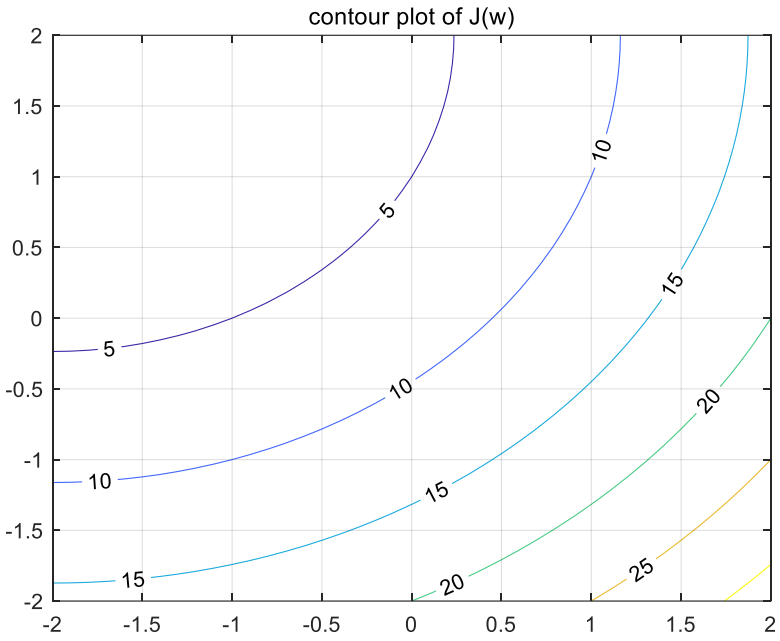
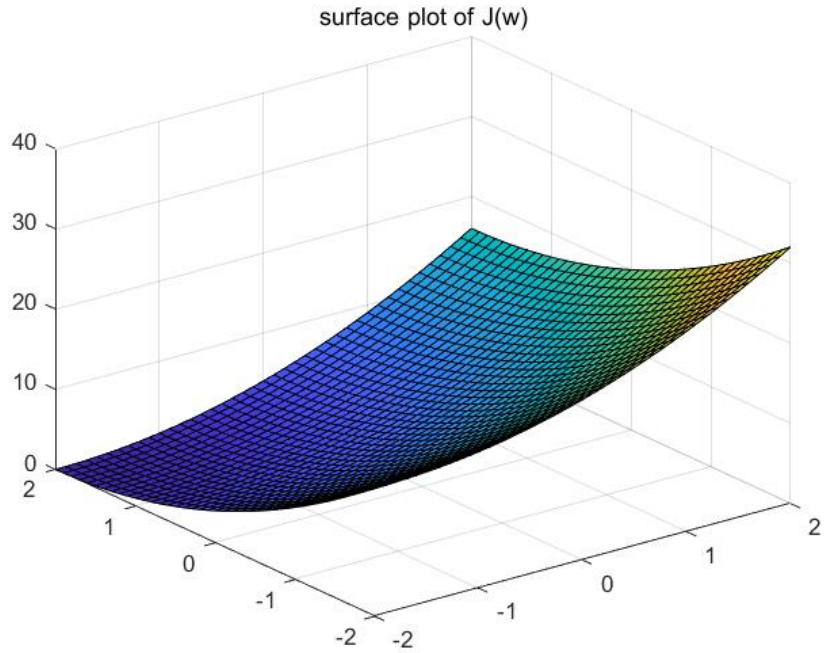
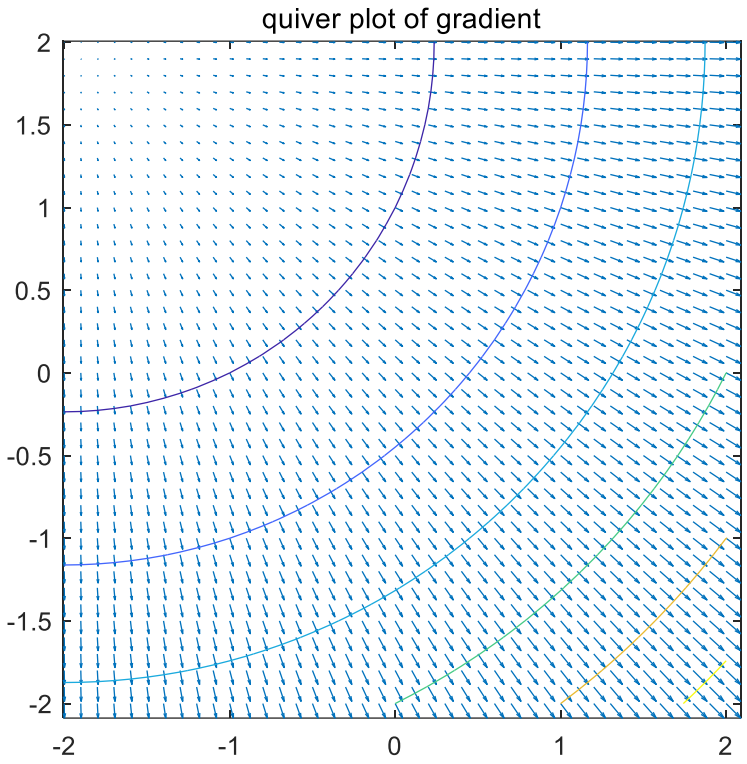
ECE 251B HW3

1.a)

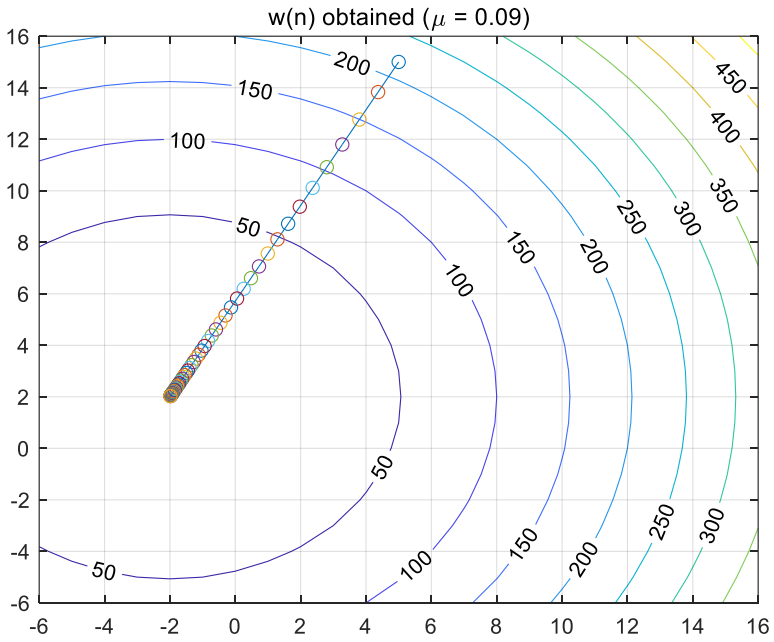




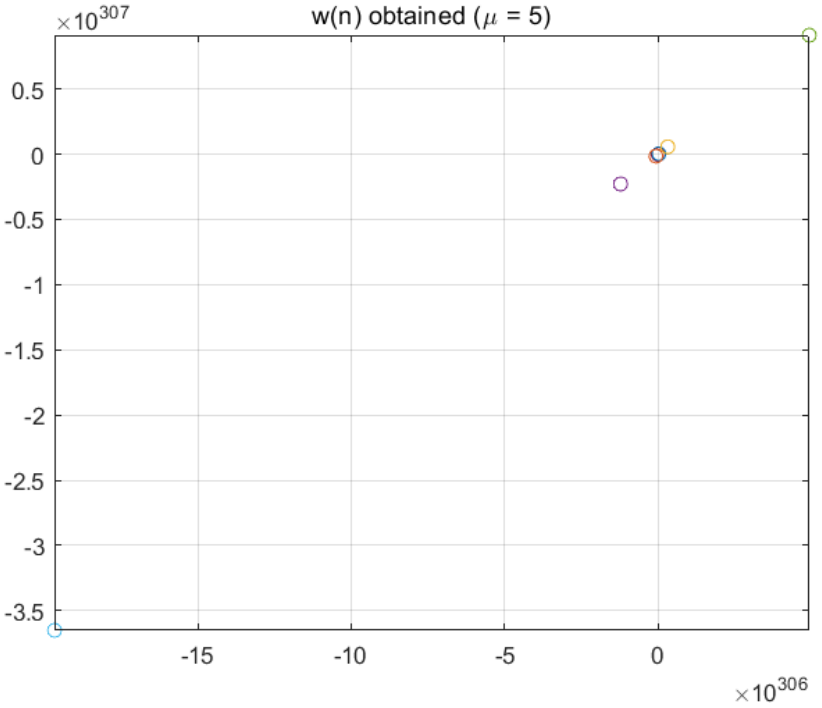
b) It shows that the gradient is perpendicular to iso-contour.

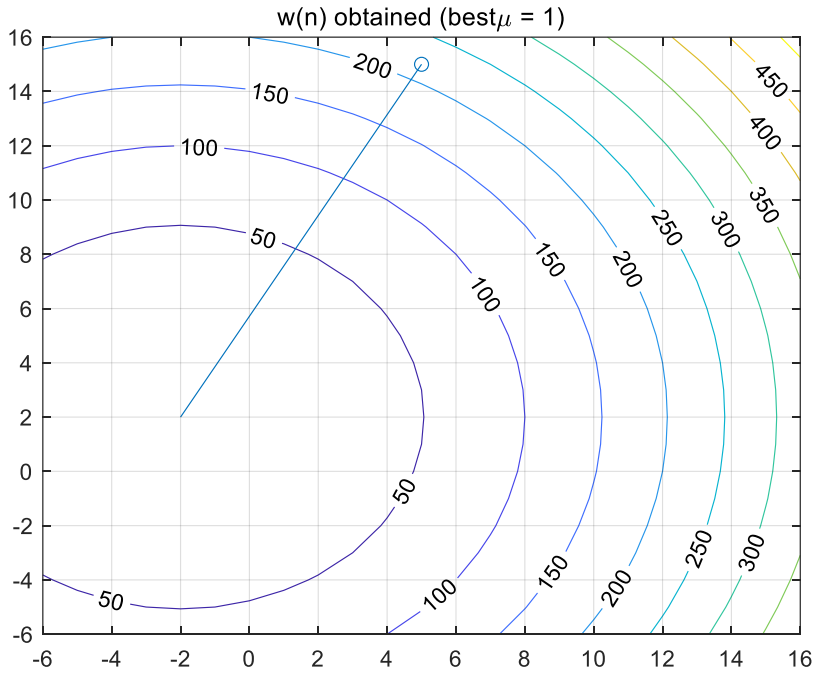


c) The expression for the parameter updates under the gradient descent algorithm is $w^{(n+1)} = w^n - \mu \nabla w J\left(w^{(n)}\right) = w^{(n)} - \mu A(w - w_0)$ .When the step size μ is 0.09, the iteration number is 66.



d)



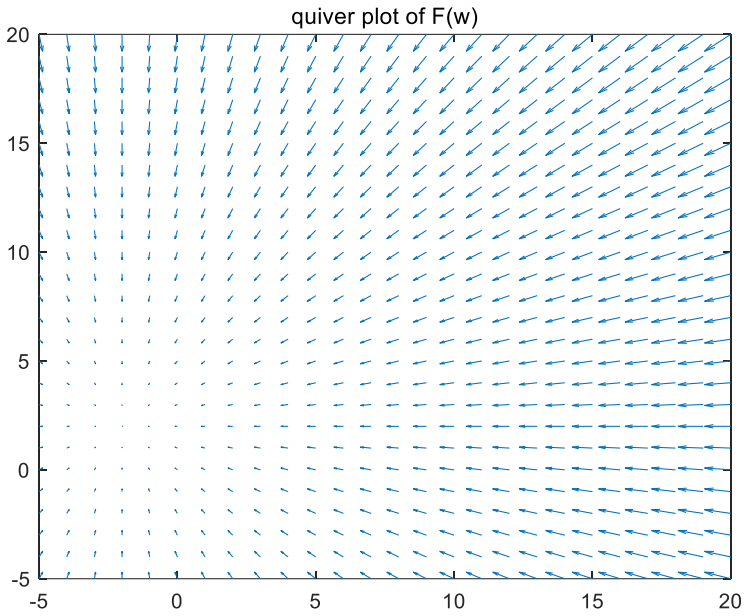


Comment: When μ is 5, the values cannot converge even after 511 times iteration, as is shown in the graph that the values of w(n) is exploded. Moreover, the contour plot is not shown since the range is too huge. When μ is 1, which is found by line search it can easily and quickly converge since the iteration times is only 1. My explanation for this is that the step size is very important for gradient descent method, if the step size is too big, it can hardly reach to the minimum point for convergence.

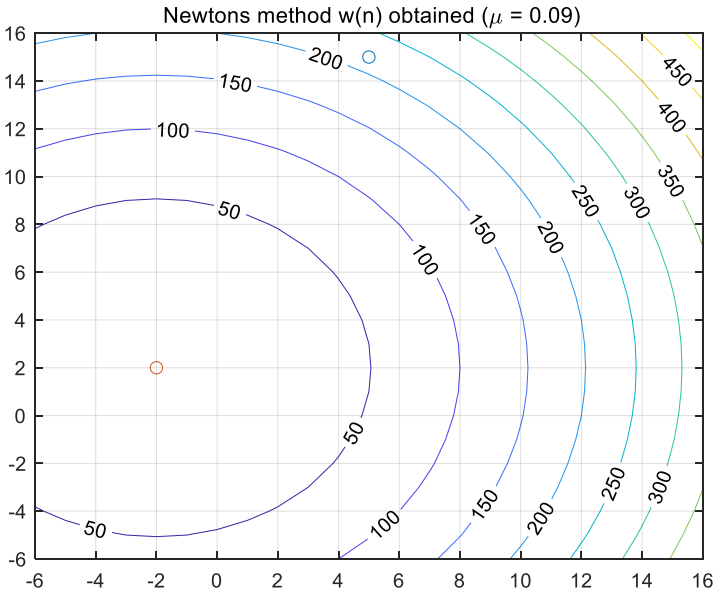
e) The expression for parameters updates under Newton's method:

$$w^{(n+1)} = w^n - \mu \nabla w J\left(w^{(n)}\right) = w^{(n)} - \frac{1}{2} * 2A^{-1}A(w - w_0) = w_0$$

Compared with the plot in (b), the direction is opposite, for F(w), it points to the optimal point (-2 ,2)



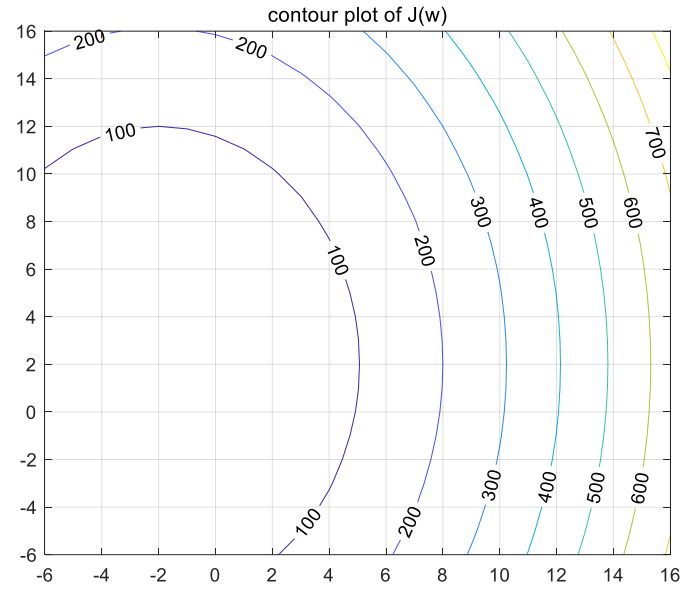
f) It only uses one time iteration to converge.

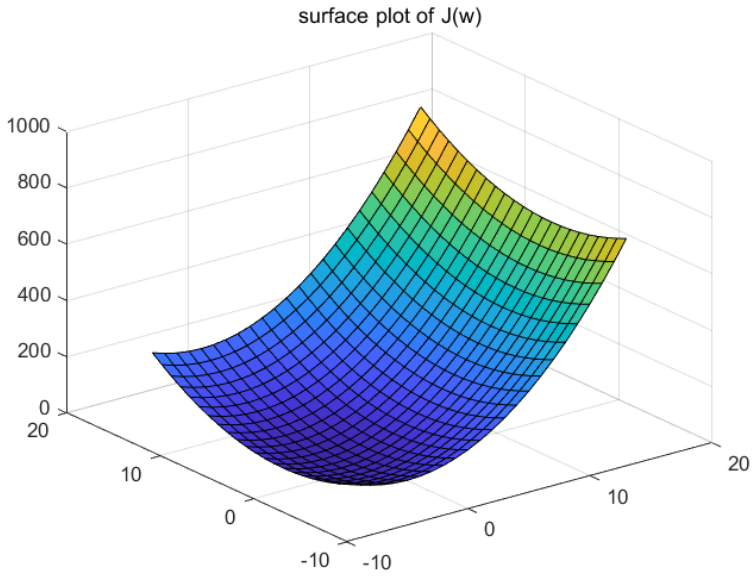


g) Comment: In question c, it uses 66 times to convergence, however, by using line search in question d, when the step size μ is 1, it only takes 1 iteration to converge. In question e, it only uses 1 time to get to the optimal.

So it shows that generally the newton's method has faster convergence rate compared with the gradient descent method. The step size is crucial for the performance of gradient descent method. The reason is that gradient descent uses a plane to fit the current local surface. The direction of the gradient f'(x) is the direction in which the function becomes larger. Newton's descent method is to use quadratic surface to fit the current local surface(twice differential).

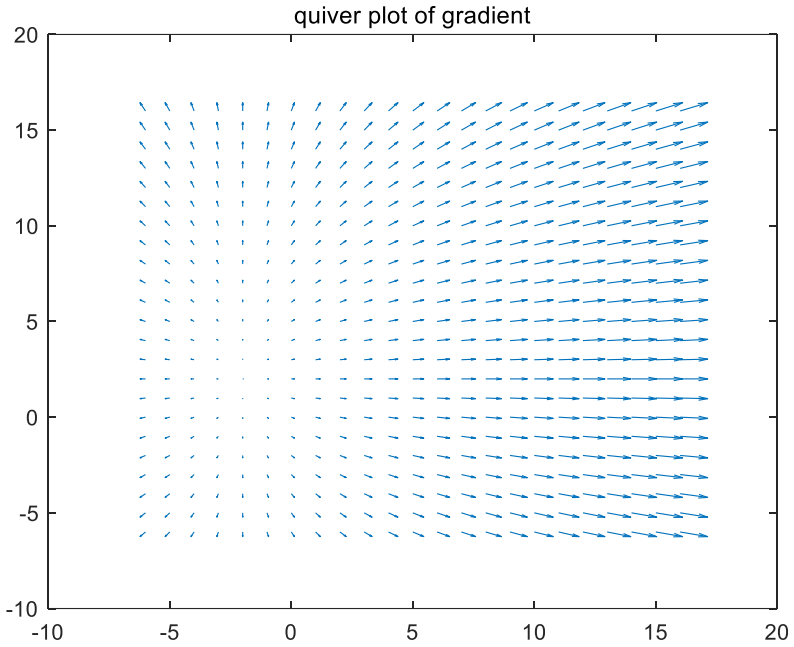
2.a)





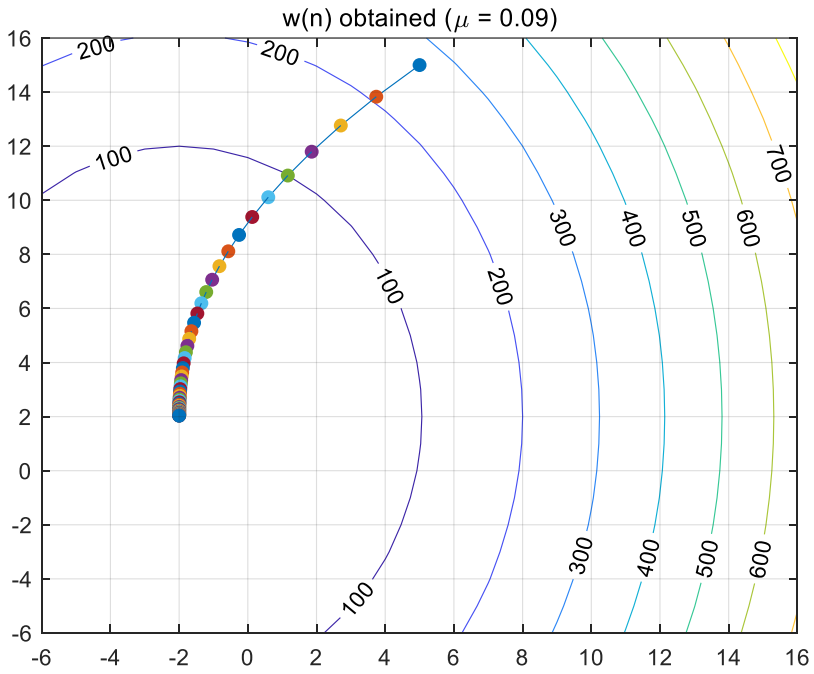
Compared with question 1, the contour plot is more ellipse because of the condition number. And the surface plot is more radian.

b)



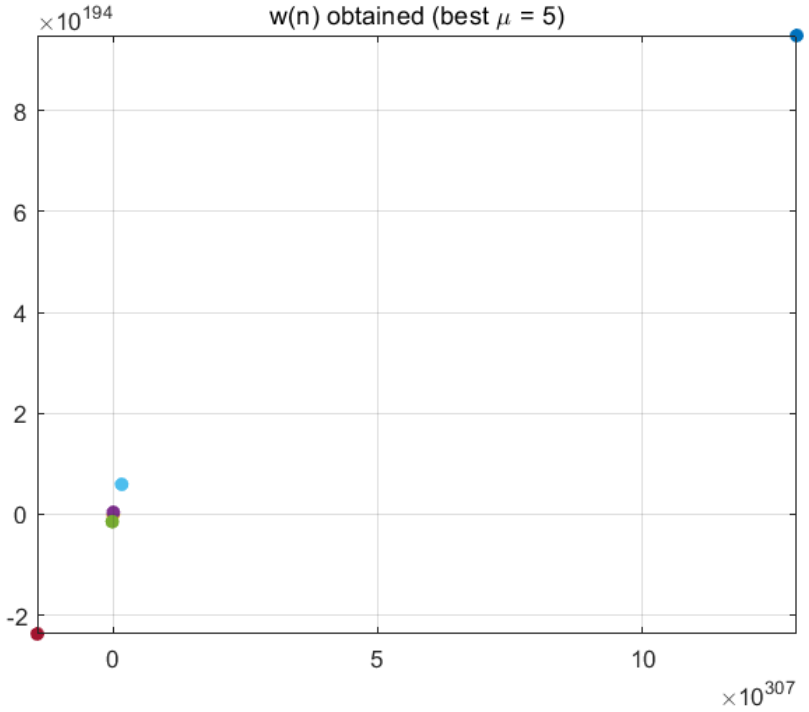
The quiver plot is perpendicular to the contour plot, so in this case, it is not as evenly distributed as that in question 1.

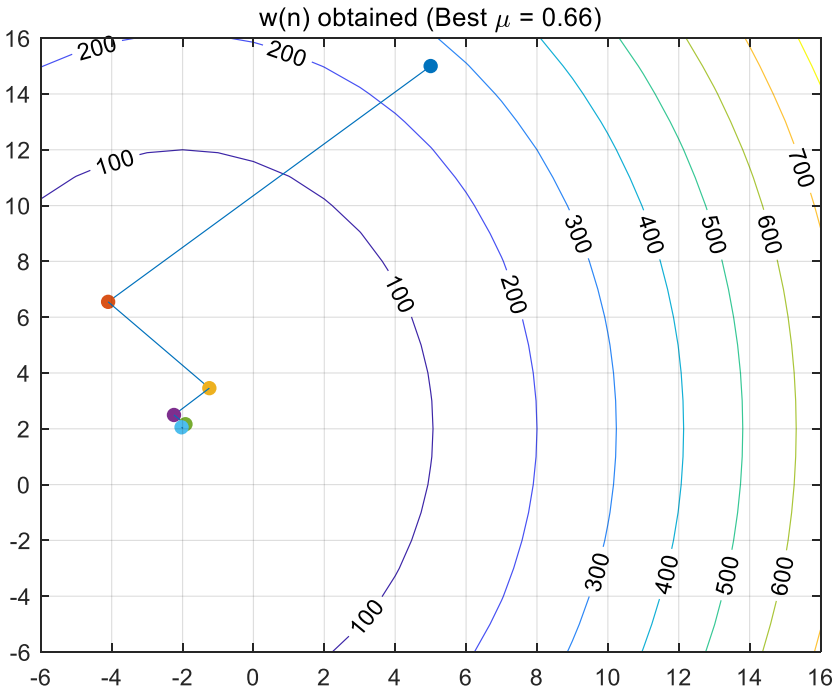
c)



The expression for the parameter updates is the same with that in question1. The difference is that it costs a little shorter time to converge (the iteration times is 64). And the track is curve and steep rather than straight.

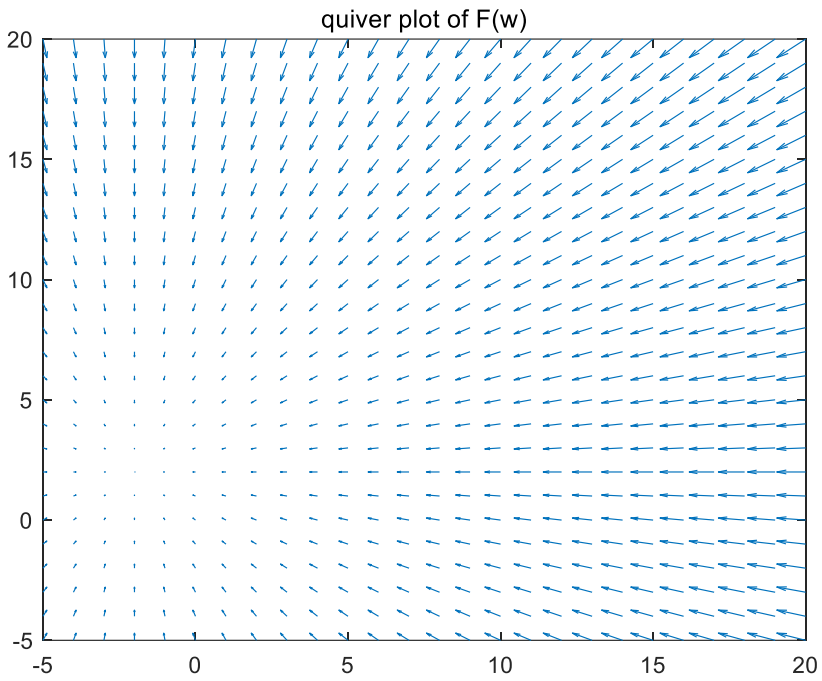
d)



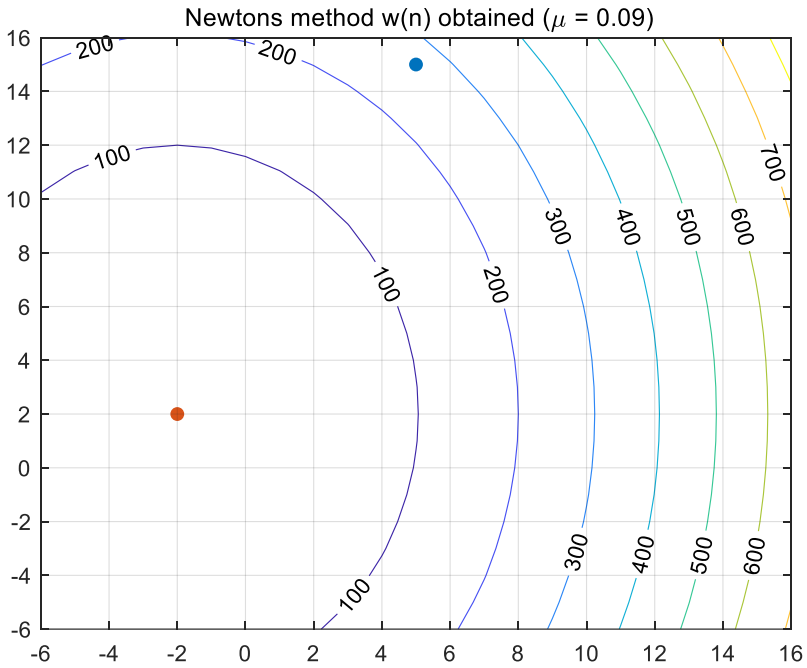


If μ equals to 5, the values cannot converge even after 324 times iterations. By doing the line search, the step size should be 0.66, under this condition, it needs to iterate 6 times to converge which is quite fast, but slower than that in question 1.

e)

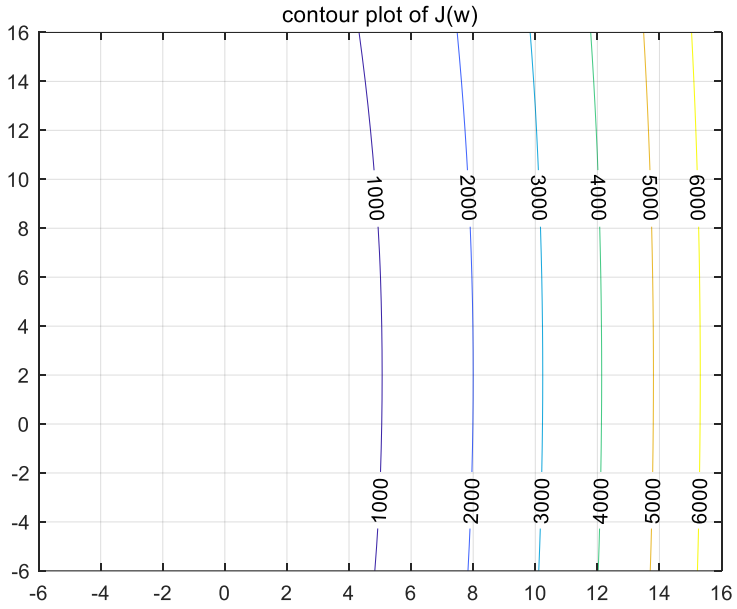


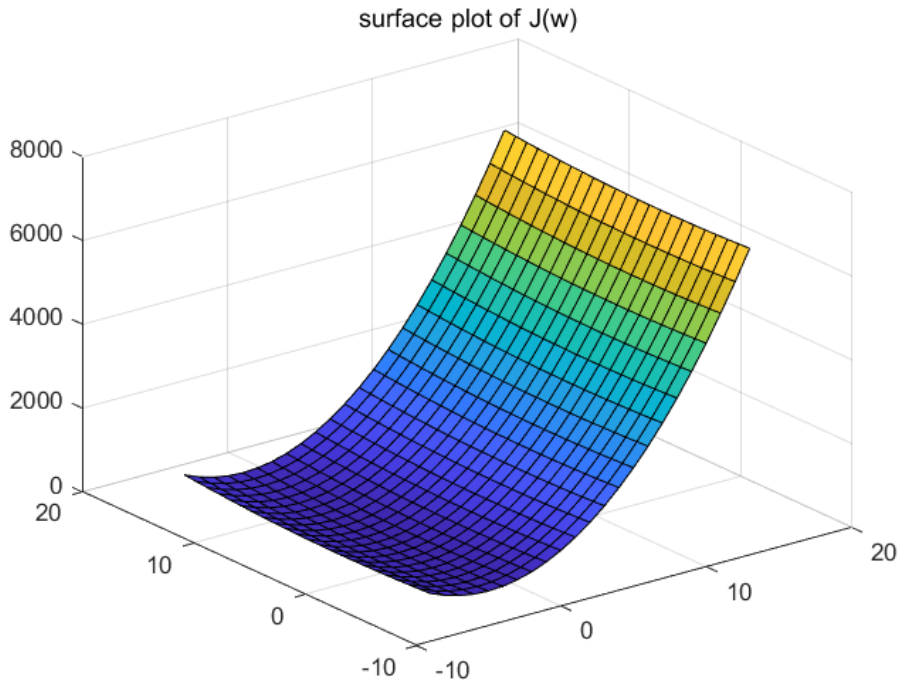
f)



g) The graph and expression of F(w) is the same with question 1, and the Newton's method still only use one time iteration.

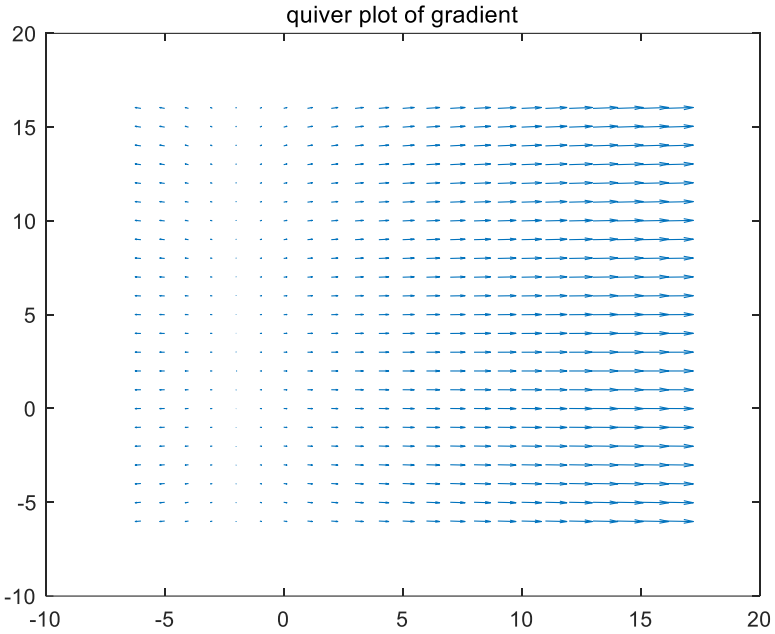
3.a)





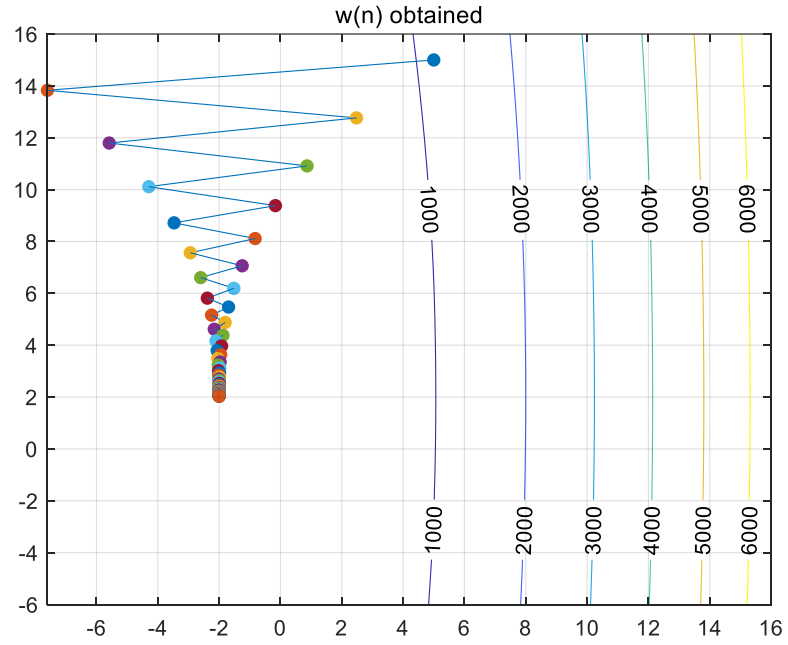
With $A = diag(20, 1)$, the contour function is hard to show compactly since the total range is very big. But the whole plot is very similar with question 2, except for the coordinate range.

b)



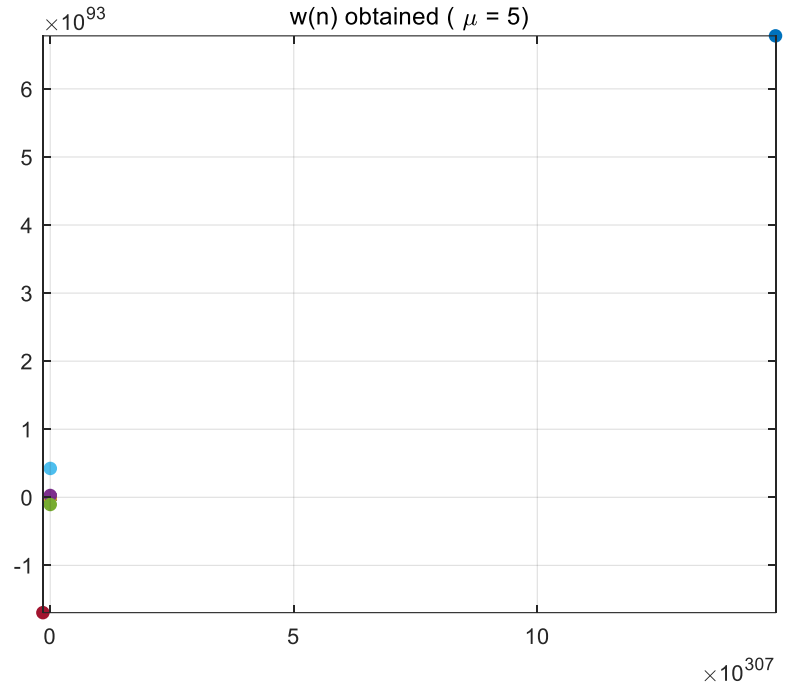
Since the quiver plot has a relatively small range, and the value of J(w) is quite big, the quiver plot has a steady and parallel gradient change.

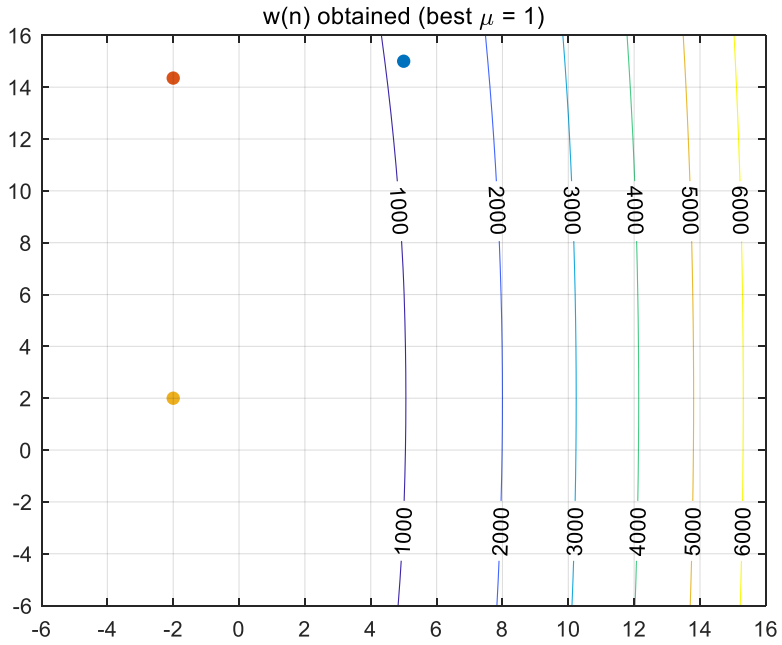
c)



As is shown, there are zig-zags and the solution oscillates. Moreover, it's slower covergence rate(longer and repetitive lines) compared with the same iteration times (64),of with other graphs.

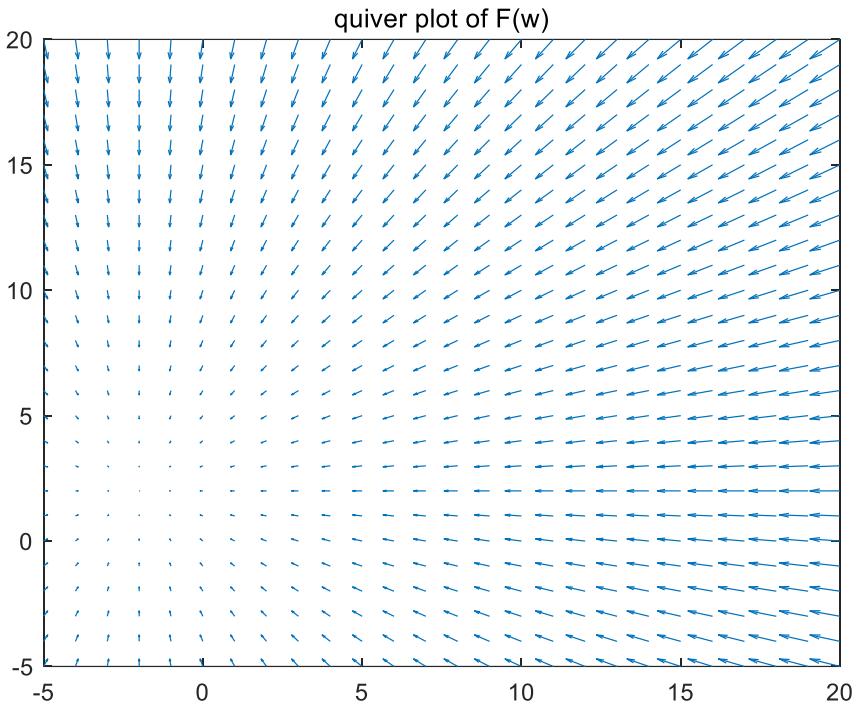
d)



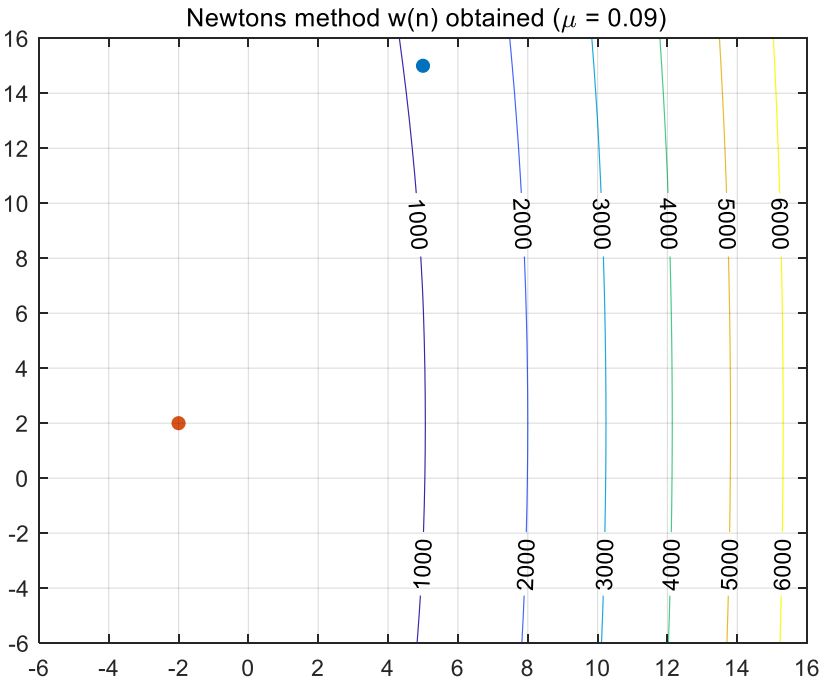


When μ is 5, it still cannot converge. By line searching, the best step size is 0.05 for first step and 1 for second step, under this case, the iteration times is only 2.

e)



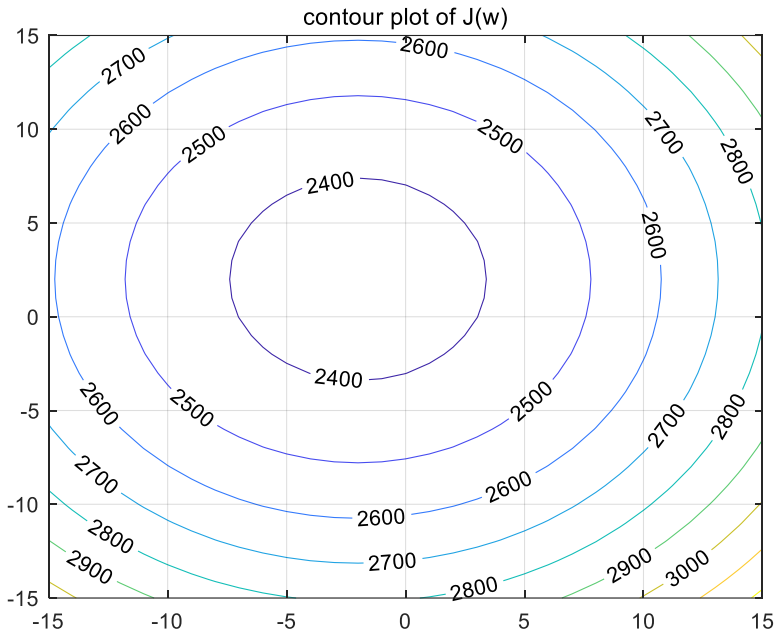
f)

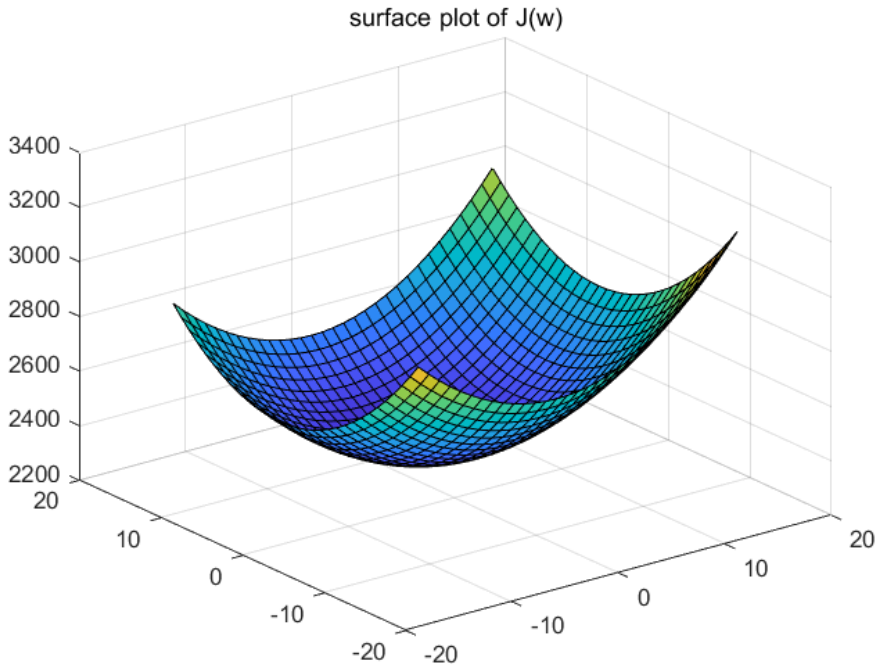


g) Same. The direction of quiver function of F(w) is opposite to the direction of d). quiver plot of gradient. In addition, the newton's method still only use up one time step to reach the optimal value.

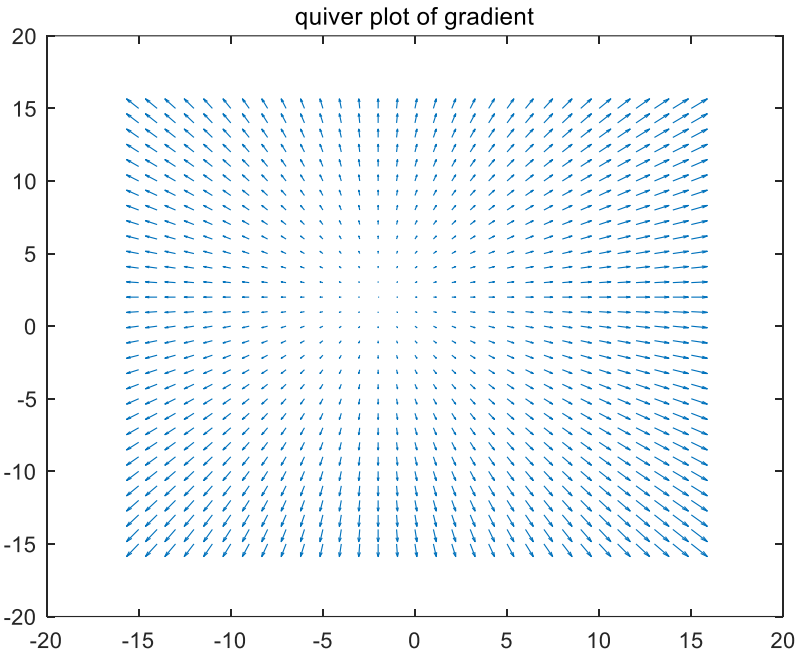
4. a) Since the eigenvalues of A is 2 and 1. By calculation, matrix A is $\begin{bmatrix} \frac{3}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{3}{2} \end{bmatrix}$. Under this condition, since the shape is ellipse

$$J = 1.5 * ((X + 2).\hat{}\,2 + (Y - 2).\hat{}\,2) + (X + 2) * (Y - 2);$$

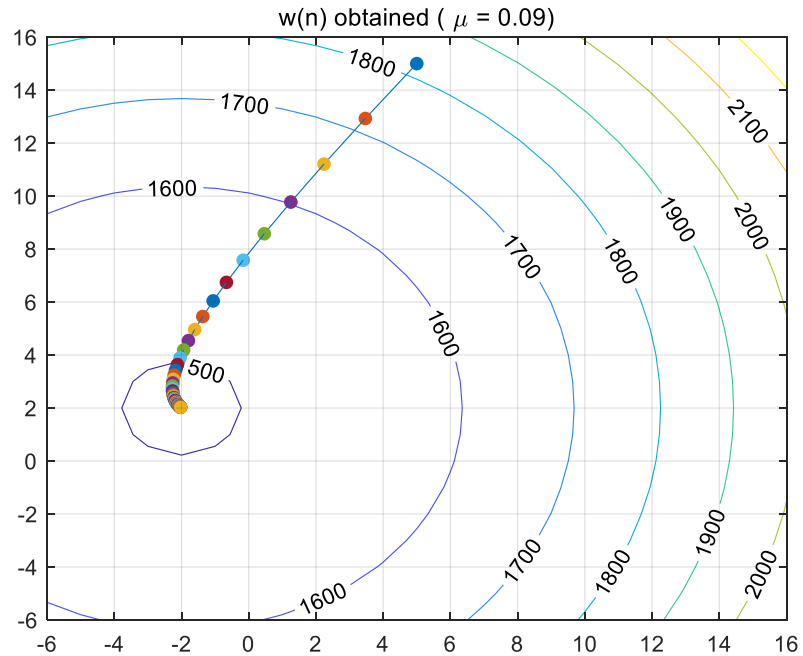




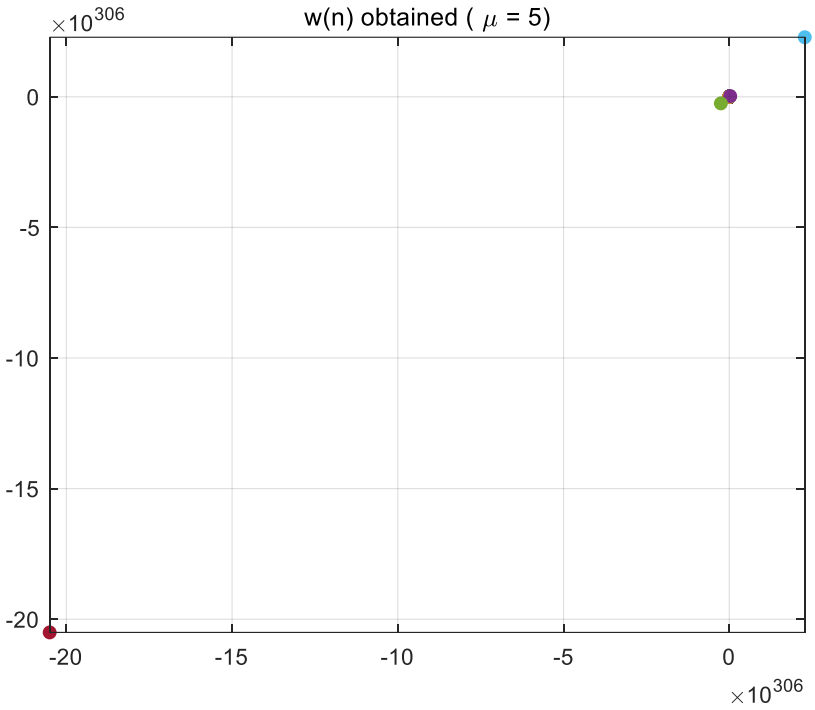
b)



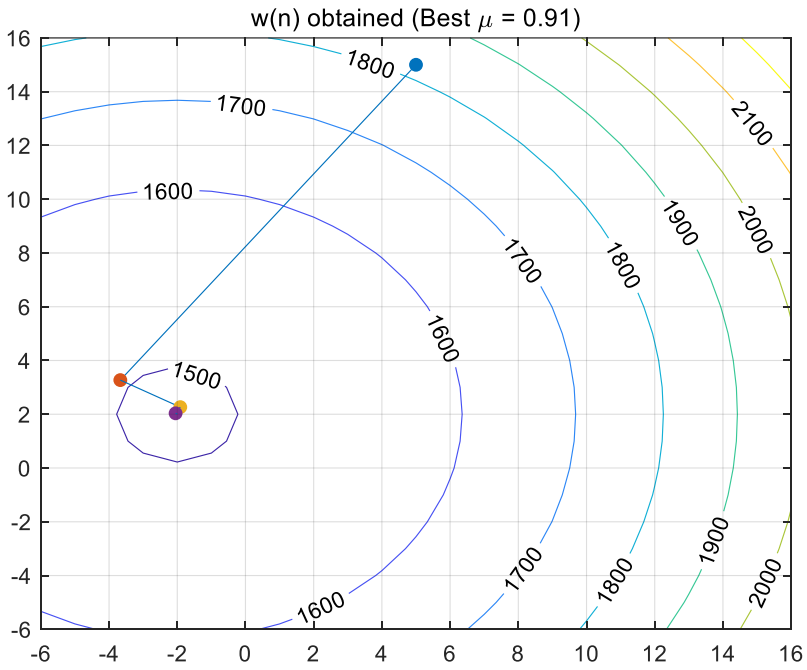
c) The parameter update is the same



d)

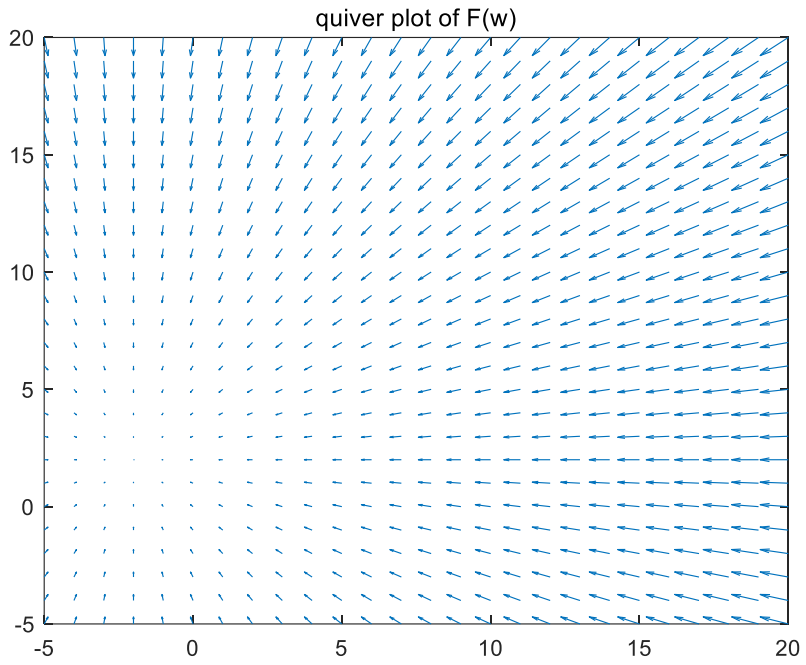


It is not convergent when μ=5, as is shown, the values of objective function is explosively huge



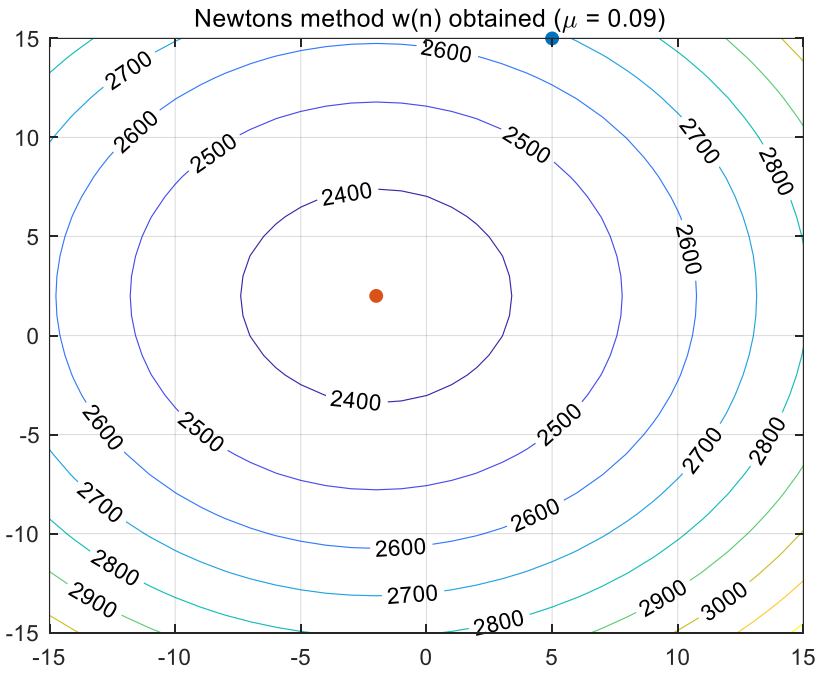
For line search, the iteration needed for convergence is 4 times.

e)



Still, the direction is opposite compared with the gradient quiver plot.

f)

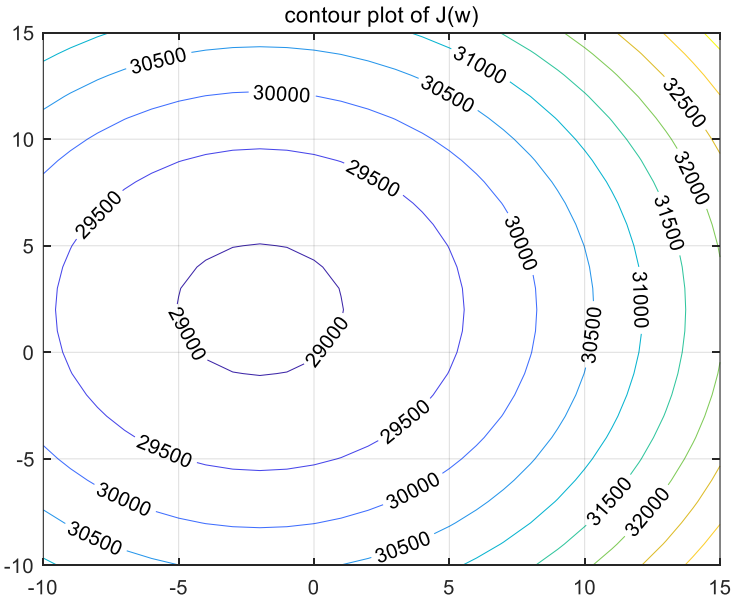


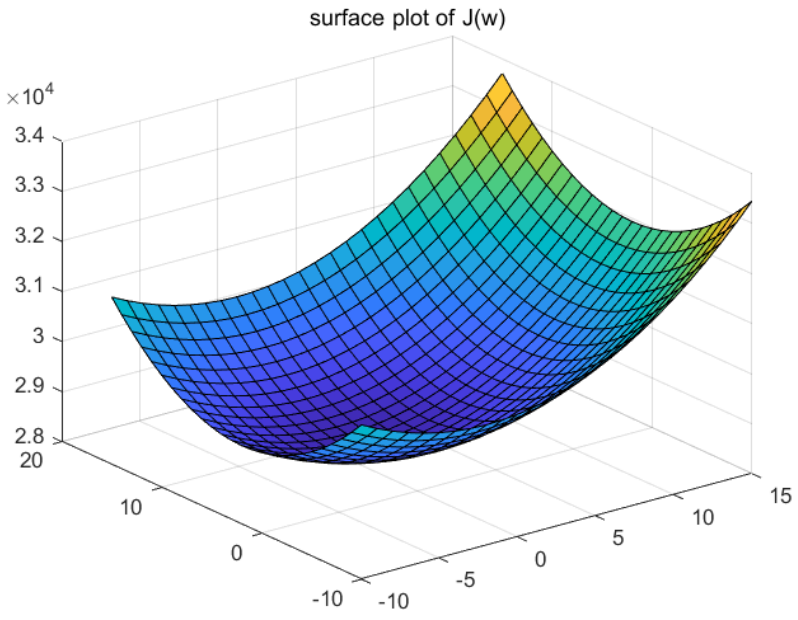
g) Still, the newton's method only uses up one time step to get to the optimal value.

As for the difference for question 4 compared with the first three questions, since the cost function (J(w)) is ellipse with principal direction which have angles $\pi/4$ and - $\pi/4$, it actually rotate the contour function by multiplying with different eigenvectors. The eigenvectors in question 4 is $\begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$ and $\begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{-\sqrt{2}} \end{bmatrix}$ and A is calculated as A = [1.5, 0.5;0.5, 1.5]. It makes the contour function rotate for $\pi/4$ angels. Other properties are quite similar.

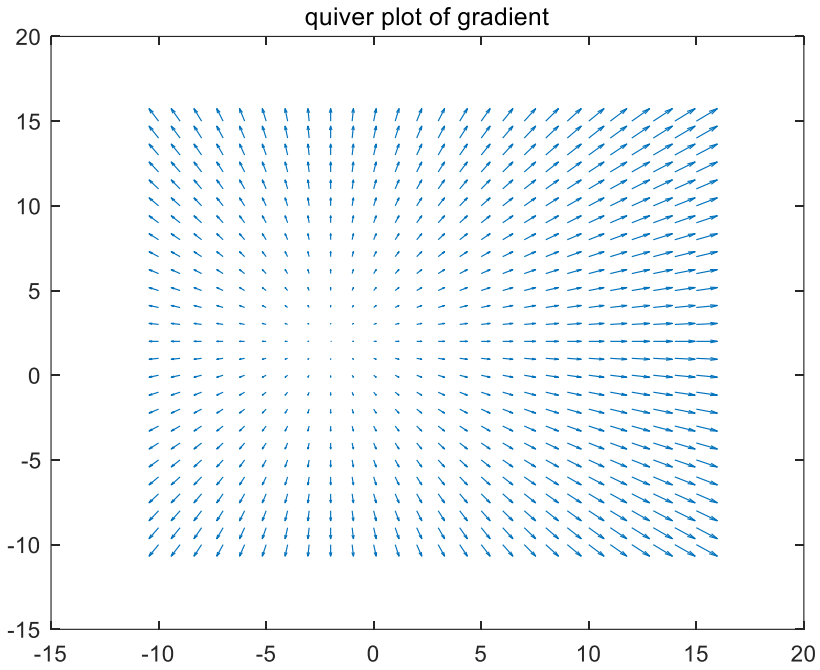
5. a) Since the eigenvalues of A is 20 and 1. By calculation, matrix A is $\begin{bmatrix} \frac{21}{2} & \frac{19}{2} \\ \frac{19}{2} & \frac{21}{2} \end{bmatrix}$

$$J = 10.5 * ((X + 2) .\hat{} 2 + (Y - 2) .\hat{} 2) + 19 * (X + 2) * (Y - 2)$$

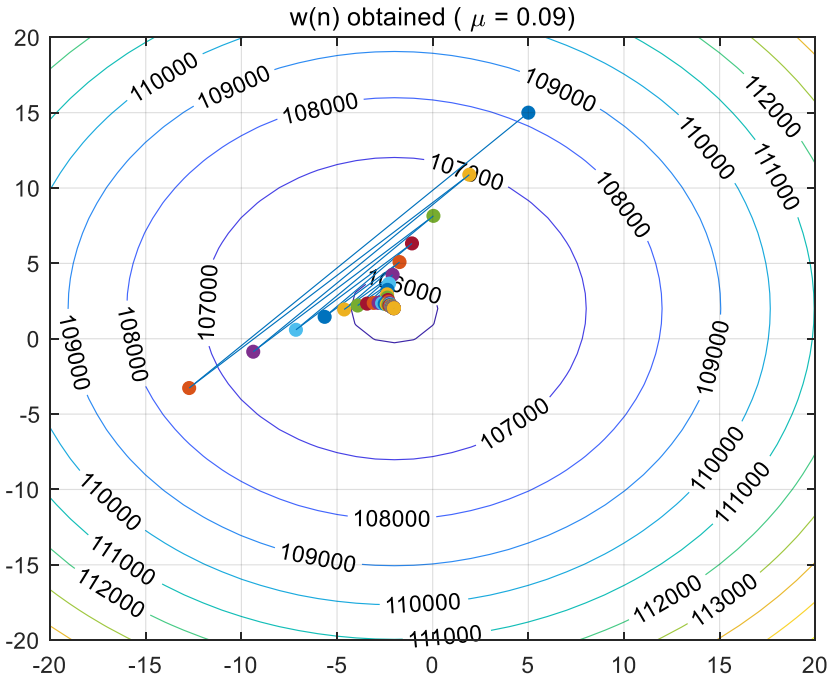




b)

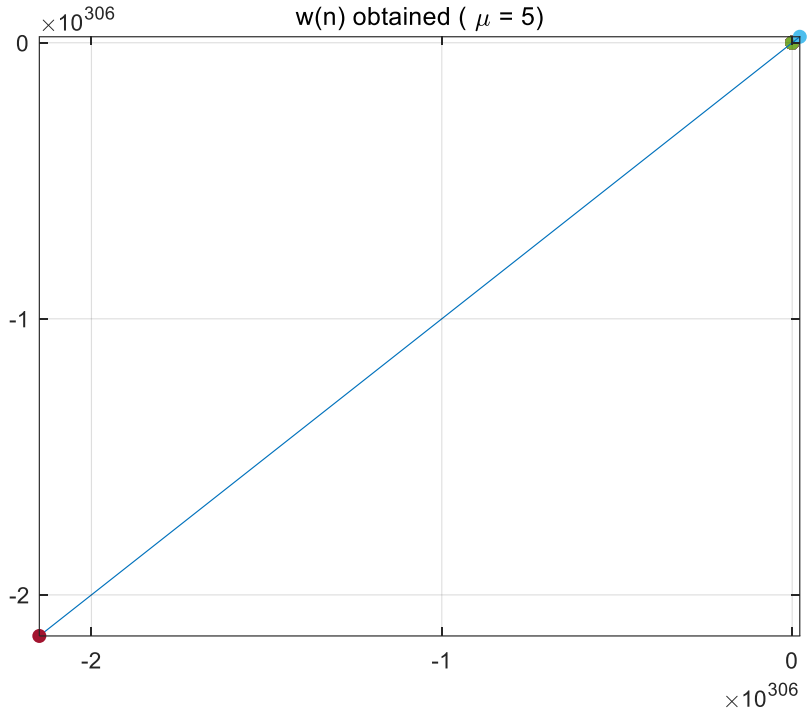


c)

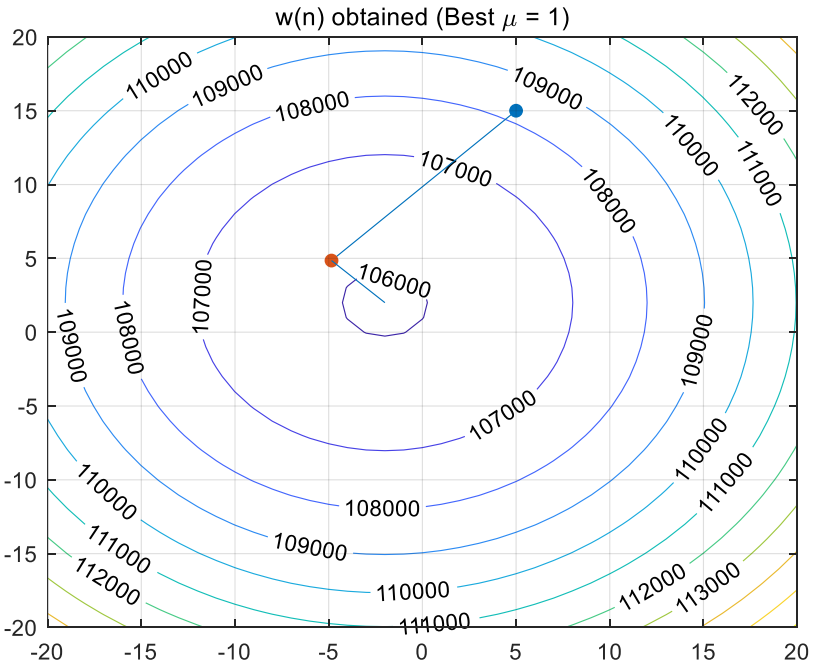


The expression for parameter update is the same, and it is also a zig-zag convergence. The iteration time is 52.

d)

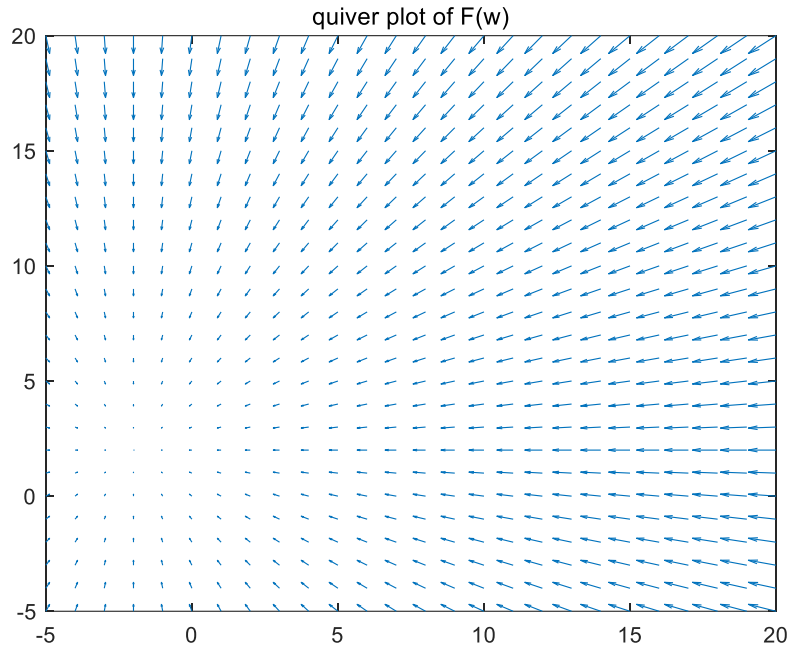


It is not convergent when μ=5, as is shown, the values of objective function is explosively huge



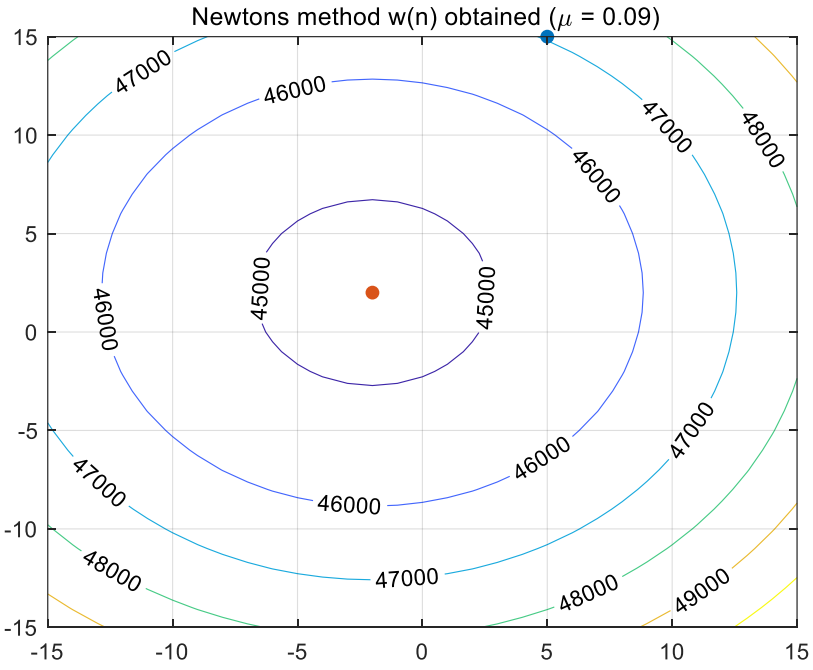
For line searching, the iteration is 2 times to get to the optimal value.

e)



Still, the direction is opposite compared with the gradient quiver plot.

f)



g) Still, the newton's method only uses up one time step to get to the optimal value.

As for the difference of question 5 to the above questions, firstly, it has the principal direction of $\pi/4$ and - $\pi/4$, so the eigenvalues are $\begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$ and $\begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{-\sqrt{2}} \end{bmatrix}$. Since it has different eigen values, so A is calculated as A = [10.5, 9.5;9.5, 10.5]; And other properties are quite similar and discussed before.

APPENDIX

Question(a,b)

```
clear
clc
x = -10:15; y =  -10:15;
[X, Y] = meshgrid(x, y);
J = 10.5*((X + 2) .^ 2+(Y - 2) .^ 2) + 19*(X+2)*(Y-2);
% x = -10 : 10;y = -10 : 10;
% [X, Y] = meshgrid(x, y);
% J = (X + 2) .^ 2 + (Y - 2) .^ 2;
spacing = 0.2;
[DX,DY] = gradient(J,spacing);

figure()
contour(X, Y, J, 'ShowText', 'on');grid;
title('contour plot of J(w)');
figure();
surf(X, Y, J);
title('surface plot of J(w)');
figure();
quiver(X,Y,DX,DY);
title('quiver plot of gradient');
hold on

A = [10.5, 9.5;9.5, 10.5];
% A = eye(2);
% axis equal
% hold off
w=[5, 15]';
w_f = [-2, 2]';
w_a(:,1)=w;
EPS = 0.001;
stepsize = 0.09;
it = 1;
% while (sum((w - w_f) .^ 2) >= EPS)
%
```

```
%     w = w - stepsize * A * (w - w_f);
%     it = it + 1;
%     w_a(:,it) = w;
%
% end


%
% for i = 1 : size(w_a, 2)
% scatter(w_a(1, i), w_a(2, i));
% end
% title('w(n) obtained (\mu = 5)');
```

Question(c,d)

```
clc
clear
w=[5, 15]';
w_f = [-2, 2]';
% A = [1.5, 0.5;0.5, 1.5];
A = [10.5, 9.5;9.5, 10.5];
w_a(:,1)=w;
EPS = 0.001;
it = 1;
miu = linspace(0.01,5.0,500);
Upper = 1000;
bestlr = 0;
% stepsize = 5;
while (sum((w - w_f) .^ 2) >= EPS)
    for i=1:1:500
        inter = w - miu(i) * (A * (w - w_f));
        Jls = (inter - w_f)'*A*(inter - w_f);
        if Jls<Upper
            Upper = Jls;
            bestlr = miu(i);
        end

    end
    stepsize = bestlr;

    w = w - stepsize * A * (w - w_f);
    it = it + 1;
    w_a(:,it) = w;
end
```

```
x = -20:20; y = -20:20;
[X, Y] = meshgrid(x, y);
% J = 1.5*((X + 2) .^ 2+(Y - 2) .^ 2) + (X+2)*(Y-2);
% J = (X + 2) .^ 2 + (Y - 2) .^ 2;
J = 10.5*((X + 2) .^ 2+(Y - 2) .^ 2) + 19*(X+2)*(Y-2);
contour(X, Y, J, 'ShowText', 'on');grid;hold on;


for i = 1 : size(w_a, 2)-1
scatter(w_a(1, i), w_a(2, i),"filled");
line([w_a(1, i), w_a(1, i + 1)], [w_a(2, i), w_a(2, i + 1)]);
end
title('w(n) obtained (Best \mu = 1)');
```

Question e,f

```
clear
clc
% A = [1.5, 0.5;0.5, 1.5];
A = [10.5, 9.5;9.5, 10.5];
% A = diag(20,1);
w=[5, 15]';
w_f = [-2, 2]';
w_a(:,1)=w;
EPS = 0.001;
x1 = -5:20; y1 = -5:20;
[X1, Y1] = meshgrid(x1, y1);
U = -1 * (X1 + 2);
V = -1 * (Y1 - 2);
quiver(X1,Y1,U,V);
title('quiver plot of F(w)');
% J = (X + 2) .^ 2 + (Y - 2) .^ 2;
% spacing = 0.2;
% [DX,DY] = gradient(J,spacing);

figure()
% contour(X1, Y1, J, 'ShowText', 'on');grid;
% title('contour plot of J(w)');
% figure();
% surf(X, Y, J);
% title('surface plot of J(w)');


% stepsize = 0.09;
```

```
it = 1;
% while (sum((w - w_f) .^ 2) >= EPS)
%     Fw = ((w - w_f) .^ 2)./(A * (w - w_f));
%     w = w -  Fw;
%     it = it + 1;
%     w_a(:,it) = w;
% end
while (sum((w - w_f) .^ 2) >= EPS)
    Fw = -(w - w_f);
    w = w +  Fw;
    it = it + 1;
    w_a(:,it) = w;

end

x = -15:15; y =  -15:15;
[X, Y] = meshgrid(x, y);
% J = 1.5*((X + 2) .^ 2+(Y - 2) .^ 2) + (X+2)*(Y-2);

J = 10.5*((X + 2) .^ 2+(Y - 2) .^ 2) + 19*(X+2)*(Y-2);
contour(X, Y, J, 'ShowText', 'on');grid;hold on;


% quiver(Fw);
% title('quiver plot of gradient');
% hold on
%
for i = 1 : size(w_a, 2)
scatter(w_a(1, i), w_a(2, i),"filled");
end
title(' Newtons method w(n) obtained (\mu = 0.09)');
```