Практическое задание Найдите посредством NumPy SVD матрицы

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

Для матрицы из предыдущего задания найдите:

а) евклидову норму;

б) норму Фробениуса.

In [1]:

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

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]]
```

Задача 1

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 наг
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]]
```

In [8]:

```
# Убедимся, что она действительно ортогональна
print(np.dot(V.T, V))
```

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

In [9]:

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

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

Задача 2

In [10]:

```
D[0]
```

Out[10]:

```
array([8.82, 0.  , 0.  ])
```

In [11]:

```
DL = D[0] +D[1] + D[2]
DL
```

Out[11]:

```
array([8.82, 6.14, 2.53])
```

In [12]:

```
NF = np.sqrt(np.dot(DL,DL))
NF
```

Out[12]:

```
11.045361017187263
```

In [ ]:

```

```