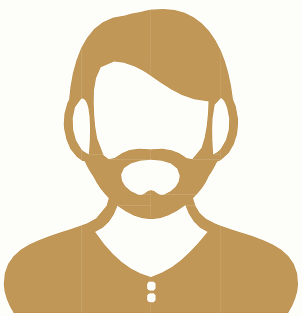
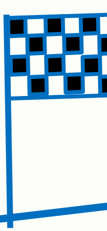
第3章 矩阵

第04讲 矩阵的基础知识

传媒与信息工程学院
欧 新 宇



- 矩阵的定义及基本操作
- 基于矩阵的向量
- 特殊形态的矩阵
- 矩阵的四则运算
- 矩阵的秩和矩阵的迹
- 矩阵的分块
- 张量的常用操作
- 矩阵的应用



矩阵的定义及基本操作

1. 矩阵的定义和基本描述

历史与目标

在数学中，**矩阵（Matrix）**是一个按照长方阵列排列的复数或实数集合，最早来自于方程组的系数及常数所构成的方阵。这一概念由19世纪英国数学家凯利首先提出。作为解决线性方程的工具，矩阵也有不短的历史。成书最早在东汉前期的《九章算术》中，用分离系数法表示线性方程组。

学习线性代数的主要目标就是：**学会利用矩阵来描述系统，并用矩阵软件工具去解决各种问题。**

1. 矩阵的定义和基本描述

矩阵的定义

【定义1】由$m \times n$个数(i=1,2,…,m; j=1,2,…,n）排成的m行n列的矩形数表就称为**矩阵**。如下所示，可以使用***加粗斜体大写英文字母***来表示一个矩阵。

$$\boldsymbol{A}=\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

若矩阵***A***是一个m行n列的矩阵，则称它为$m \times n$(阶)矩阵。为表示它是一个整体，总是加一个括弧或者方括号来表示它。矩阵中的$m \times n$个数称为矩阵***A***的元素，其中 a_{ij} 表示矩阵***A***的第 i 行第 j 列的元素。$m \times n$ 矩阵也可以被记作 $\boldsymbol{A}_{m \times n}$。

1. 矩阵的定义和基本描述

矩阵的定义

在python中，一般使用numpy数组来表示矩阵，实际上对于包括向量、矩阵及张量在内，都习惯使用numpy数组来表示，并使用numpy.array()来实现对数组的定义。

```
import numpy as np
A = np.array([
    [1, 2],
    [3, 4],
    [5, 6],
    [7, 8]])
print("矩阵A = \n{}".format(A))
print("矩阵A的形态为：{}".format(A.shape))
```

```
矩阵A =
[[1 2]
 [3 4]
 [5 6]
 [7 8]]
矩阵A的形态为：(4, 2)
```

Numpy中也有numpy.mat()和numpy.matrix()，它是array的一个子集，同时拥有更方便的计算方法，但没有array那么通用，它只适用于二维。

1. 矩阵的定义和基本描述

同型矩阵及矩阵相等

【定义2】如果两个矩阵的行数相等、列数也相等，则称它们为**同型矩阵**。若矩阵 $A=a_{ij}$ 与矩阵 $B=b_{ij}$ 是同型矩阵，且它们所有对应位置的元素均相等，即：

$$a_{ij}=b_{ij}\,(i=1,2,..,m;j=1,2,...,n)\ ,$$

则称矩阵A与矩阵B相等，记作：$A=B$。

```
import numpy as np
A1 = np.array([[1,2,3],[4,5,6]])
A2 = np.array([[1,2,3],[4,5,6]])
A3 = np.array([[1,1,1],[2,2,2]])
print("A1与A2的形态相同：{}，A1与A2相等：{}。".format(A1.shape==A2.shape, (A1==A2).all()))
print("A1与A3的形态相同：{}，A1与A3相等：{}。".format(A1.shape==A3.shape, (A1==A3).all()))
```

```
A1与A2的形态相同：True，A1与A2相等：True。
A1与A3的形态相同：True，A1与A3相等：False。
```

矩阵的转置

转置(Transpose)

【定义3】 给定矩阵$A_{m\times n}$，若将其行和列的元素进行位置互换，可以得到一个新的矩阵$B_{n\times m}$。那么矩阵B就称为矩阵A的**转置矩阵**，并记作 $B=A^T$。同时，矩阵A也称为矩阵B的**转置矩阵**。行和列的互换操作就称为矩阵的转置。

下面给出矩阵转置的Python代码：

```
import numpy as np
A = np.array([[1,2,3,4],[5,6,7,8]])

print("矩阵A=\n{}\n".format(A))
print("矩阵A的转置=\n{}".format(A.T))
```

```
矩阵A=
[[1 2 3 4]
 [5 6 7 8]]

矩阵A的转置=
[[1 5]
 [2 6]
 [3 7]
 [4 8]]
```

注意向量（一维数组）无法执行转置运算。

1. 矩阵的定义和基本描述

转置(Transpose)

矩阵的转置：转置是矩阵的重要操作之一，矩阵的转置是以对角线为轴的镜像，这条对角线从左上角到右下角被称为主对角线。

我们将矩阵$\boldsymbol{A}$的转置表示为$\boldsymbol{A}^T$，定义如下：

$$(A^T)_{i,j} = A_{j,i}$$

具体而言：

$$A = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \\ A_{3,1} & A_{3,2} \end{bmatrix} \Rightarrow A^T = \begin{bmatrix} A_{1,1} & A_{2,1} & A_{3,1} \\ A_{1,2} & A_{2,2} & A_{3,2} \end{bmatrix}$$

1. 矩阵的定义和基本描述

转置(Transpose)

- **标量的转置**：标量可以看成是只有一个元素的矩阵。因此，标量的转置等于它本身，即: $a = a^T$。

- **向量的转置**：向量可以看作是只有一列（行）的矩阵。因此，向量的转置可以看作是只有一行（列）的矩阵。
  - 若将向量元素作为行向量写在**文本行**中，则可以通过转置将其转换为标准的列向量，比如： $\boldsymbol{x} = [x_1, x_2, x_3]^T$。
  - 若将向量直接写在**文本行**中，则向量$\boldsymbol{a}$本身就是标准列向量，其转置$\boldsymbol{a}^T$为行向量。

矩阵的转置

转置的运算规律

矩阵的转置也是一种运算，满足下述运算规律（假设运算都是可行的）：

- $(A^T)^T=A$
- $(A+B)^T=A^T+B^T$
- $(\lambda A)^T=\lambda A^T$
- $\boxed{(AB)^T=B^TA^T}$ 可以脱括号，但是要注意顺序

矩阵的转置

【例1】矩阵转置的例子

已知 $A=\begin{bmatrix}2 & 0 & -1\\ 1 & 3 & 2\end{bmatrix}$，$B=\begin{bmatrix}1 & 7 & -1\\ 4 & 2 & 3\\ 2 & 0 & 1\end{bmatrix}$，求$(AB)^T$。

此处只给出Python代码的实现方法：

```python
import numpy as np
A = np.array([[2,0,-1],[1,3,2]])
B = np.array([[1,7,-1],[4,2,3],[2,0,1]])
print(np.dot(A,B).T)
```

```
[[ 0 17]
 [14 13]
 [-3 10]]
```

向量的范数

矩阵的Frobenius范数

弗罗贝尼乌斯范数（Frobenius范数）：简称为F范数，是一种定义在矩阵上的范数，用于衡量矩阵的大小。F范数表示矩阵中各元素的平方和开方。矩阵$\boldsymbol{A}=(a_{ij})$的Frobenius范数为：

$$\|\boldsymbol{A}\|_F = \sqrt{\sum_{i,j} A_{i,j}^2}$$

在Python中，我们可以使用np.linalg.norm(A)来求矩阵的F范数。当A为矩阵时，np.linalg.norm()方法默认为F范数。

```python
import numpy as np

A = np.array([[1,1,2],[1,1,1]])
print(np.linalg.norm(A))
```

Last executed at 2020-05-18 09:44:49 in 3ms

```
3.0
```

基于矩阵的向量

2. 基于矩阵的向量

为了规范和便于计算，所有的量（向量、矩阵、张量）都规范成张量，并同时使用矩阵（张量）来进行表示。在程序中，我们统一使用numpy数组来表示这种量。此时，

- 一个 $1\times n$ 的行向量 a^T 就表示成一个只有一行的矩阵；
- 一个 $n\times 1$ 的列向量 b 则表示成一个只有一列的矩阵。

```python
import numpy as np
a = np.array([[1,2,3,4]])
print("a={}".format(a))
print("b=\n{}".format(a.T))
```

```
a=[[1 2 3 4]]
b=
[[1]
 [2]
 [3]
 [4]]
```

【结果分析】我们使用一个二维数组来显示向量（**两层**中括号），这种方法基本上贯穿于整个计算机领域。其中 $\boldsymbol{a}$ 用来表示一个四维行向量(二阶张量)，$\boldsymbol{b}$ 表示一个四维列向量(二阶张量)。

课堂互动一 Link

特殊形态的矩阵

3. 特殊形态的矩阵

- 方阵
- 对称矩阵
- 零矩阵
- 对角矩阵
- 单位矩阵
- 逆矩阵
- 正定矩阵
- 半正定矩阵
- 负定矩阵
- 正交矩阵

3. 特殊形态的矩阵

方阵

【定义4】行数和列数相等的矩阵称为**方阵**，即存在$A_{m\times n}, m=n$。方阵的行数或列数称为矩阵的**阶数**。例如，一个 n 阶方阵可记为

```
import numpy as np
A = np.array([[1,2,3],[4,5,6],[7,8,9]])
B = np.array([[1,1,1,1,1],[2,2,2,2,2],[3,3,3,3,3],[4,4,4,4,4],[5,5,5,5,5]])

print("矩阵A=\n{}, \n 矩阵B=\n{}".format(A,B))
print("矩阵A的形态为：{}，矩阵B的形态为：{}".format(A.shape, B.shape))
```

```
矩阵A=
[[1 2 3]
 [4 5 6]
 [7 8 9]],
矩阵B=
[[1 1 1 1 1]
 [2 2 2 2 2]
 [3 3 3 3 3]
 [4 4 4 4 4]
 [5 5 5 5 5]]
矩阵A的形态为：(3, 3)，矩阵B的形态为：(5, 5)
```

3. 特殊形态的矩阵

对称矩阵(Symmetric Matrix)

【定义5】给定矩阵$A_{m\times n}$，若其转置矩阵A^T与原矩阵相等，即：$A^T = A$，则矩阵***A***称为**对称矩阵。**

不难发现，矩阵对称的前提条件有两点：

1. 矩阵***A***是一个方阵
2. 矩阵***A***的每一个元素都满足 $A_{ij}=A_{ji}$

$$
\begin{array}{c|cccc|}
 & 0 & 1 & 2 & 3 \\
\hline
0 & 1 & 2 & 3 & 4 \\
1 & 2 & 8 & 5 & 6 \\
2 & 3 & 5 & 9 & 7 \\
3 & 4 & 6 & 7 & 0 \\
\hline
\end{array}
$$

3. 特殊形态的矩阵

对称矩阵(Python描述)

```
import numpy as np
A = np.array([[1,5,6,7],[5,2,8,9],[6,8,3,0],[7,9,0,4]])

print("矩阵A = \n{}\n".format(A))
print("矩阵A的对称矩阵(转置矩阵) = \n{}\n".format(A.T))
```

```
矩阵A =
[[1 5 6 7]
 [5 2 8 9]
 [6 8 3 0]
 [7 9 0 4]]
```

```
矩阵A的对称矩阵(转置矩阵) =
[[1 5 6 7]
 [5 2 8 9]
 [6 8 3 0]
 [7 9 0 4]]
```

3. 特殊形态的矩阵

零矩阵

所有元素都为0的矩阵称为**零矩阵**，记作$\boldsymbol{O}$。此外还可以通过下标法标识出零矩阵的形态，例如一个4×5的零矩阵，可以表示为$O_{4\times5}$。值得注意的是，不同型的零矩阵是不同(不相等)的，例如：$O_{4\times5}\neq O_{2\times3}$。

零矩阵最重要的作用就是用来初始化矩阵，

- 一方面可以使用零矩阵来表示实际存储数据矩阵的规模，达到初始化矩阵和申请内存空间的功能；
- 另一方面零矩阵也是占用存储空间最小的矩阵。

3. 特殊形态的矩阵

零矩阵

- ✓ **任意匹配：**

  (A==B).any()
- ✓ **所有匹配**：

  (A==B).all()
- ✓ **按位匹配：**（A==B）
- ✓ **形态匹配：**

  A.shape==B.shape

```python
import numpy as np
A = np.zeros([4,5])
print(A)
```

```
[[0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]]
```

```python
import numpy as np
A1 = np.zeros([4,5])
A2 = np.zeros([2,3])
A3 = np.zeros([4,5])
print("A1 = A2 is {}。".format(A1==A2))
print("A1 = A3 is {}。".format((A1==A3).all()))
```

```
A1 = A2 is False。
A1 = A3 is True。
```

3. 特殊形态的矩阵

对角矩阵（Diagonal Matrix）

【定义6】除了主对角线上的元素外所有的元素都为0，这种矩阵就称为**对角矩阵**，即：$a_{ij}=0, i\neq j$。

例如：

$$
\boldsymbol{A}=\begin{bmatrix} 1 & & & & \\ & 2 & & & \\ & & 3 & & \\ & & & 4 & \\ & & & & 5 \end{bmatrix}
$$

在对角矩阵中，为0的元素位置可以省去不写。

```
import numpy as np
A = np.diag([1,2,3,4,5])
print(A)
```

```
[[1 0 0 0 0]
 [0 2 0 0 0]
 [0 0 3 0 0]
 [0 0 0 4 0]
 [0 0 0 0 5]]
```

3. 特殊形态的矩阵

单位矩阵（Identity Matrix）

【定义7】在对角矩阵中，如果对角线上的元素都为1，则该矩阵称为**单位矩阵**。任意向量和单位矩阵相乘，都不会改变。我们把 n 阶单位矩阵记作 $\boldsymbol{I}_n$ （也被记作 $\boldsymbol{E}_n$ ），形式上：$\forall x \in \mathbb{R}^n, \boldsymbol{I}_n x = x$。

$$\boldsymbol{I} = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix}$$

```python
import numpy as np
I = np.eye(4)
print(I)
```

3. 特殊形态的矩阵

逆矩阵（Matrix Inversion）

矩阵$\boldsymbol{A}$的**逆矩阵**记作$\boldsymbol{A}^{-1}$，其定义的矩阵满足如下条件：

$$A^{-1}A = \boldsymbol{I}_n$$

通常可以通过矩阵逆解的方式求解逆矩阵，但是首先需要考虑逆矩阵是否存在。更一般地说，相同的逆矩阵可以用于多次求解不同向量***b***的方程，如后续的基底变换。但实际应用中，逆矩阵主要作为理论工具，因为逆矩阵A^{-1}在数字计算机上只能表现有限的精度。求解逆矩阵，在python中可以通过如下代码实现:

```
from scipy import linalg
linalg.inv(A)
```

3. 特殊形态的矩阵

逆矩阵的性质

1. 如果矩阵A可逆，则A的逆矩阵唯一。

2. 若A、B为同阶可逆**方阵**，且满足AB=I，则BA=I，即A和B互逆

3. 若A可逆，则A^{-1}也可逆，且$(A^{-1})^{-1}=A$ 。

4. 若A可逆，数λ≠0，则λA可逆，且 $(\lambda A)^{-1}= A^{-1}\lambda^{-1}$

5. 若A、B均为n阶可逆方阵，则AB也可逆，且$(AB)^{-1} = (B^{-1})(A^{-1})$。此性质可推广至k个同阶方阵连乘的情况：

$$(A_1 A_2 \ldots A_k)^{-1}= A_k^{-1} A_{k-1}^{-1} \ldots A_1^{-1}$$

3. 特殊形态的矩阵

正交矩阵(Orthogonal Matrix)

正交矩阵：逆矩阵等于它的转置矩阵的方阵。正交矩阵的行向量和列向量分别都是标准正交，即该矩阵的一个内积空间的正交基是元素两两正交的基，这意味着基向量的模长都是单位长度。

正交矩阵求逆矩阵的代价很小，因此备受关注，此外，它还具有很多有趣的性质，例如：

1). $\boldsymbol{A}^{-1}=\boldsymbol{A}^{T}$;

2). $\boldsymbol{A}^{T}$也是正交矩阵;

3). $\boldsymbol{A}$的行列式值等于1或-1;

4). $\boldsymbol{A}$的各行（列）是单位向量，且两两正交

3. 特殊形态的矩阵

正定、半正定和负定矩阵

- **正定矩阵(Positive Definite Matrix)**：对于矩阵M，任意非零向量z，都有：$z^T Mz > 0$。
- **半正定矩阵(Positive Semidefinite Matrix)**：对于矩阵M，任意非零向量z，都有：$z^T Mz \geq 0$。
- **负定矩阵(Negative Definite Matrix)**：对于矩阵M，任意非零向量z，都有：$z^T Mz < 0$。

对于以上三种特殊矩阵，它们都有很多不同的判定方法和性质，目前只需要记得它们的定义即可。

课堂互动二 Link

读万卷书 行万里路 只为最好的修炼

QQ：14777591 (宇宙骑士)

Email：ouxinyu@alumni.hust.edu.cn

Tel：18687840023