Белорусский государственный университет  
Факультет прикладной математики и информатики

Далецкий Денис Андреевич

**МЕТОД РИЧАРДСОНА**

Отчет по лабораторной работе №2Студента 2 курса 10 группы

**Преподаватель**: Никифоров Иван Васильевич

доцент кафедры ВМ,

канд. физ.-мат. наук

Минск, 2018

**Постановка задачи**

Построить программу решения системы линейных алгебраических уравнений методом Ричардсона.

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

**Краткая теория**

Имеем систему из n уравнений:

Верхний индекс показывает значение на -м шаге вычислений.

Метод Ричардсона заключается в последовательном приближении решения членами рекуррентной последовательности (1)

Теорема: Пусть матрица положительно определена, - её собственные значения. Стационарный метод Ричардсона (1) сходится тогда и только тогда, когда

Оптимальным выбором с точки зрения скорости сходимости будет

**Код программы (написанной на языке Python)**

import numpy as np

from matplotlib import pyplot as plt

def richardson\_step(x, A, b, lr):

  x\_next = x + lr \* (b - A @ x)

  return x\_next

def richardson\_generator(A, b, eps, lr):

  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 τ")

  plt.xlabel("τ")

  plt.ylabel('steps')

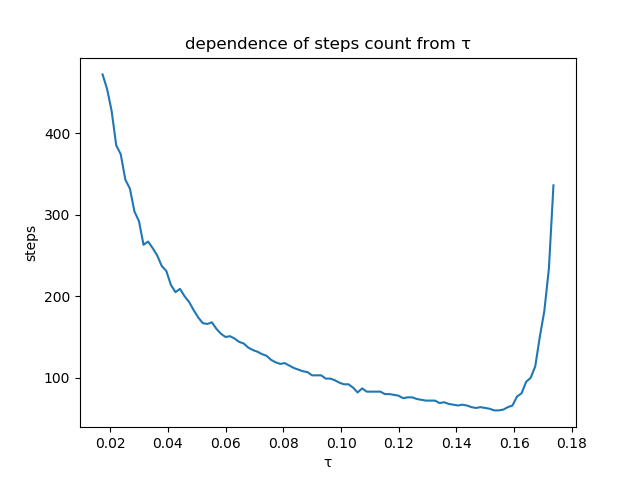
  plt.savefig("out.png")

  print("x' =", x\_)

  print("|x - x'| = ", max(abs(x - x\_)))

  plt.show()

**Результаты**

График зависимости кол-ва итераций от параметра :

**Исходная матрица:**

f = [ 25.6600000000000001 -30.1999999999999993 -6.3700000000000045

-2.7200000000000060 -1.2199999999999998]

x = [ 5.7000000000000002 -8.5199999999999996 3.2999999999999998

-7.3600000000000003 3.3700000000000001]

x' = [ 5.6999987658244500 -8.5199999994861120 3.2999994904101717

-7.3599999993418539 3.3699991940078888]