```
%Here we have created a function LSSNEqn
function [X]=LSSNEqn(A,b)
temp1=A.';     %temp1 is transpose of A

C=temp1*A;

D=chol(C);    %chol function is basically do a cholskey decomposition
temp3=D.';

%Solving the system of linear equation using backward and forward
%substitution

Y=FdSubs(temp3,temp1*b);
X=BdSubs(D,Y);

end
```

```
%Gram-Schmidt Method for QR decomposition

function [Q,R] = GramSelf(A)


    [m,n] = size(A);
    Q = zeros(m,n);
    R = zeros(n,n);

    for j = 1:n
        v=A(:,j);
        for i=1:j-1


            R(i,j)=Q(:,i).'*A(:,j);
            v=v-R(i,j)*Q(:,i);
        end
        R(j,j)=norm(v);
        Q(:,j)=v/R(j,j);
    end
```

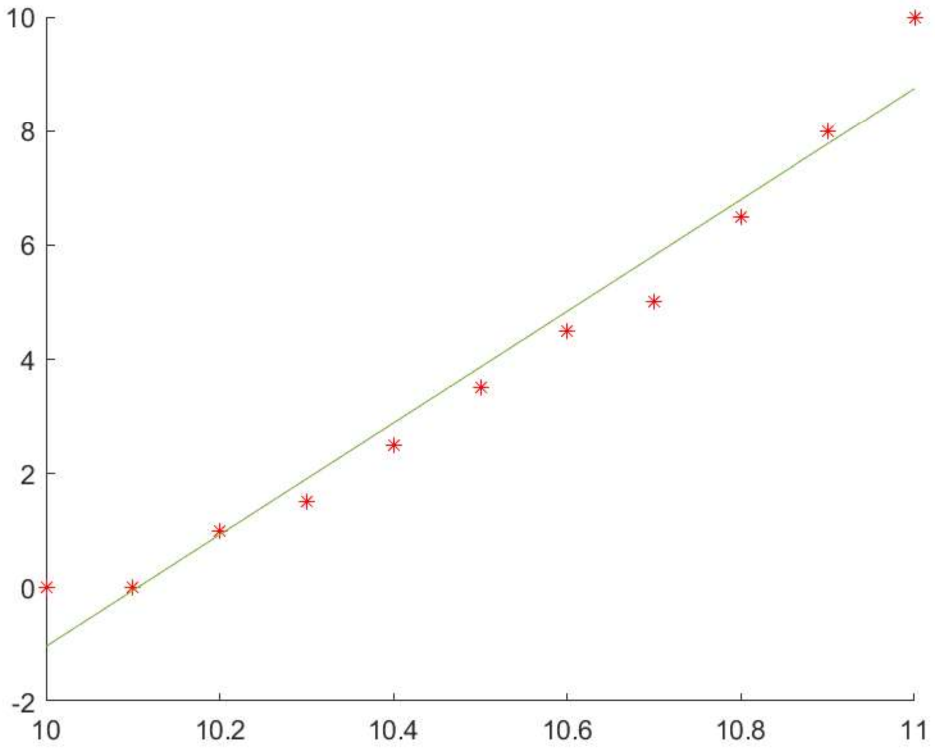
```

%Householder's Method for QR decomposition
function [Q,R] = HouseSelf(A)

[m,n] = size(A);
Q = eye(m);
R = A;

for i = 1:n
z=A(i:m,i);
v=[ -sign(z(1))*norm(z)- z(1);-z(2:end)];
v=v/sqrt(v.'*v);

for j=1:n
    A(i:m,j)=A(i:m,j)-v*(2*(v.'*A(i:m,j)));
end
for j=1:m
    Q(i:m,j)=Q(i:m,j)-v*(2*(v.'*Q(i:m,j)));
end


end

Q=Q.';
R=triu(A);
```

```matlab
% Task2
% To solve this question we are creating a function FdSubs which is taking
% two input L and b and returning an ouput matrix x
% Note----We have to first give the L and b as input
function [x] = FdSubs(L, b)

[m, n] = size(L);       %copying the size of U in m and n variables

%As we are doing this procedure only for Square matrix so program will give
%an error if L is not square matrix
if m ~= n
error('L must be a square matrix');
end

%if L is square matrix then we create a column matrix with all elements zero
x = zeros(n, 1);

for i = 1:n
%now we are using this equation
% In reality this equation is just back substitution method
% We can observe this thing when we run the for loop and put the required
% values in it

x(i) = (b(i) - L(i, 1:n) * x) / L(i, i);

end

end
```

```
% To solve this question we are creating a function BsSubs which is taking
% two input U and b and returning an ouput matrix x
% Note----We have to first give the U and b as input

function [x] = BdSubs(U, b)

[m, n] = size(U);               %copying the size of U in m and n variables

%As we are doing this procedure only for Square matrix so program will give
%an error if U is not square matrix
if m ~= n
error('U must be a square matrix');
end

%if U is square matrix then we create a column matrix with all elements zero
x = zeros(n, 1);

for i = 1:n
j = n+1-i;
%now we are using this equation
% In reality this equation is just back substitution method
% We can observe this thing when we run the for loop and put the required
% values in it
%Here we are creating a variable j and first we are calculating the value
%nth row of x ......We can easily observe this all just write these all
%matrix in systematic manner on notebook and then we can see that first we
%calculate the nth element of x then using that n-1th element of x and so on...
x(j) = (b(j) - U(j, 1:n) * x) / U(j, j);

end

end
```

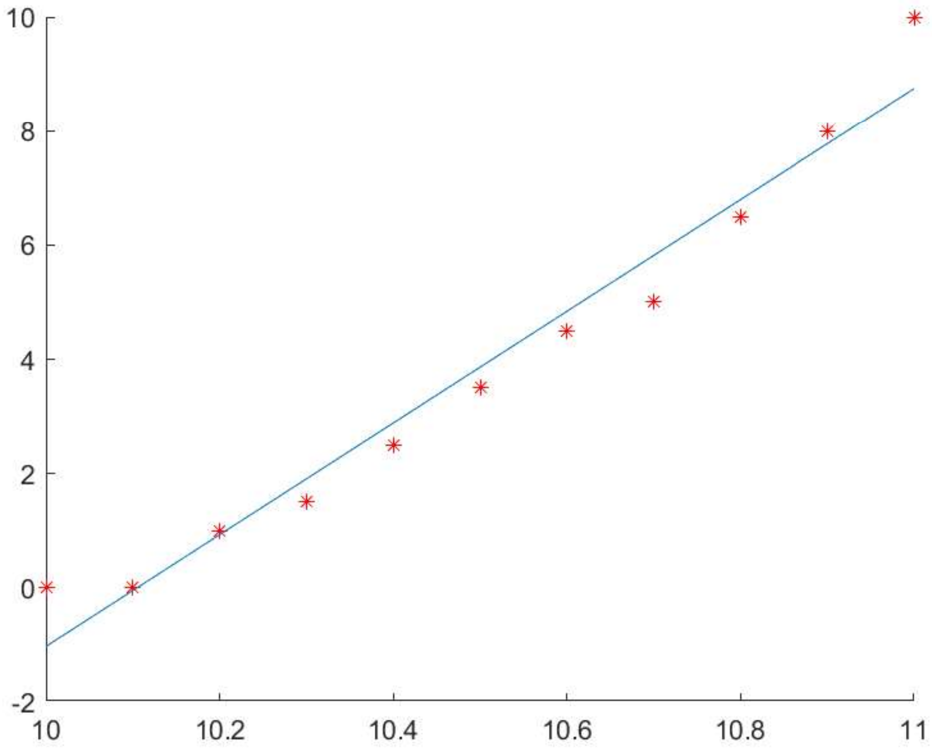
```
A=[10.0,1;10.1,1;10.2,1;10.3,1;10.4,1;10.5,1;10.6,1;10.7,1;10.8,1;10.9,1;11.0,1];
b=[0.0;0.0;1.0;1.5;2.5;3.5;4.5;5.0;6.5;8.0;10.0];
X=LSSNEqn(A,b);
for i=1:11
    temp(i,1)=A(i,1);
end
disp("A=");
disp(A);
disp("b=");
disp(b);
disp("Solution matrix x by least square method is ");
disp('X=');
disp(X);
for i=1:11
y(i,1)=X(1,1)*A(i,1)+X(2,1);
end
for i=1:11
hold on
plot(A(i,1),b(i,1),'r*');
end
plot(temp,y);

hold off
```

```
>> A4P1
A=
   10.0000    1.0000
   10.1000    1.0000
   10.2000    1.0000
   10.3000    1.0000
   10.4000    1.0000
   10.5000    1.0000
   10.6000    1.0000
   10.7000    1.0000
   10.8000    1.0000
   10.9000    1.0000
   11.0000    1.0000

b=
         0
         0
    1.0000
    1.5000
    2.5000
    3.5000
    4.5000
    5.0000
    6.5000
    8.0000
   10.0000

Solution matrix x by least square method is
X=
    9.7727
  -98.7500

>>
```


10
8
6
4
2
0
-2
10
10.2
10.4
10.6
10.8
11

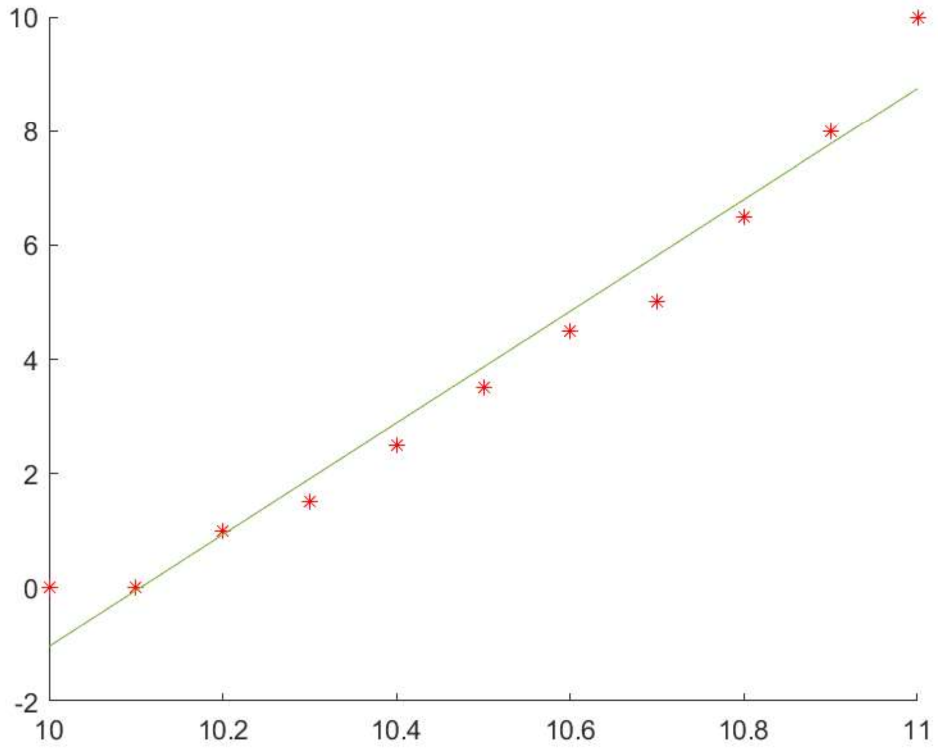
```
%Doing QR decomposition using GramSelf method

A=[10.0,1;10.1,1;10.2,1;10.3,1;10.4,1;10.5,1;10.6,1;10.7,1;10.8,1;10.9,1;11.0,1];
b=[0.0;0.0;1.0;1.5;2.5;3.5;4.5;5.0;6.5;8.0;10.0];
%disp("A=");
%disp(A);
for i=1:11
    temp(i,1)=A(i,1);
end
[Q,R]=GramSelf(A);
disp("A=");
disp(A);
disp("b=");
disp(b);

Y=Q.'*b;
%disp(Y);
X=BdSubs(R, Y);
disp('X=');
disp(X);

for i=1:11
y(i,1)=X(1,1)*A(i,1)+X(2,1);
end
for i=1:11
hold on
plot(A(i,1),b(i,1),'r*');
end

plot(temp,y);
hold off
```

```
>> A4P2
A=
   10.0000    1.0000
   10.1000    1.0000
   10.2000    1.0000
   10.3000    1.0000
   10.4000    1.0000
   10.5000    1.0000
   10.6000    1.0000
   10.7000    1.0000
   10.8000    1.0000
   10.9000    1.0000
   11.0000    1.0000

b=
         0
         0
    1.0000
    1.5000
    2.5000
    3.5000
    4.5000
    5.0000
    6.5000
    8.0000
   10.0000

X=
    9.7727
  -98.7500

>>
```


10
8
6
4
2
0
-2
10
10.2
10.4
10.6
10.8
11

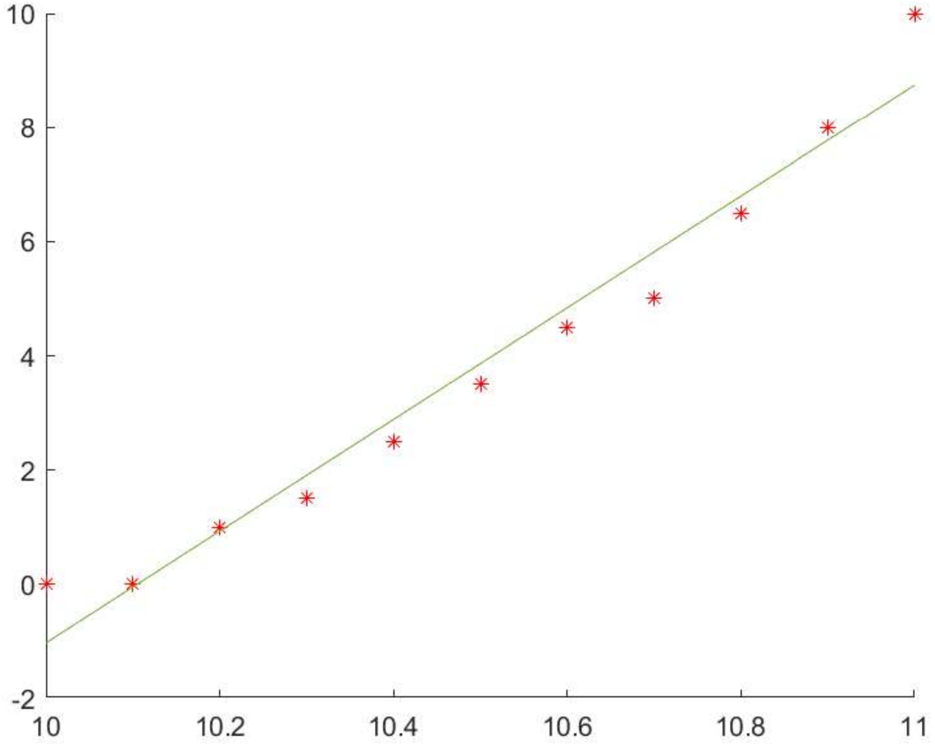
```

A=[10.0,1;10.1,1;10.2,1;10.3,1;10.4,1;10.5,1;10.6,1;10.7,1;10.8,1;10.9,1;11.0,1];
b=[0.0;0.0;1.0;1.5;2.5;3.5;4.5;5.0;6.5;8.0;10.0];
%disp("A=");
%disp(A);
for i=1:11
    temp(i,1)=A(i,1);
end

[Q,R]=HouseSelf(A);
disp(Q);
disp(R);
temp3=[eye(2,2);zeros(9,2)];
temp2=[R(1,1),R(1,2);R(2,1),R(2,2)];
R=temp2;
Q=Q*temp3;
Y=Q.'*b;
%disp(Y);
X=BdSubs(R, Y);
disp(X);
for i=1:11
y(i,1)=X(1,1)*A(i,1)+X(2,1);
end
for i=1:11
hold on
plot(temp(i,1),b(i,1),'r*');
end

plot(temp,y);
hold off
```

```
>> A4P3
  Columns 1 through 10

   -0.2870   -0.4856   -0.3210   -0.3091   -0.2972   -0.2854   -0.2735   -0.2617 ↙
-0.2498   -0.2380
   -0.2899   -0.3903   -0.2568   -0.1578   -0.0587    0.0403    0.1394    0.2385 ↙
0.3375    0.4366
   -0.2928   -0.2950    0.9004   -0.0830   -0.0664   -0.0498   -0.0333   -0.0167 ↙
-0.0001    0.0165
   -0.2956   -0.1997   -0.0864    0.9229   -0.0677   -0.0584   -0.0490   -0.0396 ↙
-0.0303   -0.0209
   -0.2985   -0.1044   -0.0733   -0.0711    0.9310   -0.0669   -0.0647   -0.0626 ↙
-0.0605   -0.0583
   -0.3014   -0.0091   -0.0601   -0.0652   -0.0703    0.9246   -0.0805   -0.0856 ↙
-0.0907   -0.0958
   -0.3042    0.0862   -0.0469   -0.0592   -0.0716   -0.0839    0.9038   -0.1085 ↙
-0.1209   -0.1332
   -0.3071    0.1815   -0.0337   -0.0533   -0.0728   -0.0924   -0.1119    0.8685 ↙
-0.1510   -0.1706
   -0.3100    0.2768   -0.0206   -0.0474   -0.0741   -0.1009   -0.1277   -0.1545 ↙
0.8188   -0.2080
   -0.3129    0.3721   -0.0074   -0.0414   -0.0754   -0.1094   -0.1434   -0.1774 ↙
-0.2114    0.7546
   -0.3157    0.4674    0.0058   -0.0355   -0.0767   -0.1179   -0.1592   -0.2004 ↙
-0.2416   -0.2828

  Column 11

   -0.2261
    0.5356
    0.0331
   -0.0116
   -0.0562
   -0.1009
   -0.1455
   -0.1901
   -0.2348
   -0.2794
    0.6759

  -34.8404   -3.3151
         0   -0.0998
         0         0
         0         0
         0         0
         0         0
         0         0
         0         0
         0         0
         0         0
         0         0

    9.7727
  -98.7500

>>
```


10
8
6
4
2
0
-2
10
10.2
10.4
10.6
10.8
11

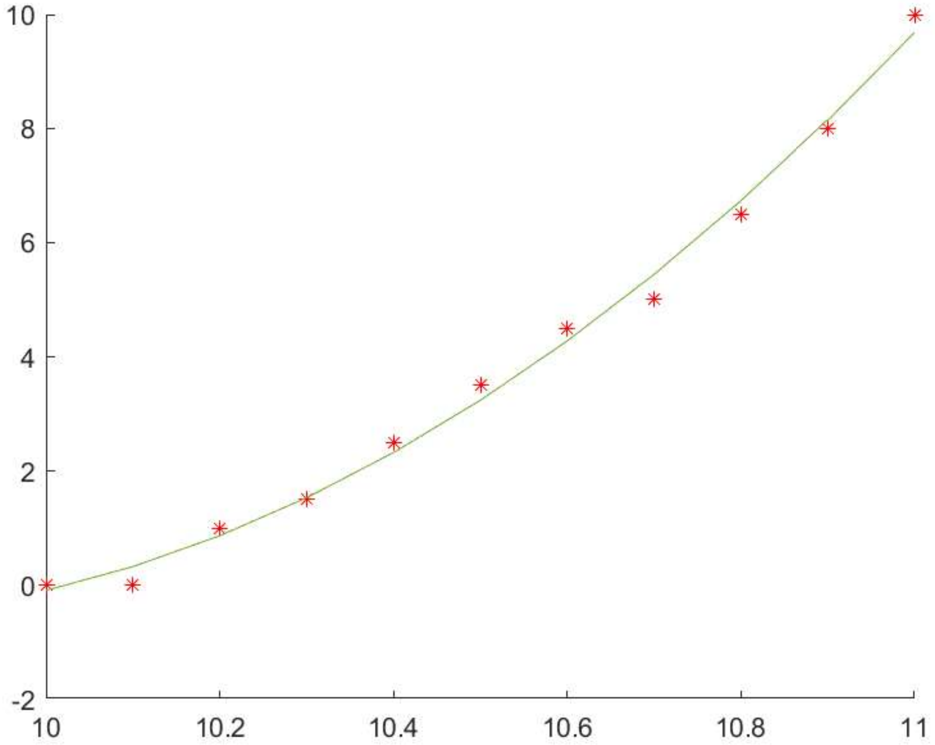
```

A=[10.0,1;10.1,1;10.2,1;10.3,1;10.4,1;10.5,1;10.6,1;10.7,1;10.8,1;10.9,1;11.0,1];
b=[0.0;0.0;1.0;1.5;2.5;3.5;4.5;5.0;6.5;8.0;10.0];
%disp("A=");
%disp(A);
for i=1:11
    temp(i,1)=A(i,1);
end
[U,S,V]=svd(A,0);

X= V*((U'*b)./diag(S));
disp(X);
for i=1:11
y(i,1)=X(1,1)*A(i,1)+X(2,1);
end
for i=1:11
hold on
plot(A(i,1),b(i,1),'r*');
end

plot(temp,y);
hold off
```

```
>> A4P4
    9.7727
  -98.7500

>>
```


10
8
6
4
2
0
-2
10
10.2
10.4
10.6
10.8
11

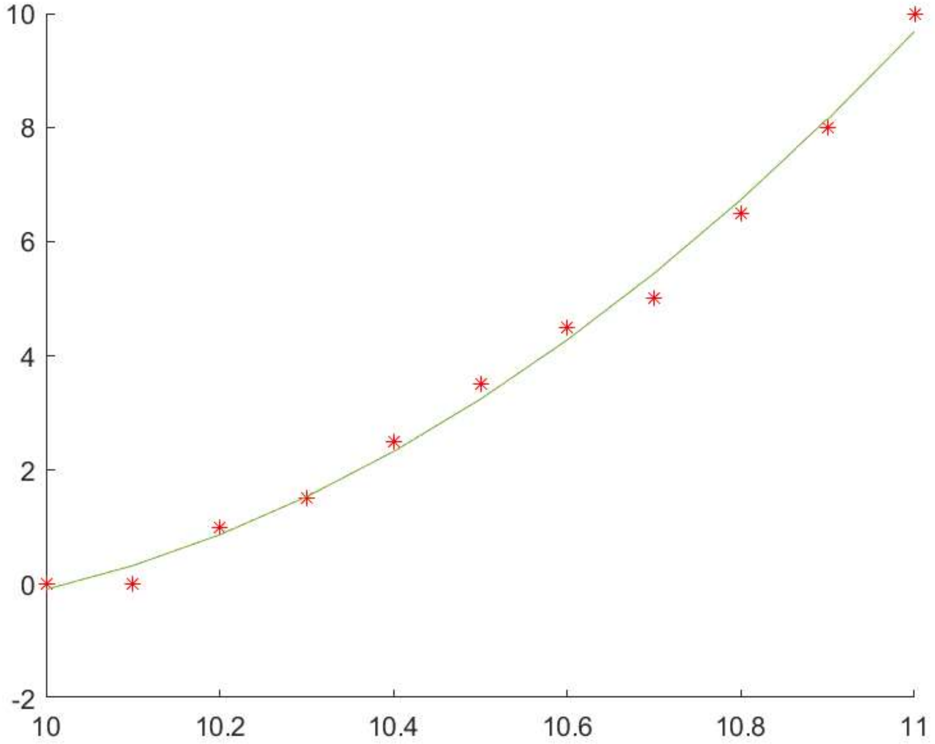
```
%Basically we have already created a function LSSNEqn and now using it we are trying
%to solve a system of linear equation by least square method

%prompt = "Enter the matrix A \nA=";
%A = input(prompt);
%prompt = "Enter the matrix b \nb=";
%b = input(prompt);
A=[10.0*10.0,10.0,1;10.1*10.1,10.1,1;10.2*10.2,10.2,1;10.3*10.3,10.3,1;10.4*10.4,10.4,1;10.5*10.5,10.5,1;10.6*10.6,10.6,1;10.7*10.7,10.7,1;10.8*10.8,10.8,1;10.9*10.9,10.9,1;11.0*11.0,11.0,1];
b=[0.0;0.0;1.0;1.5;2.5;3.5;4.5;5.0;6.5;8.0;10.0];
X=LSSNEqn(A,b);
for i=1:11
    temp(i,1)=A(i,2);
end

disp("Solution matrix x by least square method is ");
disp('X=');
disp(X);
for i=1:11
y(i,1)=X(1,1)*A(i,1)+X(2,1)*A(i,2)+X(3,1);
end
for i=1:11
hold on
plot(A(i,2),b(i,1),'r*');
end

plot(temp,y);
hold off
```

```
>> A4P5
Solution matrix x by least square method is
X=
    6.2354
 -121.1713
  588.0828

>>
```


10
8
6
4
2
0
-2
10
10.2
10.4
10.6
10.8
11

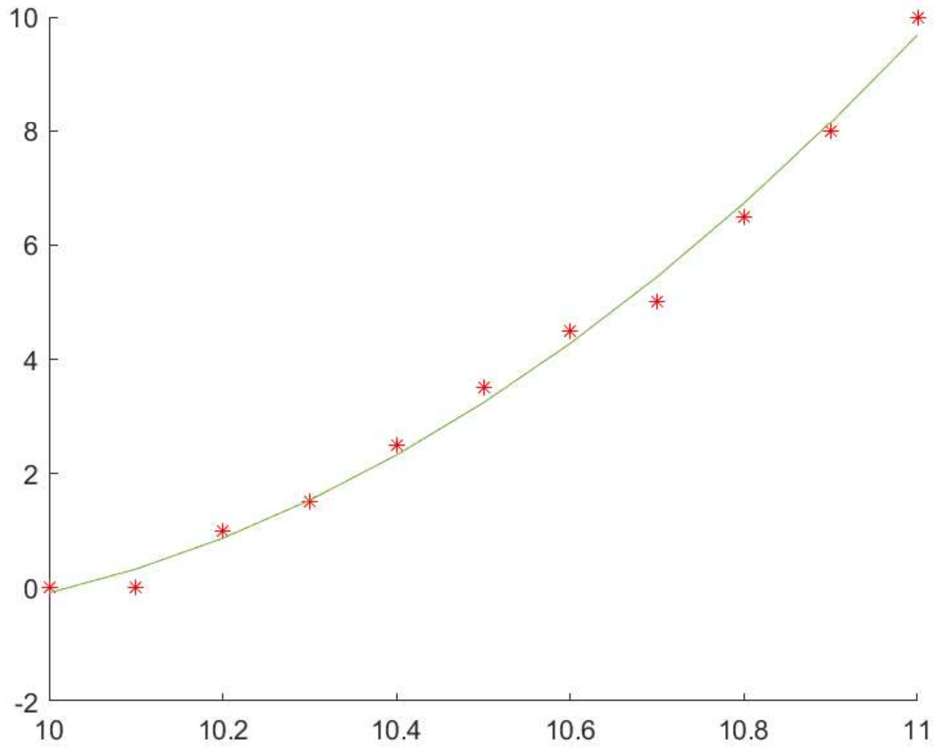
```
%Doing QR decomposition using GramSelf method

A=[10.0*10.0,10.0,1;10.1*10.1,10.1,1;10.2*10.2,10.2,1;10.3*10.3,10.3,1;10.4*10.4,10.4,1;10.5*10.5,10.5,1;10.6*10.6,10.6,1;10.7*10.7,10.7,1;10.8*10.8,10.8,1;10.9*10.9,10.9,1;11.0*11.0,11.0,1];
b=[0.0;0.0;1.0;1.5;2.5;3.5;4.5;5.0;6.5;8.0;10.0];
%disp("A=");
%disp(A);
for i=1:11
    temp(i,1)=A(i,2);
end
[Q,R]=GramSelf(A);
%disp(Q);
%disp(R);
Y=Q.'*b;
%disp(Y);
X=BdSubs(R, Y);
disp(X);
for i=1:11
y(i,1)=X(1,1)*A(i,1)+X(2,1)*A(i,2)+X(3,1);
end
for i=1:11
hold on
plot(A(i,2),b(i,1),'r*');
end

plot(temp,y);
hold off
```

```
>> A4P6
    6.2354
 -121.1713
  588.0828

>>
```


10
8
6
4
2
0
-2
10
10.2
10.4
10.6
10.8
11

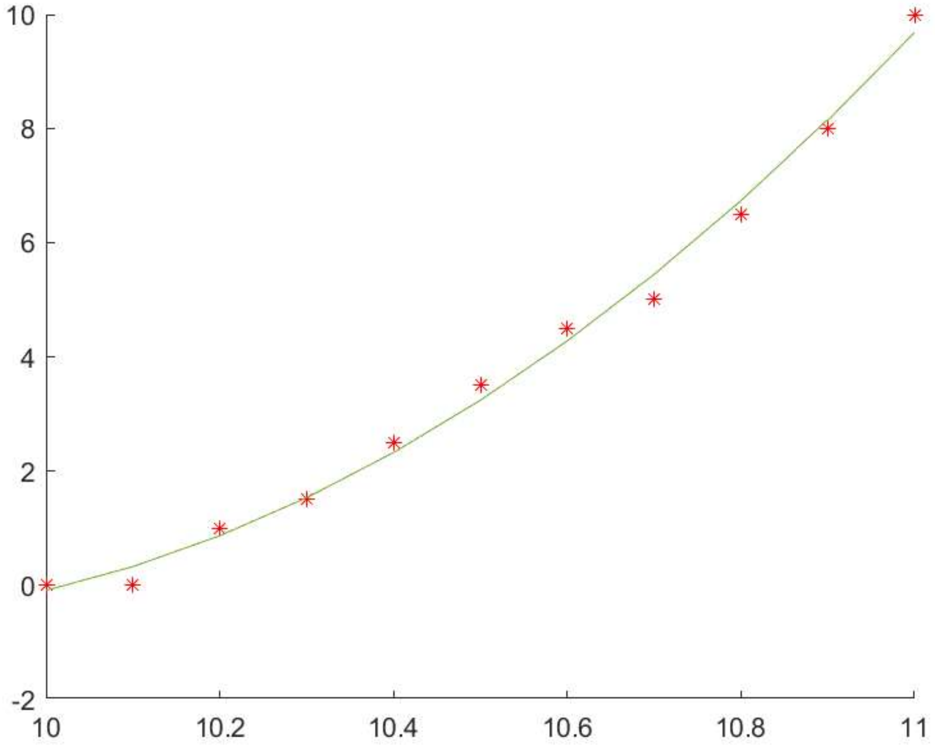
```matlab
A=[10.0*10.0,10.0,1;10.1*10.1,10.1,1;10.2*10.2,10.2,1;10.3*10.3,10.3,1;10.4*10.4,10.4,1;10.5*10.5,10.5,1;10.6*10.6,10.6,1;10.7*10.7,10.7,1;10.8*10.8,10.8,1;10.9*10.9,10.9,1;11.0*11.0,11.0,1];
b=[0.0;0.0;1.0;1.5;2.5;3.5;4.5;5.0;6.5;8.0;10.0];
%disp("A=");
%disp(A);
for i=1:11
    temp(i,1)=A(i,2);
end
flag=A;
[Q,R]=HouseSelf(A);
disp(Q);
disp(R);
temp3=[eye(3,3);zeros(8,3)];
temp2=[R(1,1),R(1,2),R(1,3);R(2,1),R(2,2),R(2,3);R(3,1),R(3,2),R(3,3)];
R=temp2;
Q=Q*temp3;
Y=Q.'*b;
%disp(Y);
X=BdSubs(R, Y);
disp(X);
for i=1:11
y(i,1)=X(1,1)*flag(i,1)+X(2,1)*flag(i,2)+X(3,1);
end
for i=1:11
hold on
plot(flag(i,2),b(i,1),'r*');
end

plot(temp,y);
hold off
```

```
>> A4P7
  Columns 1 through 10

   -0.2727   -0.4802    0.5248   -0.0689    0.0127    0.0449    0.0278   -0.0386 ↙
-0.1542   -0.3192
   -0.2782   -0.3932    0.2152   -0.0357    0.1023    0.2139    0.2989    0.3574 ↙
0.3893    0.3948
   -0.2838   -0.3044   -0.0263   -0.3454   -0.4087   -0.4170   -0.3704   -0.2687 ↙
-0.1121    0.0994
   -0.2893   -0.2138   -0.1994    0.8904   -0.1095   -0.1025   -0.0888   -0.0683 ↙
-0.0409   -0.0068
   -0.2950   -0.1214   -0.3044   -0.1366    0.8465   -0.1568   -0.1465   -0.1225 ↙
-0.0849   -0.0337
   -0.3007   -0.0272   -0.3410   -0.1442   -0.1725    0.8157   -0.1797   -0.1587 ↙
-0.1212   -0.0672
   -0.3064    0.0689   -0.3094   -0.1325   -0.1664   -0.1851    0.8115   -0.1767 ↙
-0.1496   -0.1072
   -0.3123    0.1667   -0.2096   -0.1016   -0.1354   -0.1591   -0.1728    0.8235 ↙
-0.1702   -0.1538
   -0.3181    0.2664   -0.0415   -0.0512   -0.0792   -0.1064   -0.1327   -0.1583 ↙
0.8170   -0.2069
   -0.3240    0.3679    0.1949    0.0184    0.0020   -0.0269   -0.0681   -0.1219 ↙
-0.1880    0.7334
   -0.3300    0.4712    0.4995    0.1073    0.1082    0.0794    0.0209   -0.0673 ↙
-0.1852   -0.3328

  Column 11

   -0.5336
    0.3738
    0.3660
    0.0342
    0.0311
    0.0032
   -0.0496
   -0.1274
   -0.2300
   -0.3576
    0.4899

 -366.6518  -34.8247   -3.3106
         0   -1.0454   -0.1992
         0         0    0.0027
         0         0         0
         0         0         0
         0         0         0
         0         0         0
         0         0         0
         0         0         0
         0         0         0
         0         0         0

    6.2354
 -121.1713
  588.0828

>>
```


10
8
6
4
2
0
-2
10
10.2
10.4
10.6
10.8
11

```


A=[10.0*10.0,10.0,1;10.1*10.1,10.1,1;10.2*10.2,10.2,1;10.3*10.3,10.3,1;10.4*10.4,10.4,1;10.5*10.5,10.5,1;10.6*10.6,10.6,1;10.7*10.7,10.7,1;10.8*10.8,10.8,1;10.9*10.9,10.9,1;11.0*11.0,11.0,1];
b=[0.0;0.0;1.0;1.5;2.5;3.5;4.5;5.0;6.5;8.0;10.0];
%disp("A=");
%disp(A);
for i=1:11
    temp(i,1)=A(i,2);
end

[U,S,V]=svd(A,0);

X= V*((U'*b)./diag(S));
disp(X);
for i=1:11
y(i,1)=X(1,1)*A(i,1)+X(2,1)*A(i,2)+X(3,1);
end
for i=1:11
hold on
plot(A(i,2),b(i,1),'r*');
end

plot(temp,y);
hold off
```

```
>> A4P8
    6.2354
 -121.1713
  588.0828

>>
```


10
8
6
4
2
0
-2
10
10.2
10.4
10.6
10.8
11