Практическое задание, семинар 2

In [3]:
```
import numpy as np
from numpy import linalg as LA
```

Задача 1

In [4]:
```
A = [[1, 1, 1, 1], [1, 2, 3, 4], [1, 4, 9, 16], [1, 8, 27, 64]]
```

In [7]:
```
# 1) Обратная матрица

B = LA.inv(A)
print(B)
```

```
[[ 4.         -4.33333333  1.5        -0.16666667]
 [-6.          9.5        -4.          0.5       ]
 [ 4.         -7.          3.5        -0.5       ]
 [-1.          1.83333333 -1.          0.16666667]]
```

In [9]:
```
print(A@B)
print("--------")
print(B@A)
```

```
[[ 1.00000000e+00  0.00000000e+00  0.00000000e+00  2.77555756e-17]
 [-3.55271368e-15  1.00000000e+00 -3.55271368e-15  4.44089210e-16]
 [ 0.00000000e+00  7.10542736e-15  1.00000000e+00  0.00000000e+00]
 [ 1.42108547e-14  1.42108547e-14  0.00000000e+00  1.00000000e+00]]
--------
[[ 1.00000000e+00  0.00000000e+00  5.21804822e-15  3.55271368e-15]
 [-3.33066907e-16  1.00000000e+00 -1.43218770e-14 -7.10542736e-15]
 [ 1.33226763e-15  3.55271368e-15  1.00000000e+00  1.42108547e-14]
 [-1.94289029e-16 -6.66133815e-16 -3.02535774e-15  1.00000000e+00]]
```

Проверим вычисления:

In [17]:
```
# Единичная матрица
I = [[1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1]]

np.allclose(A@B, I)
```

Out[17]:
```
True
```

In [18]:
```
np.allclose(B@A, I)
```

Out[18]:
```
True
```

In [11]:
```
# 2) Собственные значения и векторы

LA.eig(A)
```

Out[11]:
```
(array([7.15987322e+01, 3.61988880e+00, 7.16785074e-01, 6.45939334e-02]),
 array([[-0.01817783,  0.31773458, -0.71011705,  0.35330856],
        [-0.0665799 ,  0.52637506, -0.19851846, -0.73817342],
        [-0.25130845,  0.68928238,  0.63367794,  0.55538905],
        [-0.96544329, -0.38322817, -0.23404373, -0.1477026 ]]))
```

In [12]:
```
# 3) Определитель

LA.det(A)
```

Out[12]:
```
11.999999999999947
```

In [13]:
```
# 4) Число обусловленности

LA.cond(A)
```

Out[13]:
```
1171.0126859149357
```

Задача 2

In [70]:
```
B = [[1, 2], [0, 2]]
```

In [71]:
```
# 1) 1-norm
LA.norm(B, 1)
```

Out[71]: 4.0

In [72]:
```
#Checking:
print(max(1 + 0, 2 + 2))
```

```
4
```

In [73]:
```
# 2) 2-norm
```

In [74]:
```
LA.norm(B, 2)
```

Out[74]: 2.9208096264818897

In [88]:
```
#Checking:
import math

# С видео-лекции: 2-норма - максимум среди сингулярных чисел матрицы
evals, evecs = LA.eig(B)
uu, ss, vv = LA.svd(B)
print(max(ss))
```

```
2.9208096264818897
```

In [76]:
```
# 3) inf-norm
LA.norm(B, np.inf)
```

Out[76]: 3.0

In [77]:
```
#Checking:
print(max(1 + 2, 0 + 2))
```

```
3
```

Задача 3

In [54]:
```
B = [[7, 2, -5], [-9, 8, -5], [24, -6, 8]]
```

In [55]:
```
u, s, v = np.linalg.svd(B)
```

In [56]:
```
print(u, s, v)
```

```
[[-0.14943848 -0.85524101 -0.49621665]
 [ 0.41476859 -0.50978439  0.75371539]
 [-0.89757181 -0.09318099  0.43090851]] [28.91006204  8.80482678  3.26853761] [[-0.91043423  0.29071894 -0.29426519]
 [-0.41283849 -0.59395503  0.69049388]
 [ 0.02595936  0.75013327  0.66077696]]
```

Самостоятельная реализация алгоритма сингулярного разложения

In [57]:
```
# 1) B = u * s * v. Find "u"
W = B @ np.transpose(B)

# find eigenvectors and eigenvalues
evalues, evectors = LA.eig(W)

I = [[1, 0, 0], [0, 1, 0], [0, 0, 1]]
W_new = np.array(W) - evalues[0] * np.array(I)

#We have W_new * x = 0. Solve it:
b1 = [0, 0, 0]
x_1 = LA.solve(W_new, b1)
```

In [58]:
```
# We put evectors into the columns of "u"
u = evectors
```

In [66]:
```
# Practically the same we do to find "v"
W_2 = np.transpose(B) @ B
evalues_2, evectors_2 = LA.eig(W_2)
v_star = evectors_2
v = LA.inv(v_star)
```

In [67]:
```
#find s
s = [[math.sqrt(evalues[0]), 0, 0], [0, math.sqrt(evalues[1]), 0], [0, 0, math.sqrt(evalues[2])]]
```

In [68]:
```
print(u, s, v)
```

```
[[-0.14943848  0.85524101 -0.49621665]
 [ 0.41476859  0.50978439  0.75371539]
 [-0.89757181  0.09318099  0.43090851]] [[28.91006204293315, 0, 0], [0, 8.804826777354561, 0], [0, 0, 3.2685376079428288]] [[ 0.91043423 -0.29071894  0.29426519]
 [-0.41283849 -0.59395503  0.69049388]
 [ 0.02595936  0.75013327  0.66077696]]
```

In [ ]: