МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«Дальневосточный федеральный университет» (ДВФУ)

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ

Департамент математического и компьютерного моделирования

ОТЧЁТ по лабораторной работе № 3

«Метод квадратного корня»

Вариант № 8

Выполнила: студент гр. Б9122-02.03.01сцт

Ф.И.О.

Ильяхова Алиса Алексеевна

Проверил: преподаватель

Ф.И.О.

Павленко Елизавета Робертовна

Владивосток

2024

Цели работы:

Решить систему линейных алгебраических уравнений в виде Ax = b методом квадратного корня.

Ход работы:

1. Находим матрицу U по следующим формулам:
   - Задаём первый элемент $u_{11} = \sqrt{a_{11}}$.
   - При i = 1 вычисляем первую строку матрицы U: $u_{1j} = a_{1j} / u_{11}\ \forall j > 1$.
   - Если i = j находим $u_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} u^2_{ki}}$.
   - Если i < j находим $u_{ij} = (a_{ij} - \sum_{k=1}^{i-1} u_{ki} * u_{kj}) / u_{ii}$.
2. Найти транспонированную U^T.
3. Решить СЛАУ в два этапа:
   - Из системы $U^Ty = b$ находим вектор значений y.
   - Вычислив массив “y” решаем СЛАУ вида Ux = y.
   - Полученный массив x будет являться решением исходной системы Ax = b.
4. Сравнить полученные результаты с точным решением x^* тестовых СЛАУ, по-другому отладить свой алгоритм на тестах, приведенных в таблице 1.
5. После отладки программы решить два СЛАУ и вывести их.

Основное:

1) Функции:

Функция **holetskiy** реализует метод Холецкого для разложения симметричной положительно определенной матрицы. Метод Холецкого позволяет представить матрицу A в виде произведения $A = LL^T$, где L - нижняя треугольная матрица.

1. Инициализация: создается матрица matrixU (в данном случае, это нижняя треугольная матрица) с нулями, такой же размерности, как и исходная матрица original.
2. Первый элемент: вычисляется первый элемент на главной диагонали matrixU как квадратный корень из первого элемента исходной матрицы.
3. Заполнение первой строки: заполняется первая строка matrixU, деля элементы исходной матрицы на первый элемент matrixU.
4. Основной цикл: вложенные циклы проходят по всем элементам матрицы:
   - Если индексы равны (диагональные элементы), то вычисляется значение как квадратный корень из разности между элементом исходной матрицы и суммой квадратов предыдущих элементов текущего столбца.
   - Если индекс строки меньше индекса столбца (верхняя часть матрицы), то элемент вычисляется по формуле, учитывающей уже заполненные значения.
5. Возврат: функция возвращает матрицу matrixU, которая является нижней треугольной матрицей.

Функция **solution** решает систему линейных уравнений Ax = b с использованием разложения Холецкого.

1. Разложение: вызывается функция holetskiy, чтобы получить нижнюю треугольную матрицу hM из исходной матрицы matrix.
2. Решение системы: сначала решается система $L^T y = b$ с помощью функции solve из библиотеки NumPy, где L^T - это транспонированная матрица Холецкого. Затем решается система Lx = y для нахождения вектора x.
3. Возврат: Функция возвращает решение вектора x.

Этот код реализует метод Холецкого для решения систем линейных уравнений и включает тестирование на нескольких примерах. Он эффективен для симметричных положительно определенных матриц, что является требованием для использования метода Холецкого.

2) Проверка по тестовым СЛАУ:

```
Решение системы A1: [[ 6.]
 [-5.]
 [-4.]]
Разность с ожидаемым решением: [[0.0000000e+00]
 [0.0000000e+00]
 [8.8817842e-16]]
----------------------------------------
Решение системы A2: [[ 0.8]
 [-2. ]
 [ 1. ]]
Разность с ожидаемым решением: [[ 0.00000000e+00]
 [-4.44089210e-16]
 [ 2.22044605e-16]]
```

3) Результаты СЛАУ по заданию:

```
Решение системы A3: [[ 0.56206926]
 [-0.44359823]
 [ 0.13461121]]
----------------------------------------
Решение системы A4: [[ 1.45653917]
 [-1.93309996]
 [ 3.46970779]
 [ 1.51312749]]
```

4) Вывод:

На основе разложения Холецкого была разработана функция для решения системы линейных уравнений Ax = b. Метод продемонстрировал свою эффективность и точность, что видно из результатов тестов.

Метод квадратного корня имеет несколько преимуществ:

- Он требует меньше операций по сравнению с другими методами, такими как метод Гаусса, особенно для больших систем.
- Позволяет эффективно работать с симметричными положительно определенными матрицами, что часто встречается в практических задачах.

5) Листинг программы:

```
import numpy as np
from numpy.linalg import solve

def holetskiy(original: np.matrix) -> np.matrix:
    matrixU = np.matrix(np.zeros(original.shape))
    matrixU[0, 0] = np.sqrt(original[0, 0])
    for i in range(1, original.shape[0]):
        matrixU[0, i] = original[0, i] / matrixU[0, 0]
    for i in range(original.shape[0]):
        for j in range(original.shape[1]):
            if i == j:
                matrixU[i, j] = np.sqrt(original[i, j] -
sum(matrixU[k, i] ** 2 for k in range(i)))
            elif i < j:
                matrixU[i, j] = (original[i, j] - sum(matrixU[k, i]
* matrixU[k, j] for k in range(i))) / matrixU[i, i]
    return matrixU

def solution(matrix, b):
    hM = holetskiy(matrix)
    y = solve(hM.T, b)
    x = solve(hM, y)
    return x

#матрицы для проверки
mA1 = np.matrix("81 -45 45; -45 50 -15; 45 -15 38")
b1 = np.matrix("531; -460; 193")
x1 = np.matrix("6; -5; -4")
x_c1 = solution(mA1, b1)
print("Решение системы A1:", x_c1)
print("Разность с ожидаемым решением:", x1 - x_c1)
print("-----------------------------------------")

mA2 = np.matrix("6.25 -1 0.5; -1 5 2.12; 0.5 2.12 3.6")
b2 = np.matrix("7.5; -8.68; -0.24")
```

```
x2 = np.matrix("0.8; -2; 1")
x_c2 = solution(mA2, b2)
print("Решение системы A2:", x_c2)
print("Разность с ожидаемым решением:", x2 - x_c2)
print("-----------------------------------------")

#решение СЛАУ по заданию
mA3 = np.matrix("5.8 0.3 -0.2; 0.3 4 -0.7; -0.2 -0.7 6.7")
b3 = np.matrix("3.1; -1.7; 1.1")
x_c3 = solution(mA3, b3)
print("Решение системы A3:", x_c3)
print("-----------------------------------------")

mA4 = np.matrix("4.12 0.42 1.34 0.88; 0.42 3.95 1.87 0.43; 1.34
1.87 3.2 0.31; 0.88 0.43 0.31 5.17")
b4 = np.matrix("11.17; 0.115; 9.909; 9.349")
x_c4 = solution(mA4, b4)
print("Решение системы A4:", x_c4)
```