Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
факультет программной инженерии и компьютерной техники

Лабораторная работа №2

по дисциплине

Вычислительная математика

Выполнил:
Студент группы P3230
Толстых М.А.

Преподаватель:
Перл И. А.
Перл О. В.

Санкт-Петербург, 2024

Описание метода

Метод простых итераций.

Изначально имеем набор данных: количество неизвестных (n) и расширенную матрицу коэффициентов (A|B) СЛАУ, а также точность ε.

Для применения метода простых итераций требуется проверить условие сходимости итерационного процесса (условие преобладания диагональных коэффициентов): диагональные элементы матрицы A удовлетворяют ли условию:

$$|a_{ii}| > \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}|$$

, т.е. если модули диагональных коэффициентов для каждого уравнения системы больше суммы модулей всех остальных коэффициентов (не считая свободных членов)

Если в исходной системе все элементы, стоящие на главной диагонали, по модулю больше, чем сумма модулей остальных элементов в этой же строке (столбце) матрицы A, то для приведения к нужному виду в левой части оставляют только диагональные элементы, а остальные переносят в правую часть и каждое уравнение делят на диагональные элементы.

Аналогичным образом поступают и тогда, когда диагонального преобладания можно добиться перестановкой уравнений. В случае если диагонального преобладания добиться не удалось метод простых итераций не применим.

Соответственно, если проверка оказалась удачной возьмем любое начальное приближение $X^{(0)} = (x_1^{(0)}, x_2^{(0)}, x_3^{(0)})$, например, $x_1^{(0)} = 0,\ x_2^{(0)} = 0, x_3^{(0)} = 0$.

Далее вычисляем новое приближение решения $X^{(1)} = \left(x_1^{(1)}, x_2^{(1)}, x_3^{(1)}\right)$, подставляя значения из предыдущего приближения в уравнения СЛАУ. Формула итерационного процесса будет следующей:

$$x_i^{(k)} = \frac{1}{a_{ii}}\left(b_i - \sum\nolimits_{\substack{j=1 \\ i\neq j}}^{n} a_{ij} \times x_j^{(k-1)}\right), i = \overline{1, n}$$

k – номер приближения

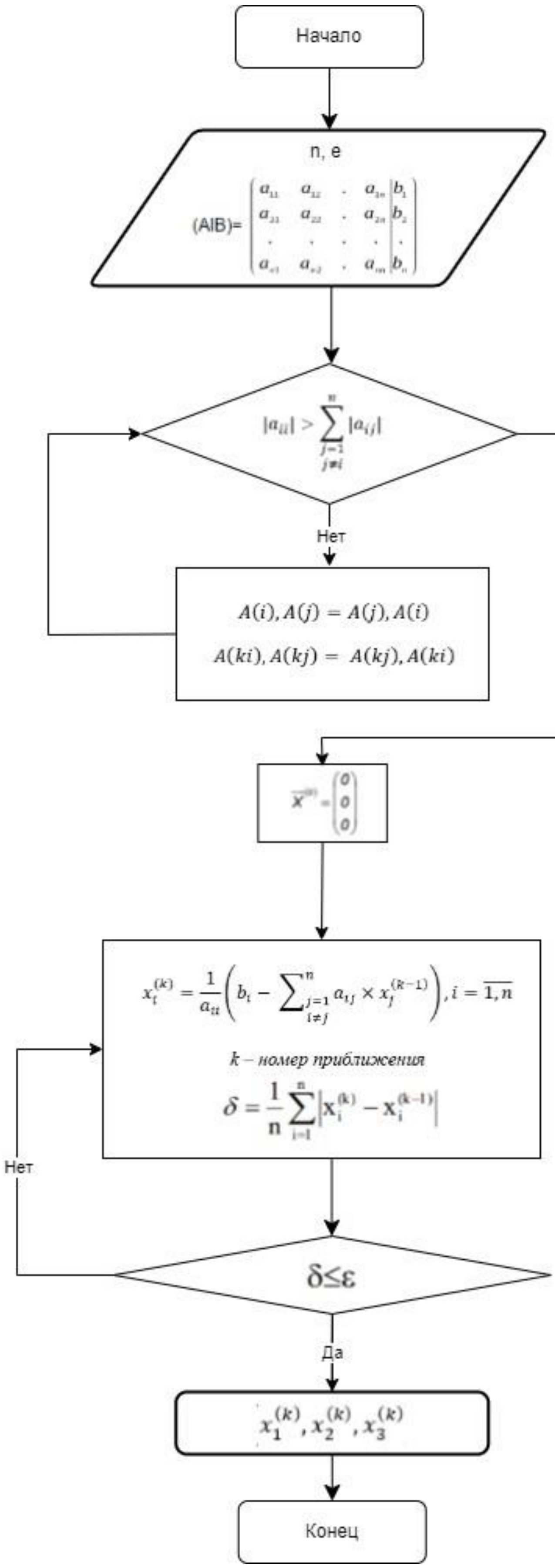
Оцениваем достигнутую точность по уравнению (условие сходимости):

$$\delta = \max_{1\leq i\leq 3}\left|x_i^{(1)} - x_i^{(0)}\right| = \max\left(\left|x_1^{(1)} - x_1^{(0)}\right|, \left|x_2^{(1)} - x_2^{(0)}\right|, \left|x_3^{(1)} - x_3^{(0)}\right|\right)$$

$$\delta = \frac{1}{n}\sum_{i=1}^{n}\left|x_i^{(k)} - x_i^{(k+1)}\right|$$

Итерационный процесс нужно продолжать, пока не выполнится $\delta \leq \varepsilon$.

Блок-схема численного метод



Код

```
class Result:
    isMethodApplicable = True
    errorMessage = ""

    def solveBySimpleIterations(n, matrix, epsilon):
        def checkDiagonalDominance(matrix):
            for i in range(n):
                row_sum = sum(abs(matrix[i][j]) for j in range(n-1) if j != 
i)
                if abs(matrix[i][i]) <= row_sum:
                    for j in range(n-1):
                        if j != i and abs(matrix[j][j]) > row_sum:
                            matrix[i], matrix[j] = matrix[j], matrix[i]
                            for k in range(n):
                                matrix[k][i], matrix[k][j] = matrix[k][j], 
matrix[k][i]
                            return True
                    return False
            return True
        print(checkDiagonalDominance(matrix))
        x = [0] * n
        converged = False

        if not checkDiagonalDominance(matrix):
            Result.isMethodApplicable = False
            Result.errorMessage = "The system has no diagonal dominance for 
this method. Method of the simple iterations is not applicable."
            return None

        while not converged :
            x_new = [0] * n  # Инициализация вектора приближенных значений
            # Итерационный процесс: обновляем значения вектора x_new по 
формуле метода простых итераций
            for i in range(n):
                s = sum(matrix[i][j] * x[j] for j in range(n-1) if j != i)
                x_new[i] = (matrix[i][-1] - s) / matrix[i][i]
            # Проверка условия сходимости: все новые значения достаточно 
близки к предыдущим
            if all(abs(x_new[i] - x[i]) < epsilon for i in range(n)):
                converged = True

            x = x_new   # Обновляем вектор приближенных значений

        return x
```

Метод простых итераций. Если для текущей матрицы нет диагонального преобладания, вам следует попытаться найти его путем перестановки столбцов или / и строк. Если после такой операции преобладание диагонали по-прежнему отсутствует, должно быть напечатано следующее сообщение:

"The system has no diagonal dominance for this method. Method of the simple iterations is not applicable.". Для этого задайте значение переменной isMethodApplicable и сообщение об ошибке.

Примеры работы программы

1. Тест на простой матрице с диагональным преобладанием из трех неизвестных

```
>? 3
>? 1 0.25 0.25 0.25
>? 0.25 1 0.25 0.5
>? 0.25 0.25 1 0.75
>? 0.0001
-2.0343810319900513e-05
0.33331298828125
0.6666463203728199
```

2. Тест на простой матрице с диагональным преобладанием из двух неизвестных

```
>? 2
>? 3 4 5
>? 23 5 8
>? 0.0001
-2.866679109877568e+39
-2.205137776828899e+38
```

3. Тест с матрицей без диагонального преобладания. Но выполнилось условие преобладания диагональных коэффициентов путем перестановки столбцов или / и строк

```
>? 2
>? 1 2 3
>? 4 5 6
>? 0.001
3.0
-1.2
```

4. Тест с матрицей без диагонального преобладания. Но не выполнилось условие преобладания диагональных коэффициентов даже путем перестановки столбцов или / и строк, поэтому вывелась ошибка на экран

```
>? 2
>? 2 1 6
>? 5 1 17
>? 0.0001
The system has no diagonal dominance for this method. Method of the simple iterations is not applicable.
```

5. Тестовый случай с нулевым определителем матрицы (нет единственного решения)

```
>? 2
>? 1 2 3
>? 2 4 6
>? 0.0001
0.0
0.0
```

Вывод

Этот метод простых итераций для решения СЛАУ может быть удобен, поскольку это одношаговый итерационный процесс. Но метод требует обязательного условия диагонального преобладания матрицы (обработку данного условия и как оно влияет на использование метода можно увидеть в примерах 3 и 4). Используя данный метод можно получить последовательность приближённых значений, сходящуюся к точному решению системы (на примерах 1 и 2 можно заметить множество цифр, которое является нецелой частью значения).
Полученные решения не являются точными значениями неизвестных, а приближенными, в отличии, например, от прямых методов (таких как метод Гаусса, метод Крамера и метод квадратных корней). Но с помощью итерационных методов возможно решать эффективнее задачи большой размерности за приемлемое время с минимумом операций. Это связанно с возможностью существенного использования разреженности матриц.

Метод простых итераций схож с методом Зейделя. Метод Зейделя тоже является итерационным методом. Его можно даже назвать модификацией метода простых итераций, поскольку условия сходимости метода Зейделя те же.

Метод Зейделя при вычислении (k+1)-ого приближения неизвестного $x_i (i > 1)$ используется уже вычисленные ранее (k+1)-ое приближение неизвестных.

Вычисления по методу Зейделя:

$$\begin{cases} x_1^{(k+1)} = c_{11}x_1^{(k)} + c_{12}x_2^{(k)} + \cdots + c_{1n}x_n^{(k)} + d_1 \\ x_2^{(k+1)} = c_{21}x_1^{(k+1)} + c_{22}x_2^{(k)} + \cdots + c_{2n}x_n^{(k)} + d_2 \\ x_2^{(k+1)} = c_{21}x_1^{(k+1)} + c_{22}x_2^{(k+1)} + \cdots + c_{2n}x_n^{(k)} + d_2 \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots.. \\ x_n^{(k+1)} = c_{n1}x_1^{(k+1)} + c_{n2}x_2^{(k+1)} + \cdots + c_{n-1,n-1}x_{n-1}^{(k+1)} + c_{n,n}x_n^{(k)} + d_n \end{cases}$$

Однако, области сходимости методов лишь частично совпадают. В случаях, когда оба метода сходятся, метод Зейделя дает лучшую сходимость, чем метод простых итераций.

Алгоритмическая сложность метода:

Основные затраты на i-й итерации – порядка $O(n^2)$ операций – состоят в умножении матрицы А на вектор текущего приближения x_i.

Анализ численной ошибки в самом численном методе:

Анализ численной ошибки в численном методе простых итераций включает в себя такие факторы, как ошибки округления, потому что при каждой итерации происходит округление чисел, что может привести к накоплению ошибок округления. Это особенно заметно при большом количестве итераций. потерю значимости, начальное приближение.

Потеря значимости, так как при вычислении следующего приближения может происходить потеря значимости, особенно если приближения близки к 1 или к 0.

Неудачный выбор начального приближения также может привести к увеличению численной ошибки. Если начальное приближение далеко от корня уравнения, метод может сходиться медленно или вообще расходиться.