

Практическое задание

In [1]:

```
1 import numpy as np
2 np.set_printoptions(precision=2, suppress=True)
```

1. Найти с помощью NumPy SVD для матрицы

$$\begin{pmatrix} 1 & 2 & 0 \\ 0 & 0 & 5 \\ 3 & -4 & 2 \\ 1 & 6 & 5 \\ 0 & 1 & 0 \end{pmatrix}.$$

In [2]:

```
1 A = np.array([[1, 2, 0],
2               [0, 0, 5],
3               [3, -4, 2],
4               [1, 6, 5],
5               [0, 1, 0],])
6 print(f'Матрица A:\n{A}')
```

Матрица A:

```
[[ 1  2  0]
 [ 0  0  5]
 [ 3 -4  2]
 [ 1  6  5]
 [ 0  1  0]]
```

In [3]:

```
1 U, s, W = np.linalg.svd(A)
2
3 # Транспонируем матрицу W
4 V = W.T
5
6 # s - список диагональных элементов, его нужно привести к виду диагональной матрицы для
7 D = np.zeros_like(A, dtype=float)
8 D[np.diag_indices(min(A.shape))] = s
```

In [4]:

```
1 print(f'Матрица D:\n{D}')
```

Матрица D:

```
[[8.82 0.  0. ]
 [0.  6.14 0. ]
 [0.  0.  2.53]
 [0.  0.  0. ]
 [0.  0.  0. ]]
```

In [5]:

```
1 print(f'Матрица U:\n{U}')
```

Матрица U:

```
[[ 0.17  0.16 -0.53 -0.8  -0.16]
 [ 0.39 -0.53  0.61 -0.43  0.03]
 [-0.14 -0.82 -0.52  0.14  0.07]
 [ 0.89  0.06 -0.25  0.38 -0.06]
 [ 0.08  0.11 -0.08 -0.11  0.98]]
```

In [6]:

```
1 print(f'Матрица V:\n{V}')
```

Матрица V:

```
[[ 0.07 -0.37 -0.93]
 [ 0.72  0.67 -0.21]
 [ 0.69 -0.65  0.31]]
```

In [7]:

```
1 # Проведем проверку
2 print(np.dot(np.dot(U, D), V.T))
```

```
[[ 1.  2. -0.]
 [-0.  0.  5.]
 [ 3. -4.  2.]
 [ 1.  6.  5.]
 [ 0.  1.  0.]]
```

Мы видим, что при умножении трех матриц получилась исходная матрица, следовательно SVD найдено верно.

2. Для матрицы из предыдущего задания найти:

а) евклидову норму;

Принимая во внимание, что евклидова норма матрицы равна евклидовой норме диагональной матрицы из ее сингулярных чисел D и принимая во внимание факт сортировки по убыванию сингулярных чисел, получим, что евклидова норма будет равна максимальному сингулярному числу μ_{max} , Т.е.

$$\|A\|_E = \mu_1.$$

Для матрицы A - это 8,82.

Проверим

In [8]:

```
1 np.linalg.norm(A, ord=2, axis=None, keepdims=False)
```

Out[8]:

8.824868854820442

Видим, что евклидову норму определили правильно

6) норму Фробениуса.

In [9]:

```
1 A_F = np.sqrt(sum(s**2))
2 A_F
```

Out[9]:

11.045361017187265

Проверим

In [10]:

```
1 np.linalg.norm(A, ord="fro", axis=None, keepdims=False)
```

Out[10]:

11.045361017187261

Видим, что норму Фробениуса определили правильно