В файле есть картинки с решениями и заданиями, но онлайн просмотрщик на гитхабе их не отображает. Необходимо склонировать репозиторий.

In [1]:

```
import numpy as np
```

Задания к уроку №6

1. **Решите линейную систему**:

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

In [2]:

```
M_6_1 = np.array([[1,2,3],[4,0,6],[7,8,9]])
B = np.array([12,2,1])
```

In [3]:

```
solve_1 = np.linalg.solve(M_6_1, B)
solve_1
```

Out[3]:

```
array([-9.2       ,  0.9       ,  6.46666667])
```

In [4]:

```
M_6_1_inv = np.linalg.inv(M_6_1)
M_6_1_inv
```

Out[4]:

```
array([[-0.8       ,  0.1       ,  0.2       ],
       [ 0.1       , -0.2       ,  0.1       ],
       [ 0.53333333,  0.1       , -0.13333333]])
```

In [5]:

```
solve_2 = np.dot(M_6_1_inv, B)
solve_2
```

Out[5]:

```
array([-9.2       ,  0.9       ,  6.46666667])
```

In [6]:

```
M_6_1 @ solve_1
```

Out[6]:

```
array([12.,  2.,  1.])
```

In [7]:

```
M_6_1 @ solve_2
```

Out[7]:

```
array([12.,  2.,  1.])
```

2. **Найдите псевдорешение**:

$$x + 2y - z = 1$$
$$3x - 4y = 7$$
$$8x - 5y + 2z = 12$$
$$2x - 5z = 7$$
$$11x + 4y - 7z = 15$$

In [8]:

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

In [9]:

```
xyz_answer = np.linalg.lstsq(A,B, rcond=None)
xyz_answer
```

Out[9]:

```
(array([ 1.13919353, -0.90498444, -0.9009803 ]),
 array([0.71523211]),
 3,
 array([15.2817306 ,  9.59852942,  3.65197794]))
```

In [10]:

```
def Q(x, y, z):
    return ((np.linalg.norm(np.dot(A, [x, y, z]) - B))**2)
```

In [11]:

```
Q(*xyz_answer[0])
```

Out[11]:

```
0.7152321111819713
```

3. **Сколько решений имеет линейная система**:

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

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

In [12]:

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

In [13]:

```
C = np.concatenate((A,B.T), axis=1)
C
```

Out[13]:

```
array([[ 1,  2,  3, 12],
       [ 4,  5,  6,  2],
       [ 7,  8,  9,  1]])
```

In [14]:

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

Out[14]:

```
(2, 3)
```

В данный момент система имеет ноль решений, так как она несовместна.
Изменим 'B' так чтобы ранг основной и расширенной матрицы стали равны количеству неизвестных:

In [15]:

```
B = np.array([[2,2,2]])
C = np.concatenate((A,B.T), axis=1)
print(C)
np.linalg.matrix_rank(A), np.linalg.matrix_rank(C)
```

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

Out[15]:

```
(2, 2)
```

Получившаяся совместная СЛАУ имеет бесконечное множество решений, так как ранг меньше числа неизвестных.

4. **Вычислите LU-разложение матрицы**:

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

После этого придумайте вектор правых частей и решите полученную линейную систему трех уравнений с данной матрицей.

In [16]:

```
A = np.array([[1,2,3],[2,16,21],[4,28,73]])
A
```

Out[16]:

```
array([[ 1,  2,  3],
       [ 2, 16, 21],
       [ 4, 28, 73]])
```

In [17]:

```
import scipy.linalg
```

In [18]:

```
P, L, U = scipy.linalg.lu(A)
```

In [19]:

```
P
```

Out[19]:

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

In [20]:

```
L
```

Out[20]:

```
array([[ 1.  ,  0.  ,  0.  ],
       [ 0.25,  1.  ,  0.  ],
       [ 0.5 , -0.4 ,  1.  ]])
```

In [21]:

```
U
```

Out[21]:

```
array([[  4.  ,  28.  ,  73.  ],
       [  0.  ,  -5.  , -15.25],
       [  0.  ,   0.  , -21.6 ]])
```

In [22]:

```
B = np.array([3,12,48])
B
```

Out[22]:

```
array([ 3, 12, 48])
```

In [23]:

```
np.linalg.solve(A, B)
```

Out[23]:

```
array([ 1.63888889, -0.40277778,  0.72222222])
```

In [24]:

```
np.dot(P, A)
```

Out[24]:

```
array([[ 2., 16., 21.],
       [ 4., 28., 73.],
       [ 1.,  2.,  3.]])
```

In [25]:

```
A2 = np.dot(L, U)
A2
```

Out[25]:

```
array([[ 4., 28., 73.],
       [ 1.,  2.,  3.],
       [ 2., 16., 21.]])
```

In [26]:

```
np.linalg.solve(A2, B)
```

Out[26]:

```
array([ 9.18055556,  4.95138889, -2.36111111])
```

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.

In [27]:

```
A = np.array([[1,2,-1],[8,-5,2]])
A
```

Out[27]:

```
array([[ 1,  2, -1],
       [ 8, -5,  2]])
```

In [28]:

```
B = np.array([1,12])
B
```

Out[28]:

```
array([ 1, 12])
```

In [29]:

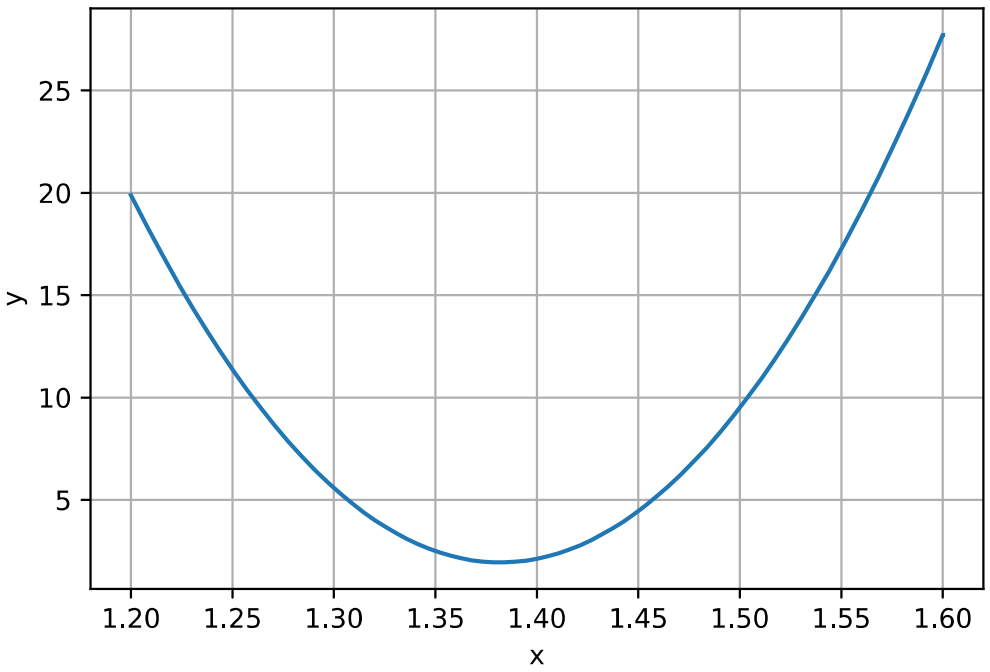
```
def Q(x,y,z):
    return (x**2 + y**2 + z**2)
```

In [30]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
%config InlineBackend.figure_format = 'svg'
import seaborn as sns
```

In [31]:

```
x = np.linspace(-1, 4, 301)
x = np.linspace(1.2, 1.6, 301)
plt.plot(x, Q(x, 10*x - 14, 6 - 4*x + 2.5 * (10 * x - 14)))
plt.xlabel('x')
plt.ylabel('y')
plt.grid(True)
plt.show()
```



In [32]:

```
solve_answer = np.linalg.lstsq(A, B, rcond=None)
solve_answer
```

Out[32]:

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

In [33]:

```
np.dot(A, solve_answer[0])
```

Out[33]:

```
array([ 1., 12.])
```

6. **Найдите одно из псевдорешений вырожденной системы**:

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

Попробуйте также отыскать и нормальное псевдорешение.

In [34]:

```
A = np.array([[1,2,3],[4,5,6],[7,8,9]])
B = np.array([2,5,1])
print("A:", A, sep="\n")
print("B:", B, sep="\n")
```

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

In [35]:

```
Q, R = np.linalg.qr(A)
print("Q:", Q, sep="\n")
print("R:", R, sep="\n")
```

```
Q:
[[-0.12309149  0.90453403  0.40824829]
 [-0.49236596  0.30151134 -0.81649658]
 [-0.86164044 -0.30151134  0.40824829]]
R:
[[-8.12403840e+00 -9.60113630e+00 -1.10782342e+01]
 [ 0.00000000e+00  9.04534034e-01  1.80906807e+00]
 [ 0.00000000e+00  0.00000000e+00 -1.11164740e-15]]
```

In [36]:

```
np.dot(Q, R)
```

Out[36]:

```
array([[1., 2., 3.],
       [4., 5., 6.],
       [7., 8., 9.]])
```

In [37]:

```
R1 = R[:2, :2]
print("R1:", R1, sep="\n")
```

```
R1:
[[-8.1240384  -9.6011363 ]
 [ 0.          0.90453403]]
```

In [38]:

```
B1 = np.dot(Q.T, B)[:2]
print("B1:", B1, sep="\n")
```

```
B1:
[-3.56965324  3.01511345]
```

In [39]:

```
X1 = np.linalg.solve(R1, B1)
print("X1:", X1, sep="\n")
```

```
X1:
[-3.5         3.33333333]
```

In [40]:

```
X = np.append(X1, 0)
print("X:", X, sep="\n")
```

```
X:
[-3.5         3.33333333  0.        ]
```

In [41]:

```
np.linalg.norm(X)
```

Out[41]:

```
4.833333333333305
```

In [42]:

```
np.linalg.norm(np.dot(A, X) - B)
```

Out[42]:

```
2.857738033247041
```

In [43]:

```
lstsq_answer = np.linalg.lstsq(A, B, rcond=None)
lstsq_answer
```

Out[43]:

```
(array([-1.80555556, -0.05555556,  1.69444444]),
 array([], dtype=float64),
 2,
 array([1.68481034e+01, 1.06836951e+00, 3.33475287e-16]))
```

In [44]:

```
np.linalg.norm(lstsq_answer[0]), np.linalg.norm(np.dot(A, lstsq_answer[0]) - B)
```

Out[44]:

```
(2.4767436805731915, 2.8577380332470415)
```

In [ ]:

```
```