Лабораторная работа №5

Научное программирование

Хохлачева Яна Дмитриевна, НПМмд-02-22

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Научиться решать общую проблему подгонки полинома к множеству точек с помощью Octave.

2 Задание

Рассмотреть методы матричного преобразования, вращения, отражения, а также дилатации.

3 Выполнение лабораторной работы

3.1 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. 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.

```
>> 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-')
```

Рис. 3.1: Введённая матрица данных в Octave и извлечённые вектора x и y

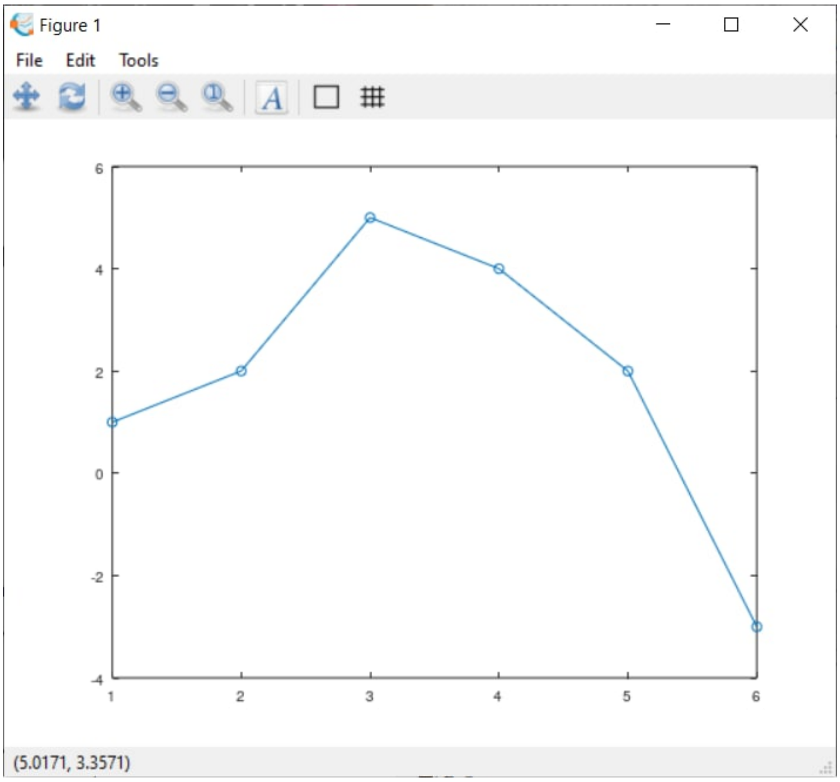

Рис. 3.2: Точки на графике

Построим уравнение вида y = ax2 + bx + c. Подставляя данные, получаем следующую систему линейных уравнений. 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.

```
>> 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. 3.3: Система линейных уравнений

Решение по методу наименьших квадратов получается из решения уравнения ATAb = ATy, где b – вектор коэффициентов полинома. Используем Octave для построения уравнений.

```
>> A'*A
ans =

   2275    441     91
    441     91     21
     91     21      6

>>  A' * ydata
ans =

   60
   28
   11

>> B = A' * A;
>> B (:,4) = A' * ydata;
>> 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
>>  x = linspace (0,7,50);
>> y = a1 * x .^ 2 + a2 * x + a3;
>> plot (xdata,ydata, 'o' ,x,y, 'linewidth', 2)
>> |
```

Рис. 3.4: Решение задачи методом Гаусса и построение соответствующего графика параболы

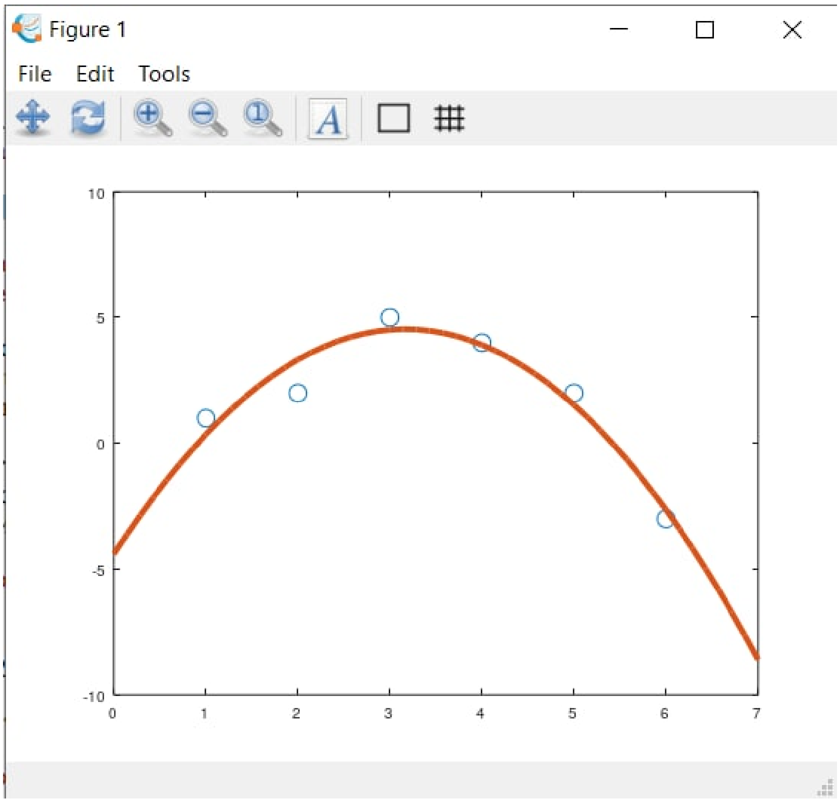

Рис. 3.5: График параболы

Процесс подгонки может быть автоматизирован встроенными функциями Octave. Для этого мы можем использовать встроенную функцию для подгонки полинома polyfit. Значения полинома P в точках, задаваемых вектором-строкой x можно получить с помощью функции polyval. Получим подгоночный полином

```
>> 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,'+-')
>>
```

Рис. 3.6: Подгоночный полином polyfit

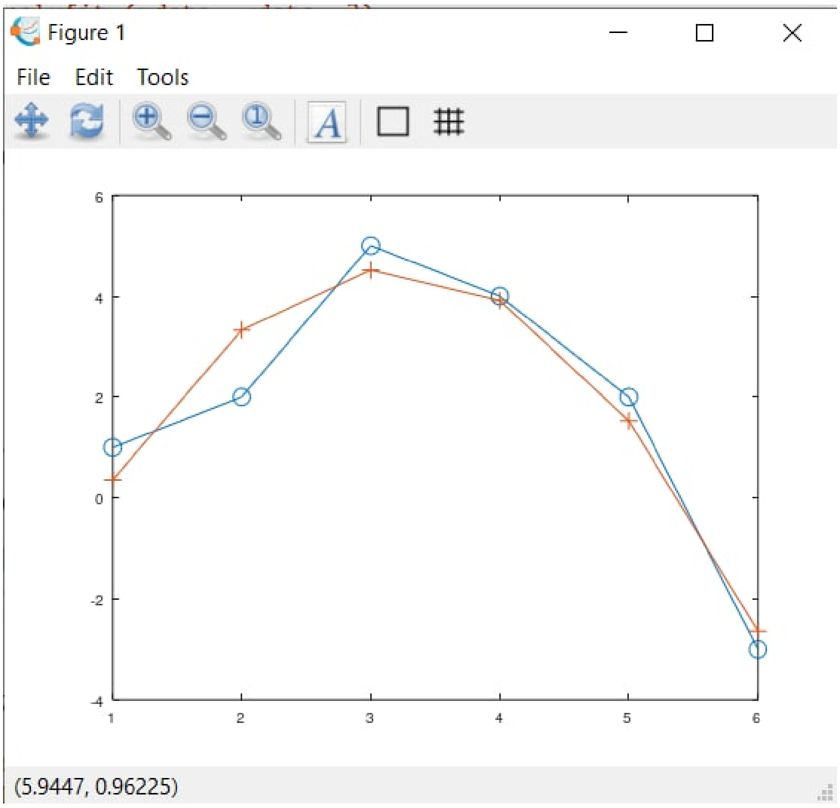

Рис. 3.7: Построение исходных и подгоночных данных

3.2 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 × 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 = [ 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. 3.8: Матричные преобразования

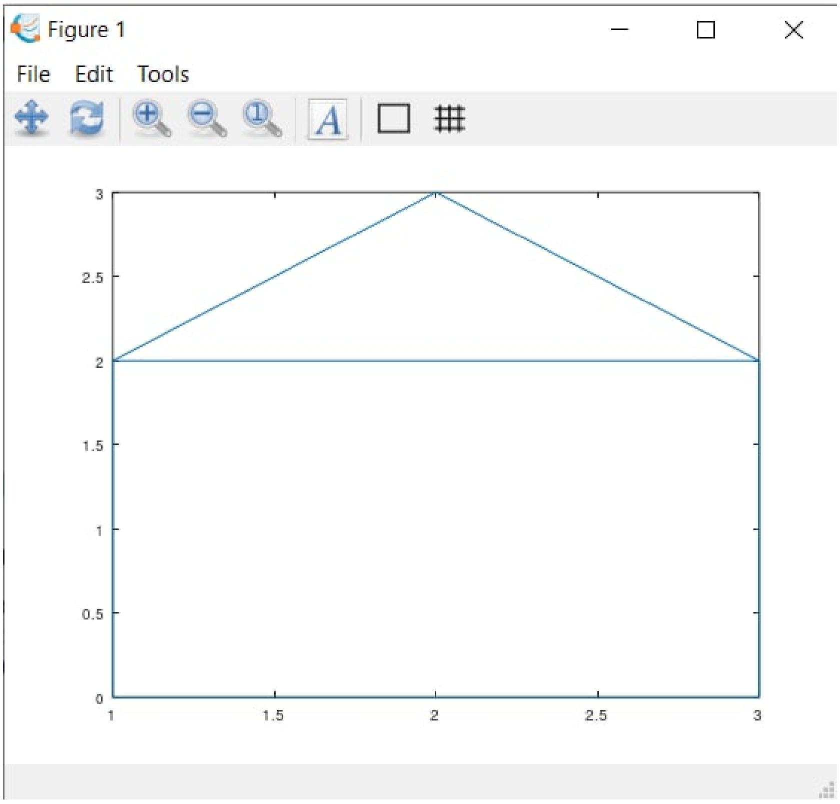

Рис. 3.9: Граф

3.3 Вращение

Рассмотрим различные способы преобразования изображения. Вращения могут быть получены с использованием умножения на специальную матрицу. Чтобы произвести повороты матрицы данных D, нам нужно вычислить произведение матриц RD. Повернём граф дома на 90 и 225. Вначале переведём угол в радианы.

```
>> 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

>> 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-' )
>> |
```

Рис. 3.10: Вращение

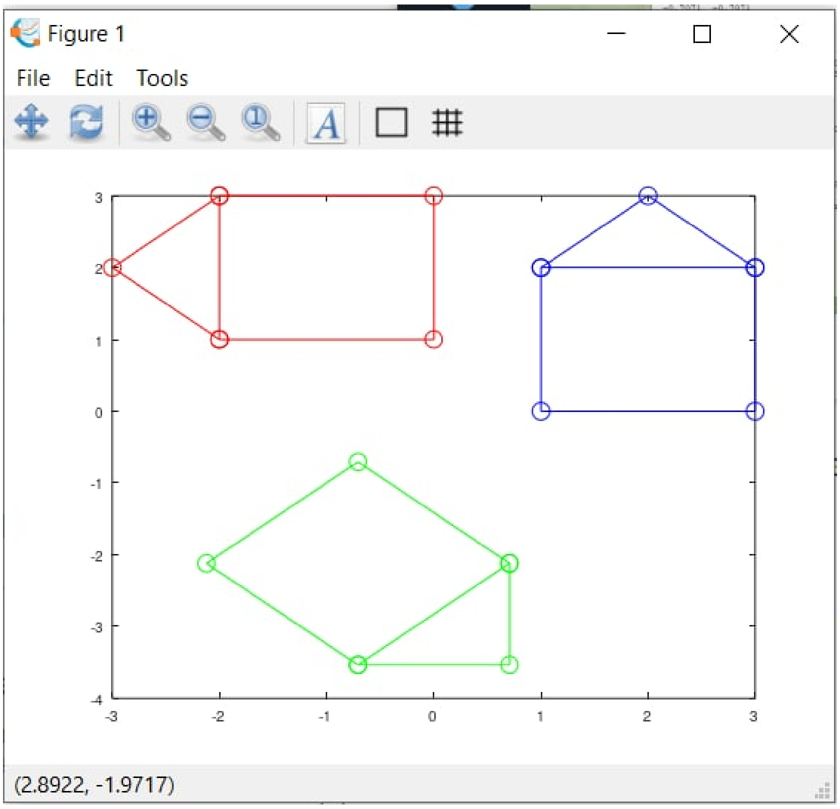

Рис. 3.11: Граф

3.4 Отражение

Отразим граф дома относительно прямой y = x. Зададим матрицу отражения.

```
>> 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-')
>>
```

Рис. 3.12: Отражение

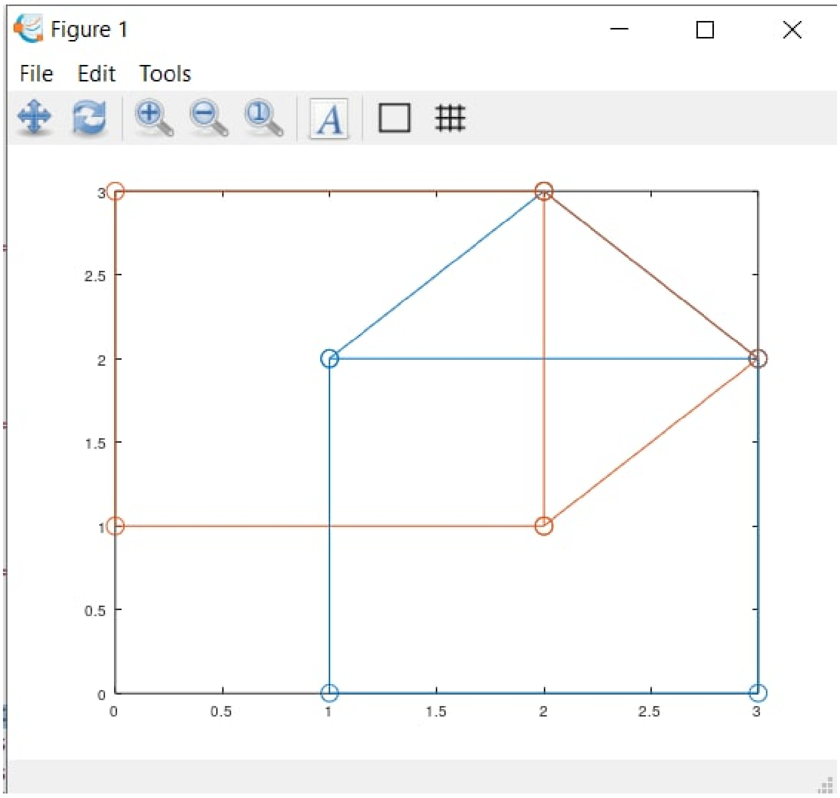

Рис. 3.13: Граф

3.5 Дилатация

Дилатация (то есть расширение или сжатие) также может быть выполнено путём умножения матриц. Тогда матричное произведение T D будет преобразованием дилатации D с коэффициентом k. Увеличим граф дома в 2 раза

```
>> 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-')
>> |
```

Рис. 3.14: Отражение

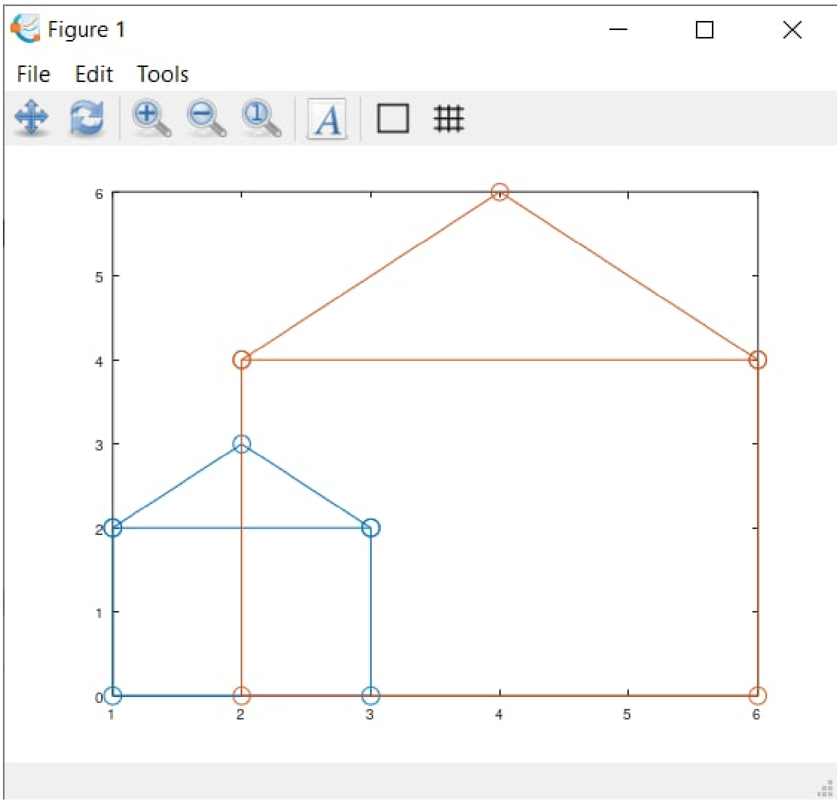

Рис. 3.15: Граф

4 Выводы

Ознакомилась с решением общей проблемы подгонки полинома к множеству точек с помощью Octave. Рассмотрены методы матричного преобразования, вращения, отражения, а также дилатации.

Список литературы