Практические задания к лекции № 4 часть 2

In [1]:
```
import numpy as np
```

1. Решить систему уравнений методом Крамера:

а) $\begin{cases} x_1 - 2x_2 = 1 \\ 3x_1 - 4x_2 = 7 \end{cases}$

б) $\begin{cases} 2x_1 - x_2 + 5x_3 = 10 \\ x_1 + x_2 - 3x_3 = -2 \\ 2x_1 + 4x_2 + x_3 = 1 \end{cases}$

Решение:
а)

In [2]:
```
a = np.array([[1, -2], [3, -4]])
print(f'Определитель:\n{np.linalg.det(a):.0f}')
```

```
Определитель:
2
```

Следовательно, система совместна.

Найдем определители $detA_1$, $detA_2$:

In [3]:
```
a_1 = np.array([[1, -2], [7, -4]])
print(f'Определитель:\n{np.linalg.det(a_1):.0f}')
```

```
Определитель:
10
```

In [4]:
```
a_2 = np.array([[1, 1], [3, 7]])
print(f'Определитель:\n{np.linalg.det(a_2):.0f}')
```

```
Определитель:
4
```

Найдем решение по формулам Крамера:

In [5]:
```
x_1 = 10 / 2
x_1
```

Out[5]:
```
5.0
```

In [6]:
```
x_2 = 4 / 2
x_2
```

Out[6]:
```
2.0
```

Ответ:

$$x_1 = 5,$$

$$x_2 = 2.$$

б) $\begin{cases} 2x_1 - x_2 + 5x_3 = 10 \\ x_1 + x_2 - 3x_3 = -2 \\ 2x_1 + 4x_2 + x_3 = 1 \end{cases}$

In [7]:
```
a = np.array([[2, -1, 5], [1, 1, -3], [2, 4, 1]])
print(f'Определитель:\n{np.linalg.det(a):.0f}')
```

```
Определитель:
43
```

Следовательно, система совместна.

Найдем определители $detA_1$, $detA_2$, $detA_3$:

Найдем определители det_1, det_2, det_3:

In [8]:
```
a_1 = np.array([[10, -1, 5], [-2, 1, -3], [1, 4, 1]])
print(f'Определитель:\n{np.linalg.det(a_1):.0f}')
```

```
Определитель:
86
```

In [9]:
```
a_2 = np.array([[2, 10, 5], [1, -2, -3], [2, 1, 1]])
print(f'Определитель:\n{np.linalg.det(a_2):.0f}')
```

```
Определитель:
-43
```

In [10]:
```
a_3 = np.array([[2, -1, 10], [1, 1, -2], [2, 4, 1]])
print(f'Определитель:\n{np.linalg.det(a_3):.0f}')
```

```
Определитель:
43
```

Найдем решение по формулам Крамера:

In [11]:
```
x_1 = 86 / 43
x_1
```

Out[11]: 2.0

In [12]:
```
x_2 = -43 / 43
x_2
```

Out[12]: -1.0

In [13]:
```
x_3 = 43 / 43
x_3
```

Out[13]: 1.0

Ответ:

$$x_1 = 2,$$
$$x_2 = -1,$$
$$x_3 = 1.$$

2*. Найти L-матрицу LU-разложения для матрицы коэффициентов:

а)

$$\begin{pmatrix} 1 & 2 & 4 \\ 2 & 9 & 12 \\ 3 & 26 & 30 \end{pmatrix}$$

б)

$$\begin{pmatrix} 1 & 1 & 2 & 4 \\ 2 & 5 & 8 & 9 \\ 3 & 18 & 29 & 18 \\ 4 & 22 & 53 & 33 \end{pmatrix}$$

Решение:

In [14]:
```
def LU_matrix(a):
    num = len(a)
    l = np.zeros((num, num))
    u = list(a).copy()

    for i in range(num):
        l[i][i] = 1
        for j in range(i, num-1):
            if u[i][i] != 0:
                l[j+1][i] = u[j+1][i] / u[i][i]
                u[j+1] = u[j+1] - l[j+1][i] * u[i]
            else:
                continue

    print(f'L-матрица:\n {l} \nU-матрица:\n {np.array(u)}')
    return(l, np.array(u))
```

а)

In [15]:
```
a = np.array([[1, 2, 4], [2, 9, 12], [3, 26, 30]])
lu = LU_matrix(a)
```

```
L-матрица:
 [[1. 0. 0.]
 [2. 1. 0.]
 [3. 4. 1.]]
U-матрица:
 [[1. 2. 4.]
 [0. 5. 4.]
 [0. 0. 2.]]
```

б)

In [16]:
```
a = np.array([[1, 1, 2, 4], [2, 5, 8, 9], [3, 18, 29, 18], [4, 22, 53, 33]])
lu = LU_matrix(a)
```

```
L-матрица:
 [[1. 0. 0. 0.]
 [2. 1. 0. 0.]
 [3. 5. 1. 0.]
 [4. 6. 7. 1.]]
U-матрица:
 [[1. 1. 2. 4.]
 [0. 3. 4. 1.]
 [0. 0. 3. 1.]
 [0. 0. 0. 4.]]
```

3*. Решить систему линейных уравнений методом LU-разложения

$$\begin{cases} 2x_1 + x_2 + 3x_3 = 1 \\ 11x_1 + 7x_2 + 5x_3 = -6 \\ 9x_1 + 8x_2 + 4x_3 = -5 \end{cases}$$

Решение:

In [17]:
```
def LU_method(a, b, check=False):
    num = len(a)
    y = np.zeros(num)
    l, u = LU_matrix(a)
    status = True

    for i in range(num):
        y[i] = b[i]
        for j in range(i):
            y[i] = y[i] - l[i][j]*y[j]
    print(f'y = {y}')

    x = np.zeros(num)
    for i in range(num-1, -1, -1):
        x[i] = y[i] / u[i][i]
        for j in range(num-1, i, -1):
            x[i] = x[i] - u[i][j] * x[j] / u[i][i]

    if check:
        print('\nCHECKING...')
        eps = 0.005
        for i in range(num):
            summa = 0
            for j in range(num):
                summa += a[i][j] * x[j]
            if summa - b[i] < eps:
                print(f'{i+1} eq status: OK')
            else:
                print(f'{i+1} eq status: WRONG')
                status = False

    if status:
        print(f'\nРешение уравнения: x = {x}')
    else:
        print('\nРешено неверно (')
    return(x)
```

In [18]:
```
a = np.array([[2, 1, 3], [11, 7, 5], [9, 8, 4]])
b = np.array([1, -6, -5])
LU_method(a, b, check=True)
```

```
L-матрица:
 [[1.         0.         0.        ]
 [5.5        1.         0.        ]
 [4.5        2.33333333 1.        ]]
U-матрица:
 [[  2.          1.          3.        ]
 [  0.          1.5       -11.5       ]
 [  0.          0.         17.33333333]]
y = [  1.         -11.5         17.33333333]

CHECKING...
1 eq status: OK
2 eq status: OK
3 eq status: OK

Решение уравнения: x = [-1.  0.  1.]
```

Out[18]:
```
array([-1.,  0.,  1.])
```

4*. Решить систему линейных уравнений методом Холецкого

$$\begin{cases} 81x_1 - 45x_2 + 45x_3 = 531 \\ -45x_1 + 50x_2 - 15x_3 = -460 \\ 45x_1 - 15x_2 + 38x_3 = 193 \end{cases}$$

Решение:

In [19]:
```
a = np.array([[81, -45, 45], [-45, 50, -15], [45, -15, 38]])
b = np.array([531, -460, 193])
```

In [20]:
```
from scipy.linalg import cholesky
L = cholesky(a, lower=True)
L
```

Out[20]:
```
array([[ 9.,  0.,  0.],
       [-5.,  5.,  0.],
       [ 5.,  2.,  3.]])
```

In [21]:
```
def cholesky_method(a, b, check=False):
    num = len(a)
    y = np.zeros(num)
    l = cholesky(a, lower=True)
    u = l.T
    status = True

    for i in range(num):
        y[i] = b[i] / l[i][i]
        for j in range(i):
            y[i] = y[i] - l[i][j]*y[j] / l[i][i]
    print(f'y = {y}')

    x = np.zeros(num)
    for i in range(num-1, -1, -1):
        x[i] = y[i] / u[i][i]
        for j in range(num-1, i, -1):
            x[i] = x[i] - u[i][j] * x[j] / u[i][i]

    if check:
        print('\nCHECKING...')
        eps = 0.005
        for i in range(num):
            summa = 0
            for j in range(num):
                summa += a[i][j] * x[j]
            if summa - b[i] < eps:
                print(f'{i+1} eq status: OK')
            else:
                print(f'{i+1} eq status: WRONG')
                status = False

    if status:
        print(f'\nРешение уравнения: x = {x}')
    else:
        print(f'\nРешено неверно ( x = {x}')
    return(x)
```

In [22]:
```
cholesky_method(a, b, check=True)
```

```
y = [ 59. -33. -12.]

CHECKING...
1 eq status: OK
2 eq status: OK
3 eq status: OK

Решение уравнения: x = [ 6. -5. -4.]
```

Out[22]:
```
array([ 6., -5., -4.])
```

In [23]:
```
LU_method(a, b, check=True)
```

```
L-матрица:
 [[ 1.          0.          0.        ]
 [-0.55555556  1.          0.        ]
 [ 0.55555556  0.4         1.        ]]
U-матрица:
 [[ 81. -45.  45.]
 [  0.  25.  10.]
 [  0.   0.   9.]]
y = [ 531. -165.  -36.]

CHECKING...
1 eq status: OK
2 eq status: OK
3 eq status: OK

Решение уравнения: x = [ 6. -5. -4.]
```

Out[23]:
```
array([ 6., -5., -4.])
```

5*. Написать на Python программу с реализацией одного из изученных алгоритмов решения СЛАУ.

Решение:

In [24]:
```
def LU_matrix(a):
    num = len(a)
    l = np.zeros((num, num))
    u = list(a).copy()

    for i in range(num):
        l[i][i] = 1
        for j in range(i, num-1):
            if u[i][i] != 0:
                l[j+1][i] = u[j+1][i] / u[i][i]
                u[j+1] = u[j+1] - l[j+1][i] * u[i]
            else:
                continue

    print(f'L-матрица:\n {l} \nU-матрица:\n {np.array(u)}')
    return(l, np.array(u))


def LU_method(a, b, check=False):
    num = len(a)
    y = np.zeros(num)
    l, u = LU_matrix(a)
    status = True

    for i in range(num):
        y[i] = b[i]
        for j in range(i):
            y[i] = y[i] - l[i][j]*y[j]
    print(f'y = {y}')

    x = np.zeros(num)
    for i in range(num-1, -1, -1):
        x[i] = y[i] / u[i][i]
        for j in range(num-1, i, -1):
            x[i] = x[i] - u[i][j] * x[j] / u[i][i]

    if check:
        print('\nCHECKING...')
        eps = 0.005
        for i in range(num):
            summa = 0
            for j in range(num):
                summa += a[i][j] * x[j]
            if summa - b[i] < eps:
                print(f'{i+1} eq status: OK')
            else:
                print(f'{i+1} eq status: WRONG')
                status = False

    if status:
        print(f'\nРешение уравнения: x = {x}')
    else:
        print('\nРешено неверно (')
    return(x)
```

Проверка на основе примера из лекции:

In [25]:
```
a = np.array([[2, 1, -1], [4, -6, -2], [-2, 7, -3]])
b = np.array([5, -2, 7])
LU_method(a, b, check=True)
```

```
L-матрица:
 [[ 1.  0.  0.]
 [ 2.  1.  0.]
 [-1. -1.  1.]]
U-матрица:
 [[ 2.  1. -1.]
 [ 0. -8.  0.]
 [ 0.  0. -4.]]
y = [  5. -12.   0.]

CHECKING...
1 eq status: OK
2 eq status: OK
3 eq status: OK

Решение уравнения: x = [ 1.75  1.5  -0.  ]
```

Out[25]:
```
array([ 1.75,  1.5 , -0.  ])
```

In [26]:
```
a = np.array([[1, -1, 2], [2, 1, -3], [3, 0, 2]])
b = np.array([0, 0, 0])
LU_method(a, b, check=True)
```

```
L-матрица:
 [[1. 0. 0.]
 [2. 1. 0.]
 [3. 1. 1.]]
U-матрица:
 [[ 1. -1.  2.]
 [ 0.  3. -7.]
 [ 0.  0.  3.]]
y = [0. 0. 0.]

CHECKING...
1 eq status: OK
2 eq status: OK
3 eq status: OK

Решение уравнения: x = [0. 0. 0.]
```

Out[26]:
```
array([0., 0., 0.])
```