Homework 2

1. We want to carry out clustering on this data. Consider this matrix, where each row is one data point:

$$X = \begin{bmatrix} 1 & 1 & 1 \\ 1.5 & 2 & 0 \\ -0.5 & 0 & -3 \\ 0 & -0.5 & -1 \end{bmatrix}.$$

   Carry out K-means on the data. (If you wish, you can use a computer to determine the distances (and the indicator vectors) but show the other computations by hand.)

   (a) Starting with R-vector

$$R = \begin{bmatrix} 3 & 3 & 3 \\ -1 & -1 & 0 \end{bmatrix}$$

   (b) Starting with R-vector

$$R = \begin{bmatrix} -0.5 & 0 & -3 \\ -1 & -1 & 1 \end{bmatrix}$$

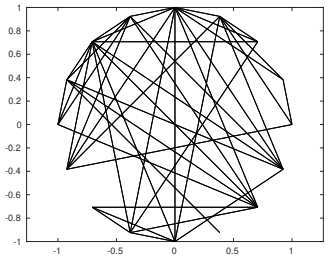
   Interpret similarities and differences between the solution in (a) and (b). Which solution is "best"?

A short help on constructing and plotting similarity graphs in Matlab is available in Canvas.

2. We will now build a graph based on the following data:

```
n0=5; p=3;
randn('seed',0);
c=2;
A1=randn(n0,p)+c*[1,0,0];
A2=randn(n0,p)+c*[0,1,0];
A3=randn(n0,p)+c*[0,0,1];
A4=[0,0,0];
A=[A1;A2;A3;A4];
```

Build a similarity graph using: ε-neighborhood for the standard Euclidean norm with $\varepsilon = 2.5$. Visualize it with circular node placement or matlabs node placement (as on the similarity graphs help page in canvas). The circular placement should lead to a figure like the one to the right.



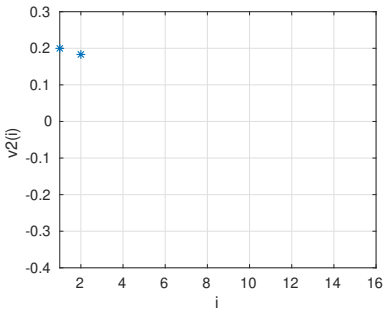
(a) From the figure, determine how many connected components does the graph have.

(b) Construct the unnormalized Laplacian. Run `eig(L)`. Relate the result in (a) with Lemma 2.3.3 in EJ2 (or Proposition 2 in UvL).

(c) Change the c-variable: $c = 3$. Provide a plot of the graph and repeat (a)-(b) for the new graph.

(d) Set $c = 3.1$ and $\varepsilon = 2.8$. Plot the graph, and plot the second eigenvector. Provide the second eigenvector in a plot like below. How can a separation of the graph (clustering) be read out from the graph?

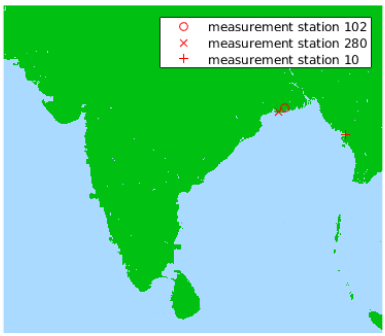
The figure provides the first two values. You may get slightly different values depending on MATLAB version.

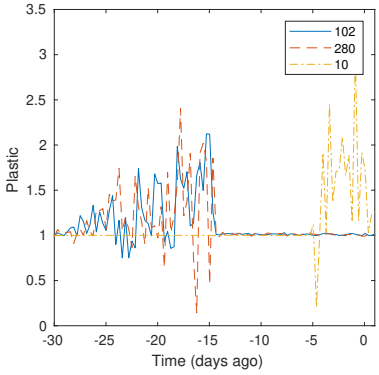


3. **Bengali cleanup simulation.** The Bengali bay countries want to carry out joint efforts against marine pollution, first by collecting information about plastic. They collect data in areas close to the shore for one month, in 937 locations. The goal is to establish seven regions where the plastic pollution is similar in order to coordinate efforts (such that the number of sampling points can be reduced and monitor the pollution easier over time).

This exercise is inspired by marine pollution investigations on the Canary islands: *Microplastic and tar pollution on three Canary Islands beaches: An annual study*, Marine Pollution Bulletin, April 2018, Pages 494-502 https://www.sciencedirect.com/science/article/pii/S0025326X1730838X

The time-series for three locations are visualized below. From the time-series data we see that the plastic pollution characteristics close to Kolkata are rather similar to each other, in comparison to the given location in Burma.





The data is stored in `bengali_cleanup.mat` and the map over the area is stored in `bengali_map.png`. Variables in the mat-file:

- `x_coords` and `y_coords` map coordinates for the 937 locations.
- `timeseries`: A matrix where every row is the data from one measurement station, the corresponding time-points are given in `tv` (same for all stations).

The plot above can be obtained from:

```
load('bengali_cleanup.mat');
A=imread('bengali_map.png');
figure(1); clf;
imshow(A); hold on;
jv=[102,280,10];
plot(y_coords(jv),x_coords(jv),'r*') % Note: x and y reversed since
figure(2);                           % images have swapped x and y axis.
plot(tv,timeseries(jv(1),:),'-'); hold on
plot(tv,timeseries(jv(2),:),'--')
plot(tv,timeseries(jv(3),:),'-.')
```

Only the time-series (not coordinates) will be used for the clustering. We will do sanity check with the coordinate vectors.

(a) Familiarize yourself with the data: Load the data and provide a map with all measurement stations.

(b) Construct a distance matrix from the time-series. Define the distance between two nodes (in the naive way) by computing the two-norm of the difference between the two time-series vectors. What is the time series distances between 102 and 280 and 10? Does it make sense?

(c) Construct a weight matrix by using kNN (version: or) with $k = 3$. Provide the graph (using MATLABs placement of nodes). Can a human identify the (seven) clusters?

(d) Carry out spectral clustering with seven clusters (RatioCUT version). You may use the matlab built-in function `kmeans` for the final step. Answer with seven maps, one for each cluster. Plot the clusters in the map.

(e) Change the k in (c) to $k = 2$. Does the general conclusions change? Plot the clusters in one or several maps.

(f) (optional) Find hidden conclusions in the data. For instance, inland Sri Lanka seems to have similarities with some other part of Bengali bay, which one?

4. Download the `zalando_clustering.mat` file from Canvas. Every column in the variable `items` correspond to an image, and `correct` is a vector containing the correct classification of each image. Although we know the correct item type in the variable `correct` we will design an algorithm not using that information, but only to determine how good the method works.

In machine learning terminology, the classification setup in problem 4 would be called unsupervised learning. It is unsupervised in the sense that we do not know the item type of images at all (except in the final stage when we check how good the method works). This is in contrast to supervised learning where you would use the item type information for one data set to determine item types for another data set.

(a) The items contains two types of item. Which ones? Use the `zalando_plot` function.

Build a graph as follows: Let $D \in \mathbb{R}^{n\times n}$ where $n = 1000$ be the distance matrix of the images with the Euclidean distance with weighting:

$$D_{i,j} = \|w. * (items(:,i) - items(:,j))\|_2$$

where $w \in \mathbb{R}^{784}$ is a weight vector. Implement the distance function like this

```
function D=build_distance(items,w)
# items: a matrix where every column is an image of length 784
# w a weight vector (784 x 1) for the weighted Euclidean product

   n=size(items,2);
   D=zeros(n,n);
   for i=...
   ...
end
```

In the initial simulations we use the unweighted Euclidean product, so w will consist of ones.

(b) Based on the D-matrix compute and plot four images:
(1) item(:,1), (2) the image closest to item(:,1), (3) item(:,2), (4) the image closest to item(:,2).

Build a weight matrix from the distance matrix as follows.

```
alpha=0.5;
for i=1:n
    sgma=std(D(:,i));
    for j=1:n
        W(i,j)=exp(-alpha*D(i,j)^2/sgma^2);
    end
end
W=W-diag(diag(W));
W=(W+W')/2; % We want an undirected graph
```

Carry out spectral clustering for two clusters. For this problem, you do not have to run k-means as post-processing, but can instead determine the cluster based on the values of the second eigenvector of the unnormalized Laplacian (using the thresholding).

(c) Plot the second eigenvector `plot(v2/norm(v2),'*');`. Spectral clustering is expected to work well when we have many items close to two discrete values. Does this happen in this case? Adjust the y-axis so you do not get distracted by outliers.

(d) Classify an image an image i to cluster 1 if `v2(i)<tau` and cluster 2 otherwise. You may now use that there the a priori knowledge that there are 500 images of each item, leading to the suggestion that τ should be the median of the `v2`-values. (Change to `v2(i)>tau` if you get more than 50% incorrect.) What percentage of the images are labeled correctly? Use the correct solution `correct` to compute the number of correctly classified.

Note that you can never get less than 50% correct, because you can always switch the sign in the eigenvector, or equivalently switch 1 and 2 in the `correct` vector.

(e) Now we want to investigate the incorrectly classified cases. Take four incorrectly classified images and plot them in first row in a subplot. In a second row, do a search for each image in the first row and plot the image that is the most similar but of different class.

Skeleton code in matlab: `hw2_plot_bad_skeleton.m` under files.

```
plotonly=4;
%% II is an index vector with indices of incorrectly classified images
for j=1:plotonly
    subplot(2,plotonly,j);
    zalando_plot(items(:,II(j)))
    correctlabel=correct(II(j));
    classifiedlabel=classification(II(j));
    title([num2str(II(j)),': correct=',...
    num2str(correctlabel),' classified=',num2str(classifiedlabel) ]);
end
for j=1:plotonly
    i=II(j);
    % ** Add code here to compute k: the image closest to item(:,i)
    %    but correct(i) ~= correct(k)
    dist=norm(item(:,i)-item(:,j));
    subplot(2,plotonly,j+4);
    zalando_plot(items(:,j));
    title([num2str(j),': dist from ',num2str(i),':',num2str(dist)]);
end
```

(f) The technique can be improved by using a weighted norm. Use the following weight:

```
x=2;
W_weight=ones(28,28); W_weight(12:16,:)=x;
w_weight=W_weight(:);
w_weight=w_weight/norm(w_weight);
```

Plot the weight using `zalando_plot(w_weight)`. What part of the image is this? Why can this be expected to improve the classification? (You may want to relate to the images in (d).) Run your clustering again with this weight. How much better is it?
Optional: Can you get a better classification by changing x or something else in the weighting?

5. Conceptual questions related to the asynchronous learning activities. Summarize in your own words (2-5 sentences for each activity):

   - Video quiz 6: Eigenvalue multiplicity
   - Video quiz 7a: Mincut
   - Video quiz 7b: Rayleigh-Ritz theorem
   - Video quiz 8: Eigenvalue derivatives
   - Video quiz 9: Proof of the Rayleigh-Ritz theorem