# Лабораторная работа №8

## Научное программирование

Николаев Дмитрий Иванович, НПМмд-02-24

6 октября 2024

Российский университет дружбы народов имени Патриса Лумумбы, Москва, Россия

# Прагматика выполнения

# Прагматика выполнения

- Повышение навыков владения Octave;
- Повышение навыков владения Julia;
- Применение полученных знаний на практике в дальнейшем.

# Цели

# Цели

Целью работы является изучение собственных значений и собственных векторов матриц, а также моделирование цепей Маркова на языках программирования Octave и Julia.

# Задачи

# Задачи

1. Нахождение собственных значений и векторов матриц;
2. Анализ случайного блуждания с использованием цепей Маркова;
3. Вычисление равновесных состояний цепей Маркова.

# Выполнение работы

# Octave. Собственные значения и векторы (1/2)

```
>> A = [1 2 -3; 2 4 0; 1 1 1]
A =

   1   2  -3
   2   4   0
   1   1   1

>> diary on
>> [v lambda] = eig(A)
v =

  -0.2400 +      0i  -0.7920 +      0i  -0.7920 -      0i
  -0.9139 +      0i   0.4523 + 0.1226i   0.4523 - 0.1226i
  -0.3273 +      0i   0.2322 + 0.3152i   0.2322 - 0.3152i

lambda =

Diagonal Matrix

   4.5251 +      0i                  0                  0
                  0   0.7374 + 0.8844i                  0
                  0                  0   0.7374 - 0.8844i
```

Рис. 1: Собственные значения и векторы матрицы A на Octave

# Octave. Собственные значения и векторы (2/2)

```
>> C = A' * A
C =

    6   11   -2
   11   21   -5
   -2   -5   10

>> [v lambda] = eig(C)
v =

   0.876137   0.188733  -0.443581
  -0.477715   0.216620  -0.851390
  -0.064597   0.957839   0.279949

lambda =

Diagonal Matrix

    0.1497         0         0
         0    8.4751         0
         0         0   28.3752
```

```
>> T = [1 0.5 0 0 0; 0 0 0.5 0 0; 0 0.5 0 0.5 0; 0 0 0.5 0 0; 0 0 0 0.5 1]
T =

   1.0000   0.5000        0        0        0
        0        0   0.5000        0        0
        0   0.5000        0   0.5000        0
        0        0   0.5000        0        0
        0        0        0   0.5000   1.0000

>> a = [0.2; 0.2; 0.2; 0.2; 0.2];
>> b = [0.5; 0; 0; 0; 0.5];
>> c = [0; 1; 0; 0; 0];
>> d = [0; 0; 1; 0; 0];
```

**Рис. 3:** Матрицы переходных вероятностей и начальные состояния на Octave

# Octave. Цепи Маркова: Случайное блуждание (2/2)

```
>> T^5 * a
ans =

   0.450000
   0.025000
   0.050000
   0.025000
   0.450000

>> T^5 * b
ans =

   0.5000
        0
        0
        0
   0.5000

>> T^5 * c
ans =

   0.6875
        0
   0.1250
        0
   0.1875

>> T^5 * d
ans =

   0.3750
   0.1250
        0
   0.1250
   0.3750
```

## Octave. Цепи Маркова: Равновесное состояние (1/2)

```
>> T = [0.48 0.51 0.14; 0.29 0.04 0.52; 0.23 0.45 0.34]
T =

   0.480000   0.510000   0.140000
   0.290000   0.040000   0.520000
   0.230000   0.450000   0.340000

>> [v lambda] = eig(T)
v =

  -0.6484  -0.8011   0.4325
  -0.5046   0.2639  -0.8160
  -0.5700   0.5372   0.3835

lambda =

Diagonal Matrix

   1.0000        0        0
        0   0.2181        0
        0        0  -0.3581
```

```
>> x = v(:,1)/sum(v(:,1))
x =

   0.3763
   0.2929
   0.3308

>> T^10 * x
ans =

   0.3763
   0.2929
   0.3308

>> T^50 * x
ans =

   0.3763
   0.2929
   0.3308

>> T^50 * x - T^10*x
ans =

   4.4409e-16
   2.7756e-16
   3.8858e-16

>> diary off
>>
```

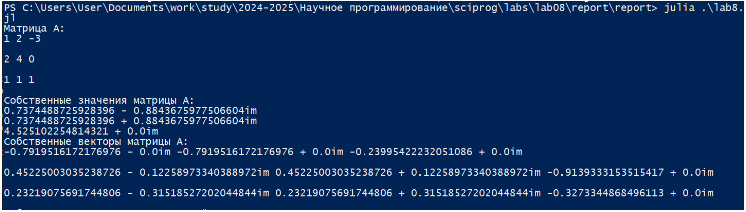
# Julia. Собственные значения и векторы (1/2)

```julia
using LinearAlgebra
function Print_Matrix(A::Matrix)
    for i in 1:size(A)[1]
        for j in 1:size(A)[2]
            print(A[i,j], " ")
        end
        println("\n")
    end
end
function Print_Vector(a::Vector)
    for i in 1:length(a)
        println(a[i])
    end
end

A = [1 2 -3; 2 4 0; 1 1 1]
println("Матрица A:")
Print_Matrix(A)
# Нахождение собственных значений и векторов
eigvals, eigvecs = eigen(A)

# Отображение результатов
println("Собственные значения матрицы A:")
Print_Vector(eigvals)
println("Собственные векторы матрицы A:")
Print_Matrix(eigvecs)
```

# Julia. Собственные значения и векторы (2/2)



```
PS C:\Users\User\Documents\work\study\2024-2025\Научное программирование\sciprog\labs\lab08\report\report> julia .\lab8.
jl
Матрица A:
1 2 -3

2 4 0

1 1 1

Собственные значения матрицы A:
0.7374488725928396 - 0.8843675977506604im
0.7374488725928396 + 0.8843675977506604im
4.525102254814321 + 0.0im
Собственные векторы матрицы A:
-0.7919516172176976 - 0.0im -0.7919516172176976 + 0.0im -0.23995422232051086 + 0.0im

0.45225003035238726 - 0.12258973340388972im 0.45225003035238726 + 0.12258973340388972im -0.9139333153515417 + 0.0im

0.23219075691744806 - 0.31518527202044844im 0.23219075691744806 + 0.31518527202044844im -0.3273344868496113 + 0.0im
```

**Рис. 8:** Собственные значения и векторы матрицы A на Julia (2/2)

# Julia. Собственные значения и векторы симметричной матрицы (1/2)

```
# Создание симметричной матрицы
C = A' * A

# Нахождение собственных значений и векторов симметричной матрицы
eigvals_sym, eigvecs_sym = eigen(C)

# Отображение результатов для симметричной матрицы
println("Собственные значения симметричной матрицы:")
Print_Vector(eigvals_sym)
println("Собственные векторы симметричной матрицы:")
Print_Matrix(eigvecs_sym)
```

**Рис. 9:** Собственные значения и векторы симметричной матрицы C = A'A на Julia (1/2)

# Julia. Собственные значения и векторы симметричной матрицы (2/2)

```
Собственные значения симметричной матрицы:
0.1496983766913473З
8.4751451450З277
28.375156478275926
Собственные векторы симметричной матрицы:
0.876137029821832B 0.18873290083336247 -0.44358065458042806

-0.477714508467550B 0.216619951644148 -0.8513898313635729

-0.06459685266778223 0.9578390724399956 0.27994920598691814
```

**Рис. 10:** Собственные значения и векторы симметричной матрицы C = A’A на Julia (2/2)

# Julia. Цепи Маркова: Случайное блуждание (1/2)

```julia
# Цепи Маркова.
println("\nЦепи Маркова. Матрица вероятностных переходов T:")
T = [1 0.5 0 0 0; 0 0 0.5 0 0; 0 0.5 0 0.5 0; 0 0 0.5 0 0; 0 0 0 0.5 1]
a = [0.2; 0.2; 0.2; 0.2; 0.2]
b = [0.5; 0; 0; 0; 0.5]
c = [0; 1; 0; 0; 0]
d = [0; 0; 1; 0; 0]
Print_Matrix(T)
# Вероятностное распределение состояний через 5 шагов
println("\nНачальное состояние: a = $(a)")
println("Вероятностное распределение состояний через 5 шагов: ", T^5 * a)
println("Начальное состояние: b = $(b)")
println("Вероятностное распределение состояний через 5 шагов: ", T^5 * b)
println("Начальное состояние: c = $(c)")
println("Вероятностное распределение состояний через 5 шагов: ", T^5 * c)
println("Начальное состояние: d = $(d)")
println("Вероятностное распределение состояний через 5 шагов: ", T^5 * d)

```

**Рис. 11:** Вектора вероятностей после 5 шагов на Julia (1/2)

# Julia. Цепи Маркова: Случайное блуждание (2/2)

```
Цепи Маркова. Матрица вероятностных переходов T:
1.0 0.5 0.0 0.0 0.0

0.0 0.0 0.5 0.0 0.0

0.0 0.5 0.0 0.5 0.0

0.0 0.0 0.5 0.0 0.0

0.0 0.0 0.0 0.5 1.0


Начальное состоение: a = [0.2, 0.2, 0.2, 0.2, 0.2]
Вероятностное распределение состояний через 5 шагов: [0.45000000000000007, 0.025, 0.05, 0.025, 0.45]
Начальное состоение: b = [0.5, 0.0, 0.0, 0.0, 0.5]
Вероятностное распределение состояний через 5 шагов: [0.5, 0.0, 0.0, 0.0, 0.5]
Начальное состоение: c = [0, 1, 0, 0, 0]
Вероятностное распределение состояний через 5 шагов: [0.6875, 0.0, 0.125, 0.0, 0.1875]
Начальное состоение: d = [0, 0, 1, 0, 0]
Вероятностное распределение состояний через 5 шагов: [0.375, 0.125, 0.0, 0.125, 0.375]
```

**Рис. 12:** Вектора вероятностей после 5 шагов на Julia (2/2)

# Julia. Цепи Маркова: Равновесное состояние (1/2)

```julia
# Равновесное состояние
println("\nРавновесное состояние. Матрица вероятностных переходов T:")
T = [0.48 0.51 0.14; 0.29 0.04 0.52; 0.23 0.45 0.34]
Print_Matrix(T)
# Нахождение собственных значений и векторов
eigvals, eigvecs = eigen(T)
println("Собственные значения матрицы T:")
Print_Vector(eigvals)
println("Собственные векторы матрицы T:")
Print_Matrix(eigvecs)
# Нормализация третьего собственного вектора
# (соответствующего равновесному состоянию)
x = eigvecs[:,3] / sum(eigvecs[:,3])

# Проверка равновесного состояния
println("Равновесное состояние (третий нормированный собственный вектор): ")
Print_Vector(x)

# Проверка через умножение
println("T^10 * x: ", T^10 * x)
println("T^50 * x: ", T^50 * x)
println("T^50 * x - T^10 * x: ", T^50 * x - T^10 * x)
```

**Рис. 13:** Равновесное состояние матрицы переходных вероятностей на Julia (1/2)

```
Равновесное состояние. Матрица вероятностных переходов T:
0.48 0.51 0.14

0.29 0.04 0.52

0.23 0.45 0.34

Собственные значения матрицы T:
-0.3580972058177587
0.218097205817759
1.0000000000000016
Собственные векторы матрицы T:
0.43249347882754974 -0.8011125885588817 -0.6483964861371586

-0.8160067273078764 0.263940170269989 -0.504632900872513

0.38351324848032664 0.5371724182888928 -0.5700242381881476

Равновесное состояние (третий нормированный собственный вектор):
0.37630662020905936
0.2928712684810245
0.3308221113099163
T^10 * x: [0.3763066202090593, 0.2928712684810246, 0.3308221113099163]
T^50 * x: [0.3763066202090595, 0.2928712684810248, 0.33082211130991646]
T^50 * x - T^10 * x: [1.6653345369377348e-16, 1.6653345369377348e-16, 1.6653345369377348e-16]
```

**Рис. 14:** Равновесное состояние матрицы переходных вероятностей на Julia (2/2)

# Результаты

По результатам работы, я изучил возможности вычисления собственных значений и собственных векторов матриц, а также моделирования цепей Маркова на языках программирования Octave и Julia.