Лабораторная работа №2

«Решение систем линейных уравнений на Python»

(трудоемкость 4 часа[1])

Цель работы: познакомиться с возможностями библиотеки NumPy языка программирования Python по решению систем линейных уравнений (СЛУ).

Порядок выполнения работы:

1) Ознакомиться с примерами реализации различных способов решения СЛУ средствами библиотеки NumPy языка программирования Python (см. стр. 37-46[2])

2) Решить одну СЛУ (двумя способами[3]) согласно варианту:

номер варианта	**Задача 1** (см. стр. 45-46)
1	70
2	71
3	72
4	73
5	74
6	75
7	76
8	77
9	78
10	79
11	80
12	81
13	82
14	83
15	70
16	71
17	72
11	73
19	74
20	75
21	76
22	77
23	78
24	79
25	80

3) Оформить отчет, содержащий:

---

1 4 часа = 2 пары

2 Здесь и далее указаны оригинальные номера страниц книги (видны на скриншотах).

3 Решая одну систему двумя способами, можно сравнить результаты решения.

- титульный лист;
- цель работы;
- задание для своего варианта;
- листинг программного кода на языке программирования Python с реализацией решения СЛУ и распечаткой результатов. Программный код снабдить смысловыми комментариями;
- вывод по итогам выполнения лабораторной работы.

59.

$$\begin{pmatrix} 2 & 3 & 5 \\ 3 & 7 & 8 \\ 1 & -6 & 1 \\ 7 & -2 & 15 \end{pmatrix}.$$

60.

$$\begin{pmatrix} 1 & 1 & 1 & 6 \\ 2 & -1 & 1 & 3 \\ 1 & -1 & 2 & 5 \\ 3 & -6 & 5 & 6 \end{pmatrix}.$$

61.

$$A = \begin{pmatrix} -7 & -1 & -5 & 2 & 6 & -2 \\ -1 & -8 & 8 & 4 & -1 & -6 \\ -5 & 8 & 1 & 2 & 7 & -3 \\ 2 & 4 & 2 & 6 & 0 & -1 \\ 6 & -1 & 7 & 0 & -9 & 7 \\ -2 & -6 & -3 & -1 & 7 & 0 \end{pmatrix}.$$

СИСТЕМЫ ЛИНЕЙНЫХ УРАВНЕНИЙ (СЛУ)

Напомним, что система из m линейных уравнений с n переменными имеет вид:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m \end{cases},$$

где a_{ij} и b_j (для всех значений $i=1,2,...,m;\ j=1,2,...,n$) — это произвольные числа, называемые соответственно коэффициентами при переменных $x_1, x_2, ..., x_n$ и свободными членами. Матричная форма записи СЛУ имеет вид $A \cdot X = B$. Где

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{1n} \\ a_{21} & a_{22} & a_{2n} \\ \dots & \dots & \dots \\ a_{m1} & a_{m2} & a_{mn} \end{pmatrix}$$ — матрица из коэффициентов при неизвестных, $X = \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix}$ — матрица из неизвестных системы и $B = \begin{pmatrix} b_1 \\ b_2 \\ \dots \\ b_m \end{pmatrix}$ — матрица из свободных членов. Решением системы m линейных уравнений с n переменными называют упорядоченный набор чисел , при подстановке которых в каждое уравнение системы вместо соответствующих переменных получают верное равенство. Вообще, как доказывает Теорема Кронеккера-Капели, СЛУ могут иметь 1 решение, бесконечное множество решений и не иметь решения.

Сначала рассмотрим случай, когда система имеет единственное решение. Это значит, что матрица системы невырожденная, то есть ее определитель не равен нулю, и эту систему можно решать по правилу Крамера или методом обратной матрицы.

62. Решить СЛУ (случай единственного решения)

$$\begin{cases} 3x_1 - x_2 + x_3 = 5 \\ x_1 - x_2 - x_3 = 2 \\ 5x_1 - 3x_2 - x_3 = 10 \end{cases};$$

Решение

Для решения системы воспользуемся функцией **numpy.linalg.solve** модуля numpy (документация — http://docs.scipy.org/doc/numpy/reference/generated/numpy.linalg.solve.html). Функция принимает на вход 2 параметра:

1-й — матрица коэффициентов перед переменными

2-й — вектор свободных членов

```
A=np.array([[3,-1,1],[1,-1,-1],[5,-3,-1]])
B=np.array([3,11,8])
X=np.linalg.solve(A,B)
for t, x in zip(X, ['x1=','x2=', 'x3=']):
print (x,t)
x1= -1.2857142857142854
x2= -2.142857142857143
x3= 6.714285714285135
```

Функция zip используется для вывода решение системы на экран в виде x1=..., x2=..., x3=...

Обратим внимание, что определитель основной матрицы системы не равен нулю, мы получили единственное решение, хотя и не целочисленное.

```
import numpy as np
from numpy import linalg as ln
A=np.array([[3,-1,1],[2,1,1],[1,1,2]])
Det_A=ln.det(A)
print (Det_A)
7.000000000000001
```

63. Решить СЛУ

$$\begin{cases} x_1 - x_2 + x_3 = 3, \\ 2x_1 + x_2 + x_3 = 11, \\ x_1 + x_2 + 2x_3 = 8. \end{cases}$$

Решение

```
A=np.array([[1,-1,1],[2,1,1],[1,1,2]])
B=np.array([3,11,8])
X=np.linalg.solve(A,B)
for t, x in zip(X, ['x1=','x2=', 'x3=']):
print (x,t)
x1= 4.0
x2= 2.0
x3= 1.0
```

Решение системы линейных уравнений AX=B методом обратной матрицы

Детальное изложение метода можно найти в учебнике [1] часть 1.3.3 Реализуем следующий алгоритм:

1. Вычислим определитель матрицы A. Если определитель равен нулю, то конец решения. Система имеет бесконечное множество решений.

2. При определителе отличном от нуля, через алгебраические дополнения находится обратная матрица A.

3. Вектор решения X={ $x_1, x_2, \ldots, x_n$ } получается умножением обратной матрицы на вектор результата B.

64. Решить СЛУ методом обратной матриц

Найдем обратную матрицу с помощью функции np.linalg.inv(a) и умножим ее на столбец свободных членов с помощью функции np.dot(a_inv, b)

$$\begin{cases} 2x_1 + x_2 - 2x_3 = -3 \\ x_1 - 2x_2 + x_3 = 5 \\ 3x_1 + x_2 - x_3 = 0 \end{cases}$$

```
import numpy as np
a = np.array([[2, 1, -2], [1, -2, 1], [3, 1, -1]])
print ("Матрица:\n", a)
```

Матрица:

```
[[ 2 1 -2]
[ 1 -2 1]
[ 3 1 -1]]
b = np.array([-3, 5, 0])
det = np.linalg.det(a)
if det==0:
print («Определитель матрицы равен нулю, система имеет бесконечно много решений»)
else:
a_inv = np.linalg.inv(a)
X=np.dot(a_inv, b)
print («Решение системы:\n», X)
```

Решение системы:

```
[1. -1. 2.]
```

Решение системы линейных уравнений AX=B с использованием правила Крамера.

Если определитель матрицы системы не равен нулю, ее можно решить по правилу Крамера. Для этого находят определитель матрицы системы и делят на него определители, полученные заменой i-го столбца матрицы системы на столбец свободных членов. Эти отношения и будут составлять решение системы. Отметим, что оно будет единственным.

Детальное изложение метода можно найти в учебнике [1] пункт 1.3.4.

65. Решить СЛУ по правилу Крамера

$$\begin{cases} 2x_1 + x_2 - 2x_3 = -3 \\ x_1 - 2x_2 + x_3 = 5 \\ 3x_1 + x_2 - x_3 = 0 \end{cases}$$

Отметим, что ниже инструкция a_1=np.column_ stack((b,s_1,s_2) записывает вместо 1-го столбца матрицы a столбец свободных членов. Ниже, так же используется обращение к элементам массива, s_i=a[:,i] — срез, который записывает i-тый столбец матрицы a, причем первый нумеруется как нулевой и т.д.

```
det = np.linalg.det(a)
if det==0:
print( «Определитель матрицы равен нулю, система имеет бесконечно много решений»)
else:
s_0=a[:,0]
s_1=a[:,1]
s_2=a[:,2]
a_1=np.column_stack((b,s_1,s_2))
det_1 = np.linalg.det(a_1)
a_2=np.column_stack((s_0,b,s_2))
det_2 = np.linalg.det(a_2)
a_3=np.column_stack((s_0,s_1,b))
det_3 = np.linalg.det(a_3)
x_1=det_1/det
x_2=det_2/det
x_3=det_3/det
print( «Решение системы:», x_1,x_2,x_3)
```

Решение системы: 0.9999999999999996 -1.0 2.0000000000000004

Теперь рассмотрим пример системы, имеющей бесконечное множество решений.

Этапы решения СЛУ¶

1. Найти определитель матрицы системы det A.
2. Если он не равен нулю, вывести решение с помощью np.linalg.solve(A,B).
3. Иначе, сравниваем ранги расширенной и основной матриц системы и используем теорему Кронеккера- Капели, [1] пункт 1.3.4.

Ниже приведен возможный вариант решения задачи о нахождении решения системы линейных уравнений в общем виде. Отметим, что команда B.reshape(1,B.shape[0]) изменяет форму вектора свободных членов, который изначально представлен как массив-строка, в массив-столбец. Команда np.vstack((A,B1)) объединяет матрицу коэффициентов и столбец свободных членов в один двумерный массив для дальнейшей работы.

67. Решить СЛУ

$$\begin{cases} x_1 + 2x_2 - x_3 = 7 \\ 2x_1 - 3x_2 + x_3 = 3 \\ x_1 + x_2 - x_3 = 16 \end{cases}$$

Решение.

```
import numpy as np
A=np.array([[1,2,-1],[2,-3,1],[4,1,-1]])
B=np.array([7,3,16])
def solve_slu(A,B):
Det_A=np.linalg.det(A)
B1=B.reshape(1,B.shape[0])
AB=np.vstack((A,B1))
if Det_A!=0:
X=np.linalg.solve(A,B)
```

```
print ('EDINSVENNOE RESHENIE')
print( X)
if Det_A==0:
rank_A=np.linalg.matrix_rank(A)
rank_AB=np.linalg.matrix_rank(AB)
if rank_A==rank_AB:
print ('Сиситема имеет бесконечное множество решений, частное решение CHASTNOE'+X)
else: print ('Система не имеет решений')
return
solve_slu(A,B)
print (np.linalg.matrix_rank(A))
B1=B.reshape(1,B.shape[0])
AB=np.vstack((A,B1))
print (np.linalg.matrix_rank(AB))
```

Система не имеет решений

2

3

ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Решить СЛУ

68. $$\begin{cases} -6x_1+2x_2+x_3-5x_4+2x_5-3x_6-x_7+7x_8=49, \\ -4x_1-6x_3-5x_4+x_5+4x_6-6x_7-2x_8=23, \\ 5x_1-5x_2-x_3-3x_4+5x_5+2x_6+6x_7+3x_8=-62, \\ -5x_1+3x_2-4x_3+5x_4-3x_5-8x_7+x_8=74, \\ 9x_1-4x_3-3x_4-3x_5+4x_6-5x_7-3x_8=-24, \\ -5x_1-5x_2-3x_3+7x_4+2x_5-5x_6-2x_7-x_8=34, \\ -x_1+x_2+5x_3+2x_5-10x_6-x_7-x_8=39, \\ 7x_1-x_2+4x_3+3x_4-4x_5+8x_8=-14 \end{cases}$$

69. $\begin{cases} -5x_1 - 5x_2 + 11x_4 + 2x_5 - 9x_6 + 3x_7 - x_8 = 34, \\ -11x_1 - 4x_2 + 4x_3 - 5x_4 - 8x_5 + 4x_6 + 2x_7 + 8x_8 = 62, \\ -x_1 + 2x_3 - 7x_4 + 2x_5 - 4x_6 - 11x_8 = 53, \\ x_1 - 7x_2 + 13x_3 - 4x_4 + 2x_5 - x_6 - 4x_7 - 8x_8 = 93, \\ 2x_2 + 7x_3 - 4x_4 - 5x_6 - 6x_7 + 6x_8 = 32, \\ -3x_1 - x_2 + 5x_3 - x_6 + x_7 + 6x_8 = 44, \\ 7x_1 + 8x_3 - 4x_4 + x_5 - 5x_6 + 7x_7 - 3x_8 = 99, \\ -4x_1 - 3x_2 + 2x_4 + 9x_5 - 8x_6 + 3x_7 - 2x_8 = -75 \end{cases}$

70

$\begin{cases} 2x_1 + 3x_2 - x_3 + x_4 = 5 \\ 3x_1 - x_2 + 2x_3 + x_4 = 1 \\ x_1 + 2x_2 + 3x_3 + 4x_4 = 6 \\ 6x_1 + 4x_2 + 4x_3 + 6x_4 = 1 \end{cases}$

71. $\begin{cases} 2x_1 + x_2 + x_3 + x_4 = 1 \\ x_2 - x_3 + 2x_4 = 2 \\ 2x_1 + 2x_2 + 3x_4 = 3 \end{cases}$

72 $\begin{cases} x_1 + 2x_2 - x_3 = 7 \\ 2x_1 - 3x_2 + x_3 = 3 \\ 4x_1 + x_2 - x_3 = 16 \end{cases}$

73. $\begin{cases} x_1 + 2x_2 + 3x_3 - 2x_4 = 6 \\ 2x_1 + 4x_2 - 2x_3 - 3x_4 = 18 \\ 3x_1 + 2x_2 - x_3 + 2x_4 = 4 \\ 2x_1 - 3x_2 + 2x_3 + x_4 = -8 \end{cases}$

74. $\begin{cases} -6x_1 + 9x_2 + 3x_3 + 2x_4 = 4 \\ -2x_1 + 3x_2 + 5x_3 + 4x_4 = 2 \\ -4x_1 + 6x_2 + 4x_3 + 3x_4 = 3 \end{cases}$

75. $\begin{cases} 5x_1 - x_2 + 2x_3 + x_4 = 7 \\ 2x_1 + x_2 + 4x_3 - 2x_4 = 1 \\ x_1 - 3x_2 - 6x_3 + 5x_4 = 0 \end{cases}$

76. $\begin{cases} 3x_1 + 2x_2 - x_3 + x_4 = -4 \\ 2x_1 - x_2 + 3x_3 - x_4 = 7 \\ -x_1 + 3x_2 + 3x_3 + x_4 = 14 \\ 5x_1 - 3x_2 - x_3 - 2x_4 = -10 \end{cases}$

77. $\begin{cases} x + 2y + z = 8 \\ -2x + 3y - 3z = -5 \\ 3x - 4y + 5z = 10 \end{cases}$

78. $\begin{cases} 3x + 2y + z = -8 \\ 2x + 3y + z = -3 \\ 2x + y + 3z = -1 \end{cases}$

79. $\begin{cases} -6x - y + 4z = 1 \\ -4x - 4y + 3z = -3 \\ 3x - 2y - 2z = -3 \end{cases}$

80. $\begin{cases} 3x + 2y + z = 1 \\ 6x + 5y + 4z = -2 \\ 9x + 8y + 7z = 3 \end{cases}$

81. $\begin{cases} 2x - 3y - z = -6 \\ 3x + 4y + 3z = -5 \\ x + y + z + 2 = 0 \end{cases}$

82. $\begin{cases} 6x + 4y + z = -17 \\ 5x + 6y + 2z = -10 \\ x + y - z + 1 = 0 \end{cases}$

83. $\begin{cases} -2x + y + 6 = 0 \\ x - 2y - z = 5 \\ 3x + 4y - 2z = 13 \end{cases}$