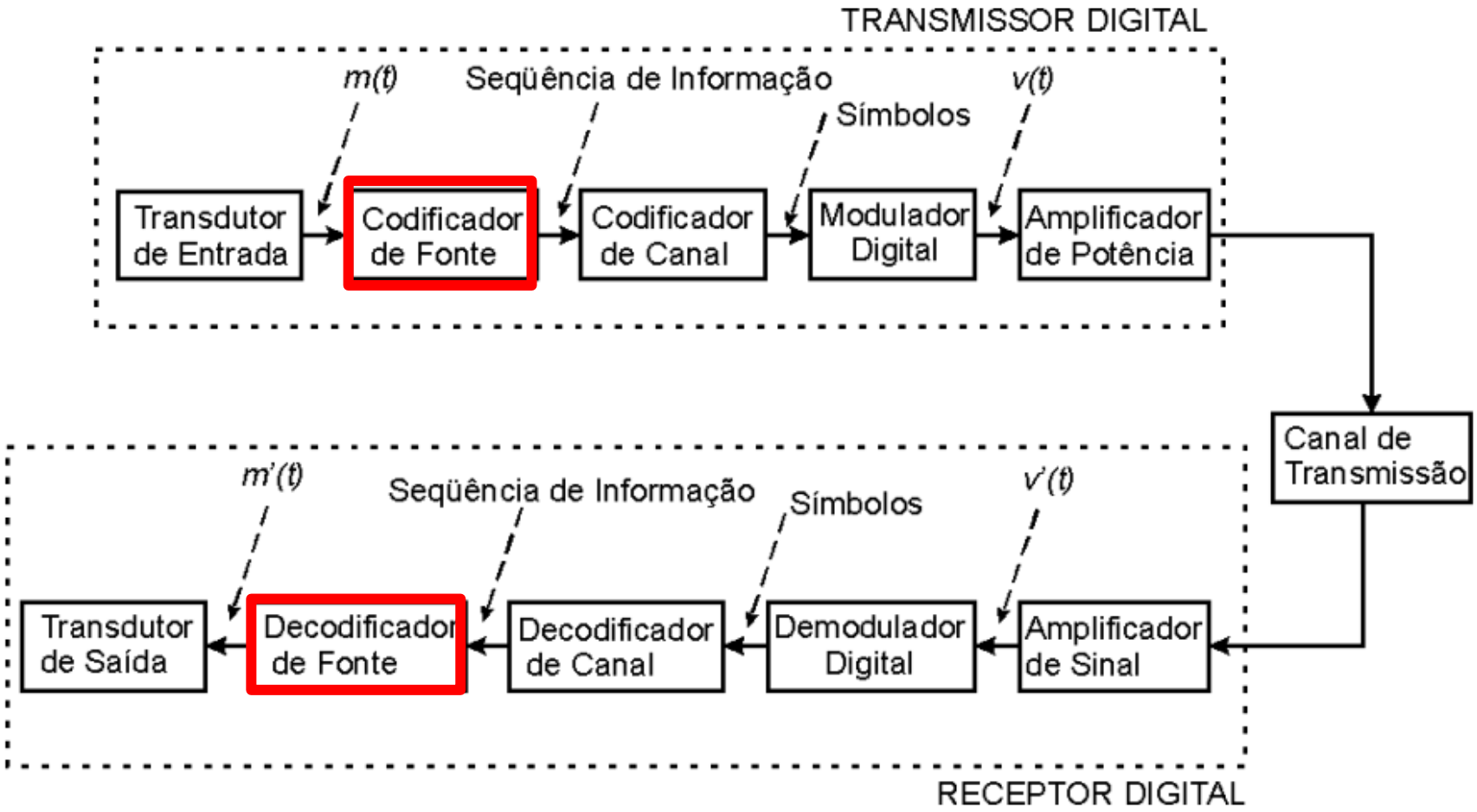
Codificação de Fonte: Conversão A/D, teorema da amostragem de Nyquist, DPCM (*differential pulse code modulation*) , DM (*delta modulation*) e ADM (*adaptive delta modulation*), codificação por entropia (código de Huffman)

Centro de Tecnologia – Departamento de Eletrônica e Computação

UFSM00261 – SISTEMAS DE COMUNICAÇÃO DIGITAL I

Prof. Fernando DeCastro

Codificação de Fonte



Conforme já brevemente discutido nos slides 3, 5, 6 e 7 do Cap I , o Codificador de Fonte (*source encoder*) efetua a digitalização (ADC) e a compressão da informação original p/ efeito de minimizar a banda ocupada no canal de transmissão.

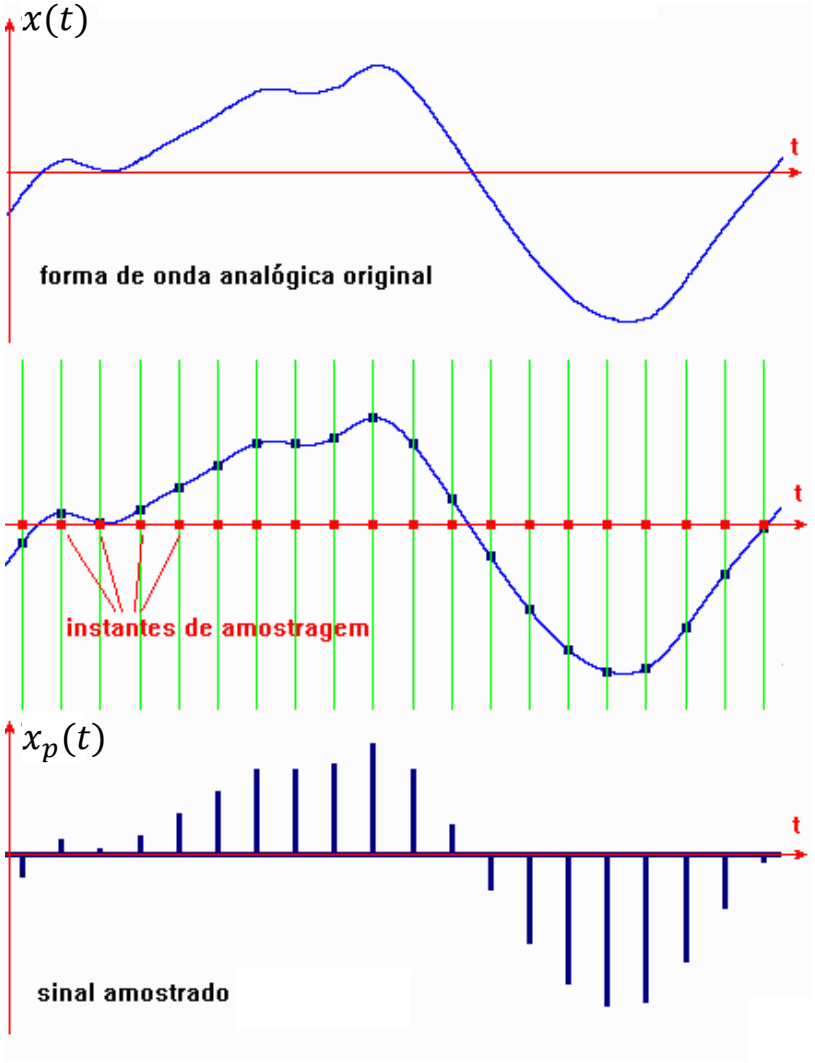
Especificamente, a Codificação de Fonte é o processo que visa reduzir o máximo possível a informação redundante da sequência de Informação em sua saída, sequência esta obtida a partir do processamento do sinal de entrada.

Conversão A/D

O processo de digitalização de um sinal analógico efetuado em um ADC (*analog to digital converter*) compreende 3 etapas que ocorrem em sequência: (I) **Amostragem** no tempo do sinal analógico, (II) **Quantização** das amplitudes do sinal amostrado, (III) **Codificação**, i.e., mapeamento de cada amostra do sinal quantizado em uma respectiva palavra binária. O processo de digitalização de um sinal analógico já foi estudado na disciplina de Sinais e Sistemas – ver https://www.fccdecastro.com.br/pdf/SS_Aula3_16032020.pdf . Faremos aqui uma breve revisão.

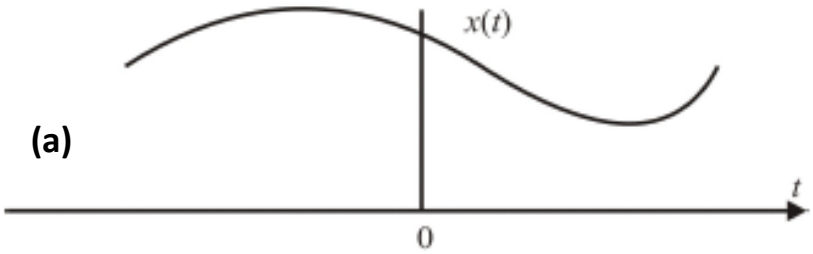
(I) Amostragem: Processo através do qual o sinal contínuo no tempo $x(t)$ é transformado em um sinal discreto no tempo, representado por $x_p(t)$ ou $x_p[n]$, onde n é interpretado como o instante de tempo no qual o valor do sinal $x(t)$ é levado à saída do processo de amostragem.

Nota: Nos próximos slides faremos uso do conceito de espectro de um sinal e do conceito de função de transferência de um sistema (no caso, um filtro passa-baixa). Para aqueles que ainda não cursaram a disciplina Sinais e Sistemas, é aconselhável estudarem com atenção o capítulo introdutório da referida disciplina disponível em https://www.fccdecastro.com.br/pdf/SS_Aula2_12032020.pdf como também assistir as videoaulas relativas a este capítulo introdutório disponíveis em https://www.fccdecastro.com.br/A2SSUFSM.html .

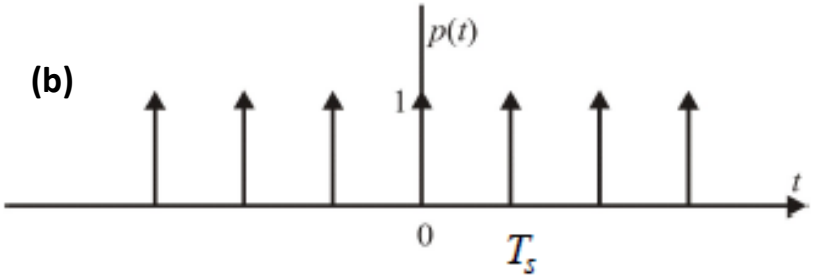


Conversão A/D

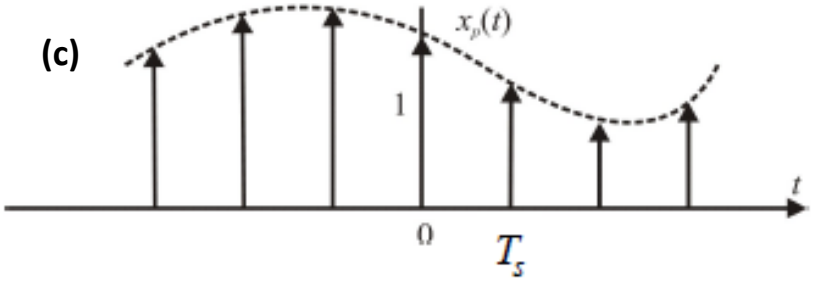
A amostragem no domínio do tempo de um sinal analógico $x(t)$ pode ser modelada pela multiplicação entre sinal $x(t)$ e uma sequência de impulsos periódicos $p(t)$ de período $T_s = 1/f_s$ [s], sendo f_s [Hz ou Sa/s – *samples per second*] a frequência de amostragem, conforme mostrado abaixo:



(a) sinal analógico $x(t)$ a ser amostrado

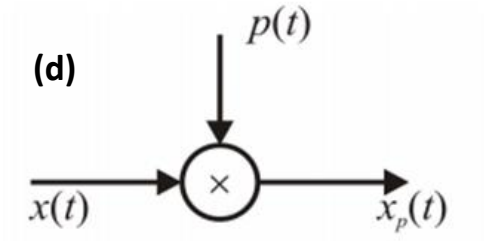


(b) função de amostragem $p(t)$



(c) operação de amostragem resultando no sinal amostrado $x_p(t) = p(t)x(t)$

(e) representação no domínio frequência da operação de amostragem $x_p(t) = x(t)\,p(t)$, onde $\mathcal{F}$ é o operador Transformada de Fourier. O operador "$*$" na equação (1) denota a operação de convolução (ver propriedade *multiplication* na tabela no slide 27 de https://www.fccdecastro.com.br/pdf/SS_Aulas9a12_27042020.pdf)



(e)

$$x(t).p(t) \overset{\mathcal{F}}{\longleftrightarrow} \frac{1}{2\pi} X(\omega) * P(\omega) \qquad (1)$$

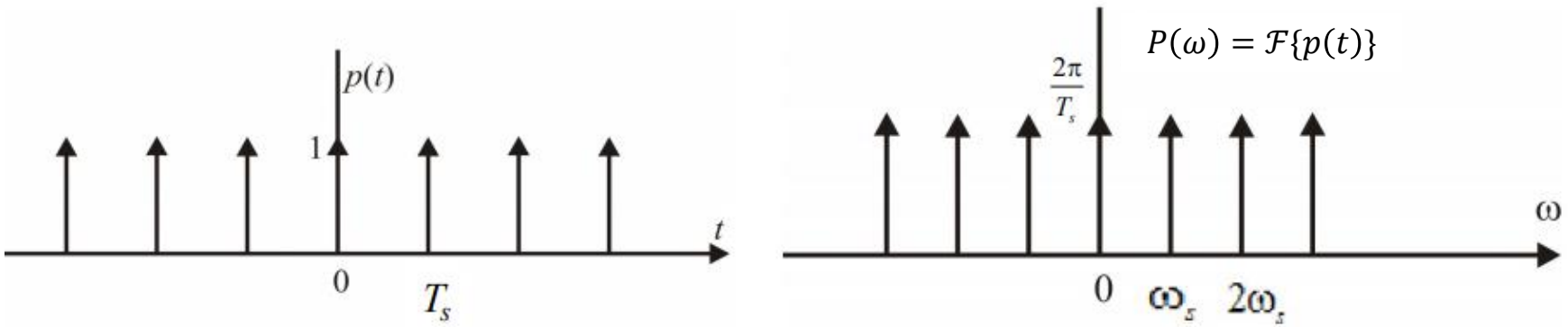
(d) representação da operação de amostragem $x_p(t) = p(t)x(t)$ através de um grafo de fluxo de sinal.

Conversão A/D

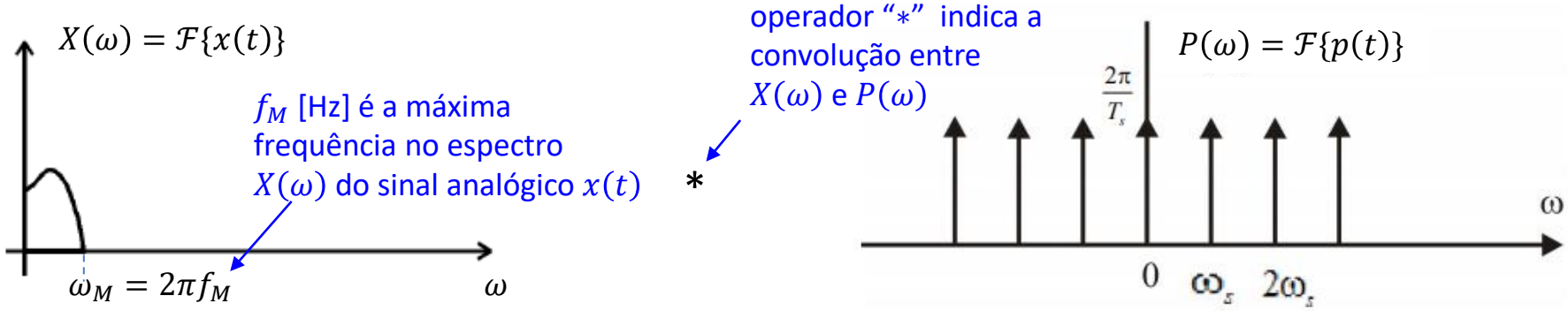
O espectro $P(\omega)$ no domínio frequência $\omega = 2\pi f$ do sinal $p(t)$ formado por impulsos periódicos de período T_s é dado por $P(\omega) = \mathcal{F}\{p(t)\}$, onde $\mathcal{F}\{\cdot\}$ é o operador que retorna a Transformada de Fourier do argumento $\{\cdot\}$, e resulta conforme equação (2) e figura abaixo (ver última linha da tabela no slide 29 de https://www.fccdecastro.com.br/pdf/SS_Aulas9a12_27042020.pdf):

$$p(t) \overset{\mathcal{F}}{\longleftrightarrow} P(\omega) = \frac{2\pi}{T_s} \sum_{k=-\infty}^{\infty} \delta(\omega - k\omega_s) \qquad (2)$$

sendo $\omega_s = 2\pi f_s$ e $f_s = 1/T_s$.



Da equação (1) no slide anterior, note que o produto $x(t)p(t) = x_p(t)$ no domínio tempo resulta na convolução $\frac{1}{2\pi}X(\omega) * P(\omega) = X_p(\omega)$ no domínio frequência, convolução que é efetuada entre o espectro $X(\omega) = \mathcal{F}\{x(t)\}$ do sinal analógico $x(t)$ e o espectro $P(\omega) = \mathcal{F}\{p(t)\}$ da função de amostragem $p(t)$ conforme mostra a figura abaixo:

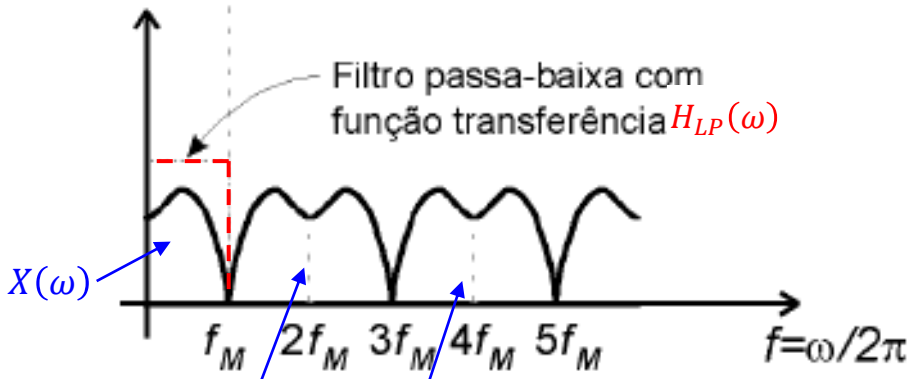


O resultado da convolução entre $X(\omega)$ e $P(\omega)$, i.e., o resultado de $X_p(\omega) = \frac{1}{2\pi}X(\omega) * P(\omega)$ referido no slide anterior é mostrado na figura abaixo p/ a situação em que $f_s = 2f_M$, onde f_M é a máxima frequência no espectro do sinal $x(t)$.

Note que o espectro $X_p(\omega)$ do sinal amostrado $x_p(t)$ é formado por múltiplos espectros transladados em frequência que são réplicas do espectro original $X(\omega) = \mathcal{F}\{x(t)\}$ do sinal analógico $x(t)$. Cada translação em frequência do espectro $X(\omega)$ ocorre em uma frequência central kf_s , onde $k = 1, 2, 3$ ...

Observe também que o espectro $X(\omega)$ do sinal analógico original $x(t)$ pode ser recuperado a partir do espectro $X_p(\omega)$ do sinal amostrado $x_p(t)$ aplicando-se um filtro passa-baixa, cuja magnitude da função de transferência $H_{LP}(\omega)$ é indicada pela "caixa quadrada" formada pela linha tracejada vermelha na figura abaixo. A função de transferência $H_{LP}(\omega)$ atenua todos os espectros transladados para as frequências centrais kf_s , $k = 1, 2, 3$ ..., mas deixa passar o espectro original $X(\omega)$ sem atenuação. Portanto, submetendo o sinal amostrado $x_p(t)$ a um filtro passa-baixa com função de transferência $H_{LP}(\omega)$, o sinal analógico original $x(t)$ pode ser recuperado.

$$X_p(\omega) = \frac{1}{2\pi}X(\omega) * P(\omega)$$



1ª réplica de $X(\omega)$ transladada em frequência e centrada em $f_s = 2f_M$

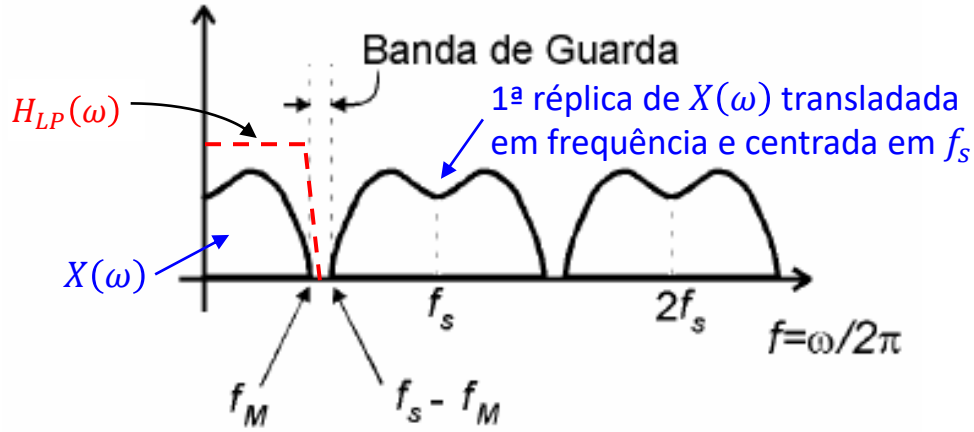
2ª réplica de $X(\omega)$ transladada em frequência e centrada em $2f_s = 4f_M$

A figura abaixo mostra o resultado da convolução $X_p(\omega) = \frac{1}{2\pi}X(\omega) * P(\omega)$ p/ a situação em que $f_s > 2f_M$, onde f_M é a máxima frequência no espectro do sinal $x(t)$.

Note que, como $f_s > 2f_M$, ocorre a formação de uma banda de guarda que separa a maior frequência f_M do espectro $X(\omega)$ original e a menor frequência $f_s - f_M$ da 1ª réplica de $X(\omega)$ transladada em frequência e centrada em f_s.

Observe que, para fins de recuperação do sinal analógico original $x(t)$ através de se um filtro passa-baixa com função de transferência $H_{LP}(\omega)$ referida no slide anterior, a existência de uma banda de guarda elimina a necessidade de que a curva da magnitude da função de transferência $H_{LP}(\omega)$ seja uma "caixa quadrada", facilitando a implementação do filtro que, nesta situação, pode ter uma curva de magnitude suave entre a faixa de passagem e a faixa de rejeição (filtros com curva de magnitude de $H_{LP}(\omega)$ na forma de uma "caixa quadrada" são difíceis de serem implementados, demandando muitos coeficientes).

$$X_p(\omega) = \frac{1}{2\pi}X(\omega) * P(\omega)$$

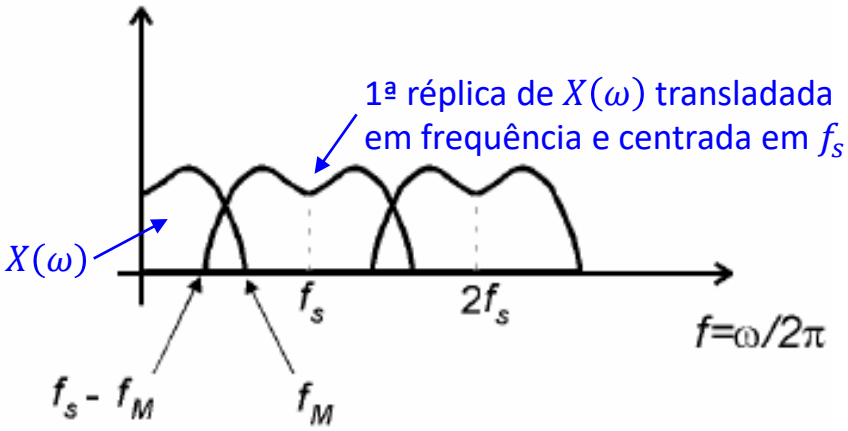


Vamos analisar agora a situação em que $f_s < 2f_M$, conforme mostrado na figura abaixo.

Note que para $f_s < 2f_M$ o espectro $X(\omega)$ original é superposto pela 1ª réplica de $X(\omega)$ transladada em frequência e centrada em f_s. Portanto, para esta situação torna-se impossível recuperar o sinal original através de um filtro passa-baixa porque o espectro $X(\omega)$ original é interferido pela superposição da 1ª réplica de $X(\omega)$ transladada em frequência e centrada em f_s, o que causaria distorção no sinal recuperado pelo filtro passa-baixa.

Esta distorção é denominada ***aliasing*** (*alias*: pseudônimo – em inglês), porque o espectro original $X(\omega)$ sofre interferência de uma réplica dele mesmo com "outro nome", isto é, sofre interferência dele mesmo, só que transladado em frequência.

$$X_p(\omega) = \frac{1}{2\pi} X(\omega) * P(\omega)$$

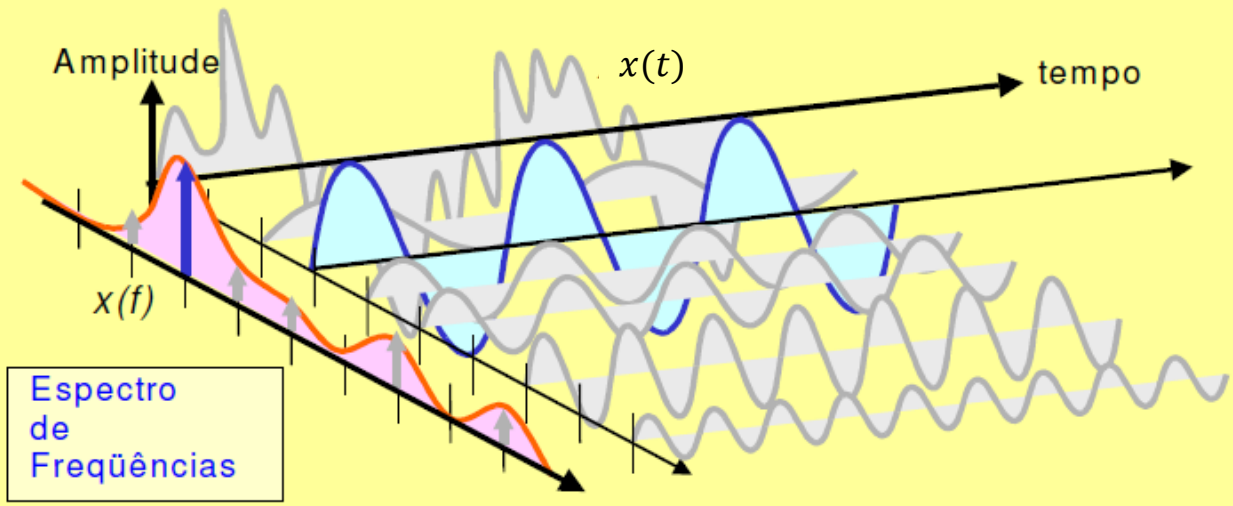


Teorema da Amostragem de Nyquist

Seja $x(t)$ um sinal limitado em banda, tal que f_M seja a frequência mais alta de seu espectro, frequência a partir da qual as componentes espectrais de $x(t)$ podem ser consideradas de magnitude desprezível.

Sejam os valores de $x(t)$ determinados a intervalos constantes de T_s segundos, tais que $T_s \leq \frac{1}{2f_M}$, isto é, $x(t)$ é periodicamente amostrado a cada $T_s \leq \frac{1}{2f_M}$ segundos. Desta forma, sendo $T_s \leq \frac{1}{2f_M}$, então a frequência de amostragem será maior ou igual a $2f_M$ ($f_s = 1/T_s \geq 2f_M$).

Uma vez obedecidas as condições acima as amostras $x(nT_s) = x[n]$ de $x(t)$ univocamente determinam $x(t)$, sendo $n = 0,1, \ldots$ o índice das amostras. Nesta situação, o sinal $x(t)$ pode ser reconstruído a partir do sinal amostrado $x[n]$ através de um filtro adequado.



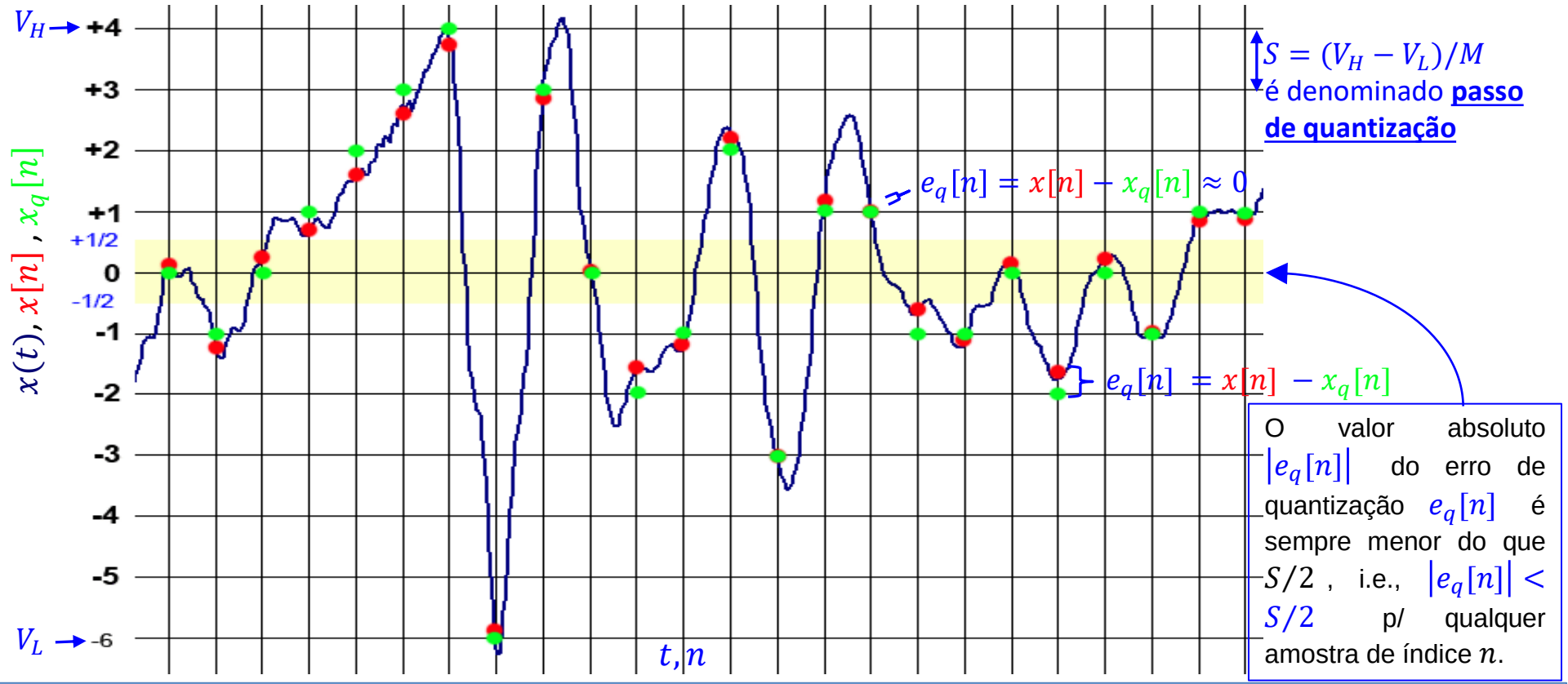
f_s = freqüência de amostragem

f_M = componente espectral de $m(t)$ de maior freqüência $x(t)$

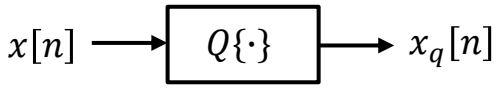
Se $f_s = 1/T_s \geq 2f_M$, após o processo de amostragem, o sinal $m(t)$ original pode ser recuperado $x(t)$ distorção, na saída de um filtro passa-baixa com freqüência de corte f_M, próximo do filtro ideal.

A frequência de amostragem mínima $f_s = 2f_M$ para que não haja ocorrência de *aliasing* é denominada de ***Nyquist Rate*** (**https://en.wikipedia.org/wiki/Nyquist_rate**). Não confundir *Nyquist Rate* com frequência de Nyquist (ver **https://en.wikipedia.org/wiki/Nyquist_frequency**)

(II) Quantização: Processo através do qual o sinal amostrado $x[n]$ contínuo em amplitude é transformado em um sinal $x_q[n]$ discreto em amplitude, viabilizando que o posterior processo "**(III) Codificação**" mapeie uma palavra binária de N bits a cada um dos $M = 2^N$ respectivos valores de amplitudes quantizados. Por exemplo, a figura abaixo mostra um sinal $x_q[n]$ quantizado em um conjunto de $M = 11$ possíveis amplitudes dadas por $m_k = \{-6, -5, -4, -3, -2, -1, 0, +1, +2, +3, +4\}$ [Volts], com $k = 0,1 \cdots M-1$. O sinal $x_q[n]$ quantizado é originado a partir da quantização de um sinal amostrado $x[n]$ contínuo em amplitude, que, por sua vez, é originado da amostragem no domínio tempo de um sinal analógico $x(t)$. A excursão de amplitude do sinal $x[n]$ varia entre os limites V_L e V_H. Especificamente, note na figura abaixo que para cada instante discreto n, o processo de quantização calcula M erros instantâneos de quantização dados por $e_q[n] = x[n] - m_k$, com k assumindo os valores $0,1 \cdots M-1$. O valor de m_k que resultar no menor $|e_q[n]|$ é então atribuído a $x_q[n]$.



Portanto, o processo de quantização pode ser representado pelo bloco funcional mostrado na figura abaixo



onde $x_q[n] = Q\{x[n]\}$, sendo $Q\{\cdot\}$ o operador definido por

$$Q\{\cdot\} = \arg\min_{m_k} |\{\cdot\} - m_k| \tag{3}$$

e onde $m_k = \{m_0, m_1 \cdots m_{M-1}\}$ é o conjunto de M amplitudes quantizadas (=níveis de quantização) c/ $k = 0,1 \cdots M-1$.

Desta maneira, lendo matematicamente a equação (3), o operador de quantização $Q\{\cdot\}$ executa o procedimento: Dado um valor do argumento $\{\cdot\}$ do operador $Q\{\cdot\}$ o operador determina os M valores resultantes da operação $|\{\cdot\} - m_k|$ para cada argumento m_k da operação $|\{\cdot\} - m_k|$ e então retorna o valor de m_k que resultou no menor $|\{\cdot\} - m_k|$.

Note que quanto maior for o número M de níveis de quantização menor será o passo de quantização $S = (V_H - V_L)/M$ e, portanto, menor será o erro de quantização $e_q[n] = x[n] - x_q[n]$ (ver slide anterior). Consequentemente, menor será o valor médio quadrático de $e_q[n]$, identificado por $\overline{e_q{}^2}$ e denominado de **potência do ruído de quantização** dada por (ver dedução na seção 2.2.1 de http://www.fccdecastro.com.br/pdf/cd2.pdf):

$$\overline{e_q{}^2} = \frac{S^2}{12} \quad \text{[Watts]} \tag{4}$$

Os valores do erro de quantização $e_q[n]$ são aleatórios porque as amplitudes do sinal analógico $x(t)$ são também aleatórias (um sinal de voz ou de vídeo, por exemplo). Portanto, em razão da aleatoriedade, o erro de quantização $e_q[n]$ pode ser interpretado como um ruído de quantização. Consequentemente, quanto maior for o número M de níveis de quantização menor será o passo de quantização S e mais $x_q[n]$ se assemelhará à $x[n]$ porque $x_q[n]$ será menos degradado pelo ruído aditivo de quantização. A relação sinal-ruído de quantização SNR_Q[dB] é dada por (ver dedução na seção 2.2.1 de http://www.fccdecastro.com.br/pdf/cd2.pdf):

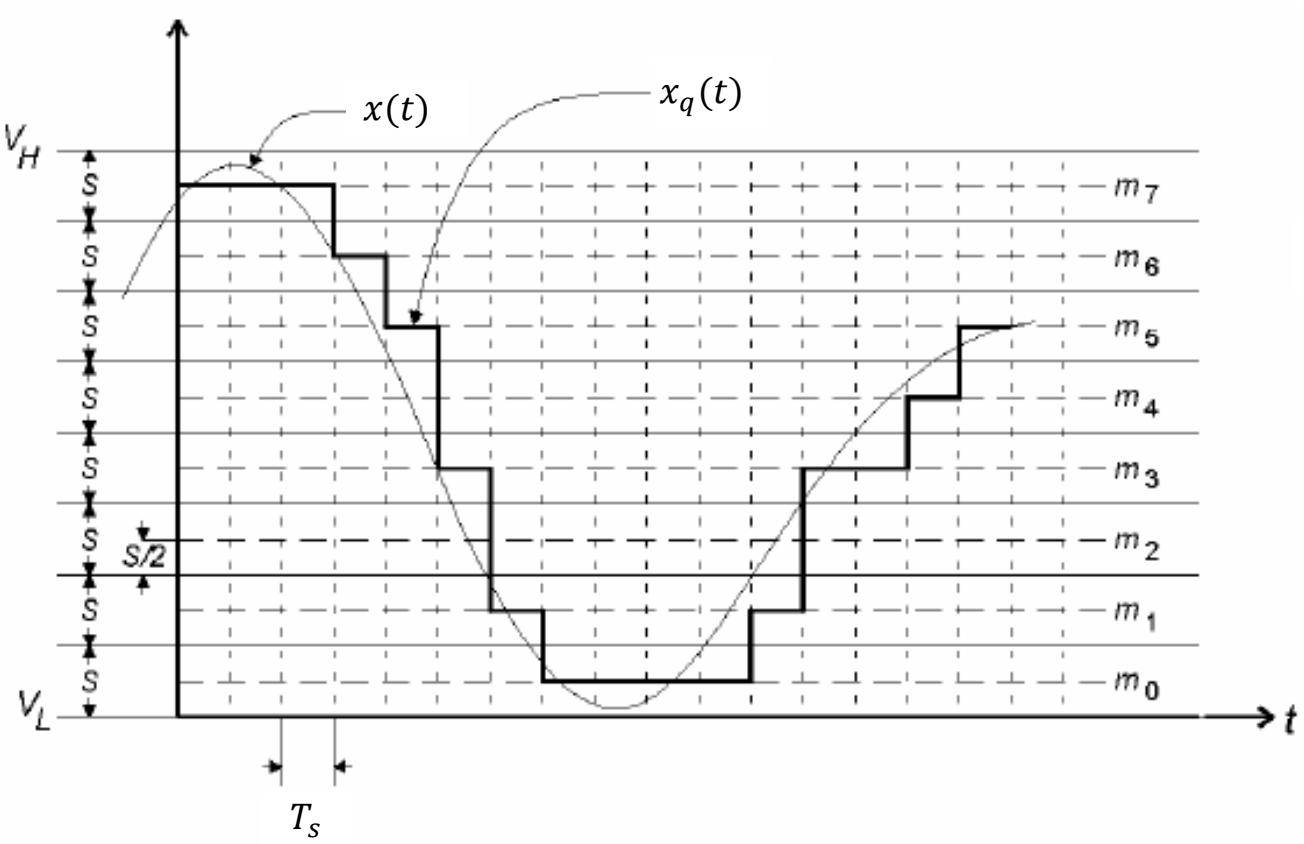
$$SNR_Q = 6N \text{ [dB]} \tag{5}$$

sendo $N = \log_2 M$ o número de bits da palavra binária respectivamente mapeada pelo posterior processo "**(III) Codificação**" a cada um dos $M = 2^N$ respectivos valores m_k de amplitudes quantizadas, sendo $k = 0,1 \cdots M-1$.

Conversão A/D

Por exemplo, a figura abaixo ilustra o processo de quantização de um conversor A/D (ADC) que utiliza palavras binárias de $N = 3$ bits/amostra ($M = 2^N = 8$ respectivos valores m_k de amplitudes quantizadas, $k = 0,1 \cdots M-1$). A relação sinal-ruído de quantização dada por (5) resulta $SNR_Q = 6N = 18$ [dB].

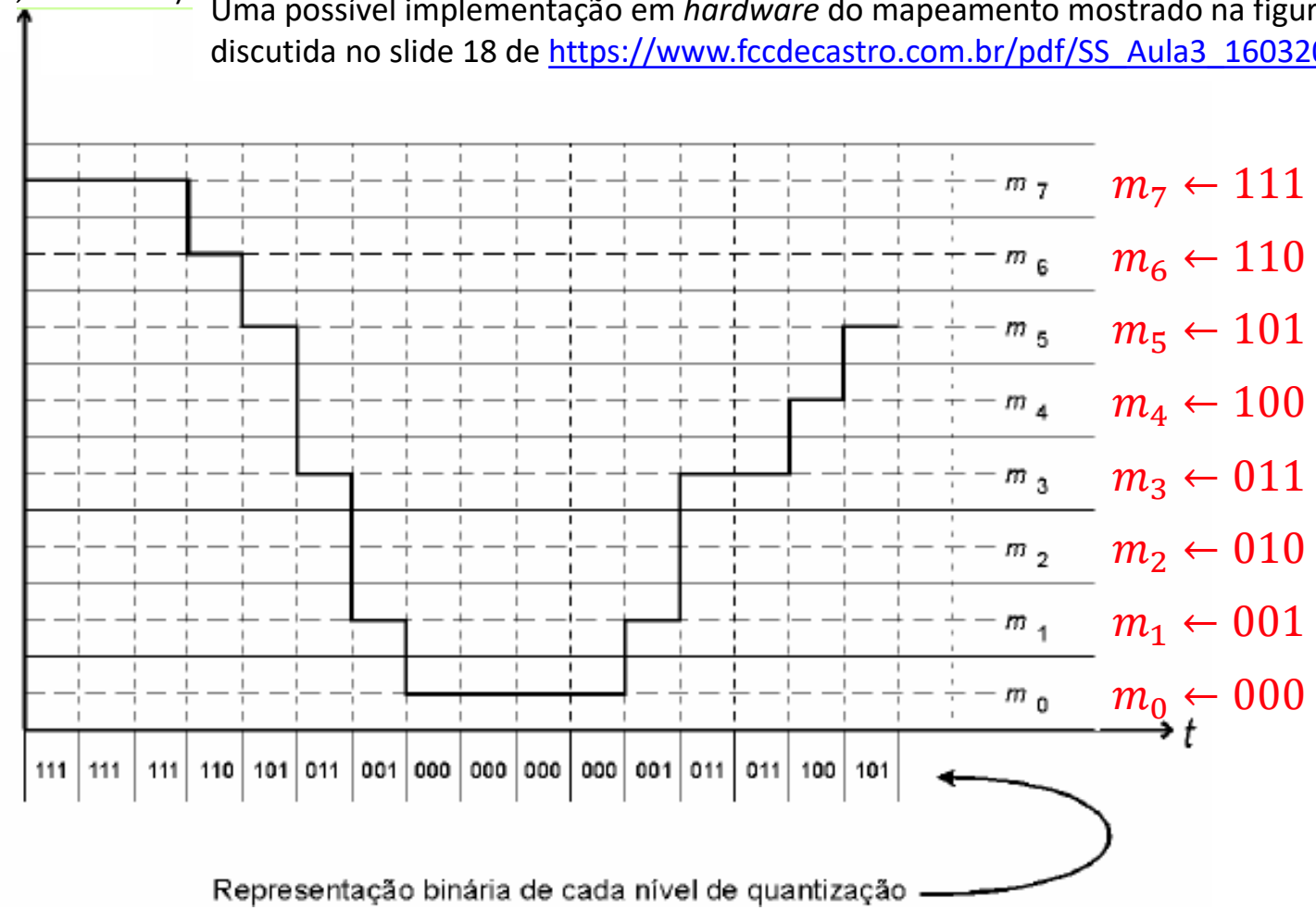
Se o ADC utilizasse palavras binárias de $N = 4$ bits/amostra ($M = 2^N = 16$ respectivos valores m_k de amplitudes quantizadas) obteríamos $SNR_Q = 6N = 24$ [dB], indicando que um ADC de $N = 4$ bits digitaliza com mais fidelidade o sinal analógico do que um ADC de $N = 3$ bits.



$$x(nT_s) = x[n]$$
$$x_q(nT_s) = x_q[n]$$

(III) Codificação (PCM – *Pulse Code Modulation*): Processo através do qual cada n-ésima amostra no sinal quantizado $x_q[n]$ é codificada (=mapeada) em uma respectiva palavra binária de $N = \log_2 M$ bits, sendo $M = 2^N$ o número de valores m_k de amplitudes quantizadas utilizadas no processo de quantização, com $k = 0,1 \cdots M-1$. Por exemplo, a figura abaixo mostra uma possível codificação adotada no mapeamento $\boldsymbol{m_k \leftarrow}$ **palavra binária** em um ADC hipotético que gera uma sequência de palavras binárias de $N = 3$ bits/amostra ($M = 2^N = 8$ respectivos valores m_k de amplitudes quantizadas, $k = 0,1 \cdots M-1$).

Uma possível implementação em *hardware* do mapeamento mostrado na figura abaixo é discutida no slide 18 de https://www.fccdecastro.com.br/pdf/SS_Aula3_16032020.pdf.

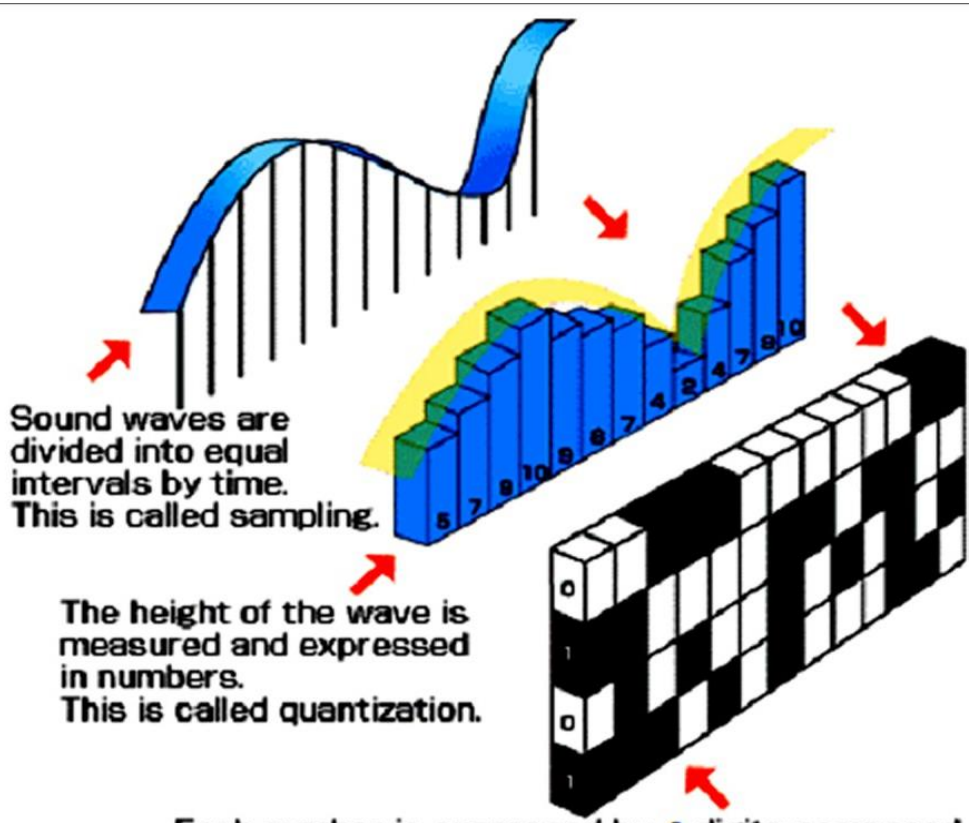


O processo de codificação $\boldsymbol{m_k \leftarrow}$ **palavra binária** é muitas vezes referido como **PCM – *Pulse Code Modulation*.**

Pulse Code Modulation (PCM) seguida de compressão de dados

Após o processo de codificação PCM, na grande maioria das situações práticas alguma forma de compressão é aplicada na sequência de palavras binárias gerada. A compressão por entropia (que estudaremos adiante neste capítulo), mostrada abaixo em um exemplo para um sinal de áudio, é uma das possíveis formas de compressão de dados utilizada. Note abaixo na coluna "Codificação original" que as palavras binárias de 4 bits com maior frequência de ocorrência são re-mapeadas em palavras binárias na coluna "Compressão (Código)" com menor número de bits. Este processo reduz, portanto, o número de bits necessários na sequência de bits que representa o sinal.

Amostra quantizada	Codificação original	Nº. de Ocorrências	Nº. de Bits Necessários	Compressão (Código)	Nº. de Bits Necessários
0	1111	0	0	001	0
1	0110	0	0	010	0
2	0010	1	4	10	2
3	1110	0	0	100	0
4	0100	2	8	00	4
5	0101	1	4	11	2
6	0001	0	0	111	0
7	0111	3	12	0	3
8	0000	1	4	000	3
9	1001	3	12	1	3
10	1010	2	8	01	4
...	...	...	...	...	...
15	1000	0	0		0
		(13)	(13x4=52)		(21)



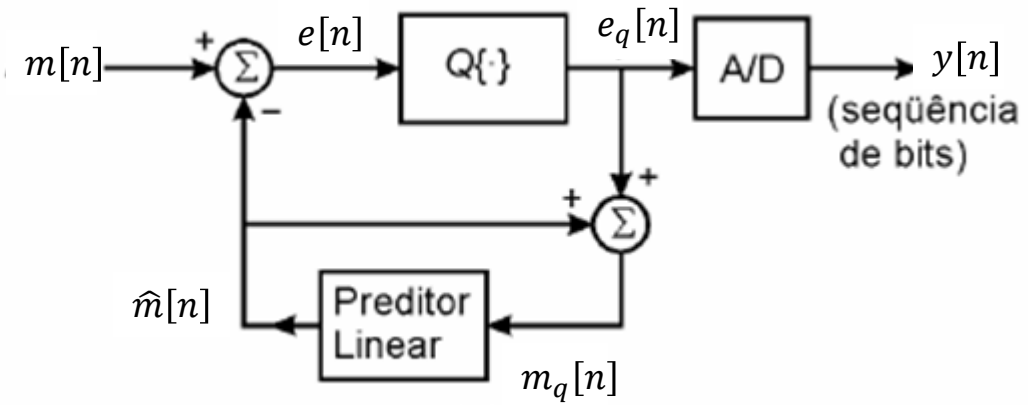
(explo.: sinal de voz $f_M = 3.3k\,\mathrm{Hz}$, $f_s = 8.0k\,\mathrm{Hz}$)

Pulse Code Modulation com compressão diferencial de dados (DPCM – *Differential* PCM)

Quando um sinal analógico $m(t)$ de áudio ou vídeo é amostrado a uma frequência de amostragem f_s muito maior do que o $Nyquist\ Rate = 2f_M$ (ver slide 9), o sinal amostrado $m[n]$ resultante exibe uma alta correlação entre amostras adjacentes. O significado desta alta correlação é que, na média, o sinal não varia rapidamente de uma amostra para a outra.

A consequência disto é que, quando amostras altamente correlacionadas são codificadas em um codificador PCM padrão, a sequência resultante de bits apresentará informação redundante desnecessária. Portanto, mensagens que não são absolutamente essenciais à transmissão de informação são geradas como resultado do processo de codificação. Remover esta redundância resulta em um aumento da eficiência do sinal codificado em transportar informação, no sentido de que serão necessários menos bits para transportar a informação.

Se conhecemos uma parcela suficiente do sinal redundante $m[n]$ podemos inferir o resto do sinal, ou pelo menos tentar fazer uma estimativa provável. Em particular, se conhecemos o comportamento passado de um sinal até um determinado ponto no tempo, então é possível fazer alguma inferência sobre seus valores futuros. Tal processo de inferência é conhecido como **predição**. Embora existam inúmeros métodos de predição, no contexto de codificação DPCM nos limitaremos à denominada **Predição Linear**. Em um codificador DPCM, situado no TX digital, o bloco "Preditor Linear" efetua a predição de uma amostra futura que é obtida como uma combinação linear de um conjunto de amostras passadas, conforme indicado abaixo no diagrama funcional de um codificador DPCM:

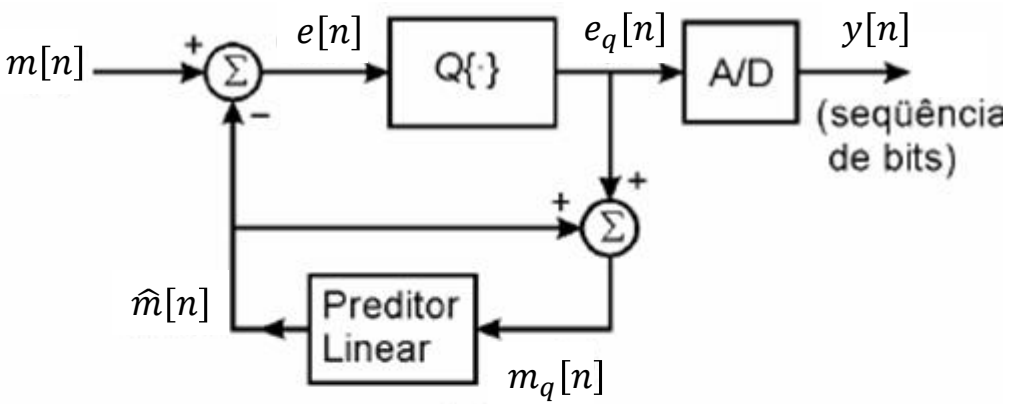


Pulse Code Modulation com compressão diferencial de dados (DPCM – *Differential* PCM)

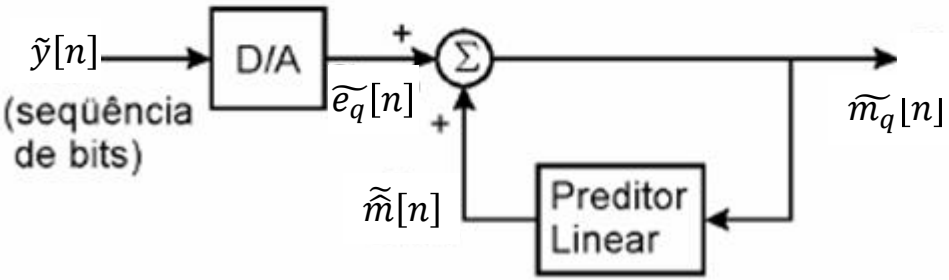
Consideremos a situação em que o sinal analógico $m(t)$ seja amostrado a uma frequência de amostragem $f_s = 1/T_s$ muito maior do que o $Nyquist\ Rate = 2f_M$ (ver slide 9). Esta super-amostragem produz uma sequência de amostras $m[n]$ altamente correlacionadas e espaçadas no tempo de um intervalo T_s.

Esta alta correlação entre as amostras da sequência $m[n]$ na entrada do codificador DPCM mostrado em (a) ao lado, permite que o decodificador DPCM mostrado em (b) efetue a predição de valores futuros de $m[n]$ a partir da sequência de bits $y[n]$ recebida do codificador DPCM e gerada no codificador através da diferença entre o sinal original $m[n]$ e sua predição $\hat{m}[n]$.

Note em (a) que o a entrada do quantizador $Q\{\cdot\}$ não é o sinal a ser codificado, mas sim o sinal de erro $e[\mathrm{n}] = m[\mathrm{n}] - \hat{m}[\mathrm{n}]$, que é a diferença entre o sinal super-amostrado $m[\mathrm{n}]$ na entrada do codificador e a predição $\hat{m}[\mathrm{n}]$ do sinal $m[\mathrm{n}]$.



(a) Codificador DPCM (no TX)



(b) Decodificador DPCM (no RX)

No codificador DPCM em (a), o sinal $\hat{m}[\mathrm{n}]$ é o resultado do processo de predição linear aplicado sobre um conjunto de amostras passadas do sinal $m_q[n]$, este último sendo uma versão quantizada do sinal $m[\mathrm{n}]$. O sinal de erro $e[n]$ é denominado de **erro de predição** visto que seu valor representa a incapacidade do Preditor Linear em predizer $m[\mathrm{n}]$ com exatidão. $Q\{\cdot\}$ efetua o processo de quantização do erro de predição $e[n]$, gerando o erro de predição quantizado $e_q[n]$. Como ocorre em todo e qualquer processo de quantização, um erro de quantização na forma de um ruído aleatório $q_e[n]$ é superposto ao sinal de saída do quantizador $Q\{\cdot\}$, de modo que a saída do quantizador $Q\{\cdot\}$ pode ser descrita como

$$e_q[n] = Q\{e[n]\} = e[n] + q_e[n] \qquad (6)$$

Pulse Code Modulation com compressão diferencial de dados (DPCM – *Differential* PCM)

O sinal de entrada do Preditor Linear do codificador DPCM mostrado em (a) é formado pela soma do sinal predito $\hat{m}[n]$ e da saída do quantizador $e_q[n]$, i.e.:

$$m_q[n] = \hat{m}[n] + e_q[n] \qquad (7)$$

Substituindo (6) em (7):

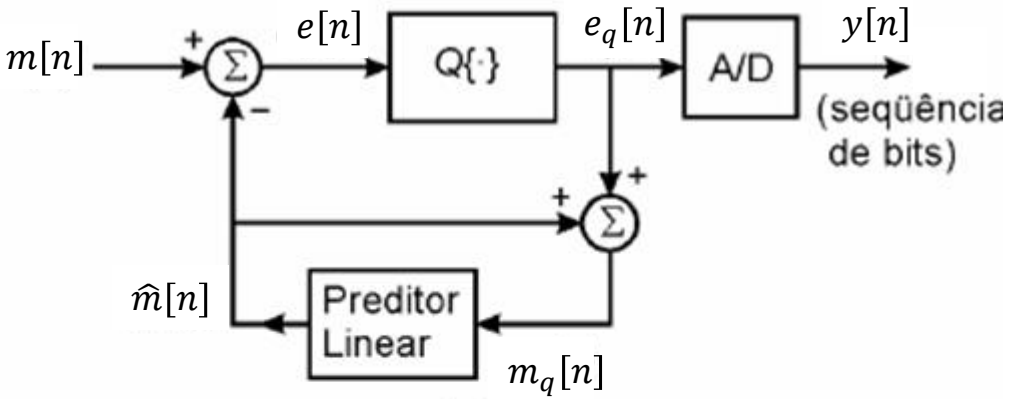
$$m_q[n] = \hat{m}[n] + e[n] + q_e[n] \qquad (8)$$

E do somador à esquerda em (a) temos que $e[n] = m[n] - \hat{m}[n]$, logo, substituindo em (8) obtemos:

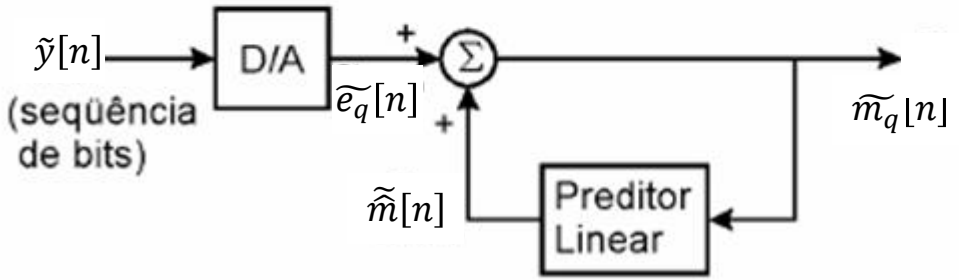
$$m_q[n] = m[n] - e[n] + e[n] + q_e[n]$$

$$m_q[n] = m[n] + q_e[n] \qquad (9)$$

Portanto, de (9), infere-se que não importa a capacidade de predição do Preditor Linear no codificador mostrado em (a): o sinal $m_q[n]$ aplicado na sua entrada difere do sinal $m[n]$ original apenas do valor do erro de quantização $q_e[n]$.



(a) Codificador DPCM (no TX)



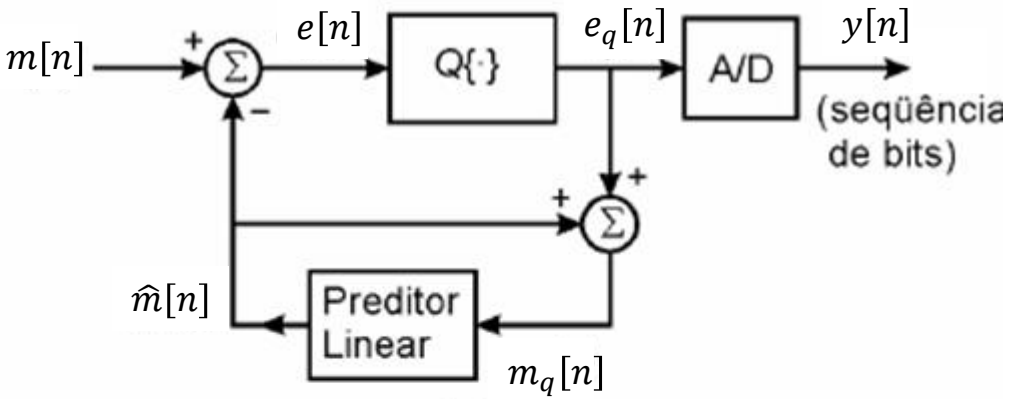
(b) Decodificador DPCM (no RX)

Note que, ao aplicar a saída do quantizador $e_q(n) = Q\{e(n)\}$ ao conversor A/D do codificador DPCM mostrado em (a), obtemos sequência $y[n]$ de bits correspondentes às mensagens codificadas em DPCM.

Após ser submetida aos processos dos demais blocos funcionais mostrados no slide 2 (codificador de canal, modulador, amplificador de potência, canal de transmissão, amplificador de sinal, demodulador e decodificador de canal), a sequência de bits $y[n]$ na saída do codificador DPCM mostrado em (a) é recebida na entrada $\tilde{y}[n]$ do decodificador DPCM mostrado em (b). Quando o canal de transmissão não gera degradação na onda EM (EM – eletromagnética) que nele se propaga e que transporta a informação a ser transmitida da origem (TX) ao destino (RX), a sequência de bits $\tilde{y}[n]$ recebida na entrada do decodificador DPCM no RX corresponde à sequência de bits $y[n]$ na saída do codificador DPCM no TX, i.e., $\tilde{y}[n] = y[n]$.

Pulse Code Modulation com compressão diferencial de dados (DPCM – *Differential* PCM)

A seguir, o conversor D/A do decodificador DPCM mostrado em (b) converte a sequência de bits $\tilde{y}[n]$ recebida na entrada do decodificador DPCM em uma aproximação $\widetilde{e_q}[n]$ do erro de predição quantizado $e_q[n]$. É uma aproximação porque o canal de transmissão pode degradar a informação transportada pela onda EM que nele se propaga ao ponto de o demodulador e o decodificador de canal não conseguirem recuperar o sinal original.

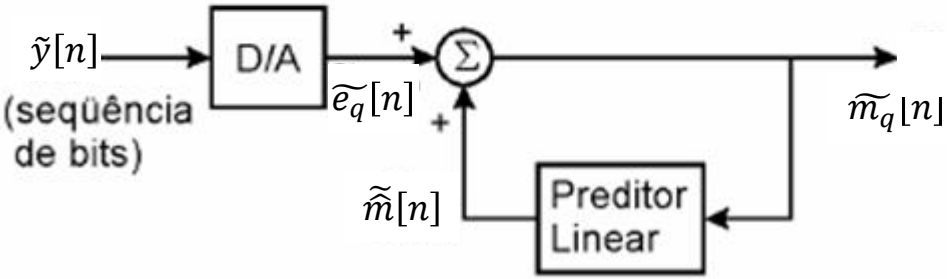


(a) Codificador DPCM (no TX)

O processo de predição efetuado no decodificador DPCM é idêntico ao processo de predição realizado no codificador. A saída $\widetilde{m}_q[n]$ do decodificador em (b) é uma aproximação quantizada de $m[n]$ na entrada do codificador em (a) e é dada pela soma do sinal predito $\widetilde{\hat{m}}[n]$ com o erro de predição quantizado $\widetilde{e_q}[n]$:

$$\widetilde{m}_q[n] = \widetilde{e_q}[n] + \widetilde{\hat{m}}[n] \qquad (9A)$$

Se o canal de transmissão não introduz nenhuma degradação, então $\widetilde{m}_q[n] = m_q[n]$.



(b) Decodificador DPCM (no RX)

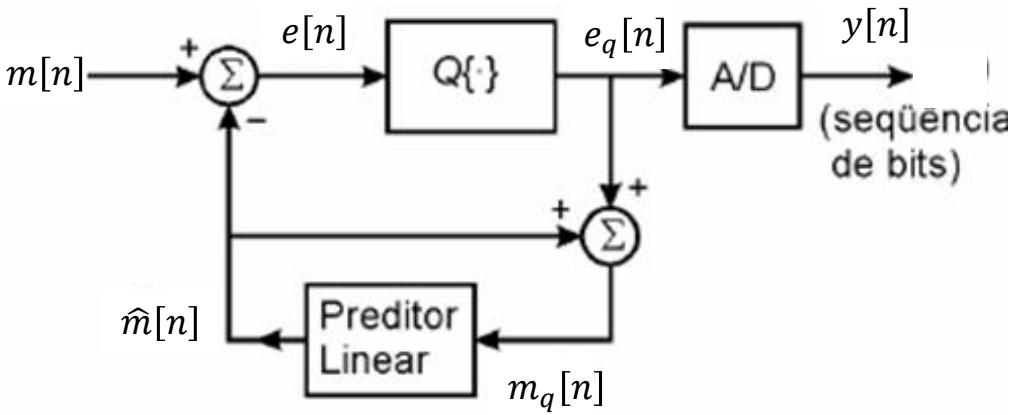
No âmbito da implementação em *hardware* de um codificador DPCM, uma possível abordagem é o sinal original $m[n]$ resultar da digitalização de um sinal analógico $m(t)$ através de um A/D de 16 bits, prévio ao codificador DPCM mostrado em (a), de modo que o ruído de quantização em $m[n]$ resulte mínimo (i.e., resulte em uma alta relação sinal ruído p/ $m[n]$: $SNR_Q = 96$ dB – ver equação (5) slide 11). Nesta situação, o sinal $m[n]$ pode ser representado por uma variável em ponto flutuante e o quantizador $Q\{\cdot\}$ e o preditor linear podem ser implementados em um microprocessador (ARM, por exemplo), com todas as demais variáveis de sinal $e[n]$, $m_q[n]$, $\hat{m}[n]$ e $e_q[n]$ também sendo representadas em ponto flutuante. Dado que todos os sinais já são digitais, o bloco "A/D" do codificador passa a ser um bloco que converte a variável $e_q[n]$ em ponto flutuante para uma variável inteira com um reduzido número de bits, número apenas o necessário para acomodar a faixa de excursão de amplitudes de $e_q[n]$. Mesma abordagem é aplicável ao decodificador DPCM mostrado em (b).

A redução de bits obtida com DPCM em comparação ao PCM

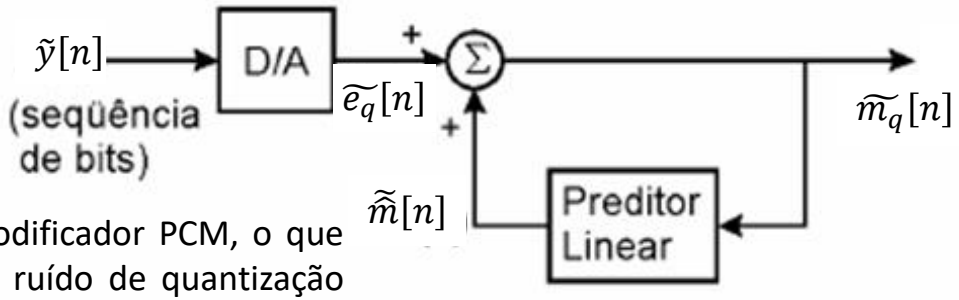
Consideremos um codificador PCM (ver slide 13) que adota um quantizador $Q\{\cdot\}$ com $M = 2^N$ níveis de quantização e passo de quantização $S = (V_H - V_L)/M$, sendo $V_H - V_L$ a faixa dinâmica do sinal na entrada de $Q\{\cdot\}$ e N o número de bits da palavra binária que representa cada amostra quantizada (ver discussão nos slides 10 e 11).

Consideremos agora o quantizador $Q\{\cdot\}$ de um codificador DPCM, conforme (a) ao lado. Se o Preditor Linear prediz $m[n]$ com alguma exatidão, então a **faixa de excursão de amplitude $V_H - V_L$ do sinal $e[n]$ na entrada de $Q\{\cdot\}$ será muito menor do que a faixa de excursão de amplitude do sinal $m[n]$ na entrada do quantizador $Q\{\cdot\}$ do codificador PCM.**

Portanto, para um mesmo número M de níveis de quantização, o passo de quantização $S = (V_H - V_L)/M$ será menor para o codificador DPCM do que para o codificador PCM, o que implica da equação (4) do slide 11 que a potência do ruído de quantização $\overline{e_q{}^2} = \frac{S^2}{12}$ é menor no codificador DPCM do que no codificador PCM.



(a) Codificador DPCM (no TX)

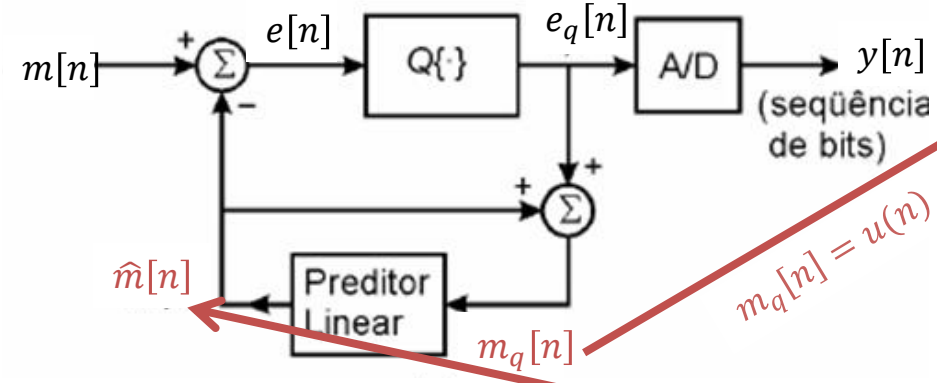


(b) Decodificador DPCM (no RX)

A consequência da asserção no parágrafo anterior é que, se um codificador DPCM apresenta a mesma relação sinal-ruído de quantização SNR_Q de um codificador PCM, então certamente o número de bits N_{DPCM} da palavra binária que representa cada amostra quantizada no codificador DPCM será menor do que o número de bits N_{PCM} da palavra binária que representa cada amostra quantizada no codificador PCM. O quanto N_{DPCM} será menor do que N_{PCM} depende das características estatísticas do sinal $m[n]$, e, principalmente, da ordem de predição do preditor linear. O quão precisa é a predição determina, portanto, o grau de compressão de bits obtido com um codificador DPCM. Para um sinal de voz digitalizado c/ $f_s = 8KHz$, um preditor linear de ordem 5 em um codificador DPCM necessita de 2 bits a menos por amostra do que um codificador PCM. Deste modo, um sistema DPCM economiza $8KHz \cdot 2\mathrm{bits} = 16\mathrm{Kbps}$ na taxa de transmissão, resultando em menor banda passante que o sistema PCM. Nos próximos slides passaremos a analisar o processo de predição linear.

Predição Linear

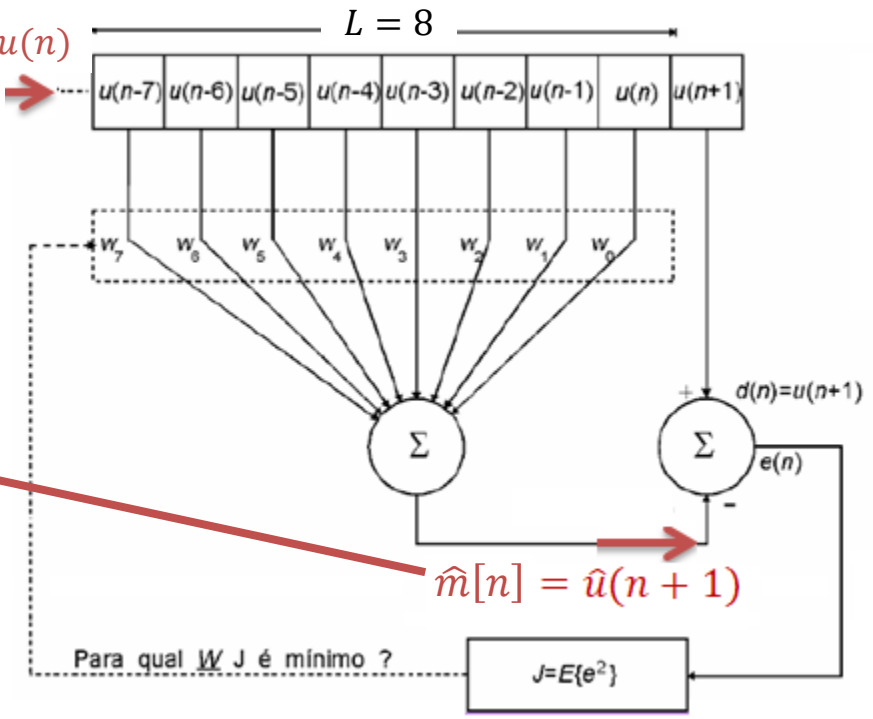
Em (a) abaixo é mostrado um codificador DPCM e em (b) é mostrada a arquitetura do bloco "Preditor Linear" adotado no codificador. No exemplo em questão o preditor linear é um filtro FIR (https://pt.wikipedia.org/wiki/Filtro_FIR) de ordem $L = 8$, i.e., o seu *buffer* que armazena as amostras passadas de $m_q[n]$ representado pelo vetor $\underline{u}$ e o vetor de coeficientes $\underline{W}$ do filtro FIR são ambos de tamanho $L = 8$. **Nota:** Vamos usar a representação $u(n)$ para expressar a n-ésima amostra da sequência u porque, no âmbito de predição linear, há descrição de sinais através de vetores e matrizes e a representação $u[n]$ para a sequência de amostras u poderia gerar confusão com a representação vetorial/matricial.



(a) Codificador DPCM (no TX)

A saída $\hat{u}(n+1)$ do preditor linear é o valor da amostra predita, e é dada por:

$$\hat{u}(n+1) = \underline{W}^T \underline{u} = \sum_{k=0}^{L-1} w_k u(n-k) \qquad (10)$$



$$\underline{W} = [w_0 \quad w_1 \quad w_2 \quad w_3 \quad w_4 \quad w_5 \quad w_6 \quad w_7]^T$$

$$\underline{u} = [u(n) \quad u(n-1) \quad u(n-2) \quad u(n-3) \quad u(n-4) \quad u(n-5) \quad u(n-6) \quad u(n-7)]^T$$

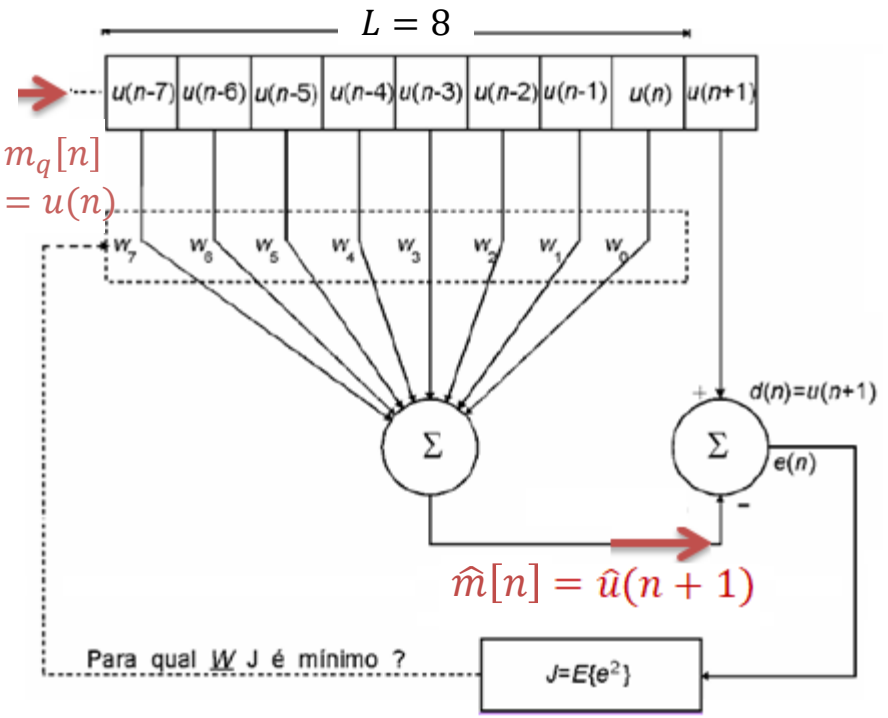
(b) Preditor linear do Codificador DPCM.

Predição Linear

Os coeficientes $\underline{W}$ do preditor são definidos de forma a minimizar de uma função de custo J (ver discussão sobre o conceito de função de custo no slide 122 de https://www.fccdecastro.com.br/pdf/CE_Aula16a18_15122020.pdf e demais orientações no link lá indicado).

No caso de predição linear, conforme mostra a figura ao lado, a função de custo J mede o erro médio quadrático $E\{e^2\}$ entre a o valor $\hat{u}(n+1)$ estimado p/ a predição da amostra que ocorre no instante $n+1$ e o valor $u(n+1) = d(n)$ da amostra que efetivamente ocorre no instante $n+1$, onde $d(n) = u(n+1)$ é a saída desejada do preditor. $e(n) = d(n) - \hat{u}(n+1) = u(n+1) - \hat{u}(n+1)$ é o erro de predição no instante discreto n e $E\{\cdot\}$ é o operador que retorna a esperança estatística de seu argumento (ver https://en.wikipedia.org/wiki/Expected_value).

De maneira semelhante ao teorema do cálculo em que os valores de x que fazem $\frac{df(x)}{dx} = 0$ correspondem à pontos de mínimo de uma função quadrática $f(x)$, no caso vetorial o operador gradiente $\underline{\nabla}$ é aplicado à função de custo J com o intuito de obter os coeficientes w_k do filtro FIR que minimizam J. Isto é obtido da solução da equação $\underline{\nabla} J = 0$, onde $\underline{\nabla} J = \frac{\partial J}{\partial \underline{W}}$.



$$\underline{W} = [w_0 \quad w_1 \quad w_2 \quad w_3 \quad w_4 \quad w_5 \quad w_6 \quad w_7]^T$$

$$\underline{u} = \begin{bmatrix} u(n) & u(n-1) & u(n-2) & u(n-3) \\ u(n-4) & u(n-5) & u(n-6) & u(n-7) \end{bmatrix}^T$$

A solução de $\frac{\partial J}{\partial \underline{W}} = 0$ é dada pela denominada Equação de Wiener-Hopf (ver https://en.wikipedia.org/wiki/Wiener_filter):

$$\underline{W} = \boldsymbol{R}^{-1}\underline{P} \qquad (11)$$

onde $\boldsymbol{R}$ é uma matriz $L \times L$ denominada de **matriz de autocorrelação** da sequência de amostras representada no vetor $\underline{u}$ e $\underline{P}$ é um vetor $L \times 1$ denominado de **vetor de correlação cruzada** entre a sequência de amostras representada no vetor $\underline{u}$ e o valor $u(n+1)$ da amostra que efetivamente ocorre no instante $n+1$. Veremos nos próximo slides como determinar $\boldsymbol{R}$ e $\underline{P}$ para um preditor linear de ordem L.

Predição Linear

Para determinar a matriz de autocorrelação $\boldsymbol{R}$ de dimensão $L \times L$ e o vetor de correlação cruzada $\underline{P}$ de dimensão $L \times 1$ utiliza-se um conjunto de $NSamp$ amostras passadas mais recentes da sequência de amostras $u(n) = m_q[n]$, sendo $NSamp$ suficientemente grande para que $\boldsymbol{R}$ "capte" a autocorrelação da sequência de amostras $u(n) = m_q[n]$. As $NSamp$ amostras da sequência $u(n)$ são transformadas em um conjunto de $NVet = NSamp - 1$ vetores $\underline{u}(n-k)$ de dimensão $L \times 1$, obtendo-se a matriz $\boldsymbol{R}$ e o vetor $\underline{P}$ através de:

$$\boldsymbol{R} = \frac{1}{NVet} \sum_{k=0}^{NVet-1} \underline{u}(n-k)\, \underline{u}^T(n-k) \qquad (12)$$

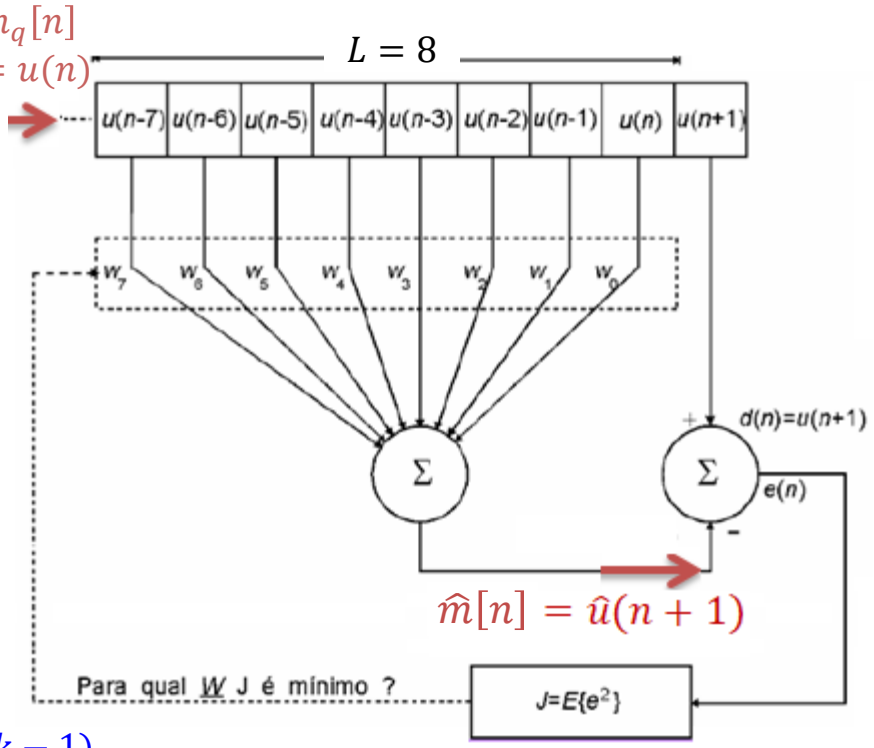
$$\underline{P} = \frac{1}{NVet-1} \sum_{k=0}^{NVet-2} u(n-k)\underline{u}(n-k-1) \qquad (13)$$

$u(n-k)$ é a saída desejada $d(n)$ do preditor p/ o vetor $\underline{u}(n-k-1)$

$\boldsymbol{R}$ e $\underline{P}$ são reavaliados a cada N_p predições, sendo N_p determinado experimentalmente com base em um compromisso entre minimização da complexidade computacional e precisão da predição.

A cada reavaliação de $\boldsymbol{R}$ e $\underline{P}$, os novos coeficientes do filtro $\underline{W}$ são obtidos e enviados ao receptor através do canal de comunicação.

Note que, a partir da inicialização de um sistema DPCM, devido à saída do preditor ser realimentada para a sua entrada, tanto o preditor do TX quanto o preditor do RX necessitam de um razoável número de amostras até que consigam reduzir o erro de predição a níveis aceitáveis.



$$\underline{W} = [w_0 \quad w_1 \quad w_2 \quad w_3 \quad w_4 \quad w_5 \quad w_6 \quad w_7]^T$$

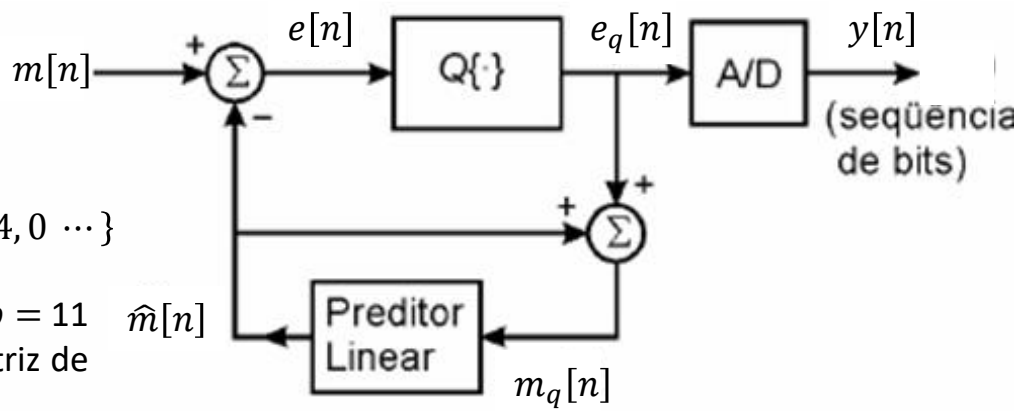
$$\underline{u} = [u(n) \quad u(n-1) \quad u(n-2) \quad u(n-3) \quad u(n-4) \quad u(n-5) \quad u(n-6) \quad u(n-7)]^T$$

$$\underline{W} = \boldsymbol{R}^{-1}\underline{P} \qquad (11)$$

Exemplo 1: Consideremos o codificador DPCM mostrado em (a) ao lado. A sequência de amostras na entrada do bloco Preditor Linear é dada por:

$$u(n) = m_q[n] =$$
$$= \{-2, -1.414, 0, 1.414, 2, 1.414, 0, -1.414, -2, -1.414, 0 \cdots\}$$

O preditor linear é de ordem $L = 2$ e utiliza $NSamp = 11$ amostras consecutivas de $m_q[n]$ para a definição da matriz de correlação $\boldsymbol{R}$.



(a) Codificador DPCM (no TX)

Dica: Dado $\boldsymbol{R}[2 \times 2]$ temos que

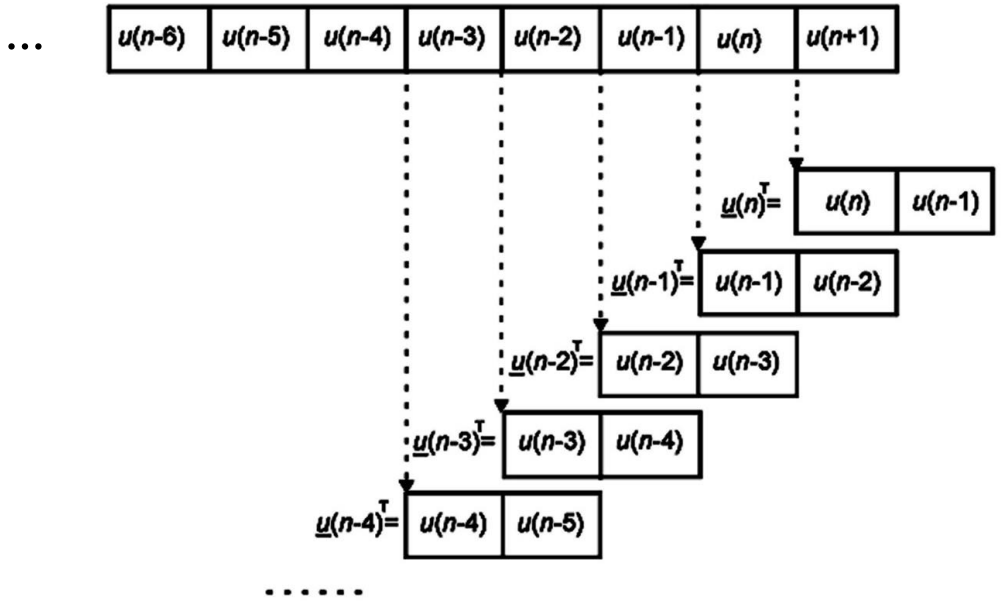
$$\boldsymbol{R} = \begin{bmatrix} r00 & r01 \\ r10 & r11 \end{bmatrix} \rightarrow \boldsymbol{R}^{-1} = \frac{1}{(r00 \cdot r11 - r01 \cdot r10)} \begin{bmatrix} r11 & -r01 \\ -r10 & r00 \end{bmatrix}$$

Pede-se:

(a) Determine o valor de $\hat{m}[n] = \hat{u}(n+1)$ que é a estimativa da predição da amostra que ocorre no instante $n+1$.

(b) Determine o erro de predição $e[n]$ para $\hat{m}[n]$ obtido em (a) sabendo que o valor da amostra que efetivamente ocorre no instante $n+1$ é $u(n+1) = 1.414$.

Solução: **(a)** A estrutura de dados para a predição é composta por $NVet = NSamp - 1 = 10$ vetores $\underline{u}(n-k)$, cada vetor $\underline{u}(n-k)$ possui $L = 2$ componentes, sendo L a ordem do preditor:



$$u(n) = m_q[n] = \{-2, -1.414, \quad 0, \quad 1.414, \quad 2, \quad 1.414, \quad 0, \quad -1.414, \quad -2, \quad -1.414, \quad 0 \cdots\}$$
$$= \{u(n-10),\ u(n-9), u(n-8),\ u(n-7),\ u(n-6),\ u(n-5), u(n-4),\ u(n-3), u(n-2),\ u(n-1), u(n)\cdots\}$$

Em conformidade com a equação (12), os $NVet = NSamp - 1 = 10$ vetores $\underline{u}(n-k)$ resultam:

$$\underline{u}(n) = \begin{bmatrix} u(n) \\ u(n-1) \end{bmatrix} = \begin{bmatrix} 0 \\ -1.414 \end{bmatrix}$$

$$\underline{u}(n-1) = \begin{bmatrix} u(n-1) \\ u(n-2) \end{bmatrix} = \begin{bmatrix} -1.414 \\ -2 \end{bmatrix}$$

$$\underline{u}(n-2) = \begin{bmatrix} u(n-2) \\ u(n-3) \end{bmatrix} = \begin{bmatrix} -2 \\ -1.414 \end{bmatrix}$$

...

$$\underline{u}(n-9) = \begin{bmatrix} u(n-9) \\ u(n-10) \end{bmatrix} = \begin{bmatrix} -1.414 \\ -2 \end{bmatrix}$$

Predição Linear

Determinando as $NVet = NSamp - 1 = 10$ matrizes $\underline{u}(n-k)\,\underline{u}^T(n-k)$ usadas na formação de matriz de autocorrelação $\boldsymbol{R}$ de dimensão $L \times L$, em conformidade com a equação (12):

Reproduzindo $u(n)$ para efeito de facilitar a visualização da sequência de amostras: $u(n) = m_q[n] = \{-2, -1.414,\ 0,\ 1.414,\ 2,\ 1.414,\ 0,\ -1.414,\ -2,\ -1.414,\ 0\ \cdots\} =$

$= \{u(n-10),\ u(n-9),\ u(n-8),\ u(n-7),\ u(n-6),\ u(n-5),\ u(n-4),\ u(n-3),\ u(n-2),\ u(n-1),\ u(n)\cdots\}$

$$\underline{u}(n)\underline{u}^T(n) = \begin{bmatrix} 0 \\ -1.414 \end{bmatrix}\begin{bmatrix} 0 & -1.414 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1.9994 \end{bmatrix}_0 \leftarrow k = 0$$

$$\underline{u}(n-1)\underline{u}^T(n-1) = \begin{bmatrix} -1.414 \\ -2 \end{bmatrix}\begin{bmatrix} -1.414 & -2 \end{bmatrix} = \begin{bmatrix} 1.9994 & 2.8280 \\ 2.8280 & 4 \end{bmatrix}_1 \leftarrow k = 1$$

$$\underline{u}(n-2)\underline{u}^T(n-2) = \begin{bmatrix} -2 \\ -1.414 \end{bmatrix}\begin{bmatrix} -2 & -1.414 \end{bmatrix} = \begin{bmatrix} 4 & 2.8280 \\ 2.8280 & 1.9994 \end{bmatrix}_2 \leftarrow k = 2$$

$$\underline{u}(n-3)\underline{u}^T(n-3) = \begin{bmatrix} -1.414 \\ 0 \end{bmatrix}\begin{bmatrix} -1.414 & 0 \end{bmatrix} = \begin{bmatrix} 1.9994 & 0 \\ 0 & 0 \end{bmatrix}_3 \leftarrow k = 3$$

$$\underline{u}(n-4)\underline{u}^T(n-4) = \begin{bmatrix} 0 \\ 1.414 \end{bmatrix}\begin{bmatrix} 0 & 1.414 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1.9994 \end{bmatrix}_4 \leftarrow k = 4$$

$$\underline{u}(n-5)\underline{u}^T(n-5)=\begin{bmatrix}1.414\\2\end{bmatrix}\begin{bmatrix}1.414 & 2\end{bmatrix}=\begin{bmatrix}1.9994 & 2.8280\\2.8280 & 4\end{bmatrix}_5 \longleftarrow k=5$$

$$\underline{u}(n-6)\underline{u}^T(n-6)=\begin{bmatrix}2\\1.414\end{bmatrix}\begin{bmatrix}2 & 1.414\end{bmatrix}=\begin{bmatrix}4 & 2.8280\\2.8280 & 1.9994\end{bmatrix}_6 \longleftarrow k=6$$

$$\underline{u}(n-7)\underline{u}^T(n-7)=\begin{bmatrix}1.414\\0\end{bmatrix}\begin{bmatrix}1.414 & 0\end{bmatrix}=\begin{bmatrix}1.9994 & 0\\0 & 0\end{bmatrix}_7 \longleftarrow k=7$$

$$\underline{u}(n-8)\underline{u}^T(n-8)=\begin{bmatrix}0\\-1.414\end{bmatrix}\begin{bmatrix}0 & -1.414\end{bmatrix}=\begin{bmatrix}0 & 0\\0 & 1.9994\end{bmatrix}_8 \longleftarrow k=8$$

$$\underline{u}(n-9)\underline{u}^T(n-9)=\begin{bmatrix}-1.414\\-2\end{bmatrix}\begin{bmatrix}-1.414 & -2\end{bmatrix}=\begin{bmatrix}1.9994 & 2.8280\\2.8280 & 4\end{bmatrix}_9 \longleftarrow k=9$$

Substituindo na equação (12) as $NVet = NSamp - 1 = 10$ matrizes $\underline{u}(n-k)\,\underline{u}^T(n-k)$ acima determinadas obtemos a matriz de autocorrelação $\boldsymbol{R}$:

$$\boldsymbol{R}=\frac{1}{10}\sum_{k=0}^{9}\underline{u}(n-k)\underline{u}^T(n-k)=\frac{1}{10}\begin{bmatrix}17.9970 & 14.1400\\14.1400 & 21.9970\end{bmatrix}=\begin{bmatrix}1.7997 & 1.4140\\1.4140 & 2.1997\end{bmatrix}$$

Predição Linear

Determinando os $NVet-1=9$ vetores $u(n-k)\underline{u}(n-k-1)$ usados na formação do vetor de correlação cruzada $\underline{P}$ de dimensão $L\times 1$, em conformidade com a equação (13) (reproduzida abaixo por facilidade de visualização):

$$\underline{P}=\frac{1}{NVet-1}\sum_{k=0}^{NVet-2}u(n-k)\underline{u}(n-k-1)=\frac{1}{9}\sum_{k=0}^{8}u(n-k)\underline{u}(n-k-1)$$

$u(n-k)$ é a saída desejada $d(n)$ do preditor p/ o vetor $\underline{u}(n-k-1)$

Reproduzindo $u(n)$ para efeito de facilitar a visualização da sequência de amostras: $u(n)=m_q[n]=$
$\{-2,-1.414,\ 0,\ 1.414,\ 2,1.414,\ 0,\ -1.414,\ -2,\ -1.414,\ 0\cdots\}=$
$=\{u(n-10),\ u(n-9),u(n-8),\ u(n-7),\ u(n-6),\ u(n-5),u(n-4),u(n-3),u(n-2),u(n-1),u(n)\cdots\}$

$$k=0\ \rightarrow d(n)\underline{u}(n-1)=u(n)\underline{u}(n-1)=0\begin{bmatrix}-1.414\\2\end{bmatrix}=\begin{bmatrix}0\\0\end{bmatrix}$$

$$k=1\ \rightarrow d(n)\underline{u}(n-2)=u(n-1)\underline{u}(n-2)=-1.414\begin{bmatrix}-2\\-1.414\end{bmatrix}=\begin{bmatrix}2.8280\\1.9994\end{bmatrix}$$

$$k=2\ \rightarrow d(n)\underline{u}(n-3)=u(n-2)\underline{u}(n-3)=-2\begin{bmatrix}-1.414\\0\end{bmatrix}=\begin{bmatrix}2.8280\\0\end{bmatrix}$$

$$k=3\ \rightarrow d(n)\underline{u}(n-4)=u(n-3)\underline{u}(n-4)=-1.414\begin{bmatrix}0\\-1.414\end{bmatrix}=\begin{bmatrix}0\\-1.9994\end{bmatrix}$$

$$k=4\ \rightarrow d(n)\underline{u}(n-5)=u(n-4)\underline{u}(n-5)=0\begin{bmatrix}1.414\\2\end{bmatrix}=\begin{bmatrix}0\\0\end{bmatrix}$$

$$k=5 \rightarrow d(n)\underline{u}(n-6)=u(n-5)\underline{u}(n-6)=1.414\begin{bmatrix}2\\1.414\end{bmatrix}=\begin{bmatrix}2.8280\\1.9994\end{bmatrix}$$

$$k=6 \rightarrow d(n)\underline{u}(n-7)=u(n-6)\underline{u}(n-7)=2\begin{bmatrix}1.414\\0\end{bmatrix}=\begin{bmatrix}2.8280\\0\end{bmatrix}$$

$$k=7 \rightarrow d(n)\underline{u}(n-8)=u(n-7)\underline{u}(n-8)=1.414\begin{bmatrix}0\\-1.414\end{bmatrix}=\begin{bmatrix}0\\-1.9994\end{bmatrix}$$

$$k=8 \rightarrow d(n)\underline{u}(n-9)=u(n-8)\underline{u}(n-9)=0\begin{bmatrix}-1.414\\-2\end{bmatrix}=\begin{bmatrix}0\\0\end{bmatrix}$$

Substituindo na equação (13) os $NVet-1=9$ vetores $u(n-k)\underline{u}(n-k-1)$ acima determinados obtemos do vetor de correlação cruzada $\underline{P}$:

$$\underline{P}=\frac{1}{NVet-1}\sum_{k=0}^{NVet-2}u(n-k)\underline{u}(n-k-1)=\frac{1}{9}\sum_{k=0}^{8}u(n-k)\underline{u}(n-k-1)=\frac{1}{9}\begin{bmatrix}11.3120\\0\end{bmatrix}=\begin{bmatrix}1.2569\\0\end{bmatrix}$$

Substituindo na equação (11) do slide 21 o valor obtido para $\boldsymbol{R}$ (slide 26) e substituindo na equação (11) o valor obtido para $\underline{P}$ (acima) obtemos o vetor $\underline{W}$ cujos componentes são os coeficientes do filtro FIR do preditor:

$$\underline{W}=\boldsymbol{R}^{-1}\underline{P}=\begin{bmatrix}1.7997 & 1.4140\\1.4140 & 2.1997\end{bmatrix}^{-1}\begin{bmatrix}1.2569\\0\end{bmatrix}=\begin{bmatrix}1.1226 & -0.7216\\-0.7216 & 0.9185\end{bmatrix}\begin{bmatrix}1.2569\\0\end{bmatrix}=\begin{bmatrix}1.4110\\-0.9070\end{bmatrix}$$

Predição Linear

O valor estimado $\hat{u}(n+1)$ para a 1ª amostra subsequente à sequencia de amostras $u(n) = m_q[n] = \{-2, -1.414,\ 0,\ 1.414,\ 2, 1.414,\ 0,\ -1.414,\ -2,\ -1.414,\ 0\ \cdots\} =$

$$= \{u(n-10),\ u(n-9), u(n-8),\ u(n-7),\ u(n-6),\ u(n-5), u(n-4), u(n-3), u(n-2), u(n-1), u(n)\cdots\}$$

é dado pela equação (10) do slide 20:

$$\hat{u}(n+1) = \underline{W}^T \underline{u} = \sum_{k=0}^{L-1} w_k u(n-k) = \sum_{k=0}^{1} w_k u(n-k) = w_o u(n) + w_1 u(n-1) =$$
$$= (1.4110)(0) + (-0.9070)(-1.414) = 1.2825$$

(b) A sequencia de amostras $u(n)$ incluindo o valor da amostra $u(n+1) = 1.414$ que efetivamente ocorre no instante $n+1$ é $u(n) = m_q[n] = \{-2, -1.414,\ 0,\ 1.414,\ 2, 1.414,\ 0,\ -1.414,\ -2,\ -1.414,\ 0, 1.414, \cdots\}$.

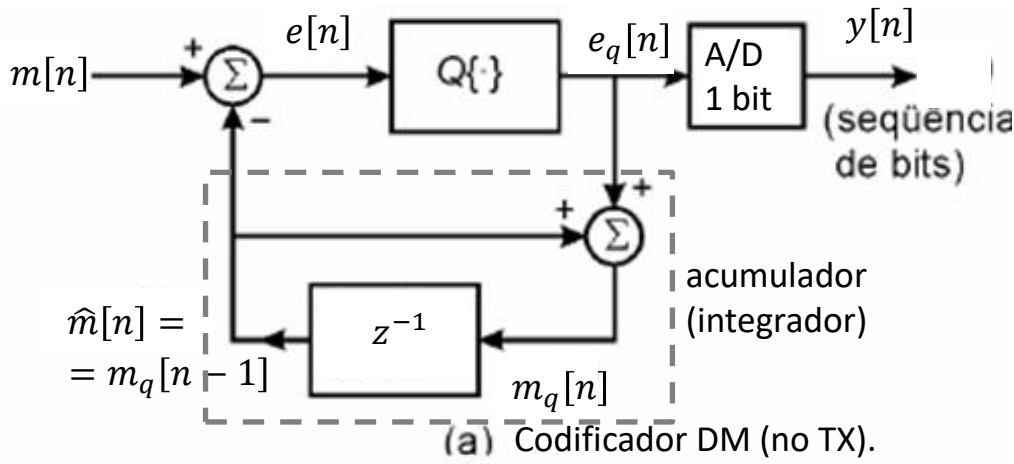
Portanto, o erro de predição no instante discreto $n+1$ é

$$e(n) = d(n) - \hat{u}(n+1) = u(n+1) - \hat{u}(n+1) = 1.414 - 1.2825 = 0.1315$$

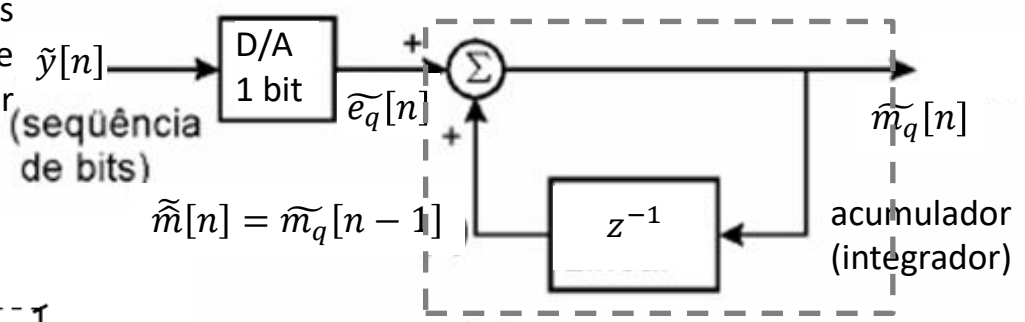
Alternativamente, no link Videoaula DPCM - Profa. Cristina De Castro encontra-se disponível uma videoaula com a revisão passo a passo da solução do Exemplo 1.

Modulação Delta (DM – *Delta Modulation*)

Quando o sinal analógico $m(t)$ é amostrado a uma frequência de amostragem f_s muitíssimo maior do que o $Nyquist\ Rate = 2f_M$ (ver slide 9), caracterizando uma super-amostragem, o sinal super-amostrado $m[n]$ resultante exibe uma altíssima correlação entre amostras adjacentes. Esta altíssima correlação entre as amostras reduz drasticamente o erro de predição do bloco Preditor Linear ao ponto de a faixa de excursão de amplitude $V_H - V_L$ do sinal $e[n]$ na entrada do quantizador $Q\{\cdot\}$ tender a zero, permitindo reduzir o número de níveis de quantização $M = 2^N$ para $M = 2$ e, portanto, as palavras binárias na saída $y[n]$ do codificador DM são palavras binárias de tamanho $N = 1$ bit. Uma aplicação clássica de DM é o link de voz do *space shuttle* da NASA (ver https://www.fccdecastro.com.br/pdf/SSGTDM.pdf).

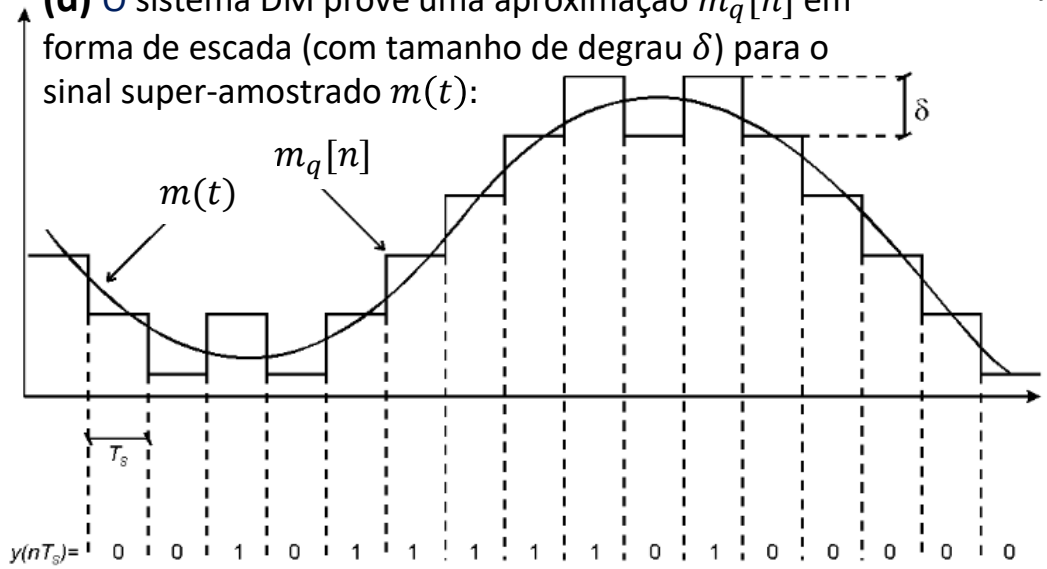


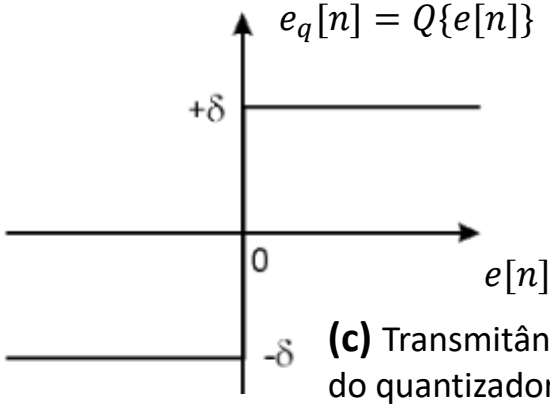
(a) Codificador DM (no TX).



(b) Decodificador DM (no RX).

(d) O sistema DM provê uma aproximação $m_q[n]$ em forma de escada (com tamanho de degrau δ) para o sinal super-amostrado $m(t)$:





(c) Transmitância do quantizador.

Modulação Delta (DM – *Delta Modulation*)

O erro de predição $e[n]$ na entrada do quantizador $Q\{\cdot\}$ do codificador DM é:

$$e[n] = m[n] - m_q[n-1] \qquad (14)$$

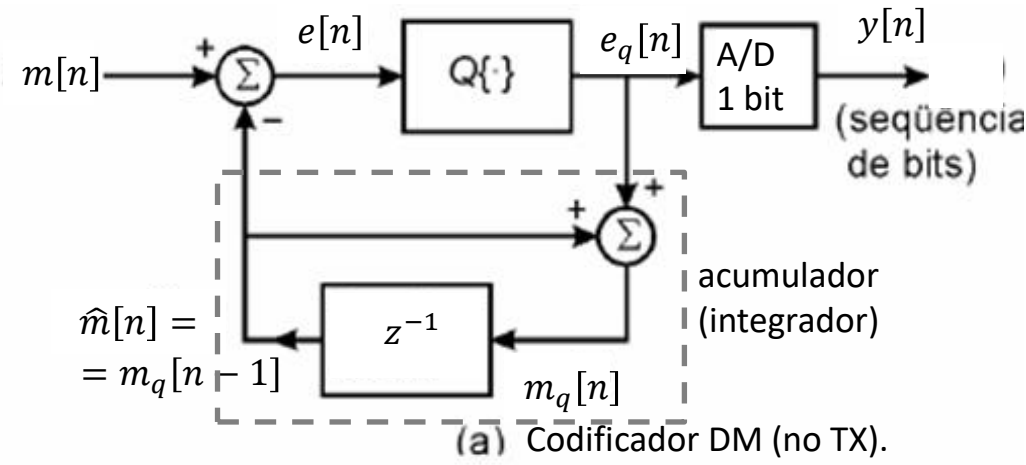
A saída $e_q[n]$ do quantizador é dada por:

$$e_q[n] = \delta \,\mathrm{sgn}(e[n]) \qquad (15)$$

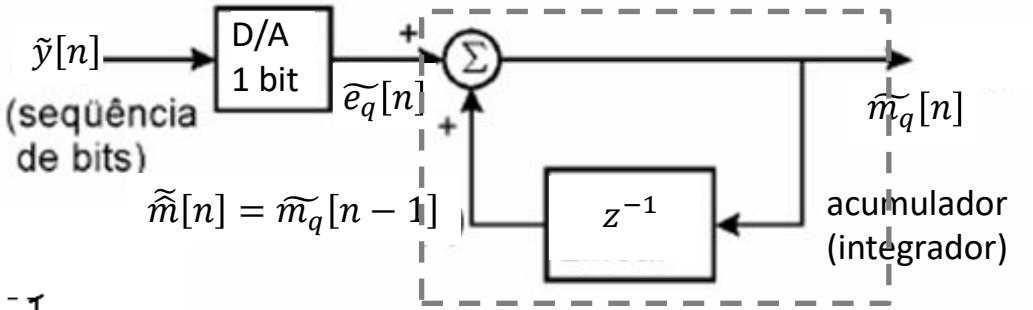
onde

$$\mathrm{sgn}(x) = \begin{cases} +1, x \geq 0 \\ -1, x < 0 \end{cases}$$

sendo δ o tamanho do degrau da "escada" do sinal $m_q[n]$ que aproxima o sinal super-amostrado $m(t)$ (ver (d) abaixo).

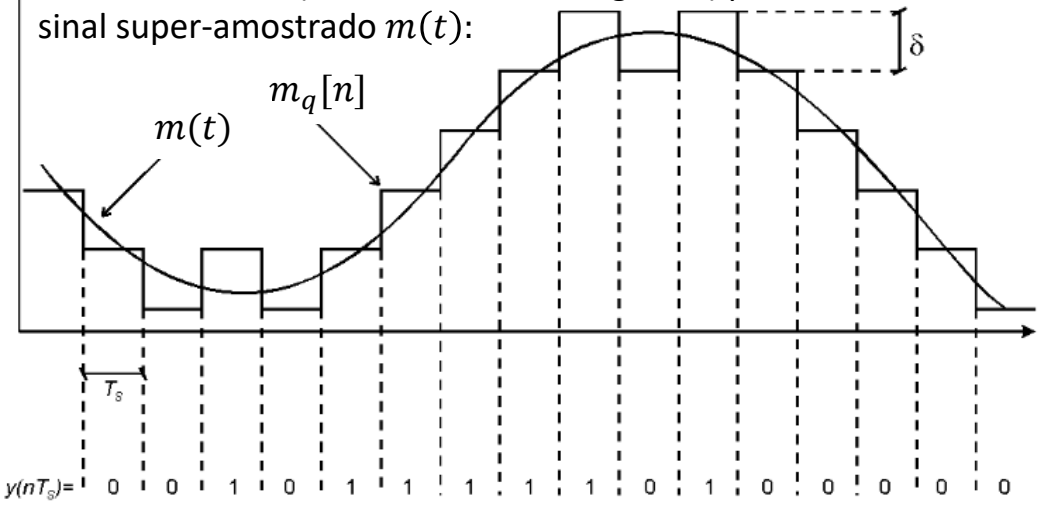


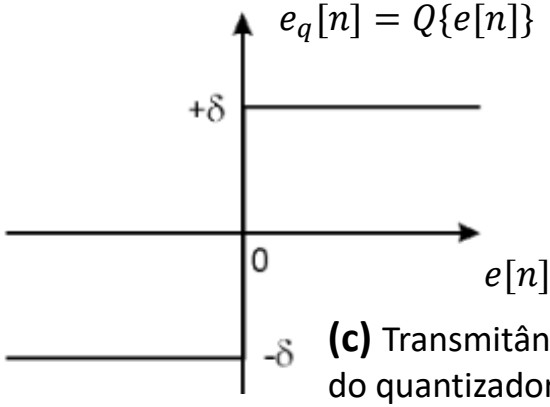
(a) Codificador DM (no TX).



(b) Decodificador DM (no RX).

(d) O sistema DM provê uma aproximação $m_q[n]$ em forma de escada (com tamanho de degrau δ) para o sinal super-amostrado $m(t)$:

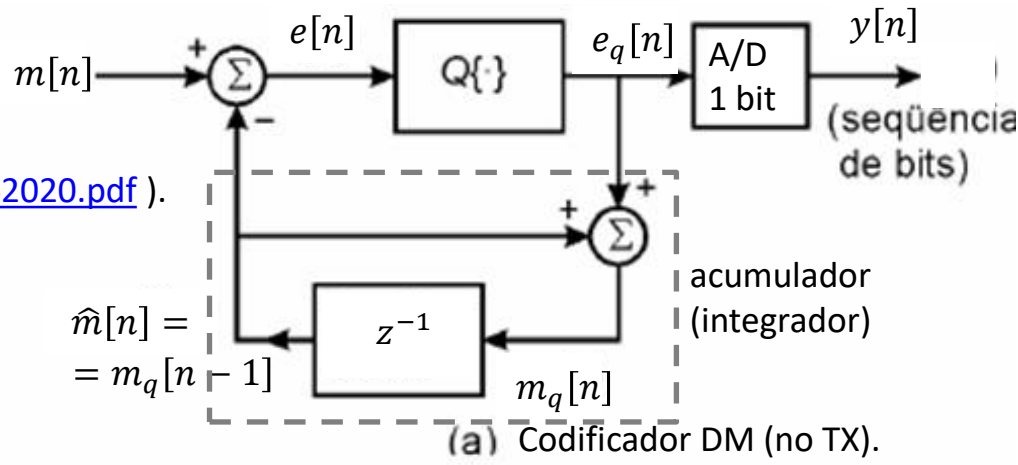




(c) Transmitância do quantizador.

Modulação Delta (DM – *Delta Modulation*)

O bloco preditor z^{-1} é um preditor pela última amostra que, no instante n copia para a sua saída $\hat{m}[n]$ a amostra anterior $m_q[n-1]$ presente em sua entrada no instante $n-1$ (significado de z^{-1}: ver transformada Z em https://www.fccdecastro.com.br/pdf/SS_aula23a26_25062020.pdf).



(a) Codificador DM (no TX).

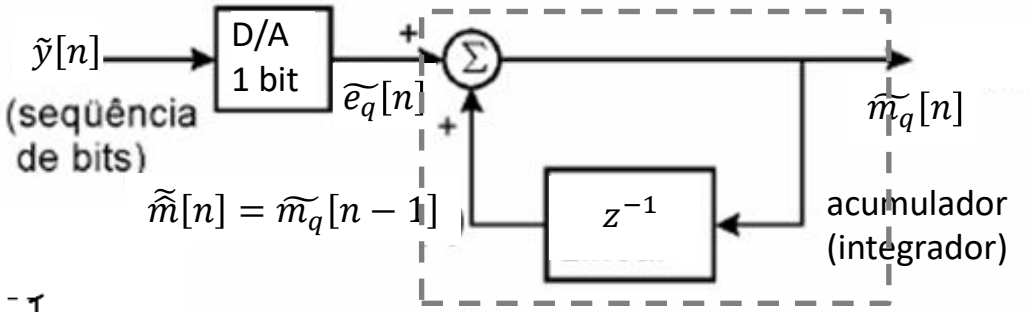
A entrada do preditor z^{-1} é dada por:

$$m_q[n] = m_q[n-1] + e_q[n] \qquad (16)$$

A realimentação da saída $\hat{m}[n]$ do preditor z^{-1} à sua entrada através do somador Σ forma um acumulador:

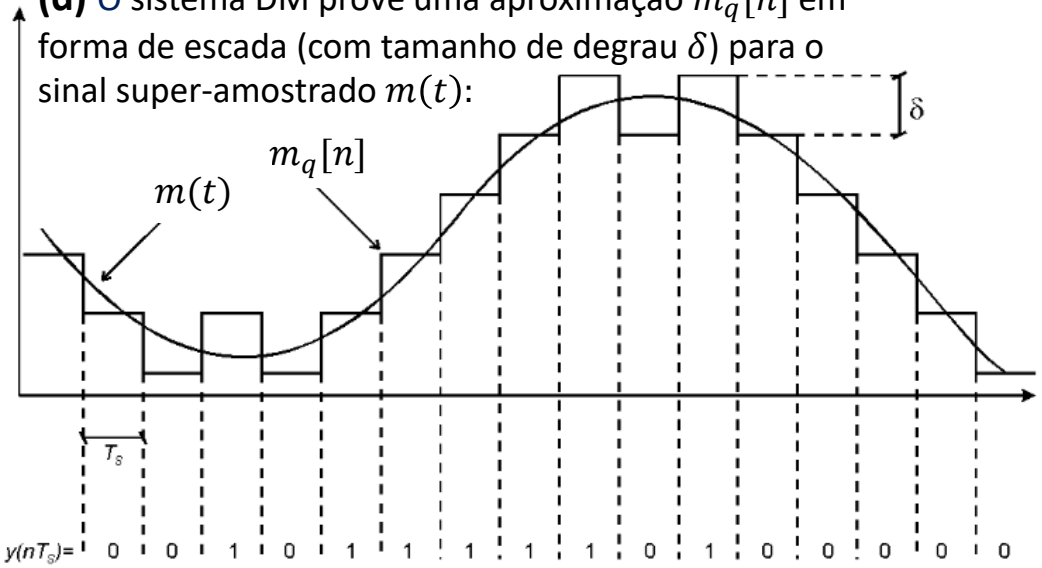
$$m_q[n] = \sum_{i=0}^{n} \left(e_q[n] + m_q[n-1]\right) =$$

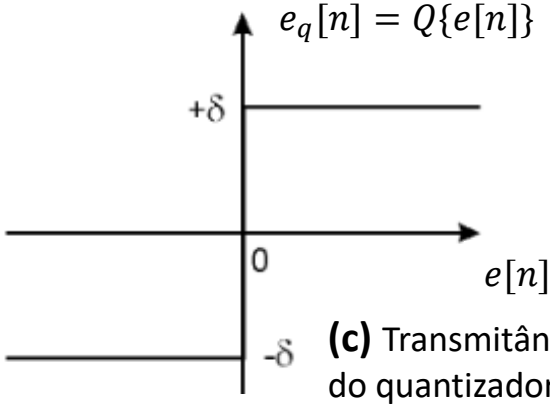
$$= \sum_{i=0}^{n} e_q[n] + \sum_{i=0}^{n} m_q[n-1] = \sum_{i=0}^{n} m_q[n-1] \qquad (17)$$



(b) Decodificador DM (no RX).

(d) O sistema DM provê uma aproximação $m_q[n]$ em forma de escada (com tamanho de degrau δ) para o sinal super-amostrado $m(t)$:





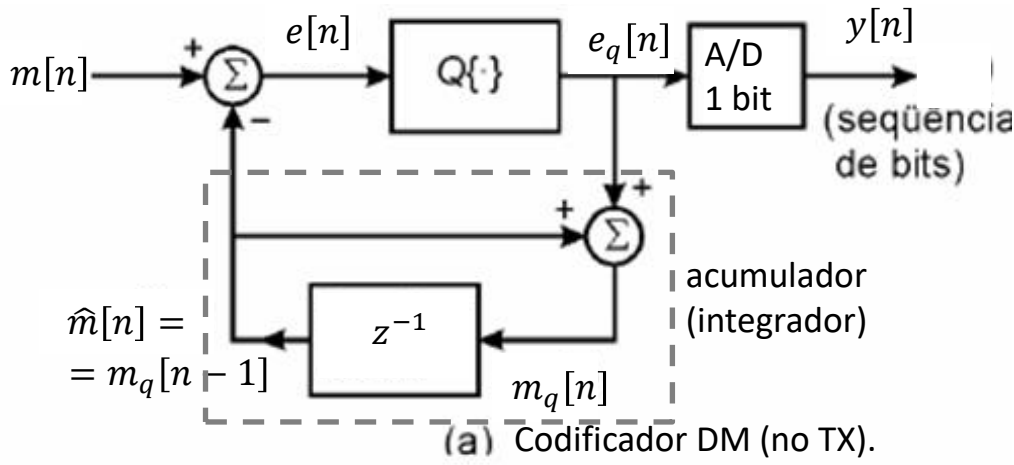
(c) Transmitância do quantizador.

Modulação Delta (DM – *Delta Modulation*)

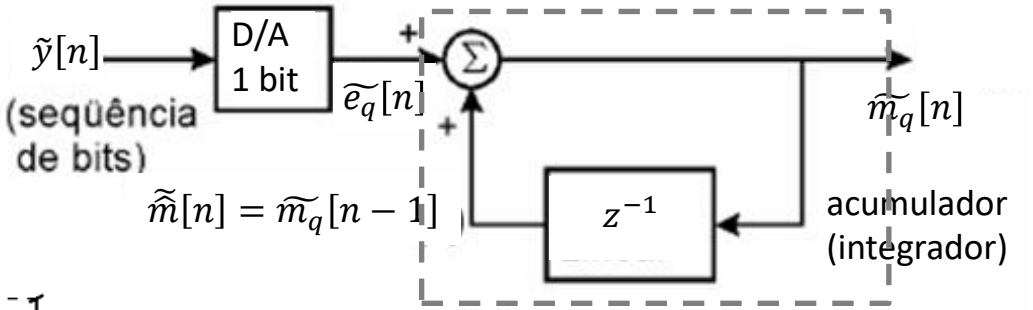
No decodificador DM em (b), cada bit é transformado pelo D/A em um pulso $+\delta/-\delta$, que representa o sinal $\widetilde{e_q}[n]$.

O acumulador gera então o sinal $\widetilde{m_q}[n]$ em sua saída, que é uma aproximação quantizada de $m[n]$ na entrada do codificador DM em (a):

$$\widetilde{m_q}[n] = \widetilde{e_q}[n] + \widetilde{\hat{m}}[n] \qquad (18)$$

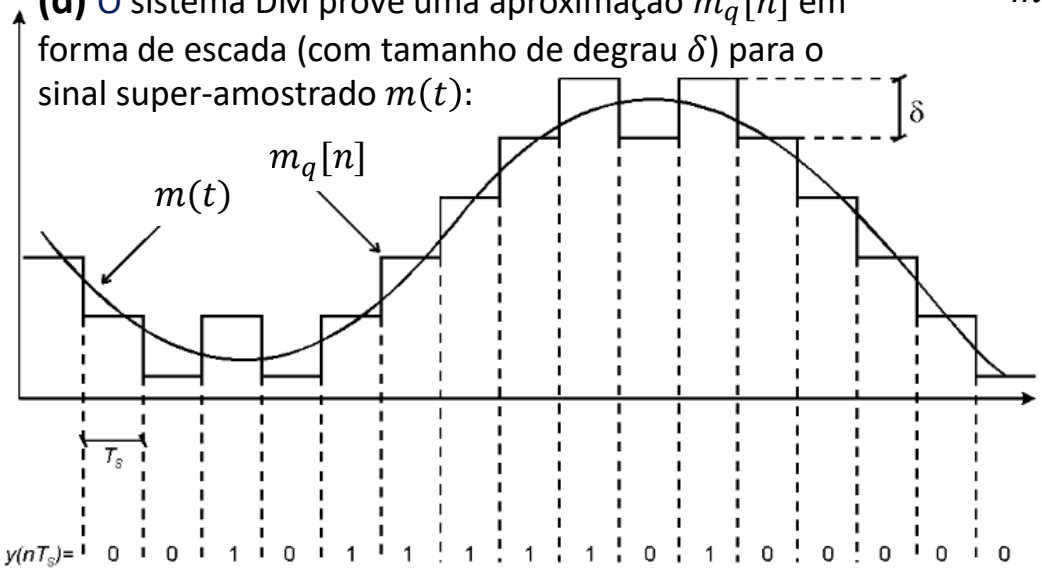


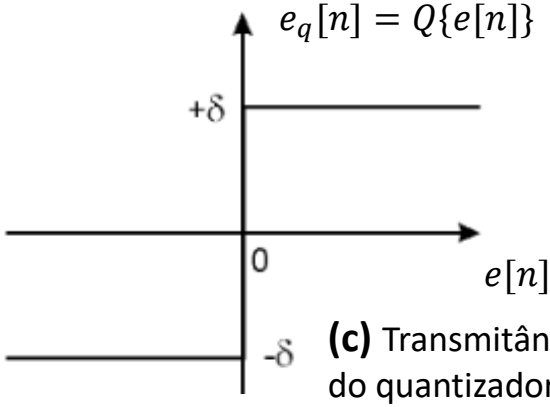
(a) Codificador DM (no TX).



(b) Decodificador DM (no RX).

(d) O sistema DM provê uma aproximação $m_q[n]$ em forma de escada (com tamanho de degrau δ) para o sinal super-amostrado $m(t)$:





(c) Transmitância do quantizador.

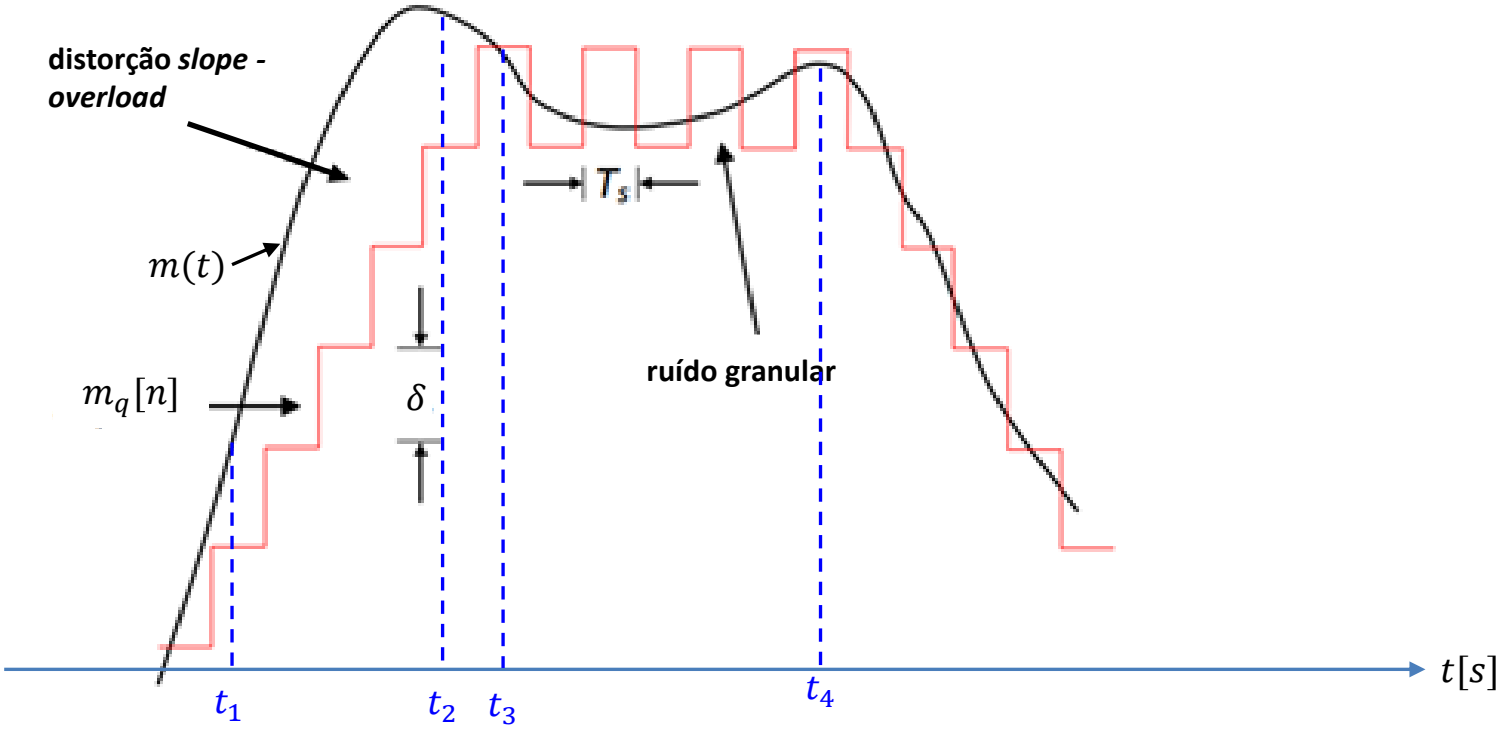
Distorção de sinal na Modulação Delta (DM – *Delta Modulation*)

Sistemas DM são sujeitos a dois tipos de erro de quantização (ver figura abaixo):

(I) Distorção por declividade excessiva em $m(t)$ (*slope-overload distortion*): No intervalo de tempo t_1 a t_2 o valor de δ é muito pequeno para que $m_q[n]$ consiga acompanhar a alta razão de variação de $m(t)$.

(II) Ruído granular: No intervalo de tempo t_3 a t_4 a razão de variação de $m(t)$ é pequena e o valor de δ é desnecessariamente grande, gerando variações $\pm\delta$ em torno do valor de $m(t)$ que representa um ruído de amplitude $\pm\delta$ sobreposto ao sinal $m(t)$.

Percebe-se, portanto, que existe a necessidade de um valor maior para δ objetivando acomodar um sinal $m(t)$ com alta razão de variação, enquanto que um menor valor de δ é necessário para representar com precisão um sinal $m(t)$ aproximadamente invariável no tempo. Neste contexto, o valor de δ adequado resulta em um compromisso entre distorção *slope-overload* e ruído granular.



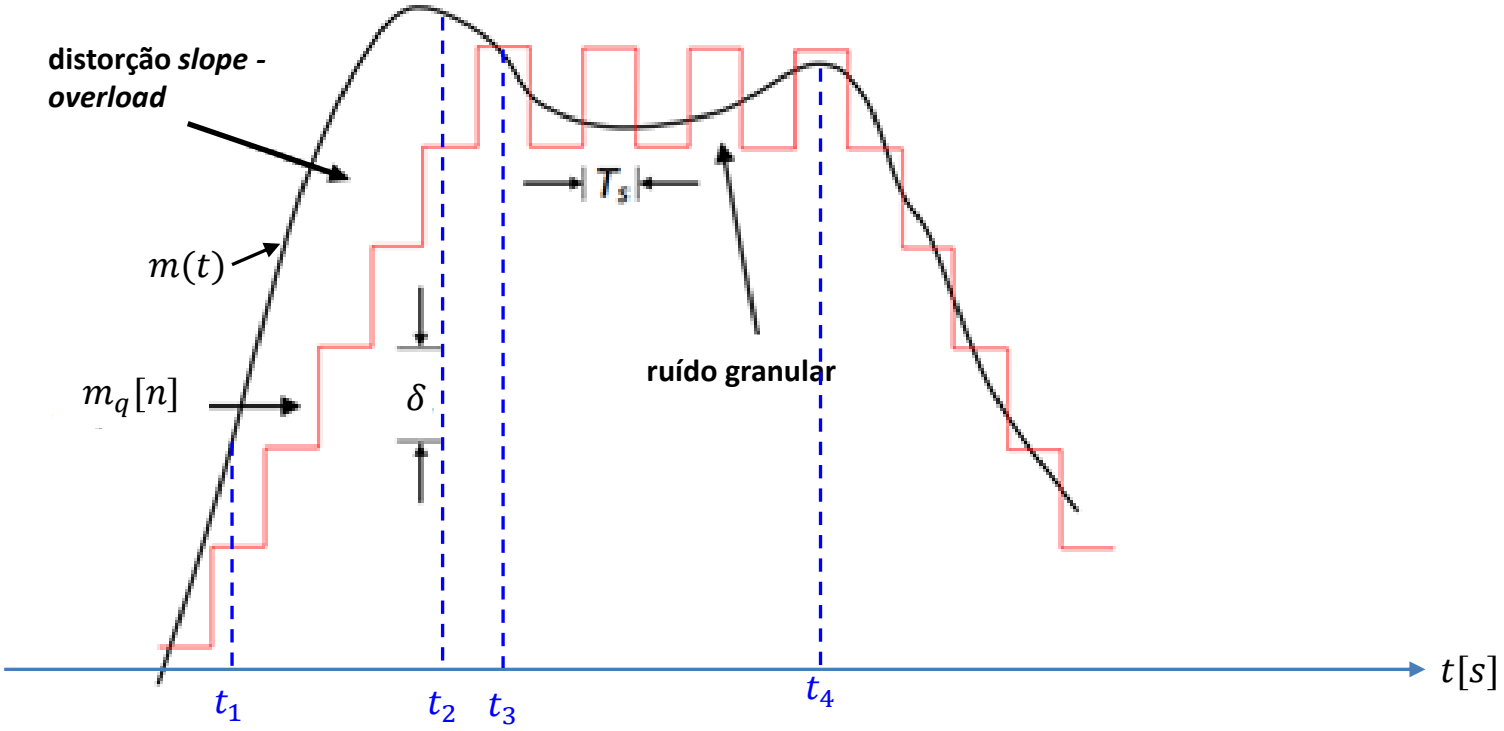
Modulação Delta Adaptativa (ADM – *Adaptive Delta Modulation*)

O sistema ADM objetiva otimizar δ para que o mesmo se adapte à dinâmica do sinal $m(t)$ minimizando o compromisso entre distorção *slope-overload* e ruído granular referido no slide anterior.

Um sistema ADM tipicamente utiliza a seguinte regra de adaptação para δ:

$$\delta_n = \delta_{n-1}\gamma^{\rho(b_n)\rho(b_{n-1})} \tag{19}$$

onde $\gamma > 1$ é uma constante determinada experimentalmente objetivando a redução do erro de quantização. $\rho(b) = 2b - 1$, com b_n e b_{n-1} sendo respectivamente o último e o penúltimo bit gerados na saída $y[\mathrm{n}]$ do A/D do codificador de um sistema DM.

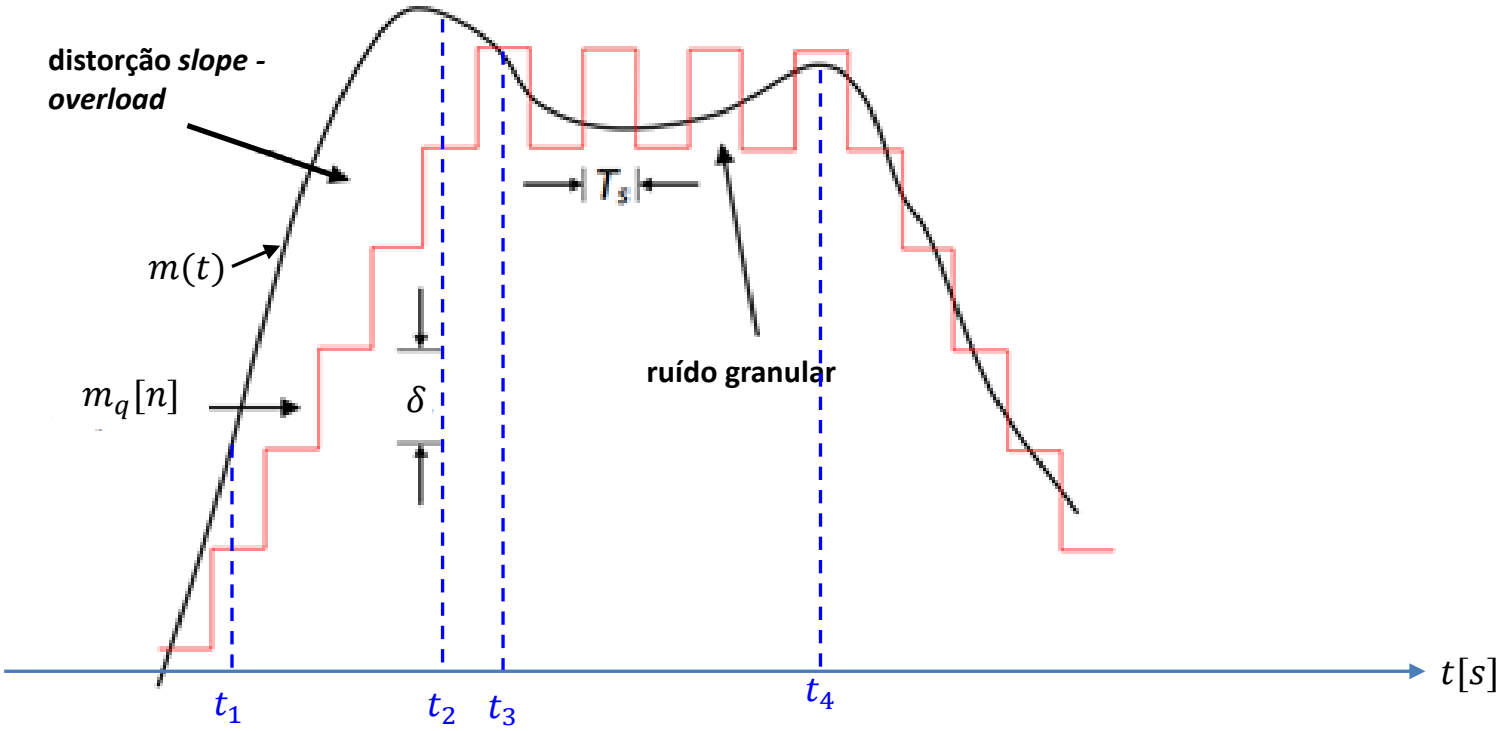


Modulação Delta Adaptativa (ADM – *Adaptive Delta Modulation*)

$$\delta_n = \delta_{n-1}\gamma^{\rho(b_n)\rho(b_{n-1})} \tag{19}$$

com $\rho(b) = 2b - 1$. Assim, se em um determinado instante n começa a ocorrer distorção *slope-overload*, então forçosamente $b_n = b_{n-1} \rightarrow \rho(b_n)\rho(b_{n-1}) = 1 \rightarrow$ de (19) $\delta_n = \delta_{n-1}\gamma$. Portanto δ_n é multiplicado por γ, aumentando δ e reduzindo a distorção *slope-overload*.

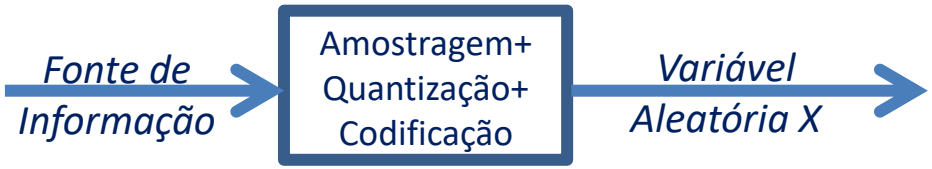
Por outro lado, se em um determinado instante n começa a ocorrer ruído granular, então forçosamente $b_n \neq b_{n-1} \rightarrow \rho(b_n)\rho(b_{n-1}) = -1 \rightarrow$ de (19) $\delta_n = \delta_{n-1}(1/\gamma)$. Portanto δ_n é dividido por γ, diminuindo δ e reduzindo o ruído granular.



Entropia – uma possível medida de informação

Conforme discutimos nos slides 5,6 e 7 do Cap I, é necessário que o Codificador de Fonte efetue a compressão da informação a ser transmitida de modo a minimizar a largura do espectro BW da onda EM que se propaga no canal de transmissão. Neste contexto, é necessário definir o que se entende por "informação". Consideremos o sinal de uma fonte de informação (áudio, vídeo) submetida ao processo de amostragem, quantização e codificação do Codificador de Fonte, conforme figura abaixo. A sequência de amostras quantizadas em amplitude (e, portanto, a sequência de palavras binárias na saída do Codificador de Fonte) pode ser interpretada como uma variável aleatória X. Neste contexto, dois postulados nos auxiliam a definir informação:

Postulado (I): "A observação da ocorrência de um evento do espaço amostral de uma variável aleatória nos dá informação":

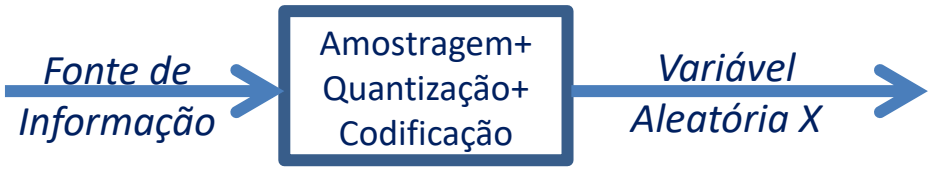


Postulado (II): "Eventos que ocorrem raramente no espaço amostral de uma variável aleatória contêm mais informação do que eventos que ocorrem frequentemente". Por exemplo:

- **"O sol nasceu à leste hoje pela manhã": evento frequente → pouca informação.**
- **"Porto Alegre foi atingida por um terremoto hoje pela manhã": evento raro → maior conteúdo de informação**.

A Entropia (proposta por Hartley, em 1928) é uma medida logarítmica de informação que reflete este raciocínio intuitivo.

Entropia – uma possível medida de informação



Ao registrarmos a sequência de amostras quantizadas em amplitude (e, portanto, ao registrarmos a sequência de palavras binárias de N bits correspondentes) em um Codificador de Fonte que opera com M níveis de quantização, após o registro de um número suficiente de amostras, podemos fazer um estudo estatístico da probabilidade de ocorrência de cada uma das M possíveis amostras (mensagens de $N = \log_2 M$ bits).

A saída do quantizador pode ser considerada uma variável aleatória discreta X, com espaço de amostras definido pelo conjunto $\Omega_X = \{m_k\} = \{m_0, m_1 \cdots m_{M-1}\}$ de M mensagens (M palavras binárias) m_k com probabilidade de ocorrência p_k, $k = 0,1 \cdots M-1$, conforme exemplo na tabela abaixo para um Codificador de Fonte que adota um ADC de N =3 bits (M =8 níveis de quantização).

Segundo Hartley, a **auto-informação** $h(m_k)$ implícita na ocorrência de uma mensagem (palavra binária) m_k, com probabilidade de ocorrência p_k, é definida por:

$$h(m_k) = -\log_2(p_k) \text{ [bits]} \qquad (20)$$

onde $\log_2(x) = \frac{\ln(x)}{\ln(2)}$. Analisando a equação (20) infere-se que:

(I) Como $0 \leq p_k \leq 1$, $h(m_k)$ é sempre um número positivo.
(II) $h(m_k)$ é medida em [bits] devido a função logarítmica de base 2.
(III) Como $\log_2(u)$ é uma função monotonicamente crescente com u, a auto-informação $h(m_k) = -\log_2(p_k)$ de uma mensagem (palavra binária) rara é maior do que a de uma mensagem (palavra binária) frequente, conforme exemplo abaixo para um Codificador de Fonte que adota um ADC de N =3 bits (M =8 níveis de quantização):

p_k=	0.05	0.1	0.15	0.2	0.25	0.15	0.075	0.025
palavra binária (mensagem m_k):	000	001	010	011	100	101	110	111
$h(m_k) = -\log_2(p_k) =$	4.322	3.322	2.737	2.322	2.000	2.737	3.737	5.322

mensagem (palavra binária) + frequente: menor auto-informação $h(m_k)$

mensagem (palavra binária) + rara: maior auto-informação $h(m_k)$

Entropia – uma possível medida de informação

A média da auto-informação $h(m_k) = -\log_2(p_k)$ transportada por cada uma das M mensagens m_k do conjunto $\Omega_X = \{m_k\} = \{m_0, m_1 \cdots m_{M-1}\}$ é denominada **entropia** da variável aleatória X, ou, equivalentemente, **entropia do conjunto Ω_X de mensagens.**

Assim, a entropia $H(X)$ da variável aleatória X, cujo espaço de amostras é o conjunto Ω_X de M mensagens, é dada por:

$$H(X) = E\{h(m_k)\} = E\{-\log_2(p_k)\} = -\sum_{k=0}^{M-1} p_k \log_2(p_k) \quad \text{[bits/mensagem]} \qquad (21)$$

onde $E\{.\}$ é o operador estatístico que retorna o valor esperado do argumento (ver https://pt.wikipedia.org/wiki/Valor_esperado).

Note que, se as M mensagens apresentam probabilidade de ocorrência iguais (mensagens equiprováveis), então $p_k = 1/M$ para $k = 0,1, \ldots, M-1$ e daí (21) simplifica para:

$$H(X) = -\sum_{k=0}^{M-1} \frac{1}{M} \log_2\left(\frac{1}{M}\right) = -\frac{1}{M} \sum_{k=0}^{M-1} -\log_2(M) =$$

$$= \frac{1}{M} \sum_{k=0}^{M-1} \log_2(M) = \frac{1}{M} M \log_2(M) = \log_2(M) \quad \text{[bits/mensagem]} \qquad (22)$$

Por exemplo, para um Codificador de Fonte que adota um ADC de $N = 2$ bits ($M = 4$ níveis de quantização) em que o sinal digitalizado apresenta níveis de amplitude equiprováveis, temos que $\Omega_X = \{m_k\} = \{m_0, m_1, m_2, m_3, \}$, $p_0 = p_1 = p_2 = p_3 = 0.25$ e a entropia $H(X) = E\{h(m_k)\}$ de cada mensagem do conjunto Ω_X de $M = 4$ mensagens de $N = 2$ bits resulta de (22) como sendo $H(X) = \log_2(4) = 2$ [bits/mensagem].

Entropia – uma possível medida de informação

Exemplo 1:

Seja um sistema para transmissão digital que utiliza no Codificador de Fonte um conjunto $\Omega_X = \{m_0, m_1\}$ com $M = 2$ possíveis mensagens (palavras binárias) de N =1 bit (ou $M = 2$ possíveis níveis de quantização).

Seja q a probabilidade de ocorrência que a saída X do quantizador assuma valor m_0, isto é, $q = P(X = m_0)$.

Pede-se: Determine o gráfico da Entropia de X em função de q.

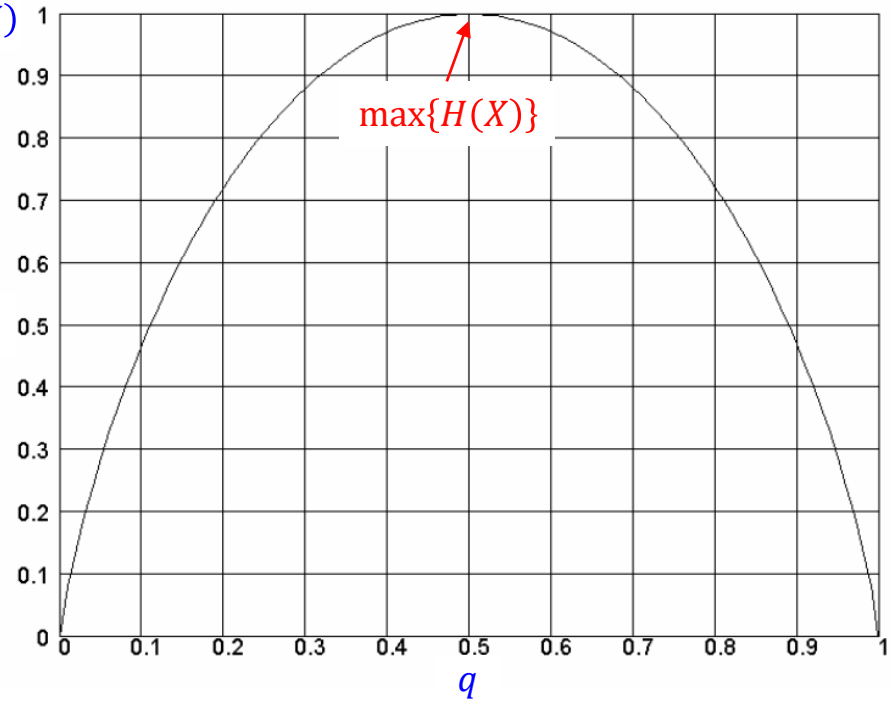
Solução: Para determinar o gráfico da Entropia de X em função de q, consideremos que

$$q = P(X = m_0) \rightarrow P(X = m_1) = 1 - q$$

Portanto,

$$H(X) = -\sum_{k=0}^{M-1} p_k \log_2(p_k) = -p_0 \log_2(p_0) - p_1 \log_2(p_1) = -q\log_2(q) - (1-q)\log_2(1-q) \quad \text{[bits/mensagem]} \qquad (23)$$

Plotando $H(X) \times q$ a partir de (23) obtemos o gráfico



Note no gráfico que a entropia, i.e., a média da auto-informação $h(m_k) = -\log_2(p_k)$ transportada por cada uma das $M = 2$ mensagens m_k do conjunto $\Omega_X = \{m_k\} = \{m_0, m_1\}$ é máxima e de valor $\max\{H(X)\}$ quando as mensagens são equiprováveis, i.e., quando $q = P(X = m_0) = P(X = m_1) = 0.5$.

Embora este resultado tenha sido obtido para $M = 2$ mensagens, o resultado é geral e válido para qualquer valor de M, i.e., **a média $H(X)$ da auto-informação transportada por cada uma das M mensagens m_k do conjunto de mensagens $\Omega_X = \{m_k\}$ de um sistema de transporte de informação é máxima e dada por $\max\{H(X)\} = \log_2(M)$ quando a probabilidade de ocorrência das M mensagens são iguais.**

Taxa de transporte da informação

Consideremos uma fonte de informação conectada à entrada do Codificador de Fonte de um sistema digital que transporta informação através de um conjunto $\Omega_X = \{m_k\} = \{m_0, m_1 \cdots m_{M-1}\}$ com M possíveis mensagens (=palavras binárias), conforme mostra a figura abaixo.

Suponhamos que estamos registrando em disco a saída X do Codificador de Fonte, contando a frequência de ocorrência (=probabilidade p_k, $k = 0,1 \cdots M-1$) das M possíveis palavras binárias, e determinando através de (21) a média da auto-informação transportada por cada uma das M palavras binárias m_k do conjunto Ω_X (i.e., determinando a entropia $H(X)$).

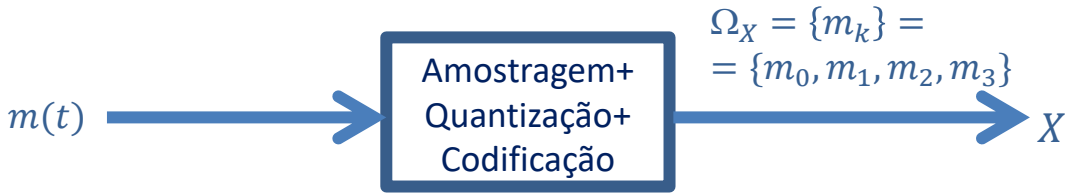
Se a fonte de informação é amostrada a uma frequência de amostragem f_s [Sa/s – *samples per second*] então na saída X do Codificador de Fonte são geradas f_s $[mensagens/segundo]$ com uma entropia $H[bits/mensagem]$. Portanto, a **Taxa de Informação** R , **que meda o número médio de bits por segundo transportados pelo Codificador de Fonte**, é dada por:

$$R[\frac{\text{bits}}{\text{s}}] = f_s \left[\frac{\text{mensagens}}{\text{s}}\right] H \left[\frac{\text{bits}}{\text{mensagem}}\right] \qquad (24)$$

Taxa de transporte da informação

Exemplo 2:

Seja um sistema para transmissão digital cujo Codificador de Fonte digitaliza um sinal analógico $m(t)$ através de um conjunto $\Omega_X = \{m_0, m_1, m_2, m_3\}$ com $M = 4$ possíveis mensagens, conforme mostra a figura abaixo:



As palavras binárias na saída X do Codificador são tais que a ocorrência de uma não altera a probabilidade de ocorrência da outra (i. é, as mensagens são estatisticamente independentes).

As probabilidades de ocorrência são: $P(X = m_0) = P(X = m_3) = 1/8$ e $P(X = m_1) = P(X = m_2) = 3/8$. O intervalo de amostragem adotado no ADC do Codificador de Fonte é $T_s = \frac{1}{f_s} = 50\mu s$.

Pede-se: Determine a taxa de informação $R[\frac{\text{bits}}{\text{s}}]$ gerada pelo sinal $m(t)$ na saida X do Codificador de Fonte.

Solução: A entropia, i.e., a média da auto-informação transportada por cada uma das $M = 4$ mensagens m_k do conjunto $\Omega_X = \{m_0, m_1, m_2, m_3\}$ é dada pela equação (21):

$$H(X) = -\sum_{k=0}^{M-1} p_k \log_2(p_k) = -\frac{1}{8}\log_2\left(\frac{1}{8}\right) - \frac{3}{8}\log_2\left(\frac{3}{8}\right) - \frac{3}{8}\log_2\left(\frac{3}{8}\right) - \frac{1}{8}\log_2\left(\frac{1}{8}\right) = 1.8 \text{ [bits/mensagem]}$$

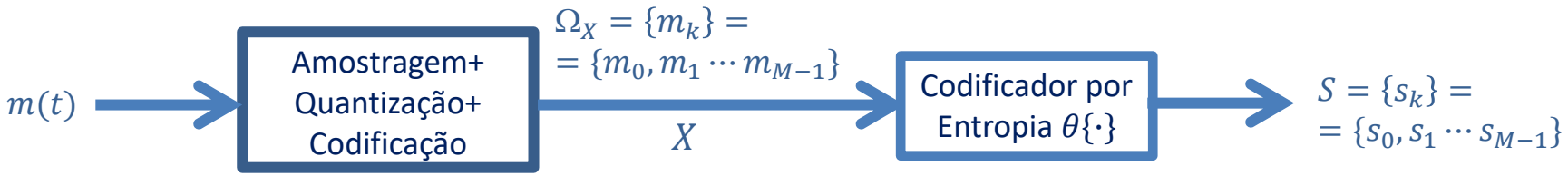
$$f_s = \frac{1}{T_s} = \frac{1}{50\mu s} = 20 \text{ [KSa/s]} = 20000 \text{ [mensagens/segundo]}$$

Da equação (24):

$$R\left[\frac{\text{bits}}{\text{s}}\right] = f_s\left[\frac{\text{mensagens}}{\text{s}}\right] H\left[\frac{\text{bits}}{\text{mensagem}}\right] = 20000 \text{ [mensagens/segundo]} \times 1.8 \text{ [bits/mensagem]} = 36 \text{ [Kbps]}$$

Codificação por entropia

Consideremos o Codificador de Fonte de um sistema digital que digitaliza a informação através de um conjunto $\Omega_X = \{m_k\} = \{m_0, m_1 \cdots m_{M-1}\}$ com M possíveis mensagens (=palavras binárias) $N = \log_2(M)$ [bits], seguido pelo bloco "Codificador por Entropia" conforme mostra a figura abaixo:



O Codificador por Entropia processa cada uma das M possíveis palavras binárias $\Omega_X = \{m_k\} = \{m_0, m_1 \cdots m_{M-1}\}$ de $N = \log_2(M)$ [bits] em sua entrada e associa a cada uma delas uma respectiva palavra-código (=símbolo) $S = \{s_k\} = \{s_0, s_1 \cdots s_{M-1}\}$ cujo número de bits é controlado por uma função $\beta(p_k)$ da probabilidade de ocorrência p_k da mensagem m_k em sua entrada. **A função $\boldsymbol{\beta(p_k)}$ é tal que quanto maior a probabilidade de ocorrência $\boldsymbol{p_k}$ da mensagem $\boldsymbol{m_k}$ menor será o número de bits atribuídos à respectiva palavra-código $\boldsymbol{s_k}$ associada.**

O efeito da função $\beta(p_k)$ resulta que **mensagens que ocorrem frequentemente necessitem de menos bits para serem transmitidas** e, portanto, **o efeito global é o de reduzir a quantidade de bits transmitidos.**

Podemos considerar o bloco "Codificador por Entropia" na figura acima como um **operador $\boldsymbol{\theta}\{\cdot\}$**, tal que $s_k = \theta\{m_k\}$, onde a associação da mensagem m_k à palavra-código s_k através do ***codebook*** definido por $s_k = \theta\{m_k\}$ obedece à função $\beta(p_k)$ referida no parágrafo anterior. Existem várias possíveis definições para a função $\beta(p_k)$. Veremos adiante neste capítulo que a função $\beta(p_k)$ ótima é implementada pelo **Código de Huffman**.

Quando um sistema digital é projetado, é feito um estudo estatístico da frequência de ocorrência p_k de cada uma das M possíveis mensagens m_k para que o mapeamento $s_k = \theta\{m_k\}$ efetuado pelo código compressor possa ser especificado, i.e., de modo que a função $\beta(p_k)$ possa ser determinada. O conjunto de M valores obtidos para p_k pode ser considerado uma boa aproximação das probabilidades de ocorrência de cada uma das M possíveis mensagens desde que seja utilizado um número suficientemente grande de amostras.

Por exemplo, o Código Morse, brevemente discutido nos slides 6 e 7 do Cap I, teve a sua função $\beta(p_k)$ determinada experimentalmente a partir de um estudo estatístico que determinou a frequência de ocorrência p_k dos caracteres alfabéticos utilizados na escrita de textos em língua inglesa.

Codificação por entropia

Um código p/ compressão por entropia é, portanto, um operador $\theta\{\cdot\}$, tal que $s_i = \theta\{x_i\}$, onde a associação da mensagem x_i à palavra-código s_i através do *codebook* definido por $s_i = \theta\{x_i\}$ obedece à uma função $\beta(p_i)$, onde $i = 0,1 \cdots M-1$, sendo M o número de possíveis mensagens x_i (=palavras binárias) de $N = \log_2(M)$ [bits] que constam no conjunto $\Omega_X = \{x_i\} = \{x_0, x_1 \cdots x_{M-1}\}$.

A função $\beta(p_i)$ é tal que quanto maior a probabilidade de ocorrência p_i da mensagem x_i menor será o número de bits atribuídos à respectiva palavra-código s_k associada.

Os dados de entrada e saída do operador $\theta\{\cdot\}$ são respectivamente:

$\Omega = \{x_i\} = \{x_0, x_1 \cdots x_{M-1}\}$ é o conjunto de M possíveis mensagens a serem codificadas através de $s_i = \theta\{x_i\}$.
$S = \{s_i\} = \{s_0, s_1 \cdots s_{M-1}\}$ é o conjunto de M possíveis palavras-código ou símbolos resultantes da codificação $s_i = \theta\{x_i\}$.

É imperativo que o mapeamento $s_i = \theta\{x_i\}$ efetuado pelo operador $\theta\{\cdot\}$ seja um **mapeamento unívoco**, de modo que as palavras-código $\tilde{s}_i$ recebidas no decodificador por entropia do RX sejam **univocamente decodificáveis** através de $\tilde{x}_i = \theta^{-1}\{\tilde{s}_i\}$, caso contrário o decodificador do RX encontrará ambiguidades no conjunto de palavras-código $\tilde{s}_i$ recebidas, incorrendo em erros de decodificação na recuperação das mensagens $\tilde{x}_i$. Veremos adiante neste capítulo como definir códigos univocamente decodificáveis (códigos UD).

O **conjunto de caracteres do código** ou **alfabeto do código** é o conjunto $A = \{a_0, a_1, \ldots, a_{D-1}\}$ composto por D elementos, de cuja composição são formadas cada uma das palavra-código. Neste estudo, o escopo enfocará códigos binários, em que o alfabeto é $A = \{0,1\}$.

O **tamanho** ℓ_i de uma palavra-código ou símbolo s_i é definido pelo número de caracteres do alfabeto A utilizado na construção da palavra-código. Por exemplo, no *codebook* abaixo é mostrado o tamanho ℓ_i de cada símbolo s_i p/ um código binário:

Mensagem x_i	Palavra bin	Palavra-Código s_i associada a x_i por $s_i = \theta\{x_i\}$	ℓ_i
x_0	00	0	1
x_1	01	010	3
x_2	10	01	2
x_3	11	10	2

Codificação por entropia

O objetivo de um codificador por entropia é, através de um código $s_i = \theta\{x_i\}$, minimizar o **tamanho médio** $\bar{L}$ dos símbolos s_i do conjunto de M possíveis símbolos $S = \{s_i\} = \{s_0, s_1, \ldots, s_{M-1}\}$ na saída do codificador respectivamente correspondentes às mensagens no conjunto $\Omega = \{x_i\} = \{x_0, x_1 \cdots x_{M-1}\}$ aplicadas à sua entrada, sendo o tamanho médio $\bar{L}$ dos símbolos no conjunto dos M possíveis símbolos $S = \{s_i\}$ dado por:

$$\bar{L} = \sum_{i=0}^{M-1} p_i \ell_i \tag{25}$$

onde p_i é a probabilidade de ocorrência da mensagem x_i e onde ℓ_i é o tamanho do símbolo s_i associado à mensagem x_i através do código $s_i = \theta\{x_i\}$.

A Codificação por Entropia assume que a fonte de informação é **sem memória.** Uma fonte é considerada sem memória quando as mensagens emitidas pela fonte são estatisticamente independentes, i.e., a ocorrência de uma determinada mensagem x_i não afeta a probabilidade de ocorrência da mensagem x_j , com $i, j = 0,1,\cdots, M-1$.

Esta condição de a fonte de informação ser sem memória é necessária pois, caso contrário, a função $\bar{L} = f(p_i, \ell_i)$ a ser minimizada dependeria do desenrolar temporal da sequência de mensagens emitidas pela fonte, **o que resultaria em um *codebook* $\theta\{\cdot\}$ variável no tempo**.

Embora poucas fontes físicas sigam exatamente o modelo de uma fonte sem memória, códigos $\theta\{\cdot\}$ constantes no tempo (resultantes da suposição de que as mensagens emitidas pela fonte são estatisticamente independentes) são amplamente utilizados como códigos compressores, mesmo quando a dependência estatística da fonte resulta na impossibilidade de minimização de $\bar{L}$ durante a totalidade do tempo de codificação.

Em alguns sistemas que operam com sinais altamente dinâmicos, a probabilidade de ocorrência p_i da mensagem x_i é avaliada periodicamente para cada uma das M mensagens e um novo *codebook* $s_i = \theta\{x_i\}$ é então definido após cada período de avaliação. Esta redefinição adaptativa e periódica do *codebook* $s_i = \theta\{x_i\}$ obviamente aumenta o custo computacional do processo de compressão. Um exemplo de técnica de compressão que utiliza um *codebook* adaptativo é o CELP – *Code Excited Linear Prediction* (ver https://en.wikipedia.org/wiki/Code-excited_linear_prediction), técnica que é largamente utilizada na compressão de voz em tempo real em telefones celulares.

Codificação por entropia

Exemplo 3: Seja um sistema para transmissão digital que utilize no Codificador de Fonte um conjunto $\Omega = \{x_0, x_1, x_2, x_3\} = \{00,01,10,11\}$ com $M = 4$ possíveis mensagens . As mensagens são tais que a ocorrência de uma não altera a probabilidade de ocorrência da outra (i.e., as mensagens são estatisticamente independentes). As probabilidades p_i de ocorrência de cada mensagem são : $P(X = x_0) = \frac{1}{2}$, $P(X = x_1) = \frac{1}{4}, P(X = x_2) = P(X = x_3) = \frac{1}{8}$. 0 *codebook* do código $\theta\{\cdot\}$ é conforme tabela ao lado.

Mensagem x_i	Palavra bin	Palavra-Código s_i associada a x_i por $s_i = \theta\{x_i\}$
x_0	00	0
x_1	01	10
x_2	10	110
x_3	11	111

Pede-se:

(a) Determine a entropia da fonte $H(X)$.

(b) Determine o comprimento médio $\bar{L}\ (\theta)$ do código $\theta\{\cdot\}$.

Solução: Da tabela do *codebook* dado no enunciado temos que as probabilidades de ocorrência p_i e o tamanho ℓ_i de cada palavra-código(símbolo) s_i são conforme tabela abaixo.

x_i	p_i	Símbolo s_i associado a x_i por $s_i = \theta\{x_i\}$	ℓ_i
x_0	1/2	0	1
x_1	1/4	10	2
x_2	1/8	110	3
x_3	1/8	111	3

(a) Da equação (21) no slide 39 e das probabilidades p_i na tabela ao lado temos que a entropia da fonte $H(X)$ resulta em:

$$H(X) = -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{4}\log_2\left(\frac{1}{4}\right) - \frac{1}{8}\log_2\left(\frac{1}{8}\right) - \frac{1}{8}\log_2\left(\frac{1}{8}\right) = 1.75\ [\text{bits/mensagem}]$$

(b) Da equação (25) no slide anterior e das probabilidades p_i e tamanhos ℓ_i na tabela acima temos que o comprimento médio $\bar{L}\ (\theta)$ do código $\theta\{\cdot\}$ resulta em:

$$\bar{L}(\theta) = \frac{1}{2}\cdot 1 + \frac{1}{4}\cdot 2 + \frac{1}{8}\cdot 3 + \frac{1}{8}\cdot 3 = 1.75\ [\text{bits/símbolo}]$$

Note, por exemplo, que se transmitíssemos as palavras binárias de 2 bits sem compressão, para 10000 mensagens a serem enviadas do TX ao RX teríamos que enviar 20000 bits através o canal de transmissão, mas o uso do *codebook* $\theta\{\cdot\}$ reduz este número para 17500 bits.

Codificação por entropia

Exemplo 4: Seja o código compressor $\theta\{\cdot\}$ conforme definido no *codebook* na tabela abaixo:

x_i	p_i	Símbolo s_i associado a x_i por $s_i = \theta\{x_i\}$	ℓ_i
x_0	1/3	0	1
x_1	1/3	10	2
x_2	1/3	11	2

Pede-se: (a) Determine a entropia da fonte $H(X)$. **(b)** Determine o comprimento médio $\bar{L}\ (\theta)$ do código $\theta\{\cdot\}$.

Solução:

(a) Da equação (21) no slide 39 e das probabilidades p_i na tabela acima temos que a entropia da fonte $H(X)$ resulta em:

$$H(X) = -\frac{1}{3}\log_2\left(\frac{1}{3}\right) - \frac{1}{3}\log_2\left(\frac{1}{3}\right) - \frac{1}{3}\log_2\left(\frac{1}{3}\right) = 1.58\ [\text{bits/mensagem}]$$

(b) Da equação (25) no slide 45 e das probabilidades p_i e tamanhos ℓ_i na tabela acima temos que o comprimento médio $\bar{L}\ (\theta)$ do código $\theta\{\cdot\}$ resulta em:

$$\bar{L}(\theta) = \frac{1}{3}\cdot 1 + \frac{1}{3}\cdot 2 + \frac{1}{3}\cdot 2 = 1.67\ [\text{bits/símbolo}]$$

Note no Exemplo 3 no slide anterior que o comprimento médio $\bar{\boldsymbol{L}}(\boldsymbol{\theta}) = \mathbf{1.75}\ [\mathbf{bits/símbolo}]$ das palavras-código s_i geradas pelo *codebook* do código $\theta\{\cdot\}$ resultou igual à média da auto-informação $\boldsymbol{H(X)} = \mathbf{1.75}\ [\mathbf{bits/mensagem}]$ transportada pelas M mensagens x_i do conjunto Ω. Esta situação em que $\bar{\boldsymbol{L}}(\boldsymbol{\theta}) = \boldsymbol{H(X)}$ maximiza a compressão porque indica que toda a informação redundante nas M mensagens x_i do conjunto Ω foi eliminada através do mapeamento $s_i = \theta\{x_i\}$ efetuado pelo *codebook* do código $\theta\{\cdot\}$.

Já com o *codebook* do código $\theta\{\cdot\}$ do presente Exemplo 4 esta condição ótima não consegue ser atingida porque $\bar{L}(\theta) = 1.67\ [\text{bits/símbolo}]$ é maior que $H(X) = 1.58\ [\text{bits/mensagem}]$, indicando que nem toda a informação redundante nas M mensagens x_i do conjunto Ω foi eliminada através do mapeamento $s_i = \theta\{x_i\}$ efetuado pelo *codebook* do código $\theta\{\cdot\}$.

Códigos univocamente decodificáveis (Códigos UD)

Conforme brevemente discutido no slide 44, é imperativo que o mapeamento $s_i = \theta\{x_i\}$ efetuado pelo *codebook* do código $\theta\{\cdot\}$ seja um **mapeamento unívoco**, de modo que as palavras-código $\widetilde{s_i}$ recebidas no decodificador por entropia do RX sejam **univocamente decodificáveis** através de $\widetilde{x_i} = \theta^{-1}\{\widetilde{s_i}\}$, caso contrário o decodificador do RX encontrará ambiguidades no conjunto de palavras-código $\widetilde{s_i}$ recebidas, incorrendo em erros de decodificação na recuperação das mensagens $\widetilde{x_i}$. Um código $\theta\{\cdot\}$ cujo *codebook* defina um mapeamento $s_i = \theta\{x_i\}$ unívoco é denominado de **código univocamente decodificável** (código UD).

Um código é Univocamente Decodificável (UD) quando o seu *codebook* for tal que qualquer sequência de caracteres do alfabeto A passível de ser formada a partir da **justaposição** de um número qualquer de símbolos pertencentes ao conjunto $S = \{s_i\} = \{s_0, s_1, \dots, s_{M-1}\}$ puder ser associada, ao ser decodificada no RX, a uma única mensagem em $\Omega = \{x_i\} = \{x_0, x_1, \dots, x_{M-1}\}$.

Conceito de justaposição: A justaposição de N símbolos (ou palavras-código) é a sequência $\alpha = [s_i, s_{i+1} \cdots s_{i+N-1}]$ formada pela transmissão do símbolo s_i, seguido da transmissão do símbolo s_{i+1}, e assim sucessivamente até a transmissão do símbolo s_{i+N-1}.

Exemplo 5: Seja o código $\theta\{\cdot\}$ conforme definido no *codebook* na tabela abaixo. **Pede-se:** Determine se $\theta\{\cdot\}$ é UD.

Mensagem	Palavra-código s_i associada a x_i por $s_i = \theta\{x_i\}$
x_0	0
x_1	010
x_2	01
x_3	10

Solução: Vamos partir da palavra-código com maior numero de bits ($s_1 = 010$) e verificar se ela, ao ser transmitida pelo TX, é passível de ser decomposta em uma justaposição das demais palavras-código. Especificamente, a pergunta que temos que responder é: Como decodificar $s_1 = 010$ no RX? As possíveis inferências que o decodificador do RX faria são:

$s_1 = 010 \rightarrow \widetilde{x_i} = \theta^{-1}\{\widetilde{s_i}\} = x_1$ (correto)

$s_2 s_0 = 010 \rightarrow \widetilde{x_i} = \theta^{-1}\{\widetilde{s_i}\} = x_1$ (incorreto)

$s_0 s_3 = 010 \rightarrow \widetilde{x_i} = \theta^{-1}\{\widetilde{s_i}\} = x_1$ (incorreto)

Dado que ocorrem as ambiguidades $s_2 s_0$ e $s_0 s_3$ no decodificador do RX quando o TX transmite s_1, então **o código $\theta\{\cdot\}$ não é UD**.

Códigos instantâneos

As ambiguidades do *codebook* do código $\theta\{\cdot\}$ do Exemplo 5 no slide anterior talvez pudessem ser resolvidas se aguardássemos a recepção de bits adicionais para resolver a incerteza, mas tal tempo de espera é indesejável, dada a constante busca por velocidade de decodificação (é desejável que o RX seja capaz de decodificar instantaneamente as palavras-código s_i à medida que as mesmas são recebidas no decodificador do RX).

Uma maneira de assegurar que um código seja UD e que nenhum tempo de espera seja necessário para a correta decodificação é utilizar **códigos prefixos ou instantâneos**.

A denominação "instantâneo" para tais códigos decorre de não haver necessidade de aguardar a recepção de bits adicionais para que se resolva ambiguidades.

Um código instantâneo ou prefixo pode ser decodificado sem referência a palavras-código futuras, porque o final de uma palavra-código é imediatamente reconhecido no decodificador.

Todos os códigos instantâneos são UD, mas nem todos os códigos UD são instantâneos. Ou seja, o conjunto dos códigos instantâneos é um sub-conjunto do conjunto dos códigos UD, conforme mostra a figura ao lado.

Um código é instantâneo se nenhuma palavra-código é prefixo de nenhuma outra palavra-código pertencente ao código.

Conceito de sequência prefixo: Sejam as sequências α_a, α_b e α_c formadas pela justaposição de, respectivamente, N_a, N_b e N_c palavras-código s_i, pertencentes ao *codebook* do código $\theta\{\cdot\}$, sendo $N_a = N_b + N_c$ um número qualquer de palavras-código. Dizemos que a sequência α_b é prefixo da sequência α_a, se α_a puder ser representada por $\alpha_b\alpha_c$, para alguma sequência α_c denominada sufixo.

Códigos instantâneos

Exemplo 6: Seja o código $\theta\{\cdot\}$ conforme definido no *codebook* na tabela abaixo. **Pede-se:** Determine se $\theta\{\cdot\}$ é instantâneo.

Mensagem	Palavra-Código s_i associada a x_i por $s_i = \theta\{x_i\}$
x_0	10
x_1	00
x_2	11
x_3	110

Solução:

Como 11 é prefixo de 110, $\theta\{\cdot\}$ não é instantâneo.

No entanto, não podemos afirmar que não seja UD, pelo fato de não ser instantâneo:

Teste para verificar se um código é UD

Seja um código $\theta\{\cdot\}$ com alfabeto $A = \{\alpha_0, \alpha_1, \ldots, \alpha_{D-1}\}$ e conjunto imagem $S = \{s_i\} = \{s_0, s_1, \ldots, s_{M-1}\}$ definido pelo seu *codebook*. Para testar se $\theta\{\cdot\}$ é UD, constrói-se uma sequência de conjuntos de símbolos $S_0, S_1, \ldots$ obedecendo ao seguinte critério de formação:

1. S_0 é o próprio conjunto imagem do *codebook*, i.e., $S_0 = S = \{s_i\} = \{s_0, s_1, \ldots, s_{M-1}\}$.

2. Para definir S_1, forma-se a partir de S_0 o conjunto **P** de todos os pares $s_i s_j$ de palavras-código, $s_i \neq s_j$ possíveis de serem formados por justaposição de duas palavras-código distintas pertencentes ao conjunto S_0:

	s_0	s_1	...	s_{M-1}
s_0	-	$s_0 s_1$	...	$s_0 s_{M-1}$
s_1	$s_1 s_0$	-	...	$s_1 s_{M-1}$
...	...	...	-	
s_{M-1}	$s_{M-1} s_0$	$s_{M-1} s_1$	...	-

3. Se a palavra-código $s_i \in S_0$ é prefixo da palavra-código $s_j \in S_0$, i.e. $s_j = s_i\sigma$, então o sufixo σ é um elemento do conjunto S_1, i.e. $\sigma \in S_1$.

Executa-se a verificação $s_j = s_i\sigma$ para todos os elementos de **P** até que todos os sufixos sejam atribuídos ao conjunto $S_1 = \{\alpha_0, \alpha_1, \ldots\}$, onde cada sequência α_k de caracteres de A é um sufixo originado pelo resultado positivo do teste $s_j = s_i\sigma$.

4. Para definir S_n, $n > 1$, compara-se S_0 e S_{n-1} de modo bidirecional:

I) Se uma palavra-código $s_i \in S_0$ é prefixo de uma sequência $\alpha_j \in S_{n-1}$, tal que $\alpha_j = s_i\sigma$, então o sufixo $\sigma \in S_n$.

II) Se uma sequência $\alpha'_j \in S_{n-1}$ é prefixo de uma palavra-código $s'_i \in S_0$ tal que $s'_i = \alpha'_j\sigma'$, então o sufixo $\sigma' \in S_n$.

5. Define-se tantos conjuntos S_n até um valor de n tal que $S_n = \{\emptyset\}$ ou até um valor de n tal que $S_n = S_{n-1}$.

6. O código $\theta\{\,.\,\}$ é UD se e somente se **nenhum** dos conjuntos da sequência de conjuntos S_1, S_2,... contenha uma palavra-código que pertença ao conjunto S_0 .

Teste para verificar se um código é UD

Exemplo 7: Usando o algoritmo descrito no slide anterior verifique se o código $\theta\{\cdot\}$ abaixo, com alfabeto $A = \{a, b, c, d, e\}$ é instantâneo e/ou UD.

Mensagem	Palavra-Código s_i associada a m_i por $s_i = \theta\{x_i\}$
x_0	a
x_1	c
x_2	ad
x_3	abb
x_4	bad
x_5	deb
x_6	$bbcde$

Teste para verificar se um código é UD

Solução:

S_0	S_1							
a	d							
c	bb							
ad								
abb								
bad								
deb								
$bbcde$								

- Construir S_1:
- S_1 contém todos os sufixos encontrados ao mapear palavras de S_0 que são prefixo de outras palavras de S_0.
- Alguma palavra de $S_1 \in S_0$?
  - Se sim, $\theta\{\,.\,\}$ não é UD.
  - Se não, construir S_2, a partir de S_0 e S_1.

Teste para verificar se um código é UD

S_0	S_1	S_2						
a	d	eb						
c	bb	cde						
ad								
abb								
bad								
deb								
$bbcde$								

- Construir S_2:
- S_2 contém todos os sufixos encontrados ao mapear palavras de S_0 que são prefixo de palavras de S_1 e
- S_2 contém todos os sufixos encontrados ao mapear palavras de S_1 que são prefixo de palavras de S_0.
- Alguma palavra de $S_2 \in S_0$?
  - Se sim, $\theta\{\,.\,\}$ não é UD.
  - Se não, construir S_3, a partir de S_0 e S_2.

Teste para verificar se um código é UD

S_0	S_1	S_2	S_3					
a	d	eb	de					
c	bb	cde						
ad								
abb								
bad								
deb								
$bbcde$								

- Construir S_3:
- S_3 contém todos os sufixos encontrados ao mapear palavras de S_0 que são prefixo de palavras de S_2 e
- S_3 contém todos os sufixos encontrados ao mapear palavras de S_2 que são prefixo de palavras de S_0.
- Alguma palavra de $S_3 \in S_0$?
  - Se sim, $\theta\{\,.\,\}$ não é UD.
  - Se não, construir S_4, a partir de S_0 e S_3.

Teste para verificar se um código é UD

S_0	S_1	S_2	S_3	S_4				
a	d	eb	de	b				
c	bb	cde						
ad								
abb								
bad								
deb								
$bbcde$								

- Construir S_4:
- S_4 contém todos os sufixos encontrados ao mapear palavras de S_0 que são prefixo de palavras de S_3 e
- S_4 contém todos os sufixos encontrados ao mapear palavras de S_3 que são prefixo de palavras de S_0.
- Alguma palavra de $S_4 \in S_0$?
  - Se sim, $\theta\{\,.\,\}$ não é UD.
  - Se não, construir S_5, a partir de S_0 e S_4.

Teste para verificar se um código é UD

S_0	S_1	S_2	S_3	S_4	S_5			
a	d	eb	de	b	ad			
c	bb	cde			$bcde$			
ad								
abb								
bad								
deb								
$bbcde$								

- Construir S_5:
- S_5 contém todos os sufixos encontrados ao mapear palavras de S_0 que são prefixo de palavras de S_4 e
- S_5 contém todos os sufixos encontrados ao mapear palavras de S_4 que são prefixo de palavras de S_0.
- Alguma palavra de $S_5 \in S_0$?

Teste para verificar se um código é UD

S_0	S_1	S_2	S_3	S_4	S_5			
a	d	eb	de	b	ad			
c	bb	cde			$bcde$			
ad								
abb								
bad								
deb								
$bbcde$								

- Construir S_5:
- S_5 contém todos os sufixos encontrados ao mapear palavras de S_0 que são prefixo de palavras de S_4 e
- S_5 contém todos os sufixos encontrados ao mapear palavras de S_4 que são prefixo de palavras de S_0.
- Alguma palavra de $S_5 \in S_0$?

Visto que $ad \in S_5$ e $ad \in S_0$, logo $\theta\{\cdot\}$ não é UD.

Teste para verificar se um código é UD

S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8
a	d	eb	de	b	ad	d	eb	$\{\emptyset\}$
c	bb	cde			$bcde$			
ad								
abb								
bad								
deb								
$bbcde$								

- Note que poderíamos ter encerrado o procedimento ao obter S_5 no slide anterior quando, então, já tínhamos elementos suficientes para decidir que $\theta\{\,.\,\}$ não é UD.
- No entanto, neste slide, o procedimento foi continuado até não encontrarmos mais elementos na coluna S construída.
- Este é o procedimento que deve obrigatoriamente ser seguido caso não se encontre uma palavra pertencente a S_0 em alguma das colunas construídas. Somente ao encontrar $\{\emptyset\}$ é possível afirmar que o código é UD.

Teste para verificar se um código é UD

Exemplo 8: Usando o algoritmo descrito nos slide 51 verifique se os códigos $\theta_I\{\cdot\}$, $\theta_{II}\{\cdot\}$ e $\theta_{III}\{\cdot\}$ abaixo, com alfabeto $A = \{0,1\}$, são instantâneos e/ou UD.

$\theta_I\{\cdot\}$

S_0	S_1	S_2
1	0	0
00		1
01		
10		

Resposta: Não é instantâneo, nem UD.

$\theta_{II}\{\cdot\}$

S_0	S_1	S_2
0	1	11
01	11	1
011		
111		

Resposta: Não é instantâneo, mas é UD.

$\theta_{III}\{\cdot\}$

S_0	S_1
0	$\{\emptyset\}$
10	
110	
111	

Resposta: É Instantâneo (e, portanto, é UD).

No link Videoaula Códigos Instantâneos e UD - Profa. Cristina De Castro encontra-se disponível uma videoaula com a solução passo a passo do Exemplo 8.

Teorema da Codificação de Fonte (TCF) de Shannon (*Noiseless Coding Theorem*)

Seja uma variável aleatória discreta *X* com espaço de amostras definido pelo conjunto $\Omega = \{x_i\} = \{x_0, x_1 \cdots x_{M-1}\}$de M eventos estatisticamente independentes x_i, com probabilidade de ocorrência $p_i, i = 0,1, \dots, M-1$.

Então é possível construir um **código Instantâneo** $\theta\{\cdot\}$ através de um *codebook* de palavras-código $S = \{s_i\} = \{s_0, s_1, \dots, s_{M-1}\}$ formadas a partir do alfabeto $A = \{0,1\}$, tal que o conjunto $L = \{\ell_i\} = \{\ell_0, \ell_1 \cdots \ell_{M-1}\}$ dos tamanhos das palavras-código respectivas em S satisfaça à desigualdade

$$H(X) \leq \bar{L} < H(X) + 1 \tag{26}$$

onde:

$H(X)$ é a entropia da fonte X (dada pela equação (21) do slide 39)

$\bar{L}$ é o tamanho médio das palavras-códigos (dado pela equação (25) do slide 45)

(ver https://en.wikipedia.org/wiki/Shannon%27s_source_coding_theorem)

O TCF garante a viabilidade teórica de implementação de códigos instantâneos, cujo tamanho médio $\bar{L}$ dos símbolos pode ser reduzido a um valor tão pequeno quanto o valor da Entropia $H(X)$ da fonte, ou, se impossível, pelo menos a um valor menor que $H(X) + 1$.

Uma decorrência do TCF é a definição da **Eficiência de Codificação η** dada por (ver discussão na solução do Exemplo 4 no slide 47):

$$\eta = \frac{H(X)}{\bar{L}} \tag{27}$$

• Um código $\theta\{\cdot\}$ é **Absolutamente Ótimo** *(matched to the source* - casado com a fonte) quando $\eta = 1.0$, isto é, quando $\bar{L} = H(X)$. Ver, por exemplo, o código do Exemplo 3 no slide 46.

• Um código $\theta\{\cdot\}$ é **Quase Absolutamente Ótimo** quando $H(X) \leq \bar{L} < H(X) + 1$.

Códigos Ótimos

Embora o TCF nos garanta que é possível obter códigos instantâneos com $\bar{L}$ tão pequeno quanto a própria entropia $H(X)$ da fonte, nenhuma informação é dada sobre como construir tais códigos.

A construção de códigos ótimos baseia-se na minimização do tamanho médio $\bar{L}$ das palavras-códigos do *codebook*, sendo $\bar{L}$ dado pela equação (25) do slide 45.

Um código instantâneo que minimize $\bar{L}$ é denominado de **Código Ótimo**.

Há um teorema que prova que, se um código ótimo $\theta^*\{\cdot\}$ resulta em $\bar{L}^*$, então é impossível existir um outro código instantâneo $\theta\{\cdot\}$ com tamanho médio $\bar{L}$ tal que $\bar{L} < \bar{L}^*$.

Um Código Ótimo binário cujas palavras-código $S = \{s_i\} = \{s_0, s_1 \cdots s_{M-1}\}$ são formadas a partir do alfabeto $A = \{0,1\}$ satisfaz as seguintes propriedades:

(I) Palavras-código com maior probabilidade possuem menor tamanho (menor número de bits).

(II) As 2 palavras-código menos prováveis possuem o mesmo tamanho (mesmo número de bits).

(III) As 2 palavras-código menos prováveis diferem somente no valor do último bit.

Códigos Ótimos

Exemplo 9: Verifique se o código $\theta\{\,.\,\}$ definido no *codebook* abaixo é um Código Ótimo.

Mensagem	p_i	Palavra-Código s_i associada a x_i por $s_i = \theta\{x_i\}$
x_0	0.6	0
x_1	0.2	100
x_2	0.1	101
x_3	0.04	1101
x_4	0.06	1110

Solução:

Do slide anterior, temos que um Código Ótimo binário cujas palavras-código $S = \{s_i\} = \{s_0, s_1 \cdots s_{M-1}\}$ são formadas a partir do alfabeto $A = \{0,1\}$ satisfaz as seguintes propriedades:

(I) Palavras-código com maior probabilidade possuem menor tamanho (menor número de bits).

(II) As 2 palavras-código menos prováveis possuem o mesmo tamanho (mesmo número de bits).

(III) As 2 palavras-código menos prováveis diferem somente no valor do último bit.

Note no *codebook* acima que:

As propriedades (I) e (II) são satisfeitas.

A propriedade (III) não é satisfeita: s_3 e s_4 não diferem somente no último bit.

Portanto o código $\theta\{\cdot\}$ definido no *codebook* acima não é um Código Ótimo.

Algoritmo para construção de códigos ótimos – Códigos de Huffman

Para a construção de um código ótimo binário $\theta\{\cdot\}$ a partir do algoritmo de Huffman, efetua-se o seguinte procedimento:

Seja, inicialmente, $k = j = 0$.

(1) Organizar as probabilidades p_i de alto a baixo em uma coluna **em ordem decrescente de valor**, denominada Coluna k**.**

(2) Somar as 2 menores probabilidades p_i na Coluna k e transferi-las para a próxima coluna (à direita), denominada Coluna $k+1$**, obedecendo a ordem decrescente**. As demais probabilidades da Coluna k são transferidas inalteradas para a Coluna $k+1$ **.**

(3) Incrementar k de 1 e repetir (1) a (3) até restarem somente 2 probabilidades na Coluna $k+1$**,** então denominada Coluna j.

(4) Na Coluna j, atribuir a palavra-código representada pelo bit 0 à maior probabilidade e atribuir a palavra-código representada pelo bit 1 à menor probabilidade.

(5) Localizar na Coluna $j+1$ **,** imediatamente à esquerda da Coluna j, quais as 2 probabilidades geradoras que, ao serem somadas, resultaram na probabilidade gerada na Coluna j. Atribuir às 2 probabilidades geradoras na Coluna $j+1$ a palavra-código já atribuída à probabilidade gerada na Coluna j. Às probabilidades não-geradoras na Coluna $j+1$ são atribuídas as palavras-código já atribuídas às respectivas probabilidades não-geradas por soma na Coluna j .

(6) Na Coluna $j+1$**,** nas palavras-códigos já atribuídas em (5) às 2 probabilidades geradoras, justapor o bit 0 à palavra-código geradora de maior probabilidade e justapor o bit 1 à palavra-código geradora de menor probabilidade.

(7) Incrementar j de 1 e repetir (5) a (7) até que todas as colunas tenham palavras-código associadas às probabilidades nelas contidas.

(8) Após a execução de (7), o *codebook* p/ o Código de Huffman estará definido na coluna mais a esquerda.

Códigos ótimos – Códigos de Huffman

Exemplo 10: Seja uma fonte de informação representada pela variável aleatória discreta X com espaço de amostras definido pelo conjunto $\Omega = \{x_i\} = \{x_0, x_1 \cdots x_{M-1}\}$ de $M = 6$ eventos estatisticamente independentes x_i com probabilidades de ocorrência p_i, $i = 0,1, \ldots, M-1$, conforme tabela abaixo.

Mensagem	p_i
x_0	0.4
x_1	0.3
x_2	0.1
x_3	0.1
x_4	0.06
x_5	0.04

Pede-se:

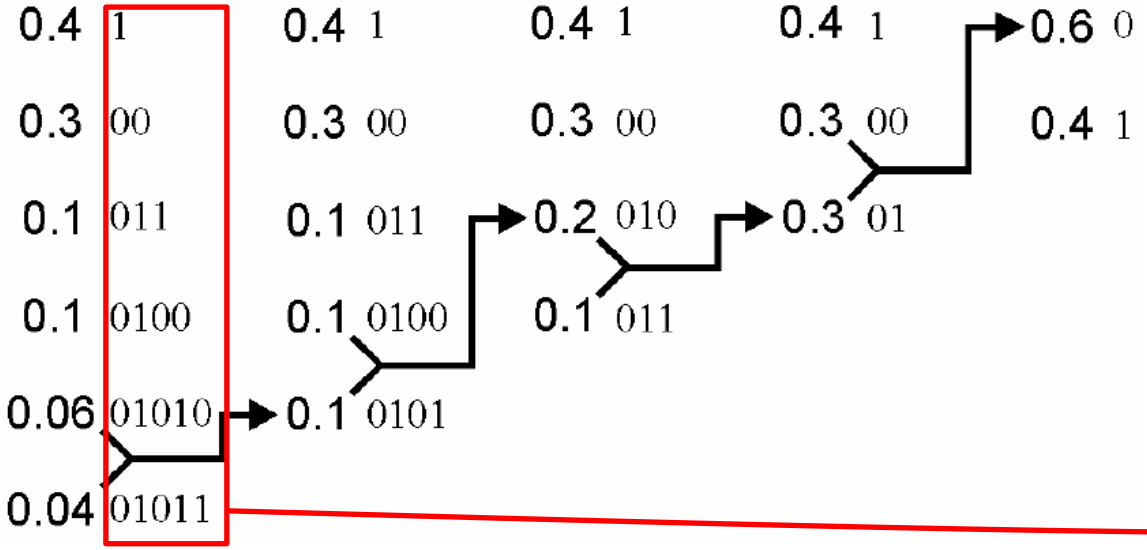
(a) Utilizando o algoritmo de Huffman descrito no slide anterior, determine um código ótimo $\theta\{\cdot\}$ cujo conjunto de palavras-código $S = \{s_i\} = \{s_0, s_1 \cdots s_{M-1}\}$ é formado a partir do alfabeto $A = \{0,1\}$.

(b) Determine a eficiência de $\theta\{\cdot\}$.

(c) Determine se $\theta\{\cdot\}$ é absolutamente ótimo ou quase absolutamente ótimo.

Solução:

(a) Utilizando o algoritmo de Huffman descrito no slide anterior:



Mensagem	p_i	s_i
x_0	0.4	1
x_1	0.3	00
x_2	0.1	011
x_3	0.1	0100
x_4	0.06	01010
x_5	0.04	01011

Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:

0.4

0.3

0.1

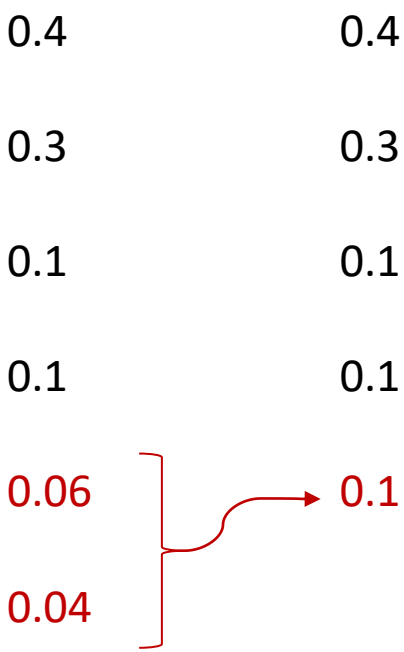
0.1

0.06

0.04

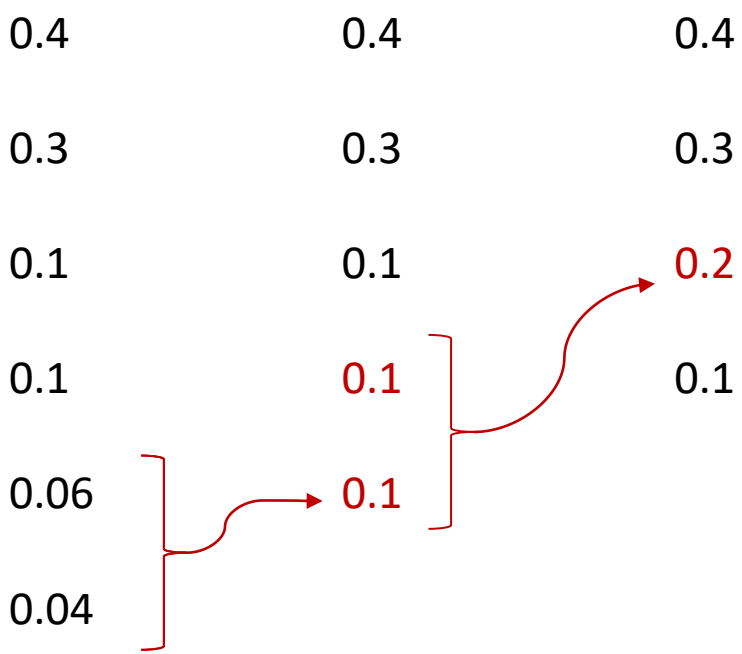
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



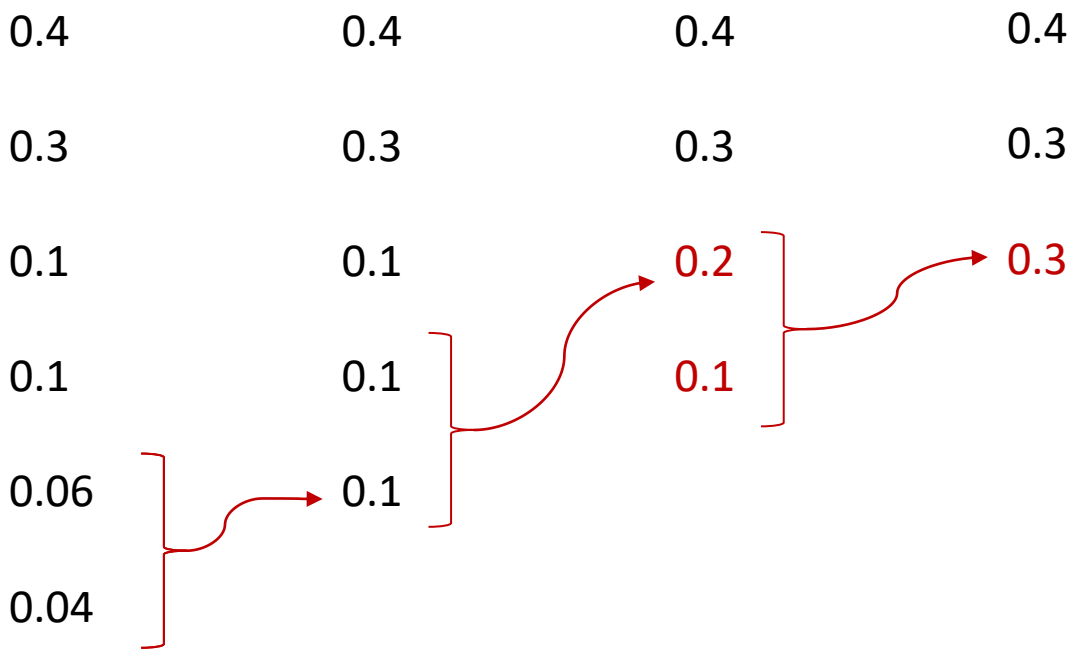
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



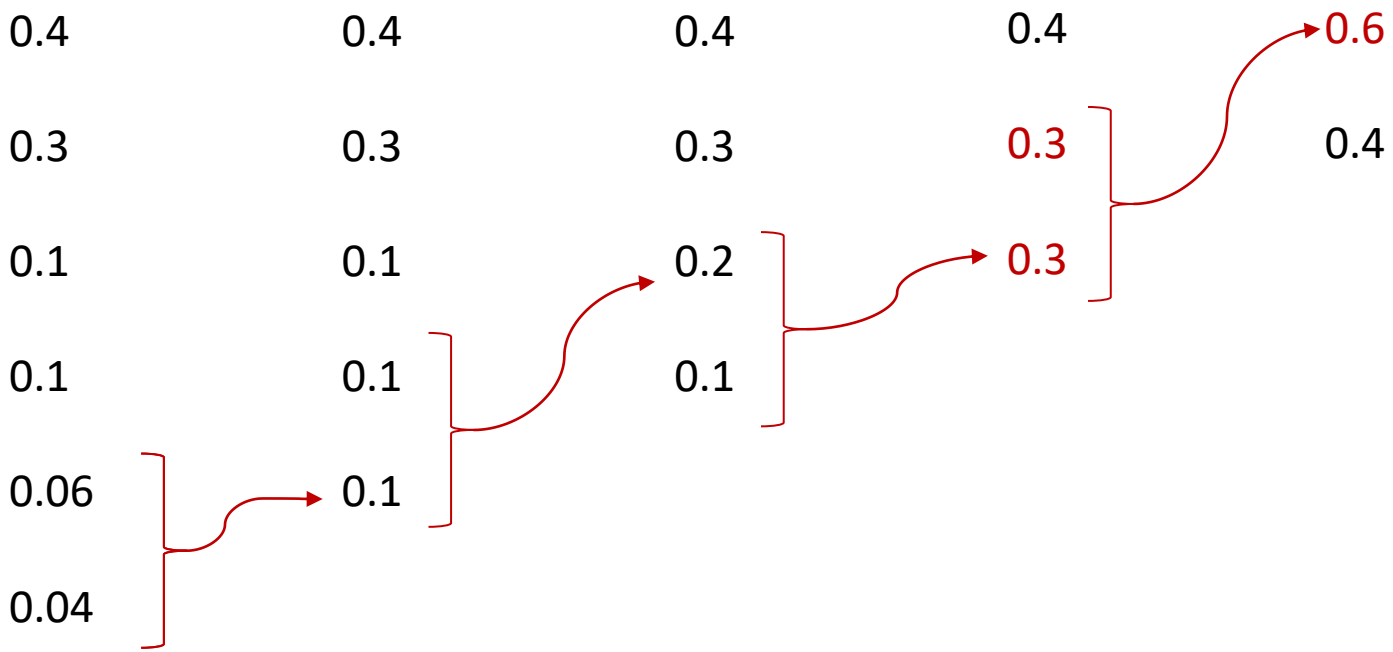
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



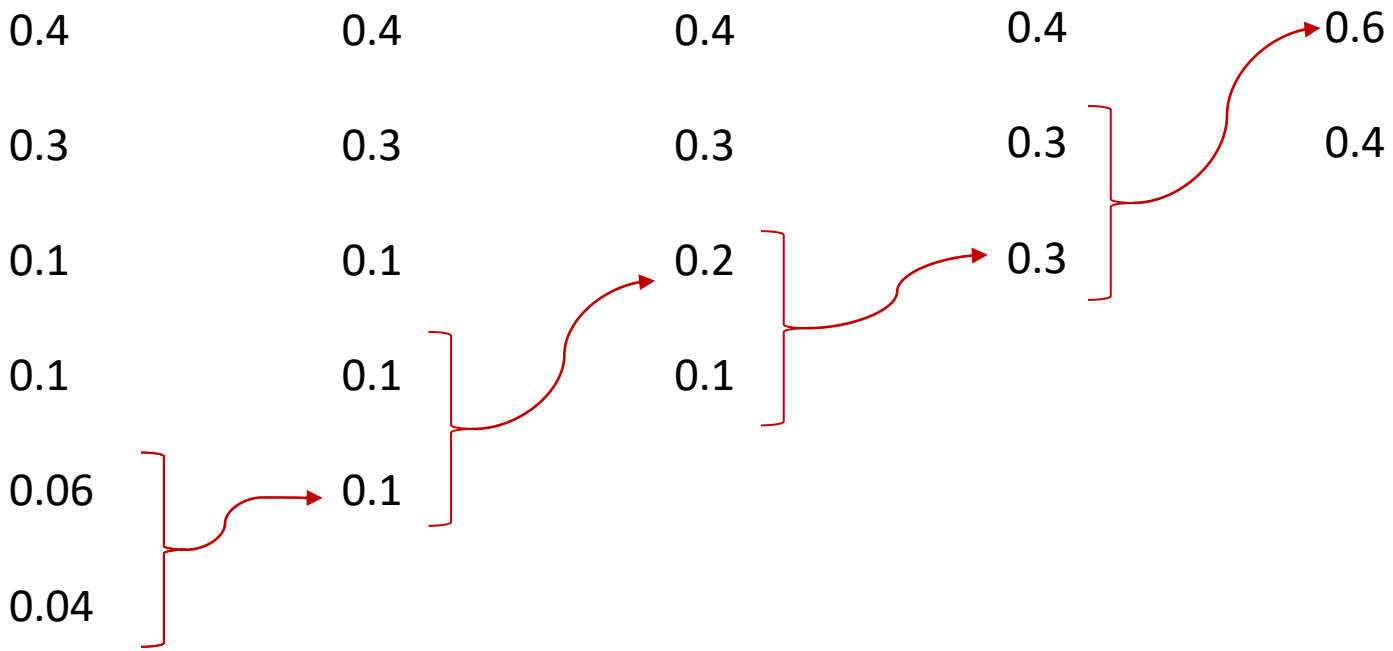
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



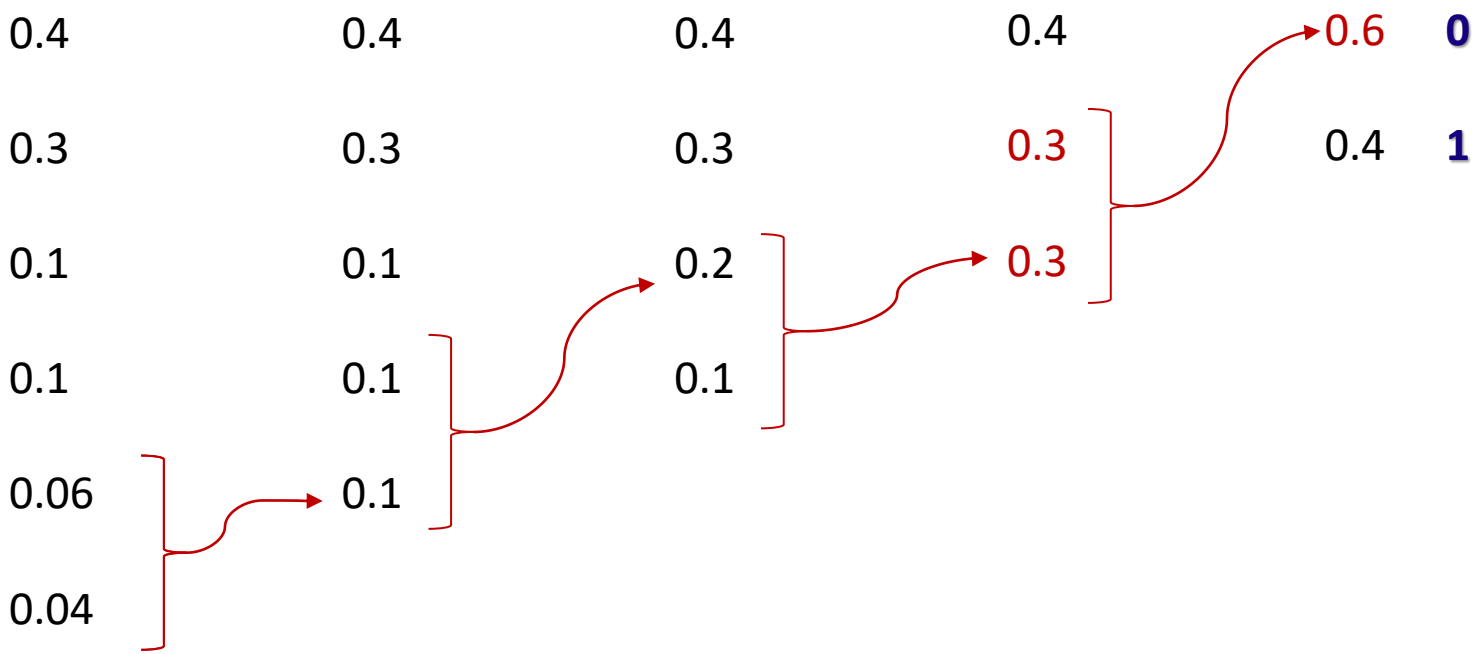
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



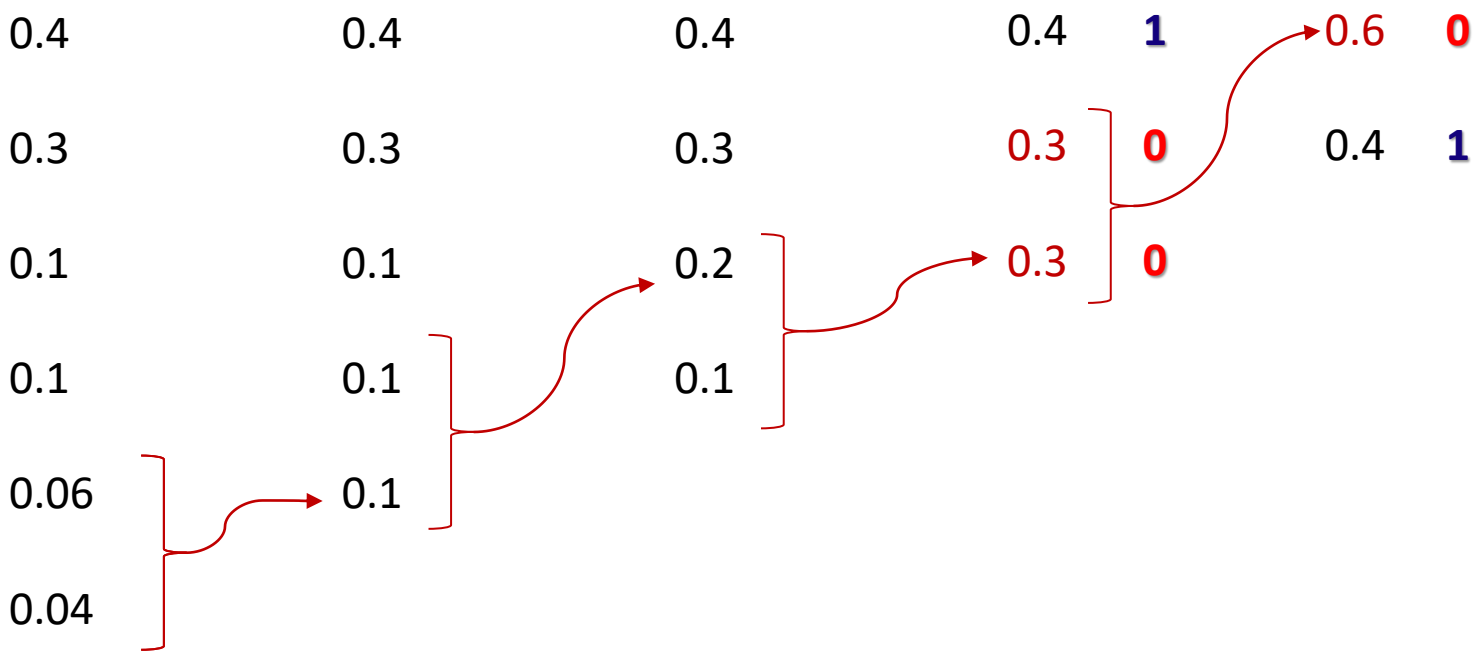
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



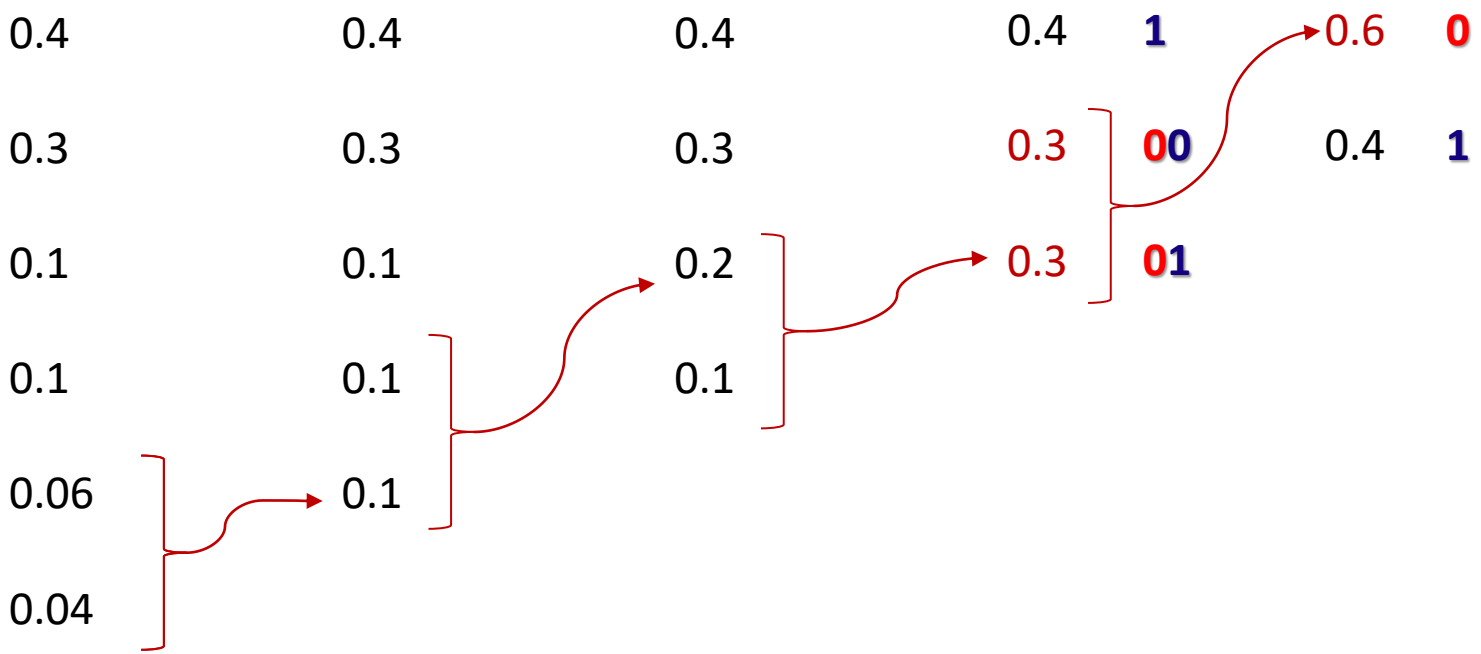
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



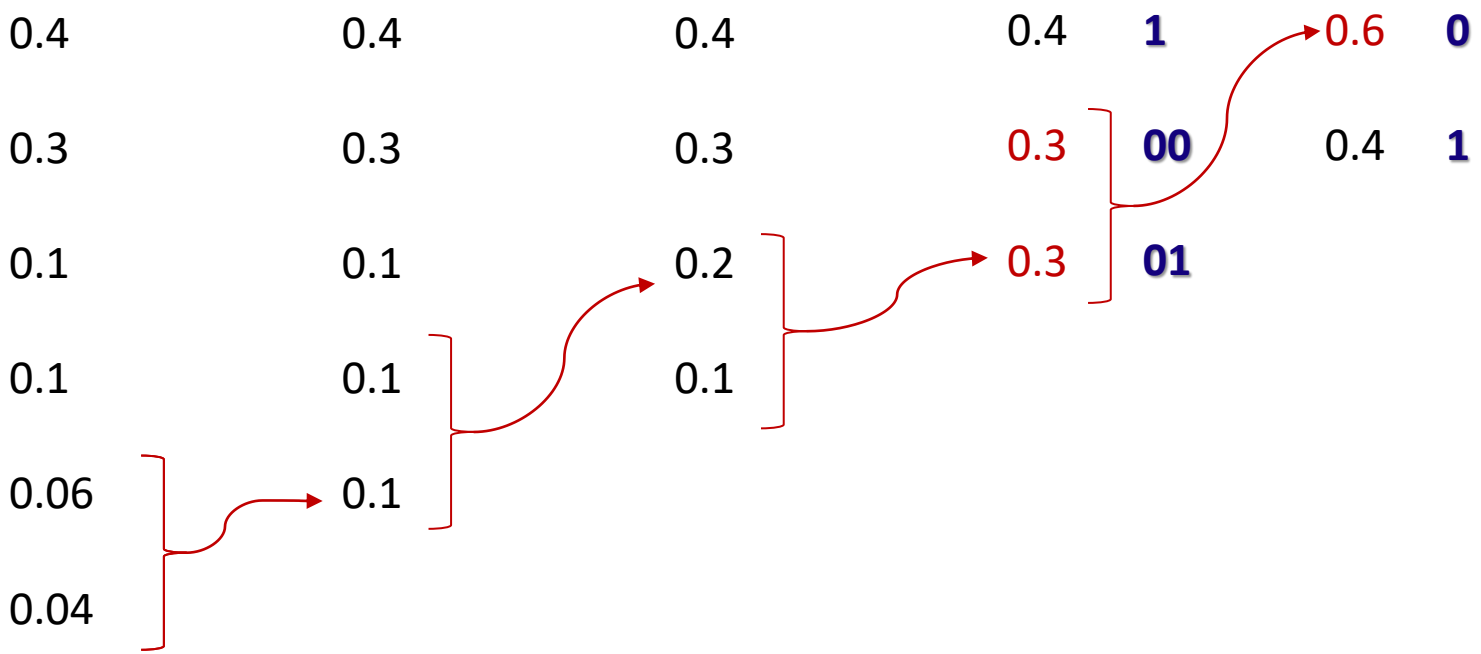
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



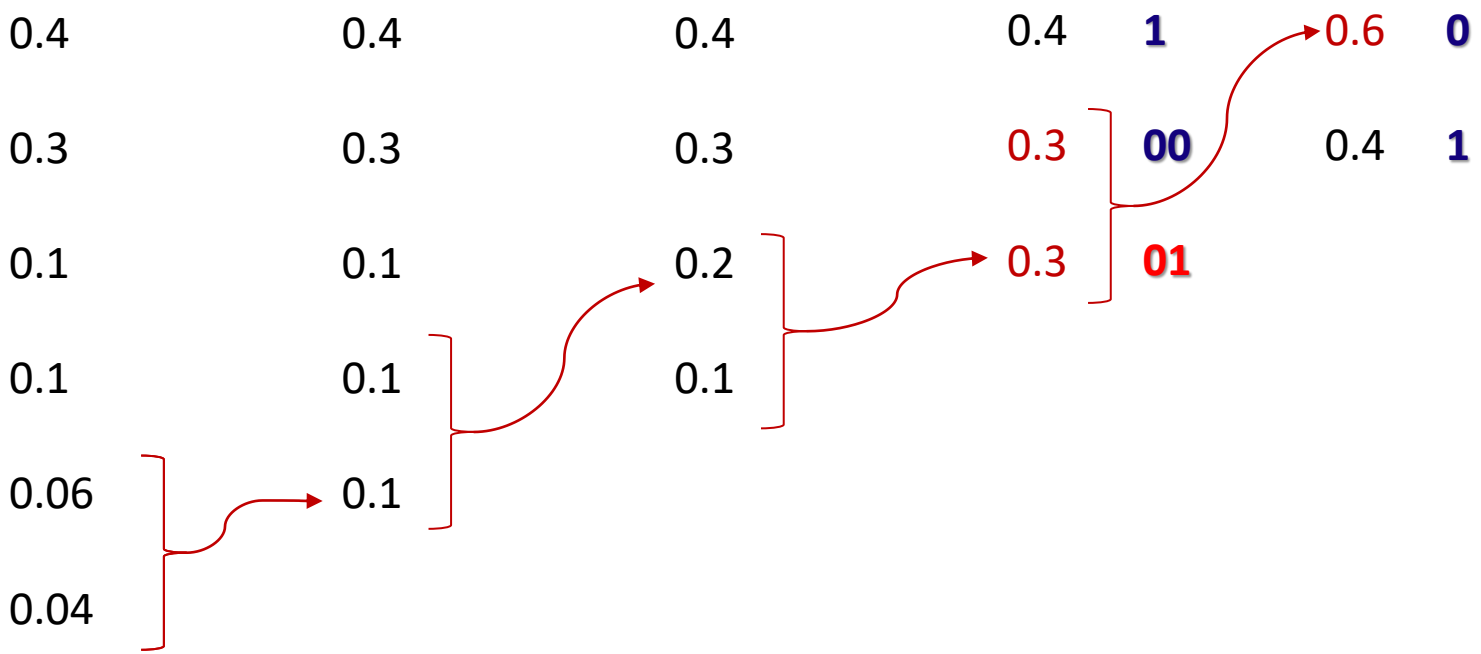
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



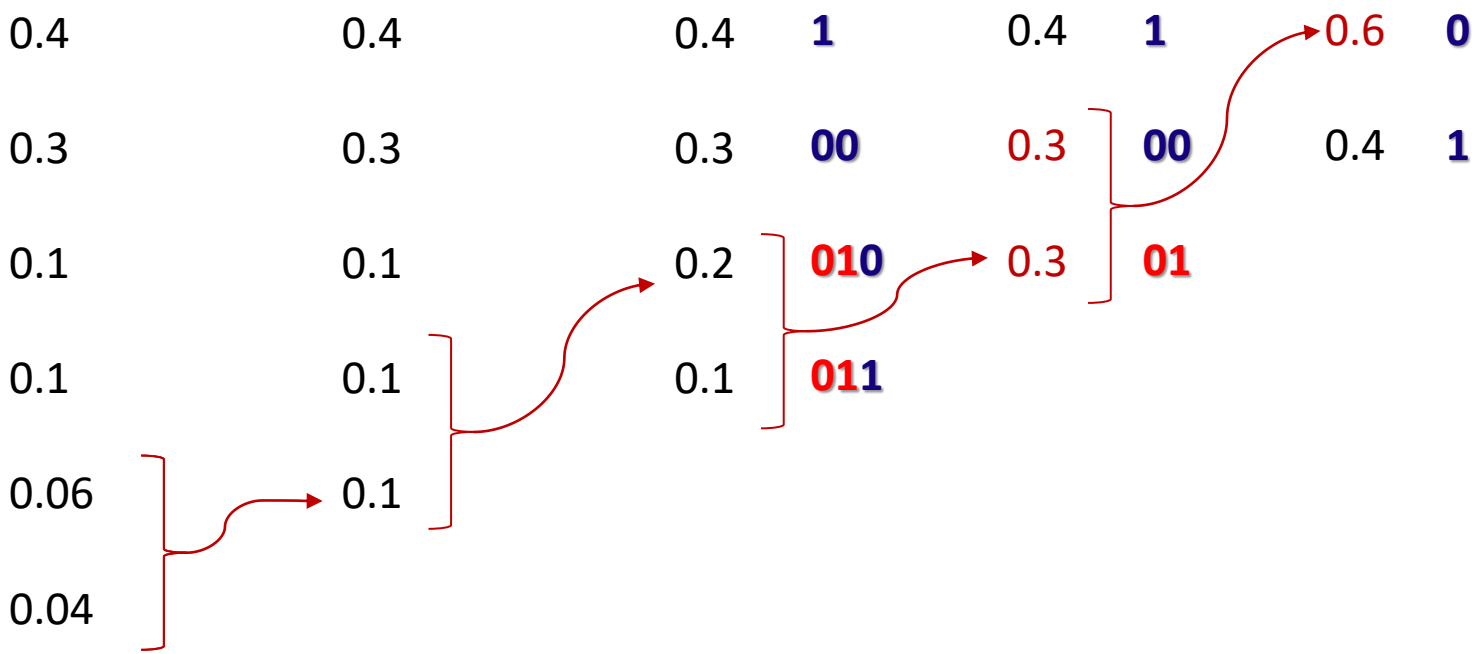
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



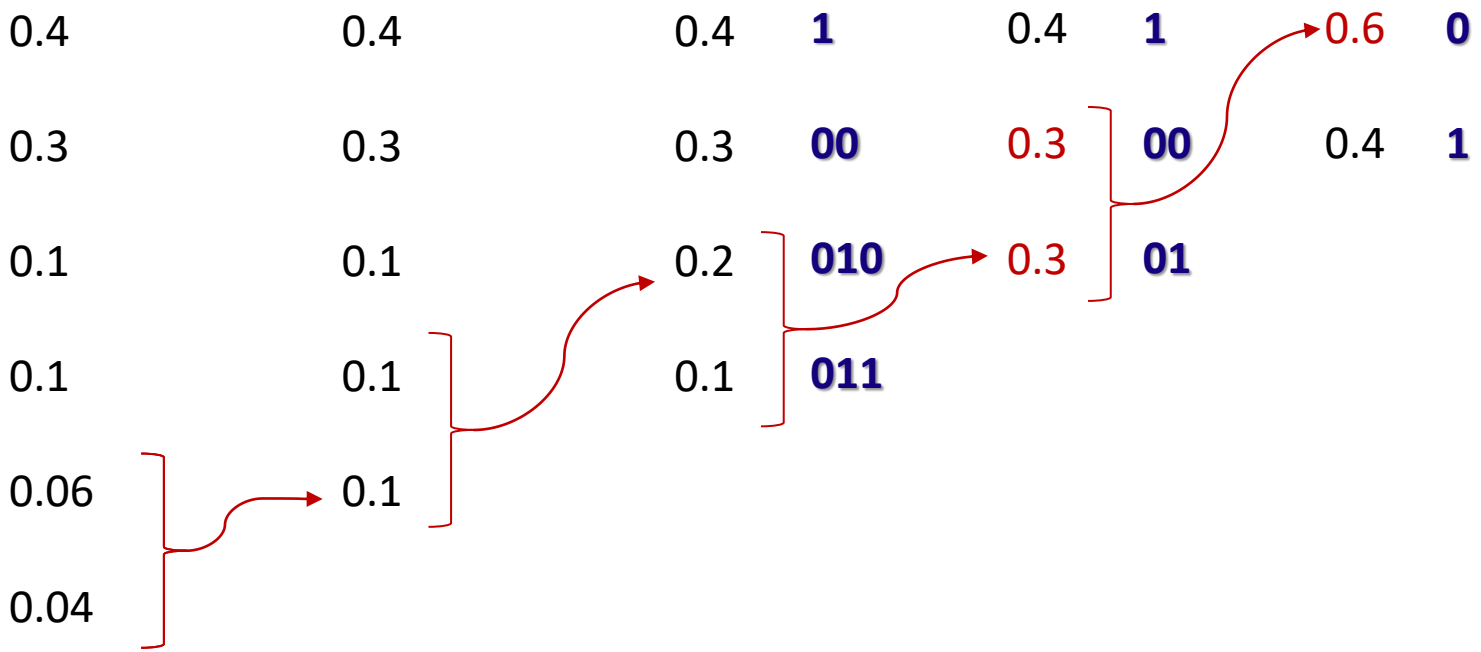
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



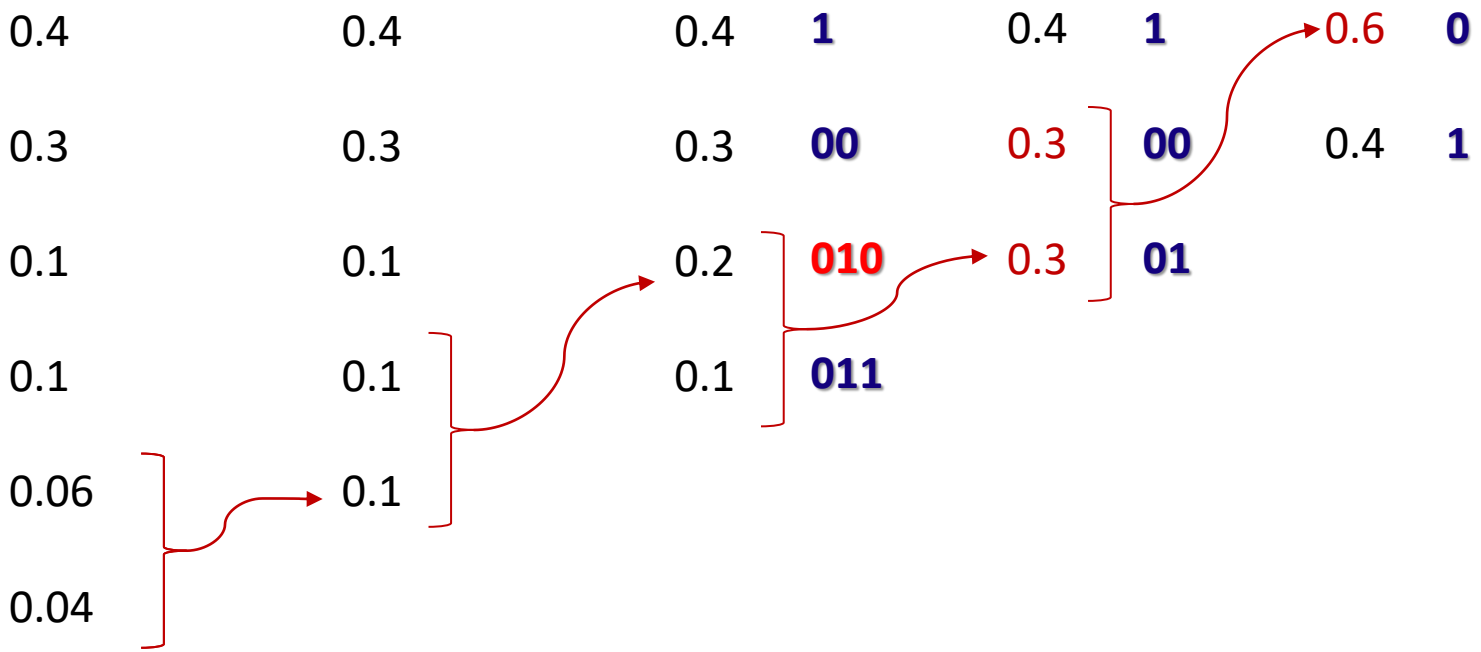
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



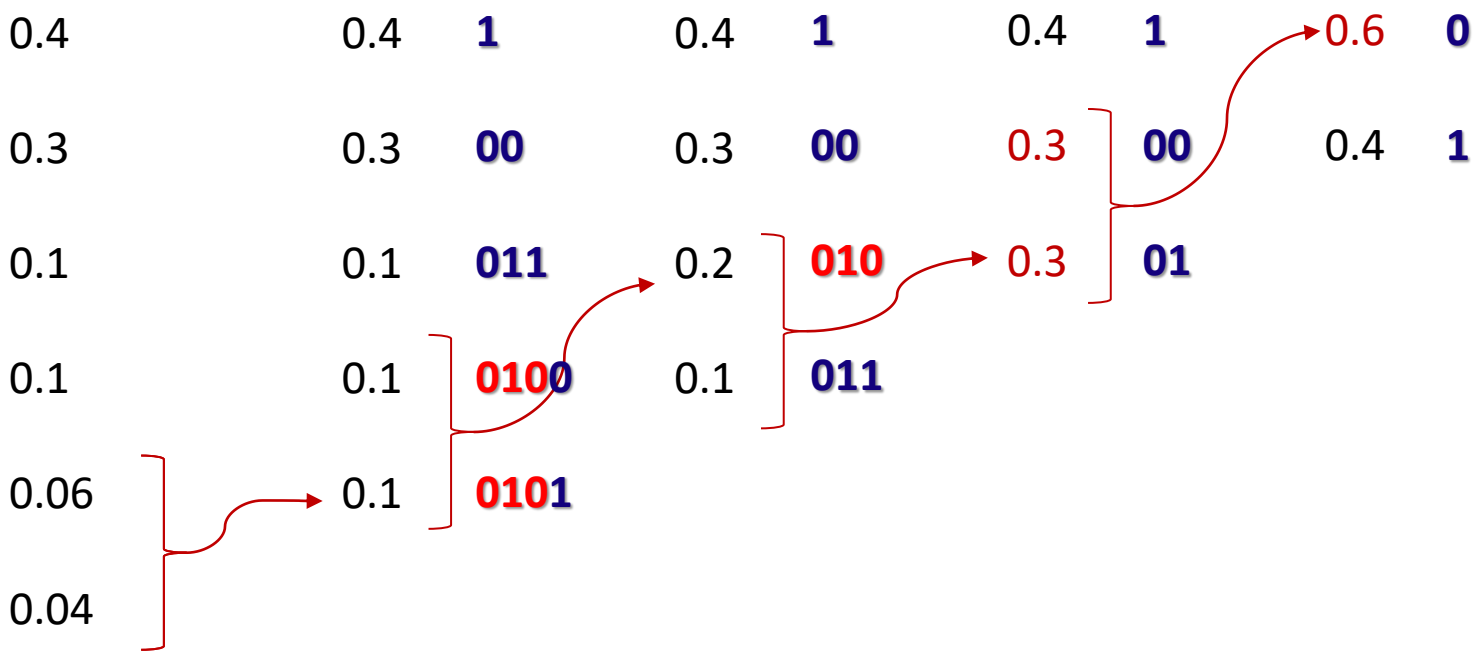
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



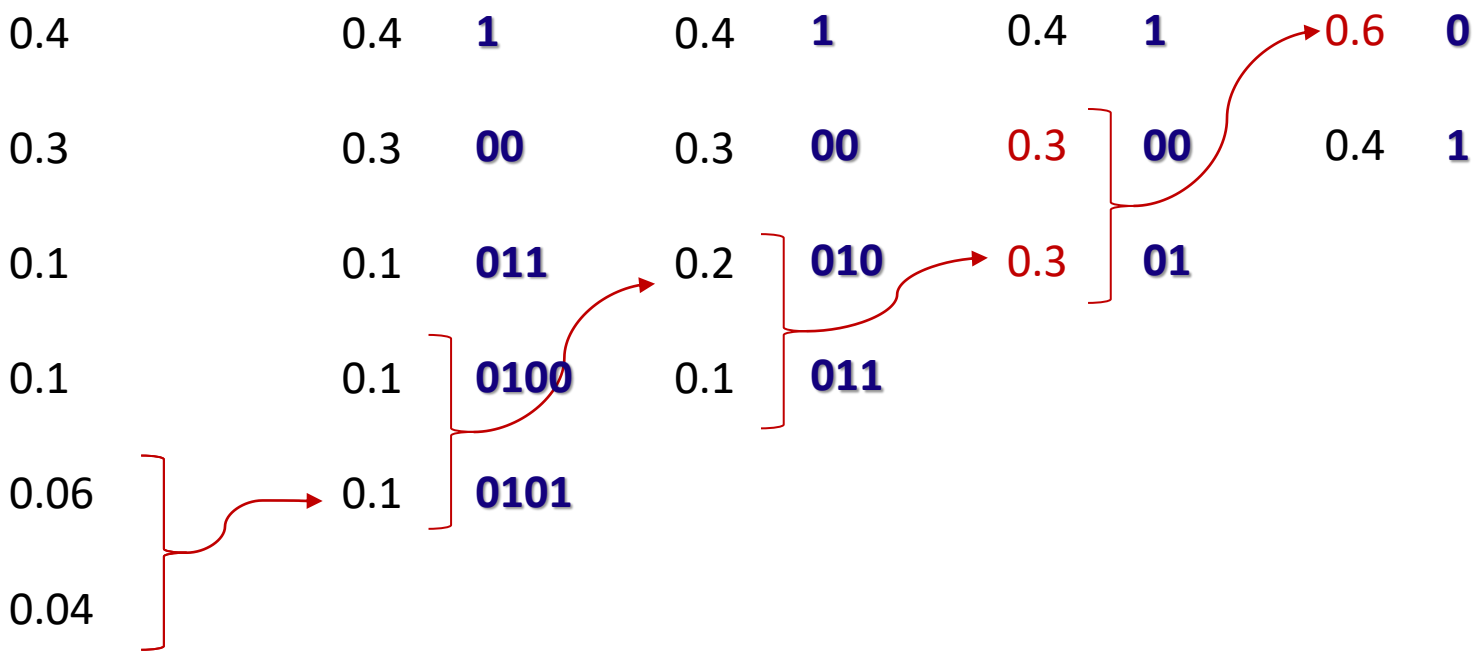
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



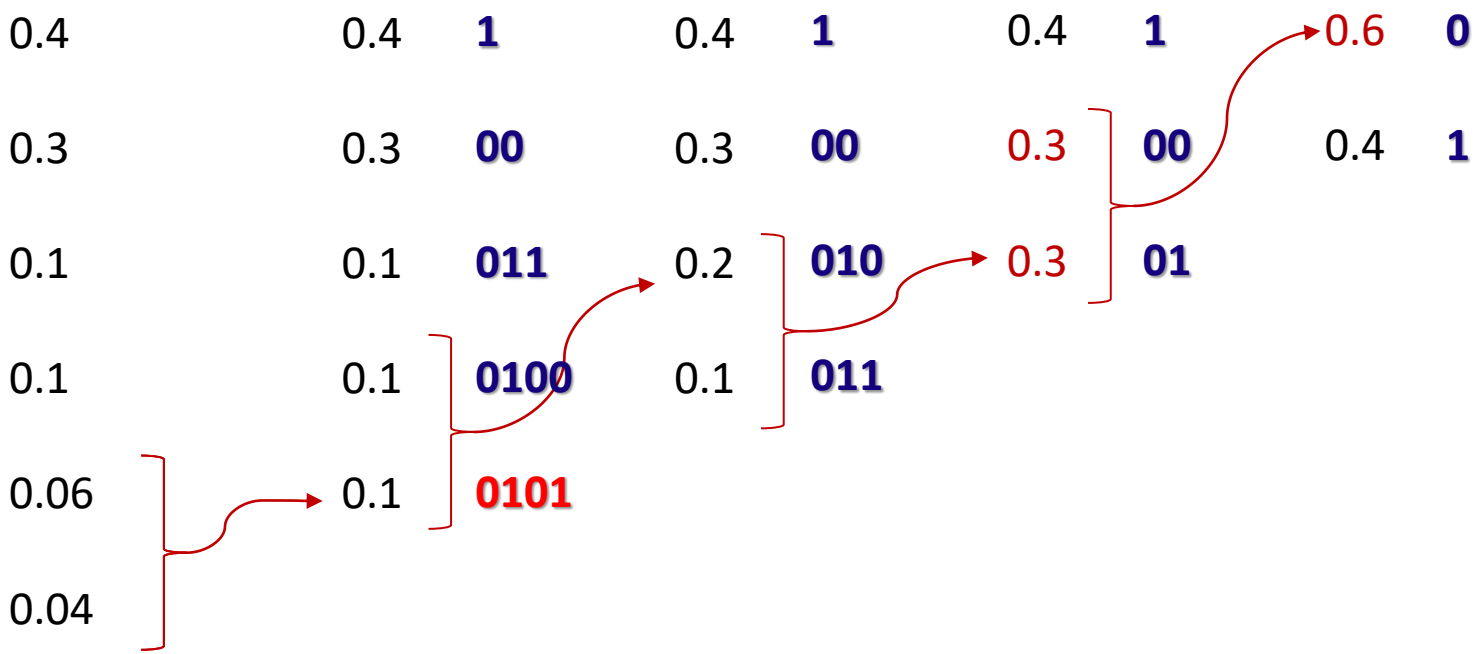
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



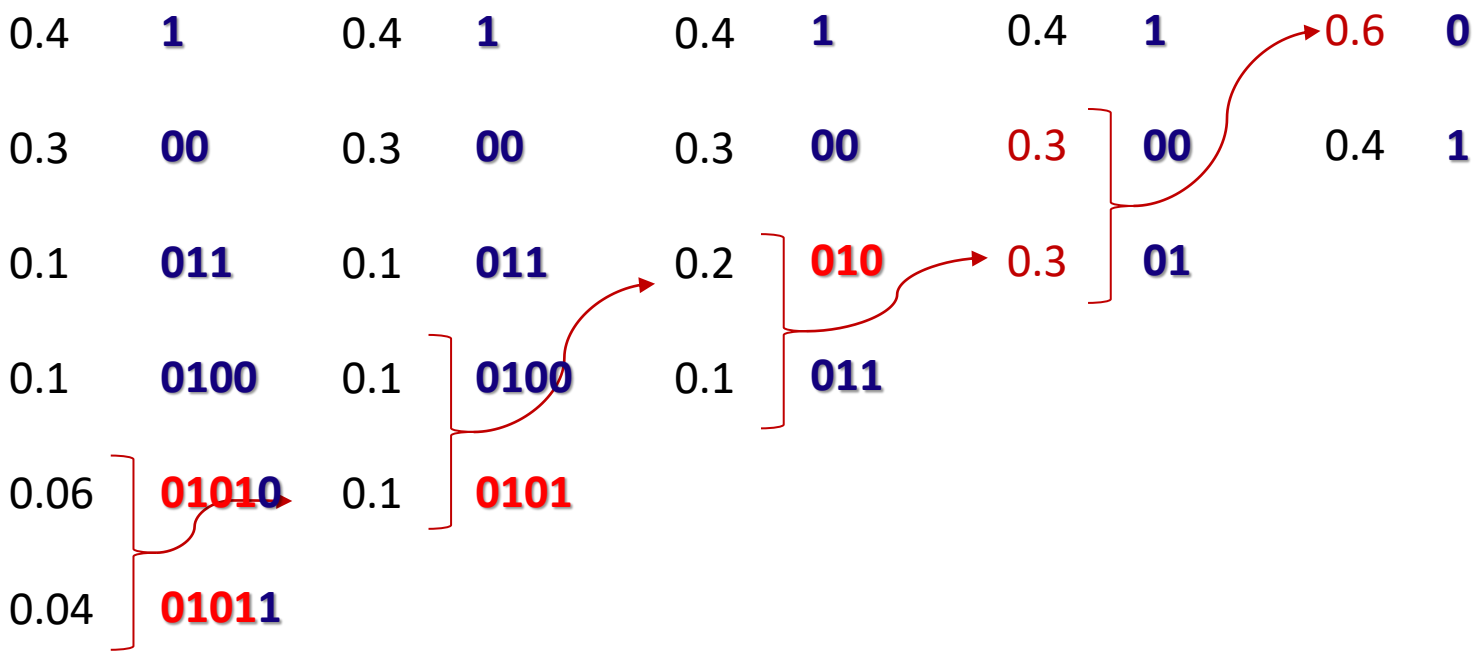
Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:



Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:

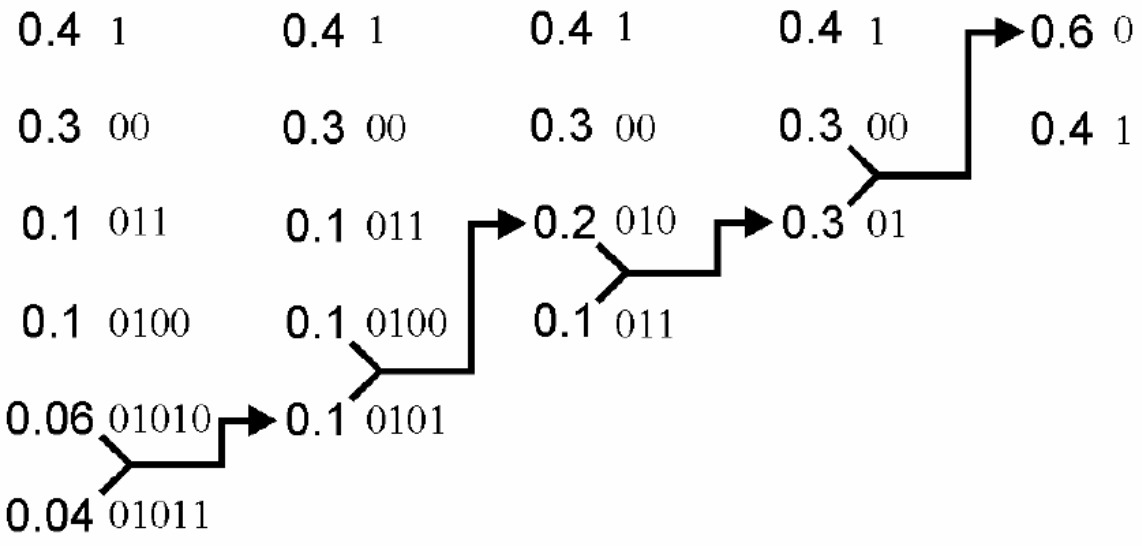


Códigos Ótimos – Códigos de Huffman

Detalhamento passo a passo do algoritmo de Huffman descrito no slide 64:

0.4	**1**	0.4	**1**	0.4	**1**	0.4	**1**	0.6	**0**
0.3	**00**	0.3	**00**	0.3	**00**	0.3	**00**	0.4	**1**
0.1	**011**	0.1	**011**	0.2	**010**	0.3	**01**		
0.1	**0100**	0.1	**0100**	0.1	**011**				
0.06	**01010**	0.1	**0101**						
0.04	**01011**								

Códigos Ótimos – Códigos de Huffman



Mensagem	p_i	s_i
x_0	0.4	1
x_1	0.3	00
x_2	0.1	011
x_3	0.1	0100
x_4	0.06	01010
x_5	0.04	01011

(b) A entropia $H(X)$ da fonte X é dada pela equação (21) do slide 39: $H(X) = -\sum_{i=0}^{M-1} p_i \log_2(p_i) = 2.14$ [bits/mensagem]

O tamanho médio $\bar{L}$ das palavras-códigos é dado pela equação (25) do slide 45: $\bar{L} = \sum_{i=0}^{M-1} p_i \ell_i = 2.20$ [bits/símbolo]

A eficiência η do código é dada pela equação (27) do slide 61: $\eta = \frac{H(X)}{\bar{L}} = \frac{2.14}{2.20} = 0.973 = 97.3\%$

(c) Visto que $H(X) \leq \bar{L} < H(X) + 1$ então $\theta\{\cdot\}$ é **quase absolutamente ótimo**.

No link Videoaula Códigos de Huffman - Profa. Cristina De Castro encontra-se disponível uma videoaula com a revisão passo a passo de códigos de Huffman.