```
%===========Algorithm Parameters==============

               ALG_NUM = 2;

%===========Set Parameters====================
n = 2; %number of dimensions
Cv = .01;

num_nodes = 30; %Randomly generate nodes
grid_size = 25; %25x25
r = 17; %set communication range
r_s = 5*ones(num_nodes,1); %sensing range
scale = [.5 -6 6]; %for survaillance region [scale factor, range_min, range_max
    ]
meshdim = scale(2):scale(1):scale(3);

%generate scalar field with gaussian mixture
dim = [grid_size grid_size]; %dimensions
theta = [20 50 35 40]; %weights of the distributions
corr = .1333*ones(4,1); %correlations
var = [2.25 2.25; 1.25 1.25; 1.25 1.25; 1.25 1.25]; %variances
mu = [2 2; 1 .5; 4.3 3.5; 3 -3]; %means
F = generate_scalar_field(dim, scale, theta, corr, var, mu);

%compute graph
nodes = (scale(3)-scale(2))*rand(num_nodes, n) + scale(2); %Randomly generate
    initial positions of MSN
[Nei_agent, A] = findneighbors(nodes, n, r);

plot_graph(true, num_nodes, nodes, Nei_agent)
plot_graph(false, num_nodes, nodes, Nei_agent)

%compute variance vector\
Var = zeros(num_nodes,grid_size,grid_size);
q_ave = mean(nodes);
for k = 1:num_nodes
    for i = 1:grid_size
        for j = 1:grid_size
            Var(k,i,j) = (norm(nodes(k,:) - [meshdim(i) meshdim(j)])^2 + Cv) /
                (r_s(k)^2); %variance matrix
        end
    end
end

%set initial measurement matrix
nodes_va = zeros(num_nodes,grid_size,grid_size);
for k = 1:num_nodes
    for i = 1:grid_size
        for j = 1:grid_size
            if norm(nodes(k,:) - [meshdim(i) meshdim(j)]) <= r_s(k) %acts as
                observance matrix
                nodes_va(k,i,j) = F(i,j) + .001*normrnd(0,Var(i)); %sets noisy
                    measurement
            else
                nodes_va(k,i,j) = 0; %sets to zero
            end
        end
    end
end
nodes_va0 = nodes_va; %save the initial measurement

%Implement Algorithm
iteration = 2;
nodes_va_next = zeros(num_nodes,grid_size,grid_size);
nodes_va_f = zeros(num_nodes,grid_size,grid_size);
history{1} = nodes_va0;
while(1) %loop until convergance (then break)
    for k = i:grid_size     %loop through grid
        for l = 1:grid_size %... and then do same operation as p1
            for i = 1:num_nodes %sum the neighbor weights
                %compute Wii
                if ALG_NUM == 1 %weighted alg2
                    Wii = weighted_design2(i ,i ,Var(:,k,l) ,Nei_agent ,Cv ,r ,
                        r_s);
                elseif ALG_NUM == 2 %metropolis
                    Wii = weighted_metropolis(i, i, Nei_agent);
                end

                %compute sum of Wij_k*X_j_k
                sum = 0;
                for j = 1:size(Nei_agent{i},1) %iterates through neighbors
                    if ALG_NUM == 1 %weighted alg2
                        sum = sum + (nodes_va(Nei_agent{i}(j),k,l) *
                            weighted_design2(i ,Nei_agent{i}(j) ,Var(:,k,l) ,
                            Nei_agent ,Cv ,r ,r_s));
                    elseif ALG_NUM == 2 %metropolis
                        sum = sum + (nodes_va(Nei_agent{i}(j),k,l) *
                            weighted_metropolis(i, Nei_agent{i}(j), Nei_agent))
                            ;
                    end
                end

                %get estimates X_i_k
                nodes_va_next(i,k,l) = Wii*nodes_va(i,k,l) + sum;
            end
        end
    end

    nodes_va = nodes_va_next; %iterate estimates
    history{iteration} = nodes_va;
    %plot_error(F, nodes_va_next, nodes_va_f, scale);

    error = zeros(size(meshdim,2), size(meshdim,2));
    final = zeros(size(meshdim,2), size(meshdim,2));
    for k = 1:size(meshdim,2)
        for l = 1:size(meshdim,2)
            final(k,l) = nodes_va_next(1,k,l);
            error(k,l) = F(k,l) - final(k,l);
        end
    end

    subplot(2,3,4), %new grid
    pcolor(meshdim, meshdim, -final)
    hold on

    subplot(2,3,5), %new mesh
    surf(meshdim, meshdim, -final)
    hold on

    subplot(2,3,3), %stem error
    stem(error(:))
    hold on

    subplot(2,3,6), %mesh error
    surf(meshdim, meshdim, error)
    hold on
end
```