# РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

## Факультет физико-математических и естественных наук

## Кафедра прикладной информатики и теории вероятностей

#### ОТЧЕТ ПО

#### ЛАБОРАТОРНОЙ РАБОТЕ №5

*дисциплина: Научное программирование*

Студент: Хиссен Али Уэддей
Группа: НПМмд-02-20
Ст. билет № 1032209306

#### Цель работы

Ознакомление с некоторыми операциями в среде Octave для решения таких задач, как подгонка полиномиальной кривой, матричных преобразований, вращений, отражений и дилатаций.

#### Ход работы

#### Подгонка полиномиальной кривой

В статистике часто рассматривается проблема подгонки прямой линии к набору данных. Решим более общую проблему подгонки полинома к множеству точек. Пусть нам нужно найти параболу по методу наименьших квадратов для набора точек, заданных матрицей

$$ D = \left( \begin{array}{cc} 1 & 1 \ 2 & 2 \ 3 & 5 \ 4 & 4 \ 5 & 2 \ 6 & -3 \end{array} \right) $$

В матрице заданы значения $x$ в столбце 1 и значения $y$ в столбце 2. Введём матрицу данных в Octave и извлечём вектора $x$ и $y$. Данные операции показаны на Рис. 1.

Командное окно

```
>>  D = [ 1 1 ; 2 2 ; 3 5 ; 4 4 ;
D =

   1   1
   2   2
   3   5
   4   4
   5   2
   6  -3

>> xdata = D(:,1)
xdata =

   1
   2
   3
   4
   5
   6

>>  ydata = D(:,2)
ydata =

   1
   2
   5
   4
   2
  -3
```

Рис.1 Ввод матрицы данных

Нарисуем точки на графике, см. Рис. 2.

```
>> plot(xdata,ydata,'o-')
```

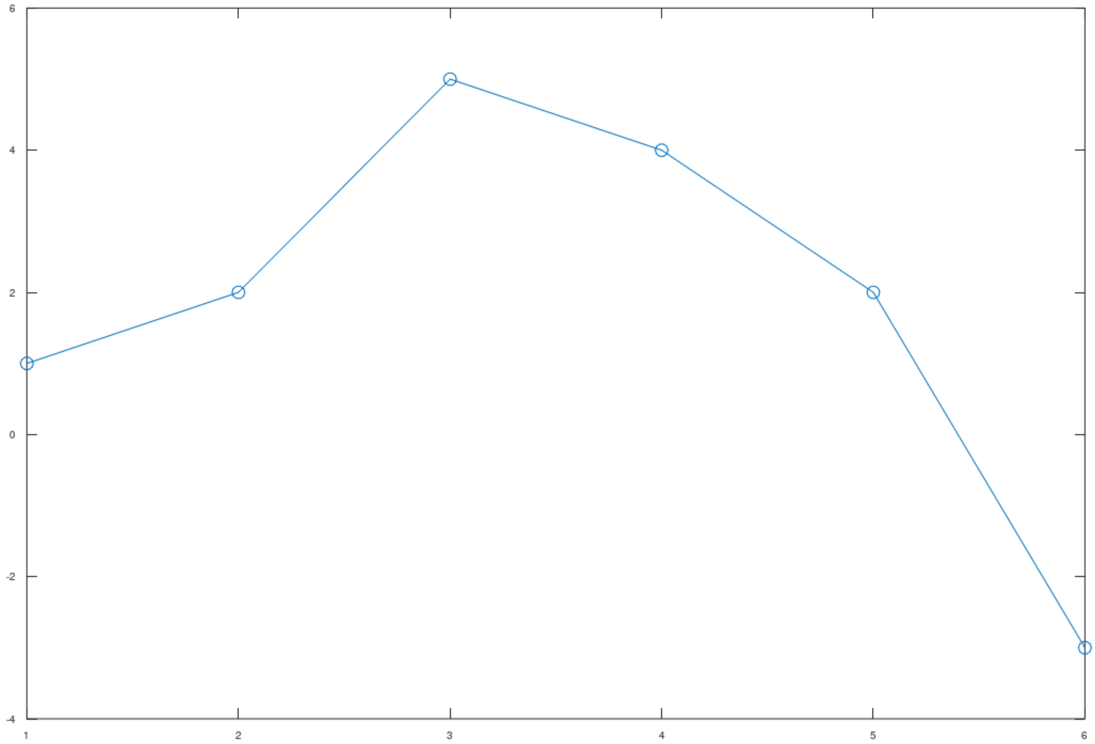


Рис.2 Нанесение точек на плоскость

Построим уравнение вида $y = ax^2 + bx + c$. Подставляя данные, получаем следующую систему линейных уравнений.

# $$ \left( \begin{array}{ccc} 1 & 1 & 1 \ 4 & 2 & 1 \ 9 & 3 & 1 \ 16 & 4 & 1 \ 25 & 5 & 1 \ 36 & 6 & 1 \end{array} \right) \left( \begin{array}{c} a \ b \ c \end{array} \right)

\left( \begin{array}{c} 1 \ 2 \ 5 \ 4 \ 2 \ -3 \end{array} \right). $$

Обратим внимание на форму матрицы коэффициентов $A$. Третий столбец – все единицы, второй столбец – значения $x$, а первый столбец – квадрат значений $x$. Правый вектор – это значения $y$. Есть несколько способов построить матрицу коэффициентов в Octave. Один из подходов состоит в том, чтобы использовать команду ones для создания матрицы единиц соответствующего размера, а затем перезаписать первый и второй столбцы необходимыми данными. Это показано на Рис. 3.

```
>> A = ones(6,3)
A =

   1   1   1
   1   1   1
   1   1   1
   1   1   1
   1   1   1
   1   1   1

>>  A(:,1) = xdata .^ 2
A =

    1    1    1
    4    1    1
    9    1    1
   16    1    1
   25    1    1
   36    1    1

>>  A(:,2) = xdata
A =

    1    1    1
    4    2    1
    9    3    1
   16    4    1
   25    5    1
   36    6    1
```

Рис.3 Создание матрицы A

Решение по методу наименьших квадратов получается из решения уравнения $A^T Ab = A^T b$, где $b$ – вектор коэффициентов полинома. Используем Octave для построения уравнений, как показано на Рис. 4

```
>> A'*A
ans =

   2275    441     91
    441     91     21
     91     21      6

>>  A' * ydata
ans =

   60
   28
   11
```

Рис.4 Построение уравнений по методу наименьших квадратов

Решим задачу методом Гаусса (См. Рис.5). Для этого запишем расширенную матрицу:

$$ B = \left( \begin{array}{cccc} 2275 & 441 & 91 & 60 \\ 441 & 91 & 21 & 28 \\ 91 & 21 & 6 & 11 \end{array} \right). $$

Таким образом, искомое квадратное уравнение имеет вид

$$ y = -0.89286 x^2 + 5.65 x - 4.4 $$

```
>>  B = A' * A;
>> B (:,4) = A' * ydata;
>>  B_res = rref (B)
B_res =

   1.00000   0.00000   0.00000  -0.89286
   0.00000   1.00000   0.00000   5.65000
   0.00000   0.00000   1.00000  -4.40000

>>  a1=B_res(1,4)
a1 = -0.89286
>>  a2=B_res(2,4)
a2 =  5.6500
>> a3=B_res(3,4)
a3 = -4.4000
```

Рис.5 Решение задачи методом Гаусса

После чего построим соответствующий график параболы. Построение можно увидеть на Рисунке 6, а вид самой параболы на рисунке 7.

```
>>  x = linspace (0,7,50);
>>  y = a1 * x .^ 2 + a2 * x + a3;
>> plot (xdata,ydata, 'o' ,x,y, 'linewidth', 2)
>>  grid on;
>>  legend ('data values', 'least-squares parabola')
>> title ('y = -0.89286 x^2 + 5.65 x - 4.4')
```

Рис.6 Построение графика параболы

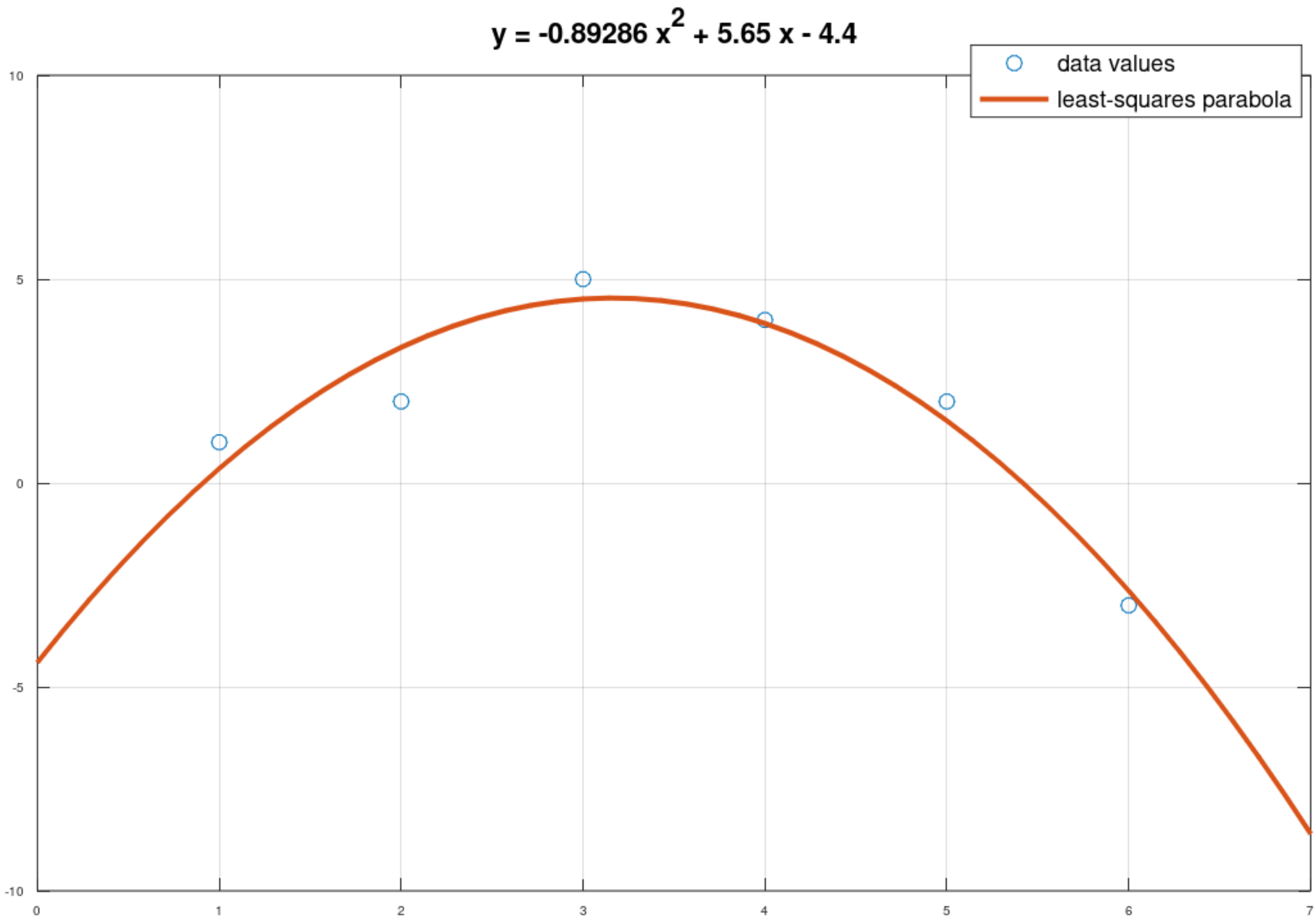


Рис.7 График параболы

Процесс подгонки может быть автоматизирован встроенными функциями Octave. Для этого мы можем использовать встроенную функцию для подгонки полинома polyfit. Синтаксис: polyfit (x, y, order), где order – это степень полинома. Значения полинома P в точках, задаваемых вектором-строкой x можно получить с помощью функции polyval. Синтаксис: polyval (P, x).

На рисунке 8 получим подгоночный полином.

```
>> P = polyfit (xdata, ydata, 2)
P =

  -0.89286   5.65000  -4.40000

>> y = polyval (P,xdata)
y =

   0.35714
   3.32857
   4.51429
   3.91429
   1.52857
  -2.64286

>>  plot(xdata,ydata,'o-',xdata,y,'+-')
>> grid on ;
>>  legend ('original data' , 'polyfit data' ) ;
```

Рис.8 Подгоночный полином

После чего рассчитаем значения в точках и построим исходные данные. Это показано на Рис.9.

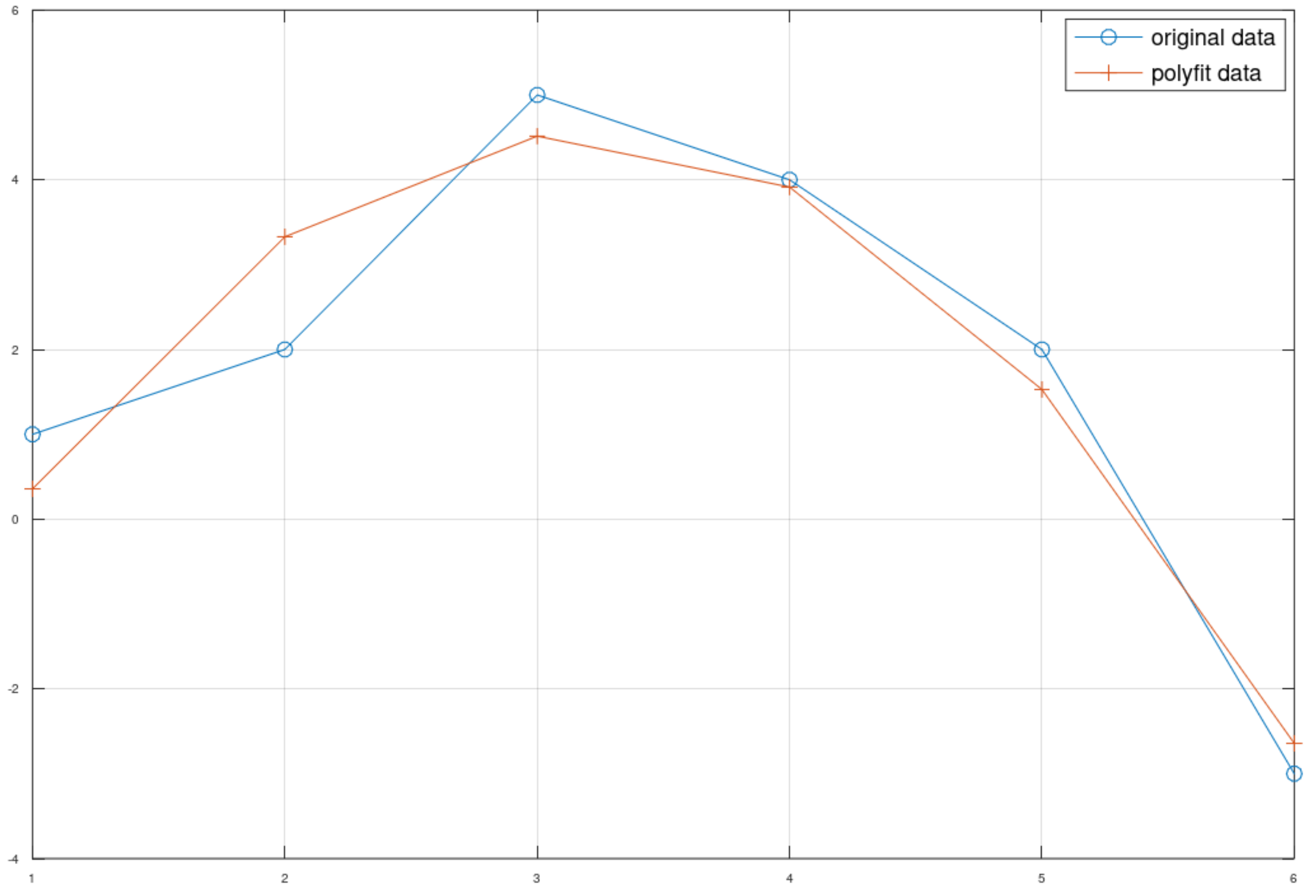


Рис.9 Граф исходных и подгоночных данных

**Матричные преобразования**
Матрицы и матричные преобразования играют ключевую роль в компьютерной графике. Существует несколько способов представления изображения в виде матрицы. Подход, который мы здесь используем, состоит в том, чтобы перечислить ряд вершин, которые соединены последовательно, чтобы получить ребра простого графа. Мы записываем это как матрицу $2 \times n$, где каждый столбец представляет точку на рисунке. В качестве простого примера, давайте попробуем закодировать граф-домик. Есть много способов закодировать это как матрицу. Эффективный метод состоит в том, чтобы выбрать путь, который проходит по каждому ребру ровно один раз (цикл Эйлера).

$$ D = \left( \begin{array}{ccccccc} 1 & 1 & 3 & 3 & 2 & 1 & 3 \ 2 & 0 & 0 & 2 & 3 & 2 & 2 \end{array} \right). $$

Реализация показана на рисунке 10.

```
>>  D = [ 1 1 3 3 2 1 3 ; 2 0 0 2 3 2 2 ]
D =

   1   1   3   3   2   1   3
   2   0   0   2   3   2   2

>>  x = D(1,:)
x =

   1   1   3   3   2   1   3

>>  y = D(2,:)
y =

   2   0   0   2   3   2   2

>>  plot (x,y)
```

Рис.10 Реализация построения графа

Полученный граф можно увидеть на рисунке 11.

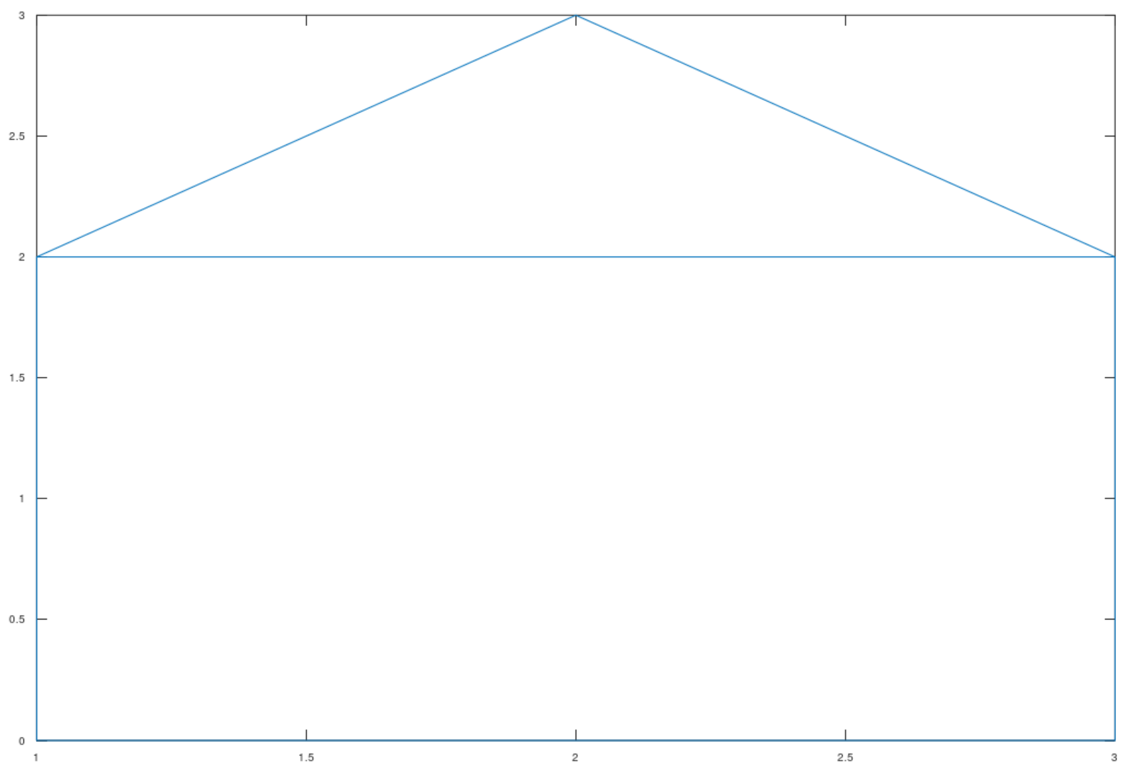


Рис.11 Полученный граф.

**Вращение**

Рассмотрим различные способы преобразования изображения. Вращения могут быть получены с использованием умножения на специальную матрицу. Вращение точки $(x, y)$ относительно начала координат определяется как

$$ R \left( \begin{array}{c} x \ y \end{array} \right), $$

где

$$ R = \left( \begin{array}{cc} cos(\theta) & -sin(\theta) \ sin(\theta) & cos(\theta) \end{array} \right), $$

$\theta$ - угол поворота (измеренный против часовой стрелки).

Теперь, чтобы произвести повороты матрицы данных $D$, нам нужно вычислить произведение матриц $RD$. Повернём граф дома на $90^{\circ}$ и $225^{\circ}$. Вначале переведём угол в радианы. Произведенные действия показаны на рисунках 12 - 14.

```
>> theta1 = 90*pi/180
theta1 =  1.5708
>>  R1 = [cos(theta1) -sin(theta1); sin(theta1) cos(theta1)]
R1 =

   6.1232e-17  -1.0000e+00
   1.0000e+00   6.1232e-17

>> RD1 = R1*D
RD1 =

  -2.0000e+00   6.1232e-17   1.8370e-16  -2.0000e+00  -3.0000e+00  -2.0000e+00
   1.0000e+00   1.0000e+00   3.0000e+00   3.0000e+00   2.0000e+00   1.0000e+00

>> x1 = RD1(1,:)
x1 =

  -2.0000e+00   6.1232e-17   1.8370e-16  -2.0000e+00  -3.0000e+00  -2.0000e+00

>>  y1 = RD1(2,:)
y1 =

   1.0000   1.0000   3.0000   3.0000   2.0000   1.0000   3.0000
```

Рис.12 Поворот на 90 градусов

```
>> theta2 = 225*pi/180
theta2 =  3.9270
>> R2 = [cos(theta2) -sin(theta2); sin(theta2) cos(theta2)]
R2 =

  -0.70711   0.70711
  -0.70711  -0.70711

>> RD2 = R2*D
RD2 =

   0.70711  -0.70711  -2.12132  -0.70711   0.70711   0.70711  -0.70711
  -2.12132  -0.70711  -2.12132  -3.53553  -3.53553  -2.12132  -3.53553

>> x2 = RD2(1,:)
x2 =

   0.70711  -0.70711  -2.12132  -0.70711   0.70711   0.70711  -0.70711

>>  y2 = RD2(2,:)
y2 =

  -2.12132  -0.70711  -2.12132  -3.53553  -3.53553  -2.12132  -3.53553
```

Рис.13 Поворот на 225 градусов

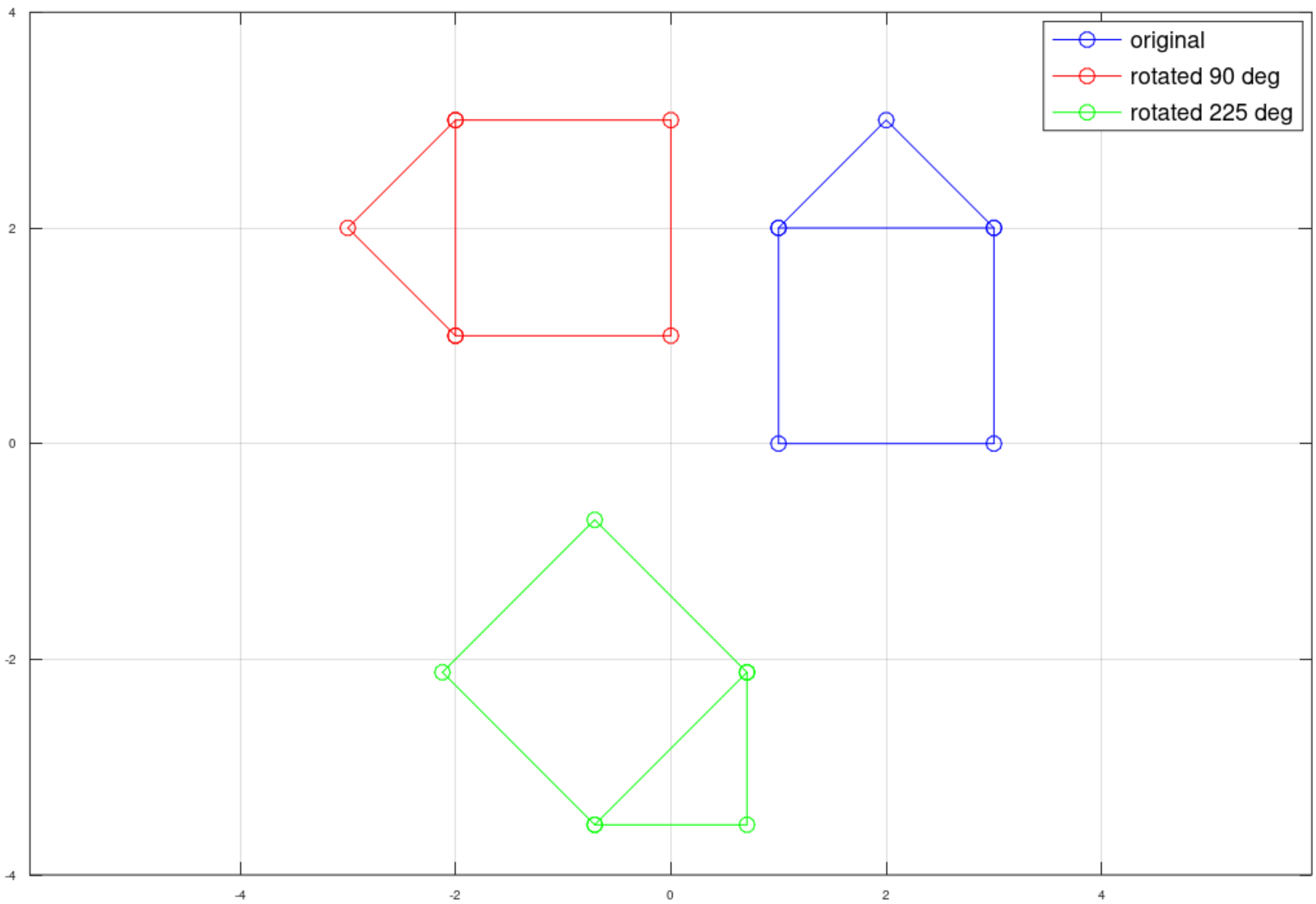

Рис.14 Результаты вращения

**Отражение**

Если $l$ – прямая, проходящая через начало координат, то отражение точки $(x, y)$ относительно прямой $l$ определяется как

$$ R \left( \begin{array}{c} x \ y \end{array} \right), $$

где

$$ R = \left( \begin{array}{cc} cos(2\theta) & sin(2\theta) \ sin(2\theta) & -cos(2\theta) \end{array} \right), $$

$\theta$ - угол между прямой $l$ и осью абсцисс (измеренный против часовой стрелки). Отразим граф дома относительно прямой $y = x$. Зададим матрицу отражения, как показано на рисунке 15.

```
>> R = [0 1; 1 0]
R =

   0   1
   1   0

>>  RD = R * D
RD =

   2   0   0   2   3   2   2
   1   1   3   3   2   1   3

>> x1 = RD(1,:)
x1 =

   2   0   0   2   3   2   2

>> y1 = RD(2,:)
y1 =

   1   1   3   3   2   1   3

>> plot (x,y,'o-',x1,y1,'o-')
>>  axis([-1 4 -1 4], 'equal');
>>  axis([-1 5 -1 5], 'equal');
>> grid on ;
>> legend ( 'original' , 'reflected' )
```

Рис.15 Задание отражения

Далее на рисунке 16 показано, какой результат получился в ходе этих действий.

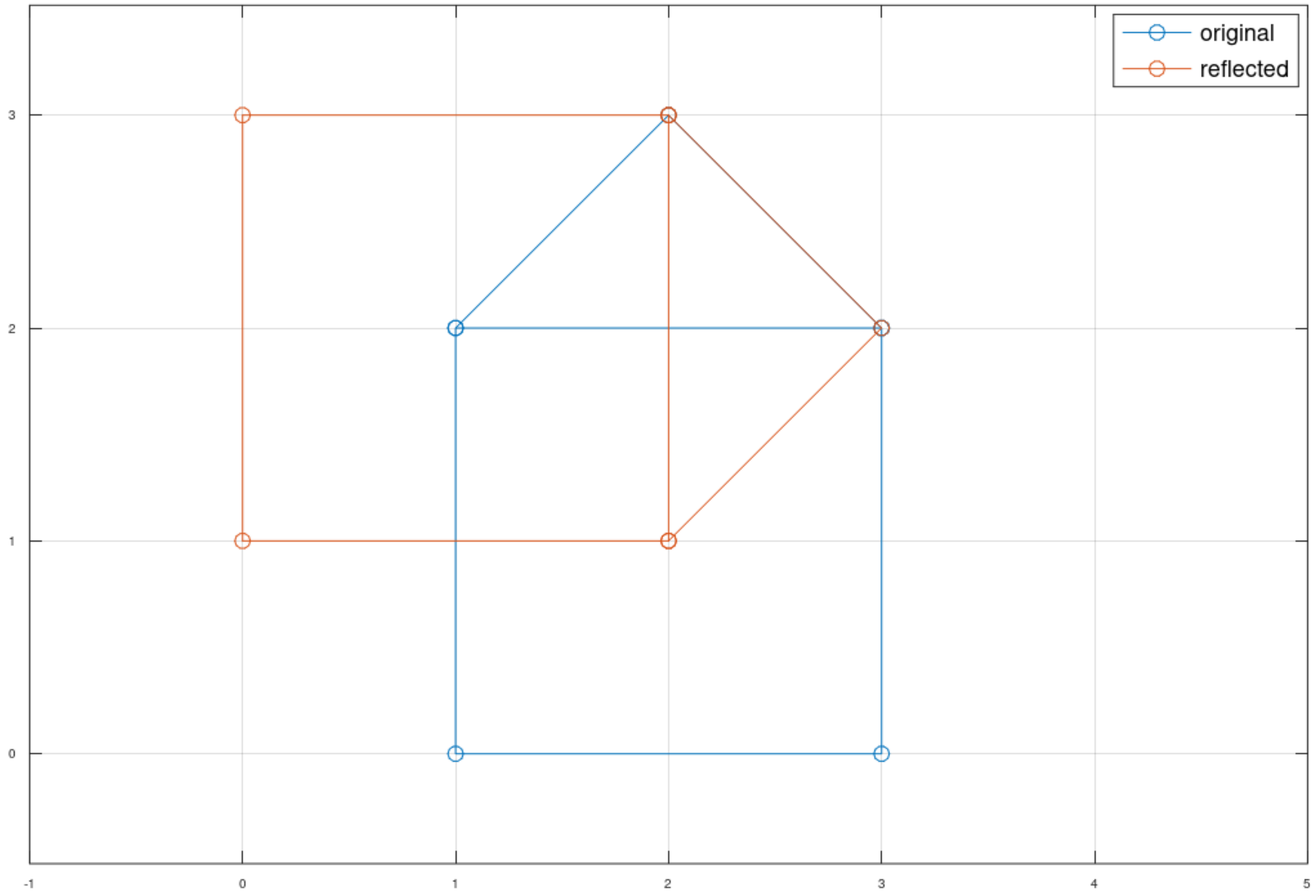


Рис.16 Результат отражения

**Дилатация**

Дилатация (то есть расширение или сжатие) также может быть выполнено путём умножения матриц. Пусть

$$ T = \left( \begin{array}{cc} k & 0 \ 0 & k \end{array} \right), $$

Тогда матричное произведение $TD$ будет преобразованием дилатации $D$ с коэффициентом $k$. Увеличим граф дома в 2 раза. Реализация выполнена на Рисунке 17.

```
>> T = [2 0; 0 2]
T =

   2   0
   0   2

>> TD = T*D;
>>  x1 = TD(1,:); y1 = TD(2,:);
>> plot (x, y, 'o-', x1, y1,'o-')
>>  axis ([-1 7 -1 7], 'equal');
>>  grid on;
>> legend ('original', 'expanded')
```

Рис.17 Реализация дилатации

После чего на рисунке 18 можно увидеть результат данной операции.

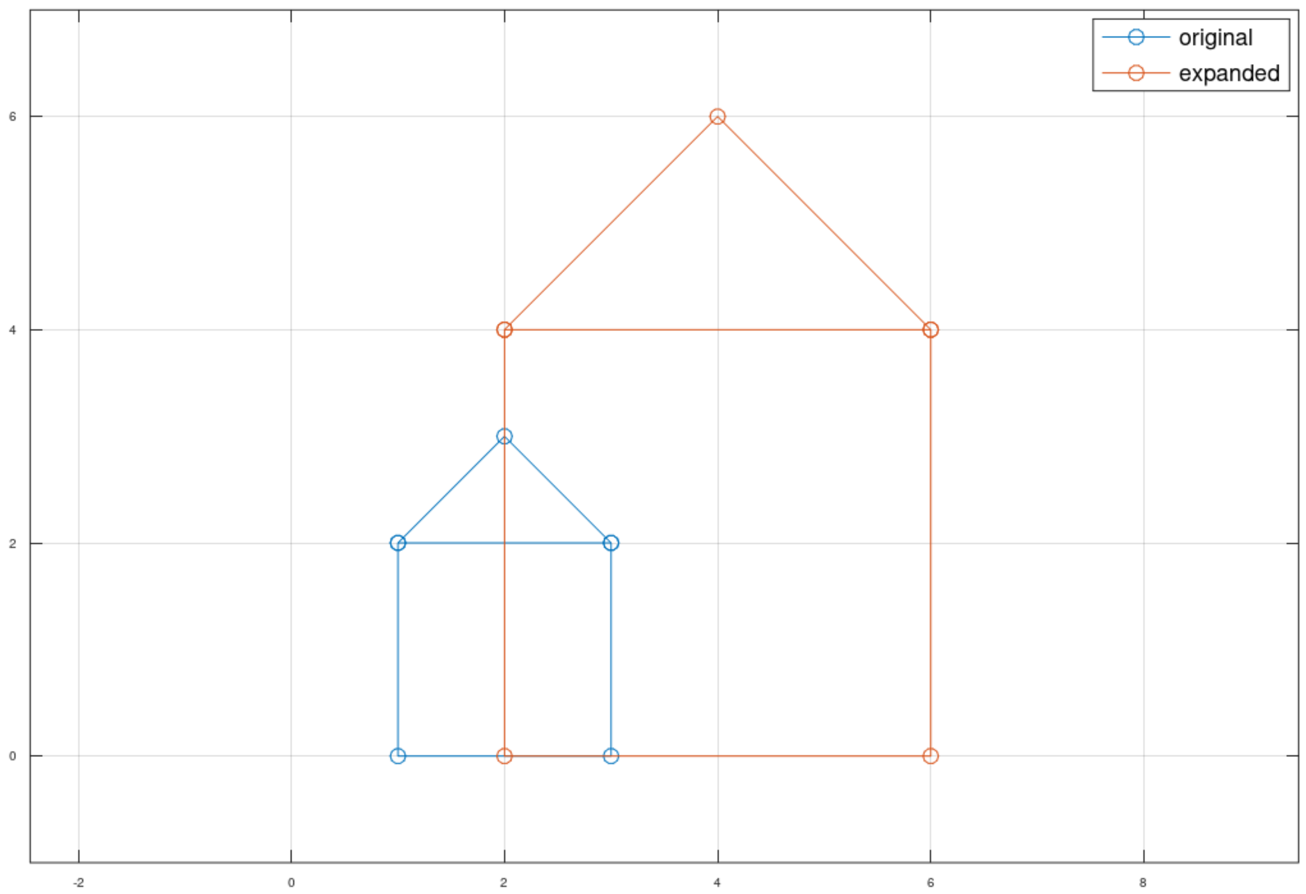


Рис.18 Результат увеличения

**Вывод**

В ходе выполнения данной работы я ознакомилась с некоторыми операциями в среде Octave для решения таких задач, как подгонка полиномиальной кривой, матричных преобразований, вращений, отражений и дилатаций.