



Линейная алгебра

Лекция 2. Матрицы

Денис Волк

Senior Data Scientist @ KPMG



- Математик, кандидат наук, МГУ
- Специальность: динамические системы и случайные процессы
- Работал в университетах Триеста, Рима, Стокгольма
- Автор 14 научных статей в международных журналах

Денис Волк

Senior Data Scientist @ KPMG

1. Что такое матрицы? Зачем они нужны?
2. Матричные операции
3. Какие бывают матрицы?
4. Операции матрицы и вектора
5. Матрицы как преобразования пространства
6. Композиция преобразований пространства
7. Обратная матрица
8. Матрицы и системы линейных уравнений
9. Определитель матрицы

Что такое матрицы?

Матрицы

44

Матрица - прямоугольная таблица чисел:

$$\begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,1} & x_{m,2} & \cdots & x_{m,n} \end{pmatrix}$$

Матрица размера $m \times n$

Матрица размера 2×2

$$\begin{pmatrix} 0 & 1 \\ -1 & 2 \end{pmatrix}$$

Матрица размера 2×3

$$\begin{pmatrix} 0 & 1 & 5 \\ -1 & 2 & -2 \end{pmatrix}$$

Матрица размера 3×1

$$\begin{pmatrix} 3 \\ -1 \\ 8 \end{pmatrix}$$

Матрицы обычно используются для записи:

1. наборов векторов
2. преобразований векторов и систем координат (“линейных отображений”)
3. соотношений между векторами (“билинейных форм”)

Матрицы на python

46

```
[16]: import numpy as np
```

```
[17]: A = np.array([[1, 2, 3], [4, 5, 6]])  
A
```

Создаём матрицу 2×3

```
[17]: array([[1, 2, 3],  
           [4, 5, 6]])
```

```
[18]: B = np.zeros((2, 2))  
B
```

Создаём нулевую матрицу 2×2

```
[18]: array([[0., 0.],  
           [0., 0.]])
```

```
[19]: C = np.eye(3)  
C
```

Создаём матрицу 3×3 с единицами на
диагонали

```
[19]: array([[1., 0., 0.],  
           [0., 1., 0.],  
           [0., 0., 1.]])
```

Матричные операции

Сложение и вычитание матриц

48

Как и векторы, две матрицы одного и того же размера $m \times n$ можно складывать и вычитать поэлементно:

$$A = \begin{pmatrix} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} & \cdots & a_{m,n} \end{pmatrix} \quad B = \begin{pmatrix} b_{1,1} & \cdots & b_{1,n} \\ \vdots & \ddots & \vdots \\ b_{m,1} & \cdots & b_{m,n} \end{pmatrix}$$

$$A + B = \begin{pmatrix} a_{1,1} + b_{1,1} & \cdots & a_{1,n} + b_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} + b_{m,1} & \cdots & a_{m,n} + b_{m,n} \end{pmatrix}$$

А также поэлементно умножать на скаляры, как и векторы.

Умножение матриц

49

Две матрицы, одна размера $m \times n$, другая $n \times k$, можно перемножить по правилу “строка на столбец”:

$$A = \begin{pmatrix} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} & \cdots & a_{m,n} \end{pmatrix} \quad B = \begin{pmatrix} b_{1,1} & \cdots & b_{1,k} \\ \vdots & \ddots & \vdots \\ b_{n,1} & \cdots & b_{n,k} \end{pmatrix}$$

$$A \cdot B = \begin{pmatrix} (a_1, b_1) & \cdots & (a_1, b_k) \\ \vdots & \ddots & \vdots \\ (a_m, b_1) & \cdots & (a_m, b_k) \end{pmatrix}$$

Здесь (a_i, b_j) - скалярные
произведения векторов
 $a_i = (a_{i,1}, \dots, a_{i,n})$
 $b_j = (b_{1,j}, \dots, b_{n,j})$

Свойства умножения матриц

50

Некоторые свойства умножения чисел всё ещё работают:

$$A \cdot (B + C) = A \cdot B + A \cdot C$$

С умножением на скаляры тоже всё хорошо (как у векторов).

!

Но, как правило, $A \cdot B \neq B \cdot A$ (даже когда размерности подходящие!)

Умножение матриц на python

51

```
[29]: A = np.array([[1, 2], [4, 5]])  
A
```

```
[29]: array([[1, 2],  
           [4, 5]])
```

```
[30]: B = np.array([[-2, 3], [0, 1]])  
B
```

```
[30]: array([[-2, 3],  
           [0, 1]])
```

```
[31]: A @ B
```

```
[31]: array([[-2, 5],  
           [-8, 17]])
```

```
[32]: B @ A
```

```
[32]: array([[10, 11],  
           [4, 5]])
```

Операция умножения

$AB \neq BA$

Умножение матриц на python

52

```
[29]: A = np.array([[1, 2], [4, 5]])  
A
```

```
[29]: array([[1, 2],  
           [4, 5]])
```

```
[30]: B = np.array([[-2, 3], [0, 1]])  
B
```

```
[30]: array([[-2, 3],  
           [0, 1]])
```

```
[31]: A @ B
```

```
[31]: array([[-2, 5],  
           [-8, 17]])
```

```
[32]: B @ A
```

```
[32]: array([[10, 11],  
           [4, 5]])
```

Два варианта записи



```
[34]: np.dot(B, A)
```

```
[34]: array([[10, 11],  
           [4, 5]])
```

Перемножение матриц и векторов

53

Вектор-строку длины n можно рассматривать как матрицу размера $1 \times n$.

Вектор-столбец длины n можно рассматривать как матрицу размера $n \times 1$.

Умножаем по правилу умножения матриц:

$$(u_1, \dots, u_m) \cdot \begin{pmatrix} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} & \cdots & a_{m,n} \end{pmatrix} = (v_1, \dots, v_n)$$

$$\begin{pmatrix} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} & \cdots & a_{m,n} \end{pmatrix} \cdot \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} = \begin{pmatrix} v_1 \\ \vdots \\ v_m \end{pmatrix}$$

Единичная матрица I_n размера $n \times n$, где n любое:

$$I_n = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

Обладает свойством: для любой матрицы A размера $n \times k$

$$A \cdot I_k = A, \quad I_n \cdot A = A$$

Узнали:

1. Что такое матрицы
2. Сложение и вычитание матриц
3. Умножение матриц
4. Единичная матрица
5. Реализация на python

Линейные отображения

Линейные отображения

57

Линейное отображение – такое отображение F пространства векторов, что для любых векторов u и v и любых чисел α_1, α_2 выполняется условие

$$F(\alpha_1 u + \alpha_2 v) = \alpha_1 F(u) + \alpha_2 F(v)$$

Оказывается, любое линейное отображение задаётся умножением на некоторую матрицу A :

$$F(v) = A \cdot v$$

Например, тождественное отображение задаётся умножением на единичную матрицу:

$$F(v) = I \cdot v = v$$

Когда векторов много

58

Часто мы хотим применить отображение не к одному вектору, а сразу к большому количеству. На python это можно было бы реализовать циклом, но можно лучше.

1. Поместим все векторы в одну большую матрицу $n \times K$, K – количество векторов:

$$V = \begin{pmatrix} v_{1,1} & \cdots & v_{1,K} \\ \vdots & \ddots & \vdots \\ v_{n,1} & \cdots & v_{n,K} \end{pmatrix}$$
$$\quad \quad \quad v_1 \quad \quad \quad v_K$$

2. Умножим матрицу отображения A на матрицу V :

$$R = AV$$

3. Вуаля! Столбцы полученной матрицы R есть образы наших векторов!

Композиция линейных отображений

59

Пусть F и G – линейные отображения. Композиция F и G – новое линейное отображение $F \cdot G$, полученное как последовательное применение сначала G , потом F :

$$(F \cdot G)(v) = F(G(v))$$

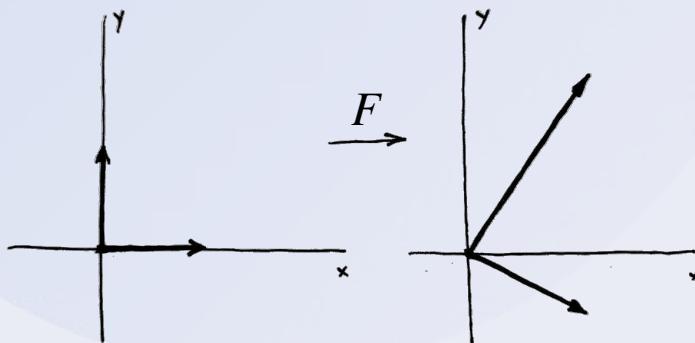
Порядок имеет значение: композиция F и G и композиция G и F – вообще говоря, разные отображения!

Матрица композиции отображений есть произведение матриц самих отображений.

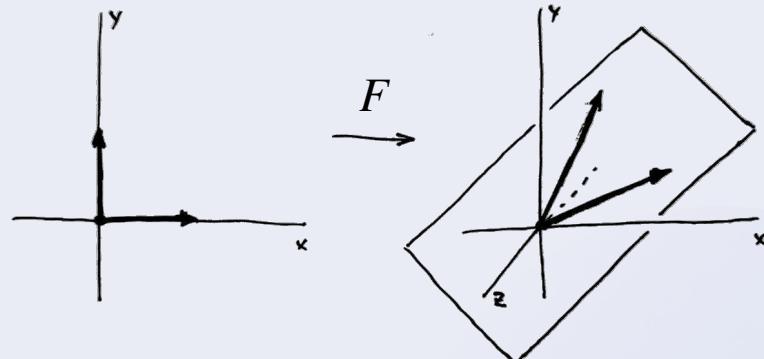
Линейные отображения: примеры

60

Квадратные матрицы $n \times n$ задают преобразование пространства:



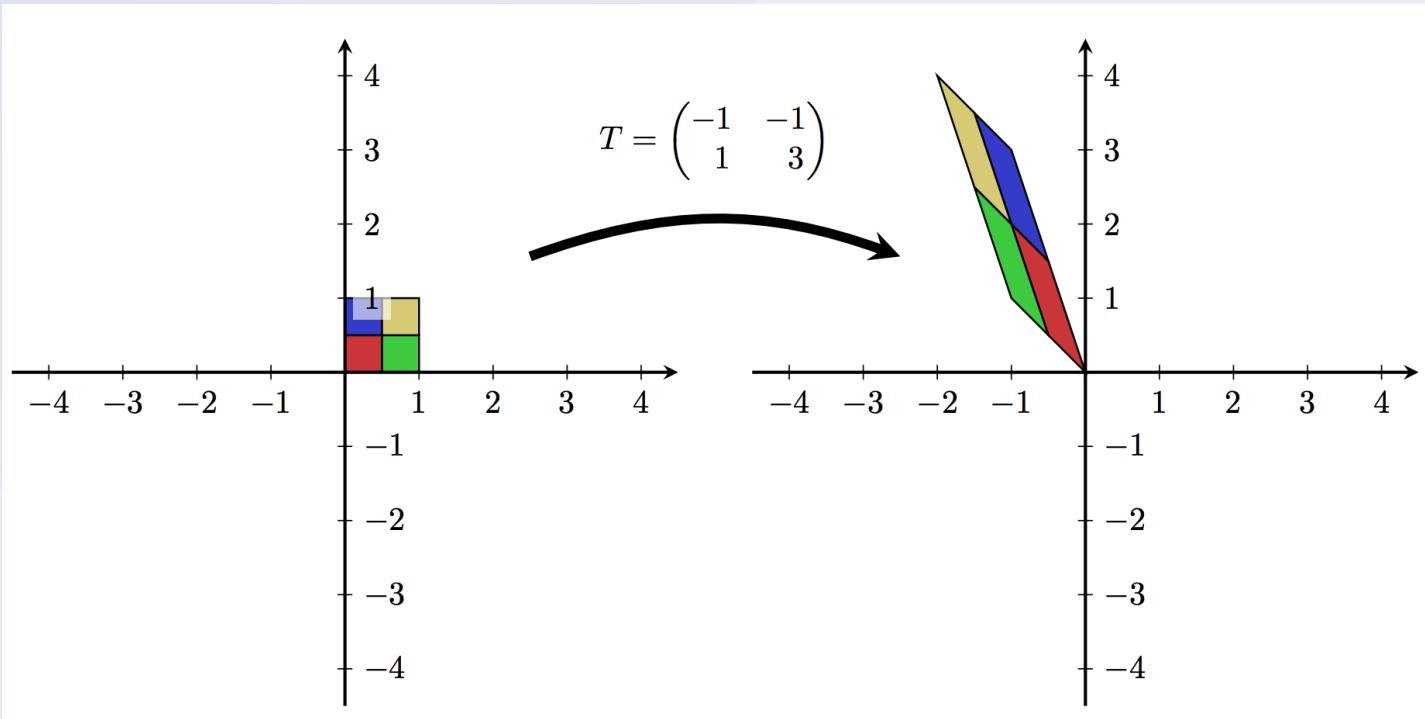
Матрицы размера $n \times k$, $k > n$, задают отображение в пространство большей размерности:



Матрицы размера $n \times k$, $k < n$, задают отображение в пространство меньшей размерности.

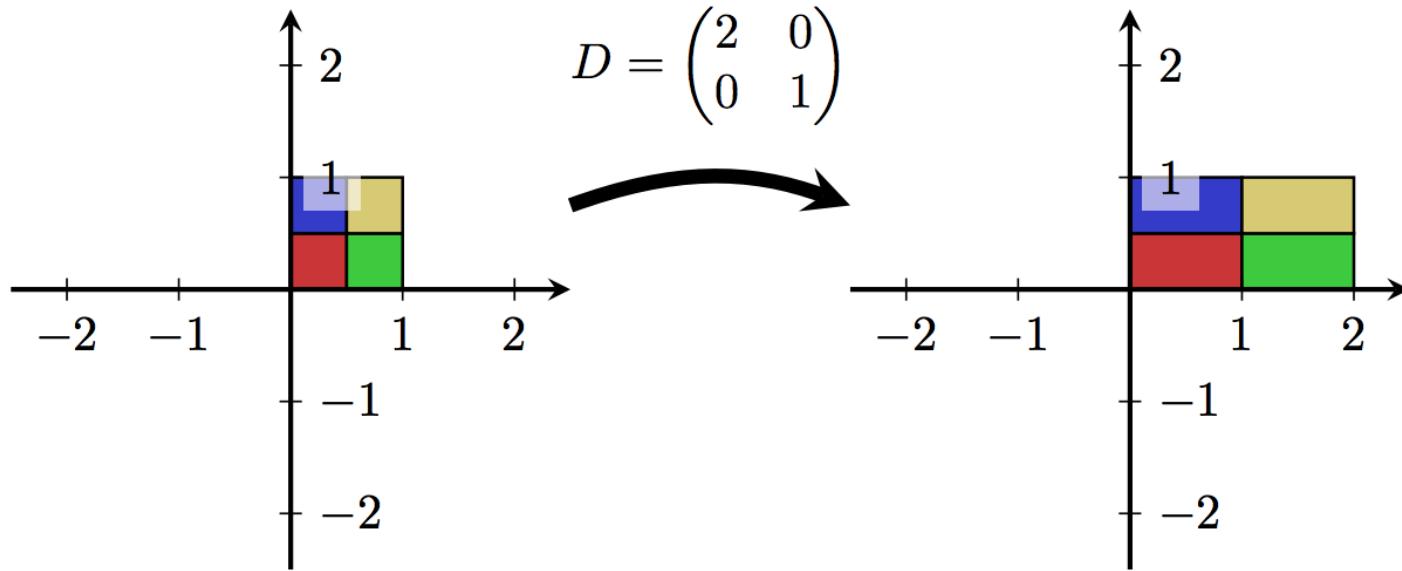
Линейные отображения: примеры

61



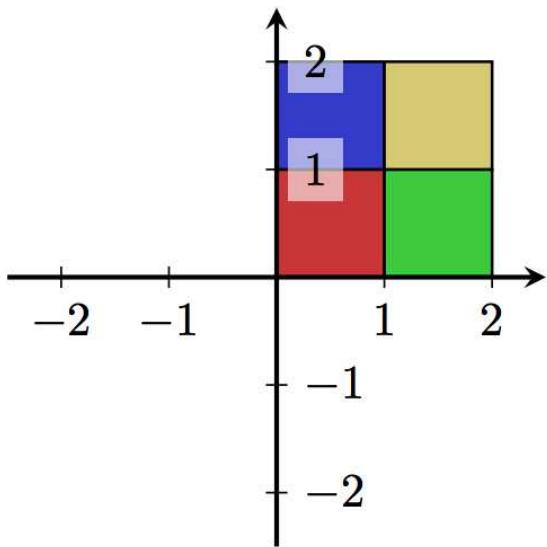
Линейные отображения: примеры

62



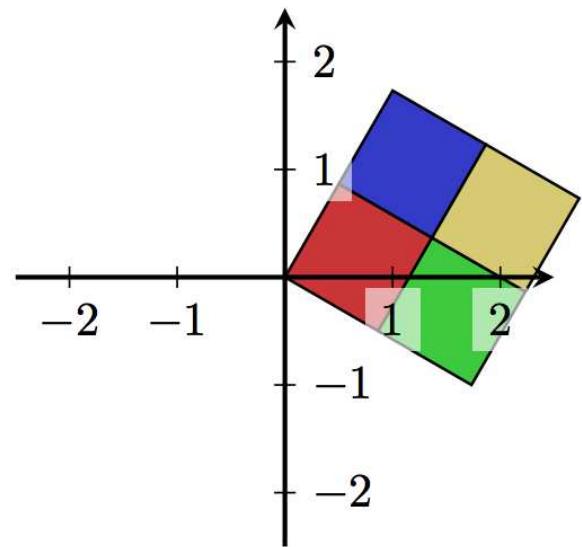
Линейные отображения: примеры

63



$$A = \begin{pmatrix} \cos \frac{\pi}{6} & -\sin \frac{\pi}{6} \\ \sin \frac{\pi}{6} & \cos \frac{\pi}{6} \end{pmatrix}$$

A large black curved arrow points from the matrix equation to the second coordinate system.



Обратная матрица

64

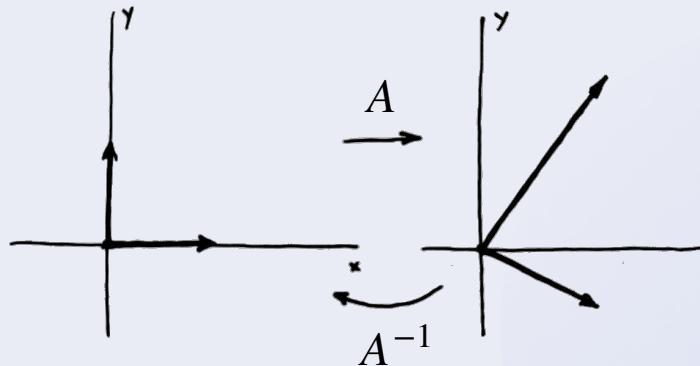
Матрица, обратная к данной, — такая матрица, что при их перемножении получается единичная матрица:

$$A \cdot A^{-1} = A^{-1} \cdot A = I$$

Существует только для квадратной матрицы A (и то не для любой!).

Если существует, то единственна.

Обратная матрица задаёт обратное преобразование пространства.



Обратная матрица: вычисление на python

65

```
[1]: import numpy as np
```

```
[2]: A = np.array([[1, 2], [4, 5]])
A
```

```
[2]: array([[1, 2],
           [4, 5]])
```

```
[3]: B = np.linalg.inv(A)
B
```

```
[3]: array([[-1.66666667,  0.66666667],
           [ 1.33333333, -0.33333333]])
```

```
[4]: A @ B
```

```
[4]: array([[1., 0.],
           [0., 1.]])
```

```
[5]: B @ A
```

```
[5]: array([[1., 0.],
           [0., 1.]])
```

```
[6]: C = np.array([[1, 2], [2, 4]])
C
```

```
[6]: array([[1, 2],
           [2, 4]])
```

```
[7]: D = np.linalg.inv(C)
```

LinAlgError

...

LinAlgError: Singular matrix

У матрицы **C** нет обратной.

Узнали:

1. Что такое линейные отображения
2. Как они связаны с матрицами
3. Как применить отображение к одному и нескольким векторам
4. Что такое обратное отображение
5. Реализация на python

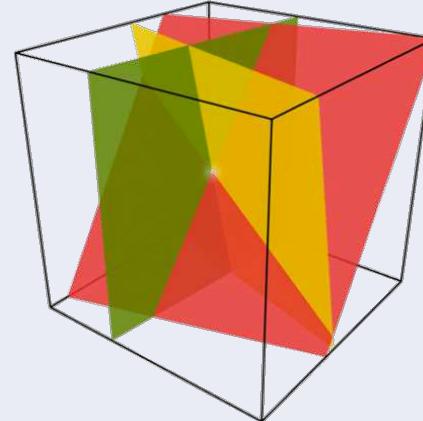
Системы линейных уравнений

Системы линейных уравнений

$$\begin{cases} a_{1,1}x_1 + \dots + a_{n,1}x_n = y_1 \\ \dots \\ a_{m,1}x_1 + \dots + a_{m,n}x_n = y_m \end{cases}$$

Возникают повсеместно, в том числе в

- приближении функций методом наименьших квадратов
- задачах оптимизации
- линейной и логистической регрессии
- интерполяции пропущенных значений



$$\begin{cases} a_{1,1}x_1 + \dots + a_{n,1}x_n = y_1 \\ \dots \\ a_{m,1}x_1 + \dots + a_{m,n}x_n = y_m \end{cases}$$

Матричная форма записи:

$$\begin{pmatrix} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} & \cdots & a_{m,n} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix}$$

A x y

или просто $Ax = y$

Решение системы линейных уравнений

70

Если матрица A обратима, то умножим обе части слева на A^{-1} :

$$Ax = y \implies A^{-1}Ax = A^{-1}y \iff x = A^{-1}y$$

Когда же A^{-1} существует, а когда нет?

Оказывается, A^{-1} существует всегда, когда информации необходимо и достаточно, чтобы однозначно решить систему уравнений.

А именно, когда

- 1) неизвестных ровно столько же, сколько уравнений (т.е. $m = n$, квадратная матрица)
- 2) нельзя сократить число уравнений, складывая их и умножая на числа (т.е. строки матрицы линейно независимы)

Определитель матрицы

71

Но как понять, когда строки матрицы линейно независимы? Когда $\det A \neq 0$.

Определитель (детерминант) матрицы:

$$\det(a_{1,1}) = a_{1,1} \quad \det \begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} = a_{1,1}a_{2,2} - a_{2,1}a_{1,2}$$

$$\det \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} = a_{1,1}a_{2,2}a_{3,3} + a_{1,2}a_{2,3}a_{3,1} + a_{2,1}a_{1,3}a_{3,2} \\ - a_{3,1}a_{2,2}a_{1,3} - a_{1,2}a_{2,1}a_{3,3} - a_{1,1}a_{2,3}a_{3,2}$$

Для большей размерности – более сложная формула.

Определитель матрицы

72

Но как понять, когда строки матрицы линейно независимы? Когда $\det A \neq 0$.

Определитель (детерминант) матрицы:

$$\det(a_{1,1}) = a_{1,1}$$

$$\det \begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} = \underline{a_{1,1}a_{2,2}} - \underline{a_{2,1}a_{1,2}}$$

$$\det \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} = a_{1,1}a_{2,2}a_{3,3} + a_{1,2}a_{2,3}a_{3,1} + a_{2,1}a_{1,3}a_{3,2} - a_{3,1}a_{2,2}a_{1,3} - a_{1,2}a_{2,1}a_{3,3} - a_{1,1}a_{2,3}a_{3,2}$$

Для большей размерности – более сложная формула.

Определитель матрицы

73

Но как понять, когда строки матрицы линейно независимы? Когда $\det A \neq 0$.

Определитель (детерминант) матрицы:

$$\det(a_{1,1}) = a_{1,1}$$

$$\det \begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} = \underline{a_{1,1}a_{2,2}} - \underline{a_{2,1}a_{1,2}}$$

$$\det \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} = \frac{a_{1,1}a_{2,2}a_{3,3} + a_{1,2}a_{2,3}a_{3,1} + a_{2,1}a_{1,3}a_{3,2}}{-a_{3,1}a_{2,2}a_{1,3} - a_{1,2}a_{2,1}a_{3,3} - a_{1,1}a_{2,3}a_{3,2}}$$

Для большей размерности – более сложная формула.

Определитель матрицы

74

Но как понять, когда строки матрицы линейно независимы? Когда $\det A \neq 0$.

Определитель (детерминант) матрицы $\det A$ или $|A|$:

$$\det(a_{1,1}) = a_{1,1}$$

$$\det \begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} = \underline{a_{1,1}a_{2,2}} - \underline{a_{2,1}a_{1,2}}$$

$$\det \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} = \underline{a_{1,1}a_{2,2}a_{3,3} + a_{1,2}a_{2,3}a_{3,1} + a_{2,1}a_{1,3}a_{3,2}} \\ \underline{-a_{3,1}a_{2,2}a_{1,3} - a_{1,2}a_{2,1}a_{3,3} - a_{1,1}a_{2,3}a_{3,2}}$$

Для большей размерности – более сложная формула.

Узнали:

1. Что такое системы линейных уравнений
2. Для чего они нужны
3. Как их решать
4. Связь с обратными матрицами
5. Зачем нужен определитель матрицы, и как его вычислить