Итерационные методы решения СЛАУ

Вариант 7

Постановка задачи

Даны две матрицы и стационарный итерационный метод. Для каждой матрицы необходимо выполнить следующие пункты.

1. Сгенерировать вектор b таким образом, чтобы точным решением СЛАУ Ax = b был вектор $x = (1, 2, ..., n)^T$.

2. Реализовать решение полученных СЛАУ, построить диаграмму сходимости.

3. Вычислить матрицу B, соответствующую вашему методу.

4. Вычислить спектральный радиус матрицы B степенным методом.

5. Сделать выводы.

Метод релаксации, ω=1.2

A1 =

5.863818863708440 -0.277544116671075 -1.995537762171580 -0.137083711556688 0.206342778890080
-0.277544116671075 6.079151014088120 -0.744896794397772 -0.536319403140064 -2.700694409782450
-1.995537762171580 -0.744896794397772 5.680655755691160 0.247609252787904 -3.082572310388470
-0.137083711556688 -0.536319403140064 0.247609252787904 7.674360409136210 -0.412397742515552
0.206342778890080 -2.700694409782450 -3.082572310388470 -0.412397742515552 5.702013957376080

A2 =

537.537709718568000 -136.375861396643000 -18.367452727628500 228.051289483657000 113.636226542262000
-136.375861396642000 373.310283520029000 277.678528402533000 92.044049805949400 35.929463020161900
-18.367452727628500 277.678528402533000 518.767439631917000 4.644690779493230 -25.043206116413600
228.051289483657000 92.044049805949400 4.644690779493230 205.055439410727000 119.468823384937000
113.636226542262000 35.929463020161900 -25.043206116413600 119.468823384937000 519.329127718759000

Решение

1.

Векторы b для решения $x = (1, 2, 3, 4, 5)^T$

1 матрица	2 матрица
-0,1945036079248017 -6,002682133160657 -0,8657886246843756 28,16855816449398 12,417715844977963	1690,0699193883343 1991,103805175622 1986,6546555090931 1843,6393360016282 3184,886466366888

2.

```
0,33669714532173645 -0,28743928102006966 0,512110868028708 4,164582786956093 2,8290078053687133
0,08303500169543622 0,9014870785133999 1,5158820326434796 3,7727711143963596 3,86714196655426
0,5563835463325484 1,349184619946093 2,2816105519126415 3,936120852563748 4,4043468686119365
0,7817455983158819 1,6882649375473444 2,6180895271123585 3,9583311423890204 4,700054125823088
0,881478587261159 1,8353833260918373 2,8073799876659353 3,9801041394656718 4,844887614149758
0,9416847008320399 1,916607990229981 2,900855198685152 3,989572083583356 4,920934034859783
0,9694843429359801 1,9571738186922973 2,9492862443964323 3,9947049851229903 4,9594381193922095
0,9845246521429956 1,978076109973935 2,9740335723655082 3,9972784571531763 4,979242103071281
0,992045955286406 1,9887766262878874 2,9866996408723603 3,9986089966607365 4,989368884945352
0,9959316886418679 1,9942513324328601 2,9931905900235622 3,9992870046685995 4,994556168443822
0,9979162218180857 1,997056694862711 2,9965127565896625 3,9996350773490863 4,99721241626201
0,9989329478420376 1,9984927267672985 2,9982143530820577 3,999813089638046 4,998572539556041
0,9994536181589548 1,9992281838928663 2,9990856103541734 3,999904298387984 4,999269043472604
0,9997202049464898 1,999604773671314 2,999531768416543 3,999950991960459 4,999625697237596
0,9998567270556322 1,9997976158150372 2,9997602321920596 3,9999749047575914 4,9998083304452035
0,9999266336799605 1,999896365011984 2,9998772217062464 3,9999871493900754 4,999901851532955
0,9999624312577859 1,9999469314585219 2,9999371287985652 3,999993419579709 4,999949740986944
0,9999807621303984 1,999972825137587 2,9999678054583776 3,999996630355463 4,99997426380746
0,9999901488371394 1,9999860845322677 2,9999835141027122 3,999998274501908 4,999986821235328
0,9999949555039792 1,9999928742895763 2,999991558045953 3,9999991164220825 4,99999325153382
0,999997416858377 1,9999963511286096 2,9999956771181546 3,9999995475451566 4,9999965443045395 iteration:21
```

На 21 итерации и на 1395 итерации для второй матрицы

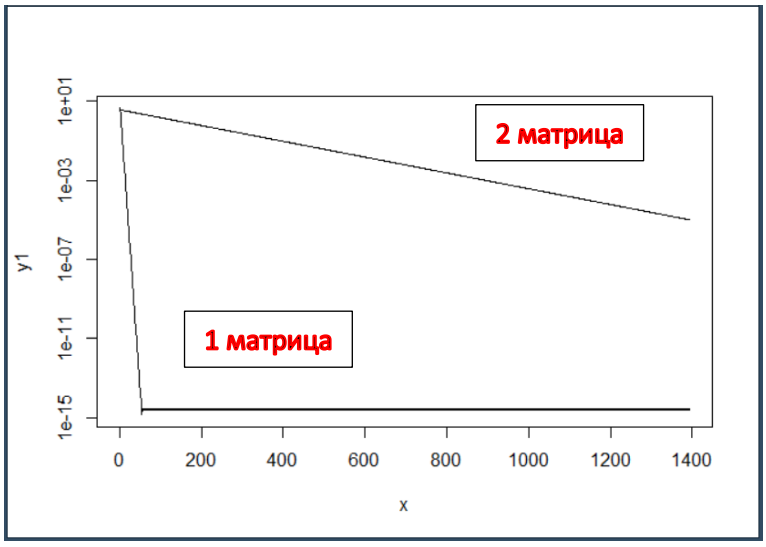
Метод сошелся с точностью 0.00001

Код на C#

```
using System;
using System.Globalization;
using System.IO;
using System.Linq;

namespace Relaxasion
{
    public static class Program
    {
        private const double Omega = 1.2;
        private const string APath = "A1.txt";
        private const string BPath = "b1.txt";
        private const string ResidualPath = "residual1.txt";

        private static double[][] A;
        private static double[] b;
        private static double[] x = new []{1.5, 1.5, 1.5, 1.5, 1.5};
        private static double[] realX = new[] {1.0, 2.0, 3.0, 4.0, 5.0};

        public static void Main()
        {
            var nfi = new NumberFormatInfo { NumberDecimalSeparator = "." };
            var rowsA = File.ReadAllLines(APath);
            var strB = File.ReadAllText(BPath);
            using var writerResidual = new StreamWriter(ResidualPath);

            // чтение из файла данные
            A = (from i in rowsA select i.Split(' ').Select(double.Parse).ToArray()).ToArray();
            b = strB.Split().Select(double.Parse).ToArray();
```

```
            var iteration = 1;
            while (iteration < 1395)
            {
                for (var i = 0; i < 5; ++i) {
                    var nextX_i = b[i];
                    for (var j = 0; j < 5; ++j) {
                        if (j == i)
                            continue;
                        nextX_i -= A[i][j] * x[j];
                    }

                    x[i] = (1 - Omega) * x[i] + Omega * nextX_i / A[i][i];
                }

                foreach (var item in x)
                    Console.Write($"{item} ");

                var residual = Residual();
                Console.WriteLine();

                writerResidual.WriteLine($"{residual.ToString(nfi)} ");
                iteration++;
            }

            Console.WriteLine($"iteration:{iteration}");
        }

        private static double Residual() =>
            Math.Sqrt(x.Select((t, i) => (t - realX[i]) * (t - realX[i])).Sum());
    }
}
```

Диаграмма сходимости



Код на R

```
x <- 1:1394
y1 <- scan('residual1.txt')
y2 <- scan('residual2.txt')

plot(x,y1,log="y", type = 'l')
lines(x,y2,log="y", type = 'l')
```

3.

Матрица B первой матрицы

$$B_\omega = (D + \omega L)^{-1}((1 - \omega)D - \omega R),$$

-0.20000000 0.05679796 0.40837641 0.02805347 -0.04222698
-0.01095722 -0.19688826 0.16941298 0.10740424 0.53079278
-0.08603293 -0.00703841 -0.00119357 -0.02357948 0.71689455
-0.00187493 -0.01502137 0.02300701 -0.18947866 0.08033616
-0.05351783 -0.12024089 0.07977728 0.02808510 0.57556430

Матрица B второй матрицы

-0.20000000 0.30444568 0.04100353 -0.50910205 -0.25368168
-0.08767561 -0.06653770 -0.87461819 -0.51905330 -0.22670317
0.04781829 0.05567346 0.36352609 0.30102358 0.19276689
0.31284130 -0.37197794 0.40650875 0.75083935 -0.24371001
-0.02379965 0.03149128 -0.02933674 -0.01308201 -0.03613628

Код на R

```r
w = 1.2
N = 5

A <- matrix(c(
  5.863818863708440,-0.277544116671075,-1.995537762171580,-0.137083711556688,0.206342778890080,
  -0.277544116671075,6.079151014088120,-0.744896794397772,-0.536319403140064,-2.700694409782450,
  -1.995537762171580,-0.744896794397772,5.680655755691160,0.247609252787904,-3.082572310388470,
  -0.137083711556688,-0.536319403140064,0.247609252787904,7.674360409136210,-0.412397742515552,
  0.206342778890080,-2.700694409782450,-3.082572310388470,-0.412397742515552,5.702013957376080),
  N,N,byrow = TRUE)

L = matrix(rep(0, N ^ 2), N, N, byrow = TRUE)
D = matrix(rep(0, N ^ 2), N, N, byrow = TRUE)
R = matrix(rep(0, N ^ 2), N, N, byrow = TRUE)

for(i in 1:N){
  D[i,i] = A[i,i]
  for (j in 1:N) {
    if (j < i)
      L[i,j] <- A[i,j]
    if (j > i)
      R[i,j] <- A[i,j]
  }
}

B <- solve(D + w*L) %*% ((1-w)*D - w*R)
```

4.

Спектр первой и второй матриц B

```
> epsilon <- 0.00001

> maxEigen1 <- PowerIteration(B1)

> maxEigen1
[1] 0.5120711

> maxEigen2 <- PowerIteration(B2)

> maxEigen2
[1] 0.9908545

> realEigen1 <- eigen(B1)

> realEigen2 <- eigen(B2)

> realEigen1$val
[1]  0.51207124+0.0000000i -0.19446018+0.0977232i -0.19446018-0.0977232i -0.20083965+0.0000000i  0.06569258+0.0000000i

> realEigen2$val
[1]  0.99085443+0.0000000i -0.04865789+0.2291164i -0.04865789-0.2291164i -0.12787991+0.0000000i  0.04603272+0.0000000i
>
```

Код на R

```r
norma <- function(x) sqrt(sum(x^2))

normalize <- function(x) x / norma(x)

PowerIteration <- function(A){
  u <- normalize(y)
  Au <- A %*% u
  lamda <- sum(Au * u)
  while(norma(Au - lamda*u) > epsilon){
    y <- Au
    u <- normalize(y)
    Au <- A %*% u
    lamda <- sum(Au * u)
  }

  lamda
}

B1 <- matrix(c(
  -0.2, 0.05679796, 0.40837641, 0.02805347, -0.04222698,
   -0.01095722, -0.19688826, 0.16941298, 0.10740424, 0.53079278,
   -0.08603293, -0.00703841, -0.00119357, -0.02357948, 0.71689455,
   -0.00187493, -0.01502137, 0.02300701, -0.18947866, 0.08033616,
   -0.05351783, -0.12024089, 0.07977728, 0.0280851, 0.5755643
  ),N,N,byrow = TRUE)

B2 <- matrix(c(
  -0.2, 0.30444568, 0.04100353, -0.50910205, -0.25368168,
   -0.08767561, -0.0665377, -0.87461819, -0.5190533, -0.22670317,
    0.04781829, 0.05567346, 0.36352609, 0.30102358, 0.19276689,
    0.3128413, -0.37197794, 0.40650875, 0.75083935, -0.24371001,
   -0.02379965, 0.03149128, -0.02933674, -0.01308201, -0.03613628
),N,N,byrow = TRUE)
```

```r
y <- c(2, 1, 3, 2, 1)
epsilon <- 0.00001

maxEigen1 <- PowerIteration(B1)
maxEigen1

maxEigen2 <- PowerIteration(B2)
maxEigen2

realEigen1 <- eigen(B1)
realEigen2 <- eigen(B2)
realEigen1$val
realEigen2$val
```

5.

Выводы

Как видно из 4 пункта спектральный радиус матриц B меньше 1 значит итерационный процесс решения должен сходится но у 2 матрицы радиус ближе к 1 значит он будет медленее сходится что мы и видим на графике в 2 пункте.