МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«Дальневосточный федеральный университет»
(ДВФУ)

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ

Департамент математического и компьютерного моделирования

Отчет
Вариант №4

по дисциплине
«Вычислительная математика»

Направление подготовки
02.03.01 «Математика и компьютерные науки»

Выполнила студентка
группы Б9122-02.03.01сцт
Винницкая Д.
(ФИО) *(подпись)*

« 15 » декабря 20 24 г.

г. Владивосток
2024

Цель работы

Цель лабораторной работы заключается в решении системы линейных алгебраических уравнений (СЛАУ) методом Якоби и методом последовательной верхней релаксации с заданной точностью $\varepsilon = 10^{-4}$. Также необходимо провести анализ сходимости этих методов и сравнить их эффективность.

Постановка задачи

Рассматривается система линейных уравнений вида:

$$A\mathbf{x} = \mathbf{b},$$

где:

$$A = \begin{bmatrix} 6.22 & 1.42 & -1.72 & 1.91 \\ 1.42 & 5.33 & 1.11 & -1.82 \\ -1.72 & 1.11 & 5.24 & 1.42 \\ 1.91 & -1.82 & 1.42 & 6.55 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 7.53 \\ 6.06 \\ 8.05 \\ 8.06 \end{bmatrix}.$$

Требуется:

1. Реализовать метод Якоби.
2. Реализовать метод последовательной верхней релаксации (SOR) с параметром ω.
3. Найти решение системы с заданной точностью $\varepsilon = 10^{-4}$.
4. Вывести количество итераций, необходимых для сходимости.
5. Сравнить эффективность двух методов.

1 Теоретические сведения

Итерационные методы решения систем линейных алгебраических уравнений (СЛАУ) являются альтернативой прямым методам (таким как метод Гаусса). Прямые методы обеспечивают точное решение системы за конечное число шагов, однако их вычислительная сложность и чувствительность к ошибкам округления делают их менее эффективными для больших систем. Итерационные методы позволяют получить приближённое решение с заданной точностью, что делает их особенно полезными для больших и разреженных матриц.

В данной работе рассматриваются два итерационных метода:

1. **Метод Якоби** (простая итерация), который является базовым методом, подходящим для симметричных матриц.
2. **Метод верхней релаксации (SOR)**, расширяющий метод Зейделя за счёт введения параметра ускорения ω.

Основная идея итерационных методов заключается в разбиении матрицы A системы $Ax = b$ на части, что позволяет на каждом шаге вычислять новое приближение $x^{(k+1)}$ через уже известное $x^{(k)}$.

Метод Якоби

Метод Якоби предполагает, что матрица A может быть разложена на диагональную часть D и внедиагональную часть R, то есть $A = D + R$. Тогда система $Ax = b$ переписывается в виде:

$$Dx = b - Rx, \tag{1}$$

откуда выводится итерационная формула:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j \neq i} a_{ij} x_j^{(k)} \right), \quad i = 1, \ldots, n. \tag{2}$$

Метод Якоби хорошо подходит для разреженных матриц, но сходимость достигается только при выполнении следующих условий:

- Матрица A должна быть диагонально доминирующей, то есть $|a_{ii}| > \sum_{j \neq i} |a_{ij}|$ для всех i.

- Либо A должна быть симметричной и положительно определённой.

Метод верхней релаксации (SOR)

Метод последовательной верхней релаксации (SOR) является усовершенствованием метода Зейделя, который в свою очередь улучшает метод Якоби за счёт использования нового значения $x_i^{(k+1)}$ уже на текущей итерации. Итерационная формула метода SOR выглядит следующим образом:

$$x_i^{(k+1)} = (1-\omega)x_i^{(k)} + \frac{\omega}{a_{ii}} \left(b_i - \sum_{j<i} a_{ij} x_j^{(k+1)} - \sum_{j>i} a_{ij} x_j^{(k)} \right), \quad i = 1, \ldots, n. \tag{3}$$

Здесь ω — параметр релаксации, значение которого выбирается из интервала $0 < \omega < 2$. При $\omega = 1$ метод SOR совпадает с методом Зейделя. Значения $\omega > 1$ ускоряют сходимость, но при слишком больших ω метод может стать неустойчивым.

Критерий остановки

Для обоих методов итерационный процесс прекращается, когда достигается заданная точность ε, то есть:

$$\|x^{(k+1)} - x^{(k)}\|_\infty < \varepsilon. \tag{4}$$

Данный критерий проверяет максимальную разницу между соответствующими компонентами вектора на двух последовательных итерациях.

Преимущества и ограничения

Итерационные методы имеют следующие преимущества:

- Экономия памяти за счёт работы с разреженными матрицами.
- Возможность настройки точности ε в зависимости от требований задачи.

Ограничения:

- Зависимость от структуры матрицы A и её свойств.
- Более медленная сходимость по сравнению с прямыми методами для плохо обусловленных матриц.

Код программы

```python
import numpy as np


def jacobi_method(A, b, tol=1e-4, max_iterations=1000):
    n = len(b)
    x = np.zeros(n)
    x_new = np.zeros(n)
    iterations = 0

    while True:
        for i in range(n):
            x_new[i] = (b[i] - sum(A[i, j] * x[j] for j in range(n) if j != i))
                / A[i, i]
        iterations += 1

        if np.linalg.norm(x_new - x, ord=np.inf) < tol or iterations >=
            max_iterations:
            break
        x = x_new.copy()

    return x_new, iterations


def sor_method(A, b, omega, tol=1e-4, max_iterations=1000):
    n = len(b)
    x = np.zeros(n)
    iterations = 0

    while True:
        x_old = x.copy()

        for i in range(n):

            sigma = sum(A[i, j] * x[j] for j in range(i))
            + sum(A[i, j] * x_old[j] for j in range(i + 1, n))

            x[i] = (1 - omega) * x_old[i] +
            (omega / A[i, i]) * (b[i] - sigma)
        iterations += 1

        if np.linalg.norm(x - x_old, ord=np.inf) <
        tol or iterations >= max_iterations:
            break

    return x, iterations


def main():
    A = np.array([[6.22, 1.42, -1.72, 1.91],
                  [1.42, 5.33, 1.11, -1.82],
                  [-1.72, 1.11, 5.24, 1.42],
                  [1.91, -1.82, 1.42, 6.55]])
```

```
51      b = np.array([7.53, 6.06, 8.05, 8.06])
52      tol = 1e-4
53
54      print("Решение методомЯкоби :")
55      jacobi_solution, jacobi_iterations = jacobi_method(A, b, tol)
56      print(f"Решение: {jacobi_solution}")
57      print(f"Количество итераций: {jacobi_iterations}")
58
59      omega = 1.1
60      print("\Решениеn методомверхнейрелаксации    (SOR):")
61      sor_solution, sor_iterations = sor_method(A, b, omega, tol)
62      print(f"Решение: {sor_solution}")
63      print(f"Количество итераций: {sor_iterations}")
64
65      print("\Сравнениеn сходимости:")
66      print(f"Метод Якоби: {jacobi_iterations} итераций")
67      print(f"Метод верхнейрелаксации : {sor_iterations} итераций")
68
69
70  if __name__ == "__main__":
71      main()
```

Листинг 1: Реализация итерационных методов решения СЛАУ»

2 Описание

Программа решает системы линейных алгебраических уравнений (СЛАУ) вида $Ax = b$ с использованием двух методов: метода Якоби и метода верхней релаксации (SOR). Рассмотрим, как работает программа и ключевые этапы её алгоритма.

Назначение программы

Программа предназначена для численного решения СЛАУ вида $Ax = b$ итерационными методами. Она сравнивает скорость сходимости и точность двух методов: метода Якоби и метода верхней релаксации (SOR). Оба метода возвращают приближённое решение вектора x с заданной точностью $\varepsilon = 10^{-4}$.

Основные этапы работы программы:

- **Инициализация данных:** Задаются матрица A и вектор b, а также параметры точности (tol) и максимального числа итераций ($\mathrm{max\_iterations}$).
- **Метод Якоби:** Выполняется последовательное уточнение решений с использованием фиксированного начального приближения.
- **Метод верхней релаксации (SOR):** Выполняется уточнение решений с использованием параметра релаксации $\omega = 1.1$.
- **Сравнение методов:** Программа оценивает и выводит количество итераций, необходимых для достижения заданной точности.

Функция $\mathrm{jacobi\_method}$

Функция реализует метод Якоби, который основывается на итерационном пересчёте решения.

Алгоритм работы:

1. Задаётся начальное приближение x, равное нулевому вектору.
2. На каждой итерации пересчитываются новые значения элементов вектора x по формуле:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j \neq i} a_{ij} x_j^{(k)} \right), \quad i = 1, \ldots, n. \tag{5}$$

3. Проверяется условие остановки:

$$\|x^{(k+1)} - x^{(k)}\|_\infty < \varepsilon. \tag{6}$$

Если условие выполнено или достигнуто максимальное число итераций, итерационный процесс прекращается.

4. Функция возвращает приближённое решение x и число итераций.

Особенности:

- Метод Якоби требует, чтобы матрица A была диагонально доминирующей для гарантированной сходимости.
- Простая структура алгоритма делает его удобным для реализации, но скорость сходимости может быть медленной.

Функция sor_method

Функция реализует метод верхней релаксации (SOR), который улучшает метод Зейделя за счёт параметра релаксации ω.

Алгоритм работы:

1. Задаётся начальное приближение x, равное нулевому вектору.
2. На каждой итерации пересчитываются новые значения элементов вектора x по формуле:

$$x_i^{(k+1)} = (1-\omega)x_i^{(k)} + \frac{\omega}{a_{ii}}\left(b_i - \sum_{j<i} a_{ij}x_j^{(k+1)} - \sum_{j>i} a_{ij}x_j^{(k)}\right). \tag{7}$$

3. Проверяется условие остановки:

$$\|x^{(k+1)} - x^{(k)}\|_\infty < \varepsilon. \tag{8}$$

Если условие выполнено или достигнуто максимальное число итераций, итерационный процесс прекращается.

4. Функция возвращает приближённое решение x и число итераций.

Особенности:

- Параметр релаксации ω ускоряет сходимость, но его значение должно быть оптимальным ($0 < \omega < 2$).
- Метод SOR часто быстрее метода Якоби, особенно для больших систем.

Функция main

Функция main выполняет основную часть работы программы.

Шаги:

1. Определяется матрица A и вектор b для СЛАУ.
2. Задаются параметры точности tol и параметр релаксации omega.
3. Вызывается функция jacobi_method для решения системы методом Якоби. Результаты (решение и число итераций) выводятся.
4. Вызывается функция sor_method для решения системы методом SOR. Результаты выводятся.
5. Сравниваются результаты двух методов, включая количество итераций.

Особенности:

- Функция демонстрирует преимущества метода SOR по сравнению с методом Якоби.
- Чётко разделены этапы решения и анализа.

Выводы

Программа наглядно демонстрирует работу двух итерационных методов: Якоби и SOR. Метод верхней релаксации показывает лучшую скорость сходимости, что делает его предпочтительным для задач, где важна эффективность. Код легко модифицируется для работы с другими системами линейных уравнений.

Основная система для варианта VI

Программа решает систему линейных алгебраических уравнений (СЛАУ) вида $Ax = b$, используя два итерационных метода: метод Якоби и метод верхней релаксации (SOR). Ниже представлены входные данные и полученные результаты.

Входные данные

Заданы следующие матрица A и вектор b:

$$A = \begin{pmatrix} 6.22 & 1.42 & -1.72 & 1.91 \\ 1.42 & 5.33 & 1.11 & -1.82 \\ -1.72 & 1.11 & 5.24 & 1.42 \\ 1.91 & -1.82 & 1.42 & 6.55 \end{pmatrix}, \quad b = \begin{pmatrix} 7.53 \\ 6.06 \\ 8.05 \\ 8.06 \end{pmatrix}.$$

Параметры для итерационных методов:

- Заданная точность: $\varepsilon = 10^{-4}$.
- Максимальное количество итераций: 1000.
- Параметр релаксации для метода SOR: $\omega = 1.1$.

Полученные результаты

Метод Якоби:

- Приближённое решение:

$$x_{\text{Якоби}} \approx \begin{pmatrix} 1.2872 \\ 0.6938 \\ 1.6231 \\ 0.6961 \end{pmatrix}.$$

- Количество итераций: 32.

Метод верхней релаксации (SOR):

- Приближённое решение:

$$x_{\text{SOR}} \approx \begin{pmatrix} 1.2877 \\ 0.6934 \\ 1.6235 \\ 0.6957 \end{pmatrix}.$$

- Количество итераций: 14.

Сравнение сходимости

Метод верхней релаксации (SOR) показал значительно более быструю сходимость по сравнению с методом Якоби:

- Метод Якоби: 32 итерации.
- Метод SOR: 14 итераций.

Выводы

Результаты показывают, что метод верхней релаксации (SOR) является более эффективным с точки зрения скорости сходимости, особенно при выборе оптимального параметра ω. Тем не менее, оба метода дают приближённое решение, близкое к точному, с заданной точностью $\varepsilon = 10^{-4}$.

3 Заключение

В ходе выполнения лабораторной работы были изучены и реализованы два итерационных метода решения систем линейных алгебраических уравнений (СЛАУ): метод Якоби и метод верхней релаксации (SOR). Целью работы было сравнение их эффективности с точки зрения скорости сходимости и точности.

Основные результаты

- Оба метода успешно решают СЛАУ с заданной точностью $\varepsilon = 10^{-4}$.
- Метод Якоби потребовал 32 итерации для достижения решения:

$$x_{\text{Якоби}} \approx \begin{pmatrix} 1.2872 \\ 0.6938 \\ 1.6231 \\ 0.6961 \end{pmatrix}.$$

- Метод верхней релаксации (SOR) с параметром $\omega = 1.1$ достиг решения за 14 итераций:

$$x_{\text{SOR}} \approx \begin{pmatrix} 1.2877 \\ 0.6934 \\ 1.6235 \\ 0.6957 \end{pmatrix}.$$

- Метод SOR оказался быстрее метода Якоби, что обусловлено использованием параметра релаксации, который ускоряет процесс сходимости.

Выводы

1. Метод верхней релаксации (SOR) является более эффективным для задач, где важна скорость решения, благодаря использованию параметра релаксации ω.
2. Метод Якоби остаётся хорошим выбором для задач, где требуется простота реализации, особенно при разреженной структуре матрицы A.

3. Сходимость обоих методов зависит от свойств матрицы A. Для метода Якоби важно выполнение диагональной доминантности, а для метода SOR требуется оптимизация параметра ω.