МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

имени М.В.Ломоносова

Факультет вычислительной математики и кибернетики

Компьютерный практикум по учебному курсу

«ВВЕДЕНИЕ В ЧИСЛЕННЫЕ МЕТОДЫ»

ЗАДАНИЕ № 1

Подвариант № 1

РЕШЕНИЕ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ МЕТОДОМ ГАУССА И МЕТОДОМ ГАУССА С ВЫБОРОМ ГЛАВНОГО ЭЛЕМЕНТА

ОТЧЕТ

о выполненном задании

студента 202 учебной группы факультета ВМК МГУ

Плеханова Антона Дмитриевича

гор. Москва

2020 год

Содержание

Цель работы

Изучить классический и модифицированный методы Гаусса, применяемые для решение систем линейных алгебраических уравнений

Постановка задачи

Дана система уравнений $Ax = f$, с невырожденной матрицей коэфицентов A размера $n \times n$. Написать программу, которая решает заданную систему линейных алгебраических уравнений методом Гауса и методом Гауса с выбором главного элемента. Предусмотреть возможность задания элементов матрицы коэфицентов и вектора-столбца правой части как во входном файле данных, так и путем задания специальных формул.

Задачи практической работы

1. Решить заданную СЛАУ методом Гаусса и методом Гаусса с выбором главного элемента;
2. Вычислить $\det A$;
3. Вычислить A^{-1};
4. Определить число обусловленности $M_A = ||A|| \cdot ||A^{-1}||$;
5. Исследовать вопрос вычислительной устойчивости метода Гаусса (при больших значениях параметра n);
6. Правильность решения СЛАУ подтвердить системой тестов;

Алгоритмы

Классический метод Гаусса

Метод Гаусса решения уравнения Ax = f с невырожденной матрицей A разделим на два этапа: прямой ход и обратный ход.

1. Во время прямого метода Гаусса осуществляется сведение исходной матрицы системы к эквивалентной верхней треугольной матрице следующим образом: На k-ом шаге прямого хода рассматриваем k-й столбец матрицы коэффициентов. Поскольку матрицы невырожденна, в нем существует элемент $a_{kl} \neq 0$, где $l \geq k$. Если $a_{kk} = 0$, меняем k-ю и l-ю строки матрицы коэффициентов, тогда элемент a_{kk}, называемый ведущим, будет не равен нулю. Затем делим k-ю строку на a_{kk} и из каждой строки с номером $m > k$ вычитаем k-ю строку, умноженную на a_{km}. Таким образом, k-й столбец будет состоять из нулей ниже k-й компоненты, а матрица системы будет сведена к эквивалентной верхней треугольной.

2. Во время обратного метода Гаусса осуществляется последовательное вычисление компонент решения начиная с последней.

Модифицированный метод Гаусса

Модифицированный метод Гаусса (метод Гаусса с выбором главного элемента) так же может быть разделен на прямой ход и обратный лишь с тем отличием, что на k-ом шаге прямого хода ведущий элемент выбирается не из k-ого столбца произвольным образом, а из k-ой строки максимальным по модулю. Такая модификация гарантирует, что все элементы матрицы коэффициентов после прямого хода будут удовлетворять неравенству $|c_{ij}| \leq 1$, а это приведет к существенному уменьшению погрешности вычислений

Описание программы

Программа написана на языке программирования Python3. Для ее использования необходима блиботека numpy.
При запуске программы пользователь должен выбрать способ задания матрицы (матрица задается вариантом задания, в файле или со стандартного потока ввода) Внимание! Программа ожидает введение невырожденной матрицы, в ином случае исход не определен. Затем пользователь должен выбрать операции на матрицей из списка доступных:

1. Найти определитель матрицы
2. Найти обратную матрицу
3. Найти число обусловленности матрицы
4. Решить систему классическим методом Гаусса
5. Решить систему модифицированным методом Гаусса

(Для квадратной матрицы доступны команды 1-3, для расширенной матрицы системы доступны команды 1-5, причем команды 1-3 используют матрицу коэффициентов)
В данной программе также были реализованы следующие функции:

- GaussStraightRun(A, $for\_det = False$) - функция осуществляет прямой ход метода Гаусса на матрице A и в зависимости от значения параметра for_det возвращает либо (при $for\_det == True$) массив ведущих элементов матрицы A, полученный в ходе прямого метода Гаусса, с последним элементом - $(-1)^k$, где k - количество перестановок строк, произведенных в ходе работы функции, либо при $for\_det == False$) возвращает матрицу, полученную в ходе прямого метода Гаусса. (Матрица A может быть как расширенной, так и просто матрицей коэффициентов)

- GaussReverseRun(C) - функция принимает в качестве аргумента матрицу C, полученную из функции GaussStraightRun(A, $for\_det = False$) (A должна быть расширенной матрицей), и осуществляет обратный ход метода Гаусса. Возвращаемое значение - вектор-столбец решения системы, заданной расширенной матрицей A.

- GaussMethod(A) - функция объединяет в себе две предыдущие функции (решает систему с расишренной матрицей A)

- ModifiedGaussStraightRun(A, $for\_det = False$) - функция осуществляет прямой ход модифицированного метода Гаусса на матрице A. Аналогично GaussStraightRun, в зависимости от значения параметра for_det возвращает либо кортеж из матрицы, полученной в ходе прямого хода, и массива нумерации переменных (в этом методе столбцы могут меняться местами), либо массив ведущих элементов с послежним элементом, отвечающим за знак определителя.

- ModifiedGaussReverseRun(C, $var_indexes$) - функция принимает в качестве аргументов элементы кортежа, возвращаемого функцией ModifiedGaussStraightRun(A, $for\_det = False$)

- det(A, modified=False) - функция вычисляет определитель матрицы A (если матрица A - расширенная матрица системы, то вычисляется определитель матрицы коэффициентов). В зависимости от аргумента modified в функции будет использоваться функция GaussStraightRun или ModifiedGaussStraightRun с аргументом $for\_det = True$. (функция просто перемножает элементы возвращаемого массива ведущих элементов)

- $inverse\_matrix(A)$ - Функция находит матрицу $^(-1)$, обратную к квадратной матрице A, методом Гаусса (Применяем прямой ход метода Гаусса к матрице A|E, затем левую подматрицу приводим к единичной, и результат есть правая подматрица)

- $calc\_condition\_number(A)$ - функция находит число обусловленности квадратной матрицы A

- $create\_LAES(x)$ - функция генерирует расширенную матрицу системы, заданной вариантом 2-4 приложения 2.

Код программы

```
import numpy as np
from numpy import linalg as LA
import sys

# # Классический метод Гаусса

# В данной функции реализован прямой ход метода Гаусса нахождения
# решения неоднородной СЛАУ с расширенной матрицей A из R^(n*n+1)
# Функция( вернет расширенную матрицу, получившуюся в результате)
#
# В случае, если for_det=True, функция вернет массив с ведущими элементами
# матрицы на каждом шаге прямого хода метода Гаусса, причем матрица A
# будет инетрпретироваться как матрица из R^(n*n) последним(
# элементом массива ведущих элементов будет число (-1)^k, где k - количество
# перестановок строк)
# Эта опция предусмотрена для нахожедния определителя матрицы A методом Гаусса

def GaussStraightRun(A, for_det=False):
    n = A.shape[0]

    # количество перестановок строк
    k = 0

    # массив ведущих элементов
    lead_elems = np.zeros((n+1,))

    # B - копия матрицы A, будет укорачиваться на каждом шаге прямого хода
    C = A.copy()
    B = C

    for i in range(n):
        # находим номер строки с ведущим элементом
        nonzero_indexes = (B[:, 0].nonzero())[0]
        if nonzero_indexes.shape[0] == 0:
            if for_det == True:
                return lead_elems
            else:
                return None
        lead_el_row_num = nonzero_indexes[0]

        # меняем первую строку и строку с ведущим элементом
        if (lead_el_row_num != 0):
            B[0], B[lead_el_row_num] = B[lead_el_row_num], B[0].copy()
            k += 1
        # сохраняем ведущий элемент
        lead_elems[i] = B[0][0]
        B[0] = B[0] / B[0][0]

        for j in range(1, B.shape[0]):
            B[j] = B[j] - B[0] * B[j][0]

        # укорачиваем"" матрицу
```

```
            B = B[1:, 1:]

    lead_elems[n] = (-1)**k

    if (for_det == True):
        return lead_elems
    else:
        return C

# В функции реализован обратный ход метода Гаусса
def GaussReverseRun(C):
    if type(C) == type(None):
        return None
    n = C.shape[0]
    x = np.zeros((n,))

    for i in range(n-1, -1, -1):
        x[i] = C[i][n] - C[i, :-1].dot(x)

    return x

# Функция объединяет прямой и обратный ход метода Гаусса
def GaussMethod(A):
    C = GaussStraightRun(A)
    return GaussReverseRun(C)

# # Метод Гаусса с выбором главного элемента

# В данной функции реализован прямой ход модифицированого метода Гаусса нахождения
# решения неоднородной СЛАУ с расширенной матрицей A из R^(n*n+1)
# Функция( вернет расширенную матрицу, получившуюся в результате)
#
# В случае, если for_det=True, функция вернет массив с ведущими элементами
# матрицы на каждом шаге прямого хода метода Гаусса, причем матрица A
# будет инетрпретироваться как матрица из R^(n*n) последним(
# элементом массива ведущих элементов будет число (-1)^k, где k - количество
# перестановок строк)
# Эта опция предусмотрена для нахожедния определителя матрицы A методом Гаусса

def ModifiedGaussStraightRun(A, for_det=False):
    n = A.shape[0]

    # массив нумерации переменных
    var_indexes = np.array(list(range(n)), dtype='int32')

    # количество перестановок столбцов
    k = 0

    # массив ведущих элементов
    lead_elems = np.zeros((n+1,))

    # C - копия матрицы A
    C = A.copy()

    for i in range(n):
```

```
        main_el_col_num = (np.fabs(C[i,:-1])).argmax()

        # меняем первый столбец и столбец с главным элементом
        if (main_el_col_num != i):
            C[:, i], C[:, main_el_col_num] = C[:, main_el_col_num], C[:, i].copy()
            var_indexes[i], var_indexes[main_el_col_num] = var_indexes[main_el_col_num], var_indexes[i]
            k += 1

        # запоминаем"" ведущий элемент
        lead_elems[i] = C[i][i]
        C[i] = C[i] / C[i][i]

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

    lead_elems[n] = (-1)**k

    if for_det == False:
        return C, var_indexes
    else:
        return lead_elems

def ModifiedGaussReverseRun(C, var_indexes):
    n = C.shape[0]
    x = np.zeros((n,))

    for i in range(n-1, -1, -1):
        x[i] = C[i][n] - C[i, :-1].dot(x)

    # возвращаем решение системы в соответствии с нумерацией
    return x[np.argsort(var_indexes)]

# Функция объединяет в себе прямой и обратный ход модифицированного метода Гаусса
def ModifiedGaussMethod(A):
    C, indexes = ModifiedGaussStraightRun(A)
    return ModifiedGaussReverseRun(C, indexes)


# ## Функции нахождения определителя матрицы, обратной матрицы и числа обусловленности


# Функция вычисляет определитель матрицы A
def det(A, modified=False):
    if modified == False:
        return GaussStraightRun(A, for_det=True).prod()
    else:
        return ModifiedGaussStraightRun(A, for_det=True).prod()

# Функция находит матрицу A^(-1), обратную к A
def inverse_matrix(A):
    n = A.shape[0]
    E = np.eye(n)
```

```
    AE = np.concatenate((A, E), axis=1)
    C = GaussStraightRun(AE)

    for i in range(n - 1, 0, -1):
        for j in range(i):
            C[j] = C[j] - C[i] * C[j][i]

    return C[:, n:]

# Функция вычисляет число обусловленности матрицы A
def calc_condition_number(A):
    inv_A = inverse_matrix(A)
    return LA.norm(A) * LA.norm(inv_A)


# ## Функция создания матрицы
# приложение( 2, пример 2, вариант 4)

# Функция генерирует расширенную матрице СЛАУ, заданной вариантом задания
def create_LAES(x):
    n = 100
    M = 4
    q = 1.001 - 2 * M * 10**(-3)

    # матрица коэффициентов системы
    A = np.zeros((n, n + 1))

    for i in range(n):
        for j in range(n):
            if (i == j):
                A[i][j] = (q - 1)**(i + j)
            else:
                A[i][j] = q**(i + j) + 0.1 * (j - i)

    # вычисляем столбец правой части системы
    vec_x = np.full(n, x)
    vec_x_div_i = np.array([x / i for i in range(1, n+1)])

    A[:, n] = vec_x * np.exp(vec_x_div_i) * np.cos(vec_x_div_i)

    return A

def main():
    mode = int(input('1 - выбор матрицы из варианта задания\n'
        '2 - ввод матрицы из файла\n'
        '3 - ввод матрицы через стандартный поток ввода\n'))

    if mode == 1:
        print("Выберите номер матрицы (1, 2, 3, 4):")
        num = input()
        if num == '4':
            x = float(input("Введите x:"))
            A = create_LAES(x)
        else:
            file_name = 'matrix_' + num + '.txt'
            A = np.loadtxt(file_name, delimiter=' ')
```

```
    elif mode == 2:
        file_name = input('Введите имя файла:  ')
        A = np.loadtxt(file_name, delimiter=' ')
    elif mode == 3:
        A = np.loadtxt(sys.stdin, delimiter=' ')
    else:
        return 1

    if A.shape[0] == A.shape[1]:
        print('Вы ввели квадратную матрицу. Для нее доступны следующие операции:\n'
            '1 - найти определитель матрицы\n'
            '2 - найти обратную матрицу\n'
            '3 - найти число обусловленности\n')
        cmds = input('Для выполнения операций введите строку с соответствующими цифрами\n'
                    'Пример(: при вводе 123 программа найдет определитель матрицы, обратную к ней и ее число обусловленности)')

        for cmd in cmds:
            if cmd == '1':
                print(f"Определитель матрицы: {det(A)}")
            elif cmd == '2':
                print(f"Обратная матрица:\n{inverse_matrix(A)}")
            elif cmd == '3':
                print(f"число обусловленности: {calc_condition_number(A)}")
    else:
        print('Вы ввели расширенную матрицу СЛАУ. Для нее доступны следующие операции:\n'
            '1 - найти определитель матрицы коэффициентов\n'
            '2 - найти обратную матрицу к матрице коэффициентов\n'
            '3 - найти число обусловленности матрицы коэффициентов\n'
            '4 - решить систему классическим методом Гаусса\n'
            '5 - решить систему методом Гаусса с выбором главного элемента\n')
        cmds = input('Для выполнения операций введите строку с соответствующими цифрами\n'
            'Пример(: при вводе 123 программа найдет определитель матрицы, обратную к ней и ее число обусловленности)\n')

        for cmd in cmds:
            if cmd == '1':
                print(f"Определитель матрицы: {det(A)}")
            elif cmd == '2':
                print(f"Обратная матрица:\n{inverse_matrix(A)}")
            elif cmd == '3':
                print(f"число обусловленности: {calc_condition_number(A)}")
            elif cmd == '4':
                print(f"Решение системы классическим методом Гаусса:")
                print(GaussMethod(A))
            elif cmd == '5':
                print(f"Решение системы модифицированным методом Гаусса:")
                print(ModifiedGaussMethod(A))

```

```
if __name__ == '__main__':
    main()
```

Тестирование программы

Корректность работы программы проверялась на вариантах задания с помощью модуля numpy.linalg библиотеки numpy (в частности, для проверки правильности решения использовалась функция numpy.linalg.solve). Все ответы сошлись.

Приложение 1. Вариант 11.

Первая система

ВНИМАНИЕ! Некоторые элементы матрицы были изменены в связи с вырожденностью исходной матрицы. В частности, был изменен элемент a_{24} (изначально был равен -3)

$$A = \begin{pmatrix} 4 & -3 & 1 & 5 \\ 1 & -2 & -2 & 3 \\ 3 & -1 & 2 & 0 \\ 2 & 3 & 2 & -8 \end{pmatrix}, f = \begin{pmatrix} 7 \\ 3 \\ -1 \\ 7 \end{pmatrix}$$

Обратная матрица:

$$A^{-1} = \begin{pmatrix} 5.3333e-01 & 2.2204e-16 & -6.0000e-01 & 3.3333e-01 \\ 1.2888e+00 & -6.6667e-01 & -1.8666e+00 & 5.5555e-01 \\ -1.5555e-01 & -3.3333e-01 & 4.6666e-01 & -2.2222e-01 \\ 5.7777e-01 & -3.3333e-01 & -7.3333e-01 & 1.1111e-01 \end{pmatrix}$$

Определитель матрицы коэффициентов:

$$\det(A) = 45.0$$

Решение системы классическим методом Гаусса:

$$x = (2, 5, -1, 3)^T$$

Решение системы модифицированным методом Гаусса:

$$x = (2, 5, -1, 3)^T$$

Число обусловленности = 36.338735582071415

Вторая система

ВНИМАНИЕ! Некоторые элементы матрицы были изменены в связи с вырожденностью исходной матрицы. В частности, был изменен элемент a_{33} (изначально был равен 5)

$$A = \begin{pmatrix} 2 & -2 & 1 & -1 \\ 1 & 2 & -1 & 1 \\ 4 & -10 & 6 & -5 \\ 2 & -14 & 7 & -11 \end{pmatrix}, f = \begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \end{pmatrix}$$

Обратная матрица:

$$A^{-1} = \begin{pmatrix} 0.33333333 & 0.33333333 & 0 & 0 \\ -2.04166667 & 1.83333333 & 0.5 & 0.125 \\ -3 & 2 & 1 & 0 \\ 0.75 & -1 & -0 & -0.25 \end{pmatrix}$$

Определитель матрицы коэффициентов:

$$\det(A) = -24.0$$

Решение системы методом Гаусса без выбора главного элемента:

$$x = (0.66666667, 0.16666667, 0, 0)^T$$

Решение системы методом Гаусса с выбором главного элемента:

$$x = (0.66666667, 0.16666667, 0, 0)^T$$

Число обусловленности = 115.47420346842262

Третья система

ВНИМАНИЕ! Некоторые элементы матрицы были изменены в связи с вырожденностью исходной матрицы. В частности, были изменены элементы a_{22} (изначально был равен -2) и a_{43} (изначально был равен 9)

$$A = \begin{pmatrix} 2 & -1 & 3 & 4 \\ 4 & -1 & 5 & 6 \\ 6 & -3 & 7 & 8 \\ 8 & -4 & 10 & 10 \end{pmatrix}, f = \begin{pmatrix} 5 \\ 7 \\ 9 \\ 11 \end{pmatrix}$$

Обратная матрица:

$$A^{-1} = \begin{pmatrix} -1.5 & 0.5 & 1 & -0.5 \\ -0.5 & 1 & -0.5 & 0 \\ 0.5 & 0 & -1.5 & 1 \\ 0.5 & 0 & 0.5 & -0.5 \end{pmatrix}$$

Определитель матрицы коэффициентов:

$$\det(A) = 8.0$$

Решение системы методом Гаусса без выбора главного элемента:

$$x = (-0.5, 0, 0, 1.5)^T$$

Решение системы методом Гаусса с выбором главного элемента:

$$x = (-0.5, 0, 0, 1.5)^T$$

Число обусловленности $= 72.02083032012335$

Приложение 2. Вариант 2-4

$M = 4, n = 100$
$q_M = 1.001 - 2 \cdot M \cdot 10^{-3}$

$$A_{ij} = \begin{cases} \mathrm{q}_M^{i+j} + 0.1 \cdot (j - i) & , i \neq j \\ (\mathrm{q}_M - 1)^{i+j} & , i = j \end{cases}$$

$f = \begin{pmatrix} f_1 \\ f_2 \\ \ldots \\ f_n \end{pmatrix}$, где $f_i = n \cdot e^{\frac{x}{i}} \cdot \cos x$

Определитель матрицы коэффициентов:

$$\det(A) = 5.528382539510184e - 27$$

Число обусловленности $= 19045.391466210534$

Для данной матрицы в связи с ее большим размером (100) компоненты вектора будут приведены с точность до 5 знака после запятой

$x = 1$

Решение системы классическим методом Гаусса
$(-14.79041, 0.017, 0.14376, 0.21837, 0.26578, 0.29839, 0.32216$
$0.34024, 0.35446, 0.36591, 0.3753, 0.38311, 0.38967, 0.39521$
$0.39991, 0.4039, 0.40728, 0.41012, 0.41247, 0.41439, 0.41592$
$0.41707, 0.41788, 0.41836, 0.41853, 0.41839, 0.41797, 0.41725$
$0.41626, 0.41499, 0.41344, 0.41163, 0.40953, 0.40717, 0.40453$
$0.40161, 0.39841, 0.39494, 0.39117, 0.38712, 0.38278, 0.37813$
$0.37318, 0.36793, 0.36236, 0.35647, 0.35025, 0.3437, 0.33682$
$0.32958, 0.322, 0.31405, 0.30573, 0.29704, 0.28796, 0.27849$
$0.26861, 0.25833, 0.24763, 0.23649, 0.22492, 0.2129, 0.20042$
$0.18748, 0.17405, 0.16013, 0.14571, 0.13078, 0.11533, 0.09934$
$0.0828, 0.0657, 0.04803, 0.02977, 0.01092, -0.00854, -0.02863$
$-0.04936, -0.07075, -0.0928, -0.11553, -0.13896, -0.1631, -0.18797$
$-0.21359, -0.23997, -0.26712, -0.29507, -0.32383, -0.35342, -0.38385$
$-0.41515, -0.44734, -0.48043, -0.51444, -0.54939, -0.58531, -0.62221$
$-0.66011, -0.69904)^T$

Решение системы модифицированным методом Гаусса:
$(-14.79041, 0.017, 0.14376, 0.21837, 0.26578, 0.29839, 0.32216$
$0.34024, 0.35446, 0.36591, 0.3753, 0.38311, 0.38967, 0.39521$
$0.39991, 0.4039, 0.40728, 0.41012, 0.41247, 0.41439, 0.41592$
$0.41707, 0.41788, 0.41836, 0.41853, 0.41839, 0.41797, 0.41725$
$0.41626, 0.41499, 0.41344, 0.41163, 0.40953, 0.40717, 0.40453$
$0.40161, 0.39841, 0.39494, 0.39117, 0.38712, 0.38278, 0.37813$
$0.37318, 0.36793, 0.36236, 0.35647, 0.35025, 0.3437, 0.33682$
$0.32958, 0.322, 0.31405, 0.30573, 0.29704, 0.28796, 0.27849$

$0.26861, 0.25833, 0.24763, 0.23649, 0.22492, 0.2129, 0.20042$
$0.18748, 0.17405, 0.16013, 0.14571, 0.13078, 0.11533, 0.09934$
$0.0828, 0.0657, 0.04803, 0.02977, 0.01092, -0.00854, -0.02863$
$-0.04936, -0.07075, -0.0928, -0.11553, -0.13896, -0.1631, -0.18797$
$-0.21359, -0.23997, -0.26712, -0.29507, -0.32383, -0.35342, -0.38385$
$-0.41515, -0.44734, -0.48043, -0.51444, -0.54939, -0.58531, -0.62221$
$-0.66011, -0.69904)^T$

$x = 2$

Решение системы классическим методом Гаусса
$(323.33317, -9.12311, -9.28415, -9.14357, -9.01976, -8.92764, -8.85927$
$-8.8072, -8.7662, -8.7327, -8.70424, -8.67908, -8.65595, -8.63391$
$-8.61223, -8.59034, -8.56779, -8.54418, -8.51921, -8.49261, -8.46413$
$-8.43357, -8.40073, -8.36543, -8.32753, -8.28685, -8.24326, -8.19661$
$-8.14677, -8.09361, -8.037, -7.9768, -7.91289, -7.84515, -7.77344$
$-7.69764, -7.61763, -7.53326, -7.44443, -7.35098, -7.2528, -7.14975$
$-7.04168, -6.92847, -6.80998, -6.68606, -6.55657, -6.42136, -6.28028$
$-6.13319, -5.97992, -5.82032, -5.65423, -5.48149, -5.30193, -5.11538$
$-4.92167, -4.72063, -4.51207, -4.29581, -4.07167, -3.83945, -3.59898$
$-3.35003, -3.09243, -2.82596, -2.55041, -2.26557, -1.97122, -1.66715$
$-1.35313, -1.02893, -0.69431, -0.34903, 0.00714, 0.37446, 0.75317$
$1.14354, 1.54584, 1.96031, 2.38725, 2.82693, 3.27963, 3.74564$
$4.22526, 4.71878, 5.22652, 5.74879, 6.2859, 6.83817, 7.40595$
$7.98956, 8.58935, 9.20567, 9.83888, 10.48933, 11.15741, 11.84349$
$12.54794, 13.27118)^T$

Решение системы модифицированным методом Гаусса:
$(323.33317, -9.12311, -9.28415, -9.14357, -9.01976, -8.92764, -8.85927$
$-8.8072, -8.7662, -8.7327, -8.70424, -8.67908, -8.65595, -8.63391$
$-8.61223, -8.59034, -8.56779, -8.54418, -8.51921, -8.49261, -8.46413$
$-8.43357, -8.40073, -8.36543, -8.32753, -8.28685, -8.24326, -8.19661$
$-8.14677, -8.09361, -8.037, -7.9768, -7.91289, -7.84515, -7.77344$
$-7.69764, -7.61763, -7.53326, -7.44443, -7.35098, -7.2528, -7.14975$
$-7.04168, -6.92847, -6.80998, -6.68606, -6.55657, -6.42136, -6.28028$
$-6.13319, -5.97992, -5.82032, -5.65423, -5.48149, -5.30193, -5.11538$
$-4.92167, -4.72063, -4.51207, -4.29581, -4.07167, -3.83945, -3.59898$
$-3.35003, -3.09243, -2.82596, -2.55041, -2.26557, -1.97122, -1.66715$
$-1.35313, -1.02893, -0.69431, -0.34903, 0.00714, 0.37446, 0.75317$
$1.14354, 1.54584, 1.96031, 2.38725, 2.82693, 3.27963, 3.74564$
$4.22526, 4.71878, 5.22652, 5.74879, 6.2859, 6.83817, 7.40595$
$7.98956, 8.58935, 9.20567, 9.83888, 10.48933, 11.15741, 11.84349$
$12.54794, 13.27118)^T$

Выводы

В ходе работы были реализованы классический и модифицированный методы Гаусса, а также функции нахождения определителя матрицы и обратной матрицы, основанные на прямом ходе метода Гаусса. При сравнении результатов нахождения решения СЛАУ классическим методом Гаусса и с выбором главного элемента отличий не выявлено, что объясняется небольшими коэффициентами матрицы.