Labs

Set 13 (DM857, DS830)

DM562	Scientific Programming
DM857	Introduction to Programming
DS830	Introduction to Programming

1 Clustering

Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group (*cluster*) are more similar (according to some similarity measure) to each other than to those in other groups. It is a main task of exploratory data analysis, and a common technique for statistical data analysis, used in many fields, including pattern recognition, image analysis, information retrieval, bioinformatics, data compression, computer graphics and machine learning.

Depending on the task, there are several clustering procedures. One of the most well-known is the the *k-Mean (Clustering) Algorithm*. This algorithm clusters data in a way that minimizes the sum of distances between entries in each cluster and the centroid (i.e., an entry in the cluster that acts as a surrogate centre) of the cluster. The number of clusters k is a parameter of the algorithm and is fixed beforehand. The algorithm consists of the following steps.

(a) Select k random entries of the dataset to be the centroids.

(b) Assign all entries to the cluster with the closest centroid.

(c) Check which point in each cluster is closest to the cluster centre (which might be different than the centroid).

(d) If at least one cluster changes its centroid, repeat from step (b), otherwise stops.

For this lab, you will implement the k-mean algorithm. For simplicity, you will consider datasets of two-dimensional points and use the euclidean distance as a notion of similarity. (Type annotations are omitted and left to you.)

1. Write a function `random_points(width,height,n)` that returns a list of `n` random, distinct points in a `width` by `height` area.

2. Write a function `show_clusters(clusters)` that displays a scatter plot of the given clusters and their centroids.

3. Write a function `k_mean(data, k)` that returns a clustering of `data` computed using the `k`-mean algorithm. (Hint: use your `show_clusters` to visualise the evolution of the clusters during the execution of the algorithm).