Aula 02 - Parte 02 - Sistemas Lineares e Matrizes

Álgebra Linear e Teoria da Informação

Prof. Tiago Tavares

Um sistema linear

Podemos descrever como populações de presas (sapos, s) e predadores (carcarás, c) mudam ao longo do tempo com um sistema de equações. A população em um mês (t) depende diretamente da população no mês anterior ($t-1$).

$$\begin{cases} c_t = 0.8c_{t-1} + 0.2s_{t-1} \\ s_t = -0.1c_{t-1} + 1.1s_{t-1} \end{cases}$$

Este é um **sistema linear**.

Sistemas lineares: forma matricial

O mesmo sistema de equações pode ser escrito de forma muito mais compacta usando matrizes. Agrupamos as populações em um **vetor de variáveis** e as regras de interação em uma **matriz de coeficientes**:

$$\underbrace{\begin{bmatrix} c_t \\ s_t \end{bmatrix}}_{\text{Variá veis em } t} = \underbrace{\begin{bmatrix} 0.8 & 0.2 \\ -0.1 & 1.1 \end{bmatrix}}_{\text{Matriz de Coeficientes (A)}} \underbrace{\begin{bmatrix} c_{t-1} \\ s_{t-1} \end{bmatrix}}_{\text{Variá veis em } t-1}$$

A equação inteira se resume a: $\boldsymbol{X_t = AX_{t-1}}$

Exercício: a forma matricial

Escreva o sistema abaixo na forma matricial:

$$\begin{cases} x + 3y + z = 3 \\ y - x = 0 \\ z + 2y = 1 \end{cases}$$

A Evolução do Sistema

Com a formulação matricial, simular a evolução do sistema ao longo do tempo se torna uma série de multiplicações matriciais. A matriz `A` encapsula todas as regras de como o sistema passa de um estado para o próximo.

```
# O estado inicial (populações no mês 0)
estado_anterior = np.array( [[c0], [s0]] )

# A matriz que define as regras do ecossistema
A = np.array([ [0.8, 0.2], [-.1, 1.1]])

# Dentro de um laço, para cada mês:
estado_atual = A @ estado_anterior
```

Mudando a Perspectiva: Transposição

A transposição A^T é uma operação que transforma as linhas de uma matriz em colunas e vice-versa.

$$X = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \rightarrow X^T = \begin{bmatrix} a & d \\ b & e \\ c & f \end{bmatrix}$$

Ela nos permite inverter a ordem da multiplicação matricial, uma propriedade muito útil:

$$(\boldsymbol{AX})^T = \boldsymbol{X}^T \boldsymbol{A}^T$$

Isso nos dá flexibilidade para organizar nossos vetores de estado como linhas ou colunas.

O Algoritmo PageRank

O PageRank, algoritmo que deu origem ao Google, modela a relevância de uma página web pela probabilidade de um "surfista aleatório" estar nela.

- O surfista começa em uma página qualquer.
- Ele clica em um link aleatório para ir à próxima página.
- Páginas que recebem mais links (especialmente de páginas importantes) serão visitadas com mais frequência.

A *relevância* de uma página é a probabilidade de, numa iteração aleatória, o surfista aleatório estar nessa página.

Pagerank: variáveis formam um Vetor de Probabilidade

Vamos representar a localização do surfista como um **vetor de estado probabilístico**. Cada elemento do vetor representa a probabilidade de o surfista estar naquela página em um determinado momento.

Se temos 4 páginas e o surfista começa na página 0, nosso vetor de estado inicial é:

$$x_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} P(e_t = s_0) \\ P(e_t = s_1) \\ P(e_t = s_2) \\ P(e_t = s_3) \end{bmatrix} \quad \begin{matrix} \text{(100\% de chance de estar na pá gina 0)} \\ \text{(0\% de chance de estar na pá gina 1)} \\ \text{(0\% de chance de estar na pá gina 2)} \\ \text{(0\% de chance de estar na pá gina 3)} \end{matrix}$$

A Matriz de Transição do PageRank

Assim como no modelo presa-predador, podemos criar uma matriz de coeficientes $\boldsymbol{A}$ que descreve como as probabilidades mudam a cada "clique". O elemento $\boldsymbol{a_{i,j}}$ é a **probabilidade de ir para a página i vindo da página j**, isto é, $P(e_{t+1} = s_i | e_t = s_j)$.

$$A = \begin{bmatrix} 0. & 0.5 & 0. & 0. \\ 0.33 & 0. & 0.5 & 0. \\ 0.33 & 0. & 0. & 1. \\ 0.34 & 0.5 & 0.5 & 0. \end{bmatrix}$$

As colunas sempre somam 1, pois representam a distribuição de probabilidade dos links de uma página.

Exercício: nesta matriz, quais páginas linkam para quais outras páginas?

PageRank: demonstração

Sabemos que nosso surfista está sempre em uma única página a cada clique. Se *sabemos* que ele está na página e_k, então a probabilidade de estar na página e_i será $P(e_{t+1} = s_i | e_t = s_k)$, isto é, $a_{i,k}$. Porém, sabemos a probabilidade $P(e_t = s_j)$ para cada página j através do vetor x_t. Sabendo que "estar em uma página" é um evento disjunto (porque só podemos estar em uma página), ficamos com:

$$P(e_{t+1} = s_i) = P(e_{t+1} = x_i | e_t = x_0)P(e_t = x_0) + P(e_{t+1} = x_i | e_t = x_1)P(e_t = x_1) + \cdots + P(e_{t+1} = x_i | e_t = x_J)P(e_t = x_J)$$

Veja como podemos re-escrever essa equação como:

$$P(e_{t+1} = s_i) = \sum_{k=0}^{K-1} P(e_{t+1} = x_i | e_t = x_k)P(e_t = x_k)$$

Simplificando

Partindo da equação:

$$P(e_{t+1} = s_i) = \sum_{k=0}^{K-1} P(e_{t+1} = x_i | e_t = x_k) P(e_t = x_k)$$

Podemos substituir as probabilidades por suas representações matriciais:

$$x_{t+1,i} = \sum_{k=0}^{K-1} a_{i,k} x_{t,k}$$

E isso nos leva imediatamente à forma de multiplicações matriciais:

$$x_{t+1} = A x_t$$

Resumo: PageRank

A evolução das probabilidades do surfista segue a mesma regra do nosso ecossistema: o novo estado é a matriz de transição multiplicada pelo estado anterior:

$$\boldsymbol{x_t} = \boldsymbol{A}\boldsymbol{x_{t-1}}$$

- $\boldsymbol{x_{t-1}}$: vetor com as probabilidades de estar em cada página.
- $\boldsymbol{A}$: matriz com as probabilidades de transição entre páginas.
- $\boldsymbol{x_t}$: o novo vetor de probabilidades após um clique.

Repetindo essa multiplicação várias vezes, o vetor $\boldsymbol{x}$ converge para as probabilidades de longo prazo: o PageRank.

Exercício

Considere a matriz de transições abaixo:

$$A = \begin{bmatrix} 0.25 & 0.5 & 0. & 0. \\ 0.25 & 0. & 0.5 & 0. \\ 0.25 & 0. & 0. & 1. \\ 0.25 & 0.5 & 0.5 & 0. \end{bmatrix}$$

Sabendo que o surfista iniciou sua jornada em $t = 0$ no estado 0, calcule a probabilidade de, após dois cliques, ele estar no estado 1.