```
clear
format shortG
warning off;
SetRNG(111);
% generate input data
%hierarchy 1 and 2 generation
k = 4;
dim = 100;
mu_range = [-1000 1000];
sigma = 100;
split_num = 2;
[center1,mu_mat] = subcluster_centroid(k,dim,split_num,mu_range,sigma);
means = [];


%hierarchy 3 generation
center2 = [];
for i = 1:size(center1,1)
    r = normrnd(0,20,[3,100])+center1(i,:);
    center2 = [center2;r];
end
center = center2;

% data generation with the given centroids
dim = 100;
num = 100;
sigma = 5;
[inputs,c] = subcluster_simulate(center,dim,num,sigma);
assign_id1 = repelem([1:4],600)';
assign_id2 = repelem([1:8],300)';
assign_id3 = repelem([1:24],100)';

% inputs = [inputs,assign_id1,assign_id2,assign_id3];
% %inputs = inputs(randperm(size(inputs, 1)),:);
% assign_id1 = inputs(:,end-2);
% assign_id2 = inputs(:,end-1);
% assign_id3 = inputs(:,end);
% inputs = inputs(:,1:end-3);
data = inputs';

c2c_dist_layer1 = norm(mu_mat(2,:)'-mu_mat(1,:)',2);
c2c_dist_layer2 = norm(center1(2,:)'-center1(1,:)',2);
c2c_dist_layer3 = norm(center2(2,:)'-center2(1,:)',2);

layers = ["centroid A and centroid B";"centroid A1 and centroid
A2";"centroid A1α and centroid A1β"];
distance = [c2c_dist_layer1;c2c_dist_layer2;c2c_dist_layer3];
T_overall = table(layers,distance)
```

```
T_overall = 3×2 table
```

	layers	distance
1	"centroid A and centroid B"	8287.6
2	"centroid A1 and centroid A2"	1398.9
3	"centroid A1α and centroid A1β"	292.56

```
% generate synaptic weights
SetRNG(111);
n_src = 100;
n_dst = 2500;
n_per_src = 20;
synaptic_weights_mat = randn(n_src,n_dst)*1000;
[srcIdx,dstIdx] = ConnectHypergeometric(n_dst, n_src, n_per_src);
index = [srcIdx;dstIdx];
for i = 1:n_dst;
    nonzero_idx = index(2,find(index(1,:) == i));
    zero_idx = setdiff(1:n_src,nonzero_idx);
    synaptic_weights_mat(zero_idx,i) = 0;
end
cells = synaptic_weights_mat; %original
```

```
tree1 = linkage(data','average','cosine');


cluster_assign_cycle = zeros(size(data,2),4);

% cycle 1, find the cluster winners
cycle = 1;
ori_cycle1_cells = cells;
cycle1_cells = normc(cells);
cycle1_cells_iter = normc(cells);

for k = 1:20;
    cycle1_winners = [];
    sampled_data = data(:, randperm(size(data, 2)));
    winner_len_mat = [];
    winner_average = zeros(100,1,2500);
    for col = 1:size(sampled_data,2); % loop over all inputs
        lr = 0.009;
        input1 = sampled_data(:,col);% each input
        len_input = norm(input1);
        input1 = normc(input1);
        winner_per_cycle = [];
        product = input1'*cycle1_cells; % the dot products of the input and
all cells
        winning_value = max(product); % max dot product value
```

```
        winning_idx = find(product == winning_value); % index(indices) of
winning cell(s)
        winning_cell = cycle1_cells(:,winning_idx); % the winning cell set,
which may contain more than one winning cell
        % loop over each winning cell in the set, in case of tied winners,
        % which may not happen
        updated_winningset = []; % in case of tied winners
        for cell_idx = winning_idx;
            winner = cycle1_cells_iter(:,cell_idx);
            cycle1_winners = [cycle1_winners;cell_idx];
            update_winner_ori = winner+(input1-winner)*lr;
            winner_average(:,:,cell_idx) = update_winner_ori +
winner_average(:,:,cell_idx);

            len_update_winner = norm(update_winner_ori);
            update_winner = update_winner_ori/len_update_winner;
            updated_winningset = [updated_winningset;update_winner];
            winner_len_mat = [winner_len_mat;
[cell_idx,len_input]];
            cycle1_cells(:,cell_idx) = update_winner;
            cycle1_cells_iter(:,cell_idx) = update_winner_ori;
            %ori_cycle1_cells(:,cell_idx) = update_winner*len_input;
        end

    end
end

[~,~,ix] = unique(cycle1_winners,"stable");
winner_stats = [unique(cycle1_winners,"stable"),accumarray(ix,1)];

for i = 1:size(winner_stats,1);
winner_average(:,:,winner_stats(i,1)) =
winner_average(:,:,winner_stats(i,1))./winner_stats(i,2);
cycle1_cells(:,winner_stats(i,1)) = winner_average(:,:,winner_stats(i,1));
end


% each input wins which?
cycle1_winner=cycle1_winners;
cycle1_winners = unique(cycle1_winners,"stable");
[~,idx] = max(normr(inputs)*cycle1_cells(:,cycle1_winners),[],2);
cluster_assign_cycle(:,cycle) = cycle1_winners(idx);
cycle1_winners = cycle1_winners(unique(idx,"stable"));


for c = cycle1_winners';
    each_input = winner_len_mat(winner_len_mat(:,1)== c,2);
    ori_cycle1_cells(:,c) = cycle1_cells(:,c)*mean(each_input);
end
cycle1_winner_demask = ori_cycle1_cells(:,cycle1_winners);
```

```
% before cycle 1 plots
final_all_data = [inputs;cells(:,cycle1_winners)'];
[coeff,~,~,~,explained,~] = pca(final_all_data);
Zb1=final_all_data*coeff(:,1:3);
explained_b = round(explained);

final_all_data = [inputs;mu_mat;ori_cycle1_cells(:,cycle1_winners)'];
[coeff,~,~,~,explained,~] = pca(final_all_data);
Za1=final_all_data*coeff(:,1:3);
explained_a = round(explained);



id = ["A","B","C","D"];


cycle1_winner_demask = ori_cycle1_cells(:,cycle1_winners);

[~,sort] = min(pdist2(mu_mat,cycle1_winner_demask'),[],2);
cycle1_winner_demask = cycle1_winner_demask(:,sort);
m1 = round(pdist2(mu_mat,cycle1_winner_demask'),1);
m2 = round(pdist2(mu_mat,mu_mat),1);
m3 = round(pdist2(cycle1_winner_demask',cycle1_winner_demask'),1);
T1_cw = array2table(m1);
T1_cc= array2table(m2);
T1_ww = array2table(m3);


[~,p] = ttest2(diag(m1),setdiff(m1,diag(m1)));
[~,p] = ttest2(diag(m1),setdiff(m1,diag(m1)),'Tail','left');


true_label = assign_id3;
assigned_cluster = cluster_assign_cycle(:,1);
PTY1 = purity(assigned_cluster,true_label);
NMI1 = nmi(true_label, assigned_cluster);
[RI1, ARI1] = randindex(true_label, assigned_cluster);
[s1] = ClusterEvalSilhouette (data', assigned_cluster, 'cosine');
ch1 = ClusterEvalCalinskiHarabasz(data', assigned_cluster);
[db1] = ClusterEvalDaviesBouldin (data', assigned_cluster);
```

```
%cycle 2, find the sub-cluster winners
% determine the clutser

cycle = 2;
ori_cycle2_cells = ori_cycle1_cells;
cycle2_cells = cycle1_cells;
cycle2_cells_iter = cycle1_cells;
```

```
ori_cycle2_cells(:,cycle1_winners) = 0;
cycle2_cells(:,cycle1_winners) = 0;
cycle2_cells_iter(:,cycle1_winners) = 0;

cycle2_winner_mat = [];
this_winner_mat = [];

for c = 1:size(cycle1_winners,1);
    lr = 0.02;
    this_idx = c;
    this_winner_idx = cycle1_winners(this_idx);
    rawinput_idx = find(cluster_assign_cycle(:,cycle-1) == this_winner_idx);
    sampled_data_fix = data(:,rawinput_idx);  %select each cluster data
    this_winner = ori_cycle1_cells(:,this_winner_idx); % the cell that
"this" cluster wins in cycle 1
    sampled_data_fix = sampled_data_fix-this_winner;
    this_winner_mat = [this_winner_mat;this_winner';this_winner'];
    for k = 1:20; % each cluster learns 100 rounds
        winner_len_mat = [];
        cycle2_winners = [];
        sampled_data = sampled_data_fix(:, randperm(size(sampled_data_fix,
2)));
        winner_average = zeros(100,1,2500);
        %sampled_data(1)
        for col = 1:size(sampled_data,2); % loop over all inputs
            input1 = sampled_data(:,col);
            len_input = norm(input1);
            input1 = normc(input1);
            product = input1'*cycle2_cells; % the dot products of the input
and all cells
            winning_value = max(product); % max dot product value
            winning_idx = find(product == winning_value);
            for cell_idx = winning_idx;
                winning_cell = cycle2_cells_iter(:,cell_idx);
                cycle2_winners = [cycle2_winners;cell_idx];
                winner_len_mat = [winner_len_mat;[cell_idx,len_input]];
                update_winner = winning_cell+(input1-winning_cell)*lr;
                winner_average(:,:,winning_idx) = update_winner +
winner_average(:,:,winning_idx);
                cycle2_cells_iter(:,winning_idx) = update_winner;

                update_winner = normc(update_winner);
                cycle2_cells(:,cell_idx) = update_winner;
                %cycle2_cells_copy(:,cell_idx) = update_winner;

            end
        end
    end

    [~,~,ix] = unique(cycle2_winners,"stable");
```

```
    winner_stats = [unique(cycle2_winners,"stable"),accumarray(ix,1)];
    for i = 1:size(winner_stats,1);
        winner_average(:,:,winner_stats(i,1)) =
winner_average(:,:,winner_stats(i,1))./winner_stats(i,2);
        cycle2_cells_iter(:,winner_stats(i,1)) =
winner_average(:,:,winner_stats(i,1));
    end
    cycle2_winners = unique(cycle2_winners);

    [~,idx] = max(normc(sampled_data_fix)'*cycle2_cells(:,cycle2_winners),
[],2);
    cluster_assign_cycle(rawinput_idx,cycle) = cycle2_winners(idx);

    cycle2_winners = cycle2_winners(unique(idx,"stable"));

%     [~,idx]= max((center1(2*c-1:2*c,:)-
this_winner')*cycle2_cells_copy(:,cycle2_winners),[],2);
%     cycle2_winners = cycle2_winners(idx); % now it's the right #
    cycle2_winner_mat = [cycle2_winner_mat;cycle2_winners];

    for c1 = cycle2_winners';
        each_input = winner_len_mat(winner_len_mat(:,1)== c1,2);
        ori_cycle2_cells(:,c1) = cycle2_cells_iter(:,c1)*mean(each_input);
    end
    cycle2_cells(:,cycle2_winners) = 0;
    cycle2_cells_iter(:,cycle2_winners) = 0;
end


cycle2_winnerA_idx = cycle2_winners(1);
cycle2_winnerB_idx = cycle2_winners(2);
cycle2_winnerA_b = ori_cycle1_cells(:,cycle2_winnerA_idx);
cycle2_winnerB_b = ori_cycle1_cells(:,cycle2_winnerB_idx);
cycle2_winnerA = ori_cycle2_cells(:,cycle2_winnerA_idx);
cycle2_winnerB = ori_cycle2_cells(:,cycle2_winnerB_idx);
cycle2_winner_demask = ori_cycle2_cells(:,cycle2_winner_mat)
+this_winner_mat';


final_all_data = [sampled_data';cycle2_winnerA_b';cycle2_winnerB_b'];
[coeff,~,~,~,explained,~] = pca(final_all_data);
Zb2=final_all_data*coeff(:,1:3);
explained_b = round(explained);

final_all_data = [sampled_data';cycle2_winnerA';cycle2_winnerB'];
[coeff,~,~,~,explained,~] = pca(final_all_data);
Za2=final_all_data*coeff(:,1:3);
explained_a = round(explained);
```

```
final_all_data =
[inputs;center1;cycle1_winner_demask';cycle2_winner_demask'];
[coeff,~,~,~,explained,~] = pca(final_all_data);
Z14=final_all_data*coeff(:,1:3);
explained = round(explained);



[~,sort] = min(pdist2(center1,cycle2_winner_demask'),[],2);
cycle2_winner_demask = cycle2_winner_demask(:,sort);
m1 = round(pdist2(center1,cycle2_winner_demask'),1);
T2_cw = array2table(m1);
[~,p] = ttest2(diag(m1),setdiff(m1,diag(m1)));


m2 = round(pdist2(center1,center1),1);
T2_cc = array2table(m2);
[~,p] = ttest2(diag(m1),setdiff(m2,diag(m2)));

m3 = round(pdist2(cycle2_winner_demask',cycle2_winner_demask'),1);
T2_ww = array2table(m3);
[~,p] = ttest2(diag(m1),setdiff(m3,diag(m3)));


tree2 = linkage(sampled_data_fix','average','cosine');


true_label = assign_id3;
assigned_cluster = cluster_assign_cycle(:,2);
PTY2 = purity(assigned_cluster,true_label);
NMI2 = nmi(true_label, assigned_cluster);
[RI2, ARI2] = randindex(true_label, assigned_cluster);
[s2] = ClusterEvalSilhouette (data', assigned_cluster, 'cosine');
ch2 = ClusterEvalCalinskiHarabasz(data', assigned_cluster);
[db2] = ClusterEvalDaviesBouldin (data', assigned_cluster);
```

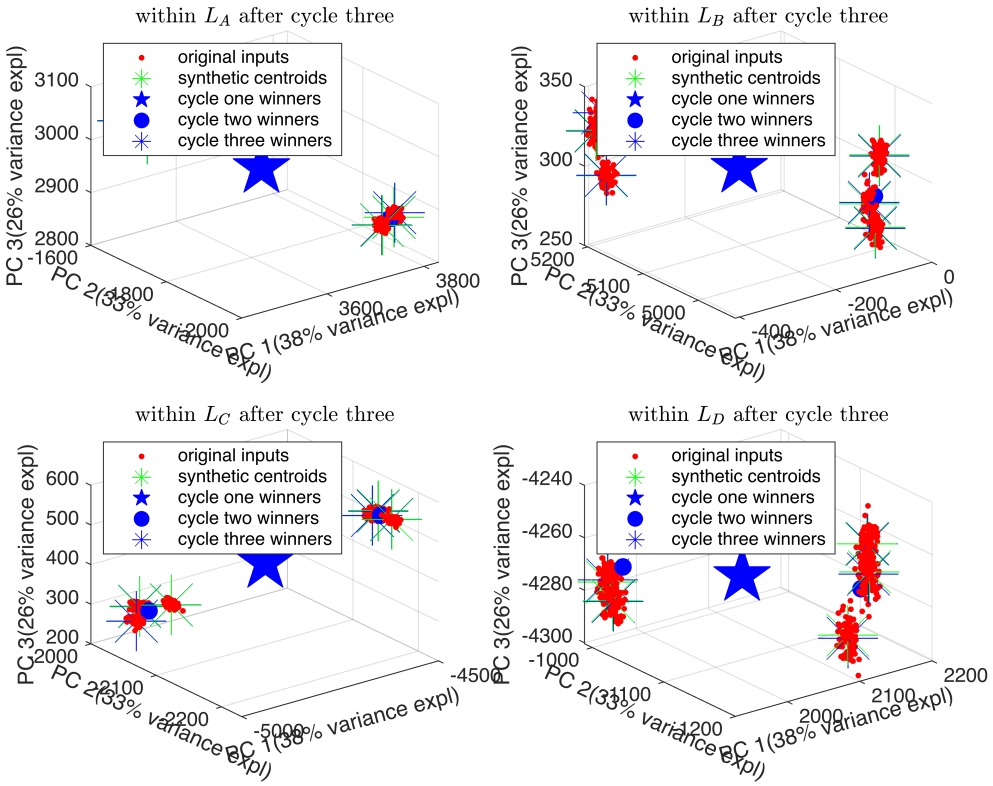
```
%cycle 3 sub-sub-cluster
cycle = 3;
ori_cycle3_cells = ori_cycle2_cells;
cycle3_cells = cycle2_cells;
cycle3_cells_iter = cycle2_cells;

ori_cycle3_cells(:,cycle2_winner_mat) = 0;
cycle3_cells(:,cycle2_winner_mat) = 0;
cycle3_cells_iter(:,cycle2_winner_mat) = 0;
```

```
cycle3_winner_mat = [];
this_winner1_mat = [];
this_winner2_mat = [];

for c = 1:size(cycle2_winner_mat,1);
    this_winner_idx = cycle2_winner_mat(c);
    rawinput_idx = find(cluster_assign_cycle(:,cycle-1) == this_winner_idx);
    sampled_data_fix = data(:,rawinput_idx);

    this_idx_1 = unique(cluster_assign_cycle(rawinput_idx,1));
    this_winner_1 = ori_cycle1_cells(:,this_idx_1);
    this_winner_2 = ori_cycle2_cells(:,cycle2_winner_mat(c));
    this_winner1_mat =
[this_winner1_mat;this_winner_1';this_winner_1';this_winner_1'];
    this_winner2_mat =
[this_winner2_mat;this_winner_2';this_winner_2';this_winner_2'];
    sampled_data_fix = sampled_data_fix-this_winner_1-this_winner_2;

    for k = 1:20;
        lr = 0.009;
        sampled_data = sampled_data_fix(:, randperm(size(sampled_data_fix,
2)));
        winner_average = zeros(100,1,2500);
        cycle3_winners = [];
        winner_len_mat = [];
        for col = 1:size(sampled_data,2);
            input1 = sampled_data(:,col);
            len_input = norm(input1);
            input1 = normc(input1);
            product = input1'*cycle3_cells;
            winning_value = max(product);
            winning_idx = find(product == winning_value);

            for cell_idx = winning_idx;
                winning_cell = cycle3_cells_iter(:,cell_idx);
                cycle3_winners = [cycle3_winners;cell_idx];
                winner_len_mat = [winner_len_mat;[cell_idx,len_input]];
                update_winner = winning_cell+(input1-winning_cell)*lr;
                winner_average(:,:,cell_idx) = update_winner +
winner_average(:,:,cell_idx);
                %cycle3_cells_copy(:,cell_idx) = update_winner;
                cycle3_cells_iter(:,cell_idx) = update_winner;
                update_winner = normc(update_winner);
                cycle3_cells(:,cell_idx) = update_winner;

            end
        end
    end
```

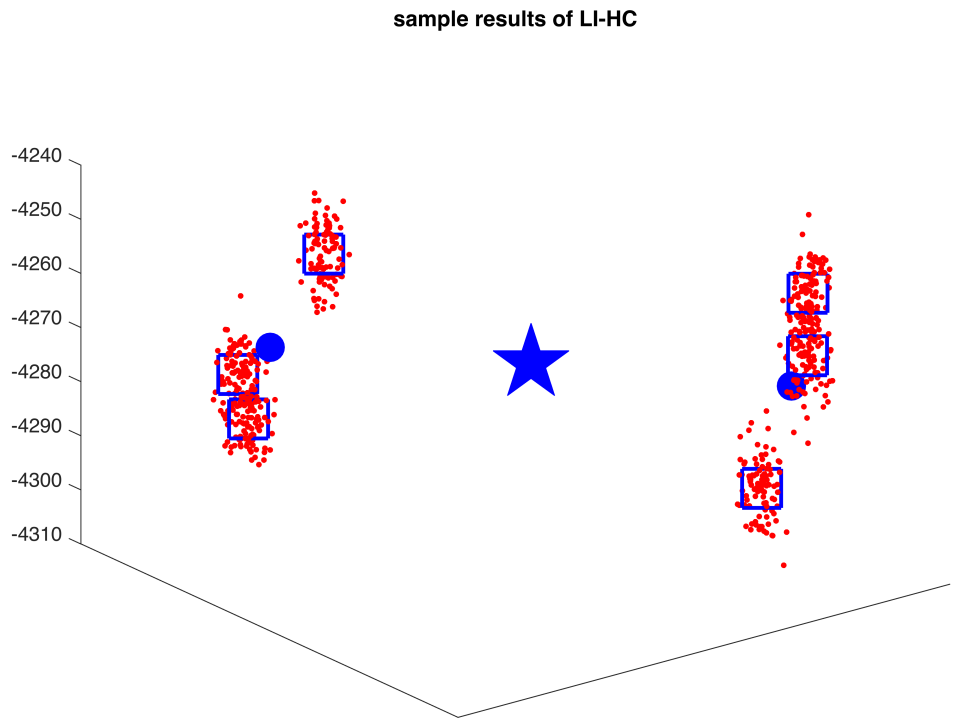
```
    [~,~,ix] = unique(cycle3_winners,"stable");
    winner_stats = [unique(cycle3_winners,"stable"),accumarray(ix,1)];
    %
    for k = 1:size(winner_stats,1);
        winner_average(:,:,winner_stats(k,1)) =
winner_average(:,:,winner_stats(k,1))./winner_stats(k,2);
        cycle3_cells_iter(:,winner_stats(k,1)) =
winner_average(:,:,winner_stats(k,1));
    end

    [~,idx] = max(normc(sampled_data_fix)'*cycle3_cells(:,cycle3_winners),
[],2);
    cluster_assign_cycle(rawinput_idx,cycle) = cycle3_winners(idx);
    cycle3_winners = cycle3_winners(unique(idx,"stable"));


    %          [~,idx]= max((center(3*c-2:3*c,:)-this_winner_1'-
this_winner_2')*cycle3_cells(:,cycle3_winners),[],2);
    %          cycle3_winners = cycle3_winners(idx); % now it's the right #
    cycle3_winner_mat = [cycle3_winner_mat;cycle3_winners];

    for c1 = cycle3_winners';
        each_input = winner_len_mat(winner_len_mat(:,1)== c1,2);
        ori_cycle3_cells(:,c1) = cycle3_cells_iter(:,c1)*mean(each_input);
    end
    cycle3_cells(:,cycle3_winners) = 0;
    cycle3_cells_iter(:,cycle3_winners) = 0;
end

cycle3_winnerA_idx = cycle3_winner_mat(22);
cycle3_winnerB_idx = cycle3_winner_mat(23);
cycle3_winnerC_idx = cycle3_winner_mat(24);
cycle3_winnerA = ori_cycle3_cells(:,cycle3_winnerA_idx);
cycle3_winnerB = ori_cycle3_cells(:,cycle3_winnerB_idx);
cycle3_winnerC = ori_cycle3_cells(:,cycle3_winnerC_idx);

cycle3_winnerA_b = ori_cycle2_cells(:,cycle3_winnerA_idx);
cycle3_winnerB_b = ori_cycle2_cells(:,cycle3_winnerB_idx);
cycle3_winnerC_b = ori_cycle2_cells(:,cycle3_winnerC_idx);

final_all_data =
[sampled_data';cycle3_winnerA_b';cycle3_winnerB_b';cycle3_winnerC_b'];
[coeff,~,~,~,explained,~] = pca(final_all_data);
Zb3=final_all_data*coeff(:,1:3);
explained_b = round(explained);

final_all_data =
[sampled_data';cycle3_winnerA';cycle3_winnerB';cycle3_winnerC'];
[coeff,~,~,~,explained,~] = pca(final_all_data);
```

```
Za3=final_all_data*coeff(:,1:3);
explained_a = round(explained);



cycle3_winner_demask = ori_cycle3_cells(:,cycle3_winner_mat)
+this_winner1_mat'+this_winner2_mat';
final_all_data =
[inputs;center;cycle1_winner_demask';cycle2_winner_demask';cycle3_winner_dem
ask'];
[coeff,~,~,~,explained,~] = pca(final_all_data);
Z14=final_all_data*coeff(:,1:3);
explained = round(explained);



id = ["A","B","C","D"];
figure;
for j = 1:4;
    data_idx = find(cluster_assign_cycle(:,1) == cycle1_winners(j));
    subplot(2,2,j);
    hold on
    view(3)

plot3(Z14(data_idx,1),Z14(data_idx,2),Z14(data_idx,3),'r.','MarkerSize',10)

plot3(Z14(2400+6*j-5:2400+6*j,1),Z14(2400+6*j-5:2400+6*j,2),Z14(2400+6*j-5:2
400+6*j,3),'g*','MarkerSize',30)

plot3(Z14(2424+j,1),Z14(2424+j,2),Z14(2424+j,3),'bp','MarkerFaceColor','blue
','MarkerSize',30)

plot3(Z14(2428+2*j-1:2428+2*j,1),Z14(2428+2*j-1:2428+2*j,2),Z14(2428+2*j-1:2
428+2*j,3),'b.','MarkerSize',30)

plot3(Z14(2436+6*j-5:2436+6*j,1),Z14(2436+6*j-5:2436+6*j,2),Z14(2436+6*j-5:2
436+6*j,3),'b*','MarkerSize',30)
    title("within $L_{"+id(j)+"}$ after cycle three", 'Interpreter',
'latex')
    legend('original inputs','synthetic centroids','cycle one
winners','cycle two winners','cycle three winners','Location','NW')
    xlabel('PC 1(' + string(explained(1))+"% variance expl)")
    ylabel('PC 2(' + string(explained(2))+"% variance expl)")
    zlabel('PC 3(' + string(explained(3))+"% variance expl)")
    xh = get(gca,'XLabel'); % Handle of the x label
    set(xh, 'Units', 'Normalized')
    pos = get(xh, 'Position');
    set(xh, 'Position',pos.*[1,-0.05,1],'Rotation',15)
    yh = get(gca,'YLabel'); % Handle of the y label
    set(yh, 'Units', 'Normalized')
```

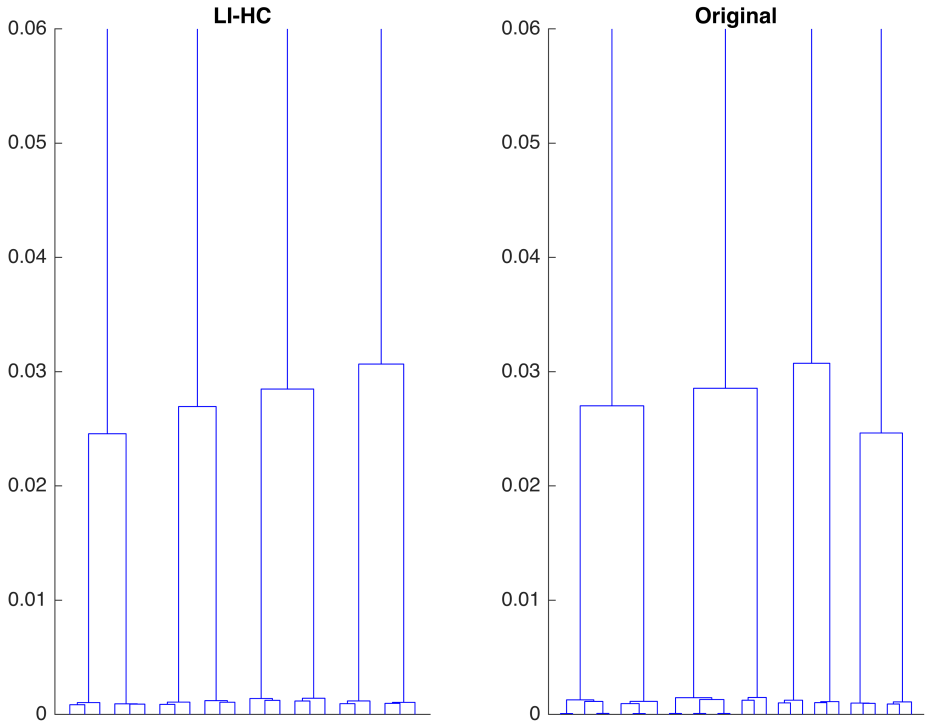
```
    pos = get(yh, 'Position');
    set(yh, 'Position',pos.*[1,-0.07,1],'Rotation',-25)
    grid on
    hold off
end
```




```
j = 4;
figure;
hold on
view(3)
plot3(Z14(data_idx,1),Z14(data_idx,2),Z14(data_idx,3),'r.','MarkerSize',10)
plot3(Z14(2424+j,1),Z14(2424+j,2),Z14(2424+j,3),'bp','MarkerFaceColor','blue
','MarkerSize',40)
plot3(Z14(2428+2*j-1:2428+2*j,1),Z14(2428+2*j-1:2428+2*j,2),Z14(2428+2*j-1:2
428+2*j,3),'b.','MarkerFaceColor','blue','MarkerSize',50)
plot3(Z14(2436+6*j-5:2436+6*j,1), Z14(2436+6*j-5:2436+6*j,2),
Z14(2436+6*j-5:2436+6*j,3), 'bs', 'MarkerSize', 25, 'LineWidth', 3)
title("sample results of LI-HC")
set(gca,'XTick',[], 'YTick', [])
hold off
```




```
tree3 = linkage(cycle3_winner_demask','average','cosine');
figure;
subplot(1,2,1)
dendrogram(tree3)
title("LI-HC")
set(gca,'XTick',[])
ylim([0 0.06])
subplot(1,2,2)
dendrogram(tree1)
title("Original")
set(gca,'XTick',[])
ylim([0 0.06])
```


LI-HC
Original
0.06
0.05
0.04
0.03
0.02
0.01
0