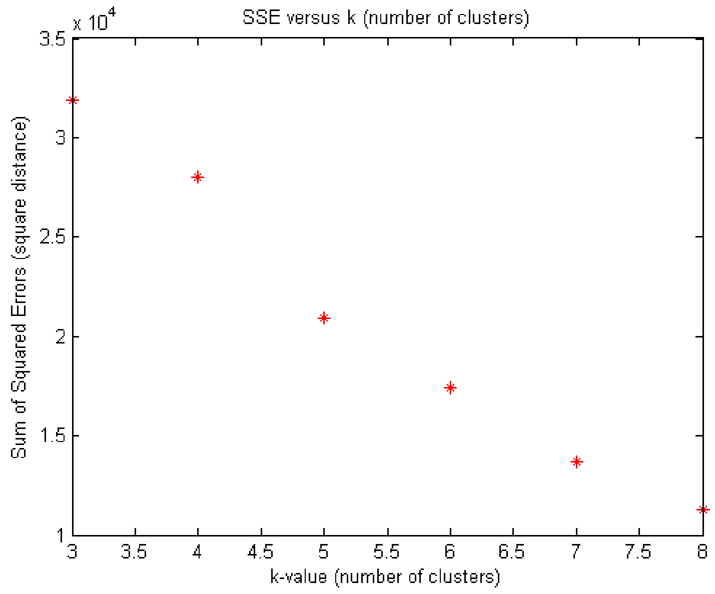
Garrett Scholtes

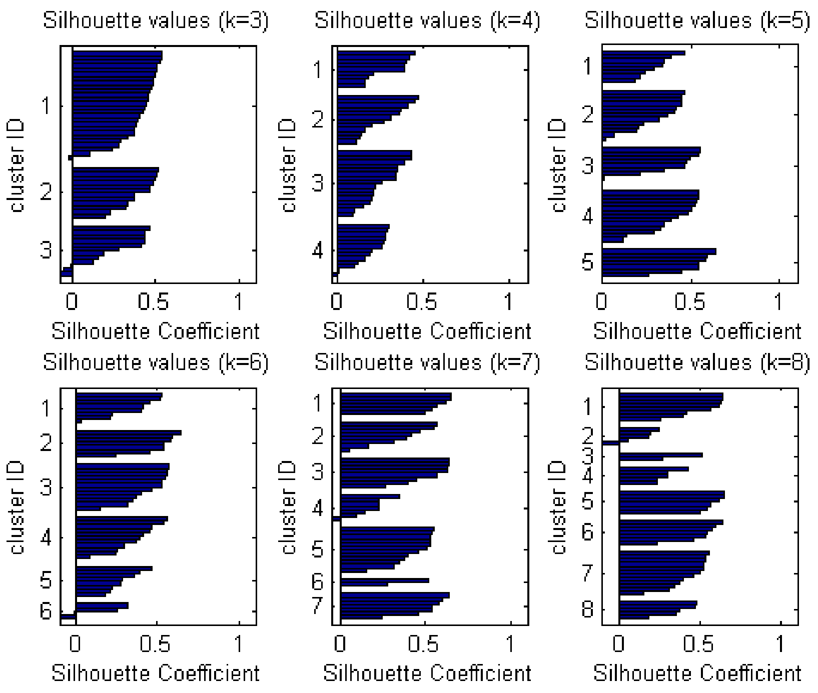
Intelligent Data Analysis
Homework #3
Due Date: Nov 14th, 2016, 7PM

Consider the data file attached with this homework. It contains scores for fifty students in four different subjects (Physics, Maths, English, and Music). Perform the following tasks with this data set.

1. Perform k-means clustering with this dataset for values of k to be 3, 4, 5, 6, 7, and 8. For each case of k run the clustering algorithm with three different initial cluster centers and select the one with the lowest SSE value. Plot the SSE against the values of k. Report the following in the submitted work: (Use Matlab kmeans function or any other similar toolbox)
   a. A plot of the SSE values against the values of k.

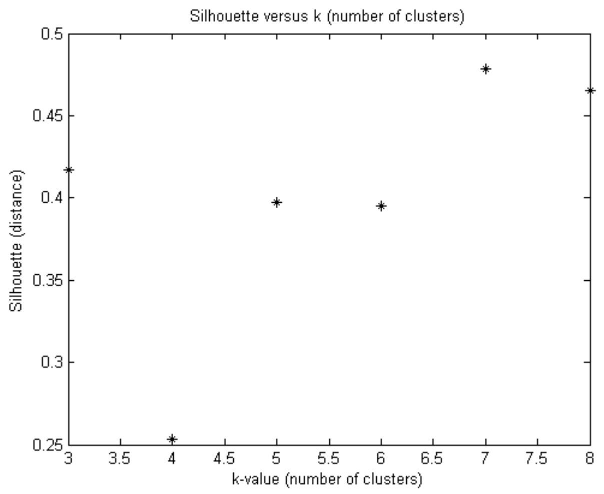


   b. A plot of the silhouette coefficients for the data points in each clustering. (Each value of k results in one clustering)



c. What is the best number of clusters for this dataset? Justify your choice for the best number of clusters.

I would choose k=5 as the best clustering



For k = 6, 7, and 8, there are at least a few clusters which only contain a few data points and should be considered anomalous.

K=1 does not look so bad, but it does have some data points which are closer to other clusters than to their own (see coefficients which are less than zero).

To determine whether k=4 or k=5 is a better clustering, let us also look at a total median silhouette for each clustering. Notice that k=4 has a much lower median silhouette than k=5.

d. For your choice of the best number of clusters report the centroids of all the clusters (Call this as Clustering-1).

Every row is a centroid (Physics, Math, English, Music)

```
56.1250   90.5000   56.2500   80.8750
64.4615   54.7692   82.4615   70.9231
49.8889   37.2222   44.7778   64.3333
93.3077   90.5385   79.0769   76.2308
90.2857   94.1429   53.5714   58.8571
```

e. Generate 50 random 4-dimensional random data points such that each attribute can take values between 0 and 100. With this dataset form the same number of clusters as selected by you in (c) above. Report the centroids and populations of the clusters. Compare the SSE for this dataset with the SSE for the provided dataset. Comment on the differences between the two values.

Centroids:

```
70.7720   46.5613   16.0088   88.4757
36.5051   58.0593   85.9093   58.5997
80.2180   27.0793   69.6453   49.8465
 9.8650   41.8185   41.7200   32.8028
55.7743   74.2697   28.4587   31.0965
```

Population of clusters 1, 2, 3, 4, and 5 (respectively) for random data:

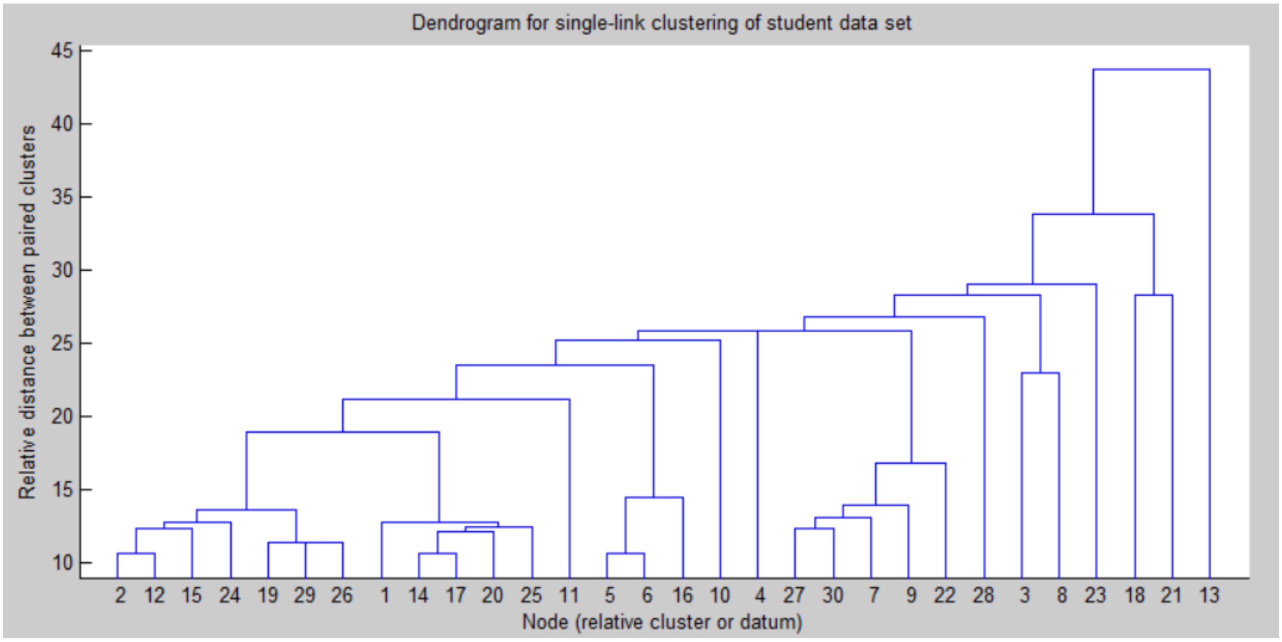
```
8   11   12   9   10
```

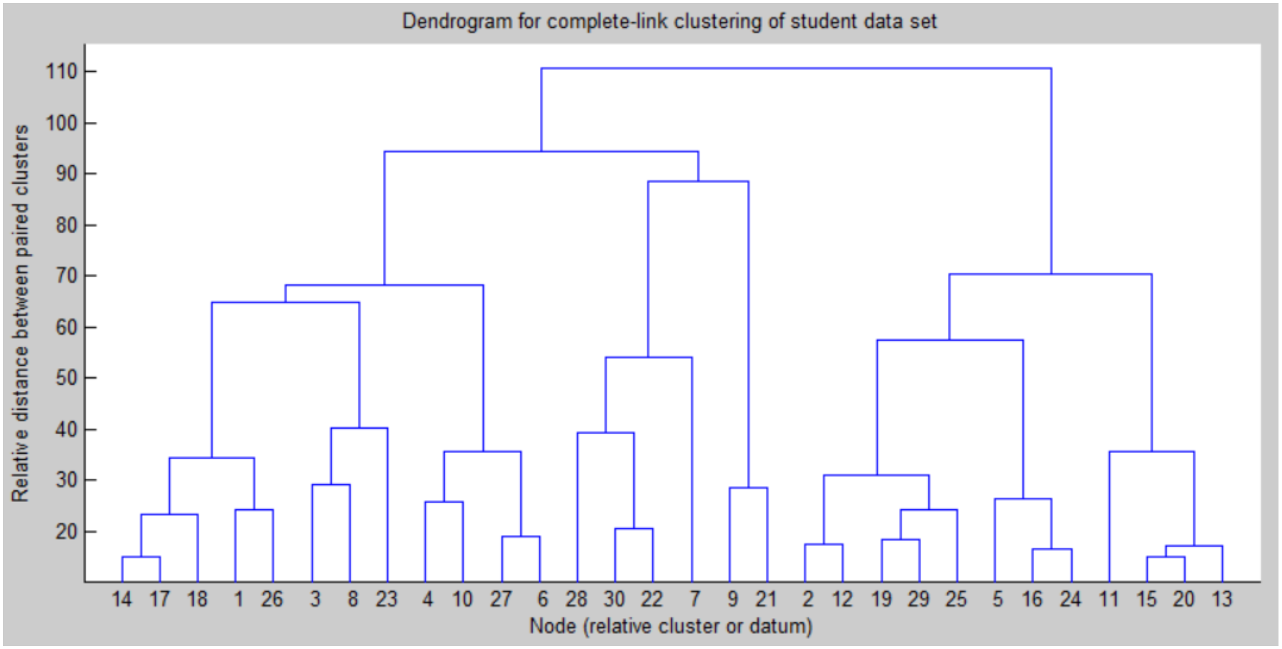
SSE for random data = 7.1622 e4
SSE for real data = 2.0927 e4

It is not surprising that the SSE for real data is lower than the SSE for random data. Because both the random data and real data fall within similar ranges and contain the same number of data, we can conclude from the SSE's that the random data is less clustered (i.e., more evenly spread out) than the real data. Therefore, we can expect that the real data exhibits more clustering and does not follow a uniform random distribution.

2. Perform hierarchical clustering for the students' scores dataset. Generate and show dendrograms for the cases (i) Single-Linkage clustering (Clustering-2), and (ii) Complete-Linkage clustering (Clustering-3). Use Euclidean distance for computing distance between data points. Report the following in the submitted work: (Use Matlab functions pdist and linkage, or any other similar toolbox.)

a. Dendrograms for the two clusterings (Clustering-2 and Clustering-3)





b. Cluster compositions for each case when we need only four clusters. Write the data points included in each cluster and compute their centroids.

Below are the number of data for each cluster (sorted by most populous clusters):

Single link:	46, 2, 1, 1
Complete link:	25, 18, 5, 2

Notice how clusters in the complete link clustering are more evenly distributed

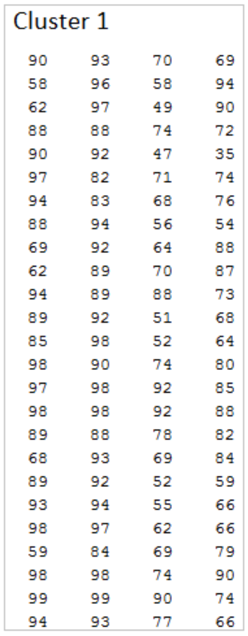
Centroids:

Single link:
```
58.5000   58.5000   58.5000   58.5000
73.7174   73.1087   68.0435   71.5217
35.5000   86.5000   35.5000   62.5000
54.7500   54.7500   54.7500   54.7500
```
Complete link:
```
35.5000   86.5000   35.5000   62.5000
55.0000   35.4000   37.4000   74.0000
60.5000   52.6667   76.0000   66.2778
85.8400   92.3600   68.0800   74.5200
```
Data points included in each cluster (per clustering):

Single Link clustering

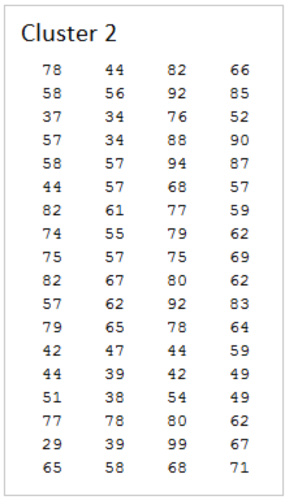
Cluster 1
```
78   44   82   66          57   62   92   83
90   93   70   69          98   90   74   80
58   56   92   85          79   65   78   64
37   34   76   52          42   47   44   59
58   96   58   94          58   43   57   89
62   97   49   90          97   98   92   85
88   88   74   72          43   37   39   64
57   34   88   90          98   98   92   88
58   57   94   87          89   88   78   82
44   57   68   57          44   39   42   49
90   92   47   35          51   38   54   49
97   82   71   74          68   93   69   84
94   83   68   76          47   32   36   69
82   61   77   59          89   92   52   59
88   94   56   54          93   94   55   66
69   92   64   88          98   97   62   66
74   55   79   62          77   78   80   62
62   89   70   87          37   33   27   79
94   89   88   73          59   84   69   79
89   92   51   68          98   98   74   90
85   98   52   64          65   58   68   71
75   57   75   69          99   99   90   74
82   67   80   62          94   93   77   66
```
Cluster 2
```
29   94   32   53
42   79   39   72
```
Cluster 3
```
29   39   99   67
```
Cluster 4
```
90   32   28   69
```

Complete Link



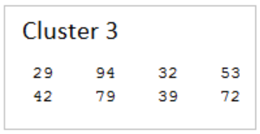
Cluster 1

90	93	70	69
58	96	58	94
62	97	49	90
88	88	74	72
90	92	47	35
97	82	71	74
94	83	68	76
88	94	56	54
69	92	64	88
62	89	70	87
94	89	88	73
89	92	51	68
85	98	52	64
98	90	74	80
97	98	92	85
98	98	92	88
89	88	78	82
68	93	69	84
89	92	52	59
93	94	55	66
98	97	62	66
59	84	69	79
98	98	74	90
99	99	90	74
94	93	77	66



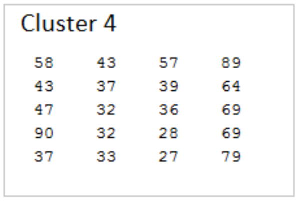
Cluster 2

78	44	82	66
58	56	92	85
37	34	76	52
57	34	88	90
58	57	94	87
44	57	68	57
82	61	77	59
74	55	79	62
75	57	75	69
82	67	80	62
57	62	92	83
79	65	78	64
42	47	44	59
44	39	42	49
51	38	54	49
77	78	80	62
29	39	99	67
65	58	68	71



Cluster 3

29	94	32	53
42	79	39	72

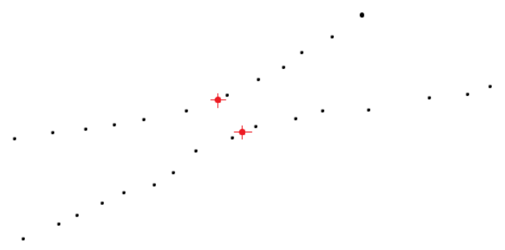


Cluster 4

58	43	57	89
43	37	39	64
47	32	36	69
90	32	28	69
37	33	27	79

c. Comment on any differences in the cluster centers and cluster compositions for the two different clusterings as performed in (b) above.

The first and last centroids in the single link clustering are very close together, whereas none of the centroids for the complete link clustering are as close.
This behavior can be explained by observing that the single-link measure can tend to produce long, "filament"-like clusters. For example, this data would likely produce two clusters via the single link method, with cluster centers that are closer together than average data within clusters themselves:



We can also notice that the single-link method produced a few clusters with only 1 data point each. This might indicate that either (1) the single link method with 4 clusters is not suitable for this dataset, or (2) the single link method *is* appropriate for this dataset, but there are some noise points. Looking more closely at the individual data seems to indicate that (1) is more likely.

d. Compute Rand Index for the comparison of Clustering-2 and Clustering-3 and show the counts a, b, c, and d as determined for computing the Rand index. Explain the meaning of each count and why such counts have been obtained for this dataset and these clusterings in this comparison.

The Rand Index is a measure which attempts to quantify similarity between clusterings.
It is computed as the number of data pairs which are placed in either the same or different cluster in both clusterings, divided by the total number of data pairs.

"a" is the number of pairs which contain data points that are in the same cluster in both clusterings
"b" is the number of pairs which contain data points that are in different clusters in both clusterings

"c" and "d" are, together, the number of pairs which contains data points that are in the same cluster in one clustering, but not in the other clustering (i.e., the number of pairs of points in which the cluster assignments disagree between the clusterings).

Rand Index = 0.4988

a =	443
b =	168
c =	593
d =	21

From the Rand Index, we can see that there is generally low agreement between these clusterings. This is also apparent in that "c" is the larger than both "a" and "b".

As for why "c" is larger than "d", this is likely due to the fact that the single-link clustering produced a few clusters with very few data, so there is a large asymmetry between these measures. Note that the *order* of "c" and "d" is not particularly meaningful by itself – had the clusterings been run through the Rand Index function in reverse order, the values of "" and "d" would be flipped.

3. Compute Rand Index for the comparison of Clustering-1 and Clustering-2 and show the counts a, b, c, and d as determined for computing the Rand index. Explain the meaning of each count and why such counts have been obtained for this dataset and these clusterings in this comparison.

Rand Index = 0.2988

a =	209
b =	157
c =	32
d =	827

See (2.d) for an explanation of the meanings of the above coefficients. The explanation for the above answer is similar here. The rand index is low because k-means and single-link do not produce similar clusterings for this problem.

As for why the rand index for k-means and single-link is so much lower than for single-link and complete link, there could be many reasons. One is that both single-link and complete-link hierarchical clustering tend to handle data of variable density better, whereas k-means does not. If the data is not equally dense, k-means would perform sub-optimal in this regard. Another is that, although complete-link and k-means both tend to produce hyperspheroidal clusters, a datum clustered via complete-link may be closer to the centroids of other clusters than to its own cluster. This tends to be a strength when it comes to clustering non-uniformly dense data.

For more completeness of analysis, let us also observe the Rand Index between k-means and complete link. In this case, it turns out that (for my random run of k-means):

Rand Index = 0.7461

The similarity of these two clusterings may be explained by both methods' tendency to produce hyphersphere clusters.

On the other hand, as for why the index is not higher (closer to 1), this could attributed to (1) the same density problem explained above, or (2) the fact that we created 5 clusters with k-means but only 4 with complete link for this example.

Code

hw3scholtes.m

```
% Garrett Scholtes
% 2016-11-09
% Homework #3
% Intelligent data analysis
%

% Close tree view windows
hiddenfigs = findall(0,'Type','figure', '-not', 'HandleVisibility', 'on');
close(hiddenfigs);
% Clear and close all
clear all;
close all;
% Seed the random number generator for convenience
% Can be removed if desired
rng(256);

% Load the data
raw_data = xlsread('StudentData2.xlsx');
% Ignore the ID column
data = raw_data(1:50, 2:5);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Part 1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Perform k-means for k = 3, 4, 5, 6, 7, and 8

% These figure coefficients are for silhouette plots
figure;

k_values = 3:8;
sse = [];
silhouettes = [];
clusterings = cell(0);
for k = k_values
    best_sse = Inf;
    for i = 1:3
        [sub_idx, sub_C, sub_sumd, sub_D] = kmeans(data, k);
        % Note:
        % The default k-means algorithm uses squared euclidean distance
        % to find clusters.  This method produces the same clusters as
        % euclidean distance, so the "sum of errors" produced by MATLAB's
        % kmeans function is just a "sum of *squared* errors" for a
        % Euclidean distance.
        % Therefore this works
        sub_sse = sum(sub_sumd);
        if sub_sse < best_sse
            best_sse = sub_sse;
               idx = sub_idx;
            C = sub_C;
            sumd = sub_sumd;
            D = sub_D;
        end
    end
    % Sum of squared errors (one for each k)
    sse(end+1) = best_sse;
    % Silhouettes
    silhouettes(end+1) = median(silhouette(data, idx, 'Euclidean'));
    % Plot the silhouettes
    subplot(2,3,k-2);
    silhouette(data, idx, 'Euclidean');
    title(sprintf('Silhouette values (k=%d)', k));
    ylabel('cluster ID');
    xlabel('Silhouette Coefficient');
    % Save clusterings for later use
    clusterings(end+1) = {struct('idx',idx,'C',C,'sumd',sumd,'D',D)};
end

% 1.a) Plot SSE versus k
figure;
plot(k_values, sse, 'r*');
title('SSE versus k (number of clusters)');
xlabel('k-value (number of clusters)');
ylabel('Sum of Squared Errors (square distance)');

% Plot *total* Silhouette versus k
figure;
plot(k_values, silhouettes, 'k*');
title('Silhouette versus k (number of clusters)');
xlabel('k-value (number of clusters)');
ylabel('Silhouette (distance)');

% "Best" clustering choice
% We will use silhouette/sse as a metric we wish to maximize
% best_metric = silhouettes./sse;
% [~, best_idx] = max(best_metric);
% It appears that k=5 is the best for this set
best_idx = 3;

% 1.c) Best k-value %
best_k = k_values(best_idx);
% 1.d) Best clustering %
% Note: best_clustering.C is the centroids of the "best" clustering
best_clustering = clusterings{best_idx};

% 1.e) Perform with random data
rand_data = 100*rand(50,4);
[rand_idx, rand_C, rand_sumd, rand_D] = kmeans(rand_data, best_k);
rand_sse = sum(rand_sumd);
rand_cluster_counts = histc(rand_idx, unique(rand_idx));

% Naming this variable explicitly will be useful for part 3
clustering1 = best_clustering.idx;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Part 2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Hierarchical clustering, with single-linkage and complete-linkage
```

```
pairdists = pdist(data);
single_link = linkage(pairdists, 'single');
complete_link = linkage(pairdists, 'complete');

% 2.a) Report dendrograms
figure;
dendrogram(single_link);
title('Dendrogram for single-link clustering of student data set');
xlabel('Node (relative cluster or datum)');
ylabel('Relative distance between paired clusters');
figure;
dendrogram(complete_link);
title('Dendrogram for complete-link clustering of student data set');
xlabel('Node (relative cluster or datum)');
ylabel('Relative distance between paired clusters');

% 2.b) 4 clusters only (for each case)
% 2.c) Also part 2.c

% Cluster the data for each linkage, but with a cutoff of 4 clusters
clustering2 = cluster(single_link, 'maxclust', 4);
clustering3 = cluster(complete_link, 'maxclust', 4);

% Data and centroids for each
% Just manually computing these.  Sloppy maybe, but works
clustering2_1 = data(clustering2==1,:);
clustering2_2 = data(clustering2==2,:);
clustering2_3 = data(clustering2==3,:);
clustering2_4 = data(clustering2==4,:);
c2centroids = zeros(4);
c2centroids(1, :) = mean(clustering2_1);
c2centroids(2, :) = mean(clustering2_2);
c2centroids(3, :) = mean(clustering2_3);
c2centroids(4, :) = mean(clustering2_4);

clustering3_1 = data(clustering3==1,:);
clustering3_2 = data(clustering3==2,:);
clustering3_3 = data(clustering3==3,:);
clustering3_4 = data(clustering3==4,:);
c3centroids = zeros(4);
c3centroids(1, :) = mean(clustering3_1);
c3centroids(2, :) = mean(clustering3_2);
c3centroids(3, :) = mean(clustering3_3);
c3centroids(4, :) = mean(clustering3_4);

% 2.d) Rand index for single and complete link
[r23, a23, b23, c23, d23] = randidx(clustering2, clustering3);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Part 3
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% 3) Rand index for k-means and single-link
[r12, a12, b12, c12, d12] = randidx(clustering1, clustering2);
```

```
% bonus) Randidx for k-means and complete link
% Doing this part just for a more complete analysis
[r13, a13, b13, c13, d13] = randidx(clustering1, clustering3);
```

Published with MATLAB® R2013a

randidx.m

```
% Garrett Scholtes
% 2016-11-12

% Note: a soft way to validate this function is to
% check that a+b+c+d = n(n-1)/2

function [ ridx, a, b, c, d ] = randidx( c1, c2 )
%randidx - computes Rand Index of two partitionings c1 and c2
%   Inputs: c1 and c2 are n-by-1 matricies, each corresponding to
%           a clustering of the same dataset, containing n data.
%           The i-th element of c1 is which cluster element i belongs
%           to in clustering 1.  Similar for c2
%   Outputs: a, b, c, and d are the standard Rand index coefficients.
%            ridx is the standard rand index computed from a, b, c, and d.

a = 0; % same c1 same c2
b = 0; % different c1 different c2
c = 0; % same c1 different c2
d = 0; % different c1 same c2

total = size(c1, 1);

for i = 1:total
    for j = (i+1):total
        same1 = c1(i) == c1(j);
        same2 = c2(i) == c2(j);
        if same1 && same2
            a = a+1;
        elseif (~same1) && (~same2)
            b = b+1;
        elseif same1 && (~same2)
            c = c+1;
        else
            d = d+1;
        end
    end
end

ridx = (a + b) / (a + b + c + d);

end
```

Published with MATLAB® R2013a