Homework 1

Name: Aris Podotas

University: National and Kapodistrian University of Athens

Program: Data Science and Informaion Technologies

Specialization: Bioinformatics - Biomedical Data

Lesson: Clustering Algorithms

Date: November 2024

Contents

1 Analysis **1**

1.1 Exercise 1 . . . 1
1.2 Exercise 2 . . . 9
1.3 Exercise 3 . . . 13
1.4 Exercise 4 . . . 14
1.5 Exercise 5 . . . 18
1.6 Exercise 6 . . . 26
1.6.1 k-means . . . 29
1.6.2 k-meadians . . . 30
1.6.3 k-harmonic means . . . 31
1.7 Exercise 7 . . . 31
1.7.1 k-means . . . 34
1.7.2 k-meadians . . . 35
1.7.3 k-harmonic means . . . 36
1.8 Exercise 8 . . . 36

1 Analysis

1.1 Exercise 1

Since:

$$\begin{cases} med(x_1, x_2, \ldots, x_n) = \frac{x_{\frac{n}{2}} + x_{\frac{n}{2}+1}}{2} & \text{When } n \text{ is even.} \\ med(x_1, x_2, \ldots, x_n) = x_{\frac{n+1}{2}} & \text{When } n \text{ is odd.} \end{cases}$$

We will consider both cases. For each, let us compare the distances produced by the median and any other representative p. It holds true then that: $p > \mu \mid p < \mu$ where $\mid$ is the binary or gate. These two circumstances must be analyzed separately. Thus, there is a 2×2 table of circumstances to consider.

$$A_m = \sum_{i=1}^{n} |x_i - \mu| = |x_1 - \mu| + |x_2 - \mu| + \cdots + |x_n - \mu|$$
$$A_p = \sum_{i=1}^{n} |x_i - p| = |x_1 - p| + |x_2 - p| + \cdots + |x_n - p|$$

When n is odd:

When $p > \mu$: Since $p > \mu \iff p = \mu + c$ where $c > 0$. Then:

$$A_p = \sum_{i=1}^{n} |x_i - (\mu + c)| = \sum_{i=1}^{n} |x_i - \mu - c| = |x_1 - \mu - c| + |x_2 - \mu - c| + \cdots + |x_n - \mu - c|$$

It is at this point that in order to deal with the absolute values we must consider more cases for the values of x_i.

$$\begin{cases} |x_i - \mu| = -x_i + \mu, \quad |x_i - \mu - c| = -x_i + \mu + c & \text{for } x_i \leq \mu \\ |x_i - \mu| = x_i - \mu, \quad |x_i - \mu - c| = -x_i + \mu + c & \text{for } \mu < x_i \leq \mu + c \\ |x_i - \mu| = x_i - \mu, \quad |x_i - \mu - c| = x_i - \mu - c & \text{for } x_i > \mu + c \end{cases}$$

Then

$$A_\mu = \sum_{j=1}^{\frac{n+1}{2}} (-x_j + \mu) + \sum_{i=\frac{n+1}{2}+1}^{n} (x_i - \mu) = -\sum_{j=1}^{\frac{n+1}{2}} (x_j) + \sum_{j=1}^{\frac{n+1}{2}} (\mu) + \sum_{i=\frac{n+1}{2}+1}^{n} (x_i) - \sum_{i=\frac{n+1}{2}+1}^{n} (\mu)$$
$$= -\sum_{j=1}^{\frac{n+1}{2}} (x_j) + \frac{n+1}{2}\mu + \sum_{i=\frac{n+1}{2}+1}^{n} (x_i) - (n - \frac{n+1}{2})\mu = -\sum_{j=1}^{\frac{n+1}{2}} (x_j) + \frac{n+1}{2}\mu + \sum_{i=\frac{n+1}{2}+1}^{n} (x_i)$$
$$- n\mu + \frac{n+1}{2}\mu = -\sum_{j=1}^{\frac{n+1}{2}} (x_j) + \sum_{i=\frac{n+1}{2}+1}^{n} (x_i) + \cancel{n\mu} + \mu - \cancel{n\mu} = -\sum_{j=1}^{\frac{n+1}{2}} (x_j) + \sum_{i=\frac{n+1}{2}+1}^{n} (x_i) + \mu \tag{1}$$

Before we begin with A_p we need the index of the point p in the set. $I(p) = I(\mu + c)$. Let $q = I(\mu + c)$ then $\exists r \in [1, n] : \frac{n+1}{2} + r = q$

$$A_p = \sum_{i=1}^{\frac{n+1}{2}+r} (-x_i + \mu + c) + \sum_{j=\frac{n+1}{2}+r+1}^{n} (x_j - \mu - c) =$$

$$\sum_{i=1}^{\frac{n+1}{2}+r} (-x_i) + \sum_{j=1}^{\frac{n+1}{2}+r} (\mu) + \sum_{j=1}^{\frac{n+1}{2}+r} (c) + \sum_{j=\frac{n+1}{2}+r+1}^{n} x_j - \sum_{j=\frac{n+1}{2}+r+1}^{n} (\mu) - \sum_{j=\frac{n+1}{2}+r+1}^{n} (c)$$

$$= \sum_{i=1}^{\frac{n+1}{2}+r} (-x_i) + (\frac{n+1}{2} + r)\mu + (\frac{n+1}{2} + r)c$$

$$+ \sum_{j=\frac{n+1}{2}+r+1}^{n} x_j - (n - (\frac{n+1}{2} + r))\mu - (n - (\frac{n+1}{2} + r))c$$

$$= \sum_{i=1}^{\frac{n+1}{2}+r} (-x_i) + \frac{n+1}{2}\mu + r\mu + \frac{n+1}{2}c + rc \tag{2}$$

$$+ \sum_{j=\frac{n+1}{2}+r+1}^{n} x_j - n\mu + \frac{n+1}{2}\mu + r\mu - nc + \frac{n+1}{2}c + rc$$

$$= \sum_{i=1}^{\frac{n+1}{2}+r} (-x_i) + \cancel{n\mu} + \mu + 2r\mu + \cancel{nc} + c + 2rc + \sum_{j=\frac{n+1}{2}+r+1}^{n} (x_j) - \cancel{n\mu} - \cancel{nc}$$

$$= \sum_{i=1}^{\frac{n+1}{2}+r} (-x_i) + \mu + 2r\mu + c + 2rc + \sum_{j=\frac{n+1}{2}+r+1}^{n} (x_j)$$

$$= \sum_{i=1}^{\frac{n+1}{2}} (-x_i) + \sum_{k=\frac{n+1}{2}+1}^{\frac{n+1}{2}+r} (-x_k) + \mu + 2r\mu + c + 2rc + \sum_{j=\frac{n+1}{2}+r+1}^{n} (x_j)$$

If we solve for $\sum_{j=1}^{\frac{n+1}{2}} x_j$ in 1 and substitute the same term in 2 we get:

$$
\begin{aligned}
&(1) \sum_{j=1}^{\frac{n+1}{2}} -x_j = A_\mu - \mu - \sum_{i=\frac{n+1}{2}+1}^{n} x_i \\
&(2) \sum_{i=1}^{\frac{n+1}{2}} (-x_i) + \sum_{k=\frac{n+1}{2}+1}^{\frac{n+1}{2}+r} (-x_k) + \mu + 2r\mu + c + 2rc + \sum_{j=\frac{n+1}{2}+r+1}^{n} (x_j) \\
&\iff A_\mu - \cancel{\mu} - \sum_{i=\frac{n+1}{2}+1}^{n} x_i + \sum_{k=\frac{n+1}{2}+1}^{\frac{n+1}{2}+r} -x_k + \cancel{\mu} + 2r\mu + c + 2rc + \sum_{j=\frac{n+1}{2}+r+1}^{n} (x_j) \\
&\iff A_\mu = A_p + \sum_{i=\frac{n+1}{2}+1}^{n} x_i - \sum_{k=\frac{n+1}{2}+1}^{\frac{n+1}{2}+r} -x_k - 2r\mu - c - 2rc - \sum_{j=\frac{n+1}{2}+r+1}^{n} (x_j)
\end{aligned}
\tag{3}
$$

Before the final transformation for 3 we should notice that $\sum_{i=\frac{n+1}{2}+1}^{n} x_i$ contains $-\sum_{k=\frac{n+1}{2}+1}^{\frac{n+1}{2}+r} -x_k, \quad \sum_{j=\frac{n+1}{2}+r+1}^{n} (x_j)$ since it is a sum over all x_i after the median. Essentially:

$$
\sum_{i=\frac{n+1}{2}+1}^{n} x_i = \sum_{k=\frac{n+1}{2}+1}^{\frac{n+1}{2}+r} x_k + \sum_{j=\frac{n+1}{2}+r+1}^{n} (x_j)
$$

$$
\begin{aligned}
&A_\mu = A_p + \sum_{i=\frac{n+1}{2}+1}^{n} x_i - \sum_{k=\frac{n+1}{2}+1}^{\frac{n+1}{2}+r} -x_k - 2r\mu - c - 2rc - \sum_{j=\frac{n+1}{2}+r+1}^{n} (x_j) \\
&\iff A_\mu = A_p + \cancel{\sum_{j=\frac{n+1}{2}+r+1}^{n} (x_j)} + 2 \sum_{k=\frac{n+1}{2}+1}^{\frac{n+1}{2}+r} x_k - 2r\mu - c - 2rc - \cancel{\sum_{j=\frac{n+1}{2}+r+1}^{n} (x_j)}
\end{aligned}
$$

Now the final task is to show that:

$$
2 \sum_{k=\frac{n+1}{2}+1}^{\frac{n+1}{2}+r} x_k - 2r\mu - c - 2rc < 0 \iff \sum_{k=\frac{n+1}{2}+1}^{\frac{n+1}{2}+r} x_k < r\mu + \frac{c}{2} + rc
$$

This sum goes from the point right after the median μ to p and will have r terms. So let us re-write it as follows:

$$
\underbrace{x_{k=1} + \cdots + p}_{r \text{ terms}} < r\cancel{(\mu + c)} \overset{p}{} + \frac{c}{2} \iff \underbrace{x_{k=1} + \cdots + p}_{r \text{ terms}} < \underbrace{p + \cdots + p}_{r \text{ terms}} + \frac{c}{2}
$$

Which is true. (since $\forall x_k \in [\frac{n+1}{2} + 1, \frac{n+1}{2} + r), x_k < p$)

So in conclusion we have proven that when n is odd and $p > \mu,\ A_p > A_\mu$.

When $p < \mu$: Since $p < \mu \iff p = \mu - c$ where $c > 0$. Then:

$$A_p = \sum_{i=1}^{n} |x_i - (\mu - c)| = \sum_{i=1}^{n} |x_i - \mu + c| = |x_1 - \mu + c| + |x_2 - \mu + c| + \cdots + |x_n - \mu + c|$$

It is at this point that in order to deal with the absolute values we must consider more cases for the values of x_i.

$$\begin{cases} |x_i - \mu| = -x_i + \mu, & |x_i - \mu + c| = -x_i + \mu - c & \text{for } x_i \leq \mu - c \\ |x_i - \mu| = -x_i + \mu, & |x_i - \mu + c| = x_i - \mu + c & \text{for } \mu - c < x_i \leq \mu \\ |x_i - \mu| = x_i - \mu, & |x_i - \mu + c| = x_i - \mu + c & \text{for } x_i > \mu \end{cases}$$

Then

$$A_\mu = \sum_{j=1}^{\frac{n+1}{2}} (-x_j + \mu) + \sum_{i=\frac{n+1}{2}+1}^{n} (x_i - \mu) = -\sum_{j=1}^{\frac{n+1}{2}} (x_j) + \sum_{j=1}^{\frac{n+1}{2}} (\mu) + \sum_{i=\frac{n+1}{2}+1}^{n} (x_i) - \sum_{i=\frac{n+1}{2}+1}^{n} (\mu)$$

$$= -\sum_{j=1}^{\frac{n+1}{2}} (x_j) + \frac{n+1}{2}\mu + \sum_{i=\frac{n+1}{2}+1}^{n} (x_i) - (n - \frac{n+1}{2})\mu = -\sum_{j=1}^{\frac{n+1}{2}} (x_j) + \frac{n+1}{2}\mu + \sum_{i=\frac{n+1}{2}+1}^{n} (x_i)$$

$$- n\mu + \frac{n+1}{2}\mu = -\sum_{j=1}^{\frac{n+1}{2}} (x_j) + \sum_{i=\frac{n+1}{2}+1}^{n} (x_i) + \cancel{n\mu} + \mu - \cancel{n\mu} = -\sum_{j=1}^{\frac{n+1}{2}} (x_j) + \sum_{i=\frac{n+1}{2}+1}^{n} (x_i) + \mu$$

It is at this point that we must notice that $\sum_{j=1}^{\frac{n+1}{2}} (x_j)$ contains $\sum_{i=1}^{\frac{n+1}{2}-r} (x_i)$, $\sum_{i=\frac{n+1}{2}-r+1}^{\frac{n+1}{2}} x_i$. Essentially:

$$\sum_{j=1}^{\frac{n+1}{2}} (x_j) = \sum_{i=1}^{\frac{n+1}{2}-r} (x_i) + \sum_{i=\frac{n+1}{2}-r+1}^{\frac{n+1}{2}} x_i$$

Then

$$A_\mu = -\sum_{i=1}^{\frac{n+1}{2}-r} (x_i) - \sum_{i=\frac{n+1}{2}-r+1}^{\frac{n+1}{2}} (x_i) + \sum_{i=\frac{n+1}{2}+1}^{n} x_i + \mu \tag{4}$$

This will be helpful later.

Before we begin with A_p we need the index of the point p in the set. $I(p) = I(\mu - c)$. Let $q = I(\mu - c)$ then $\exists r \in [1, n] : \frac{n+1}{2} - r = q$

$$
\begin{aligned}
A_p &= \sum_{i=1}^{\frac{n+1}{2}-r} (-x_i + \mu - c) + \sum_{j=\frac{n+1}{2}-r+1}^{n} (x_j - \mu + c) = \\
&\sum_{i=1}^{\frac{n+1}{2}-r} (-x_i) + \sum_{j=1}^{\frac{n+1}{2}-r} (\mu) - \sum_{j=1}^{\frac{n+1}{2}-r} (c) + \sum_{j=\frac{n+1}{2}-r+1}^{n} x_j - \sum_{j=\frac{n+1}{2}-r+1}^{n} (\mu) + \sum_{j=\frac{n+1}{2}-r+1}^{n} (c) \\
&= \sum_{i=1}^{\frac{n+1}{2}-r} (-x_i) + (\frac{n+1}{2} - r)\mu - (\frac{n+1}{2} - r)c \\
&+ \sum_{j=\frac{n+1}{2}-r+1}^{n} x_j - (n - (\frac{n+1}{2} - r))\mu + (n - (\frac{n+1}{2} - r))c \\
&= \sum_{i=1}^{\frac{n+1}{2}-r} (-x_i) + \frac{n+1}{2}\mu - r\mu - \frac{n+1}{2}c + rc \\
&+ \sum_{j=\frac{n+1}{2}-r+1}^{n} x_j - n\mu + \frac{n+1}{2}\mu - r\mu + nc - \frac{n+1}{2}c + rc \\
&= \sum_{i=1}^{\frac{n+1}{2}-r} (-x_i) + \cancel{n\mu} + \mu - 2r\mu - \cancel{nc} - c + 2rc + \sum_{j=\frac{n+1}{2}-r+1}^{n} (x_j) - \cancel{n\mu} + \cancel{nc} \\
&= \sum_{i=1}^{\frac{n+1}{2}-r} (-x_i) + \mu - 2r\mu - c + 2rc + \sum_{j=\frac{n+1}{2}-r+1}^{n} (x_j)
\end{aligned}
\tag{5}
$$

If we solve for $\sum_{j=1}^{\frac{n+1}{2}-r} x_j$ in 4 and substitute the same term in 5 we get:

$$
A_p = A_\mu + \sum_{i=\frac{n+1}{2}-r+1}^{\frac{n+1}{2}} x_i - \sum_{i=\frac{n+1}{2}+1}^{n} x_i - \cancel{\mu} + \cancel{\mu} - 2r\mu - c + 2rc + \sum_{j=\frac{n+1}{2}-r+1}^{n} (x_j) \tag{6}
$$

Before the final transformation for 6 we should notice that $\sum_{i=\frac{n+1}{2}-r+1}^{n} x_i$ contains $\sum_{k=\frac{n+1}{2}-r+1}^{\frac{n+1}{2}} x_k, \quad \sum_{j=\frac{n+1}{2}+1}^{n} (x_j)$ since it is a sum over all x_i after the median. Essentially:

$$
\sum_{i=\frac{n+1}{2}-r+1}^{n} x_i = \sum_{k=\frac{n+1}{2}-r+1}^{\frac{n+1}{2}} x_k + \sum_{j=\frac{n+1}{2}+1}^{n} (x_j)
$$

$$
A_p = A_\mu + 2 \sum_{i=\frac{n+1}{2}-r+1}^{\frac{n+1}{2}} x_i - \cancel{\sum_{i=\frac{n+1}{2}+1}^{n} x_i} - 2r\mu - c + 2rc + \cancel{\sum_{j=\frac{n+1}{2}+1}^{n} (x_j)}
$$

Now the final task is to show that:

$$2\sum_{i=\frac{n+1}{2}-r+1}^{\frac{n+1}{2}} x_i - 2r\mu - c + 2rc > 0 \iff \sum_{i=\frac{n+1}{2}-r+1}^{\frac{n+1}{2}} x_i > r\mu + \frac{c}{2} - rc$$

This sum goes from the point right after p to μ and will have r terms. So let us re-write it as follows:

$$\underbrace{x_{k=1} + \cdots + \mu}_{r \text{ terms}} > r\cancel{(\mu - c)}^{p} + \frac{c}{2} \iff \underbrace{x_{k=1} + \cdots + \mu}_{r \text{ terms}} > \underbrace{p + \cdots + p}_{r \text{ terms}} + \frac{c}{2}$$

Which is true. (since $\forall x_k \in [\frac{n+1}{2} - r + 1, \frac{n+1}{2}], x_k > p$)

So in conclusion we have proven that when n is odd and $p < \mu,\ A_p > A_\mu$.

When n is even:
When $p > \mu$: Since $p > \mu \iff p = \mu + c$ where $c > 0$. Then:

$$A_p = \sum_{i=1}^{n} |x_i - (\mu + c)| = \sum_{i=1}^{n} |x_i - \mu - c| = |x_1 - \mu - c| + |x_2 - \mu - c| + \cdots + |x_n - \mu - c|$$

It is at this point that in order to deal with the absolute values we must consider more cases for the values of x_i.

$$\begin{cases} |x_i - \mu| = -x_i + \mu, & |x_i - \mu - c| = -x_i + \mu + c & \text{for } x_i \le \mu \\ |x_i - \mu| = x_i - \mu, & |x_i - \mu - c| = -x_i + \mu + c & \text{for } \mu < x_i \le \mu + c \\ |x_i - \mu| = x_i - \mu, & |x_i - \mu - c| = x_i - \mu - c & \text{for } x_i > \mu + c \end{cases}$$

Then

$$\begin{aligned} A_\mu &= \sum_{j=1}^{\frac{n}{2}} (-x_j + \mu) + \sum_{i=\frac{n}{2}+1}^{n} (x_i - \mu) = \\ &\sum_{j=1}^{\frac{n}{2}} (-x_j) + \frac{n}{2}\mu + \sum_{i=\frac{n}{2}+1}^{n} (x_i) - (n - (\frac{n}{2} + 1))\mu = \\ &\sum_{j=1}^{\frac{n}{2}} (-x_j) + \cancel{\frac{n}{2}}\mu + \sum_{i=\frac{n}{2}+1}^{n} (x_i) - \cancel{n\mu} + \cancel{\frac{n}{2}}\mu + \mu = \\ &\sum_{j=1}^{\frac{n}{2}} (-x_j) + \sum_{i=\frac{n}{2}+1}^{n} (x_i) + \mu \end{aligned} \tag{7}$$

Before we begin with A_p we need the index of the point p in the set. $I(p) = I(\mu + c)$. Let $q = I(\mu + c)$ then $\exists r \in [1, n] : \frac{n}{2} + 1 + r = q$. Notice that we use the index $\frac{n}{2} + 1$

instead of the medians index because $p > \mu$ and all points in the range $[\frac{n}{2}, \frac{n}{2}+1]$ would suffice.

$$A_p = \sum_{i=1}^{\frac{n}{2}+1+r} (-x_i + \mu + c) + \sum_{j=\frac{n}{2}+2+r}^{n} (x_j - \mu - c) =$$

$$\sum_{i=1}^{\frac{n}{2}+1+r} (-x_i) + \cancel{\frac{n}{2}}\mu + \mu + r\mu + \cancel{\frac{n}{2}}c + c + rc + \sum_{j=\frac{n}{2}+2+r}^{n} (x_j) - \cancel{n\mu} + \cancel{\frac{n}{2}}\mu + \mu + r\mu - \cancel{rc} + \cancel{\frac{n}{2}}c + c + rc =$$

$$\sum_{i=1}^{\frac{n}{2}+1+r} (-x_i) + 2\mu + 2r\mu + 2c + 2rc + \sum_{j=\frac{n}{2}+2+r}^{n} (x_j)$$

$$- \sum_{i=1}^{\frac{n}{2}} x_i - \sum_{i=\frac{n}{2}+1}^{\frac{n}{2}+1+r} x_i + 2\mu + 2r\mu + 2c + 2rc + \sum_{j=\frac{n}{2}+2+r}^{n} x_j \tag{8}$$

If we solve for $\sum_{j=1}^{\frac{n}{2}} (x_j)$ in 7 and substitute the same term in 8 we get:

$$A_p = A_\mu - \sum_{i=\frac{n}{2}+1}^{n} x_i - \mu - \sum_{j=\frac{n}{2}+1}^{\frac{n}{2}+1+r} x_j + 2\mu + 2r\mu + 2c + 2rc + \sum_{j=\frac{n}{2}+2+r}^{n} x_j$$

$$\sum_{i=\frac{n}{2}+1}^{n} x_i = \sum_{k=\frac{n}{2}+1}^{\frac{n}{2}+1+r} x_k + \sum_{j=\frac{n}{2}+r+2}^{n} (x_j)$$

$$A_p = A_\mu - 2\sum_{k=\frac{n}{2}+1}^{\frac{n}{2}+1+r} x_k - \cancel{\sum_{j=\frac{n}{2}+r+2}^{n} (x_j)} - \mu + 2\mu + 2r\mu + 2c + 2rc + \cancel{\sum_{j=\frac{n}{2}+2+r}^{n} x_j}$$

Now the final task is to show that:

$$-2\sum_{k=\frac{n}{2}+1}^{\frac{n}{2}+1+r} x_k + \mu + 2r\mu + 2c + 2rc > 0 \iff \mu + 2r\mu + 2c + 2rc > 2\sum_{k=\frac{n}{2}+1}^{\frac{n}{2}+1+r} x_k$$

This sum goes from the point right after the median μ to p and will have r terms. So let us re-write it as follows:

$$\underbrace{x_{k=1} + \cdots + p}_{r \text{ terms}} < r\cancel{(\mu + c)}^{p} + c + \frac{\mu}{2} \iff \underbrace{x_{k=1} + \cdots + p}_{r \text{ terms}} < \underbrace{p + \cdots + p}_{r \text{ terms}} + c + \frac{\mu}{2}$$

Which is true. (since $\forall x_k \in [\frac{n}{2}+1, \frac{n}{2}+1+r), x_k < p$)

So in conclusion we have proven that when n is even and $p > \mu,\ A_p > A_\mu$.

When $p < \mu$: Since $p < \mu \iff p = \mu - c$ where $c > 0$. Then:

$$A_p = \sum_{i=1}^{n} |x_i - (\mu - c)| = \sum_{i=1}^{n} |x_i - \mu + c| = |x_1 - \mu + c| + |x_2 - \mu + c| + \cdots + |x_n - \mu + c|$$

It is at this point that in order to deal with the absolute values we must consider more cases for the values of x_i.

$$\begin{cases} |x_i - \mu| = -x_i + \mu, & |x_i - \mu + c| = -x_i + \mu - c & \text{for } x_i \leq \mu - c \\ |x_i - \mu| = -x_i + \mu, & |x_i - \mu + c| = x_i - \mu + c & \text{for } \mu - c < x_i \leq \mu \\ |x_i - \mu| = x_i - \mu, & |x_i - \mu + c| = x_i - \mu + c & \text{for } x_i > \mu \end{cases}$$

Then

$$A_\mu = \sum_{j=1}^{\frac{n}{2}} (-x_j + \mu) + \sum_{i=\frac{n}{2}+1}^{n} (x_i - \mu) = \sum_{j=1}^{\frac{n}{2}} (-x_j) + \cancel{\frac{n}{2}}\mu + \sum_{i=\frac{n}{2}+1}^{n} (x_i) - \cancel{n\mu} + \cancel{\frac{n}{2}}\mu + \mu \quad (9)$$

Before we begin with A_p we need the index of the point p in the set. $I(p) = I(\mu - c)$. Let $q = I(\mu - c)$ then $\exists r \in [1, n] : \frac{n}{2} + r = q$. Notice that we use the index $\frac{n}{2}$ instead of the medians index because $p < \mu$ and all points in the range $[\frac{n}{2}, \frac{n}{2} + 1]$ would suffice.

$$\begin{aligned} A_p &= \sum_{i=1}^{\frac{n}{2}-r} (-x_i + \mu - c) + \sum_{j=\frac{n}{2}-r+1}^{n} (x_j - \mu + c) = \\ &\sum_{i=1}^{\frac{n}{2}-r} (-x_i) + \cancel{\frac{n}{2}}\mu - r\mu - \cancel{\frac{n}{2}}c + rc + \sum_{j=\frac{n}{2}-r+1}^{n} (x_j) - \cancel{n\mu} + \cancel{\frac{n}{2}}\mu - r\mu + \cancel{nc} - \cancel{\frac{n}{2}}c + rc \\ &\sum_{i=1}^{\frac{n}{2}-r} (-x_i) - r\mu + rc + \sum_{j=\frac{n}{2}-r+1}^{n} (x_j) - r\mu + rc \\ &\sum_{i=1}^{\frac{n}{2}-r} (-x_i) - 2r\mu + 2rc + \sum_{j=\frac{n}{2}-r+1}^{n} (x_j) \end{aligned} \quad (10)$$

It is at this point that we must notice that $\sum_{j=\frac{n}{2}-r+1}^{n} (x_j)$ contains $\sum_{i=\frac{n}{2}-r+1}^{\frac{n}{2}} (x_i),\ \sum_{i=\frac{n}{2}+1}^{n} x_i$. Essentially:

$$\sum_{j=\frac{n}{2}-r+1}^{n} (x_j) = \sum_{i=\frac{n}{2}-r+1}^{\frac{n}{2}} (x_i) + \sum_{i=\frac{n}{2}+1}^{n} x_i$$

Then

$$A_p = \sum_{i=1}^{\frac{n}{2}-r} (-x_i) - 2r\mu + 2rc + \sum_{i=\frac{n}{2}-r+1}^{\frac{n}{2}} (x_i) + \sum_{i=\frac{n}{2}+1}^{n} x_i$$

If we solve for $\sum_{i=\frac{n}{2}+1}^{n} x_i$ in 9 and substitute the same term in 10 we get:

$$A_p = \sum_{i=1}^{\frac{n}{2}-r} (-x_i) - 2r\mu + 2rc + \sum_{i=\frac{n}{2}-r+1}^{\frac{n}{2}} (x_i) + A_\mu - \sum_{j=1}^{\frac{n}{2}} (-x_j) - \mu$$

$\sum_{j=1}^{\frac{n}{2}} (-x_j)$ contains $\sum_{i=1}^{\frac{n}{2}-r} (-x_i)$ and $\sum_{j=\frac{n}{2}-r+1}^{\frac{n}{2}} -x_j$

$$A_p = \cancel{\sum_{i=1}^{\frac{n}{2}-r} (-x_i)} - 2r\mu + 2rc + \sum_{i=\frac{n}{2}-r+1}^{\frac{n}{2}} (x_i) + A_\mu - \cancel{\sum_{i=1}^{\frac{n}{2}-r} (-x_i)} - \sum_{j=\frac{n}{2}-r+1}^{\frac{n}{2}} -x_j - \mu$$

Now the final task is to show that:

$$-2r\mu + 2rc + 2 \sum_{i=\frac{n}{2}-r+1}^{\frac{n}{2}} (x_i) - \mu > 0 \iff \sum_{i=\frac{n}{2}-r+1}^{\frac{n}{2}} (x_i) > r\mu - rc + \frac{\mu}{2}$$

let us re-write it as follows:

$$\underbrace{x_{i=1} + \cdots + \mu}_{r \text{ terms}} > r\cancel{(\mu - c)}^{p} + \frac{c}{2} \iff \underbrace{x_{i=1} + \cdots + \mu}_{r \text{ terms}} > \underbrace{p + \cdots + p}_{r \text{ terms}} + \frac{c}{2}$$

Which is true. (since $\forall x_i \in [\frac{n}{2} - r + 1, \frac{n}{2}], x_k > p$)

So in conclusion we have proven that when n is even and $p < \mu,\ A_p > A_\mu$.

Thus we have proven that the quantity $A = \sum_{i=1}^{n} |x_i - \mu|$ is minimized by the median when $x_1 < x_2 < \cdots < x_n$ and $x_1, x_2, \ldots, x_n \in \mathrm{R}$.

1.2 Exercise 2

i)

Since the Hard K-medians algorithm belongs to the Generalized Hard Algorithm Scheme (Generalized Hard Cost Function Optimization Scheme) it holds true that: Each point belongs exclusively to one cluster. Each cluster is denoted by a representative θ. We must provide the number of clusters a-priori. We define $u_{ij} = \begin{cases} 1, & \text{if } x_i \in C_j \\ 0, & \text{otherwise} \end{cases}$ It holds true that:

$$\sum_{j=1}^{m} u_{ij} = 1, \quad i = 1, 2, \ldots, N$$

The given cost function:

$$J(\theta, U) = \sum_{i=1}^{N} \sum_{j=1}^{m} u_{ij} ||x_i - \theta_j||_1$$

Is similar to the cost function of the hard K-means algorithm. It is for this reason that the algorithm will look very similar and thus we can identify parts of the k-means algorithm (common traits held by the Generalized Hard Cost Function Optimization Scheme) that are not affected by the cost function to re-use them within this algorithm.

The only aspects of the hard k-means that are reliant on the cost function are the $\theta_j(t)$ in the parameter updating. Changing these respective parts of the algorithm with the cost function of the hard k-medians algorithm should result in the k-medians algorithm.

Note: The determination of the partition and the minimization follow the same rules as the hard k-means algorithm. It is easy to see how the result in the following pseudocode is derived if we set θ_j as constant and solve for u_{ij}. It is the same as in the general case.

The general form we will be using then, in *Pseudocode* is:

```
Initialize θ_j(0) % This is done with whatever a-priori method we have at
    our desposal for the dataset
t = 0
Repeat
    for i = 1 to N
        for j = 1 to m
            u_ij(t) = { 1,  if ||x_i − θ_j(t)|| = argmin_{q=1,...,m} ||x_i − θ_q(t)||
                      { 0,  otherwise
        end for{j}
    end for{i}
    t = t + 1
    for k = 1 to m
        θ_k(t) = argmin_{θ_k} J(θ, U(t − 1))
    end for{k}
Until θ_j(t) = θ_j(t − 1),   ∀θ_j ∈ [1, m]
```

Let us determine the form of $J(U(t-1), \theta)$ when θ is minimized. We will encounter the issue of the absolute function being non differentiable at 0. Assume fixed u_{ij}'s As long as $x_i \neq \theta_j$:

Since the cost function is:

$$J(U,\theta) = \sum_{i=1}^{N}\sum_{j=1}^{m} u_{ij}||x_i - \theta_j||_1 \iff$$

$$\sum_{i=1}^{N}\sum_{j=1}^{m} u_{ij}\sum_{r=1}^{l}|x_{ir} - \theta_{jr}|$$

and since we analyze a single θ_j term at a time in the algorithm,

we remove $\sum_{j=1}^{m}$ from our terms to proceed (we have kept u_{ij}'s constant up till now).

$$\iff \sum_{i=1}^{N} u_{ij}\sum_{r=1}^{l}|x_{ir} - \theta_{jr}|$$

We analyze one dimension of the data at a time, in one dimension
the points are arranged in ascending order
(if the order of the data points is rearranged).
Since the terms of our sum can be re arranged in any order, essentially
this data is equivalent to the data in exercise 1 within the sum.
it is already known that for such data the minimization point is the median.
What the minimization would have looked like without the u_{ij}

$$\theta_{jr} = \sum_{i=1}^{N} med(x_{ir})$$

Factoring in u_{ij}

$$\theta_{jr} = \sum_{i=1}^{N} u_{ij} \times med(x_{ir})$$

Thus the algorithm in *Pseudocode* is:

```
Initialize θ_j(0) % This is done with whatever a-priori method we have at
    our desposal for the dataset
t = 0
Repeat
    for i = 1 to N
        for j = 1 to m
            u_ij(t) = { 1,  if ||x_i − θ_j(t)|| = argmin_{q=1,...,m} ||x_i − θ_q(t)||
                      { 0,  otherwise
        end for{j}
    end for{i}
    t = t + 1
    for k = 1 to m
        for r = 1 to l
            θ_kr(t) = Σ_{i=1}^{N} u_ij(t − 1) × med(x_ir)
        end for{r}
    end for{k}
Until  θ_j(t) = θ_j(t − 1),    ∀θ_j ∈ [1, m]
```

ii)

In the probabilistic cost function optimization algorithms we have that:

Each points belongs to all clusters, to a degree of compatibility.
$u_{ij} \in [0,1] i = 1, \ldots, N, j = 1, \ldots, m$
$0 < \sum_{i=1}^{N} u_{ij} < N$

Our algorithm belongs to the Generalized Possibilistic Algorithm Scheme. For this reason we will identify parts of the Generalized Possibilitic Algorithm Scheme that need modification and modify them with the specific results produced by our circumstances.

The given cost function

$$J(U, \Theta) = \sum_{i=1}^{N} \sum_{j=1}^{m} u_{ij} ||x_i - \theta_j||_1 + \sum_{j=1}^{m} \eta_j \sum_{i=1}^{N} u_{ij} \ln u_{ij} - u_{ij}$$

We should note that the general possibilistic algorithm scheme contains two such algorithms and we should choose one iteration for the algorithm. The given cost function is in the form of the General Possibilistic Algorithm Scheme 2 and thus the following algorithm will be part of that shceme.

To derive the general form below we should keep the θ_j constant and solve for the minimization of the u_{ij}'s. This is done in exactly the same way as in the general possibilistic algorithm scheme 2. This means that there is no q value to be chosen.

It is important to note that the values of the η_j's have 2 methods of determining them and the one to choose in the following algorithms is up to the user within their specific dataset.

Derivation of the u_{ij}'s: For fixed θ_j's

$$\frac{\partial J(U, \Theta)}{\partial u_{ij}} = \sum_{i=1}^{N} \sum_{j=1}^{m} ||x_i - \theta_j||_1 + \sum_{j=1}^{m} \eta_j \sum_{i=1}^{N} ln u_{ij} + \not{1} - \not{1}$$

When this quantity is set to 0

$$\sum_{j=1}^{m} \eta_j \sum_{i=1}^{N} ln u_{ij} = - \sum_{i=1}^{N} \sum_{j=1}^{m} ||x_i - \theta_j||_1$$

for any one u_{ij}

$$u_{ij} = e^{-\frac{||x_i - \theta_j||_1}{\eta_j}}$$

The general form we will be using then, in *Pseudocode* is:

```
Initialize ηj's % This is done with whatever a-priori method we have at our desposal for the dataset.
Initialize θj(0) % This is done with whatever a-priori method we have at our desposal for the dataset.
t = 0
Repeat
    for i = 1 to N
        for j = 1 to m
            uij(t) = e^{-||xi - θj(t)||1 / ηj}
        end for{j}
    end for{i}
    t = t + 1
    for k = 1 to m
        θk(t) = argmin_{θk} J(θ, U(t-1))
    end for{k}
Until θj(t) = θj(t-1),   ∀θj ∈ [1, m]
```

To derive the final algorithm we should consider the case where u_{ij}'s are constant and solve for θ_j. This will be largely similar to the hard k-meadians and exercise 1. We will again notice that per one dimension of the l-dimensional vectors the data is arranged (in a different order that the data is fed to the algorithm) in a ascending order. Then we also know that the quantity $\sum_{i=1}^{n} |x_i - \mu|$ is minimized by the meadian. In conclusion we will substitute θ_j with the median of the data in the cost function.

$$argmin J(U, \Theta) = \sum_{i=1}^{N} \sum_{j=1}^{m} u_{ij} \sum_{r=1}^{l} |x_{ir} - med(x_{i=1,r}, \ldots, x_{i=N,r})| + \sum_{j=1}^{m} \eta_j \sum_{i=1}^{N} u_{ij} \ln u_{ij} - u_{ij}$$

Thus the algorithm in *Pseudocode* is:

```
Initialize ηj's % This is done with whatever a-priori method we have at our desposal for the dataset.
Initialize θj(0) % This is done with whatever a-priori method we have at our desposal for the dataset.
t = 0
Repeat
    for i = 1 to N
        for j = 1 to m
            uij(t) = e^{-(Σ_{r=1}^{l} |xir - θjr(t)|) / ηj}
        end for{j}
    end for{i}
    t = t + 1
    for k = 1 to m
        for r = 1 to l
            θkr(t) = Σ_{i=1}^{N} Σ_{j=1}^{m} uij(t-1) Σ_{r=1}^{l} |xir - med(x_{i=1,r}, ..., x_{i=N,r})| +
    Σ_{j=1}^{m} ηj Σ_{i=1}^{N} uij(t-1) ln uij(t-1) - uij(t-1)
        end for{r}
    end for{k}
Until θj(t) = θj(t-1),   ∀θj ∈ [1, m]
```

1.3 Exercise 3

To minimize the quantity A we must notice that $A \mapsto f(x) : \sum_{i=1}^{n} (\frac{1}{x_i} - \frac{1}{\mu})^2$ Then we must notice that $f(x)$ is differentiable. We will differentiate with respect to μ and find

the value of the derivative that is equal to 0.

$$\frac{\partial A}{\partial \mu} \iff \frac{\partial f(x)}{\partial \mu} = \frac{\partial \sum_{i=1}^{n} (\frac{1}{x_i} - \frac{1}{\mu})^2}{\partial \mu} = \frac{\partial \sum_{i=1}^{n} (\frac{1}{x_i^2} - 2\frac{1}{x_i}\frac{1}{\mu} + \frac{1}{\mu^2})}{\partial \mu}$$

$$= \frac{\partial \sum_{i=1}^{n} \frac{1}{x_i^2}}{\partial \mu}^{0} - \frac{\partial \sum_{i=1}^{n} 2\frac{1}{x_i}\frac{1}{\mu}}{\partial \mu} + \frac{\partial \sum_{i=1}^{n} \frac{1}{\mu^2}}{\partial \mu} = +\sum_{i=1}^{n} 2\frac{1}{x_i}\frac{1}{\mu^2} - \sum_{i=1}^{n} \frac{2}{\mu^3}$$

$$= \sum_{i=1}^{n} 2\frac{1}{x_i}\frac{1}{\mu^2} - n \times \frac{2}{\mu^3}$$

Thus when

$$\frac{\partial f(x)}{\partial \mu} = 0 \iff \sum_{i=1}^{n} (\frac{2}{x_i \mu^2}) - \frac{2n}{\mu^3} = 0$$

$$\sum_{i=1}^{n} \frac{2}{x_i \mu^2} = \frac{2n}{\mu^3} \iff \frac{2}{\mu^2} \sum_{i=1}^{n} \frac{1}{x_i} = \frac{2n}{\mu^3} \iff \mu = \frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}}$$

1.4 Exercise 4

Since the Hard K-harmonic means algorithm belongs to the Generalized Hard Algorithm Scheme (Generalized Hard Cost Function Optimization Scheme) it holds true that: Each point belongs exclusively to one cluster. Each cluster is denoted by a representative θ. We must provide the number of clusters a-priori. We define $u_{ij} = \begin{cases} 1, & \text{if } x_i \in C_j \\ 0, & \text{otherwise} \end{cases}$ It holds true that:

$$\sum_{j=1}^{m} u_{ij} = 1, \quad i = 1, 2, \ldots, N$$

The given cost function:

$$J(\theta, U) = \sum_{i=1}^{N} \sum_{j=1}^{m} u_{ij} \sum_{r=1}^{l} (\frac{1}{x_{ir}} - \frac{1}{\theta_{jr}})^2$$

Is similar to the cost function of the hard K-means algorithm. It is for this reason that the algorithm will look very similar and thus we can identify parts of the k-means algorithm (common traits held by the Generalized Hard Cost Function Optimization Scheme) that are not affected by the cost function to re-use them within this algorithm.

The only aspects of the hard k-means that are reliant on the cost function are the $\theta_j(t)$ in the parameter updating. Changing these respective parts of the algorithm with the cost function of the hard k-medians algorithm should result in the k-medians algorithm.

Note: The determination of the partition and the minimization follow the same rules as the hard k-means algorithm. It is fairly easy to see how the u_{ij} in the following pseudocode was derived if we set θ_j as constant and solve for u_{ij}.

i)

The general form we will be using then, in *Pseudocode* is:

```
Initialize θ_j(0) % This is done with whatever a-priori method we have at
    our desposal for the dataset
t = 0
Repeat
    for i = 1 to N
        for j = 1 to m
            u_ij(t) = { 1, if Σ_{r=1}^{l} (1/x_ir − 1/θ_jr(t))^2 = argmin_{q=1,...,m} Σ_{r=1}^{l} (1/x_ir − 1/θ_qr(t))^2
                      { 0, otherwise
        end for{j}
    end for{i}
    t = t + 1
    for k = 1 to m
        θ_k(t) = argmin_{θ_k} J(θ, U(t − 1))
    end for{k}
Until θ_j(t) = θ_j(t − 1),   ∀θ_j ∈ [1, m]
```

Let us determine the form of $J(U(t-1),\theta)$ when θ is minimized. Assume fixed u_{ij}'s.

$$\frac{\partial J}{\partial \theta_j} = \frac{\partial \sum_{i=1}^{N}\sum_{j=1}^{m} u_{ij}\sum_{r=1}^{l}(\frac{1}{x_{ir}}-\frac{1}{\theta_{jr}})^2}{\partial \theta_j} =$$

$$\sum_{i=1}^{N}\sum_{j=1}^{m} u_{ij}\sum_{r=1}^{l}\frac{2}{x_{ir}\theta_{jr}^2} - l\frac{2}{\theta_{jr}^3}$$

when this quantity is 0

$$\sum_{i=1}^{N}\sum_{j=1}^{m} u_{ij}\sum_{r=1}^{l}\frac{2}{x_{ir}\theta_{jr}^2} = \sum_{i=1}^{N}\sum_{j=1}^{m} u_{ij} l\frac{2}{\theta_{jr}^3}$$

considering any one r value and removing the $\sum_{r=1}^{l}$.

Since we analyze a single θ_j term at a time in the algorithm,

we remove $\sum_{j=1}^{m}$ from our terms to proceed (we have kept u_{ij}'s constant up till now).

$$\sum_{i=1}^{N} u_{ij}\frac{2}{x_{ir}\theta_{jr}^2} = \sum_{i=1}^{N} u_{ij} l\frac{2}{\theta_{jr}^3} \iff$$

$$\theta_{jr} = \frac{l\sum_{i=1}^{N} u_{ij}}{\sum_{i=1}^{N} u_{ij}\frac{1}{x_{ir}}}$$

Thus the algorithm in *Pseudocode* is:

```
Initialize θ_j(0) % This is done with whatever a-priori method we have at
    our desposal for the dataset
t = 0
Repeat
```

```
    for i = 1 to N
        for j = 1 to m
            u_ij(t) = { 1,  if Σ_{r=1}^{l} (1/x_ir − 1/θ_jr(t))^2 = argmin_{q=1,...,m} Σ_{r=1}^{l} (1/x_ir − 1/θ_qr(t))^2
                      { 0,  otherwise
        end for{j}
    end for{i}
    t = t + 1
    for k = 1 to m
        for r = 1 to l
            θ_kr(t) = l Σ_{i=1}^{N} u_ij(t−1) / Σ_{i=1}^{N} u_ij(t−1) 1/x_ir
        end for{r}
    end for{k}
Until θ_j(t) = θ_j(t − 1),   ∀θ_j ∈ [1, m]
```

ii)

In the probabilistic cost function optimization algorithms we have that:

Each points belongs to all clusters, to a degree of compatibility.
$u_{ij} \in [0,1], i = 1, \ldots, N, j = 1, \ldots, m$
$0 < \sum_{i=1}^{N} u_{ij} < N$

Our algorithm belongs to the Generalized Possibilistic Algorithm Scheme. For this reason we will identify parts of the Generalized Possibilitic Algorithm Scheme that need modification and modify them with the specific results produced by our circumstances.

The given cost function

$$J(U, \Theta) = \sum_{i=1}^{N} \sum_{j=1}^{m} u_{ij} d(x_i, \theta_j) + \sum_{j=1}^{m} \eta_j \sum_{i=1}^{N} u_{ij} \ln u_{ij} - u_{ij}$$

We should note that the general possibilistic algorithm scheme contains two such algorithms and we should choose one iteration for the algorithm. The given cost function is in the form of the General Possibilistic Algorithm Scheme 2 and thus the following algorithm will be part of that shceme.

To derive the general form below we should keep the θ_j constant and solve for the minimization of the u_{ij}'s. This is done in exactly the same way as in the general possibilistic algorithm scheme 2. This means that there is no q value to be chosen.

It is important to note that the values of the η_j's have 2 methods of determining them and the one to choose in the following algorithms is up to the user within their specific dataset.

Derivation of the u_{ij}'s: For fixed θ_j's

$$\frac{\partial J(U,\Theta)}{\partial u_{ij}} = \sum_{i=1}^{N}\sum_{j=1}^{m} d(x_i,\theta_j) + \sum_{j=1}^{m}\eta_j \sum_{i=1}^{N} lnu_{ij} + \not{1} - \not{1}$$

When this quantity is set to 0

$$\sum_{j=1}^{m}\eta_j \sum_{i=1}^{N} lnu_{ij} = -\sum_{i=1}^{N}\sum_{j=1}^{m} d(x_i,\theta_j)$$

for any one u_{ij}

$$u_{ij} = e^{-\frac{d(x_i-\theta_j)}{\eta_j}} = e^{-\frac{\sum_{r=1}^{l}\left(\frac{1}{x_{ir}}-\frac{1}{\theta_{jr}}\right)^2}{\eta_j}}$$

The general form we will be using then, in *Pseudocode* is:

```
Initialize ηj's % This is done with whatever a-priori method we have at
    our desposal for the dataset.
Initialize θj(0) % This is done with whatever a-priori method we have at
    our desposal for the dataset.
t = 0
Repeat
    for i = 1 to N
        for j = 1 to m
            uij(t) = e^{-(Σ_{r=1}^{l} (1/xir - 1/θjr(t))^2)/ηj}
        end for{j}
    end for{i}
    t = t + 1
    for k = 1 to m
        θk(t) = argmin_{θk} J(θ, U(t-1))
    end for{k}
Until θj(t) = θj(t-1),   ∀θj ∈ [1, m]
```

To derive the final algorithm we should consider the case where u_{ij}'s are constant and solve for θ_j. This will be largely similar to the hard k-harmonic means.

$$argmin_{\theta_j} J(U,\Theta) = \frac{\partial J(U,\Theta)}{\partial \theta_j} = \frac{\partial \sum_{i=1}^{N}\sum_{j=1}^{m} u_{ij}\sum_{r=1}^{l}\frac{1}{\left(x_{ir}-\frac{1}{\theta_{jr}}\right)^2} + \cancel{\sum_{j=1}^{m}\eta_j\sum_{i=1}^{N} u_{ij}\ln u_{ij} - u_{ij}}}{\partial \theta_j}$$

This is the same as in i), we will take the result from there.

$$argmin_{\theta_j} J(U,\Theta) \iff \theta_{jr} = \frac{l\sum_{i=1}^{N} u_{ij}}{\sum_{i=1}^{N} u_{ij}\frac{1}{x_{ir}}}$$

Thus the algorithm in *Pseudocode* is:

```
Initialize ηj's % This is done with whatever a-priori method we have at
    our desposal for the dataset.
Initialize θj(0) % This is done with whatever a-priori method we have at
    our desposal for the dataset.
t = 0
Repeat
```

```
    for i = 1 to N
        for j = 1 to m
            u_ij(t) = e^{-(Σ_{r=1}^{l} (1/x_ir - 1/θ_jr(t))^2)/η_j}
        end for{j}
    end for{i}
    t = t + 1
    for k = 1 to m
        for r = 1 to l
            θ_kr(t) = (l Σ_{i=1}^{N} u_ij(t-1)) / (Σ_{i=1}^{N} u_ij(t-1) 1/x_ir)
        end for{r}
    end for{k}
Until θ_j(t) = θ_j(t-1),   ∀θ_j ∈ [1, m]
```

1.5 Exercise 5

Note: Since using a pen and paper would lead to a less aesthetic output in the report, calculations shall be done manually here.

Note: The hard version of these algorithms will be implemented since the possibilistic version derived above use the General Possibilistic Algorithm Scheme 2 and our data do not seem to fit the criteria for using this scheme.

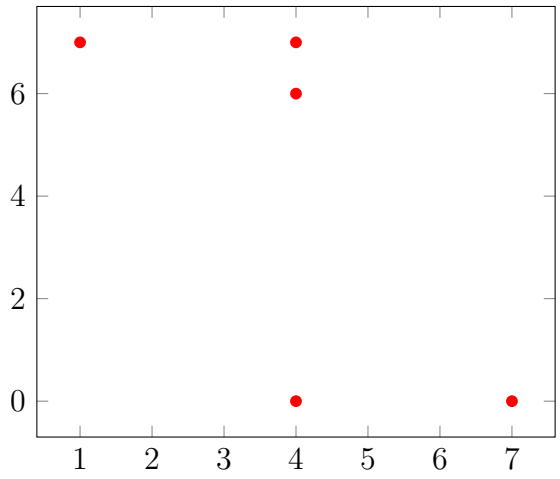
Note: Clusters are represented with blue points

Note: We will represent points within the cluster with representative θ_1 as brown

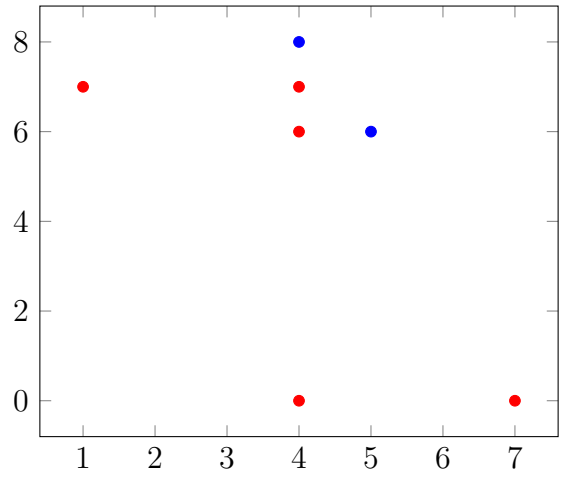
Note: We will represent points within the cluster with representative θ_2 as pink

(a)

Our data looks like this:

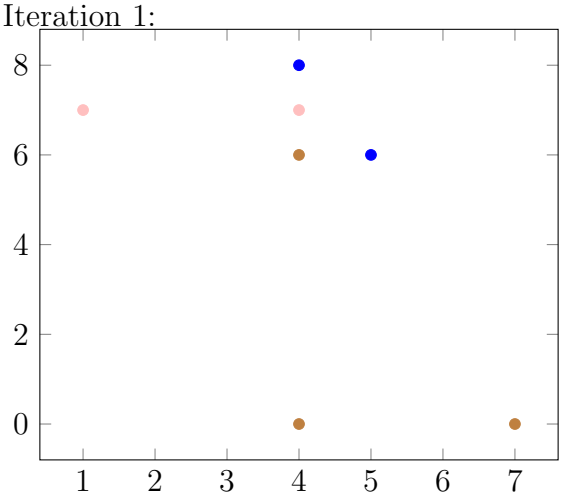


Initializing cluster representatives at $\theta_1(0) = \begin{bmatrix}5 & 6\end{bmatrix}^T, \theta_2(0) = \begin{bmatrix}4 & 8\end{bmatrix}^T$



The case of k-means:

Iteration 1:



$$U = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

New θ_j's:

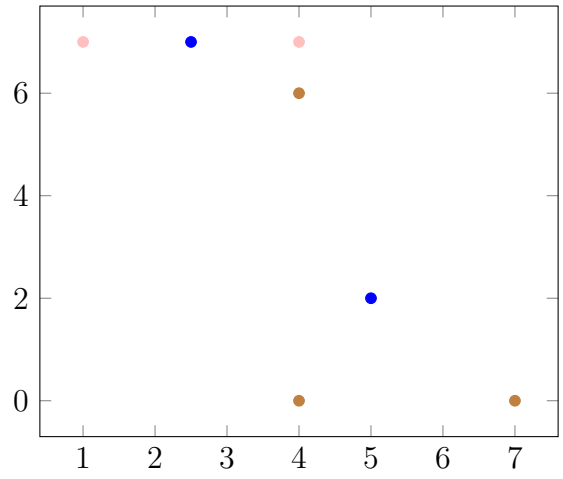
$$\theta_1(1) = \left[\frac{4+7+4}{3} = 5 \quad \frac{0+0+6}{3} = 2\right]^T$$

$$\theta_2(1) = \left[\frac{1+4}{2} = 2.5 \quad \frac{7+7}{2} = 7\right]^T$$

clusters:

$$C_1(1) = \left[[4 \quad 0]^T \quad [7 \quad 0]^T \quad [4 \quad 6]^T\right]^T$$

$$C_2(1) = \left[[1 \quad 7]^T \quad [4 \quad 7]^T\right]^T$$



Iteration 2:

$$U = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

New θ_j's:

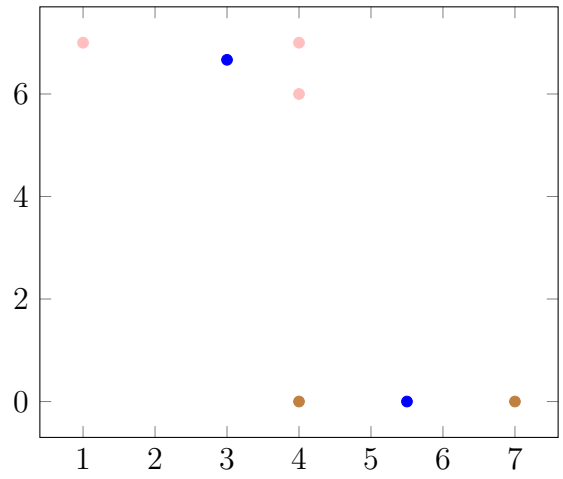
$$\theta_1(1) = \left[\frac{4+7}{2} = 5.5 \quad \frac{0+0}{2} = 0\right]^T$$

$$\theta_2(1) = \left[\frac{1+4+4}{3} = 3 \quad \frac{7+7+6}{3} = \frac{20}{3}\right]^T$$

clusters:

$$C_1(2) = \left[[4 \quad 0]^T \quad [7 \quad 0]^T\right]^T$$

$$C_2(2) = \left[[1 \quad 7]^T \quad [4 \quad 7]^T \quad [4 \quad 6]^T\right]^T$$



Iteration 3:

$$U = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

New θ_j's:

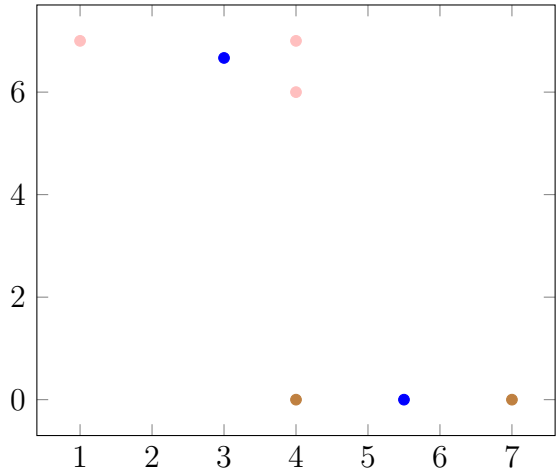
$$\theta_1(1) = \left[\frac{4+7}{2} = 5.5 \quad \frac{0+0}{2} = 0\right]^T$$

$$\theta_2(1) = \left[\frac{1+4+4}{3} = 3 \quad \frac{7+7+6}{3} = \frac{20}{3}\right]^T$$

clusters:

$$C_1(2) = \left[[4 \quad 0]^T \quad [7 \quad 0]^T\right]^T$$

$$C_2(2) = \left[[1 \quad 7]^T \quad [4 \quad 7]^T \quad [4 \quad 6]^T\right]^T$$



Termination of the algorithm since no change in θ_1, θ_2 has occurred between iteration 2 and 3.

The case of k-meadians:

Iteration 1:

$$U = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

New θ_j's:

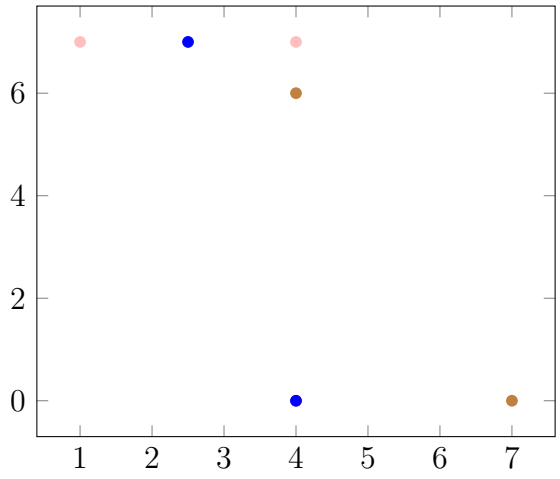
$$\theta_1(1) = \left[4 \quad 0\right]^T$$

$$\theta_2(1) = \left[2.5 \quad 7\right]^T$$

clusters:

$$C_1(1) = \left[[4 \quad 0]^T \quad [7 \quad 0]^T \quad [4 \quad 6]^T\right]^T$$

$$C_2(1) = \left[[4 \quad 7]^T \quad [1 \quad 7]^T\right]^T$$



Iteration 2:

$$U = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

New θ_j's:

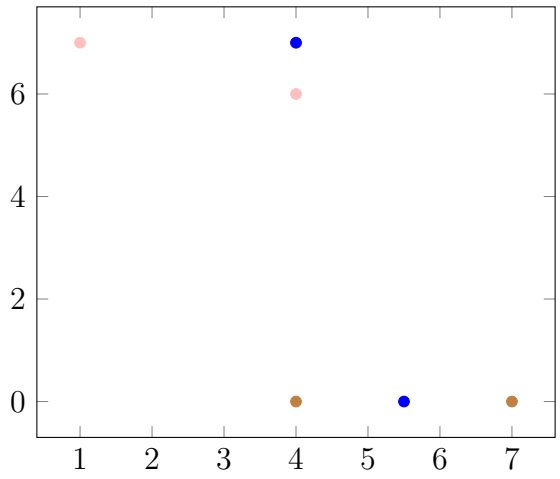
$$\theta_1(1) = \begin{bmatrix} 5.5 & 0 \end{bmatrix}^T$$

$$\theta_2(1) = \begin{bmatrix} 4 & 7 \end{bmatrix}^T$$

clusters:

$$C_1(1) = \begin{bmatrix} [4 & 0]^T & [7 & 0]^T \end{bmatrix}^T$$

$$C_2(1) = \begin{bmatrix} [4 & 7]^T & [1 & 7]^T & [4 & 6]^T \end{bmatrix}^T$$



Iteration 3:

$$U = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

New θ_j's:

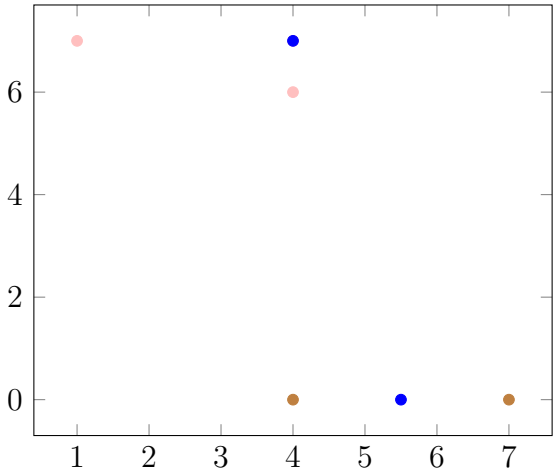
$$\theta_1(1) = \begin{bmatrix} 5.5 & 0 \end{bmatrix}^T$$

$$\theta_2(1) = \begin{bmatrix} 4 & 7 \end{bmatrix}^T$$

clusters:

$$C_1(1) = \begin{bmatrix} [4 & 0]^T & [7 & 0]^T \end{bmatrix}^T$$

$$C_2(1) = \begin{bmatrix} [4 & 7]^T & [1 & 7]^T & [4 & 6]^T \end{bmatrix}^T$$



The algorithm terminates because there is no change between two successive iteration's θ_j's.

The case of k-harmonic means:

Before any of the further analysis we have to consider how to handle undefined values. For the k-harmonic means any undefined values will be ignored. Usually the problem itself would give us some insight into how to deal with these values but no a-priori knowledge exists in this problem and thus we will consider the easiest solution of ignoring (set to 0) these values.

Iteration 1:

Calculations:

$$(\frac{1}{4}-\frac{1}{5})^2+(\frac{1}{0}-\frac{1}{6})^2=0.0302$$
$$(\frac{1}{4}-\frac{1}{4})^2+(\frac{1}{0}-\frac{1}{8})^2=0.015$$
$$(\frac{1}{7}-\frac{1}{5})^2+(\frac{1}{0}-\frac{1}{6})^2=0.0310$$
$$(\frac{1}{7}-\frac{1}{4})^2+(\frac{1}{0}-\frac{1}{8})^20.03925$$
$$(\frac{1}{4}-\frac{1}{5})^2+(\frac{1}{6}-\frac{1}{6})^2=0.0025$$
$$(\frac{1}{4}-\frac{1}{4})^2+(\frac{1}{6}-\frac{1}{8})^20.001736$$
$$(\frac{1}{4}-\frac{1}{5})^2+(\frac{1}{7}-\frac{1}{6})^2=0.003066$$
$$(\frac{1}{4}-\frac{1}{4})^2+(\frac{1}{7}-\frac{1}{8})^2=0.00031$$
$$(\frac{1}{1}-\frac{1}{5})^2+(\frac{1}{7}-\frac{1}{6})^2=0.64$$
$$(\frac{1}{1}-\frac{1}{4})^2+(\frac{1}{7}-\frac{1}{8})^2=0.56$$

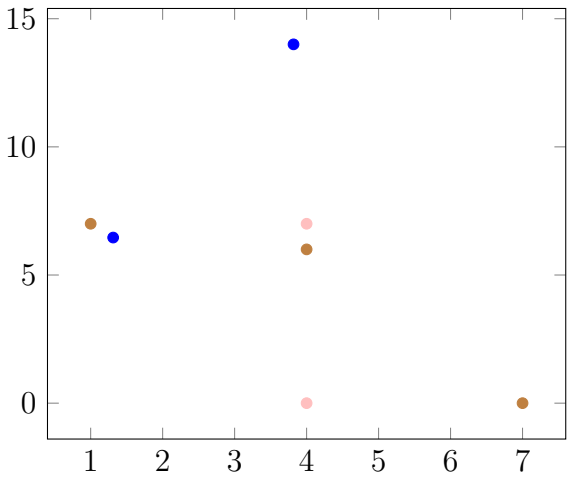
$$U=\begin{bmatrix}0&1\\1&0\\0&1\\1&0\\0&1\end{bmatrix}$$

New θ_j's:

$$\theta_1(1)=\left[\frac{2\times2}{2\times(1/7+1/4)}=\frac{2}{1/7+1/4}\quad\frac{2}{1/0+1/7}\right]^T$$
$$\theta_2(1)=\left[\frac{2\times3}{3\times(1/7+1/4+1/1)}=\frac{2}{1+1/7+1/4}\quad\frac{2}{1/0+1/6+1/7}\right]^T$$

clusters:

$$C_1(1)=\left[[4\quad7]^T\quad[7\quad0]^T\right]^T$$
$$C_2(1)=\left[[1\quad7]^T\quad[4\quad0]^T\quad[4\quad6]^T\right]^T$$



Iteration 2:

$$U = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

New θ_j's:

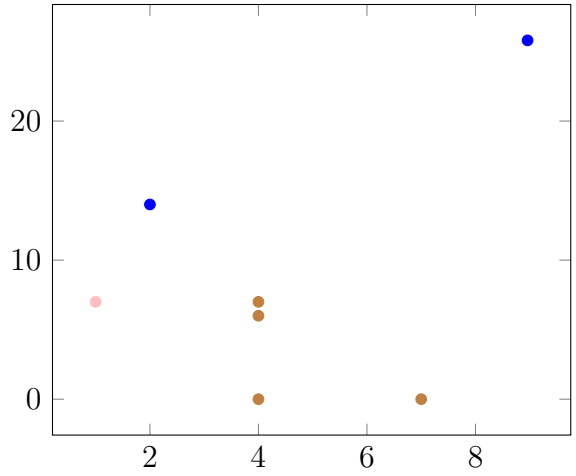
$$\theta_1(1) = \left[\frac{8}{3/4+1/7} = 8.96 \quad \frac{8}{1/6+1/7} = 25,8\right]^T$$

$$\theta_2(1) = \left[\frac{2}{1} = 2 \quad \frac{2}{1/7} = 14\right]^T$$

clusters:

$$C_1(2) = \left[[4 \quad 7]^T \quad [7 \quad 0]^T \quad [4 \quad 0]^T \quad [4 \quad 6]^T\right]^T$$

$$C_2(2) = \left[[1 \quad 7]^T\right]^T$$



Iteration 3:

$$U = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

New θ_j's:

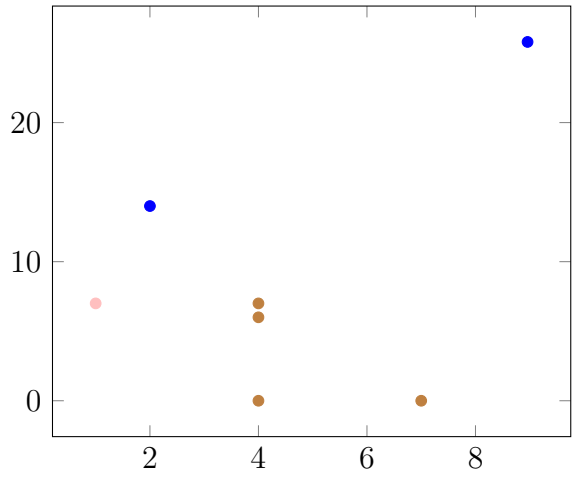
$$\theta_1(1) = \left[\frac{8}{3/4+1/7} = 8.96 \quad \frac{8}{1/6+1/7} = 25,8\right]^T$$

$$\theta_2(1) = \left[\frac{2}{1} = 2 \quad \frac{2}{1/7} = 14\right]^T$$

clusters:

$$C_1(2) = \left[[4 \quad 7]^T \quad [7 \quad 0]^T \quad [4 \quad 0]^T \quad [4 \quad 6]^T\right]^T$$

$$C_2(2) = \left[[1 \quad 7]^T\right]^T$$



The algorithm terminates since no change in θ_j's has occurred.

(b)

In the case of the k-medians there would be a difference in the result for a cluster $\theta_2(0) = \begin{bmatrix}4 & 20\end{bmatrix}$. The distance of the new cluster to the data as compared to the other cluster would be larger for all datapoints in the set. We know a-priori that the k-means algorithm is susceptible to local minimum and in this instance our poor initialization of the clusters leads to a different result, a different local minimum. This is why we try to initialize our clusters within a convex hull formed by the data. In the case of the k-medians the final result will differ again since the new cluster is too far from the data points, all the data points will be clustered with in the first cluster that is much closer (see the new distances in the first iteration. In the case of the k-harmonic means.

We know that the amount of clusters is proportional to the amount of representatives for these 3 algorithms, thus adding one representative would lead to a 3-clustering of the data. Should the new cluster be within the convex hull of the data, the result would change. There is an area outside the convex hull that would change the results once more. It is possible however, for us to add a cluster whose representative is far enough from the data that the clustering remains the same since these algorithms allow empty clusters.

1.6 Exercise 6

Note: All new MATLAB files will be provided in the .zip
Note: All clusters are initialized in the same positions for all algorithms so that a valid comparison can be applied. These positions are

$$\begin{bmatrix} \theta_1 = \begin{bmatrix}0 & 0\end{bmatrix} \\ \theta_2 = \begin{bmatrix}20 & 0\end{bmatrix} \\ \theta_3 = \begin{bmatrix}0 & 20\end{bmatrix} \\ \theta_4 = \begin{bmatrix}20 & 20\end{bmatrix} \end{bmatrix}$$

A file names ./k_harm.m has implemented the hard k-harmonic means algorithm.

a)

The codefile to implement this requirement is ./EX6A.m. Brevity is already outside the scope of this report, so the code will also be shown below.

```
% This file is for the first requirement of exercise 6.
N = 100;  % The number of data vectors

randn('seed',0)
m = [10 10; 20 10; 10 19; 19 18]; % The distribution means of the
    clusters
S(:,:,1)=eye(2);
S(:,:,2)=[1.0 .2; .2 1.5];
S(:,:,3)=[1.0 .4; .4 1.1];
S(:,:,4)=[.3 .2; .2 .5];
n_points=N*ones(1,4);
X=[];

figure(1), hold on
for i=1:4
    X=[X; mvnrnd(m(i,:),S(:,:,i),n_points(i))];
end
X=X';

for i=1:4
    figure(1), plot(X(1,(i-1)*100+1:i*100),X(2,(i-1)*100+1:i*100),'.',
    'Color', rand(1, 3))
end
figure(1), axis equal
```

Output:

b)

Note: The hard versions of these algorithms have been provided so the hard versions of these algorithms will be used.

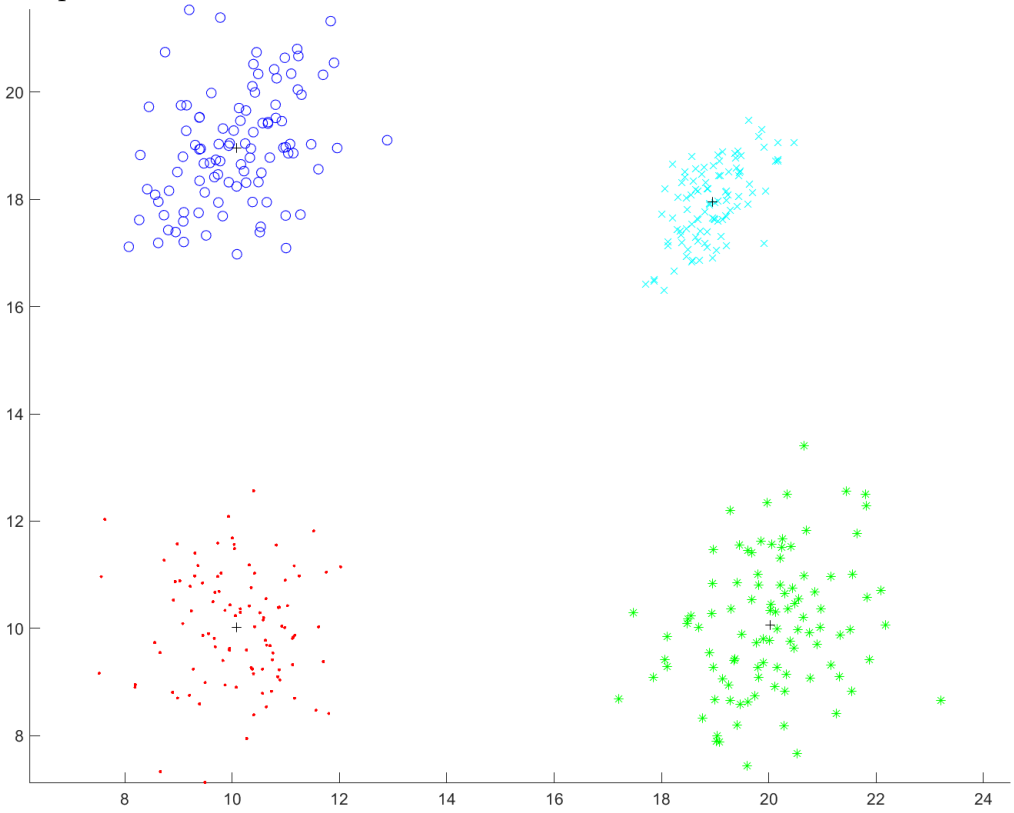
The general form of the algorithms is as follows:

```
% This file is for the first requirement of exercise 6.
l = 2;  % The dimension of the data vectors
N = 100;  % The number of data vectors
% This will be the theta input into the algorithm
theta = [0, 20, 0, 20; 0, 0, 20, 20];

randn('seed',0)
m = [10 10; 20 10; 10 19; 19 18]; % The distribution means of the
    clusters
S(:,:,1)=eye(2);
S(:,:,2)=[1.0 .2; .2 1.5];
S(:,:,3)=[1.0 .4; .4 1.1];
S(:,:,4)=[.3 .2; .2 .5];
n_points=N*ones(1,4);
X=[];

figure(1), hold on
for i=1:4
    X=[X; mvnrnd(m(i,:),S(:,:,i),n_points(i))];
end
X=X';

% Plotting the way the data should be.
for i=1:4
    figure(1), plot(X(1,(i-1)*100+1:i*100),X(2,(i-1)*100+1:i*100),'.',
    'Color', rand(1, 3))
end
figure(1), axis equal

% run the algorithm
[theta,bel,J]=algorithm(X,theta);

% Plot the clusters
figure(2), hold on
figure(2), plot(X(1,bel==1),X(2,bel==1),'r.',...
X(1,bel==2),X(2,bel==2),'g*',X(1,bel==3),X(2,bel==3),'bo',...
X(1,bel==4),X(2,bel==4),'cx',X(1,bel==5),X(2,bel==5),'md',...
X(1,bel==6),X(2,bel==6),'yp',X(1,bel==7),X(2,bel==7),'ks')
figure(2), plot(theta(1,:),theta(2,:),'k+')
figure(2), axis equal
```

Note: Each algorithm has it's own file ./algorithm.m and the file belonging to the above shceme that will run it with the required input ./EX6Balgorithm.m.

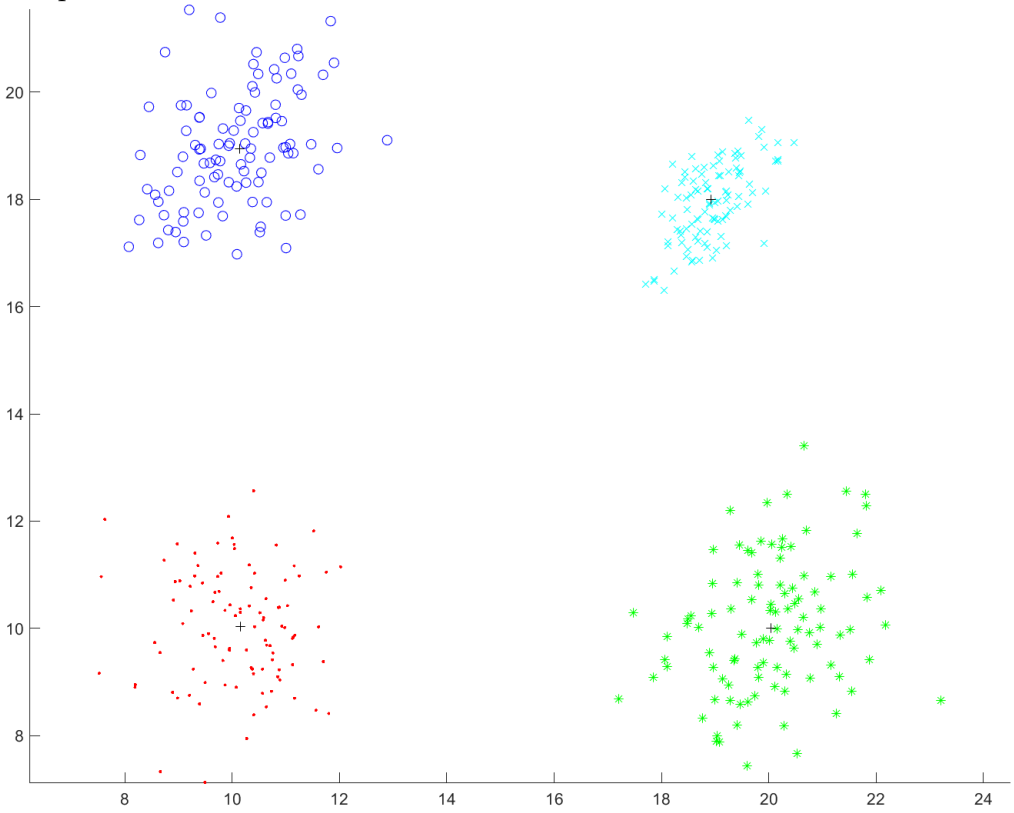
1.6.1 k-means

Output:



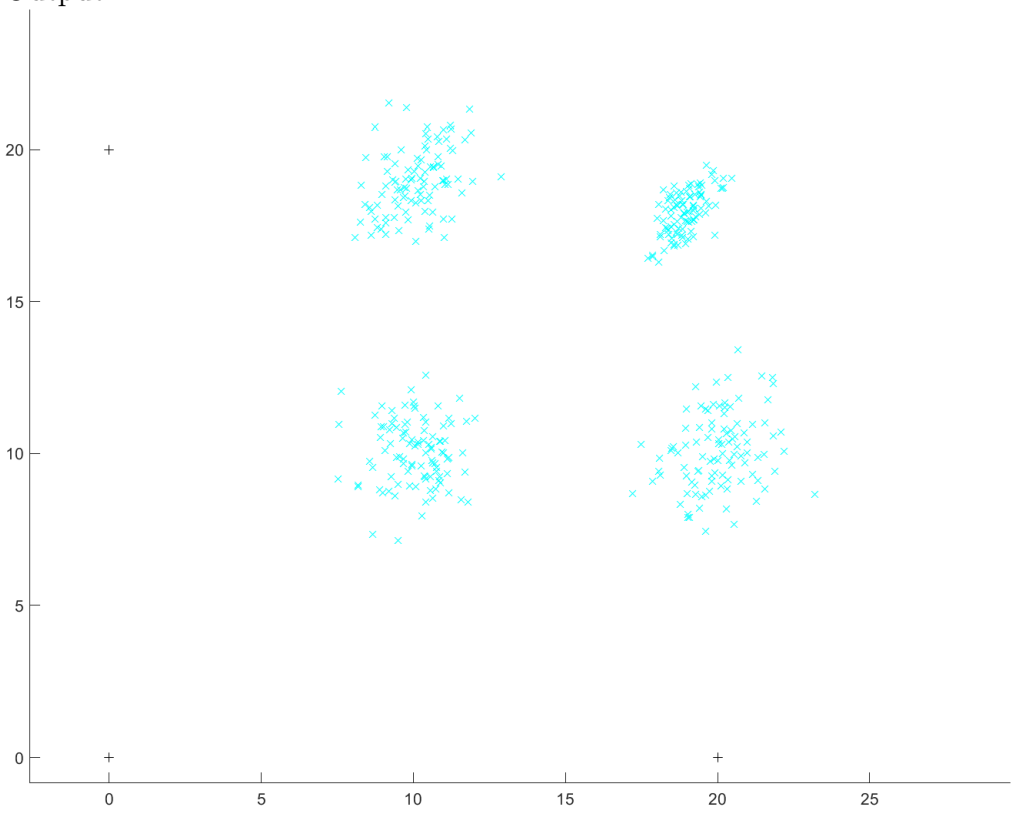
1.6.2 k-meadians

Output:



1.6.3 k-harmonic means

Output:



c)

The k-harmonic means algorithm has a questionable output since all the dataset is within the 4th cluster and the other 3 are empty, the cluster representative for the 4th cluster is very distant from the points of the means of the distributions that made the data.

Thus the following is for the other two algorithms Due to the initialization of the clusters and the shape of the data all algorithms have output the same result. This result is correct in that the clusters contain the correct data. In this case correct just means that each cluster contains all the points stemming from one of the distributions that created the data, and that the cluster contains no more than those points. The very cluster representatives themselves are not synonymous withe the means of the distributions. The margin of error between the cluster representative and the distribution mean is minuscule and not significantly different between the different distributions. The covariance matrix seems to have little effect on the cluster representative position in these examples. We also notice that the different algorithms do not have significant differences in the positions of the cluster representatives. The data distribution is such that the mean and the median and all characteristics match in such a way that produce similar results.

1.7 Exercise 7

a)

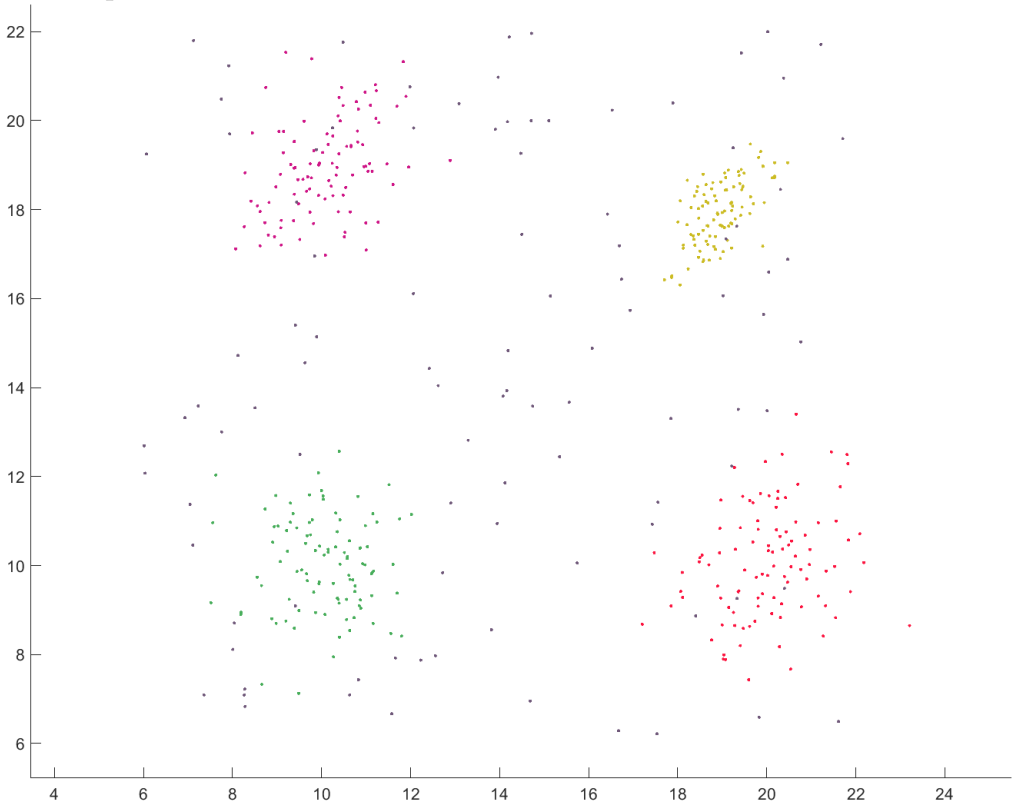
This step has been completed in the k-harm.m file.

b)

The codefile to implement this requirement is ./EX7B.m.

```
% This file is for the first requirement of exercise 6.
N = 100;  % The number of data vectors

randn('seed',0)
m = [10 10; 20 10; 10 19; 19 18]; % The distribution means of the
    clusters
S(:,:,1)=eye(2);
S(:,:,2)=[1.0 .2; .2 1.5];
S(:,:,3)=[1.0 .4; .4 1.1];
S(:,:,4)=[.3 .2; .2 .5];
n_points=N*ones(1,5);
noise=rand(2,100)*16+6;
X=[];

figure(1), hold on
for i=1:4
    X=[X; mvnrnd(m(i,:),S(:,:,i),n_points(i))];
end
X=X';
X=[X, noise];

for i=1:5
    figure(1), plot(X(1,(i-1)*100+1:i*100),X(2,(i-1)*100+1:i*100),'.',
    'Color', rand(1, 3))
end
figure(1), axis equal
```

Output:



c)

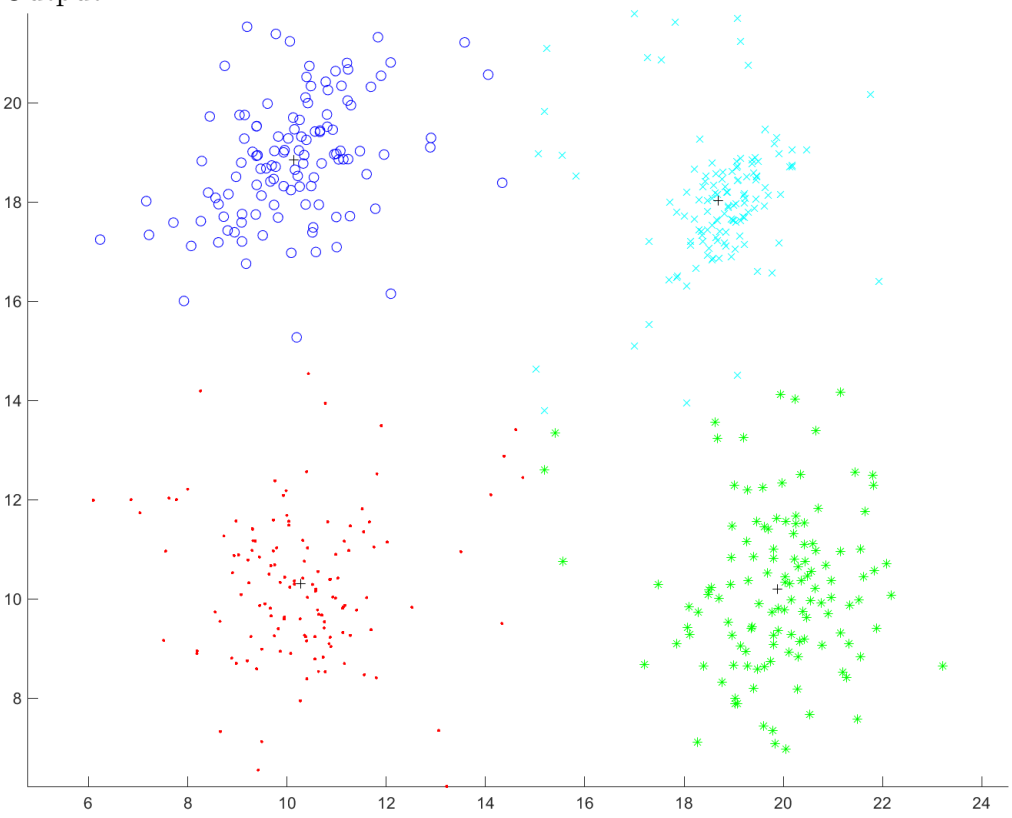
The general form of the algorithms is as follows:

```
% This file is for the first requirement of exercise 6.
l = 2;  % The dimension of the data vectors
N = 100;  % The number of data vectors
% This will be the theta input into the algorithm
theta = [0, 20, 0, 20; 0, 0, 20, 20];

randn('seed',0)
m = [10 10; 20 10; 10 19; 19 18]; % The distribution means of the
    clusters
S(:,:,1)=eye(2);
S(:,:,2)=[1.0 .2; .2 1.5];
S(:,:,3)=[1.0 .4; .4 1.1];
S(:,:,4)=[.3 .2; .2 .5];
n_points=N*ones(1,5);
noise=rand(2,100)*16+6;
X=[];

figure(1), hold on
for i=1:4
    X=[X; mvnrnd(m(i,:),S(:,:,i),n_points(i))];
end
X=X';
X=[X, noise];

% Plotting the way the data should be.
for i=1:5
    figure(1), plot(X(1,(i-1)*100+1:i*100),X(2,(i-1)*100+1:i*100),'.',
    'Color', rand(1, 3))
end
figure(1), axis equal

% run the algorithm
[theta,bel,J]=algorithm(X,theta);

% Plot the clusters
figure(2), hold on
figure(2), plot(X(1,bel==1),X(2,bel==1),'r.',...
X(1,bel==2),X(2,bel==2),'g*',X(1,bel==3),X(2,bel==3),'bo',...
X(1,bel==4),X(2,bel==4),'cx',X(1,bel==5),X(2,bel==5),'md',...
X(1,bel==6),X(2,bel==6),'yp',X(1,bel==7),X(2,bel==7),'ks')
figure(2), plot(theta(1,:),theta(2,:),'k+')
figure(2), axis equal
```

Note: Each algorithm has it's own file ./algorithm.m and the file belonging to the above shceme that will run it with the required input ./EX7Calgorithm.m.

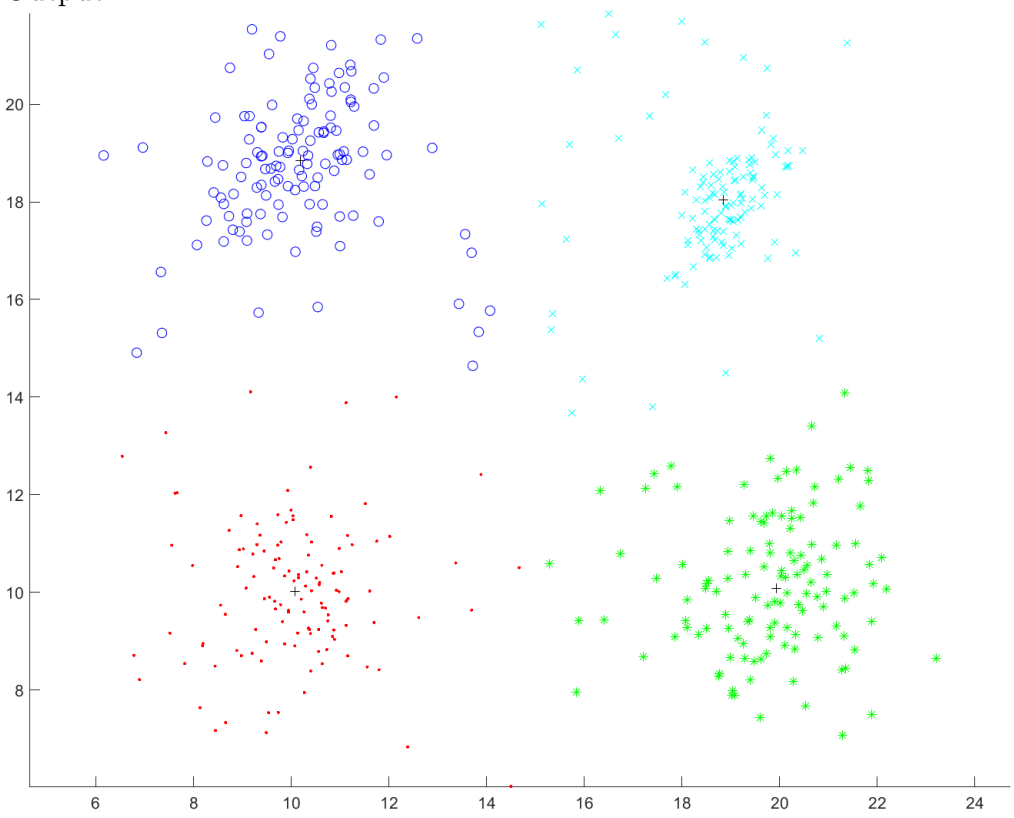
1.7.1 k-means

Output:



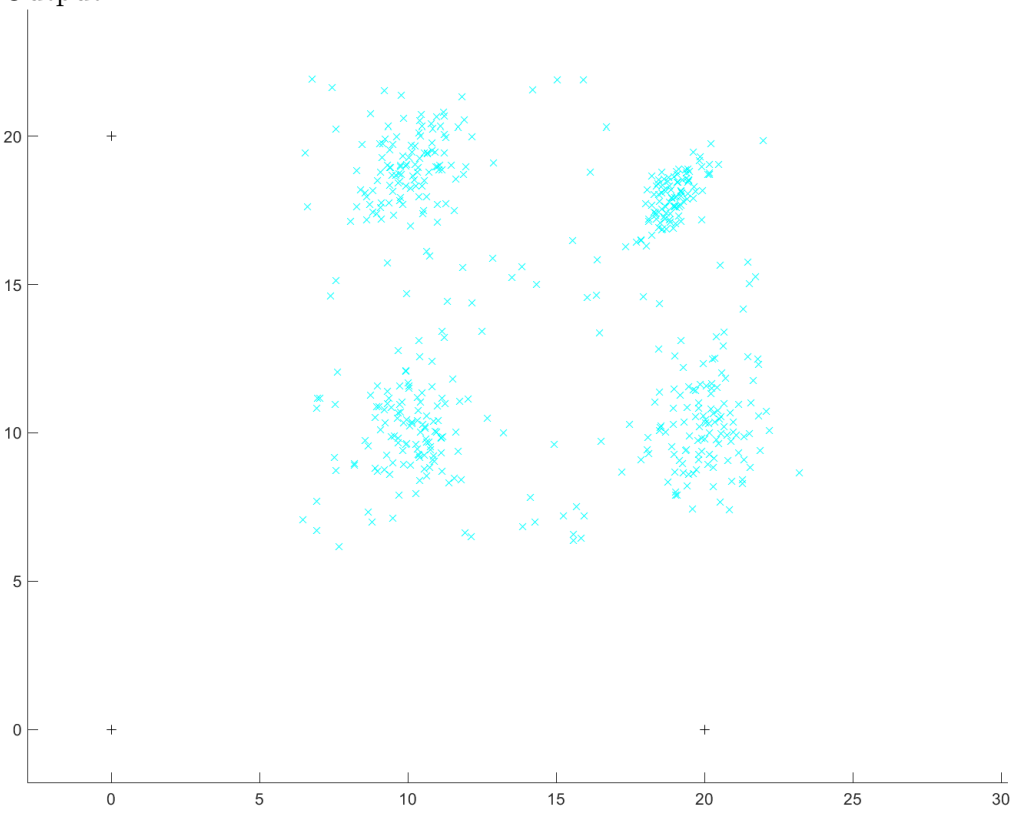
1.7.2 k-meadians

Output:



1.7.3 k-harmonic means

Output:



d)

The k-harmonic means seems to be the wrong algorithm for this data set. Only one cluster contains all the data and it differs from the mean of the distribution that has produced any of the clusters.

The following thus is for the other two algorithms. One thing to notice about the outputs of the algorithms is that they differ between the different algorithms. The results are not exactly correct, in that the clusters contain more than just the points produced from one distribution. Each cluster is "contaminated" with data produced by the noise. The overall clustering however is not of bad quality for each cluster representative is at a "Dense in data" position. The cluster representatives do not differ heavily from the positions of the means of the distributions that produced the data for each cluster. The noise and the covariance matrix of each distributions seem to have little effect on the position of the representatives themselves (as it pertains to the relation with the mean of the distribution producing the data). There is a difference in the way the different algorithms have clustered the data but we cannot say that any of the algorithms are unfit. Between the algorithms themselves the positions of the representatives are not that different from the each other and from the distribution means.

1.8 Exercise 8

Since the k-harmonic means has a less than optimal output the following is for the other two algorithms. There are a few factors to discuss, we should consider the difference in

position of the cluster representatives from the distribution means, the percent coverage of the points produced from a specific distribution within one continuous cluster, the amount of points within a cluster that stem from different distributions, the amount of noise each cluster wrongly characterized, the difference of the algorithms themselves, the user defined parameters (cluster initialization positions), the effect of the covariance matrix of each distribution.

First let us refer to the clustering themselves. For each algorithm the clustering has a high sensitivity, specificity for the datasets with and without the noise. The number of false positives, true positives, false negatives, true negatives obtained by any of the algorithms seem un-problematic. This would depend on the final interpretation and the dataset at hand but for our data it seems appropriate a conclusion. The position of the final cluster representatives is of little importance to the clustering problem but is interesting to note that the difference in position of a cluster representative form the nearest distribution mean was small. The noise had little effect on the distance from a cluster representative to its nearest distribution mean. The degree to which a cluster would identify noise as part of the cluster is higher than the degree to which it would characterize points stemming from a distribution other than the one from the nearest distribution mean.

The initialization of the data has played a big role in this result. The clusters are initialized close to any one distribution mean and yet sufficiently far away from all the other distribution means. The employed strategy was simply fit for the data. Other attempts with significantly different initializations yielded much worse results.

The change in result from the two datasets seems insignificant, which is surprising since noise should have an effect on the algorithms themselves. It would seem that for this data and this initialization these algorithms are fit for "noisy" data.

The separate algorithms had little effect on the distance from the distribution means, but a more significant change in the clustering produced. This was apparent more so in the data set that had noise within it. It can be said that for this data set each algorithm is appropriate for the clustering (the specifics of each dataset would have more insight on this point).

Note: In order to end up with the initialization strategy the other initializations considered where, one where all the clusters are along the $y = x$ line. Once a clustering seemed appropriate (for 2 clusters in total) it was kept. The final strategy came from analyzing the way the data looked.