

se $N_{/b} = a_r \cdots a_1 a_0 , a_{-1} a_{-2} \cdots a_{-p}$

então

$$N_{/10} = \sum_{i=-p}^{r} a_i b^i$$

(*) onde "b" é representado pelo seu valor na base 10

15 *Curso de Cálculo Numérico - 2015*

Cálculo Numérico

Conversão de Base

✓ **Conversão de número inteiro para a base 10 (exemplos)**

Base 2:

$1011/_2 = 1x2^0 + 1x2^1 + 0x2^2 + 1x2^3 = 11/_{10}$

Base 8:

$1011/_8 = 1x8^0 + 1x8^1 + 0x8^2 + 1x8^3 = 521/_{10}$

Base 16:

$1011/_{16} = 1x16^0 + 1x16^1 + 0x16^2 + 1x16^3 = 4113/_{10}$

Base 5:

$1011/_5 = 1x5^0 + 1x5^1 + 0x5^2 + 1x5^3 = 131/_{10}$

16 *Curso de Cálculo Numérico - 2015*

Cálculo Numérico

Conversão de Base

✓ **Conversão de número fracionário para a base 10 (exemplos)**

Base 2:

$1011.01/_2 = 1*2^{-2} + 0x2^{-1} + 1x2^0 + 1x2^1 + 0x2^2 + 1x2^3 = 11.25/_{10}$

Base 8:

$1011.01/_8 = 1*8^{-2} + 0x8^{-1} + 1x8^0 + 1x8^1 + 0x8^2 + 1x8^3 = 521.015625/_{10}$

Base 16:

$1011.01/_2 = 1*16^{-2} + 0x16^{-1} + 1x16^0 + 1x16^1 + 0x16^2 + 1x16^3 = 4113.00390625/_{10}$

Base 5:

$1011.01/_2 = 1*5^{-2} + 0x5^{-1} + 1x5^0 + 1x5^1 + 0x5^2 + 1x5^3 = 131.04/_{10}$

17 *Curso de Cálculo Numérico - 2015*

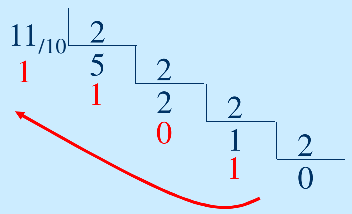
Cálculo Numérico

Conversão de Base

✓ **Conversão de número inteiro da base 10 para outra (exemplo)**

Usa-se o inverso da operação anterior, ou seja, divide-se o número sucessivamente pela base destino e toma-se como resposta, os restos da divisão em cada etapa considerados no sentido inverso (de traz para frente).

Para a Base 2:



$1011_{/2}$

18 *Curso de Cálculo Numérico - 2015*

Cálculo Numérico

Conversão de Base

✓ **Conversão de número fracionário da base 10 para outra (exemplo)**

Para a Base 2:

$11.25/_{10}$ parte inteira ➔ 1011

$0.25 \times 2 = 0.5$ ➔ 0

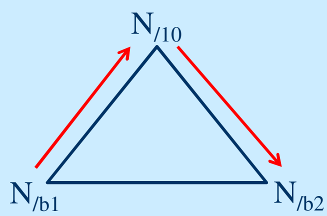
$0.5 \times 2 = 1.0$ ➔ 1

$\boxed{1011.01_{/2}}$

Cálculo Numérico

Conversão de Base

✓ **Conversão entre duas bases quaisquer (exemplo)**

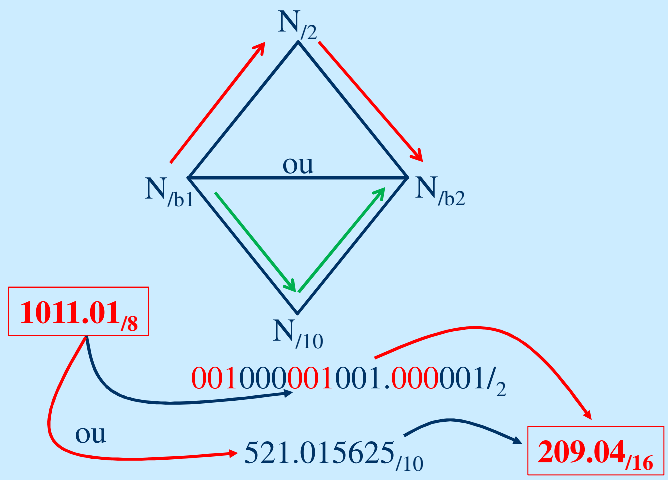


$\boxed{1011.01_{/2}}$ → $11.25/_{10}$ → $\boxed{13.2_{/8}}$

Cálculo Numérico

Conversão de Base

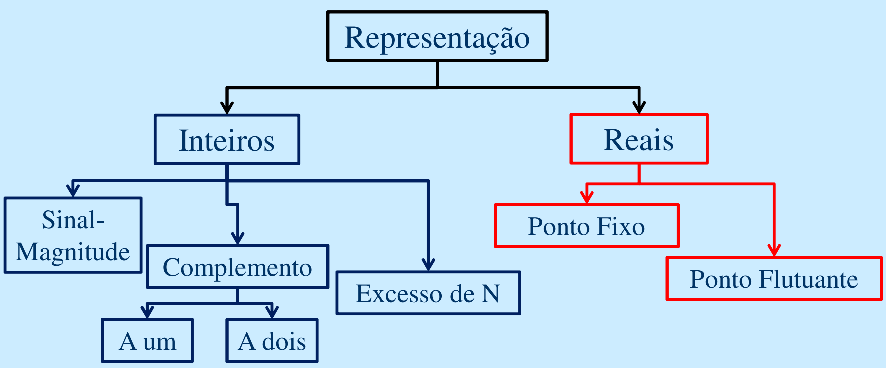
✓ **Conversão entre duas bases potência de dois (exemplo)**



Cálculo Numérico

Representação Binária

Compreende as formas padronizadas para representar números inteiros ou fracionários, positivos ou negativos, em um sistema binário para tratamento pelo hardware de um computador.



Cálculo Numérico

Representação Binária - Inteiros

Em matemática, os números negativos em qualquer base são representados prefixando-os com um sinal de "–". No entanto, em um hardware computacional, os números são representados em binário apenas, sem símbolos extras, requerendo um método de codificação para os números negativos.

Ex.: Tamanho da palavra – 16 bits

Representação em Sinal-Magnitude

$$\left.\begin{array}{ll} +25_{/10} \rightarrow +11001_{/2} \Longrightarrow & \color{red}{0}000000000011001 \\ -25_{/10} \rightarrow -11001_{/2} \Longrightarrow & \color{red}{1}000000000011001 \end{array}\right\} \text{16 bits}$$

Cálculo Numérico

Representação Binária - Inteiros

Representação Sinal-Magnitude

Limites para uma palavra de 8 bits

Binário	com Sinal	sem Sinal
00000000	+0	0
00000001	1	1
...	...	...
01111111	**127**	127
10000000	–0	128
10000001	–1	129
...	...	...
11111111	**–127**	**255**

Representação Binária - Inteiros

Representação em Complemento a Um

$+25_{/10} \rightarrow +11001_{/2} \Longrightarrow 0000000000011001$

$-25_{/10} \rightarrow -11001_{/2} \Longrightarrow 1111111111100110$

16 bits

(*) o número negativo é representado em complemento

Representação em Complemento a Dois

$+25_{/10} \rightarrow +11001_{/2} \Longrightarrow 0000000000011001$

$-25_{/10} \rightarrow -11001_{/2} \Longrightarrow 1111111111100111$

16 bits

(*) Complemento a dois = complemento a 1 + 1

Representação Binária - Inteiros

Complemento a Um

Binário	Interpretação de complemento para um	Interpretação sem sinal
00000000	+0	0
00000001	1	1
...	...	...
01111101	125	125
01111110	126	126
01111111	127	127
10000000	−127	128
10000001	−126	129
10000010	−125	130
...	...	...
11111110	−1	254
11111111	−0	255

Complemento a Dois

Binário	Interpretação de complemento para dois	Interpretação sem sinal
00000000	0	0
00000001	1	1
...	...	...
01111110	126	126
01111111	127	127
10000000	−128	128
10000001	−127	129
10000010	−126	130
...	...	...
11111110	−2	254
11111111	−1	255

Cálculo Numérico

Representação Binária - Inteiros

Excesso-*N*

A representação em **Excesso-*N*** usa um número pré-especificado *N* como um valor de polarização. Um valor é representado pelo número sem sinal que é *N* unidades maior do que o valor pretendido. Assim, 0 é representado por *N*, e -*N* é representado pelo padrão de bits zeros (tudo zerado).

Valor binário	Interpretação de Excesso-127	Interpretação sem-sinal
00000000	-127	0
00000001	-126	1
...	...	...
01111111	0	127
10000000	1	128
...	...	...
11111111	+128	255

N = 127

$\boxed{127 = 2^{8-1}-1}$

Cálculo Numérico

Representação Binária - Reais

Ponto Fixo

Quando a palavra binária é dividida, de forma fixa, em bit de sinal, bits para a parte inteira e bits para a parte fracionária.

Ponto fixo virtual

Exemplo com 16 bits: 0 1111111111 00000

Sinal – **parte inteira** – **parte fracionária**

Maior número positivo/negativo: $\pm 2^{10}-1 = \pm 1023_{/10}$
Precisão: $2^{-5} = 0.03125_{/10}$

Cálculo Numérico

Representação Binária - Reais

Ponto Flutuante

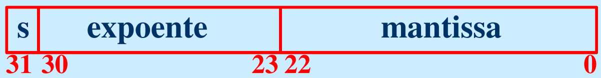
Usado fundamentalmente para números fracionários e para números inteiros que ultrapassem o limite de sua representação.

Faz uso concatenado de registradores. A mantissa é armazenada na forma normalizada e o expoente na forma de excesso a N, onde N depende do número de bits destinados para este campo.

Cálculo Numérico

Representação Binária - Reais

Ponto Flutuante



Maior número positivo/negativo: $\pm 2^{+128} \approx \pm 10^{+38}$

Menor número positivo/negativo: $\pm 2^{-127} \approx \pm 10^{-38}$

Precisão: $2^{-23} \approx 10^{-7}$

normalizada

Notação Científica: $+101101.101_{/2} \rightarrow +1.\underbrace{01101101}\text{x}2^{+5}{}_{/2}$

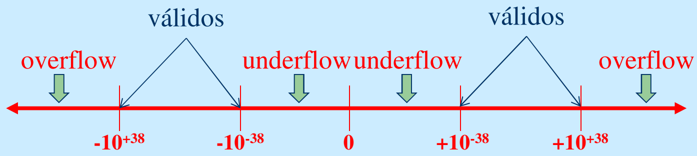
s	expoente	mantissa
0	10000100	01101101000000000000000

Cálculo Numérico

Representação Binária - Reais

- ***ERROS de Conversão***
- ***ERROS de Representação***

✓ OVERFLOW – o valor supera o máximo que pode ser representado.
✓ UNDERFLOW – o valor é menor que o mínimo possível.
✓ ROUNDING – perda por arredondamento
✓ TRUNCATING – perda por truncamento



Cálculo Numérico

Erro Absoluto e Erro Relativo

Erro Absoluto é dado pela diferença entre os valores exato e aproximado.

$$\boxed{EA_x = x - \overline{x}}$$

Em geral, apenas o valor aproximado é conhecido e, neste caso, torna-se impossível obter o valor exato do erro absoluto.

A solução é obter um limitante superior ou uma estimativa para o módulo do erro absoluto.

Exemplo: sabendo que $\pi \in (3.14, 3.15)$ toma-se $|EA_\pi| = |\pi - \overline{\pi}| < 0.01$

Cálculo Numérico

Erro Absoluto e Erro Relativo

Todo Erro Absoluto apresenta a mesma precisão?

Exemplo:

$$x \in (2112.8, 2113) \; / \; \bar{x} = 2112.9 \Rightarrow |EA_x| < 0.1$$

$$y \in (5.29, 5.35) \; / \; \bar{y} = 5.3 \Rightarrow |EA_y| < 0.1$$

O Erro Relativo é calculado dividindo-se o Erro Absoluto pelo valor aproximado.

$$ER_x = \frac{EA_x}{\bar{x}} = \frac{x - \bar{x}}{\bar{x}}$$

$$|ER_x| = \frac{|EA_x|}{|\bar{x}|} < \frac{0.1}{2112.9} \simeq 4.7 \times 10^{-5}$$

$$|ER_y| = \frac{|EA_y|}{|\bar{y}|} < \frac{0.1}{5.3} \simeq 0.02,$$

A precisão em x é maior que em y.

Cálculo Numérico

As medidas 'Épsilon da máquina' e ULP

1. Épsilon da máquina - ε

Em ponto-flutuante, "ε" da máquina representa o menor número que somado a 1 produza resultado diferente de 1, ou seja, não é arredondado.

IEEE 754 - 2008	Nome usual	Tipo de dado em C++	Base b	Precisão p	Épsilon de máquina[a] $b^{-(p-1)}/2$
binary16	meia precisão	indisponível	2	11 (um bit implícito)	$2^{-11} = 4.88e-04$
binary32	precisão singular	float	2	24 (um bit implícito)	$2^{-24} = 5.96e-08$
binary64	precisão dupla	double	2	53 (um bit implícito)	$2^{-53} = 1.11e-16$

(*) representa o menor valor (mais próximo de zero) representável

Cálculo Numérico

As medidas 'Épsilon da máquina' e ULP

2. Unit in the last position - ULP

ULP(x) representa a distância entre dois números reais (floating- points) que sejam representáveis, adjacentes e que "x" seja um deles ou esteja entre eles.

MATLAB ➔ a instrução eps(x) opera tanto como ε ou ulp()

- **eps(X) is the positive distance from abs(X) to the next larger in magnitude floating point number of the same precision as X. Quando X=1 opera com "ε" da máquina. (precisão é o nr. Bits da mantissa)**

Precisão Simples	Precisão Dupla
eps(single(1)) = 2^{-23}	eps(double(1)) = 2^{-52}
eps(single(1/2)) = 2^{-24}	eps(double(1/2)) = 2^{-53}

Cálculo Numérico

Acurácia x Precisão

Acurácia (exatidão): é um conceito qualitativo

Grau de concordância entre o resultado de uma medição e um valor verdadeiro do mensurando.

Precisão: é um conceito quantitativo

Grau de concordância entre vários resultados da medição obtidos sob as mesmas condições (repetitividade).