Университет ИТМО

Факультет программной инженерии и компьютерной техники

Лабораторная работа №1

По дисциплине «Вычислительная математика»

Вариант 4. Метод простых итераций

Выполнил:

Студент группы P3208

Дашкевич Егор Вячеславович

Преподаватель:

Машина Екатерина Алексеевна

Санкт-Петербург

2024

Оглавление

Цель работы

Ознакомиться со способами автоматизации решения СЛАУ, составить программу для решения СЛАУ при помощи метода простых итераций

Текст задания

- В программе численный метод должен быть реализован в виде отдельной подпрограммы/метода/класса, в который исходные/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).

  Для итерационных методов должно быть реализовано:

  • 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: $x1, x2, \dots, xn$

  • Вывод количества итераций, за которое было найдено решение.

  • Вывод вектора погрешностей: $|xi\ (k) - xi\ (k-1)\ |$

Описание метода, расчетные формулы

Вектор решений можно представить в виде:

$$x_i = \sum_{j=1}^{n} c_{ij} x_j + d_i \ , \qquad i = 1, 2, \dots, n$$

$$c_{ij} = \begin{cases} 0, \ \text{при } i = j \\ -\frac{a_{ij}}{a_{ii}}, \ \text{при } i \neq j \end{cases} \qquad d_i = \frac{b_i}{a_{ii}} \quad i = 1, 2, \dots, n$$

Рабочая формула итерации:

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

Где k – номер итерации

Условие преоблодания диагональных элементов:

$$|a_{ii}| \geq \sum_{j \neq i} |a_{ij}| \ , \qquad i = 1, 2, \dots, n$$

Листинг программы

Полное решение доступно на GitHub

Реализация метода простых итераций:

```
1 usage  new *
def do_simple_iteration(c: [[float]], d: [float], x: [float]) -> [float]:
    ans_x = []
    for i in range(len(d)):
        ans_x.append(d[i] + sum([x[j] * c[i][j] for j in range(len(c[i]))]))
    return ans_x


1 usage  new *
def iteration_algo(mrx: [[float]], precision: float):
    n = len(mrx)
    d = [mrx[i][-1] / mrx[i][i] for i in range(n)]
    iter_count = 0
    x = d.copy()
    c = []
    for i in range(n):
        c_row = []
        for j in range(n):
            if i == j:
                c_row.append(0)
            else:
                c_row.append(-(mrx[i][j] / mrx[i][i]))
        c.append(c_row)
    last_err = sys.float_info.max
    while True:
        iter_count += 1
        x_1 = do_simple_iteration(c, d, x)
        errors = [abs(x_1[i] - x[i]) for i in range(len(x))]
        if max(errors) < precision:
            print(f"Found answer with given precision!\n"
                  f"answer vector:")
            for i in range(n):
                print(f"x_{i + 1}: {x_1[i]}")
            print("errors vector:")
            for i in range(n):
                print(f"|x_{i + 1}({iter_count}) - x_{i + 1}({iter_count - 1})|: {errors[i]}")
            print(f"Answer was found in {iter_count} iterations!")
            return
        if max(errors) > last_err:
            print("Answer cannot be found, aborting", file=sys.stderr)
            return
        last_err = max(errors)
        x = x_1
```

Примеры и результаты работы

Корректная работа со чтением с консоли:

```
Select an input type:
1. Manual
2. By file
3. Random matrix generation
Input type (number or name): 1
Enter your matrix. Please note the following:
- matrix should be n + 1 by n in size, extra column being the answers vector
- separate entries inside a row via single space
- n can't be more than 20
Enter rows one by one below:
2 2 10 14
2 10 1 13
10 1 1 12
Your matrix:
--  --  --  --
10   1   1  12
 2  10   1  13
 2   2  10  14
--  --  --  --
Enter desired precision: 0,01
```

```
Found answer with given precision!
answer vector:
x_1: 0.999568
x_2: 0.99946
x_3: 0.9993159999999999
errors vector:
|x_1(5) - x_1(4)|: 0.0019320000000000448
|x_2(5) - x_2(4)|: 0.0024600000000001288
|x_3(5) - x_3(4)|: 0.0030840000000000867
Answer was found in 5 iterations!

Process finished with exit code 0
```

Пример работы с ошибками ввода:

```
Select an input type:
1. Manual
2. By file
3. Random matrix generation
Input type (number or name): фыв
Invalid input, please try again
Input type (number or name): 3
Choose your way to enter n (should be <= 20):
1. Manual
2. By file
Input way: 1
n: ф
n is not a valid number
n: 0
n is not in range
n: 3
Your matrix:
--------  --------  -------  -------
105.975    12.6209  92.0686  54.6689
 78.9995  183.411   34.8724  43.6983
 32.5777   31.8041  86.7366  35.7069
--------  --------  -------  -------
Enter desired precision:
```

Пример с матрицей неудовлетворяющей критерию:

```
Select an input type:
1. Manual
2. By file
3. Random matrix generation
Input type (number or name): 2
Reading matrix from file, please notice:
- matrix should be n + 1 by n in size, extra column being the answers vector
- separate entries inside a row via single space
- n can't be more than 20
Enter a file name:matrix.txt
No diagonal dominance presence, result may not be correct
Your matrix:
-  -  --  --
2  2  10  14
1  1  10  12
2  1  10  13
-  -  --  --
Enter desired precision: 0,01
Answer cannot be found, aborting

Process finished with exit code 0
```

Вывод

В ходе выполнения работы удалось на примере решения СЛАУ разобрать методы автоматизации решения задач. Получилось для случая метода простых итераций перенести алгоритм решения «с бумаги» в программу, отработать учет ошибок ввода пользователя и погрешностей.