Robust Kernel Methods in Clustering and Dimensionality Reduction Problems

Jian Guo, Debadyuti Roy, Jing Wang
University of Michigan, Department of Statistics

1 Introduction

In this report we propose robust methods for kernel K-means clustering and kernel Principal component analysis (PCA). Often these methods tend to get sensitive towards outliers because of the use of ML estimators. Here we replace these estimators with robust M-estimators (Huber's), and thus make the methods more robust towards outliers. These robust versions can also be applied to non-convex clustering and non-linear PCA by using inner product kernels. The usual setup for K-means clustering and PCA are given below:

- *K-means clustering* : The goal of this method is to partition the data $x_1, x_2 \ldots, x_n \in \mathbb{R}^d$ into K different clusters, so that the points in the same cluster are more similar compared to points in the other cluster [4]. Here the squared Euclidean distance measure is adopted as the dissimilarity measure, i.e., $d_{i,j} = \|x_i - x_j\|^2$. The K-mean clustering method seeks to find the optimal cluster map C^* that minimizes the within cluster scatter, i.e.,

$$W(C^*) = \arg\min_C \sum_{k=1}^{K} \sum_{i:C(i)=k} \|x_i - \bar{x_k}\|^2, \tag{1}$$

  where $n_k = \sum_{i=1}^{n} \mathbf{1}_{\{C(i)=k\}}$ and $\bar{x_k} = \frac{1}{n_k} \sum_{i:C(i)=k} x_i$.
  The K-means clustering algorithm is :

  1. Initialize $\bar{x}_k, k = 1, 2, \ldots, K$
  2. Repeat
     - $C(i) = \arg\min_k \|x_i - \bar{x}_k\|^2$
     - $\bar{x_k} = 1/n_k \sum_{i:C(i)=k} x_i$

  until clusters do not change.

- *Principle Component Analysis (PCA)*: PCA is an orthogonal transformation of the coordinate system. Given data $x_1, x_2 \ldots, x_n \in \mathbb{R}^d$, we wish to find the directions (linear combinations of the d variables) $Y_1, \ldots, Y_d$, where

$$Y_j = A_j' X = \sum_{i=1}^{d} A_{ij} X_i,$$

Y_1 has maximum variance over all linear combinations, Y_2 has the largest variance over all linear combinations orthogonal to the direction of Y_1, and so on. It can be shown that the problem reduces to finding the eigenvalue decomposition of Σ, the covariance matrix of data, i.e., $A_j = V_j$, where $\Sigma = V \Lambda V'$ and V_j is the eigenvector corresponding to the jth largest eigenvalue of Σ.

2 Background

The Kernel k-means algorithm [1] is a generalization of the standard k-means algorithm described in the previous section. By mapping the data points to a higher dimensional space, kernel k-mean can discover clusters that are non-linearly separable in the input space. The weighted kernel k-means and its computational complexity is discussed in [2]. We adopt a method similar to robust kernel density estimation as discussed in [3].
The kernel PCA method is well discussed in [7]. We used the robust version of the mean and covariance matrix to implement the robust kernel PCA, as discussed in [5].

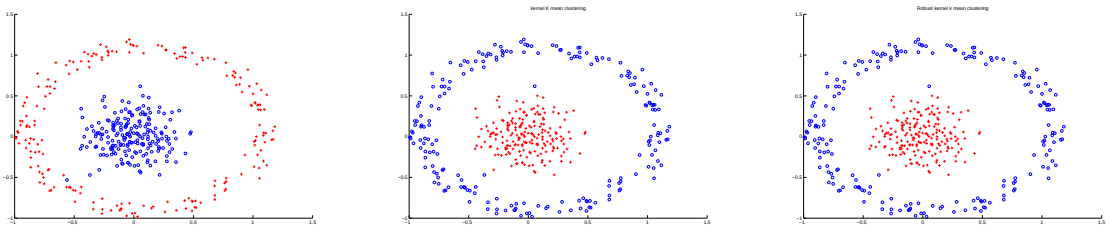
Robust M-estimators: Given iid data $x_1, x_2, \ldots, x_n \in \mathbb{R}^d$ an estimate of the data centroid can be $\hat{\theta} = \arg\min_\theta \sum_{i=1}^{n} \rho(\|x_i - \theta\|)$ where ρ is any suitable robust function. We can solve for $\hat{\theta}$ from

$$-\sum_{i=1}^{n} \left\{ (x_i - \hat{\theta}). \frac{\rho'(\|x_i - \hat{\theta}\|)}{\|x_i - \hat{\theta}\|} \right\} = 0 \tag{2}$$

The choice of the Huber's ρ function [5] would make the estimate robust and is consider in the sequel. On a separate note recently it is shown that the principal components are the continuous solutions to the discrete cluster membership indicators for K-means clustering [6].

3 Robust Kernel K-Mean Clustering

The data is first transformed using an inner product kernel. For an infinite dimensional Hilbert spaces $\mathcal{H}$, let us define a map $\Phi : \mathbb{R}^d \to \mathcal{H}$, and the corresponding inner product kernel $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ such that $K(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$. The K-means clustering algorithm now find the optimal cluster map C^* that minimizes (1) with x_i replaced by $\Phi(x_i)$



(a) (b) (c)

Figure 1: *(a) scatter plot of data with no noise, (b) K-Means clustering, (c) Robust Kernel K-means clustering*

and $\bar{x}_k$ replaced by $\bar{\Phi}_k := \frac{1}{n_k}\sum_{i:C(i)=k}\Phi(x_i)$. Thus in the algorithm for K-means clustering $C(i)$ can be written as

$$\begin{aligned} C(i) &= \arg\min_k \|\Phi(x_i) - \bar{\Phi}_k\|^2 \\ &= \arg\min_k \left(K(x_i,x_i) - \frac{2}{n_k}\sum_{j:C(j)=k} K(x_i,x_j) + \frac{1}{n_k^2}\sum_{i:C(i)=k}\sum_{j:C(j)=k} K(x_i,x_j) \right) \end{aligned} \tag{3}$$

where $n_k = \sum_{i=1}^n \mathbf{1}_{C(i)=k}$

To make the clustering procedure robust we replace the usual sample mean in (1) by a more robust estimate $\hat{m}_k$ of the cluster centroid in the high-dimensional feature space [3]. This can be achieved by replacing the usual squared loss ($\rho(t) = t^2$) by the Huber's ρ function. By the M-estimator criterion obtained in the preliminaries section, $\hat{m}_k$ satisfies

$$\sum_{i:C(i)=k}\left((\Phi(x_i) - \hat{m}_k).\frac{\rho'(\|\Phi(x_i) - \hat{m}_k\|)}{\|\Phi(x_i) - \hat{m}_k\|} \right) = 0$$

$$\Rightarrow \hat{m}_k = \frac{\sum_{i=1}^n \tilde{w}_i\Phi(x_i)}{\sum_{i=1}^n \tilde{w}_i} = \sum_{i=1}^n w_i\Phi(x_i)$$

where $\tilde{w}_i = \rho'(\|\Phi(x_i) - \hat{m}_k\|)/\|\Phi(x_i) - \hat{m}_k\|$ and $w_i = \tilde{w}_i/\sum_{i=1}^n \tilde{w}_i$

The M-estimators $\hat{m}_k$ can be computed using the iteratively re-weighted least squares (IRLS) method. The cluster map can be computed as

$$\begin{aligned} C(i) &= \arg\min_k \|\Phi(x_i) - \hat{m}_k\|^2 \\ &= \arg\min_k \|\Phi(x_i) - \sum_{j=1}^n w_j\Phi(x_j)\|^2 \\ &= \arg\min_k \left(K(x_i,x_i) - 2\sum_{j:C(j)=k} w_jK(x_i,x_j) + \sum_{i:C(i)=k}\sum_{j:C(j)=k} w_iw_jK(x_i,x_j) \right) \end{aligned} \tag{4}$$

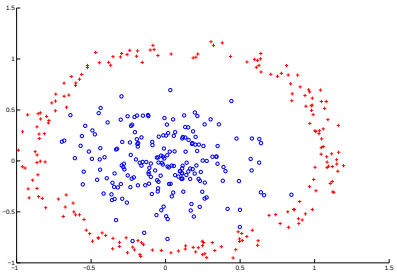
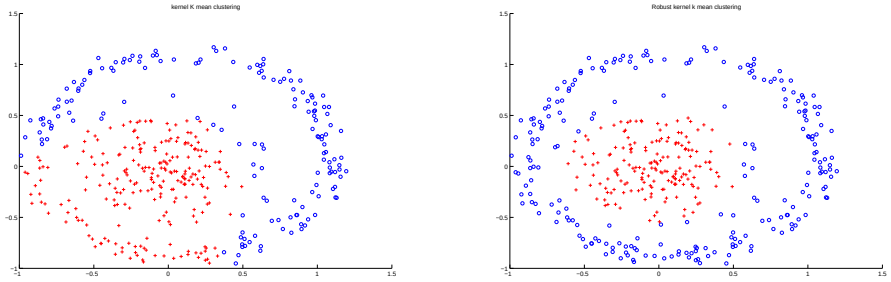


Figure 2: *(a) scatter plot of data with noise*



(a) (b)

Figure 3: *(a) kernel k-means clustering, (b) Robust kernel k-means clustering*

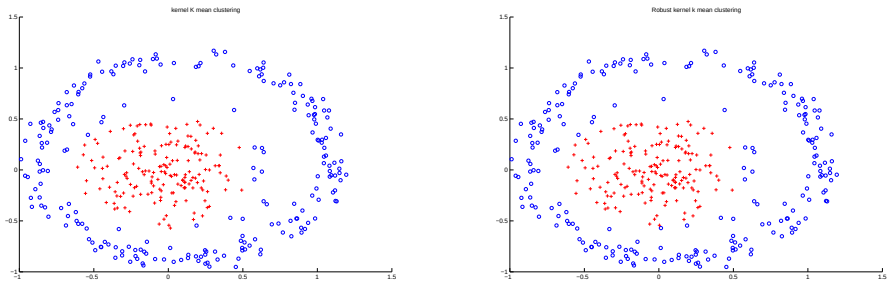
3.1 Experimental Results

We applied kernel k-means and robust kernel k-means on a simulated 2 dimensional data with one convex and another non-convex cluster (red and blue). Figure 1 shows their performance on data with no noise/outliers. The clustering solutions are the same for the two methods.

Figure 2 is the scatter plot of the data with high variance for the inner cluster. Figure 3 shows the performance of kernel k-means (partitioning the clusters by a hyperplane) and the robust version of it (correctly identifying two clusters) with random initialization. Note that a different random initialization may result in good clustering solution by kernel k-means algorithm (Figure 4).

Kernel k-means can produce both good and bad clustering solution for non-convex clusters depending on the initialization of the class labels, whereas the robust kernel k-means algorithm performs better in most cases.

As a method of comparing the clustering solution we have calculated the Sum of Squares of Error (SSE) by seldom assigning some class label to the clusters. The average SSE and the standard error of it is also calculated based on 100 simulations. The result is given is Table 1. We could not use the Silhoutte coefficient criterion (a usual clustering solution comparison method) here because the clusters used in our setup are nonconvex in nature. But the



(a) (b)

Figure 4: *(a) Kernel k-means clustering, (b) Robust kernel k-means clustering*

Error	Kernel K Means	Robust Kernel K Means
Clear data	0.0075	0.0075
Initial Setup 1	0.2050	0.0675
Initial Setup 2	0.0650	0.0675
Average	0.2082	0.1200
Standard Error	0.0106	0.0056

Table 1: *Comparison of kernel k-means and robust kernel k-means for different random initialization*

criterion used here is not really applicable in practice because the class labels are usually unknown. We have used random initialization of class labels and a Gaussian Kernel function with bandwidth 0.5 since it gives the smallest error in this setting. From our simulation study we found that the robust kernel k-mean method performs significantly better than the kernel k-means method irrespective of the initialization condition.

4 Robust Kernel PCA

We define a map $\Phi : \mathbb{R}^d \to \mathcal{H}$ to enrich the feature space. For the sake of simplicity, here we deal with centered data $\Phi(x_i) - \bar{\Phi}$ where $\bar{\Phi}$ is the usual sample mean $\frac{1}{n}\sum_{i=1}^{n}\Phi(x_i)$ or a more robust estimate $\hat{m}$, where $\hat{m}$ is computed as in Section 3. Using the robust estimate of the mean, the centered data is then $\tilde{\Phi}(x_i) := \Phi(x_i) - \hat{m}$. Now the inner product Kernel matrix can be defined as $K_{ij} = \langle\tilde{\Phi}(x_i), \tilde{\Phi}(x_j)\rangle$.

The inner product matrix K can be computed easily from K^{old}, the kernel with the enlarged data. If the robust mean $\hat{m} = \sum_{i=1}^{n} w_i\Phi(x_i)$, where w_i is exactly the same w_i as in K-means, is used, then the new inner product kernel matrix K is

$$K_{ij} = K_{ij}^{old} - \sum_{l=1}^{n} w_l K_{il}^{old} - \sum_{l=1}^{n} w_l K_{jl}^{old} + \sum_{l=1}^{n}\sum_{m=1}^{n} w_l K_{lm}^{old} w_m.$$

The covariance matrix Σ takes the form $\Sigma = \frac{1}{n}\sum_{j=1}^{n} \tilde{\Phi}(x_j)\tilde{\Phi}(x_j)'$. Now if $\mathcal{H}$ is infinite dimensional, $\tilde{\Phi}(x_j)\tilde{\Phi}(x_j)'$ can be thought of as a linear operator on $\mathcal{H}$ mapping $\mathbf{y} \mapsto \tilde{\Phi}(x_j)\langle\tilde{\Phi}(x_j), \mathbf{y}\rangle$.

We now need to find eigenvalues $\lambda \geq 0$ and nonzero eigenvectors $\mathbf{v} \in \mathcal{H}\backslash\{0\}$ satisfying

$$\lambda\mathbf{v} = \Sigma\mathbf{v} = \frac{1}{n}\sum \tilde{\Phi}(x_i)\langle\tilde{\Phi}(x_i), \mathbf{v}\rangle$$

Thus, all solutions $\mathbf{v}$ with $\lambda \neq 0$ lie in the span of $\tilde{\Phi}(x_1), \ldots, \tilde{\Phi}(x_n)$. So there exists coefficients $\alpha_1, \ldots, \alpha_n$, such that $\mathbf{v} = \sum_{i=1}^{n} \alpha_i\tilde{\Phi}(x_i)$. Using this result, for a fixed i, we have

$$\begin{aligned}
\langle\tilde{\Phi}(x_i), \lambda\mathbf{v}\rangle &= \lambda\sum_{j=1}^{n} \alpha_j\langle\tilde{\Phi}(x_i), \tilde{\Phi}(x_j)\rangle = \lambda\sum_{j=1}^{n} \alpha_j K_{ij} \text{ and also} \qquad (5)\\
\langle\tilde{\Phi}(x_i), \lambda\mathbf{v}\rangle &= \langle\tilde{\Phi}(x_i), \Sigma\mathbf{v}\rangle = \left\langle \tilde{\Phi}(x_i), \frac{1}{n}\sum_{j=1}^{n} \tilde{\Phi}(x_j)\langle\tilde{\Phi}(x_j), \mathbf{v}\rangle \right\rangle \\
&= \frac{1}{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \alpha_k\langle\tilde{\Phi}(x_i), \tilde{\Phi}(x_j)\rangle\langle\tilde{\Phi}(x_j), \tilde{\Phi}(x_k)\rangle \\
&= 1/n\sum_{j=1}^{n}\sum_{k=1}^{n} \alpha_k K_{ij}K_{jk}. \qquad (6)
\end{aligned}$$

Thus varying $i = 1, \ldots, n$ and using the above two equations we need to solve $n\lambda K\alpha = K^2\alpha$, or equivalently, solve the dual $n\lambda\alpha = K\alpha$. The principal components are eigenvectors of the Gram (Kernel) matrix which is the projection of the data in the principal directions of the covariance matrix. Here we also want to control the contribution of outliers in covariance matrix, thus we use the weighted feature vectors to estimate Σ, i.e. $\Sigma = \sum_{j=1}^{n} w_i^2\tilde{\Phi}(x_j)\tilde{\Phi}(x_j)'$. The corresponding centered kernel matrix becomes WKW, where W is a diagonal matrix with i^{th} entry w_i.

Thus Robust Kernel PCA algorithm is :

1. Compute the robust estimate of $\hat{m}$.
2. Find the kernel matrix K with centered data.
3. Diagonalize WKW to obtain the nonzero eigenvalues $\lambda_1, \ldots, \lambda_p$ and the corresponding eigenvectors $\alpha^1, \ldots, \alpha^p$. To ensure that the eigenvectors $\mathbf{v}^1, \ldots, \mathbf{v}^p$ for the enlarged feature space are normalized α^j must satisfy $\lambda_j\langle\alpha^j, \alpha^j\rangle = 1$.
4. Finally, to extract the principal components (corresponding to a kernel function k) of a test point x, we compute the projections onto the eigenvectors by

$$\langle\mathbf{v}^j, \Phi(x)\rangle = \sum_{i=1}^{n} \alpha_j k(x_i, x), j = 1, \ldots, p.$$

   Here $k(x_i, x)$ is the entry of WKW.

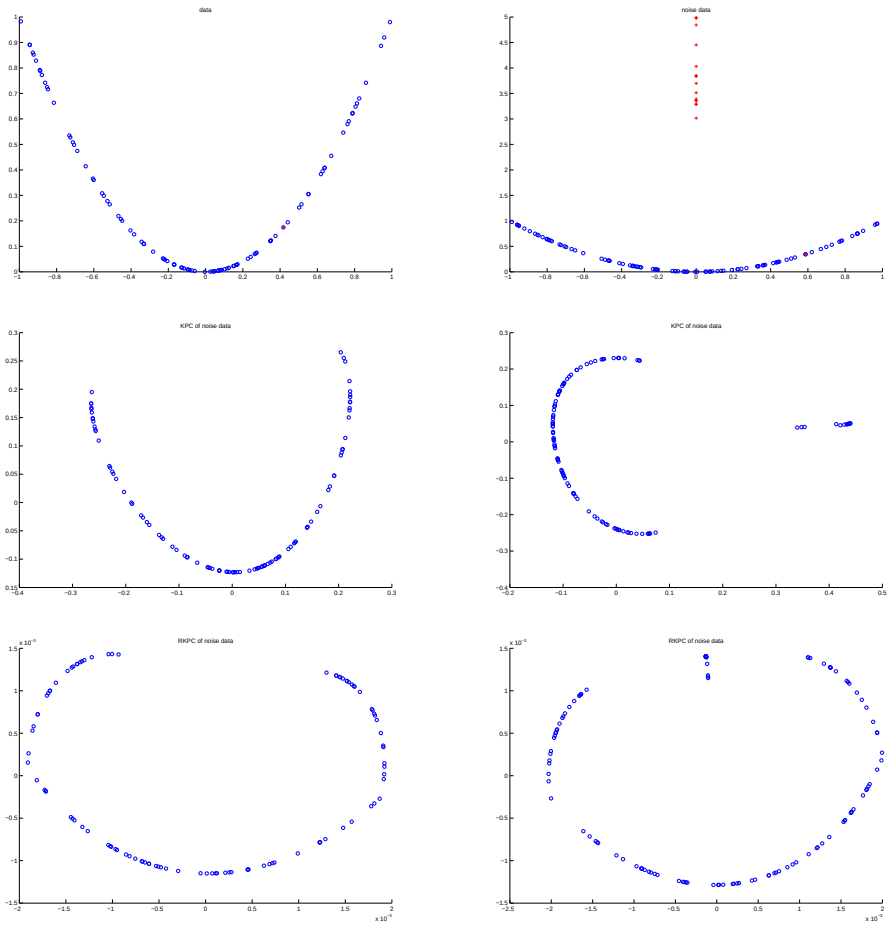


Figure 5: *Biplot of PC1 and PC2. Left top: simulated data without noise; Right top: simulated data with noise; Left middle: biplot with kernel PCA on the data without noise; Right middle: biplot with kernel PCA on the data with noise; Left bottom: biplot with robust kernel PCA on the data without noise; Right bottom: biplot with robust kernel PCA on the data with noise;*

4.1 Experimental Results

Figure 5 compares the performance of kernel PCA with robust kernel PCA on simulated data with noise and without noise respectively. As a method of goodness of fit and to quantitatively evaluate the performance of robust kernel PCA, we match the first PC scores of robust kernel PCA and those of kernel PCA on noised data with those of kernel PCA on the data without noise (treated as a baseline) using procrustes rotation, respectively. The Sum of Square Error from the Procrustes Rotation (SSEPR) was calculated for both matches. This procedure was repeated on 100 simulated data and SSEPR were averaged on these 100 experiment. SSEPR from the match of kernel PCA on noised data and kernel PCA on the data without noise is 0.4751 and SSEPR from the match of robust kernel PCA on noised data and kernel PCA on the data without noise is 0.0483. Therefore, robust kernel PCA outperforms kernel PCA in terms of SSEPR.

5 Modified Robust PCA

In the robust (kernel) PCA, we use a robust way to estimate the mean $\widehat{m}$ and Σ separately. As a matter of fact, we can use the robust $\widehat{m}$ and Σ to update the estimation of each other iteratively. The key point is we estimate w_i differently:

$$w_i = \frac{\psi(\sqrt{(x_i - \widehat{m})^T \Sigma^{-1} (x_i - \widehat{m})})}{(x_i - \widehat{m})^T \Sigma^{-1} (x_i - \widehat{m}))}$$

where ψ is the derivative of ρ.

We formulate the algorithm as follows:

1. Initialize

$$\begin{aligned} w_i &= 1/n, \quad 1 \le i \le n \\ \widehat{m} &= \sum_{i=1}^{n} w_i x_i \\ \widehat{\Sigma} &= (X - \widehat{m}\mathbf{1}_n)^T \begin{bmatrix} w_1^2 & & \\ & \ddots & \\ & & w_n^2 \end{bmatrix} (X - \widehat{m}\mathbf{1}_n) \end{aligned}$$

2. Update

$$\begin{aligned} w_i^{new} &= \frac{\psi(\sqrt{((x_i - \widehat{m}^{old})^T (\Sigma^{old})^{-1} (x_i - \widehat{m}^{old}))})}{(x_i - \widehat{m}^{old})^T (\Sigma^{old})^{-1} (x_i - \widehat{m}^{old}))} \\ \widehat{m}^{new} &= \sum_{i=1}^{n} w_i^{new} x_i \\ \widehat{\Sigma}^{new} &= (X - \widehat{m}^{new}\mathbf{1}_n)^T \begin{bmatrix} (w_1^{new})^2 & & \\ & \ddots & \\ & & (w_n^{new})^2 \end{bmatrix} (X - \widehat{m}^{new}\mathbf{1}_n) \end{aligned}$$

3. Repeat step 2 until convergence, i.e., stop the iteration when

$$\left| \sum_{i=1}^{n} \rho(\sqrt{(x_i - \widehat{m}^{new})^T (\Sigma^{new})^{-1} (x_i - \widehat{m}^{new})}) - \sum_{i=1}^{n} \rho(\sqrt{(x_i - \widehat{m}^{old})^T (\Sigma^{old})^{-1} (x_i - \widehat{m}^{old})}) \right| < \epsilon$$

4. Do eigen-decomposition for $\widehat{\Sigma}^{new}$

In above algorithm, ρ can be choosen as Hampel's function or Huber's function. For Huber's function, the parameter k is chosen as the median absolute deviation (MAD) which is calculated as follows:

$$k = median(x_i - \widehat{m})$$

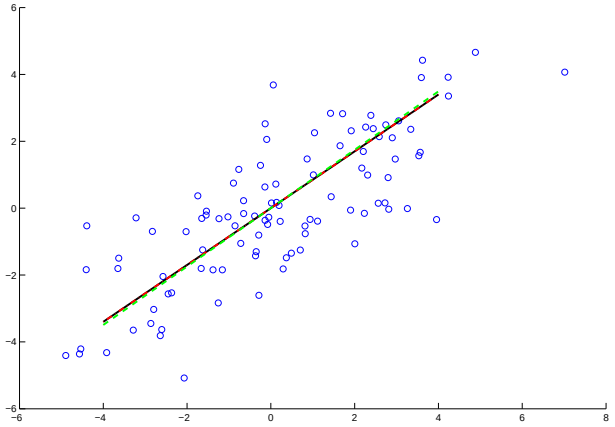


Figure 6: *Comparison of Robust PCA with PCA on clear data. Black solid line: first PC from the data without noise; green dashed line: first PC from robust PCA; red dashed dot line: first PC from PCA;*

where $\widehat{m}$ is the generalized geometric mean defined as

$$\widehat{m} = \arg\min_{m} \sum_{i=1}^{n} \|\Sigma^{-1/2}(x_i - m)\|$$

5.1 Experimental Results

We simulate 100 samples from a 2-D normal distribution. Figure 6 compares the modified robust PCA with PCA when no noise is added to the regular samples. It is not surprising that the first PC line from the two methods overlap. Figure 7 compares the modified robust PCA with PCA when the center of the noise is far from that of the regular samples as well as close to the regular samples. In both figures, the first PC line of PCA moves toward the noise samples significantly while that of the modified robust PCA only moves slightly (compared with the ground true black solid line). The results demonstrate the advantage of the modified robust PCA over PCA in terms of robustness to the noise.

6 Discussion and Conclusion

In this report, we proposed a method to improve the robustness of the regular kernel k-means and PCA by introducing the M-estimator. The performance of the robust kernel K-means and PCA are tested on simulated data sets. Now we discuss the advantages and limitations of our method.

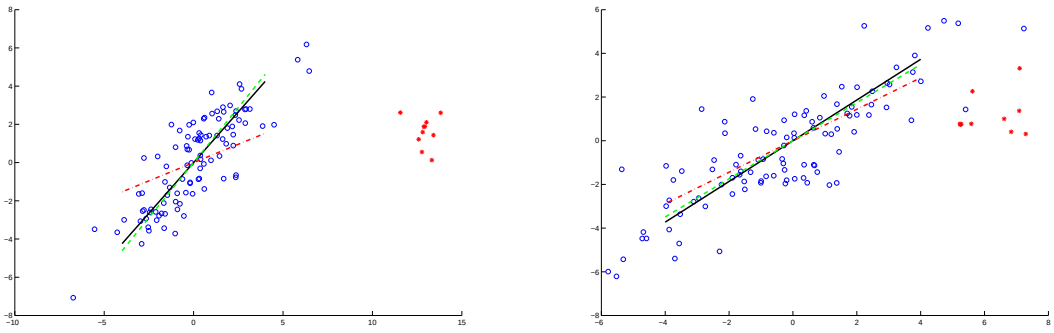


Figure 7: *Comparison of Robust PCA with PCA. (Left): when the center of the noise samples is far from data. (Right): when the center of the noise samples is close to data. Black solid line: first PC from the data without noise (ground true line); green dashed line: first PC from robust PCA; blue circle: data; red star: noise sample.*

6.1 Advantages and Limitations of Robust k-means

The simulation results show Robust kernel k-means performs better than kernel k-means in terms of the robustness to the noise. There are still some limitations in this method.

1. The performance of robust kernel k-means is highly dependent on the initialization. If we treat kernel k-means algorithm as a special case of the EM algorithm, it iteratively optimizes an implicit likelihood function. Due to the greedy nature of EM algorithm (the likelihood may not decrease after each iteration), kernel k-means only converges to some local optimal point which depends on the initialization condition. Occasionally if the likelihood is convex, the local optimal point is also the global optimal point.

2. In simulation, the performance of the robust kernel k-means is measured by comparing the true labels (seldom assigned) and labels given by the clustering solution. Unfortunately, in real data, we do not know the true labels, so the evaluation method for simulation data will not work for real data. A practical way is to use silhouette coefficient to evaluate the performance of convex clusters.

3. In practice, the performance of robust kernel k-means depends much on the choice of kernel function. A paradox is that robust kernel k-means may perform even worse under inappropriate kernel function.

6.2 Advantages and Limitations of Robust PCA and Modified Robust PCA

In this report, we introduced both a (kernel) robust PCA and a modified robust PCA. Due to the difficulty of formulation and the limitation of time, we did not study the kernelized version of the modified robust PCA. Now we discuss the performance of robust PCA.

1. Under gaussian kernel, robust kernel PCA generally outperforms kernel PCA in terms of SSEPR. We have to say, however, the choice of the kernel function significantly affect the performance of robust kernel PCA. In reality, how to choose an appropriate kernel function still needs to be studied.

2. The modified robust PCA updates a robust mean and a robust covariance matrix as the same time. The comparison between modified robust PCA has shown its advantage over PCA.

7 Team Effort

This project has truly been a joint effort by all of us. We had frequent meetings discussing the formulation/derivation of our method, running simulations and checking the results. Jian worked on the data generation code. Debi worked towards the code for k-means and Jing worked towards the code for 2 different PCA methods. Debi and Jian shared the work on the the report write up. The poster write up and the presentation is shared by all three of us.

References

[1] B. Scholkopf, A. Smola, and K.R. Muller, "Nonlinear component analysis as a kernel eigenvalue problem,?Neural Computation, vol. 10, pp. 1299?319, 1998.

[2] I. Dhillon, Y. Guan and B. Kulis, "Kernel k-means, Spectral clustering and Normalized Cuts", KDD?4, August 22?5, 2004, Seattle, Washinton, USA.

[3] J. Kim and C. Scott, "Robust Kernel Density Estimation", submitted to ICASSP 2008.

[4] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, New York, 2001.

[5] P. Huber, *Robust Statistics*, Wiley, New York, 1981.

[6] C. Ding and X. He, "K-means Clustering via Principal Component Analysis", Proceedings of the 21st International Conference on Machine Learning, Banff, Canada, 2004.

[7] B. Schoelkopf and A. Smola, *Learning with Kernels*, MIT Press, 2002