Самарский национальный исследовательский университет
имени академика С.П. Королева
Естественнонаучный институт
Механико-математический факультет

Кафедра информатики и вычислительной математики

Отчет по лабораторной работе №4 «Численное решение систем линейных алгебраических уравнений итерационными методами»

Выполнила:
студентка группы 4345-020303D
С.С. Ильметов

Проверил:
доцент В.П. Сироченко

Самара-2024

1. РЕШЕНИЕ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ ИТЕРАЦИОННЫМ МЕТОДОМ ЗЕЙДЕЛЯ

1.1 Постановка задачи

Реализовать итерационный метод Зейделя в виде компьютерной программы для решения системы уравнений Ax = b.

Программа должны быть рассчитана для решения систем уравнений произвольного порядка.

Входные данные для программ: число уравнений n, матрица коэффициентов A, вектор правых частей b, параметр ε для окончания итераций, максимальное число итераций itmax (ограничение для числа итераций в случае медленной сходимости итераций или зацикливания).

Итерации прекращаются, если норма разности двух соседних итераций для x станет меньше ε, или, если число итераций сравняется с itmax.

Выходные данные: приближенное решение системы вектор x, количество итераций it, норма разности двух последних итераций для x, параметр ε для окончания итераций, максимальное число итераций itmax.

Выходные данные должны сопровождаться соответствующими комментариями (обозначениями). В случае равенства it и itmax сравнить норму разности двух последних итераций для x с ε, чтобы проверить, выполнилось ли условие окончания итераций с ε.

1.2 Краткое описание численного метода

Решение систем линейных алгебраических уравнений (СЛАУ) является одной из самых распространенных и важных задач вычислительной математики, т.к. к ней сводятся многочисленные научные и практические задачи.

Запишем систему n линейных алгебраических уравнений с n неизвестными:

$$\begin{gathered} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \cdots \\ a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n = b_i \\ \cdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n \end{gathered}$$

Часто систему записывают в векторно-матричной форме:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$ – квадратная матрица коэффициентов

$$X = (x_j) = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$ – векторы-столбцы неизвестных

$$b = (b_i) = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$ – свободные члены

Где i,j = $\overline{1,n}$

В матричном виде СЛАУ запишется:

AX=b

Метод Зейделя — это один из итерационных методов решения систем линейных уравнений (СЛАУ) вида AX=b

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \quad \text{i} = \overline{1,n}$$

Предположим, что известен

$$x^k = (x_1^k, x_2^k, \dots, x_n^k)$$

Нужно получить x^{k+1} приближение

Идея метода Зейделя заключается в последовательном уточнении приближенных значений решения системы линейных уравнений путем выражения каждой компоненты x_i нового вектора x^{k+1} через компоненты предыдущего вектора x^k.

Расчетная формула:

$x_i^{k+1} = (b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^k)/a_{ii} \quad a_{ii} \neq 0, \text{i} = \overline{1,n}$

На практике используются различные критерии остановки итераций метода Зейделя, например, достижение заданной точности $\left||x^{k+1} - x^k|\right| < \varepsilon$,

$$\varepsilon > 0 - \text{задано}$$

Достаточное условие сходимости метода Зейделя:

- *A* является симметричным положительно определенным
- A является строго или неприводимо доминирующим по диагонали.

Метод Гаусса–Зайделя иногда сходится, даже если эти условия не выполняются.

1.3 Листинг программы

Для реализации алгоритма использовался язык программирования Python в интегрированной среде разработки Visual Studio Code.

```
def norm(a):
    sum = 0
    for i in a:
        sum += i*i
    return sqrt(sum)
```

```
def seidel_solver(A, b, n, epsilon, itmax):
    x = [0]*n
    it = 0
    while it < itmax:
        x_new = x.copy()
        for i in range(n):
            s1 = sum(A[i][j] * x_new[j] for j in range(i))
            s2 = sum(A[i][j] * x[j] for j in range(i + 1, n))
            x_new[i] =  (b[i] - s1 - s2) / A[i][i]

        dx = norm(np.array(x_new) - np.array(x))

        if dx < epsilon:
            break

        x = x_new
        it += 1

    return x, it, dx
```

1.4 Результаты расчетов

Для проверки правильности работы программы решена тестовая система уравнений с расширенной матрицей:

$$\left(\begin{array}{ccc|c} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{array}\right)$$

Точное решение равно (1, 2, 3).

```
Матрица для метода Зейделя:
 [[1 0 0]
 [0 1 0]
 [0 0 1]]
Вектор b для метода Зейделя:
 [1, 2, 3]
Матрица для метода релаксации:
 [[1 0 0]
 [0 1 0]
 [0 0 1]]
Вектор b для метода релаксации:
 [1, 2, 3]

Введите параметр ε для окончания итераций: 0.001
Введите максимальное число итераций (itmax): 100

Метод Зейделя:
Решение (вектор x):
[1. 2. 3.]
Количество итераций (it): 1
Норма разности двух последних итераций для x: 0.0
```

Рисунок 1 – Результат выполнения программы

Решена вторая тестовая система уравнений с расширенной матрицей:

$$\left(\begin{array}{ccc|c} -1 & 0 & 0 & -1 \\ 0 & -1 & 0 & -2 \\ 0 & 0 & -1 & -3 \end{array}\right)$$

Точное решение равно (1, 2, 3).

```
Матрица для метода Зейделя:
 [[-1  0  0]
 [ 0 -1  0]
 [ 0  0 -1]]
Вектор b для метода Зейделя:
 [-1, -2, -3]
Матрица для метода релаксации:
 [[-1  0  0]
 [ 0 -1  0]
 [ 0  0 -1]]
Вектор b для метода релаксации:
 [-1, -2, -3]

Введите параметр ε для окончания итераций: 0.001
Введите максимальное число итераций (itmax): 100

Метод Зейделя:
Решение (вектор x):
[1. 2. 3.]
Количество итераций (it): 1
Норма разности двух последних итераций для x: 0.0
```

Рисунок 2 – Результат выполнения программы

Решена еще одна тестовая система уравнений с расширенной матрицей

$$\left(\begin{array}{ccc|c} 3 & 1 & 1 & 8 \\ 1 & 3 & 1 & 10 \\ 1 & 1 & 3 & 12 \end{array}\right)$$

Точное решение равно (1, 2, 3).

```
Матрица для метода Зейделя:
 [[3 1 1]
 [1 3 1]
 [1 1 3]]
Вектор b для метода Зейделя:
 [8, 10, 12]
Матрица для метода релаксации:
 [[3 1 1]
 [1 3 1]
 [1 1 3]]
Вектор b для метода релаксации:
 [8, 10, 12]

Введите параметр ε для окончания итераций: 0.001
Введите максимальное число итераций (itmax): 100

Метод Зейделя:
Решение (вектор x):
[0.99969073 1.9997941  3.00017172]
Количество итераций (it): 6
Норма разности двух последних итераций для x: 0.00038456148384592384
```

Рисунок 3 – Результат выполнения программы

В соответствии с номером варианта были решены итерационным методом конкретные системы уравнений

$$\left(\begin{array}{ccc|c} 2.56 & 0.67 & -1.78 & 1.14 \\ 0.67 & -2.67 & 1.35 & 0.66 \\ -1.78 & 1.35 & -1.55 & 1.72 \end{array}\right)$$

```
Матрица для метода Зейделя:
 [[ 2.56  0.67 -1.78]
 [ 0.67 -2.67  1.35]
 [-1.78  1.35 -0.55]]
Вектор b для метода Зейделя:
 [1.14 0.66 1.72]
Матрица для метода релаксации:
 [[ 2 -1  0]
 [-1  2 -1]
 [ 0 -1  2]]
Вектор b для метода релаксации:
 [ 0.33  1.   -0.33]

Введите параметр ε для окончания итераций: 0.01
Введите максимальное число итераций (itmax): 38

Метод Зейделя:
Решение (вектор x):
[-11.2790777  -10.74237069   7.00828703]
Количество итераций (it): 38
Норма разности двух последних итераций для x: 32.61947149038136

Внимание: метод Зейделя достиг максимального числа итераций без удовлетворения критерия с
ходимости.
Условие окончания итераций с ε не выполнено.
```

Рисунок 4 – Результат выполнения программы

Также решение проверено с помощью калькулятора онлайн. Получено следующее решение, где N_i – i-ая итерация.

38	-11.279	-10.742	7.008	4.131	6.219	-8.151

Рисунок 5 – Результат выполнения программы в онлайн-калькуляторе

Вывод: решение, полученное с помощью программы, совпадает с точным решением тестовых систем и с решением, полученным с помощью калькулятора онлайн. Можно сделать вывод, что программа работает верно

2. РЕШЕНИЕ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ ИТЕРАЦИОННЫМ МЕТОДОМ РЕЛАКСАЦИИ

2.1 Постановка задачи

Реализовать итерационный метод релаксации в виде компьютерной программы для решения системы уравнений Ax = b. В программе для метода релаксации подбирать оптимальное значение параметра релаксации для заданной системы уравнений с помощью пробных расчетов.

Программа должны быть рассчитана для решения систем уравнений произвольного порядка.

Входные данные для программ: число уравнений n, матрица коэффициентов A, вектор правых частей b, параметр ε для окончания итераций, максимальное число итераций itmax (ограничение для числа итераций в случае медленной сходимости итераций или зацикливания).

Итерации прекращаются, если норма разности двух соседних итераций для x станет меньше ε, или, если число итераций сравняется с itmax.

Выходные данные: приближенное решение системы вектор x, количество итераций it, норма разности двух последних итераций для x, параметр ε для окончания итераций, максимальное число итераций itmax.

Выходные данные должны сопровождаться соответствующими комментариями (обозначениями). 2 В случае равенства it и itmax сравнить норму разности двух последних итераций для x с ε, чтобы проверить, выполнилось ли условие окончания итераций с ε.

2.2 Краткое описание численного метода

Метод релаксации (метод верхней релаксации) — это итерационный метод численного решения системы линейных уравнений, который основан на принципе постепенного приближения к точному решению путем последовательного уточнения значений неизвестных переменных. Метод релаксации относится к группе простых итерационных методов, в которых каждое новое приближение вычисляется на основе предыдущего.

Расчетная формула:

$$x_i^{k+1} = w(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{k+1} - \sum_{j=i+1}^{n} a_{ij}x_j^k)/a_{ii} + (1-w)x_i^k \quad \text{i} = \overline{1,n} \quad \text{k=0,1…}$$

$$0 < w < 2 - \text{параметр релаксации}$$

Идея метода релаксации заключается в добавлении некоторого "релаксационного" параметра к каждому уравнению, чтобы ускорить сходимость. Для каждого уравнения этот параметр может быть разным. Параметр обычно выбирается таким образом, чтобы минимизировать число итераций, необходимых для достижения заданной точности.

На практике используются различные критерии остановки итераций метода релаксации, например, достижение заданной точности $\left||x^{k+1} - x^k|\right| < \varepsilon$,

$$\varepsilon > 0 - \text{задано}$$

Оценка оптимального значения параметра релаксации

Ax=b $w \in (0; 2)$

Оптимальное значение параметра релаксации w будет лежать в диапазоне от 0 до 2.

$$\Delta w = \frac{2}{l+1}$$

Необходимо провести пробные расчеты, делая k итераций метода релаксации с w_s(каждого значения w из диапазона), где , $s = \overline{1,l}$

$$x^0, x_s^1, x_s^2, \dots, x_s^k, S = \overline{1,l}$$

Для каждой последовательности x^k вычислить норму разности $\left\|x_s^k - x_s^{k-1}\right\|$. Далее необходимо выбрать оптимальное значение параметра релаксации w, соответствующее наименьшей норме разности

$$\min\left\|x_s^k - x_s^{k-1}\right\| = \left\|x_*^k - x_*^{k-1}\right\|$$

2.3 Листинг программы

Для реализации алгоритма использовался язык программирования C++ в интегрированной среде разработки Visual Studio.

```
  def relaxation_solver(A, b, n, omega, epsilon, itmax):

    x = [0]*n
    it = 0
    while it < itmax:
        x_new = x.copy()
        for i in range(n):
            s1 = sum(A[i][j] * x_new[j] for j in range(i))
            s2 = sum(A[i][j] * x[j] for j in range(i + 1, n))
            x_new[i] = (1 - omega) * x[i] + omega * ( (b[i] - s1 - s2)/ A[i][i] )

        dx = norm(np.array(x_new) - np.array(x))

        if dx < epsilon:
            break

        x = x_new
        it += 1

    return x, it, dx


def optimal_omega(A, b, omega_start=0.1, omega_end=2.0, omega_step=0.001):
    best_omega = omega_start
    dx_min = 0
```

```
    for omega in np.arange(omega_start, omega_end + omega_step, omega_step):
        x = [0]*n
        it = 0
        dx_min = 1
        while it < 25:
            x_new = x.copy()
            for i in range(n):
                s1 = sum(A[i][j] * x_new[j] for j in range(i))
                s2 = sum(A[i][j] * x[j] for j in range(i + 1, n))
                x_new[i] = (1 - omega) * x[i] + omega * ( (b[i] - s1 - s2)/ A[i][i]
)

            dx = norm(np.array(x_new) - np.array(x))

            if dx < dx_min:
                dx_min = dx
                best_omega = omega

            x = x_new
            it += 1

    return best_omega
```

2.4 Результаты расчетов

Для проверки правильности работы программы решена тестовая система уравнений с расширенной матрицей:

$$\left(\begin{array}{ccc|c} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{array}\right)$$

Точное решение равно (1, 2, 3).

```
Матрица для метода Зейделя:
 [[1, 0, 0], [0, 1, 0], [0, 0, 1]]
Вектор b для метода Зейделя:
 [1, 2, 3]
Матрица для метода релаксации:
 [[1, 0, 0], [0, 1, 0], [0, 0, 1]]
Вектор b для метода релаксации:
 [1, 2, 3]

Введите параметр ε для окончания итераций: 1.e-7
Введите максимальное число итераций (itmax): 100

Метод Зейделя:
Решение (вектор x):
[1.0, 2.0, 3.0]
Количество итераций (it): 1
Норма разности двух последних итераций для x: 0.0

Оптимальное значение параметра релаксации (omega): 1.0

Метод релаксации:
Решение (вектор x):
[1.0, 2.0, 3.0]
Количество итераций (it): 1
Норма разности двух последних итераций для x: 0.0
```

Рисунок 6 – Результат выполнения программы

Решена вторая тестовая система уравнений с расширенной матрицей:

$$\left(\begin{array}{ccc|c} -1 & 0 & 0 & -1 \\ 0 & -1 & 0 & -2 \\ 0 & 0 & -1 & -3 \end{array}\right)$$

Точное решение равно (1, 2, 3).

```
Матрица для метода Зейделя:
 [[-1, 0, 0], [0, -1, 0], [0, 0, -1]]
Вектор b для метода Зейделя:
 [-1, -2, -3]
Матрица для метода релаксации:
 [[-1, 0, 0], [0, -1, 0], [0, 0, -1]]
Вектор b для метода релаксации:
 [-1, -2, -3]

Введите параметр ε для окончания итераций: 0.01
Введите максимальное число итераций (itmax): 100

Метод Зейделя:
Решение (вектор x):
[1.0, 2.0, 3.0]
Количество итераций (it): 1
Норма разности двух последних итераций для x: 0.0

Оптимальное значение параметра релаксации (omega): 1.0

Метод релаксации:
Решение (вектор x):
[1.0, 2.0, 3.0]
Количество итераций (it): 1
Норма разности двух последних итераций для x: 0.0
```

Рисунок 7 – Результат выполнения программы

Решена еще одна тестовая система уравнений с расширенной матрицей

$$\left(\begin{array}{ccc|c} 3 & 1 & 1 & 8 \\ 1 & 3 & 1 & 10 \\ 1 & 1 & 3 & 12 \end{array}\right)$$

Точное решение равно (1, 2, 3).

```
Матрица для метода Зейделя:
 [[3, 1, 1], [1, 3, 1], [1, 1, 3]]
Вектор b для метода Зейделя:
 [8, 10, 12]
Матрица для метода релаксации:
 [[3, 1, 1], [1, 3, 1], [1, 1, 3]]
Вектор b для метода релаксации:
 [8, 10, 12]

Введите параметр ε для окончания итераций: 0.01
Введите максимальное число итераций (itmax): 100

Метод Зейделя:
Решение (вектор x):
[0.9972182550210968, 2.0000008363006327, 3.000926969559423]
Количество итераций (it): 5
Норма разности двух последних итераций для x: 0.0025935066554554956

Оптимальное значение параметра релаксации (omega): 1.1

Метод релаксации:
Решение (вектор x):
[1.003487520148661, 1.999530861585072, 2.9988344381723824]
Количество итераций (it): 5
Норма разности двух последних итераций для x: 0.0035078509904344674
```

Рисунок 8 – Результат выполнения программы

В соответствии с номером варианта были решены итерационным методом конкретные системы уравнений

$$\left(\begin{array}{ccc|c} 2.56 & 0.67 & -1.78 & 1.14 \\ 0.67 & -2.67 & 1.35 & 0.66 \\ -1.78 & 1.35 & -0.55 & 1.72 \end{array}\right)$$

```
Матрица для метода релаксации:
 [[2.56, 0.67, -1.78], [0.67, -2.67, 1.35], [-1.78, 1.35, -6.55]]
Вектор b для метода релаксации:
 [1.14, 0.66, 1.72]

Введите параметр ε для окончания итераций: 0.01
Введите максимальное число итераций (itmax): 100

Оптимальное значение параметра релаксации (omega): 0.9

Метод релаксации:
Решение (вектор x):
[0.2582941040964753, -0.3788162541939168, -0.4094151915480059]
Количество итераций (it): 3
Норма разности двух последних итераций для x: 0.009886192271910135
```

Рисунок 9 – Результат выполнения программы