Assignment 4

Aleksandr Salo

Due April 30, 2015

1 K-means clustering (20 points)

1. **Number of clusters k**
   Given three natural clusters, if we use k = 4 then one cluster inevitably splits on two. Also, mathematically coherently, the cost function decreases with the number of clusters even though they are not good. (Use plotting obj function).

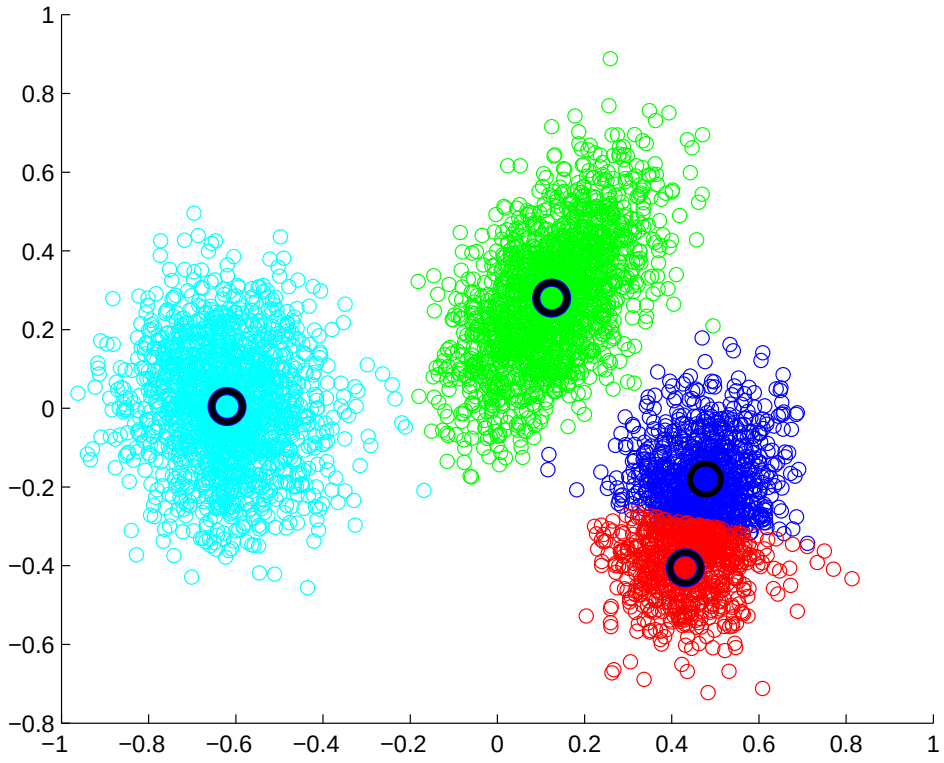
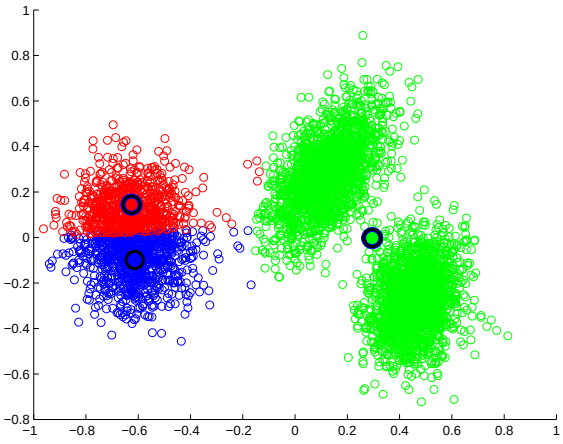


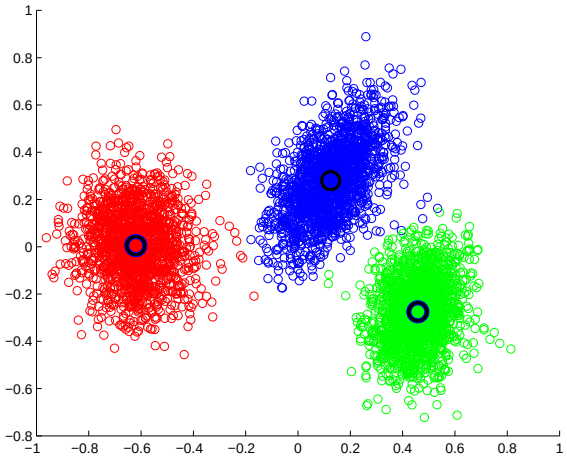
Figure 1: Optimal solution, k = 4

2. **Importance of initialization**
Success of clustering highly depends on the initial guess. Although with given data even random data points assignment to the role of centroids works relatively well, there are common cases when k-mean gets stuck in suboptimal (albeit not necesserilarly bad one) solution.

Given the lucky initialization (with fair chances on this data set) we arrive at seemingly optimal solution.

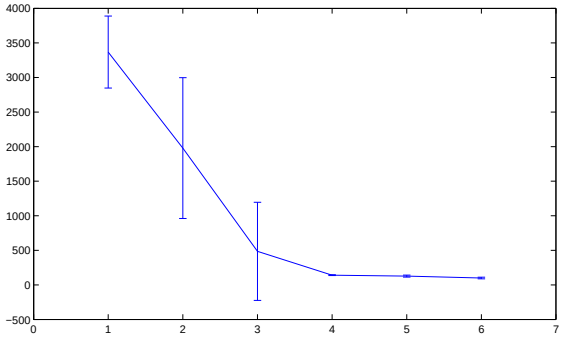


(a) Suboptimal solution, k = 3

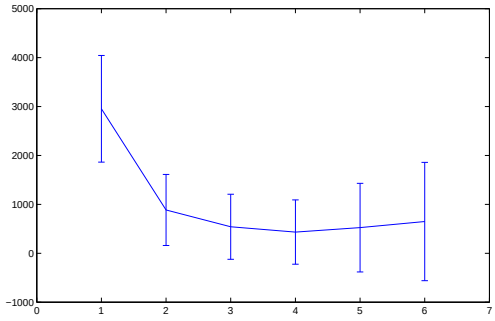


(b) Optimal solution, k = 3

3. **How to determine the optimal number of clusters k?**
That is one of the biggest drawbacks for using k-means - that you need to define it yourself. However, we can plot the average value of cost function as a function of number of clusters k. Logically, cost is decreasing with the increase of k. However, at some point making more clusters should not make much of the improvement (if there are some natural clusters). We want to find that sweet spot that likely to indicate the optimal number of clusters.



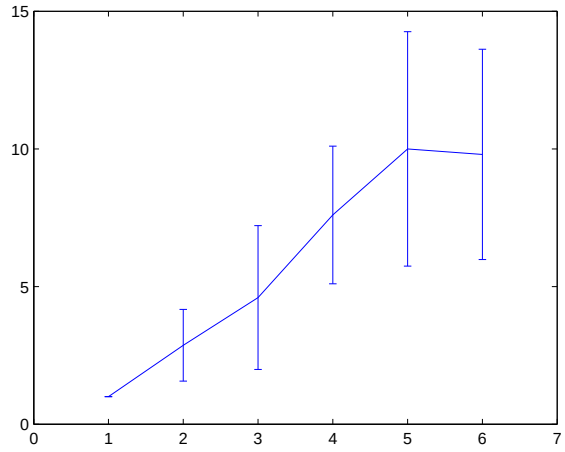
(a) Cost function as function of k, rough descent



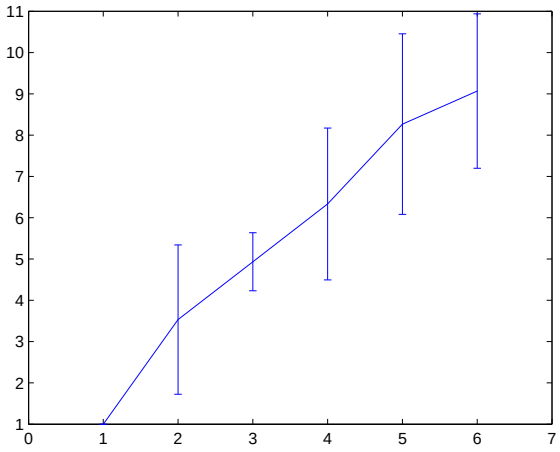
(b) Cost function as function of k, soft descent

Thus we can conclude that the **optimal k = 3**, which is coherent with our observation of the data. However, we need to bear in mind that different initialization could lead to different results thus the plots could look very different and yield slightly different results. Plots above a made with 5 runs for each value of k.

4. **Convergence**.
I used $J > J_{prev} - \epsilon_{convergence}$ stopping condition, where $\epsilon_{convergence} = 0.1$.
Empirically, more clusters we have - more steps it takes to converge. Visually: Seemingly, there is no



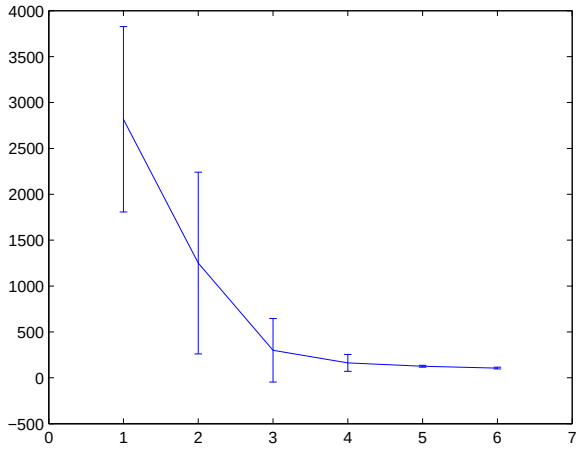
(a) Number of steps to converge, random initialization



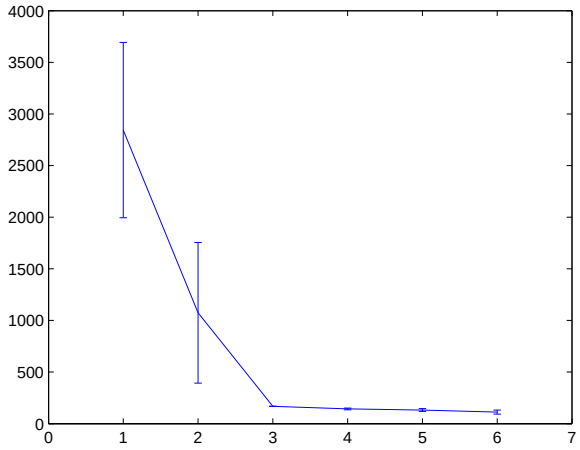
(b) Number of steps to converge, k-means++ init

significant difference between different initialization methods in terms of number of steps to converge. However, standard deviation seem to be smaller in case of k++ init.

5. **Gain of J by k-means++ initialization**



(a) Cost function (15 runs), random initialization



(b) Cost function (15 runs), k-means++ init

Notably, smart initialization *(k-means++, choosing the farthest point from all the existing centroids)* gives significantly better results in terms of cost function J, especially when k is close to optimal. Cohesively, for k=3, there is no variation in optimized cost using smart init - we always arrive in the optimal solution on the given dataset.

K-means Summary

(a) Choice of k is of crucial importance. By choosing wrong number of clusters we doom ourself on failure to determine the natural clusters in data. However, optimized cost function will nonetheless yield suboptimal solution that could be of use. This could be especially true, when clusters are not so naturally separable as in the given dataset.

(b) To avoid "guessing" the k, one can deploy the analysis of cost function J as a function of k. By analyzing the such plot one can find a "sweet" spot with, hopefully, the natural number of clusters present in the data. Once again, this would be not so easy given complex overlapping dataset.

(c) Initialization of centroids could be of paramount significance. While simple guess among present data points yields satisfactory results, there is a fair probability of getting stuck in suboptimal solution if the guess was unlucky. To fight this problem, one could use:

i. Smart initialization i.e. k-means++, where we choose centroids iteratively as the farthest point from all existing centroids.

ii. Run k-means several times and choose the run with the best J.
i.e. on given dataset, the chance of getting bad guess is seemed to be around 30 percent. Thus running k-means for 10 times would make the chance of getting bad guess on all 10 runs is negligible $0.3^{10} = 0.000006$

First option seems to be a preferable one (execution time wise).

(d) Convergence properties of k-means are impressive. No matter how we initialize the centroids we arrive to the optimal (or suboptimal) solution very fast. On given data set we can observe an empirical rule of: number of steps to converge = 1.5k, where k is a number of clusters.

2 The Generalized EM Algorithm (20 points)

GEM algorithm makes the following M-step:

$$\theta \leftarrow \theta + \alpha \nabla_\theta \sum_{i=1}^{m} \sum_{z^{(i)}} Q_i(z^{(i)}) log \frac{p(x^{(i)}, z^{(i)}; \theta)}{Q_i(z^{(i)})}$$

(a) Prove that the GEM algorithm described above converges. To do this, you should show that the likelihood is monotonically improving, as it does for the EM algorithm i.e., show that $\ell(\theta^{(t+1)}) \geq \ell(\theta^{(t)})$

First let us prove that $\ell(\theta^{(t+1)}) \geq \ell(\theta^{(t)})$ for EM.
Since we know that $\forall Q_i, \forall \theta$:

$$\ell(\theta) \geq \sum_{i=1}^{m} \sum_{z^{(i)}} Q_i(z^{(i)}) log \frac{p(x^{(i)}, z^{(i)}; \theta)}{Q_i(z^{(i)})}$$

Then this would hold true for a particular Q, θ:

$$\ell(\theta^{(t+1)}) \geq \sum_{i=1}^{m} \sum_{z^{(i)}} Q_i(z^{(i)}) log \frac{p(x^{(i)}, z^{(i)}; \theta^{(t+1)})}{Q_i(z^{(i)})}$$

Now, since we explicitly choose $\theta^{(t+1)}$ as:

$$\theta^{(t+1)} = arg \max_\theta \sum_{i=1}^{m} \sum_{z^{(i)}} Q_i(z^{(i)}) log \frac{p(x^{(i)}, z^{(i)}; \theta)}{Q_i(z^{(i)})}$$

Then:

$$\sum_{i=1}^{m}\sum_{z^{(i)}} Q_i(z^{(i)})log\frac{p(x^{(i)},z^{(i)};\theta^{(t+1)})}{Q_i(z^{(i)})} \geq \sum_{i=1}^{m}\sum_{z^{(i)}} Q_i(z^{(i)})log\frac{p(x^{(i)},z^{(i)};\theta^{(t)})}{Q_i(z^{(i)})}$$

Lastly, since we specifically chosen $Q_i(z^{(i)}) = p(z^{(i)}|x^{(i)};\theta)$ to make make Jensen's inequality to hold with equality, we can conclude that:

$$\ell(\theta^{(t)}) = \sum_{i=1}^{m}\sum_{z^{(i)}} Q_i(z^{(i)})log\frac{p(x^{(i)},z^{(i)};\theta^{(t)})}{Q_i(z^{(i)})}$$

Considering all three steps together, we proven that:

$$\ell(\theta^{(t+1)}) \geq \ell(\theta^{(t)})$$

Now let us prove that that holds for GEM:
We can apply the same logic, shortly:

$$\begin{aligned}\ell(\theta^{(t+1)}) &\geq \sum_{i=1}^{m}\sum_{z^{(i)}} Q_i(z^{(i)})log\frac{p(x^{(i)},z^{(i)};\theta^{(t+1)})}{Q_i(z^{(i)})} \\ &\geq \sum_{i=1}^{m}\sum_{z^{(i)}} Q_i(z^{(i)})log\frac{p(x^{(i)},z^{(i)};\theta^{(t)})}{Q_i(z^{(i)})} \\ &= \ell(\theta^{(t)})\end{aligned}$$

By the argument given in EM case (Jensen's inequality) the first line holds true. By the choice of distribution for Q we made it hold with equality - thus last line is true.
Lastly, our assumption *"where is a learning rate which we assume is chosen small enough such that we do not decrease the objective function when taking this gradient step."* leads to the second line to be true.

Hence, the likelihood is indeed monotonically improving with each gradient step.

$$\ell(\theta^{(t+1)}) \geq \ell(\theta^{(t)})$$

(b) Instead of using the EM algorithm at all, suppose we just want to apply gradient ascent to maximize the log-likelihood directly. In other words, we are trying to maximize the (non-convex) function

$$\ell(\theta) = \sum_{i=1}^{m} log\sum_{z^{(i)}} p(x^{(i)},z^{(i)};\theta)$$

so we could simply use the update:

$$\theta \leftarrow \theta + \alpha\nabla_\theta \sum_{i=1}^{m} log\sum_{z^{(i)}} p(x^{(i)},z^{(i)};\theta)$$

First, let's take the partial derivative of $\ell(\theta)$ of the new approach with respect to θ_j:

$$\begin{aligned}\frac{\partial}{\partial\theta_j}\sum_{i=1}^{m} log\sum_{z^{(i)}} p(x^{(i)},z^{(i)};\theta) &= \sum_{i=1}^{m}\frac{1}{\sum_{z^{(i)}} p(x^{(i)},z^{(i)};\theta)}\sum_{z^{(i)}}\frac{\partial}{\partial\theta_j}p(x^{(i)},z^{(i)};\theta) \\ &= \sum_{i=1}^{m}\sum_{z^{(i)}}\frac{1}{p(x^{(i)};\theta)}\frac{\partial}{\partial\theta_j}p(x^{(i)},z^{(i)};\theta)\end{aligned}$$

Now let's do the same for GEM:

$$\frac{\partial}{\partial\theta_j}\sum_{i=1}^{m}\sum_{z^{(i)}}Q_i(z^{(i)})log\frac{p(x^{(i)},z^{(i)};\theta)}{Q_i(z^{(i)})}=\sum_{i=1}^{m}\sum_{z^{(i)}}\frac{Q_i(z^{(i)})}{p(x^{(i)},z^{(i)};\theta)}\frac{\partial}{\partial\theta_j}p(x^{(i)},z^{(i)};\theta)$$

Since we specifically chosen $Q_i(z^{(i)}) = p(z^{(i)}|x^{(i)};\theta) = \frac{p(x^{(i)},z^{(i)};\theta)}{p(x^{(i)};\theta)}$ on the E-step of GEM, then by substitution in the previous equation we have:

$$\sum_{i=1}^{m}\sum_{z^{(i)}}\frac{\frac{p(x^{(i)},z^{(i)};\theta)}{p(x^{(i)};\theta)}}{p(x^{(i)},z^{(i)};\theta)}\frac{\partial}{\partial\theta_j}p(x^{(i)},z^{(i)};\theta)=\sum_{i=1}^{m}\sum_{z^{(i)}}\frac{1}{p(x^{(i)};\theta)}\frac{\partial}{\partial\theta_j}p(x^{(i)},z^{(i)};\theta)$$

That is to say we derived exactly the same result as derivative of the log likelihood.
Thus, $\theta \leftarrow \theta + \alpha\nabla_\theta \sum_{i=1}^{m} log \sum_{z^{(i)}} p(x^{(i)},z^{(i)};\theta)$ procedure gives the same update as the GEM algorithm.