HOMEWORK 5

>>Nevindu M. Batagoda<<
>>9081677594<<

Instructions: Use this latex file as a template to develop your homework. Submit your homework on time as a single pdf file. Please wrap your code and upload to a public GitHub repo, then attach the link below the instructions so that we can access it. Answers to the questions that are not within the pdf are not accepted. This includes external links or answers attached to the code implementation. Late submissions may not be accepted. You can choose any programming language (i.e. python, R, or MATLAB). Please check Piazza for updates about the homework. It is ok to share the experiments results and compare them with each other.

1 Clustering

1.1 K-means Clustering (14 points)

1. **(6 Points)** Given n observations $X_1^n = \{X_1, \dots, X_n\}$, $X_i \in \mathcal{X}$, the K-means objective is to find k ($< n$) centres $\mu_1^k = \{\mu_1, \dots, \mu_k\}$, and a rule $f : \mathcal{X} \to \{1, \dots, K\}$ so as to minimize the objective

$$J(\mu_1^K, f; X_1^n) = \sum_{i=1}^{n} \sum_{k=1}^{K} \mathbb{1}(f(X_i) = k) \|X_i - \mu_k\|^2 \tag{1}$$

Let $\mathcal{J}_K(X_1^n) = \min_{\mu_1^K, f} J(\mu_1^K, f; X_1^n)$. Prove that $\mathcal{J}_K(X_1^n)$ is a non-increasing function of K.

2. **(8 Points)** Consider the K-means (Lloyd's) clustering algorithm we studied in class. We terminate the algorithm when there are no changes to the objective. Show that the algorithm terminates in a finite number of steps.

1. Assume that μ_1^K and f is the optimal solution for K clusters. Then we have,

$$J(\mu_1^K, f; X_1^n) = \sum_{i=1}^{n} \sum_{k=1}^{K} \mathbb{1}(f(X_i) = k) \|X_i - \mu_k\|^2 .$$

Now consider the case where we have $K+1$ clusters. Let μ_1^{K+1} be the optimal solution for $K+1$ clusters with the assignmet rule $\hat{f}$. Then we have,

$$J(\mu_1^{K+1}, f; X_1^n) = \sum_{i=1}^{n} \sum_{k=1}^{K+1} \mathbb{1}(f(X_i) = k) \|X_i - \mu_k\|^2 .$$

We can acheive the above clustering by keeping the $K-1$ clusters from μ_1^K and splitting the remaining cluster into two clusters. The objective for the $K-1$ clusters remains the same and also the new assignmet rule $\hat{f}$ will also remain the same. By splitting one cluster, we create two new cluster centers that will be much closer to the data points in the cluster. Therefore, the objective for $K+1$ clusters will be less than or same as the objective for K clusters. That is, when we increase the number of clusters, the objective will be less than or same as the previous objective. Therefore, $\mathcal{J}_K(X_1^n)$ is a non-increasing function of K.

2. The KMeans algorithm acts on a finite dataset on a finite space, and tries to minimize the leaset squares distance between the data points and the cluster centers. This is a convex objective function where zero is the global minimum, and at each update to the centroid location the objective function value decreases. Therefore, the algorithm will terminate in a finite number of steps either by acheiving the global minima or plateauing at a local minima.

1.2 Experiment (20 Points)

In this question, we will evaluate K-means clustering and GMM on a simple 2 dimensional problem. First, create a two-dimensional synthetic dataset of 300 points by sampling 100 points each from the three Gaussian distributions shown below:

$$P_a = \mathcal{N}\left(\begin{bmatrix}-1\\-1\end{bmatrix}, \sigma\begin{bmatrix}2, & 0.5\\0.5, & 1\end{bmatrix}\right), \quad P_b = \mathcal{N}\left(\begin{bmatrix}1\\-1\end{bmatrix}, \sigma\begin{bmatrix}1, & -0.5\\-0.5, & 2\end{bmatrix}\right), \quad P_c = \mathcal{N}\left(\begin{bmatrix}0\\1\end{bmatrix}, \sigma\begin{bmatrix}1 & 0\\0, & 2\end{bmatrix}\right)$$

Here, σ is a parameter we will change to produce different datasets.

First implement K-means clustering and the expectation maximization algorithm for GMMs. Execute both methods on five synthetic datasets, generated as shown above with $\sigma \in \{0.5, 1, 2, 4, 8\}$. Finally, evaluate both methods on *(i)* the clustering objective (1) and *(ii)* the clustering accuracy. For each of the two criteria, plot the value achieved by each method against σ.

Guidelines:

- Both algorithms are only guaranteed to find only a local optimum so we recommend trying multiple restarts and picking the one with the lowest objective value (This is (1) for K-means and the negative log likelihood for GMMs). You may also experiment with a smart initialization strategy (such as kmeans++).
- To plot the clustering accuracy, you may treat the 'label' of points generated from distribution P_u as u, where $u \in \{a, b, c\}$. Assume that the cluster id i returned by a method is $i \in \{1, 2, 3\}$. Since clustering is an unsupervised learning problem, you should obtain the best possible mapping from $\{1, 2, 3\}$ to $\{a, b, c\}$ to compute the clustering objective. One way to do this is to compare the clustering centers returned by the method (centroids for K-means, means for GMMs) and map them to the distribution with the closest mean.

Points break down: 7 points each for implementation of each method, 6 points for reporting of evaluation metrics.

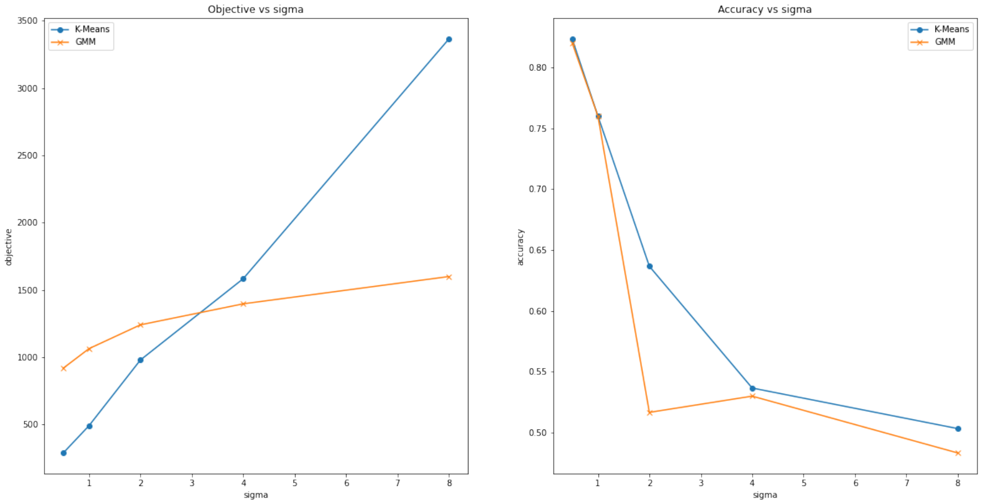

Figure 1: KMeans Clustering Vs GMM Clustering

2 Linear Dimensionality Reduction

2.1 Principal Components Analysis (10 points)

Principal Components Analysis (PCA) is a popular method for linear dimensionality reduction. PCA attempts to find a lower dimensional subspace such that when you project the data onto the subspace as much of the information is preserved. Say we have data $X = [x_1^\top; \dots; x_n^\top] \in \mathbb{R}^{n\times D}$ where $x_i \in \mathbb{R}^D$. We wish to find a d ($< D$) dimensional subspace $A = [a_1, \dots, a_d] \in \mathbb{R}^{D\times d}$, such that $a_i \in \mathbb{R}^D$ and $A^\top A = I_d$, so as to maximize $\frac{1}{n}\sum_{i=1}^n \|A^\top x_i\|^2$.

1. **(4 Points)** Suppose we wish to find the first direction a_1 (such that $a_1^\top a_1 = 1$) to maximize $\frac{1}{n}\sum_i (a_1^\top x_i)^2$. Show that a_1 is the first right singular vector of X.
2. **(6 Points)** Given $a_1, \dots, a_k$, let $A_k = [a_1, \dots, a_k]$ and $\tilde{x}_i = x_i - A_k A_k^\top x_i$. We wish to find a_{k+1}, to maximize $\frac{1}{n}\sum_i (a_{k+1}^\top \tilde{x}_i)^2$. Show that a_{k+1} is the $(k+1)^{th}$ right singular vector of X.

1. Finding the first direction amounts to solving the following optimization problem:

$$\begin{aligned} \max_{a_1} \quad & \frac{1}{n}\sum_{i=1}^n (a_1^\top x_i)^2 \\ \text{s.t.} \quad & a_1^\top a_1 = 1\,. \end{aligned}$$

   We can define this as a Lagrangian L with multiplier λ as follow:

$$L(a_1, \lambda) = \frac{1}{n}\sum_{i=1}^n (a_1^\top x_i)^2 - \lambda(a_1^\top a_1 - 1)\,.$$

   To find the optimal a_1, we can take the derivative of L with respect to a_1 and set it to zero:

$$\begin{aligned} \frac{\partial L}{\partial a_1} &= \frac{2}{n}\sum_{i=1}^n x_i(x_i^\top a_1) + 2\lambda a_1 = 0 \\ &= \frac{1}{n}\sum_{i=1}^n x_i(x_i^\top a_1) - \lambda a_1 = 0 \\ &= \frac{1}{n}\sum_{i=1}^n x_i(x_i^\top a_1) = \lambda a_1 \\ &\Rightarrow \frac{1}{n}X^\top X a_1 = \lambda a_1\,. \end{aligned}$$

   Note that, assuming X is zero-mean centered, the component $\frac{1}{n}X^T X$ is propotional to the sample covariance. As such, we need to find a_1 that maximizes $\frac{1}{n}X^T X$. Also, note that above is formatted as an eigenvalue problem as such, a_1 is the eigenvector that maximizes $\frac{1}{n}X^T X$. Since the eigenvalues of $X^\top X$ are related to the singular values of X, we can further simplify the above equation by taking the singular value decomposition of X:

   Let $X = U\Sigma V^\top$ be the singular value decomposition of X. Then we have

$$\frac{1}{n}X^\top X a_1 = \frac{1}{n}V\Sigma^2 V^\top a_1 = \lambda a_1\,.$$

   Note from above the eigenvectors V of $X^\top X$ are the right singular vectors of X. And since the singular values of Σ are ordered in decending order, the first right singular vector is associated with the largest eigenvalue, which is what we are trying to maximize. Therefore, the first direction a_1 is the first right singular vector of X.

2.

2.2 Dimensionality reduction via optimization (22 points)

We will now motivate the dimensionality reduction problem from a slightly different perspective. The resulting algorithm has many similarities to PCA. We will refer to method as DRO.

As before, you are given data $\{x_i\}_{i=1}^n$, where $x_i \in \mathbb{R}^D$. Let $X = [x_1^\top; \dots x_n^\top] \in \mathbb{R}^{n\times D}$. We suspect that the data actually lies approximately in a d dimensional affine subspace. Here $d < D$ and $d < n$. Our goal, as in PCA, is

to use this dataset to find a d dimensional representation z for each $x \in \mathbb{R}^D$. (We will assume that the span of the data has dimension larger than d, but our method should work whether $n > D$ or $n < D$.)
Let $z_i \in \mathbb{R}^d$ be the lower dimensional representation for x_i and let $Z = [z_1^\top; \dots; z_n^\top] \in \mathbb{R}^{n \times d}$. We wish to find parameters $A \in \mathbb{R}^{D \times d}$, $b \in \mathbb{R}^D$ and the lower dimensional representation $Z \in \mathbb{R}^{n \times d}$ so as to minimize

$$J(A, b, Z) = \frac{1}{n}\sum_{i=1}^{n} \|x_i - Az_i - b\|^2 = \|X - ZA^\top - \mathbf{1}b^\top\|_F^2. \tag{2}$$

Here, $\|A\|_F^2 = \sum_{i,j} A_{ij}^2$ is the Frobenius norm of a matrix.

1. **(3 Points)** Let $M \in \mathbb{R}^{d \times d}$ be an arbitrary invertible matrix and $p \in \mathbb{R}^d$ be an arbitrary vector. Denote, $A_2 = A_1M^{-1}$, $b_2 = b_1 - A_1M^{-1}p$ and $Z_2 = Z_1M^\top + \mathbf{1}p^\top$. Show that both (A_1, b_1, Z_1) and (A_2, b_2, Z_2) achieve the same objective value J (2).

Therefore, in order to make the problem determined, we need to impose some constraint on Z. We will assume that the z_i's have zero mean and identity covariance. That is,

$$\bar{Z} = \frac{1}{n}\sum_{i=1}^{n} z_i = \frac{1}{n}Z^\top\mathbf{1}_n = 0, \qquad S = \frac{1}{n}\sum_{i=1}^{n} z_i z_i^\top = \frac{1}{n}Z^\top Z = I_d$$

Here, $\mathbf{1}_d = [1, 1 \dots, 1]^\top \in \mathbb{R}^d$ and I_d is the $d \times d$ identity matrix.

2. **(16 Points)** Outline a procedure to solve the above problem. Specify how you would obtain A, Z, b which minimize the objective and satisfy the constraints.

   Hint: The rank k approximation of a matrix in Frobenius norm is obtained by taking its SVD and then zeroing out all but the first k singular values.

3. **(3 Points)** You are given a point x_* in the original D dimensional space. State the rule to obtain the d dimensional representation z_* for this new point. (If x_* is some original point x_i from the D–dimensional space, it shoud be the d–dimensional representation z_i.)

1. Consider the objective function $J(A,_2 b_2, Z_2)$, we have,

$$\begin{aligned}
J(A_2, b_2, Z_2) &= \|X - Z_2A_2^\top - \mathbf{1}b_2^\top\|_F^2 \\
&= \|X - (Z_1M^\top + \mathbf{1}p^\top)(A_1M^{-1})^\top - \mathbf{1}(b_1 - A_1M^{-1}p)^\top\|_F^2 \quad \text{(Substituting)} \\
&= \|X - (Z_1M^\top + \mathbf{1}p^\top)(M^{-1})^\top A_1^\top - \mathbf{1}(b_1 - A_1M^{-1}p)^\top\|_F^2 \\
&= \|X - Z_1M^\top(M^{-1})^T A_1^T - \mathbf{1}p^T(M^{-1})^T A_1^T - \mathbf{1}b^T + \mathbf{1}p^T(M^{-1})^T A_1^T\|_F^2 \\
&= \|X - Z_1A_1^\top - \mathbf{1}b^\top\|_F^2 \quad \text{(Since } M \text{ is invertible)} \\
&= J(A_1, b_1, Z_1)\,.
\end{aligned}$$

2. Consider the following optimization problem:

$$\begin{aligned}
\min_{A,b,Z} \quad & J(A, b, Z) = \frac{1}{n}\sum_{i=1}^{n} \|x_i - Az_i - b\|^2 = \|X - ZA^\top - \mathbf{1}b^\top\|_F^2. \\
\text{s.t.} \quad & \frac{1}{n}Z^\top\mathbf{1} = 0 \\
& \frac{1}{n}Z^\top Z = I_d
\end{aligned}$$

Take the derivative of J with respect to b and set them to zero to obtain the optimal b, we have

$$\begin{aligned}
\frac{\partial J}{\partial b} &= \frac{2}{n}\sum_{i=1}^{n}(x_i - Az_i - b) = 0 \\
\Rightarrow b &= \frac{1}{n}\sum_{i=1}^{n}(x_i - Az_i)\,.
\end{aligned}$$

Because we constrain Z to have zero mean, we have $\sum_{i=1}^{n}(Az_i) = 0$. Therefore, we have

$$b = \frac{1}{n}\sum_{i=1}^{n} x_i \, .$$

That is, the optimal b is the mean of the data.

Now let y be the zero-centered data, i.e., $Y = X - b$. Then we have

$$J(A, b, Z) = \|Y - ZA^\top\|_F^2 \, .$$

Take the derivative of J with respect to A and set them to zero to get the optimal A, we have

$$\frac{\partial J}{\partial A} = \frac{2}{n}\sum_{i=1}^{n}(Y_i - Z_i A^\top)Z_i = 0$$
$$\Rightarrow A = \left(\sum_{i=1}^{n} Z_i Z_i^\top\right)^{-1}\left(\sum_{i=1}^{n} Y_i Z_i^\top\right) .$$

From the constraints we know the covariance matrix of Z is I_d. Therefore, we have

$$A = \left(\sum_{i=1}^{n} Y_i Z_i^\top\right) = Y Z^\top$$

Rearrange the above equation, we have

$$Y = ZA^\top \, .$$

Now take the truncated singular value decomposition of Y to obtain the rank-d decomposition of Y, we have

$$Y \approx Y_d = U_d \Sigma_d V_d^\top \, .$$

Where, U_d is the $n \times d$ matrix of the first d left singular vectors of Y, Σ_d is the $d \times d$ diagonal matrix of the first d singular values of Y, and V_d is the $d \times D$ matrix of the first k right singular vectors of Y.

From the above two equations, we have

$$ZA^\top \approx Y_d = U_d \Sigma_d V_d^\top \, .$$

We can choose $Z = U_d$ as the lower dimensional representation and $A = \Sigma_d V_d^\top$ to minimize the objective function J. These choices of Z and A will also satisfy the constraints.

2.3 Experiment (34 points)

Here we will compare the above three methods on two data sets.

- We will implement three variants of PCA:
  1. "buggy PCA": PCA applied directly on the matrix X.
  2. "demeaned PCA": We subtract the mean along each dimension before applying PCA.
  3. "normalized PCA": Before applying PCA, we subtract the mean and scale each dimension so that the sample mean and standard deviation along each dimension is 0 and 1 respectively.
- One way to study how well the low dimensional representation Z captures the linear structure in our data is to project Z back to D dimensions and look at the reconstruction error. For PCA, if we mapped it to d dimensions via $z = Vx$ then the reconstruction is $V^\top z$. For the preprocessed versions, we first do this and then reverse the preprocessing steps as well. For DRO we just compute $Az + b$. We will compare all methods by the reconstruction error on the datasets.

- Please implement code for the methods: Buggy PCA (just take the SVD of X) , Demeaned PCA, Normalized PCA, DRO. In all cases your function should take in an $n \times d$ data matrix and d as an argument. It should return the the d dimensional representations, the estimated parameters, and the reconstructions of these representations in D dimensions.
- You are given two datasets: A two Dimensional dataset with 50 points `data2D.csv` and a thousand dimensional dataset with 500 points `data1000D.csv`.
- For the $2D$ dataset use $d = 1$. For the $1000D$ dataset, you need to choose d. For this, observe the singular values in DRO and see if there is a clear "knee point" in the spectrum. Attach any figures/ Statistics you computed to justify your choice.
- For the $2D$ dataset you need to attach the a plot comparing the orignal points with the reconstructed points for all 4 methods. For both datasets you should also report the reconstruction errors, that is the squared sum of differences $\sum_{i=1}^{n} \|x_i - r(z_i)\|^2$, where x_i's are the original points and $r(z_i)$ are the D dimensional points reconstructed from the d dimensional representation z_i.
- **Questions:** After you have completed the experiments, please answer the following questions.
  1. Look at the results for Buggy PCA. The reconstruction error is bad and the reconstructed points don't seem to well represent the original points. Why is this?
     Hint: Which subspace is Buggy PCA trying to project the points onto?
  2. The error criterion we are using is the average squared error between the original points and the reconstructed points. In both examples DRO and demeaned PCA achieves the lowest error among all methods. Is this surprising? Why?
- Point allocation:
  - Implementation of the three PCA methods: **(6 Points)**
  - Implementation of DRO: **(6 points)**
  - Plots showing original points and reconstructed points for 2D dataset for each one of the 4 methods: **(10 points)**
  - Implementing reconstructions and reporting results for each one of the 4 methods for the 2 datasets: **(5 points)**
  - Choice of d for $1000D$ dataset and appropriate justification: **(3 Points)**
  - Questions **(4 Points)**

1. Plots for the 2D dataset:

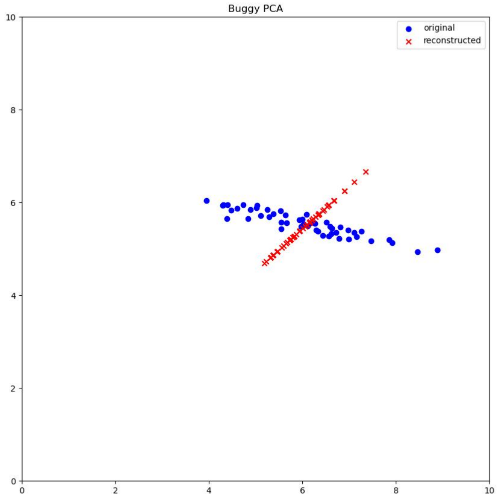

Figure 2: Buggy PCA

–

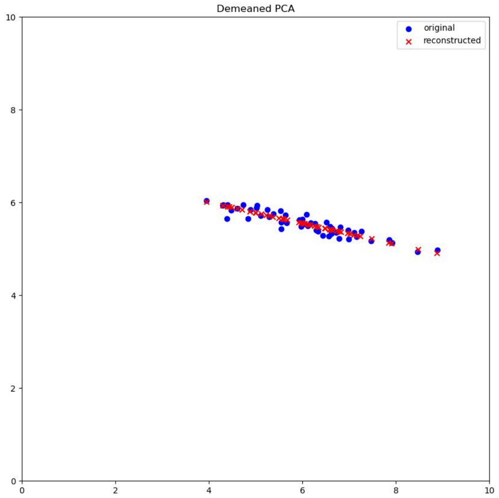

Figure 3: Demeaned PCA

–

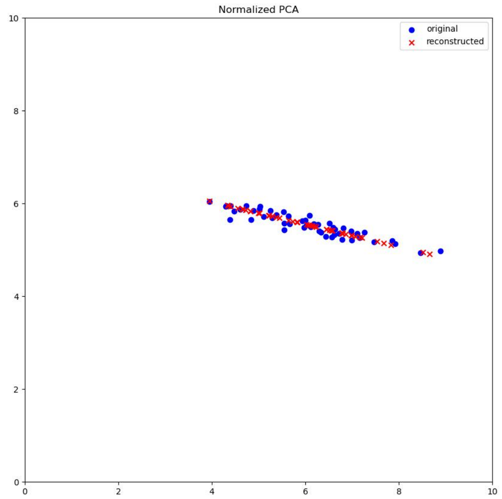

Figure 4: Normalized PCA

-

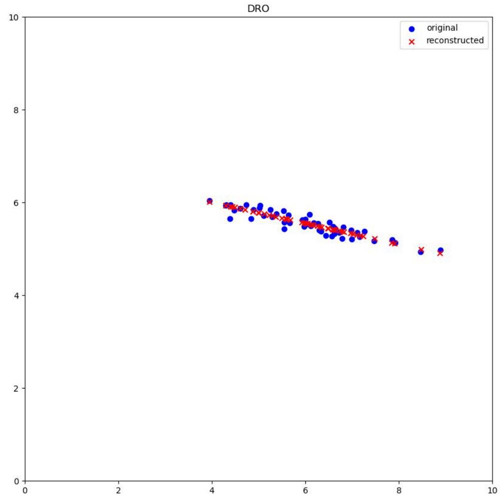

Figure 5: DRO

-

2. Choosing d for the 1000D dataset:

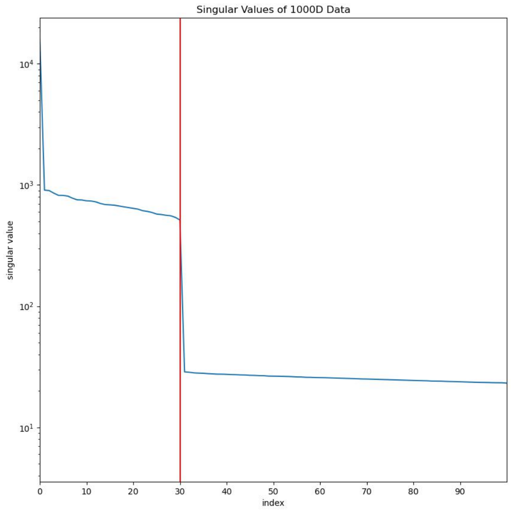

Figure 6: Singular Values

–

From the above figure, we can see that the singular values drop off significantly after the 30th singular value creating a ”knee point” in the graph.

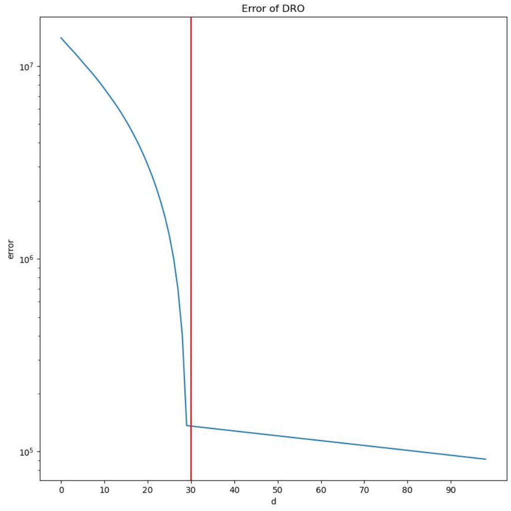

Figure 7: Singular Values

I also plot the DRO error rate for values of $d = 1...100$, and we can again see a clear ”knee point” at $d = 30$. From the above two plots, we can conclude that the optimal value for d is 30.

3. Reconstruction errors for the two datasets.

Dataset /Method	Buggy PCA	Demeaned PCA	Normalized PCA	DROP
2D	44.36	0.50	2.47	0.50
1000D (d = 31)	136384.97	135697.80	136029.46	135697.80

Table 1: Reconstruction Errors

4. a. In the buggy PCA, we are doing PCA on a dataset that has not been centered. Therefore, the subspace that we are projecting the points onto is not the subspace that captures the most variance in the data. That is, the principal components are not in the direction of the maximum variance. This is because the assumption that XX^T is the covariance matrix is only true when the data is centered. Therefore, the reconstructed points will not be close to the original points.

   b. We acheive the lowest error for DROP and demeaned PCA because we center the data before doing PCA. Hence, the subspace that we are projecting the points onto the subspace that captures the most variance in the data. Therefore, the reconstructed points will be close to the original points.