SM 2022-2023

Série clássica de Fourier

$x(t)$ é uma função periódica de período T : $\forall t \quad x(t+T)=x(t)$

Podemos expandir $x(t)$ numa soma de cossenos e senos:

$$x(t) = \frac{a_0}{2} + \sum_{m=1}^{+\infty} a_m \cos\left(2\pi m \frac{t}{T}\right) + \sum_{m=1}^{+\infty} b_m \sin\left(2\pi m \frac{t}{T}\right),$$

$$a_m = \frac{2}{T}\int_0^T x(t)\cos\left(2\pi m \frac{t}{T}\right) dt \, , \quad m=0,1,\ldots$$

$$b_m = \frac{2}{T}\int_0^T x(t)\sin\left(2\pi m \frac{t}{T}\right) dt \, , \quad m=1,2,\ldots$$

Mas como $(j=\sqrt{-1})$

é preciso saber $\left\{ \cos x = \dfrac{e^{jx}+e^{-jx}}{2} \quad \text{e} \quad \sin x = \dfrac{e^{jx}-e^{-jx}}{2j} \right.$

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} + \ldots$$

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

$$\sin(x) = \frac{x}{1!} - \frac{x^3}{3!} + \frac{x^5}{5!} \ldots$$

é mais simples usar exponenciais complexas em vez de cossenos e senos

$$\boxed{x(t) = \sum_{m=-\infty}^{+\infty} c_m e^{j2\pi m\frac{t}{T}} \, , \qquad c_m = \frac{1}{T}\int_0^T x(t)\, e^{-j2\pi m \frac{t}{T}} dt \, , \quad m=0,\pm1,\pm2,\ldots}$$

no formulário do teste

Para quem quizer saber mais!

$$\underbrace{x(t) = \sum_{m=-\infty}^{+\infty} c_m e^{j2\pi m \frac{t}{T}}}_{\text{combinação linear das funções periódicas } e^{j2\pi m \frac{t}{T}}} \quad , \text{ com } \quad c_m = \frac{1}{T}\int_0^T x(t)\, e^{-j2\pi m \frac{t}{T}} dt = \frac{1}{T}\int_{-T/2}^{+T/2} x(t)\, e^{-j2\pi m \frac{t}{T}} dt$$

conjugado

Produto interno $\langle x, y \rangle = \frac{1}{T}\int_0^T x(t)\, y^*(t)\, dt$

$$\langle e^{j2\pi n \frac{t}{T}}, e^{j2\pi m \frac{t}{T}} \rangle = \begin{cases} 1, & m = n \\ 0, & m \neq 0 \end{cases}$$

Norma $\quad ||x|| = \sqrt{\langle x, x \rangle}$

As funções $e^{j2\pi m \frac{t}{T}}$ são ortonormais (ortogonais umas às outras e normais, norma = 1)

Decompomos pois a função periódica $x(t)$ numa soma de funções periódicas elementares.

Interpretação da série clássica de Fourier

período T : $x(t) = x(T+t)$ período mínimo → frequência fundamental

frequência : $\frac{1}{T}$

$$x(t) = \sum_{m=-\infty}^{+\infty} c_m \, e^{j2\pi m \frac{t}{T}}$$

$m \neq 0$

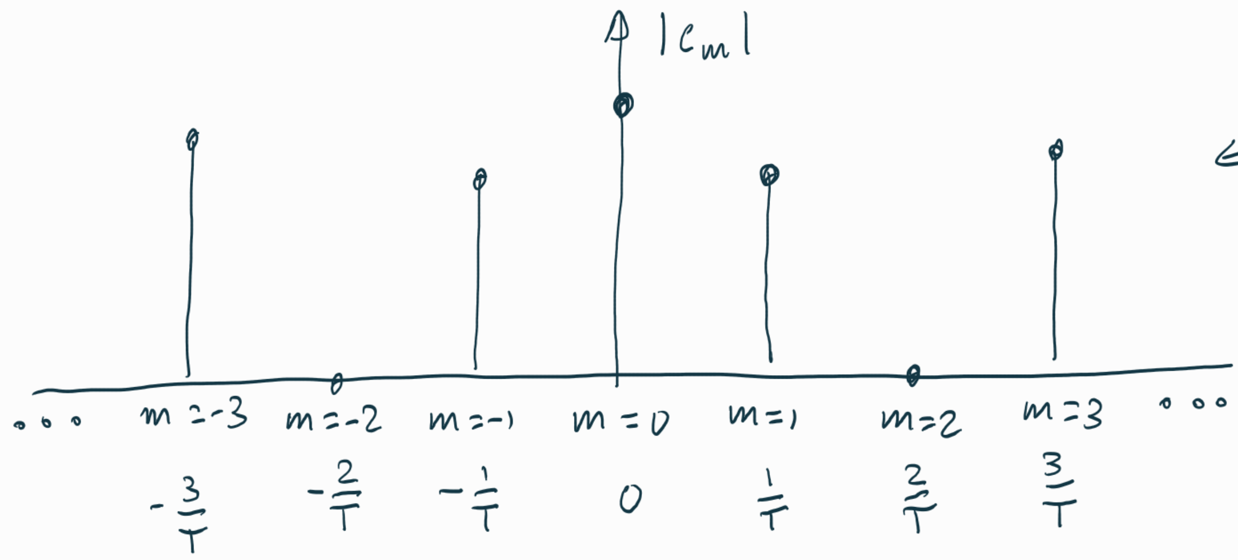
função com período $\frac{T}{|m|}$

ou seja,

com frequência $f = \frac{m}{T}$

A frequência mede-se em períodos por segundo (Hertz, Hz)

(dá aqui jeito ter frequência 0 (para $m=0$) e frequências negativas (para $m<0$))



ficamos a saber como é que a energia de $x(t)$ está espalhada pelas várias frequências.

[Frequência de Hz]

Exercício

Dado: $x(t) = 2\cos(4\pi t) - \sin(6\pi t)$

1) Qual é a frequência fundamental de $x(t)$?

2) Expanda $x(t)$ na sua série clássica de Fourier

3) Faça um gráfico de $|c_m|$ em função da frequência

Solução:

1) $\cos(4\pi t) = \cos\left(2\pi \frac{t}{T_1}\right) \rightarrow T_1 = 1/2$

$\sin(6\pi t) = \sin\left(2\pi \frac{t}{T_2}\right) \rightarrow T_2 = 1/3$

Período = mínimo múltiplo comum de T_1 e T_2

$\rightarrow k_1 T_1 = k_2 T_2$, k_1 e k_2 inteiros

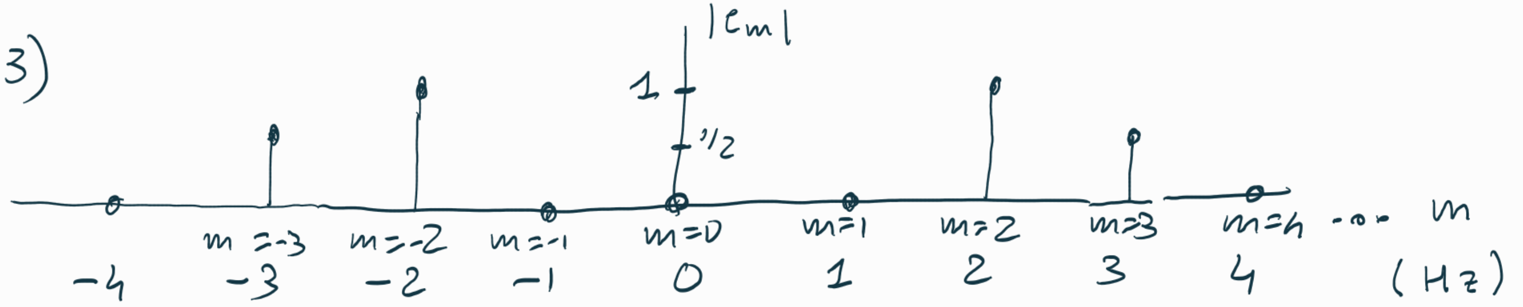
$\frac{k_1}{2} = \frac{k_2}{3} \rightarrow k_1 = 2$, $k_2 = 3$

período = 1 $\quad T = 1$

Ou: $Freq_1 = 1/(1/2) = 2\,Hz \quad Freq_2 = 3\,Hz$ $\quad Freq$ = máximo divisor comum de 2 e 3 = 1

2) $$x(t) = 2 \cdot \frac{e^{j2\pi 2t} + e^{-j2\pi 2t}}{2} - \frac{e^{j2\pi 3t} - e^{-j2\pi 3t}}{2j}$$

3)



A transformada de Fourier

E se $x(t)$ não for uma função periódica? Mas com $\int_{-\infty}^{+\infty} |x(t)|\,dt < \infty$.

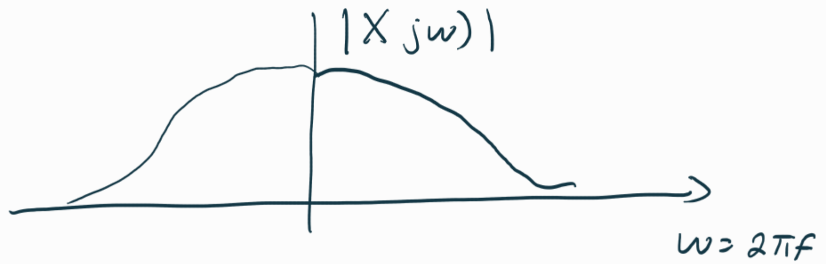
$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} X(j\omega)\, e^{j\omega t}\, d\omega \quad , \text{ com } \quad X(j\omega) = \int_{-\infty}^{+\infty} x(t)\, e^{-j\omega t}\, dt$$

É um caso particular da transformada <u>bilateral</u> de Laplace

$$X(s) = \int_{-\infty}^{+\infty} x(t)\, e^{-st}\, dt$$

Nota: ω é uma frequência angular = $2 \cdot \pi \cdot$ frequência linear. Mede-se em radianos por segundo

Basta fazer $s = j\omega$. $X(j\omega)$ também mostra como a energia de $x(t)$ está espalhada pelas várias frequências).



O teorema da amostragem

Considere o sinal $x(t) = \cos(2\pi f_0 t)$, que tem uma frequência de f_0 Hz.

O processo de amostragem consiste em fazer $t = nT_a$, onde T_a é o período de amostragem. Obtemos ($f_a = 1/T_a$ é a frequência de amostragem)

$$x_a(n) = x(nT_a) = \cos(2\pi f_0 n T_a) = \cos\left(2\pi \frac{f_0}{f_a} n\right)$$

é habitual omitir T_a

Só interessa $\frac{f_0}{f_a}$

Note que $\cos\left(2\pi \frac{f_0}{f_a} n\right) = \cos\left(\underbrace{2\pi \frac{f_0 + K f_a}{f_a} n}_{= 2\pi \frac{f_0}{f_a} n + \cancel{2\pi K n}}\right)$, $K \in \mathbb{Z}$

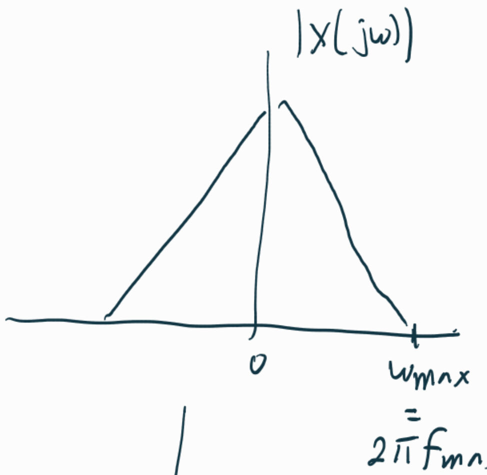
Para que não haja ambiguidade

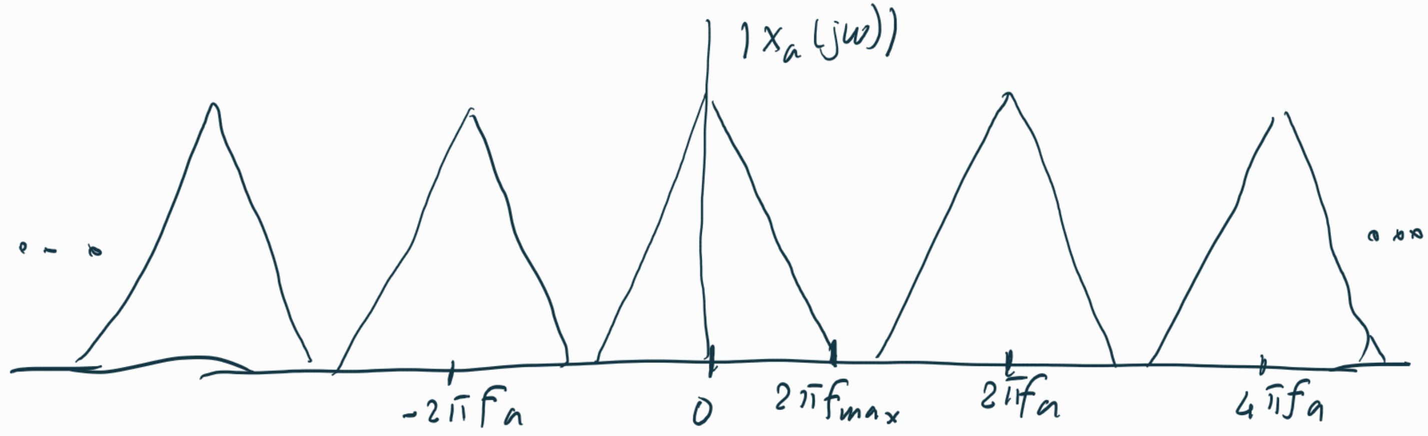
$\left|\frac{f_0}{f_a}\right| < 1/2$, ou seja, a frequência de amostragem deve ser maior que o dobro da frequência do sinal

Para quem quer saber mais!

$x(t) \longleftrightarrow X(j\omega)$

$x(nT_a) \longleftrightarrow X_a(j\omega) \simeq \underbrace{\sum_{k=-\infty}^{+\infty} X(j\omega + jk2\pi f_a)}_{\text{Passa a ser uma função periódica com período } 2\pi f_a \text{ !}}$





Nota: fazer algo deste género pode sair no teste

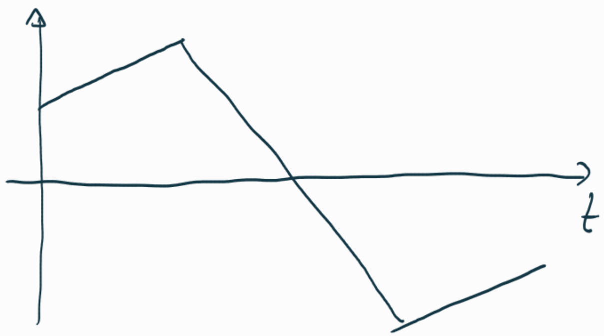
Se $f_a > 2 f_{max}$ não há sobreposição

$\downarrow$

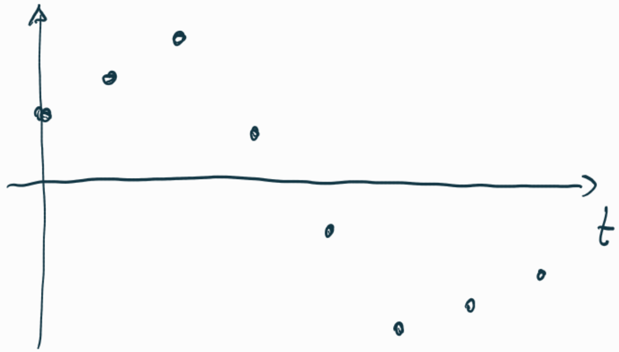
É possível voltar a recuperar $X(j\omega)$

e por isso é possível voltar a recuperar $x(t)$

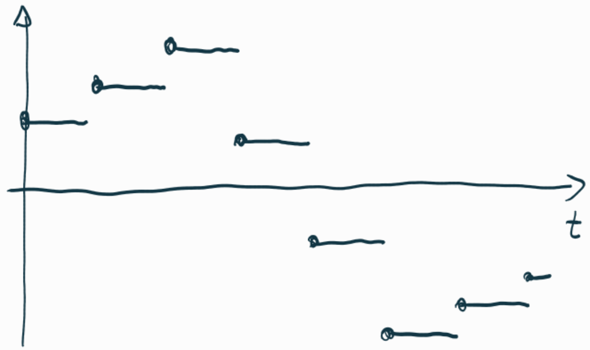
Amostragem e reconstrução de sinais



$\downarrow t = nT_a$



$\downarrow$ zero-order hold



A conversão analógica-digital (Analog to Digital Converter, ADC)

Faz isto e também <u>quantiza</u> as amplitudes

(mais à frente...)

A conversão digital-analógica (Digital to Analog Converter, DAC)

Faz isto

Pode também ser feita uma "filtragem passa-baixo" para eliminar a energia nas frequências $\geq f_a/2$

Exercício

Se o sinal $x(t) = \cos(8\pi t)$ for amostrado a 3 Hz, com que outros sinais é que se confunde? Esboce também o gráfico de $|X(j\omega)|$ (do sinal não amostrado)

Solução:

$x(t) = \cos(2\pi f_0 t)$, com $f_0 = 4$

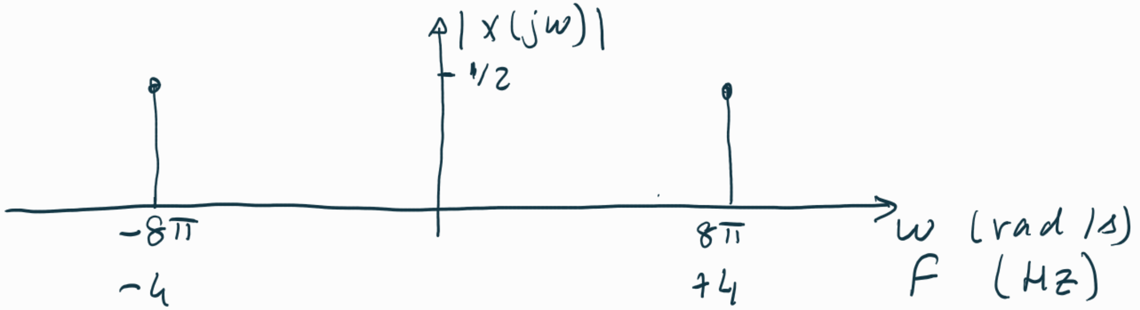
$x(nT_a) = \cos\left(2\pi \frac{f_0}{f_a} t\right)$, com $f_a = 3$

$= \cos\left(2\pi \frac{f_0 + k f_a}{f_a} t\right)$.

$$f_0 + k f_a = \begin{cases} \dots \\ -2, & \text{para } k=-2 \\ +1, & \text{para } k=-1 \\ +4, & \text{para } k=0 \\ +7, & \text{para } k=+1 \\ \dots \end{cases}$$

$x(t)$ confunde-se pois com outros cossenos com frequência $-5, -2, +1, +7, +10$, etc.

Quanto ao gráfico de $|X(j\omega)|$ temos $\cos(8\pi t) = \frac{\overbrace{e^{j2\pi 4t}}^{+4\text{ Hz}} + \overbrace{e^{-j2\pi 4t}}^{-4\text{ Hz}}}{2}$, pelo que



A transformada discreta de Fourier (DFT)

Seja $x(n)$ um sinal <u>amostrado</u> com um período de N amostras: $x(N+n) = x(n)$

Tanto n como N são números inteiros e N é positivo.

Tal como na série clássica de Fourier, podemos expandir $x(n)$ numa combinação linear, agora finita, de funções periódicas.

$$x(n) = \sum_{m=0}^{N-1} X(m)\, e^{j\frac{2\pi}{N}mn} \quad , \quad \text{com} \quad X(m) = \frac{1}{N} \underbrace{\sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi}{N}nm}}$$

transformada inversa (é quase igual...)

transformada

O MATLAB calcula isto de uma forma eficiente com a função fft (Fast Fourier Transform)

Interpretação da DFT

$x(t)$, sinal periódico com período T segundos

amostra-se este sinal, usando N amostras por período [não vamos tratar aqui o caso em que o número de amostras por período não é um número inteiro], $x_a(n) = x(nT_a)$

$T_a = T/N$, ou seja $f_a = N/T = N \cdot F$, onde F é frequência do sinal

A parcela $\underbrace{e^{j\frac{2\pi}{N}mn}}_{\text{vezes } X(m)}$ de $x_a(n)$ corresponde a $\underbrace{e^{j2\pi m\frac{t}{T}}}_{\text{vezes } c_m}$ da série clássica de $x(t)$

$$\boxed{\text{A frequência associada a } X(m) \text{ é então } \frac{m}{T} = \frac{m}{NT_a} = \frac{m}{N} f_a}$$

Importante saber

$m = 0 \longrightarrow$ frequência 0

$m = 1 \longrightarrow$ frequência f_a/N

em geral

$m \longrightarrow$ frequência $\frac{m}{N} f_a$

Não esquecer que $X(m)$ também é uma função periódica com período N !!!

$m = N-1$ é o mesmo que $m = -1$

Exercício

Um sinal periódico com uma frequência fundamental de 10 Hz foi amostrado 5 vezes num período. Os valores $X(m)$ da sua DFT são

m	$X(m)$
0	2
1	$3+4j$
2	-1
3	-1
4	$3-4j$

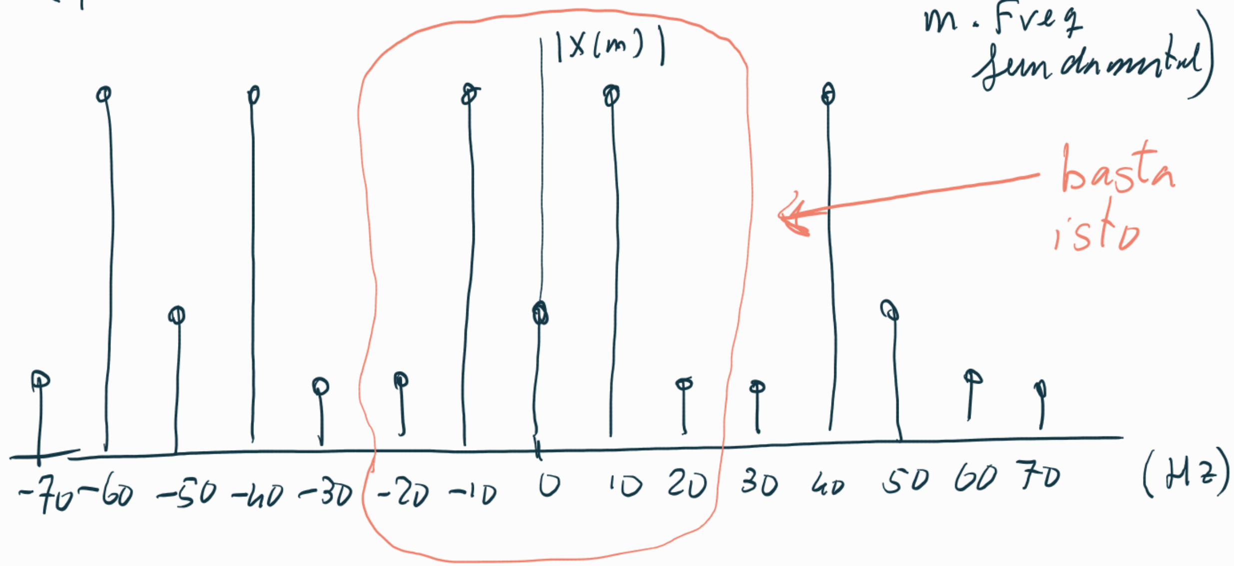
Desenhe o gráfico de $|X(m)|$ em que o eixo dos x está calibrado em Hz.

Solução: $f_a = \underset{N}{5} \times \underset{\text{frequência fundamental.}}{10} = 50$ Hz

$m \rightarrow$ frequência de $\frac{m}{N} f_a = mF = 10m$

(podia ser direto: $m \rightarrow$ frequência de m . freq fundamental)

m	$\lvert X(m) \rvert$	Freq
−5, 0, 5	2	−50, 0, 50, ...
−4, 1, 6	5	−40, 10, 60, ...
−3, 2, 7	1	−30, 20, 70, ...
−2, 3, 8	1	−20, 30, 80, ...
−1, 4, 9	5	−10, 40, 90, ...



Representações tempo-frequência

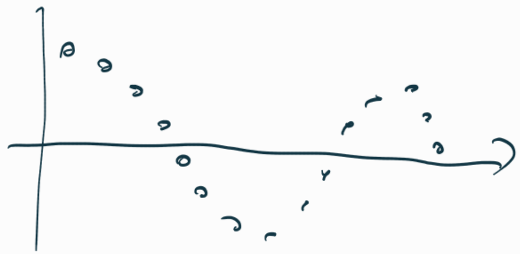
$\times$

janela deslizante ao longo do tempo $\rightarrow$

fica-se apenas com N amostras...

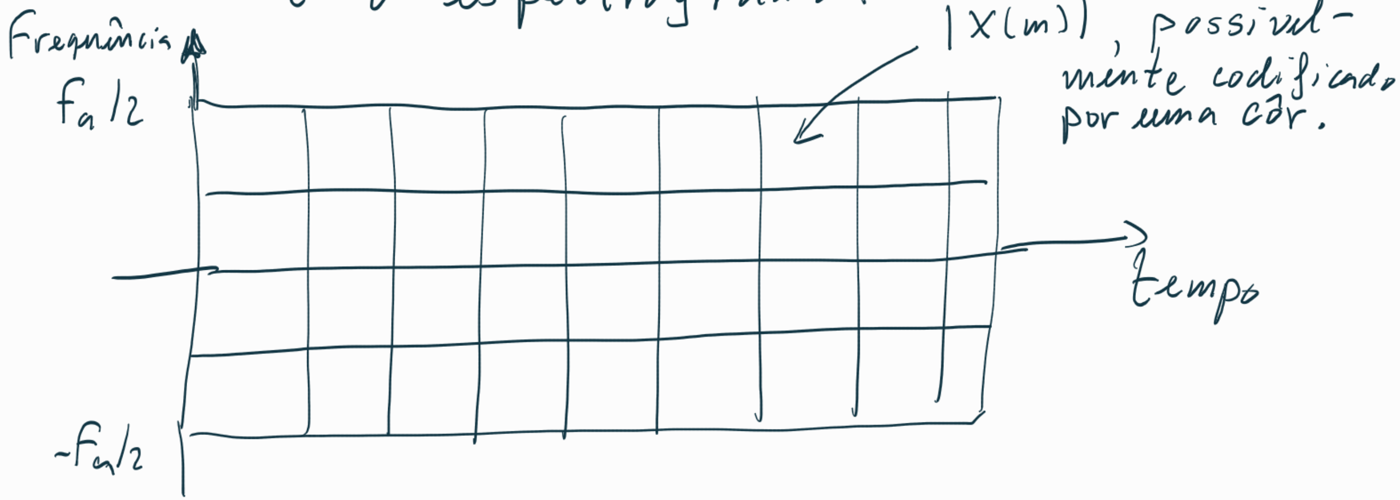
t_0

$$x(t_0, n) = x(n - t_0),$$
$$n = 0, 1, \ldots, N-1$$



DFT

Para cada posição da janela (informação temporal) obtemos os coeficientes da DFT (frequências). O resultado é o espectrograma:



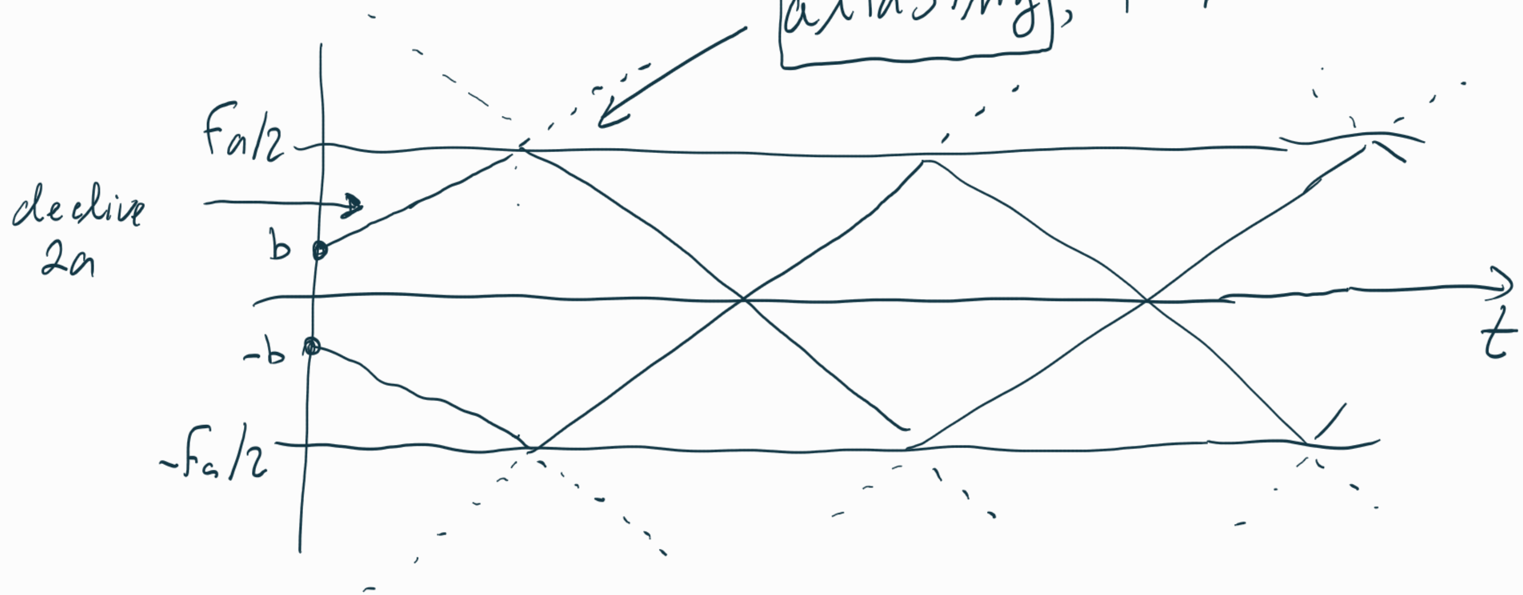
Exemplo:

$$\cos(2\pi\,\theta(t)), \text{ com } \theta(t) = at^2 + bt + c$$

Por analogia com $\cos(2\pi f_0 t)$ podemos considerar que a frequência instantânea de $\cos(2\pi\,\theta(t))$ é dada por $\frac{d\,\theta(t)}{dt} = 2at + b$.

Se $a > 0$, a frequência aumenta ao longo do tempo. Um esboço do espectrograma seria:

$\boxed{\text{aliasing}}$, porque a frequência seria maior que $f_a/2$.

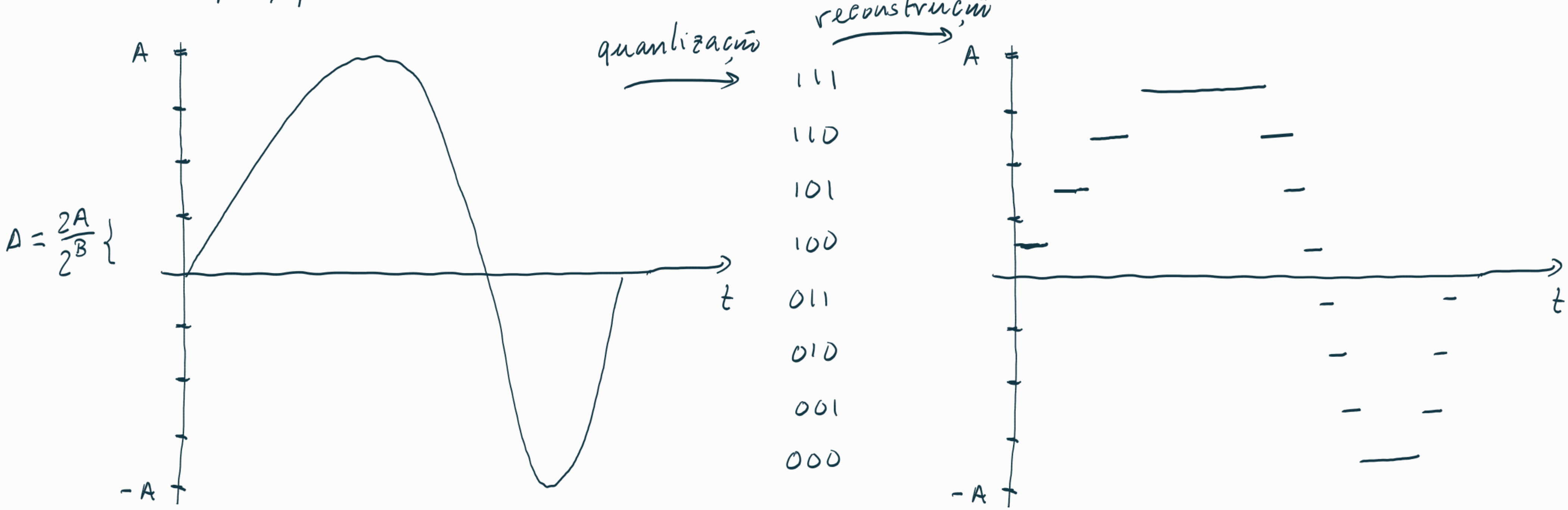


Quantização

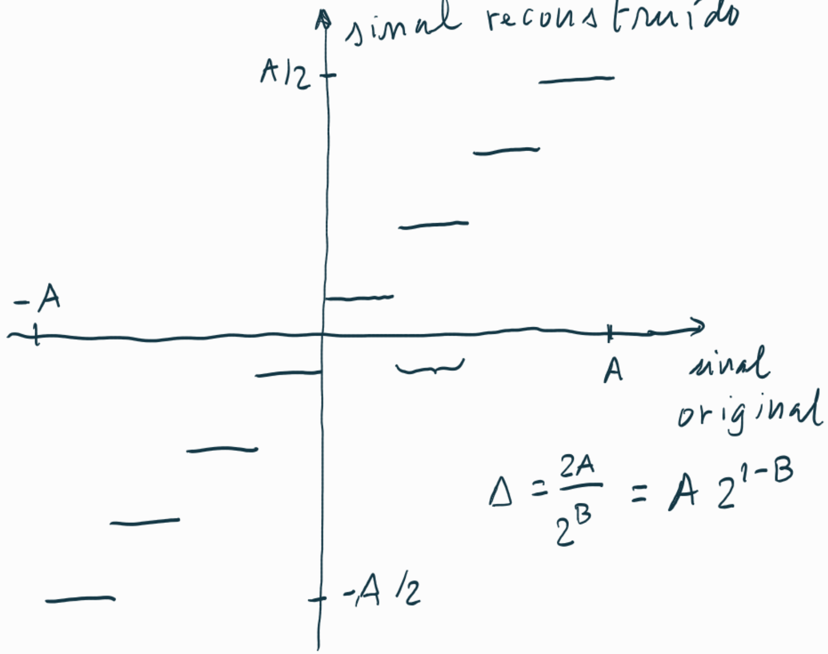
Além de se amostrar os sinais, também é preciso discretizar as suas amplitudes. Chama-se a isso quantização.

Considere que o sinal $x(n)$ tem amplitude A, isto é, que $|x(n)| \leq A$, e que se pretende quantizar $x(n)$ com B bits, isto é, com até 2^B valores distintos.

Por exemplo, para $B = 3$, teremos 8 níveis:

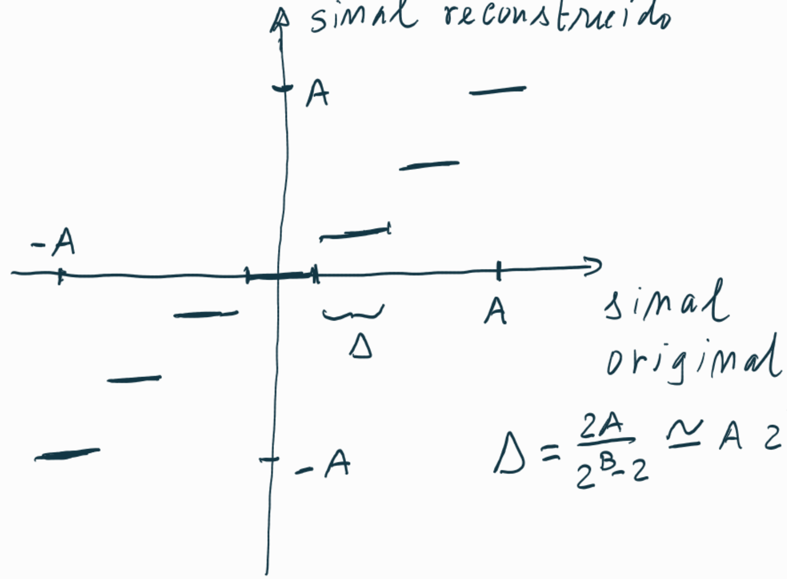


A maneira de fazer a quantização apresentada na página anterior tem o problema de zero não ser reconstruído com zero:



$$\Delta = \frac{2A}{2^B} = A\,2^{1-B}$$

É melhor algo do género ($2^B - 1$ níveis!)



$$\Delta = \frac{2A}{2^B - 2} \simeq A\,2^{1-B}$$

O sinal quantizado é $x_q(n) = \text{round}\,(x(n)/\Delta)$ e o sinal reconstruído é $x_r(n) = \Delta\, x_q(n) = \Delta\, \text{round}\,(x(n)/\Delta)$. O erro absoluto da quantização não pode ser maior do que $\Delta/2$ e o erro quadrático médio, pressuponto uma distribuição uniforme do erro de quantização, é

$$\int_{-\Delta/2}^{+\Delta/2} e^2 \frac{1}{\Delta}\, de = \frac{e^3}{3\Delta}\Big|_{-\Delta/2}^{+\Delta/2} = \frac{\Delta^2}{12}.$$

variância! (média nula)

densidade de probabilidade

$\boxed{\text{Por cada bit extra o erro de quantização decresce 6 dB}}$

Em decibeis (dB) temos erro quadrático médio $= 10 \log_{10} \frac{\Delta^2}{12} = 10 \log_{10} \frac{4A^2}{12} \frac{1}{4^B} = \text{Const} - 6.0206\,B$

Exercício

Pretende-se quantizar o sinal

$$x(n) = 2\cos(8\pi t + 1) - 2\sin(13\pi t) + \cos(17\pi t)$$

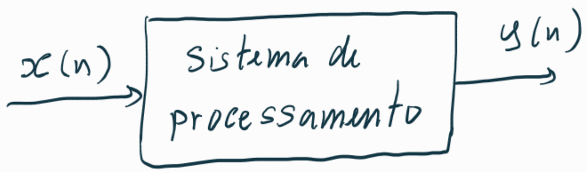
com 10 bits. Estime a amplitude do sinal, o passo de quantização (Δ), indique fórmulas para os sinais quantizado e reconstruídos, e estime os erros de quantização absoluto e quadrático médio.

- Amplitude: $|x(n)| \leq |2\cos(8\pi t+1)| + |-2\sin(13\pi t)| + |\cos(17\pi t)| = 2+2+1 = 5$
- Δ: $\Delta = \frac{2A}{2^B-2} = \frac{2 \cdot 5}{1024-2} \simeq \frac{10}{1000} = \frac{1}{100}$
- Sinal quantizado: $x_q(n) = \text{round}(x(n)/\Delta) = \text{round}(100\, x(n))$
- Sinal reconstruído: $x_r(n) = \Delta\, x_q(n) = 0.01\, \text{round}(100\, x(n))$
- Erro absoluto: $\Delta/2 = 0.005$
- Erro quadrático médio: $\frac{\Delta^2}{12} = \frac{0.0001}{12} \simeq 0.00001 = 10^{-5}$

Nota fora do exercício.

Também se podia falar na relação sinal/ruído, mas fórmulas concretas para isso dependem da forma do sinal, pelo que não falaremos disso neste resumo.

Processamento (filtragem) de sinais



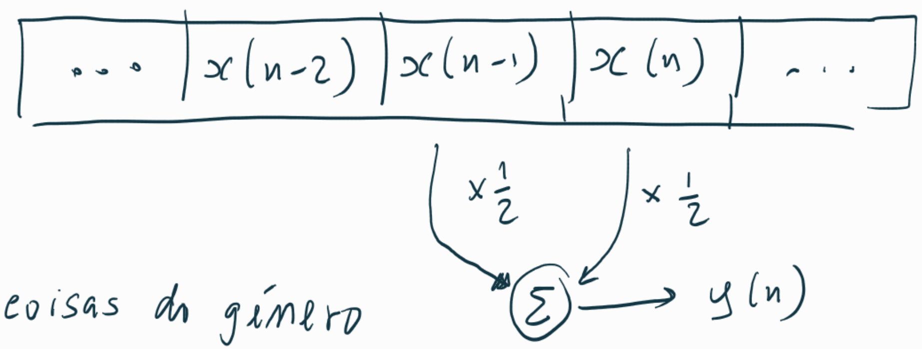
Exemplo: $y(n) = \frac{x(n) + x(n-1)}{2}$

amostra atual

amostra anterior

número da amostra atual

Implementação: guardam-se os vários $x(n-k)$, neste caso apenas 2, num array (circular!):



Também é possível ter coisas do género

$$y(n) = x(n) + 0.9\, y(n-1)$$

que é equivalente a

$$y(n) = x(n) + 0.9\left[x(n-1) + 0.9\, y(n-2)\right] = x(n) + 0.9\, x(n-1) + 0.9^2 x(n-2) + 0.9^3 x(n-3) + \dots$$

Em geral, temos $y(n) = \sum_{k=0}^{+\infty} h(k)\, x(n-k)$.

Efeito da filtragem

O que é que

$$x(n) \longrightarrow \boxed{h(k)} \longrightarrow y(n) = \sum_{k=0}^{+\infty} h(k)\, x(n-k)$$

faz às frequências de um sinal?

Nota: se $x(n) = \begin{cases} 1, & n=0 \\ 0, & n \neq 0 \end{cases}$ (um impulso)

pode-se verificar que $y(n) = h(n)$

Chama-se por isso a $h(k)$ resposta impulsional

Usando o sinal de teste $x(n) = e^{j2\pi F_0 n}$ (só uma frequência!) temos

$$y(n) = \sum_{k=0}^{+\infty} h(k)\, e^{j2\pi F_0 (n-k)} = \underbrace{e^{j2\pi F_0 n}}_{x(n)} \underbrace{\left[\sum_{k=0}^{+\infty} h(k)\, e^{-j2\pi F_0 k}\right]}$$

$\boxed{\text{número complexo que depende de } F_0 \text{. É a resposta em frequência do sistema}}$

E se o sinal de entrada for a soma de vários sinais destes?

Linearidade:

$$\left.\begin{aligned} x_1(n) &\longrightarrow y_1(n) \\ x_2(n) &\longrightarrow y_2(n) \end{aligned}\right\} \Longrightarrow \alpha_1 x_1(n) + \alpha_2 x_2(n) \Longrightarrow \alpha_1 y_1(n) + \alpha_2 y_2(n),$$

isto é, podemos analizar individualmente o que se passa para cada frequência.

Exercício

Sabe-se que o sinal de saída de um sistema é dado por

$$y(n) = x(n) + 2x(n-1) + x(n-2)$$

Esboce o gráfico do valor absoluto da resposta em frequência deste sistema.

Resposta:

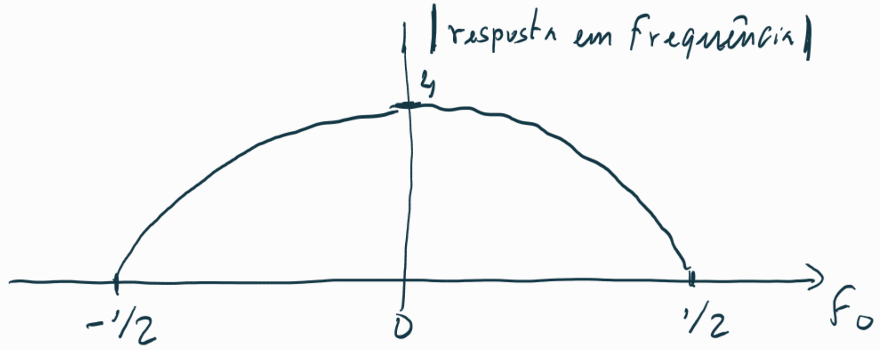
Para $x(n) = e^{j2\pi f_0 n}$ temos $y(n) = x(n)\Big[\underset{k=0}{1} + \underset{k=1}{2e^{-j2\pi f_0}} + \underset{k=2}{e^{-j4\pi f_0}}\Big]$

tem módulo 1

A resposta em frequência é então

$$1 + 2e^{-j2\pi f_0} + e^{-j4\pi f_0} = e^{-j2\pi f_0}\left[e^{+j2\pi f_0} + 2 + e^{-j2\pi f_0}\right] = e^{-j2\pi f_0}\left[2 + 2\cos(2\pi f_0)\right]$$

O seu valor absoluto é então $2\left[1+\cos(2\pi f_0)\right]$. O gráfico correspondente é



Representação de informação

Exemplo: "strings"

Em C:

's'	't'	'r'	'i'	'n'	'g'	's'	0

← array

Outra possibilidade:

7

← inteiro

's'	't'	'r'	'i'	'n'	'g'	's'

Mas precisamos de especificar como se codifica cada carater!

× ASCII × ISO latin1 (iso 8859-) • utf8

No que se segue vamos trabalhar com símbolos genéricos $s_0, s_1, s_2, \ldots$, que pertencem a um determinado alfabeto. Cada um deles representa um item de informação (letra, palavra, ...). A informação que queremos manipular (por exemplo, comprimir) será uma sequência destes símbolos.

Cada símbolo, s_k, terá uma probabilidade de ocorrência, p_k.

Entropia

Se $p_i = 0$ usa-se $p_i \log_2 p_i = 0$

Definição:

$$H = -\sum p_i \log_2 p_i$$

Quando se usa o logaritmo base 2 mede-se em bits.

Exemplo: 2 símbolos, S_0 e S_1, cada um com probabilidade $P_0 = P_1 = 1/2$.

$$H = -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} = -\frac{1}{2}(-1) - \frac{1}{2}(-1) = \frac{1}{2} + \frac{1}{2} = 1$$

Codificação da fonte (source code, aqui não tem nada a ver com programação!)

Pretende-se codificar cada símbolo S_k por um código binário C_k (em que cada C_k é uma sequência de bits), de modo a que a nossa informação possa ser representada pelo número médio de bits o mais baixo possível.

Mas queremos mais! A partir da sequência de códigos binários que representam a sequência de símbolos da nossa informação, queremos que a descodificação seja única e eficiente (por exemplo, não queremos que a descodificação de um símbolo dependa de um ou mais símbolos seguintes).

Códigos de Huffman

Algoritmo: enquanto existirem 2 ou mais símbolos, aglomeram-se os dois símbolos menos prováveis num novo símbolo, e usa-se um bit nos códigos binários respetivos para os distinguir. Esses códigos binários começam com o código do novo símbolo e terminam com o bit de desambiguação.

Exemplo:

Símbolo	Probabilidade
s_0	0.1
s_1	0.6
s_2	0.1
s_3	0.2

$\xrightarrow{0}$ (s_0), $\xrightarrow{1}$ (s_2)

s_{02}	0.2
s_1	0.6
s_3	0.2

$\xrightarrow{0}$ (s_{02}), $\xrightarrow{1}$ (s_3)

s_{023}	0.4
s_1	0.6

$\xrightarrow{0}$ (s_{023}), $\xrightarrow{1}$ (s_1)

s_{0123}	1

Os códigos binários obtêm-se andando para trás:

s_{0123}	

$\xleftarrow{0}$ (s_{023}), $\xleftarrow{1}$ (s_1)

s_{023}	$c_{023} = 0$
s_1	$c_1 = 1$

$\xleftarrow{0}$ (s_{02}), $\xleftarrow{1}$ (s_3)

s_{02}	$c_{02} = 00$
s_1	$c_1 = 1$
s_3	$c_3 = 01$

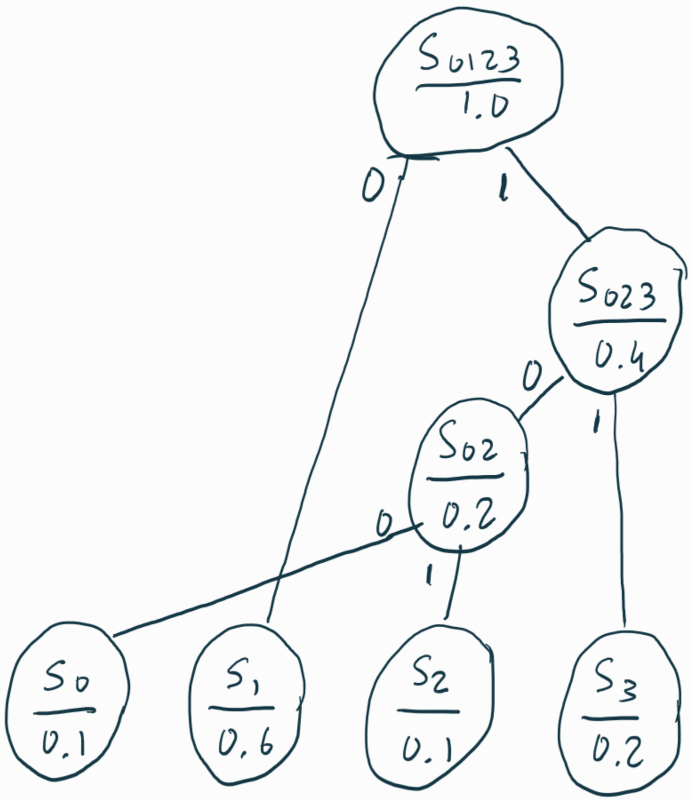
acrescenta-se o novo bit no fim

$\xleftarrow{0}$ (s_0), $\xleftarrow{1}$ (s_2)

Símbolo	Código
s_0	$c_0 = 000$
s_1	$c_1 = 1$
s_2	$c_2 = 001$
s_3	c_3 01

Também se podem organizar o que precisa de ser feito de outras maneiras. Por exemplo, usando uma árvore binária, temos

Legenda $\dfrac{\text{símbolo}}{\text{probabilidade}}$



De baixo para cima cada nível corresponde a aglomerar os dois nós da árvore que ainda não têm ascendentes.

A atribuição do código binário pode ser feita de uma maneira arbitrária: em cada bifurcação um dos lados fica com um bit a 0 e o outro lado fica com um bit a 1. Aqui estamos a pôr zeros à esquerda e uns à direita, o que dá um código diferente, mas equivalente, ao obtido na página anterior.

Símbolo	Código	Prob.
s_0	100	0.1
s_1	0	0.6
s_2	101	0.1
s_3	11	0.2

O número médio de bits por símbolo é

$$\underset{\text{nº bits}}{3} \times \underset{\text{prob}}{0.1} + 1 \times 0.6 + 3 \times 0.1 + 2 \times 0.2 = 1.6$$

Existe um teorema que diz que se os símbolos forem codificados um a um então

$$\boxed{H \leq \text{número médio de bits} < H+1}$$

No exemplo anterior

$$H = -\left(0.1 \log_2 0.1 + 0.6 \log_2 0.6 + 0.1 \log_2 0.1 + 0.2 \log_2 0.2\right) = 1.57095\ldots$$

pelo que o número médio de bits do código de Huffman está muito próximo do limite inferior.

Outro exemplo

(original)

Símbolo	Probabilidade	Código
S_0	0.9	0
S_1	0.1	1

considerando símbolos independentes

(novo)

Símbolo	Probabilidade	Código
$S_0 S_0$	0.81	0
$S_0 S_1$	0.09	10
$S_1 S_0$	0.09	110
$S_1 S_1$	0.01	111

$H = 0.46899\ldots$

número médio de bits por símbolo = 1

originais

número médio de bits por 2 símbolos = 1.29

número médio de bits por símbolo = 0.645

Exercício

Os símbolos S_0 a S_4 têm as seguintes probabilidades:

S_0	1/8
S_1	3/8
S_2	1/8
S_3	2/8
S_4	1/8

Calcule: a) Entropia b) código de Huffman

c) número médio de bits do código de Huffman

d) codifique $S_0 S_3 S_1 S_4 S_0 S_0$

e) descodifique 0010001110110

Sabe-se que $\log_2 3 \simeq 1.585$

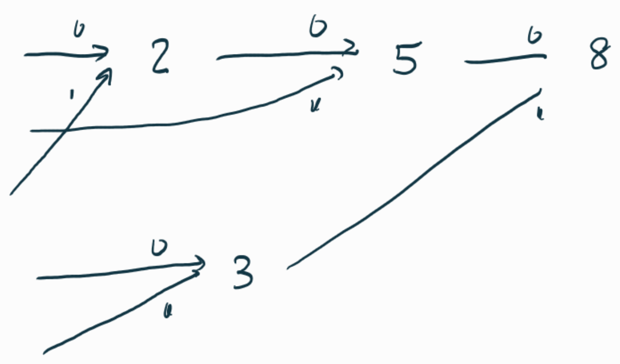
Solução

a) $-\log_2 \frac{1}{8} = 3$ $\quad -\log_2 \frac{3}{8} = 3 - 1.585 = 1.415$ $\quad -\log_2 2/8 = 2$

$$H = (3 + 3 \cdot 1.415 + 3 + 2 \cdot 2 + 3)/8 \simeq 2{,}156$$

b)

000	S_0	1
01	S_1	3
001	S_2	1
10	S_3	2
11	S_4	1



c) $(3\times 1 + 2\times 3 + 3\times 1 + 2\times 2 + 2\times 1)/8$

$= 9/4 = 2{,}25$

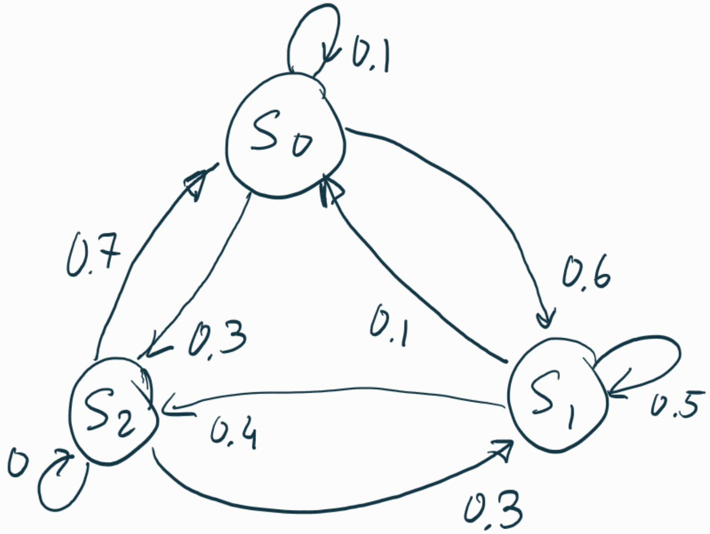
d) |000|10|01|11|000|000|

e) |001|000|11|10|11|0

S_2 S_0 S_4 S_3 S_4 incompleto

Para quem quer saber mais!

Códigos de Huffman e cadeias de Markov



Construimos 3 códigos de Huffman

- um para codificar um símbolo que vem a seguir a S_0

S_0	0.1	0.4	1.0	00
S_1	0.6			1
S_2	0.3			01

1.4 bits por símbolo

- outro para codificar um símbolo que vem a seguir a S_1

S_0	0.1	0.5	1.0	00
S_1	0.5			1
S_2	0.4			01

1.5 bits por símbolo

- e um outro para codificar um símbolo que vem a seguir a S_2

S_0	0.7	0
S_1	0.3	1
S_2	0.0	X

1.0 bits por símbolo

Outras maneira de compactar informação

Run Lenght Encoding (RLE)

Para compactar uma sequência de símbolos no RLE contam-se quantas vezes aparece consecutivamente o mesmo símbolo e envia-se o símbolo e a respetiva contagem. Por exemplo, para codificar (os símbolos são aqui letras maiúsculas)

A A A B B C A A A

envia-se

(A, 3) (B, 2) (C, 1) (A, 3)

Estes códigos são especialmente úteis quando existem poucos símbolos e quando a probabilidade de repetição é muito elevada.

Variante: temos 256 símbolos, pelo que cada símbolo é codificado num byte.

Envia-se dois tipos de informação:

1) 2 bytes → $\boxed{N \mid S}$, $N = 0, \ldots, 127$, $(N+1)$ repetições de S

N-125 bytes → $\boxed{N \mid S_1 \mid S_2 \mid \cdots \mid S_{N-126}}$, $N = 128, \ldots, 255$, $N-126$ símbolos (pelo menos 2)

usa-se quando aparecem 2 ou mais símbolos sem repetição

Codificação aritmética

Na codificação aritmética uma sequência de símbolos é codificada por um sub-intervalo do intervalo $[0,1[$. Envia-se apenas informação (por exemplo, centro do intervalo com precisão suficiente) para identificar sem ambiguidade o intervalo.

Exemplo (para facilitar as contas, vamos trabalhar em base 10 em vez de base 2)

Símbolo	Probabilidade	Intervalo
S_0	0.3	$[0.0, 0.3[$
S_1	0.1	$[0.3, 0.4[$
S_2	0.2	$[0.4, 0.6[$
S_3	0.1	$[0.6, 0.7[$
S_4	0.3	$[0.7, 1.0[$

↑ largura do intervalo

↑ limite superior de um intervalo = limite inferior do intervalo seguinte

$$\boxed{\text{número médio de bits por símbolo} = H}$$

Codificação de $S_2 S_4$

0.0 S_0 0.3 S_1 0.4 S_2 0.6 S_3 0.7 S_4 1.0

$$0.46 = 0.4 + 0.3 \times (0.6 - 0.4)$$
$$0.48 = 0.4 + 0.4 \times (0.6 - 0.4)$$
$$\ldots$$

0.40 S_0 0.46 S_1 0.48 S_2 0.52 S_3 0.54 S_4 0.60

$S_2 S_4$ corresponde ao intervalo $[0.54, 0.60[$

Envia-se, por exemplo, 0.57.

Para descodificar, $0.57 \in [0.4, 0.6[$ pelo que o primeiro símbolo é S_2. Expande-se esse intervalo e repete-se o raciocínio.

Exercício

Pretende-se codificar $S_1 S_0 S_2$ usando um codificador aritmético. Sabe-se que $P_0 = 0.1$, $P_1 = 0.5$, $P_2 = 0.3$ e $P_3 = 0.1$ (sim, há 4 símbolos). Descodifique 0.52

Solução:

codificar

0.0 S_0 0.1 S_1 0.6 S_2 0.9 S_3 1.0

0.10 S_0 0.15 S_1 0.40 S_2 0.55 S_3 0.60

0.100 S_0 0.105 S_1 0.125 S_2 0.145 S_3 0.150

basta enviar

0.13 ou 0.14

descodificar

Está errado

Corrija!

0.0 S_0 0.1 S_1 0.6 S_2 0.9 S_3 1.0

0.10 S_0 0.15 S_1 0.40 S_2 0.55 S_3 0.60

$0.52 \in [0.40, 0.55[$

Como não há mais algarismos termina-se a descodificação:

$S_1 S_2$

Compressão LZW (Lempel-Ziv-Welch)

Baseado num dicionário (uma string é uma sequência de 1 ou mais símbolos)

Para codificar:

1) iniciar o dicionário com todas as strings de comprimento 1.
2) P = empty string
3) repetir até se chegar ao fim dos dados para comprimir
   - 3a) C = próximo símbolo
   - 3b) Se PC estiver no dicionário P = concatenação de P e C. Voltar a 3)
   - 3c) enviar o índice correspondente a P, adicionar PC ao dicionário, P = C e voltar a 3)
4) enviar o índice correspondente a P

Notas:

- à medida que o dicionário vai ficando maior, o número de bits necessário para codificar os índices aumenta.
- é possível ter um índice especial para se reiniciar o dicionário

Demasiado complicado: não sai no teste

Para descodificar:

1) iniciar o dicionário com todas as strings de comprimento 1 (*)
2) P = primeiro índice. Enviar correspondente entrada do dicionário. C = empty string.
3) repetir até se chegar ao fim da informação a descomprimir
   3a) N = próximo índice
   3b) Se N ainda não estiver no dicionário
           S = entrada do dicionário correspondente a P concatenado com C
       Caso contrário
           S = entrada do dicionário correspondente a N
   3c) Enviar S
   3d) C = primeiro símbolo de S
   3e) Adicionar a concatenação de P com C ao dicionário
   3f) P = N
   3g) voltar a 3)

Notas:
- A descodificação parece mais complicada porque pode acontecer que se receba um índice que ainda não foi inicializado no dicionário. No entanto, nessa altura já existe toda a informação para o inicializar.

Exercício

Para um dicionário inicializado com A, B e C, comprima a string ABAACABA

Solução:

Dicionário

índice	string
0	A
1	B
2	C
3	AB
4	BA
5	AA
6	AC
7	CA
8	ABA

P	C	P+C	índice
	A	A	X
A	B	AB	0
B	A	BA	1
A	A	AA	0
A	C	AC	0
C	A	CA	2
A	B	AB	x
AB	A	ABA	3
A	x		0