Автор: Анна Смелова

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

Решение домашнего задания к уроку 8 “Сингулярное разложение матриц”

1. Задача

Найти с помощью NumPy SVD для матрицы

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

Решение:

In [2]:
```
A = np.array([[1, 2, 0],
              [0, 0, 5],
              [3, -4, 2],
              [1, 6, 5],
              [0, 1, 0]])
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]:
```
U, s, W = np.linalg.svd(A)

# Транспонируем матрицу W
V = W.T

# s - список диагональных элементов, его нужно привести к виду диагональной матрицы для наглядности
D = np.zeros_like(A, dtype=float)
D[np.diag_indices(min(A.shape))] = s
```

In [4]:
```
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]:
```
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]:
```
# Убедимся, что она действительно ортогональна
print(np.dot(U.T, U))
```

```
[[ 1.  0. -0. -0. -0.]
 [ 0.  1.  0. -0.  0.]
 [-0.  0.  1. -0. -0.]
 [-0. -0. -0.  1. -0.]
 [-0.  0. -0. -0.  1.]]
```

In [7]:
```
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]]
```

```python
In [8]: # Убедимся, что она действительно ортогональна
print(np.dot(V.T, V))
```

```
[[ 1.  0.  0.]
 [ 0.  1. -0.]
 [ 0. -0.  1.]]
```

```python
In [9]: print(f'Матрица A:\n{A}')
print(f'Матрица U:\n{U}')
print(f'Матрица D:\n{D}')
print(f'Матрица V:\n{V}')
```

```
Матрица A:
[[ 1  2  0]
 [ 0  0  5]
 [ 3 -4  2]
 [ 1  6  5]
 [ 0  1  0]]
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]]
Матрица D:
[[8.82 0.   0.  ]
 [0.   6.14 0.  ]
 [0.   0.   2.53]
 [0.   0.   0.  ]
 [0.   0.   0.  ]]
Матрица V:
[[ 0.07 -0.37 -0.93]
 [ 0.72  0.67 -0.21]
 [ 0.69 -0.65  0.31]]
```

```python
In [10]: # Проведем проверку
A_ = np.dot(np.dot(U, D), V.T)
print(f'Матрица A:\n{A}')
print(f'Матрица A_:\n{A_}')
```

```
Матрица A:
[[ 1  2  0]
 [ 0  0  5]
 [ 3 -4  2]
 [ 1  6  5]
 [ 0  1  0]]
Матрица A_:
[[ 1.  2.  0.]
 [ 0. -0.  5.]
 [ 3. -4.  2.]
 [ 1.  6.  5.]
 [-0.  1.  0.]]
```

2. Задача

Для матрицы из предыдущего задания найти:

а) евклидову норму;

б) норму Фробениуса.

Решение:

а) Евклидова норма: $\|A\|_E = \mu_1 = 8.82.$

```python
In [11]: # а) евклидова норма
norm_A_e = np.linalg.norm(A, ord=2, axis=None, keepdims=False)
print(f'Евклидова норма = {norm_A_e}')
```

```
Евклидова норма = 8.824868854820444
```

б) Норма Фробениуса: $\|A\|_F = \sqrt{\sum_{k=1}^{r} \mu_k^2} = \sqrt{(8.82^2 + 6.14^2 + 2.53^2)} = \sqrt{(77.7924 + 37.6996 + 6.4009)} = \sqrt{121.8929} = 11.041.$

```python
In [12]: # б) норма Фробениуса
norm_A_f = np.linalg.norm(A, ord='fro', axis=None, keepdims=False)
print(f'Норма Фробениуса = {norm_A_f}')
```

```
Норма Фробениуса = 11.045361017187261
```