Алгебра матриц. Системы линейных уравнений

Матрицы

Матрицы в библиотеке numpy

Ввод [1]:

```
import numpy as np
```

Ввод [2]:

```
def Minor_elem(A,i, j):
    ''' Вычисляет минор элемента a_ij '''
    m,n = A.shape
    if m != n:
        raise ValueError('Матрица должна быть квадратной')
    if (0 < i <= n) & (0 < j <= n):
        A.row_del(i-1) # нумерация элементов массива с 0
        A.col_del(j-1)
    else:
        raise ValueError('индекс элемента больше размера матрицы')
    return(det(A))
```

Ввод [3]:

```
def Algebr_compl(A,i,j):
    m = Minor_elem(A,i,j)
    return (-l)**(i+j)*m
```

Ввод [4]:

```
def Algebr_compl(A,i,j):
    m = Minor_elem(A,i,j)
    return (-1)**(i+j)*m
```

Ввод [5]:

```
def Minor_Matrix(A,Row,Col):
    n = len(Row)
    m = len(Col)
    if n != m:
        raise ValueError('The quantities of the given \
        rows and columns must be equal')
    if (n < 1) or (n > A.shape[0]):
        raise ValueError('Invalid number of minor rows')
    M_Row = A.row(Row[0]-1)
    for i in range(1,n):
        M_Row = M_Row.row_insert(i,A.row(Row[i]-1))
    M_Col = M_Row.col(Col[0]-1)
    for j in range(1,m):
        M_Col = M_Col.col_insert(j,M_Row.col(Col[j]-1))
    return det(M_Col)
```

Ввод [6]:

```
def silvestr(A):

    m,n = A.shape
    if m!=n:
        raise ValueError('Матрица должна быть квадратной')
    Ml = A[0,0]
    if Ml == 0:
        return('Не является знакоопределенной')
    elif Ml > 0: # проверка на положительную определенность
        for k in range(2,n+1):
            Mk = det(A[0:k,0:k])
            if Mk <=0:
                return('Не является знакоопределенной')
        return('Положительно определена')
    else: # проверка на отрицательную определенность
        for k in range(2,n+1):
            Mk = det(A[0:k,0:k])
            if Mk == 0:
                return('Не является знакоопределенной')
            else:
                s1 = Ml/abs(Ml)
                s2 = Mk/abs(Mk)
                if s1*s2 > 0:
                    return('Не является знакоопределенной')
            Ml = Mk
        return('Отрицательно определена')
```

Создание матрицы.

Ввод [7]:

```
A = np.array([[-7,4,0],
             [0,-1,0],
             [-1,5,7]])
A
```

Out[7]:

```
array([[-7,  4,  0],
       [ 0, -1,  0],
       [-1,  5,  7]])
```

Ввод [8]:

```
'''Единичная матрица третьего порядка'''
E = np.eye(3)
E
```

Out[8]:

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

Ввод [9]:

```
''' Матрица-строка '''
A = np.array([1,2,3])
A
```

Out[9]:

```
array([1, 2, 3])
```

Элементы матрицы

Ввод [10]:

```
A = np.array([[-7,4,0],
[0,-1,0],
[-1,5,7]])
''' Элемент a12 (нумерация с 0) '''
A[0,1]
```

Out[10]:

```
4
```

Ввод [11]:

```
''' Первая строка '''
A[0]
```

Out[11]:

```
array([-7,  4,  0])
```

Ввод [12]:

```
''' Столбцы, начиная со второго
и заканчивая третьим '''
A[:,1:3]
```

Out[12]:

```
array([[ 4,  0],
       [-1,  0],
       [ 5,  7]])
```

Сложение, вычитание, умножение на число.

Ввод [13]:

```
A = np.array([[1,2,3],
              [4,5,6] ,
              [7,8,9] ])
E = np.eye(3)
A-E
```

Out[13]:

```
array([[0., 2., 3.],
       [4., 4., 6.],
       [7., 8., 8.]])
```

Ввод [14]:

```
3*A
```

Out[14]:

```
array([[ 3,  6,  9],
       [12, 15, 18],
       [21, 24, 27]])
```

Поэлементное умножение - операция *

Ввод [15]:

```
A = np.array([[1,2,3],
              [4,5,6] ]),
B = np.array([[0,0,0],
              [2,2,2]])
A * B
```

Out[15]:

```
array([[[ 0,  0,  0],
        [ 8, 10, 12]]])
```

Транспонирование - метод .T

Пример 1

Ввод [16]:

```
B = np.array([[1,0],
              [0,-2],
              [1,1] ])
B1 = B.T
B1
```

Out[16]:

```
array([[ 1,  0,  1],
       [ 0, -2,  1]])
```

Пример 2.

Ввод [17]:

```
A = np.array([[-7,4,0],
              [0,-1,0],
              [-1,5,7]])
B = np.array([[1,0],
              [0,-2],
              [1,1] ])
F = A@B
F
```

Out[17]:

```
array([[-7, -8],
       [ 0,  2],
       [ 6, -3]])
```

Определитель матрицы

Пример 3.

Ввод [18]:

```
A = np.array([[7,-3],
              [1,1]])
detA = np.linalg.det(A)
detA
```

Out[18]:

```
9.999999999999998
```

Пример 4.

Ввод [19]:

```
A = np.array([[7,-3],
              [1,1] ])
A1 = np.linalg.inv(A)
A1
```

Out[19]:

```
array([[ 0.1,  0.3],
       [-0.1,  0.7]])
```

Матрицы в библиотеке sympy

Ввод [20]:

```
''' Подключение функций библиотеки '''
from sympy import *
```

Создание матрицы

Ввод [21]:

```
a = Matrix([[1,2,3], [0,-1, 1]])
a
```

Out[21]:

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & -1 & 1 \end{bmatrix}$$

Может содержать символы - переменные:

Ввод [22]:

```
x,y,z = symbols('x y z')
v = Matrix([[1,x],[y,z]])
v
```

Out[22]:

$$\begin{bmatrix} 1 & x \\ y & z \end{bmatrix}$$

Ввод [23]:

```
''' Создание единичной матрицы '''
eye(3)
```

Out[23]:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Ввод [24]:

```
''' Создание нулевой матрицы '''
zeros(2,3)
```

Out[24]:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Ввод [25]:

```
'''Создание матрицы, все элементы которой равны 1'''
ones(3,2)
```

Out[25]:

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

Ввод [26]:

```
''' Создание диагональной матрицы '''
diag(1,5,-2)
```

Out[26]:

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

Ввод [27]:

```
''' Элементами диагонали могут быть матрицы '''
diag(-1, ones(2, 2), Matrix([5, 7, 5]))
```

Out[27]:

$$\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 5 \\ 0 & 0 & 0 & 7 \\ 0 & 0 & 0 & 5 \end{bmatrix}$$

Ввод [28]:

```
''' Матрица 1x3 (вектор-строка) '''
A = Matrix([[1,2,3]])
A
```

Out[28]:

$$\begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$$

Ввод [29]:

```
''' Матрица 3x1 (вектор-столбец)'''
A = Matrix([[1], [2],[3]])
A
```

Out[29]:

$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

Ввод [30]:

```
''' Отличие от правила в модуле numpy:
одна пара квадратных скобок приводит к созданию вектор-столбца '''
A = Matrix([1,2,3])
A
```

Out[30]:

$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

Пример 5. Матрица размера 2 X 3 (2 строки, 3 столбца):

Ввод [31]:

```
A = Matrix([[1,2,3], [0,-1, 1]])
A.shape
```

Out[31]:

```
(2, 3)
```

Пример 6. Матрица третьего порядка

Ввод [32]:

```
all, a12, a13, a21, a22, a23, a31, a32, a33 = \
symbols('a11 a12 a13 a21 a22 a23 a31 a32 a33')
A = Matrix([[all, a12, a13],
            [a21, a22, a23],
            [a31, a32, a33]])
A
```

Out[32]:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

Ввод [33]:

```
''' Элемент в третьей строке, в первом столбце (нумерация с 0'''
A[2,0]
```

Out[33]:

a_{31}

Ввод [34]:

```
''' Второй столбец '''
A[:, 1:2]
```

Out[34]:

$$\begin{bmatrix} a_{12} \\ a_{22} \\ a_{32} \end{bmatrix}$$

Методы получения строки и столбца - ,row(), .col()

Ввод [35]:

```
A = Matrix([[1,2,3],
[4,5,6] ,
[7,8,9] ])
''' Первая строка '''
A.row(0)
```

Out[35]:

$$\begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$$

Ввод [36]:

```
''' Второй столбец '''
A.col(1)
```

Out[36]:

$$\begin{bmatrix} 2 \\ 5 \\ 8 \end{bmatrix}$$

Пример 7

Ввод [37]:

```
A = Matrix([[1,2,3],
            [4,5,6] ,
            [7,8,9] ])
A
```

Out[37]:

$$\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 9\end{bmatrix}$$

Ввод [38]:

```
A.row_del(0)
A
```

Out[38]:

$$\begin{bmatrix}4 & 5 & 6\\7 & 8 & 9\end{bmatrix}$$

Ввод [39]:

```
A.col_del(1)
A
```

Out[39]:

$$\begin{bmatrix}4 & 6\\7 & 9\end{bmatrix}$$

Пример 8.

Ввод [40]:

```
B = Matrix([[1,2,3],
            [7,8,9] ])
A = B.row_insert(1, Matrix([[4,5,6]]))
A
```

Out[40]:

$$\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 9\end{bmatrix}$$

Ввод [41]:

```
''' Матрица B не изменилась '''
B
```

Out[41]:

$$\begin{bmatrix}1 & 2 & 3\\7 & 8 & 9\end{bmatrix}$$

Ввод [42]:

```
D = B.col_insert(3,Matrix([4,10]))
D
```

Out[42]:

$$\begin{bmatrix}1 & 2 & 3 & 4\\7 & 8 & 9 & 10\end{bmatrix}$$

Пример 9.

Ввод [43]:

```
V = Matrix([[1,x],[y,z]])
V*V
```

Out[43]:

$$\begin{bmatrix}xy + 1 & xz + x\\zy + y & xy + z^{2}\end{bmatrix}$$

Пример 10.

Ввод [44]:

```
a11, a12, a21, a22, a31, a32, b11, b12, b21, b22 = \
symbols('a11 a12 a21 a22 a31 a32 b11 b12 b21 b22')
A = Matrix([[a11, a12], [a21, a22], [a31, a32]])
B = Matrix([[b11, b12], [b21, b22]])
A*B
```

Out[44]:

$$\begin{bmatrix}a_{11} b_{11} + a_{12} b_{21} & a_{11} b_{12} + a_{12} b_{22}\\a_{21} b_{11} + a_{22} b_{21} & a_{21} b_{12} + a_{22} b_{22}\\a_{31} b_{11} + a_{32} b_{21} & a_{31} b_{12} + a_{32} b_{22}\end{bmatrix}$$

Пример 11.

Ввод [45]:

```
B = Matrix( [ [b11, b12], [b21, b22]])
B
```

Out[45]:

$$\begin{bmatrix}b_{11} & b_{12}\\b_{21} & b_{22}\end{bmatrix}$$

Ввод [46]:

```
B.T
```

Out[46]:

$$\begin{bmatrix} b_{11} & b_{21} \\ b_{12} & b_{22} \end{bmatrix}$$

Пример 12.

Ввод [47]:

```
D = Matrix([[0,1], [1,0]])
det(D)
```

Out[47]:

$$-1$$

Пример 13.

Ввод [48]:

```
x11, x12, x21, x22 = symbols('xll x12 x21 x22')
X = Matrix([[x11, x12], [x21, x22]])
det(X)
```

Out[48]:

$$-x_{12}x_{21} + xllx_{22}$$

Ввод [49]:

```
y11, y12, y13, y21, y22, y23, y31, y32, y33 = \
symbols('y11 y12 y13 y21 y22 y23 y31 y32 y33')
Y = Matrix([[y11, y12, y13], [y21, y22, y23], [y31, y32, y33]])
det(Y)
```

Out[49]:

$$y_{11}y_{22}y_{33} - y_{11}y_{23}y_{32} - y_{12}y_{21}y_{33} + y_{12}y_{23}y_{31} + y_{13}y_{21}y_{32} - y_{13}y_{22}y_{31}$$

Пример 14.

Ввод [50]:

```
A = Matrix( [[ 2,-3,-8],
             [-2,-1, 2],
             [ 1, 0,-3]] )
A.inv()
```

Out[50]:

$$\begin{bmatrix} \frac{3}{10} & -\frac{9}{10} & -\frac{7}{5} \\ -\frac{2}{5} & \frac{1}{5} & \frac{6}{5} \\ \frac{1}{10} & -\frac{3}{10} & -\frac{4}{5} \end{bmatrix}$$

Ввод [51]:

```
x11, x12, x21, x22 = symbols('x11 x12 x21 x22')
X = Matrix([[x11, x12], [x21, x22]])
X.inv()
```

Out[51]:

$$\begin{bmatrix} \frac{x_{22}}{x_{11}x_{22}-x_{12}x_{21}} & -\frac{x_{12}}{x_{11}x_{22}-x_{12}x_{21}} \\ -\frac{x_{21}}{x_{11}x_{22}-x_{12}x_{21}} & \frac{x_{11}}{x_{11}x_{22}-x_{12}x_{21}} \end{bmatrix}$$

Ввод [52]:

```
A = Matrix( [[ 2,-3,-8],
             [-2,-1, 2],
             [ 1, 0,-3]] )
A**-1
```

Out[52]:

$$\begin{bmatrix} \frac{3}{10} & -\frac{9}{10} & -\frac{7}{5} \\ -\frac{2}{5} & \frac{1}{5} & \frac{6}{5} \\ \frac{1}{10} & -\frac{3}{10} & -\frac{4}{5} \end{bmatrix}$$

Пример 15.

Ввод [53]:

```
A = Matrix([[2,4,5,6,0,4],
            [8,-2,0,2,4,-2],
            [6,-6,-5,-4,4,-6],
            [-4,0,2,-2,2,0],
            [-2,-4,-5,-6,0,-4],
            [0,1,0,1,0,1]])
A.rank()
```

Out[53]:

4

Пример 16.

Ввод [54]:

```
a = Matrix([[1,3,4,5,0],
            [4,-1,0,1,2],
            [3,2,5,5,3],
            [-2,0,1,-1,1],
            [4,6,7,11,-1]])
a.rank()
```

Out[54]:

3

Пример 17.

Ввод [55]:

```
A = Matrix([[1,3,4],
            [4,-1,0],
            [3,2,5],
            [-2,0,11],
            [4,6,7]])
A
```

Out[55]:

$$\begin{bmatrix} 1 & 3 & 4 \\ 4 & -1 & 0 \\ 3 & 2 & 5 \\ -2 & 0 & 11 \\ 4 & 6 & 7 \end{bmatrix}$$

Ввод [56]:

```
A.T.columnspace()
```

Out[56]:

```
[Matrix([
 [1],
 [3],
 [4]]),
 Matrix([
 [ 4],
 [-1],
 [ 0]]),
 Matrix([
 [3],
 [2],
 [5]])]
```

Ввод [57]:

```
A.T.rref()[1]
```

Out[57]:

```
(0, 1, 2)
```

Пример 18.

Ввод [58]:

```
A = Matrix([[1,0,-3,9],
[2,-7,11,5],
[-9,4,25,84],
[3,12,-5,58]])
''' Матрица после удаления 3-й строки и 2-го столбца'''
M = Matrix([[1,-3,9], [2,11,5], [3,-5,58]])
''' Определитель '''
det(M)
```

Out[58]:

$$579$$

Пример 19.

Ввод [59]:

```
A = Matrix([[1,0,-3,9],
            [2,-7,11,5],
            [-9,4,25,84],
            [3,12,-5,58]])
Algebr_compl(A, 3,2)
```

Out[59]:

$$-579$$

Пример 20.

Ввод [60]:

```
A = Matrix([[1,0,-3,9],
            [2,-7,11,5],
            [-9,4,25,84],
            [3,12,-5,58]])
Row = [1,3]
Col = [3,4]
Minor_Matrix(A,Row,Col)
```

Out[60]:

$$-477$$

Пример 21.

Ввод [61]:

```
A = Matrix([[1,3,2,4,5],
[0,0,-1,2,7],
[3,9,6,12,15],
[5,15,9,26,22],
[1,3,1,10,2]])
A
```

Out[61]:

$$\begin{bmatrix} 1 & 3 & 2 & 4 & 5 \\ 0 & 0 & -1 & 2 & 7 \\ 3 & 9 & 6 & 12 & 15 \\ 5 & 15 & 9 & 26 & 22 \\ 1 & 3 & 1 & 10 & 2 \end{bmatrix}$$

Ввод [62]:

```
A.rank()
```

Out[62]:

```
3
```

Системы линейных уравнений

Пример 22.

Ввод [63]:

```
A = np.array([[3, 2, 0],
[1, -1, 0],
[0, 5, 1]])
b = np.array([2, 4, -1])
u = np.linalg.solve(A,b)
u
```

Out[63]:

```
array([ 2., -2.,  9.])
```

Ввод [64]:

```
np.dot(A, u) == b
```

Out[64]:

```
array([ True,  True,  True])
```

Пример 23.

Ввод [65]:

```
A = Matrix([[3, 2, 0],
[1, -1, 0],
[0, 5, 1]])
b = Matrix([2, 4, -1])
x = A.inv()*b
x
```

Out[65]:

$$\begin{bmatrix} 2 \\ -2 \\ 9 \end{bmatrix}$$

Разложение вектора по системе векторов

Пример 24.

Ввод [66]:

```
F = Matrix([[2,-1],
            [6,2]])
a = Matrix([0,5])
x = F.inv()*a
x
```

Out[66]:

$$\begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}$$

Линейные системы уравнений произвольного вида. Общее решение системы

Пример 25.

Ввод [67]:

```
A = Matrix([[1,1,3],
[2,-1,9]])
A.rank()
```

Out[67]:

2

Ввод [68]:

```
x1,x2,x3 = symbols('xl x2 x3')

y1 = x1 + x2 + 3*x3 - 18
y2 = 2*x1 - x2 + 9*x3 - 30

linsolve([y1,y2], [x1,x2])
```

Out[68]:

$$\{(16 - 4x_3,\ x_3 + 2)\}$$

Ввод [69]:

```
A = Matrix([[1, 1, 3, 18],
[2, -1, 9, 30]])
A.rref()
```

Out[69]:

```
(Matrix([
 [1, 0,  4, 16],
 [0, 1, -1,  2]]),
 (0, 1))
```

Ввод [70]:

```
rref_matrix, rref_pivots = A.rref()
rref_matrix
```

Out[70]:

$$\begin{bmatrix} 1 & 0 & 4 & 16 \\ 0 & 1 & -1 & 2 \end{bmatrix}$$

Ввод [71]:

```
rref_pivots
```

Out[71]:

```
(0, 1)
```

Пример 26.

Ввод [72]:

```
A = Matrix([[1,-2,4],
[1,-2,1],
[-3,6,-12]])
A.rank()
```

Out[72]:

```
2
```

Ввод [73]:

```
Ab = Matrix([[1,-2,4,6],
[1,-2,1,4],
[-3,6,-12,-18]])
A.rank()
```

Out[73]:

2

Ввод [74]:

```
A = Matrix([[1,-2,4,6],
[1,-2,1,4],
[-3,6,-12,-18]])
rref_matrix, rref_pivots = A.rref()
rref_matrix
```

Out[74]:

$$\begin{bmatrix} 1 & -2 & 0 & \frac{10}{3} \\ 0 & 0 & 1 & \frac{2}{3} \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Ввод [75]:

```
rref_pivots
```

Out[75]:

(0, 2)

Пример 27.

Ввод [76]:

```
x, y, z = symbols('x, y, z')
A = Matrix([[1, 2, 3], [4, 5, 6], [7, 8, 10]])
b = Matrix([3, 6, 9])
A
```

Out[76]:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 10 \end{bmatrix}$$

Ввод [77]:

```
b
```

Out[77]:

$$\begin{bmatrix} 3 \\ 6 \\ 9 \end{bmatrix}$$

Ввод [78]:

```
linsolve((A, b), [x, y, z])
```

Out[78]:

$$\{(-1,\ 2,\ 0)\}$$

Пример 28.

Ввод [79]:

```
A = Matrix([[1, 2, 3],
[4, 5, 6],
[7, 8, 9]])
b = Matrix([3, 6, 9])
linsolve((A, b), x, y, z)
```

Out[79]:

$$\{(z-1,\ 2-2z,\ z)\}$$

Ввод [80]:

```
linsolve((A, b))
```

Out[80]:

$$\{(\tau_0 - 1,\ 2 - 2\tau_0,\ \tau_0)\}$$

Пример 29.

Ввод [81]:

```
Eqns = [3*x + 2*y - z - 1, 2*x - 2*y + 4*z + 2, -x + y/2 - z]
linsolve(Eqns, x, y, z)
```

Out[81]:

$$\{(1,\ -2,\ -2)\}$$

Пример 30.

Ввод [82]:

```
A = Matrix([[2, 1, 3, 1],
[2, 6, 8, 3],
[6, 8, 18, 5]])
linsolve(A, x, y, z)
```

Out[82]:

$$\left\{\left(\frac{3}{10},\ \frac{2}{5},\ 0\right)\right\}$$

Пример 31.

Ввод [83]:

```
a, b, c, d, e, f = symbols('a b c d e f')
eqns = [a*x + b*y - c, d*x + e*y - f]
linsolve(eqns, x, y)
```

Out[83]:

$$\left\{\left(\frac{-bf+ce}{ae-bd}, \frac{af-cd}{ae-bd}\right)\right\}$$

Однородные системы уравнений

Пример 32.

Ввод [84]:

```
A = Matrix([[1,-1,2],
[2,1,-3],
[3,0,2]])
''' Ранг матрицы системы '''
A.rank()
```

Out[84]:

```
3
```

Пример 33.

Ввод [85]:

```
A = Matrix([[1,2,3],
[4,5,6],
[7,8,9]])

A.rank()
```

Out[85]:

```
2
```

Ввод [86]:

```
A = Matrix( [[1,2,3],
[4,5,6],
[7,8,9]])
A.nullspace()
```

Out[86]:

```
[Matrix([
 [ 1],
 [-2],
 [ 1]])]
```

Пример 34.

Ввод [87]:

```
A = Matrix([[1,3,4, -2],
            [0,5,7,-4],
            [1,8,11,-6],
            [-1,2,3,-2]])

A.rank()
```

Out[87]:

2

Преобразование координат вектора при переходе к новому базису

Пример 35.

Ввод [88]:

```
x = Matrix([-2,3,1])
e1 = Matrix([1,2,-1])
e2 = Matrix([-2,0,3])
e3 = Matrix([-1,1,-1])
T = Matrix([e1.T,e2.T,e3.T]).T
T
```

Out[88]:

$$\begin{bmatrix} 1 & -2 & -1 \\ 2 & 0 & 1 \\ -1 & 3 & -1 \end{bmatrix}$$

Ввод [89]:

```
xn = T.inv()*x
xn
```

Out[89]:

$$\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Пример 36.

Ввод [90]:

```
T = Matrix([[3,-1],
[2,5]])
T1 = T.inv()
T1
```

Out[90]:

$$\begin{bmatrix} \frac{5}{17} & \frac{1}{17} \\ -\frac{2}{17} & \frac{3}{17} \end{bmatrix}$$

Ввод [91]:

```
A = Matrix([[2,-3],
[1,-4]])
T1*A*T
```

Out[91]:

$$\begin{bmatrix} -\frac{5}{17} & -\frac{106}{17} \\ -\frac{15}{17} & -\frac{29}{17} \end{bmatrix}$$

Собственные векторы

Пример 37.

Ввод [92]:

```
A = np.array([[3,6],
[1, 4]])
np.linalg.eig(A)
```

Out[92]:

```
(array([1., 6.]),
 array([[-0.9486833 , -0.89442719],
        [ 0.31622777, -0.4472136 ]]))
```

Ввод [93]:

```
L,V = np.linalg.eig(A)
L
```

Out[93]:

```
array([1., 6.])
```

Ввод [94]:

```
V[:,0]
```

Out[94]:

```
array([-0.9486833 ,  0.31622777])
```

Ввод [95]:

```
V[:,1]
```

Out[95]:

```
array([-0.89442719, -0.4472136 ])
```

В библиотеке sympy.

Ввод [96]:

```
A = Matrix([[3,6],
[1, 4]])
A.eigenvals()
```

Out[96]:

```
{6: 1, 1: 1}
```

Ввод [97]:

```
list(A.eigenvals().keys())
```

Out[97]:

```
[6, 1]
```

Пример 39.

Ввод [98]:

```
A = Matrix([[3,6],
[1, 4]])
A.eigenvects()
```

Out[98]:

```
[(1,
  1,
  [Matrix([
   [-3],
   [ 1]])]),
 (6,
  1,
  [Matrix([
   [2],
   [1]])])]
```

Ввод [99]:

```
A.eigenvects()[0][2]
```

Out[99]:

```
[Matrix([
 [-3],
 [ 1]])]
```

Ввод [100]:

```
[list(t[2][0]) for t in A.eigenvects()]
```

Out[100]:

```
[[-3, 1], [2, 1]]
```

Характеристический многочлен

Пример 40.

Ввод [101]:

```
A = Matrix([[3, -2, 4, -2],
           [5, 3, -3, -2],
           [5, -2, 2, -2],
           [5, -2, -3, 3]])
lamda = symbols('lamda')
p = A.charpoly(lamda)
p
```

Out[101]:

$$\operatorname{PurePoly}\left(\lambda^{4} - 11\lambda^{3} + 29\lambda^{2} + 35\lambda - 150, \lambda, domain = \mathbb{Z}\right)$$

Ввод [102]:

```
factor(p)
```

Out[102]:

$$\operatorname{PurePoly}\left(\lambda^{4} - 11\lambda^{3} + 29\lambda^{2} + 35\lambda - 150, \lambda, domain = \mathbb{Z}\right)$$

Ввод [103]:

```
A.eigenvals()
```

Out[103]:

```
{3: 1, -2: 1, 5: 2}
```

Приведение матрицы линейного оператора к диагональному виду

Пример 41.

Ввод [104]:

```
A = Matrix([[3, -2, 4, -2],
[5, 3, -3, -2],
[5, -2, 2, -2],
[5, -2, -3, 3]])
T, D = A.diagonalize()
```

Ввод [105]:

```
T
```

Out[105]:

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

Ввод [106]:

```
D
```

Out[106]:

$$\begin{bmatrix} -2 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 5 \end{bmatrix}$$

Ввод [107]:

```
T*D*T**-1
```

Out[107]:

$$\begin{bmatrix} 3 & -2 & 4 & -2 \\ 5 & 3 & -3 & -2 \\ 5 & -2 & 2 & -2 \\ 5 & -2 & -3 & 3 \end{bmatrix}$$

Квадратичные формы

Пример 43.

Ввод [108]:

```
A = Matrix([[-1,1,2],
            [1,-3,5],
            [2,5,-2]])
xl,x2,x3 = symbols('xl x2 x3')
```

Ввод [109]:

```
X = Matrix([[x1,x2,x3]])
Q = X*A*X.T
Q.simplify()
Q
```

Out[109]:

$$\left[-xl^2 + 2xl\mathrm{x}_2 + 4xl\mathrm{x}3 - 3\mathrm{x}_2^2 + 10\mathrm{x}_2\mathrm{x}3 - 2\mathrm{x}3^2 \right]$$

Пример 44

Ввод [110]:

```
A = Matrix( [[-1,1,2],
[1,-3,5],
[2,5,-2]])
silvestr(A)
```

Out[110]:

```
'Положительно определена'
```

Приведение квадратичной формы к каноническому виду

Пример 45.

Ввод [111]:

```
A = Matrix([[-2, 2],
            [2, 1]])

T,D = A.diagonalize()
T
```

Out[111]:

$$\begin{bmatrix} -2 & 1 \\ 1 & 2 \end{bmatrix}$$

Ввод [112]:

```
D
```

Out[112]:

$$\begin{bmatrix} -3 & 0 \\ 0 & 2 \end{bmatrix}$$

Пример 46.

Ввод [113]:

```
q = Matrix([30,60,40,80,50])
r = Matrix([5,3,7,2,4])
t = Matrix([7,10,8,15,8])
p = Matrix([45,20,50,25,30])
```

Ввод [114]:

```
R = q.T*r
R
```

Out[114]:

$$\left[\begin{matrix}970\end{matrix}\right]$$

Ввод [115]:

```
T = q.T*t
T
```

Out[115]:

$$\left[\begin{matrix}2730\end{matrix}\right]$$

Ввод [116]:

```
P = q.T*p
P
```

Out[116]:

$$\left[\begin{matrix}8050\end{matrix}\right]$$

Пример 47.

Ввод [117]:

```
Q = Matrix([[3,5,4,4,6],
[4,2,3,5,2] ,
[2,3,5,2,4],
[7,4,2,8,3] ])
N = Matrix([120,200,150,170,220]).T
B = Matrix([[4,2,6,3],
[3,1,4,5],
[2,5,4,2]])
p = Matrix([60,80,50]).T
```

Ввод [118]:

```
Qy = zeros(4,5)
for j in range(0,5):
    for i in range(0,4):
        Qy[i, j] = Q[i, j] * N[j]
Qy
```

Out[118]:

$$\left[\begin{matrix}360 & 1000 & 600 & 680 & 1320\\480 & 400 & 450 & 850 & 440\\240 & 600 & 750 & 340 & 880\\840 & 800 & 300 & 1360 & 660\end{matrix}\right]$$

Ввод [119]:

```
BQ = B*Q
BQ
```

Out[119]:

$$\begin{bmatrix} 53 & 54 & 58 & 62 & 61 \\ 56 & 49 & 45 & 65 & 51 \\ 48 & 40 & 47 & 57 & 44 \end{bmatrix}$$

Ввод [120]:

```
BQy = zeros(3,5)
for j in range(0,5):
    for i in range(0,3):
        BQy[i,j] = BQ[i,j]*N[j]
BQy
```

Out[120]:

$$\begin{bmatrix} 6360 & 10800 & 8700 & 10540 & 13420 \\ 6720 & 9800 & 6750 & 11050 & 11220 \\ 5760 & 8000 & 7050 & 9690 & 9680 \end{bmatrix}$$

Ввод [121]:

```
P = p*BQy
P
```

Out[121]:

$$\begin{bmatrix} 1207200 & 1832000 & 1414500 & 2000900 & 2186800 \end{bmatrix}$$

Примеры решения задач

1. Найти $A - A^T$, если $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$.

Ввод [122]:

```
A = Matrix([[1,2], [3,4]])
A - A.T
```

Out[122]:

$$\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

2. Найти AB и BA, если $A = \begin{pmatrix} 1 & 2 \\ 4 & -1 \end{pmatrix}$, $B = \begin{pmatrix} 2 & -3 \\ -4 & 1 \end{pmatrix}$.

Ввод [123]:

```
A = Matrix([[1,2], [4,-1]])
B = Matrix([[2,-3], [-4,1]])
A * B
```

Out[123]:

$$\begin{bmatrix} -6 & -1 \\ 12 & -13 \end{bmatrix}$$

Ввод [124]:

```
B * A
```

Out[124]:

$$\begin{bmatrix} -10 & 7 \\ 0 & -9 \end{bmatrix}$$

3. Квадрат ненулевой матрицы, в отличие от чисел, может быть нулевым. Проверить равенство: $\begin{pmatrix} 2 & 1 \\ -4 & -2 \end{pmatrix}^2 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$.

Ввод [125]:

```
A = Matrix([[2,1], [-4,-2]])
A**2
```

Out[125]:

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

4. Пусть $f(x) = x^3 - 5x^2 + 3x$, $A = \begin{pmatrix} 2 & -1 \\ -3 & 3 \end{pmatrix}$. Найти $f(A)$.

Ввод [126]:

```
x = symbols('x')
y = x**3 - 5*x**2 + 3*x
A = Matrix([[2,-1], [-3,3]])
y.subs(x,A)
```

Out[126]:

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

5. Вычислить $\begin{pmatrix} 2 & -1 \\ 3 & -2 \end{pmatrix}^n$.

Ввод [127]:

```
A = Matrix([[2,-1],
[3,-2]])
n = symbols('n')
A**n
```

Out[127]:

$$\begin{bmatrix} \frac{3}{2} - \frac{(-1)^n}{2} & \frac{(-1)^n}{2} - \frac{1}{2} \\ \frac{3}{2} - \frac{3(-1)^n}{2} & \frac{3(-1)^n}{2} - \frac{1}{2} \end{bmatrix}$$

7. Вычислить $\sqrt{A}$, где $A = \begin{pmatrix} 20 & -4 \\ 4 & 12 \end{pmatrix}$.

Ввод [128]:

```
A = Matrix([[20,-4],
[4,12]])
A**(1/2)
```

Out[128]:

$$\begin{bmatrix} 4.5 & -0.5 \\ 0.5 & 3.5 \end{bmatrix}$$

9. Вычислить определитель $\begin{vmatrix} 1 & -2 & 1 \\ 2 & 1 & 4 \\ 3 & 5 & 1 \end{vmatrix}$.

Ввод [129]:

```
A = Matrix([[1,-2,1], [2,1,4], [3,5,1]])
det(A)
```

Out[129]:

$$-32$$

10. Решить уравнение $\begin{vmatrix} x^2 - 4 & 4 \\ x - 2 & x + 2 \end{vmatrix} = 0.$

Ввод [130]:

```
x = symbols('x')
A = Matrix([[x**2-4, 4], [x-2, x+2]])
solve(det(A),x)
```

Out[130]:

```
[-4, 0, 2]
```

12. Найти $\left(A^{-1}\right)^T$ и $\left(A^T\right)^{-1}$, если $A = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 1 & 0 \\ 2 & 2 & 1 \end{pmatrix}$.

Ввод [131]:

```
A = Matrix([[1,0,-1], [2,1,0], [2,2,1]])
A.inv().T
```

Out[131]:

$$\begin{bmatrix} -1 & 2 & -2 \\ 2 & -3 & 2 \\ -1 & 2 & -1 \end{bmatrix}$$

Ввод [132]:

```
A.T.inv()
```

Out[132]:

$$\begin{bmatrix} -1 & 2 & -2 \\ 2 & -3 & 2 \\ -1 & 2 & -1 \end{bmatrix}$$

13. Найти ранг матрицы $\begin{pmatrix} 1 & 4 & 4 & 3 \\ 2 & 6 & 4 & 0 \\ 2 & -8 & 1 & 10 \\ 5 & 10 & 5 & 5 \end{pmatrix}$.

Ввод [133]:

```
A = Matrix([[1,4,4,3], [2,6,4,0], [2,-5,-3,2], [5,5,5,5]])
A.rank()
```

Out[133]:

```
3
```

16. Являются ли векторы $\boldsymbol{a} = (-1, 0, 1)$, $\boldsymbol{b} = (2, -1, 0)$ и $\boldsymbol{c} = (3, 2, -1)$ линейно независимыми?

Ввод [134]:

```
Matrix([[-1,0,1], [2,-1,0], [3,2,-1]]).rank()
```

Out[134]:

```
3
```

22. Решить систему уравнений $\begin{cases} 2x + 3y - z = 3 \\ 3x + 4y - 2z = 5 \\ x + 2y - 3z = 6 \end{cases}$

Ввод [135]:

```
A = np.array([[2, 3, -1],
[3, 4, -2],
[1, 2, -3]])
b = np.array([3, 5, 6])
w = np.linalg.solve(A, b)
w
```

Out[135]:

```
array([-0.33333333,  0.66666667, -1.66666667])
```

28. Найти общее и базисное решения системы $\begin{cases} x_1 + 3x_2 + 4x_3 - 2x_4 = -2, \\ 5x_2 + 7x_3 - 4x_4 = 4 \\ x_1 + 8x_2 + 11x_3 - 6x_4 = 2, \\ -x_1 + 2x_2 + 3x_3 - 2x_4 = 6. \end{cases}$

Ввод [136]:

```
A = Matrix([[1,3,4,-2],
[0,5,7,-4],
[1,8,11,-6],
[-1,2,3,-2]])
b = Matrix([-2,4,2,6])
Ab = Matrix([[1,3,4,-2,-2],
[0,5,7,-4,4],
[1,8,11,-6,2],
[-1,2,3,-2,6]])
```

Ввод [137]:

```
xl,x2,x3,x4, = symbols('xl x2 x3 x4')
gensolve = linsolve((A,b), [xl,x2,x3,x4])
gensolve
```

Out[137]:

$$\left\{\left(\frac{x_3}{5} - \frac{2x_4}{5} - \frac{22}{5},\ -\frac{7x_3}{5} + \frac{4x_4}{5} + \frac{4}{5},\ x_3,\ x_4\right)\right\}$$

41. Предприятие производит продукцию четырех видов P_1, P_2, P_3, P_4, и использует сырье пяти типов S_1, S_2, S_3, S_4, S_5. Нормы затрат сырья (по строкам) на единицу продукции каждого вида (по столбцам) заданы матрицей$A = \begin{pmatrix} 3 & 2 & 5 & 2 \\ 1 & 6 & 3 & 0 \\ 5 & 0 & 4 & 5 \\ 2 & 4 & 1 & 3 \\ 4 & 1 & 0 & 4 \end{pmatrix}$.Стоимость единицы сырья каждого типа S_i

задана матрицей $B = \begin{pmatrix} 25 & 10 & 18 & 20 & 15 \end{pmatrix}$.Каковы общие затраты предприятия на

Ввод [138]:

```
A = Matrix([[3,2,5,2],
[1,6,3,0],
[5,0,4,5],
[2,4,1,3] ,
[4,1,0,4]])
B = Matrix([25,10,18,20,15])
Q = Matrix([200,300,250,350])
```

Ввод [139]:

```
B
```

Out[139]:

$$\begin{bmatrix} 25 \\ 10 \\ 18 \\ 20 \\ 15 \end{bmatrix}$$

Ввод [140]:

```
P = B.T*A
P
```

Out[140]:

$$\begin{bmatrix} 275 & 205 & 247 & 260 \end{bmatrix}$$

Ввод [141]:

```
P*Q
```

Out[141]:

$$\begin{bmatrix} 269250 \end{bmatrix}$$

Задачи для самостоятельного решения

Задача 16:

Найти значение многочлена $f(x) = x^3 - 7x^2 + 13x - 5$ от матрицы $\begin{pmatrix} 5 & 2 & -3 \\ 1 & 3 & -1 \\ 2 & 2 & -1 \end{pmatrix}$.

Ввод [142]:

```
A = Matrix([
    [5, 2, -3],
    [1, 3, -1],
    [2, 2, -1]
])
E = eye(3)
A**3 - 7*A**2 + 13*A - 5*E
```

Out[142]:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Индивидуальное задание

Задача: Вы инженер-механик, которому поручено улучшить аэродинамику автомобиля. Первоначальный коэффициент аэродинамического сопротивления автомобиля составляет 0,445, и вы хотите максимально снизить его, чтобы улучшить топливную экономичность и производительность. Одним из способов снижения коэффициента сопротивления является изменение формы кузова автомобиля, включая угол наклона лобового стекла и расположение воздухозаборников и выхлопных труб.

Для моделирования аэродинамики автомобиля используется матричное уравнение, которое связывает силу сопротивления, действующую на автомобиль, с коэффициентом сопротивления и скоростью автомобиля: $F = \frac{1}{2} * rho * v^2 * A * Cd$

где F - сила сопротивления, действующая на автомобиль, rho - плотность воздуха, v - скорость автомобиля, A - лобовая площадь автомобиля, Cd - коэффициент сопротивления.

Чтобы оптимизировать аэродинамику автомобиля, можно варьировать параметры конструкции и вычислять результирующий коэффициент сопротивления для каждой комбинации параметров. Затем стоит использовать эти коэффициенты сопротивления для построения матричного уравнения, которое свяжет новый коэффициент сопротивления со старым коэффициентом сопротивления и изменениями параметров конструкции.

К параметрам конструкции, которые можно регулировать, относятся:

Угол наклона лобового стекла, который может изменяться от 20 до 30 градусов

Размер воздухозаборников, который можно варьировать от $0,2$ до $0,4$ квадратных метров

размер выхлопных труб, который может изменяться от $0,1$ до $0,2$ квадратных метров.

Начальный коэффициент сопротивления 0,445

Примечание: можно предположить, что частные производные уравнения сопротивления по отношению к параметрам конструкции постоянны и равны следующим значениям:

$$\frac{df(Cd)}{dWindshield} = -0,01$$

$$\frac{df(Cd)}{dAirIntake} = -0.005$$

$$\frac{df(Cd)}{dExhaust} = -0.0025$$

Решение:

Импорт необходимых библиотек

Ввод [143]:

```
import matplotlib.pyplot as plt
%matplotlib inline
```

Определение диапазонов параметров конструкции

Ввод [144]:

```
windshield_range = np.arange(20, 31, 1)
air_intake_range = np.arange(0.2, 0.41, 0.05)
exhaust_range = np.arange(0.1, 0.21, 0.05)
```

Определение констант Нам необходимо определить константы, которые мы будем использовать при расчете новых коэффициентов сопротивления.

Ввод [145]:

```
rho = 1.225 # kg/m^3
A = 2.5 # m^2
v = 100 * 1000 / 3600 # m/s
Cd_initial = 0.445
dCd_dWindshield = -0.01
dCd_dAirIntake = -0.005
dCd_dExhaust = -0.0025
```

Рассчитываем новые коэффициенты сопротивления для каждой комбинации параметров конструкции

Ввод [146]:

```
drag_coefficients = np.zeros((len(windshield_range), len(air_intake_range), len(exhaust_r
for i, windshield in enumerate(windshield_range):
    for j, air_intake in enumerate(air_intake_range):
        for k, exhaust in enumerate(exhaust_range):
            Cd_new = Cd_initial + dCd_dWindshield * (windshield - 25) + dCd_dAirIntake *
            F = 0.5 * rho * v**2 * A * Cd_new
            drag_coefficients[i, j, k] = Cd_new
```

Ввод [147]:

```
Cd_new
```

Out[147]:

0.39437500000000003

Ввод [148]:

```
F
```

Out[148]:

465.96197434413585

Найдем комбинацию параметров конструкции, при которой достигается наименьший коэффициент сопротивления

Ввод [149]:

```
i_min, j_min, k_min = np.unravel_index(np.argmin(drag_coefficients), drag_coefficients.sh
windshield_min = windshield_range[i_min]
air_intake_min = air_intake_range[j_min]
exhaust_min = exhaust_range[k_min]
Cd_min = drag_coefficients[i_min, j_min, k_min]
```

Ввод [150]:

```
windshield_min
```

Out[150]:

30

Ввод [151]:

```
air_intake_min
```

Out[151]:

0.39999999999999997

Ввод [152]:

```
exhaust_min
```

Out[152]:

```
0.20000000000000004
```

Ввод [153]:

```
Cd_min
```

Out[153]:

```
0.39437500000000003
```

Построение трехмерной диаграмму рассеяния коэффициентов сопротивления

Ввод [154]:

```
fig = plt.figure(figsize=(7, 7))
ax = fig.add_subplot(111, projection='3d')
X, Y, Z = np.meshgrid(windshield_range, air_intake_range, exhaust_range, indexing='ij')
ax.scatter(X, Y, Z, c=drag_coefficients.flatten(), cmap='viridis')
ax.set_xlabel('Угол наклона лобового стекла (градусы)')
ax.set_ylabel('Размер воздухозаборника (м^2)')
ax.set_zlabel('Размер выхлопной трубы (м^2)')
ax.set_title('Коэффициент сопротивления в зависимости от конструктивных параметров')
ax.text(25, air_intake_min, 0.22, f'минимальное Cd = {Cd_min:.4f}', color='red')
plt.show()
```