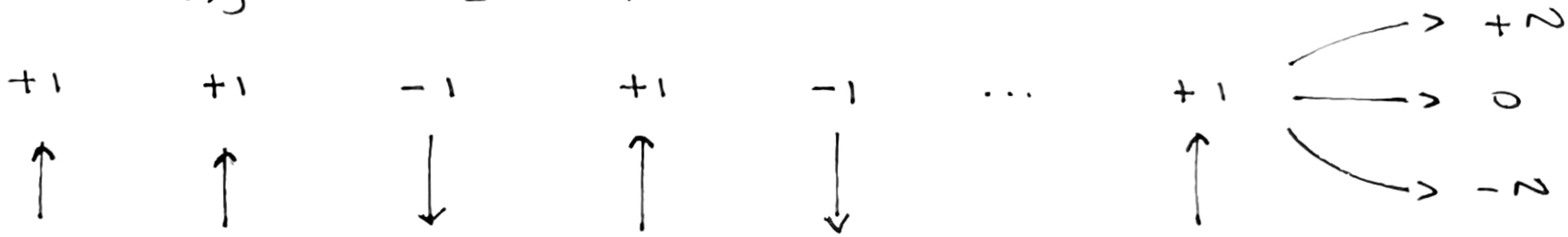


Resumindo, o algoritmo consiste em começar de um estado  $\mu$  (ou seja, um estado  $x$ ) e executar o "simulated annealing" básico (do pseudo-código

$$\text{com } S(\mu) = \sum_{i>j} \left( \lambda - \frac{a_{ij}}{2} \right) \mu_i \mu_j, \text{ onde os números } a_{ij} \text{ são dados.}$$



Método de perturbação: escolher uma posição aleatória do vetor  $\mu$  e alterá-la de " $-1$ " para " $+1$ " ou vice-versa.

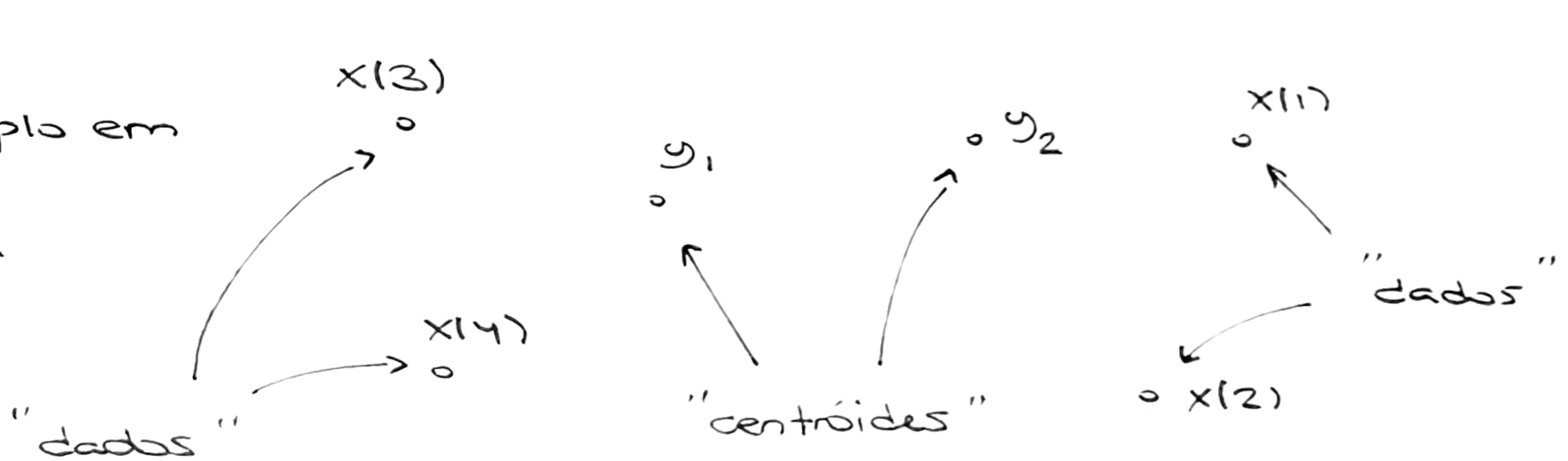
Método de resfriamento: exponencial (resulta em mínimo local, por não ser logarítmico).

Simulated Annealing — Aula 03

Continuação dos exemplos de SA básicos que vimos na aula passada...

Exemplo 4 — "clustering" ("Clusterização" ou Agrupamento)

Nós vamos usar este exemplo em  
várias partes da disciplina.



Os dados são normalmente organizados em uma "matriz de dados"  $X$ . (X)

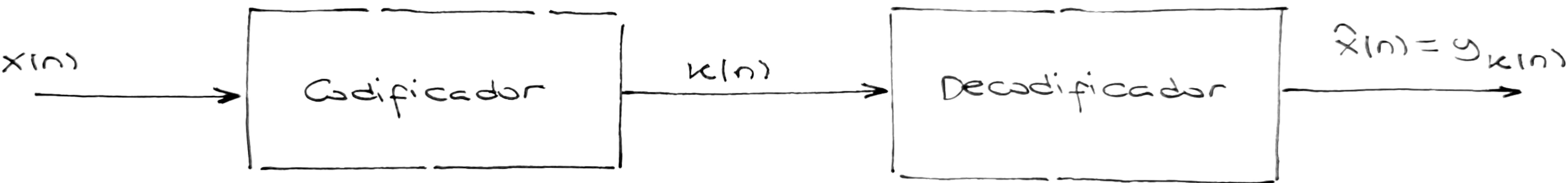
Não confunda esta matriz  $X$  com o estado  $X$  do Simulated Annealing.

$$X = \begin{bmatrix} | & | & | & | \\ x(1) & x(2) & x(3) & x(4) \\ | & | & | & | \end{bmatrix} \rightarrow \text{dados (constantes), e não "estado" (variável)}$$

$$Y = \begin{bmatrix} | & | \\ y_1 & y_2 \\ | & | \end{bmatrix} \rightarrow \text{a matriz que contém os centrífides é chamada de "dicionário", por motivos que são explicados a seguir.}$$

Considere que temos um "canal de comunicação" através do qual desejamos ...

... transmitir os vetores  $x(n)$  usando um número restrito de bits (por exemplo, digamos, 1 bit por vetor):



$$k(n) = \alpha(x(n)) = \operatorname{argmin}_i \|x(n) - y_i\|$$

Toda vez que um vetor  $x(n)$  aparece, transmitimos um bit através do canal de comunicação:

$x(n)$	$k(n)$
$x(1)$	1
$x(2)$	1
$x(3)$	0
$x(4)$	0

Considerando que, neste exemplo, o objetivo é escolher os vetores  $y$  a serem usados neste "canal de comunicação", nós vamos escolher os vetores  $y$  que minimizam o erro médio quadrático  $D$  com o qual os vetores  $x(n)$  são "reconstruídos":

$$D = \frac{1}{N} \sum_{n=1}^N \|x(n) - y_{k(n)}\|^2 \quad (D \text{ é uma função de } y)$$

MSE, de "mean squared error"  
(erro médio quadrático)

$y_1$  e  $y_2$  estão sujeitos a otimização,  
por exemplo, através de Simulated Annealing.

As  $N$  parcelas  $\|x(n) - y_{k(n)}\|^2$  também podem ser definidas a partir das distâncias Euclidianas, ao quadrado, entre os vetores  $x$  e  $y$ :

$$d(x, y) = \|x - y\|^2 \quad \text{note que este quadrado já está incluído nesta definição de distância.}$$

E então:  $D = \frac{1}{N} \sum_{n=1}^N d(x(n), y_{k(n)}) \longrightarrow D$  é uma função de  $y$

E o dicionário ótimo é dado por  $y^* = \underset{Y}{\operatorname{argmin}} D(Y)$

Podemos então executar o Simulated Annealing básico (do pseudo - código dado na Aula 2) colocando, no lugar da função custo genérica, a função custo "erro médio quadrático" da "clusterização":

$$J(Y) = \frac{1}{N} \sum_n d(x_{1n}, y_{kn})$$

Começamos com um estado  $y_0$  aleatório (note que, neste exemplo, reservamos o símbolo  $x$  para os dados do problema) e perturhamos os sucessivos estados " $y_{\text{actual}}$ " aleatoriamente, conforme o Algoritmo de Metropolis. À medida em que a temperatura é reduzida, são obtidas ...

... soluções sucessivamente melhores, escapando de mínimos locais. <sup>(\*)</sup>

Ideialmente — com convergência em probabilidade para a p.d.f. final e com temperatura  $T$  tendendo lentamente a zero — o algoritmo converge para o ponto mínimo global ( $y^*$ ) de  $J(y)$ .

(\*) Este é um problema de otimização não-convexa, que tem ótimos locais.

Uma forma simples de demonstrar isso é inicializar o estado  $y_0$  do otimizador em dois pontos diferentes, com  $T$  baixo, e notar que, para cada inicialização, o otimizador converge para um resultado diferente.

Simulated Annealing — Análise da Convergência — Geman, 1984

Cadeia de Markov ("Markov Chain")

$f_{xyz}(x, y, z) \rightarrow$  p.d.f. conjunta de três variáveis aleatórias.

Se  $f_{z|xy}(z|x, y) = f_{z|x}(z|x)$ , então:

$x \rightarrow y \rightarrow z$

(as variáveis aleatórias  $x, y$  e  $z$  compõem uma Cadeia de Markov)

Quando a sequência das variáveis aleatórias é temporal, temos um

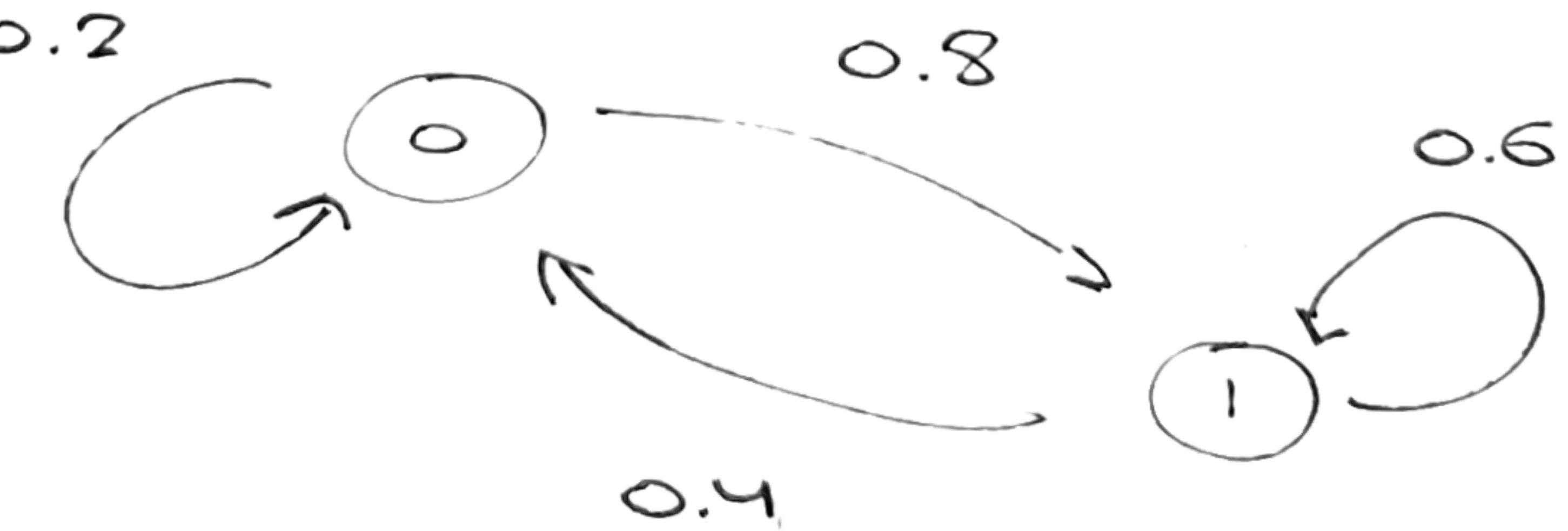
Processo de Markov:

$$x(0) \longrightarrow x(1) \longrightarrow \dots \longrightarrow x(n-1) \longrightarrow x(n)$$

$$f_{X_n | X_{n-1}, \dots, X_0}(x_n | x_{n-1}, \dots, x_0) = f_{X_n | X_{n-1}}(x_n | x_{n-1})$$

Origens

Exemplo:



Destinos

$$\begin{bmatrix} 0 & 1 \\ 0 & 0.2 & 0.4 \\ 1 & 0.8 & 0.6 \end{bmatrix} = \Sigma$$

$$M = \begin{bmatrix} 0.2 & 0.4 \\ 0.8 & 0.6 \end{bmatrix} = \begin{bmatrix} p(x_n=0 | x_{n-1}=0) & p(x_n=0 | x_{n-1}=1) \\ p(x_n=1 | x_{n-1}=0) & p(x_n=1 | x_{n-1}=1) \end{bmatrix}$$

$$M = \begin{bmatrix} p(0|0) & p(0|1) \\ p(1|0) & p(1|1) \end{bmatrix} \quad (\text{abreviando } p_{x_n|x_{n-1}}) \quad (\rightarrow p_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix})$$

Considere que "sempre começamos" do estado  $x_0 = 1$ :

$$\left\{ \begin{array}{l} p(x_0=0) = p(0) = 1 \\ p(x_0=1) = p(1) = 0 \end{array} \right.$$

$$(P(A \cap B) = P(B|A) \cdot P(A) \text{ e } P(A \cup B) = P(A) + P(B) - P(A \cap B))$$

$$p(x_1=0) = p(0|0)p(0) + p(0|1)p(1) = 0.2 \times 1 + 0.4 \times 0 = 0.2$$

$$p(x_1=1) = p(1|0)p(0) + p(1|1)p(1) = 0.8 \times 1 + 0.6 \times 0 = 0.8$$

$$p_1 = M p_0$$

Temos, então,  $p_1 = Mp_0$ ,  $p_2 = M^2 p_0 = M^2 p_0, \dots \Rightarrow$

$$\boxed{p_n = M p_{n-1}}$$

Para saber o que acontece quando  $n \rightarrow \infty$ , devemos olhar os autovalores

e autovetores de  $M$ . Mais especificamente, devemos calcular o autovetor " $\pi$ "

para o qual o autovalor é 1.  $\rightarrow \boxed{M\pi = \pi}$  ("autovetor invariante de  $M$ ")

$$|M - \lambda I| = 0 \Rightarrow \begin{vmatrix} 0.2 - \lambda & 0.4 \\ 0.8 & 0.6 - \lambda \end{vmatrix} = 0.12 - 0.8\lambda + \lambda^2 - 0.32 = 0$$

$$\lambda^2 - 0.8\lambda - 0.2 = 0 \rightarrow \lambda = \frac{0.8 \pm \sqrt{0.64 + 0.8}}{2} \rightarrow \frac{0.8 + 1.2}{2} = 1 \quad \checkmark$$

este autovetor  
não é de nosso  
interesse

$$M\pi = \pi \implies (M - I)\pi = 0 \implies \begin{bmatrix} -0.8 & 0.4 \\ 0.8 & -0.4 \end{bmatrix} \begin{bmatrix} \pi_1 \\ \pi_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Arbitrando  $\pi_1 = 1$ , encontramos então  $\pi_2 = 2$ . Neste caso,  $\pi_1 + \pi_2 = "2" = 3$ .

Normalizando  $\pi_1$  e  $\pi_2$  de modo que  $\pi_1 + \pi_2 = 1$  (correspondendo a probabilidades), temos  $\pi = \begin{bmatrix} 1/3 \\ 2/3 \end{bmatrix}$ .

Leitura do artigo German, 1984: Seções IX, X, XI, XII e Apêndice