

HW4

Computer Exercise

Part a

In Solution PDF

Part b

In Solution PDF

Part c

i. Varying n, sigma constant

Code

```
1 % Part a and b done on paper
2 % Part c_i (varying n, sigma constant)
3
4 count = 0;
5 kernel = 3; % based on w_true = [1,3,2]'
6 mse_vector = zeros(5,1);
7 w_average_matrix = zeros(kernel,5);
8 w_true_matrix = zeros(kernel,5);
9
10
11 for n = [10,20,30,40,50]
12
13     count = count + 1;
14     A = zeros(2*n+1,kernel);
15     sigma = 1;
16     x_interim = -n:n;
17     x = transpose(x_interim*0.1); % turning to vector
18
19     % A generation
20
21     for i = 1:kernel
22         A(:,i) = x.^i;
23     end
24
25     % t generation
26     t = A(:,1) + 3*A(:,2) + 2*A(:,3);
27
28     w_true = pinv(A)*t; % calculating the true w directly from t
29     w_true_matrix(:,count) = w_true;
30
31     w_average_interim = zeros(kernel,1);
```

```

32
33 -     mse_interim = 0;
34
35 -     for j = 1:100
36
37         % v generation
38         v = sigma*randn(2*n+1,1);
39
40         % y generation
41         y = t + v;
42
43         w_estimated = pinv(A)*y;      % calculating the estimate of w
44
45         w_average_interim = w_average_interim + w_estimated; % running sum of w_average
46
47         mse_interim = mse_interim + (norm(w_true - w_estimated))^2; % running sum of mse
48
49
50     end
51
52     mse = mse_interim/100;
53     w_average = w_average_interim/100;
54
55     mse_vector(count,1) = mse;
56     w_average_matrix(:,count) = w_average;
57
58
59
60 - end

```

```

61
62 - n_range = [10;20;30;40;50];
63 - disp("The matrix of true w, where each column corresponds to a value of n and the column vector the vector of w, is")
64 - disp(w_true_matrix)
65 - disp("The matrix of estimated w (averaged over 100 iterations, for each n), where each column corresponds to a value of n and the column vector the vector of w, is")
66 - disp(w_average_matrix)
67 - disp("The averaged MSE over 100 iterations for each n, where rows represent MSE value for each n, is")
68 - disp(mse_vector)
69
70 - plot(n_range,mse_vector,'-o')
71 - xlabel('values of n, sigma = 1 (constant)')
72 - ylabel('MSE')
73

```

Results

The matrix of true w, where each column corresponds to a value of n and the column vector the vector of w, is,

1.0000	1.0000	1.0000	1.0000	1.0000
3.0000	3.0000	3.0000	3.0000	3.0000
2.0000	2.0000	2.0000	2.0000	2.0000

As expected, it is [1,3,2]

The matrix of estimated w (averaged over 100 iterations, for each n), where each column corresponds to a value of n and the column vector the vector of averaged w estimate, is,

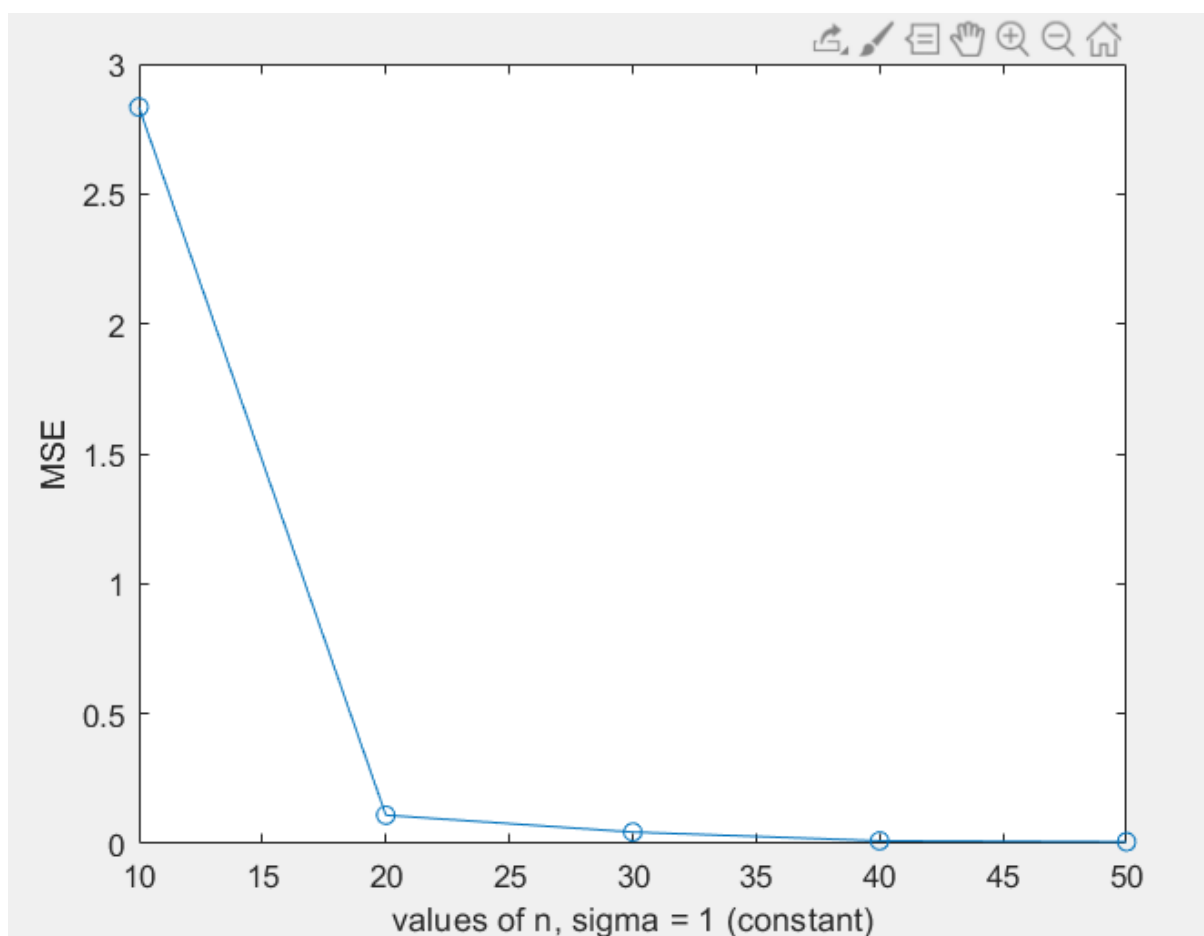
0.9333	1.0302	1.0023	1.0099	1.0042
3.0322	2.9993	2.9981	2.9999	3.0000
2.0261	1.9950	2.0001	1.9990	1.9996

As n increases, the average (over 100 trials) of estimate of w becomes more accurate.

The averaged MSE over 100 iterations for each n , where rows represent MSE value for each n , is,

2.3952
0.1106
0.0316
0.0174
0.0076

The MSE values also reduce as the n value increases.



The above shows the MSE versus n plot with sigma constant

ii. Varying sigma, n constant (n = 20)

Code

```
1      % Part c_ii (varying sigma, n constant)
2
3
4 -    kernel = 3;                % based on w_true = [1,3,2]'
5 -    mse_vector = zeros(4,1);
6 -    w_average_matrix = zeros(kernel,4);
7
8
9 -    n = 20;
10 -    A = zeros(2*n+1,kernel);
11 -    x_interim = -n:n;
12 -    x = transpose(x_interim*0.1); % turning to vector
13
14     % A generation
15
16 -    for i = 1:kernel
17 -        A(:,i) = x.^i;
18 -    end
19
20     % t generation
21 -    t = A(:,1) + 3*A(:,2) + 2*A(:,3);
22
23 -    mp_i = pinv(A); % as A stays same, irrespective of sigma, for a given n, we
24
25 -    w_true = mp_i*t;          % calculating the true w directly from t
26
27 -    count = 0;
28
29 -    for sigma = [0.1,1,3,5]
30
31 -        count = count + 1;
32
```

```

32
33 -     w_average_interim = zeros(kernel,1);
34
35 -     mse_interim = 0;
36
37 -     for j = 1:100
38
39         % v generation
40         v = sigma*randn(2*n+1,1);
41
42         % y generation
43         y = t + v;
44
45         w_estimated = mp_i*y;           % calculating the estimate of w
46
47         w_average_interim = w_average_interim + w_estimated; % running sum
48
49         mse_interim = mse_interim + (norm(w_true - w_estimated))^2; % runni
50
51
52 -     end
53
54     mse = mse_interim/100;
55     w_average = w_average_interim/100;
56
57     mse_vector(count,1) = mse;
58     w_average_matrix(:,count) = w_average;
59
60
61
62 - end

```

```

63
64 -     sigma_range = [0.1;1;3;5];
65 -     disp("The matrix of true w, obtained same for all sigma (obviously) is")
66 -     disp(w_true)
67 -     disp("The matrix of estimated w (averaged over 100 iterations, for each sig
68 -     disp(w_average_matrix)
69 -     disp("The averaged MSE over 100 iterations for each sigma, where rows repre
70 -     disp(mse_vector)
71
72 -     plot(sigma_range,mse_vector,'-o')
73 -     xlabel('values of sigma, n = 20 (constant)')
74 -     ylabel('MSE')
75
76

```

The code is pretty similar to c, Part i.

Results

The matrix of true w , obtained same for all sigma (obviously) is,

```
1.0000
3.0000
2.0000
```

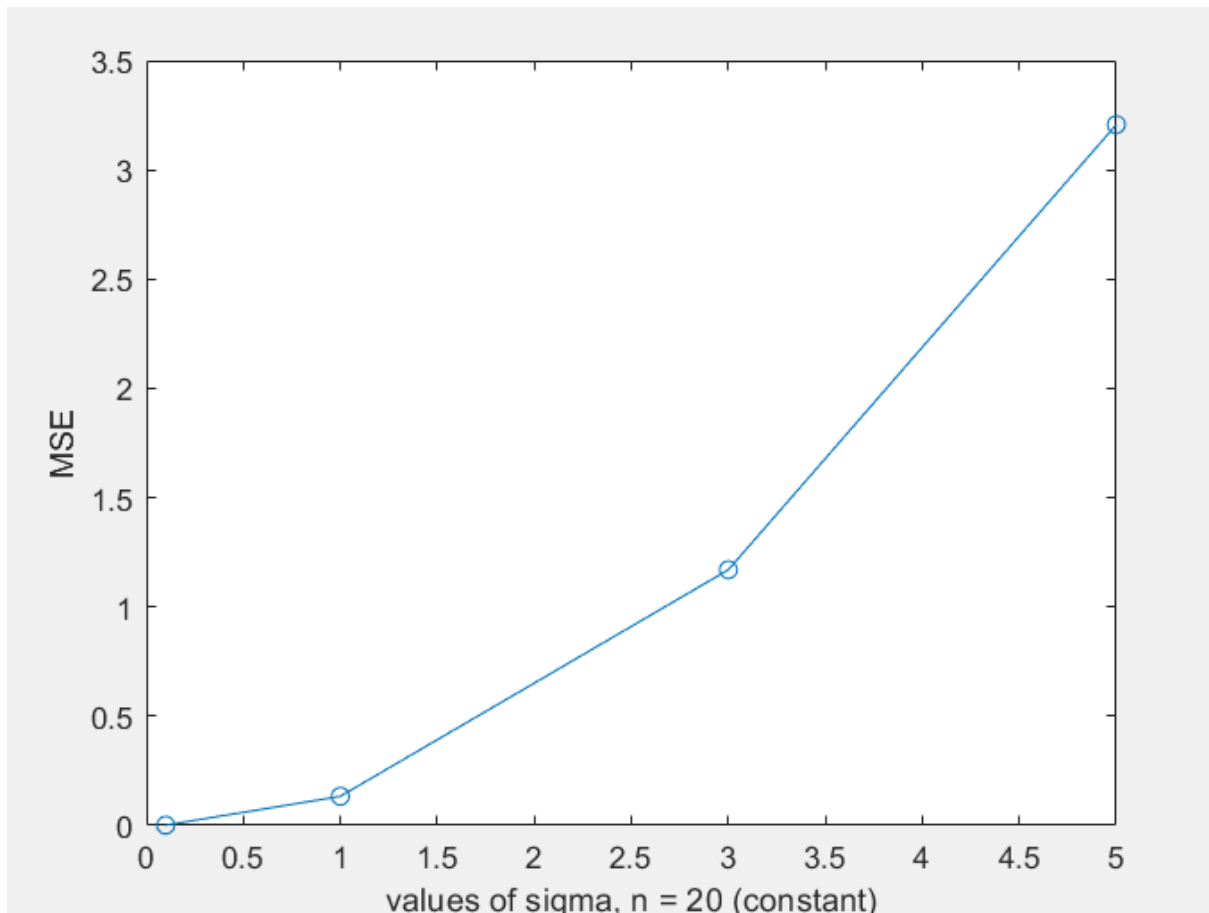
The matrix of estimated w (averaged over 100 iterations, for each sigma), where each column corresponds to a value of sigma and the column vector the vector of averaged w estimate, is,

```
0.9986    0.9978    1.1565    1.2948
3.0007    2.9989    2.9924    3.0163
2.0009    1.9951    1.9407    1.9570
```

The averaged MSE over 100 iterations for each sigma, where rows represent MSE value for each sigma, is,

```
0.0012
0.1332
1.1697
3.2069
```

The curve is,



As expected, as the variance of the added noise increases, the MSE increases and the estimate of \mathbf{w} becomes more and more inaccurate.

Variance of $\mathbf{z}^* \sigma = (\sigma)^2$ Variance of $\mathbf{z} = (\sigma)^2$

Since \mathbf{z} is standard normal, with zero mean and unit variance.

Thus, as the value of added distortion to \mathbf{t} increases, error in \mathbf{w} , increases.

$$\mathbf{y} = \mathbf{t} + \mathbf{v} = \mathbf{t} + \mathbf{z}^* \sigma$$

Thus, by increasing σ , we increase both the variance (how much the noise varies) as well as the overall value of the noise (the probability of larger values appearing in noise term increases as the value of σ increases).

Thus, with increase in σ , due to these reasons, MSE increases.

iii. For Model with x and x^3 (underfit)

PART 1 – Varying n with sigma constant

Code

```
1 % c_iii_Part1
2
3 % NOTE :
4 % We assume that the t observed is observed still from the true model with
5 % the added noise. Our model has changed, true model has not.
6
7 % Thus, A will be formed with respect to our new model, w corresponds to w
8 % of our new model, y = t + v is generated with v generated as before and t
9 % generated according to our true model, not the mismatched model. We
10 % observe x and t and we are simulating using y, the observed t. Thus y
11 % doesn't change despite the change in our model. Thus the rule to generate
12 % t still stays the same.
13
14 % w true will be obtained without the noise added and w estimate with the
15 % noise added.
16
17 % t according to our new mismatched model will be calculated after we
18 % obtain w, using the new model. t true will then be the observed values of
19 % t, or the values to which model is fit, i.e. true t value is the one
20 % generated by the true model alongside the added error (or t which will be
21 % used to obtain w)
22
23 % NOTE : True t and obtained t won't be stored for each n (averaged
24 % obviously over 100 trials) and stored, since it hasn't been asked, and we
25 % don't want to increase unnecessary computation. Obviously doing so
26 % wouldn't at all be difficult and would need around 1-2 more lines of
27 % code. So it has been avoided here.
28
29 % Part c_iii_Model1_Part1 (varying n, sigma constant) (Model is the one
30 % which is underfit to the true model, i.e. the non-noisy version of
31 % the true model)
32
33
```

```

32
33 - count = 0;
34 - kernel = 3; % for t generation, the true one
35 - mse_vector = zeros(5,1);
36 - recon_mse_vector = zeros(5,1);
37 - kernel_mis = 2;
38 - w_average_matrix = zeros(kernel_mis,5);
39 - w_true_matrix = zeros(kernel_mis,5);
40
41
42 - □ for n = [10,20,30,40,50]
43
44 -     count = count + 1;
45 -     A = zeros(2*n+1,kernel);
46 -     sigma = 1;
47 -     x_interim = -n:n;
48 -     x = transpose(x_interim*0.1); % turning to vector
49
50     % A generation, true A, A that gives true t
51
52 - □ for i = 1:kernel
53 -     A(:,i) = x.^i;
54 - end
55
56     % t generation
57 - t = A(:,1) + 3*A(:,2) + 2*A(:,3); % true t
58
59     % A, for our model, generation

```

```

60
61 - A_mis = zeros(2*n+1, kernel_mis);
62
63 - for i = 1:kernel_mis
64 -     A_mis(:,i) = x.^(2*i-1);           % our kernel model
65 - end
66
67
68 - w_true = pinv(A_mis)*t;
69 - w_true_matrix(:,count) = w_true;
70
71 - w_average_interim = zeros(kernel_mis,1);
72
73 - mse_interim = 0;
74
75 - recon_mse_interim = 0;
76
77 - for j = 1:100
78
79     % v generation
80     v = sigma*randn(2*n+1,1);
81
82     % y generation
83     y = t + v;           % observed y
84
85     w_estimated = pinv(A_mis)*y;           % calculating the estimate of w
86
87     y_estimated = A_mis*w_estimated;
88
89     w_average_interim = w_average_interim + w_estimated; % running sum of w_average
90
91     mse_interim = mse_interim + (norm(w_true - w_estimated))^2; % running sum of mse

```

```

92         recon_mse_interim = recon_mse_interim + (norm(y - y_estimated))^2;
93     end
94
95     mse = mse_interim/100;
96     recon_mse = recon_mse_interim/100;
97     w_average = w_average_interim/100;
98
99     mse_vector(count,1) = mse;
100     recon_mse_vector(count,1) = recon_mse;
101     w_average_matrix(:,count) = w_average;
102
103 end
104
105 n_range = [10;20;30;40;50];
106 disp("The matrix of true w for our model, where each column corresponds to
107 disp(w_true_matrix)
108 disp("The matrix of estimated w for our model(averaged over 100 iterations,
109 disp(w_average_matrix)
110 disp("The averaged MSE of w over 100 iterations for each n, where rows rep
111 disp(mse_vector)
112 disp("The averaged MSE of reconfiguration over 100 iterations for each n, w
113 disp(recon_mse_vector)
114
115 figure(1);
116 plot(n_range,mse_vector,'o-')
117 xlabel('values of n, sigma = 1 (constant)')
118 ylabel('MSE for w')
119
120
121 figure(2);
122 plot(n_range,recon_mse_vector,'o-')
123 xlabel('values of n, sigma = 1 (constant)')
124 ylabel('MSE for reconfiguration')
125

```

Results

>> c_iii_Model1_Part1

The matrix of true w for our model, where each column corresponds to a value of n and the column vector the vector of w, is,

1.0000	1.0000	1.0000	1.0000	1.0000
2.0000	2.0000	2.0000	2.0000	2.0000

The matrix of estimated w for our model (averaged over 100 iterations, for each n), where each column corresponds to a value of n and the column vector the vector of averaged w estimate, is,

0.9973	0.9585	1.0251	1.0157	0.9945
1.9959	2.0160	1.9950	1.9988	2.0002

The averaged MSE of w over 100 iterations for each n , where rows represent MSE of w value for each n , is,

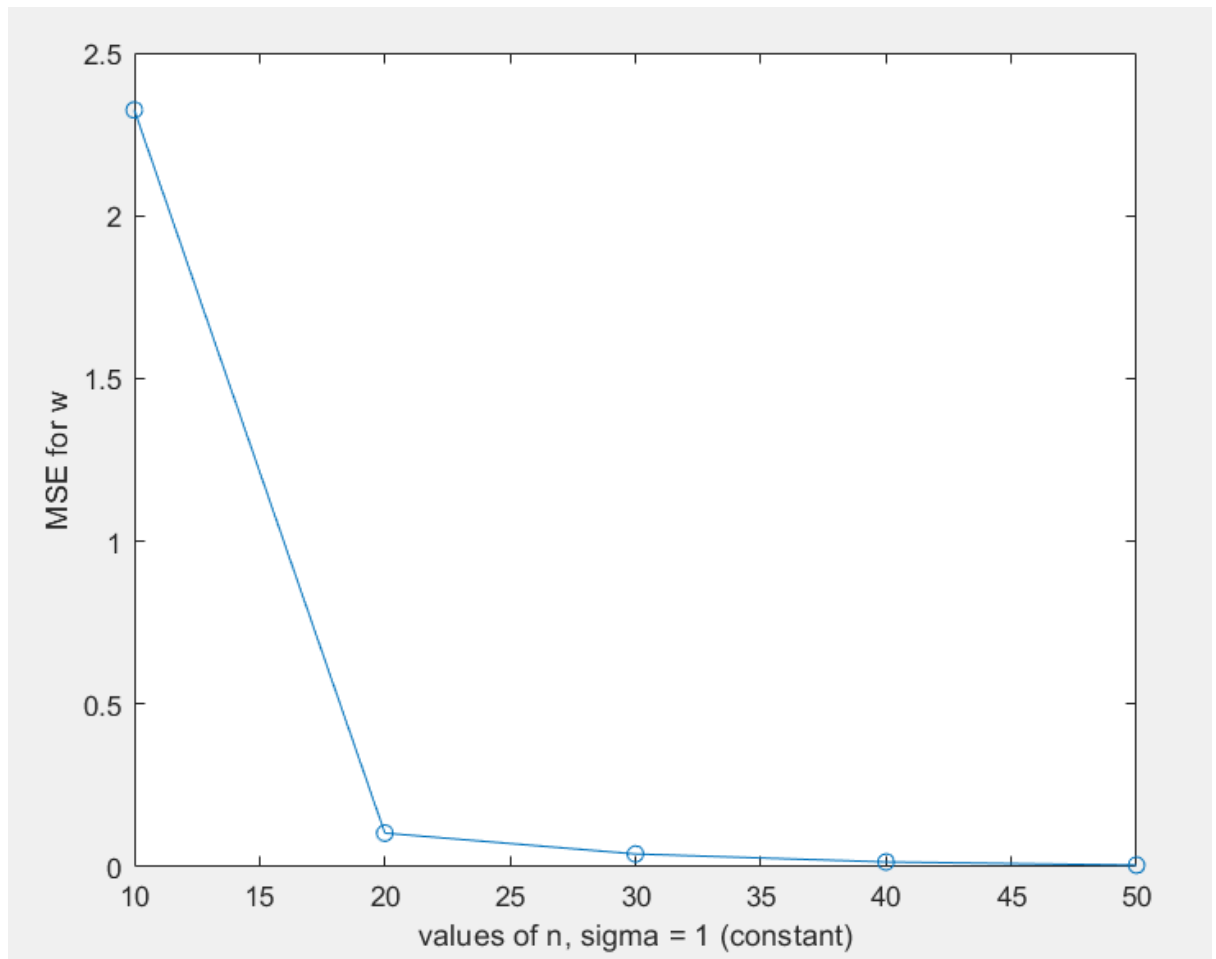
2.3247
0.1036
0.0395
0.0149
0.0052

The averaged MSE of reconfiguration over 100 iterations for each n , where rows represent MSE of reconfiguration value for each n , is,

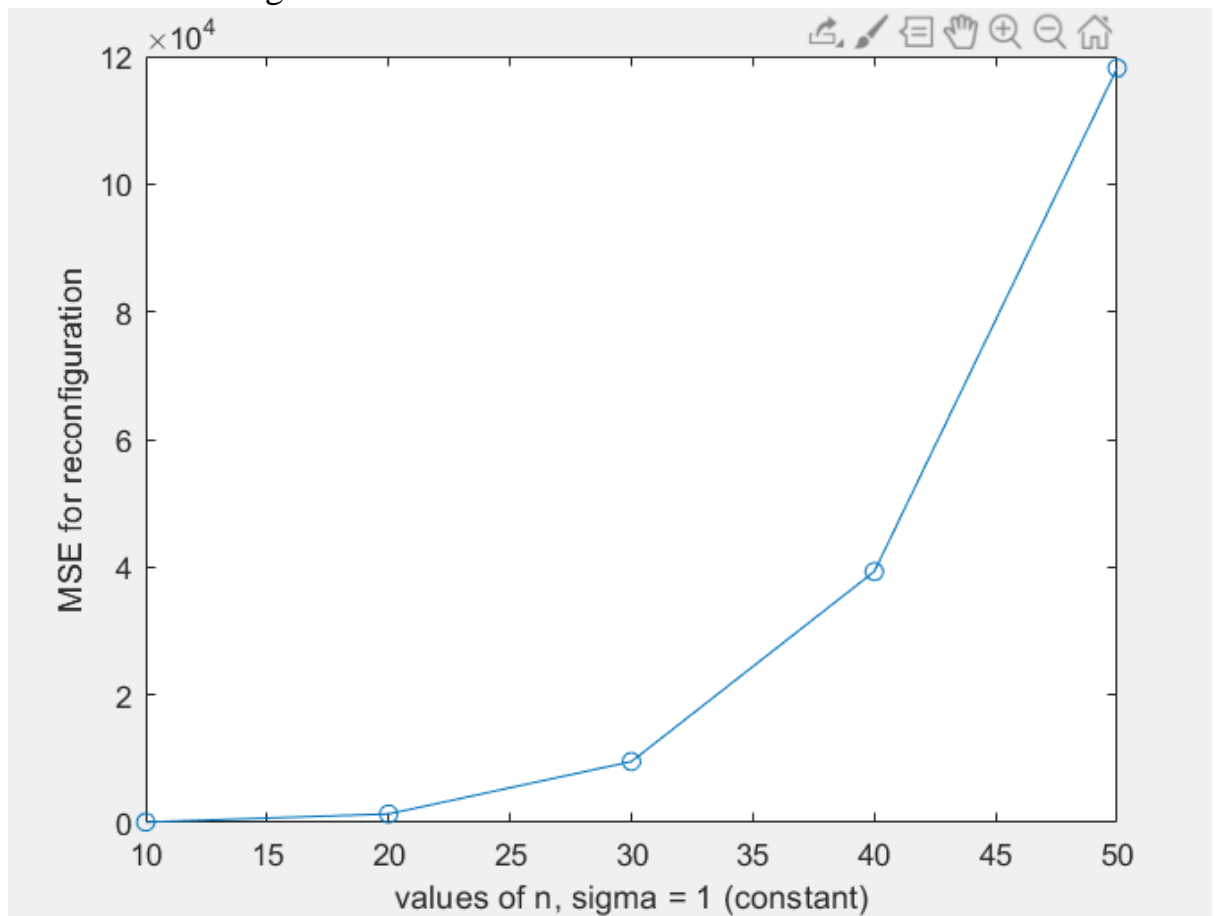
1.0e+05 *
0.0006
0.0134
0.0955
0.3931
1.1815

Plots

MSE for w



MSE for Reconfiguration



The MSE for reconfiguration is extremely large, since, the coefficients of x and x^3 are still close to 1 and 2 as before, but now, we have the x^2 factor missing due to which our estimated y deviates largely from our actual y in general.

PART 2 – Varying sigma with n constant, same model

Code

```

1      % Part c_iii_Modell_Part2 (varying sigma, n constant) (Model is the one
2      % which is underfit to the true model, i.e. the non-noisy version of
3      % the true model, with x and x^3 only)
4
5      % Everything same, but now only sigma varies
6
7
8      count = 0;
9      kernel = 3;          % for t generation, the true one
10     mse_vector = zeros(4,1);
11     recon_mse_vector = zeros(4,1);
12     kernel_mis = 2;
13     w_average_matrix = zeros(kernel_mis,4);
14
15
16     n = 20;
17     A = zeros(2*n+1,kernel);
18     x_interim = -n:n;
19     x = transpose(x_interim*0.1); % turning to vector
20
21     % A generation, true A, A that gives true t
22
23     for i = 1:kernel
24         A(:,i) = x.^i;
25     end
26
27     % t generation
28     t = A(:,1) + 3*A(:,2) + 2*A(:,3);          % true t
29
30     % A, for our model, generation
31
32     A_mis = zeros(2*n+1,kernel_mis);

```



```

33
34 - for i = 1:kernel_mis
35 -     A_mis(:,i) = x.^(2*i-1);           % our kernel model
36 - end
37
38 - pm_i = pinv(A_mis);
39 - w_true = pm_i*t;
40
41 - for sigma = [0.1,1,3,5]
42
43 -     count = count + 1;
44
45 -     w_average_interim = zeros(kernel_mis,1);
46
47 -     mse_interim = 0;
48
49 -     recon_mse_interim = 0;
50
51 -     for j = 1:100
52
53 -         % v generation
54 -         v = sigma*randn(2*n+1,1);
55
56 -         % y generation
57 -         y = t + v;           % observed y
58
59 -         w_estimated = pm_i*y;   % calculating the estimate of w
60
61 -         y_estimated = A_mis*w_estimated;
62
63 -         w_average_interim = w_average_interim + w_estimated; % running sum of w_average
64

```

```

65 -         mse_interim = mse_interim + (norm(w_true - w_estimated))^2; % runni
66 -
67 -         recon_mse_interim = recon_mse_interim + (norm(y - y_estimated))^2;
68 -
69 -     end
70 -
71 -     mse = mse_interim/100;
72 -     recon_mse = recon_mse_interim/100;
73 -     w_average = w_average_interim/100;
74 -
75 -     mse_vector(count,1) = mse;
76 -     recon_mse_vector(count,1) = recon_mse;
77 -     w_average_matrix(:,count) = w_average;
78 -
79 -
80 -
81 - end
82 -
83 - sigma_range = [0.1;1;3;5];
84 - disp("The true w vector for our model, which is obviously constant for all
85 - disp(w_true)
86 - disp("The matrix of estimated w for our model(averaged over 100 iterations,
87 - disp(w_average_matrix)
88 - disp("The averaged MSE of w over 100 iterations for each sigma, where rows
89 - disp(mse_vector)
90 - disp("The averaged MSE of reconfiguration over 100 iterations for each sigma
91 - disp(recon_mse_vector)
92 -
93 - figure(1);
94 - plot(sigma_range,mse_vector,'o-')
95 - xlabel('values of sigma, n = 20 (constant)')
96 -
97 -
98 - ylabel('MSE for w')
99 -
100 - figure(2);
101 - plot(sigma_range,recon_mse_vector,'o-')
102 - xlabel('values of sigma, n = 20 (constant)')
103 - ylabel('MSE for reconfiguration')

```

Results

>> c_iii_Model1_Part2

The true w vector for our model, which is obviously constant for all sigma, since n is constant is

1.0000

2.0000

The matrix of estimated w for our model(averaged over 100 iterations, for each sigma), where each column corresponds to a value of sigma and the column vector the vector of averaged w estimate, is

0.9991	0.9937	0.8778	0.7348
2.0004	2.0037	2.0665	2.1051

The averaged MSE of w over 100 iterations for each sigma, where rows represent MSE of w value for each sigma, is

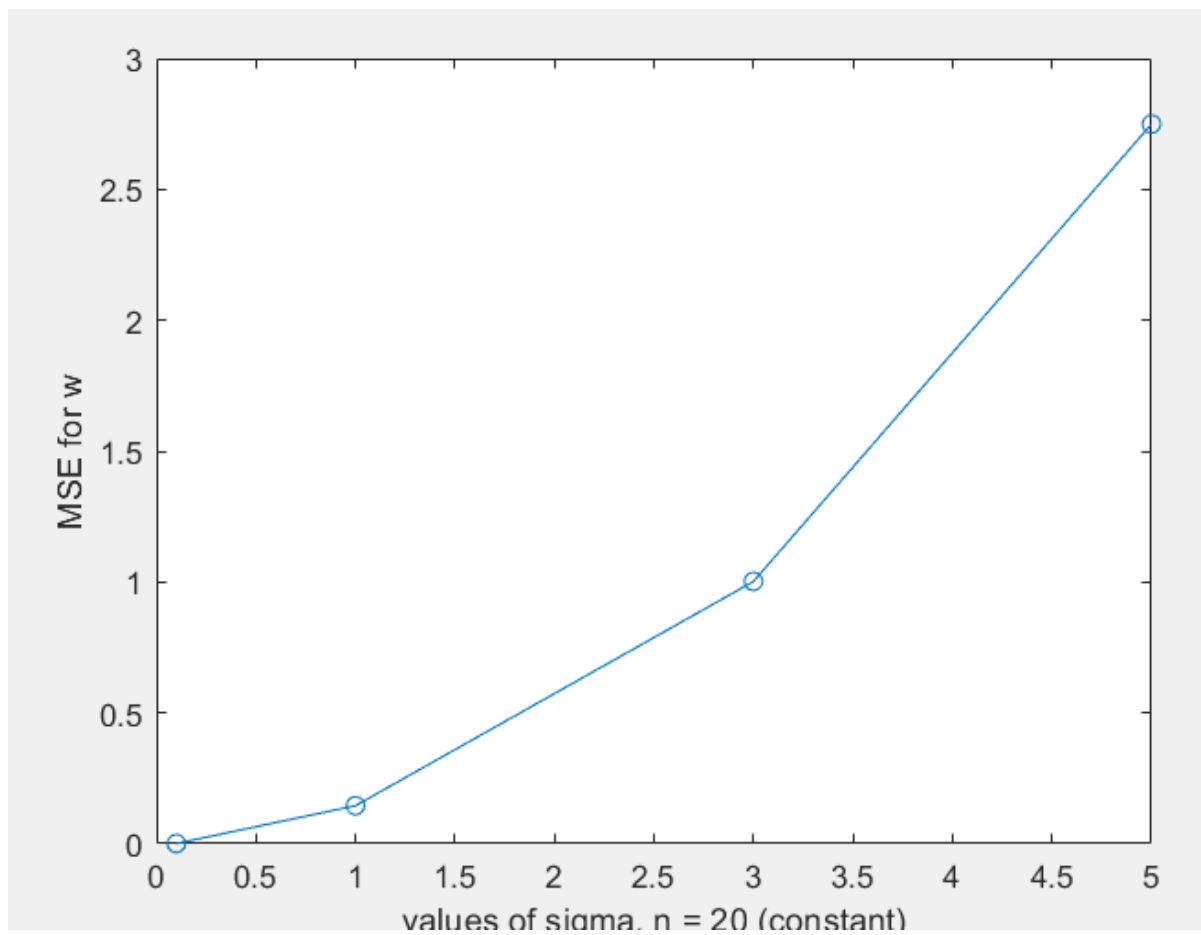
0.0010
0.1453
1.0021
2.7495

The averaged MSE of reconfiguration over 100 iterations for each sigma, where rows represent MSE of reconfiguration value for each sigma, is

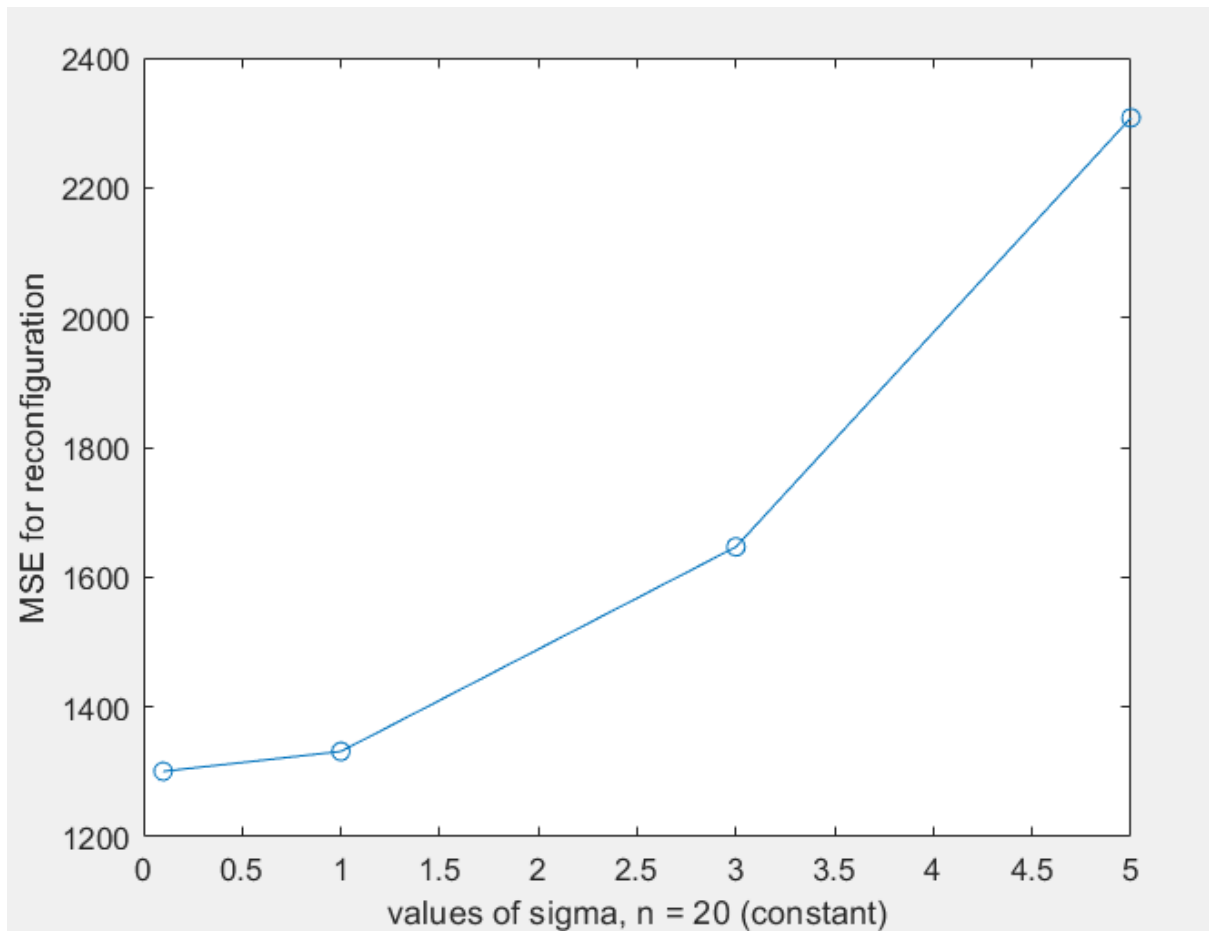
1.0e+03 *
1.3007
1.3314
1.6466
2.3078

Plots

MSE for w



MSE for Reconfiguration



As seen before, as sigma increases, both the MSE has a tendency to increase. The same previous explanation goes here as well.

iii. For Model with terms till and including x^4 (overfit)

PART 1 – Varying n with sigma constant

Code

```

1 % Part c_iii_Model2_Part1 (varying n, sigma constant) (Model is the one
2 % which is overfit to the true model)
3
4 % Things same as before
5
6 count = 0;
7 kernel = 3; % for t generation, the true one
8 mse_vector = zeros(5,1);
9 recon_mse_vector = zeros(5,1);
10 kernel_mis = 5; % the overfit model
11 w_average_matrix = zeros(kernel_mis,5);
12 w_true_matrix = zeros(kernel_mis,5);
13
14
15 for n = [10,20,30,40,50]
16
17     count = count + 1;
18     A = zeros(2*n+1,kernel);
19     sigma = 1;
20     x_interim = -n:n;
21     x = transpose(x_interim*0.1); % turning to vector
22
23     % A generation, true A, A that gives true t
24
25     for i = 1:kernel
26         A(:,i) = x.^i;
27     end
28
29     % t generation
30     t = A(:,1) + 3*A(:,2) + 2*A(:,3); % true t
31
32     % A, for our model, generation

```

```

34 - A_mis = zeros(2*n+1, kernel_mis);
35
36 - for i = 1:kernel_mis
37 -     A_mis(:,i) = x.^(i-1);           % our kernel model
38 - end
39
40
41 - w_true = pinv(A_mis)*t;
42 - w_true_matrix(:,count) = w_true;
43
44 - w_average_interim = zeros(kernel_mis,1);
45
46 - mse_interim = 0;
47
48 - recon_mse_interim = 0;
49
50 - for j = 1:100
51
52 -     % v generation
53 -     v = sigma*randn(2*n+1,1);
54
55 -     % y generation
56 -     y = t + v;           % observed y
57
58 -     w_estimated = pinv(A_mis)*y;           % calculating the estimate of w
59
60 -     y_estimated = A_mis*w_estimated;
61
62 -     w_average_interim = w_average_interim + w_estimated; % running sum of w
63
64 -     mse_interim = mse_interim + (norm(w_true - w_estimated))^2; % running su

```

```

65
66 -         recon_mse_interim = recon_mse_interim + (norm(y - y_estimated))^2
67
68 -     end
69
70 -     mse = mse_interim/100;
71 -     recon_mse = recon_mse_interim/100;
72 -     w_average = w_average_interim/100;
73
74 -     mse_vector(count,1) = mse;
75 -     recon_mse_vector(count,1) = recon_mse;
76 -     w_average_matrix(:,count) = w_average;
77
78
79
80 - end
81
82 - n_range = [10;20;30;40;50];
83 - disp("The matrix of true w for our model, where each column corresponds t
84 - disp(w_true_matrix)
85 - disp("The matrix of estimated w for our model(averaged over 100 iteration
86 - disp(w_average_matrix)
87 - disp("The averaged MSE of w over 100 iterations for each n, where rows re
88 - disp(mse_vector)
89 - disp("The averaged MSE of reconfiguration over 100 iterations for each n,
90 - disp(recon_mse_vector)
91
92 - figure(1);
93 - plot(n_range,mse_vector,'o-')
94 - xlabel('values of n, sigma = 1 (constant)')
95 - ylabel('MSE for w')
96
91
92 - figure(1);
93 - plot(n_range,mse_vector,'o-')
94 - xlabel('values of n, sigma = 1 (constant)')
95 - ylabel('MSE for w')
96
97 - figure(2);
98 - plot(n_range,recon_mse_vector,'o-')
99 - xlabel('values of n, sigma = 1 (constant)')
100 - ylabel('MSE for reconfiguration')
101
102
103

```

Results

>> c_iii_Model2_Part1

The matrix of true w for our model, where each column corresponds to a value of n and the column vector the vector of w , is

0.0000	0.0000	-0.0000	0.0000	-0.0000
1.0000	1.0000	1.0000	1.0000	1.0000
3.0000	3.0000	3.0000	3.0000	3.0000
2.0000	2.0000	2.0000	2.0000	2.0000
0.0000	0.0000	-0.0000	-0.0000	-0.0000

The matrix of estimated w for our model(averaged over 100 iterations, for each n), where each column corresponds to a value of n and the column vector the vector of averaged w estimate, is

-0.0809	-0.0246	-0.0109	-0.0084	0.0028
0.9675	1.0193	0.9783	1.0082	0.9960
3.3731	3.0623	3.0005	2.9953	2.9994
2.0593	1.9947	2.0051	1.9995	2.0006
-0.2965	-0.0184	-0.0003	0.0005	-0.0000

The averaged MSE of w over 100 iterations for each n , where rows represent MSE of w value for each n , is

14.9549
0.4332
0.1175
0.0672
0.0372

The averaged MSE of reconfiguration over 100 iterations for each n , where rows represent MSE of reconfiguration value for each n , is

14.9236
35.6173

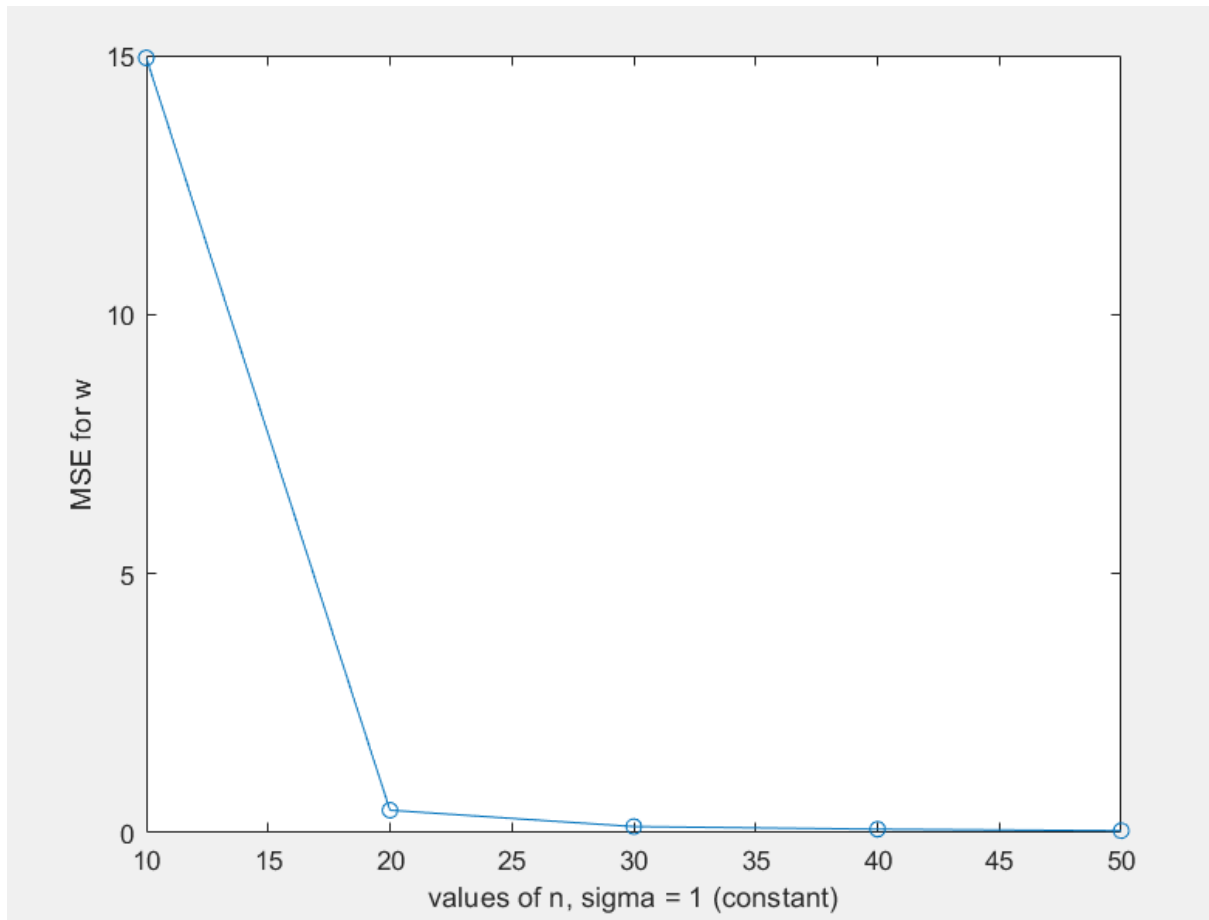
56.9790

76.7711

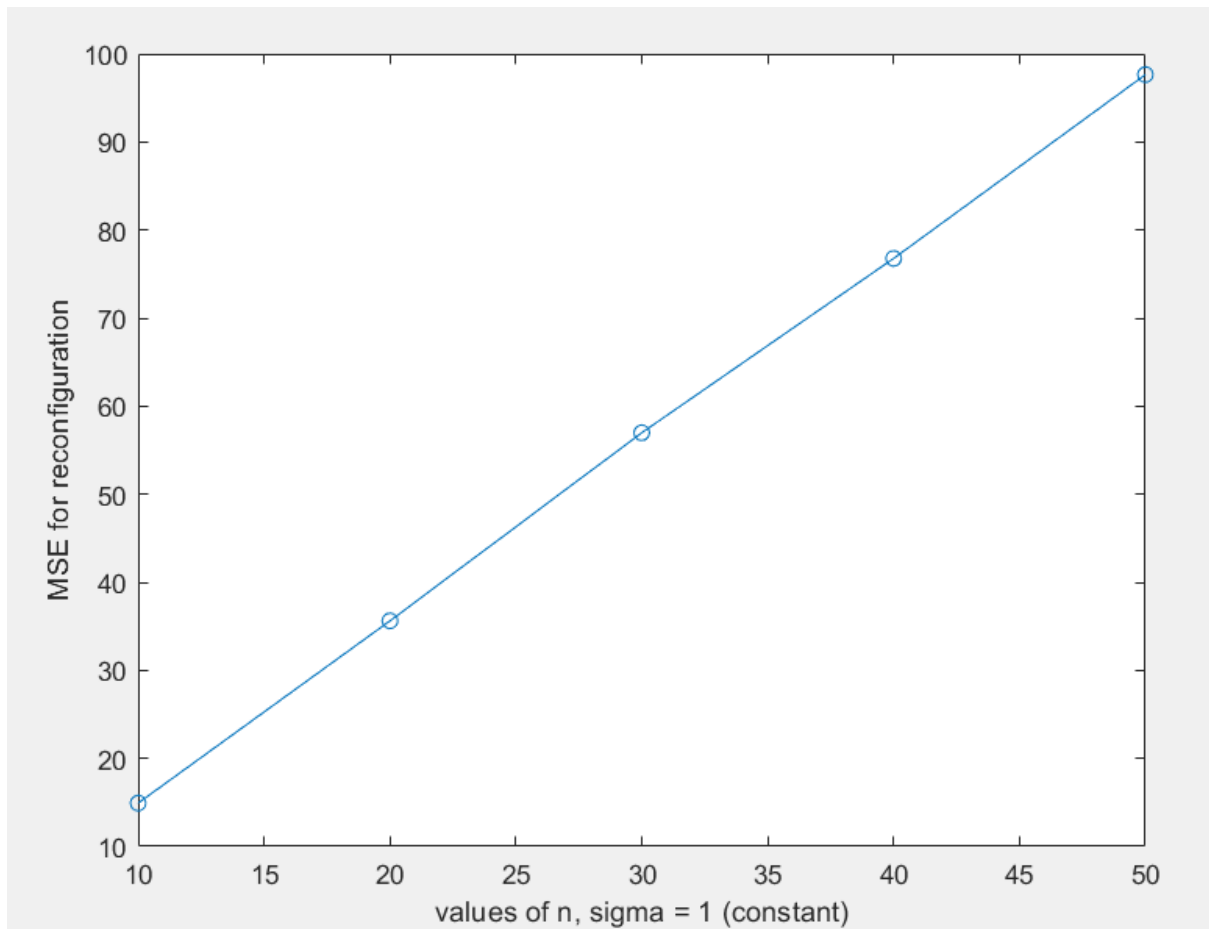
97.6454

Plots

MSE for w



MSE for Reconstruction



As can be seen the MSE for the overfit case is much less than that for the underfit case. Hence, it is obvious that if MSE of reconstruction is our main criteria then overfit model should be selected since it gives us an MSE which is extremely less in comparison to the underfit model.

PART 1 – Varying sigma with n constant

Code

```

1 % Part c_iii_Model2_Part2 (varying sigma, n constant) (Model is the one
2 % which is overfit to the true model)
3
4 % Everything same, but now only sigma varies
5
6
7 - count = 0;
8 - kernel = 3; % for t generation, the true one
9 - mse_vector = zeros(4,1);
10 - recon_mse_vector = zeros(4,1);
11 - kernel_mis = 5;
12 - w_average_matrix = zeros(kernel_mis,4);
13
14
15 - n = 20;
16 - A = zeros(2*n+1,kernel);
17 - x_interim = -n:n;
18 - x = transpose(x_interim*0.1); % turning to vector
19
20 % A generation, true A, A that gives true t
21
22 - for i = 1:kernel
23 -     A(:,i) = x.^i;
24 - end
25
26 % t generation
27 - t = A(:,1) + 3*A(:,2) + 2*A(:,3); % true t
28
29 % A, for our model, generation
30
31 - A_mis = zeros(2*n+1,kernel_mis);
32

```

```

33 - for i = 1:kernel_mis
34 -     A_mis(:,i) = x.^(i-1);           % our kernel model
35 - end
36
37 - pm_i = pinv(A_mis);
38 - w_true = pm_i*t;
39
40 - for sigma = [0.1,1,3,5]
41
42 -     count = count + 1;
43
44 -     w_average_interim = zeros(kernel_mis,1);
45
46 -     mse_interim = 0;
47
48 -     recon_mse_interim = 0;
49
50 -     for j = 1:100
51
52 -         % v generation
53 -         v = sigma*randn(2*n+1,1);
54
55 -         % y generation
56 -         y = t + v;           % observed y
57
58 -         w_estimated = pm_i*y;   % calculating the estimate of w
59
60 -         y_estimated = A_mis*w_estimated;
61
62 -         w_average_interim = w_average_interim + w_estimated; % running sum
63
64 -         mse_interim = mse_interim + (norm(w_true - w_estimated))^2; % running

```

```

65 -
66 -         recon_mse_interim = recon_mse_interim + (norm(y - y_estimated))^2;
67 -
68 -     end
69 -
70 -     mse = mse_interim/100;
71 -     recon_mse = recon_mse_interim/100;
72 -     w_average = w_average_interim/100;
73 -
74 -     mse_vector(count,1) = mse;
75 -     recon_mse_vector(count,1) = recon_mse;
76 -     w_average_matrix(:,count) = w_average;
77 -
78 -
79 -
80 - end
81 -
82 - sigma_range = [0.1;1;3;5];
83 - disp("The true w vector for our model, which is obviously constant for all
84 - disp(w_true)
85 - disp("The matrix of estimated w for our model(averaged over 100 iterations,
86 - disp(w_average_matrix)
87 - disp("The averaged MSE of w over 100 iterations for each sigma, where rows
88 - disp(mse_vector)
89 - disp("The averaged MSE of reconfiguration over 100 iterations for each sig
90 - disp(recon_mse_vector)
91 -
92 - figure(1);
93 - plot(sigma_range,mse_vector,'o-')
94 - xlabel('values of sigma, n = 20 (constant)')
95 - ylabel('MSE for w')
96 -
97 - figure(2);
98 - plot(sigma_range,recon_mse_vector,'o-')
99 - xlabel('values of sigma, n = 20 (constant)')
100 - ylabel('MSE for reconfiguration')
101 -

```

Results

>> c_iii_Model2_Part2

The true w vector for our model, which is obviously constant for all sigma, since n is constant is

0.0000

1.0000

3.0000

2.0000

0.0000

The matrix of estimated w for our model(averaged over 100 iterations, for each sigma), where each column corresponds to a value of sigma and the column vector the vector of averaged w estimate, is

-0.0032	-0.0155	0.1085	-0.0525
1.0011	1.0133	0.8659	1.0348
3.0037	3.0665	2.9011	3.0538
2.0001	2.0015	2.0399	1.9652
-0.0007	-0.0222	0.0186	-0.0108

The averaged MSE of w over 100 iterations for each sigma, where rows represent MSE of w value for each sigma, is

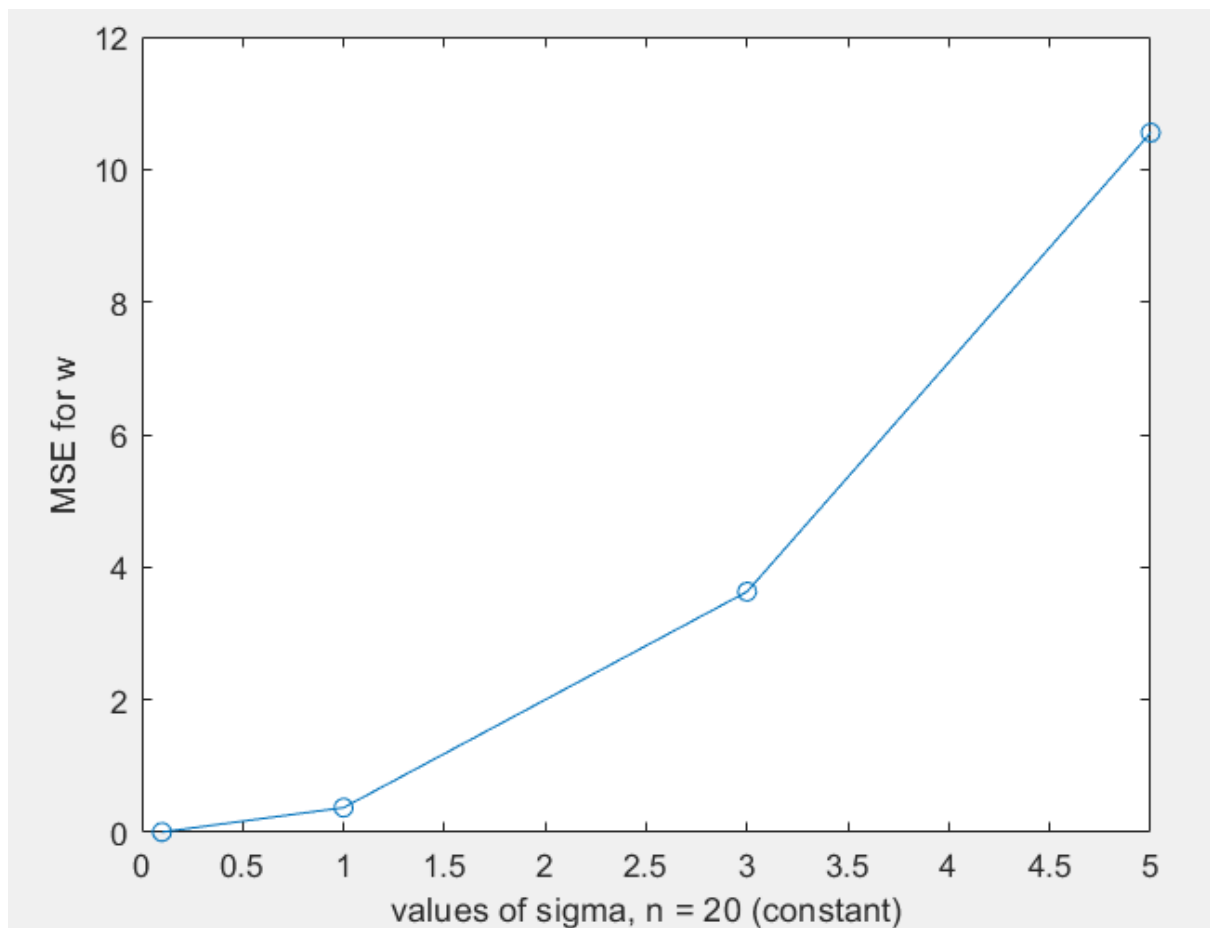
0.0043
0.3726
3.6304
10.5548

The averaged MSE of reconfiguration over 100 iterations for each sigma, where rows represent MSE of reconfiguration value for each sigma, is

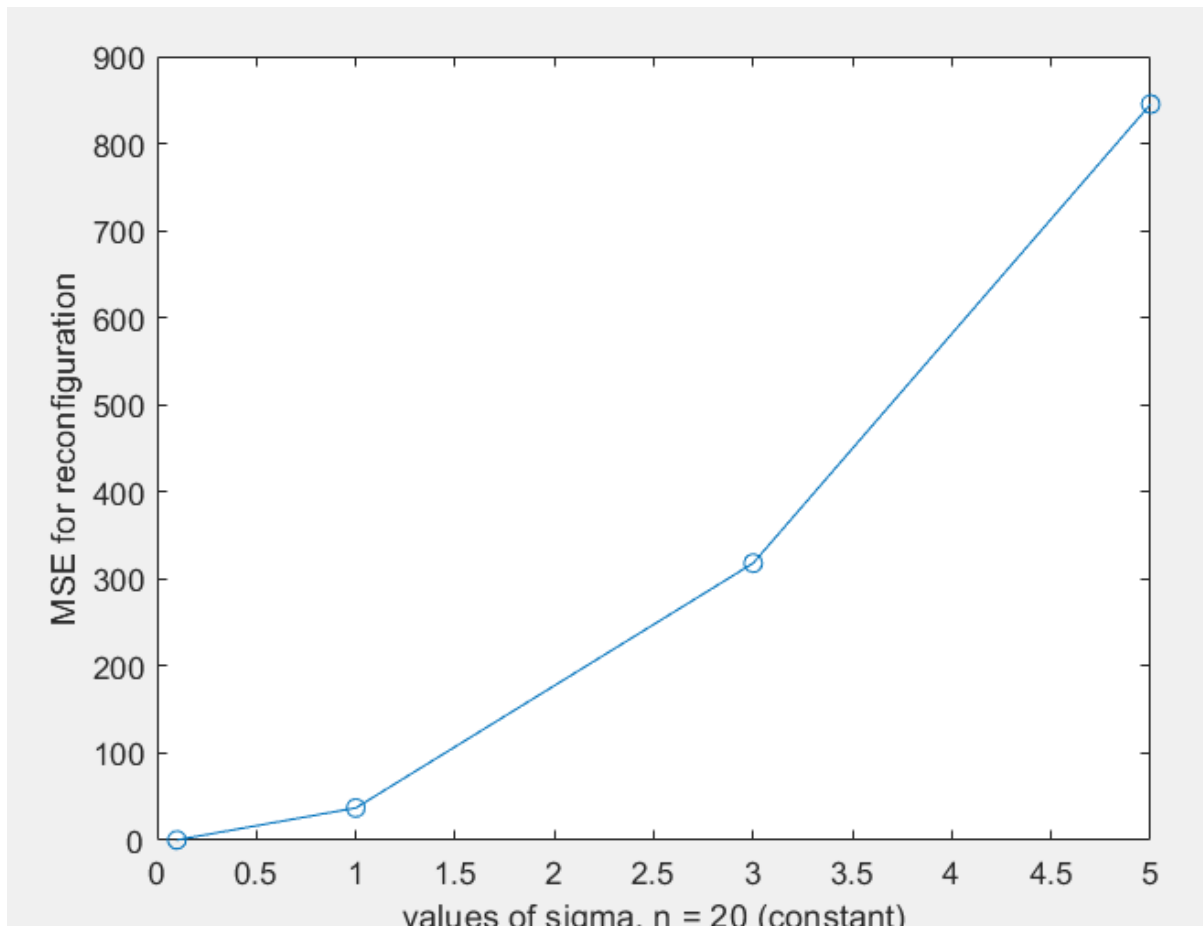
0.3584
37.0618
318.1502
845.2264

Plots

MSE for w



MSE for Reconstruction



Similar results with similar explanation

iv. Comment on Results

It is definitely worse, MSE wise, to underestimate/underfit the model. Thus, the underestimated/underfit model had an excruciatingly worse MSE compared to the overfit/overestimated version which had a comparatively much better MSE.

However, with overfitting we do have an issue where the model is so overfit, that we can't have good model generalization to the test data, increasing the test MSE. This fact is also known as the bias-variance trade-off in machine learning algorithms.

Underfit model, have, higher training MSE but generalize better and may have low test MSE, sometimes even lower than the Overfit model.

-----END-----