Case 1. Estimate a Single Cell

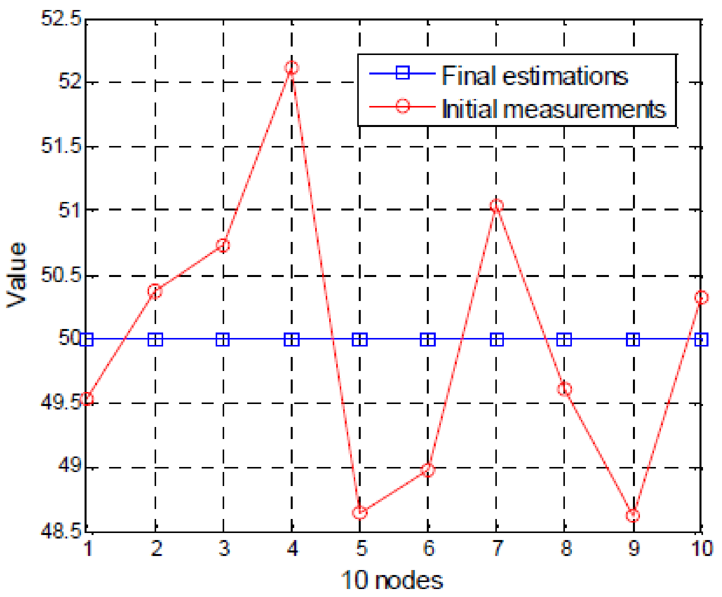
Project 2- Consensus Filter Design

We randomly generate a connected network of 10 nodes in the area of 4x4. The cell is located at the center of this area. The ground truth of the measurement at this location is 50.

Each node makes a measurement as

$$m_i^1 = F^1 + n_i^1. \qquad (1)$$

here $F^1 = 50$, and n_i^1 is the Gaussian noise, $N(0, V_i^1)$, with $V_i^1 = \frac{\|q_i - \overline{q}\|^2 + c_v}{(r_i^s)^2}$, $c_v = 0.01$, $r_1^s = r_2^s = ... = r_{10}^s = 1.6$, and $\overline{q} = \frac{1}{10}\sum_{i=1}^{10} q_i$. The initial condition for running the Consensus Filter 1 is $x_i^1(l = 0) = m_i^1$.

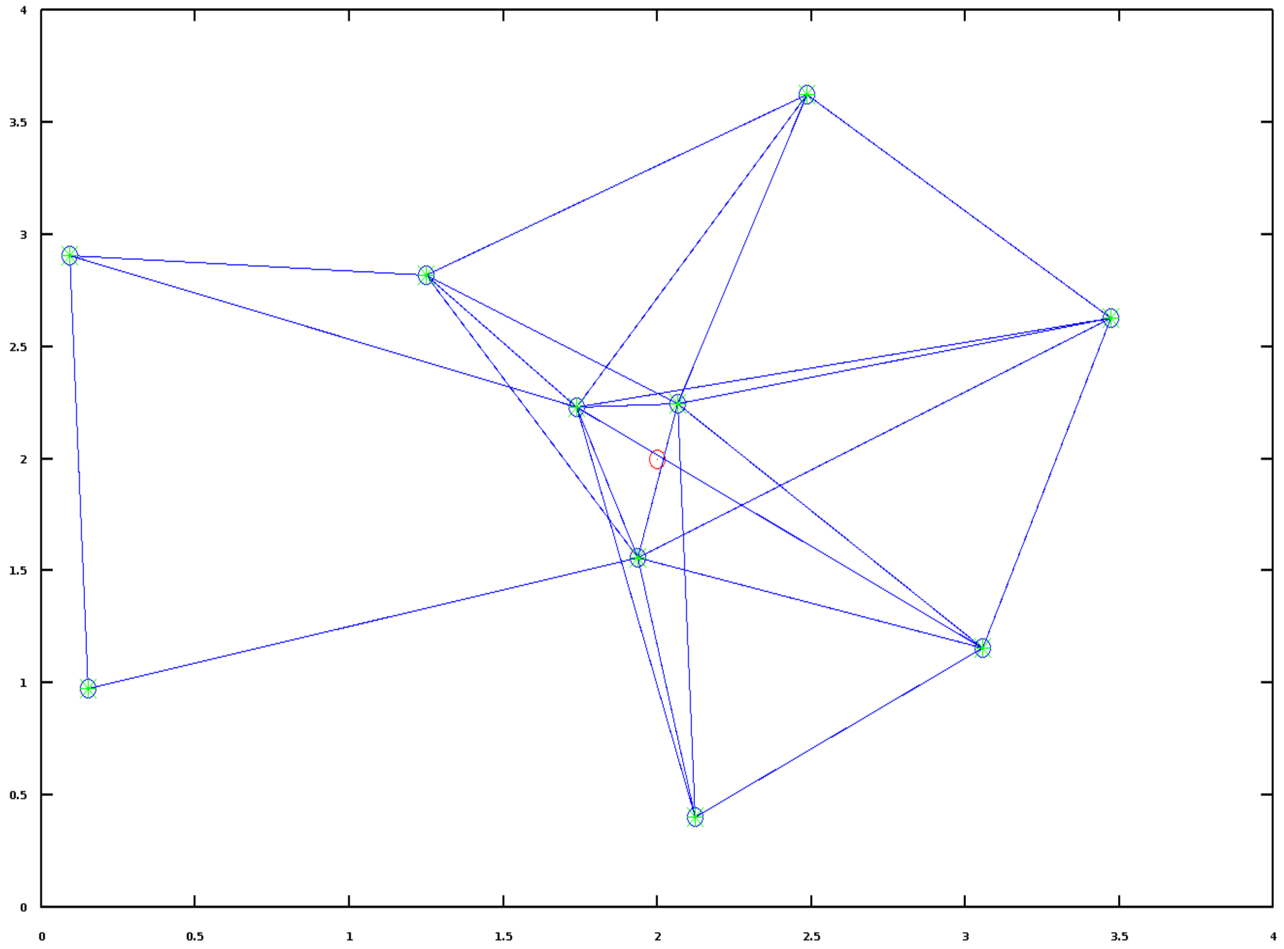


Add noise in Matlab:
N=10;
measure = 50 + randn(N,1).*V;

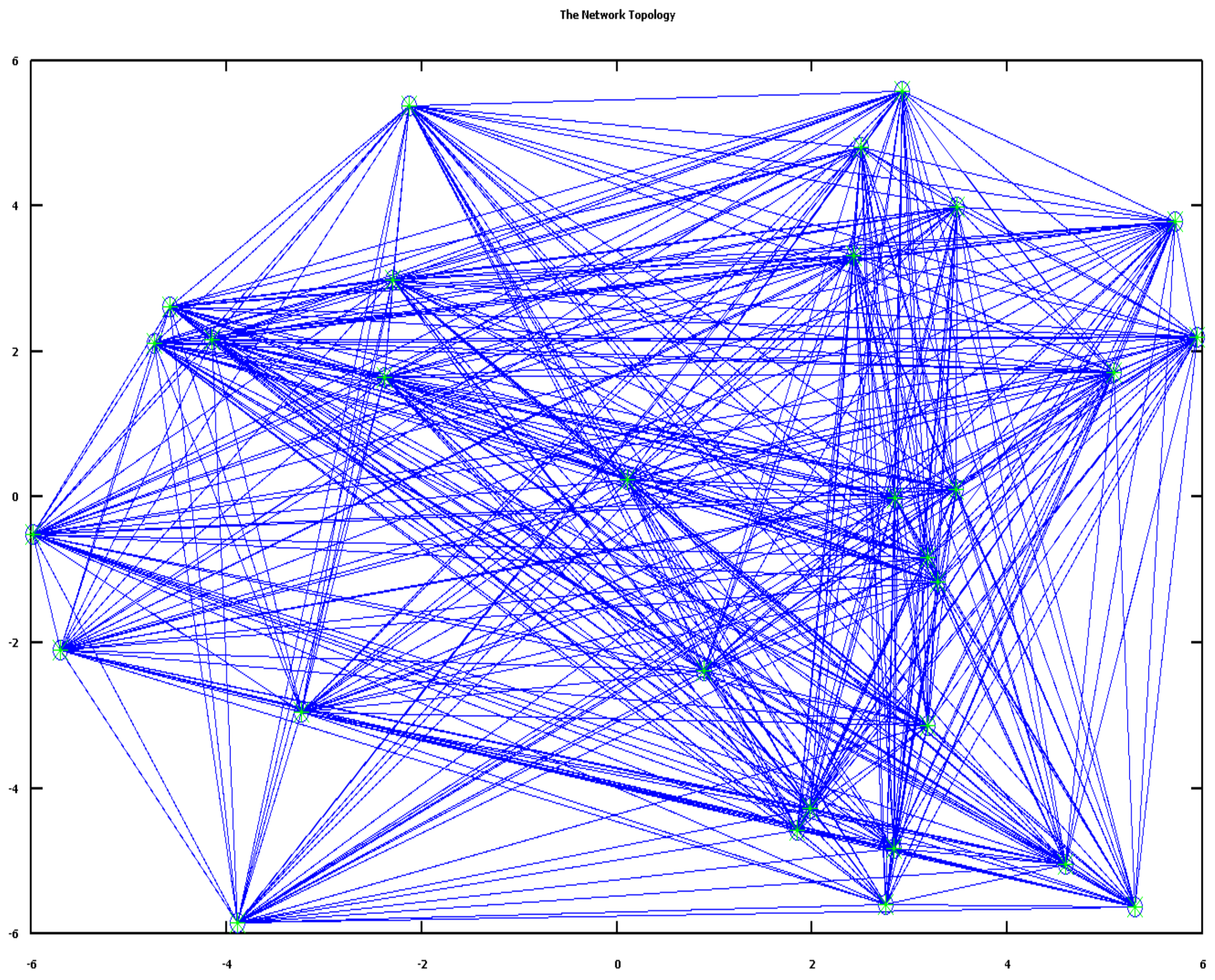
Add noise in Python:

```
for i in xrange(num_nodes):
1. V1[i]=((np.linalg.norm(nodes_position[i]-q)**2)+cv)/(rs**2)
2. n1[i]=np.random.normal(0.0,V1[i])
3. m1[i]=F+n1[i]
```

Example of a network of 10 nodes



All nodes has a communication range, r, of 2 and a sensing range, rs, of 3



The network topology used in case 1. All nodes has a communication range, r, of 17 and a sensing range, r_s, of 17

1. Show the results of the convergence of consensus filter 1 (Weighted Average Consensus) associated with two different weights, i.e., weight design 1 and weight design 2. Explain the obtained results.

Weighted Average Consensus

- Each sensor node makes the observation at cell *k* at time step *t* based on its own confidence (weight).
- Need to find an agreement among the estimates at cell *k* from all sensor nodes in the network.

$$x_i^k(l+1) = w_{ii}^k(t)x_i^k(l) + \sum_{j \in N_i(t)} w_{ij}^k(t)x_j^k(l).$$

$$x_i^k(l=0) = m_i^k(t)$$

$$N_i(t) = \{j \in \vartheta : \|q_j - q_i\| \leq r, \vartheta = \{1, 2, \ldots, n\}, j \neq i\}$$

Weight Design 1 and 2

Weight Design 1

$$w_{ij}^k(l) = \begin{cases} \frac{c_1^w}{V_i^k(t)+V_j^k(t)}, & if \quad i \neq j, j \in N_i(t), \\ 1 - \sum_{j \in N_i(t)} w_{ij}^k(l), & if \quad i = j, \\ 0, & otherwise. \end{cases}$$

Weight Design 2

$$w_{ij}^k(l) = \begin{cases} \frac{c_2^w}{V_i^k(t)}, & if \quad i = j, \\ \frac{1-w_{ii}^k(l)}{|N_i(t)|}, & if \quad i \neq j, j \in N_i(t), \\ 0, & otherwise. \end{cases}$$

Parameter Selection

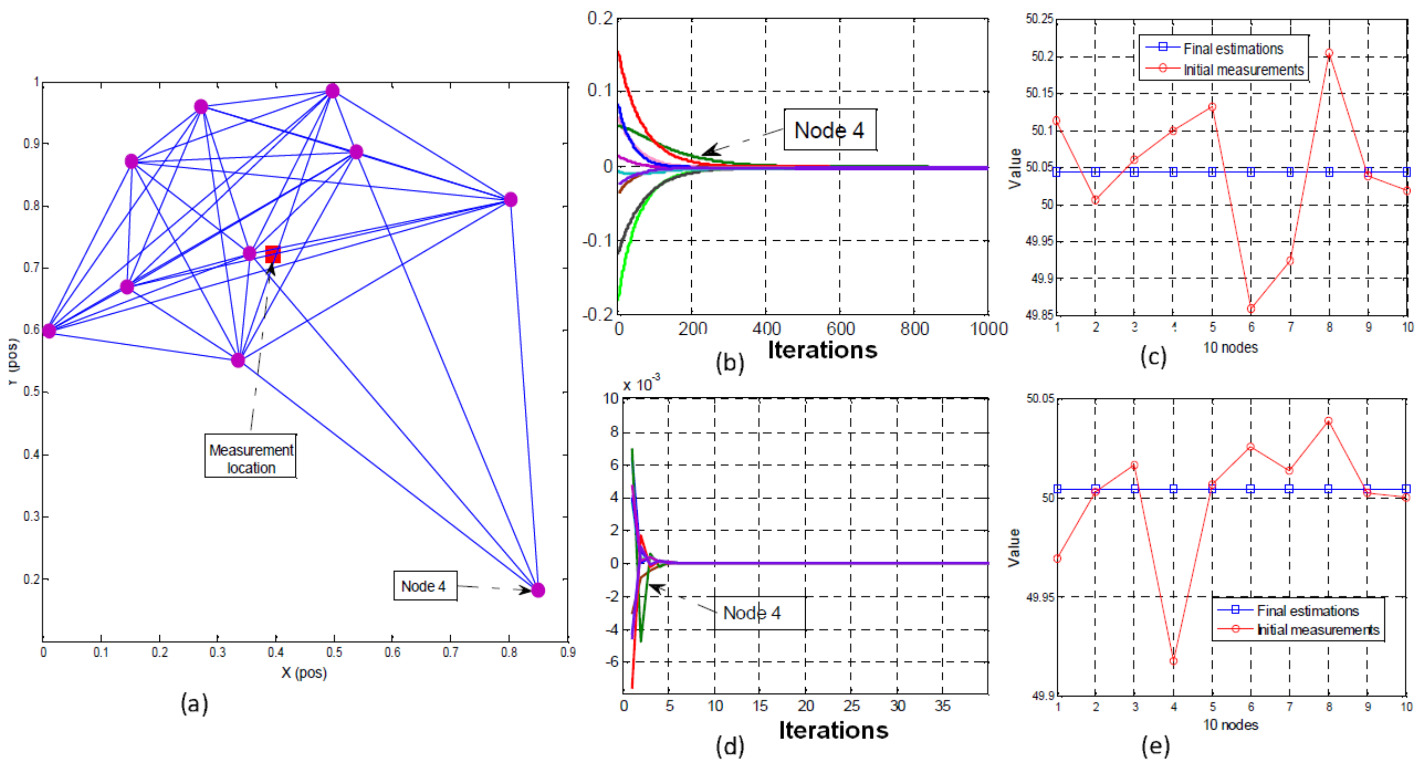
$$\begin{cases} 0 < c_1^w < \frac{2c_v}{(r_i^s)^2|N_i(t)|}, & \text{if } r_i^s = r_j^s = r^s \\ 0 < c_1^w < \frac{1}{|N_i(t)|}\left(\frac{c_v}{\left(r_i^s\right)^2} + \frac{c_v}{\left(r_j^s\right)^2}\right), & \text{otherwise.} \end{cases}$$

$$0 < c_2^w < \frac{c_v}{\left(r_i^s\right)^2}$$

Note: to select c_1w, you can assign $|N_i(t)|$ = total number of nodes -1.

Reference: **H. M. La** and W. Sheng, [Distributed sensor fusion for scalar field mapping using mobile sensor networks](), ***IEEE Transactions on Cybernetics***, pp. 766-778, Vol. 43, No. 2, April, 2013.

Weighted Average Consensus



Compare Weight 1 and 2

2. Show the results of the convergence of consensus filter 2 (Average Consensus) with both Max-degree and Metropolis weights. Explain the obtained results.

Average Consensus

- Each sensor node has its own confidence of the measurement of the value of the scalar field at each cell at each time step *t.*

- Need to find an agreement among the confidences of sensor nodes.

$$y_i^k(l+1) = w_{ii}^k(l)y_i^k(l) + \sum_{j \in N_i(t)} w_{ij}^k(l)y_j^k(l)$$

$$y_i^k(l=0) = w_{ii}^k(t)$$

Using Metropolis weight (Xiao *et al.* 2005)

$$w_{ij}^k(l) = \begin{cases} \frac{1}{1+max(|N_i(t)|,|N_j(t)|)}, & if \quad i \neq j, j \in N_i(t) \\ 1 - \sum_{j \in N_i(t)} w_{ij}^k(l), & if \quad i = j, \\ 0, & \quad otherwise. \end{cases}$$

Weight Design

- **Maximum-degree weights.** Here we use the constant weight $1/n$ on all the edges, and choose the self-weights so that the sum of weights at each node is 1:

$$W_{ij}(t) = \begin{cases} \dfrac{1}{n} & \text{if } \{i,j\} \in \mathcal{E}(t), \\ 1 - \dfrac{d_i(t)}{n} & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

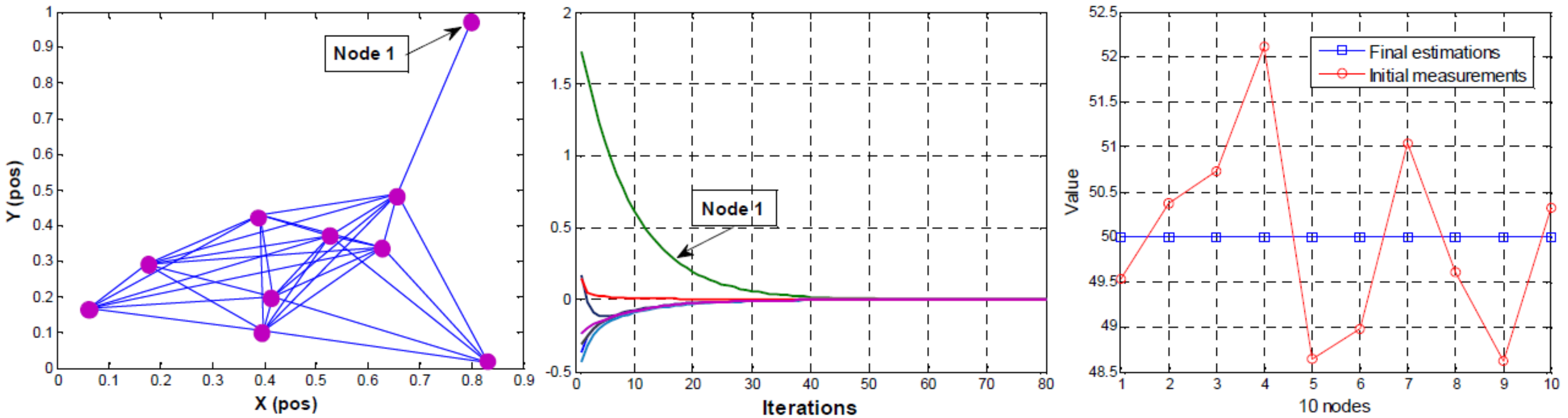
- **Metropolis weights.** The Metropolis weight matrix is defined as

$$W_{ij}(t) = \begin{cases} \dfrac{1}{1+\max\{d_i(t), d_j(t)\}} & \text{if } \{i,j\} \in \mathcal{E}(t), \\ 1 - \displaystyle\sum_{\{i,k\} \in \mathcal{E}(t)} W_{ik}(t) & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

Average Consensus

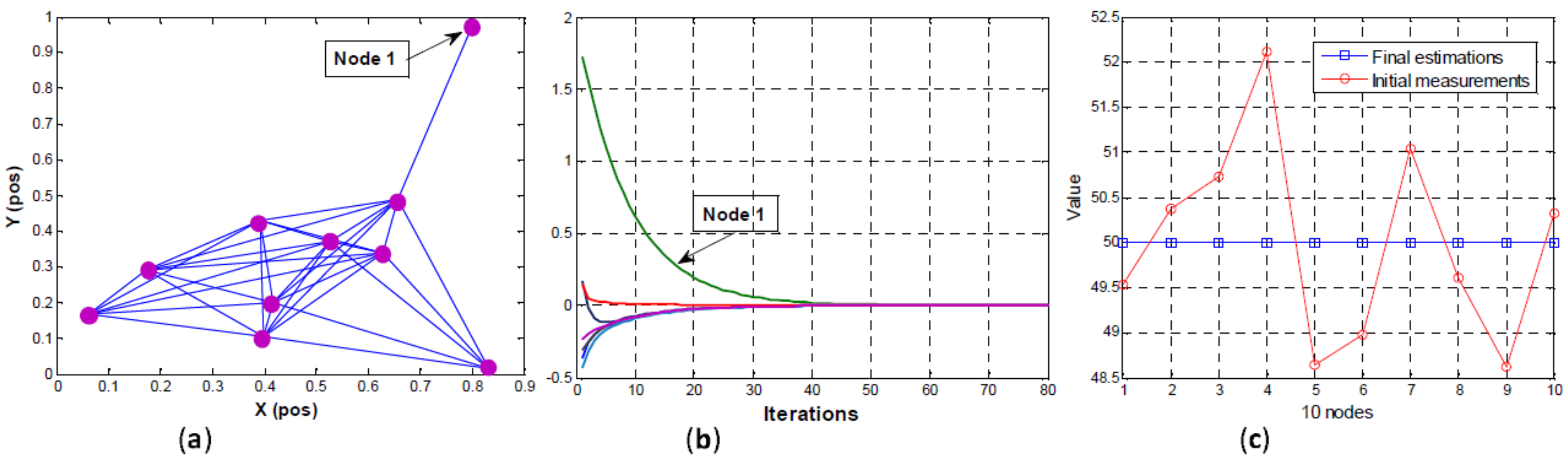
$$y_i^k(l+1) = w_{ii}^k(l) y_i^k(l) + \sum_{j \in N_i(t)} w_{ij}^k(l) y_j^k(l) \qquad y_i^k(l=0) = w_{ii}^k(t)$$

$$w_{ij}^k(l) = \begin{cases} \frac{1}{1+max(|N_i(t)|,|N_j(t)|)}, & if \quad i \neq j, j \in N_i(t) \\ 1 - \sum_{j \in N_i(t)} w_{ij}^k(l), & if \quad i = j, \\ 0, & \quad otherwise. \end{cases}$$

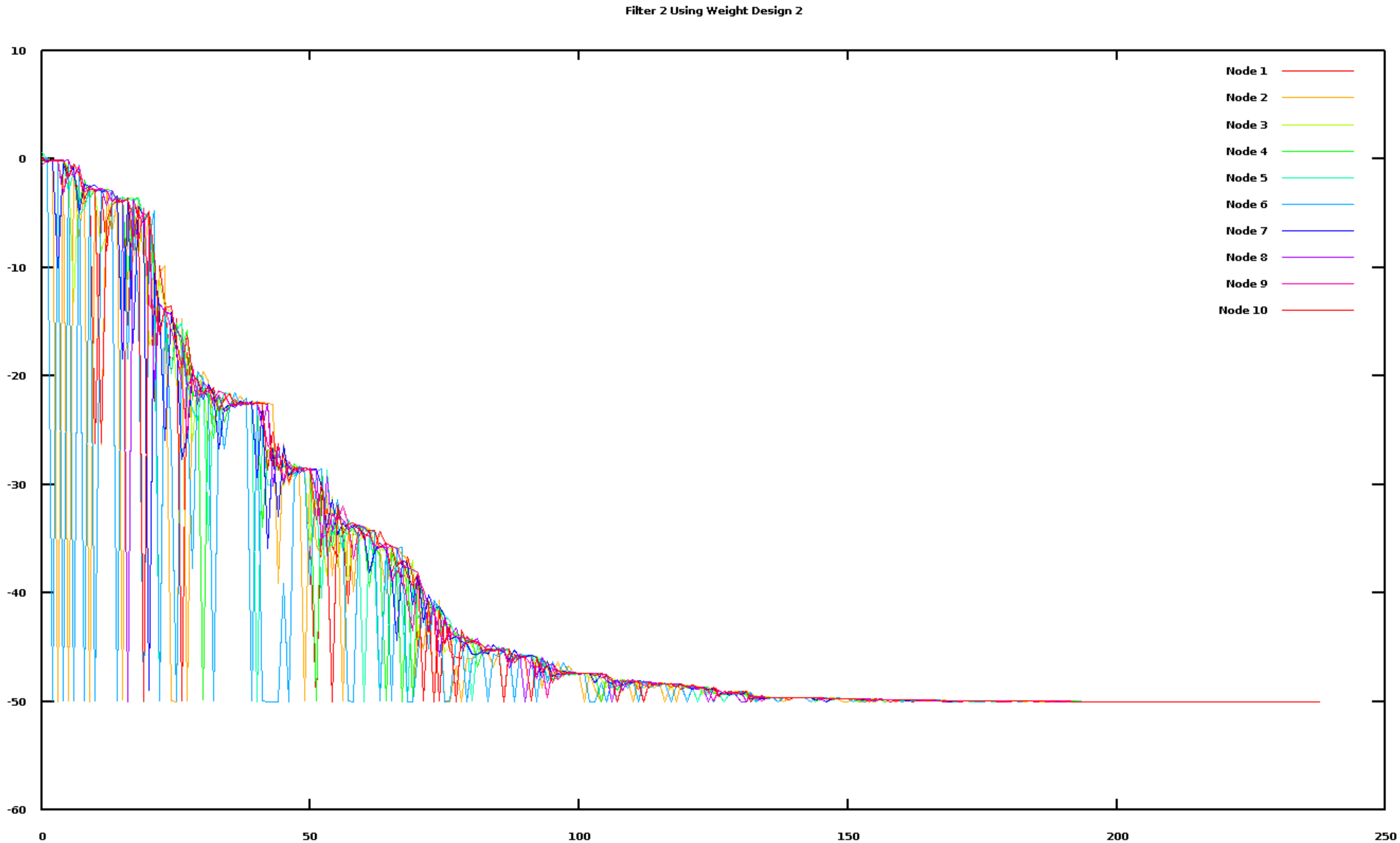


(a) (b) (c)

3. Show the convergence of the node which has the smallest number of neighbors and the node which has the largest number of neighbors. Observe the obtained results and give explanation.



4. Show the convergence of the node which has the smallest number of neighbors and the node which has the largest number of neighbors in the dynamic network case where the node's neighbors are changing over time. Observe the obtained results and give explanation. (Grad Students Only)



```
r = 1; %set communication range
num_nodes = 10;  %Randomly generate nodes
n=2; %number of dimensions
%delta_t = 0.008; %0.008
delta_t_update = 0.008;%0.008; %for sine wave SN1
nodes = rand(num_nodes, n);
%Add measurement for each node: yi= theta + v_i
nodes_va = 50.*ones(num_nodes,1)+ 1*randn(num_nodes,1);
nodes_va0 = nodes_va; %save the initial measurement
```

```
nodes_va_old = nodes_va; %to update the consensus
[Nei_agent, A] = findneighbors(nodes,r,n,
delta_t_update);
figure(1), plot(nodes(:,1),nodes(:,2),
'm>','LineWidth',.2,...
                    'MarkerEdgeColor','m',...
                    'MarkerFaceColor','m',...
                    'MarkerSize',5)
 hold on
for i = 1:num_nodes
    %Line the neighbors together
    tmp=nodes(Nei_agent{i},:);
    for j = 1:size(nodes(Nei_agent{i},1))
       line([nodes(i,1),tmp(j,1)],[nodes(i,2),tmp(j,2)])
    end
end
```

Main Program

```
iteration = 80
Con = [];
for kk = 1: iteration
%===========Compute Metropolis weights===============


%Compute vertex and edge weights

end
```

Case 2. Estimate multiple cells (scalar field)

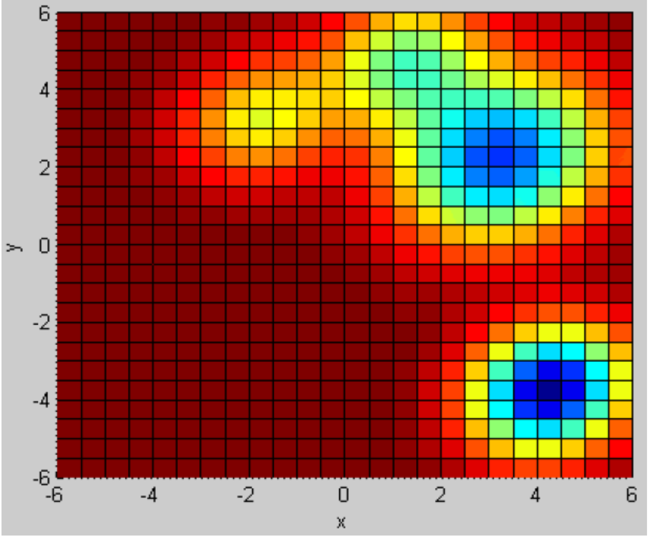
Model of the Scalar Field:

$$F = \Theta\Phi^T \qquad \text{here } \Theta = [\theta_1, \theta_2, \ldots, \theta_K], \text{ and } \Phi = [\phi_1, \phi_2, \ldots, \phi_K]$$

$$\phi_j = \frac{1}{\sqrt{det(C_j)(2\pi)^2}} e^{\frac{1}{2}(x-\mu_x^j)C_j^{-1}(y-\mu_y^j)^T}, j \in [1, 2, \ldots, K]$$

here $[\mu_x^j \ \mu_y^j]$ is the mean of the distribution of function ϕ^j

C_j is covariance matrix $\qquad C_j = \begin{bmatrix} (\sigma_x^j)^2 & c_j^o \sigma_x^j \sigma_y^j \\ c_j^0 \sigma_x^j \sigma_y^j & (\sigma_y^j)^2 \end{bmatrix}$



Each Gaussian kernel can represent an oil leak or chemical leak

Divide the scalar field into cells

Measurement Model:

Each sensor make a measurement at cell k *th*:

$$m_i^k(t) = O_i^k(t)[F^k(t) + n_i^k(t)]$$

here $n_i^k(t)$ is the Gaussian noise with zero mean and variance $V_i^k(t)$ at time step t.

n_i^k is uncorrelated noise which satisfies $Cov(n_i^k(s), n_i^k(t)) = \begin{cases} V_i^k, & if \quad s = t \\ 0, & otherwise, \end{cases}$

$O_i^k(t)$ is the observability of sensor node i at cell k

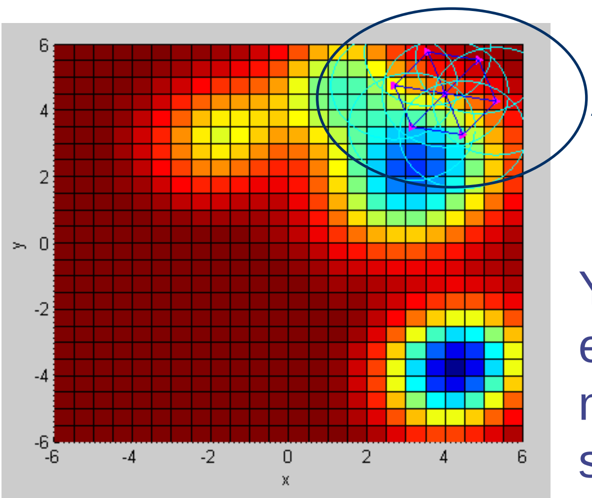
$$O_i^k(t) = \begin{cases} 1, & if \quad \|q_i(t) - q_c^k\| \leq r_i^s \\ 0, & otherwise, \end{cases}$$

Noise variance model:

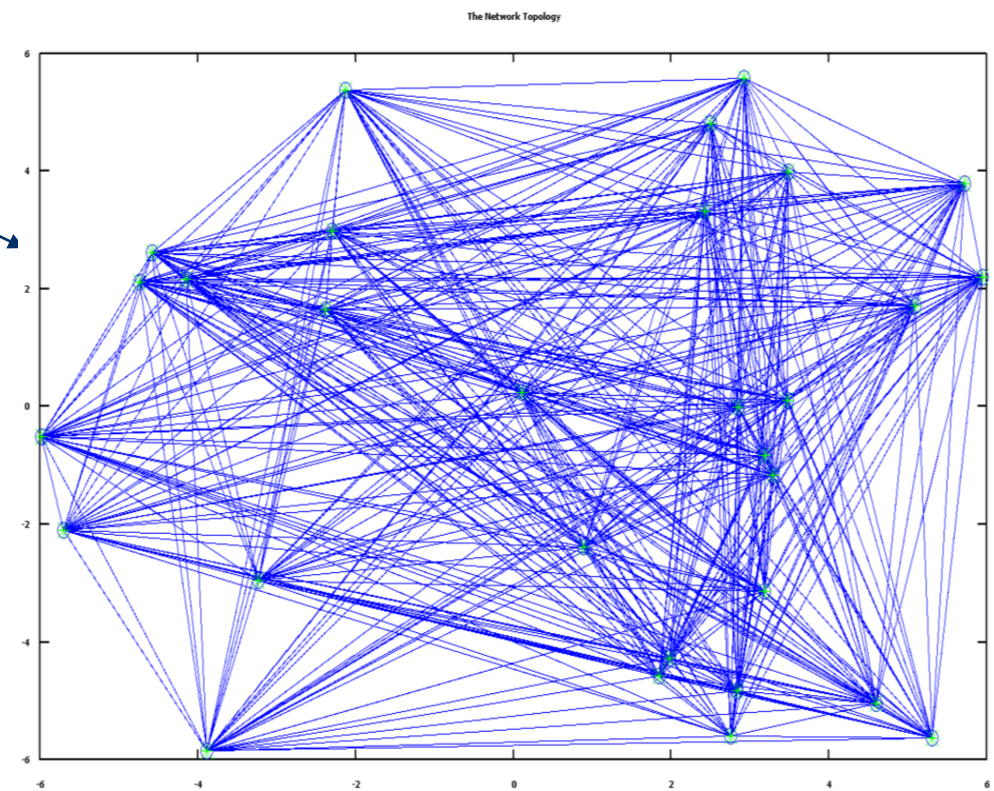
$$V_i^k(t) = \begin{cases} \frac{\|q_i(t) - q_c^k\|^2 + c_v}{(r_i^s)^2}, & if \quad \|q_i(t) - q_c^k\| \leq r_i^s \\ 0, & otherwise, \end{cases}$$

The field *F* has a size *x*×*y* = 12×12, and it is partitioned into 25×25 = 625 cells. You can set variables *x* and *y* run as: *-6 to 6* with scale of *0.5* as presented in the above figure.

1. Generate a connected network of 30 nodes in the area of 12x12 so that they can cover the entire area. You can select the node's sensing range (may be $r_s = 5$).



You need to enlarge the network size to cover the whole field F



2. Running the Consensus 1 (Weighted Average Consensus: can try either Weight design 1 or 2) to obtain the estimate at each cell of the field F

$$x_i^k(l+1) = w_{ii}^k(t)x_i^k(l) + \sum_{j \in N_i(t)} w_{ij}^k(t)x_j^k(l).$$

$$x_i^k(l=0) = m_i^k(t)$$

Weight Design 1

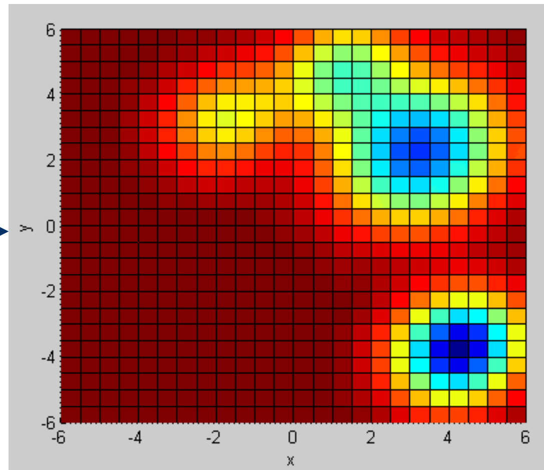
$$w_{ij}^k(l) = \begin{cases} \frac{c_1^w}{V_i^k(t)+V_j^k(t)}, & if \quad i \neq j, j \in N_i(t), \\ 1 - \sum_{j \in N_i(t)} w_{ij}^k(l), & if \quad i = j, \\ 0, & otherwise. \end{cases}$$

Weight Design 2

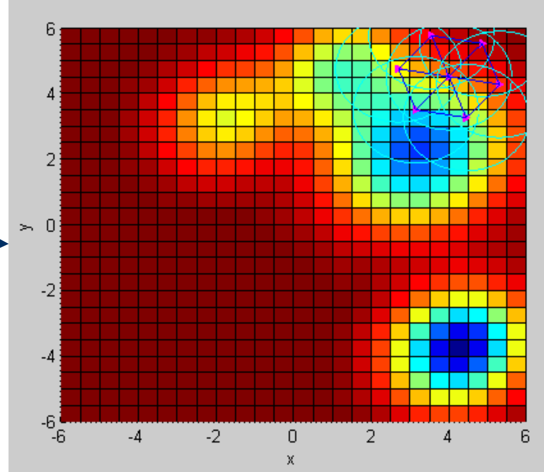
$$w_{ij}^k(l) = \begin{cases} \frac{c_2^w}{V_i^k(t)}, & if \quad i = j, \\ \frac{1-w_{ii}^k(l)}{|N_i(t)|}, & if \quad i \neq j, j \in N_i(t), \\ 0, & otherwise. \end{cases}$$

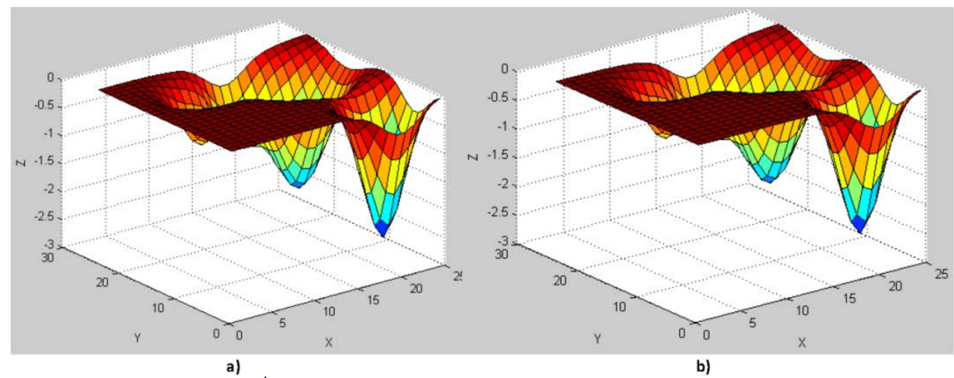
3. Build the map in 2D and/or 3D of this scalar field

Built Map in 2D



Original Map in 2D

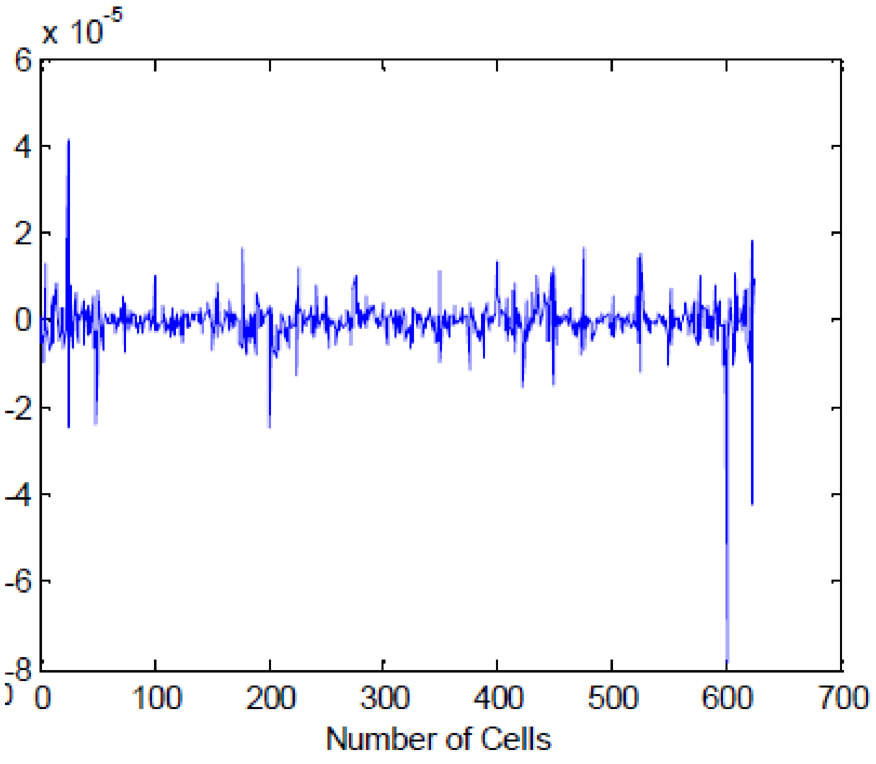


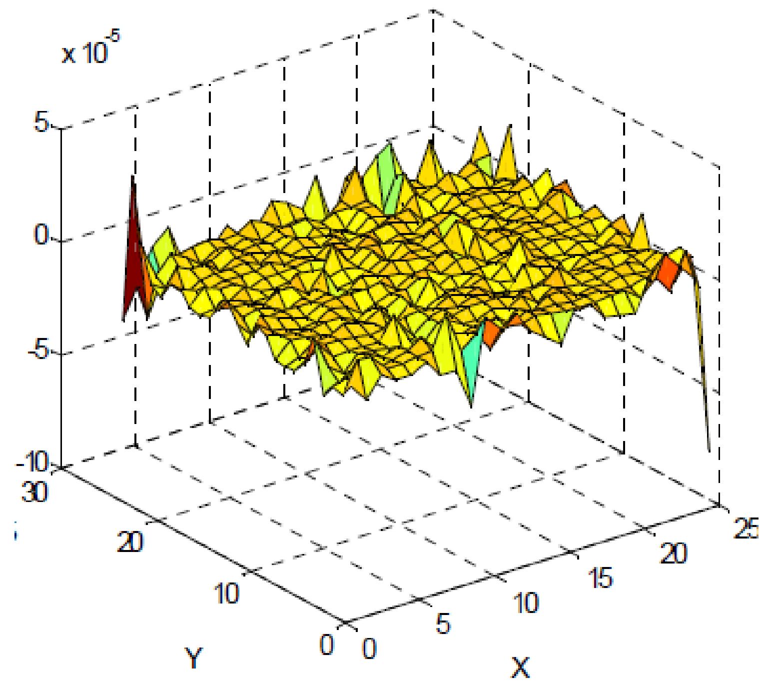


Original Map in 3D

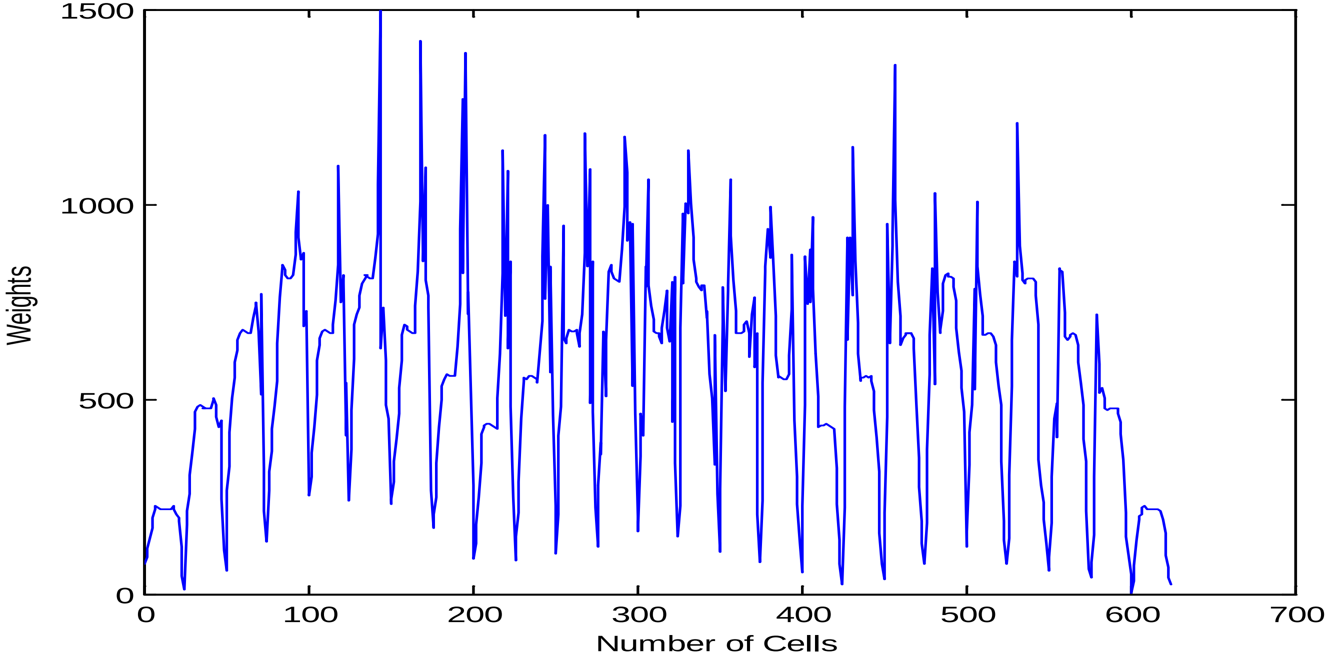
Built Map in 3D

4. Plot the error between the build map and the original one in both 2D and 3D (ignore if difficult), respectively, and give explanation for the obtained results





5. Running the Consensus 2 (Average Consensus) to find out the confidence (weight) of the estimate at each cell, then plot the confidence (weight) in both 2D and 3D **(Graduate Students Only)**



Scalar Field Model

```
%====PARAMETER OF GAUSSIAN MODEL (MULTIVARIATE NORMAL
DISTRIBUTION)===
x_neg = -6;
x_pos = 6;
y_neg = -6;
y_pos = 6;

%Gaussian distribution 0
mu_x1 = 3;%2
mu_y1 = 2;%0
mu1 = [mu_x1 mu_y1];
variance_x1 = 2.25;
variance_y1 = 2.25;
rho1 = 0.1333; %correlation between 2 variabes (#1)
covariance_xy1 = rho1*sqrt(variance_x1*variance_y1);
%Sigma = 0.25*[.25 .3; .3 1];
Sigma1 = [variance_x1 covariance_xy1; covariance_xy1
variance_y1];
```

Scalar Field Model (Continued)

```
mu_x2 = 1;%0;
mu_y2 = 4.5;%2
mu2 = [mu_x2 mu_y2];
variance_x2 = 1.25;
variance_y2 = 1.25;
rho2 = 0.1333; %correlation between 2 variabes (#1)
covariance_xy2 = rho2*sqrt(variance_x2*variance_y2);
%Sigma = 0.25*[.25 .3; .3 1];
Sigma2 = [variance_x2 covariance_xy2; covariance_xy2
variance_y2];

mu_x3 = -2;%0;
mu_y3 = 3;%2
mu3 = [mu_x3 mu_y3];
variance_x3 = 1.25;
variance_y3 = 1.25;
rho3 = 0.1333; %correlation between 2 variabes (#1)
covariance_xy3 = rho3*sqrt(variance_x3*variance_y3);
%Sigma = 0.25*[.25 .3; .3 1];
Sigma3 = [variance_x3 covariance_xy3; covariance_xy3
variance_y3];
```

Scalar Field Model (Continued)

```
mu_x4 = 4;%0;
mu_y4 = -4;%2
mu4 = [mu_x4 mu_y4];
variance_x4 = 1.25;
variance_y4 = 1.25;
rho4 = 0.1333; %correlation between 2 variabes (#1)
covariance_xy4 = rho4*sqrt(variance_x4*variance_y4);
%Sigma = 0.25*[.25 .3; .3 1];
Sigma4 = [variance_x4 covariance_xy4; covariance_xy4
variance_y4];

%Devide the Region F into cells
scal = 0.5;%0.1;
x = x_neg:scal:x_pos; %x dimension of the survellance Region
y = y_neg:scal:y_pos; %y dimension of the survellance Region
[X1,X2] = meshgrid(x,y);
Theta = [30 10 8 20]; %The true constant vector
```

Scalar Field Model (Continued)

```
Phi = [mvnpdf([X1(:) X2(:)],mu1,Sigma1) mvnpdf([X1(:)
X2(:)],mu2,Sigma2) mvnpdf([X1(:) X2(:)],mu3,Sigma3)
mvnpdf([X1(:) X2(:)],mu4,Sigma4)]; %The distribution of
the environment

F1 = Theta*Phi';

num_cells = length(F1);

F = reshape(F1,length(y),length(x)); %Find cell's value
(y by x matrix)
%F =[F2(2,:); F2(1,:)];
figure(1), surf(-F) %Plot the true map
```