Unsupervised Machine Learning Projects
Project 3
Clustering and Nonlinear projection methods

Deadline: May 22nd 2015 (midnight the latest).

Exercise 1: Clustering

1. Reproduce Figure 12.5 in the lecture notes by drawing 2000 samples from a Gaussian mixture with two clusters (see Eq. 12.9 and section 12.4).
   In the report: Explain the construction in your own words. Make a scatter plot of the data. Specify the parameters employed. Explain their relation to the shape of the data using your own words, equations, and scatter plots proving your claims.
   Hints:
   Recall from the first exercise that a covariance matrix $\mathbf{C}$ can be written as $\mathbf{C} = \mathbf{V}\,\mathbf{D}\,\mathbf{D}\,\mathbf{V}^{\mathrm{T}}$, being $\mathbf{V}$ the matrix with principal directions as columns and $\mathbf{D}$ a diagonal matrix with the standard deviations along the principal directions.

2. Implement the Expectation-Maximization (EM) algorithm (sections 12.5 and 12.6) for data of arbitrary dimensionality and number of clusters. The algorithm must return the estimated cluster probabilities π_c, mean values μ_c, and covariance matrices C_c (Eqs. 12.17–19), as well as the values of the objective function during the optimization (Eq. 12.21).
   In the report: Explain your implementation and its relation to the underlying theory in your own words. Test the algorithm on the data generated before, make a figure with the objective function, and compare the estimates of the parameters with their true values. Restart the algorithm with different initial points and assess the sensitivity of your estimates.
   Hints:
   (i) The algorithm can "blow up" if it puts a Gaussian onto a single data point. In that case, restart the algorithm from different initial point.
   (ii) To avoid triggering errors, Use $u\ \log(u+\delta)$, with $\delta = 10^{-20}$ or less, instead of $u\,log(u)$ (see Eq. 12.20).

3. Using the estimated parameters, compute the posterior probability that a data point belongs to each cluster ($p(r(t)|\mathbf{x}(t))$ in Eq. 12.14) and assign each data point $\mathbf{x}(t)$ to the cluster $\hat{r}(t)$ that maximizes it (this operation is known as MAP estimate).
   In the report: Explain what you did in your own words. Make a scatter plot where the color of each data point indicates the cluster it belongs to according to the MAP estimate.

4. Implement the K-means algorithm (section 12.2) for data of arbitrary dimensionality and number of clusters. The algorithm must return the centre points $\mathbf{w}_c$ of each cluster c (Eq. 12.2), the closest centre point to each data point, and the value of the objective function during the optimization (Eq. 12.3).
   In the report: Explain your implementation and its relation to the underlying theory in your own words. Test the algorithm on the data generated before, make a figure with the objective function, and compare the estimates of the parameters with their true values and with the estimates of the

EM algorithm. Restart the algorithm with different initial points and assess the sensitivity of your estimates.

Hints:

(i) The code for computing Gaussian mixtures using Expectation Maximization can be reused for K-means as well.

(ii) Initialize the centre points so that they are preferably inside the data cloud. Otherwise, they may not be assigned to any data point, thereby becoming "dead units".

(iii) Should dead units occur, reassign them randomly somewhere inside the data cloud while still running the algorithm, thereby ensuring that you obtain the number of clusters you specified.

5. Create a mixture of 4 Gaussian distributions with the following means μ_c, covariance matrices C_c, and cluster probabilities π_c:

$$\mu_1 = [3, 0] \qquad \mu_2 = [-1, 4] \qquad \mu_3 = [-3, -3] \qquad \mu_4 = [0, 0] \tag{1}$$

$$C_1 = \begin{bmatrix} 0.09 & 0 \\ 0 & 3 \end{bmatrix} \qquad C_2 = \begin{bmatrix} 5 & -2 \\ -2 & 1 \end{bmatrix} \qquad C_3 = \begin{bmatrix} 2 & 1.9 \\ 1.9 & 2 \end{bmatrix} \qquad C_4 = \begin{bmatrix} 4 & 0 \\ 0 & 0.25 \end{bmatrix} \tag{2}$$

$$\pi_1 = 0.1 \qquad \pi_2 = 0.2 \qquad \pi_3 = 0.3 \qquad \pi_4 = 0.4 \tag{3}$$

Run both the EM and the K-means algorithms from 5 different starting points. Set the cluster probabilities to be equal and repeat the estimation. Now set the covariances to be their average and repeat the analysis. **In the report:** Visualize the MAP estimates and the objective functions for both algorithm in all runs for the three cases. Compare the estimates between different algorithms and with the original values. Discuss how effective are the algorithms in each case.

Exercise 2: Multidimensional scaling

1. Generate two data sets $\mathbf{Y_A}$ and $\mathbf{Y_B}$ of 5000 samples each according to the following equations:

$$\mathbf{y_A} = \begin{bmatrix} \sqrt{2+2\,x_1}\ \cos(2\pi\ \sqrt{2+2\,x_1}) \\ \sqrt{2+2\,x_1}\ \sin(2\pi\ \sqrt{2+2\,x_1}) \\ 2\,x_2 \end{bmatrix} \qquad \mathbf{y_B} = \begin{bmatrix} (10+x_4)\ \cos(\pi x_3) \\ (10+x_4)\ \sin(\pi x_3) \end{bmatrix} \tag{4}$$

   where x_1, x_2 and x_3 are uniformly distributed in the interval $[-1, 1]$, x_4 follows a Gaussian distribution with zero mean and unit variance.
   In the report: Make scatter plots of the two data sets.

2. Implement an algorithm that calculates the squared Euclidean distance matrix (Eq. 13.3) and the kernel-PCA distance matrix (Eq. 13.8), taking as input the data set and the type of distance to calculate. **In the report:** Describe the algorithm and its relation to the underlying theory. Make a figure with four separate histograms showing the distribution of each distance in each data set. Compare and discuss the results.

3. Implement the multidimensional scaling (section 13.1) taking as inputs the distance matrix $\mathbf{D}$ and the number of projections N as follows:
   (i) Normalise $\mathbf{D}$ (Eq. 13.2) to obtain the double-centred distance matrix $\tilde{D}$.
   (ii) Compute the first N smallest eigenvalues λ_n and their eigenvectors $\mathbf{U_n}$ $(1 \leq n \leq N)$.
   (iii) Compute $\hat{\mathbf{X}} = {}^1\!/\!\sqrt{2}\ \mathbf{U}\,(-\lambda)^{1/2}$, being $\mathbf{U}$ a matrix with eigenvectors in its columns and λ a diagonal matrix sorted in the same order as the eigenvectors.
   (iv) The output of the algorithm is $\hat{\mathbf{X}}$, constituting the projections of the data set.

   Perform Euclidean MDS and kernel-PCA by feeding the algorithm with the appropriate distance matrix to project the two data sets onto the one-dimensional feature that best represents x_1 or x_3. Use the distance histograms to aid the choice of the width of the Gaussian kernel.
   In the report: Describe the algorithm and its relation to the underlying theory. Make a figure with four separate panels showing the data sets in their original spaces, each point with a colour given by its projection onto a one dimensional space (top panels in Figure 1 below). Compare the projections with the values of x_1 and x_3.

4. Implement Laplacian eigenmaps (section 13.2.2) taking as inputs the kernel-PCA distance matrix $\mathbf{D^{kPCA}}$ and the number of projections N as follows:
   (i) Construct the diagonal matrix $\mathbf{\Delta}$ with elements given by $\mathbf{\Delta}_{ii} = \left(-\sum_{j=1}^{M} \mathbf{D^{kPCA}}_{ij}\right)^{-1/2}$ (M being the number of samples).
   (ii) Compute $\mathbf{D^{LE}} = \mathbf{\Delta}\,\mathbf{D^{kPCA}}\,\mathbf{\Delta} + \mathbb{1}$ where $\mathbb{1}$ represents the identity matrix.
   (iii) Compute the first $N+1$ smallest eigenvalues of $\mathbf{D^{LE}}$ and their corresponding eigenvectors $\mathbf{U}$.
   (iv) Discard the smallest eigenvalue, and its eigenvector, as it is trivially zero.
   (v) Estimate the projection of the data as $\hat{\mathbf{X}} = \mathbf{\Delta}\,\mathbf{U}$.

   Apply the algorithm to the two data sets analogously to the previous task. Use the distance histograms to aid the choice of the width of the Gaussian kernel.
   In the report: Make a figure with two separate panels showing the data sets in their original spaces, each point with a colour given by its projection onto a one dimensional space (top panels

in Figure 1 below). Compare the projections with the values of x_1 and x_3 and with the previous results using Euclidean MDS and kernel-PCA.

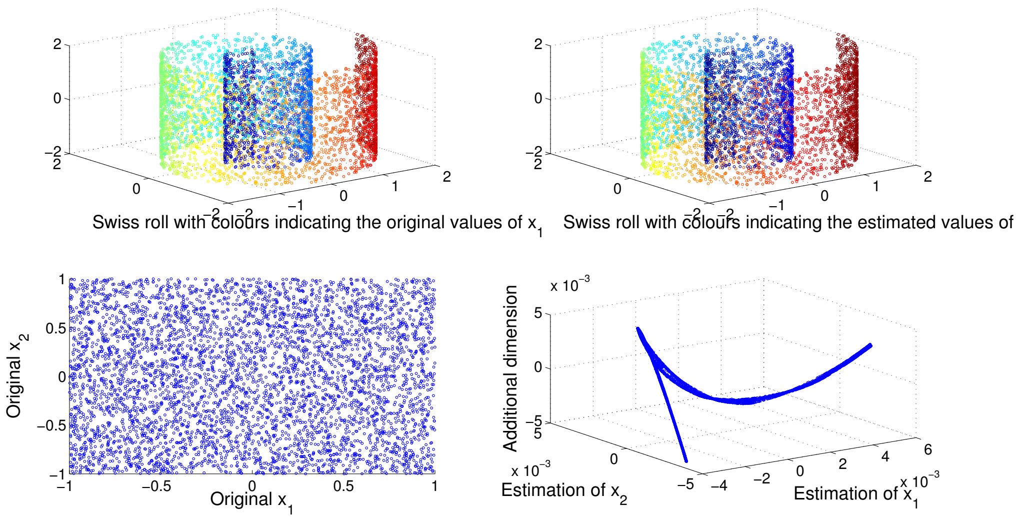

Figure 1: Application of Laplacian eigenmaps to $\mathbf{y_A}$. Original data set with colours indicating the value of x_1 (top left) and indicating the value of the estimation of x_1 obtained using Laplacian eigenmaps (top right). The estimation represents an approximately scaled version of x_1, but does not completely capture the intrinsic dimension of the underlying manifold. The bottom left panel shows the 2D space in which the underlying manifold is defined, whereas the bottom right panel shows that Laplacian eigenmaps actually projects the data into a 3-dimensional space.

5. Load the data from the file `flightnet.txt`, each row representing a different flight connection. The data contains 8 columns: the first four corresponding to the city of departure and the last four to the city of arrival. In each group, the first and second columns represent cities and their countries, with the third and forth columns being their longitudes and latitudes. Cities and countries are represented by different integer numbers. Construct a vector with the longitudes and latitudes of all cities in the data. Compute two great-circle distance matrices, one for all cities and another for cities connected by flights, using the Haversine formula given below:

$$d_{ij} = 12742 \arcsin\left(\sqrt{\sin^2\left(\frac{\phi_j - \phi_i}{2}\right) + \cos(\phi_i)\cos(\phi_j)\sin^2\left(\frac{\lambda_j - \lambda_i}{2}\right)}\right), \tag{5}$$

where λ and ϕ represent the longitude and latitude of the cities. Apply Euclidean MDS to both matrices and extract three main components.
In the report: Make three scatter plots giving different colors for cities in different countries. The first one should use the latitudes and longitudes as the coordinates of the cities. It should look just like Figure 2. The second and third should employ as coordinates the projections calculated before. Compare the plots, discussing the effect of city density and flight connections. Make scatter plots in which the data is projected into a 3D space, and discuss the result.

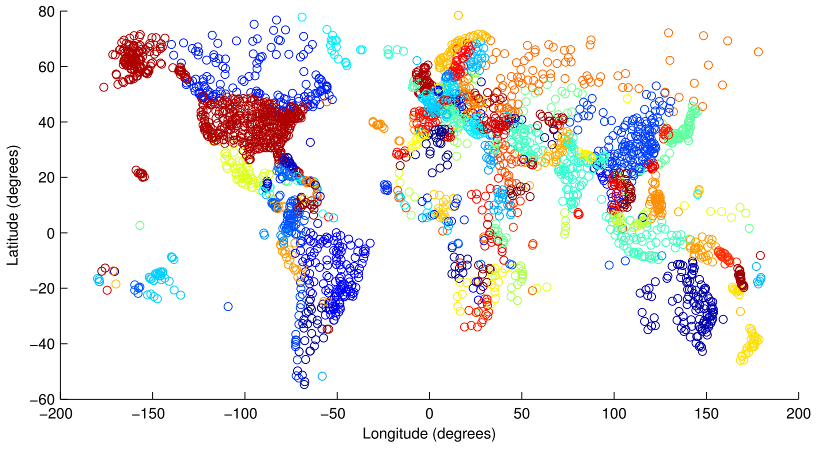

Figure 2: Fight connections data from 2010 - http://www.visualizing.org/datasets/global-flights-network.

Exercise 3: Isomap and Kohonen's self-organizing map

1. Implement an algorithm for computing the neighbours of each data sample. The algorithm must take as input the Euclidean distance matrix and the type of rule for computing the neighbours, described below:

   (i) K-rule: Each sample is a neighbour of the K-nearest data samples.

   (ii) ϵ-rule: Each sample is a neighbour of all other samples lying at a distance shorter than ϵ.

   (iii) Data-rule: For each sample $\mathbf{y_i}$, determine the set of neighbours $\mathbf{c_1}, \ldots, \mathbf{c_M}$ using one of the previous rules, and determine whether each pair of samples $\mathbf{c_u}$ and $\mathbf{c_v}$ fulfil the following three conditions:

   $$||\mathbf{y_i} - \mathbf{c_u}|| + ||\mathbf{y_i} - \mathbf{c_v}|| < C_1\, ||\mathbf{c_v} - \mathbf{c_u}|| \tag{6a}$$
   $$||\mathbf{y_i} - \mathbf{c_u}|| < C_2\, ||\mathbf{y_i} - \mathbf{c_v}|| \tag{6b}$$
   $$||\mathbf{y_i} - \mathbf{c_v}|| < C_2\, ||\mathbf{y_i} - \mathbf{c_u}|| \tag{6c}$$

   where $C_1 = \sqrt{S^2+1}$ and $C_2 = \dfrac{1+S}{1-S}$, being S a value chosen by you. The effect of S is shown in Figure 3. If so, those samples are neighbours.

   The algorithm must return a matrix in which only the distance between neighbours is preserved, usually called adjacency matrix. Make sure that samples are not connected with themselves, and that the matrix is symmetric. Test your implementation with the data set $\mathbf{y_A}$ generated in the previous exercise. To that end, preprocess the data using the K-means clustering algorithm developed in the first exercise to reduce the number of points from 5000 to 500.
   In the report: Make a figure with four panels showing the number of connections: in the first panel, as a function of the number of neighbours using the K-rule; in the second panel, as a function of the distance ϵ using the ϵ-rule (second panel); in the third panel, as a function of the number of neighbours for each value of S shown in Figure 3 using the Data-rule; and in the forth panel, as a function of ϵ for each value of S shown in Figure 3 using the Data-rule. Compare the results obtained with the different methods and with the radius of the spiral.

2. Implement the Isomap algorithm as follows:

   (i) Take the adjacency matrix as input and compute the geodesic distances, for example, using Dijkstra's algorithm, which you need not code yourself.

   (ii) Square the distances and plug them into the MDS algorithm developed in the previous exercise.

   (iii) The output is the embedding of the data set provided by Isomap.

   Test your implementation on both data sets created in the previous exercise. To that end, preprocess the data using the K-means clustering algorithm developed in the first exercise to reduce the number of points from 5000 to 500. Set up the parameters taking into account the typical distances between neighbours and points far away determined in the previous exercise.
   In the report: Make scatter plots of the results showing the data points with colours proportional to the estimations of x_1 and x_3, defined in the previous exercise. Compare the results with the one obtained in the previous exercise. Discuss the performance of the algorithm and its sensitivity to the parameters.

3. Implement the Kohonen's self-organizing map algorithm (SOM, section 13.3) and apply it to the preprocessed data. Define the neighbourhood of node i as node $i-1$ and node $i+1$. In the case

of the circle, make the nodes periodic, that is, make the last node a neighbour of the first. This neighbourhood relation establishes a kind of ordering of the model vectors (cluster means) $\mathbf{w_i}$.
In the report: Run the algorithm for various numbers of model vectors (about 10 to 100), various initializations and make figures similar to Figure 13.8 in the lecture notes. Increase the size of the neighbourhood to include the second and the third nearest neighbour, respectively.Compare the results. Discuss their performance in capturing the variation of x_1 and x_3. .

4. Load the data from the files `prion.txt` and `prioncn.txt`. The former contains the positions of all atoms of the prion protein and the latter only contains the positions of carbon and nitrogen atoms. The prion protein (Figure 4) is thought to be responsible for what are known as transmissible spongiform encephalopaties, of which the most well-known example is perhaps the mad-cow disease. The function of the proteins is not only given by the sequence of amino acids they are composed of (which is known as primary structure) but also by how nearby amino acids interact with each other, twisting the sequence (secondary structure) and folding it (tertiary structure, top right panel of Figure 4). The disease is caused by an appropriate unfolding of the protein which drives other proteins to fold in similar ways, and from there, to form aggregates of proteins. The files provided here contain data providing the positions of the atoms in the tertiary structure. Using the information provided in `prioncn.txt`, apply both Isomap and SOM to uncover the trace of the tertiary structure (top right panel in Figure 4). To that end, use either the ϵ-rule or the data-rule in Isomap and between 50 and 100 nodes in SOM, and search for the best set of parameters that could reproduce the bottom panel in Figure 4.
In the report: For Isomap, make a scatter plot with each point corresponding to the average of the positions of 10 consecutive atoms (you can do that by using a sliding window, for example, using `conv(atompositions,ones(10,1),'same')` in Matlab), and the colour given by the value of the first component extracted by Isomap. For SOM, plot the atoms with colours given by their position in the sequence, and the nodes connecting them with lines, with colours given by their position too. Apply both methods to the data in `prion.txt` using the optimal parameters found before (do not search for optimal parameters here). Compare your results with the figures provided. Did you manage to obtain a good match? Explain the possible reasons for your success or failure in terms of the neighbourhoods defined, the distances between neighbouring atoms, and the distance between atoms in different parts of the trace of the protein.

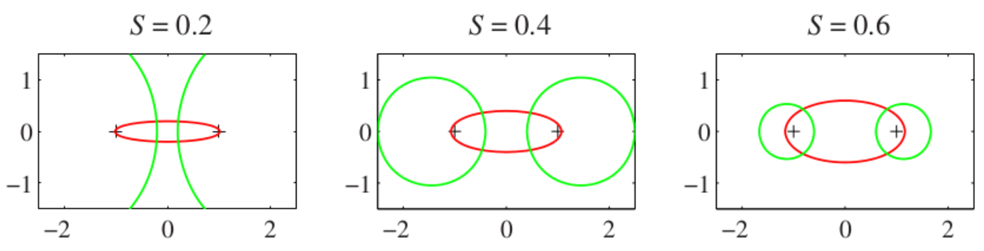

Fig. E.3. The two conditions to fulfill in order to create a graph edge between the vertices associated to two prototypes (black crosses). The points that create the edge must be inside the ellipse and outside both circles. The ellipse and circles are shown for different values of S.

Figure 3: Effect of the value of S on the shape of the neighbourhood using the data-rule. Extracted from the book Nonlinear Dimensionality Reduction, John A Lee and Michel Verleysen.

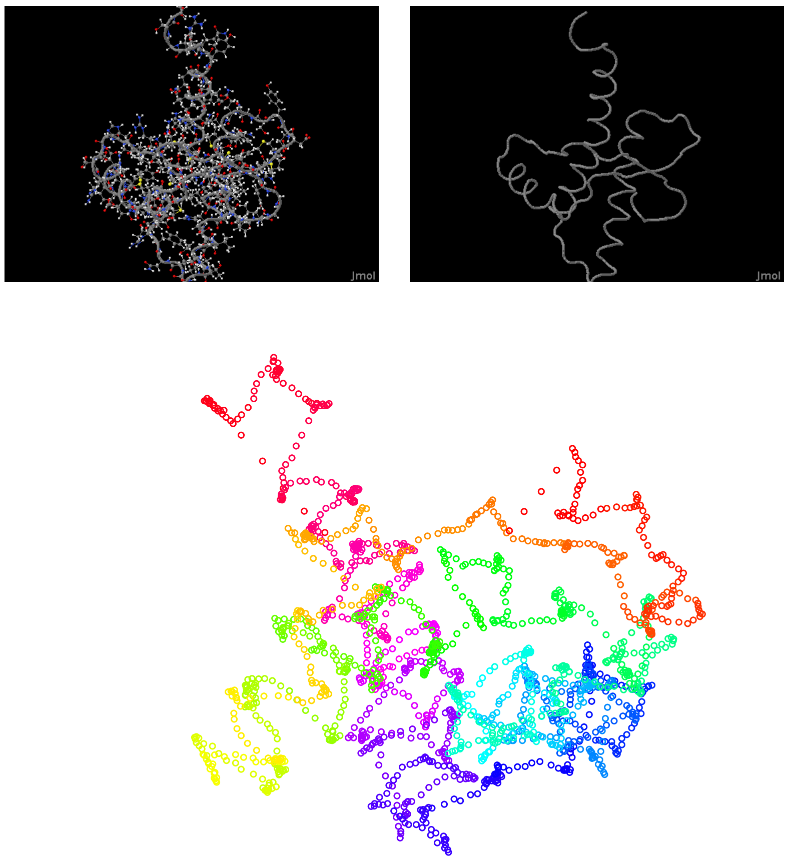

Figure 4: Description of the prion protein. (*top left*) Positions of atoms constituting the prion protein. (*top right*) Trace of the prion protein, which indicates the way the sequence of amino acids is twisted and folded. (*bottom*) Approximation of the trace of the prion protein using the data provided here and averaging the positions of the atoms with a sliding window of length 10. The colour map chosen was hsv.