```
0 00000000000 0000000000000000000000000000000000000000000000000000  x=0.
0 01111111111 0000000000000000000000000000000000000000000000000000  x=1.
0 10000000000 0000000000000000000000000000000000000000000000000000  x=2.
0 10000000000 1000000000000000000000000000000000000000000000000000  x=3.
0 10000000001 0000000000000000000000000000000000000000000000000000  x=4.
0 10000000001 0100000000000000000000000000000000000000000000000000  x=5.
0 10000000001 1000000000000000000000000000000000000000000000000000  x=6.
0 10000000001 1100000000000000000000000000000000000000000000000000  x=7.
0 10000000010 0000000000000000000000000000000000000000000000000000  x=8.
0 10000000010 0010000000000000000000000000000000000000000000000000  x=9.
0 01111111110 0000000000000000000000000000000000000000000000000000  x=0.5
0 01111111001 0100011110101110000101000111101011100001010001111011  x=0.02
0 01111111000 0100011110101110000101000111101011100001010001111011  x=0.01
0 01111110101 0000011000100100110111010010111100011010100111111100  x=0.001
0 01111110110 1001100110011001100110011001100110011001100110011010 x=0.003125
0 10000001001 0000000000000000000000000000000000000000000000000000  x=1024
0 10000001010 0000000000000000000000000000000000000000000000000000  x=2048
0 10000001011 0000000000000000000000000000000000000000000000000000  x=4096
0 10000001011 0000000000010000000000000000000000000000000000000000  x=4097
0 10000001011 0000000000001000000000000000000000000000000000000000  x=4096.5
0 10000001011 0000000000000100000000000000000000000000000000000000  x=4096.25
1 10000000010 0010000000000000000000000000000000000000000000000000  x=-9.
1 01111111110 0000000000000000000000000000000000000000000000000000  x=-0.5
1 01111111001 0100011110101110000101000111101011100001010001111011  x=-0.02
```

Pode-se observar como o sinal é guardado no bit mais significativo, os $q = 11$ bits seguintes são reservados ao exponente ($2^{11} = 2048$), guardado com excesso de 1023 (por isso $1 = 2^0$ tem $(01111111111)_2 = 1023$ no espaço do expoente), e os 52 bits restantes são reservados à mantissa ($p = 53$ é a precisão), onde o primeiro algarismo ($1.\cdots$) não é guardado, posto que sempre é 1 (salvo para o 0). Para o número 0., único com mantissa que não começa pelo 1, reserva-se a expressão cheia de 0's. Esta expressão deve interpretar-se como 0 e não como $1.0\ldots\cdot 2^{-1023}$.

Quando o expoente é $(00000000000)_2 = 0$, não deve tomar-se $0 - 1023 = -1023$ e interpretar-se como um número real de expoente 2^{-1023}. Além do número 0 antes indicado, este valor de expoentes está reservado no IEEE754 para indicar certos objetos de controlo, os "denormalized numbers". Por outro lado, quando o expoente é $(11111111111)_2 = 2047$, não deve tomar-se $2047 - 1023 = 1024$ e interpretar-se como um número real de expoente 2^{1024}. Este valor de expoente no IEEE754 está reservado para indicar certos objetos de controlo como $\pm\infty$ (quando a mantissa é $0\ldots0$), NaN ("not a number", quando a mantissa é não nula),

1.3 Aproximações e erros.

No trabalho com modelos matemáticos criados para representar situações reais as grandezas com que se trabalha não são precisas. A diferencia entre o valor numérico (o número) que usamos para representar aproximadamente a grandeza e o autêntico valor desta (valor que pode ser desconhecido) é o que se conhece como **o erro da aproximação**.

Os resultados nos nossos cálculos são afetados por vários tipos de erro. A interpretação correta do resultado exige então conhecer quais as possíveis origens e se há forma de evitar as causas destes erros.

Ao fazer um cálculo matemático (aplicar um procedimento algorítmico) a partir de dados, o resultado é suscetível de conter erros que se podem classificar em dois tipos:

1. O erro devido à falta de precisão dos dados de partida.
2. e o erro que o algoritmo produz quando faz o cálculo com dados exatos.

A falta de precisão nos dados de partida está provocada pela forma em que esses dados são obtidos e representados. Este erro tem uma **componente sistemática** e uma **componente aleatória**. A parte sistemática pode dever-se, por exemplo, a uma má calibração dos aparelhos de medição, ao arredondamento decimal que se faz nos dados, ou ao erro herdado duma série de operações feitas para gerar esses dados. A parte aleatória aparece quando os dados são obtidos através de um processo de tipo aleatório. Assim, uma mesma medição realizada em condições "idênticas" (até onde se possa controlar) é influenciada por fatores não controláveis, e produz resultados diferentes quando repetida, é uma experiência aleatória onde cada resultado individual está afetado por fatores aleatórios.

Os erros dos dados de partida podem também dever-se a que estes dados sejam o resultado da aplicação dum método numérico (que pode introduzir erros) desde valores conhecidos anteriormente (valores que podiam também conter erros)

Por outra parte, os algoritmos numéricos podem também produzir erros quando aplicados a dados exatos. Os resultados contêm então um erro que será propagado se se aplicam outros algoritmos. As fontes de erro mais comuns na aplicação de algoritmos são:

1. Não correspondência do modelo matemático com a realidade. As ciências oferecem-nos modelos que pretendem ser fiel reflexo da realidade. No entanto, isto não é assim, os modelos costumam ter um âmbito no qual se ajustam em forma aceitável à realidade e um âmbito no qual é preciso um modelo diferente para tratar o problema. Os algoritmos derivados destes modelos levam a resultados com maior ou menor erro em função de como se ajuste este modelo com a realidade.
2. Erros humanos e da máquina. O programador de algoritmos tem que ter um extremo cuidado na interpretação da teoria e na forma de representar esta no código que se programa ou no desenho de circuitos que se ocupam das operações internas. Pode acontecer, até com os melhores programadores e fabricantes, que estes nem sempre, frente a todos os dados, produzam os resultados esperados.
3. Erros de arredondamento. A causa deste erro é a impossibilidade de representar números reais numa memória finita. O erro de arredondamento aparece quando um número real é substituído por um valor expressado por um número fixo de algarismos (decimais ou binários). A limitação da aritmética de ponto flutuante leva a perda de informação que, dependendo do contexto pode ser mais ou menos importante. Como exemplo, a transformação de um número em ponto flutuante de base 10 a base 2 pode introduzir um erro de arredondamento: quase sempre que uma máquina começa tratar números que uma pessoa introduz, o primeiro que a máquina faz é adicionar um erro (o número é guardado como aquele valor em ponto flutuante em base 2 que melhor o representa).
4. Erros de truncatura. A causa deste erro é a impossibilidade de computar valores limite, que exigiriam tempo infinito para serem obtidos. Muitas ferramentas matemáticas usam a noção de limite duma sucessão. No entanto, a sucessão infinita não pode ser calculada explicitamente e o valor limite não pode ser obtido através da sucessão e da definição de limite. Programar um algoritmo que use um valor limite exige então truncar o cálculo dos termos da sucessão num determinado ponto em que espera-se ter obtido um valor o bastante próximo do valor limite, que não pode ser alcançado. Um exemplo grosseiro de truncatura é mudar uma determinada função $f(x)$ pelo polinómio de Taylor de segunda ordem associado $f(0) + f'(0) \cdot x + f''(0)/2 \cdot x^2$ (trunca-se a expressão infinita $f(x) = f(0) + f'(0) \cdot x + f''(0)/2 \cdot x^2 + f'''(0)/6 \cdot x^3 + \ldots$ de Taylor na segunda parcela). O erro de arredondamento na representação de ponto flutuante pode-se interpretar como um erro de truncatura ao substituir uma soma infinita $\sum_{k=1}^{\infty} a_{-k} b^{-k}$ pela soma truncada num passo dado.

Medidas de erro

Pensemos numa grandeza x que nós aproximamos pelo valor x^*. Na secção anterior demos vários motivos pelo qual o valor aproximado, no geral, pode não coincidir com o autêntico.

O valor x^* trata-se então duma aproximação. Esta questão apresenta-se quando trabalhamos com valores reais, mas também irá aparecer quando trabalharmos com pontos do plano $\mathbb{R}^2$, do espaço tridimensional $\mathbb{R}^3$ ou em geral com elementos de $\mathbb{R}^n$ (os elementos $x = (x_1, x_2, \ldots, x_n)$ sendo cada x_i um valor real)

Pretendemos distinguir quais são aproximações mais precisas e quais menos. Iremos associar a cada valor x e cada possível aproximação x^* um valor não negativo, uma **medida do erro** de x^* como aproximação de x.

Aparece uma dúvida natural, nomeadamente se consideramos $x^* = 2581$ como aproximação de $x = 2581 + \frac{2}{3}$, ou se consideramos $y^* = 1$ como aproximação de $y = 1 + \frac{1}{3}$, qual delas consideramos melhor aproximação? Por uma parte a distância entre x^* e x é $\frac{2}{3}$ (no primeiro caso), enquanto a distância entre y^* e y é $\frac{1}{3}$ (menor neste segundo caso). Por outra parte a quantidade $\frac{1}{3}$, em termos das unidades que estamos a usar, resulta maior do que $\frac{2}{3}$, em termos das unidades que estamos a usar.

Para medir o erro em $\mathbb{R}^n$ precisamos de uma noção de distância entre x^* e x (para o erro absoluto), e de uma forma de ponderar esta distância em termos da grandeza de x (para o erro relativo).Isto pode ser resolvido com ajuda da noção de **norma**:

Definição 1.18. *Uma **norma** $\|\cdot\|$ em $\mathbb{R}^n$ é uma função real $\mathbb{R}^n \to \mathbb{R}$ que satisfaz:*

1. *A norma é definido-positiva:* $\|x\| > 0, \quad \forall x \in \mathbb{R}^n, \quad x \neq (0, \ldots, 0)$
2. *A norma é homogénea:* $\|\lambda \cdot x\| = |\lambda| \cdot \|x\|, \quad \forall x \in \mathbb{R}^n, \forall \lambda \in \mathbb{R}$
3. *A norma é sub-aditiva:* $\|x + x'\| \leq \|x\| + \|x'\|, \quad \forall e, e' \in E$

Exemplos de normas são as seguintes:

Definição 1.19. *Seja $p \geq 1$ um número real. Chamamos **norma**-p dum ponto $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ o valor:*

$$\boxed{\|x\|_p = \sqrt[p]{|x_1|^p + |x_2|^p + \ldots + |x_n|^p}}$$

*Observamos que para $p \to +\infty$, isto define uma **norma**-∞*

$$\boxed{\|x\|_\infty = \max(|x_1|, |x_2|, \ldots, |x_n|)}$$

No caso $n = 1$ (o conjunto dos números reais) estas normas são todas a mesma, o cálculo de $|x|$

No caso $n = 2$ (os pontos do plano), temos três normas importantes muito usadas:

$$\|(x_1, x_2)\|_1 = |x_1| + |x_2|, \quad \|(x_1, x_2)\|_2 = \sqrt{|x_1|^2 + |x_2|^2}, \quad \|(x_1, x_2)\|_\infty = \max(|x_1|, |x_2|)$$

Denominadas norma-1, norma-2 (ou Euclidiana) e norma-∞ no plano.

A norma Euclidiana é frequente, em particular por ter a sua origem na geometria Euclidiana, ainda que o seu cômputo supõe múltiplas operações. As normas $\|\cdot\|_\infty$ e $\|\cdot\|_1$ são frequentes pela sua simplicidade de cálculo e uso intuitivo.

A diferença entre dois pontos de $\mathbb{R}^n$ é um novo elemento $x - y \in \mathbb{R}^n$. A norma desde vetor é chamada a **distância entre os pontos**, com respeito da norma escolhida.

Definição 1.20. *Chamamos **produto escalar** $x \cdot y$ de dois elementos de $\mathbb{R}^n$ o seguinte valor real (escalar):*

$$(x_1, \ldots, x_n) \cdot (y_1, \ldots, y_n) = x_1 y_1 + x_2 y_2 + \ldots + x_n y_n$$

As seguintes propriedades são importantes para esta operação

Proposição 1.21. *O produto escalar em $\mathbb{R}^n$ satisfaz:*

- *É simétrico* $x \cdot y = y \cdot x$

- *É linear em cada componente, para escalares $\alpha \in \mathbb{R}$ e vetores $x, y, \bar{x}, \bar{y} \in \mathbb{R}^n$:*

$$(x + \bar{x}) \cdot y = x \cdot y + \bar{x} \cdot y, \quad x \cdot (y + \bar{y}) = x \cdot y + x \cdot \bar{y}$$

$$(\alpha x) \cdot y = \alpha(x \cdot y) = x \cdot (\alpha y)$$

- *Com respeito da norma euclidiana:*

$$|x \cdot y| \leq \|x\|_2 \cdot \|y\|_2, \qquad x \cdot x = \|x\|_2^2$$

- *Para vetores $u \in \mathbb{R}^n$ unitários em norma-p (aqueles com $\|u\|_p = 1$):*

$$\max_{\|u\|_p = 1} |x \cdot u| = \|x\|_q$$

sendo q o valor que satisfaz $\frac{1}{p} + \frac{1}{q} = 1$

Esta última propriedade permite afirmar que a norma-p tem uma **norma dual**, a norma-q, sendo que:

$$\boxed{|x \cdot y| \leq \|x\|_q \cdot \|y\|_p \qquad (\frac{1}{p} + \frac{1}{q} = 1)}$$

Esta desigualdade é precisa (em inglês "sharp"), no sentido de que para qualquer x é possível encontrar um y onde é satisfeita a igualdade, e reciprocamente.
(A norma dual da norma-2 é novamente a norma-2. A norma dual da norma-1 é a norma-∞. A norma dual da norma-∞ é a norma-1)

Mais uma desigualdade precisa é a seguinte:

$$\boxed{|x \cdot y| \leq |x_1| \cdot |y_1| + |x_2| \cdot |y_2| + \ldots + |x_n| \cdot |y_n|}$$

(a norma do produto escalar não é superior ao produto escalar das normas de cada componente)

Definição 1.22. *Se fixamos uma norma $\|\cdot\|$ em $\mathbb{R}^n$, chamamos **distância** associada a esta norma a função $\Delta(x, y) = \|x - y\|$. Esta função satisfaz as seguintes propriedades:*

- *É não negativa $\Delta(x, y) \geq 0$, com $\Delta(x, y) = 0 \Leftrightarrow x = y$*
- *É simétrica $\Delta(x, y) = \Delta(y, x)$*
- *Satisfaz a desigualdade triangular: $\Delta(x, y) \leq \Delta(x, z) + \Delta(z, y)$*

Distâncias muito usadas são a distância euclidiana Δ_2 (comprimento dum percurso que une dois pontos, se usamos qualquer direção retilínea), a distância Δ_1 chamada "taxicab" ou "Manhattan" (é o comprimento dum percurso entre os pontos, se só usamos direções paralelas aos eixos), e a distância Δ_∞, chamada também de Chebychev ou "do rei no xadrez".

Os conceitos de norma e de distância vão permitir a introdução duma **medida do erro**:

Definição 1.23. *Chamamos **medida do erro absoluto, com respeito duma norma** $\|\cdot\|$ em $\mathbb{R}^n$, dum ponto x^* como aproximação doutro ponto x o valor:*

$$\boxed{\Delta(x^*, x) = \|x^* - x\|}$$

*Se $x \neq (0, \ldots, 0)$, chamamos **medida de erro relativo** de x^* como aproximação de x, o valor:*

$$\boxed{\delta(x^*, x) = \frac{\Delta(x^*, x)}{\|x\|}}$$

*O produto do erro relativo com 100 é chamado **percentagem de erro** da aproximação.*

Como notação abreviada para indicar um erro absoluto ou um erro relativo, usamos as seguintes

$$x^* = x \pm \epsilon \Leftrightarrow \Delta(x^*, x) \leq \epsilon$$

$$x^* = x \cdot (1 \pm \alpha) \Leftrightarrow \delta(x^*, x) \leq \alpha$$

Uma utilidade da norma é que permite definir quando uma sucessão de pontos $x(1), x(2), x(3), \ldots$ é convergente a um determinado limite $a \in \mathbb{R}^n$.

Definição 1.24. *Diremos que uma sucessão de pontos $x(1), x(2), x(3), \ldots$ é **convergente** a um determinado ponto limite $a \in \mathbb{R}^n$ quando a sucessão de erros absolutos associada $\Delta(k) = \|x(k) - a\|$ tiver limite zero (equivalentemente, se a sucessão de erros relativos tiver limite zero). Escrevemos então*

$$\boxed{\lim_{k\to\infty} x(k) = a \qquad (\lim \Delta(k) = 0)}$$

*Diremos que há **convergência linear** se existe uma constante $c < 1$ para a qual*

$$\Delta(k+1) < c \cdot \Delta(k) \qquad (c < 1)$$

*Diremos que há **convergência quadrática** (mais rápida que a linear) se existe alguma constante c para a qual $\Delta(k+1) < c \cdot (\Delta(k))^2$, e em geral diremos que tem **convergência de ordem** q se existe uma constante c para a qual*

$$\boxed{\Delta(k+1) < c \cdot (\Delta(k))^q}$$

(A definição se trocamos Δ por δ resulta ser coincidente)

Por exemplo a sucessão $\frac{1}{2^k}$ tem convergência linear a $\overline{x} = 0$ Uma propriedade notável é que todas as normas em $\mathbb{R}^n$ são equivalentes, em qualquer questão relativa à convergência ou à ordem de convergência de sucessões. Se uma sucessão de pontos é convergente a um ponto a com ordem de convergência q quando usamos uma norma, a mesma propriedade continua a ser válida se usamos qualquer outra norma. Também resulta equivalente definir a convergência ou a ordem de convergência através do erro relativo.

Nota 1.25. Se pensamos em $\mathbb{R}$, onde a igualdade $\|x\| = a$ só tem como soluções $x = a$ ou $x = -a$, conhecer o erro relativo $\delta = \delta(x, x^*)$ permite escrever $x = x^* \cdot (1 \pm \delta)$. Podemos também escrever $x = x^* \pm \Delta$ (onde $\Delta = \Delta(x^*, x)$)

Os erros absolutos são claramente simétricos. Isto é, temos $\Delta(x^*, x) = \Delta(x, x^*)$. Os erros relativos no entanto, não sã simétricos. No caso da medição de erros relativos em $\mathbb{R}$ temos:

$$\delta(x, x^*) = \pm\frac{x^* - x}{x^*} = \pm\frac{x^* - x}{x}\left(1 + \frac{x^* - x}{x}\right)^{-1} = \frac{\pm\delta(x^*, x)}{1 \pm \delta(x^*, x)} = \pm\delta(x^*, x) - \delta(x^*, x)^2 + \ldots$$

(onde $\ldots$ representa o desenvolvimento de Taylor de $\frac{\delta}{1+\delta}$ para δ próximo de zero)

Portanto se $\delta(x^*, x)$ é pequeno o seu quadrado é muito menor e temos $\delta(x, x^*) \simeq \delta(x^*, x)$. As duas formas de definir o erro relativo são aproximadamente iguais, para valores do erro pequenos. Podemos assim escrever $x = x^* \cdot (1 \pm \delta)$ se conhecemos $\delta = \delta(x^*, x) \simeq \delta(x, x^*)$

Por outra parte, utilizaremos com frequência a notação $x^* \pm \epsilon$ como aproximação de x. Com esta expressão queremos dizer que existe um valor $e \in [-\epsilon, \epsilon]$ tal que $x^* = x + e$. Neste caso um valor aproximado para x é x^* e o valor absoluto do erro cometido na aproximação é $|x^* - x| \leq \epsilon$.

Quando trabalhamos em $\mathbb{R}$ há uma forma de indicar se o erro relativo é grande. Esta é a noção de **número de algarismos significativos da aproximação**. Suponhamos que temos um valor x desconhecido cuja representação em notação científica em base b é:

$$x = \pm(a_0.a_1a_2 \ldots a_{p-1} \ldots)_b \cdot b^e$$

com $a_0 \neq 0$. Isto é, temos $b^e \leq x < b^{e+1}$. Suponhamos que temos um valor x^* como aproximação de x.

Definição 1.26. *Dado $b^e \leq x < b^{e+1}$, diremos que x^* aproxima x* ***com p algarismos significativos*** *ou* ***com p algarismos de precisão*** *em base b se:*

$$\boxed{\left|\frac{x^* - x}{b^e}\right| \leq \frac{b}{2} \cdot b^{-p}}$$

quando b é par, isto quer dizer que:

$$|x^* - x| \leq (0.00\ldots\overbrace{0}^{p-1}\overbrace{(b/2)}^{p})_b \cdot b^e$$

Por exemplo a aproximação 0.008234 ± 0.000004 dum número x desconhecido, podemos afirmar que é uma aproximação com 3 algarismos significativos: $x \in [0.00823, 0.008238] \subset [10^{-3}, 10^{-2}]$ (portanto $e = -3$) e $|x^* - x|/10^{-3} \leq 0.004 \leq 5 \cdot 10^{-3}$. No entanto, 0.008234 ± 0.000006 não podemos saber se é uma boa aproximação com 3 algarismos significativos, o máximo que podemos afirmar é que tem 2 algarismos significativos:

$$\left|\frac{x^* - x}{10^{-3}}\right| \leq 0.006 \begin{cases} \not\leq 5 \cdot 10^{-3} \\ \leq 5 \cdot 10^{-2} \end{cases}$$

dependendo de qual é o valor real desconhecido, pode ser que a diferença alcance até 0.006, que não é menor que $5 \cdot 10^{-3}$ mas sim é menor do que $5 \cdot 10^{-2}$. é uma aproximação com 2 algarismos significativos.

O número de algarismos significativos numa aproximação é uma forma de expressar o erro relativo cometido na aproximação.

Porquê tomamos $b/2 \cdot b^{-p}$ para falar de algarismos significativos duma aproximação?. Isto é devido ao método de arredondamento:

Nota 1.27. Seja x um valor real não nulo. Ao determinar os seu primeiros p algarismos significativos em base b e a fração restante temos

$$x = \pm\left((a_0\,.\,a_1\ldots a_{p-1})_b \cdot b^e + \epsilon\right)$$

com $a_0 \neq 0$, $0 \leq \epsilon < (0.0\ldots\overbrace{1}^{p-1})_b \cdot b^e = b^{e-p+1}$.

O valor $x^* = \pm(a_0\,.\,a_1\ldots a_{p-1})_b \cdot b^e$ era a aproximação de x obtida por **arredondamento por corte com p algarismos** ("aproximação ao zero", "por corte" ou "por defeito"). O valor $x^{**} = x^* + (0.0\ldots\overbrace{1}^{p-1})_b \cdot b^e = x^* + b^{e-p+1}$ é o **arredondamento "por excesso" com p algarismos**.

Podemos observar então:

$$\frac{|x^{**} - x^*|}{b^e} = b^{-p+1}$$

Como x está situado entre o arredondamento por corte x^* e o arredondamento por excesso x^{**}, podemos afirmar que nestes dois métodos de arredondamento temos:

$$\frac{|fl(x) - x|}{b^e} \leq b^{-p+1} \quad \text{por corte ou por excesso}$$

Por outra parte seja qual for x, ao considerar o mais próximo destes dois arredondamentos iremos ter o arredondamento ao mais próximo. Para este arredondamento portanto:

$$\frac{|fl(x) - x|}{b^e} \leq \frac{1}{2} b^{-p+1} \text{arredondamento simétrico}$$

No **arredondamento simétrico com p algarismos** em base b, podemos ter a certeza que o valor arredondado $\hat{x}$ representa uma **aproximação de x com p algarismos significativos**.

O erro relativo numa aproximação feita por arredondamento fica então limitado pelo número de algarismos usados no arredondamento. Assim o espaço de memória que utilizarmos para guardar mantissa dum número influencia o erro relativo que admitimos cometer nos arredondamentos.

O espaço de memória que usamos para guardar o expoente influencia a diversidade de expoentes que conseguimos guardar.

Reservar muito espaço para os expoente irá permitir guardar de maneira arredondada um intervalo maior de números, e reservar muito espaço para a mantissa irá permitir que estes números sejam guardados com maior precisão (menor erro relativo). A metodologia do IEEE fixa um compromisso entre precisão e amplitude do sistema números usados.

Nota 1.28. Medir a velocidade de convergência duma sucessão através da "ordem de convergência" pode interpretar-se como uma maneira de medir quantos algarismos significativos ganhamos em cada novo termo da sucessão.

Podemos visualizar graficamente se uma sucessão é convergente com ordem de convergência q, e interpretar o significado em termos dos algarismos significativos de precisão de cada um dos termos como aproximação do valor limite.

Consideremos uma sucessão $(x(k))$ convergente a um valor $\overline{x} > 0$, e que o expoente binário de k é o valor $e \in \mathbb{Z}$, isto é, $\overline{x} \in [2^e, 2^{e+1}[$. O número de algarismos binários de precisão de $x(k)$ como aproximação de $\overline{x}$ está dada por $prec(k) = -\log_2 \frac{|x(k)-\overline{x}|}{2^e} = e - \log_2 \Delta(k)$.

A condição $\Delta(k+1) < \text{cte} \cdot (\Delta(k))^q$ equivale a $\log_2 \Delta(k+1) < \log_2 \text{cte} + q \log_2 \Delta(k)$, portanto a $e - \log_2 \Delta(k+1) > -\log_2 \text{cte} + (1-q)e + q(e - \log_2 \Delta(k))$, isto é, o número de algarismos binários de precisão $prec(k)$ devem verificar $prec(k+1) > \alpha + q \cdot prec(k)$. A cada novo termo da sucessão, o número de algarismos de precisão fica multiplicado com q (salvo um termo α somado, que será pouco relevante se o número de algarismos significativos for já elevado)

Portanto um método para calcular a ordem de convergência duma sucessão seria calcular os pontos $(prec(k), prec(k+1))$. Se estes pontos podem ser situados num semiplano $y > \alpha + qx$ limitado inferiormente por uma reta de declive q, a sucessão tem ordem de convergência q.

Da mesma maneira, com ajuda dos erros absolutos: se os pontos $(\log \Delta(k), \log \Delta(k+1))$ ficam situados num semiplano $y < q \cdot x$ limitado superiormente por uma reta de declive q, a sucessão tem ordem de convergência q. Podemos estudar assim a ordem de convergência se representamos graficamente os pontos $(\log_2 \Delta(k+1), \log_2 \Delta(k))$.

Nota 1.29. Para o arredondamento por corte $a^* = fl(a)$ dum número $a \neq 0$ numa máquina binária com precisão p temos $\delta(a^*, a) \leq 2^{-p}$. Portanto no caso dum elemento $a \in \mathbb{R}^n$ as componentes (valores reais) são aproximadas com $|a_i^* - a_i| \leq 2^{-p} \cdot a_i$.

Em muitas situações queremos representar no computador não um número, senão um ponto, uma sequência $a = (a_1, \ldots, a_n)$ com n componentes reais. O procedimento mais normal é usar um conjunto de n posições de memória e introduzir em cada uma delas o valor arredondado a_i^*. Com a medição de erros através da norma infinito temos:

$$\delta_\infty(a^*, a) = \frac{\|a^* - a\|_\infty}{\|a\|_\infty} = \frac{\max|a_i^* - a_i|}{\max|a_i|} \leq \frac{\max|2^{-p} \cdot a_i|}{\max|a_i|} = 2^{-p} \cdot \frac{\max|a_i|}{\max|a_i|} = 2^{-p}$$

Portanto do ponto de vista da norma-∞, numa máquina com representação de números em ponto flutuante com p algarismos de mantissa, os vetores podem ser representados com erros relativos não superiores a 2^{-p}

Definição 1.30. *Chama-se* ***unidade de arredondamento*** *duma máquina o supremo dos erros relativos introduzidos pelos arredondamentos existentes na máquina. Isto é, a unidade de arredondamento é o menor valor u que verifica:*

$$\boxed{\frac{|fl(x) - x|}{|x|} < u}$$

nos diferentes valores x que não sejam overflow/underflow.

Numa máquina binária com precisão p e que faz arredondamentos ao mais próximo, a unidade de arredondamento é então $u = 2^{-p}$.

Outro valor que produz informação análoga à precisão da máquina é o conhecido como épsilon da máquina.

Definição 1.31. *Chama-se **épsilon da máquina** num sistema de cômputo em ponto flutuante o menor número positivo ϵ representável na máquina e tal que:*

$$\boxed{1 \oplus \epsilon > 1}$$

onde $1 \oplus \epsilon$ quer dizer a soma de 1 e ϵ tal como feita pela máquina para números em ponto flutuante, normalmente $fl(1+\epsilon)$.

1.4 Propagação do erro

A noção de valor aproximado pode entender-se também para funções aproximadas ou algoritmos.

Pensemos num problema matemático com solução perfeitamente determinada mas dependente de parâmetros de entrada. Por exemplo, pensemos no problema de encontrar a menor das raízes reais do polinómio $x^2 + b \cdot x + c$. Sabemos que existe uma solução deste problema, e que está determinada pela aplicação da fórmula resolvente:

$$\text{Menor raiz de } x^2 + b \cdot x + c \text{ é } z = \frac{-b - \sqrt{b^2 - 4c}}{2}$$

A solução do problema matemático está assim descrita por $z = f(b, c)$ onde f é uma função nos parâmetros b, c que determinam o problema.

Se temos um sistema numérico com finitos números e onde existe um procedimento para somar, multiplicar, dividir, e extrair raízes destes números em forma aproximada, poderíamos criar um algoritmo f^* como aproximação da fórmula resolvente, ou seja uma série de instruções que:

- Precisa da introdução de dois valores b^*, c^* existentes no sistema numérico.
- Através das instruções programadas, produz como resultado um valor f^*
- O valor f^* devolvido irá ser usado como aproximação de $f(b, c)$, sempre que (b^*, c^*) sejam considerados aproximação de (b, c)

Assim o problema matemático tem parâmetros de entrada x (no caso anterior $x = (b, c)$), e conhecido o parâmetro existe uma solução determinada (aqui $f(x) = \frac{-b-\sqrt{b^2-4c}}{2}$).

No computador temos um algoritmo $f^* \colon x^* \mapsto y^*$ que admite como parâmetros de entrada valores x^* representáveis no computador e que consegue devolver um valor $y^* = f^*(x^*)$ representável no computador.

Muitas questões na matemática exigem a utilização de determinados valores (parâmetros) Suponhamos que temos um problema a resolver, que este problema depende de parâmetros numéricos e que este problema é resolvido através da função $f \colon x \to f(x)$. Suponhamos que conhecemos um algoritmo numérico $f^* \colon x^* \mapsto f^*(x^*)$.

Existem duas questões pertinentes para ver se f^* é bom para representar f:

- Se tivéssemos parâmetros representáveis em forma exata no computador $x = x^*$, será que o valor devolvido pelo algoritmo $f^*(x^*)$ é a melhor aproximação existente de $f(x^*)$? Será que o computador não consegue distinguir $f(x^*)$ de $f^*(x^*)$? Ou seja: $f^*(x^*) = fl(f(x^*))$ para qualquer x^* do sistema numérico usado?