

Лабораторная работа №3

Введение в работу с Octave

Смирнов-Мальцев Егор Дмитриевич

Содержание

Цель работы	4
Задание	5
Теоретическое введение	6
Выполнение лабораторной работы	7
Выводы	14
Список литературы	15

Список иллюстраций

1	Простейшие операции	7
2	Операции с векторами	8
3	Вычисление проектора	8
4	Операции с двумя матрицами	8
5	Операции с одной матрицей	9
6	Вычисление суммы	13

Цель работы

Научиться выполнять основные вычисления и рисовать простейшие двумерные графики с помощью системы для математических вычислений Octave.

Задание

- Выполнить простейшие операции.
- Выполнить операции с векторами.
- Выполнить матричные операции.
- Построить простейшие графики.
- Сравнить циклы и операции с векторами

Теоретическое введение

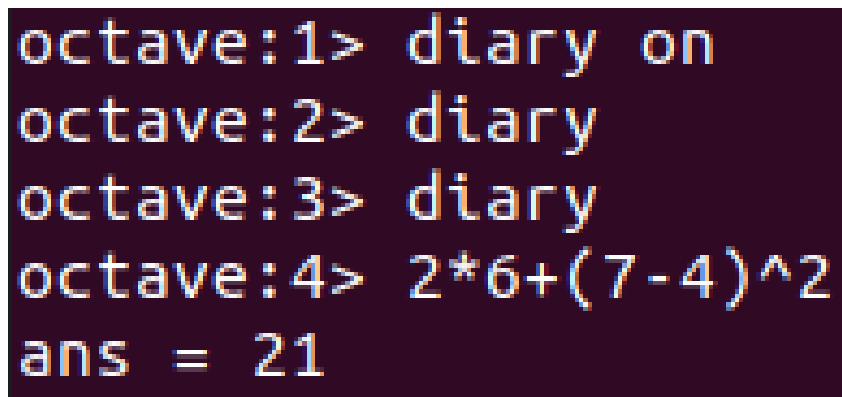
GNU Octave — это система математических вычислений, использующая совместимый с MATLAB язык высокого уровня `[@octave:bash]`.

По заявлениям разработчиков данная программа обладает следующими свойствами[]:

- Мощный синтаксис, ориентированный на математику, со встроенными инструментами 2D/3D-графики и визуализации.
- Бесплатное программное обеспечение, работающее на GNU/Linux, macOS, BSD и Microsoft Windows.
- Вставка, совместимая со многими скриптами Matlab

Выполнение лабораторной работы

Включим журналирование с помощью `diary on`. Затем воспользуемся Octave как простейшим калькулятором, вычислив выражение $2 * 6 + (7 - 4)^2$ (рис. [-@fig:001])



```
octave:1> diary on
octave:2> diary
octave:3> diary
octave:4> 2*6+(7-4)^2
ans = 21
```

Рис. 1: Простейшие операции

Зададим вектор-строку u . Затем создадим вектор-столбец u и матрицу A . Зададим ещё один вектор-столбец v и посчитаем $2*v + 3*u$. Перемножим эти векторы скалярно с помощью функции `dot()` и векторно с помощью функции `cross()`. Также найдём норму этих векторов функцией `norm()` (рис. [-@fig:002])

```

octave> u = [1 -4 6]
u =
    1   -4    6
octave> u = [1; -4; 6]
u =
     1
    -4
     6
octave> A = [1 2 -3; 2 4 0; 1 1 1]
A =
     1     2    -3
     2     4     0
     1     1     1
octave> v = [2; 1; -1]
v =
     2
     1
    -1
octave> 2*v+3*u
ans =
     7
    -10
    16
octave> dot(u,v)
ans = 8
octave> cross(u,v)
ans =
    -2
    -13
     9
octave> norm(u)
ans = 7.2801
octave> norm(v)
ans = 2.4500

```

Рис. 2: Операции с векторами

Введём два новых вектора-строки u и v и вычислим проекцию вектора u на вектор v (рис. [-@fig:003]):

```

octave> u=[3 5]
u =
     3     5
octave> v=[7 2]
v =
     7     2
octave> prj = dot(u,v)/norm(v)^2*v
prj =
    4.0943    1.1098

```

Рис. 3: Вычисление проектора

Введём матрицу B . Вычислим $A * B$, $B^T * A$ (рис. [-@fig:004])

```

octave> B = [1 2 3 4; 0 -2 -4 0; 1 -1 0 0]
B =
     1     2     3     4
     0    -2    -4     0
     1    -1     0     0
octave> A*B
ans =
    -2     1    -5    16
     2     4    10    32
     2    -1    -1    10
octave> B'*A
ans =
     2     3    -2
    -3    -5    -7
     3    10    -9
    16    32   -12

```

Рис. 4: Операции с двумя матрицами

Вычислим $2 * A - 4 * I$, где I единичная матрица. Затем найдём определитель матрицы A , обратную ей матрицу, собственные значения и ранг матрицы A (рис. [-@fig:005])


```

octave:22> 2*5-4*eye(3)
ans =
   -2    4   -6
    4    4    0
    2    2   -2

octave:23> det(A)
ans = 6

octave:24> inv(A)
ans =
   0.6667   0.3333   2.0000
  -0.3333   0.6667  -1.0000
   0.3333   0.3333    0.0000

octave:25> eig(A)
ans =
  4.5251 + 0i
  0.2374 + 0.8844i
  0.2374 - 0.8844i

octave:26> rank(A)
ans = 3

```

Рис. 5: Операции с одной матрицей

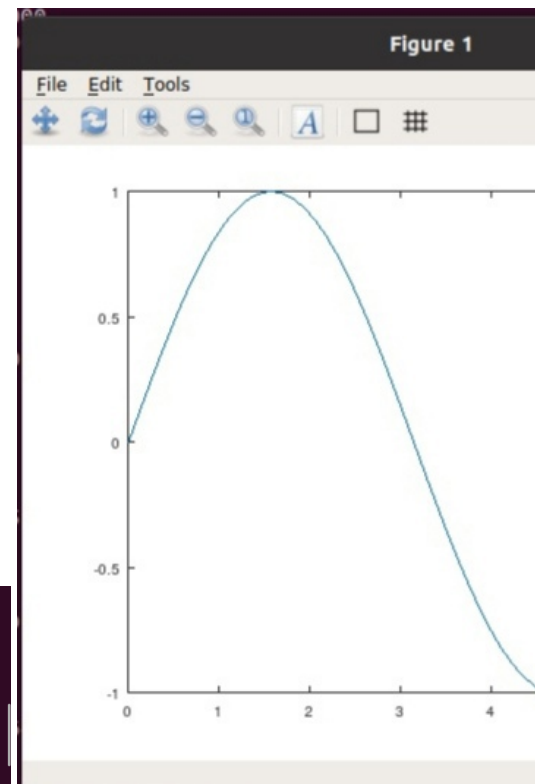
Создадим вектор значений x , зададим вектор $y = \sin x$ и построим график (рис. [-@fig:006], [-@fig:007])

```

octave:27> x = linspace(0, 2*pi, 50);
octave:28> y = sin(x)
y =
Columns 1 through 11:
    0   0.1229   0.2537   0.3753   0.4907   0.5981   0.6957   0.7818   0.8551   0.9144   0.9587
Columns 12 through 22:
   0.9872   0.9995   0.9954   0.9749   0.9385   0.8866   0.8202   0.7403   0.6402   0.5455   0.4339
Columns 23 through 33:
   0.3151   0.1912   0.0641  -0.0641  -0.1912  -0.3151  -0.4339  -0.5455  -0.6402  -0.7403  -0.8202
Columns 34 through 44:
  -0.8866  -0.9385  -0.9749  -0.9954  -0.9995  -0.9872  -0.9587  -0.9144  -0.8551  -0.7818  -0.6957
Columns 45 through 50:
  -0.5981  -0.4907  -0.3753  -0.2537  -0.1229  -0.0000

octave:29> plot(x,y)

```

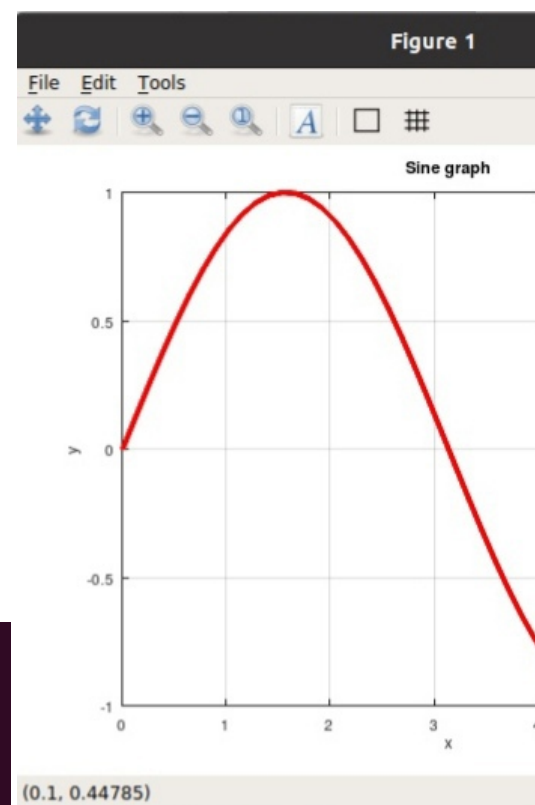


Улучшим внешний вид графика. Изменим цвет и ширину линии, подгоним диапазон осей, нарисуем сетку, подпишем оси, сделаем заголовок графика и зададим легенду (рис. [-@fig:008], [-@fig:009])

```

octave:31> plot(x,y,'r','linewidth',3)
octave:32> axis([0 2*pi -1 1])
octave:33> grid on
octave:34> xlabel('x')
octave:35> ylabel('y')

```

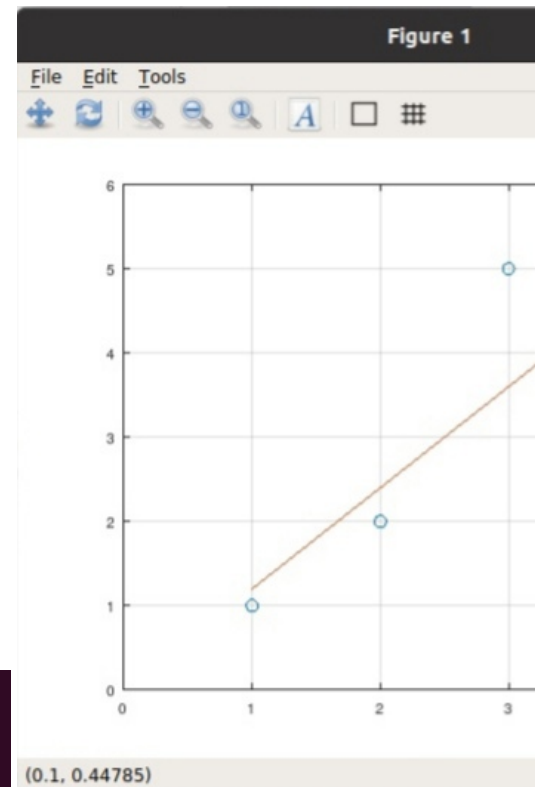


Начертим два графика на одном чертеже. Очистим память и рабочую область фигуры. Зададим два вектора и начертим эти точки, используя кружочки как маркеры. Чтобы добавить к нашему текущему графику ещё один, используем команду `hold on`. Добавим график регрессии, зададим сетку, оси и легенду. (рис. [-@fig:010], [-@fig:011])

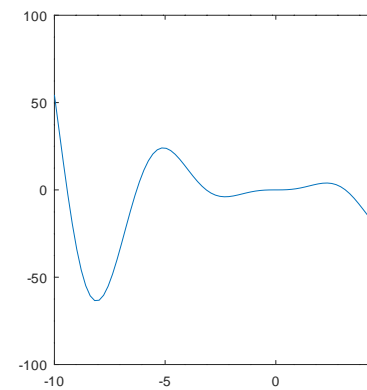
```

octave:42> plot(x,y,'o')
octave:43> hold on
octave:44> plot(x,1.2*x)
octave:45> grid on
octave:46> axis([0 5 0 6])
octave:47> legend('data points', 'regression line');

```



Очистим память и рабочую область фигуры. Создадим вектор x и попробуем построить график $y = x^2 * \sin(x)$. С помощью команды `plot(x, x^2*sin(x))` сделать это не получится, так как ей задаётся матричное умножение, а нам нужно поточечное. Сохраним графики в виде файлов, в результате получим следующий график (рис. [-@fig:012], [-@fig:013])



```
octave:42> plot(x,y,'o')
octave:43> hold on
octave:44> plot(x,1.2*x)
octave:45> grid on
octave:46> axis([0 5 0 6])
octave:47> legend('data points', 'regression line');
octave:48> clear
octave:49> clf
octave:50> x = linspace(-10, 10, 100);
octave:51> plot(x, x^2*sin(x))
error: for x*y, only square matrix arguments are permitted and one argument must be scalar. Use .^ for elementwise power.
octave:52> plot(x, x.^2.*sin(x))
octave:53> print graph2.png -dpng
octave:54> print ('graph2.pdf' '-dpdf')
```

Сравним эффективность работы с циклами и операций с векторами. Для этого вычислим сумму

$$\sum_n^{10000000} 1/n^2$$

с помощью цикла (программа loop_for.m) и с помощью операций с векторами (программа loop_vec.m). При сравнении обнаружим, что вычисление через векторы значительно быстрее. (рис. [-@fig:014])

```
octave:63> loop_for  
Elapsed time is 0.126218 seconds.  
octave:64> s  
s = 1.6449  
octave:65> loop_vec  
Elapsed time is 0.00279093 seconds.  
octave:66> s  
s = 1.6449
```

Рис. 6: Вычисление суммы

Выводы

В результате выполнения работы научились выполнять основные вычисления и рисовать простейшие двумерные графики с помощью системы Octave.

Список литературы