# Введение в высшую математику

## Урок 7. Видеоурок “Введение в линейную алгебру”

## Практическое задание к уроку 7

### 1 - часть

**Практическое задание №5**

5.1. Вектор – это частный случай матрицы 1xN и Nx1. Повторите материал для векторов, уделяя особое внимание умножению A·B.

Вычислите, по возможности не используя программирование: (5E)-1 где E – единичная матрица размера 5x5.

5.2. Вычислите определитель:

|1 2 3|
|4 0 6|
|7 8 9|

5.3.

1. Вычислите матрицу, обратную данной:
   |1 2 3|
   |4 0 6|
   |7 8 9|
2. Приведите пример матрицы 4x4, ранг которой равен 1.

5.4.
Вычислите скалярное произведение двух векторов:
(1, 5) и (2, 8)

5.5
(1, 5, 0), (2, 8, 7) и (7, 1.5, 3) Вычислите смешанное произведение трех векторов:

### 2 - часть

**Практическое задание №6**

1. Решите линейную систему:
$$\begin{bmatrix}1 & 2 & 3\\4 & 0 & 6\\7 & 8 & 9\end{bmatrix} * X = \begin{bmatrix}12\\2\\1\end{bmatrix}$$
2. Найдите псевдорешение:
   x + 2y – z = 1
   3x – 4y = 7
   8x – 5y + 2z = 12
   2x – 5z = 7
   11x +4y – 7z = 15
3. Сколько решений имеет линейная система:
$$\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 9\end{bmatrix} * X = \begin{bmatrix}12\\2\\1\end{bmatrix}$$

Если ноль – то измените вектор правой части так, чтобы система стала совместной, и решите ее.

4. Вычислите LU-разложение матрицы:
$$\begin{bmatrix}1 & 2 & 3\\2 & 16 & 21\\4 & 28 & 73\end{bmatrix}$$
5. Найдите нормальное псевдорешение недоопределенной системы:
   x + 2y – z = 1
   8x – 5y + 2z = 12
   Для этого определите функцию Q(x,y,z), равную норме решения, и найдите ее минимум.
6. Найдите одно из псевдорешений вырожденной системы:
$$\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 9\end{bmatrix} * X = \begin{bmatrix}2\\5\\11\end{bmatrix}$$

Попробуйте также отыскать и нормальное псевдорешение.

В [1]:
```
import numpy as np
```

### 1 - часть

# Задание 5.1

Вектор – это частный случай матрицы 1xN и Nx1. Повторите материал для векторов, уделяя особое внимание умножению A·B.

Вычислите, по возможности не используя программирование: $(5E)^{-1}$ где E – единичная матрица размера 5x5.

$$(5E)(5E)^{-1} = E$$

Из определения обратной матрицы $A \cdot A^{-1} = E$ и того что $E^{-1} = E$ и $det(5E) = 5$, следует

$$(5E) \cdot (5E)^{-1} = 5E \cdot \frac{1}{det(5E)} A' = 5E \cdot \frac{1}{5} A' = 5 \cdot \frac{1}{5} E \cdot A' = 5 \cdot \frac{1}{5} E \cdot E^{-1} = E$$

.

Следовательно матрица $A' = E$. Поэтому обратная матрица в нашем случае будет иметь вид $(5E)^{-1} = \frac{1}{5}E$

# Задание 5.2

Вычислите определитель:

|1 2 3|
|4 0 6|
|7 8 9|

$$defA = 1 \cdot (0 \cdot 9 - 8 \cdot 6) - 2 \cdot (4 \cdot 9 - 7 \cdot 6) + 3 \cdot (4 \cdot 8 - 7 \cdot 0) =$$
$$= -48 - 2(36 - 42) + 3(32 - 0) = -48 + 12 + 96 = 60$$

В [2]:
```
a = np.array([[1, 2, 3], [4, 0, 6], [7, 8 ,9]])
np.linalg.det(a)
```

Out[2]: 59.999999999999986

# Задание 5.3

1. Вычислите матрицу, обратную данной:

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 0 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

**Главный определитель $\Delta$**
$\Delta = 1 * (0 * 9 - 8 * 6) - 4 * (2 * 9 - 8 * 3) + 7 * (2 * 6 - 0 * 3) = 60$

Определитель отличен от нуля, следовательно, матрица является невырожденной и для нее можно найти обратную матрицу А.

**Обратная матрица** будет иметь следующий вид:

$$A^{-1} = \frac{1}{60} \begin{pmatrix} A_{11} & A_{21} & A_{31} \\ A_{12} & A_{22} & A_{32} \\ A_{13} & A_{23} & A_{33} \end{pmatrix}$$

где $Aij$ - алгебраичеякое дополнение.

**Транспонированная матрица**

$$A^T = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 0 & 8 \\ 3 & 6 & 9 \end{pmatrix}$$

Найдем алгебраические дополнения матрицы $A^T$
$A_{1,1} = (-1)^{1+1}(0 * 9 - 6 * 8) = -48$
$A_{1,2} = (-1)^{1+2}(2 * 9 - 3 * 8) = 6$
$A_{1,3} = (-1)^{1+3}(2 * 6 - 3 * 0) = 12$

$A_{2,1} = (-1)^{2+1}(4 * 9 - 6 * 7) = 6$
$A_{2,2} = (-1)^{2+2}(1 * 9 - 3 * 7) = -12$
$A_{2,3} = (-1)^{2+3}(2 * 6 - 3 * 4) = 6$

$A_{3,1} = (-1)^{3+1}(4 * 8 - 0 * 7) = 32$
$A_{3,2} = (-1)^{3+2}(1 * 8 - 2 * 7) = 6$
$A_{3,3} = (-1)^{3+3}(1 * 0 - 2 * 4) = -8$

**Обратная матрица**

$$A^{-1} = \frac{1}{60} \begin{pmatrix} -48 & 6 & 12 \\ 6 & -12 & 6 \\ 32 & 6 & -8 \end{pmatrix}$$

**Проверка**:

В [3]:
```
A=(np.matrix([[ 1,  2,  3],
 [ 4, 0, 6],
 [7, 8, 9]]))

print('A:\n', A,'\n')

A_1 = (np.matrix(
[[ -48,  6,  12],
 [ 6, -12, 6],
 [32, 6, -8]]))/60

print('A_1:\n',A_1,'\n')
print('A*A_1:\n',A*A_1)
```

```
A:
 [[1 2 3]
 [4 0 6]
 [7 8 9]]

A_1:
 [[-0.8         0.1         0.2       ]
 [ 0.1        -0.2         0.1       ]
 [ 0.53333333  0.1        -0.13333333]]

A*A_1:
 [[ 1.00000000e+00 -2.77555756e-17  2.77555756e-17]
 [-2.22044605e-16  1.00000000e+00  5.55111512e-17]
 [-7.77156117e-16  2.77555756e-17  1.00000000e+00]]
```

В [4]:
```
A_2 = np.linalg.inv(A)
print('A_2:\n', A_2)
```

```
A_2:
 [[-0.8         0.1         0.2       ]
 [ 0.1        -0.2         0.1       ]
 [ 0.53333333  0.1        -0.13333333]]
```

2. Приведите пример матрицы 4x4, ранг которой равен 1.

В [5]:
```
A=(np.matrix([[ -1/2,  -1,  3/2, 0],
 [ 1, 2, -3, 0],
 [ 4, 8, -12, 0],
 [ -1, -2, 3, 0]]))

A
```

Out[5]:
```
matrix([[ -0.5,  -1. ,   1.5,   0. ],
        [  1. ,   2. ,  -3. ,   0. ],
        [  4. ,   8. , -12. ,   0. ],
        [ -1. ,  -2. ,   3. ,   0. ]])
```

## Задание 5.4

Вычислите скалярное произведение двух векторов:
(1, 5) и (2, 8)

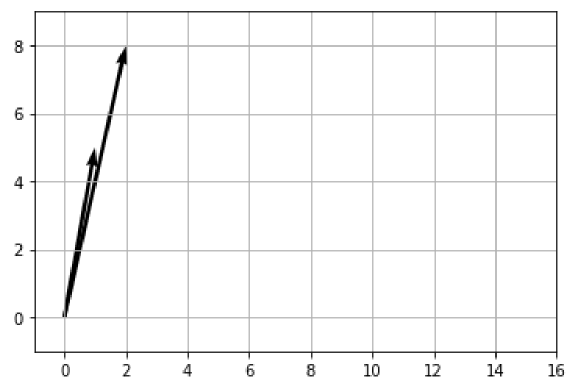
$$\begin{pmatrix} 1 & 5 \end{pmatrix} \begin{pmatrix} 2 \\ 8 \end{pmatrix} = 1 \cdot 2 + 5 \cdot 8 = 42$$

В [6]:
```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
```

В [7]:
```
i = np.array([1, 5], float)
j = np.array([2, 8], float)
X, Y = np.array([0, 0]), np.array([0, 0])
U, V = np.array([i[0], j[0]]), np.array([i[1], j[1]])

plt.quiver(X, Y, U, V, angles='xy', scale_units='xy', scale=1)

plt.xlim(-1, 16)
plt.ylim(-1, 9)
plt.grid(True)
plt.show()
np.inner(i, j)
```




Out[7]: 42.0

## Задание 5.5

(1, 5, 0), (2, 8, 7) и (7, 1.5, 3) Вычислите смешанное произведение трех векторов:

$a = \begin{pmatrix} 1 & 5 & 0 \end{pmatrix}$

$b = \begin{pmatrix} 2 & 8 & 7 \end{pmatrix}$

$c = \begin{pmatrix} 7 & 1.5 & 3 \end{pmatrix}$

$$\vec{a} \times \vec{b} = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix}$$

$\vec{a} \times \vec{b} = -\Sigma_{j,k=1}^{3} e_{ijk} \cdot a_j \cdot b_k$ , где $e_{ijk}$ - символ Леви-Чивиты

$\vec{a} \times \vec{b} = (a_y b_z - a_z b_y, a_z b_x - a_x b_z, a_x b_y - a_y b_x) = (5 \cdot 7 - 0 \cdot 2; 0 \cdot 2 - 1 \cdot 7; 1 \cdot 8 - 5 \cdot 2) = (35, -7, -2)$

$(\vec{a} \times \vec{b}) \cdot c = 35 \cdot 7 + (-7 \cdot 1.5) + (-2 \cdot 3) = 228.5$

В [8]:
```
35*7-(7*1.5)-(2*3)
```

Out[8]:
```
228.5
```

$$a * b = \begin{pmatrix} 1 & 5 & 0 \end{pmatrix} \begin{pmatrix} 2 \\ 8 \\ 7 \end{pmatrix} = 1 \cdot 2 + 5 \cdot 8 + 0 \cdot 7 = 42$$

$$a * b * c = 42 * \begin{pmatrix} 7 & 1.5 & 3 \end{pmatrix} = \begin{pmatrix} 494 & 10.5 & 126 \end{pmatrix}$$

**Смешанное произведение**

$$(\vec{a} \times \vec{b}) \cdot c = a \cdot (\vec{b} \times \vec{c})$$

В [9]:
```
a = np.array([1, 5, 0], float)
b = np.array([2, 8, 7], float)
c = np.array([7, 1.5, 3], float)
v = np.cross(a, b)
print(v)
print (np.inner(v, c))
w = np.cross(b, c)
print (np.inner(w, a))
```

```
[35. -7. -2.]
228.5
228.5
```

# 2 - часть

## Практическое задание №6

# Задание 1

Решите линейную систему:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 0 & 6 \\ 7 & 8 & 9 \end{bmatrix} * X = \begin{bmatrix} 12 \\ 2 \\ 1 \end{bmatrix}$$

В [10]:
```
A = np.array([[1, 2, 3], [4, 0, 6], [7, 8, 9]])
B = np.array([12, 2, 1])
np.linalg.solve(A, B)
```

Out[10]:
```
array([-9.2       ,  0.9       ,  6.46666667])
```

# Задание 2

Найдите псевдорешение:
x + 2y – z = 1
3x – 4y = 7
8x – 5y + 2z = 12
2x – 5z = 7
11x +4y – 7z = 15

В [49]:
```
import numpy as np
import matplotlib.pyplot as plt

# переопределенное СЛАУ
A = np.array([[1, 2, -1], [3, -4, 0], [8, -5, 2], [2, 0, -5], [11, 4, -7]], float)
B = np.array([1, 7, 12, 7, 15])
np.linalg.lstsq(A, B)[0]
```

```
<ipython-input-49-38a69aeaff98>:7: FutureWarning: `rcond` parameter will change to the default of machine precision times ``max(M, N)`` where M and N are the input matrix dimensions.
To use the future default and silence this warning we advise to pass `rcond=None`, to keep using the old, explicitly pass `rcond=-1`.
  np.linalg.lstsq(A, B)[0]
```

Out[49]:
```
array([ 1.13919353, -0.90498444, -0.9009803 ])
```

В [35]:
```
import numpy as np
from numpy import linalg as LA
print(A)

# Функция linalg.pinv() вычисляет псевдообратную матрицу (Мура-Пенроуза).
a_pinv = LA.pinv(A)
print(a_pinv)
```

```
[[ 1.  2. -1.]
 [ 3. -4.  0.]
 [ 8. -5.  2.]
 [ 2.  0. -5.]
 [11.  4. -7.]]
[[ 0.0184525  -0.01309971  0.06358533 -0.05827435  0.0571557 ]
 [ 0.05588372 -0.11181622 -0.03160448 -0.12284939  0.07073632]
 [ 0.03245479 -0.06855478  0.06726537 -0.17860416 -0.00070046]]
```

Псевдорешение $X' = A^{'} b$

В [34]:
```
X_1=np.dot(a_pinv, B)
print(f'Псевдорешение X_1:\n{X_1}')
```

```
Псевдорешение X_1:
[ 1.13919353 -0.90498444 -0.9009803 ]
```

## Задание 3

Сколько решений имеет линейная система:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} * X = \begin{bmatrix} 12 \\ 2 \\ 1 \end{bmatrix}$$

Если ноль – то измените вектор правой части так, чтобы система стала совместной, и решите ее.

В [37]:
```
import numpy as np
import matplotlib.pyplot as plt

# print(np.__version__)

A = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
B = np.array([[12, 2, 1]])
C = np.concatenate((A,B.T), axis=1)

print (A)
print (C)
np.linalg.matrix_rank(A, 0.0001), np.linalg.matrix_rank(C, 0.0001)
np.linalg.matrix_rank(A), np.linalg.matrix_rank(C)
```

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
[[ 1  2  3 12]
 [ 4  5  6  2]
 [ 7  8  9  1]]
```

Out[37]:
```
(2, 3)
```

Теорема Кронекера_Каппели утверждает, что в этом случае решения нет т. к. $rank A \neq rank \mathrm{C}$

В [13]:
```
# Подбираем вектор B
B = np.array([[1, 1, 1]])

C = np.concatenate((A,B.T), axis=1)
print (C)
np.linalg.matrix_rank(A, 0.0001), np.linalg.matrix_rank(C, 0.0001)
# np.linalg.matrix_rank(A), np.linalg.matrix_rank(C)
```

```
[[1 2 3 1]
 [4 5 6 1]
 [7 8 9 1]]
```

Out[13]:
```
(2, 2)
```

В [14]:
```
# np.linalg.solve(A, B)
```

## Задание 4

Вычислите LU-разложение матрицы:

$$\begin{bmatrix} 1 & 2 & 3 \\ 2 & 16 & 21 \\ 4 & 28 & 73 \end{bmatrix}$$

В [15]:
```
import scipy
import scipy.linalg

A = np.array([ [1, 2, 3], [2, 16, 21], [4, 28, 73]])
P, L, U = scipy.linalg.lu(A)

print(P)
print(L)
print(U)

print(np.dot(P.transpose(), A) - np.dot(L, U))
```

```
[[0. 1. 0.]
 [0. 0. 1.]
 [1. 0. 0.]]
[[ 1.    0.    0.  ]
 [ 0.25  1.    0.  ]
 [ 0.5  -0.4   1.  ]]
[[  4.    28.    73.  ]
 [  0.    -5.   -15.25]
 [  0.     0.   -21.6 ]]
[[0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]]
```

В [ ]:

## Задание 5

Найдите нормальное псевдорешение недоопределенной системы:
x + 2y – z = 1
8x – 5y + 2z = 12
Для этого определите функцию Q(x,y,z), равную норме решения, и найдите ее минимум.

Из первого выражаем x
$x = 1 + z - 2y$
x подставляем во второе и находим y:
$y = \frac{1}{21}(10z - 4)$
y подставляем в уравнение для x и находим его
$x = 1 + z - 2(\frac{1}{21}(10z - 4)) = \frac{1}{21}(29 + z)$

Решение:

$$X = \begin{pmatrix} \frac{1}{21}(29+z) \\ \frac{1}{21}(10z-4) \\ z \end{pmatrix}$$

**нормальное псевдорешение недоопределенной системы**

Необходимо найти решение удовлетворяющее СЛАУ и обладающее минимальной нормой.

Модулем (нормой) столбца $x = \begin{pmatrix} x_1 & x_2 & \dots & x_n \end{pmatrix}^T$ называется неотрицательное число

$|X| = \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}$

$|X| = \sqrt{\frac{1}{21}(29+z)^2 + \frac{1}{21}(10z-4)^2 + z^2}$

В [52]:
```
z=0
```

Решение:

$$X = \begin{pmatrix} \frac{1}{21}(29) \\ \frac{1}{21}(4) \\ 0 \end{pmatrix}$$

В [47]:
```
A = np.array([[1, 2, -1], [8, -5, 2]])
B = np.array([1, 12])
np.linalg.lstsq(A, B)
```

```
<ipython-input-47-d93be9187394>:3: FutureWarning: `rcond` parameter will change to the default of machine precision times ``max(M, N)`` where M and N are the input matrix dimensions.
To use the future default and silence this warning we advise to pass `rcond=None`, to keep using the old, explicitly pass `rcond=-1`.
  np.linalg.lstsq(A, B)
```

Out[47]:
```
(array([ 1.38191882, -0.18081181,  0.0202952 ]),
 array([], dtype=float64),
 2,
 array([9.65316119, 2.41173777]))
```

В [ ]:

В [50]:
```
# Решить в матричной форме Ax=b
# (в этом случае недоопределенная система, но мы можем использовать solve_linear_system):
from sympy import Matrix, solve_linear_system
from sympy import symbols

x, y, z = symbols('x, y, z')
A = Matrix(( (1, 2, -1, 1), (8, -5, 2, 12) ))
solve_linear_system(A, x, y, z)
```

Out[50]:
```
{y: 10*z/21 - 4/21, x: z/21 + 29/21}
```

## Задание 6

Найдите одно из псевдорешений вырожденной системы:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} * X = \begin{bmatrix} 2 \\ 5 \\ 11 \end{bmatrix}$$

Попробуйте также отыскать и нормальное псевдорешение.

В [21]:
```
A = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]], float)
B = np.array([2, 5, 11])
np.linalg.lstsq(A, B)[0]
```

```
<ipython-input-21-e974c8fd472d>:3: FutureWarning: `rcond` parameter will change to the default of machine precision times ``max(M, N)`` where M and N are the input matrix dimensions.
To use the future default and silence this warning we advise to pass `rcond=None`, to keep using the old, explicitly pass `rcond=-1`.
  np.linalg.lstsq(A, B)[0]
```

Out[21]:
```
array([ 1.25,  0.5 , -0.25])
```

**линейная-алгебра - Найти псевдорешения несовместной СЛУ**
http://math.hashcode.ru/questions/40157/линейная-алгебра-найти-псевдорешения-несовместной-слу
(http://math.hashcode.ru/questions/40157/%D0%BB%D0%B8%D0%BD%D0%B5%D0%B9%D0%BD%D0%B0%D1%8F-%D0%B0%D0%BB%D0%B3%D0%B5%D0%B1%D1%80%D0%B0-%D0%BD%D0%B0%D0%B9%D1%82%D0%B8-%D0%BF%D1%81%D0%B5%D0%B2%D0%B4%D0%BE%D1%80%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D1%8F-%D0%BD%D0%B5%D1%81%D0%BE%D0%B2%D0%BC%D0%B5%D1%81%D1%82%D0%BD%D0%BE%D0%B9-%D1%81%D0%BB%D1%83)

Дана несовместная система уравнений Ax = b. Описать все псевдорешения СЛУ. Найти нормальное псевдорешение этой системы линейных уравнений.

В [ ]: