Лабораторная работа № 4

Ли Тимофей Александрович, НФИбд-01-18

Цель работы

Изучить возможности специализированных пакетов Julia для выполнения и оценки эффективности операций над объектами линейной алгебры.

Ход работы. Примеры

[1]:
```
a=rand(1:20,(4,3))
```

[1]:
```
4×3 Matrix{Int64}:
  8  10  16
  5  17   9
 19  14   2
 15   4   3
```

[2]:
```
println(sum(a))
println(sum(a,dims=1))
println(sum(a,dims=2))
```

```
122
[47 45 30]
[34; 31; 35; 22]
```

[3]:
```
println(prod(a))
println(prod(a,dims=1))
println(prod(a,dims=2))
```

```
93768192000
[11400 9520 864]
[1280; 765; 532; 180]
```

[4]:
```
import Pkg
Pkg.add("Statistics")
```

```
    Updating registry at `C:\Users\Xiaomi\.julia\registries\General`
    Updating git-repo `https://github.com/JuliaRegistries/General.git`
   Resolving package versions...
    Updating `C:\Users\Xiaomi\.julia\environments\v1.6\Project.toml`
  [10745b16] + Statistics
  No Changes to `C:\Users\Xiaomi\.julia\environments\v1.6\Manifest.toml`
```

[5]:
```
using Statistics
println(mean(a))
println(mean(a,dims=1))
println(mean(a,dims=2))
```

```
10.166666666666666
[11.75 11.25 7.5]
[11.333333333333334; 10.333333333333334; 11.666666666666666; 7.333333333333333]
```

Рис. 1: примеры1

Ход работы. Примеры

```
[6]: Pkg.add("LinearAlgebra")
```

```
  Resolving package versions...
   Updating `C:\Users\Xiaomi\.julia\environments
 [37e2e46d] + LinearAlgebra
 No Changes to `C:\Users\Xiaomi\.julia\environme
```

```
[7]: using LinearAlgebra
     b=rand(1:20,(4,4))
```

```
[7]: 4×4 Matrix{Int64}:
      7  20   6   4
     15  11  11  15
      8  17   5  11
      2  17  15   5
```

```
[8]: transpose(b)
```

```
[8]: 4×4 transpose(::Matrix{Int64}) with eltype Int64:
      7  15   8   2
     20  11  17  17
      6  11   5  15
      4  15  11   5
```

```
[9]: tr(b)
```

```
[9]: 28
```

```
[10]: diag(b)
```

```
[10]: 4-element Vector{Int64}:
       7
      11
       5
       5
```

```
[11]: rank(b)
```

```
[11]: 4
```

```
[12]: inv(b)
```

```
[12]: 4×4 Matrix{Float64}:
       0.126543    0.0895062  -0.135802   -0.0709877
       0.0384615  -0.0384615   0.0384615   0.0
      -0.0117521   0.0395299  -0.0811966   0.0694444
      -0.14613    -0.023623    0.167142    0.0200617
```

```
[13]: det(b)
```

```
[13]: 16848.0
```

```
[14]: pinv(b)
```

```
[14]: 4×4 Matrix{Float64}:
       0.126543    0.0895062  -0.135802   -0.0709877
       0.0384615  -0.0384615   0.0384615   2.14255e-17
      -0.0117521   0.0395299  -0.0811966   0.0694444
      -0.14613    -0.023623    0.167142    0.0200617
```

Рис. 2: примеры2

Ход работы. Примеры

```
[15]: x=[2,4,-5]
println(norm(x))
p=1
println(norm(x,p))
```

```
6.708203932499369
11.0
```

```
[16]: y=[1,-1,3]
println(norm(x-y))
```

```
9.486832980505138
```

```
[17]: println(sum((x-y).^2))
```

```
90
```

```
[19]: acos((transpose(x)*y)/(norm(x)*norm(y)))
```

```
[19]: 2.4404307889469252
```

```
[20]: d=[5 -4 2; -1 2 3; -2 1 0]
println(opnorm(d))
println(opnorm(d,p))
```

```
7.147682841795258
8.0
```

```
[21]: rot180(d)
```

```
[21]: 3×3 Matrix{Int64}:
 0   1  -2
 3   2  -1
 2  -4   5
```

```
[22]: reverse(d,dims=1)
```

```
[22]: 3×3 Matrix{Int64}:
 -2   1  0
 -1   2  3
  5  -4  2
```

```
[23]: reverse(d,dims=2)
```

```
[23]: 3×3 Matrix{Int64}:
 2  -4   5
 3   2  -1
 0   1  -2
```

Рис. 3: примеры3

Ход работы. Примеры

```
[24]: a=rand(1:10,(2,3))
      b=rand(1:10,(3,4))
      a*b
```

```
[24]: 2×4 Matrix{Int64}:
       166  126  74  128
       121   95  59   90
```

```
[25]: Matrix{Int}(I,3,3)
```

```
[25]: 3×3 Matrix{Int64}:
       1  0  0
       0  1  0
       0  0  1
```

```
[26]: dot(x,y)
```

```
[26]: -17
```

```
[27]: x'y
```

```
[27]: -17
```

Рис. 4: примеры4

Ход работы. Примеры

```
[28]: a=rand(3,3)
      x=fill(1.0,3)
      b=a*x

[28]: 3-element Vector{Float64}:
       0.9660839601808051
       2.296387657479849
       0.43505222993012227

[29]: a\b

[29]: 3-element Vector{Float64}:
       0.9999999999999992
       1.0000000000000004
       1.0000000000000002

[30]: alu=lu(a)

[30]: LU{Float64, Matrix{Float64}}
      L factor:
      3×3 Matrix{Float64}:
       1.0         0.0       0.0
       0.113423    1.0       0.0
       0.0141559  -0.602373  1.0
      U factor:
      3×3 Matrix{Float64}:
       0.706353   0.695527   0.894508
       0.0       -0.0716548  0.246243
       0.0        0.0        1.03874

[31]: alu.p

[31]: 3-element Vector{Int64}:
       2
       3
       1

[32]: alu.P

[32]: 3×3 Matrix{Float64}:
       0.0  1.0  0.0
       0.0  0.0  1.0
       1.0  0.0  0.0

[33]: alu.L

[33]: 3×3 Matrix{Float64}:
       1.0         0.0       0.0
       0.113423    1.0       0.0
       0.0141559  -0.602373  1.0

[34]: alu.U

[34]: 3×3 Matrix{Float64}:
       0.706353   0.695527   0.894508
       0.0       -0.0716548  0.246243
       0.0        0.0        1.03874

[35]: a\b

[35]: 3-element Vector{Float64}:
       0.9999999999999992
       1.0000000000000004
       1.0000000000000002

[36]: alu\b

[36]: 3-element Vector{Float64}:
       0.9999999999999992
       1.0000000000000004
       1.0000000000000002

[37]: det(a)

[37]: -0.05257453494420739

[38]: det(alu)

[38]: -0.05257453494420739

[39]: aqr=qr(a)

[39]: LinearAlgebra.QRCompactWY{Float64, Matrix{Float64}}
      Q factor:
      3×3 LinearAlgebra.QRCompactWYQ{Float64, Matrix{Float64}}:
       -0.0140644   0.519318    0.854465
       -0.993531    0.0890502  -0.0704754
       -0.112689   -0.849929    0.514706
      R factor:
      3×3 Matrix{Float64}:
       -0.710952  -0.692588   -0.940604
        0.0        0.0833167   0.253119
        0.0        0.0         0.88757

[40]: aqr.Q

[40]: 3×3 LinearAlgebra.QRCompactWYQ{Float64, Matrix{Float64}}:
       -0.0140644   0.519318    0.854465
       -0.993531    0.0890502  -0.0704754
       -0.112689   -0.849929    0.514706

[41]: aqr.R

[41]: 3×3 Matrix{Float64}:
       -0.710952  -0.692588   -0.940604
        0.0        0.0833167   0.253119
        0.0        0.0         0.88757

[42]: aqr.Q'*aqr.R

[42]: 3×3 Matrix{Float64}:
        0.00999909  -0.0730369  -0.338272
       -0.369211    -0.352254   -1.2203
       -0.607484    -0.597664   -0.364714
```

Рис. 5: примеры5

Ход работы. Примеры

```
[43]: asym=a+a'

[43]: 3x3 Matrix{Float64}:
 0.0199982  0.759362  0.983193
 0.759362   1.39105   0.901742
 0.983193   0.901742  0.695402

[44]: asymeig=eigen(asym)

[44]: Eigen{Float64, Float64, Matrix{Float64}, Vector{Float64}}
values:
3-element Vector{Float64}:
 -0.6881234482197165
  0.22392954711208235
  2.5706479163830305
vectors:
3x3 Matrix{Float64}:
 -0.833222    0.350454  -0.427695
  0.0662272  -0.704675  -0.706433
  0.548958    0.616941  -0.563941

[45]: asymeig.values

[45]: 3-element Vector{Float64}:
 -0.6881234482197165
  0.22392954711208235
  2.5706479163830305

[46]: asymeig.vectors

[46]: 3x3 Matrix{Float64}:
 -0.833222    0.350454  -0.427695
  0.0662272  -0.704675  -0.706433
  0.548958    0.616941  -0.563941
```

```
[47]: inv(asymeig)*asym

[47]: 3x3 Matrix{Float64}:
   1.0          3.9968e-15   -7.77156e-15
  -3.33067e-15  1.0           9.32587e-15
   1.77636e-15  4.21885e-15   1.0

[51]: n=1000
a=randn(n,n)
asym=a+a'
issymmetric(asym)

[51]: true

[53]: asym_noisy=copy(asym)
asym_noisy[1,2]+=5eps()
issymmetric(asym_noisy)

[53]: false

[54]: asym_explicit=Symmetric(asym_noisy)

[54]: 1000×1000 Symmetric{Float64, Matrix{Float64}}:
 -3.97621   -1.80498     1.83508    …  -2.40325      2.21321     0.13888
 -1.80498   -0.228763    1.75753       -2.98848     -0.401623   -0.765896
  1.83508    1.75753    -1.62725        0.346819     1.43516     0.25021
  0.679386   0.114696    0.800428      -0.638286     1.74288     1.76747
 -1.35989   -0.844844   -0.810548       0.468689     1.92526     0.25471
 -1.00175    1.37295     0.506201   …   1.82724      0.647835    0.369493
  0.954013  -3.16234     0.209664      -0.00514881   0.844065   -0.980934
 -0.563676  -2.54702     2.81653       -1.21339     -0.201298    0.754921
 -0.484848  -2.79842     0.407308       1.42897      0.519411    1.15908
```

Рис. 6: примеры5

Ход работы. Примеры

```
[55]: Pkg.add("BenchmarkTools")

   Resolving package versions...
   Installed BenchmarkTools — v1.2.0
    Updating `C:\Users\Xiaomi\.julia\environments\v1.6\Project.toml`
  [6e4b80f9] + BenchmarkTools v1.2.0
    Updating `C:\Users\Xiaomi\.julia\environments\v1.6\Manifest.toml`
  [6e4b80f9] + BenchmarkTools v1.2.0
  [9abbd945] + Profile
Precompiling project...
  ✓ BenchmarkTools
  1 dependency successfully precompiled in 2 seconds (21 already precompiled)

[56]: using BenchmarkTools
      @btime eigvals(asym);

  335.109 ms (11 allocations: 7.99 MiB)

[57]: @btime eigvals(asym_noisy);

  1.405 s (13 allocations: 7.92 MiB)

[58]: @btime eigvals(asym_explicit);

  332.292 ms (11 allocations: 7.99 MiB)

[59]: n = 1000000;
      A = SymTridiagonal(randn(n), randn(n-1))

[59]: 1000000×1000000 SymTridiagonal{Float64, Vector{Float64}}:
 -0.699461  -2.13919     ·       …    ·          ·          ·
 -2.13919   -0.62774   -0.287085       ·          ·          ·
   ·        -0.287085   0.149479       ·          ·          ·
   ·          ·         0.720997       ·          ·          ·

[60]: @btime eigmax(A)

  552.836 ms (17 allocations: 183.11 MiB)

[60]: 6.673829087889594

[61]: B=Matrix(A)

OutOfMemoryError()

Stacktrace:
 [1] Array
   @ .\boot.jl:450 [inlined]
 [2] Array
   @ .\boot.jl:458 [inlined]
 [3] zeros
   @ .\array.jl:503 [inlined]
 [4] zeros
   @ .\array.jl:499 [inlined]
 [5] Matrix{Float64}(M::SymTridiagonal{Float64, Vector{Float64}})
   @ LinearAlgebra C:\buildbot\worker\package_win64\build\usr\share\julia\stdlib\v1.6\LinearAlg
 [6] (Matrix{T} where T)(M::SymTridiagonal{Float64, Vector{Float64}})
   @ LinearAlgebra C:\buildbot\worker\package_win64\build\usr\share\julia\stdlib\v1.6\LinearAlg
 [7] top-level scope
   @ In[61]:1
 [8] eval
   @ .\boot.jl:360 [inlined]
 [9] include_string(mapexpr::typeof(REPL.softscope), mod::Module, code::String, filename::Strin
   @ Base .\loading.jl:1116
```

Рис. 7: примеры5

Ход работы. Примеры

```
[62]: Arational = Matrix{Rational{BigInt}}(rand(1:10, 3, 3))/10

[62]: 3×3 Matrix{Rational{BigInt}}:
       3//5  3//5  1//2
       2//5  1//1  3//10
       3//5  3//5  2//5

[63]: x=fill(1,3)
      b=Arational*x

[63]: 3-element Vector{Rational{BigInt}}:
       17//10
       17//10
        8//5

[64]: Arational\b

[64]: 3-element Vector{Rational{BigInt}}:
       1//1
       1//1
       1//1
```

```
[65]: lu(Arational)

[65]: LU{Rational{BigInt}, Matrix{Rational{BigInt}}}
      L factor:
      3×3 Matrix{Rational{BigInt}}:
       1//1  0//1  0//1
       2//3  1//1  0//1
       1//1  0//1  1//1
      U factor:
      3×3 Matrix{Rational{BigInt}}:
       3//5  3//5   1//2
       0//1  3//5  -1//30
       0//1  0//1  -1//10
```

Рис. 8: примеры6

Ход работы. 1

```
[66]: v=[1; 2; 3]
      dot_v=v'v

[66]: 14

[68]: outer_v=v*v'

[68]: 3×3 Matrix{Int64}:
       1  2  3
       2  4  6
       3  6  9
```

Рис. 9: номер1

Ход работы. 2

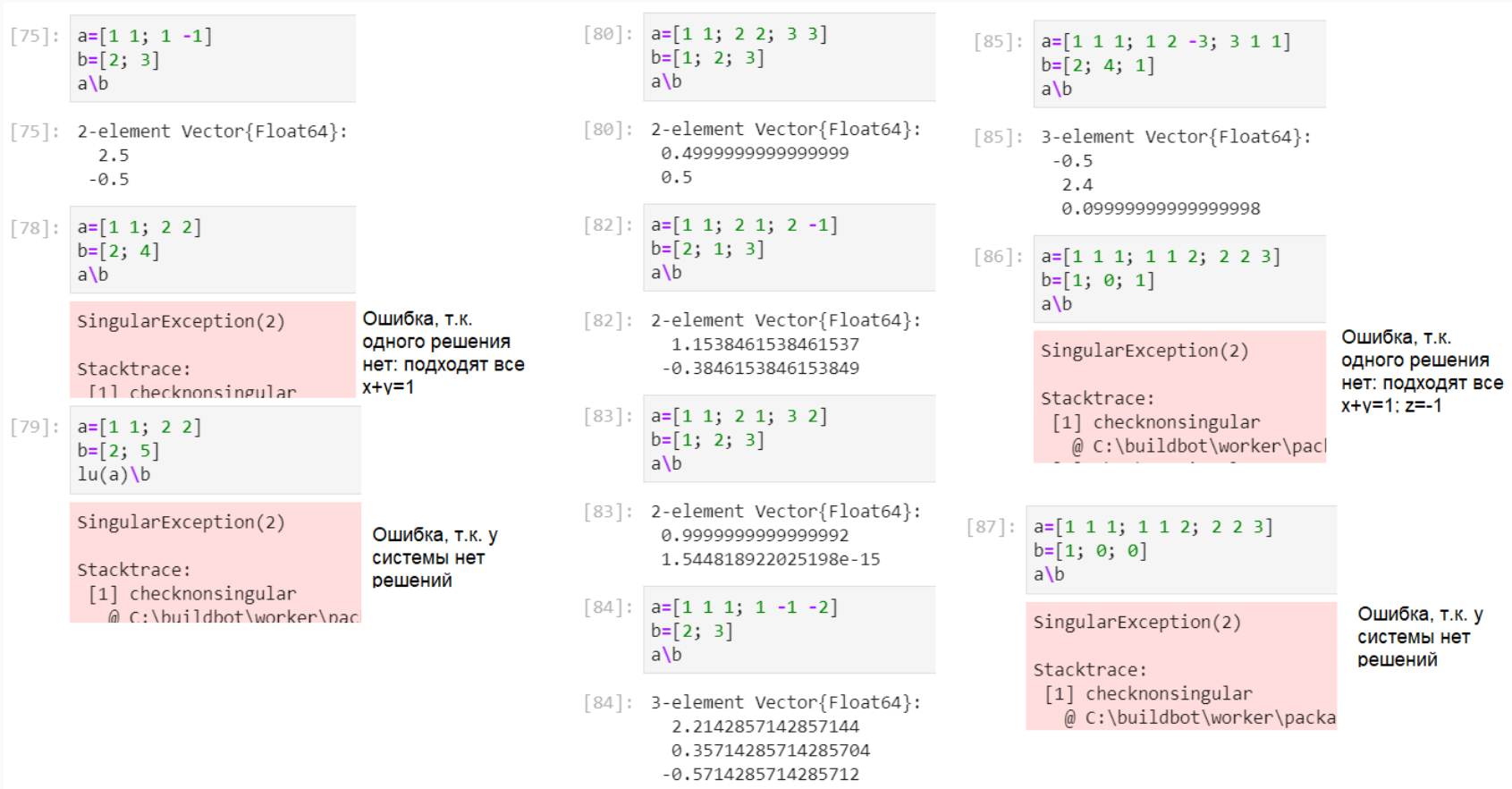

Рис. 10: номер2

Ход работы. 3

```
[128]: a=[1 -2; -2 1]
       Matrix(Diagonal(eigen(a).values))

[128]: 2×2 Matrix{Float64}:
        -1.0  0.0
         0.0  3.0

[129]: a=[1 -2; -2 3]
       Matrix(Diagonal(eigen(a).values))

[129]: 2×2 Matrix{Float64}:
        -0.236068  0.0
         0.0       4.23607
```

```
[130]: a=[1 -2 0; -2 1 2; 0 2 0]
       Matrix(Diagonal(eigen(a).values))

[130]: 3×3 Matrix{Float64}:
        -2.14134  0.0       0.0
         0.0      0.515138  0.0
         0.0      0.0       3.6262

[131]: a=[1 -2; -2 1]
       a^10

[131]: 2×2 Matrix{Int64}:
         29525  -29524
        -29524   29525
```

```
[133]: a=[5 -2; -2 5]
       a^(1/2)

[133]: 2×2 Symmetric{Float64, Matrix{Float64}}:
         2.1889   -0.45685
        -0.45685   2.1889

[134]: a=[1 -2; -2 1]
       a^(1/3)

[134]: 2×2 Symmetric{ComplexF64, Matrix{ComplexF64}}:
          0.971125+0.433013im  -0.471125+0.433013im
         -0.471125+0.433013im   0.971125+0.433013im

[135]: a=[1 2; 2 3]
       a^(1/2)

[135]: 2×2 Symmetric{ComplexF64, Matrix{ComplexF64}}:
        0.568864+0.351578im  0.920442-0.217287im
        0.920442-0.217287im  1.48931+0.134291im
```

Рис. 11: номер3

Ход работы. 3

```
[140]: 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]
       println(eigvals(a))
       Matrix(Diagonal(eigvals(a)))
```

```
[-128.49322764802145, -55.887784553056875, 42.7521672793189, 87.16111477514521, 542.4677301466143]
```

```
[140]: 5×5 Matrix{Float64}:
 -128.493    0.0      0.0      0.0       0.0
    0.0    -55.8878   0.0      0.0       0.0
    0.0      0.0     42.7522   0.0       0.0
    0.0      0.0      0.0     87.1611    0.0
    0.0      0.0      0.0      0.0     542.468
```

```
[141]: lu(a).L
```

```
[141]: 5×5 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
```

```
[144]: println(@btime eigvals(a))
       println(@btime Matrix(Diagonal(eigvals(a))))
       println(@btime lu(a).L)
```

```
  3.538 μs (10 allocations: 2.80 KiB)
[-128.49322764802145, -55.887784553056875, 42.7521672793189, 87.16111477514521, 542.4677301466143]
  4.229 μs (12 allocations: 3.09 KiB)
[-128.49322764802145 0.0 0.0 0.0 0.0; 0.0 -55.887784553056875 0.0 0.0 0.0; 0.0 0.0 42.7521672793189 0.0 0.0; 0.0 0.0 0.0 87.16111477514521 0.0; 0.0 0.0 0.0 0.0 542.4677301466143]
  952.941 ns (4 allocations: 736 bytes)
[1.0 0.0 0.0 0.0 0.0; 0.7797619047619048 1.0 0.0 0.0 0.0; 0.44047619047619047 -0.47313956627373355 1.0 0.0 0.0; 0.8333333333333333 0.18392873211554023 -0.5563115375892516 1.0 0.0; 0.5773809523809523 -0.4590119679654459 -0.1896576444121198 0.897067538972598 1.0]
```

Рис. 12: номер3

Ход работы. 4

```
[157]: a=[1 2; 3 4]
       y=rand(0:1000000, 2)
       e=Matrix{Int}(I,2,2)
       y\(e-a)

[157]: 1×2 transpose(::Vector{Float64}) with eltype Float64:
        -1.47733e-6  -3.14816e-6   непродуктивна, т.к. значения <0

[158]: a=[1/2 1; 3/2 2]
       y=rand(0:1000000, 2)
       e=Matrix{Int}(I,2,2)
       y\(e-a)

[158]: 1×2 transpose(::Vector{Float64}) with eltype Float64:
        -5.63125e-7  -1.52842e-6   непродуктивна, т.к. значения <0

[159]: a=[1/10 2/10; 3/10 4/10]
       y=rand(0:1000000, 2)
       e=Matrix{Int}(I,2,2)
       y\(e-a)

[159]: 1×2 transpose(::Vector{Float64}) with eltype Float64:
        2.11053e-7  6.89444e-7   продуктивная

[160]: a=[1 2; 3 1]
       e=Matrix{Int}(I,2,2)
       inv(e-a)

[160]: 2×2 Matrix{Float64}:
        -0.0  -0.333333          непродуктивна, т.к. значения <0
        -0.5   0.0

[161]: a=[1/2 2/2; 3/2 1/2]
       e=Matrix{Int}(I,2,2)
       inv(e-a)                  непродуктивна, т.к. значения <0

[161]: 2×2 Matrix{Float64}:
        -0.4  -0.8
        -1.2  -0.4

[162]: a=[1/10 2/10; 3/10 1/10]
       e=Matrix{Int}(I,2,2)
       inv(e-a)

[162]: 2×2 Matrix{Float64}:
        1.2  0.266667
        0.4  1.2          продуктивная

[165]: a=[1 2; 3 1]
       eigvals(a)

[165]: 2-element Vector{Float64}:
        -1.4494897427831779   непродуктивна,
         3.4494897427831783   т.к. значения по
                              модулю >1

[166]: a=[1/2 2/2; 3/2 1/2]
       eigvals(a)

[166]: 2-element Vector{Float64}:
        -0.7247448713915892   непродуктивна,
         1.724744871391589    т.к. значения по
                              модулю >1

[167]: a=[1/10 2/10; 3/10 1/10]
       eigvals(a)

[167]: 2-element Vector{Float64}:
        -0.14494897427831785
         0.34494897427831783  продуктивная

[168]: a=[0.1 0.2 0.3; 0 0.1 0.2; 0 0.1 0.3]
       eigvals(a)

[168]: 3-element Vector{Float64}:
        0.02679491924311228
        0.1
        0.37320508075688774  продуктивная
```

Рис. 13: номер4

Выводы

Изучил возможности специализированных пакетов Julia для выполнения и оценки эффективности операций над объектами линейной алгебры.