ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет программной инженерии и компьютерной техники
Системное и прикладное программное обеспечение

Дисциплина «Вычислительная математика»

Отчет к лабораторной работе № 1
«Решение системы линейных алгебраических уравнений»
Вариант 20. Метод Гаусса-Зейделя

Выполнил:
Студент группы P3213
Харламов Александр Сергеевич

Преподаватель:
Малышева Татьяна Алексеевна

Санкт-Петербург
2022 год

Цель работы

Познакомиться с задачами, решаемыми в рамках дисциплины “Вычислительная математика”, на примере решения систем линейных алгебраических уравнений с помощью итерационного метода Гаусса-Зейделя. Проанализировать полученные результаты и оценить погрешность.

Задание лабораторной работы

Создать программу, которая решает систему линейных алгебраических уравнений методом Гаусса-Зейделя.

Общие требования

- В программе численный метод должен быть реализован в виде отдельной подпрограммы или класса, в который входные/выходные данные передаются в качестве параметров.

- Размерность матрицы n<=20 (задается из файла или с клавиатуры - по выбору конечного пользователя).

- Должна быть реализована возможность ввода коэффициентов матрицы, как с клавиатуры, так и из файла (по выбору конечного пользователя).

Требования к итерационному методу

- Точность задается с клавиатуры/файла

- Проверка диагонального преобладания (в случае, если диагональное преобладание в исходной матрице отсутствует, сделать перестановку строк/столбцов до тех пор, пока преобладание не будет достигнуто). В случае невозможности достижения диагонального преобладания - выводить соответствующее сообщение.

- Вывод вектора неизвестных

- Вывод количества итераций, за которое было найдено решение.

- Вывод вектора погрешностей

Описание метода, расчетные формулы

Метод Гаусса-Зейделя является модификацией метода простой итерации и обеспечивает более быструю сходимость к решению системы уравнений. Идея метода: при вычислении компонента вектора неизвестных на n-ной итерации используются компоненты вектора неизвестных, уже вычисленные на данной итерации. Значения остальных берутся из предыдущей итерации.

Рабочая формула метода Гаусса-Зейделя:

$$x_i^{(k+1)} = \frac{b_i}{a_{ii}} - \sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}} x_j^{k+1} - \sum_{j=i+1}^{n} \frac{a_{ij}}{a_{ii}} x_j^k, i = 1, 2, \dots, n$$

Итерационный процесс продолжается до тех пор, пока модуль разности компонента вектора на данной операции и компонента вектора на предыдущей операции не превысит заданный ε.

Приведение матрицы к диагональному преобладанию реализуется при помощи алгоритма Куна нахождения наибольшего паросочетания:

1) Составляется список, где каждому элементу сопоставляется массив индексов элементов строки матрицы, превышающих сумму модулей других элементов

2) Для каждого столбца запускается поиск в глубину с целью найти увеличивающую цепь.

3) Просматриваются все индексы и проверяется, ведёт ли следующий индекс к “ненасыщенному”, либо индекс уже “насыщен”, но можно нйти увеличивающую цепь.

4) Цепь представлена в виде массива, где каждому индексу будет соответствовать новый для перестановки строк.

Листинг программы

https://github.com/itookyourboo/itmo_comp_math/tree/master/lab1_gauss-seidel

```
def converges(vars_old, vars_new, eps):
    n = len(vars_old)
    return sum((vars_new[i] - vars_old[i]) ** 2 for i in range(n)) ** 0.5 < eps



def seidel_iteration(coefficients, vars, values):
    n = len(coefficients)
    vars = vars[:]

    for i in range(n):
        s = sum(coefficients[i][j] * vars[j] for j in range(n) if i != j)
        vars[i] = (values[i] - s) / coefficients[i][i]

    return vars



def solve(coefficients, values, epsilon, precision=PRECISION):
    print('>>> Input matrix:')
```

```
    io_helper.print_matrix(coefficients, values)

    if not make_matrix_diagonal_dominance(coefficients, values):
        print("Couldn't make matrix diagonal dominant")
        return

    print('>>> Matrix after reordering:')
    io_helper.print_matrix(coefficients, values)

    n = len(coefficients)
    x = [0 for _ in range(n)]

    for i in range(1, LIMIT + 1):
        x_new = seidel_iteration(coefficients, x, values)
        if converges(x, x_new, epsilon):
            print('>>> Solution:')
            io_helper.print_x_values(x_new, precision)
            print('>>> Iterations:', i)
            print('>>> Errors:')
            io_helper.print_errors(x, x_new)
            break

        x = x_new
    else:
        print("Couldn't compute :(\nProbably diverges")
```

Примеры и результаты работы программы

Предпоследние два параметра epsilon и precision задают погрешность и количество знаков после запятой при выводе результата. По умолчанию равны 1e-6 и 6 соответственно.

Матрица 3x3, ввод с файла

Файл: tests/matrix_3.txt

3 -2 1 6

-1 2 4 9

1 3 2 2

Запуск: python main.py file tests/matrix_3.py 1e-8 4

Результат:

>>> Input matrix:

3	-2	1	| 6
-1	2	4	| 9
1	3	2	| 2

>>> Matrix after reordering:

3 -2 1 | 6

1 3 2 | 2

-1 2 4 | 9

>>> Solution:

0.122 -1.3415 2.9512

>>> Iterations: 20

>>> Errors:

8.12195879995592e-09 2.7726789753046432e-09 6.441500666198863e-10

Матрица 3x3, ввод с клавиатуры

Запуск: python main.py input 1e-8 6

3 -2 1 6

-1 2 4 9

1 3 2 2

Результат:

>>> Input matrix:

3 -2 1 | 6

-1 2 4 | 9

1 3 2 | 2

>>> Matrix after reordering:

3 -2 1 | 6

1 3 2 | 2

-1 2 4 | 9

>>> Solution:

0.121951 -1.341463 2.95122

>>> Iterations: 20

>>> Errors:

8.12195879995592e-09 2.7726789753046432e-09 6.441500666198863e-10

Матрица 7x7, ввод с файла

Запуск: python main.py file tests/matrix_7.txt

Результат:

>>> Input matrix:

1 22 7 8 1 -2 2 | 41

-9 0 71 9 0 8 14 | -2

1 10 13 100 8 -6 2 | 6

8 17 13 10 18 -10 90 | 12

92 12 0 8 1 17 2 | 18

-9 3 2 8 99 11 3 | -10

18 -20 2 4 -8 110 0 | 7

>>> Matrix after reordering:

92 12 0 8 1 17 2 | 18

1 22 7 8 1 -2 2 | 41

-9 0 71 9 0 8 14 | -2

1 10 13 100 8 -6 2 | 6

-9 3 2 8 99 11 3 | -10

18 -20 2 4 -8 110 0 | 7

8 17 13 10 18 -10 90 | 12

>>> Solution:

-0.130389 1.978737 -0.059099 -0.083699 -0.209384 0.433633 -0.120943

>>> Iterations: 10

>>> Errors:

1.3094005024694155e-07 1.0906750236294727e-07 7.475182458444403e-08 2.9603215684348427e-08 4.1271211892457416e-08 3.853810948140795e-08 1.1081970383020057e-08

Выводы

Я познакомился с итерационным методом Гаусса-Зейделя решения СЛАУ и алгоритмом Куна для перестановки строк матрицы с целью диагонального преобладания.