Отчёт по лабораторной работе 5

Супонина Анастасия Павловна

Содержание

Список иллюстраций

Элементы списка иллюстраций не найдены.

Список таблиц

Элементы списка иллюстраций не найдены.

Цель работы

Ознакомиться с подгонкой полиномиальной кривой, а также с различными матричными преобразованиями в Octave. Научиться вращать, отражать и дилатировать изображения на графике.

Теоретическая часть.

Подгонка полиномиальной кривой

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).**

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 × □, 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).

1. 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

$$R\begin{bmatrix} x \\ y \end{bmatrix}$$

где,

$$R\begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix}$$

θ – угол поворота (измеренный против часовой стрелки). Теперь, чтобы произвести повороты матрицы данных D, нам нужно вычислить произведение матриц RD.

2. Отражение

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

$$R\begin{bmatrix} x \\ y \end{bmatrix}$$

где

$$R\begin{bmatrix} cos(2\theta) & sin(2\theta) \\ sin(2\theta) & -cos(2\theta) \end{bmatrix}$$

θ – угол между прямой l и осью абсцисс (измеренный против часовой стрелки).

3. Дилатация

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

$$T\begin{bmatrix} k & 0 \\ 0 & k \end{bmatrix}$$

Тогда матричное произведение TD будет преобразованием дилатации D с коэффициентом k.

Задание.

1) Выполнить подгонку полиномиальной кривой
2) Провести матричные преобразования
    1. Вращение
    2. Отражение
    3. Дилатация

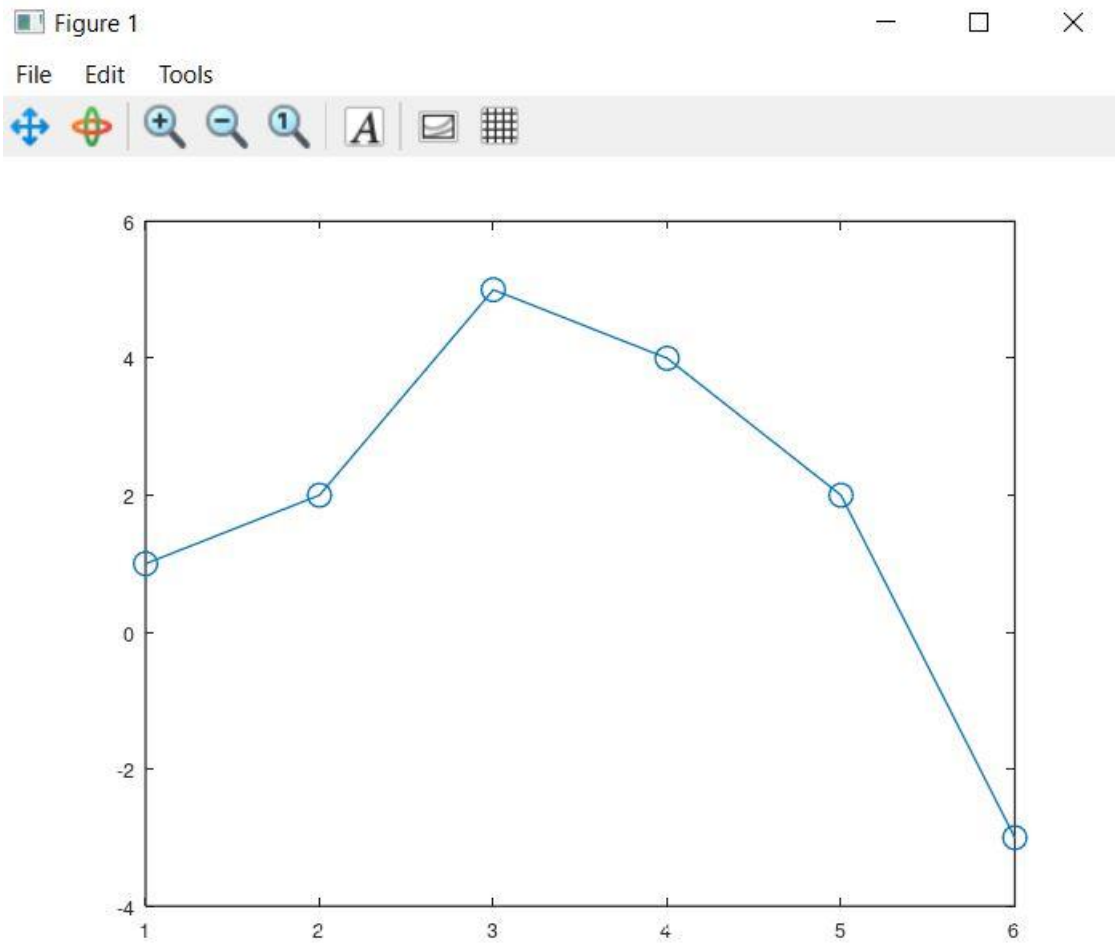
Выполнение работы

Подгонка полиномиальной кривой

Ввожу матрицу и извлекаю из неё значения x и y

```
>> D = [ 1 1 ; 2 2 ; 3 5 ; 4 4 ; 5 2 ; 6 -3]
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

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

При помощи полученных значений строю график



Строю из матриц уравнение вида

$$y = ax^2 + bx + c$$

.

Записываю исходную матрицу и решаю по методу наименьших квадратов.

```
>> 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

>> A'*A
ans =

   2275    441     91
    441     91     21
     91     21      6

>> A' * ydata
ans =

   60
   28
   11
```

Решаю задачу методом Гаусса, используя встроенную функцию rref. В конце выписывая нужные значения в отдельные переменные для отображения на графике

```
>> B = A'*A
B =

   2275    441     91
    441     91     21
     91     21      6

>>  B (:,4) = A' * ydata
B =

   2275    441     91     60
    441     91     21     28
     91     21      6     11

>> B_res = rref(B)
B_res =

   1.0000         0         0   -0.8929
        0    1.0000         0    5.6500
        0         0    1.0000   -4.4000

>> a1=B_res(1,4)
a1 = -0.8929
>> a2=B_res(2,4)
a2 = 5.6500
>> a3=B_res(3,4)
a3 = -4.4000
```

Строю график параболы:

Задаю значения для x

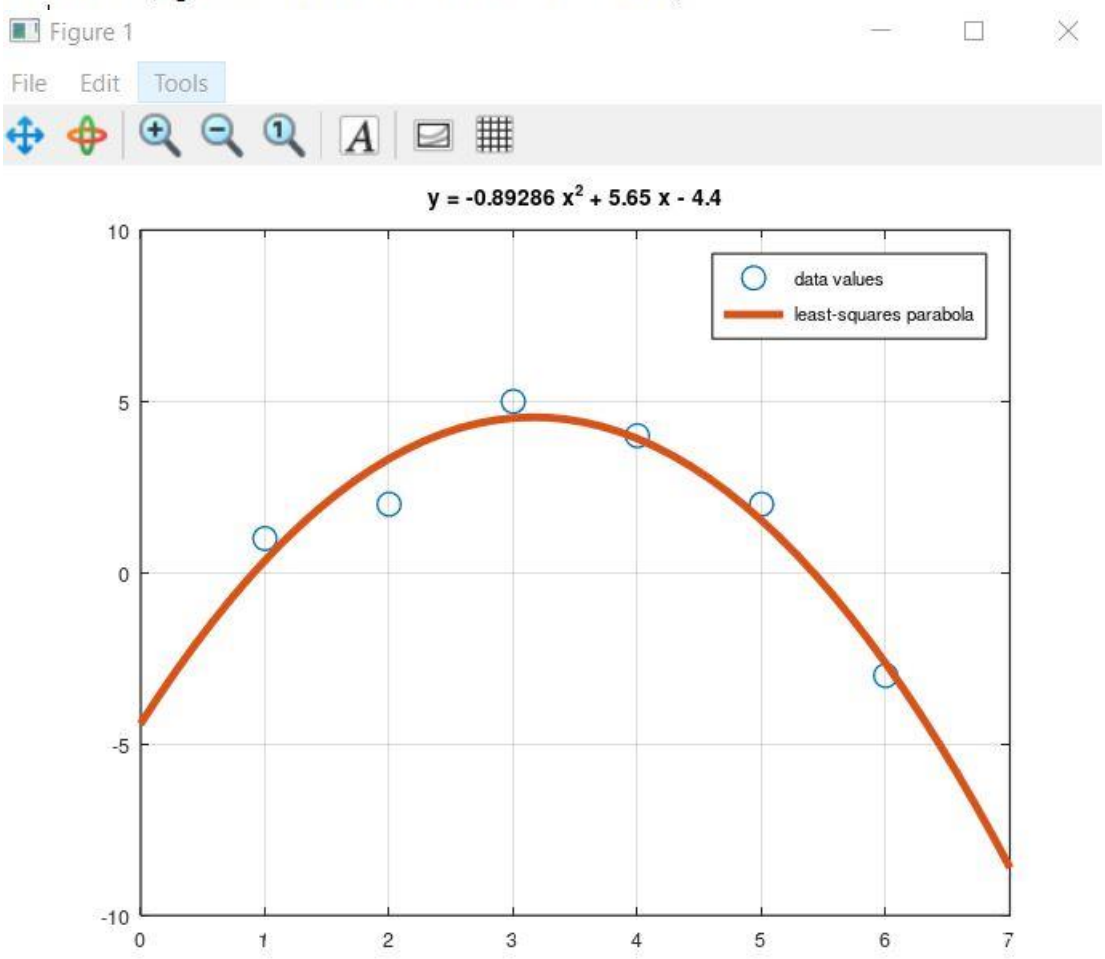
```
>> x = linspace (0,7,50)
x =

 Columns 1 through 10:

        0   0.1429   0.2857   0.4286   0.5714   0.7143   0.8571   1.0000   1.1429   1.2857

 Columns 11 through 20:

   1.4286   1.5714   1.7143   1.8571   2.0000   2.1429   2.2857   2.4286   2.5714   2.7143

 Columns 21 through 30:

   2.8571   3.0000   3.1429   3.2857   3.4286   3.5714   3.7143   3.8571   4.0000   4.1429

 Columns 31 through 40:

   4.2857   4.4286   4.5714   4.7143   4.8571   5.0000   5.1429   5.2857   5.4286   5.5714

 Columns 41 through 50:

   5.7143   5.8571   6.0000   6.1429   6.2857   6.4286   6.5714   6.7143   6.8571   7.0000
```

Считаю соответствующие каждому x, значения для y

```
>> y = a1 * x .^ 2 + a2 * x + a3
y =

 Columns 1 through 8:

  -4.400000  -3.611079  -2.858601  -2.142566  -1.462974  -0.819825  -0.213120   0.357143

 Columns 9 through 16:

   0.890962   1.388338   1.849271   2.273761   2.661808   3.013411   3.328571   3.607289

 Columns 17 through 24:

   3.849563   4.055394   4.224781   4.357726   4.454227   4.514286   4.537901   4.525073

 Columns 25 through 32:

   4.475802   4.390087   4.267930   4.109329   3.914286   3.682799   3.414869   3.110496

 Columns 33 through 40:

   2.769679   2.392420   1.978717   1.528571   1.041983   0.518950  -0.040525  -0.636443

 Columns 41 through 48:

  -1.268805  -1.937609  -2.642857  -3.384548  -4.162682  -4.977259  -5.828280  -6.715743

 Columns 49 and 50:

  -7.639650  -8.600000
```

Рисую график с полученными значениями

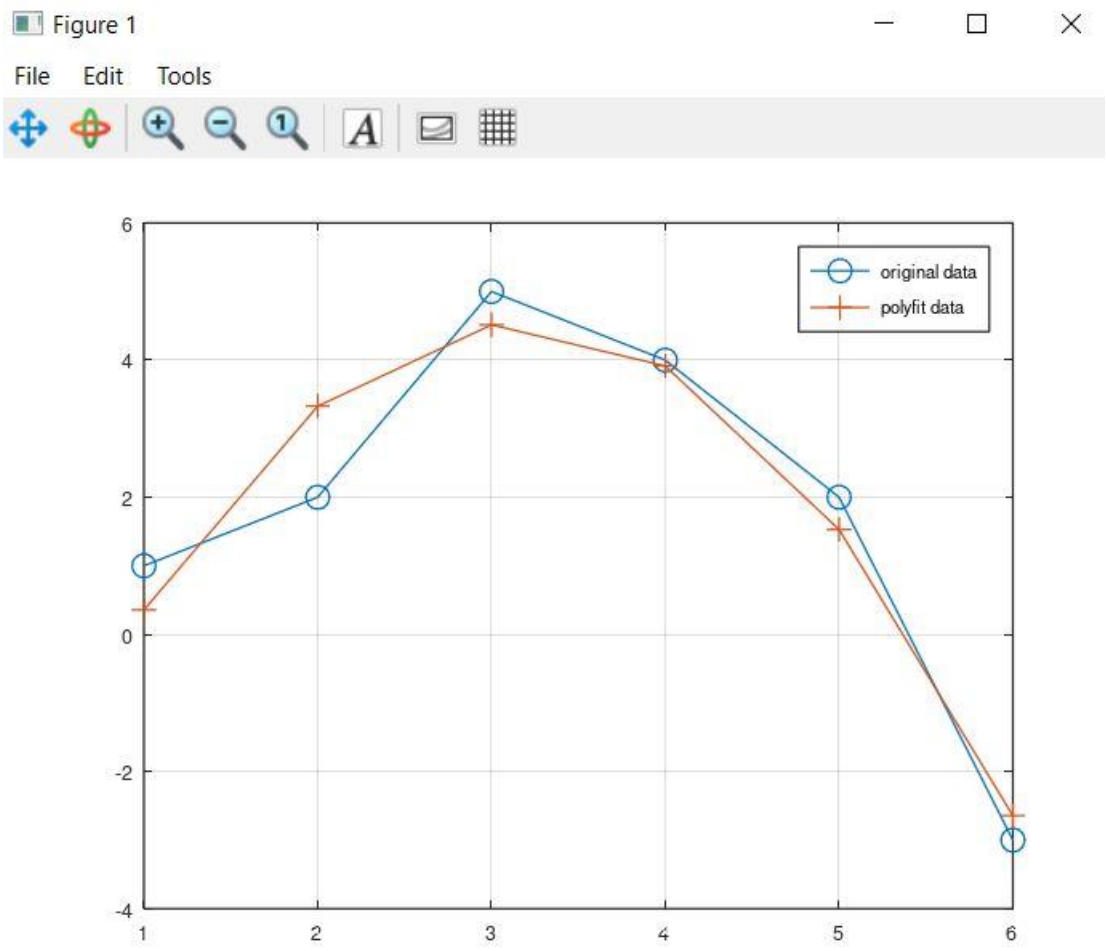
```
>> plot (xdata,ydata, 'o' ,x,y, 'linewidth', 2)
>> legend('data values', 'least-squares parabola')
>> grid on
>> title ('y = -0.89286 x^2 + 5.65 x - 4.4')
```



Использую встроенную в Octave функцию для нахождения полинома polyfit

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

  -0.8929   5.6500  -4.4000

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

   0.3571
   3.3286
   4.5143
   3.9143
   1.5286
  -2.6429

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

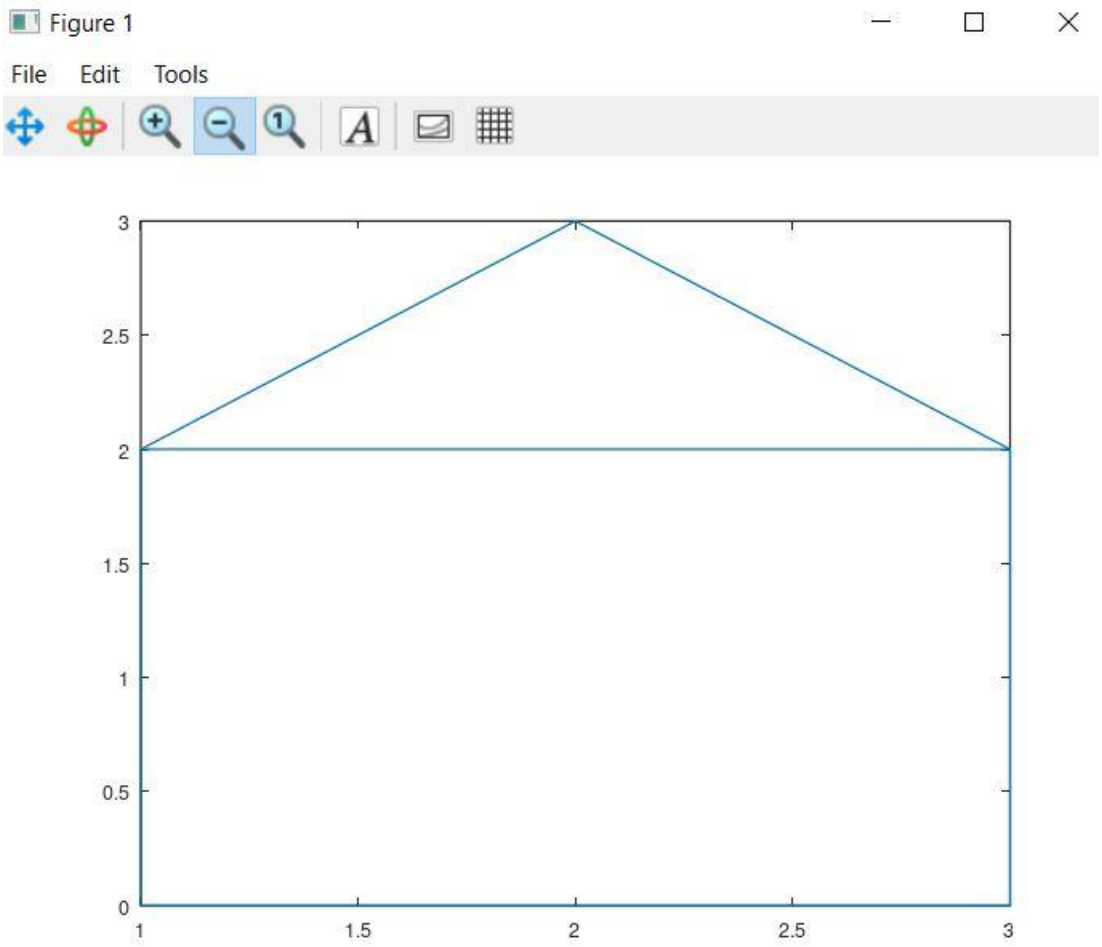
Строю график с исходными и подгоночными данными



Матричные преобразования

Рисую граф в виде дома

```
>> 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)
```

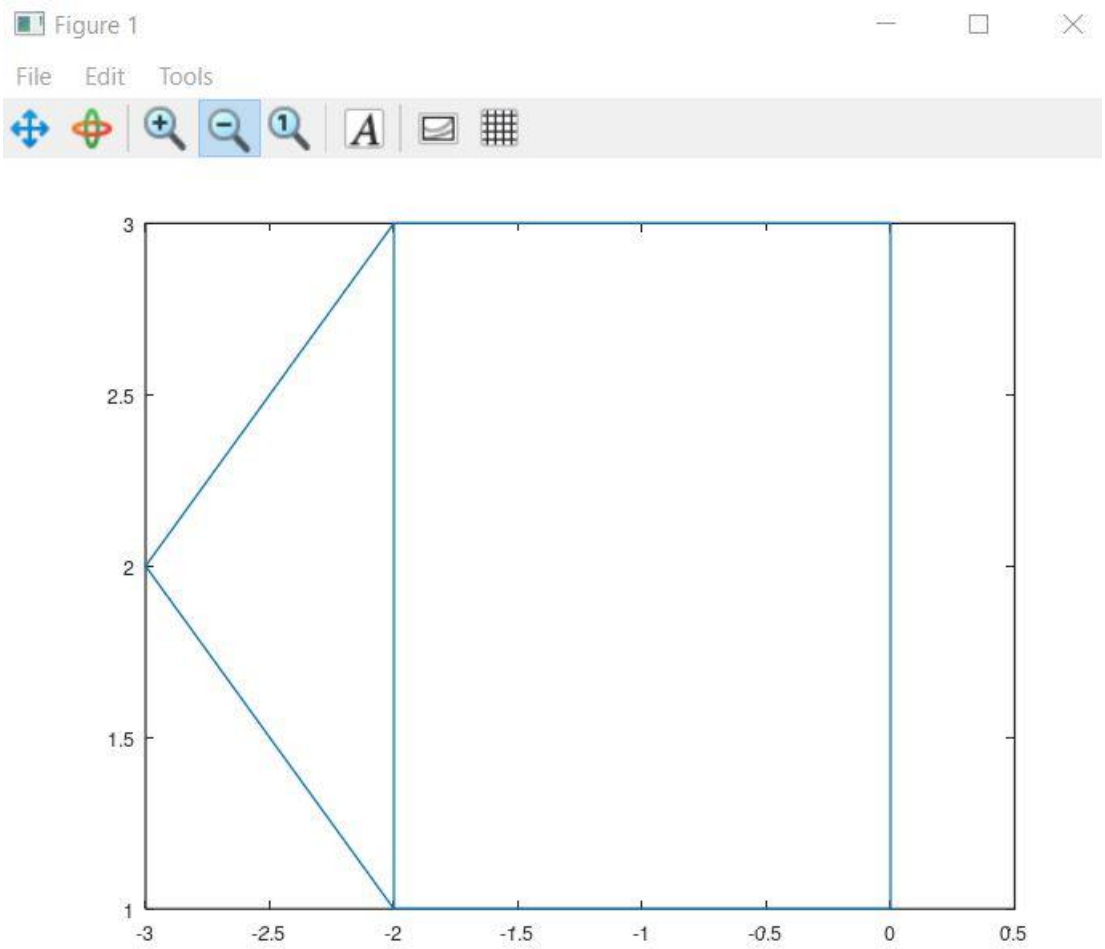


Вращение

Задаю угол вращения и записываю матрицу вращения со значением данного угла. После чего уможаю исходную матрицу для дома на полученную матрицу, для тоже чтобы развернуть исходный граф

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

   6.1230e-17  -1.0000e+00
   1.0000e+00   6.1230e-17

>> RD1 = R1*D
RD1 =

  -2.0000e+00   6.1230e-17   1.8369e-16  -2.0000e+00  -3.0000e+00  -2.0000e+00  -2.0000e+00
   1.0000e+00   1.0000e+00   3.0000e+00   3.0000e+00   2.0000e+00   1.0000e+00   3.0000e+00

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

  -2.0000e+00   6.1230e-17   1.8369e-16  -2.0000e+00  -3.0000e+00  -2.0000e+00  -2.0000e+00

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

   1   1   3   3   2   1   3

>> plot (x,y)
>> plot (x1,y1)
```

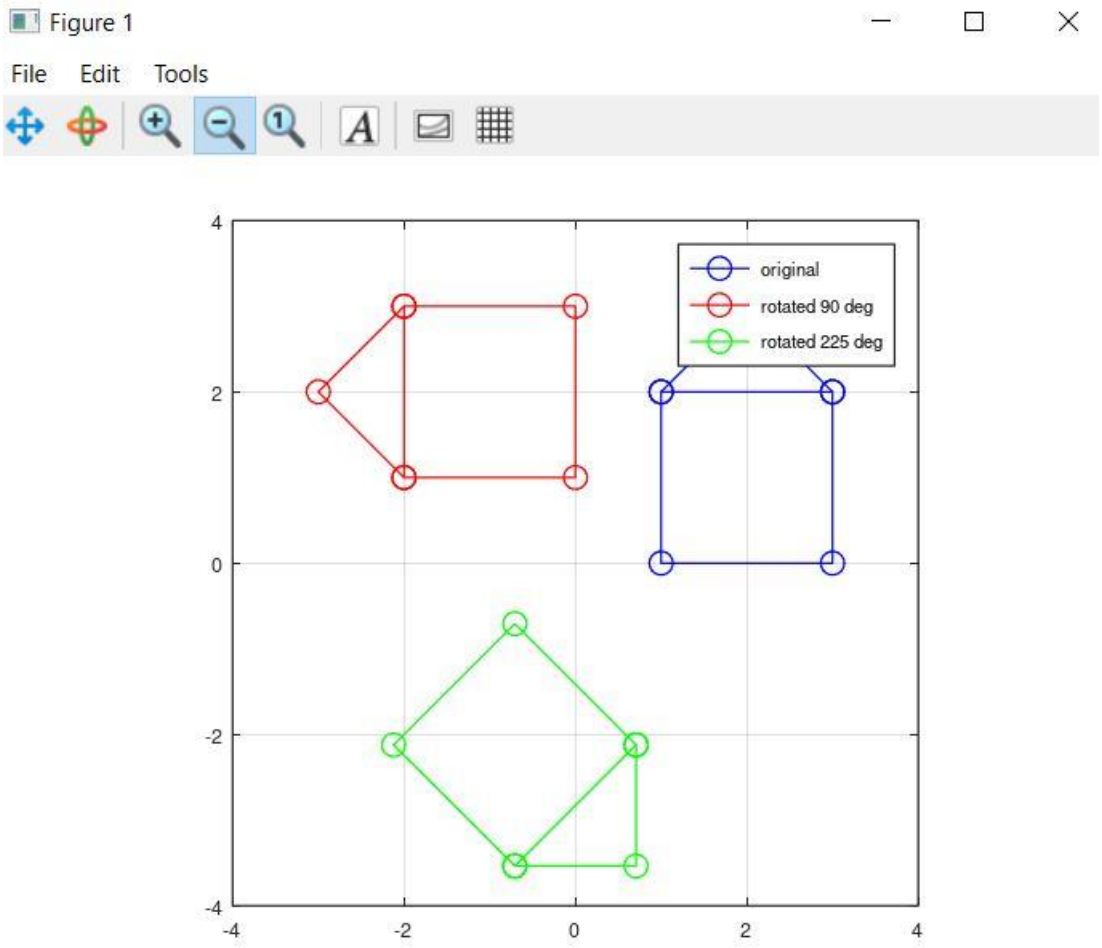
Проверяю результат на графике



Повторяю все теже действия, но уже с другим углом

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

  -0.7071   0.7071
  -0.7071  -0.7071

>> RD2 = R2*D
RD2 =

   0.7071  -0.7071  -2.1213  -0.7071   0.7071   0.7071  -0.7071
  -2.1213  -0.7071  -2.1213  -3.5355  -3.5355  -2.1213  -3.5355

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

   0.7071  -0.7071  -2.1213  -0.7071   0.7071   0.7071  -0.7071

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

  -2.1213  -0.7071  -2.1213  -3.5355  -3.5355  -2.1213  -3.5355

>> plot (x,y, 'bo-' , x1 , y1 , 'ro-' , x2 , y2 , 'go-' )
>> axis ([-4 4 -4 4] , 'equal' )
>> grid on
>> legend ('original' , 'rotated 90 deg' , 'rotated 225 deg' )
>>
```

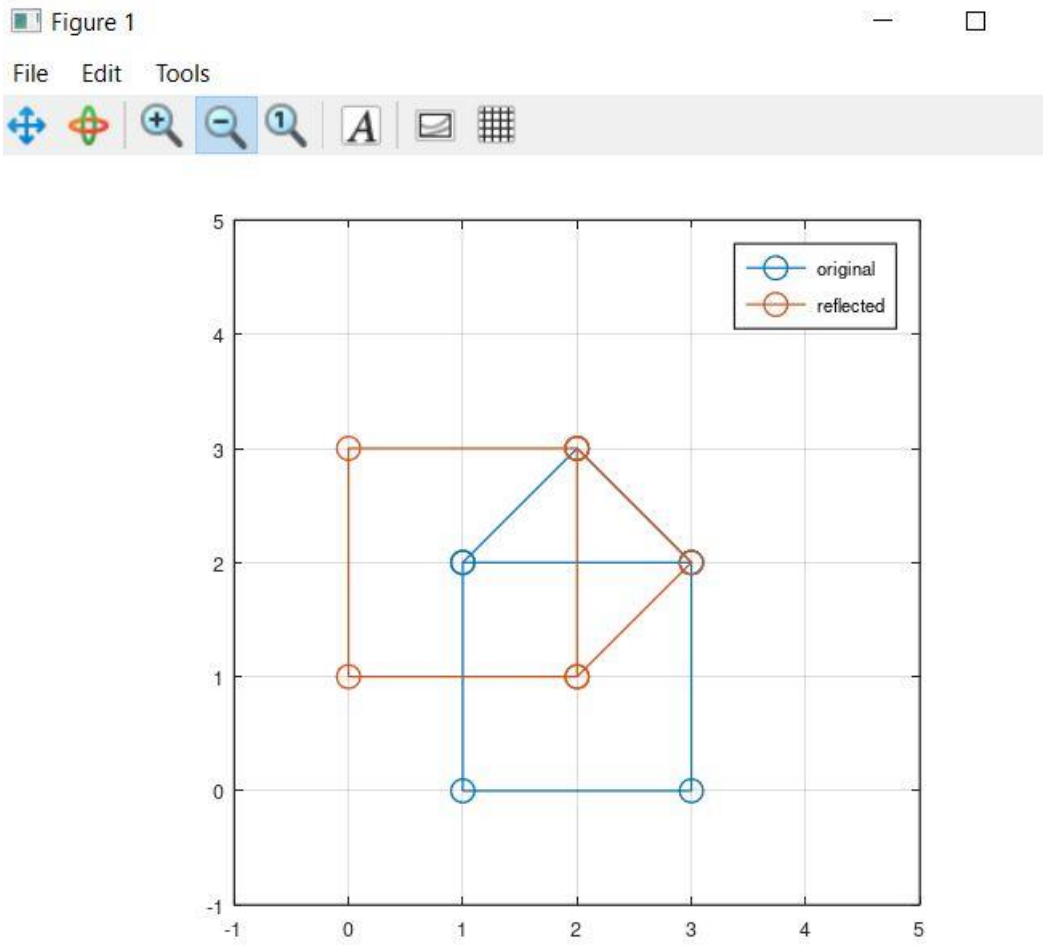
Для наглядности результата, вывожу все три графика на одном



Отражение

Задаю матрицу R параметрами для отражения и аналогично предыдущему методу, для получения результата умножаю исходную матрицу на полученную матрицу R. После чего проверяю результат отображаю полученные значения на графике вместе с исходным графиком

```
>>  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 5 -1 5], 'equal')
>> grid on
>> legend ( 'original' , 'reflected' )
```

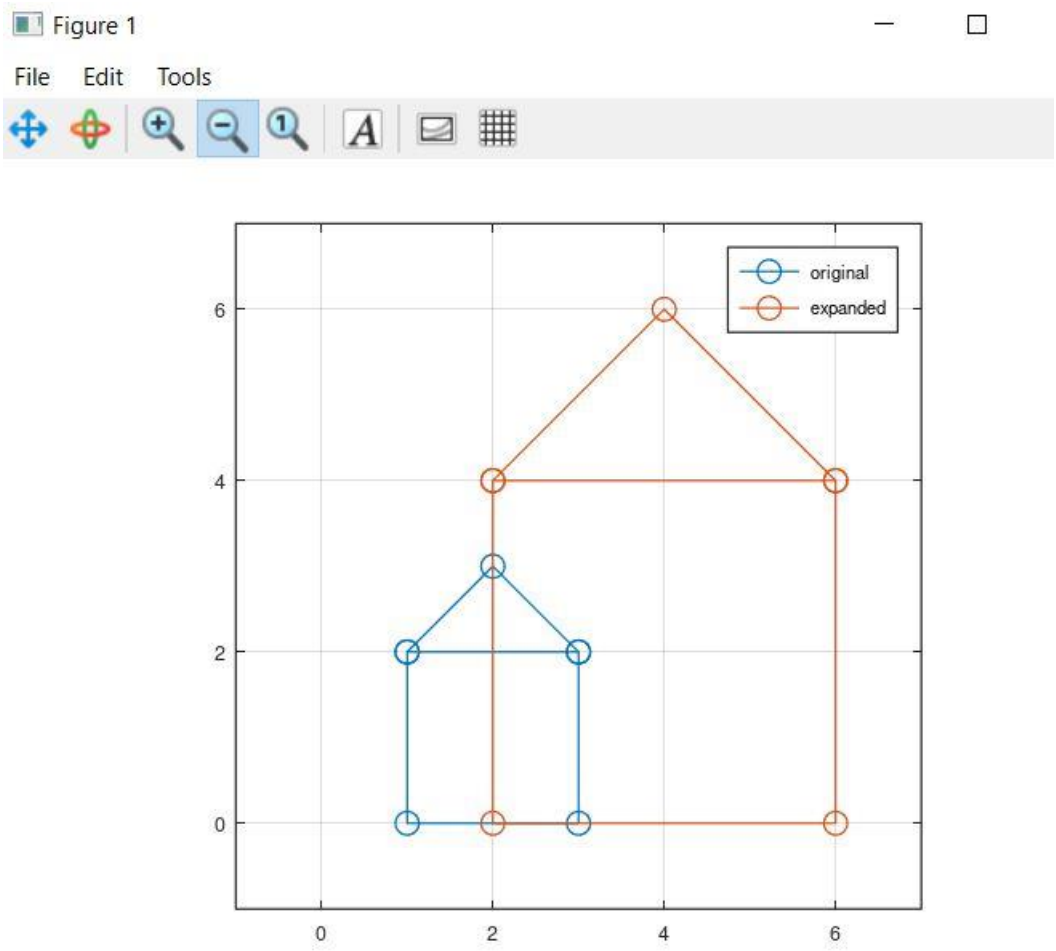


Дилатация

Для дилатации создаю новую матрицу T с ненулевыми и равными значениями на главной диагонали, остальные значения равны нулю и аналогично предыдущим методам умножаю получившуюся матрицу на исходную

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

   2   0
   0   2

>> TD = T*D
TD =

   2   2   6   6   4   2   6
   4   0   0   4   6   4   4

>> x1 = TD(1,:); y1 = TD(2,:)
y1 =

   4   0   0   4   6   4   4

>> x1
x1 =

   2   2   6   6   4   2   6

>> plot (x, y, 'o-', x1, y1,'o-')
>> axis ([-1 7 -1 7], 'equal')
>> grid on
>> legend ('original', 'expanded')
```

Оторбражаю результат на графике, вместе с исходным графов для сравнения



Выводы

В процессе выполнения работы, я познакомилась с новыми функциями polyfit и polyval, необходимыми для получения полиномиальной кривой, а также научилась без этих функций выполнять подгонку к полиномиальной кривой использую метод наименьшего квадрата и преобразование Гаусса. Изучила матрицы необходимые для матричных преобразований. Научилась изменять графы, а именно отражать, вращать и дилатировать их, используя умножение матрицы исходного графа на специальные матрицы из теоретической части.