Отчёт по лабораторной работе №4  
Линейная алгебра

Статический анализ данных

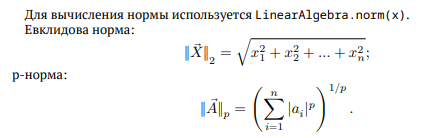
Выполнила: Коняева Марина Александровна,  
НФИбд-01-21, 1032217044

Содержание

# Цели лабораторной работы

Изучение возможностей специализированных пакетов Julia для выполнения и оценки эффективности операций над объектами линейной алгебры.

# Теоретическое введение







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.

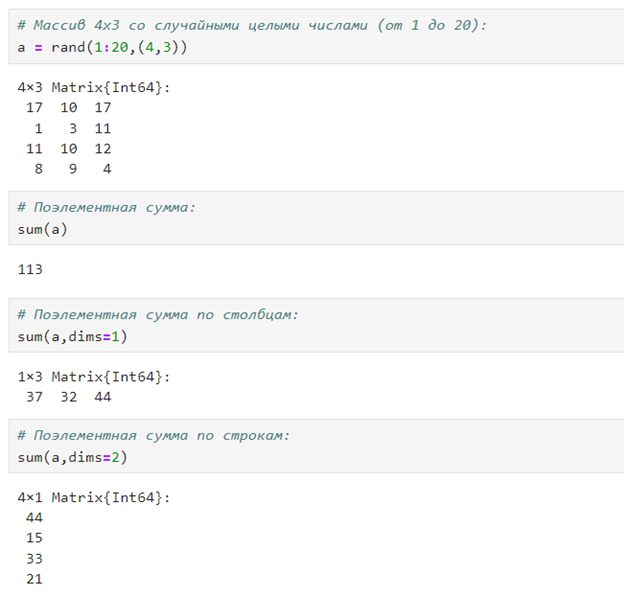
# Задачи лабораторной работы

1. Используя Jupyter Lab, повторите примеры из раздела 4.2.
2. Выполните задания для самостоятельной работы (раздел 4.4).

# Выполнение лабораторной работы

## Поэлементные операции над многомерными массивами

1. Изучим информацию о поэлементных операциях над многомерными массивами.
2. Повторим примеры с поэлементными операциями над многомерными массивами: Для матрицы 4 × 3 рассмотрим поэлементные операции сложения её элементов:



Матрица 4×3, сложения её элементов

1. Повторим примеры с поэлементными операциями над многомерными массивами: Для матрицы 4 × 3 рассмотрим поэлементные операции произведения её элементов:



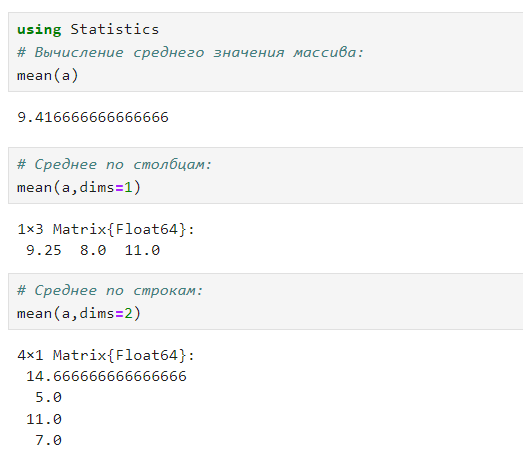
Матрица 4×3, произведение её элементов

1. Для работы со средними значениями можно воспользоваться возможностями пакета Statistics.



Добавление пакета

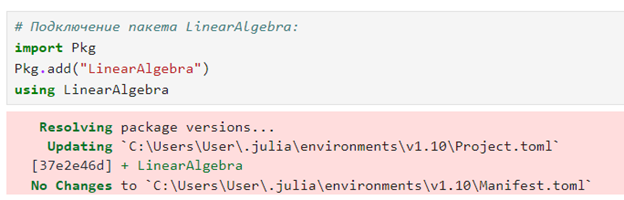
1. Повторим примеры с нахождением среднего значения массива, его среднего значения по столбцам и строкам.



Среднее значение массива

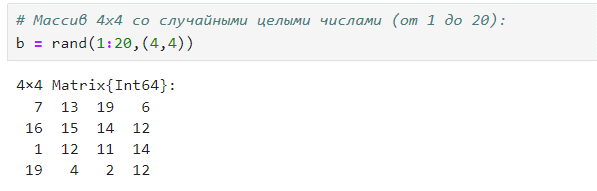
## Транспонирование, след, ранг, определитель и инверсия матрицы

1. Для выполнения таких операций над матрицами, как транспонирование, диагонализация, определение следа, ранга, определителя матрицы и т.п. можно воспользоваться библиотекой (пакетом) LinearAlgebra.



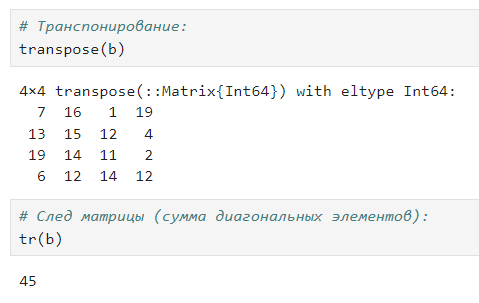
Добавление пакета

1. Повторим пример создание массива 4x4 со случайными целыми числами (от 1 до 20).



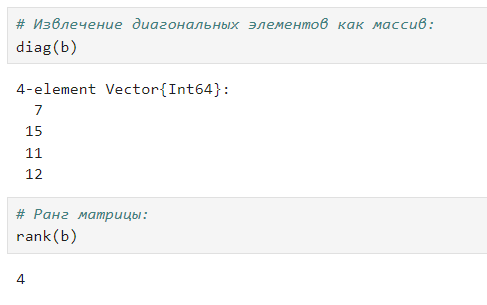
Массив 4x

1. Повторим примеры с массивом: транспонирование, след матрицы.



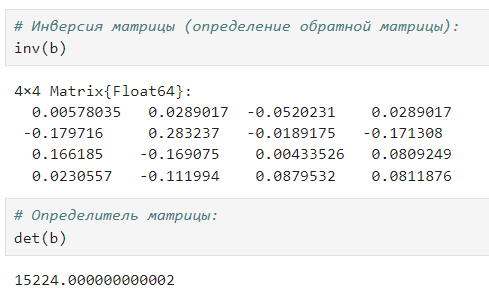
Транспонирование, след матрицы

1. Повторим примеры с массивом: извлечение диагональных элементов как массив, ранг матрицы.



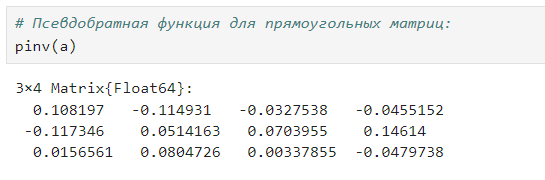
Извлечение диагональных элементов, ранг матрицы

1. Повторим примеры с массивом: инверсия матрицы (определение обратной матрицы), определитель матрицы.



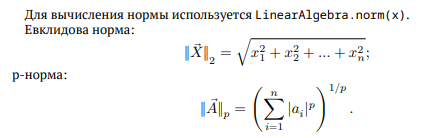
Инверсия матрицы, определитель матрицы

1. Повторим примеры с массивом: псевдобратная функция для прямоугольных матриц

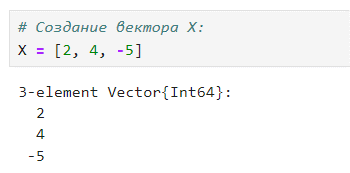


Псевдобратная функция для прямоугольных матриц

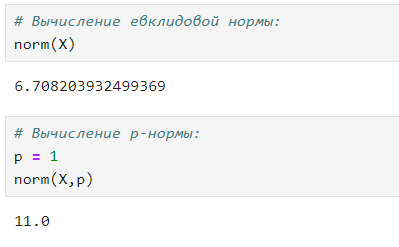
## Вычисление нормы векторов и матриц, повороты, вращения



1. Повторим пример с вычилением нормы, а именно создаем вектор, высчитываем евклидовую норму и р-норму.



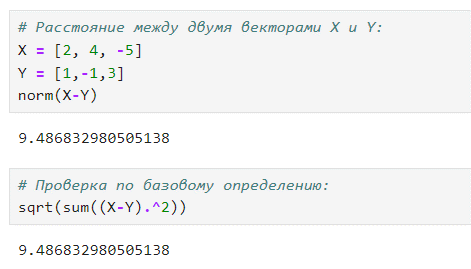
Вектор



Нормы



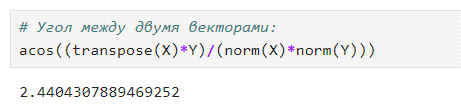
1. Повторим примеры с вычислением евклидово расстояния между двумя векторами.



Расстояние

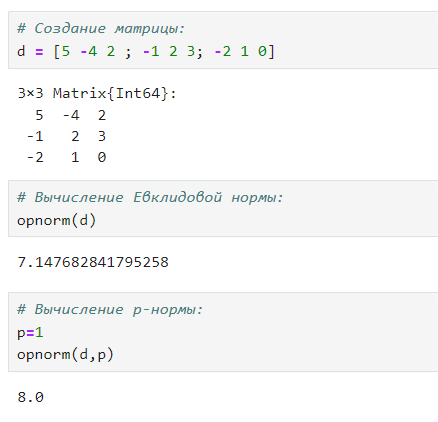


1. Повторим примеры с вычислением угла между двумя веткорами.



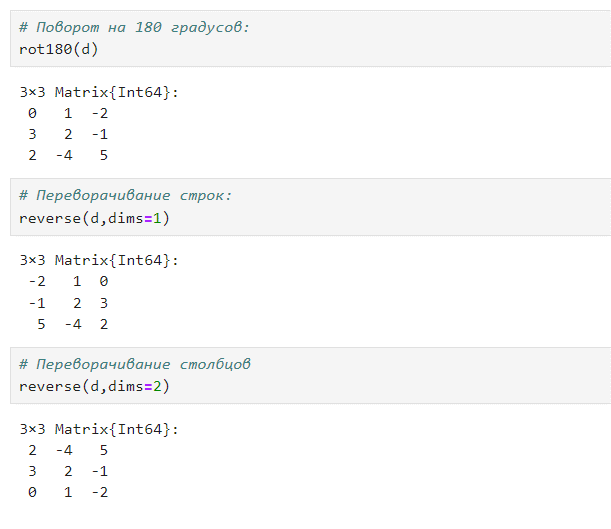
Угол

1. Повторим пример с вычилением нормы для двумерной мтарицы, а именно создаем матрицу, высчитываем евклидовую норму и р-норму.



Матрица и нормы

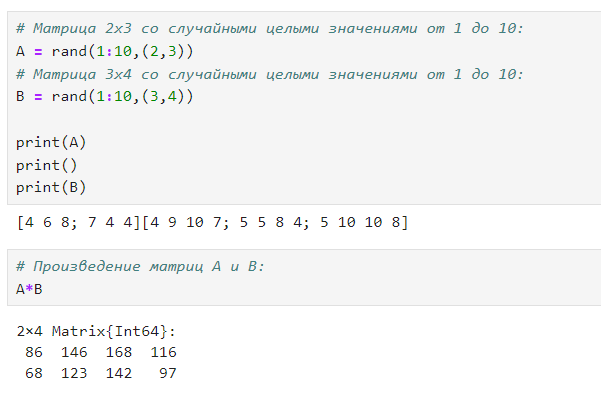
1. Выполним примеры с поворотом и переворачиваем по строкам и столбцам.



Примеры с поворотом и переворачиваем по строкам и столбцам

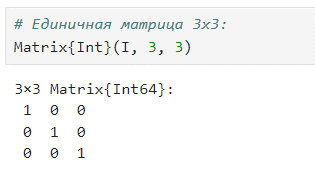
## Матричное умножение, единичная матрица, скалярное произведение

1. Повторим примеры: создадим дву матрицы и заполним случайными значения и вычислим произвидение этих матриц.



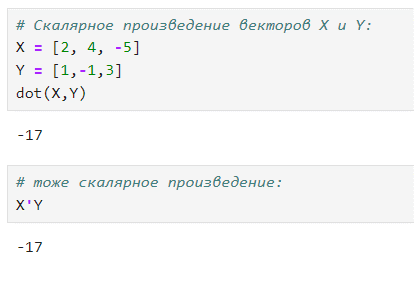
Матрицы и их произведение

1. Повторим пример создания единичной матрицы.



Матрица

1. Повторим примеры: вычислим скалярное произведение двух векторов разными способами.

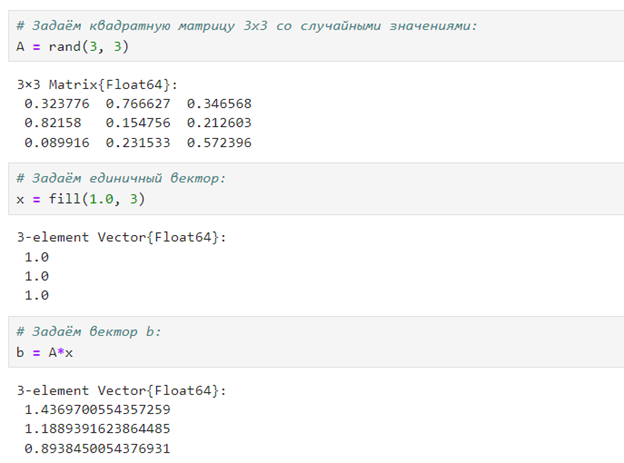


Скалярное произведение

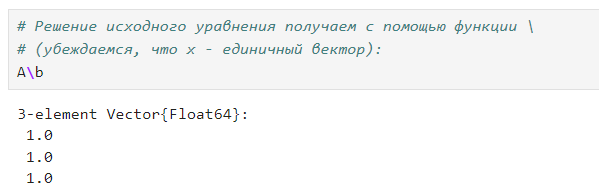
## Факторизация. Специальные матричные структуры

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. Матрица может быть факторизована на произведение матриц специального вида для приложений, в которых эта форма удобна. К специальным видам матриц относят ортогональные, унитарные и треугольные матрицы.

1. Повторим пример решение систем линейный алгебраических уравнений 𝐴𝑥 = 𝑏: зададим все начальные условия и найдем решение.

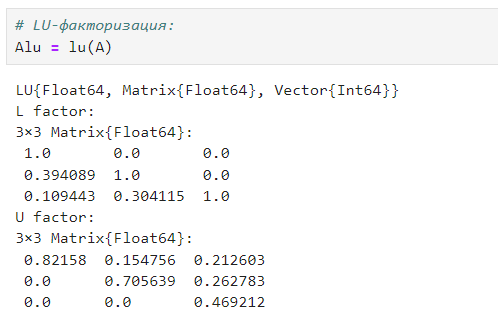


Начальные условия



Решение

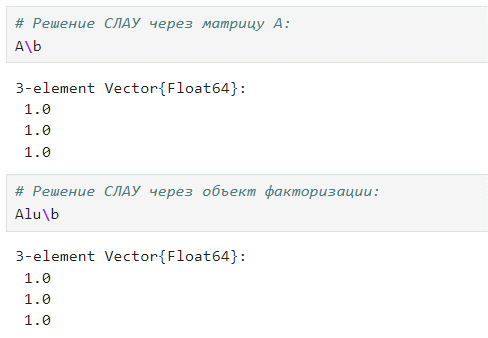
1. Вычислим факторизацию: Julia позволяет вычислять LU-факторизацию и определяет составной тип факторизации для его хранения:



LU-факторизация

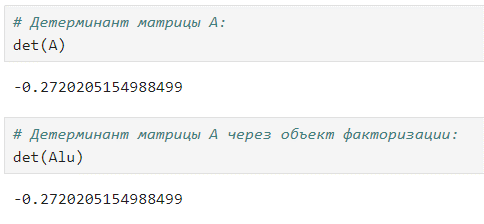
Различные части факторизации могут быть извлечены путём доступа к их специальным свойствам: Матрица перестановок: Alu.P Вектор перестановок: Alu.p Матрица L: Alu.L Матрица U: Alu.U

1. Повторим пример: исходная система уравнений 𝐴𝑥 = 𝑏 может быть решена или с использованием исходной матрицы, или с использованием объекта факторизации:



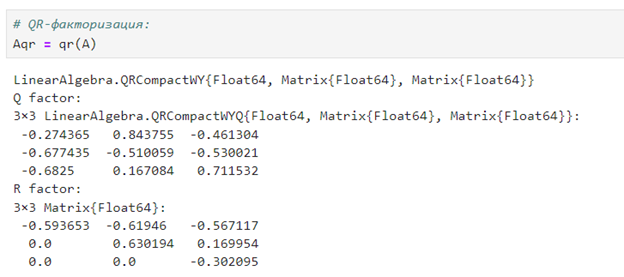
Решение

1. Повторим пример и найдем детерминат матрицы.



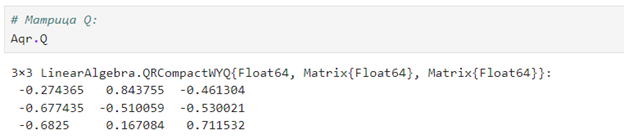
Детерминат

1. Выполним пример: Julia позволяет вычислять QR-факторизацию и определяет составной тип факторизации для его хранения.

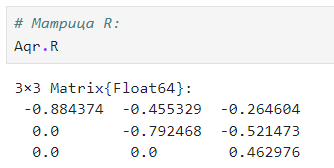


QR-факторизация

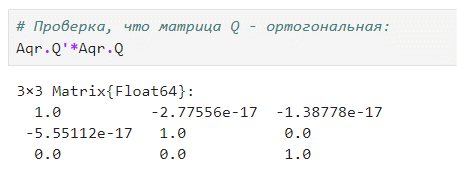
1. По аналогии с LU-факторизацией различные части QR-факторизации могут быть извлечены путём доступа к их специальным свойствам.



Q

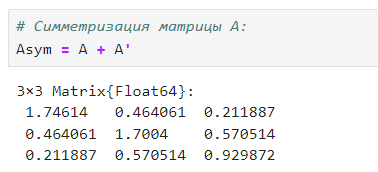


R

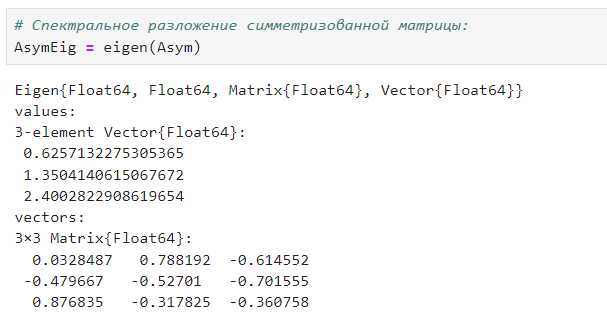


Проверка ортогональности

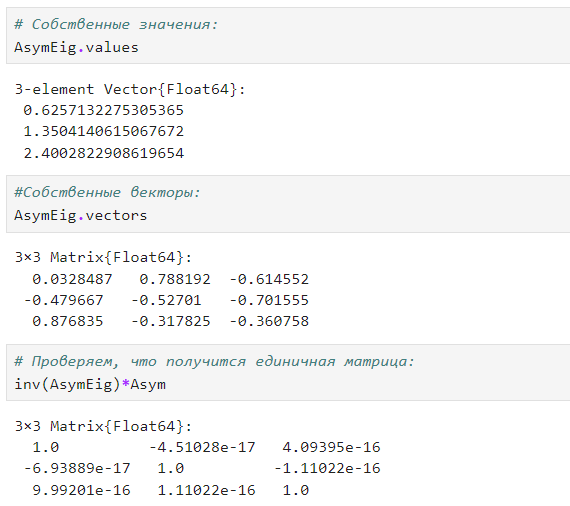
1. Выполним примеры собственной декомпозиции матрицы А, а именно симметризация матрицы A, спектральное разложение симметризованной матрицы, посик собственных значений и векторов.



Симметризация матрицы A

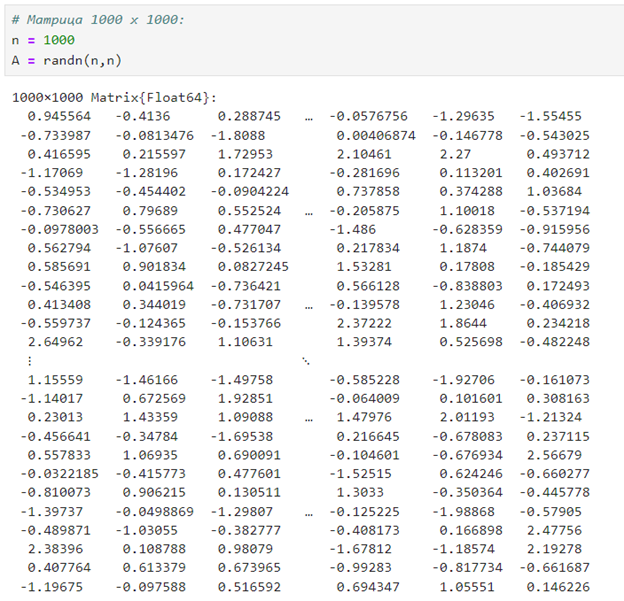


Спектральное разложение симметризованной матрицы



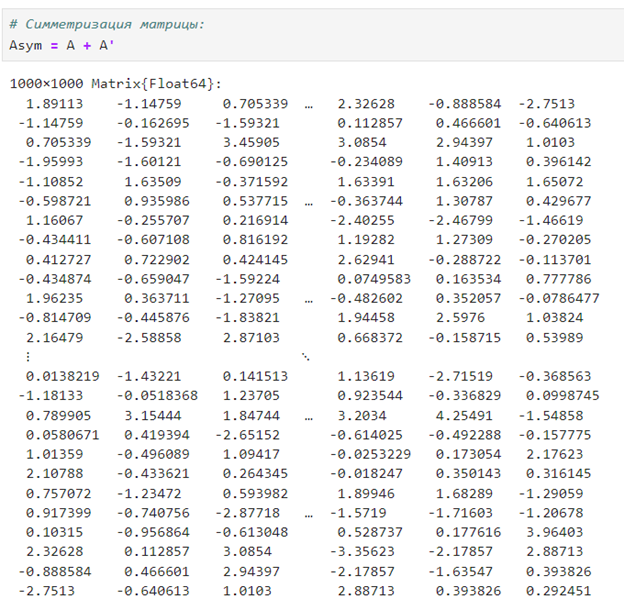
Собственные значения, векторы и проверка

1. Далее рассмотрим примеры работы с матрицами большой размерности и специальной структуры: матрица 1000 х 1000.

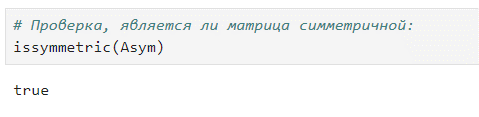


Матрица 1000 х 1000

1. Выполним для данной матрицы симметризацию, и проверку на симметрию.

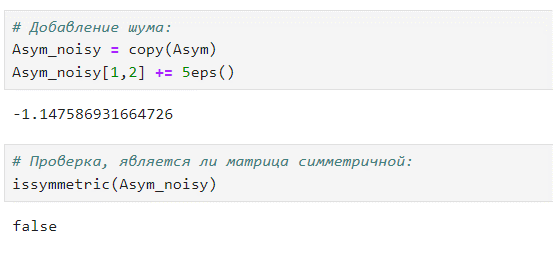


Симметризация



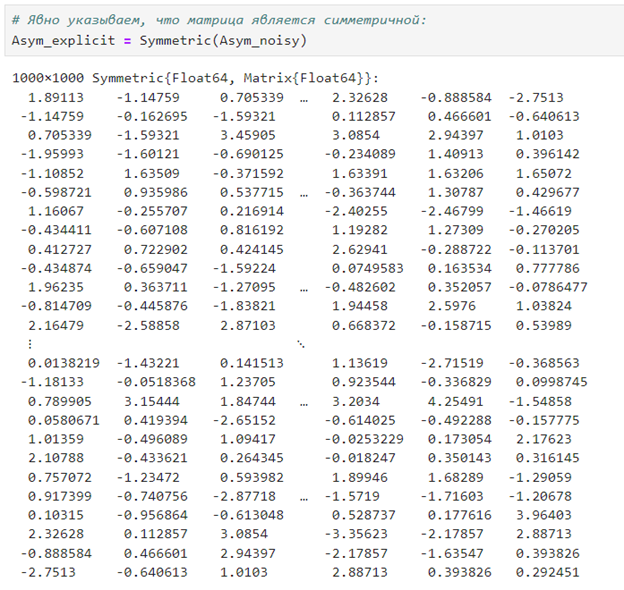
Проверка

1. Выполним пример добавления шума в симметричную матрицу (матрица уже не будет симметричной).



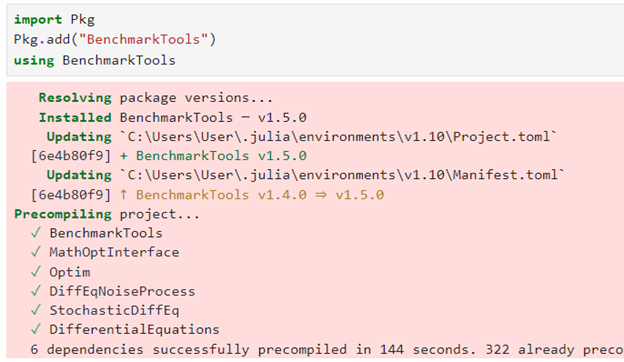
Добавление шума и проверка

1. В Julia можно объявить структуру матрица явно, например, используя Diagonal, Triangular, Symmetric, Hermitian, Tridiagonal и SymTridiagonal.



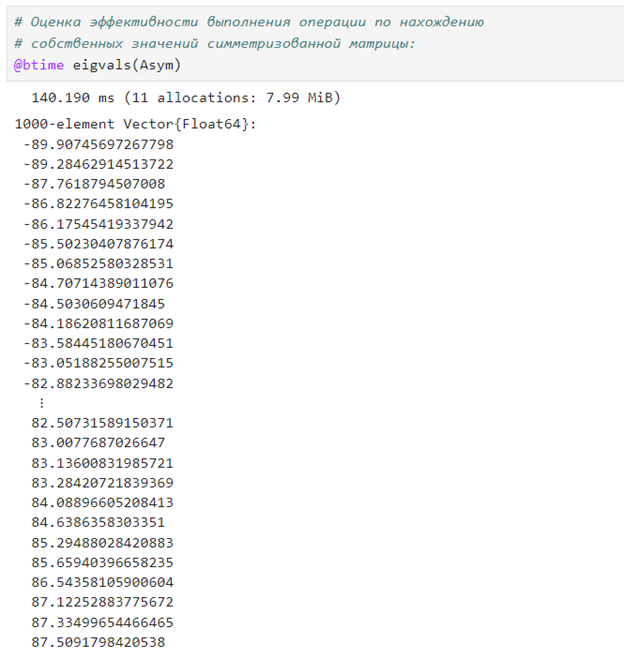
Указание на симметричность

1. Далее для оценки эффективности выполнения операций над матрицами большой размерности и специальной структуры воспользуемся пакетом BenchmarkTools, добавим его.

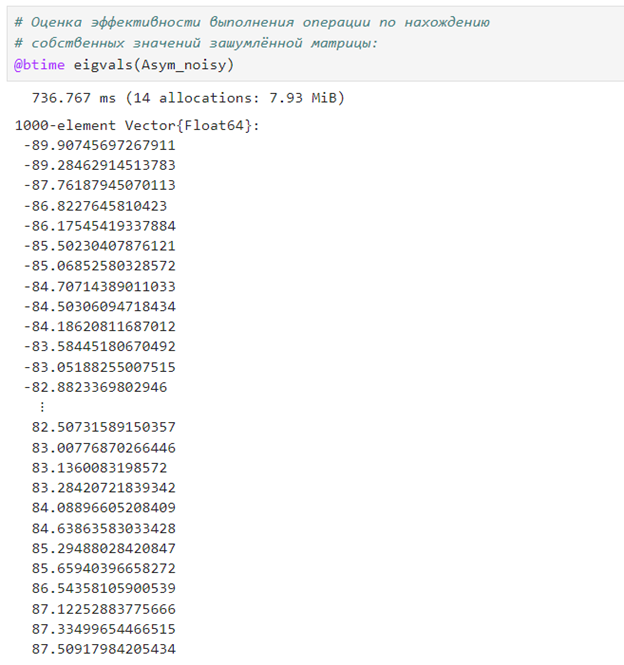


Добавление пакета

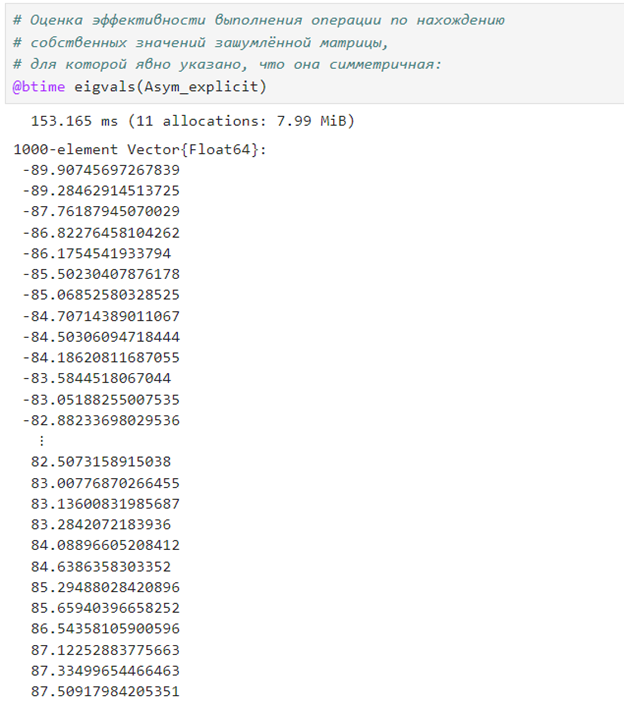
1. Выполним оценку эффективности выполнения операции по нахождению собственных значений для различных матриц.



Оценка эффективности 1

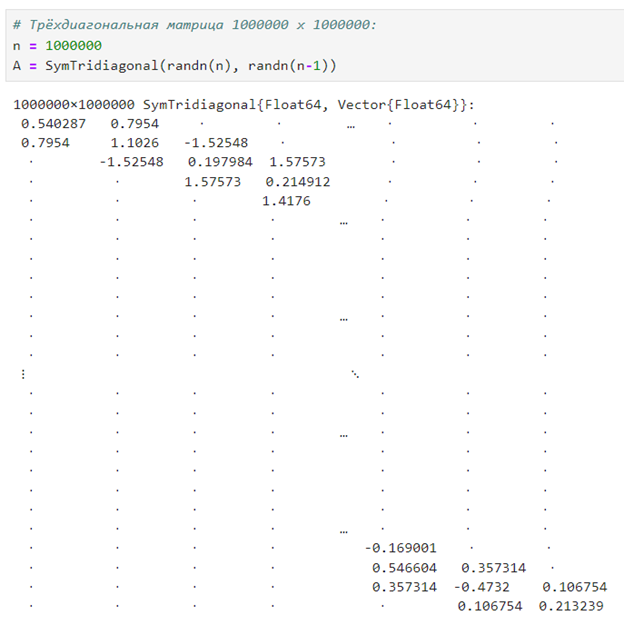


Оценка эффективности 2

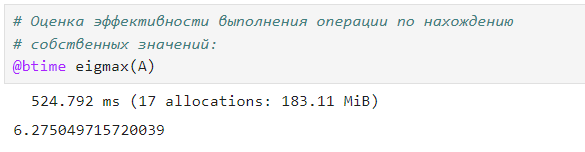


Оценка эффективности 3

1. Далее рассмотрим примеры работы с разряженными матрицами большой размерности. Использование типов Tridiagonal и SymTridiagonal для хранения трёхдиагональных матриц позволяет работать с потенциально очень большими трёхдиагональными матрицами. Выполним оценку эффективности.



Матрица 1000000х1000000



Оценка эффективности

1. При попытке задать подобную матрицу обычным способом и посчитать её собственные значения, мы получим ошибку переполнения памяти.

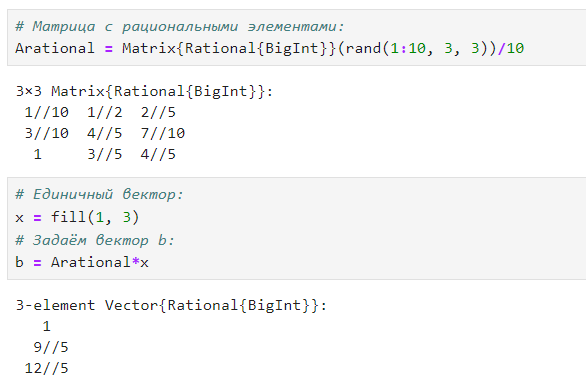


Ошибка

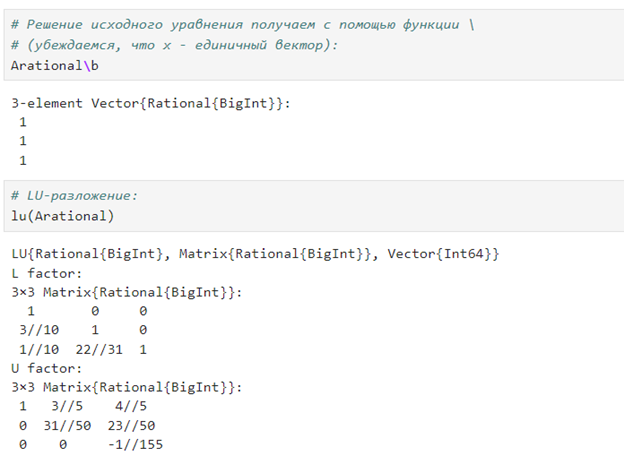
## Общая линейная алгебра

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.

1. 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

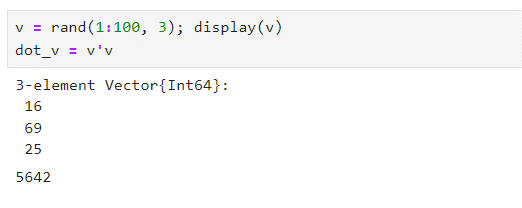


Решение и LU-разложение

## Задания для самостоятельного выполнения

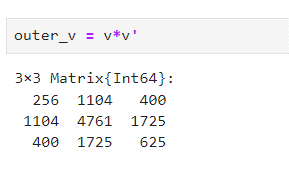
### Произведение векторов

1. Выполним 1.1 задание: задайте вектор v. Умножьте вектор v скалярно сам на себя и сохраните результат в dot\_v.



1.1 задание

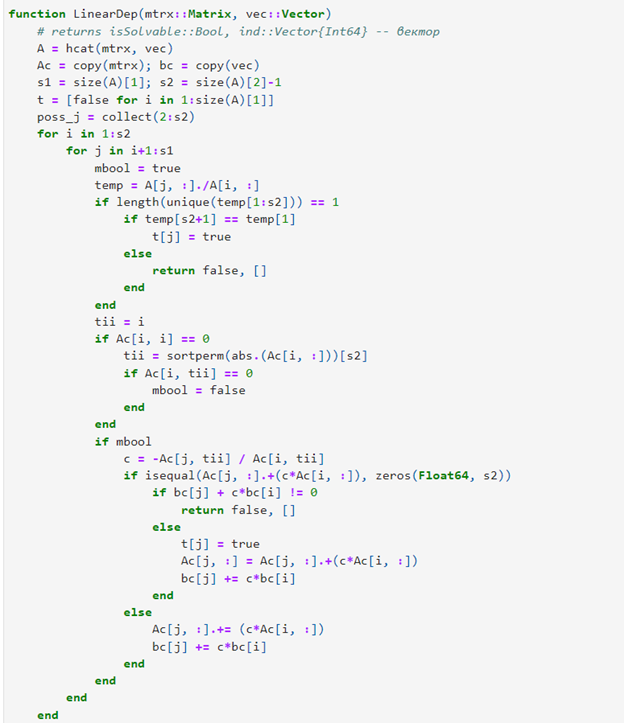
1. Выполним 1.2 задание: умножьте v матрично на себя (внешнее произведение), присвоив результат переменной outer\_v.



1.2 задание

### Системы линейных уравнений

1. Выполним 2 задание: Решить СЛАУ с двумя/тремя неизвестными.

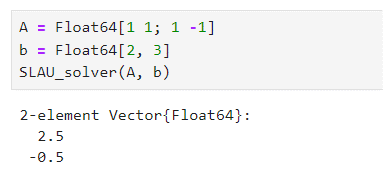


Функция для решения



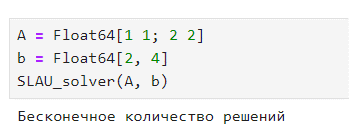
Функция для решения

1. Решение существует (система линейно независима)



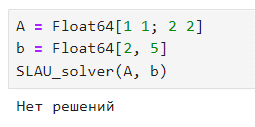
а)

1. Бесконечное количество решений (вся система линейно зависима и коэффициенты и ветокры ответов)



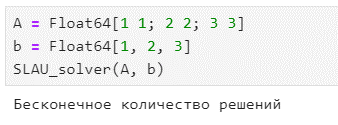
b)

1. Нет решений (матрица коэффициентов линейно зависима, при этом векторы нет)



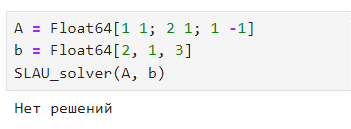
c)

1. Бесконечное количество решений (вся система линейно зависима)



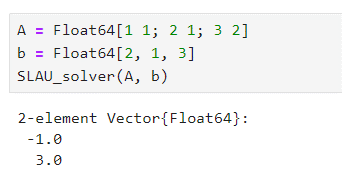
d)

1. Нет решений



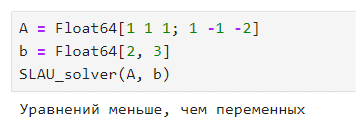
e)

1. Решение существует

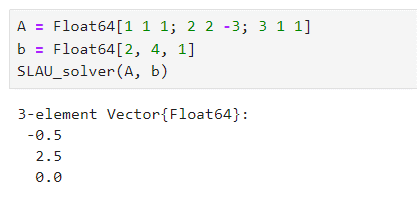


f)

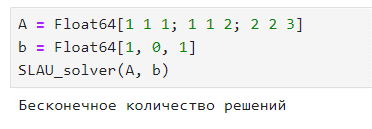
Решить СЛАУ с тремя неизвестными:



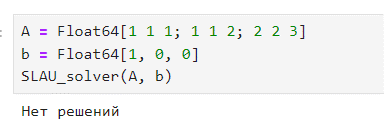
а)



b)



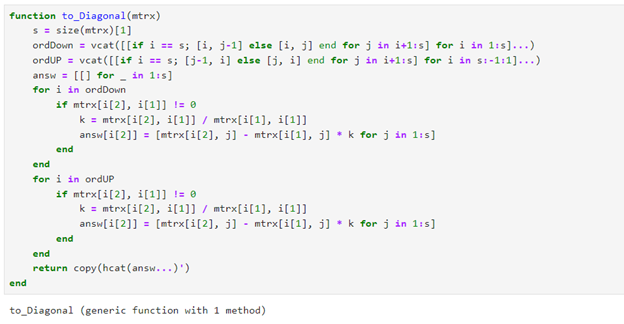
c)



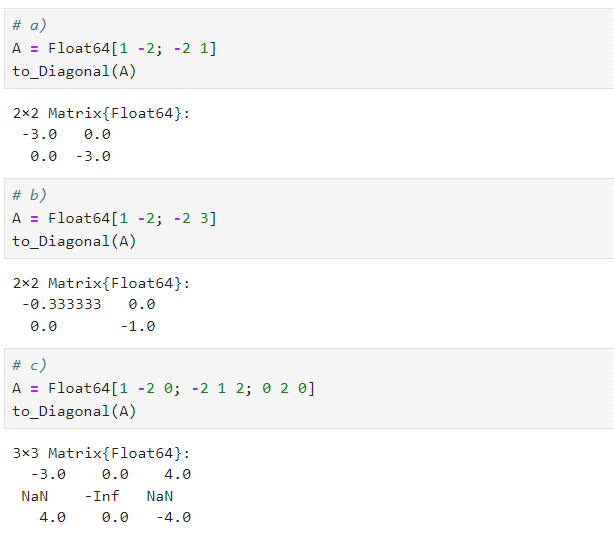
d)

### Операции с матрицами

1. Приведите приведённые ниже матрицы к диагональному виду:

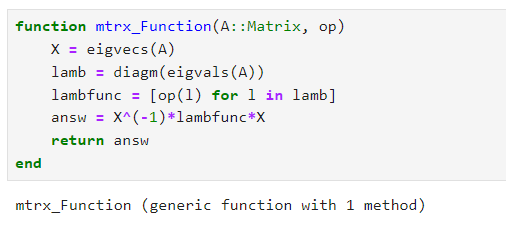


Функция для решения

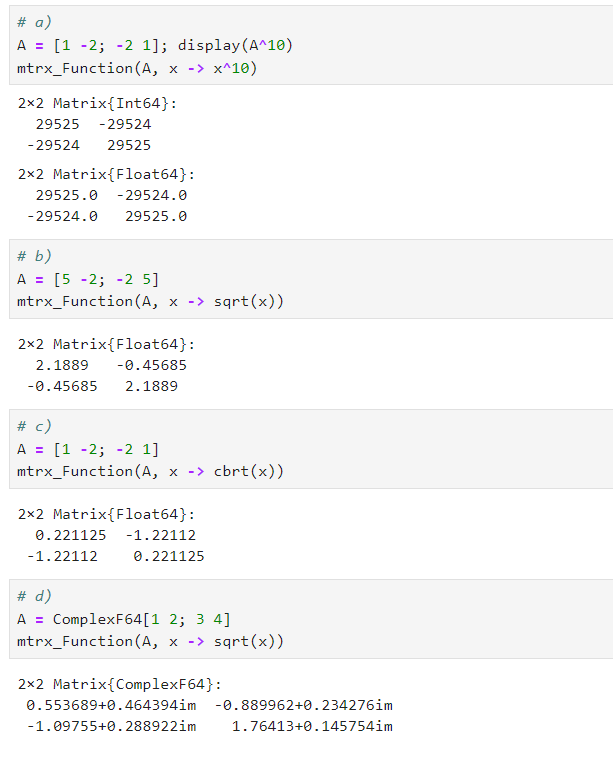


а,b,c)

* Вычислите:

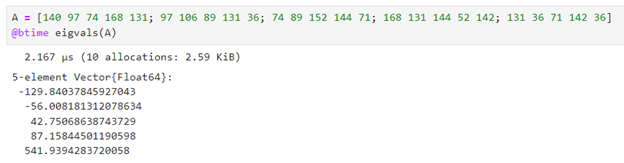


Функция для решения



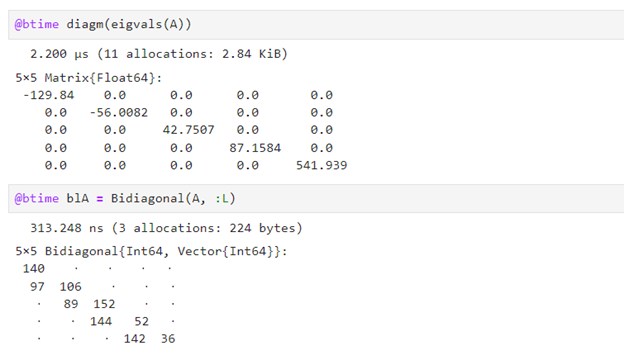
a,b,c,d)

* Найдите собственные значения матрицы 𝐴, если:



A

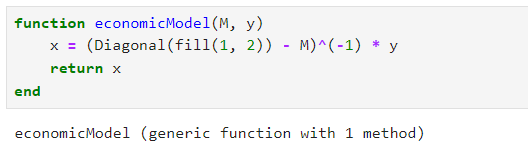
* Создайте диагональную матрицу из собственных значений матрицы 𝐴. Создайте нижнедиагональную матрицу из матрица 𝐴. Оцените эффективность выполняемых операций.



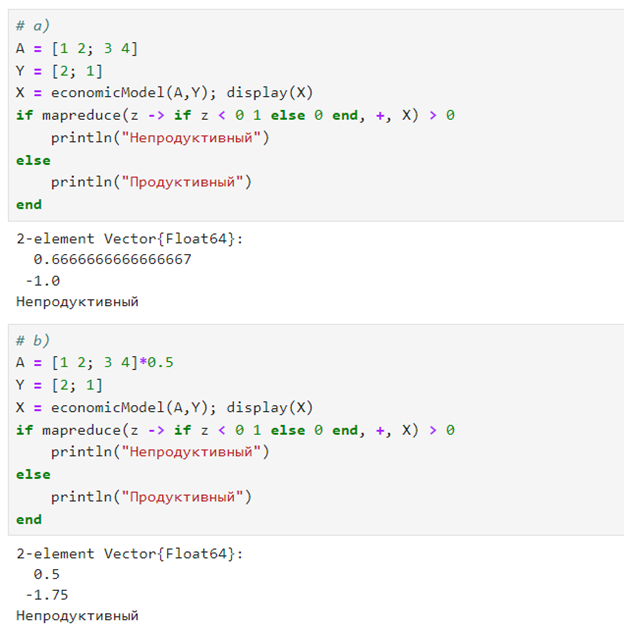
A из собственных значений

### Линейные модели экономики

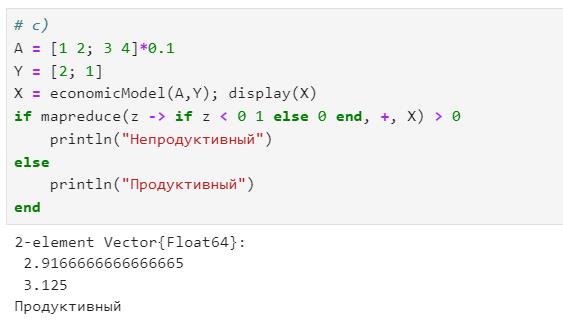
1. Выполним задание: 1. Матрица 𝐴 называется продуктивной, если решение 𝑥 системы при любой неотрицательной правой части 𝑦 имеет только неотрицательные элементы 𝑥𝑖. Используя это определение, проверьте, являются ли матрицы продуктивными.



Функция для решения

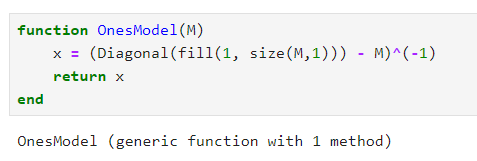


a,b)



c)

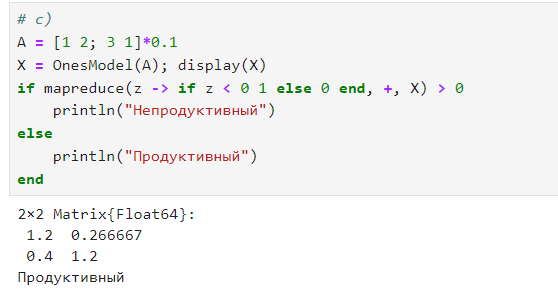
1. Выполним задание: 2. Критерий продуктивности: матрица 𝐴 является продуктивной тогда и только тогда, когда все элементы матрица (𝐸 − 𝐴)−1 являются неотрицательными числами. Используя этот критерий, проверьте, являются ли матрицы продуктивными.



Функция для решения



a,b)



c)

1. Выполним задание: 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.



a,b)



c,d)

# Выводы по проделанной работе

## Вывод

В результате выполнения работы мы изучили возможностей специализированных пакетов Julia для выполнения и оценки эффективности операций над объектами линейной алгебры. Были записаны скринкасты выполнения , создания отчета, презентации и защиты лабораторной работы.

# Список литературы

* Julia: https://ru.wikipedia.org/wiki/Julia
* https://julialang.org/packages/
* https://juliahub.com/ui/Home
* https://juliaobserver.com/
* https://github.com/svaksha/Julia.jl