To: CPE 470
From: Brendan Aguiar
Date: May 10, 2021
Subject: Consensus Filters for a Multi-Robot System

Report

Introduction

The purpose of this report is to show the convergence of a consensus agreement between sensor nodes about the measurement of k cells. The weight designs implemented are Weight Design 1, Weight Design 2, the Metropolis Design, and the Maximum-Degree Design. Each weight design is a factor in the weighted Average consensus algorithm, which outputs future node measurement predictions of the cell. The sensors and cells are both time fixed in this report, meaning both the nodes and the environment are unmoving and the neighbors are fixed.

Weighted Average & Average Consensus

The weighted average consensus uses the weights of a given nodal network along with an associated cell measurement to produce a consensus of the nodes of the measurement. Shown in Fig. 1 is the nodal network used for the first part of the experiment. The network consists of 10 nodes on a 1x1 grid. To accommodate the close proximity, the sensing range for each node has been scaled to the tenths place. Changing the sensing range will change the amount of neighbors each node has. The sensing range is used to compute the design factors for Weight Designs 1 and 2. It's also used to compute the noise variance of the model.

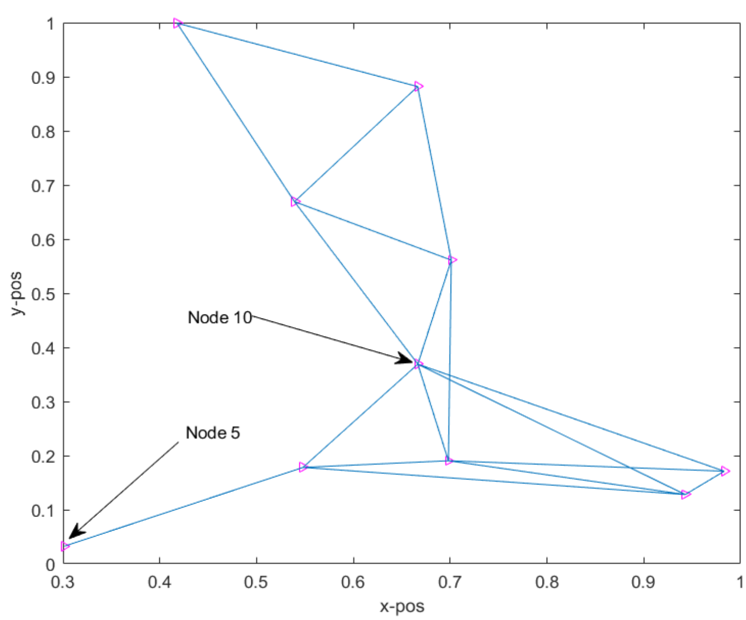

Fig. 1: The following network was used for the weight designs.

Weight Design 1

The first design converges linearly with each iteration. Eventually, the convergence will become logarithmic over many iterations. This weight design is obtained by solving and setting all the vertex and edge weights to one. The weights of each nodes' neighbors are calculated using the distance of each node from the average node and the sensing range. Shown in Fig. 2, the nodes with the most neighbors, took the most time to converge. The node that took the least time to converge also had the least amount of neighbors. The node with the most neighbors took the longest to converge. As shown in Fig. 3, the initial estimate of node 10 was the furthest from the final estimate.

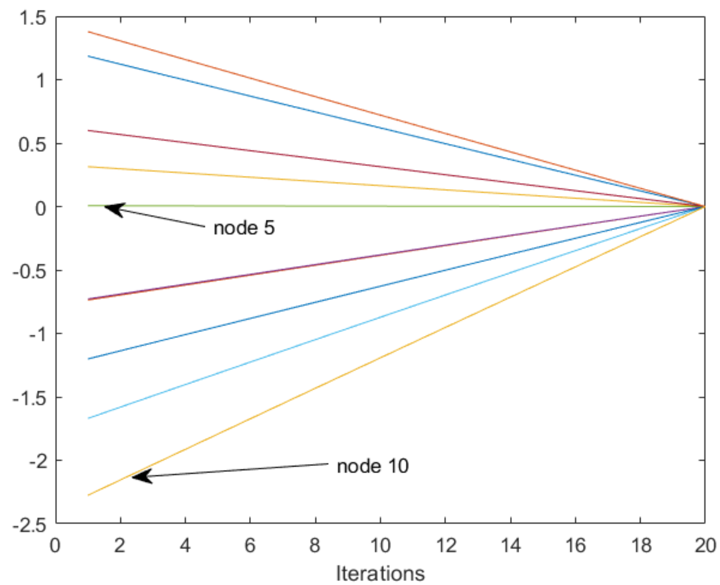

Fig. 2: Nodes five and ten have the smallest and largest neighbors respectively.

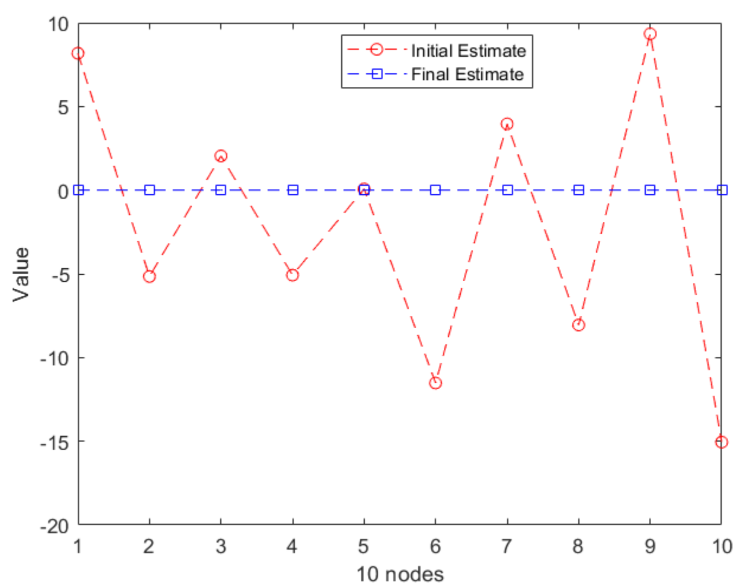

Fig. 3: Weight Design had the biggest error range of all the designs.

Weight Design 2

Weight design 2 is similar to Weight Design 1 where it uses the weights of the nodes and the sensing range to generate weights. First the weights of each node to itself is calculated using a design factor, the sensing range, and the noise variance. Then, those weights are used along with the number of neighbors each node has to calculate the remaining weights. As shown in Fig. 4. The consensus algorithm begins to fail. This is a problem with the actual code. The convergences change directions until the last iteration where the final and initial estimates are equal.

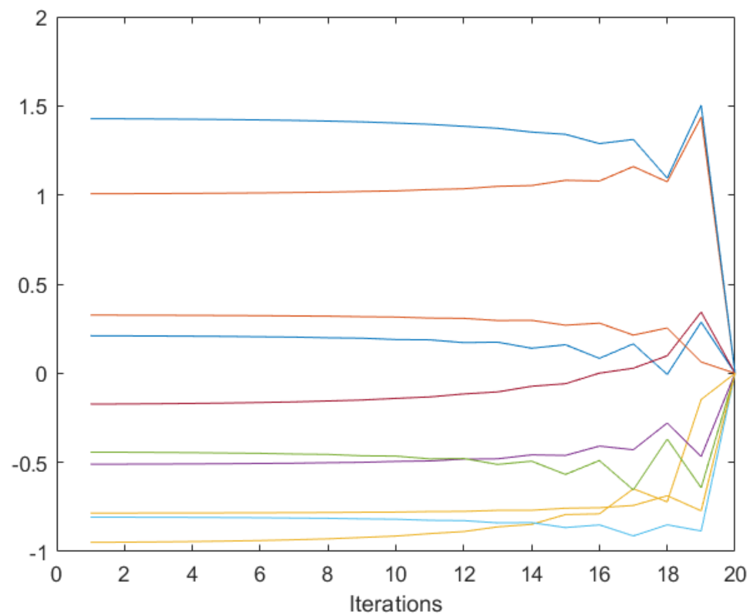

Fig. 4: The Weighted Consensus Algorithm cannot adequately find the measurement.

The performance of the consensus algorithm could be partially analyzed by comparing the initial measurements of each design. For design 1, the initial estimate errors were off by around five each time. The initial estimates for design 2 show improvement by ranging within two as shown in Fig. 5. This could indicate that Weight Design 2 performs better at first.

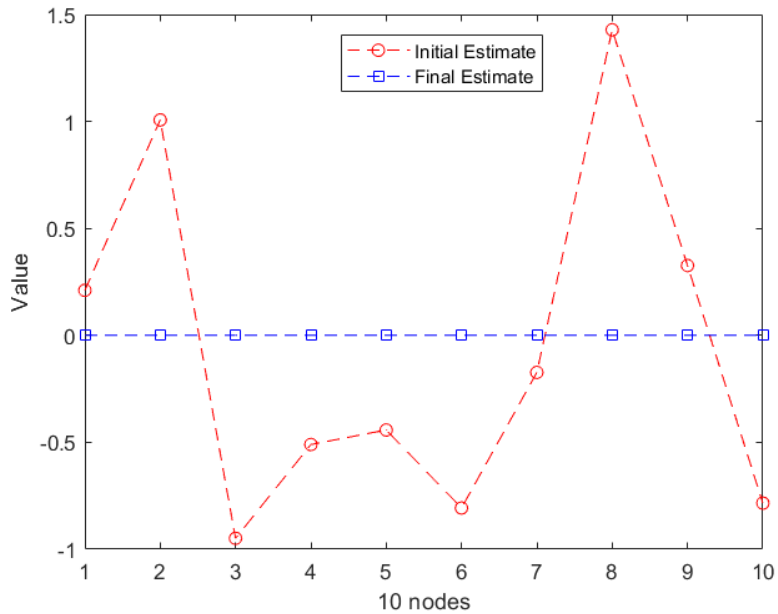

Fig. 5: Although no initial measurements are exact to the final, they are close.

Metropolis Design

The Metropolis Design is the first design that does not include the noise variance or design factors. Instead, for all neighbors of each node, the weight is calculated using the max number between the number of neighbors between the node and the number of neighbors of the node's neighbors. The sum of the weights for all the nodes should sum up to 1. The convergence is shown in Fig. 6.

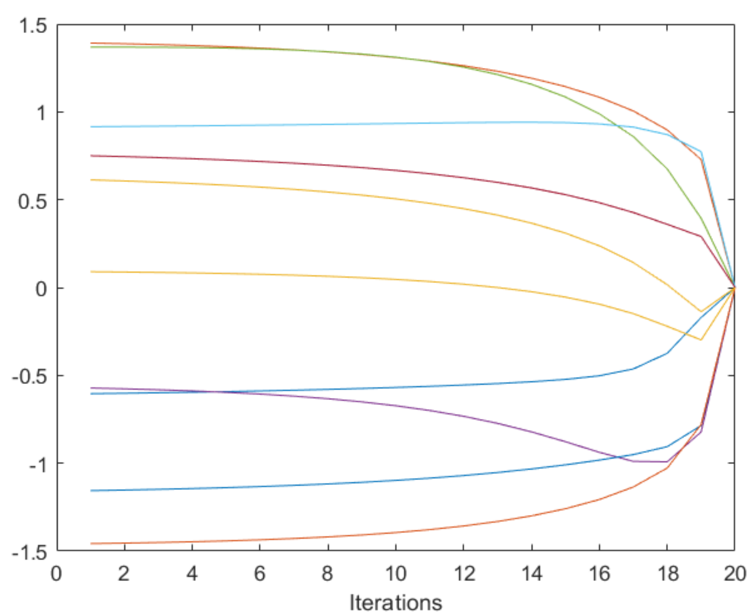

Fig. 6: The convergence is minimal of each node until the final iterations.

Similarly to Weight Design 2, the nodes struggle to converge. This may just be a graphical error. The weights of the nodes total up to 1 as expected. As shown in Fig. 7, the estimate range is close to what the actual estimate is.

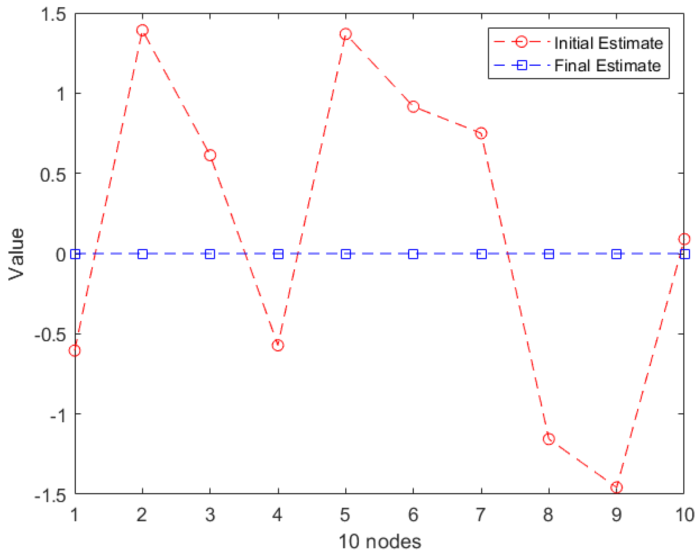

Fig. 7: The estimate range for each node is similar to Weight design 2 and Max-degree design.

Maximum-Degree Design

The Maximum-Degree Design is the simplest of the weight designs used in the experiment. The weights depend entirely on the number of nodes used to model the system and the number of neighbors each node has. Fig. 8 shows the convergence of the maximum-degree design.

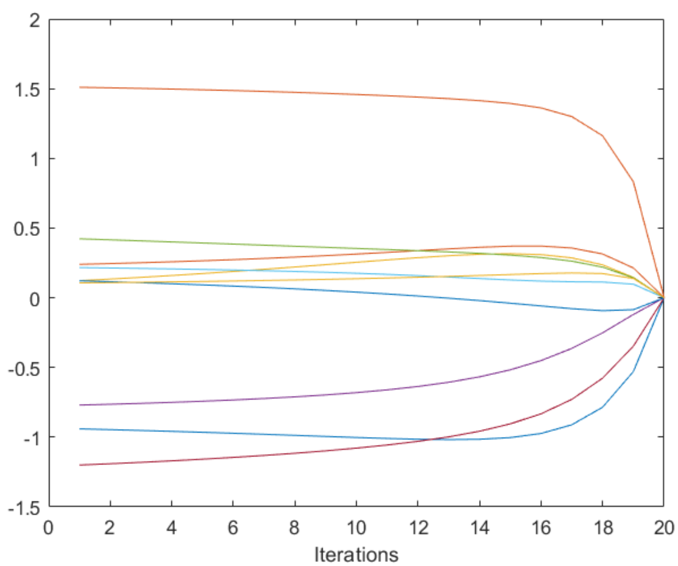

Fig. 8: Although, still suffering from issues, the curvature has increased to the convergence.

In order to optimize performance, the maximum-degree design should only be used when the number of nodes isn't too big. As the number of nodes increases, the size of the weights decrease. In the case of a larger node network, designs like Weight Design 2 should be used instead. The initial estimates using the maximum-degree function proved optimal outperforming the other designs as shown in Fig. 9. This may be due to the smaller size of the system used in the experiment.

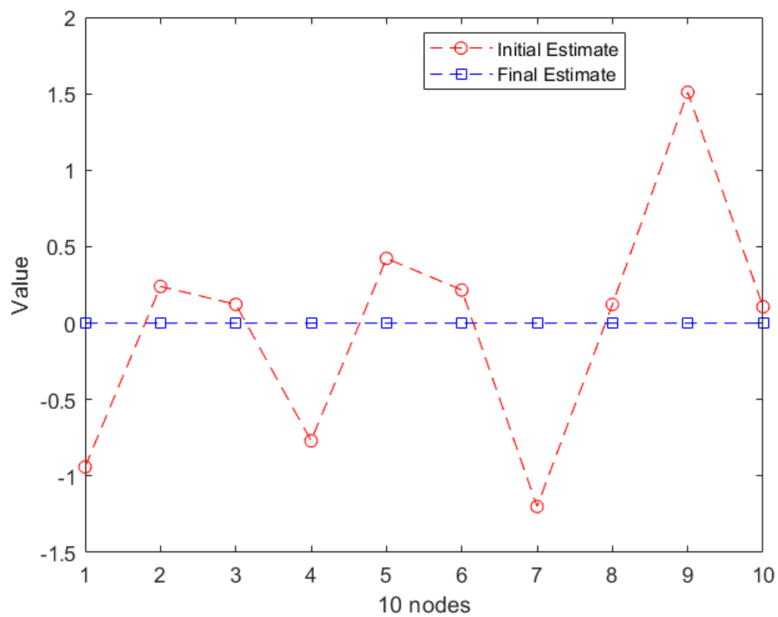

Fig. 9: The node with the most neighbors, node ten, is the best initial estimate.

Convergence of 50 node Network

A system of 50 node networks performs just as well as a network of 10 nodes. The node with the least amount of neighbors in both cases converged the fastest in Weight design . The node with the most neighbors had the closest initial estimate to the final estimate using the Maximum-degree Design. Fig. 10 shows the 50 node network generated. Fig. 11 shows the convergence of the 50 node network using Weight Design 1.

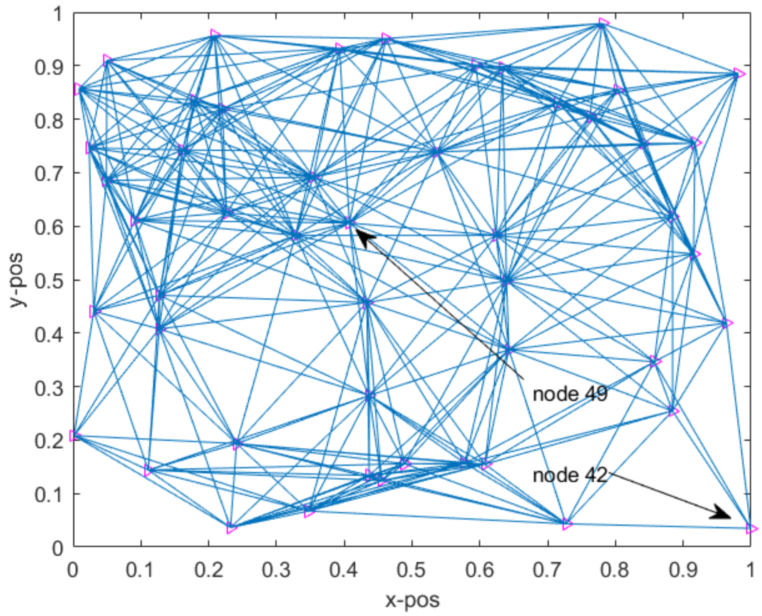

Fig. 10: Node 42 has only six neighbors while node 49 has 22 neighbors.

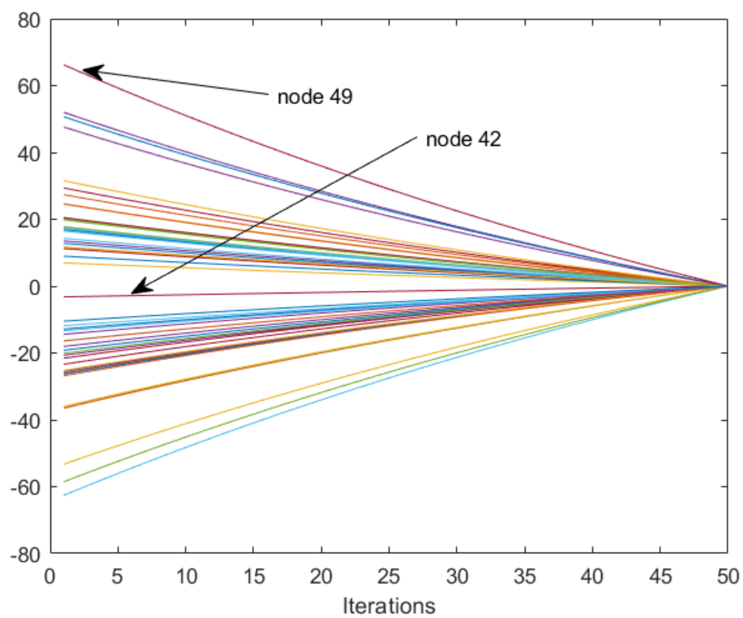

Fig. 11: The convergence of 50 nodes behaves similarly to the convergence of 10.

The remainder of this report should include the use of a multi-cell field of 625 cells. A node network of 30 nodes should cover the entire area. The weighted average consensus should be found using Weight Design 1 and Weight Design 2 and the average consensus should be found using the Metropolis and Maximum-Degree design. Finally, a scalar map of the field should be generated.

Appendix

```
clc
close all
clear

%code by Jim La
num_nodes = 10;%50; %number of nodes
r = .4; %communication range <==========================Change r here
n = 2; % number of dimensions
delta_t_update = .008;
%nodes = rand(num_nodes, n); %uncomment to generate new node set
%save('Nodes.txt', 'nodes');
nodes = importdata('Nodes.txt');
F = 50;
%nodes = nodes * 4;
%Add measurement for each node
M = F * ones(num_nodes, 1) + 1 * randn(num_nodes, 1);
m_i = M; % Save initial measurement
[Nei_agent, A] = findneighbors(nodes, r, n, delta_t_update);


%Code By Brendan Aguiar
cv = .01;
W = zeros(num_nodes, num_nodes);%Weights initialized to 0
% generate design factor
c1_w = ((2 * cv) / (r * r * (num_nodes - 1))).* rand(1,1);
c2_w = (cv / (r * r)) .*rand(1,1);
%generate V
V = zeros(num_nodes, 1);
q_bar = mean(nodes);
%Q = transpose(q_bar);
for i = 1:num_nodes
   e_d = norm(nodes(i,:) - q_bar);
   V(i) = (e_d + cv) / (r * r);
end
%Select Weight Design
str = '-Make a Selection-';
str = [str newline '1) Weight Design 1'];
str = [str newline  '2) Weight Design 2'];
str = [str newline '3) Metropolis Design'];
str = [str newline '4) Maximum Degree Design'];
str = [str newline 'Choice : '];
```

```
choice = input(str);
switch choice
   case 1 % Weight Design 1 set L to 500
      for i = 1:num_nodes % for each node
         for j = 1:num_nodes %for each weight
            for k = 1:size(Nei_agent{i})%for each neighbor
               TEST = Nei_agent{i}(k);
               if j == TEST
                  if i ~= j
                     W(i,j) = c1_w / (V(i) + V(j));
                  end
               end
            end
         end
         for j = 1:num_nodes % for each weight where i equals j
            if i == j
               W(i,j) = 1 - sum(W(i));
            end
         end
      end
   case 2 % Weight Design 2 set r to .5 and L to 50
      for i = 1:num_nodes %for each node
          W(i,i) = c2_w / V(i);
      end
      for i = 1:num_nodes % for each node
         for j = 1:num_nodes %for each weight
            m = size(Nei_agent{i});
            for k = 1:size(Nei_agent{i})%for each neighbor
               TEST = Nei_agent{i}(k);
               if j == TEST
                  if i ~= j
                     W(i,j) = (1 - W(i,i)) / m(:,1);
                  end
               end
            end
         end
      end
   case 3 % Metropolis Design set L to 15
      for i = 1:num_nodes %for each node
         for j = 1:num_nodes %for each weight
            for k = 1:size(Nei_agent{i})%for each neighbor
               TEST = Nei_agent{i}(k);
               if j == TEST
                  m = size(Nei_agent{i});
```

```
                n = size(Nei_agent{j});
                W(i,j) = 1 / (max(m(:,1), n(:,1)) + 1);
            end
        end
    end
    for j = 1:num_nodes %for each weight
        if i == j

            W(i,j) = 1 - sum(W(i,Nei_agent{i}));
        end
    end
end
case 4 % Maximum Degree Design set L to 50
    for i = 1:num_nodes %for each node
        for j = 1:num_nodes%for each weight
            if i == j
                m = size(Nei_agent{i});
                W(i,j) = 1 - (m(:,1) / num_nodes);
            end
        end
        for j = 1:num_nodes %for each weight
            for k = 1:size(Nei_agent{i})%for each neighbor
                TEST = Nei_agent{i}(k);
                if j == TEST
                    W(i,j) = 1 /num_nodes;
                end
            end
        end
    end
otherwise % Terminate program
        disp('No Design selected\n');
        quit;
end

L = 50;%<=====================================Change L here
Val = zeros(9,1);
X = zeros(L, num_nodes);
X(1,:) = m_i;% first iteration
for j = 2:L %Weighted Average Consensus
    for i = 1:num_nodes
        temp1 = transpose(X((j - 1), Nei_agent{i}));
        temp2 = W(i,Nei_agent{i});
        Val = temp2 * temp1;
        X(j,i) = W(i,i) * X((j - 1),i) + Val;
```

```
    end
end

j = L;
C = zeros(L,num_nodes);

for i = 1:L %calculating convergence C
    for k = 1:num_nodes
        C(i,k) = (X(j,k) - X(1,k));
        if floor(k/2) == k/2
            C(i,k) = C(i,k) * -1;%flip every other node
        end
    end
    j = j - 1;
end
%plotting node network
figure(1), plot(nodes(:,1), nodes(:,2), 'm>', 'Linewidth', .2, ...
                'MarkerEdgeColor', '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

%plotting Convergence
figure(2), plot(C)
%plotting final vs. initial measurements
figure(3), plot(C(1,:), '--ro')
hold on
plot(C(L,:), '--bs')
```