РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра теории вероятностей и кибербезопасности

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №4

дисциплина: Компьютерный практикум по статистическому анализу данных

Студент: Быстров Глеб

Группа: НПИбд-01-20

МОСКВА

2023 г.

Цель работы

В данной лабораторной работе мне будет необходимо изучить возможности специализированных пакетов Julia для выполнения и оценки эффективности операций над объектами линейной алгебры.

Описание процесса выполнения работы

3.3.1. Поэлементные операции над многомерными массивами

1. Для матрицы 4×3 рассмотрим поэлементные операции сложения и произведения её элементов:

```
# Массив 4х3 со случайными целыми числами (от 1 до 20):
a = rand(1:20,(4,3))
```

```
4×3 Matrix{Int64}:
  9   9  18
  6   2  19
  8  12   8
 20   2  11
```

```
# Поэлементная сумма:
sum(a)
```

```
124
```

```
# Поэлементная сумма по столбцам:
sum(a,dims=1)
```

```
1×3 Matrix{Int64}:
 43  25  56
```

```
# Поэлементная сумма по строкам:
sum(a,dims=2)
```

```
4×1 Matrix{Int64}:
 36
 27
 28
 33
```

Для работы со средними значениями можно воспользоваться возможностями пакета Statistics:

```
import Pkg
Pkg.add("Statistics")
using Statistics
```

```
    Updating registry at `C:\Users\GlebB\.julia\registries\General.toml`
   Resolving package versions...
    Updating `C:\Users\GlebB\.julia\environments\v1.9\Project.toml`
  [10745b16] + Statistics v1.9.0
  No Changes to `C:\Users\GlebB\.julia\environments\v1.9\Manifest.toml`
```

```
4×1 Matrix{Float64}:
 12.0
  9.0
  9.333333333333334
 11.0
```

```
# Вычисление среднего значения массива:
mean(a)
```

```
10.333333333333334
```

```
# Среднее по столбцам:
mean(a,dims=1)
```

```
1×3 Matrix{Float64}:
 10.75  6.25  14.0
```

```
# Среднее по строкам:
mean(a,dims=2)
```

```
4×1 Matrix{Float64}:
 12.0
  9.0
```

3.3.2. Транспонирование, след, ранг, определитель и инверсия матрицы

2. Для выполнения таких операций над матрицами, как транспонирование, диагонализация, определение следа, ранга,

определителя матрицы и т.п. можно воспользоваться библиотекой (пакетом) LinearAlgebra:

```
# Подключение пакета LinearAlgebra:
import Pkg
Pkg.add("LinearAlgebra")
using LinearAlgebra
```

```
   Resolving package versions...
    Updating `C:\Users\GlebB\.julia\environments\v1.9\Project.toml`
  [37e2e46d] + LinearAlgebra
  No Changes to `C:\Users\GlebB\.julia\environments\v1.9\Manifest.toml`
```

```
# Массив 4x4 со случайными целыми числами (от 1 до 20):
b = rand(1:20,(4,4))
# Транспонирование:
transpose(b)
```

```
4x4 transpose(::Matrix{Int64}) with eltype Int64:
  1  11   6   6
 12   5  14   7
  6  11   5  13
  2  12  14  15
```

```
# След матрицы (сумма диагональных элементов):
tr(b)
```

```
26
```

```
# Извлечение диагональных элементов как массив:
diag(b)
```

```
4-element Vector{Int64}:
  8
 12
 18
 15
```

```
# Ранг матрицы:
rank(b)
```

```
4
```

```
# Инверсия матрицы (определение обратной матрицы):
inv(b)
```

```
4x4 Matrix{Float64}:
 -0.000289603   0.16044       0.0133217  -0.140747
  0.0643885    -0.00453712    0.0381311  -0.0405445
  0.067381      0.00424751   -0.099527    0.0805097
 -0.088329     -0.0657399     0.0631335   0.0721112
```

```
# Определитель матрицы:
det(b)
```

```
10359.0
```

```
# Псевдобратная функция для прямоугольных матриц:
pinv(a)
```

```
3x4 Matrix{Float64}:
 -0.014341    -0.0241341    0.00445983   0.0619098
  0.023717    -0.0373237    0.0767987   -0.030195
  0.0212517    0.0510161   -0.0217465   -0.0161695
```

3.3.3. Вычисление нормы векторов и матриц, повороты, вращения

3. Для вычисления нормы используется LinearAlgebra.norm(x).

   Евклидова норма:

$$\|\vec{X}\|_2 = \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2};$$

p-норма:

$$\|\vec{A}\|_p = \left(\sum_{i=1}^{n} |a_i|^p \right)^{1/p}.$$

```
# Создание вектора X:
X = [2, 4, -5]
# Вычисление евклидовой нормы:
norm(X)
```

```
6.708203932499369
```

```
# Вычисление p-нормы:
p = 1
norm(X,p)
```

```
11.0
```

```
# Расстояние между двумя векторами X и Y:
X = [2, 4, -5];
Y = [1,-1,3];
norm(X-Y)
```

```
9.486832980505138
```

```
# Проверка по базовому определению:
sqrt(sum((X-Y).^2))
```

```
9.486832980505138
```

```
# Угол между двумя векторами:
acos((transpose(X)*Y)/(norm(X)*norm(Y)))
```

```
2.4404307889469252
```

```
# Создание матрицы:
d = [5 -4 2 ; -1 2 3; -2 1 0]
```

```
3×3 Matrix{Int64}:
  5  -4  2
 -1   2  3
 -2   1  0
```

```
# Вычисление Евклидовой нормы:
opnorm(d)
```

```
7.147682841795258
```

```
# Вычисление p-нормы:
p=1
opnorm(d,p)
```

```
8.0
```

```
# Поворот на 180 градусов:
rot180(d)
```

```
3×3 Matrix{Int64}:
 0   1  -2
 3   2  -1
 2  -4   5
```

```
# Переворачивание строк:
reverse(d,dims=1)
```

```
3×3 Matrix{Int64}:
```

3.3.4. Матричное умножение, единичная матрица, скалярное произведение

4.

```
# Матрица 2x3 со случайными целыми значениями от 1 до 10:
A = rand(1:10,(2,3))
```

```
2×3 Matrix{Int64}:
 6  9  10
 8  4  10
```

```
# Матрица 3x4 со случайными целыми значениями от 1 до 10:
B = rand(1:10,(3,4))
```

```
3×4 Matrix{Int64}:
 9   1  9  10
 4  10  8   3
 6   8  3   1
```

```
# Произведение матриц A и B:
A*B
```

```
2×4 Matrix{Int64}:
 150  176  156   97
 148  128  134  102
```

```
# Единичная матрица 3x3:
Matrix{Int}(I, 3, 3)
```

```
3×3 Matrix{Int64}:
 1  0  0
 0  1  0
 0  0  1
```

```
# Скалярное произведение векторов X и Y:
X = [2, 4, -5]
Y = [1,-1,3]
dot(X,Y)
```

```
-17
```

```
# тоже скалярное произведение:
X'Y
```

```
-17
```

3.3.5. Факторизация. Специальные матричные структуры

5. В математике факторизация (или разложение) объекта — его декомпозиция (например, числа, полинома или матрицы) в произведение других объектов или факторов, которые, будучи перемноженными, дают исходный объект.

```julia
# Задаём квадратную матрицу 3x3 со случайными значениями:
A = rand(3, 3)
```

```
3×3 Matrix{Float64}:
 0.675351  0.298063  0.933472
 0.873562  0.631751  0.686768
 0.264798  0.355774  0.183635
```

```julia
# Задаём единичный вектор:
x = fill(1.0, 3)
```

```
3-element Vector{Float64}:
 1.0
 1.0
 1.0
```

```julia
# Задаём вектор b:
b = A*x
```

```
3-element Vector{Float64}:
 1.9068863752704455
 2.192080508690627
 0.8042070810272597
```

```julia
# Решение исходного уравнения получаем с помощью функции \
# (убеждаемся, что x - единичный вектор):
A\b
```

```
3-element Vector{Float64}:
 1.0000000000000013
 0.9999999999999994
 0.9999999999999993
```

```julia
# LU-факторизация:
Alu = lu(A)
```

```
LU{Float64, Matrix{Float64}, Vector{Int64}}
L factor:
3×3 Matrix{Float64}:
 1.0        0.0       0.0
 0.773101   1.0       0.0
 0.303124  -0.863039  1.0
U factor:
3×3 Matrix{Float64}:
 0.873562   0.631751  0.686768
 0.0       -0.190344  0.402531
 0.0        0.0       0.322859
```

```julia
# Матрица перестановок:
Alu.P
```

```
3×3 Matrix{Float64}:
 0.0  1.0  0.0
 1.0  0.0  0.0
 0.0  0.0  1.0
```

```julia
# Вектор перестановок:
Alu.p
```

```
3-element Vector{Int64}:
```

3.3.6. Общая линейная алгебра

6. Обычный способ добавить поддержку числовой линейной алгебры - это обернуть подпрограммы BLAS и LAPACK. Собственно, для матриц с элементами Float32,Float64,Complex {Float32} или Complex {Float64} разработчики Julia использовали такое же решение. Однако Julia также

поддерживает общую линейную алгебру, что позволяет, например, работать с матрицами и векторами рациональных чисел.

Для задания рационального числа используется двойная косая черта:

1//2

В следующем примере показано, как можно решить систему линейных уравнений с рациональными элементами без преобразования в типы элементов с плавающей запятой (для избежания проблемы с переполнением используем BigInt):

```
# Матрица с рациональными элементами:
Arational = Matrix{Rational{BigInt}}(rand(1:10, 3, 3))/10
```

```
3x3 Matrix{Rational{BigInt}}:
 1//1   3//5   1//10
 7//10  7//10  9//10
 3//5   3//10  7//10
```

```
# Единичный вектор:
x = fill(1, 3)
```

```
3-element Vector{Int64}:
 1
 1
 1
```

```
# Задаём вектор b:
b = Arational*x
```

```
3-element Vector{Rational{BigInt}}:
 17//10
 23//10
  8//5
```

```
# Решение исходного уравнения получаем с помощью функции \
# (убеждаемся, что x - единичный вектор):
Arational\b
```

```
3-element Vector{Rational{BigInt}}:
 1//1
 1//1
 1//1
```

```
# LU-разложение:
lu(Arational)
```

```
LU{Rational{BigInt}, Matrix{Rational{BigInt}}, Vector{Int64}}
L factor:
3x3 Matrix{Rational{BigInt}}:
 1//1    0//1   0//1
 7//10   1//1   0//1
 3//5   -3//14  1//1
U factor:
3x3 Matrix{Rational{BigInt}}:
 1//1  3//5    1//10
 0//1  7//25   83//100
 0//1  0//1   229//280
```

3.3.7. Задания для самостоятельного выполнения

7. Задайте вектор v. Умножьте вектор v скалярно сам на себя и сохраните результат в dot_v.

   Умножьте v матрично на себя (внешнее произведение), присвоив результат переменной outer_v.

```
v = [3, 5, 2, 9]
dot_v = dot(v, v)
```

```
119
```

```
outer_v = v * v'
```

```
4×4 Matrix{Int64}:
  9  15   6  27
 15  25  10  45
  6  10   4  18
 27  45  18  81
```

8. Решить СЛАУ с двумя неизвестными

```
# Left - лево, Right - право
L1 = [1 1; 1 -1]
R1 = [2; 3]
```

```
2-element Vector{Int64}:
 2
 3
```

```
L1\R1
```

```
2-element Vector{Float64}:
  2.5
 -0.5
```

```
L2 = [1 1; 2 2]
R2 = [2; 4]
```

```
2-element Vector{Int64}:
 2
 4
```

```
L2\R2
```

```
SingularException(2)

Stacktrace:
 [1] checknonsingular
   @ C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\factorization.jl:19 [inlined]
 [2] checknonsingular
   @ C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\factorization.jl:22 [inlined]
 [3] #lu!#170
```

```
L3 = [1 1; 2 2]
R3 = [2; 5]
```

```
2-element Vector{Int64}:
 2
 5
```

```
L3\R3
```

```
SingularException(2)

Stacktrace:
 [1] checknonsingular
   @ C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\factorization.jl:19 [inlined]
 [2] checknonsingular
   @ C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\factorization.jl:22 [inlined]
 [3] #lu!#170
   @ C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\lu.jl:82 [inlined]
 [4] lu!
   @ C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\lu.jl:80 [inlined]
 [5] lu(A::Matrix{Int64}, pivot::RowMaximum; check::Bool)
   @ LinearAlgebra C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\lu.jl:299
 [6] lu (repeats 2 times)
   @ C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\lu.jl:298 [inlined]
 [7] \(A::Matrix{Int64}, B::Vector{Int64})
   @ LinearAlgebra C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\generic.jl:1115
 [8] top-level scope
   @ In[104]:1
```

```
L4 = [1 1; 2 2; 3 3]
R4 = [1; 2; 3]
```



```
R4 = [1; 2; 3]
```



```
3-element Vector{Int64}:
 1
 2
 3
```

```
L4\R4
```

```
2-element Vector{Float64}:
 0.4999999999999999
 0.5
```

```
L5 = [1 1; 2 1; 1 -1]
R5 = [2; 1; 3]
```

```
3-element Vector{Int64}:
 2
 1
 3
```

```
L5\R5
```

```
2-element Vector{Float64}:
  1.5000000000000004
 -0.9999999999999997
```

```
L6 = [1 1; 2 1; 3 2]
R6 = [2; 1; 3]
```

```
3-element Vector{Int64}:
 2
```

9. Решить СЛАУ с тремя неизвестными

```
L7 = [1 1 1; 1 -1 -2]
R7 = [2; 3]
L7\R7
```

```
3-element Vector{Float64}:
  2.2142857142857144
  0.35714285714285704
 -0.5714285714285712
```

```
L8 = [1 1 1; 2 2 -3; 3 1 1]
R8 = [2; 4; 1]
L8\R8
```

```
3-element Vector{Float64}:
 -0.5
  2.5
  0.0
```

```
L9 = [1 1 1; 1 1 2; 2 2 3]
R9 = [1; 0; 1]
L9\R9
```

```
SingularException(2)

Stacktrace:
 [1] checknonsingular
   @ C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\factorization.jl:19 [inlined]
 [2] checknonsingular
   @ C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\factorization.jl:22 [inlined]
```

```
L10 = [1 1 1; 1 1 2; 2 2 3]
R10 = [1; 0; 0]
L10\R10
```

```
SingularException(2)

Stacktrace:
 [1] checknonsingular
   @ C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\factorization.jl:19 [inlined]
 [2] checknonsingular
   @ C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\factorization.jl:22 [inlined]
 [3] #lu!#170
   @ C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\lu.jl:82 [inlined]
 [4] lu!
   @ C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\lu.jl:80 [inlined]
 [5] lu(A::Matrix{Int64}, pivot::RowMaximum; check::Bool)
   @ LinearAlgebra C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\lu.jl:299
 [6] lu (repeats 2 times)
   @ C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\lu.jl:298 [inlined]
 [7] \(A::Matrix{Int64}, B::Vector{Int64})
   @ LinearAlgebra C:\Users\GlebB\AppData\Local\Programs\Julia-1.9.3\share\julia\stdlib\v1.9\LinearAlgebra\src\generic.jl:1115
 [8] top-level scope
   @ In[112]:3
```

10. Приведите приведённые ниже матрицы к диагональному виду

```
function dia(mat)
    simm = mat + mat'
    razsimm = eigen(simm)
    return inv(razsimm.vectors) * mat * razsimm.vectors
end
```

```
dia (generic function with 1 method)
```

```
mat1 = [1 -2; -2 1]
dia(mat1)
```

```
2×2 Matrix{Float64}:
 -1.0  0.0
  0.0  3.0
```

```
mat2 = [1 -2; -2 3]
dia(mat2)
```

```
2×2 Matrix{Float64}:
 -0.236068     3.46945e-16
  4.44089e-16  4.23607
```

```
mat3 = [1 -2 0; -2 1 2; 0 2 0]
dia(mat3)
```

```
3×3 Matrix{Float64}:
 -2.14134      3.55271e-15  -1.9984e-15
  3.38618e-15  0.515138      1.11022e-16
 -6.66134e-16 -4.44089e-16   3.6262
```

11. Вычислите

```
([1 -2; -2 1])^10
```

```
2×2 Matrix{Int64}:
  29525  -29524
 -29524   29525
```

```
sqrt([5 -2; -2 5])
```

```
2×2 Matrix{Float64}:
  2.1889   -0.45685
 -0.45685   2.1889
```

```
([1 -2; -2 1])^(1/3)
```

```
2×2 Symmetric{ComplexF64, Matrix{ComplexF64}}:
  0.971125+0.433013im  -0.471125+0.433013im
 -0.471125+0.433013im   0.971125+0.433013im
```

```
sqrt([1 2; 2 3])
```

```
2×2 Matrix{ComplexF64}:
 0.568864+0.351578im  0.920442-0.217287im
 0.920442-0.217287im   1.48931+0.134291im
```

12. Найдите собственные значения матрицы A, если

$$A = \begin{pmatrix} 140 & 97 & 74 & 168 & 131 \\ 97 & 106 & 89 & 131 & 36 \\ 74 & 89 & 152 & 144 & 71 \\ 168 & 131 & 144 & 54 & 142 \\ 131 & 36 & 71 & 142 & 36 \end{pmatrix}.$$

Создайте диагональную матрицу из собственных значений матрицы A. Создайте нижнедиагональную матрицу из матрица A. Оцените эффективность выполняемых операций.

```
A = [140 97 74 168 131; 97 106 89 131 36; 74 89 152 144 71; 168 131 144 54 142; 131 36 71 142 36]
val = eigvals(A)
```

```
5-element Vector{Float64}:
 -128.49322764802145
  -55.887784553056875
   42.7521672793189
   87.16111477514521
  542.4677301466143
```

```
@btime eigvals(A)
```

```
  3.700 μs (10 allocations: 2.59 KiB)
```

```
5-element Vector{Float64}:
 -128.49322764802145
  -55.887784553056875
   42.7521672793189
   87.16111477514521
  542.4677301466143
```

```
N = zeros(4, 4)
@btime for i in 1:1:4
    N[i, i] = val[i]
end
N
```

```
  241.646 ns (4 allocations: 64 bytes)

4x4 Matrix{Float64}:
 -128.493    0.0      0.0      0.0
    0.0    -55.8878   0.0      0.0
    0.0      0.0     42.7522   0.0
    0.0      0.0      0.0     87.1611
```

```
Alu = lu(A)
@btime Alu.L
```

```
  147.407 ns (1 allocation: 256 bytes)

5x5 Matrix{Float64}:
 1.0        0.0        0.0        0.0       0.0
 0.779762   1.0        0.0        0.0       0.0
 0.440476  -0.47314    1.0        0.0       0.0
 0.833333   0.183929  -0.556312   1.0       0.0
 0.577381  -0.459012  -0.189658   0.897068  1.0
```

13. Матрица A называется продуктивной, если решение x системы при любой неотрицательной правой части y имеет только неотрицательные элементы $xi/$. Используя это определение, проверьте, являются ли матрицы продуктивными.

```
function matr(mat, s)
    ans = ""
    P = [1 0; 0 1]
    K = rand(0:100, s)
    H = P - mat
    T = H\K
    for i in 1:1:s
        if T[i] < 0
            ans = "no"
            break
        else
            ans = "yes"
        end
    end
    return ans
end
```

```
matr (generic function with 1 method)
```

```
mat1 = [1 2; 3 4]
matr(mat1, 2)
```

```
"no"
```

```
mat2 = ([1 2; 3 4])*(1/2)
matr(mat2, 2)
```

```
"no"
```

```
mat3 = ([1 2; 3 4])*(1/10)
matr(mat3, 2)
```

```
"yes"
```

Вывод

В данной лабораторной работе мне успешно удалось изучить возможности специализированных пакетов Julia для выполнения и оценки эффективности операций над объектами линейной алгебры.

Приложение

123

{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": 3,
   "id": "3f957b55",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "4×3 Matrix{Int64}:\n",
       " 9   9 18\n",
       " 6   2 19\n",
       " 8 12   8\n",
       " 20   2 11"
      ]
     },
     "execution_count": 3,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "# Массив 4x3 со случайными целыми числами (от 1 до 20):\n",
    "a = rand(1:20,(4,3))"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 8,
   "id": "d13f0281",
   "metadata": {
    "scrolled": true
   },
   "outputs": [
    {
     "data": {
      "text/plain": [
       "124"
      ]
     },
     "execution_count": 8,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "# Поэлементная сумма:\n",
    "sum(a)"

```
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 9,
  "id": "57afd29a",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "1×3 Matrix{Int64}:\n",
      " 43  25  56"
     ]
    },
    "execution_count": 9,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "# Поэлементная сумма по столбцам:\n",
   "sum(a,dims=1)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 10,
  "id": "e35176b2",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "4×1 Matrix{Int64}:\n",
      " 36\n",
      " 27\n",
      " 28\n",
      " 33"
     ]
    },
    "execution_count": 10,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "# Поэлементная сумма по строкам:\n",
   "sum(a,dims=2)"
  ]
 },
 {
  "cell_type": "code",
```

```
 "execution_count": 6,
 "id": "64dab0ab",
 "metadata": {
  "scrolled": true
 },
 "outputs": [
  {
   "data": {
    "text/plain": [
     "4×1 Matrix{Int64}:\n",
     " 1458\n",
     "  228\n",
     "  768\n",
     "  440"
    ]
   },
   "execution_count": 6,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Поэлементное произведение:\n",
  "prod(a)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 11,
 "id": "4d4cd5df",
 "metadata": {
  "scrolled": true
 },
 "outputs": [
  {
   "data": {
    "text/plain": [
     "1×3 Matrix{Int64}:\n",
     " 8640  432  30096"
    ]
   },
   "execution_count": 11,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Поэлементное произведение по столбцам:\n",
  "prod(a,dims=1)"
 ]
},
{
 "cell_type": "code",
```

```
   "execution_count": 12,
   "id": "a55a15b5",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "4×1 Matrix{Int64}:\n",
       " 1458\n",
       "  228\n",
       "  768\n",
       "  440"
      ]
     },
     "execution_count": 12,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "# Поэлементное произведение по строкам:\n",
    "prod(a,dims=2)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 7,
   "id": "f87e1005",
   "metadata": {},
   "outputs": [
    {
     "name": "stderr",
     "output_type": "stream",
     "text": [
      "\u001b[32m\u001b[1m    Updating\u001b[22m\u001b[39m registry at `C:\\Users\\GlebB\\.julia\\registries\\General.toml`\n",
      "\u001b[32m\u001b[1m   Resolving\u001b[22m\u001b[39m package versions...\n",
      "\u001b[32m\u001b[1m    Updating\u001b[22m\u001b[39m `C:\\Users\\GlebB\\.julia\\environments\\v1.9\\Project.toml`\n",
      "  \u001b[90m[10745b16] \u001b[39m\u001b[92m+ Statistics v1.9.0\u001b[39m\n",
      "\u001b[32m\u001b[1m  No Changes\u001b[22m\u001b[39m to `C:\\Users\\GlebB\\.julia\\environments\\v1.9\\Manifest.toml`\n"
     ]
    },
    {
     "data": {
      "text/plain": [
       "4×1 Matrix{Float64}:\n",
       " 12.0\n",
       "  9.0\n",
       "  9.333333333333334\n",
       " 11.0"
      ]
     },
     "execution_count": 7,
     "metadata": {},
```

"output_type": "execute_result"
}
],
"source": [
"# Подключение пакета Statistics:\n",
"import Pkg\n",
"Pkg.add(\"Statistics\")\n",
"using Statistics"
]
},
{
"cell_type": "code",
"execution_count": 13,
"id": "31612196",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"10.333333333333334"
]
},
"execution_count": 13,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Вычисление среднего значения массива:\n",
"mean(a)"
]
},
{
"cell_type": "code",
"execution_count": 14,
"id": "54f77033",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"1×3 Matrix{Float64}:\n",
" 10.75 6.25 14.0"
]
},
"execution_count": 14,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Среднее по столбцам:\n",
"mean(a,dims=1)"

```
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 15,
  "id": "f1d91985",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "4×1 Matrix{Float64}:\n",
      " 12.0\n",
      "  9.0\n",
      "  9.333333333333334\n",
      " 11.0"
     ]
    },
    "execution_count": 15,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "# Среднее по строкам:\n",
   "mean(a,dims=2)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 16,
  "id": "cda69138",
  "metadata": {},
  "outputs": [
   {
    "name": "stderr",
    "output_type": "stream",
    "text": [
     "\u001b[32m\u001b[1m   Resolving\u001b[22m\u001b[39m package versions...\n",
     "\u001b[32m\u001b[1m    Updating\u001b[22m\u001b[39m `C:\\Users\\GlebB\\.julia\\environments\\v1.9\\Project.toml`\n",
     "  \u001b[90m[37e2e46d] \u001b[39m\u001b[92m+ LinearAlgebra\u001b[39m\n",
     "\u001b[32m\u001b[1m  No Changes\u001b[22m\u001b[39m to `C:\\Users\\GlebB\\.julia\\environments\\v1.9\\Manifest.toml`\n"
    ]
   }
  ],
  "source": [
   "# Подключение пакета LinearAlgebra:\n",
   "import Pkg\n",
   "Pkg.add(\"LinearAlgebra\")\n",
   "using LinearAlgebra"
  ]
 },
 {
```

"cell_type": "code",
"execution_count": 19,
"id": "a0d78f72",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [
    "4×4 transpose(::Matrix{Int64}) with eltype Int64:\n",
    " 1 11   6   6\n",
    " 12   5 14   7\n",
    " 6 11   5 13\n",
    " 2 12 14 15"
   ]
  },
  "execution_count": 19,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "# Массив 4x4 со случайными целыми числами (от 1 до 20):\n",
 "b = rand(1:20,(4,4))\n",
 "# Транспонирование:\n",
 "transpose(b)"
]
},
{
"cell_type": "code",
"execution_count": 20,
"id": "a81efca6",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [
    "26"
   ]
  },
  "execution_count": 20,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "# След матрицы (сумма диагональных элементов):\n",
 "tr(b)"
]
},
{
"cell_type": "code",
"execution_count": 18,
"id": "86631f36",

"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [
    "4-element Vector{Int64}:\n",
    " 8\n",
    " 12\n",
    " 18\n",
    " 15"
   ]
  },
  "execution_count": 18,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "# Извлечение диагональных элементов как массив:\n",
 "diag(b)"
]
},
{
"cell_type": "code",
"execution_count": 21,
"id": "c1661674",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [
    "4"
   ]
  },
  "execution_count": 21,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "# Ранг матрицы:\n",
 "rank(b)"
]
},
{
"cell_type": "code",
"execution_count": 22,
"id": "ba00463a",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [

```
     "4×4 Matrix{Float64}:\n",
     " -0.000289603   0.16044      0.0133217  -0.140747\n",
     "  0.0643885    -0.00453712   0.0381311  -0.0405445\n",
     "  0.067381      0.00424751  -0.099527    0.0805097\n",
     " -0.088329     -0.0657399    0.0631335   0.0721112"
    ]
   },
   "execution_count": 22,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Инверсия матрицы (определение обратной матрицы):\n",
  "inv(b)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 23,
 "id": "b86741b9",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "10359.0"
    ]
   },
   "execution_count": 23,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Определитель матрицы:\n",
  "det(b)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 24,
 "id": "c195e52e",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "3×4 Matrix{Float64}:\n",
     " -0.014341   -0.0241341   0.00445983   0.0619098\n",
     "  0.023717   -0.0373237   0.0767987   -0.030195\n",
     "  0.0212517   0.0510161  -0.0217465   -0.0161695"
    ]
```

},
"execution_count": 24,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Псевдобратная функция для прямоугольных матриц:\n",
"pinv(a)"
]
},
{
"cell_type": "code",
"execution_count": 25,
"id": "a2641623",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"11.0"
]
},
"execution_count": 25,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Создание вектора X:\n",
"X = [2, 4, -5]\n",
"# Вычисление евклидовой нормы:\n",
"norm(X)\n",
"# Вычисление p-нормы:\n",
"p = 1\n",
"norm(X,p)"
]
},
{
"cell_type": "code",
"execution_count": 26,
"id": "34363091",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"9.486832980505138"
]
},
"execution_count": 26,
"metadata": {},
"output_type": "execute_result"

}
],
"source": [
"# Расстояние между двумя векторами X и Y:\n",
"X = [2, 4, -5];\n",
"Y = [1,-1,3];\n",
"norm(X-Y)"
]
},
{
"cell_type": "code",
"execution_count": 27,
"id": "7302a43a",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"9.486832980505138"
]
},
"execution_count": 27,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Проверка по базовому определению:\n",
"sqrt(sum((X-Y).^2))"
]
},
{
"cell_type": "code",
"execution_count": 28,
"id": "bab1f5e8",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"2.4404307889469252"
]
},
"execution_count": 28,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Угол между двумя векторами:\n",
"acos((transpose(X)*Y)/(norm(X)*norm(Y)))"
]
},

```
{
 "cell_type": "code",
 "execution_count": 29,
 "id": "2f673691",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "3×3 Matrix{Int64}:\n",
     "  5  -4  2\n",
     " -1   2  3\n",
     " -2   1  0"
    ]
   },
   "execution_count": 29,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Создание матрицы:\n",
  "d = [5 -4 2 ; -1 2 3; -2 1 0]"
 ]
},
{
 "cell_type": "code",
 "execution_count": 30,
 "id": "d84a7db6",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "7.147682841795258"
    ]
   },
   "execution_count": 30,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Вычисление Евклидовой нормы:\n",
  "opnorm(d)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 31,
 "id": "88647050",
 "metadata": {},
 "outputs": [
```

{
"data": {
"text/plain": [
"8.0"
]
},
"execution_count": 31,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Вычисление p-нормы:\n",
"p=1\n",
"opnorm(d,p)"
]
},
{
"cell_type": "code",
"execution_count": 32,
"id": "ca35133d",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"3×3 Matrix{Int64}:\n",
" 0 1 -2\n",
" 3 2 -1\n",
" 2 -4 5"
]
},
"execution_count": 32,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Поворот на 180 градусов:\n",
"rot180(d)"
]
},
{
"cell_type": "code",
"execution_count": 33,
"id": "c2946420",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"3×3 Matrix{Int64}:\n",
" -2 1 0\n",

```
     " -1   2  3\n",
     "  5  -4  2"
    ]
   },
   "execution_count": 33,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Переворачивание строк:\n",
  "reverse(d,dims=1)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 34,
 "id": "2996ed3a",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "3×3 Matrix{Int64}:\n",
     " 2  -4   5\n",
     " 3   2  -1\n",
     " 0   1  -2"
    ]
   },
   "execution_count": 34,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Переворачивание столбцов\n",
  "reverse(d,dims=2)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 35,
 "id": "b8003685",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "2×3 Matrix{Int64}:\n",
     " 6  9  10\n",
     " 8  4  10"
    ]
   },
```

"execution_count": 35,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Матрица 2x3 со случайными целыми значениями от 1 до 10:\n",
"A = rand(1:10,(2,3))"
]
},
{
"cell_type": "code",
"execution_count": 36,
"id": "8cf2e5c9",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"3×4 Matrix{Int64}:\n",
" 9   1 9 10\n",
" 4 10 8   3\n",
" 6   8 3   1"
]
},
"execution_count": 36,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Матрица 3x4 со случайными целыми значениями от 1 до 10:\n",
"B = rand(1:10,(3,4))"
]
},
{
"cell_type": "code",
"execution_count": 37,
"id": "e7657959",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"2×4 Matrix{Int64}:\n",
" 150 176 156   97\n",
" 148 128 134 102"
]
},
"execution_count": 37,
"metadata": {},
"output_type": "execute_result"
}

],
"source": [
"# Произведение матриц A и B:\n",
"A*B"
]
},
{
"cell_type": "code",
"execution_count": 38,
"id": "19f4be61",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"3×3 Matrix{Int64}:\n",
" 1 0 0\n",
" 0 1 0\n",
" 0 0 1"
]
},
"execution_count": 38,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Единичная матрица 3x3:\n",
"Matrix{Int}(I, 3, 3)"
]
},
{
"cell_type": "code",
"execution_count": 39,
"id": "96f606f3",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"-17"
]
},
"execution_count": 39,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Скалярное произведение векторов X и Y:\n",
"X = [2, 4, -5]\n",
"Y = [1,-1,3]\n",
"dot(X,Y)"

```
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 40,
  "id": "a8a8ed2a",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "-17"
     ]
    },
    "execution_count": 40,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "# тоже скалярное произведение:\n",
   "X'Y"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 41,
  "id": "b91b14e6",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "3×3 Matrix{Float64}:\n",
      " 0.675351  0.298063  0.933472\n",
      " 0.873562  0.631751  0.686768\n",
      " 0.264798  0.355774  0.183635"
     ]
    },
    "execution_count": 41,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "# Задаём квадратную матрицу 3x3 со случайными значениями:\n",
   "A = rand(3, 3)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 42,
  "id": "46ab8349",
```

```
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "3-element Vector{Float64}:\n",
     " 1.0\n",
     " 1.0\n",
     " 1.0"
    ]
   },
   "execution_count": 42,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Задаём единичный вектор:\n",
  "x = fill(1.0, 3)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 43,
 "id": "007bc031",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "3-element Vector{Float64}:\n",
     " 1.9068863752704455\n",
     " 2.192080508690627\n",
     " 0.8042070810272597"
    ]
   },
   "execution_count": 43,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Задаём вектор b:\n",
  "b = A*x"
 ]
},
{
 "cell_type": "code",
 "execution_count": 44,
 "id": "67557709",
 "metadata": {},
 "outputs": [
  {
```

```
   "data": {
    "text/plain": [
     "3-element Vector{Float64}:\n",
     " 1.0000000000000013\n",
     " 0.9999999999999994\n",
     " 0.9999999999999993"
    ]
   },
   "execution_count": 44,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Решение исходного уравнения получаем с помощью функции \\\n",
  "# (убеждаемся, что x - единичный вектор):\n",
  "A\\b"
 ]
},
{
 "cell_type": "code",
 "execution_count": 45,
 "id": "be0ba21b",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "LU{Float64, Matrix{Float64}, Vector{Int64}}\n",
     "L factor:\n",
     "3×3 Matrix{Float64}:\n",
     " 1.0        0.0       0.0\n",
     " 0.773101   1.0       0.0\n",
     " 0.303124  -0.863039  1.0\n",
     "U factor:\n",
     "3×3 Matrix{Float64}:\n",
     " 0.873562   0.631751  0.686768\n",
     " 0.0       -0.190344  0.402531\n",
     " 0.0        0.0       0.322859"
    ]
   },
   "execution_count": 45,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# LU-факторизация:\n",
  "Alu = lu(A)"
 ]
},
{
 "cell_type": "code",
```

```
 "execution_count": 46,
 "id": "440198f2",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "3×3 Matrix{Float64}:\n",
     " 0.0  1.0  0.0\n",
     " 1.0  0.0  0.0\n",
     " 0.0  0.0  1.0"
    ]
   },
   "execution_count": 46,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Матрица перестановок:\n",
  "Alu.P"
 ]
},
{
 "cell_type": "code",
 "execution_count": 47,
 "id": "f0700b7a",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "3-element Vector{Int64}:\n",
     " 2\n",
     " 1\n",
     " 3"
    ]
   },
   "execution_count": 47,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Вектор перестановок:\n",
  "Alu.p"
 ]
},
{
 "cell_type": "code",
 "execution_count": 48,
 "id": "d15ac2b7",
 "metadata": {},
```

"outputs": [
{
"data": {
"text/plain": [
"3×3 Matrix{Float64}:\n",
" 1.0   0.0   0.0\n",
" 0.773101   1.0   0.0\n",
" 0.303124 -0.863039 1.0"
]
},
"execution_count": 48,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Матрица L:\n",
"Alu.L"
]
},
{
"cell_type": "code",
"execution_count": 49,
"id": "eca774ba",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"3×3 Matrix{Float64}:\n",
" 0.873562   0.631751 0.686768\n",
" 0.0   -0.190344 0.402531\n",
" 0.0   0.0   0.322859"
]
},
"execution_count": 49,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Матрица U:\n",
"Alu.U"
]
},
{
"cell_type": "code",
"execution_count": 50,
"id": "bddc51cb",
"metadata": {},
"outputs": [
{
"data": {

```
    "text/plain": [
     "3-element Vector{Float64}:\n",
     " 1.0000000000000013\n",
     " 0.9999999999999994\n",
     " 0.9999999999999993"
    ]
   },
   "execution_count": 50,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Решение СЛАУ через матрицу A:\n",
  "A\\b"
 ]
},
{
 "cell_type": "code",
 "execution_count": 51,
 "id": "a4c6db8e",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "3-element Vector{Float64}:\n",
     " 1.0000000000000013\n",
     " 0.9999999999999994\n",
     " 0.9999999999999993"
    ]
   },
   "execution_count": 51,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Решение СЛАУ через объект факторизации:\n",
  "Alu\\b"
 ]
},
{
 "cell_type": "code",
 "execution_count": 52,
 "id": "502974dd",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "0.05368420908258649"
    ]
```

```
   },
   "execution_count": 52,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Детерминант матрицы A:\n",
  "det(A)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 53,
 "id": "58109a2a",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "0.05368420908258649"
    ]
   },
   "execution_count": 53,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Детерминант матрицы A через объект факторизации:\n",
  "det(Alu)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 54,
 "id": "772a0a3b",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "LinearAlgebra.QRCompactWY{Float64, Matrix{Float64}, Matrix{Float64}}\n",
     "Q factor:\n",
     "3×3 LinearAlgebra.QRCompactWYQ{Float64, Matrix{Float64}, Matrix{Float64}}:\n",
     " -0.594769   0.607444   0.526557\n",
     " -0.769329  -0.240086  -0.592024\n",
     " -0.233202  -0.757213   0.610119\n",
     "R factor:\n",
     "3×3 Matrix{Float64}:\n",
     " -1.13549  -0.74627   -1.12637\n",
     "  0.0      -0.240014   0.263098\n",
     "  0.0       0.0        0.196982"
```

```
    ]
   },
   "execution_count": 54,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# QR-факторизация:\n",
  "Aqr = qr(A)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 55,
 "id": "69a85549",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "3×3 LinearAlgebra.QRCompactWYQ{Float64, Matrix{Float64}, Matrix{Float64}}:\n",
     " -0.594769   0.607444   0.526557\n",
     " -0.769329  -0.240086  -0.592024\n",
     " -0.233202  -0.757213   0.610119"
    ]
   },
   "execution_count": 55,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Матрица Q:\n",
  "Aqr.Q"
 ]
},
{
 "cell_type": "code",
 "execution_count": 56,
 "id": "7af9f1f5",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "3×3 Matrix{Float64}:\n",
     " -1.13549  -0.74627   -1.12637\n",
     "  0.0      -0.240014   0.263098\n",
     "  0.0       0.0        0.196982"
    ]
   },
   "execution_count": 56,
```

"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Матрица R:\n",
"Aqr.R"
]
},
{
"cell_type": "code",
"execution_count": 57,
"id": "a2d8aae7",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"3×3 Matrix{Float64}:\n",
" 1.0 -1.11022e-16 0.0\n",
" 0.0   1.0   2.22045e-16\n",
" 0.0   2.22045e-16 1.0"
]
},
"execution_count": 57,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Проверка, что матрица Q - ортогональная:\n",
"Aqr.Q'*Aqr.Q"
]
},
{
"cell_type": "code",
"execution_count": 58,
"id": "9ef341c2",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"3×3 Matrix{Float64}:\n",
" 1.3507   1.17162 1.19827\n",
" 1.17162 1.2635   1.04254\n",
" 1.19827 1.04254 0.36727"
]
},
"execution_count": 58,
"metadata": {},
"output_type": "execute_result"
}

```
   ],
   "source": [
    "# Симметризация матрицы A:\n",
    "Asym = A + A'"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 60,
   "id": "801d1f00",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "Eigen{Float64, Float64, Matrix{Float64}, Vector{Float64}}\n",
       "values:\n",
       "3-element Vector{Float64}:\n",
       " -0.49037528464639485\n",
       "  0.1423009515620718\n",
       "  3.3295498969172534\n",
       "vectors:\n",
       "3×3 Matrix{Float64}:\n",
       " -0.415539   0.643362   -0.642972\n",
       " -0.243369  -0.759761   -0.602938\n",
       "  0.876412   0.0940649  -0.472285"
      ]
     },
     "execution_count": 60,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "# Спектральное разложение симметризованной матрицы:\n",
    "AsymEig = eigen(Asym)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 61,
   "id": "379d1b99",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "3-element Vector{Float64}:\n",
       " -0.49037528464639485\n",
       "  0.1423009515620718\n",
       "  3.3295498969172534"
      ]
     },
```

"execution_count": 61,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Собственные значения:\n",
"AsymEig.values"
]
},
{
"cell_type": "code",
"execution_count": 62,
"id": "6d2edb9d",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"3×3 Matrix{Float64}:\n",
" -0.415539   0.643362   -0.642972\n",
" -0.243369 -0.759761   -0.602938\n",
" 0.876412   0.0940649 -0.472285"
]
},
"execution_count": 62,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"#Собственные векторы:\n",
"AsymEig.vectors"
]
},
{
"cell_type": "code",
"execution_count": 63,
"id": "febc6204",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"3×3 Matrix{Float64}:\n",
" 1.0   -2.66454e-15 -4.44089e-16\n",
" 4.13558e-15   1.0   2.20657e-15\n",
" 1.77636e-15   2.22045e-15   1.0"
]
},
"execution_count": 63,
"metadata": {},
"output_type": "execute_result"

```
 }
 ],
 "source": [
  "# Проверяем, что получится единичная матрица:\n",
  "inv(AsymEig)*Asym"
 ]
},
{
 "cell_type": "code",
 "execution_count": 64,
 "id": "5f4f2bc7",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "1000×1000 Matrix{Float64}:\n",
     "  0.19653    -1.28792     1.16627    …  -1.84939    -1.34687     0.0181986\n",
     "  0.100823   -0.242892   -1.78588       -0.951881   -0.694639    1.01759\n",
     "  0.356048    0.687924    0.0288233     -0.81331    -1.04784     2.1041\n",
     "  0.839306    1.17316     0.102227      -2.14399    -0.172797    0.692587\n",
     " -0.0264909   0.0640003  -0.28165        0.938568   -0.516024    0.135838\n",
     "  0.839222    0.541828    1.81463    …  -0.257015    0.206856    1.07932\n",
     " -1.13509    -0.656688    0.958005      -0.872888   -0.0603781  -0.583505\n",
     "  0.0969855  -1.10011     0.414148      -0.0181358  -0.0280408  -0.655324\n",
     " -0.215069    0.685932    0.843887       1.08668    -0.308307   -0.0473633\n",
     " -0.0640355   0.0312235  -0.406964      -2.2979     -0.0320142   0.0212742\n",
     " -0.895282   -0.517428   -0.979889   …  -0.190308    0.544045    0.286461\n",
     " -0.467473    0.118284    2.29478        2.13171    -1.04756     0.552324\n",
     "  1.01328     0.832668   -0.358916      -1.3213     -1.72488     0.69898\n",
     "  ⋮                                  ⋱                          \n",
     "  0.39093    -0.355748    0.0192273     -0.659646    0.557988    1.66871\n",
     "  1.93465     1.13655    -0.312131       0.559977   -0.216137    0.824661\n",
     "  1.21781     0.434912   -2.74921    …   0.599794   -0.733041   -1.76278\n",
     "  0.943539   -0.446311    0.352814      -1.28741     0.472569   -1.2998\n",
     "  0.540156   -0.0606668   0.195867       1.2427     -1.56934    -1.33498\n",
     " -0.703261   -0.400684    0.63522        0.947157    1.66973     1.28292\n",
     " -0.0232253  -0.0330447   0.30982       -0.629506   -0.613275   -1.22788\n",
     "  1.2462      0.393643    0.179153   …   0.800256   -1.06537    -1.16544\n",
     " -0.58862    -0.922899   -1.64867        1.74774    -0.34958    -0.674475\n",
     " -0.8326      1.08633    -1.35123       -0.995951    0.414619   -1.02544\n",
     "  0.249982   -0.427203    0.415          0.869754   -1.3841     -0.0414904\n",
     " -1.77331     0.881116    0.514299       1.81118     0.206007    0.821672"
    ]
   },
   "execution_count": 64,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Матрица 1000 x 1000:\n",
  "n = 1000\n",
```

"A = randn(n,n)"
]
},
{
"cell_type": "code",
"execution_count": 65,
"id": "dbefe8ae",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"1000×1000 Matrix{Float64}:\n",
" 0.393061 -1.1871   1.52232   … -2.68199   -1.09688   -1.75511\n",
" -1.1871   -0.485784 -1.09796   0.134448 -1.12184   1.8987\n",
" 1.52232   -1.09796   0.0576466   -2.16454   -0.632841   2.6184\n",
" 2.20254   2.12601   1.37919   -0.86109   1.38753   1.14129\n",
" 1.14361   0.670914 -0.0388096   -0.222059 -0.324271   0.174666\n",
" 0.4213   0.237232   3.36525   …   0.574236   0.0863073   1.37996\n",
" -1.12982   -0.138938   0.183372   -0.725398   0.580001   -0.369564\n",
" 0.907652 -2.17058   0.748957   -1.53514   -0.627687   -0.127403\n",
" 0.43601   -0.342293   0.25039   1.02085   -0.711883   0.583164\n",
" 0.547735 -0.316938 -0.495361   -0.510295   0.578657   0.0241491\n",
" 0.127273   0.215779   0.0729088 …   1.1158   2.47708   0.760371\n",
" -0.175972   1.07572   2.89464   2.63094   -1.5264   -0.396812\n",
" 2.18405   -0.469264   1.93446   -2.98784   -3.06913   0.64797\n",
" ⋮   ⋱   \n",
" -1.39222   1.79202   -0.233084   -0.323808 -0.37531   1.79995\n",
" 4.67361   2.30666   -0.197534   2.05787   0.0193012   2.52057\n",
" 3.13532   -0.456212 -2.97602   …   2.4854   -0.266278   -1.94299\n",
" 1.4815   -0.628309 -0.447098   -2.64624   0.458817   -1.73581\n",
" 1.02467   -2.23186   0.065106   3.52976   -1.24011   0.257556\n",
" -0.105639 -1.85397   1.48033   0.30017   1.08879   1.27713\n",
" -2.016   -0.236965 -1.47008   0.523953   0.434294   -1.67341\n",
" 1.46506   1.31339   1.66461   … -0.881256 -2.62313   -0.105925\n",
" -1.64201   -2.50723   -2.068   0.708567 -0.294867   -2.18745\n",
" -2.68199   0.134448 -2.16454   -1.9919   1.28437   0.785744\n",
" -1.09688   -1.12184   -0.632841   1.28437   -2.7682   0.164517\n",
" -1.75511   1.8987   2.6184   0.785744   0.164517   1.64334"
]
},
"execution_count": 65,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Симметризация матрицы:\n",
"Asym = A + A'"
]
},
{
"cell_type": "code",

"execution_count": 66,
"id": "678a30b9",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [
    "true"
   ]
  },
  "execution_count": 66,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "# Проверка, является ли матрица симметричной:\n",
 "issymmetric(Asym)"
]
},
{
"cell_type": "code",
"execution_count": 67,
"id": "5cbf605f",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [
    "-1.1871005162529327"
   ]
  },
  "execution_count": 67,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "# Добавление шума:\n",
 "Asym_noisy = copy(Asym)\n",
 "Asym_noisy[1,2] += 5eps()"
]
},
{
"cell_type": "code",
"execution_count": 68,
"id": "944f0573",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [
    "false"

```
     ]
    },
    "execution_count": 68,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "# Проверка, является ли матрица симметричной:\n",
   "issymmetric(Asym_noisy)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 69,
  "id": "e7001897",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "1000×1000 Symmetric{Float64, Matrix{Float64}}:\n",
      "  0.393061  -1.1871     1.52232    …  -2.68199   -1.09688    -1.75511\n",
      " -1.1871    -0.485784  -1.09796        0.134448  -1.12184     1.8987\n",
      "  1.52232   -1.09796    0.0576466     -2.16454   -0.632841    2.6184\n",
      "  2.20254    2.12601    1.37919       -0.86109    1.38753     1.14129\n",
      "  1.14361    0.670914  -0.0388096     -0.222059  -0.324271    0.174666\n",
      "  0.4213     0.237232   3.36525    …   0.574236   0.0863073   1.37996\n",
      " -1.12982   -0.138938   0.183372      -0.725398   0.580001   -0.369564\n",
      "  0.907652  -2.17058    0.748957      -1.53514   -0.627687   -0.127403\n",
      "  0.43601   -0.342293   0.25039        1.02085   -0.711883    0.583164\n",
      "  0.547735  -0.316938  -0.495361      -0.510295   0.578657    0.0241491\n",
      "  0.127273   0.215779   0.0729088  …   1.1158     2.47708     0.760371\n",
      " -0.175972   1.07572    2.89464        2.63094   -1.5264     -0.396812\n",
      "  2.18405   -0.469264   1.93446       -2.98784   -3.06913     0.64797\n",
      "  ⋮                                ⋱                          \n",
      " -1.39222    1.79202   -0.233084      -0.323808  -0.37531     1.79995\n",
      "  4.67361    2.30666   -0.197534       2.05787    0.0193012   2.52057\n",
      "  3.13532   -0.456212  -2.97602    …   2.4854    -0.266278   -1.94299\n",
      "  1.4815    -0.628309  -0.447098      -2.64624    0.458817   -1.73581\n",
      "  1.02467   -2.23186    0.065106       3.52976   -1.24011     0.257556\n",
      " -0.105639  -1.85397    1.48033        0.30017    1.08879     1.27713\n",
      " -2.016     -0.236965  -1.47008        0.523953   0.434294   -1.67341\n",
      "  1.46506    1.31339    1.66461    …  -0.881256  -2.62313    -0.105925\n",
      " -1.64201   -2.50723   -2.068          0.708567  -0.294867   -2.18745\n",
      " -2.68199    0.134448  -2.16454       -1.9919     1.28437     0.785744\n",
      " -1.09688   -1.12184   -0.632841       1.28437   -2.7682      0.164517\n",
      " -1.75511    1.8987     2.6184         0.785744   0.164517    1.64334"
     ]
    },
    "execution_count": 69,
    "metadata": {},
    "output_type": "execute_result"
```

```
   }
  ],
  "source": [
   "# Явно указываем, что матрица является симметричной:\n",
   "Asym_explicit = Symmetric(Asym_noisy)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 70,
  "id": "b44642f8",
  "metadata": {},
  "outputs": [
   {
    "name": "stderr",
    "output_type": "stream",
    "text": [
     "\u001b[32m\u001b[1m   Resolving\u001b[22m\u001b[39m package versions...\n",
     "\u001b[32m\u001b[1m  No Changes\u001b[22m\u001b[39m to `C:\\Users\\GlebB\\.julia\\environments\\v1.9\\Project.toml`\n",
     "\u001b[32m\u001b[1m  No Changes\u001b[22m\u001b[39m to `C:\\Users\\GlebB\\.julia\\environments\\v1.9\\Manifest.toml`\n"
    ]
   },
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "  373.472 ms (11 allocations: 7.99 MiB)\n"
    ]
   }
  ],
  "source": [
   "import Pkg\n",
   "Pkg.add(\"BenchmarkTools\")\n",
   "using BenchmarkTools\n",
   "# Оценка эффективности выполнения операции по нахождению\n",
   "# собственных значений симметризованной матрицы:\n",
   "@btime eigvals(Asym);"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 71,
  "id": "0821a9a4",
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "  1.158 s (14 allocations: 7.93 MiB)\n"
    ]
   }
  ],
```

"source": [
 "# Оценка эффективности выполнения операции по нахождению\n",
 "# собственных значений зашумлённой матрицы:\n",
 "@btime eigvals(Asym_noisy);"
]
},
{
"cell_type": "code",
"execution_count": 72,
"id": "91f95884",
"metadata": {},
"outputs": [
 {
  "name": "stdout",
  "output_type": "stream",
  "text": [
   " 251.271 ms (11 allocations: 7.99 MiB)\n"
  ]
 }
],
"source": [
 "# Оценка эффективности выполнения операции по нахождению\n",
 "# собственных значений зашумлённой матрицы,\n",
 "# для которой явно указано, что она симметричная:\n",
 "@btime eigvals(Asym_explicit);"
]
},
{
"cell_type": "code",
"execution_count": 73,
"id": "4c3274fb",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [
    "1000000×1000000 SymTridiagonal{Float64, Vector{Float64}}:\n",
    " 0.308917 1.67869   ·   ·   …   ·   ·   · \n",
    " 1.67869   0.62622   0.829802 ·   ·   ·   · \n",
    " ·   0.829802 1.77733   0.720879   ·   ·   · \n",
    " ·   ·   0.720879 1.6302   ·   ·   · \n",
    " ·   ·   ·   0.51027   ·   ·   · \n",
    " ·   ·   ·   ·   …   ·   ·   · \n",
    " ·   ·   ·   ·   ·   ·   · \n",
    " ·   ·   ·   ·   ·   ·   · \n",
    " ·   ·   ·   ·   ·   ·   · \n",
    " ·   ·   ·   ·   ·   ·   · \n",
    " ·   ·   ·   ·   …   ·   ·   · \n",
    " ·   ·   ·   ·   ·   ·   · \n",
    " ·   ·   ·   ·   ·   ·   · \n",
    " ⋮   ⋱   \n",
    " ·   ·   ·   ·   ·   ·   · \n",

" ·   ·   ·   ·   ·   ·   · \n",
" ·   ·   ·   ·   …   ·   ·   · \n",
" ·   ·   ·   ·   ·   ·   · \n",
" ·   ·   ·   ·   ·   ·   · \n",
" ·   ·   ·   ·   ·   ·   · \n",
" ·   ·   ·   ·   ·   ·   · \n",
" ·   ·   ·   ·   …   ·   ·   · \n",
" ·   ·   ·   ·   1.14922   ·   · \n",
" ·   ·   ·   ·   1.68936   -0.746803   · \n",
" ·   ·   ·   ·   -0.746803 -0.406215 -1.27868\n",
" ·   ·   ·   ·   ·   -1.27868   -0.900195"
]
},
"execution_count": 73,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Трёхдиагональная матрица 1000000 x 1000000:\n",
"n = 1000000;\n",
"A = SymTridiagonal(randn(n), randn(n-1))"
]
},
{
"cell_type": "code",
"execution_count": 74,
"id": "8d612666",
"metadata": {},
"outputs": [
{
"name": "stdout",
"output_type": "stream",
"text": [
" 740.392 ms (17 allocations: 183.11 MiB)\n"
]
},
{
"data": {
"text/plain": [
"6.52034195883315"
]
},
"execution_count": 74,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Оценка эффективности выполнения операции по нахождению\n",
"# собственных значений:\n",
"@btime eigmax(A)"

]
},
{
"cell_type": "code",
"execution_count": 75,
"id": "d780e74f",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"3×3 Matrix{Rational{BigInt}}:\n",
" 1//1   3//5   1//10\n",
" 7//10 7//10 9//10\n",
" 3//5   3//10 7//10"
]
},
"execution_count": 75,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Матрица с рациональными элементами:\n",
"Arational = Matrix{Rational{BigInt}}(rand(1:10, 3, 3))/10"
]
},
{
"cell_type": "code",
"execution_count": 76,
"id": "08c5fa64",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"3-element Vector{Int64}:\n",
" 1\n",
" 1\n",
" 1"
]
},
"execution_count": 76,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"# Единичный вектор:\n",
"x = fill(1, 3)"
]
},
{

"cell_type": "code",
"execution_count": 77,
"id": "28d0fc04",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [
    "3-element Vector{Rational{BigInt}}:\n",
    " 17//10\n",
    " 23//10\n",
    " 8//5"
   ]
  },
  "execution_count": 77,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "# Задаём вектор b:\n",
 "b = Arational*x"
]
},
{
"cell_type": "code",
"execution_count": 78,
"id": "7abfabd6",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [
    "3-element Vector{Rational{BigInt}}:\n",
    " 1//1\n",
    " 1//1\n",
    " 1//1"
   ]
  },
  "execution_count": 78,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "# Решение исходного уравнения получаем с помощью функции \\\n",
 "# (убеждаемся, что x - единичный вектор):\n",
 "Arational\\b"
]
},
{
"cell_type": "code",
"execution_count": 79,

```
 "id": "a8aa82a7",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "LU{Rational{BigInt}, Matrix{Rational{BigInt}}, Vector{Int64}}\n",
     "L factor:\n",
     "3×3 Matrix{Rational{BigInt}}:\n",
     " 1//1    0//1   0//1\n",
     " 7//10   1//1   0//1\n",
     " 3//5   -3//14  1//1\n",
     "U factor:\n",
     "3×3 Matrix{Rational{BigInt}}:\n",
     " 1//1  3//5     1//10\n",
     " 0//1  7//25   83//100\n",
     " 0//1  0//1   229//280"
    ]
   },
   "execution_count": 79,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# LU-разложение:\n",
  "lu(Arational)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 81,
 "id": "7e78a23e",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "119"
    ]
   },
   "execution_count": 81,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "v = [3, 5, 2, 9]\n",
  "dot_v = dot(v, v)"
 ]
},
{
 "cell_type": "code",
```

```
 "execution_count": 82,
 "id": "78e87a8c",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "4×4 Matrix{Int64}:\n",
     "  9  15   6  27\n",
     " 15  25  10  45\n",
     "  6  10   4  18\n",
     " 27  45  18  81"
    ]
   },
   "execution_count": 82,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "outer_v = v * v'"
 ]
},
{
 "cell_type": "code",
 "execution_count": 91,
 "id": "1308d126",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "2-element Vector{Int64}:\n",
     " 2\n",
     " 3"
    ]
   },
   "execution_count": 91,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "# Left - лево, Right - право\n",
  "L1 = [1 1; 1 -1]\n",
  "R1 = [2; 3]"
 ]
},
{
 "cell_type": "code",
 "execution_count": 102,
 "id": "5d0fd99e",
 "metadata": {},
```

```
 "outputs": [
  {
   "data": {
    "text/plain": [
     "2-element Vector{Float64}:\n",
     "  2.5\n",
     " -0.5"
    ]
   },
   "execution_count": 102,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "L1\\R1"
 ]
},
{
 "cell_type": "code",
 "execution_count": 93,
 "id": "a374a853",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "2-element Vector{Int64}:\n",
     " 2\n",
     " 4"
    ]
   },
   "execution_count": 93,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "L2 = [1 1; 2 2]\n",
  "R2 = [2; 4]"
 ]
},
{
 "cell_type": "code",
 "execution_count": 103,
 "id": "b35aceb6",
 "metadata": {},
 "outputs": [
  {
   "ename": "LoadError",
   "evalue": "SingularException(2)",
   "output_type": "error",
   "traceback": [
```

"SingularException(2)",
"",
"Stacktrace:",
" [1] checknonsingular",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\factorization.jl:19 [inlined]",
" [2] checknonsingular",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\factorization.jl:22 [inlined]",
" [3] #lu!#170",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:82 [inlined]",
" [4] lu!",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:80 [inlined]",
" [5] lu(A::Matrix{Int64}, pivot::RowMaximum; check::Bool)",
"   @ LinearAlgebra C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:299",
" [6] lu (repeats 2 times)",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:298 [inlined]",
" [7] \\(A::Matrix{Int64}, B::Vector{Int64})",
"   @ LinearAlgebra C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\generic.jl:1115",
" [8] top-level scope",
"   @ In[103]:1"
]
}
],
"source": [
"L2\\R2"
]
},
{
"cell_type": "code",
"execution_count": 95,
"id": "d35fa0e3",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"2-element Vector{Int64}:\n",
" 2\n",
" 5"
]
},
"execution_count": 95,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"L3 = [1 1; 2 2]\n",
"R3 = [2; 5]"
]
},
{
"cell_type": "code",
"execution_count": 104,

"id": "08921ab9",
"metadata": {},
"outputs": [
{
"ename": "LoadError",
"evalue": "SingularException(2)",
"output_type": "error",
"traceback": [
"SingularException(2)",
"",
"Stacktrace:",
" [1] checknonsingular",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\factorization.jl:19 [inlined]",
" [2] checknonsingular",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\factorization.jl:22 [inlined]",
" [3] #lu!#170",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:82 [inlined]",
" [4] lu!",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:80 [inlined]",
" [5] lu(A::Matrix{Int64}, pivot::RowMaximum; check::Bool)",
"   @ LinearAlgebra C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:299",
" [6] lu (repeats 2 times)",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:298 [inlined]",
" [7] \\(A::Matrix{Int64}, B::Vector{Int64})",
"   @ LinearAlgebra C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\generic.jl:1115",
" [8] top-level scope",
"   @ In[104]:1"
]
}
],
"source": [
"L3\\R3"
]
},
{
"cell_type": "code",
"execution_count": 97,
"id": "c5651b6c",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"3-element Vector{Int64}:\n",
" 1\n",
" 2\n",
" 3"
]
},
"execution_count": 97,
"metadata": {},
"output_type": "execute_result"
}

```
  ],
  "source": [
   "L4 = [1 1; 2 2; 3 3]\n",
   "R4 = [1; 2; 3]"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 105,
  "id": "41a77ce2",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "2-element Vector{Float64}:\n",
      " 0.4999999999999999\n",
      " 0.5"
     ]
    },
    "execution_count": 105,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "L4\\R4"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 99,
  "id": "fa13d46e",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "3-element Vector{Int64}:\n",
      " 2\n",
      " 1\n",
      " 3"
     ]
    },
    "execution_count": 99,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "L5 = [1 1; 2 1; 1 -1]\n",
   "R5 = [2; 1; 3]"
  ]
```

},
{
"cell_type": "code",
"execution_count": 106,
"id": "0e42687a",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"2-element Vector{Float64}:\n",
" 1.5000000000000004\n",
" -0.9999999999999997"
]
},
"execution_count": 106,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"L5\\R5"
]
},
{
"cell_type": "code",
"execution_count": 101,
"id": "cbac12bd",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"3-element Vector{Int64}:\n",
" 2\n",
" 1\n",
" 3"
]
},
"execution_count": 101,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"L6 = [1 1; 2 1; 3 2]\n",
"R6 = [2; 1; 3]"
]
},
{
"cell_type": "code",
"execution_count": 108,
"id": "b6f00dc0",

"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"2-element Vector{Float64}:\n",
" -0.9999999999999989\n",
" 2.9999999999999982"
]
},
"execution_count": 108,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"L6\\R6"
]
},
{
"cell_type": "code",
"execution_count": 109,
"id": "3694eef8",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"3-element Vector{Float64}:\n",
" 2.2142857142857144\n",
" 0.35714285714285704\n",
" -0.5714285714285712"
]
},
"execution_count": 109,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"L7 = [1 1 1; 1 -1 -2]\n",
"R7 = [2; 3]\n",
"L7\\R7"
]
},
{
"cell_type": "code",
"execution_count": 110,
"id": "718698e6",
"metadata": {},
"outputs": [
{
"data": {

"text/plain": [
"3-element Vector{Float64}:\n",
" -0.5\n",
" 2.5\n",
" 0.0"
]
},
"execution_count": 110,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"L8 = [1 1 1; 2 2 -3; 3 1 1]\n",
"R8 = [2; 4; 1]\n",
"L8\\R8"
]
},
{
"cell_type": "code",
"execution_count": 111,
"id": "57d5657d",
"metadata": {},
"outputs": [
{
"ename": "LoadError",
"evalue": "SingularException(2)",
"output_type": "error",
"traceback": [
"SingularException(2)",
"",
"Stacktrace:",
" [1] checknonsingular",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\factorization.jl:19 [inlined]",
" [2] checknonsingular",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\factorization.jl:22 [inlined]",
" [3] #lu!#170",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:82 [inlined]",
" [4] lu!",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:80 [inlined]",
" [5] lu(A::Matrix{Int64}, pivot::RowMaximum; check::Bool)",
"   @ LinearAlgebra C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:299",
" [6] lu (repeats 2 times)",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:298 [inlined]",
" [7] \\(A::Matrix{Int64}, B::Vector{Int64})",
"   @ LinearAlgebra C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\generic.jl:1115",
" [8] top-level scope",
"   @ In[111]:3"
]
}
],
"source": [
"L9 = [1 1 1; 1 1 2; 2 2 3]\n",

"R9 = [1; 0; 1]\n",
"L9\\R9"
]
},
{
"cell_type": "code",
"execution_count": 112,
"id": "1a529133",
"metadata": {},
"outputs": [
{
"ename": "LoadError",
"evalue": "SingularException(2)",
"output_type": "error",
"traceback": [
"SingularException(2)",
"",
"Stacktrace:",
" [1] checknonsingular",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\factorization.jl:19 [inlined]",
" [2] checknonsingular",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\factorization.jl:22 [inlined]",
" [3] #lu!#170",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:82 [inlined]",
" [4] lu!",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:80 [inlined]",
" [5] lu(A::Matrix{Int64}, pivot::RowMaximum; check::Bool)",
"   @ LinearAlgebra C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:299",
" [6] lu (repeats 2 times)",
"   @ C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\lu.jl:298 [inlined]",
" [7] \\(A::Matrix{Int64}, B::Vector{Int64})",
"   @ LinearAlgebra C:\\Users\\GlebB\\AppData\\Local\\Programs\\Julia-1.9.3\\share\\julia\\stdlib\\v1.9\\LinearAlgebra\\src\\generic.jl:1115",
" [8] top-level scope",
"   @ In[112]:3"
]
}
],
"source": [
"L10 = [1 1 1; 1 1 2; 2 2 3]\n",
"R10 = [1; 0; 0]\n",
"L10\\R10"
]
},
{
"cell_type": "code",
"execution_count": 115,
"id": "0d75f8d4",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"dia (generic function with 1 method)"

```
     ]
    },
    "execution_count": 115,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "function dia(mat)\n",
   "    simm = mat + mat'\n",
   "    razsimm = eigen(simm)\n",
   "    return inv(razsimm.vectors) * mat * razsimm.vectors\n",
   "end"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 116,
  "id": "9cc16856",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "2×2 Matrix{Float64}:\n",
      " -1.0  0.0\n",
      "  0.0  3.0"
     ]
    },
    "execution_count": 116,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "mat1 = [1 -2; -2 1]\n",
   "dia(mat1)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 117,
  "id": "4a0694a0",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "2×2 Matrix{Float64}:\n",
      " -0.236068     3.46945e-16\n",
      "  4.44089e-16  4.23607"
     ]
    },
```

```
   "execution_count": 117,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "mat2 = [1 -2; -2 3]\n",
  "dia(mat2)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 118,
 "id": "573a620e",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "3×3 Matrix{Float64}:\n",
     " -2.14134       3.55271e-15  -1.9984e-15\n",
     "  3.38618e-15   0.515138      1.11022e-16\n",
     " -6.66134e-16  -4.44089e-16   3.6262"
    ]
   },
   "execution_count": 118,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "mat3 = [1 -2 0; -2 1 2; 0 2 0]\n",
  "dia(mat3)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 119,
 "id": "a6b843c4",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "2×2 Matrix{Int64}:\n",
     "  29525  -29524\n",
     " -29524   29525"
    ]
   },
   "execution_count": 119,
   "metadata": {},
   "output_type": "execute_result"
  }
```

],
"source": [
"([1 -2; -2 1])^10"
]
},
{
"cell_type": "code",
"execution_count": 120,
"id": "075a6f44",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"2×2 Matrix{Float64}:\n",
" 2.1889   -0.45685\n",
" -0.45685   2.1889"
]
},
"execution_count": 120,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"sqrt([5 -2; -2 5])"
]
},
{
"cell_type": "code",
"execution_count": 121,
"id": "b620263a",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"2×2 Symmetric{ComplexF64, Matrix{ComplexF64}}:\n",
" 0.971125+0.433013im -0.471125+0.433013im\n",
" -0.471125+0.433013im   0.971125+0.433013im"
]
},
"execution_count": 121,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"([1 -2; -2 1])^(1/3)"
]
},
{
"cell_type": "code",

"execution_count": 122,
"id": "4c2c65fc",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"2×2 Matrix{ComplexF64}:\n",
" 0.568864+0.351578im 0.920442-0.217287im\n",
" 0.920442-0.217287im   1.48931+0.134291im"
]
},
"execution_count": 122,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"sqrt([1 2; 2 3])"
]
},
{
"cell_type": "code",
"execution_count": 123,
"id": "4595a555",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"5-element Vector{Float64}:\n",
" -128.49322764802145\n",
" -55.887784553056875\n",
"   42.7521672793189\n",
"   87.16111477514521\n",
" 542.4677301466143"
]
},
"execution_count": 123,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"A = [140 97 74 168 131; 97 106 89 131 36; 74 89 152 144 71; 168 131 144 54 142; 131 36 71 142 36]\n",
"val = eigvals(A)"
]
},
{
"cell_type": "code",
"execution_count": 124,
"id": "c148c729",
"metadata": {},

"outputs": [
 {
  "name": "stdout",
  "output_type": "stream",
  "text": [
   " 3.700 μs (10 allocations: 2.59 KiB)\n"
  ]
 },
 {
  "data": {
   "text/plain": [
    "5-element Vector{Float64}:\n",
    " -128.49322764802145\n",
    " -55.887784553056875\n",
    "   42.7521672793189\n",
    "   87.16111477514521\n",
    " 542.4677301466143"
   ]
  },
  "execution_count": 124,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "@btime eigvals(A)"
]
},
{
"cell_type": "code",
"execution_count": 126,
"id": "a12f2a8e",
"metadata": {},
"outputs": [
 {
  "name": "stdout",
  "output_type": "stream",
  "text": [
   " 241.646 ns (4 allocations: 64 bytes)\n"
  ]
 },
 {
  "data": {
   "text/plain": [
    "4×4 Matrix{Float64}:\n",
    " -128.493   0.0   0.0   0.0\n",
    "   0.0   -55.8878   0.0   0.0\n",
    "   0.0   0.0   42.7522   0.0\n",
    "   0.0   0.0   0.0   87.1611"
   ]
  },
  "execution_count": 126,
  "metadata": {},

```
   "output_type": "execute_result"
  }
 ],
 "source": [
  "N = zeros(4, 4)\n",
  "@btime for i in 1:1:4\n",
  "    N[i, i] = val[i]\n",
  "end\n",
  "N"
 ]
},
{
 "cell_type": "code",
 "execution_count": 127,
 "id": "54da7dff",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "  147.407 ns (1 allocation: 256 bytes)\n"
   ]
  },
  {
   "data": {
    "text/plain": [
     "5×5 Matrix{Float64}:\n",
     " 1.0         0.0         0.0        0.0        0.0\n",
     " 0.779762   1.0         0.0        0.0        0.0\n",
     " 0.440476  -0.47314    1.0        0.0        0.0\n",
     " 0.833333   0.183929  -0.556312  1.0        0.0\n",
     " 0.577381  -0.459012  -0.189658  0.897068  1.0"
    ]
   },
   "execution_count": 127,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "Alu = lu(A)\n",
  "@btime Alu.L"
 ]
},
{
 "cell_type": "code",
 "execution_count": 128,
 "id": "a685934b",
 "metadata": {},
 "outputs": [
  {
   "data": {
```

```
    "text/plain": [
     "matr (generic function with 1 method)"
    ]
   },
   "execution_count": 128,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "function matr(mat, s)\n",
  "    ans = \"\"\n",
  "    P = [1 0; 0 1]\n",
  "    K = rand(0:100, s)\n",
  "    H = P - mat\n",
  "    T = H\\K\n",
  "    for i in 1:1:s\n",
  "        if T[i] < 0\n",
  "            ans = \"no\"\n",
  "            break\n",
  "        else\n",
  "            ans = \"yes\"\n",
  "        end\n",
  "    end\n",
  "    return ans\n",
  "end"
 ]
},
{
 "cell_type": "code",
 "execution_count": 129,
 "id": "85e80ed8",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "\"no\""
    ]
   },
   "execution_count": 129,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "mat1 = [1 2; 3 4]\n",
  "matr(mat1, 2)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 130,
```

"id": "8624dd05",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"\"no\""
]
},
"execution_count": 130,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"mat2 = ([1 2; 3 4])*(1/2)\n",
"matr(mat2, 2)"
]
},
{
"cell_type": "code",
"execution_count": 131,
"id": "217cb23e",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"\"yes\""
]
},
"execution_count": 131,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"mat3 = ([1 2; 3 4])*(1/10)\n",
"matr(mat3, 2)"
]
},
{
"cell_type": "code",
"execution_count": null,
"id": "89a4d1af",
"metadata": {},
"outputs": [],
"source": []
}
],
"metadata": {
"kernelspec": {
"display_name": "Julia 1.9.3",

```
   "language": "julia",
   "name": "julia-1.9"
  },
  "language_info": {
   "file_extension": ".jl",
   "mimetype": "application/julia",
   "name": "julia",
   "version": "1.9.3"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 5
}
```