HOMEWORK 5

>>Hemanth Sridhar Nakshatri<<
>>nakshatri@wisc.edu, 9085807346<<

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 $X_1^n = \{X_1, \ldots, X_n\}$, $X_i \in \mathcal{X}$, the K-means objective is to find k ($< n$) centres $\mu_1^k = \{\mu_1, \ldots, \mu_k\}$, and a rule $f : \mathcal{X} \to \{1, \ldots, K\}$ 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.

- Consider two cases with *K* and *K+1* clusters each.
- For *K* Clusters, it is given that $J_K(X_n) = min_{\mu^K, f} J(\mu^K, f; X^n)$, i.e a certain minimum sum of sqaured distances.
- If we add one more cluster, i.e *K+1* case, one of the clusters is split into two to minimize the sum of sqaured distances even more.
- However, since the new centroid is at least as close to the points in its new cluster as the old centroid was, the division cannot increase the sum of sqaured distances.
- Hence $J_{K+1}(X^n) \leq J_K(X^n)$
- Thus as *K* increases, $J_K(X^n)$ does not increase meaning it is a non-increasing function.

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.

- For *n* data points and *K* clusters, the total number of possible partitions is a finite number (It might be huge number though).
- In K-Means algorithm, every iteration tends to reduce the objective function *J*, unless it has reached a possible local minimum. If it's in a local minimum, change in the partition would increase the squared distances *J*.

Since the objective function *J* cannot increase and is lower bounder by 0 in K-Means, and there's only a finite number of ways to split the data points, the algorithm will reach a local minimum in finite steps. Once the descent stops, the algorithm will terminate.

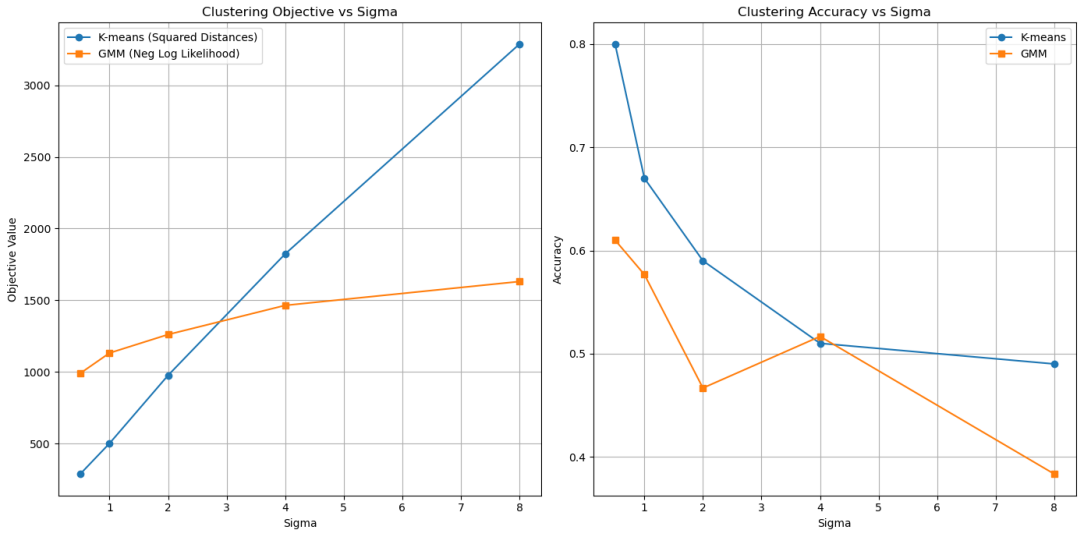

Figure 1: Clustering Objective and Clustering Accuracy plots

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.

- Five datasets are generated with $\sigma = \{0, 1, 2, 4, 8\}$.
- K-Means clustering and expectation maximization algorithm for GMMs are implemented from scracth and tested on the generated datasets.
- Clustering objective and Clustering accuracy are calculated for each case and plotted.
- Clustering objective increases significantly faster for K-Means clustering with increase in σ as compared to the GMM and can be seen in Fig. 1.
-

- From Fig. 1, it can be seen that the squared distances and the negative log likelihood increases with increase in the standard deviation of the data.
- Consecutively, the accuracy decreases due to the increased deviations in the data.
- ***Note that the accuracy and the objective changes with different initalizations of the cluster centroids. The centroids are intitialized using kmeans++ (Atleast what I consider my imference of kmeans++)***

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 $X = [x_1^\top; \ldots; x_n^\top] \in \mathbb{R}^{n \times D}$ where $x_i \in \mathbb{R}^D$. We wish to find a d ($< D$) dimensional subspace $A = [a_1, \ldots, a_d] \in \mathbb{R}^{D \times d}$, 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.

   Variance of the projected data onto a_1 is given by,

   $$Var(Xa_1) = \frac{1}{n} \sum_{i=1}^{n} (a_1^T x_i)^2$$

   Here, x_i is the *i*-th column of X.

   We need to find, $\max_{a1}(a_1^T X^T X a_1)$ where $a^T a = 1$.

   Consider a Lagrangian multiplier λ. Then the Lagrangian $\mathcal{L}$ is given by,

   $$\mathcal{L}(a_1, \lambda) = a_1^T X^T X a_1 - \lambda a_1^T a_1$$

   To find maximum, derivative $\frac{\partial \mathcal{L}}{\partial a_1} = 2X^T X a_1 - 2\lambda a_1 = 0$

   This simplifies to the eigenvalue equation: $X^T X a_1 = \lambda a_1$

   The direction $a1$ which maximizes the expression is the eigenvector corresponding to the largest eigenvalue of $X^T X$ which is also the first right singular vector of X.

2. **(6 Points)** Given $a_1, \ldots, a_k$, let $A_k = [a_1, \ldots, a_k]$ 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.

   Given the first k principal components, $A_k = [\alpha_1, \ldots, \alpha_k]$, and the residual vector $x_i - A_k A_k^T x_i$, to find α_{k+1} such that:

   $$\max_{\alpha_{k+1}} \sum_{i=1}^{n} \left(\alpha_{k+1}^T (x_i - A_k A_k^T x_i)\right)^2.$$

   Again, subject to the constraint $\alpha_{k+1}^T \alpha_{k+1} = 1$ and orthogonality to previous components $A_k^T \alpha_{k+1} = 0$. The Lagrangian in this case is:

   $$\mathcal{L}(\alpha_{k+1}, \Lambda) = \alpha_{k+1}^T X^T (I - A_k A_k^T) X \alpha_{k+1} - \Lambda(\alpha_{k+1}^T \alpha_{k+1} - 1).$$

   Differentiating with respect to α_{k+1} and setting to zero gives:

   $$\frac{\partial \mathcal{L}}{\partial \alpha_{k+1}} = 2X^T (I - A_k A_k^T) X \alpha_{k+1} - 2\Lambda \alpha_{k+1} = 0.$$

   Which leads to the generalized eigenvalue problem:

   $$X^T (I - A_k A_k^T) X \alpha_{k+1} = \Lambda \alpha_{k+1}.$$

   Solving this yields α_{k+1}, the $(k+1)$-th right singular vector of X.

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, $A_2 = A_1 M^{-1}$, $b_2 = b_1 - A_1 M^{-1} p$ and $Z_2 = Z_1 M^\top + \mathbf{1}p^\top$. Show that both (A_1, b_1, Z_1) and (A_2, b_2, Z_2) achieve the same objective value J (2).

Given: Objective function:

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

$A_2 = A_1 M^{-1}, b_2 = b_1 - A_1 M^{-1} p$ and $Z_2 = Z_1 M^T + 1p^T$.

Substituting $A_2, b_2, and Z_2$ into $J(A_2, b_2, Z_2)$,
$J(A_2, b_2, Z_2) = \frac{1}{2}||X - A_2 Z_2 - 1b_2^T||$
$= \frac{1}{2}||X - A1M^{-1}(Z_1 M^T - 1p^T) - 1(b_1 - A_1 M^{-1} p^T)||_F^2$
$= \frac{1}{2}||X - A_1 M^{-1} Z_1 M^T - A_1 M^{-1} p^T - 1b_1^T + 1(A_1 M^{-1} p)^T||_F^2$
After cancelling out the terms, the equation reduces to,

$$J(A_2, b_2, Z_2) = \frac{1}{2}||X - A_1 Z_1 - 1b_1^T||_F^2 = J(A_1, b_1, Z_1)$$

This shows that $J(A_2, b_2, Z_2) = J(A_1, b_1, Z_1)$.

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.

   To minimize J (i.e, minimize the error for $X = AZ + b$)
   Assumptions: Z has zero mean and Identity covariance.

   - Move the data to the center. This is done by subtracting the mean of the data $\mu = \frac{1}{n} \sum_{i=1}^{n} x_i$.
   - Thus the centered data would be $\tilde{x_i} = x_i - \mu$.
   - Compute the SVD on the centered data matrix $\widetilde{X} = U \Sigma V^T$.

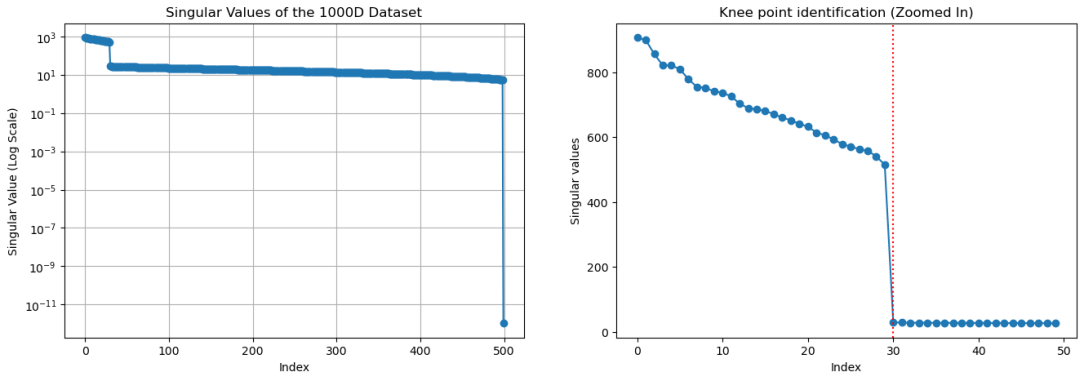

Figure 2: Knee point

- For d dimensions, the first d columns of matrix V form the matrix A (since the V contains/represents the principal components.)
- The product of first d columns of U and the largest d singular values forms Z matrix. $Z = U_d \Sigma_d$.
- The vector b is given the mean μ of the original data points. Thus $AZ + b$ restores the mean.

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.)

   From the A matrix in Q2.2, the d-dimensional representation z_i of a point x_i is obtained projecting x_i onto the subspace spanned by A.

   Thus, $z_i = A^T(x_i - \mu)$. (μ is the mean of the original data).

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.

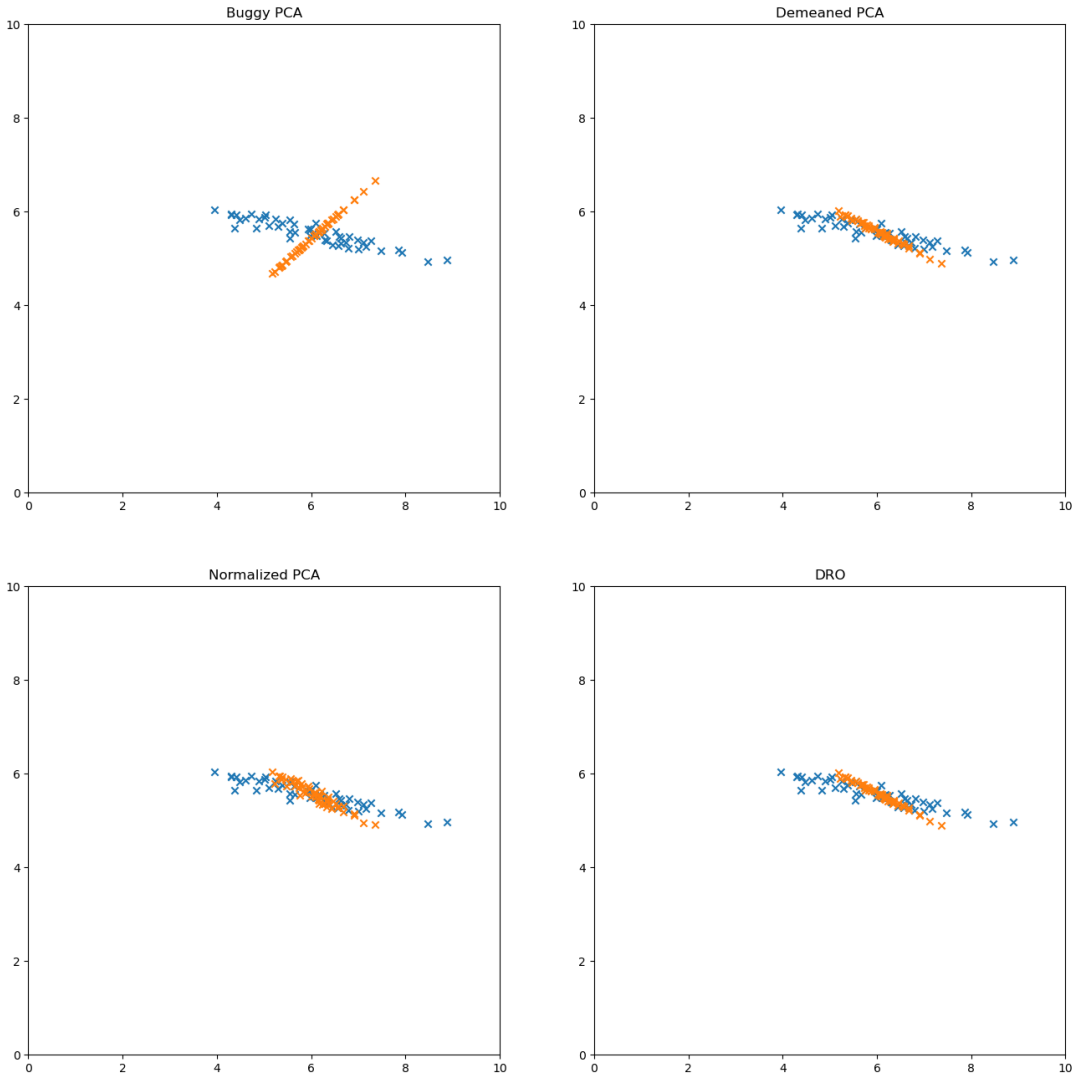

Figure 3: Comparing different methods. Blue markers are the original data and Orange markers represent the reconstructed points.

- From Fig. 2(a), it can be seen that there is a significant drop in the magnitude of the singular values near the beginning. (*Note hat the scale is Logarithmic for Fig. 2 (a)*).
- Zooming in on the intial part as seen in Fig. 2(b), we can observe a significant drop from Index 29 to Index 30. Further verification shows that the error does no reduce much with $d = 31$ and later.
- Thus $d = 30$(Index 29) is a clear "knee point".
- A vertical line is drawn in Fig. 2(b) to show the significant drop in the singular values.

- For the $2D$ dataset you need to attach the a plot comparing the orignal points with the reconstructed points for all five 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.

  Fig. 3 shows the original points and the reconstructed points using $d = 1$ for the different implementations.

  Reconstruction Errors are reported below.

  - ***2D Datasets*** ($d = 1$):
    - Error for Buggy PCA: 0.884691951047077
    - Error for Demeaned PCA: 0.007149309154951905

    - Error for Normalized PCA: 0.03603156595649728
    - Error for DRO: 0.007149309154951905
  - ***1000D Datasets (d = 30)*** :
    - Error for Buggy PCA: 2564813.1741051986
    - Error for Demeaned PCA: 868849.2657102454
    - Error for Normalized PCA: 870683.1535936354
    - Error for DRO: 868849.2657102454
  - ***1000D Datasets (d = 31)*** :
    - Error for Buggy PCA: 867932.1993035482
    - Error for Demeaned PCA: 863541.6411836385
    - Error for Normalized PCA: 865629.2071053413
    - Error for DRO: 863541.6411836385

- **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?
     PCA aims to find the subspace that maximizes the variance along the axes. For the Buggy PCA, since the mean isn't subtracted, the first PC tends to point towards the mean rather than the direction of the maximum variance.
     Thus Buggy PCA is attempting to project the points onto a subspace that is not centered around the distribution of the data.
  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?

     - Demeaned PCA and DRO give the lowest error since they both center the data by subtracting the mean.
     - Normalized PCA gives slightly higher error due to the cumulative errors that addd up when de-normalizing the data.
     - Buggy PCA gives the worst results since it points to the mean rather than the direction of the maximum variance.
     - However, both Demeaned PCA and DRO essentially follow the same operations and thus get the exact results.

     Thus, it is not surprising that demeaned PCA and DRO give the lowest average squared errors.

- 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 methods: **(10 points)**
  - Implementing reconstructions and reporting results: **(5 points)**
  - Choice of d for $1000D$ dataset and appropriate justification: **(3 Points)**
  - Questions **(4 Points)**