Лабораторная работа №4

Компьютерный практикум по статистическому анализу данных

Канева Екатерина, НФИбд-02-22

25 октября 2025

Российский университет дружбы народов, Москва, Россия

Информация

Докладчик

- Канева Екатерина Павловна
- студент группы НФИбд-02-22
- Российский университет дружбы народов
- 1132222004@rudn.ru
- https://nevseros.github.io/ru/

Вводная часть

Цель работы

Основной целью работы является изучение возможностей специализированных пакетов Julia для выполнения и оценки эффективности операций над объектами линейной алгебры.

Задания

- Используя Jupyter Lab, повторить примеры.
- Выполнить задания для самостоятельной работы.

Выполнение работы

Примеры

Примеры с поэлементными операциями

Выполнила примеры с поэлементными операциями:

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

[67]: 4×3 Matrix{Int64}:
       16  18  14
       15   1   7
       10  11   7
        2  11  20

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

[69]: (132, [43 41 48], [48; 23; 28; 33;;])

[71]: prod(a), prod(a,dims=1), prod(a,dims=2)

[71]: (143434368000, [4800 2178 13720], [4032; 105; 770; 440;;])

[13]: import Pkg
      Pkg.add("Statistics")
      using Statistics

        Updating registry at `C:\Users\ekaneva\.julia\registries\General.toml`
       Resolving package versions...
        Updating `C:\Users\ekaneva\.julia\environments\v1.11\Project.toml`
      [10745b16] + Statistics v1.11.1
       No Changes to `C:\Users\ekaneva\.julia\environments\v1.11\Manifest.toml`

[73]: mean(a), mean(a,dims=1), mean(a,dims=2)

[73]: (11.0, [10.75 10.25 12.0], [16.0; 7.666666666666667; 9.333333333333334; 11.0;;])
```

Рис. 1: Примеры с поэлементными операциями.

Примеры из второго раздела

Выполнила примеры с транспонированием, следом, рангом, определителем и инверсией матрицы:

```
[17]: import Pkg
      Pkg.add("LinearAlgebra")
      using LinearAlgebra

         Resolving package versions...
          Updating `C:\Users\ekaneva\.julia\environments\v1.11\Project.toml`
        [37e2e46d] + LinearAlgebra v1.11.0
        No Changes to `C:\Users\ekaneva\.julia\environments\v1.11\Manifest.toml`

[21]: b = rand(1:20,(4,4))

[21]: 4×4 Matrix{Int64}:
        7  11  11  9
        6  13   5  5
       11  11  16  9
        3   9  17  2

[23]: transpose(b)

[23]: 4×4 transpose(::Matrix{Int64}) with eltype Int64:
        7   6  11   3
       11  13  11   9
       11   5  16  17
        9   5   9   2

[25]: tr(b)

[25]: 38

[29]: print(diag(b))

      [7, 13, 16, 2]

[31]: rank(b)

[31]: 4

[33]: inv(b)

[33]: 4×4 Matrix{Float64}:
       -0.262664     0.0728935   0.23919     -0.0766
        0.00148258   0.113417   -0.0719051    0.033358
        0.010131    -0.0583148   0.00864838   0.06128
        0.301211    -0.124043   -0.108723    -0.0560909

[35]: det(b)

[35]: 4046.9999999999995

[75]: pinv(a)

[75]: 3×4 Matrix{Float64}:
        0.00997362   0.0548678   0.00750279  -0.0288112
        0.0489731   -0.0755042   0.0392155   -0.0215801
       -0.0253709    0.0358694  -0.0260771    0.0643323
```

Рис. 2: Примеры из второго раздела.

Выполнила примеры с вычислением нормы векторов и матриц, поворотами, вращениями:

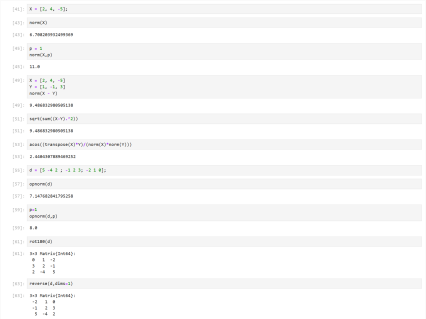


Рис. 3: Примеры из третьего раздела.

Примеры из четвёртого раздела

Выполнила примеры с матричным умножением, единичной матрицей, скалярным произведением:

```
[77]: A = rand(1:10,(2,3))
      B = rand(1:10,(3,4));

[79]: A*B

[79]: 2×4 Matrix{Int64}:
       96  154  125  111
       96  107  122  100

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

[81]: 3×3 Matrix{Int64}:
       1  0  0
       0  1  0
       0  0  1

[83]: X = [2, 4, -5]
      Y = [1,-1,3]
      dot(X,Y)

[83]: -17

[85]: X'Y

[85]: -17
```

Рис. 4: Примеры из четвёртого раздела.

Примеры из пятого раздела

Выполнила примеры с факторизацией, специальными матричными структурами:

```
[105]: det(A), det(Alu)

[105]: (0.18855350162195214, 0.18855350162195214)

[107]: Aqr = qr(A)

[107]: LinearAlgebra.QRCompactWY{Float64, Matrix{Float64}, Matrix{Float64}}
       Q factor: 3×3 LinearAlgebra.QRCompactWYQ{Float64, Matrix{Float64}, Matrix{Float64}}
       R factor:
       3×3 Matrix{Float64}:
        -0.775726  -0.445134  -0.875191
         0.0       -0.297776  -0.283805
         0.0        0.0        0.816274

[109]: Aqr.Q

[109]: 3×3 LinearAlgebra.QRCompactWYQ{Float64, Matrix{Float64}, Matrix{Float64}}

[111]: Asym = A + A'

[111]: 3×3 Matrix{Float64}:
        1.40557   0.603775  0.767926
        0.603775  0.890824  0.503309
        0.767926  0.503309  1.82323

[113]: AsymEig = eigen(Asym)

[113]: Eigen{Float64, Float64, Matrix{Float64}, Vector{Float64}}
       values:
       3-element Vector{Float64}:
        0.4918380908177505
        0.8956926855712033
        2.7320978776611917
       vectors:
       3×3 Matrix{Float64}:
         0.554209    -0.589624  -0.587534
        -0.832363    -0.396812  -0.386927
        -0.00499892   0.70548   -0.710698

[115]: inv(AsymEig)*Asym

[115]: 3×3 Matrix{Float64}:
         1.0           1.38778e-16   3.33067e-16
        -2.22045e-16   1.0          -1.66533e-16
         9.99201e-16   5.55112e-16   1.0
```

Рис. 5: Примеры из пятого раздела.

Примеры из шестого раздела

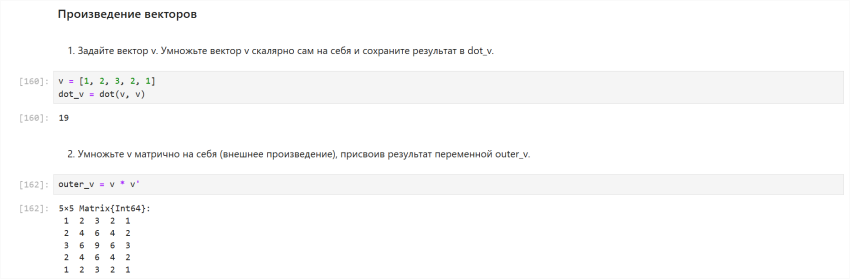
Выполнила примеры по общей линейной алгебре:

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

[132]: 3×3 Matrix{Rational{BigInt}}:
        7//10  7//10  2//5
        2//5   7//10  2//5
        3//5   3//5   9//10

[134]: x = fill(1, 3)

[134]: 3-element Vector{Int64}:
        1
        1
        1

[136]: b = Arational*x

[136]: 3-element Vector{Rational{BigInt}}:
        9//5
        3//2
        21//10

[138]: Arational\b

[138]: 3-element Vector{Rational{BigInt}}:
        1
        1
        1

[140]: lu(Arational)

[140]: LU{Rational{BigInt}, Matrix{Rational{BigInt}}, Vector{Int64}}
       L factor:
       3×3 Matrix{Rational{BigInt}}:
        1    0  0
        6//7  1  0
        6//7  0  1
       U factor:
       3×3 Matrix{Rational{BigInt}}:
        7//10  7//10   2//5
        0      3//10   6//35
        0      0      39//70
```

Рис. 6: Примеры из шестого раздела.

Задания для самостоятельного выполнения

Выполнила первое и второе задания для самостоятельной работы:



Произведение векторов

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

```
[160]: v = [1, 2, 3, 2, 1]
       dot_v = dot(v, v)
```

```
[160]: 19
```

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

```
[162]: outer_v = v * v'
```

```
[162]: 5×5 Matrix{Int64}:
        1  2  3  2  1
        2  4  6  4  2
        3  6  9  6  3
        2  4  6  4  2
        1  2  3  2  1
```

Рис. 7: Раздел 1, задания 1 и 2.

Выполнила первое задание для самостоятельной работы:

1. Решить СЛАУ с двумя неизвестными.

[164]:
```
A = [1 1; 1 -1]
b = [2, 3]
x = A \ b
```

[164]:
```
2-element Vector{Float64}:
  2.5
 -0.5
```

[174]:
```
A = [1 1; 2 2]
b = [2, 4]
try
    x = A \ b
catch
    print("No solution")
end
```

```
No solution
```

[176]:
```
A = [1 1; 2 2]
b = [2, 5]
try
    x = A \ b
catch
    print("No solution")
end
```

```
No solution
```

Рис. 8: Раздел 2, задание 1.

Выполнила второе задание для самостоятельной работы:

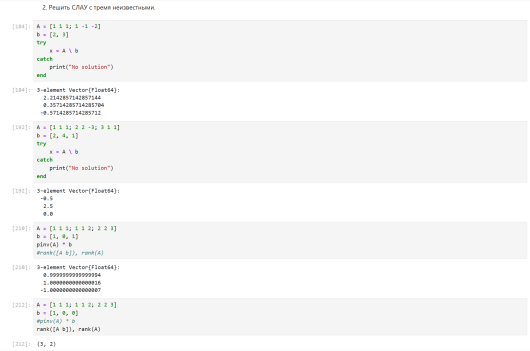
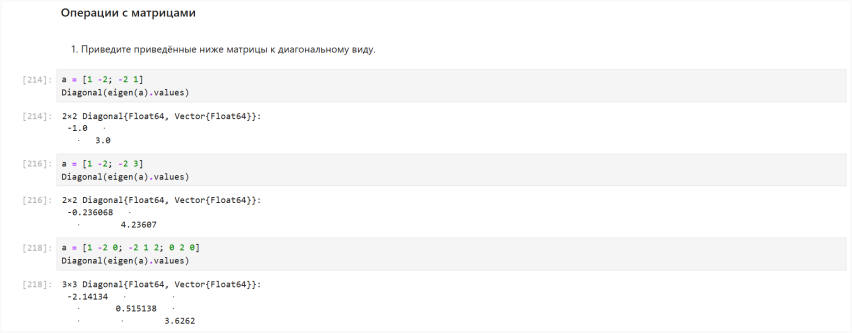


Рис. 9: Раздел 2, задание 2.

Выполнила первое задание для самостоятельной работы:



Операции с матрицами

1. Приведите приведённые ниже матрицы к диагональному виду.

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

[214]: 2×2 Diagonal{Float64, Vector{Float64}}:
        -1.0    ⋅
         ⋅     3.0

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

[216]: 2×2 Diagonal{Float64, Vector{Float64}}:
        -0.236068    ⋅
         ⋅          4.23607

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

[218]: 3×3 Diagonal{Float64, Vector{Float64}}:
        -2.14134    ⋅         ⋅
         ⋅         0.515138   ⋅
         ⋅          ⋅         3.6262
```

Рис. 10: Раздел 3, задание 1.

Выполнила второе задание для самостоятельной работы:

2. Вычислите.

```
[220]: a = [1 -2; -2 1]
       a^10
```

```
[220]: 2×2 Matrix{Int64}:
         29525  -29524
        -29524   29525
```

```
[222]: a = [5 -2; -2 5]
       sqrt(a)
```

```
[222]: 2×2 Matrix{Float64}:
         2.1889   -0.45685
        -0.45685   2.1889
```

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

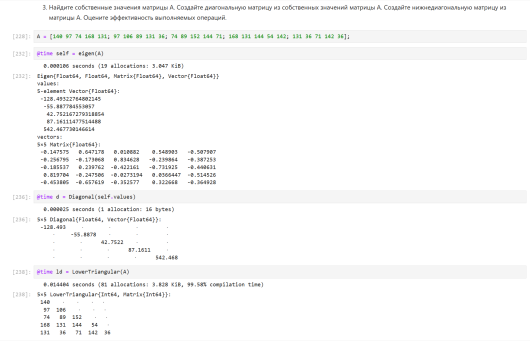
```
[224]: 2×2 Symmetric{ComplexF64, Matrix{ComplexF64}}:
         0.971125+0.433013im  -0.471125+0.433013im
        -0.471125+0.433013im   0.971125+0.433013im
```

```
[226]: a = [1 2; 2 3]
       sqrt(a)
```

```
[226]: 2×2 Matrix{ComplexF64}:
        0.568864+0.351578im  0.920442-0.217287im
        0.920442-0.217287im   1.48931+0.134291im
```

Рис. 11: Раздел 3, задание 2.

Выполнила третье задание для самостоятельной работы:



3. Найдите собственные значения матрицы A. Создайте диагональную матрицу из собственных значений матрицы A. Создайте нижнедиагональную матрицу из матрицы A. Оцените эффективность выполняемых операций.

```
[220]: 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];

[232]: @time self = eigen(A)
  0.000106 seconds (19 allocations: 3.047 KiB)

[232]: Eigen{Float64, Float64, Matrix{Float64}, Vector{Float64}}
values:
5-element Vector{Float64}:
 -128.49522764082145
  -55.88778455139073
   42.75216727931885
   87.16111477914488
  542.4677304146614
vectors:
5×5 Matrix{Float64}:
 -0.147575   0.647178   0.010082   0.548903   -0.507907
 -0.296795  -0.173068   0.834628  -0.239864   -0.387253
 -0.185537   0.238762  -0.422161  -0.731921   -0.440611
  0.819704  -0.247506  -0.0273194  0.0566447  -0.514526
 -0.453805  -0.657619  -0.352577   0.322668   -0.364928

[236]: @time d = Diagonal(self.values)
  0.000025 seconds (1 allocation: 16 bytes)
[236]: 5×5 Diagonal{Float64, Vector{Float64}}:
 -128.495     ⋅         ⋅         ⋅         ⋅
       ⋅   -55.8878      ⋅         ⋅         ⋅
       ⋅          ⋅    42.7522    ⋅         ⋅
       ⋅          ⋅          ⋅   87.1611    ⋅
       ⋅          ⋅          ⋅         ⋅   542.468

[238]: @time ld = LowerTriangular(A)
  0.014404 seconds (81 allocations: 3.828 KiB, 99.58% compilation time)
[238]: 5×5 LowerTriangular{Int64, Matrix{Int64}}:
 140
  97  106
  74   89  152
 168  131  144   54
 131   36   71  142   36
```

Рис. 12: Раздел 3, задание 3.

Выполнила первое задание для самостоятельной работы:

Линейные модели экономики

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

```
[242]: a = [1 2; 3 4]
       all(>=(0), inv(I - a)) ? "Продуктивная" : "Непродуктивная"

[242]: "Непродуктивная"

[244]: a = 0.5 * [1 2; 3 4]
       all(>=(0), inv(I - a)) ? "Продуктивная" : "Непродуктивная"

[244]: "Непродуктивная"

[246]: a = 0.1 * [1 2; 3 4]
       all(>=(0), inv(I - a)) ? "Продуктивная" : "Непродуктивная"

[246]: "Продуктивная"
```

Рис. 13: Раздел 4, задание 1.

Выполнила второе задание для самостоятельной работы:

2. Критерий продуктивности: матрица A является продуктивной тогда и только тогда, когда все элементы матрица $(E - A)^{-1}$ являются неотрицательными числами. Используя этот критерий, проверьте, являются ли матрицы продуктивными.

```
[248]: a = [1 2; 3 4]
       all(>=(0), inv(I - a)) ? "Продуктивная" : "Непродуктивная"

[248]: "Непродуктивная"

[250]: a = 0.5 * [1 2; 3 4]
       all(>=(0), inv(I - a)) ? "Продуктивная" : "Непродуктивная"

[250]: "Непродуктивная"

[252]: a = 0.1 * [1 2; 3 4]
       all(>=(0), inv(I - a)) ? "Продуктивная" : "Непродуктивная"

[252]: "Продуктивная"
```

Рис. 14: Раздел 4, задание 2.

Выполнила третье задание для самостоятельной работы:

3. Спектральный критерий продуктивности: матрица A является продуктивной тогда и только тогда, когда все её собственные значения по модулю меньше 1.
Используя этот критерий, проверьте, являются ли матрицы продуктивными.

```
[254]: a = [1 2; 3 4]
       all(x -> abs(x) < 1, eigvals(a)) ? "Продуктивная" : "Непродуктивная"

[254]: "Непродуктивная"

[256]: a = 0.5 * [1 2; 3 4]
       all(x -> abs(x) < 1, eigvals(a)) ? "Продуктивная" : "Непродуктивная"

[256]: "Непродуктивная"

[258]: a = 0.1 * [1 2; 3 4]
       all(x -> abs(x) < 1, eigvals(a)) ? "Продуктивная" : "Непродуктивная"

[258]: "Продуктивная"

[260]: a = [0.1 0.2 0.3; 0 0.1 0.2; 0 0.1 0.3]
       all(x -> abs(x) < 1, eigvals(a)) ? "Продуктивная" : "Непродуктивная"

[260]: "Продуктивная"
```

Рис. 15: Раздел 4, задание 3.

Заключение

Вывод

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