Pattern Recognition Practical 6

Group 24: Maikel Withagen (s1867733) Steven Bosch (s1861948)

October 21, 2015

Assignment 1

1

When we take $k = 2$ the Minkowski metric is the same as the Euclidean distance between the points, which is used as error function in other clustering methods such as K-means clustering.

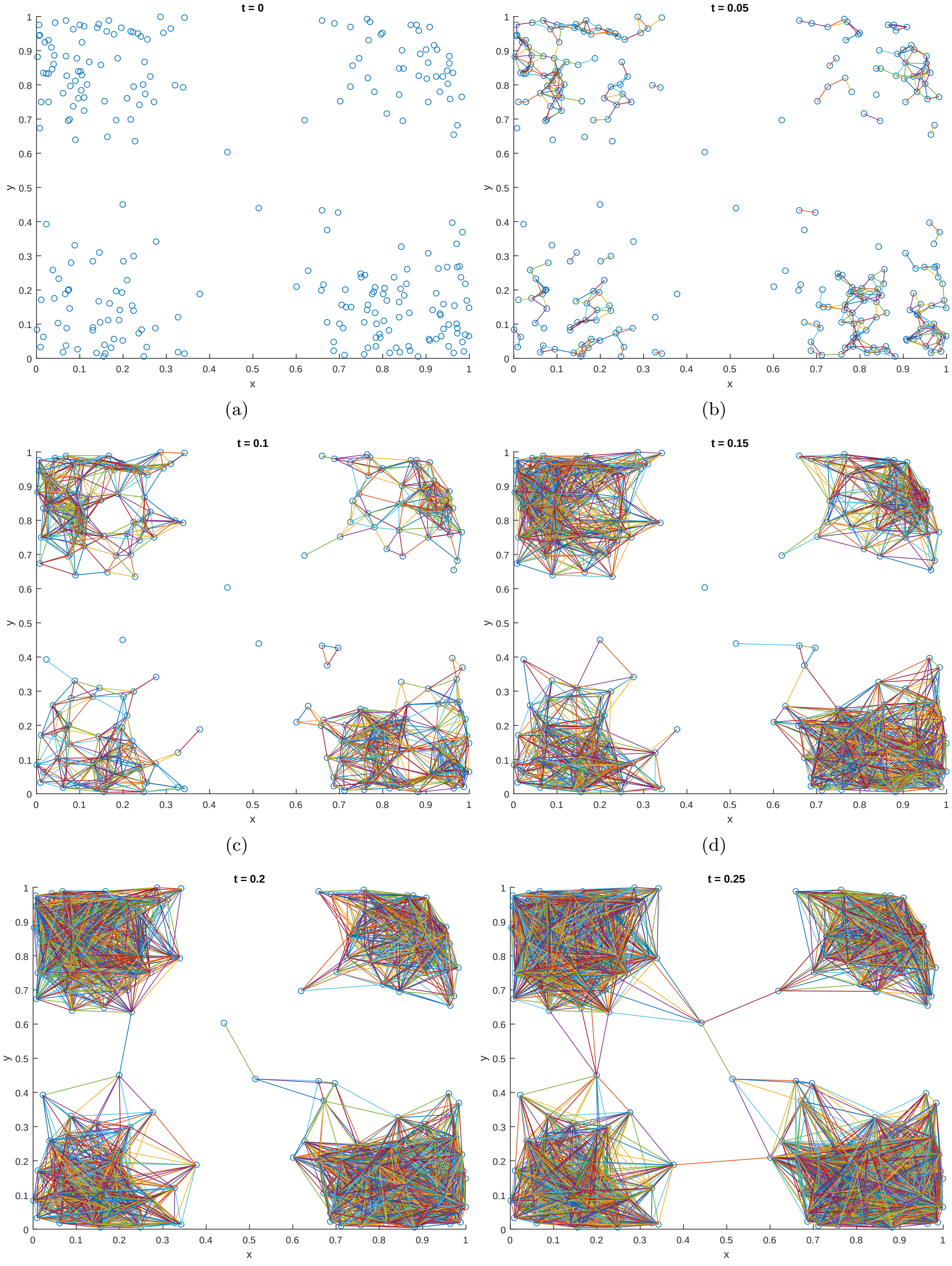
2

See section A in the appendix for our implementation.

3

Looking at figure 1a we can clearly distinguish four main clusters within the data. The figures show that for higher values of t more connections are plotted. This is logical since a higher threshold permits higher distances between two points to be plotted, which overall results in more plotted connections. As for the optimal value of t, 0.05 is clearly too low, because we can see some connections within the clusters, but a lot of points that clearly belong to the clusters are left out because their distance to the other points is too high (see figure 1b). We can still see this for a t of 0.1, but on a smaller scale (see figure 1c). On the other hand a t of 0.25 is clearly too high, because multiple different clusters get connected through outliers, causing multiple clusters to be clustered together (see figure 1f). The same thing happens with a t of 0.2, where the two left clusters are connected (see figure 1e). Since this does not happen for a t of 0.15 and almost all of the points are assigned to a cluster (see figure 1d), this seems to be the optimal value of t. There is one point that does not get assigned to a cluster, so we will have to accept this as an outlier that does not belong to any cluster.

Figure 1: Minkowski clustering for different threshold values.



(a)

(b)

(c)

(d)

(e)

(f)

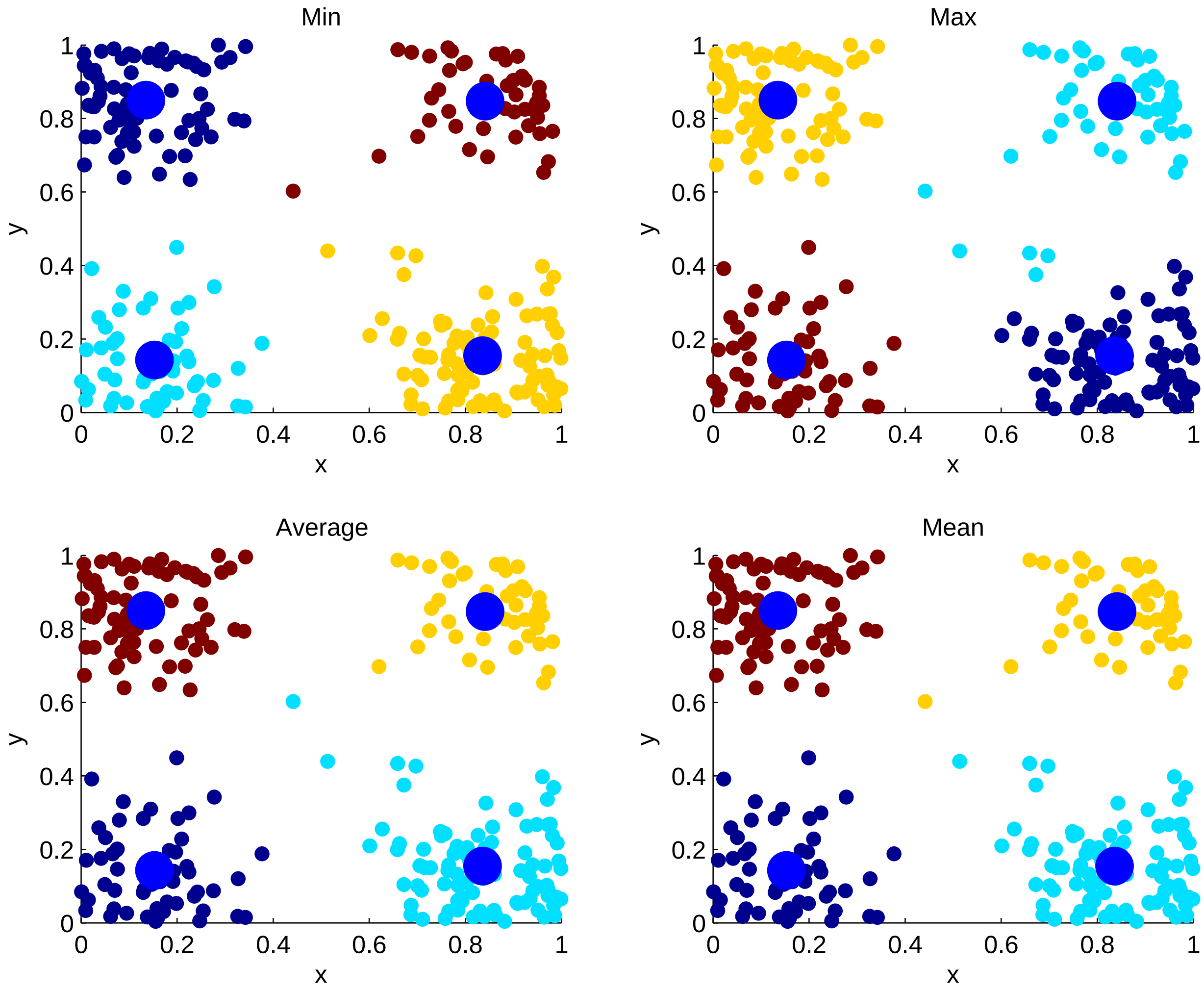


Figure 2: Agglomerative hierarchical clustering using different distance functions. The big circles are the centroids per cluster.

Assignment 2

1

See section B in the appendix for our implementation. Figure 2 shows the different clusterings the agglomerative hierarchical clustering method yields for different distances. As we can see, the different distances yield similar clusters but not entirely equal. The data points that are in the middle of the graphs get attributed to different clusters, dependening on the distance type used. When we interpret the different clusterings by just looking at the plots, it seems like the minimum and the mean distance yield the best solutions out of the four. But to know for certain an error could be computed to see which distance yields the best results.

2

Figure 3 gives the dendograms for the agglomerative hierarchical clustering using different distance functions. Let us assume four main clusters again. We can extrapolate information about the distances between

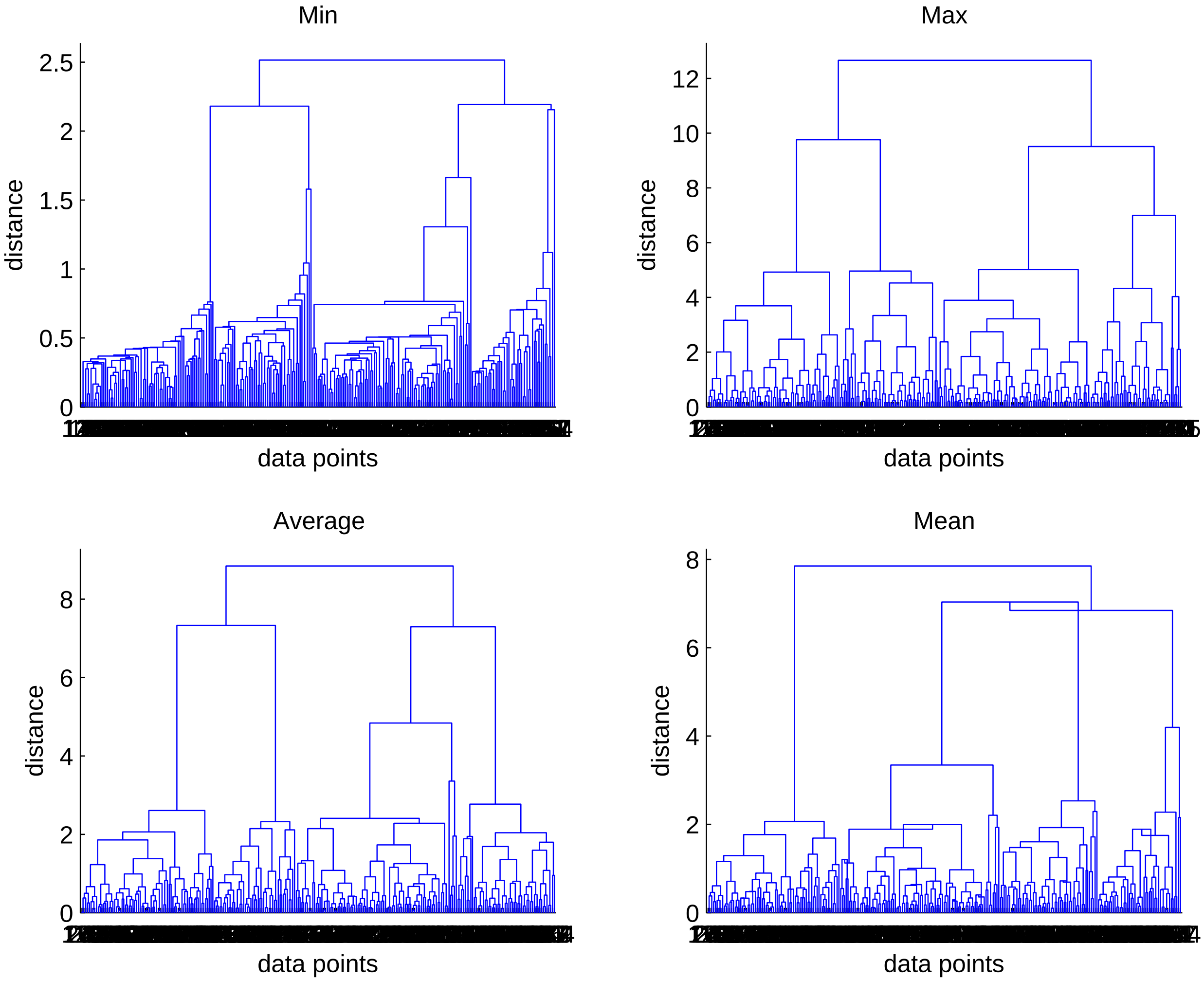


Figure 3: Dendroids for the agglomerative hierarchical clustering using different distance functions.

members of these four clusters from these graphs. For example, the minimum distance dendogram shows us the minimum distances between members of the four main clusters: the nearest members of the first two main clusters shown in the graph (from left to right) lie about 0.75 units apart, whereas the nearest members of the last two main clusters lie very close (in the order of 0.05 units) to each other. The nearest members between the first two clusters and the last two clusters are about 0.25 units apart.

Similarly we can deduce from the maximum distance graph that the farthest members from the first two main clusters lie about 5 units apart and from the last two main clusters about 3 units. Between the first and last two clusters the farthest members lie about 3 units apart. The average member of the first two clusters lie about 4 units apart and those of the last two clusters about 2 units. The average members between the first and last two clusters lie about 2 units apart.

Finally in the mean graph we can see something different with respect to the previous three. Here the centroid of one cluster is farther away from the three other clusters than their's are to each other. So there is one cluster that lies further apart from the other three, whereas the previous three graphs did not show that. This is probably due to the fact the last method uses distances between centroids, which is more robust towards outliers than the other three methods. Therefore the first three graphs do not show this, while the last one does.

Assignment 3

1

Using the code given in section C in the appendix, we computed the J-values for the different clusterings, shown in figure 1

Table 1: J_e-values for different clusters

Clustering	J_e
$\{\{x1, x2, x3\}, \{x4, x5\}\}$	13.1667
$\{\{x2, x3, x5\}, \{x1, x4\}\}$	20.6667
$\{\{x4\}, \{x1, x2, x3, x5\}\}$	17.7500
$\{\{x4, x5\}, \{x1, x2, x3\}\}$	13.1667
$\{\{x3, x5\}, \{x1, x2, x4\}\}$	22.6667

2

Table 1 shows that the clusterings $\{\{x1, x2, x3\}, \{x4, x5\}\}$ and $\{\{x4, x5\}, \{x1, x2, x3\}\}$ minimize the sum-of-squared error. The order of the clusters does not matter (different orderings of a sum yield the same result), as long as they have the same members. That is why these two clusterings yield the same result.

Appendix

A Assignment 1

../Code/Ass1.m

```
close all;
load('cluster_data.mat', 'cluster_data');
dat = cluster_data;
k = size(dat, 2);
% Calculate the minkowski distances between the points for k dimensions
dist = pdist2(dat, dat, 'minkowski', k);

% Make a new figure for every t-value and plot the relevant connections
for t = 0.00 : 0.05 : 0.25
    figure;
    hold on;
    plot(dat(:,1), dat(:,2), 'o');
    % Loop over all points and plot the connections when the distance
    % between two points is smaller than t
    for point = 1 : length(dat)
        for point2 = point+1 : length(dat)
            if dist(point, point2) < t
                plot([dat(point,1) dat(point2,1)], [dat(point,2) dat(point2,2)])
            end
        end
    end
    xlabel('x'); ylabel('y'); title(['t = ' num2str(t)]);
    print(sprintf(['../Report/Ass1_' num2str(t*100)]), '-depsc');
    hold off;
end
```

B Assignment 2

../Code/Ass2.m

```

X = cluster_data;
c_min = cluster(linkage(squareform(pdist(cluster_data)), 'single'), 'maxclust', 4);
c_max = cluster(linkage(squareform(pdist(cluster_data)), 'complete'), 'maxclust', 4);
c_avg = cluster(linkage(squareform(pdist(cluster_data)), 'average'), 'maxclust', 4);
c_mean = cluster(linkage(squareform(pdist(cluster_data)), 'centroid'), 'maxclust', 4);

hold on;
figure();
subplot(2,2,1)
scatter(X(:,1),X(:,2),[],c_min, 'filled');
hold on;
for group = 1:4
    plot(mean(X(c_min == group,1)),mean(X(c_min == group,2)), 'o', 'MarkerSize', 15, '
        MarkerFacecolor', 'b');
end
xlabel('x');ylabel('y');title('Min');
subplot(2,2,2    )
scatter(X(:,1),X(:,2),[],c_max, 'filled')
hold on;
for group = 1:4
    plot(mean(X(c_min == group,1)),mean(X(c_min == group,2)), 'o', 'MarkerSize', 15, '
        MarkerFacecolor', 'b');
end
```

```
xlabel('x');ylabel('y');title('Max');
subplot(2,2,3)
scatter(X(:,1),X(:,2),[],c_avg, 'filled')
hold on;
for group = 1:4
    plot(mean(X(c_min == group,1)),mean(X(c_min == group,2)), 'o', 'MarkerSize', 15, 'MarkerFacecolor', 'b');
end
xlabel('x');ylabel('y');title('Average');
subplot(2,2,4)
scatter(X(:,1),X(:,2),[],c_mean, 'filled')
hold on;
for group = 1:4
    plot(mean(X(c_min == group,1)),mean(X(c_min == group,2)), 'o', 'MarkerSize', 15, 'MarkerFacecolor', 'b');
end
xlabel('x');ylabel('y');title('Mean');
print(sprintf('../Report/Ass2_1'), '-depsc');

figure();
subplot(2,2,1); dendrogram(linkage(squareform(pdist(cluster_data)), 'single'), 270);
xlabel('data points');ylabel('distance');title('Min');
subplot(2,2,2); dendrogram(linkage(squareform(pdist(cluster_data)), 'complete'), 270);
xlabel('data points');ylabel('distance');title('Max');
subplot(2,2,3); dendrogram(linkage(squareform(pdist(cluster_data)), 'average'), 270);
xlabel('data points');ylabel('distance');title('Average');
subplot(2,2,4); dendrogram(linkage(squareform(pdist(cluster_data)), 'centroid'), 270);
xlabel('data points');ylabel('distance');title('Mean');
print(sprintf('../Report/Ass2_2'), '-depsc');
```

C Assignment 3

../Code/Ass3.m

```
x1 = [0 0];
x2 = [2 3];
x3 = [1 4];
x4 = [4 2];
x5 = [3 0];

% {{x1, x2, x3}, {x4, x5}}
m1 = 1/3 * (x1+x2+x3);
m2 = 1/2 * (x4+x5);
J1 = norm(x1-m1).^2+norm(x2-m1).^2+norm(x3-m1).^2+norm(x4-m2).^2+norm(x5-m2).^2

% {{x2, x3, x5}, {x1, x4}}
m1 = 1/3 * (x2+x3+x5);
m2 = 1/2 * (x1+x4);
J2 = norm(x2-m1).^2+norm(x3-m1).^2+norm(x5-m1).^2+norm(x1-m2).^2+norm(x4-m2).^2

% {{x4}, {x1, x2, x3, x5}}
m1 = x4;
m2 = 1/4 * (x1+x2+x3+x5);
J3 = norm(x4-m1).^2+norm(x1-m2).^2+norm(x2-m2).^2+norm(x3-m2).^2+norm(x5-m2).^2

% {{x4, x5}, {x1, x2, x3}}
m1 = 1/2 * (x4+x5);
m2 = 1/3 * (x1+x2+x3);
J4 = norm(x4-m1).^2+norm(x5-m1).^2+norm(x1-m2).^2+norm(x2-m2).^2+norm(x3-m2).^2

% {{x3, x5}, {x1, x2, x4}}
m1 = 1/2 * (x3+x5);
```

```
m2 = 1/3 * (x1+x2+x4);
J5 = norm(x3-m1).^2+norm(x5-m1).^2+norm(x1-m2).^2+norm(x2-m2).^2+norm(x4-m2).^2
```