Лабораторная работа №3

Научное программирование

Дарижапов Тимур Андреевич

Содержание

List of Tables

List of Figures

1 Цель работы

Изучить идеологию и применение языка Octave, познакомится с основными командами и возможностями языка.

2 Задание

Разобраться со спецификой языка и выполнить основные операции.

1. Простейшие операции.
2. Операции с векторами.
3. Вычисление проектора.
4. Матричные операции.
5. Построение простейших графиков.
6. Построение двух графиков на одном чертеже
7. Построение графика сложной функции
8. Сравнение циклов и операций над векторами

3 Выполнение лабораторной работы

Для начала работы с программой включим журналирование сессии командой diary on. Затем приступим к выполнению первого этапа - простейших операций. Сначала используем как простейший калькулятор и вычислим выражение. Зададим вектор u и покажем как сделать вектор-строку и вектор-столбец. Зададим матрицу A. (рис. 3.1)

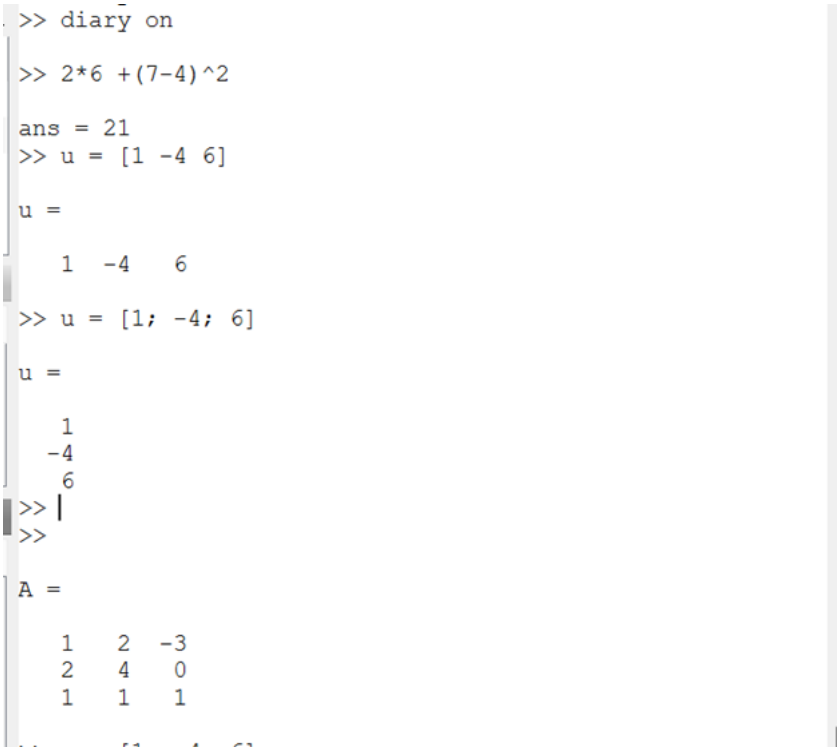

Figure 3.1: Ввод данных

Теперь зададим два вектора-столбца v и u, с которыми будем совершать операции. (рис. 3.2)

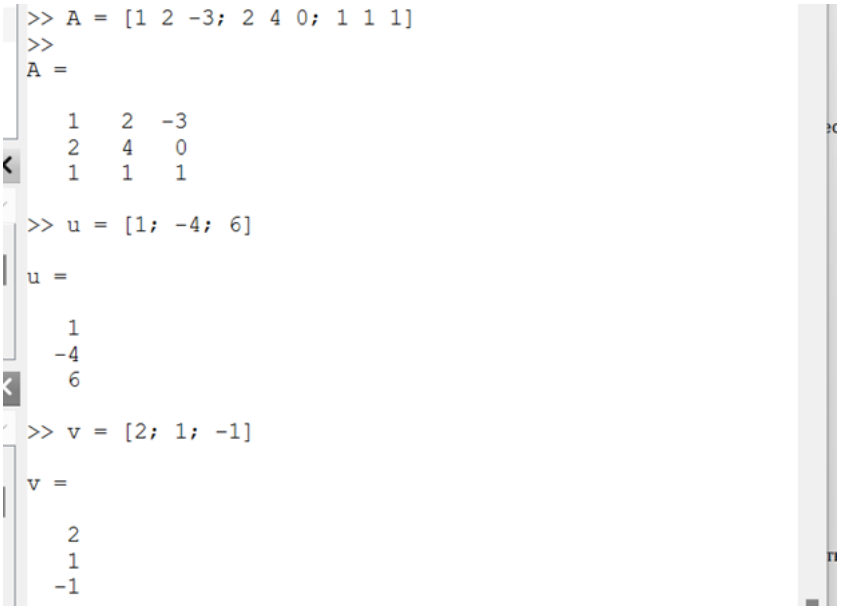

Figure 3.2: Зададим значения

Осуществим несколько операций с ними: сложение, скалярное умножение, векторное умножение, вычисление нормы. (рис. 3.3)

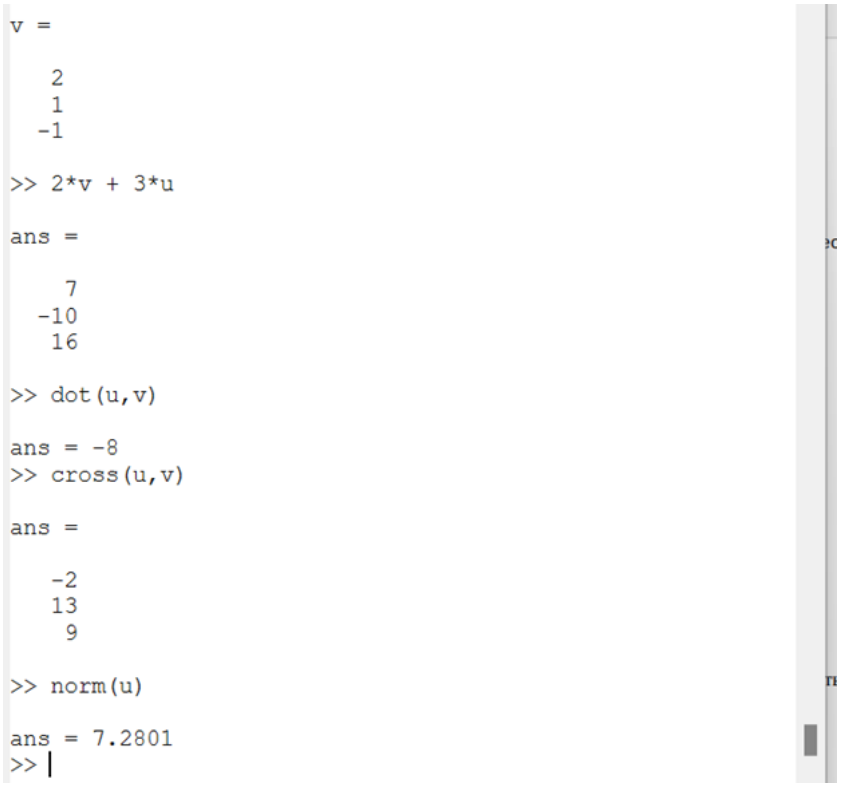

Figure 3.3: Операции над векторами 1

Приступим к следующему этапу работы: вычислим проекции. Для этого введём два вектора строки. Выведя необходимую формулу, вычислим её в octave. (рис. 3.4)

```
u =

   3   5

>> v = [7 2]

v =

   7   2

>> proj = dot(u,v)/(norm(v))^2 * v

proj =

   4.0943   1.1698
```

Figure 3.4: Операции над векторами 2

Следующий этап - матричные операции. Введём две матрицы - A и B. Далее проведём несколько операций над ними. Сначала вычислим произведение матриц A*B (рис. 3.5) , а затем произведение транспонированной B на A. Так же вычислим значение выражения 2*A − 4*I, где I - единичная матрица. (рис. 3.6)

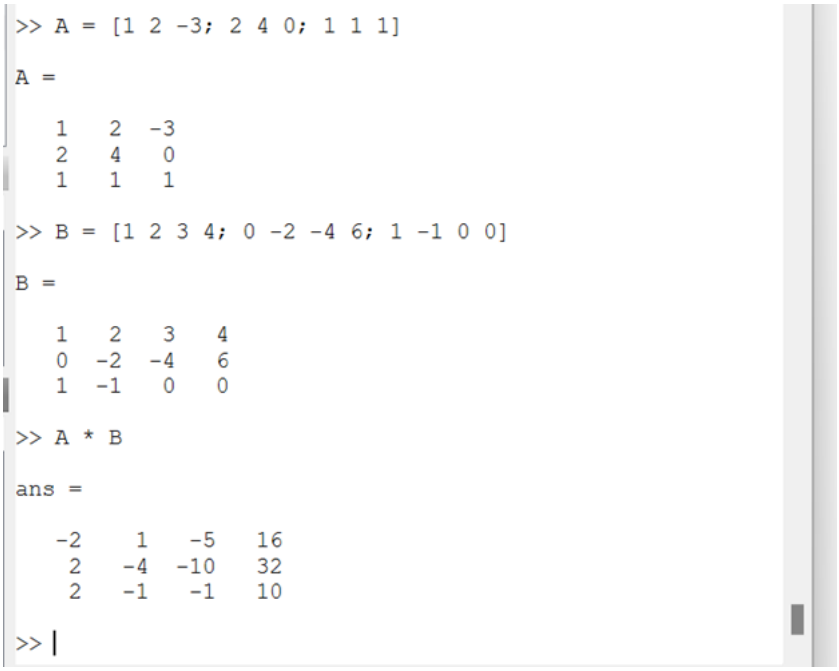

Figure 3.5: Операции над матрицами 1

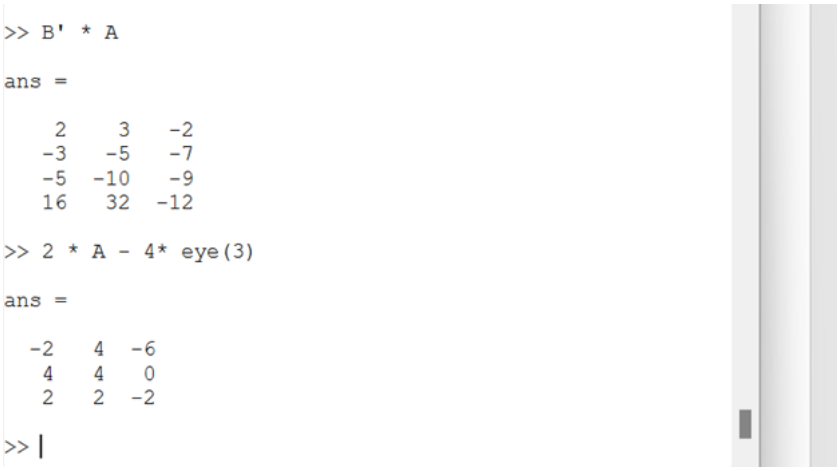

Figure 3.6: Операции над матрицами 2

Теперь найдём обратную матрицу A и найдём её собственные значения. (рис. 3.7)

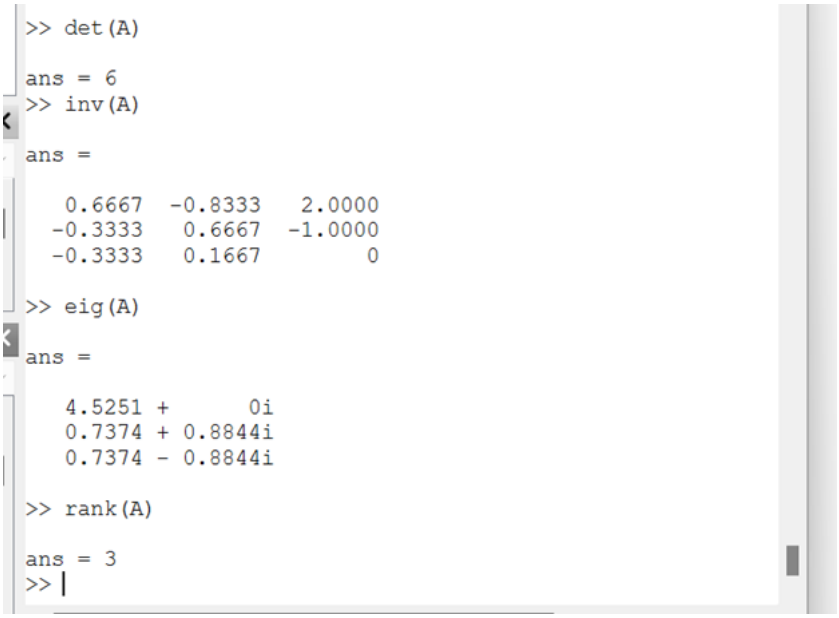

Figure 3.7: Операции над матрицами 3

Теперь научимся рисовать графики. Для начала нарисуем простейший график (рис. 3.10) , задав x (рис. 3.8) и y (рис. 3.9)

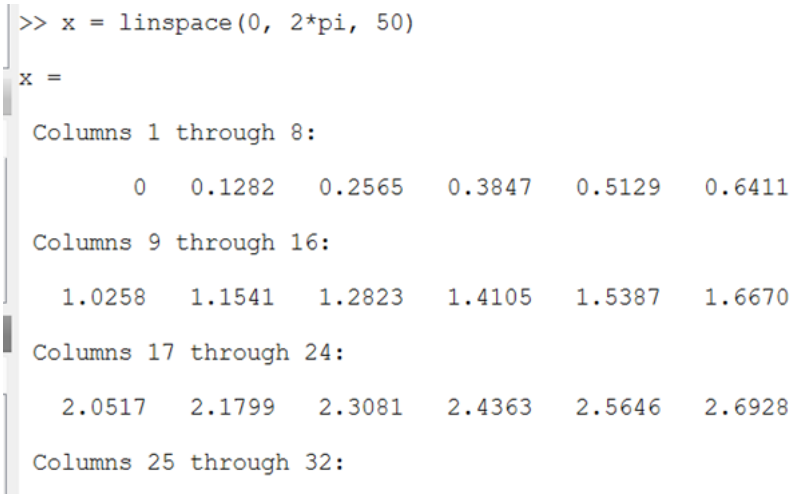

Figure 3.8: Значение x

```
>> y = sin(x)

y =

 Columns 1 through 8:

         0    0.1279    0.2537    0.3753    0.4907    0.5981

 Columns 9 through 16:

    0.8551    0.9144    0.9587    0.9872    0.9995    0.9954

 Columns 17 through 24:

    0.8866    0.8202    0.7403    0.6482    0.5455    0.4339

 Columns 25 through 32:
```

Figure 3.9: Значение y

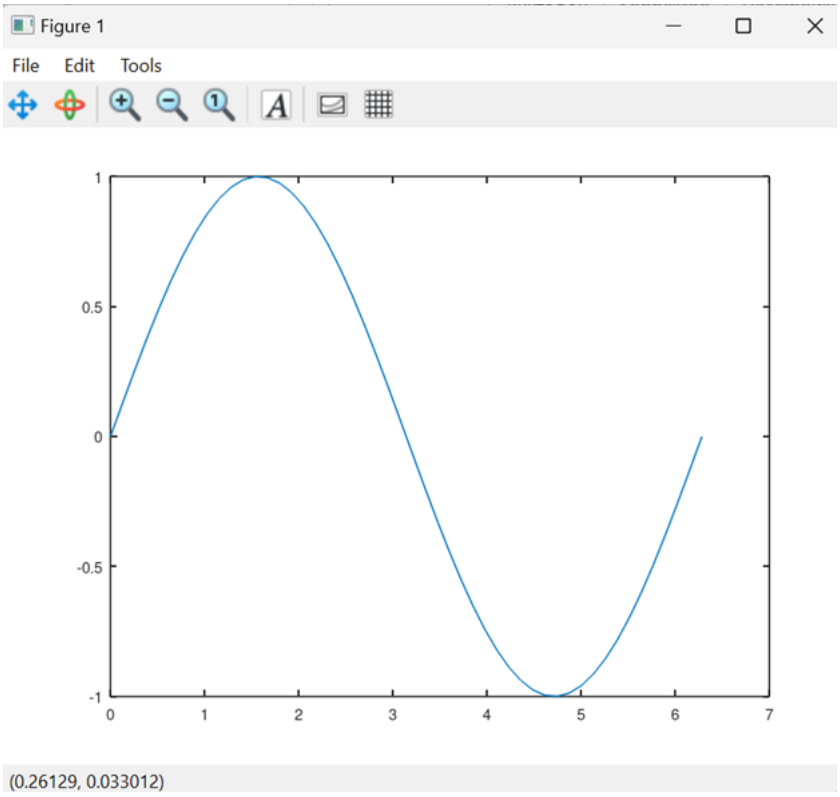

Figure 3.10: Простой график

Улучшим график, зададим красный цвет для линии и сделаем её толще. Поправим диапозон осей, нарисуем сетку, подпишем оси, сделаем заголовок графика, зададим легенду и в итоге получим такой график (рис. 3.11) (рис. 3.12)

```
>> plot(x,y)
>> clf
>> plot(x,y, 'r', 'linewidth', 3)
>> axis([0 2*pi -1 1])
>> grid on
>> xlabel('x')
>> ylabel('y')
>> title('Sine graph')
>> legend('y = sin(x)')
```

Figure 3.11: Код улучшения

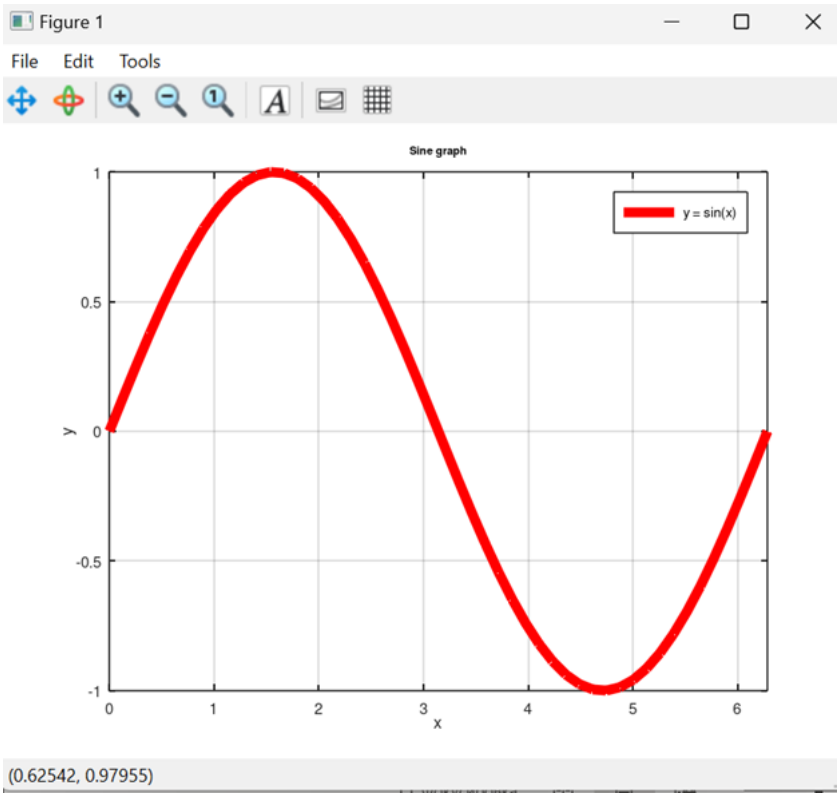

Figure 3.12: Улучшенный график

Теперь попробуем начертить два графика на одном чертеже. Для этого используем команду hold on и строим два графика: точки и регрессию. (рис. 3.13)

```
>> plot(x,y, 'o')
>> hold on
>> plot(x, 1.2*x)
>> grid on
>> axis([0 5 0 6])
>> legend('data points', 'regressionline')
>>
```

Figure 3.13: Код программы

Получаем такой график (рис. 3.14)

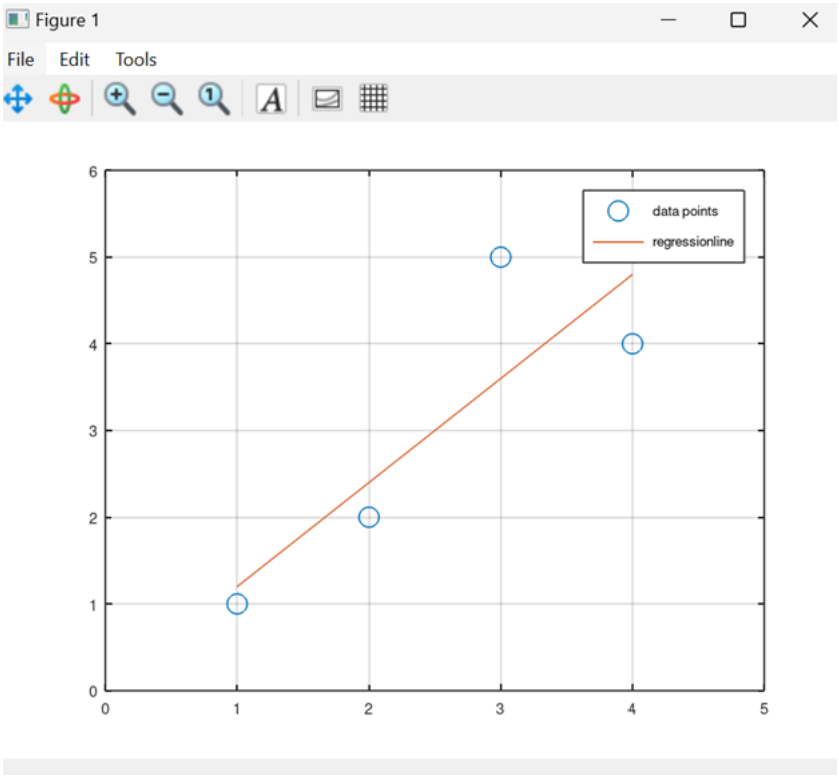

Figure 3.14: График двух функций

Теперь построим график сложной функции. Для начала попробуем задать постройку графика напрямую, однако тут же получим ошибку. Действительно, мы задали в выражении матричное умножение, однако нам необходимо поэлементное. Используем поэлементное возведение в степень .^ и поэлементное умножение .* (рис. 3.15)

```
>> plot(x, x^2*sin(x))

error: for x^y, only square matrix arguments are permitte
t be scalar.  Use .^ for elementwise power.
>> plot(x, x.^2.*sin(x))

>>
```

Figure 3.15: Код построения

В итоге получаем исправный график функции (рис. 3.16)

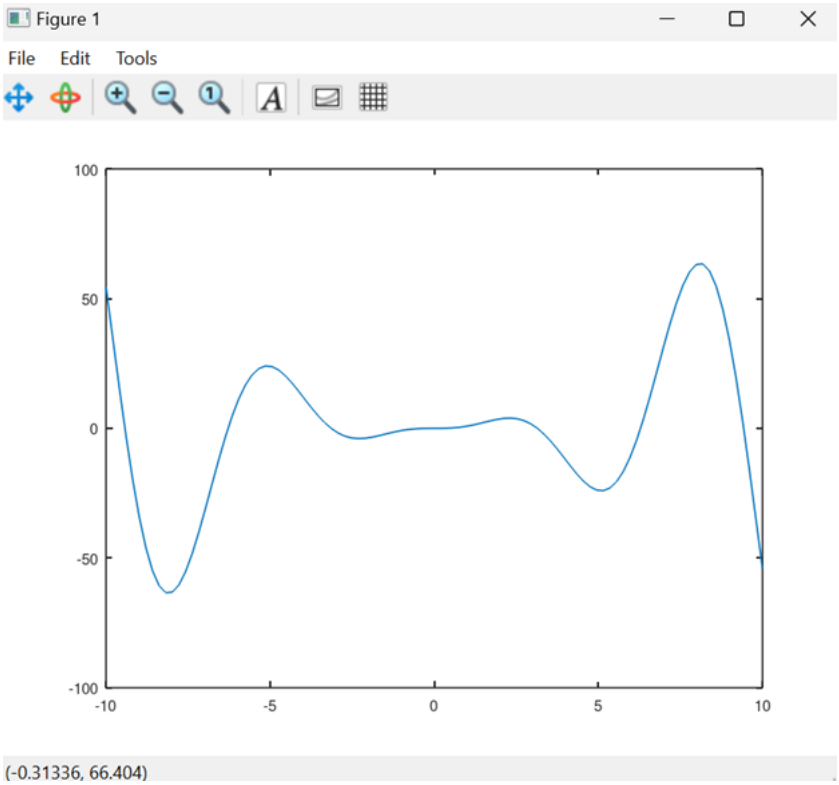

Figure 3.16: График сложной функции

Сохраним рисунок на компьютере в разных форматах (рис. 3.17)

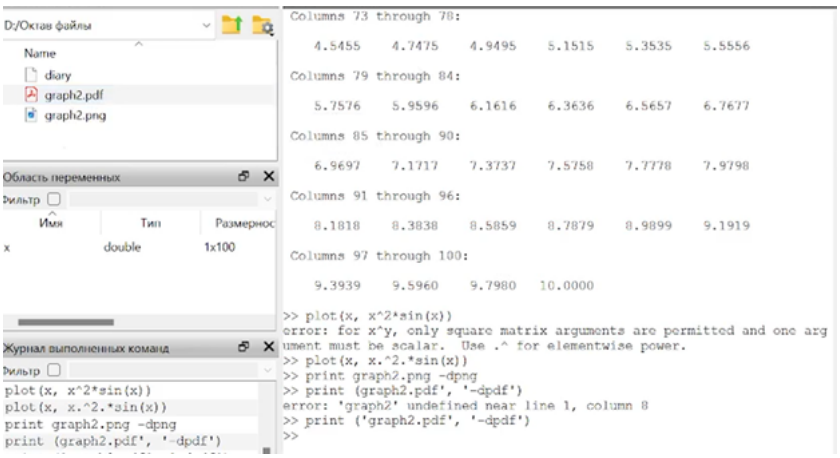

Figure 3.17: Сохранение данных

Приступим к выполнению последнего этапа: сравнение циклов и операций над векторами. Очистим память и проверим сумму чисел с помощью цикла. Так же добавим таймер, чтобы посмотреть, сколько времени понадобилось программе для реализации действия. (рис. 3.18)

```
tic
s = 0
for n = 1:1000000
s = s+1/n^2
end
toc
```

Figure 3.18: Код цикла

В итоге получим ответ (рис. 3.19)

```
s = 1.6449
s = 1.6449

>> toc
Elapsed time is 133.562 seconds.
>> # Octave 9.2.0, Wed Oct 09 00:08:59 2024 GMT <unl
```

Figure 3.19: Ответ цикла

Теперь посчитаем сумму при помощи векторов (рис. 3.20)

```
>> # Octave 9.2.0, Wed Oct 09 00:08:59 2024 GMT <unk
>> tic
>> clear(7-4)^2
>> tic
>> n = 1:1000000;
>> s = sum(1./n.^2)
s = 1.6449
>> toc
Elapsed time is 81.7055 seconds.
>> diary off
>>
```

Figure 3.20: Код и ответ

Как можно заметить, сумма равна одному и тому же значению, но на вторую операцию времени затрачено меньше. Следовательно, если есть возможность, то лучше осуществлять операции при помощи векторов.

Завершаем работу командой diary off, чтобы закрыть запись в файл. (рис. 3.20)

4 Выводы

Я изучил идеологию и применение языка Octave, познакомился с основными командами и возможностями языка.

5 Список литературы

Лабораторная работа №3

Лабораторная работа № 3. Введение в работу с Octave [Электронный ресурс]. 2019. URL:https://esystem.rudn.ru/pluginfile.php/2372902/mod_resource/content/3/003-octave-intro.pdf