

|  |  |  |
| --- | --- | --- |
| Lambda=0 | M=1 | M=2 |
| Data10 | c=[-1.76; 0.96] | c=[-1.09; -0.66; 0.80] |
| Data15 | C=[-1.64; 0.78] | C=[ 1.48; -3.12; 1.01] |
| Data100 | C=[-1.93; 0.97] | C=[0.12; -2.05; 0.99] |
|  | M=6 | M=9 |
| Data10 | C=[-35.28; 70.14; -14.45; -29.34;  4.92; 3.39; 0.33] | C=[-39174.11; 167585.39; -296212.33;  278998.99; -150461.03; 46361.19; -7661.46; 571.27; -8.49; 0.30] |
| Data15 | C=[-1.25; -1.99; 21.51; -13.84; -9.16; 4.60; 0.35] | C=[17526.16; -73960.23; 129530.72; -122070.64; 67154.11; -21876.61; 4085.20; -408.77; 20.05; 0.32] |
| Data100 | C=[183.36; -627.67; 824.55; -489.24; 113.78; -4.64; 0.22] | C=[ -2217.77; 11580.15; -24870.40; 28694.69; -19522.55; 8133.27; -2070.28; 284.96; -11.85; 0.27] |

Lambda = exp(-10)

|  |  |  |
| --- | --- | --- |
| Lambda  =exp(-10) | M=1 | M=2 |
| Data10 | c=[-1.76; 0.96] | c=[-1.09; -0.66; 0.80] |
| Data15 | C=[-1.64; 0.78] | C=[ 1.48; -3.12; 1.01] |
| Data100 | C=[-1.93; 0.97] | C=[0.12; -2.05; 0.99] |
|  | M=6 | M=9 |
| Data10 | C=[-3.87; 9.54; 6.22; -8.52; -9.65; 5.72; 0.30] | C=[-3.53; -1.85; 2.30; 6.47; 6.81; 0.25; -9.31; -6.83; 5.10; 0.31] |
| Data15 | C=[-2.04; 5.69; 6.16; -2.13; -12.72; 4.92; 0.35] | C=[2.32; -3.49; -2.32; 2.43; 6.25; 4.83; -2.95; -12.00; 4.81; 0.35] |
| Data100 | C=[-24.95; 30.77; 30.39; -35.69; -8.85; 8.72; -0.12] | C=[17.54; -22.01; -21.07; 7.04; 33.83; 20.80; -38.38; -5.44; 8.07; -0.10] |

1. For data10, when we set lambda as 0, m=9, the curve is in some way over fitting, it goes too closely according to the data points provided. However, when we set lambda to exp(-10), the phenomenon of overfitting disappeared. The fittings are shown below.



For data100, the changing of lambda does not make much difference in the fitting curve but has much effect on regression parameter matrix c. Before introducing lambda, the elements in weight matrix is huge, after adding lambda they are much smaller. The fittings are shown below.



When data is scarce, low degree polynomials are recommended for regression, since by applying higher order polynomials, the curve may overfitting and the elements in weight matrix will be huge.

By utilizing regularization method, we set a proper lambda, the curve can fit well after adding the factor lambda. Since when we only use polynomials fitting, we expect a minimum in

which may generate over fitting case. By adding a factor lambda, that formula changes to

which means we can’t take the error with data points as the only criteria. The weight cannot be very large at the same time. In this way, we prevent overfitting.

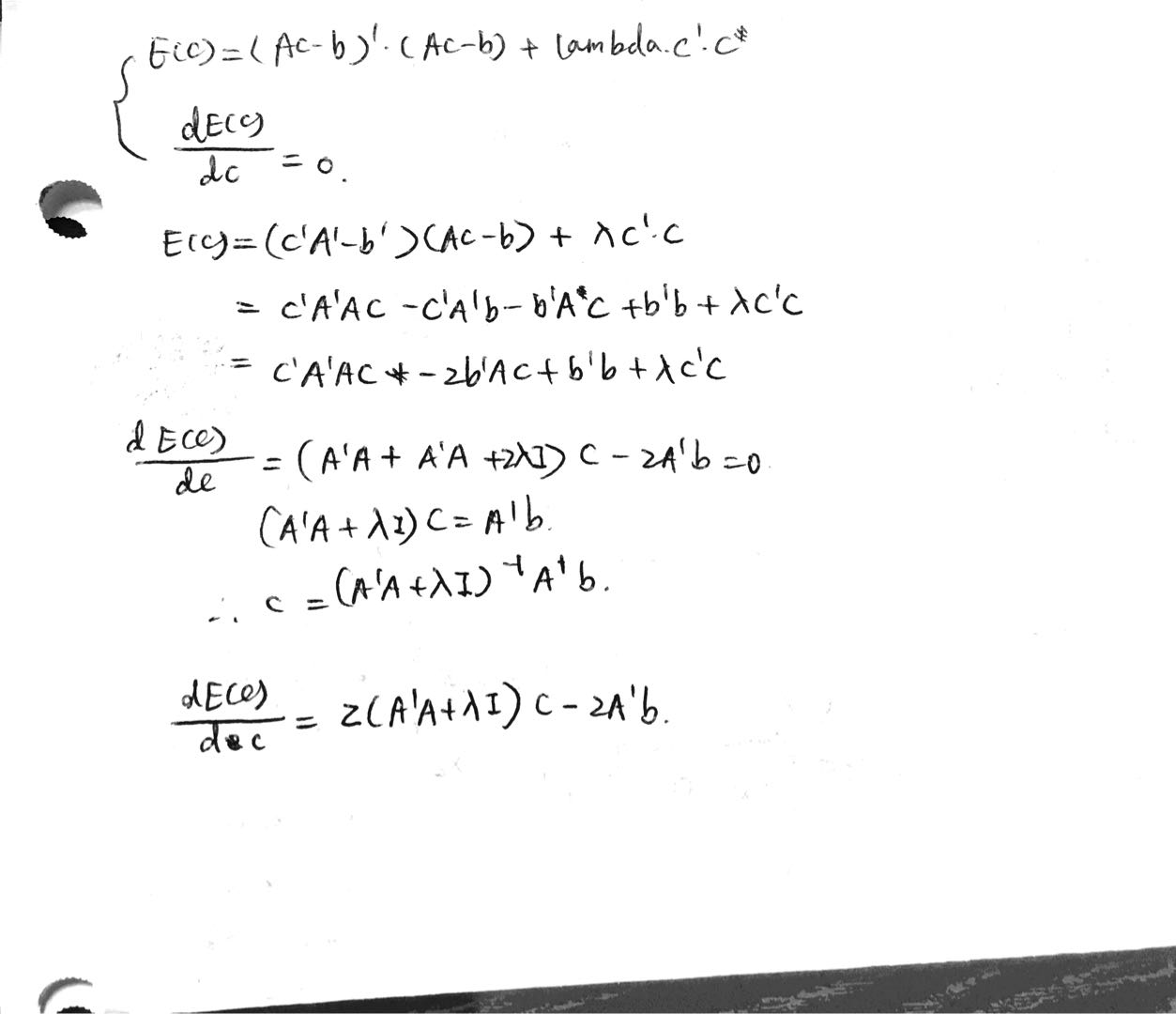
1. For data10, lambda=0:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | M=1 | M=2 | M=6 | M=9 |
| i | 0.6503 | 0.6662 | 0.3001 | 0.5038 |
| ii | 0.6832 | 0.6690 | 0.2576 | 0.0002 |



The above result shows that by increasing the m value, error between data points and estimation decreases. However, we can observe from the curve fitting result that m=9, lambda=0 is not a good fitting. Which demonstrates low error between fitting curve and data points neither means a low error between ground truth and estimation, nor means a good model. Overfitting may decrease error between data points but not error between ground truth.

1. The best value is m=6. Since it has a minimum error when using i to calculate the error between ground truth and estimation. Also, from the figures above, the curve fitting when m=6 is the closest to ground truth.



1. It won’t stuck in a local minimum, since there’s only one variable, so the Hessian matrix will only have one eigen value, which means there will only be one minimum. So it won’t stuck somewhere else.
2. Using gradient descent:

(set alpha to 0.005 instead of 0.01, it will converge perfectly)

-5.01410752669402

-0.698852405771098

3.24317049273975

5.99536900155817

6.28451628804124

2.51200573939767

-5.96943773142363

-13.3550576374343

7.31543183113573

0.0616174799470760

Difference:

22.5530541516936

-21.3076339525103

-24.3123037462126

1.04449794353538

27.5436776584116

18.2849652920149

-32.4134855523311

7.91626183882470

0.758667532548174

-0.159027641024549

1. When alpha=0.9, the value of c won’t converge since the step is too large. It will just go randomly between the function as shown below. When it’s close to the minimum point, due to large step size, it just step over the point and go farther.

