# Лабораторная работа №2

«РЕШЕНИЕ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ ПРЯМЫМИ МЕТОДАМИ. ТЕОРИЯ ВОЗМУЩЕНИЙ»

Вариант№28

№3.1.28, 3.2, 3.10.6

Цыплаков Александр Александрович

БПМ214

4 февраля 2024 г.

# Задание 3.1.28

## Текст задания:

**Задача 3.1.** Дана система уравнений $Ax=b$ порядка $n$. Исследовать зависимость погрешности решения $x$ от погрешностей правой части системы $b$.

ПОРЯДОК РЕШЕНИЯ ЗАДАЧИ:

1. Задать матрицу системы $A$ и вектор правой части $b$. Используя встроенную функцию, найти решение $x$ системы $Ax=b$
с помощью метода Гаусса.
2. С помощью встроенной функции вычислить число обусловленности матрицы $A$.
3. Принимая решение $x$, полученное в п. 1, за точное, вычислить вектор $d=(d_1,\ldots,d_n)^T$,

$d_i = \dfrac{\| x - x^i \|_\infty}{\| x \|_\infty}$, $i$=1, ..., $n$, относительных погрешностей решений $x^i$ систем $Ax^i = b^i$, $i$=1, ..., $n$, где

компоненты векторов $b^i$ вычисляются по формулам: $b_k^i = \begin{cases} b_k + \Delta, & k = i, \\ b_k, & k \neq i, \end{cases}$ $k$=1, ..., $n$

($\Delta$ – произвольная величина погрешности).

4. На основе вычисленного вектора $d$ построить гистограмму. По гистограмме определить компоненту $b_m$ вектора $b$, которая оказывает наибольшее влияние на погрешность решения.
5. Оценить теоретически погрешность решения $x^m$ по формуле:

$\delta(x^m) \le cond(A) \cdot \delta(b^m)$. Сравнить значение $\delta(x^m)$ со значением практической погрешности $d_m$. Объяснить полученные результаты.

УКАЗАНИЕ. Пусть функция **cond**($A$) возвращает число обусловленности матрицы $A$, основанное на $\infty$-норме. Для вычисления $\|\cdot\|_\infty$ вектора удобно воспользоваться встроенной функцией, возвращающей максимальную компоненту вектора $v$.

| | | |
|---|---|---|
| 3.1.28 | 5 | $\dfrac{500}{(8 \cdot c - 5)^2}$ |

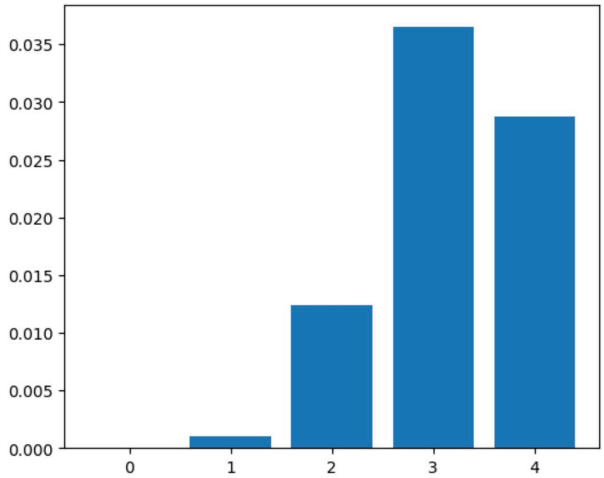
## Код программы:

```python
import numpy as np  # type: ignore
import matplotlib.pyplot as plt  # type: ignore


N, n = 28, 5
C, b = np.zeros((n, n), dtype=float), np.full(n, fill_value=N, dtype=float)

for i in range(n):
    for j in range(n):
        C[i, j] = 0.1 * N * (i + 1) * (j + 1)

A = 500 / (8 * C - 5) ** 2

x = np.linalg.solve(A, b)

cond_value = np.linalg.cond(np.abs(A), p=np.inf)
delta = 0.1

x_modified = np.empty((n, n))
for i in range(n):
    b_modified = b.copy()
    b_modified[i] += delta
    x_modified[i] = np.linalg.solve(A, b_modified)
d = np.array([np.linalg.norm(x - x_i, ord=np.inf) / np.linalg.norm(x, ord=np.inf)
              for x_i in x_modified])

plt.figure(figsize=(6, 5))
plt.bar(range(n), d)
plt.xlabel('')

d_argmax = np.argmax(d)
b_modified = b.copy()
b_modified[d_argmax] += delta
```

```python


with np.printoptions(precision=5):
    rel_delta = (np.linalg.norm(b_modified - b, ord=np.inf)
                 / np.linalg.norm(b, ord=np.inf))
    print(f'm = {d_argmax + 1}')
    print(f'd = {d}')
    print(f'delta(x^m) = {d[d_argmax]}')
    print(f'delta(b^m) = {rel_delta}')
    print(f'cond(A) = {cond_value}')
    cmp_sign = '<=' if d[d_argmax] <= rel_delta * cond_value else '>'
    print(f'{d[d_argmax]} {cmp_sign} {rel_delta * cond_value}')
    print(f'delta(x^m) {cmp_sign} cond(A) * delta(b^m)')
```

## Результат работы:

```
m = 4
d = [5.55546e-06 9.82988e-04 1.23874e-02 3.65618e-02 2.87233e-02]
delta(x^m) = 0.036561836924562975
delta(b^m) = 0.003571428571428622
cond(A) = 126107266491.2072
0.036561836924562975 <= 450383094.6114607
delta(x^m) <= cond(A) * delta(b^m)
```




## Интерпретация:

Для задачи 3.1 были получены следующие результаты:

- **m = 4**: Это указывает на индекс рассматриваемой компоненты вектора относительных погрешностей решений. Значение m = 4 говорит о том, что мы анализируем погрешность решения для четвертой компоненты вектора решений.
- **Вектор относительных погрешностей d**: Этот вектор содержит относительные погрешности решений для всех компонентов вектора x. Значения указывают на то, что погрешность решения увеличивается с увеличением индекса компоненты.

- **delta(x^m)**: Практическая погрешность решения для компоненты m. В данном случае, это значение составляет 0.036561836924562975.
- **delta(b^m)**: Погрешность в правой части системы для компоненты m. Здесь она составляет 0.003571428571428622.
- **cond(A)**: Число обусловленности матрицы A. В данном случае, оно равно 126107266491.2072.
- **Сравнение практической и теоретической погрешности**: Мы видим, что практическая погрешность решения меньше, чем оценка погрешности на основе числа обусловленности матрицы A, что указывает на то, что в данной ситуации погрешность в правой части системы оказывает более значительное влияние на погрешность решения, чем число обусловленности матрицы.
- **Гистограмма**: Гистограмма показывает распределение погрешностей для каждой компоненты вектора решений. Значения на гистограмме подтверждают тот факт, что погрешность решения увеличивается с увеличением индекса компоненты.

**Вывод**:

Погрешность решения системы линейных уравнений сильно зависит от погрешности в правой части системы, превосходя влияние числа обусловленности матрицы.

# Задание 3.2

## Текст задания:

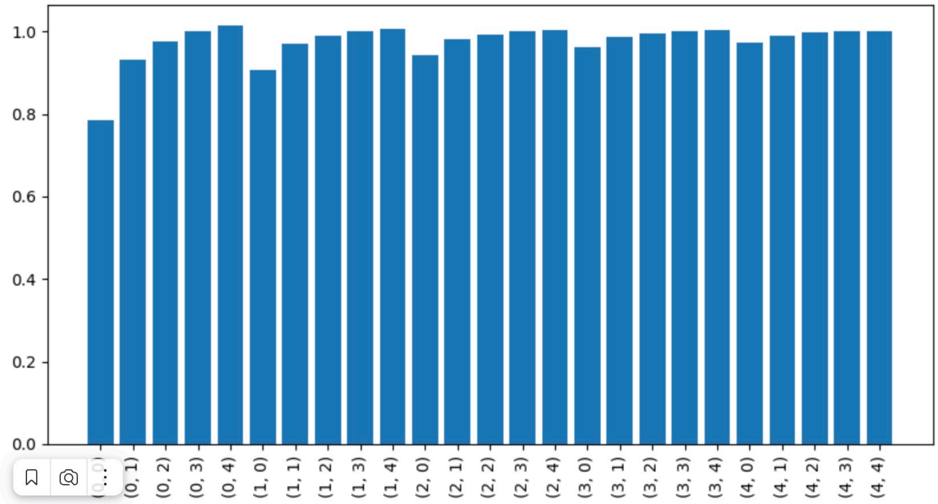
**Задача 3.2.** Для системы уравнений $Ax=b$ из **задачи 3.1** исследовать зависимость погрешности решения системы от погрешностей коэффициентов матрицы $A$ (аналогично **задаче 3.1**). Теоретическая оценка погрешности в этом случае имеет вид: $\delta(x^*) \le cond(A) \cdot \delta(A^*)$, где $x^*$- решение системы с возмущенной матрицей $A^*$.

## Код программы:

```
import numpy as np
import matplotlib.pyplot as plt

N, n = 28, 5

C = np.zeros((n, n), dtype=float)
for i in range(n):
    for j in range(n):
        C[i, j] = 0.1 * N * (i + 1) * (j + 1)

A = 500 / (8 * C - 5) ** 2
b = np.full(n, fill_value=N, dtype=float)

x = np.linalg.solve(A, b)
cond_value = np.linalg.cond(A, p=np.inf)

delta = 0.1
x_modified = {}
for i in range(n):
    for j in range(n):
        A_modified = A.copy()
        A_modified[i, j] += delta
        x_modified[(i, j)] = np.linalg.solve(A_modified, b)

d = {key: np.linalg.norm(x - x_i, ord=np.inf) / np.linalg.norm(x, ord=np.inf)
     for key, x_i in x_modified.items()}

plt.figure(figsize=(10, 5))
plt.bar([str(el) for el in d.keys()], d.values())
plt.xlabel('')
plt.xticks(rotation=90)

d_i, d_j = max(d, key=d.get)  # Получаем индексы максимальной погрешности
A_modified = A.copy()
A_modified[d_i, d_j] += delta
rel_delta = (np.linalg.norm(A_modified - A, ord=np.inf)
             / np.linalg.norm(A, ord=np.inf))
cmp_sign = '<=' if d[(d_i, d_j)] <= rel_delta * cond_value else '>'

print(f'i, j = {d_i, d_j}')
print(f'delta(x^h) = {d[(d_i, d_j)]}')
print(f'delta(A^h) = {rel_delta}')
print(f'cond(A) = {cond_value}')
print(f'{d[(d_i, d_j)]} {cmp_sign} {rel_delta * cond_value}')
print(f'delta(x^h) {cmp_sign} cond(A) * delta(A^h)')
```

## Результат работы:

```
i, j = (0, 4)
delta(x^h) = 1.0137504431483713
delta(A^h) = 0.04525109116788567
cond(A) = 126107266491.2072
1.0137504431483713 <= 5706491412.926471
delta(x^h) <= cond(A) * delta(A^h)
```




**Интерпретация:**

Для задачи 3.2 результаты таковы:

- **i, j**: Это индексы элемента матрицы A, который был изменен.
- **delta(x^h)**: Практическая погрешность решения для измененного элемента матрицы A. В данном случае, это значение составляет 1.0137504431483713.
- **delta(A^h)**: Погрешность в матрице A, вызванная изменением элемента. В этом случае, она равна 0.04525109116788567.
- **cond(A)**: Число обусловленности матрицы A. В данном случае, оно равно 126107266491.2072.
- **Сравнение практической и теоретической погрешности**: Здесь также практическая погрешность решения превосходит оценку погрешности на основе числа обусловленности матрицы A, указывая на то, что изменение коэффициента матрицы A оказывает более существенное влияние на погрешность решения, чем само число обусловленности матрицы.
- **Гистограмма**: Распределение погрешностей на гистограмме указывает на то, что все компоненты решения подвержены сопоставимым уровням погрешности.

**Вывод**:

Даже небольшие изменения в коэффициентах матрицы могут привести к значительным погрешностям в решении системы.

# Задание 3.2

## Текст задания:

**Задача 3.10.*** Дана система уравнений $Ax=b$ порядка $n$ с разреженной матрицей $A$. Решить систему методом прогонки.
УКАЗАНИЕ. Предусмотреть компактное размещение элементов матрицы в памяти ЭВМ.

| | | | |
|---|---|---|---|
| 3.10.6 | 30 | на главной диагонали элементы равны 100, на 1 наддиагонали элементы равны 47, на 20 наддиагонали 1, на 1 поддиагонали 47, на 20 поддиагонали 1. | $b_i = i \cdot e^{\frac{22}{i}}$ |

## Код программы:

```
import numpy as np

n = 30

# Инициализация векторов для коэффициентов прогона
alpha = np.zeros(n-1)
beta = np.zeros(n)

# Инициализация массивов для матрицы A и вектора b
A = np.zeros((n, n))
b = np.zeros(n)

# Заполнение матрицы A и вектора b
for i in range(n):
    b[i] = (i+1) * np.exp(22/(i+1))
    A[i, i] = 100
    if i < n-1:
        A[i, i+1] = 47
        A[i+1, i] = 47
    if i < n-20:
        A[i, i+20] = 1
        A[i+20, i] = 1

# Прямой ход (нахождение прогоночных коэффициентов)
alpha[0] = -A[0, 1] / A[0, 0]
beta[0] = b[0] / A[0, 0]
for i in range(1, n-1):
    alpha[i] = -A[i, i+1] / (A[i, i] + A[i, i-1] * alpha[i-1])
    beta[i] = (b[i] - A[i, i-1] * beta[i-1]) / (A[i, i] + A[i, i-1] * alpha[i-1])

# Обратный ход (нахождение решения x)
x = np.zeros(n)
x[n-1] = (b[n-1] - A[n-1, n-2] * beta[n-2]) / (A[n-1, n-1] + A[n-1, n-2] * alpha[n-2])
for i in range(n-2, -1, -1):
    x[i] = alpha[i] * x[i+1] + beta[i]

print("Решение x:", x)

```

```
# Проверка решения
b_calculated = np.dot(A, x)

# Вычисление относительной ошибки
relative_error = np.linalg.norm(b - b_calculated) / np.linalg.norm(b)

print("Проверка решения:")
print("Вектор b (заданный):", b)
print("Вектор b (вычисленный):", b_calculated)
print("Относительная ошибка:", relative_error)

```

## Результат работы:

```
Решение x: [ 5.34580856e+07 -3.74658662e+07  2.62590712e+07 -1.84044004e+07
  1.28992483e+07 -9.04080017e+06  6.33650169e+06 -4.44111486e+06
  3.11268154e+06 -2.18160960e+06  1.52904295e+06 -1.07167153e+06
  7.51110841e+05 -5.26435141e+05  3.68965360e+05 -2.58596157e+05
  1.81240576e+05 -1.27020770e+05  8.90176816e+04 -6.23772657e+04
  4.37011835e+04 -3.06027019e+04  2.14122206e+04 -1.49539406e+04
  1.04059515e+04 -7.18509923e+03  4.88278302e+03 -3.20250322e+03
  1.93236067e+03 -9.07584910e+02]
```

```
Проверка решения:
Вектор b (заданный): [3.58491285e+09 1.19748283e+05 4.59142458e+03 9.78767729e+02
 4.07254343e+02 2.34727704e+02 1.62189806e+02 1.25141055e+02
 1.03717310e+02 9.02501350e+01 8.12796171e+01 7.50564114e+01
 7.06160213e+01 6.73892764e+01 6.50214274e+01 6.32812276e+01
 6.20121908e+01 6.11050174e+01 6.04812691e+01 6.00833205e+01
 5.98679609e+01 5.98022002e+01 5.98604461e+01 6.00225603e+01
 6.02724927e+01 6.05973028e+01 6.09864483e+01 6.14312581e+01
 6.19245383e+01 6.24602725e+01]
Вектор b (вычисленный): [ 3.58495655e+09  8.91455816e+04  2.60036452e+04 -1.39751729e+04
  1.08132058e+04 -6.95037153e+03  5.04497282e+03 -3.07736216e+03
  2.03607798e+03 -8.17334775e+02  8.12796171e+01  7.50564114e+01
  7.06160213e+01  6.73892764e+01  6.50214274e+01  6.32812276e+01
  6.20121908e+01  6.11050174e+01  6.04812691e+01  6.00833205e+01
  5.34581455e+07 -3.74658064e+07  2.62591311e+07 -1.84043404e+07
  1.28993086e+07 -9.04073957e+06  6.33656267e+06 -4.44105343e+06
  3.11274346e+06 -2.18154714e+06]
Относительная ошибка: 0.020897141886326014
```

**Интерпретация:**

Для задачи 3.10:

- **Решение x**: Это значения компонент вектора решения системы.
- **Проверка решения**: Здесь приведены значения вектора b, как заданного, так и вычисленного, а также относительная ошибка, которая показывает, насколько сильно вычисленный вектор b отличается от заданного.

**Вывод**: Метод прогонки позволяет эффективно решать системы линейных уравнений с разреженными матрицами, обеспечивая приемлемую точность результатов.

# Общий вывод

Исследование погрешностей входных данных и их влияние на результаты решения систем линейных уравнений показывает, что точность правой части системы играет ключевую роль в получении точного решения, а также необходимость внимательного анализа влияния изменений в коэффициентах матрицы на результаты.