Revisão e Notas sobre Matrizes

Pedro Rupf Pereira Viana

5 de dezembro de 2025

1 Introdução e Objetivos

Trata-se de uma revisão teórica e expositiva sobre matrizes, suas definições fundamentais, e suas aplicações na matemática, física e na programação.

2 Desenvolvimento

2.1 Primeiros passos

Chamamos de matriz A, $m \times n$ (isto é, m por n) uma tabela de mn elementos em m linhas e n colunas. Por exemplo, ao recolhermos os dados referentes a altura, peso e idade de um grupo de 4 pessoas, podemos dispô-los na tabela:

	Altura (m)	**Peso (kg)**	**Idade (anos)**
Pessoa 1	1,70	70	23
Pessoa 2	1,75	60	45
Pessoa 3	1,60	52	25
Pessoa 4	1,81	72	30

Tabela 1: Tabela de dados de altura, peso e idade de 4 pessoas genéricas

Ao abstrairmos os significados das linhas e colunas, obtemos a matriz A abaixo:

$$A = \begin{bmatrix} 1,70 & 70 & 23 \\ 1,75 & 60 & 45 \\ 1,60 & 52 & 25 \\ 1,81 & 72 & 30 \end{bmatrix} \tag{2.1}$$

Observe que em um problema o número de variáveis e de observações pode ser muito grande, essa disposição ordenada dos dados em forma de matriz facilita a visualização e o manuseio dos mesmos. Os elementos de uma matriz podem ser números (reais ou complexos), funções, ou até mesmo outras matrizes. Representaremos uma matriz A, de ordem m x n, por:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} = [a_{ij}]_{m\times n} \tag{2.2}$$

Usaremos sempre letras maiúsculas para representar matrizes, e quando quisermos especificar a ordem de uma matriz $\mathbf{A}$ (isto é, o número de linhas e colunas), usaremos a notação $\mathbf{A}_{m\times n}$. O elemento que se encontra na i-ésima linha e j-ésima coluna da matriz $\mathbf{A}$ será representado por a_{ij}.

2.2 Tipos especiaias de matrizes

2.2.1 Matriz quadrada

Matriz cujo número de linhas é igual ao número de colunas ($m = n$). Exemplo:

$$A = \begin{bmatrix} 2 & 4 & 6 \\ 1 & 3 & 5 \\ 7 & 8 & 9 \end{bmatrix} \tag{2.3}$$

No caso de matrizes $\mathbf{A}_{m \times m}$, dizemos que a matriz é de ordem m.

2.2.2 Matriz nula

Matrizes cujos elementos são todos iguais a zero; isto é, $a_{ij} = 0$, pata todo i e j. Exemplo:

$$\mathbf{A}_{2\times 2} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} = 0 \tag{2.4}$$

2.2.3 Matriz diagnonal

São matrizes quadradas cujos elementos fora da diagonal principal são todos iguais a zero; isto é, $a_{ij} = 0$, para todo $i \neq j$. Exemplo:

$$\mathbf{A}_{3\times 3} = \begin{bmatrix} 5 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 9 \end{bmatrix} \tag{2.5}$$

2.3 Operações com matrizes

Ao utilizar matrizes, surge naturalmente a necessidade de realizar certas operações matemáticas com elas. As operações mais comuns são a adição, subtração e multiplicação de matrizes. Por exemplo, consideremos as tabelas abaixo, que descrevem a produção de grãos em dois anos consecutivos:

	Soja	**Feijão**	**Arroz**	**Milho**
Região A	3000	200	400	600
Região B	700	350	700	100
Região C	1000	100	500	800

Tabela 2: Produção de grãos (em milhares de toneladas) - Ano 1

	Soja	**Feijão**	**Arroz**	**Milho**
Região A	5000	50	200	0
Região B	2000	100	300	300
Região C	2000	100	600	600

Tabela 3: Produção de grãos (em milhares de toneladas) - Ano 2

Se quisermos montar uma tabela que dê a produção por produto e por região nos dois anos conjuntamente, podemos somar as duas matrizes correspondentes das tabelas acima:

$$\begin{bmatrix} 3000 & 200 & 400 & 600 \\ 700 & 350 & 700 & 100 \\ 1000 & 100 & 500 & 800 \end{bmatrix} + \begin{bmatrix} 5000 & 50 & 200 & 0 \\ 2000 & 100 & 300 & 300 \\ 2000 & 100 & 600 & 600 \end{bmatrix} = \begin{bmatrix} 8000 & 250 & 600 & 600 \\ 2700 & 450 & 1000 & 400 \\ 3000 & 200 & 1100 & 1400 \end{bmatrix} \tag{2.6}$$

2.3.1 Adição

A adição de matrizes é definida somente para matrizes de mesma ordem. A soma de duas matrizes $\mathbf{A} = [a_{ij}]_{m\times n}$ e $\mathbf{B} = [b_{ij}]_{m\times n}$ é uma matriz $m \times n$, que denotaremos por $\mathbf{A} + \mathbf{B}$, cujos elementos são as somas dos elementos correspondentes de $\mathbf{A}$ e $\mathbf{B}$; isto é:

$$\mathbf{A} + \mathbf{B} = [a_{ij} + b_{ij}]_{m\times n} \tag{2.7}$$

Por exemplo:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} + \begin{bmatrix} 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix} = \begin{bmatrix} 8 & 10 & 12 \\ 14 & 16 & 18 \end{bmatrix} \tag{2.8}$$

Observemos que, pela foram como foi definida, a adição de matrizes satisfaz as mesmas propriedades da adição de números reais, tais como a comutatividade e a associatividade.

2.3.2 Multiplicação por um escalar

Seja $\mathbf{A} = [a_{ij}]_{m\times n}$ uma matriz e k um número real (ou complexo). A multiplicação de $\mathbf{A}$ por k é a matriz $k\mathbf{A}$, definida por:

$$k\mathbf{A} = [ka_{ij}]_{m\times n} \tag{2.9}$$

Por exemplo:

$$3 \cdot \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} = \begin{bmatrix} 3 & 6 & 9 \\ 12 & 15 & 18 \end{bmatrix} \tag{2.10}$$

2.3.3 Matriz Transposta

Dada uma matriz $\mathbf{A} = [a_{ij}]_{m\times n}$, podemos obter uma outra matriz, $\mathbf{A}^T = [b_{ji}]_{n\times m}$, cujas linhas são as colunas de $\mathbf{A}$ e cujas colunas são as linhas de $\mathbf{A}$, isto é, $a_{ij} = b_{ji}$. A matriz $\mathbf{A}^T$ é chamada de transposta de $\mathbf{A}$ e é definida por:

Exemplos:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}^T = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \tag{2.11}$$

$$\begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix}^T = \begin{bmatrix} 7 & 9 & 11 \\ 8 & 10 & 12 \end{bmatrix} \tag{2.12}$$

2.3.4 Multiplicação entre matrizes

O produto de duas matrizes, tais que o número de colunas da primeira matriz é igual ao número de linhas da segunda matriz, é definido como segue. Sejam $\mathbf{A} = [a_{ij}]_{m\times n}$ e $\mathbf{B} = [b_{ij}]_{n\times p}$ duas matrizes. O produto $\mathbf{C} = \mathbf{A}\cdot\mathbf{B}$ é a matriz $m \times p$, definida por:

$$c_{ij} = \sum_{k=1}^{n} a_{ik}b_{kj}, \quad \text{para } i = 1, 2, \ldots, m \text{ e } j = 1, 2, \ldots, p \tag{2.13}$$

Exemplos:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \cdot \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix} = \begin{bmatrix} 58 & 64 \\ 139 & 154 \end{bmatrix} \tag{2.14}$$

$$\begin{bmatrix} 2 & 4 \\ 3 & 5 \end{bmatrix} \cdot \begin{bmatrix} 6 & 8 & 10 \\ 7 & 9 & 11 \end{bmatrix} = \begin{bmatrix} 46 & 62 & 78 \\ 57 & 77 & 97 \end{bmatrix} \tag{2.15}$$

2.3.5 Propriedades da álgebra matricial

Sejam $\mathbf{A}$, $\mathbf{B}$ e $\mathbf{C}$ matrizes de mesma ordem, e α e β escalares. As seguintes propriedades matriciais são válidas:

- $\mathbf{A} + \mathbf{B} = \mathbf{B} + \mathbf{A}$
- $\mathbf{A} + (\mathbf{B} + \mathbf{C}) = (\mathbf{A} + \mathbf{B}) + \mathbf{C}$
- A matriz $\bar{0}$, definida por $[\bar{0}]_{ij} = 0$, para $i = 1, 2, \ldots, m$ e $j = 1, 2, \ldots, n$, é tal que $\mathbf{A} + \bar{0} = \mathbf{A}$, para qualquer matriz $\mathbf{A}$, $m \times n$. A matriz $\bar{0}$ é chamada de matriz nula $m \times n$.
- Para cada matriz $\mathbf{A}$, $m\times n$, existe uma única matriz $-\mathbf{A}$, $m\times n$, onde $\mathbf{A}+(-\mathbf{A}) = \bar{0}$.
- $\alpha(\beta\mathbf{A}) = (\alpha\beta)\mathbf{A}$
- $(\alpha + \beta)\mathbf{A} = \alpha\mathbf{A} + \beta\mathbf{A}$
- $\alpha(\mathbf{A} + \mathbf{B}) = \alpha\mathbf{A} + \alpha\mathbf{B}$
- $\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C}$
- Para cada inteiro positivo p, a matriz $p\times p$, chamada de matriz identidade, definida por:

  $$\mathbf{I} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \tag{2.16}$$

  é tal que, para qualquer matriz $\mathbf{A}$, $m\times p$, e qualquer matriz $\mathbf{B}$, $p\times n$, temos $\mathbf{IA} = \mathbf{A}$ e $\mathbf{BI} = \mathbf{B}$.

- $\mathbf{A}(\mathbf{B}+\mathbf{C}) = \mathbf{AB} + \mathbf{AC}$ e $(\mathbf{B}+\mathbf{C})\mathbf{A} = \mathbf{BA} + \mathbf{CA}$
- $\alpha(\mathbf{AB}) = (\alpha\mathbf{A})\mathbf{B} = \mathbf{A}(\alpha\mathbf{B})$
- $(\mathbf{A}^T)^T = \mathbf{A}$
- $(\mathbf{A}+\mathbf{B})^T = \mathbf{A}^T + \mathbf{B}^T$
- $\alpha(\mathbf{A}^T) = (\alpha\mathbf{A})^T$
- $(\mathbf{AB})^T = \mathbf{B}^T\mathbf{A}^T$

3 Matrizes na Programação: Aplicações e Conceitos Fundamentais

No contexto da programação, uma matriz é uma estrutura de dados que armazena uma coleção de elementos do mesmo tipo, organizados em uma estrutura bidimensional ou multidimensional. Trata-se de uma das ferramentas mais fundamentais e poderosas para resolver uma ampla gama de problemas computacionais, sendo amplamente utilizada em diversas áreas da computação, como processamento de imagens, álgebra linear, análise de dados, inteligência artificial, e sistemas de controle. O uso de matrizes é essencial para representar e manipular grandes volumes de dados de maneira eficiente e escalável, o que as tornam indispensáveis na maioria dos algoritmos e técnicas computacionais.

3.1 Definição e estrutura

Tal como definido anteriormente, uma matriz é composta por linhas e colunas, onde cada elemento é identificado por dois índices: o índice da linha e o índice da coluna. O número de linhas e colunas pode variar dependendo da aplicação, o que resulta em diferentes tipos de matrizes:

- **Matriz unidimensional (vetor)**: possui apenas uma linha ou uma coluna.
- **Matriz bidimensional**: A forma mais comum de matriz, contendo múltiplas linhas e colunas.
- **Matriz Multidimensional**: Extensão das matrizes bidimensionais, onde mais de duas dimensões são usadas para armazenar dados.

3.2 Aplicações de Matrizes em Programação

As matrizes têm uma gama de aplicações práticas em diferentes áreas da ciência da computação. A seguir, serão apresentadas algumas das principais áreas de uso de matrizes na programação.

3.2.1 Processamento de Imagens

Em visão computacional e processamento de imagens, as imagens digitais são frequentemente representadas como matrizes de valores. Cada elemento da matriz corresponde a um pixel da imagem, e o valor armazenado em cada posição da matriz representa a intensidade ou cor do pixel. Uma imagem colorida pode ser representada como uma matriz tridimensional, onde as três dimensões correspondem às três cores fundamentais (vermelho, verde e azul, ou RGB) e os elementos armazenam os valores de intensidade dessas cores.

```python
from imageio import imread
import matplotlib.pyplot as plt

img = imread('lena.png')                  # shape (H, W, 3) uint8
img_float = img.astype(np.float32)/255  # normalizacao

# Conversao para escala de cinza (media ponderada ITU-R 601-2)
gray = np.dot(img_float[...,:3],
              [0.299, 0.587, 0.114])

# Filtro de convolucao 3x3 (deteccao de bordas Sobel)
sobel_x = np.array([[-1, 0, 1],
                    [-2, 0, 2],
                    [-1, 0, 1]], dtype=np.float32)

from scipy.ndimage import convolve
grad_x = convolve(gray, sobel_x)
```

Listing 1: Processamento de imagens em Python

3.2.2 Álgebra Linear e Resolução de Sistemas Lineares

Matrizes são amplamente utilizadas em álgebra linear, especialmente na resolução de sistemas de equações lineares. A solução de um sistema de equações lineares pode ser expressa como a multiplicação de uma matriz pelos valores das variáveis. Um exemplo clássico é a solução de um sistema de equações $Ax = b$, onde A é uma matriz de coeficientes, x é o vetor de variáveis desconhecidas, e b é o vetor de constantes (onde também podem ser descritos como resultados conhecidos).

```python
import numpy as np
import time

# Sistema 3x3 bem condicionado
A = np.array([[ 3.0,  2.0, -1.0],
              [ 2.0, -2.0,  4.0],
              [-1.0,  0.5, -1.0]], dtype=np.float64)

b = np.array([1.0, -2.0, 0.0])

x = np.linalg.solve(A, b)
print("Solucao (solve):", x)
# Saida: [ 1. -2.  1.]

# Verificacao: ||Ax - b||
residuo = np.linalg.norm(A @ x - b)
print(f"Residuo: {residuo:.2e}")
```

Listing 2: Resolução de sistemas lineares em Python

```python
# Sistema inconsistente: 5 equacoes, 3 variaveis
A = np.array([[1, 2, 3],
              [4, 5, 6],
              [7, 8, 9],
              [10,11,12],
              [13,14,15]], dtype=np.float64)

b = np.array([1, 2, 3, 4, 100], dtype=np.float64)  # outlier

# 3.1. Numpy (SVD)
x_lstsq, residuals, rank, s = np.linalg.lstsq(A, b, rcond=None)
print("lstsq (SVD):", x_lstsq)

# 3.2. SciPy iterativo (grande escala)
from scipy.sparse.linalg import lsqr
x_lsqr, istop, itn = lsqr(A, b, damp=0.0, atol=1e-10, btol=1e-10)[:3]
print("lsqr:", x_lsqr)
```

Listing 3: Exemplo de uso para Mínimos Quadrados em Python

A multiplicação de matrizes é um conceito central, e seu uso se estende a diversos algoritmos de machine learning, redes neurais, e até mesmo em gráficos computacionais.

3.2.3 Inteligência Artificial e Redes Neurais

Nas redes neurais artificiais, matrizes são essenciais para representar os pesos das conexões entre os neurônios e para calcular as saídas das camadas de uma rede neural. A computação das ativações de uma camada em uma rede neural é, basicamente, uma multiplicação de matrizes entre os pesos e os dados de entrada.

Além disso, a operação de retropropagação, usada para treinar redes neurais, envolve a manipulação de matrizes para calcular gradientes e ajustar os pesos.

```python
class CamadaDensa:
    def __init__(self, n_entradas: int, n_neuronios: int,
                 ativacao: str = 'relu'):
        self.W = np.random.randn(n_entradas, n_neuronios) * 0.01
        self.b = np.zeros((1, n_neuronios))
        self.ativacao = ativacao

    def forward(self, X: np.ndarray) -> np.ndarray:
        self.Z = X @ self.W + self.b
        if self.ativacao == 'relu':
            return np.maximum(0, self.Z)
        elif self.ativacao == 'sigmoid':
            return 1 / (1 + np.exp(-self.Z))
        else:
            return self.Z
```

Listing 4: Método de propagação direta de Redes Neurais em Python

4 Conclusão

O uso de matrizes é central em muitos campos da ciência da computação, sendo uma das ferramentas mais poderosas e versáteis na resolução de problemas complexos. Seu uso é fundamental em áreas que envolvem grandes quantidades de dados, como processamento de imagens, inteligência artificial, álgebra linear, análise de sistemas dinâmicos e muito mais. A eficiência na manipulação de matrizes é um aspecto essencial para garantir a performance e a escalabilidade de algoritmos em diversas aplicações computacionais.

O estudo das operações com matrizes, suas propriedades e otimizações em termos de implementação é uma habilidade crucial para desenvolvedores, especialmente aqueles que lidam com dados em grande escala ou realizam cálculos numéricos avançados.