INTRODUCTION TO LINEAR DIMENSIONALITY REDUCTION METHODS

BAICHEN TAN

ABSTRACT. This paper discusses classical techniques in dimensionality reduction methods. We study three important linear high dimensional reduction methods: principal component analysis, classical multidimensional scaling, and random projection. Detailed explanations will be given to illustrate the three methods' intuition and theoretical foundations.

CONTENTS

1. INTRODUCTION

In recent years, dimensionality reduction has been gaining more and more attention in the realm of statistics. Dimensionality reduction is a technique used to reduce the number dimensions of a dataset while retaining essential information, such as pairwise distances or other geometric features. High-dimensional datasets can lead to challenges in visualization, computational complexity, and overfitting. For example, in genetics, we often have thousands of gene snippets to test disease effects, while we often only have access to a limited number of patients as observations. In such a high-dimensional setting, traditional statistical tools, such as linear regression, would fail since there are many more variables than equations (observations). Moreover, many basic statistics such as correlation matrices scale quadratically (or worse) in size with dimensionality, making their computation intractable for the dimensionality of many real-world datasets. Therefore, dimensionality reduction is a very important technique in statistical analysis. This paper examines three linear methods for dimensionality reduction: principal component analysis (PCA), classical multidimensional scaling (CMDS), and random projections. Readers of this paper should be familiar with linear algebra. We will give detailed proofs of the important theorems and lemmas in this paper; proofs of some lemmas and propositions are relegated to [1] or to [2].

2. PRINCIPAL COMPONENT ANALYSIS (PCA)

Suppose we have a high dimensional dataset with a very large dimension p, but our data actually approximately lies in a hyperplane of dimension $k \ll p$. The goal of PCA is to project the original p dimensional data onto the k dimensional hyperplane to reduce the dimensionality of our original dataset. In Figure 1, we

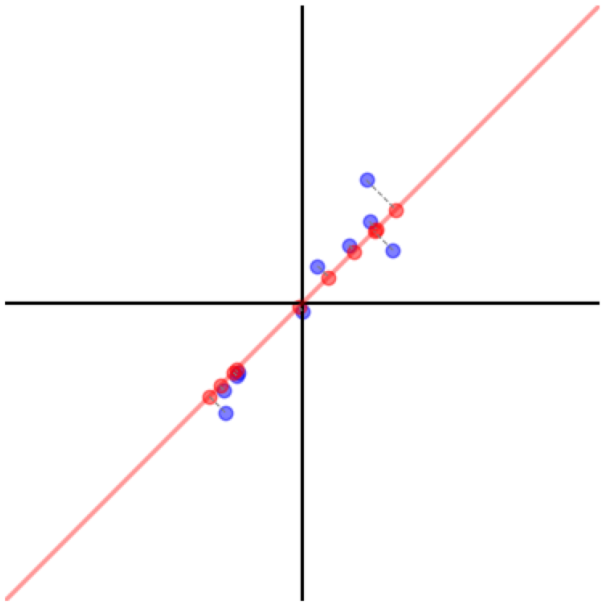


FIGURE 1. Intuition of PCA

project a two dimensional dataset to an one dimensional line. As we can see, after the projection, the overall geometry of our new dataset is roughly preserved. The idea of PCA is to find the "correct" hyperplane on which to project the data as to minimize any distortion to the relative geometry.

2.1. **Review of Linear Algebra.** We first review some prerequisite linear algebra.

Theorem 2.1 (Singular Value Decomposition)**.** *Let A be an $m \times n$ matrix with rank r. Then there always exists a singular value decomposition*

$$A = U\Sigma V^T$$

where U, Σ, V^T are given by:

$$U = \begin{bmatrix} u_1, u_2, \dots, u_r \end{bmatrix} \in \mathbb{R}^{m\times r},$$

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sigma_r \end{bmatrix} \in \mathbb{R}^{r\times r}$$

$$V^T = \begin{bmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_r^T \end{bmatrix} \in \mathbb{R}^{r\times n}$$

where u_i and v_i are called the left and right singular vectors, respectively. The values σ_i are called the singular values with the ordering $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r$. Importantly, each of the u_i's are orthogonal to each other and each of the v_i's are orthogonal to each other.

We now introduce the Frobenius norms on matrices.

Definition 2.2 (Frobenius norm)**.** The Frobenius norm of a matrix $A \in \mathbb{R}^{m\times n}$ is defined as

$$\|A\|_F = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2 \right)^{\frac{1}{2}} = \mathrm{tr}(A^T A)^{\frac{1}{2}}$$

We introduce two important properties of the Frobenius norm. The proof of the following two propositions can be found in [1].

Proposition 2.3. *The Frobenius norm is invariant under rotation by an orthogonal matrix U. That is, if U is orthogonal, then $\|AU\|_F = \|UA\|_F = \|A\|_F$ for any matrix A.*

Proposition 2.4. *Let $A \in \mathbb{R}^{m\times n}$ be a matrix with rank r. Then the Frobenius norm of A is*

$$\|A\|_F = \left(\sum_{i=1}^{r} \sigma_i^2\right)^{\frac{1}{2}}$$

where σ_i are singular values of A

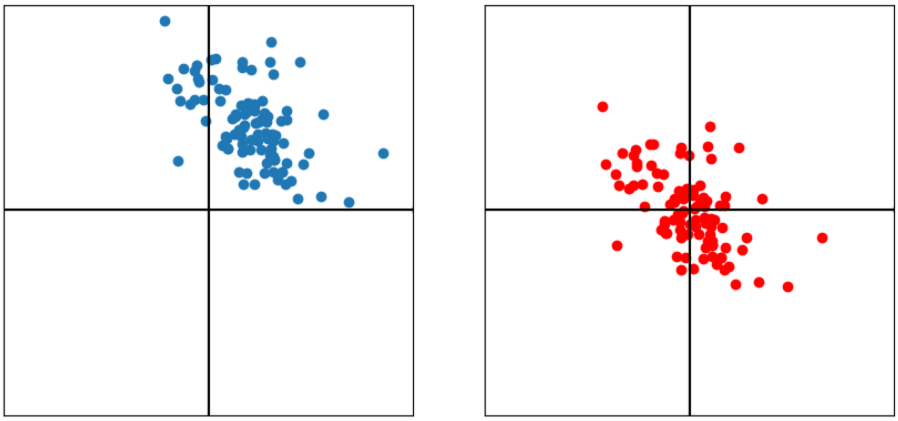
2.2. **The PCA Algorithm.** Now we will explain the PCA method. Let $X = [x_1, ..., x_n]^T \in \mathbb{R}^{n\times p}$ be our dataset with n observations. The a-shift of a dataset X is denoted as

$$X_a = X - a$$

where a is a row vector in $\mathbb{R}^p$. In practice, we often shift data by the sample mean, given by

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

As can see in Figure 2, shifting a dataset doesn't change the relative geometry.



(A) Dataset before shift by sample mean (B) Dataset after shift by sample mean

FIGURE 2. Shifting a dataset

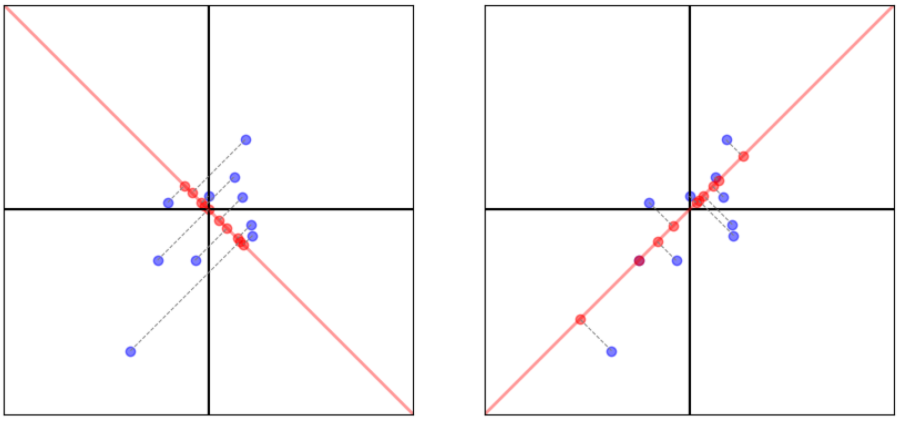
For simplicity, we shall assume in later part of this text that our dataset is already centered, that is, $X = \widehat{X} = X - \bar{X}$.

Definition 2.5. We define the energy of a dataset

$$\mathcal{E}(X) = \sum_{i=1}^{n} \|x_i\|_2^2 = \|X\|_F^2.$$

Loosely speaking, PCA is a technique that identifies the subspace in which a dataset clusters. We first consider the maximization problem of finding a one dimensional subspace S_1 that maximizes the energy of X after we project it onto S_1. We use a unit vector v_1 to represent the direction of S_1. We denote the projection transformation $\mathbb{R}^p \to S_1$ by T_{v_1}, and the set of all directions in $\mathbb{R}^p$ by the unit sphere $\mathbb{S}^{p-1}$. Thus, our new coordinate v_1 is the solution of the following maximization problem:

$$v_1 = \arg\max_{a\in\mathbb{S}^{p-1}} \mathcal{E}\left(T_a(X)\right) \tag{2.6}$$



(A) projection with low energy (B) projection with high energy

FIGURE 3. Projection onto the first coordinate

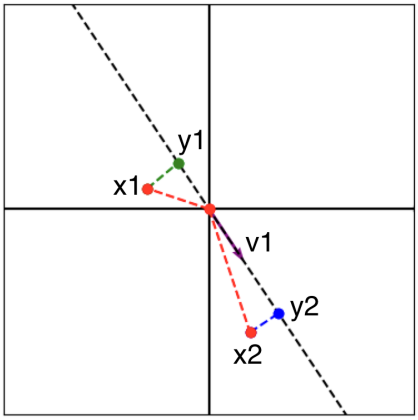


FIGURE 4. Illustration of PCA on single data points

Figure 3 provides an intuitive visualization of why we want to perform this maximization. Because the energy is the sum of the norms of the projected vectors, the projection in plot B preserves more energy than that in plot A, and the projected data points in plot B represents the original data clustering better than plot A does.

We call $v_1 \in \mathbb{S}^{p-1}$ the first principal direction. Now we take a closer look at what happens to individual data points in PCA. Figure 4 provides a visual explanation. Suppose we have two data points x_1, x_2 in the original dataset X, and we want to project the them onto the line where the direction vector v_1 lies. Then the projected points y_1, y_2 are called the first principal components of x_1, x_2 respectively. They represent the locations of the projected points in the original coordinate system. To generalize, for each $x_i \in X$, its first principal component is denoted as

$$y_{1,i} = T_{v_1}(x_i) \in \mathbb{R}^p \tag{2.7}$$

We can also write equation (2.7) in the form of matrix. We denote

$$Y_1 = [y_{1,1}, \cdots, y_{1,n}]^T = [T_{v_1}(x_1), \cdots, T_{v_1}(x_n)]^T = T_{v_1}(X) \in \mathbb{R}^{n \times p}$$

Now suppose we want to project our dataset to a second basis vector. This means finding a second principal direction v_2 onto which to project the data that maintains as much as possible the information that is not in the span by the first coordinate v_1. In other words, for each data point x_i, we want to project

$x_i - v_1 y_{1,i}$ onto a second subspace that maximizes the energy. Because $x_i - v_1 y_{1,i}$ is the part of x_i that is not in the span of the first principal direction, it is linearly independent from $v_1 y_{1,i}$, and therefore v_2 and v_1 are also independent and thus they are orthogonal to each other. We denote the span(v_1) as S_1 and the subspace of $\mathbb{S}^{p-1}$ that is orthogonal to S_1 as $S_1^{\perp} \cap \mathbb{S}^{p-1}$. Therefore, the second principal component v_2 is the solution of the following problem:

$$X_1 = [x_1 - v_1 y_{1,1}, \cdots, x_n - v_1 y_{1,n}]^T \tag{2.8}$$

$$v_2 = \underset{a \in S_1^{\perp} \cap \mathbb{S}^{p-1}}{\arg\max} \; \mathcal{E}\left(T_a(X_1)\right)$$

and for each $x_i - v_1 y_{1,i} \in X_1$, we call

$$y_{2,i} = T_{v_2}\left(x_i - v_1 y_{1,i}\right) \tag{2.9}$$

the second principal component of $x_i - v_1 y_{1,i}$. Similarly, we can write equation (2.9) in matrix form

$$Y_2 = [y_{2,1}, \cdots, y_{2,n}]^T = [T_{v_2}(x_1 - v_1 y_{1,1}), \cdots, T_{v_2}(x_n - v_1 y_{1,n})]^T = T_{v_2}(X_1) \in \mathbb{R}^{n \times p}$$

Then, using mathematical induction, we can calculate the successive principal directions as follows. Assume that the first $s-1$ principal directions $V_{s-1} = \{v_1, \cdots, v_{s-1}\} \subset \mathbb{S}^{p-1}$ and the corresponding $s-1$ principal components $\{Y_1, \cdots, Y_{s-1}\} \subset \mathbb{R}^p$ of X are well defined. We denote the subspace spanned by $v_1, \cdots, v_{s-1}$ as S_{s-1} and the subspace of $\mathbb{S}^{p-1}$ that is orthogonal to S_{s-1} as $S_{s-1}^{\perp} \cap \mathbb{S}^{p-1}$. Now, we want to project onto the sth coordinate the information in X that is not spanned by the previous $s-1$ principal components, which we denote as

$$X_{s-1} = \left\{ x_i - \sum_{j=1}^{s-1} v_j y_{j,i}, 1 \le i \le n \right\} \tag{2.10}$$

Similarly, v_s is orthogonal to $v_1 \cdots v_{s-1}$. Then the sth principal direction is defined as

$$v_s = \underset{a \in S_{s-1}^{\perp} \cap \mathbb{S}^{p-1}}{\arg\max} \; \mathcal{E}\left(T_a\left(X_{s-1}\right)\right)$$

and the sth principal component of X is $Y_s = T_{v_s}\left(X_{s-1}\right)$.

While it is hard to solve the maximization problem described above directly, we can convert it into the following minimization problem which is easy to solve:

Theorem 2.11. *The projection T_a that maximizes the energy of a dataset X is also the projection that minimizes the energy of the orthogonal vector $(I - T_a)(X)$. Namely,*

$$v_1 = \underset{a \in \mathbb{S}^{p-1}}{\arg\max} \, \mathcal{E}\left(T_a(X)\right) = \underset{a \in \mathbb{S}^{p-1}}{\arg\min} \, \mathcal{E}\left(\left(I - T_a\right)(X)\right), \tag{2.12}$$

and

$$v_s = \underset{a \in S_{s-1}^{\perp} \cap \mathbb{S}^{p-1}}{\arg\max} \; \mathcal{E}\left(T_a\left(X_{s-1}\right)\right) = \underset{a \in S_{s-1}^{\perp} \cap \mathbb{S}^{p-1}}{\arg\min} \; \mathcal{E}\left(\left(I - T_a\right)\left(X_{s-1}\right)\right). \tag{2.13}$$

Proof. This is the direct result of the Pythagorean theorem. Recall that the energy is the sum of the distances from each projected data point to the origin. Consider for now a single point. We want to find the direction on the unit sphere that maximizes the distance between the projected data point and the origin. Notice in Figure 5 that the distance from the original data point to the origin (i.e. the norm of the vector c) is unchanged as the coordinate vector a is rotated around the origin. By the Pythagorean theorem, $\|c\|^2 = \|a\|^2 + \|b\|^2$. Thus, for fixed $\|c\|$, maximizing $\|a\|$ is equivalent to minimizing $\|b\|$. The norm of b can be represented as $\|x_i - T_a(x_i)\|$ where T_a is our desired transformation that maximizes the energy.

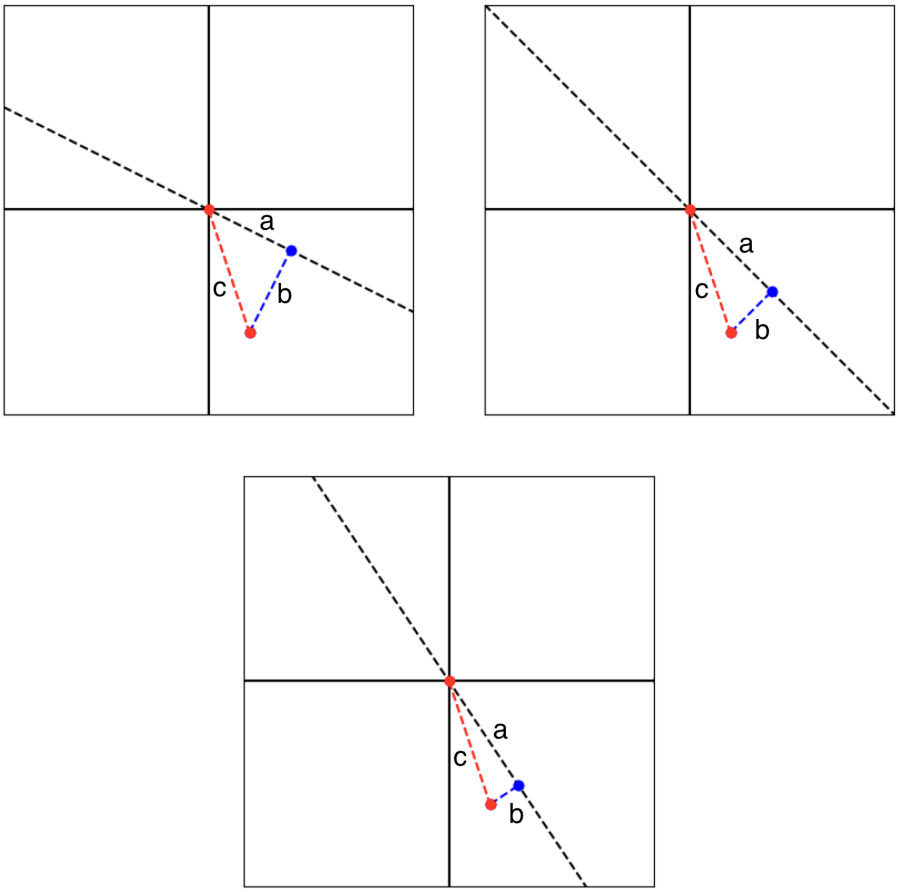


FIGURE 5. Illustration of the Pythagorean theorem

□

The solutions of principal directions can be obtained with the aid of the best $k-$rank approximation for matrices.

Theorem 2.14 (Mirsky (1960), Eckart and Young (1936))**.** *Let $X \in \mathbb{R}^{n\times p}$ be our dataset, where each column is a data point and each row is an observation. Let $r = \text{rank}(X)$ and the SVD decomposition of X be:*

$$X = U\Sigma V^T = \sum_{i=1}^{r} \sigma_i u_i v_i^T$$

where $U = [u_1, u_2, ..., u_r] \in \mathbb{R}^{n\times r}$, $\Sigma = \text{diag}(\sigma_1, ..., \sigma_r) \in \mathbb{R}^{r\times r}$ with $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r$, and $V = [v_1, ..., v_r] \in \mathbb{R}^{p\times r}$

Denote $\widetilde{X} = \sum_{i=1}^{k} \sigma_i u_i v_i^T, k \leq r$, then we have the following statement: $\widetilde{X}$ is the rank k matrix that is closest in Frobenius norm to X, i.e.

$$\|X - \widetilde{X}\|_F = \min_{\text{rank}(A)=k} \|X - A\|_F$$

We call $\widetilde{X}$ the best k-rank approximation of X.

A proof can be found in [1].

Remark 2.15. We can also write the best k-rank approximation in the form of matrix multiplication. Let the SVD of X be given by $X = U\Sigma V^T$ and $\text{rank}(X) = r$. We denote

$$U_k = [u_1, \cdots, u_k], \Sigma_k = \text{diag}(\sigma_1, \cdots, \sigma_k), V_k = [v_1, \cdots, v_k], \text{ for some } k \leq r$$

Then the best k-rank approximation of X is

$$\widetilde{X} = U_k \Sigma_k V_k^T$$

By the best k-rank approximation theorem, now we can then construct kth principal directions and components of a high dimensional dataset $X \in \mathbb{R}^{n\times p}$ using the following theorem.

Theorem 2.16. *Let $X \in \mathbb{R}^{n\times p}$ be our centered dataset, and let U_k, Σ_k, V_k be defined as remark 2.15. Then columns $v_1, v_2, \cdots, v_k$ of V_k are the k principal directions of X. Namely, they are the directions that minimize the energy of $(I - T_a)(X), (I - T_a)(X_1), ..., (I - T_a)(X_{k-1})$ respectively, where X_1 and X_{k-1} are defined as (2.8) and (2.10). The k principal components of X are the column vectors of the matrix*

$$Y = [y_1, \cdots, y_n] = U_k \Sigma_k,$$

Proof. Let $r = \text{rank}(X)$ and assume $k \leq r$. We prove the theorem by induction. For $1 \leq s \leq k$, denote

$$U_s = \begin{bmatrix} u_1 & \cdots & u_s \end{bmatrix}, \Sigma_s = \text{diag}(\sigma_1, \cdots, \sigma_s), V_s = \begin{bmatrix} v_1 & \cdots & v_s \end{bmatrix}$$

and

$$Y_s = U_s \Sigma_s.$$

Note Y_s consists of the first s rows of Y. Define the matrix $B_s = Y_s V_s^T \in \mathbb{R}^{n\times p}$. Suppose $s = 1$, then by theorem 2.14, $B_1 = Y_1 V_1^T$ is the best 1 rank approximation of X. Namely,

$$\|X - B_1\|_F = \min_{\text{rank}(B)=1} \|X - B\|_F = \|X - U_1\Sigma_1 V_1^T\|_F = \|X - XV_1^T\|_F \tag{2.17}$$

Recall that in theorem 2.11, we want to minimize $\mathcal{E}((I - T_a)(X)), a \in \mathbb{S}^{p-1}$ to find the first principal direction v_1. Rewriting the energy sign in the form of Frobenius norm, we have

$$\begin{aligned} \min \mathcal{E}((I - T_a)(X)) &= \min \|(I - T_a)(X)\|_F \\ &= \min \|X - T_a(X)\|_F \end{aligned}$$

Notice that $T_a(X)$ has rank 1, and therefore the minimization problem of finding the first principal direction is equivalent to the best 1-rank approximation in equation (2.17). Namely, v_1 is the first principal direction and $U_1\Sigma_1$ is the corresponding first principal component of X.

Now suppose $v_1, \cdots, v_{s-1} \in V_{s-1}$ are the first $s-1$ principal directions of X and $U_{s-1}\Sigma_{s-1}$ are the corresponding principal components, we want to prove that $v_s \in V_s$ is the sth principal direction of X. By theorem 2.11, we want to minimize $\mathcal{E}((I - T_a)(X_{s-1})), a \in S_{s-1}^{\perp} \cap \mathbb{S}^{p-1}$ to find the sth principal direction of X. Rewriting the energy in the form of the Frobenius norm, we have

$$\begin{aligned} \min \mathcal{E}((I - T_a)(X_{s-1})) &= \min \|(I - T_a)(X_{s-1})\|_F \\ &= \min \|X_{s-1} - T_a(X_{s-1})\|_F \\ &= \min \|X - U_{s-1}\Sigma_{s-1}V_{s-1}^T - T_a(X_{s-1})\|_F \\ &= \min \|X - \left(U_{s-1}\Sigma_{s-1}V_{s-1}^T + T_a(X_{s-1})\right)\|_F \end{aligned}$$

where the third equation holds because $X_{s-1} = X - U_{s-1}\Sigma_{s-1}V_{s-1}^T$ by the way X_{s-1} is defined in (2.10).

We denote $A_s = U_{s-1}\Sigma_{s-1}V_{s-1}^T + T_a(X_{s-1})$. Notice that A_s is a rank s matrix because $U_{s-1}\Sigma_{s-1}V_{s-1}^T$ has rank $s-1$ and $T_a(X_{s-1})$ has rank 1 and $T_a(X_{s-1})$ is not in the span of $v_1, \ldots, v_{s-1}$. Then, the energy minimization problem in finding the sth principal direction becomes

$$\min \|X - A_s\|_F, \ \text{rank}(A_s) = s$$

which is equivalent to the best s rank approximation problem defined in 2.14. Therefore, v_s is the sth principal direction of X.

To sum up, $\{v_1, \ldots, v_k\} \subset \mathbb{R}^k$ are successive principal directions of X and the columns of $Y_s = U_s\Sigma_s$ are the corresponding d principal components of X.

□

2.3. **Statistical interpretation of PCA.** Now we are going to discuss a statistical interpretation of PCA. Let $X = [x_1, \cdots, x_n]^T \in \mathbb{R}^{n\times p}$ be our dataset. We now consider each column vector $x_i \in X$ as an observation of the dataset, and each row the sample space of a variable.

For simplicity, we assume that our dataset X is already shifted by its sample mean so that all of its components have zero mean, i.e. $X = \widehat{X}$. Recall that the variance of a random variable Z with n samples $\{z_i\}_{i=1}^n$ is computed by

$$\mathrm{Var}(Z) = \mathbb{E}\left[(Z - \bar{Z})^2\right] = \frac{1}{n}\sum_{i=1}^{n}\left(z_i - \bar{Z}\right)^2,$$

where $\bar{Z}$ is the mean of Z. Particularly, when $\bar{Z} = 0$,

$$\mathrm{Var}(Z) = \frac{1}{n}\sum_{i=1}^{n}(z_i)^2 = \frac{1}{n}\|z\|_2^2.$$

We can think of each data point x_i as a realization of some random variable following a certain distribution such as Gaussian or Bernoulli depending on the data assumption. Remember that when trying to find the first principal direction, we try to find the projection of X onto a lower dimension that maximizes the energy $\arg\max \mathcal{E}\left(T_a(X)\right)$. Because energy of a dataset is defined as the sum of the square of the norm of each data point, we can rewrite the formula in 2.6 as

$$\underset{a\in\mathbb{S}^{p-1}}{\arg\max}\mathcal{E}\left(T_a(X)\right) = \underset{a\in\mathbb{S}^{p-1}}{\arg\max}\sum_{i=1}^{n}\|\left(T_a(x_i)\right)\|_2^2$$

We can see that maximizing $\sum_{i=1}^n \|\left(T_a(x_i)\right)\|^2$ is equivalent of maximizing the variance of $T_a(X)$.Therefore, the first principal direction of the dataset X is defined as the direction of maximum variance.

3. Classical Multidimensional Scaling (CMDS)

Classical multidimensional scaling (CMDS) is a dimensionality reduction method that preserves local features between the data points. Specifically, given a set of high dimensional data points X, we wish to project them onto some lower dimensional subspace in a way that preserves the pairwise distances between them as much as possible.

3.1. **Distance and Gram Matrices.** In this section we will introduce the distance matrix and gram matrix.

Definition 3.1 (Euclidean distance and square-distance matrix)**.** Let $X = [x_1, x_2, \cdots, x_n]^T \in \mathbb{R}^{n\times p}$ be our dataset. The Euclidean distance matrix is an $n \times n$ matrix $D = [D_{ij}]$ whose each entry D_{ij} is the Euclidean distance between the data point x_i and x_j:

$$[D_{ij}] = [\|x_i - x_j\|_2]_{i,j}^n \tag{3.2}$$

The Euclidean square-distance matrix S is defined as

$$[S_{ij}] = [\|x_i - x_j\|_2^2]_{i,j}^n \tag{3.3}$$

Notice that both D and S are symmetric and invariant under shifts and rotations, since these transformations do not affect pairwise distances.

From the definition 3.1 above, we can see that any matrix that can be written in the form (3.2) can be considered as a Euclidean distance matrix. The goal of CMDS is to recover the points x_i from knowledge of D alone.

Definition 3.4. Let $X = [x_1, x_2, ..., x_n]^T \in \mathbb{R}^{n\times p}$ be a dataset. We define a Gram matrix G on dataset X as

$$[G_{ij}] = [\langle x_i, x_j\rangle]_{i,j=1}^n = XX^T$$

where $\langle\cdot,\cdot\rangle$ is the standard Euclidean inner product. If the Gram matrix is defined on a centered dataset $\widehat{X}$, then it is called the centering matrix, which we often denote as G^c.

Because the entries of a Gram matrix G are vector inner products, Gram matrix is **not shift invariant**.

We also notice that G is a positive semi-definite (p.s.d) matrix because for any nonzero real column vector z,

$$z^T G z = z^T X X^T z = z^T (X^T)^T X^T z = \|X^T z\|_2^2 \geq 0 \tag{3.5}$$

For any dataset, we can easily determine its Gram matrix. Inversely, we can also think of any p.s.d matrix as a Gram matrix of some dataset. This is because if an $n \times n$ p.s.d matrix G has rank m, then it has a Cholesky decomposition

$$G = XX^T \tag{3.6}$$

where $X = [x_1, \cdots, x_n]^T$ is an $n \times m$ upper triangular matrix, which we can view as the matrix representation of some sort of dataset $\{x_1, x_2, ..., x_n\} \subset \mathbb{R}^m$. Given a p.s.d. matrix, we now want to investigate if it is a *centering* Gram matrix for some set of points in $\mathbb{R}^m$.

Definition 3.7. Write $\mathbf{1} = [1, 1, \cdots, 1]^T \in \mathbb{R}^n, E = \mathbf{1}\mathbf{1}^T$, and let I denote the $n \times n$ identity matrix. Then the $n \times n$ matrix $H = I - \frac{1}{n}E$ is called the n-centralizing matrix.

The following lemma helps us to check whether a dataset is centered and whether a positive semi-definite matrix can be viewed as a centering Gram matrix.

Lemma 3.8. *The n-centralizing matrix H have the following properties.*

(1) $H^2 = H$
(2) $\mathbf{1}^T H = H\mathbf{1} = 0$,
(3) X is a centered data set, if and only if $H^T X = X$,
(4) A positive semi-definite matrix C is a centering Gram matrix, if and only if $HCH = C$.

The proof of this lemma can be found as Lemma 1 in Chapter 6 of [2]

Remark 3.9. Let $X \in \mathbb{R}^{n \times p}$ be a data matrix and G be its Gram matrix. Then the centered data set of X is $H^T X$, and the centering Gram matrix of X is $G^c = HGH$.

Just like PCA, we always shift our dataset by the sample mean in CMDS because the centering Gram matrix has the following nice property.

Lemma 3.10. *Let G^c be a centering Gram matrix. Then $\sum_{i=1}^n G^c_{ij} = 0$ for all j. That is, the sum of each row is equal to zero.*

Proof. We can rewrite the summation in Lemma 3.10 in the form of inner products,

$$\sum_{i=1}^n G^c_{ij} = \sum_{i=1}^n \langle x_i, x_j \rangle = \left\langle \sum_{i=1}^n x_i^T, x_j \right\rangle$$

Notice that because x_i's are centered, their summation $\sum_{i=1}^n x_i^T$ adds up to 0. □

Now we want to explore the relationship between the Gram matrix and the distance matrix.

Lemma 3.11. *Let $X = [x_1, x_2, ..., x_n]^T \in \mathbb{R}^{n \times p}$ be our dataset. The Gram matrix G and the Euclidean distance matrix D has the following relation:*

$$D_{ij} = \|x_i - x_j\|_2 = \sqrt{G_{ii} + G_{jj} - 2G_{ij}}.$$

Proof. We compute

$$\begin{aligned} D_{ij} &= \sqrt{\|x_i - x_j\|_2^2} \\ &= \sqrt{\langle x_i, x_i \rangle + \langle x_j, x_j \rangle - 2\langle x_i, x_j \rangle} \\ &= \sqrt{G_{ii} + G_{jj} - 2G_{ij}} \end{aligned}$$

□

Theorem 3.12. *For a given centered dataset X, its Euclidean square-distance matrix S and Gram centering matrix G^c have the following relation.*

$$G^c = -\frac{1}{2}S$$

Proof. By lemma 3.10, G^c has the property $\sum_{i=1}^n G^c_{ij} = 0$. Hence, the relation in lemma 3.11 immediately gives us

$$\sum_{i=1}^n D^2_{ij} = nG^c_{jj} + \sum_{i=1}^n G^c_{ii}$$

and

$$\sum_{j=1}^n D^2_{ij} = nG^c_{ii} + \sum_{j=1}^n G^c_{jj}.$$

Therefore, the (i, j)-entry of S is given by

$$\begin{aligned} S_{ij} &= D^2_{ij} - \frac{1}{n}\left(\sum_{i=1}^n D^2_{ij} + \sum_{j=1}^n D^2_{ij} - \frac{1}{n}\sum_{i,j=1}^n D^2_{ij}\right) \\ &= D^2_{ij} - G^c_{ii} - G^c_{jj} \\ &= -2G^c_{ij}, \end{aligned}$$

completing the proof of the theorem. □

Theorem 3.12 shows the importance of centering our dataset in CMDS. Given a centered dataset, we can always find its corresponding Euclidean square-distance matrix by its centering Gram matrix through the equation $G^c = -\frac{1}{2}S$.

By Theorem 3.12, we have the following corollary.

Corollary 3.13. *Let A be a symmetric matrix. Then*
(1) A is a Gram matrix of a data set if and only if it is a p.s.d matrix.
(2) A is a centering Gram matrix if and only if it is a centering p.s.d matrix. A centering p.s.d matrix is a p.s.d matrix subtracted by its sample mean.
(3) A is a Euclidean square-distance matrix if and only if $-\frac{1}{2}A$ is a centering p.s.d matrix.

Proof. (1) is already proved in equation (3.5) and (3.6). (2) can be easily proved by lemma 3.8 (4). We now want to prove (3). For the backwards direction, write $G^c = -\frac{1}{2}A$. If G^c is a centering p.s.d matrix, by equation (3.6), G^c has a Cholesky decomposition

$$G = XX^T$$

where A can be seen as the Euclidean square-distance matrix of the dataset X. For the forwards direction, if A is a Euclidean square-distance matrix, then by theorem 3.12, $G^c = -\frac{1}{2}A$ is the centering Gram matrix of X so that it is a centering p.s.d matrix. □

The relationship between the centering Gram matrix and the (square) distance matrix is important in CMDS. We can observe that Corollary 3.13 helps us determine whether a given matrix can be viewed as a (centering) Gram matrix or distance matrix. Recall that the intuition of CMDS is that given a distance matrix, we want to construct a configuration of points that reflects this distance matrix. The Gram matrix can then help us find the formula for finding the low dimensional configuration set such that the distance matrix of this low dimensional configuration set is not distorted much from the original distance matrix.

3.2. **The CMDS algorithm.** Let $X = [x_1, x_2, ..., x_n]^T \in \mathbb{R}^{n\times p}$ be a centered dataset, and let $D = [D_{ij}]$ be the distance matrix of our dataset. CMDS tries to find a configuration $Y = [y_1, \cdots, y_n]^T$ on a lower k dimensional subspace such that the distance matrix of Y is as close as possible to the matrix D. Namely,

$$\|y_i - y_j\|_2 \approx D_{ij}, \forall i, j.$$

Lemma 3.14. *Assume that an $n \times n$ matrix D is a Euclidean distance matrix and S is the corresponding square-distance matrix. Let $G^c = -\frac{1}{2}S$. If the rank of G^c is r. Then there is an r-dimensional centered vector set $X = [x_1, \cdots, x_n]^T \in \mathbb{R}^{n\times r}$ such that*

$$\|x_i - x_j\|_2 = D_{ij},\ 1 \le i, j \le n.$$

Proof. By theorem 3.12, G^c is a centering Gram matrix. Since the rank of G^c is r, there exists an $n \times r$ centered data matrix X such that $G^c = XX^T$ by Cholesky decomposition. Then the centered data set X satisfies the above relation. □

We call r in Lemma 3.14 the intrinsic configuration dimension of D and X the exact configuration of D.

In section 3.1, we have shown how to determine if a p.s.d matrix is a distance matrix. Now for any distance matrix D, we can find the corresponding Gram matrix G^c by theorem 3.12 and the decomposition $G^c = XX^T$ gives you the exact dataset X for the original distance matrix. If we wish to reduce the dimensionality of the data, then we can try to find a lower rank $Y = [y_1, \cdots, y_n]^T$ such that the Gram matrix of Y is close to G^c. We use the following loss function to judge whether a certain configuration Y's distance matrix is close to the original dataset's distance matrix.

$$\eta(Y) = \sum_{i,j=1}^{n} \left(\|x_i - x_j\|_2^2 - \|y_i - y_j\|_2^2 \right), \text{ s.t. } y_i = Px_i,$$

where P is an orthogonal projection from $\mathbb{R}^p$ to a k-dimensional subspace $S_k \subset \mathbb{R}^r$ and X is an exact configuration of D. Our goal is to find the Y that minimizes the loss function η, namely,

$$Y = \underset{Y \in \mathbb{R}^{n \times k}}{\arg\min}\, \eta(Y), \quad \text{s.t. } Y = PX \tag{3.15}$$

We need to show that each term of the loss function is non-negative so that they won't cancel each other out. To prove this, we need the following theorem.

Theorem 3.16. *Let x be a vector and Px be an orthogonal projection. Then the norm of x is always greater than or equal to Px, namely,*

$$\|Px\|_2 \leq \|x\|_2$$

Before the proof of this theorem, we first take an intuitive look at why this is true. As we can see in Figure 6, when we project two vectors to a lower dimension, their distance always shrinks. The length of the projected vector is always shorter than that of the original vector when we project the data point to a lower dimension.

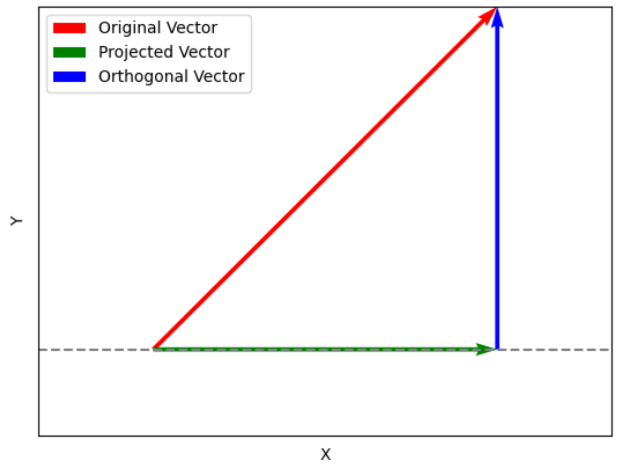


FIGURE 6. Illustration of Orthogonal Projection

Proof of Theorem 3.16. We compute

$$\begin{aligned}
\|x\|_2^2 &= \|Px + x - Px\|_2^2 \\
&= \|Px + (I - P)x\|_2^2 \\
&= \|Px\|_2^2 + \|(I - P)x\|_2^2 + 2(Px)^T(I - P)x \\
&= \|Px\|_2^2 + \|(I - P)x\|_2^2
\end{aligned}$$

where the last equation holds because

$$\begin{aligned} 2(Px)^T(I-P)x &= 2x^TP^T(I-P)x \\ &= 2x^T(P^T-P^TP)x \\ &= 0 \end{aligned}$$

because the orthogonal projection matrix P is symmetric and $P^2 = P$

Then, since $\|x\|_2^2 = \|Px\|_2^2 + \|(I-P)x\|_2^2$ and $\|(I-P)x\|_2^2 \geq 0$, we have $\|Px\|_2^2 \leq \|x\|_2^2$ and thus $\|Px\|_2 \leq \|x\|_2$, as desired. □

We can then denote $x = x_i - x_j$ for $x_i, x_j \in X$ and $Px = y_i - y_j$. By theorem 3.16, each term of $\|x_i - x_j\|_2^2 - \|y_i - y_j\|_2^2$ is always larger or equal to 0.

In order to find a solution for the above problem (3.15), we introduce the following 3 lemmas.

Lemma 3.17. *If X is a centered dataset, and A is an orthogonal projection transformation to a lower dimension, then the transformed dataset XA is still centered.*

Proof. Because X is centered, by Lemma 3.8 (3), $X = H^TX$ and so $XA = H^TXA$ where H is the n-centralizing matrix. Then by lemma 3.8 (3) again, XA is a centered dataset. □

Lemma 3.18. *Let X be a centered dataset with corresponding Euclidean square-distance matrix $S = [S_{ij}]$, and let G^c be its centering Gram matrix. Then*

$$\operatorname{tr}(G^c) = \frac{1}{2n}\sum_{i=1}^{n}\sum_{j=1}^{n} S_{ij}$$

The proof of this lemma is expanding the entrywise summation and can be found in Chapter 6 Lemma 6.4 in [2], and it immediately yields lemma 3.19.

Lemma 3.19. *Let $X = [x_1, \cdots, x_n]^T$ be the centered dataset and D to be its distance matrix. Then*

$$\|X\|_F = \frac{1}{\sqrt{2n}}\|D\|_F.$$

The proof of this lemma can be found in Chapter 6 Lemma 6.5 in [2].

We now explain the method for finding the solution of the minimization problem in the equation (3.15).

Theorem 3.20. *Let $X \in \mathbb{R}^{n\times r}$ be the configuration of $D = [D_{ij}]$. Let $\operatorname{rank}(X) = r$ and the SVD of X be given by*

$$X = U\Sigma V^T$$

For a given $k \leq r$, let $V_k = [v_1, \cdots, v_k]$ and $Y = XV_k$. Then Y is a solution of the minimization problem in (3.15) with

$$\eta(Y) = \sum_{i=k+1}^{r}\sigma_i^2$$

Proof. Let S_k be a k-dimensional subspace of $\mathbb{R}^r$ and $B \in \mathbb{R}^{r\times k}$ be a matrix whose columns form an orthonormal basis on S_k. Then BB^T represents the corresponding orthogonal projection from $\mathbb{R}^r$ to S_k. Now observe

$$\begin{aligned} \left\|\left(I-BB^T\right)(x_i-x_j)\right\|_2^2 &= \left[\left(I-BB^T\right)(x_i-x_j)\right]^T\left(I-BB^T\right)(x_i-x_j) \\ &= (x_i^T-x_j^T)(I-BB^T)(I-BB^T)(x_i-x_j) \\ (B^TB=I) \quad &= (x_i^T-x_j^T)(I-2BB^T+BB^TBB^T)(x_i-x_j) \\ &= (x_i^T-x_j^T)(I-BB^T)(x_i-x_j) \\ &= x_i^Tx_j - x_i^TBB^Tx_i - x_j^Tx_i + x_j^TBB^Tx_i - x_i^Tx_j \\ &\quad - x_i^Tx_j - x_i^Tx_j + x_i^TBB^Tx_j + x_j^Tx_j - x_j^TBB^Tx_j \end{aligned}$$

where the last term is exactly the expansion of the term

$$\|x_i - x_j\|_2^2 - \|BB^Tx_i - BB^Tx_j\|_2^2$$

and so we have

$$\left\|\left(I-BB^T\right)(x_i-x_j)\right\|_2^2 = \|x_i-x_j\|_2^2 - \|BB^Tx_i - BB^Tx_j\|_2^2 \tag{3.21}$$

Then the loss function $\eta(Y)$ becomes

$$\eta(Y) = \sum_{i,j=1}^{n} \|x_i-x_j\|_2^2 - \|BB^Tx_i, BB^Tx_j\|_2^2 = \sum_{i,j=1}^{n} \left\|\left(I-BB^T\right)(x_i-x_j)\right\|_2^2 \tag{3.22}$$

Therefore, the minimization problem is transformed to the problem of finding a $B_* \in \mathbb{R}^{r\times k}$ such that minimizes $\eta(Y)$:

$$B_* = \underset{B\in\mathbb{R}^{r\times k}}{\arg\min} \sum_{i,j=1}^{n} \left\|\left(I-BB^T\right)(x_i-x_j)\right\|^2 \tag{3.23}$$

Notice that $X\left(I-BB^T\right) \in \mathbb{R}^{n\times r}$. We write $\hat{Z} = [z_1,\cdots,z_n] = \left(X\left(I-BB^T\right)\right)^T$ such that each $z_1,\cdots,z_n \in \mathbb{R}^r$. Notice that $\hat{Z}$ is still centered by Lemma 3.17. We use $D_{\hat{Z}}$ to denote the distance matrix of $\hat{Z}$. By Lemma 3.19, we have

$$\left\|D_{\hat{Z}}\right\|_F^2 = 2n\|\hat{Z}\|_F^2 \tag{3.24}$$

By equation (3.22), we have

$$\eta(Y) = \sum_{i,j=1}^{n} \left\|\left(I-BB^T\right)(x_i-x_j)\right\|_2^2 = \left\|D_{\hat{Z}}\right\|_F^2 \tag{3.25}$$

and

$$\|\hat{Z}\|_F = \|\hat{Z}^T\|_F = \left\|X-XBB^T\right\|_F \tag{3.26}$$

Therefore, minimizing $\eta(Y)$ is equivalent of minimizing $\left\|D_{\hat{Z}}\right\|_F^2$ by equation (3.25), and is then equivalent to minimizing $\|\hat{Z}\|_F$ by equation (3.24). Namely,

$$\min\eta(Y) = \min\left\|D_{\hat{Z}}\right\|_F^2 = \min\|\hat{Z}\|_F.$$

Because $\|\hat{Z}\|_F = \|\hat{Z}^T\|_F = \left\|X-XBB^T\right\|_F$ by equation (3.26), we have

$$\min\eta(Y) = \min\|\hat{Z}\|_F = \min\left\|X-XBB^T\right\|_F$$

Then our goal of minimizing $\eta(Y)$ is equivalent to minimizing $\left\|X-XBB^T\right\|_F$. Notice that XBB^T has rank k because we are projecting X to a k-dimensional subspace by BB^T. Therefore, minimizing $\left\|X-XBB^T\right\|_F$ becomes the best k-rank approximation problem. Letting $A = XBB^T$, we have

$$\min\left\|X-XBB^T\right\|_F = \min_{\text{rank}(A)=k}\|X-A\|_F$$

By Theorem 2.14, the solution of the above minimization problem is $A = U_k\Sigma_kV_k^T = XV_k^T$ where U_k,Σ_k,V_k^T are defined as remark 2.15. Then we have that

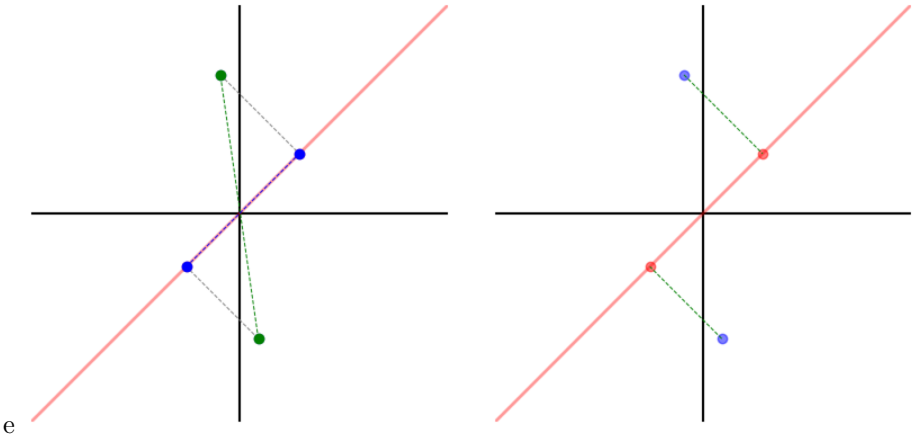
$$B_*B_*^T = V_k$$

is the solution of our minimization problem in equation (3.23), so our lower dimensional configuration is $Y = XV_k$, and the error term follows

$$\eta\left(XV_k\right) = \|X-XV_k\|_F^2 = \sum_{i=k+1}^{r}\sigma_i^2$$

by Proposition 2.4. □

3.3. **Comparison between CMDS and PCA.** The starting point between PCA and CMDS are different: in PCA, we are given the dataset and we want to project onto a lower dimensional subspace: to achieve this goal we identify the most important components in a given dataset X that retain the largest variance. In CMDS, we are given the distance matrix of a dataset, and we focus on finding a set of points in a lower-dimensional Euclidean space that maintain the greatest similarities, using an Euclidean distance matrix for n data points. Figure 7 provides an intuitive understanding of the difference between PCA and CMDS.



(A) Illustration of CMDS (B) Illustration of PCA

FIGURE 7. Comparison between CMDS and PCA

As we can see in Figure 7, in subplot 7a, CMDS tries to minimize the difference between the square of the length of the green line segment and that of the blue line segments, i.e. $\|x_i - x_j\|_2^2 - \|y_i - y_j\|_2^2$. In subplot 7b, PCA tries to minimize the sum of the length of the green line segments. Therefore, the starting point of PCA and CMDS are different.

However, PCA and CMDS yield the same results. As we can see in Theorem 2.16, the way we find the k principal directions and components is to find the k-rank approximation $\sum_{i=1}^{k} \sigma_i u_i v_i^T = U_k \Sigma_k V_k^T$. Similarly, in CMDS, the configuration is $Y = XV_k = U\Sigma V^T V_k$ which, if we write in the form of summation of row and column vectors, gives us the same result as finding the k-rank approximation in the PCA method. We can see that while PCA gives us the principal components V_k which form a basis that spans the subspace in which the transformed data lies, CMDS gives us the coordinates XV_k of the coordinates of the transformed data.

4. Random Projection

CMDS is a dimensionality reduction method without the direct access to the original dataset. In this section, we are going to introduce another method called random projection which, similar to PCA, operates on the dataset itself. While PCA is good at preserving global properties of a dataset, it may not maintain local relationships. For example, notice that in Figure 8, while the distance between the two yellow points is well preserved after PCA, the distance between the two green points is poorly preserved. This is because PCA projects the data points to a lower dimensional hyperplane in the direction of maximal variance. This means that when we project the data points onto the lower dimensional subspace, the subspace will tend to align in a way that fits points far away from the center and so the greatest pairwise distance will be preserved well while the smaller ones are less so. Thus, for points that are close to the center but are relatively far away from each other (like the yellow points in the figure), their distances may shrink to be arbitrarily small after PCA. To preserve local data distances in dimensionality reduction, we can use random projection. We will first introduce Lipschitz embeddings, which are important in the application of random projection.

Definition 4.1. A mapping $f : V \subset \mathbb{R}^p \to \mathbb{R}^k$ is called a Lipschitz mapping on V if there exist two positive constants A and B such that for each pair of vectors $u, v \in V$,

$$A\|u - v\|_2^2 \leq \|f(u) - f(v)\|_2^2 \leq B\|u - v\|_2^2$$

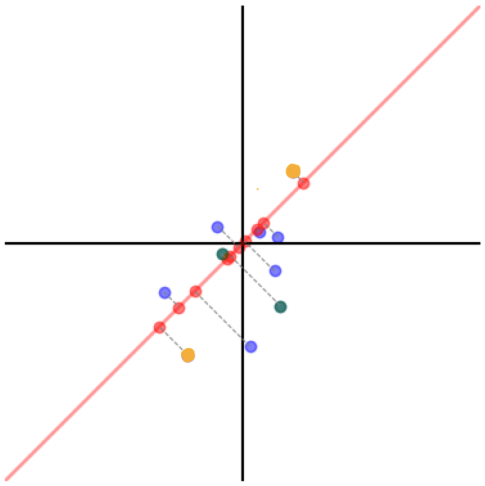


FIGURE 8. Illustration that PCA doesn't preserve local distances

From this, we can see that Lipschitz embeddings have a nice property of giving a lower and upper bound to the distance between any two vectors in the range of f. Random projections seek to employ this property of Lipschitz embeddings to preserve pairwise distances of the data points. We use random matrices to represent these random projections.

4.1. **Random Matrices.** A random matrix is a matrix such that each entry is an i.i.d random variable following a certain distribution. Here we introduce the most common type of random matrix.

Definition 4.2. A matrix R is Gaussian random matrix if each of its entry is a random variable following normal Gaussian distribution $N(0,1)$. Namely,

$$R = [r_{ij}],\ f(r_{ij}) = \frac{1}{\sqrt{2\pi}} \exp\left(\frac{-r_{ij}^2}{2}\right)$$

where f is the probability density function of normal distribution.

We shall note that while we introduce Gaussian random matrices, there exist other types of random matrices. For example, we can have a Bernoulli random matrix where each entry follows a Bernoulli random variable. For this paper, we shall assume in later part of this text that all random matrices are Gaussian random matrices.

Next we will introduce some interesting properties about random matrices. First, we shall notice that since each entry of a random matrix $R \in \mathbb{R}^{m\times n}$ is an i.i.d Gaussian random variable, if $a \in \mathbb{R}^n$ is a vector, then $Ra \in \mathbb{R}^m$ is a vector of linear combinations of Gaussians. Therefore, each entry of Ra is also a Gaussian random variable. We now explain the relationship of the mean and variances between the random matrix R and that of the random vector Ra.

Lemma 4.3. *Let $R = [r_{ij}]$ be a $k \times p$ matrix, in which all entries are i.i.d. $N(0,1)$. Let the mapping $f : \mathbb{R}^p \to \mathbb{R}^k$ be defined by*

$$f(a) = \frac{1}{\sqrt{k}} Ra, \quad a \in \mathbb{R}^p. \tag{4.4}$$

Then we have

$$\mathbb{E}\left[\|f(a)\|_2^2\right] = \|a\|_2^2.$$

Proof. We shall first notice that $f(a)$ is a random vector in $\mathbb{R}^k$, of which all components are i.i.d. random variables with 0 mean and variance $\frac{1}{k}\|a\|_2^2$. We denote the ith row vector of R by r^i and denote the inner

product $\langle r^i, a\rangle$ by c_i. Then $f(a) = c = [c_1, \cdots, c_k]^T$. We have

$$\mathbb{E}[c_i] = \frac{1}{\sqrt{k}} \sum_{j=1}^{p} a_j \mathbb{E}[r_{ij}] = 0,$$

$$\mathbb{E}[c_i c_j] = \frac{1}{k} \sum_{m=1}^{p} \sum_{l=1}^{p} a_l a_m \mathbb{E}[r_{il} r_{jm}] = 0, \quad i \neq j,$$

and

$$\begin{aligned}
\mathbb{E}\left[c_i^2\right] &= \mathbb{E}\left[\left[\sum_{j=1}^{p} a_j\left(r_{ij}\right)\right]^2\right] \\
&= \frac{1}{k}\left(\sum_{j=1}^{p} a_i^2 \mathbb{E}\left[r_{ij}^2\right] + 2\sum_{l\neq m} a_l a_m \mathbb{E}[r_{il}]\,\mathbb{E}[r_{im}]\right) \\
&= \frac{1}{k}\sum_{j=1}^{p} a_i^2,
\end{aligned}$$

which yields that all entries of $f(a)$ are i.i.d. random variables with 0 mean and variance $\frac{1}{k}\|a\|_2^2$. Finally, we have

$$\mathbb{E}\left[\|f(a)\|_2^2\right] = \sum_{i=1}^{k} \mathbb{E}\left[c_i^2\right] = k\frac{\|a\|_2^2}{k} = \|a\|_2^2.$$

□

This lemma shows that for any random matrix $R = [r_{i,j}]$ with $\mathbb{E}[r_i j] = 0$ and $\mathrm{Var}(r_{i,j}) = 1$, we have $\mathbb{E}[\|Ra\|_2^2] = \|a\|_2^2$ for any vector a. In particular, let $R = [r_{ij}]$ be a $k \times D$ random matrix. If a is a unit vector, then $\sqrt{k}Ra$ is a k-dimensional random vector whose entries are i.i.d random variables with 0 mean and variance 1.

4.2. **Dimensionality Reduction with Random Projection.** Let $X = [x_1, x_2, \cdots, x_n]^T \in \mathbb{R}^{n\times p}$ be our high dimensional dataset. Random projection tries to project our high dimensional dataset to a lower dimension by a linear mapping f defined in (4.4).

We should first notice that for any random projection $f : \mathbb{R}^p \to \mathbb{R}^k$ where $k < p$, the projection f cannot be a Lipschitz embedding. This is because by the rank-nullity theorem, f has nontrivial null space. Namely, there exists nonzero $x_1 \neq x_2$ such that $f(x_1) = f(x_2) = 0$. Therefore we cannot find positive constants A and B such that

$$A\|x_1 - x_2\|_2^2 \leq \|f(x_1) - f(x_2)\|_2^2 \leq B\|x_1 - x_2\|_2^2$$

because $A\|x_1 - x_2\|_2^2$ is larger than 0 while $\|f(x_1) - f(x_2)\|_2^2$ is equal to 0.

Therefore, random projection method does not aim to find a Lipschitz embedding $f : \mathbb{R}^p \to \mathbb{R}^k$. Rather, it seeks to find an embedding that is Lipschitz on a subset $X \subset \mathbb{R}^p$ with a high probability such that this embedding, after projecting our original dataset to lower dimension k with some restrictions, preserves the pairwise distances between data points by restricting the pairwise distances between a lower and an upper bound.

Lemma 4.5 (Johnson and Lindenstrauss Lemma (JL Lemma))**.** *For any $0 < \varepsilon < 1$ and any integer $n > 0$, let k be a positive integer such that*

$$k \geq 4\left(\varepsilon^2/2 - \varepsilon^3/3\right)^{-1} \ln n \tag{4.6}$$

Then for any dataset $X \subset \mathbb{R}^p$ of n points, let $f : \mathbb{R}^p \to \mathbb{R}^k$ be defined as (4.4). Then for any two $x_i, x_j \in X$,

$$(1-\varepsilon)\|x_i - x_j\|_2^2 \leq \|f(x_i) - f(x_j)\|_2^2 \leq (1+\varepsilon)\|x_i - x_j\|_2^2 \tag{4.7}$$

holds with probability $1 - \frac{1}{n}$.

We shall notice the intuition behind the lower bound on k. In the inequality (4.6), ε is the error tolerance, which measures how tight the bound is on $\|f(x_i) - f(x_j)\|_2$, as stated in the inequality (4.7). As ε gets smaller, we allow for less error and $4\left(\varepsilon^2/2 - \varepsilon^3/3\right)^{-1} \ln n$ becomes bigger, and thus k has a higher lower bound. Similarly, as n gets larger, $4\left(\varepsilon^2/2 - \varepsilon^3/3\right)^{-1} \ln n$ also becomes bigger, and thus k has a higher lower bound. Intuitively, with less error or with more data points, our dimensionality reduction problem becomes harder, so we cannot project k to a very low dimension while still preserving pairwise distances well.

To prove the JL lemma, we need the following lemma first.

Lemma 4.8. *Let R be a $k \times p$ $(k \leq p)$ Gaussian random matrix and $a \in \mathbb{R}^p$ be a unit vector. Let $y = Ra$ and $1 < \beta$. Then*

$$\Pr\left[\|y\|_2^2 \leq \frac{k}{\beta}\right] < \exp\left(\frac{k}{2}\left(1 - \frac{1}{\beta} - \ln\beta\right)\right) \tag{4.9}$$

$$\Pr\left[\|y\|_2^2 \geq k\beta\right] < \exp\left(\frac{k}{2}(1 - \beta + \ln\beta)\right) \tag{4.10}$$

The proof of this lemma can be found in Chapter 7 Lemma 7.3 of the book [2].

Proof of JL Lemma. Let R be a $k \times p$ random matrix whose entries are i.i.d. random variables $\sim N(0,1)$. We want to show that the linear map R satisfies the inequality (4.7). For each pair of x_i and x_j in X with $x_i \neq x_j$, we set

$$a = \frac{x_i - x_j}{\|x_i - x_j\|_2}, \quad z = R(a) \quad y = \sqrt{k}z$$

Then

$$\begin{aligned}\Pr\left(\frac{\|f(x_i) - f(x_j)\|_2^2}{\|x_i - x_j\|_2^2} \leq (1-\varepsilon)\right) &= \Pr\left(\|z\|_2^2 \leq (1-\varepsilon)\right) \\ &= \Pr\left(\|y\|_2^2 \leq (1-\varepsilon)k\right).\end{aligned}$$

Similarly,

$$\Pr\left(\frac{\|f(x_i) - f(x_j)\|_2^2}{\|x_i - x_j\|_2^2} \geq (1+\varepsilon)\right) = \Pr\left(\|y\|_2^2 \geq (1+\varepsilon)k\right).$$

Applying formula (4.9) in Lemma 4.5 to y with $\frac{1}{\beta} = 1 - \varepsilon$, we have

$$\begin{aligned}\Pr\left(\|y\|_2^2 \leq (1-\varepsilon)k\right) &< \exp\left(\frac{k}{2}(1 - (1-\varepsilon) + \ln(1-\varepsilon))\right) \\ &\leq \exp\left(\frac{k}{2}\left(1 - (1-\varepsilon) - \left(\varepsilon - \frac{\varepsilon^2}{2}\right)\right)\right) \\ &= \exp\left(-\frac{k\varepsilon^2}{4}\right) \leq \exp(-2\ln n) = \frac{1}{n^2},\end{aligned}$$

where the inequality $\ln(1-\varepsilon) \leq -\varepsilon - \varepsilon^2/2$ is used in second line, and the condition

$$k \geq 4\left(\varepsilon^2/2 - \varepsilon^3/3\right)^{-1} \ln n$$

is applied to deriving the inequality in the last line. Similarly, we have

$$\begin{aligned}\Pr\left(\|y\|_2^2 \geq (1+\varepsilon)k\right) &< \exp\left(\frac{k}{2}(1 - (1+\varepsilon) + \ln(1+\varepsilon))\right) \\ &\leq \exp\left(\frac{k}{2}\left(-\varepsilon + \left(\varepsilon - \frac{\varepsilon^2}{2} + \frac{\varepsilon^3}{3}\right)\right)\right) \\ &= \exp\left(-\frac{k\left(\varepsilon^2/2 - \varepsilon^3/3\right)}{2}\right) \leq \exp(-2\ln n) = \frac{1}{n^2}.\end{aligned}$$

Therefore, for each pair x_i and x_j in X,

$$\Pr\left(\frac{\|f(x_i) - f(x_j)\|_2^2}{\|x_i - x_j\|_2^2} \notin [1-\varepsilon, 1+\varepsilon]\right) < \frac{2}{n^2}$$

Since the cardinality of X is n, there are a total of $n(n-1)/2$ pairs. The probability of the event that at least one pair does not satisfy Lemma 4.6 is less than $1 - \frac{n(n-1)}{2}\frac{2}{n^2} = 1/n > 0$. □

From the proof above, we can see that a random matrix R, while not a Lipschitz embedding on the whole vector space $\mathbb{R}^p$, can help preserve local properties of a dataset $X \subset \mathbb{R}^p$ with high probability. Specifically, if we have a dataset X with a very large number of observations n, then the probability that we can bound the distances of all pairs of $f(x_i), f(x_j), x_i, x_j \in X$ is $1 - \frac{1}{n}$.

With the information above, now we can construct the algorithm for random projections.

4.3. **Random Projection Algorithm.** The random projection algorithm consists of two steps: first, we create a random matrix R; and second, we multiply the created random matrix with our high-dimensional dataset X and output the projected dataset RX.

Let x_i, x_j be two data points from X. Recall the formula (4.7) in JL lemma,

$$(1-\varepsilon)\|x_i - x_i\|_2^2 \leq \|f(x_i) - f(x_j)\|_2^2 \leq (1+\varepsilon)\|x_i - x_j\|_2^2$$

We can rewrite the formula as

$$(1-\varepsilon)\|x_i - x_j\|_2^2 \leq \left\| \frac{1}{\sqrt{k}} Rx_i - \frac{1}{\sqrt{k}} Rx_j \right\|_2^2 \leq (1+\varepsilon)\|x_i - x_j\|_2^2$$

We can take the constant $\frac{1}{\sqrt{k}}$ out of the norm operator, and move $\frac{1}{k}$ the the other sides of the inequations,

$$(1-\varepsilon)\|x_i - x_j\|_2^2 \leq \frac{1}{k}\|Rx_i - Rx_j\|_2^2 \leq (1+\varepsilon)\|x_i - x_j\|_2^2$$

$$k(1-\varepsilon)\|x_i - x_j\|_2^2 \leq \|Rx_i - Rx_j\|_2^2 \leq k(1+\varepsilon)\|x_i - x_j\|_2^2$$

Therefore, the JL lemma guarantees that the random projection algorithm can bound the pairwise distances of our projected dataset RX with an upper bound $k(1+\varepsilon)$ and lower bound $k(1-\varepsilon)$ with a high probability of $1 - \frac{1}{n}$.

Acknowledgments

I give sincere thanks to my mentor Phillip Lo, who gives great guidance in helping me understand the materials and write the paper.

References

[1] Severn, Katie. Multivariate Statistics https://rich-d-wilkinson.github.io/MATH3030/

[2] Wang, Jianzhong. Geometric Structure of High-Dimensional Data and Dimensionality Reduction. Higher Education Press, 2012.