|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатики и систем управления

КАФЕДРА Теоретической информатики и компьютерных технологий

**Лабораторная работа № 1**

«Решение СЛАУ с трехдиагональной матрицей методом прогонки»

по курсу «Численные методы»

Выполнил:

студент группы ИУ9-62Б

Марченко Андрей

Проверила:

Домрачева А. Б.

Москва, 2024

1. **Цель**

Целью данной работы является изучение накопления погрешности в решении системы линейных алгебраических уравнений (СЛАУ) с трехдиагональной матрицей методом прогонки.

1. **Постановка задачи**

**Дано:** , где , – трехдиагональная матрица.

**Найти:** Решение СЛАУ (вектор ) с помощью метода прогонки при исходных .

Для достижения цели работы были поставлены следующие задачи:

* ознакомление с теорией метода прогонки;
* реализация алгоритма поиска решения СЛАУ методом прогонки на языке программирования Python;
* нахождение погрешности решения и оценка точности результата.

1. **Основные теоретические сведения**

Метод прогонки является одним из способов решения систем линейных алгебраических уравнений вида , где - трехдиагональная матрица. Данный метод представляет собой вариацию метода последовательного исключения неизвестных системы.

**Описание алгоритма:**

Пусть – массив элементов под главной диагональю, – массив элементов главной диагонали, – массив элементов над главной диагональю.

Данная матрица задает следующую СЛАУ:

Из данной системы получаем:

Произведя замену , получаем .

Подставляем полученные значения во второе уравнение системы:

Получаем:

Произведя замену , получаем .

Продолжая аналогичные рассуждения, получаем формулы для нахождения :

где

вычисляется следующим образом:

Вычисление называется прямым ходом метода прогонки, а вычисление – обратным ходом метода прогонки.

**Достаточные условия метода прогонки:**

**Оценка погрешности для решения СЛАУ при отсутствии точного решения:**

Необходимо найти решение СЛАУ методом прогонкой – вектор . Для оценки погрешности вычисления вычисляем:

где – искомый вектор ошибок.

Тогда

Точное решение .

Для проверки точности реализации использовалось сравнение с известным результатом.

1. **Реализация**

Листинг 1. Метод прогонки для решения СЛАУ с трехдиагональной матрицей

from decimal import Decimal, getcontext  
  
getcontext().prec = 20  
  
def show\_vector(s):  
 for i in range(0, len(s)):  
 print(s[i], end=" ")  
 print()  
  
  
def process\_algorithm(a, b, c, d):  
 n = len(d)  
 alpha = [Decimal(0.0)] \* n  
 beta = [Decimal(0.0)] \* n  
  
 alpha[0] = -c[0] / b[0]  
 beta[0] = d[0] / b[0]  
 for i in range(1, n - 1):  
 alpha[i] = -c[i] / (alpha[i - 1] \* a[i - 1] + b[i])  
 beta[i] = (d[i] - a[i - 1] \* beta[i - 1]) / (alpha[i - 1] \* a[i - 1] + b[i])  
  
 x = [Decimal(0.0)] \* n  
 # x[n-1] = beta[n-1]  
 x[n - 1] = (d[n - 1] - a[n - 2] \* beta[n - 2]) / (a[n - 2] \* alpha[n - 2] + b[n - 1])  
 for i in range(n - 2, -1, -1):  
 x[i] = alpha[i] \* x[i + 1] + beta[i]  
 return x  
  
  
def check\_conditions(a, b, c):  
 n = len(b)  
 flag = True  
 for i in range(1, n - 1):  
 if not (abs(b[i]) >= abs(a[i - 1]) + abs(c[i])):  
 flag = False  
 break  
  
 if not (abs(b[0]) / abs(c[0]) >= 1 and abs(b[n-1]) / abs(c[n-2]) >= 1):  
 flag = False  
  
 return flag  
  
  
def main():  
 n = 4  
  
 a = c = [Decimal(1.0 / 3.), Decimal(1.0 / 3.), Decimal(1.0 / 3.)]  
 b = [Decimal(4.0 / 3.), Decimal(4.0 / 3.), Decimal(4.0 / 3.), Decimal(4.0 / 3.)]  
 d = [Decimal(5.0 / 3.), Decimal(6.0 / 3.), Decimal(6.0 / 3.), Decimal(5.0 / 3.)]  
  
 x\_expected = [Decimal(1.0), Decimal(1.0), Decimal(1.0), Decimal(1.0)]  
  
 if not check\_conditions(a, b, c):  
 print("Conditions not completed")  
 else:  
 print("Conditions completed")  
  
 print()  
 x = process\_algorithm(a, b, c, d)  
  
 print("x: ", end="")  
 show\_vector(x)  
  
 error = [Decimal(0.0)] \* n  
 for i in range(n):  
 error[i] = abs(x[i] - x\_expected[i])  
  
 print("error:", error)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

1. **Тестирование**

Для тестирования полученной программы в качестве трехдиагональной матрицы была выбрана следующая матрица:

В качестве вектора :

Таким образом, СЛАУ имеет вид:

В результате работы программы (см. листинг 1) получены значения:

Чтобы продемонстрировать ненулевую погрешность, разделим коэффициенты матрицы и вектора на 3. Тогда имеем такие входные данные:

Очевидно, что ожидаемый ответ не изменится.

В результате работы программы (см. листинг 1) получены значения:

Как видно выше, вектор погрешности не является нулевым, что связано с использованием типа данных Decimal с точностью в 20 знаков после запятой и нецелых значений в начальной матрице.

1. **Вывод**

В ходе выполнения лабораторной работы был изучен метод решения СЛАУ с трехдиагональной матрицей – метод прогонки. Алгоритм был реализован на языке программирования Python.

Для метода прогонки можно отметить эффективность, обусловленную хранением лишь части данных матрицы (ненулевые диагональные элементы). У данного метода отсутствует методологическая погрешность, однако имеет место вычислительная. В представленной реализации присутствует несущественная погрешность, т.е. решение имеет несущественные отличия от ответа. Это связано с особенностью использования чисел с плавающей точкой, а также усечением разрядной сетки результатов вычисления, что и приводит к вычислительной погрешности.