Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники
Направление подготовки: 09.03.04 — Системное и прикладное программное обеспечение
Дисциплина «Вычислительная математика»

Лабораторная работа №1

Метод Гаусса-Зейделя

Выполнил:

Капарулин Тимофей Иванович

Преподаватель:

Машина Екатерина Алексеевна

г.Санкт-Петербург 2024 г.

Цель работы

Написать программу для решения системы линейных алгебраических уравнений методом Гаусса-Зейделя с матрицей размероности до 20.

Описание метода

Общее положение

Метод Гаусса-Зейделя является модификацией метода простой итерации, обеспечивает более быструю сходимость к решению систем уравнений.

Алгоритм решения

- Так же как и в методе простых итераций строится эквивалентная СЛАУ и берется какое-то начальное приближение (обычно вектор правых частей).

$$x_1 = c_{11}x_1 + \ldots + c_{1n}x_n + d_1$$
$$\ldots$$
$$x_{n1} = c_{n1}x_1 + \ldots + c_{nn}x_n + d_n$$

$$x^0 = (d_1, \ldots, d_n)$$

- Далее необходимо проверить условие преобладания диагональных элементов:

$$|a_{ii}| \geq \sum_{(j \neq i)} |a_{ij}|, \quad j = 1,2,\ldots,n$$

  При этом хотя бы для одного уравнения неравенство должно выполняться строго. Это условие является достаточными для сходимости метода, но оно не являются необходимыми, т. е. для некоторых систем итерации сходятся и при нарушении этого условия.

- Далее начинается итерационный процесс, на k+1 итерации вычисляется вектор $x^{(k+1)}$:

$$x_1^{(k+1)} \to x_1^k \; x_2^k \; \ldots \; x_{(n-1)}{}^k \; x_n^k$$
$$x_2^{(k+1)} \to x_1^{(k+1)} \; x_2^k \; \ldots \; x_{(n-1)}{}^k \; x_n^k$$
$$\ldots$$
$$x_n^{(k+1)} \to x_1^{(k+1)} \; x_2^{(k+1)} \; \ldots \; x_{(n-1)}{}^{(k+1)} \; x_n^k$$

  Общая формулы для i-ого элемента на k+1 шаге итерации:

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

- Итерационный процесс продолжается пока

$$| x_i^{(k+1)} - x_i^k | > \epsilon$$

Листинг программы

```python
self.KMatrix = np.empty((self.dims, self.dims+1))

for line in range(self.dims):
    self.KMatrix[line] = np.concat(( -self.matrix[line], [self.answ[line]]))
    self.KMatrix[line] /= self.matrix[line][line]
    self.KMatrix[line, line] = 0

self.solution = np.concat(( self.KMatrix[:, self.dims], [1] ))

while self.variance >= self.epsilon:
    self.iter_cnt+=1
    old_solution = np.copy(self.solution)

    for line in range(self.dims):
        self.solution[line] = np.sum(self.KMatrix[line] * self.solution)

    self.variance = np.max( np.abs(self.solution - old_solution) )

self.solution = self.solution[:-1]
```

[github]()

Пример работы программы

- Пример 1

$$\begin{cases} x_1 + 2x_2 = 3 \\ 2x_1 + 4x_2 = 6 \end{cases}$$

При вводе данной системы будет выведена ошибка, так как отсутствует диагональное преобладание, хотя сама по себе система решается и имеет бесконечно много корней.

- Пример 2

$$\begin{cases} 2x_1 + 2x_2 + 10x_3 = 14 \\ 10x_1 + x_2 + x_3 = 12 \\ 2x_1 + 10x_2 + x_3 = 13 \end{cases}$$

При вводе с $\varepsilon = 0.01$ данной системы будет полученно корректное решение:

Полученное решение: (1.00017808, 0.99993686, 0.99997701)
Вектор погрешностей: (0.00050192, 0.00199286, 0.00029819)

- Пример 3

$$\begin{cases} 2x_1 - 3x_2 + 12x_3 + x_4 - 2x_5 = 10 \\ 8x_1 + 2x_2 - 3x_3 + 2x_4 + x_5 = 5 \\ -1x_1 + 4x_2 + 2x_3 - 3x_4 + 10x_5 = -2 \\ 3x_1 - 15x_2 + 6x_3 + 4x_4 - 1x_5 = 8 \\ 5x_1 + 3x_2 - 2x_3 + 15x_4 + 4x_5 = 12 \end{cases}$$

При вводе с $\varepsilon = 0.001$ данной системы будет полученно корректное решение:

Полученное решение: (9.99625707, 4.99891342, −2.00089554, 8.00049032 12)
Вектор погрешностей: (0.00023075, 0.000873, 0.00027292, 0.00065038 0.00012258)

Выводы

В данной работе был реализован метод Гаусса-Зейделя решения систем линейных уравнений. Метод был протестирован на различных примерах, включая системы с единственным решением, с бесконечным множеством решений и без решений. Результаты показали, что реализованный алгоритм успешно справляется с поставленной задачей и находит решения в пределах допустимых погрешностей.

Вследствие более быстрой сходимости, работает лучше метода простых итераций, однако является и более трудоемким. Так же одной из проблем является проверка условия сходимости. (больше сравнений с другими методами и ограничения)