

Polynomial curve fitting

b) Matlab code:

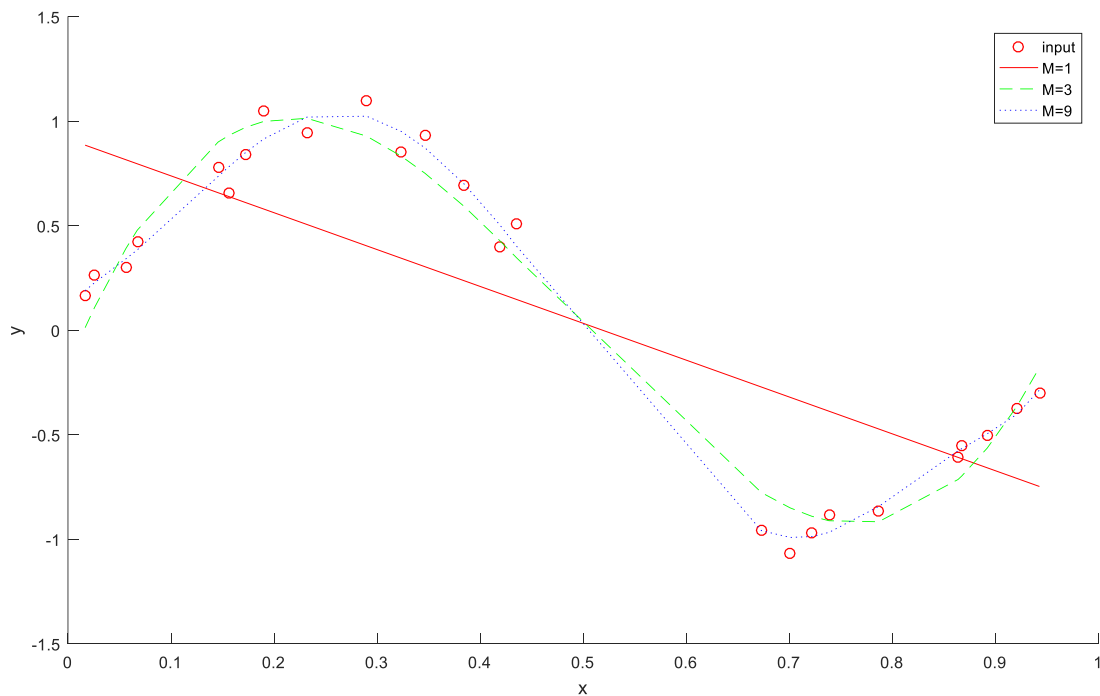
```
1. clear;
2. close all;
3. fclose all;
4.
5. % Generate data
6. rand('seed', 2);
7. randn('seed', 2);
8. n = 25;
9. x = rand(n, 1);
10. x = sort(x);
11. noise = 0.1 * randn(n, 1);
12. y = sin(2 * pi * x) + noise;
13.
14. %% problem b) without regularization term
15. % M = 1
16. M = 1;
17. m = M + 1;
18. X1 = zeros(n, m);
19. Y1 = zeros(n, 1);
20. for i = 1:n
21.     for j = 1:m
22.         X1(i, j) = x(i) ^ (j - 1);
23.     end
24. end
25. for i = 1:n
26.     Y1(i) = y(i);
27. end
28. % use CVX to solve
29. cvx_begin
30.     variable w1(m)
31.     minimize (norm(X1 * w1 - Y1, 2))
32. cvx_end
33.
34. % M = 3
35. M = 3;
36. m = M + 1;
37. X2 = zeros(n, m);
38. Y2 = zeros(n, 1);
39. for i = 1:n
40.     for j = 1:m
41.         X2(i, j) = x(i) ^ (j - 1);
42.     end
43. end
44. for i = 1:n
45.     Y2(i) = y(i);
46. end
47. % use CVX to solve
48. cvx_begin
49.     variable w2(m)
50.     minimize (norm(X2 * w2 - Y2, 2))
51. cvx_end
52.
53. % M = 9
54. M = 9;
55. m = M + 1;
56. X3 = zeros(n, m);
```

```

57. Y3 = zeros(n, 1);
58. for i = 1:n
59.     for j = 1:m
60.         X3(i, j) = x(i) ^ (j - 1);
61.     end
62. end
63. for i = 1:n
64.     Y3(i) = y(i);
65. end
66. % use CVX to solve
67. cvx_begin
68.     variable w3(m)
69.     minimize (norm(X3 * w3 - Y3, 2))
70. cvx_end
71.
72. % plot
73. y_est1 = X1 * w1;
74. y_est2 = X2 * w2;
75. y_est3 = X3 * w3;
76. hold on;
77. axis on;
78. xlabel('x');ylabel('y');
79. plot( x(:), y(:), 'ro');
80. plot( x(:), y_est1(:), 'r-' );
81. plot( x(:), y_est2(:), 'g--' );
82. plot( x(:), y_est3(:), 'b:' );
83. legend('input', 'M=1', 'M=3', 'M=9')
84. hold off;

```

result:



A little overfitting seems existed in M = 9;

c) add regularization term

Matlab code:

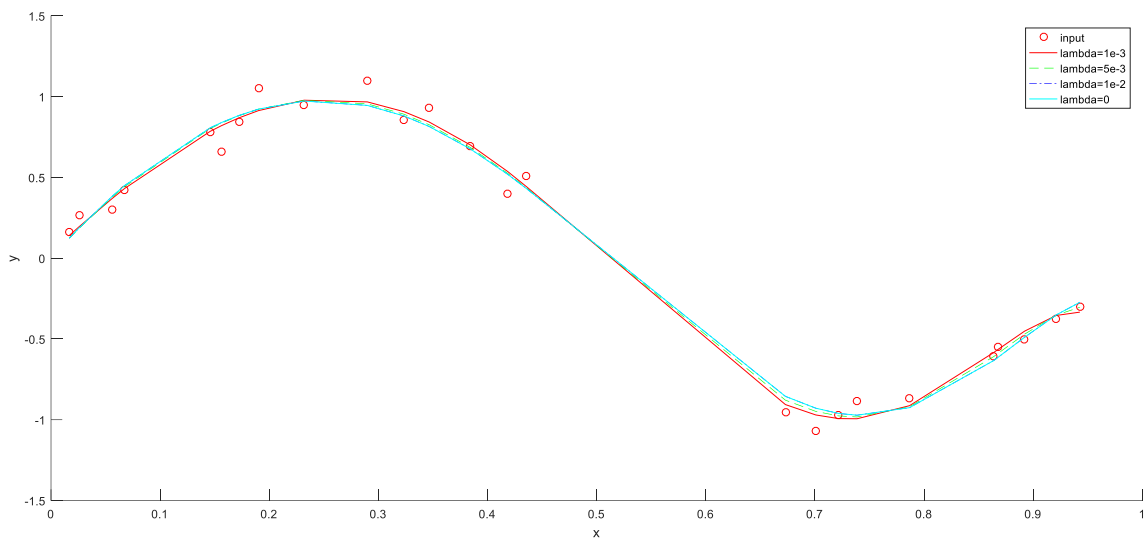
```
1. clear;
2. close all;
3. fclose all;
4.
5. % Generate data
6. rand('seed', 2);
7. randn('seed', 2);
8. n = 25;
9. x = rand(n, 1);
10. x = sort(x);
11. noise = 0.1 * randn(n, 1);
12. y = sin(2 * pi * x) + noise;
13.
14. %% problem c) with regularization term
15. % set lambda
16. lambda1 = 1e-3;
17. lambda2 = 5e-3;
18. lambda3 = 1e-2;
19. lambda4 = 0;
20. % M = 9
21. M = 9;
22. m = M + 1;
23. X3 = zeros(n, m);
24. Y3 = zeros(n, 1);
25. for i = 1:n
26.     for j = 1:m
27.         X3(i, j) = x(i) ^ (j - 1);
28.     end
29. end
30. for i = 1:n
31.     Y3(i) = y(i);
32. end
33. % use CVX to solve
34. cvx_begin
35.     variable w1(m)
36.     minimize (norm(X3 * w1 - Y3, 2) + lambda1 * norm(w1, 2))
37. cvx_end
38.
39. cvx_begin
40.     variable w2(m)
41.     minimize (norm(X3 * w2 - Y3, 2) + lambda2 * norm(w2, 2))
42. cvx_end
43.
44. cvx_begin
45.     variable w3(m)
46.     minimize (norm(X3 * w3 - Y3, 2) + lambda3 * norm(w3, 2))
47. cvx_end
48.
49. cvx_begin
50.     variable w4(m)
51.     minimize (norm(X3 * w4 - Y3, 2) + lambda4 * norm(w4, 2))
52. cvx_end
53.
54. % plot
55. y_est1 = X3 * w1;
56. y_est2 = X3 * w2;
```

```

57. y_est3 = X3 * w3;
58. y_est4 = X3 * w4;
59. hold on;
60. axis on;
61. xlabel('x');ylabel('y');
62. plot( x(:), y(:), 'ro');
63. plot( x(:), y_est1(:), 'r-' );
64. plot( x(:), y_est2(:), 'g--' );
65. plot( x(:), y_est3(:), 'b-.' );
66. plot( x(:), y_est3(:), 'c-' );
67. legend('input', 'lambda=1e-3', 'lambda=5e-3', 'lambda=1e-2', 'lambda=0')
68. hold off;

```

result:



The green curve and blue curve are nearly overlapped with each other. They are smoother than the others.

Effect of regularization term: encourage small weight values, make the model variance smaller, but with large bias.

3) compare L2 norm and L1 norm

Matlab code:

```

1. clear;
2. close all;
3. fclose all;
4.
5. % Generate data
6. rand('seed', 2);
7. randn('seed', 2);
8. n = 25;
9. x = rand(n, 1);
10. x = sort(x);
11. noise = 0.1 * randn(n, 1) + 3 * [zeros(15, 1); 1; zeros(9, 1)];

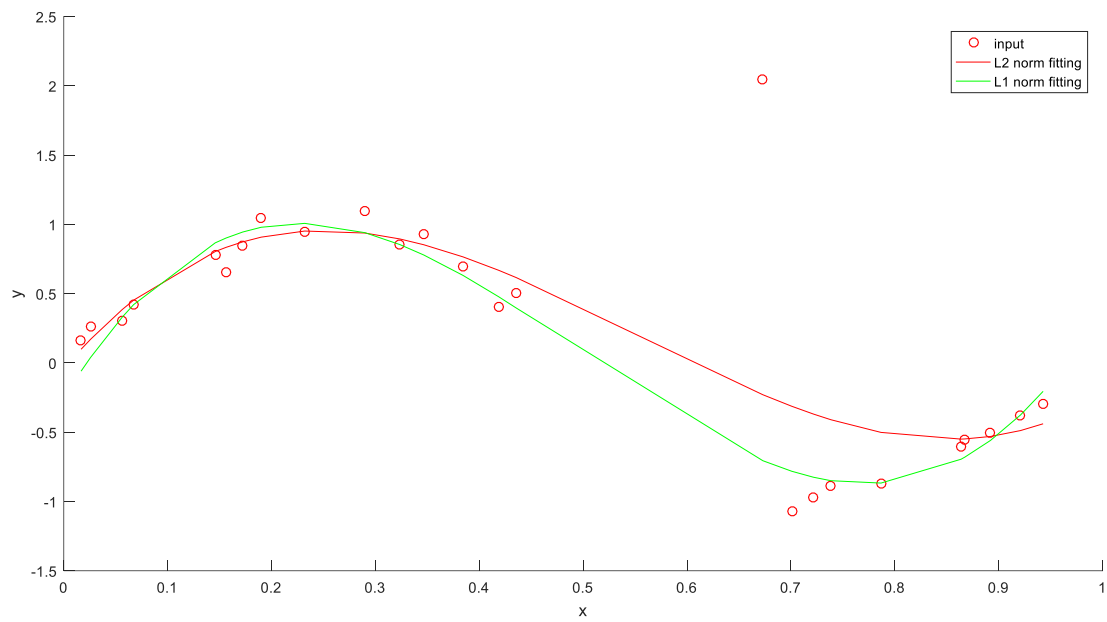
```

```

12. y = sin(2 * pi * x) + noise;
13.
14. %% problem c) with regularization term
15. % set lambda
16. % M = 9
17. M = 3;
18. m = M + 1;
19. X3 = zeros(n, m);
20. Y3 = zeros(n, 1);
21. for i = 1:n
22.     for j = 1:m
23.         X3(i, j) = x(i) ^ (j - 1);
24.     end
25. end
26. for i = 1:n
27.     Y3(i) = y(i);
28. end
29. % use CVX to solve
30. cvx_begin
31.     variable w1(m)
32.     minimize (norm(X3 * w1 - Y3, 2))
33. cvx_end
34.
35. cvx_begin
36.     variable w2(m)
37.     minimize (norm(X3 * w2 - Y3, 1))
38. cvx_end
39.
40. % plot
41. y_est1 = X3 * w1;
42. y_est2 = X3 * w2;
43. hold on;
44. axis on;
45. xlabel('x');ylabel('y');
46. plot( x(:), y(:), 'ro');
47. plot( x(:), y_est1(:), 'r-' );
48. plot( x(:), y_est2(:), 'g-' );
49. legend('input', 'L2 norm fitting', 'L1 norm fitting')
50. hold off;

```

result:



We can see that there is a very large noisy point (outliers) in the input data. L2 fitting is very sensitive to outliers, while L1 fitting is better.

Minimum Volume Covering Ellipsoid

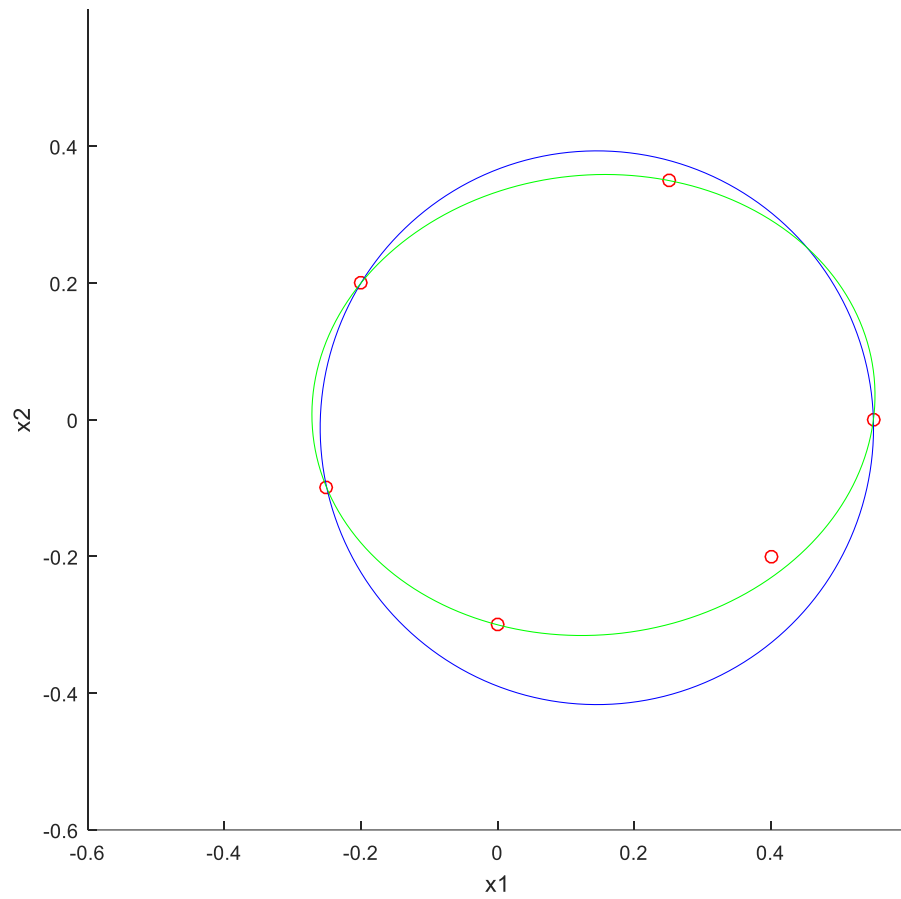
Matlab code:

```
1. clear;
2. close all;
3. fclose all;
4.
5. % Generate data
6. x = [ 0.55 0.0;
7.      0.25 0.35
8.      -0.2 0.2
9.      -0.25 -0.1
10.     -0.0 -0.3
11.     0.4 -0.2 ]';
12. [n,m] = size(x);
13. eyemat = eye(2, 2);
14.
15. % Create and solve the model for question a
16. cvx_begin
17.     variable A1
18.     variable b1(n)
19.     maximize(A1)
20.     subject to
21.         norms( A1 * eyemat * x + b1 * ones( 1, m ), 2 ) <= 1;
22. cvx_end
23.
24. % Create and solve the model for question b
25. cvx_begin
26.     variable A2(n, n)
27.     variable b2(n)
28.     maximize(det_rootn(A2))
29.     subject to
30.         norms( A2 * x + b2 * ones( 1, m ), 2 ) <= 1;
31. cvx_end
32.
33. % Plot the results
34. clf;
35. figure(1);
36.
37. noangles = 200;
38. angles = linspace( 0, 2 * pi, noangles );
39. ellipse1 = A1 * eyemat \ [ cos(angles) - b1(1) ; sin(angles) - b1(2) ];
40. ellipse2 = A2 \ [ cos(angles) - b2(1) ; sin(angles) - b2(2) ];
41. hold on;
42. axis on;
43. axis([-0.6 0.6 -0.6 0.6]); xlabel('x1');ylabel('x2');pbaspect([1, 1, 1])
44. plot( x(1,:), x(2,:), 'ro', ellipse1(1,:), ellipse1(2,:), 'b-' );
45. plot(ellipse2(1,:), ellipse2(2,:), 'g-' );
46. hold off;
```

result:

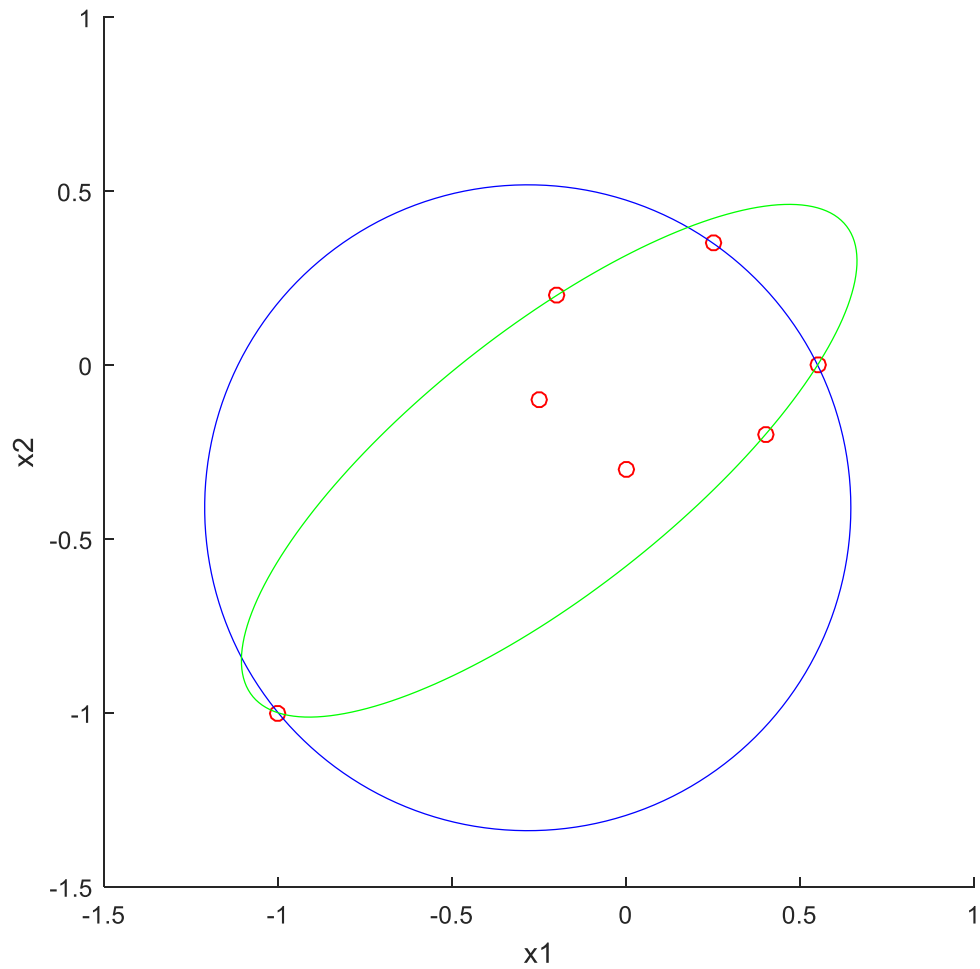
the blue curve (a circle) is the result of a).

the green curve (a ellipsoid) is the result of b).



c) add an extra point $(-1, -1)$, repeat a) and b).

the result becomes:



the parameter of a) is:

Ball center $(-0.2829, -0.4103)$, radius is 0.9284

The parameter of b) is:

Ellipsoid center $(-0.2213, -0.2752)$, shape matrix $\begin{pmatrix} 0.8135 & 0.3473 \\ 0.3473 & 0.6501 \end{pmatrix}$

The outlier makes the result far from the original one. Because the new outlier point is far from the original points, both the ball and the ellipsoid must go through the outlier. This problem is not robust to outliers.

