

# Лабораторная работа №6-72 «Сравнение скорости сходимости метода Якоби и Зейделя»

## Цель работы

Реализовать метод Зейделя и Якоби.

## Задание

1. Реализовать метод Якоби и Зейделя.
2. Ввести критерий остановки итерационного процесса используя равномерную норму.
3. Сравнить скорости сходимости методов в зависимости от размера матрицы

```
import random
from copy import deepcopy

import numpy
import numpy as np
from matplotlib import pyplot as plt

def increase_diag(a, pow):
    for i in range(0, len(a)):
        a[i][i] = abs(a[i][i]) * (10 ** pow)
    return a

def norma_vec(vec):
    max = -1
    for x in vec:
        if max <= abs(x):
            max = x
    return max

def jacobi(a, b):
    x = np.zeros_like(b, dtype=np.double)
    d = np.diagonal(a)
    t = a - np.diag(d)
    iter = 0
    while True:

        x_old = x.copy()

        x[:] = (b - np.dot(t, x)) / d
        if abs(norma_vec(x) - norma_vec(x_old)) < 10 ** (-6):
            break
        iter += 1
```

```

    iter += 1

    return x, iter

def seidel(a, b):
    x = np.zeros_like(b, dtype=np.double)
    iter = 0
    while True:

        x_old = x.copy()

        for i in range(a.shape[0]):
            x[i] = (b[i]
                    - np.dot(a[i, :i], x[:i])
                    - np.dot(a[i, (i + 1):], x_old[(i + 1):])) \
                    / a[i, i]

        if abs(norma_vec(x) - norma_vec(x_old)) < 10**(-6):
            break
        iter += 1

    return x, iter

def fill_matrix(n, a, b):
    m = [[random.uniform(-a, b) for i in range(n)] for j in range(n)]
    return m

def compare_jacobi_gauss(n, p):
    # matrix = fill_matrix(3, 0, 10)
    # b = [random.uniform(0, 10) for i in range(3)]
    # x = [1.0, 1.0, 1.0]
    a = numpy.array(increase_diag(fill_matrix(n, 0, 10), p))
    b = numpy.array([random.uniform(0, 10) for i in range(n)])

    x_jacobi, iter_jacobi = jacobi(deepcopy(a), deepcopy(b))
    x_seidel, iter_seidel = seidel(deepcopy(a), deepcopy(b))
    # print("-----")
    # for el in x_jacobi:
    #     print("%.30f " % el)

    # print(x_jacobi)
    # print("-----")
    # print("-----")
    # for el in x_seidel:
    #     print("%.30f " % el)

    # print(x_seidel)
    return iter_jacobi, iter_seidel

def compare_speed():
    for x in range(3, 7):
        x_i = 1

```

```

x_j = []
x_z = []
y = []
for i in range(3, 50):
    j, z = compare_jacobi_gauss(i, x)
    x_j.append(j)
    x_z.append(z)
    y.append(i)
plt.plot(y, x_j, color="blue")
plt.plot(y, x_z, color="red")
plt.show()

```

```

def is_diag(a):
    sum = []
    flag = True
    for i in range(0, len(a)):
        s = 0.0
        for j in range(0, len(a)):
            if i != j:
                s += abs(a[i][j])
        sum.append(s)

    for i in range(0, len(a)):
        if abs(a[i][i]) < sum[i]:
            flag = False
            break

    return flag

if __name__ == '__main__':
    compare_speed()

```





