БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет прикладной математики и информатики

Далецкий Денис Андреевич

МЕТОД РИЧАРДСОНА

Отчет по лабораторной работе №2 Студента 2 курса 10 группы

Преподаватель:

Никифоров Иван Васильевич доцент кафедры ВМ, канд. физ.-мат. наук

Постановка задачи

Построить программу решения системы линейных алгебраических уравнений методом Ричардсона.

- 1. Для заданной матрицы A и случайного вектора x вычислить f = Ax.
- 2. Решить СЛАУ Ax' = f методом Ричардсона.
- 3. Сравнить полученное решение x' с правильным решением x
- 4. Исследовать сходимость метода Ричардсона от параметра т. Показать зависимость количества итераций от выбранного параметра т.

Краткая теория

Имеем систему из п уравнений:

$$\begin{cases} a_{11}^{0} * x_{1} + a_{12}^{0} * x_{2} + \dots + a_{1n}^{0} * x_{n} = f_{1}^{0} \\ a_{21}^{0} * x_{1} + a_{22}^{0} * x_{2} + \dots + a_{2n}^{0} * x_{n} = f_{2}^{0} \end{cases}$$

$$\vdots$$

$$a_{n1}^{0} * x_{1} + a_{n2}^{0} * x_{2} + \dots + a_{nn}^{0} * x_{n} = f_{n}^{0}$$

Верхний индексk показывает значение на k-м шаге вычислений.

$$A = [a_{ij}], x = [x_i], f = [f_j]$$

Метод Ричардсона заключается в последовательном приближении решения xчленами рекуррентной последовательности $x^{[k+1]} = x^{[k]} + \tau \left(f - A x^{[k]} \right)$ (1)

<u>Теорема:</u> Пусть матрица A положительно определена, $\lambda_1 \ge \lambda_2 \ge ... \ge \lambda_n > 0$ - её собственные значения. Стационарный метод Ричардсона (1) сходится тогда и только тогда, когда $0 < \tau < \frac{2}{\lambda_1}$

Оптимальным выбором τ с точки зрения скорости сходимости будет $\tau = \frac{2}{\lambda_1 + \lambda_n}$

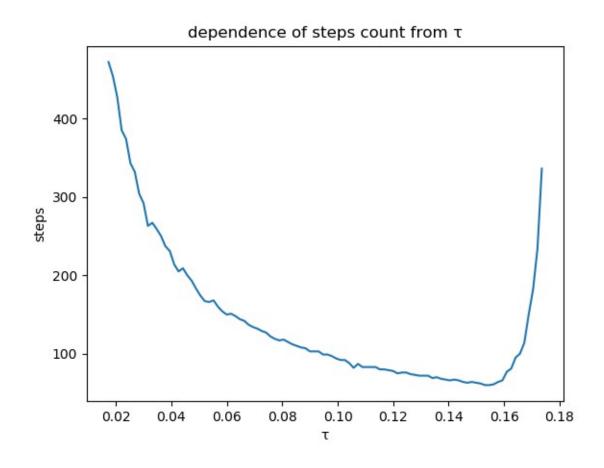
Код программы (написанной на языке Python)

```
import numpy as np
from matplotlib import pyplot as plt
def richardson_step(x, A, b, Ir):
 x_next = x + lr * (b - A @ x)
 return x_next
def richardson_generator(A, b, eps, Ir):
 n = A.shape[0]
 x_{current} = np.random.normal(size=n)
 while True:
    x_next = richardson_step(x_current, A, b, lr)
   yield x next
   if np.max(np.abs(x current - x next)) < eps:
      raise StopIteration
    x_{current} = x_{next}
def continuous_norm(mat):
 row_sums = np.sum(np.abs(mat), 1)
 return np.max(row_sums)
if __name__ == "__main__":
 np.set_printoptions(floatmode="fixed", precision=16)
 A = np.array([
   [8, 0, -4, 0, -2],
   [0, 7, 0, -4, 0],
   [-4, 0, 6, 0, -1],
   [0, -4, 0, 5, 0],
   [-2, 0, -1, 0, 4]
    ], dtype=float)
```

```
x = np.round(np.random.rand(5) * 21 - 10, 2)
b = A @ x
print(A)
print("b = ", b)
print("x = ", x)
eigenvals, _ = np.linalg.eigh(A)
min_{eig} = eigenvals[0]
max_eig = eigenvals[-1]
max_tau = 2.0 / (0.2 * min_eig + max_eig)
min_tau = max_tau / 10
opt tau = 2.0 / (min eig + max eig)
steps counts = []
taus = np.linspace(min_tau, max_tau, 100)
for tau in taus:
  xs_ = np.array(list(richardson_generator(A, b, 1e-6, tau)))
  x_{-} = xs_{-}[-1]
  steps = len(xs)
  steps_counts.append(steps)
fig, ax = plt.subplots(1, 1)
ax.plot(taus, steps_counts)
plt.title("dependence of steps count from \tau")
plt.xlabel("\tau")
plt.ylabel('steps')
plt.savefig("out.png")
print("x' =", x_)
print("|x - x'| = ", max(abs(x - x_)))
plt.show()
```

Результаты

График зависимости кол-ва итераций от параметра τ :



Исходная матрица:
$$\begin{vmatrix} 8 & 0 & -4 & 0 & -2 \\ 0 & 7 & 0 & -4 & 0 \\ -4 & 0 & 6 & 0 & -1 \\ 0 & -4 & 0 & 5 & 0 \\ -2 & 0 & -1 & 0 & 4 \end{vmatrix}$$

-2.720000000000000 -1.219999999999998]

-7.3600000000000003 3.3700000000000001]

x' = [5.6999987658244500 -8.5199999994861120 3.29999994904101717

-7.359999993418539 3.3699991940078888]

|x - x'| = 1.2341755502021101e-06