Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчет

по лабораторной работе «Решение системы линейных алгебраических уравнений СЛАУ»

по дисциплине «**Вычислительная математика**»

Автор: Билошицкий Михаил Владимирович

Факультет: ПИиКТ

Группа: P3126

Преподаватель: Малышева Татьяна Алексеевна

Санкт-Петербург, 2023

Содержание

Цель работы

Разобраться в работе и научиться программировать основные методы решения систем линейных алгебраических уравнений.

Описание метода, расчётные формулы

Метод простой итерации — это численный и приближенный метод решения СЛАУ.

Суть: нахождение по приближённому значению величины следующего приближения, которое является более точным. Метод позволяет получить значения корней системы с заданной точностью в виде предела последовательности некоторых векторов.

Для матриц вида: $Ax = B$; $detA \neq 0$

Достаточное условие сходимости: $|a_{ii}| \geq \sum_{j \neq i}^{n} |a_{ij}|, i = 1, 2, \dots, n$

Листинг программы

```
from numpy import float64, linalg
from colorama import init as init_colorama, Fore
init_colorama()

# Проверка на нули в диагональных элементах
def is_zero(matrix):
    n = len(matrix)
    for i in range(n):
        if matrix[i][i] == 0:
            return True
    return False

# Метод диагонального преобладания
def diagonal_pivot(A, b):
    n = len(A)
    for i in range(n):
        max_row = i
        for j in range(i + 1, n):
            if abs(A[j][i]) > abs(A[max_row][i]):
                max_row = j
        A[i], A[max_row] = A[max_row], A[i]
```

```
        b[i], b[max_row] = b[max_row], b[i]

# Проверка на диагональное преобладание
def is_diagonal_pivot(A):
    n = len(A)
    for i in range(n):
        row_sum = sum(abs(A[i][j]) for j in range(n) if j != i)
        if abs(A[i][i]) <= row_sum:
            return False
    return True

# Определитель матрицы
def determinant(matrix):
    n = len(matrix)
    if n == 1:
        return matrix[0][0]
    elif n == 2:
        return matrix[0][0] * matrix[1][1] - matrix[0][1] * matrix[1][0]
    else:
        det = 0
        for j in range(n):
            minor = [row[:j] + row[j + 1:] for row in matrix[1:]]
            det += (-1) ** j * matrix[0][j] * determinant(minor)
        return det

# Векторное произведение
def dot_product(v1, v2):
    return sum(x * y for x, y in zip(v1, v2))

def zero_vector(n):
    return [float64(0)] * n

def simple_iteration(A, b, tol=0.01, max_iter=1000):
    n = len(b)
    x = zero_vector(n)
    for itr in range(max_iter):
        x_new = zero_vector(n)
        # Вычисление следующего приближения
        for i in range(n):
            x_new[i] = (b[i] - dot_product(A[i][:i], x[:i]) - \
```

```
                    dot_product(A[i][i + 1:], x[i + 1:])) / A[i][i]
        # Критерий окончания итерационного процесса
        if max(abs(x_new[i] - x[i]) for i in range(n)) < tol:
            for j in range(n):
                print(f"Вектор погрешностей для x{j}: {abs(x_new[j] - x[j])}")
            return x_new
        x = x_new
        print("Итерация ", itr + 1, ": ", x)
    raise ValueError("Решение не сошлось после максимального числа итераций")

# Вычисление невязки
def residual(A, b, x):
    res = zero_vector(len(b))
    for i in range(len(b)):
        res[i] = b[i] - dot_product(A[i], x)
    return res

try:
    n = int(input('Введите размерность матрицы -> '))
    A = [list(map(float64, input(f'Введите ряд {i + 1} -> ').split())) for i in
range(n)]
    if determinant(A) == 0:
        print('Матрица вырождена, метод непременим.')
        exit()
    b = list(map(float64, input('Вектор b -> ').split()))
    if not is_diagonal_pivot(A):
        print('Матрица не соответствует диагональному преобладанию. \
              Попытка преобразования:')
        diagonal_pivot(A, b)
        if is_zero(A):
            print('После преобразования остались нули на \
                  диагноальных элементах. Метод непременим.')
            exit()
        for row in A:
            print(row)
    else:
        print('Диагональное преобладание согласовано.')
    tol = float64(input('Погрешность -> '))
    max_iter = int(input('Максимальное число итераций -> '))
    print()
```

```
except Exception:
    print('Неверные входные данные')
    exit()

try:
    solution = simple_iteration(A, b, tol, max_iter)
except ValueError:
    print(Fore.RED + '\nРешение не сошлось после максимального числа итераций')
    exit()

print(Fore.GREEN + "\nРешение:", solution)
print("Определитель:", determinant(A))
print(Fore.RED + "Невязка: ", residual(A, b, solution))
print(Fore.BLUE + "Решение при помощи numpy:", linalg.solve(A, b))
print("Определитель при помощи numpy:", linalg.det(A), end="\n\n")
```

Примеры и результаты работы программы

Пример 1.

Введите размерность матрицы -> 3
Введите ряд 1 -> 0 5 2
Введите ряд 2 -> 8 4 1
Введите ряд 3 -> 0 5 10
Вектор b -> 1 2 3
Матрица не соответствует диагональному преобладанию. Попытка преобразования:
[8.0, 4.0, 1.0]
[0.0, 5.0, 2.0]
[0.0, 5.0, 10.0]
Погрешность -> 0.01
Максимальное число итераций -> 100

Итерация 1 : [0.25, 0.2, 0.3]
Итерация 2 : [0.11249999999999999, 0.08, 0.2]
Итерация 3 : [0.185, 0.12, 0.26]
Итерация 4 : [0.1575, 0.096, 0.24]
Итерация 5 : [0.172, 0.10400000000000001, 0.252]
Вектор погрешностей для x0: 0.005500000000000005
Вектор погрешностей для x1: 0.004800000000000013
Вектор погрешностей для x2: 0.0040000000000000036

Решение: [0.16649999999999998, 0.0992, 0.248]
Определитель: 320.0

Невязка: [0.02320000000000011, 0.008000000000000007, 0.02400000000000002]
Решение при помощи numpy: [0.16875 0.1 0.25]
Определитель при помощи numpy: 319.99999999999994

Пример 2.

Введите размерность матрицы -> 7
Введите ряд 1 -> 91 5 7 9 11 13 15
Введите ряд 2 -> 1 80 5 7 9 11 13
Введите ряд 3 -> 2 3 70 7 9 11 13
Введите ряд 4 -> 3 5 7 120 11 13 15
Введите ряд 5 -> 4 5 7 9 210 13 15
Введите ряд 6 -> 5 7 9 11 13 100 10
Введите ряд 7 -> 6 7 9 11 13 15 90
Вектор b -> 1 1 1 1 1 1 1
Диагональное преобладание согласовано.
Погрешность -> 0.0000001
Максимальное число итераций -> 100

Итерация 1 : [0.01098901098901099, 0.0125, 0.014285714285714285, 0.008333333333333333, 0.004761904761904762, 0.01, 0.011111111111111112]
Итерация 2 : [0.004543432757718471, 0.007024343711843713, 0.008355529391243677, 0.003795711233211233 5, 0.0020089394732251873, 0.004643009768009768, 0.004604700854700854]
Итерация 3 : [0.007919782424099535, 0.009976178347941742, 0.011632220727118687, 0.006176927528490028, 0.0034505943036555286, 0.00738996620443049, 0.007898389441246585]
Итерация 4 : [0.006160431602170815, 0.008445707269037952, 0.00994241953843864, 0.004936935635680466, 0.0026994127782921764, 0.00594090029726464, 0.006158940540537477]
Итерация 5 : [0.007081678611328562, 0.009248225949429312, 0.010829608860532192, 0.005586532329545924, 0.0030927836561115704, 0.006696080517442744, 0.00706581690098343]
Итерация 6 : [0.006600174690159125, 0.008828962405869126, 0.010366266861608207, 0.0052470804328034, 0.0028871895360780635, 0.006300713333882723, 0.0065911846225015225]
Итерация 7 : [0.0068519941590034, 0.009048262192278046, 0.010608646396893118, 0.005424622235783855, 0.0029947135933596743, 0.006507367929987792, 0.006839308494726605]
Итерация 8 : [0.006720322023083588, 0.008933599227556193, 0.010481919849330334, 0.005331791014547091, 0.002938491319007378, 0.006399291700324344, 0.006709552687454224]
Итерация 9 : [0.006789175372097493, 0.008993559076461 2, 0.010548188652254223, 0.005380334266021208, 0.0029678908346553274, 0.006455803014660596, 0.006777401453473923]
Итерация 10 : [0.006753171598723279, 0.008962205899201969, 0.010513536608513535, 0.005354950814680973 5, 0.0029525176817935097, 0.006426252394225045, 0.00674192259294708]

Итерация 11 : [0.006771998268059409, 0.008978600755937665, 0.010531656488875022, 0.005368224041078044, 0.0029605564309440084, 0.006441704564810708, 0.006760474721093809]
Итерация 12 : [0.006762153665155536, 0.008970027769179822, 0.010522181477755549, 0.005361283375788982, 0.002956352910463334, 0.006433624497031954, 0.006750773668599039]
Итерация 13 : [0.006763014826616865, 0.00897451064952793 6, 0.010527136033975964, 0.005364912702573732, 0.002958550963222324 3, 0.006437849623344712, 0.006755846421109184]
Итерация 14 : [0.006764609650283521, 0.008972166517865377, 0.010524545259540921, 0.005363014900424825, 0.002957401584955234 6, 0.006435640272729193, 0.006753193840568652]
Итерация 15 : [0.006766017229687324, 0.008973392281962842, 0.010525899995012177, 0.005364007275454326 5, 0.002958002603499618 3, 0.006436795558728788, 0.006754580894859377]
Итерация 16 : [0.006765281195836823, 0.00897275132070483, 0.010525191593655307, 0.005363488355093149, 0.002957688326385225 7, 0.006436191450986275, 0.006753855593740024]
Итерация 17 : [0.006765666073471328, 0.008973086484166228, 0.010525562022051764, 0.005363759702451823, 0.002957852664251338 4, 0.006436507343465515, 0.006754234859148008]
Итерация 18 : [0.006765464818062676, 0.008972911224672503, 0.010525368322258416, 0.005363617812877966 5, 0.002957766730745633, 0.006436342160912964, 0.006754036538414694]
Итерация 19 : [0.006765570056028966 5, 0.00897300286917944, 0.010525469609336672, 0.005363692008002913, 0.002957811666021843 4, 0.006436428536111555, 0.006754140241817171]
Вектор погрешностей для x0: 5.5029723781269135e-08
Вектор погрешностей для x1: 4.792160168513859e-08
Вектор погрешностей для x2: 5.2963774723607804e-08
Вектор погрешностей для x3: 3.8797188652711645e-08
Вектор погрешностей для x4: 2.349699376649056e-08
Вектор погрешностей для x5: 4.516624074139136e-08
Вектор погрешностей для x6: 5.422728882945038e-08

Решение: [0.006765515026305185, 0.008972954947577754, 0.010525416645561949, 0.005363653210814260 5, 0.002957788169028077, 0.006436383369870813, 0.006754086014528341 5]
Определитель: 101034768791057.0
Невязка: [2.6185665229494504e-06, 2.004685264789252e-06, 1.938660920020041e-06, 2.4344809961274194e-06, 2.5802184866474676e-06, 2.361776685777528e-06, 2.552027132285417e-06]
Решение при помощи numpy: [0.00676553 0.00897297 0.01052543 0.00536367 0.0029578 0.0064364
0.0067541]
Определитель при помощи numpy: 101034768791057.22

Вывод

По итогам выполненной лабораторной работы я научился эффективно программировать метод простых итераций на языке программирования Python, а также хорошо разобрался в других численных методах решения СЛАУ.