

**Федеральное государственное автономное образовательное  
учреждение высшего образования  
«Национальный исследовательский университет  
«Высшая школа экономики»**

Образовательная программа  
«Прикладная математика»

**ОТЧЕТ  
По лабораторной работе № 1**

По предмету  
**«Численные методы»**

По теме  
**«Теория погрешностей и машинная арифметика»**

**Выполнил**  
студент группы БПМ211  
Кудряшов Максим Дмитриевич

**Проверил**  
Брандышев Петр Евгеньевич

Москва - 2024

## Содержание

<b>1</b>	<b>3.1.8</b>	<b>2</b>
1.1	Постановка задачи . . . . .	2
1.2	Теория . . . . .	2
1.3	аналитическое решение тестового примера и результат вычислительного эксперимента по тесту . . . . .	2
1.4	Код на Python . . . . .	2
1.5	Результаты . . . . .	4
<b>2</b>	<b>3.2</b>	<b>5</b>
<b>3</b>	<b>3.8.2</b>	<b>5</b>

## 1 3.1.8

### 1.1 Постановка задачи

Дана система уравнений  $Ax = b$  порядка  $n = 6$ . Исследовать зависимость погрешности решения  $x$  от погрешностей правой части системы  $b$ .

$$A_{ij} = \frac{1}{\sqrt{c_{ij}^2 + 0.58c_{ij}}}$$

$$b_i = N$$

$$c_{ij} = 0.1Nij$$

$$N = 8$$

### 1.2 Теория

### 1.3 аналитическое решение тестового примера и результат вычислительного эксперимента по тесту

### 1.4 Код на Python

```

1  import numpy as np
2  import matplotlib.pyplot as plt
3  import os
4  import pandas as pd
5
6  output_file_path = "3.1.8.txt"
7
8  n = 6
9  N = 8
10 delta = 0.05
11
12 os.remove(output_file_path)
13 with open(output_file_path, "a") as f:
14     # заполним матрицу A
15     A = np.empty((n, n))
16     for i in range(n):
17         for j in range(n):
18             c = 0.1 * N * (i + 1) * (j + 1)
19             A[i][j] = 1 / np.sqrt(c ** 2 + 0.58 * c)
20     # print(f'A = {A}', file=f)
21
22     # заполним вектор b
23     b = np.full(n, N, dtype=float)
24
25     # найдем лист из измененных векторов b

```

```

26 b_error_list = []
27 for i in range(n):
28     b_error = b.copy()
29     b_error[i] += delta
30     b_error_list.append(b_error)
31
32 # найдем решение через встроенную функцию
33 x = np.linalg.solve(A, b)
34 print(f"x = ", file=f)
35
36 # найдем число обусловленности через встроенную функцию
37 A_cond = np.linalg.cond(A, np.inf)
38 print(f"A_cond = ", file=f)
39
40 # найдем вектор d
41 d = np.empty(n)
42 for i in range(n):
43     b_error = b_error_list[i]
44     x_error = np.linalg.solve(A, b_error)
45     d[i] = np.linalg.norm(x - x_error, ord=np.inf) / np.linalg.norm(x, ord=np.inf)
46 print(f"d = ", file=f)
47
48 # найдем что оказывает наибольшее влияние на погрешность
49 d_max = d.max()
50 d_max_index = list(d).index(d_max)
51 print(f'd_max = {d_max} with index = {d_max_index}', file=f)
52
53 # построим гистограмму
54 plt.bar(range(n), d)
55 plt.savefig('3.1.8.png', dpi=300)
56 plt.show()
57
58 # print(f"Сравнение погрешностей:", file=f)
59 delta_rel_x_error_list = np.empty(n, dtype=float)
60 for i in range(n):
61     b_error = b_error_list[i]
62     # delta_rel_b_old = np.linalg.norm(b - b_error, ord=np.inf) / np.linalg.norm(b, ord=np.inf)
63     delta_rel_b = np.abs(delta) / np.linalg.norm(b, ord=np.inf)
64     delta_rel_x_error = A_cond * delta_rel_b
65     delta_rel_x_error_list[i] = delta_rel_x_error
66     # print(f"практически: {d[i]}, \tтеоретически: {delta_rel_x_error}", file=f)
67
68     df = pd.DataFrame({
69         'Практические': d,
70         'Теоретические': delta_rel_x_error_list
71     })
72     df.to_latex("3.1.8.tex")
73
74 # with open(output_file_path, "r") as f:
75 #     print(f.read())

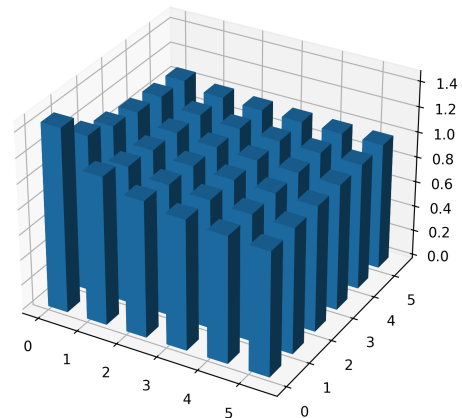
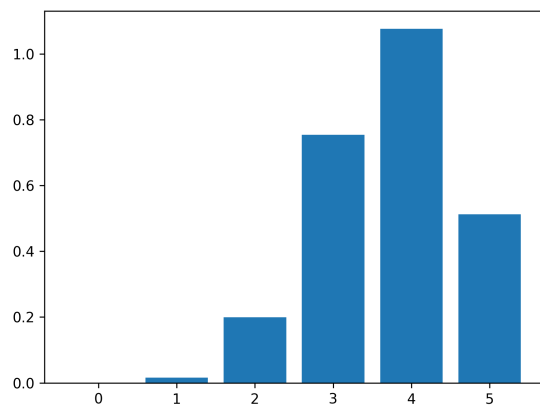
```

## 1.5 Результаты

```

1  x = array([ 2.28625963e+06, -1.97930833e+08,  2.39587644e+09, -8.96828858e+09,
2          1.27233076e+10, -6.02721331e+09])
3  A_cond = 1678868976234.5142
4  d_for_b = array([1.75867282e-04, 1.61316244e-02, 1.99311489e-01, 7.53942029e-01,
5          1.07646727e+00, 5.12130976e-01])
6  d_for_A = array([[1.45164313, 1.1693928 , 1.07560493, 1.02840626, 1.00000045,
7          0.98102369],
8          [1.23898367, 1.09095131, 1.04062713, 1.0152746 , 1.          ,
9          0.98979009],
10         [1.16321821, 1.06218071, 1.02778666, 1.01044896, 1.          ,
11         0.99301434],
12         [1.12391529, 1.04724248, 1.02111544, 1.00794103, 1.          ,
13         0.99469056],
14         [1.09987034, 1.03809273, 1.01702794, 1.00640416, 1.          ,
15         0.99571792],
16         [1.08363861, 1.03191264, 1.0142665 , 1.00536578, 1.          ,
17         0.99641211]])
18 d_for_b_max = 1.0764672729236027 with index = 4
19 d_for_A_max = 1.4516431254007967 with index = [0, 0]

```



Индекс	Практические	Теоретические
0	0.000176	10492931101.465714
1	0.016132	10492931101.465714
2	0.199311	10492931101.465714
3	0.753942	10492931101.465714
4	1.076467	10492931101.465714
5	0.512131	10492931101.465714

i	Практические	Теоретические
(0, 0)	1.451643	32548247136.209515
(0, 1)	1.169393	32548247136.209515
(0, 2)	1.075605	32548247136.209515
(0, 3)	1.028406	32548247136.209515
(0, 4)	1.000000	32548247136.209515
(0, 5)	0.981024	32548247136.209515
(1, 0)	1.238984	32548247136.209515
(1, 1)	1.090951	32548247136.209515
(1, 2)	1.040627	32548247136.209515
(1, 3)	1.015275	32548247136.209515
(1, 4)	1.000000	32548247136.209515
(1, 5)	0.989790	32548247136.209515
(2, 0)	1.163218	32548247136.209515
(2, 1)	1.062181	32548247136.209515
(2, 2)	1.027787	32548247136.209515
(2, 3)	1.010449	32548247136.209515
(2, 4)	1.000000	32548247136.209515
(2, 5)	0.993014	32548247136.209515
(3, 0)	1.123915	32548247136.209515
(3, 1)	1.047242	32548247136.209515
(3, 2)	1.021115	32548247136.209515
(3, 3)	1.007941	32548247136.209515
(3, 4)	1.000000	32548247136.209515
(3, 5)	0.994691	32548247136.209515
(4, 0)	1.099870	32548247136.209515
(4, 1)	1.038093	32548247136.209515
(4, 2)	1.017028	32548247136.209515
(4, 3)	1.006404	32548247136.209515
(4, 4)	1.000000	32548247136.209515
(4, 5)	0.995718	32548247136.209515
(5, 0)	1.083639	32548247136.209515
(5, 1)	1.031913	32548247136.209515
(5, 2)	1.014267	32548247136.209515
(5, 3)	1.005366	32548247136.209515
(5, 4)	1.000000	32548247136.209515
(5, 5)	0.996412	32548247136.209515

2 3.2

3 3.8.2