Отчёт по лабораторной работе №3

Дисциплна: Научное программирование

Живцова Анна, 1132249547

Содержание

# 1 Цель работы

* Изучить основы языков научного программирования Octave и Julia
* Выполнить практические примеры для закрепления синтаксиса
* Проверить эффективность векторных вычислений

# 2 Задание

* Изучить типы переменных
* Выполнить примеры операций из линейной алгебры
* Освоить процедуру рисования графиков
* На примере сравнить скорость выполнения программ, записанных через цикл и векторные операции

# 3 Теоретическое введение

Octave – язык научного программирования ысокого уровня. Задуманный изначально как программное пособие для проектирования химического реактора и названный в честь профессора химии Октава Левеншпиля, преподававшего автору математического пакета, Octave призван был заменить студентам Техасского Университета сложный в отладке Fortran. Версия 1.0 вышла в свет 17 февраля 1994 г.

GNU Octave — свободная программная система для математических вычислений, использующая совместимый с MATLAB язык высокого уровня. Предоставляет интерактивный командный интерфейс для решения линейных и нелинейных математических задач, а также проведения других численных экспериментов. Кроме того, Octave можно использовать для пакетной обработки. Язык Octave оперирует арифметикой вещественных и комплексных скаляров, векторов и матриц, имеет расширения для решения линейных алгебраических задач, нахождения корней систем нелинейных алгебраических уравнений, работы с полиномами, решения различных дифференциальных уравнений, интегрирования систем дифференциальных и дифференциально-алгебраических уравнений первого порядка, интегрирования функций на конечных и бесконечных интервалах. Система написана на C++ с использованием стандартной библиотеки шаблонов [1].

Julia — высокоуровневый свободный язык программирования с динамической типизацией, созданный для математических вычислений. Эффективен также и для написания программ общего назначения. Синтаксис языка схож с синтаксисом других математических языков (например, MATLAB и Octave), однако имеет некоторые существенные отличия. Julia написан на Си, C++ и Scheme. Имеет встроенную поддержку многопоточности и распределённых вычислений, реализованные в том числе в стандартных конструкциях.

Язык является динамическим, при этом поддерживает JIT-компиляцию (JIT-компилятор на основе LLVM входит в стандартный комплект), благодаря чему, по утверждению авторов языка, приложения, полностью написанные на языке (без использование низкоуровневых библиотек и векторных операций) практически не уступают в производительности приложениям, написанным на статически компилируемых языках, таких как Си или C++. Большая часть стандартной библиотеки языка написана на нём же [2].

# 4 Выполнение лабораторной работы

1. Типы данных и операции. Инициализировала переменные типа вектор-строка, вектор-столбец, матрица (см рис. 1 и 2). Выполнила арифметические операции (сложение, вычитание, умножение, возведение в степень), а также операции линейной алгебры: сложение векторов, умножения на скаляр, скалярное умножение, векторное умножение, матричное умножение, трансппонирование, сложениие матриц и обращение матриц. Нашла проекцию вектора на вектор, норму вектора, а также определитель, ранг и собственные значения матрицы (см рис. 3, 5, 4, 6).

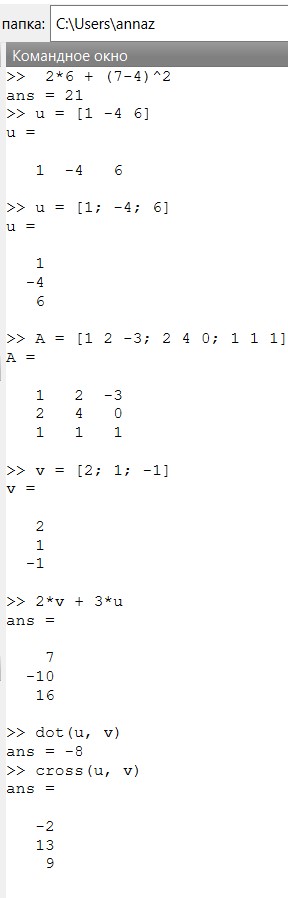


Рис. 1: Типы данных. Octave

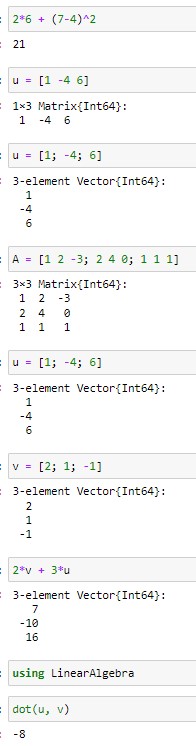


Рис. 2: Типы данных. Julia

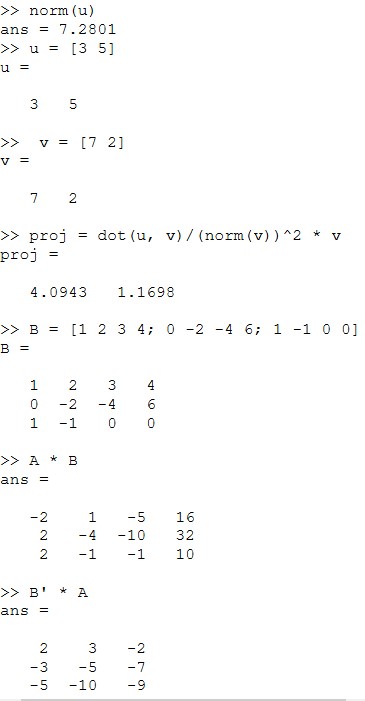


Рис. 3: Операции линейной алгебры. Octave

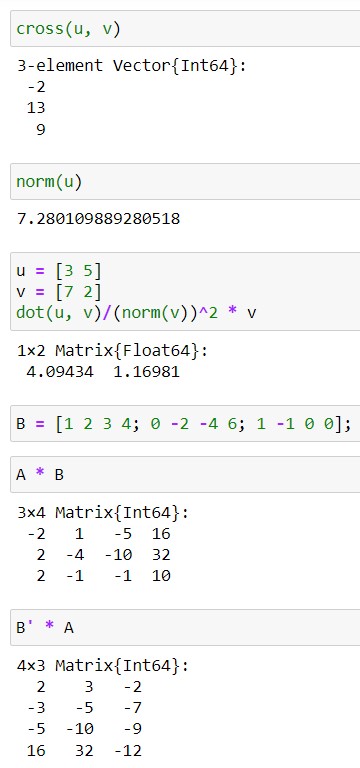


Рис. 4: Операции линейной алгебры. Julia

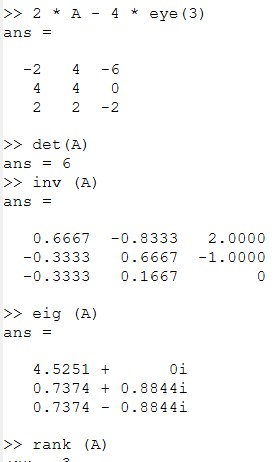


Рис. 5: Операции линейной алгебры (2). Octave

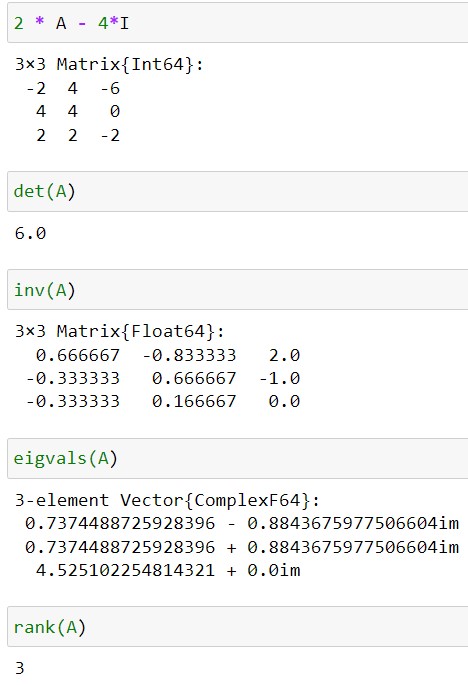


Рис. 6: Операции линейной алгебры (2). Julia

1. Рисование графиков. Освоила функцию рисования графиков и способы настройки внешнего вида графиков: подпись осей, совмещение нескольких графиков на одном рисунке, установка легенды, сетки, толщины линий, цвета линий, названия рисунка, диапазона осей (см рис. 7, 9, 11, 8, 10, 12).

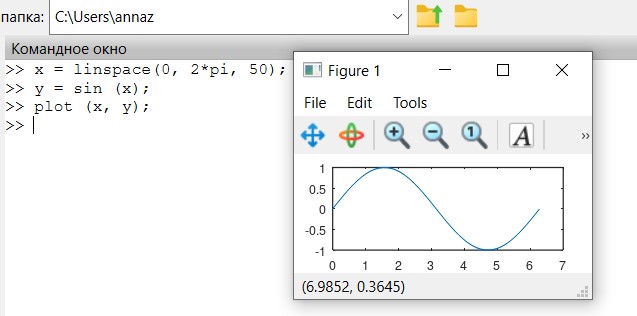


Рис. 7: Базовый вид графика. Octave

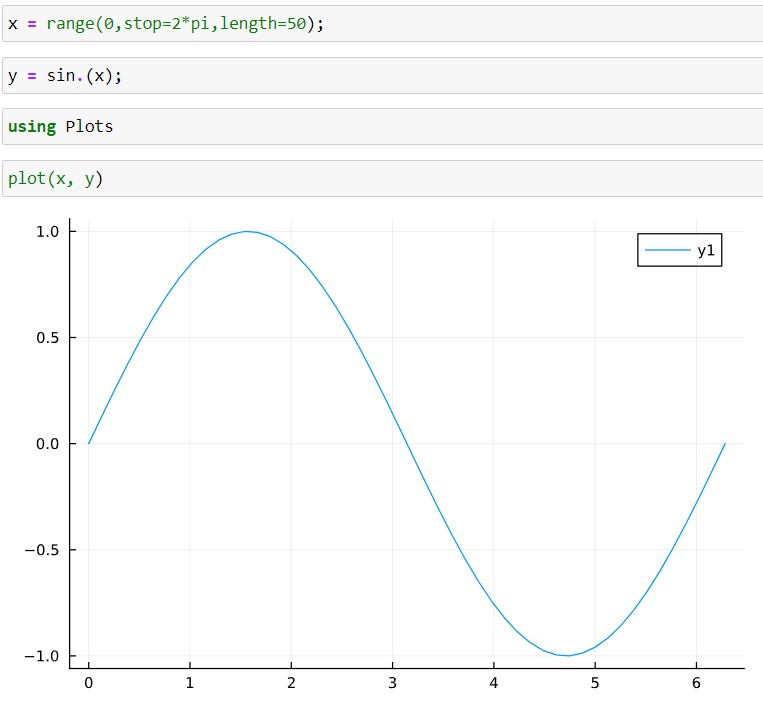


Рис. 8: Базовый вид графика. Julia

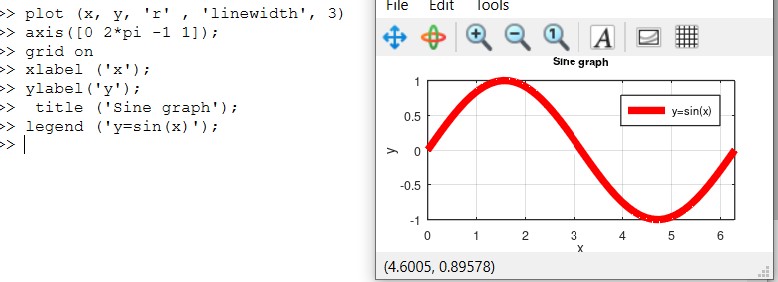


Рис. 9: График с настроенным внешним видом. Octave

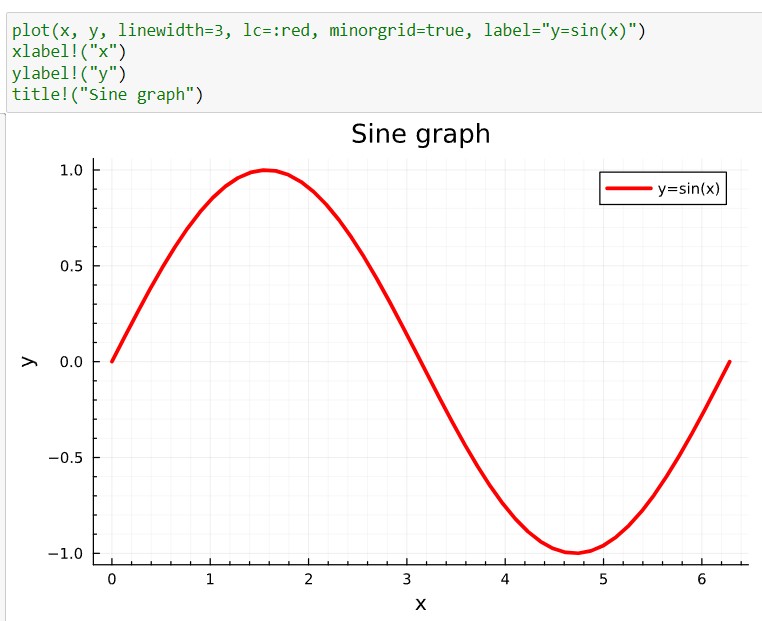


Рис. 10: График с настроенным внешним видом. Julia

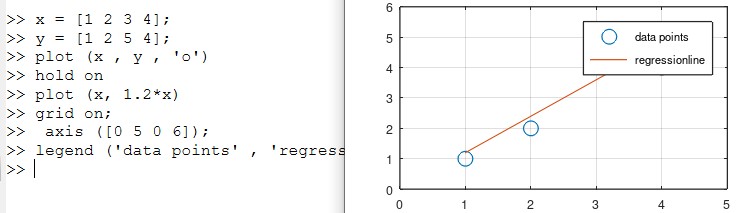


Рис. 11: Два графика на одномрисунке. Octave

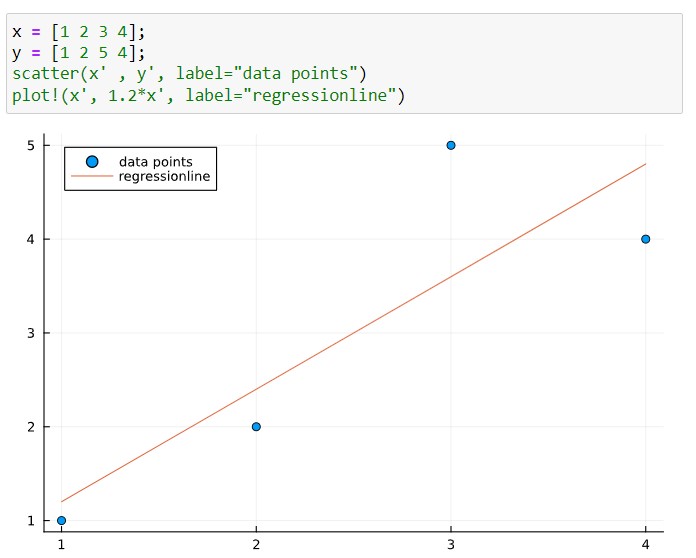


Рис. 12: Два графика на одномрисунке. Julia

1. Поэлемеентные операции. Изучила синтаксис поэлементных операций (см. рис 13 и 14). Проверила скорость выполнения программы при использовании цикла и поэлементных операций в векторах (см. рис 16 и 15).

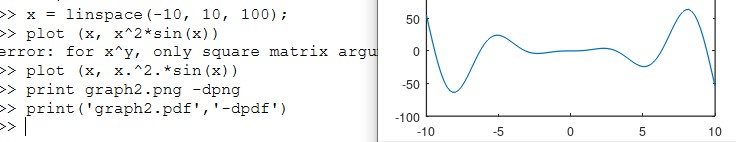


Рис. 13: Пример применения поэлементных операций. Octave

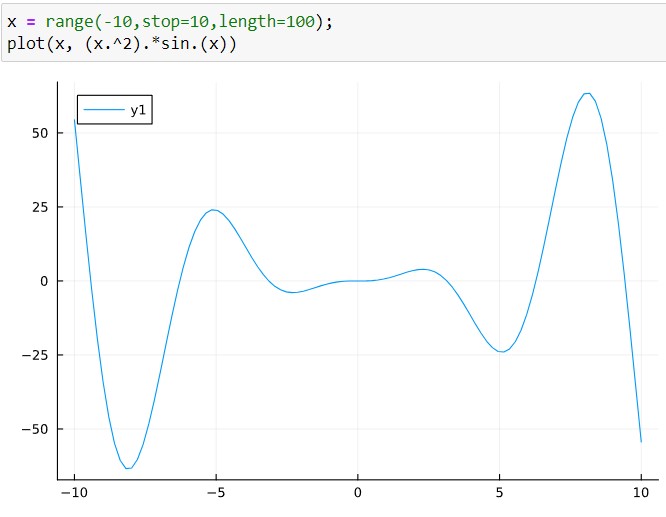


Рис. 14: Пример применения поэлементных операций. Julia

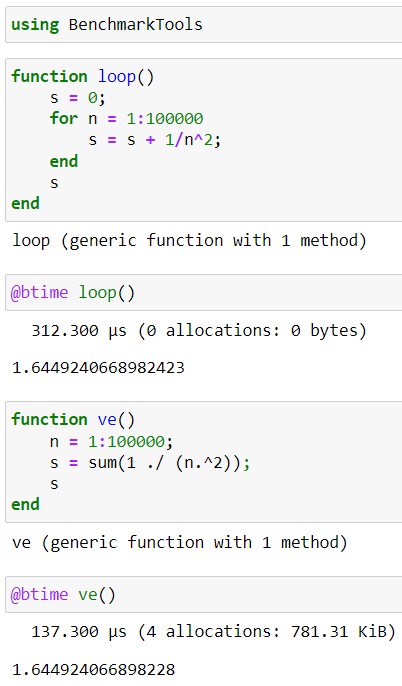


Рис. 15: Сравнение времени выполнения программы в зависимости от формы записи. Julia

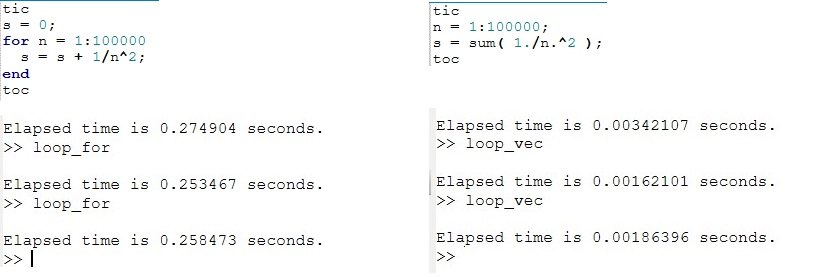


Рис. 16: Сравнение времени выполнения программы в зависимости от формы записи. Octave

# 5 Выводы

В данной работе я познакомилась с языками научного программирования Octave и Julia. Изучила некоторые типы данных, арифметические операции (сложение, вычитание, умножение, возведение в степень), а также операции линейной алгебры: сложение векторов, умножения на скаляр, скалярное умножение, векторное умножение, матричное умножение, трансппонирование, сложениие матриц и обращение матриц. Нашла проекцию вектора на вектор, норму вектора, а также определитель, ранг и собственные значения матрицы. Освоила процедуру рисования и настройки внешнего вида (цвета, легенды, подписи осей, названия рисунка, толщины линий, сетки, масштаба) графиков, а также на примере сравнила скорость выполнения программ, записанных через цикл и векторные операции. Поэлементные операции с векторами показывают лучшую производительность.

# 6 Список литературы

1. [GNU Octave documentation](https://docs.octave.org/latest/). The Octave Project Developers, 2024.

2. [Julia documentation](https://docs.julialang.org/en/v1/). Julia Programming Language., 2024.