Dimensionality Reduction Clustering

by Chieh Wu

Feb/16/17

Dong Ling's algorithm finds a lower dimension reduction of the data while simulaneously finding a good clustering result. The data $X \in \mathbb{R}^{n \times d}$, where n is the number of samples and d is the dimensionality of each sample. His cost function in the linear cast can be written as :

$$\begin{array}{ll} \min & \operatorname{Tr}(XWW^TXHUU^TH) \\ W,U & \\ s.t & W^TW = I \end{array}$$

Although this algorithm perform well by the author, it presents serious challenges when others attempt to reproduce similar results. The first obvious problem is the requirement of the Stiefel manifold constraint. Due to the non-convex nature of this constraint, the solution to the optimization problem does not have an off the shelf solution. Instead, the author proposed an algorithm called Dimension Growth that is difficult to understand, and time consuming to implement correctly. Besides the time intensive requirement of the implementation, the proposed optimization algorithm also converges slowly. As data becomes bigger and bigger, a faster algorithm might be more suitable. Another major problem with the Stiefel manifold constraint is that by removing the convexity assumption, we cannot guarentee global minimum.

Perhaps one of the most uncertain impediment is the false assumption of knowing the dimensionality of W. Since this assumption is rarely true, the dimensionality of the W becomes a questionable hyper-parameter. Without spending extensive amount of time adjusting and tuning q, the solution could vary wildly.

Ideally, we search for an algorithm that removes the Stiefel manifold constraint. We want to be able to find the solution through simple techniques with off the shelf functions. By removing these barriers to entry, we speed up both the time of implementation, and the actual execution run time. A side benefit of removing the Stiefel manifold constraint is the return to a convex cost function. This change allows us to find the global minimum instead of the local. Lastly, we want to completely remove the guess work for the size of W. Instead of tuning q, we want an algorithm that automatically discover this variable while solving for the optimal clustering quality.

In the spirite of these objectives, we have discovered that the stiefel manifold constrain can be removed by combining WW^T term into a single positive definite matrix $A = WW^T$. Assuming that the optimal Whas the dimensions of $W^* \in \mathbb{R}^{d \times q}$ and $q \ll d$, the matrix A is a positive semidefinite. By solving for A, while minimizing for its rank, instead of W, it turns out that many of the problems previously addressed simply disappears. The algorithm for solving the cost function is simple and can be resolved with off the shelf functions. It is fast to code and implement, and the run time speed is magnitudes faster. Lastly, by removing the Stiefel manifold constraint, the problem can be solved with a global solution.

Given the problem :

$$\begin{array}{ll} \min\limits_{A,U} & \operatorname{rank}(A) - \lambda \operatorname{Tr}(KHK_UH) \\ s.t & A = WW^T \\ & W^TW = I \\ & K = XAX^T \\ & A \geq 0 \\ & A \in \mathbb{R}^{n\times n} \end{array}$$

Assume that the optimal solution of $\operatorname{rank}(A^*) = q$, $q \ll n$. The rank portion of the cost function can be relaxed into a log determinant representation, and we can fill in the definition of K and K_U.

$$\min_{A,U} \quad \log(\det(A+\delta I)) - \lambda \operatorname{Tr}(XAX^THUU^TH)$$

The log determinant portion can be further approximated with the first order Taylor approximation to yield.

$$\min_{A,U} \quad \operatorname{Tr}((A_k+\delta I)^{-1}A) - \lambda \operatorname{Tr}(X^THUU^THXA)$$

$$\min_{A,U} \quad \operatorname{Tr}([(A_k+\delta I)^{-1} - \lambda X^THUU^THX]A)$$

$$\min_{A,U} \quad \operatorname{Tr}(\Phi A)$$

We know that $A \geq 0$, therefore the factorization of $A = LL^T$ is possible.

$$\min_{A,U} \quad \operatorname{Tr}(\Phi LL^T) = \operatorname{Tr}(L^T\Phi L)$$

Since we are minimizing the term $\operatorname{Tr}(L^T \, \Phi L)$, L matrix corresponds to the least dominant eigenvectors. Notice that as long as the eigenvalue is negative, the cost is further decreased. Therefore, the dimension q simply corresponds with the number of negative eigenvalues.

On the other hand when we try to optimize for U, we get :

$$\max_{U} \lambda(U^THXAX^THU)$$

Let $Y = HXAX^TH$, U corresponds with the most dominant eigenvectors of Y. Given k as the number of clusters.

$$U \in \mathbb{R}^{n\times k}$$

Experiment 1

Given data $X \in \mathbb{R}^{60\times 4}$. The first 2 dimension consist of 2 linearly separable gaussian distributions. The other 2 dimensions consist of a uniform distribution blanketing the size of the data. Running this algorithm yields a perfect clustering, and a suggested reduction of dimension down to 1.

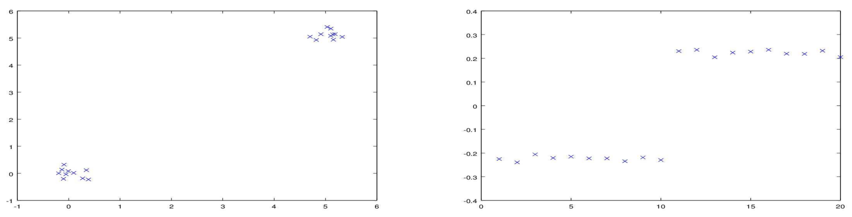


Figure 1.

Multiple other clustering techniques were used to compare the clustering quality. Since the data is synthetic, the true labels exist, and the results of the clusters can be compared visually as well as using normalized mutual information (nmi). From the figure below, it can be seen that the weights of the dimensionality reduction concentrated heavily on the first 2 dimensions. This is reasonable since the last 2 dimensions is just noise of uniform distribution. Due to the noise, also notice that spectral clustering did not perform well which DRC performed as well as GMM and Kmeans.

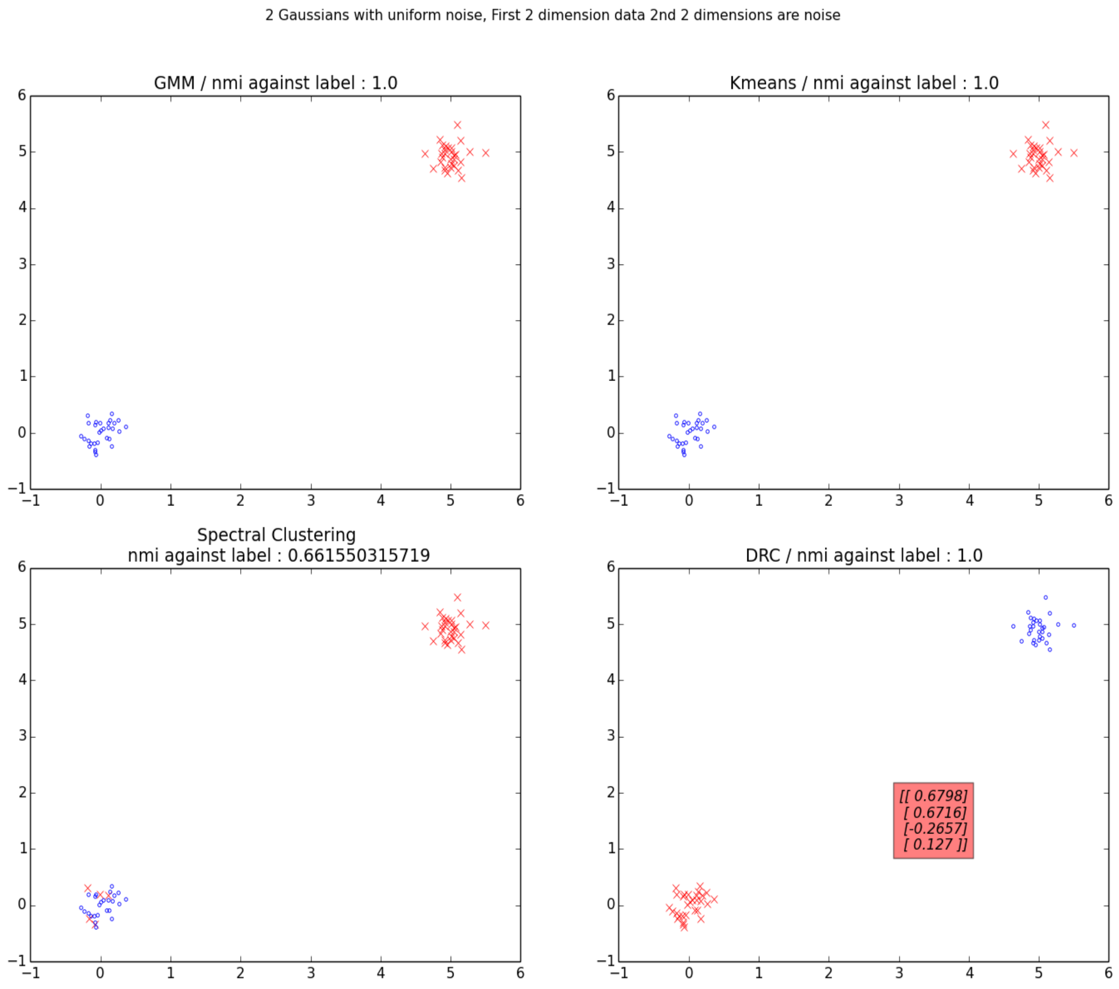


Figure 2.

Experiment 2

The purpose of this experiment is to show an opposite example from example 1. Since example 1 was designed for Spectral clustering to perform poorly and GMM/Kmeans to perform well, this experiment is designed for GMM/Kmeans to perform poorly while Spectral Clustering performing well. It can be concluded from this experiment that DRC perform equally well in both experiment 1 and 2. In terms of Mutual information, notice that DRC performed the best.

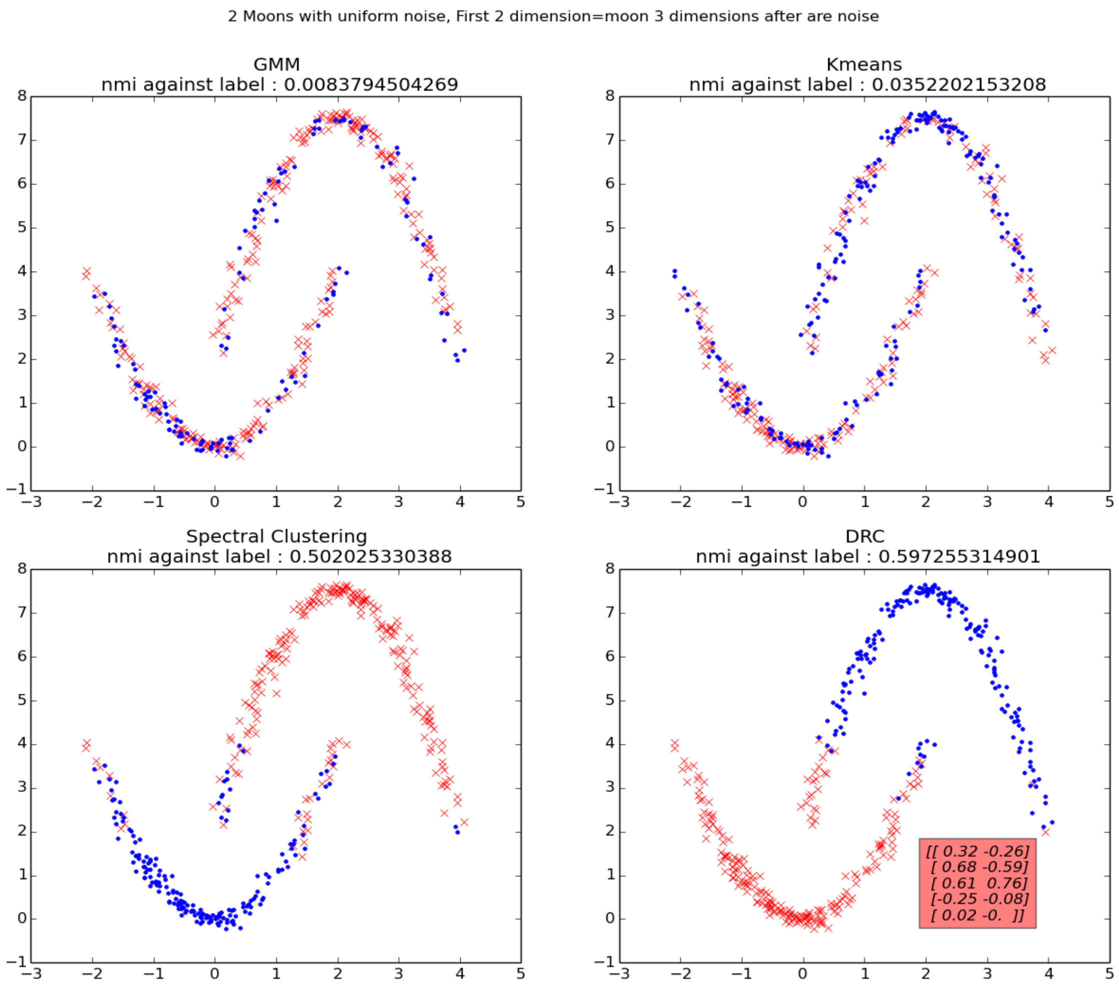


Figure 3.

Experiment 3

This experiment used the breast cancer data

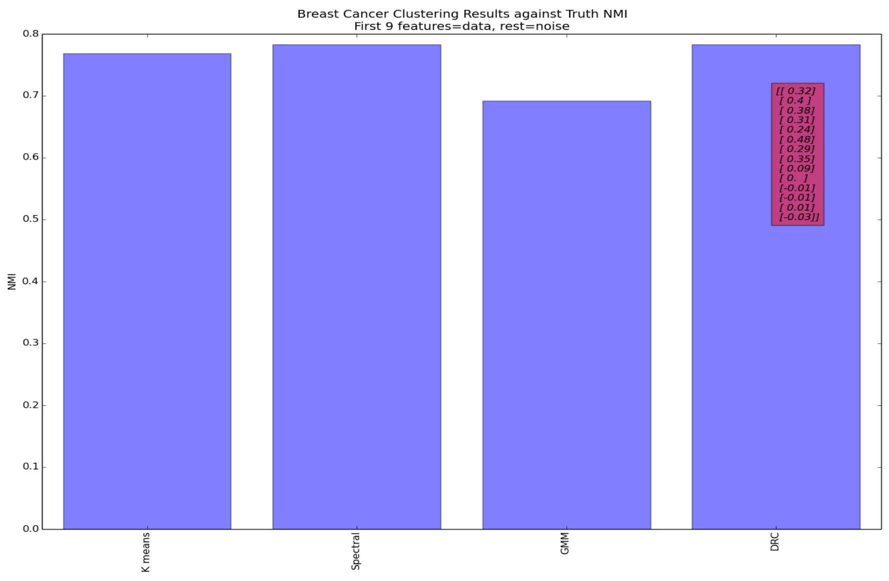


Figure 4.

Experiment 4

This experiment uses the facial data, the identity is the dominant pattern within the data.

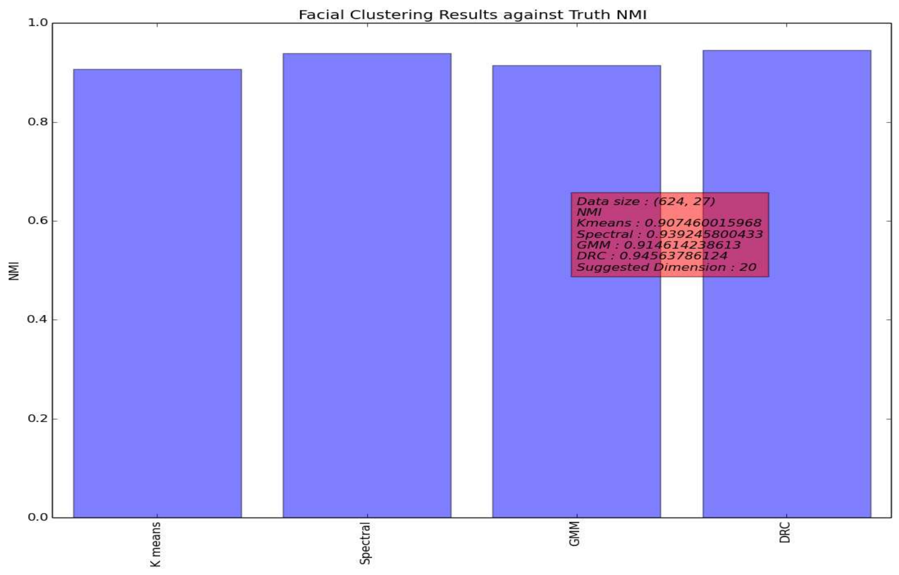


Figure 5.

This experiment uses the facial data, the pose is the less dominant pattern within the data.

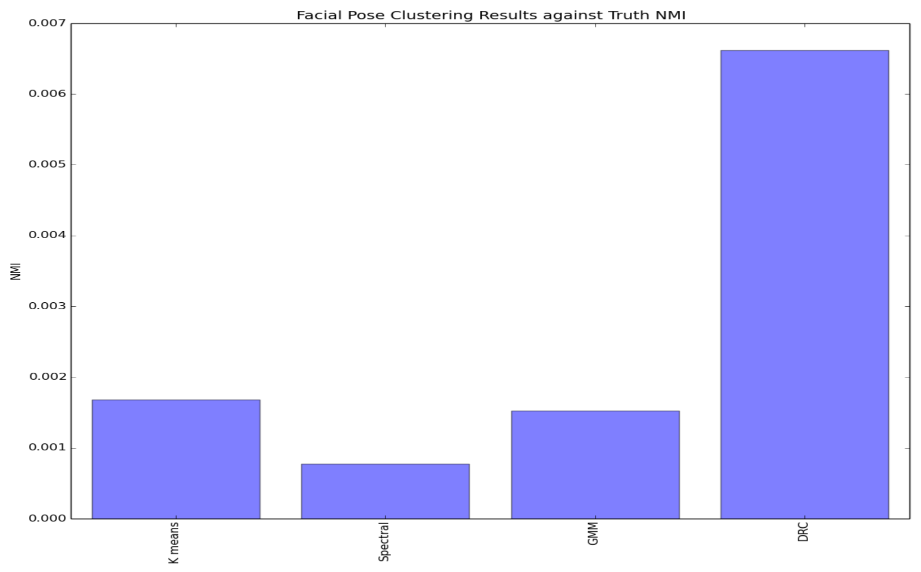
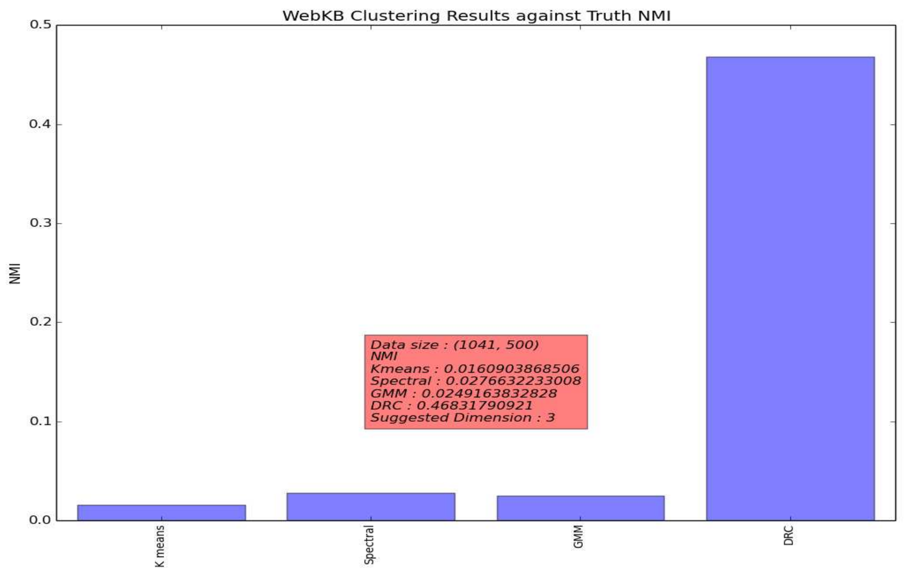


Figure 6.

This experiment used the webkb data, it is obvious that DRC performs the best. The reason for this is that the data had 500 features. As you can see, the more features, the better this algorithm performs.



Data size : (1041, 500)
NMI
Kmeans : 0.0160903868506
Spectral : 0.0276632233008
GMM : 0.0249163832828
DRC : 0.46831790921
Suggested Dimension : 3

Figure 7.

Future possibilities of kernelizing it :

$$\begin{array}{ll} \min\limits_{K,U} & \text{rank}(A) - \lambda \text{Tr}(KHUU^TH) \\ s.t & K_{i,j} = e^{-\left(\frac{\Delta x_{i,j}^T A \Delta x_{i,j}}{2\sigma^2}\right)} \\ & A = WW^T \\ & W^TW = I \\ & A \geq 0 \\ & A \in \mathbb{R}^{n \times n} \end{array}$$

Relaxation of rank.

$$\min_{A,U} \quad \text{Tr}((A_k + \delta I)^{-1} A) - \lambda \text{Tr}\left(HUU^THK\right)$$

$$\min_{A,U} \quad \text{Tr}(CA) - \lambda \text{Tr}\left(HUU^THK\right)$$

Let $Y^T = HUU^TH$, $A = WW^T$, we transform back into orthogonality constraint.

$$\min_{A,U} \quad \text{Tr}(W^TCW) - \sum_{i,j} \lambda Y_{i,j} e^{-\frac{\text{Tr}\left(W^TX_{i,j}W\right)}{2\sigma^2}} - \frac{1}{2}\text{Tr}(\Lambda(W^TW - I))$$

$$\mathcal{L} = \text{Tr}(W^TCW) - \sum_{i,j} \lambda Y_{i,j} e^{-\frac{\text{Tr}\left(W^TX_{i,j}W\right)}{2\sigma^2}} - \frac{1}{2}\text{Tr}(\Lambda(W^TW - I))$$

$$\nabla \mathcal{L} = 2CW + \sum_{i,j} \frac{\lambda}{\sigma^2} Y_{i,j} e^{-\frac{\text{Tr}\left(W^TX_{i,j}W\right)}{2\sigma^2}} X_{i,j} W - W\Lambda = 0$$

$$\left(2C + \sum_{i,j} \frac{\lambda}{\sigma^2} Y_{i,j} e^{-\frac{\text{Tr}\left(W^TX_{i,j}W\right)}{2\sigma^2}} X_{i,j}\right) W = W\Lambda$$

$$\Phi(W)\, W = W\Lambda$$

Also finding out the number of clusters

$$\begin{array}{ll} \min\limits_{A,U} & \text{rank}(A) + \rho\, \text{rank}(K_U) - \lambda \text{Tr}\left(KHK_UH\right) \\ s.t & K = XAX^T \\ & A \geq 0 \\ & A \in \mathbb{R}^{n \times n} \end{array}$$

Optimizing A is identical, but optimizing for U is now changed to optimize for the entire K_U matrix.

$$\rho\, \text{rank}(K_U) - \lambda(HXAX^THK_U)$$

$$\rho\, \text{rank}(K_U) - \lambda(\Phi K_U)$$

$$\rho \text{Tr}((K_{U_{k-1}} - \delta I)^{-1} K_U) - \lambda(\Phi K_U)$$

$$\text{Tr}([\rho(K_{U_{k-1}} - \delta I)^{-1} - \lambda \Phi] K_U)$$

This problem is now converted into a SDP problem.

$$\begin{array}{ll}\min & \mathrm{Tr}(WK_U) \\ K_U & \\ s.t & K_U \succeq 0\end{array}$$

By finding out the rank of K_U, we also know the number of clusters.