1 PCA qualitative

Statistics - A) Joe's first job as a statistician was to analyse the countries that won medals in the 2012 London Olympic Games. He started his task by collecting data from 84 countries that had won at least one medal. The data's features (dimensions) for each country (sample/observation) consisted of the number of medals won, the population size (in hundred thousands) and the GDP (in millions of US dollars) all taken in 2012. To start his analysis Joe ran PCA on the data.

1. The first eigenvector explained 99.9% of the data's variance. Joe was very happy! He was already planning on winning rivers of money by betting on the medals distribution of the 2016 Olympic Games. Assuming that Joe is a competent analyst why did he think he could predict the medals distribution of the next olympic games?

2. Joe's boss was not as happy. He ran the PCA by himself and he found that the first eigenvector explained only 74.0% of the data's variance. They compared what they both had done but the only difference they found was that Joe's boss used the GDP values in billions of dollars while Joe's GDP values were in millions of dollars. Why did they obtain different results? How should Joe and his Boss have treated the data before the PCA? What was probably the first eigenvector found by Joe?

3. Joe has finally learned the correct way to handle the data before running PCA. He was now trying to run the analysis with extra features (dimensions). He added the countries call prefix code (e.g. 41 for Switzerland) to the last column of the data. Table 1 shows the results that he obtained. How would you interpret these results?

	Medals	**Population**	**GDP**	**Codes**	**Variance Explained**
1^{st} **Eigenvector**	-0.619	-0.478	-0.621	-0.051	56%
2^{nd} **Eigenvector**	0.132	-0.263	0.149	-0.944	26%

Table 1: Joe's results.

Note: These results are based on the analysis of real data collected from `http://www.medalspercapita.com/#medals-per-capita:2012`. However, the countries call prefix code is fake, it was generated from random number generator with uniform distribution.

Dimensionality Reduction - B)

1. Jenny has a dataset of 10 000 images of faces. These are all black and white 100 x 200 px images where each pixel can have a value from 0 to 255 (1 byte) to represent the intensity. Because Jenny cares about reducing the dimension of each image but not to find common features across the images, she stores the image in a matrix X where each column is a different image and each row corresponds to each different pixel location[1].

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,10'000} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,10'000} \\ \vdots & \vdots & \vdots & \vdots \\ x_{20'000,1} & x_{20'000,2} & \cdots & x_{20'000,10'000} \end{bmatrix}$$

[1]The images are now represented as column vectors with $20'000$ (width $\times$ height) dimensions.

The dataset occupies 200MBs (1 (byte/pixel)×20′000 (pixels/image) ×10′000 (images) = 200 (MBs)) in memory. Jenny needs to compress them because she wants to make the dataset available on her internet server, and, if multiple users download it, she ends up paying a big internet bill. To find the optimal compression, she decides to run PCA on those images. She finds that the first 100 principal components explain 90% of the data's variance. She also observes that if she projects the images onto the first 100 eigenvectors and discards the other projections, her images lose resolution but remain readable. This project hence might pay off thanks to the generated decrease in memory space. Has Jenny really gained in memory? And if so, how much has she gained? How much memory does the dataset occupy now?

Hint: You can consider each float number takes 8 bytes (64bits) of space

Eigenvectors intuition for image data - C) Let's assume that we have the dataset of grayscale images of pens with either vertical or horizontal orientation (refer to Figure 1). We want to be able to classify whether a pen is rotated horizontally or vertically, using a PCA projection.

1. First step when doing PCA on any dataset is to center the data, via subtracting the mean. a) Try to sketch (i.e. draw approximately) the "mean" of the dataset presented in Figure 1. Assume that dataset is balanced, i.e. contains equal number of vertical and horizontal pens. b) How does "centered" image (i.e. after mean image subtraction) of a vertically (and horizontally) oriented pen looks?

   Hint: Assume that image is grayscale uint8, 0 is for black pixels, 255 is for white, all negative values are visualized as black. "Subtracting images" should be treated as pointwise pixel intensity v subtraction, i.e. if pixel is white ($v = 255$), then subtracting black ($v = 0$) pixel results in white pixel ($255 - 0 = 255$).

2. After centering the data, and computing the covariance matrix, one should perform eigenvalue decomposition, and use resulting eigenvectors as a projection basis for the data of reduced dimensions. Since the eigenvectors have the same dimensionality as the original images, they can be visualized, highlighting filtering features. How does such an eigenvector looks like? Try sketching two first eigenvectors of the projection basis.

3. Using these two eigenvectors as a new basis, plot two-dimensional projections of the images of vertically and horizontally oriented pens. Can they be separated easily? How many eigenvectors is required for these two classes to be linearly separable? Can you think of a single eigenvector, that can be used to separate two classes (horizontally and vertically oriented pens)?

Figure 1: Pen dataset images $i = 1..9$

Solutions

A)

1. Being a good analyst Joe would have checked if the data had similar ranges and spreads on all axes. Assuming he had done that, having an eigenvector explaining 99.9% of the

data's variance means that all factors in the analysis were linearly correlated. This means that, if we know the value of one variable, we can perfectly predict the value of all the other dimensions. Because of this Joe was planning on predicting the number of medals each country would get on the 2016 Olympics based on their population and/or GDP in that year.

2. It seems that Joe was not a good analyst after all. His first eigenvector was able to explain 99.9% of the data's variance just because the spread of the GDP variable was too high. When his boss ran the same analysis but with a smaller scale for the GDP variable he found an eigenvector that could only explain 74.0% of the data's variance. If we have a variable that has a much larger spread than the other variables, then the variance of the data along this dimension will be much larger than in the other dimensions. Therefore, the first principal component will be pointing in a direction close to that variable axis, in this case the direction [0,0,1]. To avoid these wrong conclusions, it is good practice to rescale the data to avoid order of magnitude differences across dimensions. This can be easily done by dividing each dimension by the standard deviation of the values of this dimension, normalizing their standard deviations, or simply by doing the eigenvalue decomposition on the correlation matrix as opposed to the covariance matrix.

   To get a geometric understanding of this scaling problem imagine that Joe samples data from two normal distributions with equal mean ($\mu_0 = \mu_1 = 0$) but with different scales ($\sigma_0 = 20, \sigma_1 = 1$). So that the two variables are $X \sim \mathcal{N}(0, 20)$ and $Y \sim \mathcal{N}(0, 1)$ (see Figure 2).

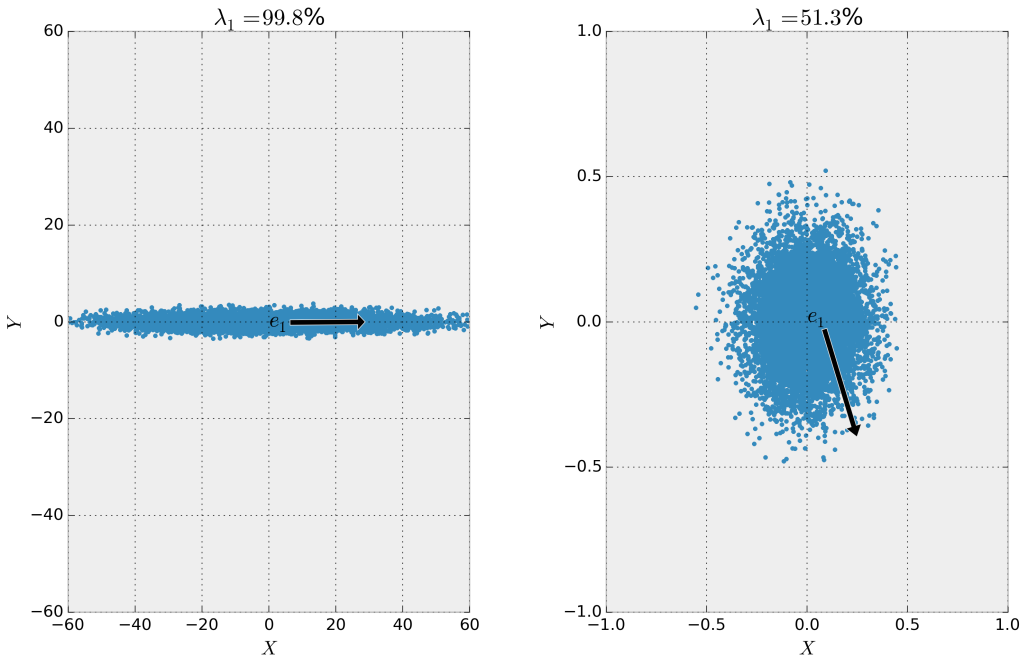


Figure 2: On the left, the principal component points in the direction of the x-axis and it explains 99.8% of the variance in the data. On the right, the data was now normalized and the principal component points in some other direction (all the directions are equally good specially for a large sample size) and it explains only 51.3% of the variance of the data.

   The PCA on the unnormalized data will return a principal component that points in the direction of the variable with the biggest scale and it will be able to explain most of the data's variance. Notice that the two variables are uncorrelated, the covariance matrix is

a 2 by 2 diagonal matrix with one value much bigger than the other. If we normalize the data (divide it by its standard deviation) before the PCA, we will not be able to predict the direction of the principal component (there is no preferential direction on uncorrelated data) and that vector will be only able to explain about 50% of the data's variance.

Note, however, that if you compare data that live in the same dimensions, e.g. when comparing distribution of datapoints in Cartesian space, you should not normalize your data, as the variance along each dimension bear meaningful information which you would loose through a re-scaling.

Re-scaling should be done only when comparing data which have different units and differ by one or more order of magnitude, e.g. when comparing measurements captured with force sensors (measured in Newton) and measurement from position sensors (measured in mm).

3. The fact that the two first eigenvectors are able to explain 82% of the data's variance variables (Table 1) is an indication that there is a correlation among the original dimensions. To know which variables are correlated, one must look at the coefficients of the eigenvectors on the original dimensions. Taking a look at these, we see that the first principal component (eigenvector) has large negative coefficients in all dimensions but the prefix codes. This reveals a correlation across the first three original dimensions, so across Medals, Population and GDP. One can see this by inspecting Figure 3 in which we see a strong correlation between GDP and Population. This is the same for the Medals (not shown here).

   There is, indeed, no reason for the countries call prefix code to be correlated with any of the other variables. Therefore, the information given by this variable is independent of the other dimensions and, unsurprisingly, we see that the second eigenvector points along the Codes axis (see Figure 4), and that the first eigenvector has no contribution from Codes. One can confirm this by looking at Figure 4 which shows the GDP-Codes slice of the data. This fact holds for Codes vs the other remaining dimensions (not shown here).

   For curiosity the normalized linear correlations between the variables are shown in Table 2.

	Medals	**Population**	**GDP**	**Codes**
Medals	1.00	0.47	0.88	0.00
Population	0.47	1.00	0.48	0.15
GDP	0.88	0.48	1.00	-0.02
Codes	0.00	0.15	-0.02	1.00

Table 2: Normalized linear correlations between the variables

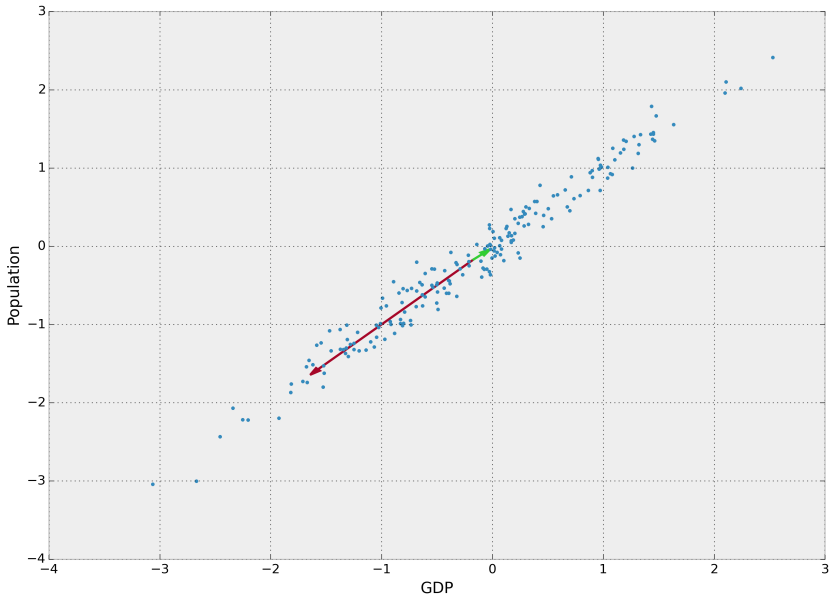


Figure 3: The first eigenvector (in red) has similar contributions from both the GDP and the population variables. The second eigenvector (in green) is very small when projected to this space because it is almost orthogonal to it.

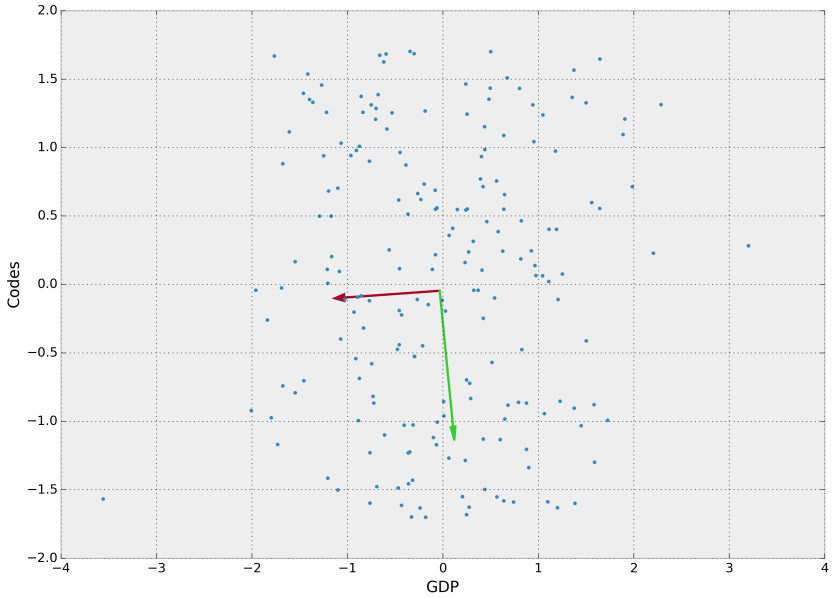


Figure 4: There is no correlation between GDP and Codes. As we can see the second eigenvector (green) is nearly completely aligned with the Codes dimension.

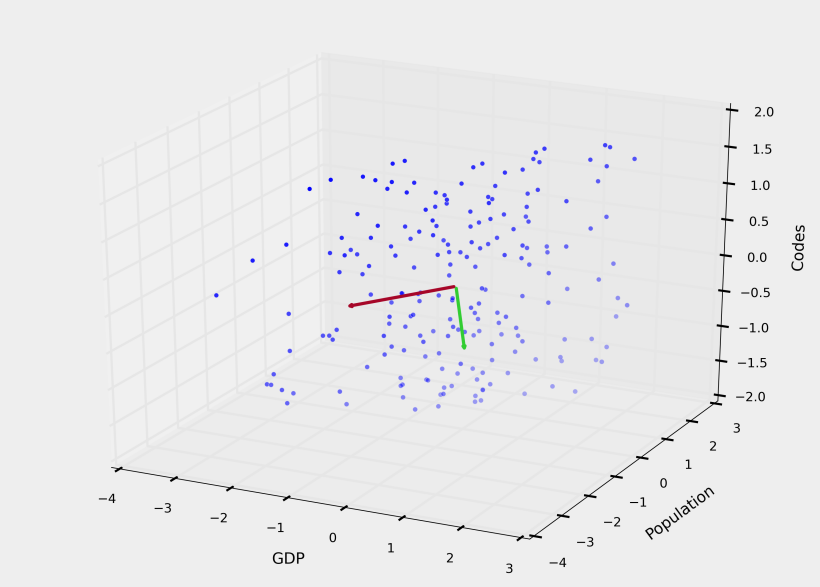


Figure 5: We can see that the Codes distribution is uniform whilst the GDP and Population distribution is not.

B) There are three solutions to this problem depending on what Jenny aims to do.

Global Data Compression:

Jenny considers this as a global dimensional reduction task and opts to use PCA to reduce the dimensionality of all image data set at once.

For doing this, she first subtracts the mean from her data set $X = X - \mathbb{E}\{X\}$ such that it is centred and then proceeds to compute the sample covariance matrix $C = \mathbb{E}\{XX^T\}$, with $C \in \mathbb{R}^{20'000 \times 20'000}$.

Jenny does an eigen decomposition of $C = V\Lambda V^{\mathrm{T}}$ where Λ is a diagonal matrix in which the entries are the eigenvalues and V is a matrix whose columns are the eigenvectors $V = [e^1, ..., e^{20'000}]$, $e^1 \in \mathbb{R}^{20'000 \times 1}$. Jenny's hope is that she can find a linear projection which can reduce the dimensionality of her data, $Y = AX$ where $A = V^{\mathrm{T}}$.

By inspecting the eigenvalues she finds that a 100 of them explains 90% of her data sets variance. She decides to discard the remaining eigenvectors obtaining the projection matrix A_p for which $p = 100$, ($A_p \in \mathbb{R}^{100 \times 20'000}$).

This implies that there exists a 100 dimensional subspace (given the 20′000 original space) in which the projected data retains 90% of the information. As Jenny wants to simply compress the information and make only the compressed data available through the net, she will project all the images to the 100 dimensional subspace and discard the eigenvectors.

$$Y = A_p X \tag{1}$$

$$Y \in \mathbb{R}^{100 \times 10'000} \tag{2}$$

$$A_p \in \mathbb{R}^{100 \times 20'000} \tag{3}$$

$$X \in \mathbb{R}^{20'000 \times 10'000} \tag{4}$$

After this PCA step the projection matrix A_p and the original data X are no longer needed, leaving Jenny with Y. The total amount of memory saved would be $(20'000 \times 10'000) - (8 \times 100 \times 10'000) = 192$Mb. This gives a space saving of $96\% = 192/200$.

Remote Data Un-compression

Let us now consider the case where Jenny is interested in not loosing any resolution in her images and allow image processing enthusiasts to recover the images in close to full resolution.

As she also wants to minimise the amount of bandwidth she uses when sending her dataset over the internet, she gathers that it is more efficient to transmit A_p, Y and $\mathbb{E}\{X\}$ rather than simply sending the raw data set X (here, she uses the compression obtained in the global compression approach). When the enthusiast receives Y, A_p and $\mathbb{E}\{X\}$ he can then reconstruct a lossy version of the original data set of images $\hat{X}$ through mean-square projection.

$$A_p^{-1}Y + \mathbb{E}\{X\} = \hat{X} \tag{5}$$

The reason we have to add the mean back to the re-projected data is because we originally centred it. If the data was already centred there would be no need to store or add the mean since it would be zero.

Note also that the reason that this is a lossy compression $\hat{X} \neq X$ is because the explained variance with $p = 100$ is not 100%. If this were the case then there would be lossless compression and $\hat{X} = X$.

Jenny would now have to store the variables A_p, Y and $\mathbb{E}\{X\}$ which now are represented by floating numbers instead of integers. So each element of the variables will take 8bytes of space instead of 1byte.

A_p:	$(8 \times 100 \times 20'000)$	$= 16$ MBs
Y:	$(8 \times 100 \times 10'000)$	$= 8$ MBs
$\mathbb{E}\{X\}$:	$(8 \times 1 \times 20'000)$	$= 0.16$ MBs

Table 3: Memory requirement per transmitted variable.

The total amount of memory the data set occupies now is 24.16MBs as opposed to the original 200MBs giving Jenny 88% compression factor.

Local Data Compression

In the previous solutions, Jenny had assumed that all data shared some common properties and hence that computing an eigenvalue decomposition on the entire dataset would be beneficial to extract these common properties and remove the distracting "noise" information. This way of proceeding is good only if the dataset is quite homogeneous, e.g. dataset of faces of people, where the face is nicely centered. If the dataset is composed of an heterogeneous dataset with all sorts of images, then, likely there is not much common across the images and the PCA will not yield a very good compression.

In this case, Jenny should rather treat each image separately and apply PCA on each image (as shown in the example of the class's slides). To compute PCA on each image **separately**, Jenny must compute 10'000 sample covariance matrices $C_i \in \mathbb{R}^{100\times100}$ for $i = 1 : 10'000$, given that the image columns are the datapoints (recall that image dimensionality is 100×200 pixels). She would first subtract the mean from each image $\hat{x_i} = x_i - \mathbb{E}\{x_i\}$ such that it is centred and proceeds to compute the sample covariance matrix $C_i = \mathbb{E}\{x_i x_i^T\}$. Then, for each image, she would select the subset of eigenvectors that yields the smallest reconstruction error. The number of eigenvectors may vary across each image. She would then project each image on its selected set of eigenvectors. The advantage of this procedure is that it yields the optimal tradeoff between dimensionality reduction and image resolution. Often, this process leads to a more efficient datacompression of the order of 90%. In other words, each projection would require much less than the 100 eigenvectors Jenny found in the global data compression approach.

However, the drawback is that a) the images end up having a different resolution; b) the process is costly (computing 10'000 times PCA in place of once).

Figure 6: Pen dataset (repetition for solutions)

C) The mean of the data looks like a cross-shape as in Figure 7. Notice that it incorporates both possible pen orientations, and, importantly, captures repeating features, such as background, light from the flash, etc, allowing for residual images to only carry non-repetitive features.

Figure 7: Average image

Images after substracting the mean are presented in Figure 8. Consider horizontal pen. Background becomes 0 (i.e. black), as it's shared across all images. Pen position (horizontal line in the center) was black already, so remains dark. However, the pixels describing vertical pen, had low value in the mean image, thus the background is not removed there. So, horizontal pen after centering looks like bright vertical line (and vertical pen looks like bright horizontal line).

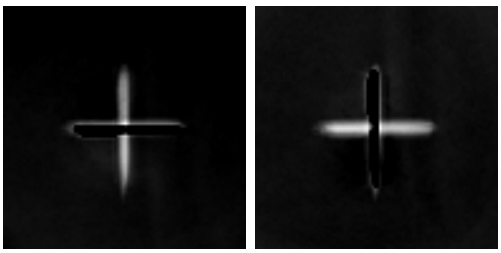


Figure 8: Centered images: horizontal pen(left) and vertical pen(right)

The first five eigenvectors are presented in Figure 9. Note that they all combine horizontal and vertical lines. It is not trivial to sketch an eigenvector even for a such simple scenario. Intuitively one could assume that first eigenvector will have only horizontal pattern, and second - only vertical (or vice versa), however this is not the case. In fact, the first eigenvector combines both these features, vertical pixels having negative values (shown black), and horizontal pixels being positive (in white). This way, all horizontal pens (that are represented by high values on vertical axis) will have large negative e_1 coordinate, and vertical pens (that are actually bright horizontal lines after substracting the mean) - large positive e_1 coordinate. Figure 10 shows the projected coordinates of images from Figure 1. Note, that the second eigenvector focuses on various features in horizontal area, thus cluster with largest distribution along second eigenvector corresponds to horizontally placed pens, and values along e_2 are proportional to the placement of the pen - slightly higher or slightly lower than the center.

Figure 9: Five first eigenvectors (first on the left)

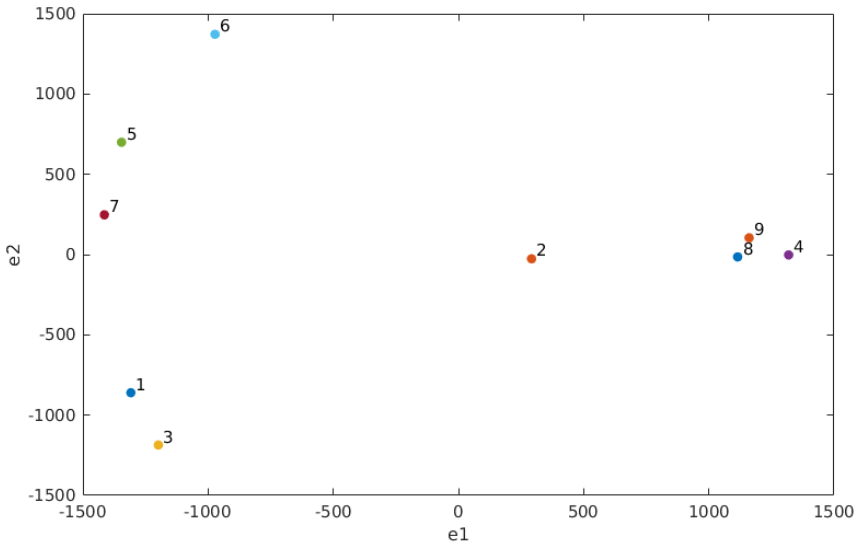
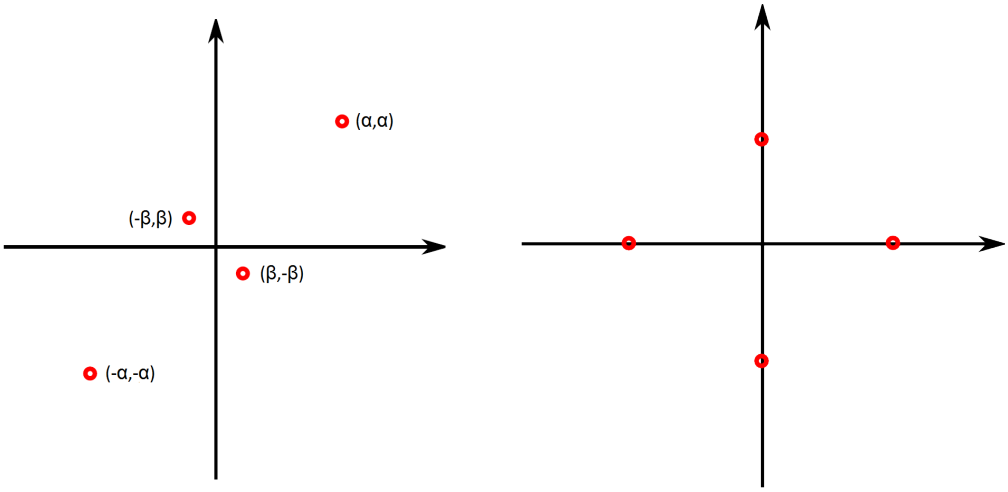


Figure 10: Projected coordinates of the images from Figure 1

2 PCA quantitative



(a) (b)

A) Let us go through through the important steps of PCA:

1. Compute the covariance matrix, C, of the dataset composed of the four points shown in the left figure $((\alpha\ \alpha), (-\alpha\ -\alpha)\ , (-\beta\ \beta)\ , (\beta\ -\beta))$.

2. Compute the eigenvectors and eigenvalues of C. Discuss how the eigenvectors and eigenvalues are affected by α and β.

3. Compute the projection of the dataset on the principal directions. Then reduce the dimensionality of the dataset by discarding the less relevant dimension in the projected space (assuming $\alpha > \beta$).

4. From the reduced dataset, reconstruct an approximation of the dataset in the original referential. Compute the average reconstruction error

$$E = \frac{1}{M}\sum_{i=1}^{M} ||\boldsymbol{x}^i - \hat{\boldsymbol{x}}^i||^2$$

   where $\boldsymbol{x}^i$ are the original data points and $\hat{\boldsymbol{x}}^i$ their approximation, and discuss its relation to the eigenvalues of C.

5. For the data from the right figure, what structure (isotropic, diagonal, full, symmetric) does the covariance matrix have?

B) Show that when an N-dim set of data points X is projected onto the eigenvectors $V = [\mathbf{e}^1, \cdots, \mathbf{e}^N]$ of its covariance matrix, $C = XX^{\mathrm{T}}$, the covariance matrix YY^{T} of the projected data Y is diagonal and hence that, in the space of the eigenvector decomposition, the distribution of X is uncorrelated.

Solutions

A)

1. The empirical covariance matrix is given by

$$\begin{aligned} C &= \frac{1}{M}\sum_{i=1}^{M}\left(\mathbf{x}^i-\boldsymbol{\mu}\right)\left(\mathbf{x}^i-\boldsymbol{\mu}\right)^{\mathrm{T}} \\ &= \frac{1}{M}X'X'^{\mathrm{T}}. \end{aligned}$$

with $X' = X - \boldsymbol{\mu}$.

Using the values of the given data points we get (as $\boldsymbol{\mu} = \mathbf{0}$)

$$\begin{aligned} C &= \frac{1}{4}\begin{bmatrix}\alpha & -\alpha & -\beta & \beta \\ \alpha & -\alpha & \beta & -\beta\end{bmatrix}\begin{bmatrix}\alpha & \alpha \\ -\alpha & -\alpha \\ -\beta & \beta \\ \beta & -\beta\end{bmatrix} \\ &= \frac{1}{2}\begin{bmatrix}\alpha^2+\beta^2 & \alpha^2-\beta^2 \\ \alpha^2-\beta^2 & \alpha^2+\beta^2\end{bmatrix} \end{aligned}$$

2. The eigenvalues λ^i and eigenvectors $\boldsymbol{e}^i$ are defined by the following property:

$$\begin{aligned} C\boldsymbol{e}^i &= \lambda^i\boldsymbol{e}^i \\ \Rightarrow \left(C-\lambda^i I\right)\boldsymbol{e}^i &= \mathbf{0} \end{aligned}$$

For a non-zero vector, $\mathbf{e}^i$, to satisfy this equation the determinant $|(C - \lambda I)| = 0$, meaning that it cannot have an inverse. If it did have an inverse then $\mathbf{v}$=0 which is the degenerate case.

$$\begin{aligned} \begin{vmatrix} \frac{1}{2}\alpha^2+\frac{1}{2}\beta^2-\lambda^i & \frac{1}{2}\alpha^2-\frac{1}{2}\beta^2 \\ \frac{1}{2}\alpha^2-\frac{1}{2}\beta^2 & \frac{1}{2}\alpha^2+\frac{1}{2}\beta^2-\lambda^i \end{vmatrix} &= 0 \\ \Rightarrow (\frac{1}{2}\alpha^2+\frac{1}{2}\beta^2-\lambda^i)^2-(\frac{1}{2}\alpha^2-\frac{1}{2}\beta^2)^2 &= 0 \\ \Rightarrow {\lambda^i}^2+\lambda^i(\alpha^2+\beta^2)+(\alpha^2\beta^2) &= 0 \\ \Rightarrow \lambda^i &= \frac{(\alpha^2+\beta^2)}{2}\pm\frac{(\alpha^2-\beta^2)}{2} \\ \Rightarrow \lambda^1=\alpha^2 \qquad \lambda^2=\beta^2 & \end{aligned}$$

Now the eigenvectors are simply the solutions to $\left(C-\lambda^i I\right)\boldsymbol{e}^i = \mathbf{0}$.

For λ^1:

$$\begin{aligned} \frac{1}{2}\begin{bmatrix}-\alpha^2+\beta^2 & \alpha^2-\beta^2 \\ \alpha^2-\beta^2 & -\alpha^2+\beta^2\end{bmatrix}\begin{bmatrix}e_1^1 \\ e_2^1\end{bmatrix} &= \begin{bmatrix}0 \\ 0\end{bmatrix} \\ \Rightarrow e_1^1 &= e_2^1 \end{aligned}$$

Additionally, the eigenvectors must be normalized (i.e $||\boldsymbol{e}^i|| = 1$)

$$\begin{aligned} \sqrt{{e_1^1}^2+{e_2^1}^2} &= 1 \\ \Rightarrow \boldsymbol{e}^1 &= \begin{bmatrix}\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\end{bmatrix}^T \end{aligned}$$

For λ^2:

$$\begin{aligned}\frac{1}{2}\begin{bmatrix}\alpha^2-\beta^2 & \alpha^2-\beta^2\\ \alpha^2-\beta^2 & \alpha^2-\beta^2\end{bmatrix}\begin{bmatrix}e_1^2\\ e_2^2\end{bmatrix} &= \begin{bmatrix}0\\0\end{bmatrix}\\ \Rightarrow e_1^2 &= -e_2^2\\ \Rightarrow \boldsymbol{e}^2 &= \begin{bmatrix}\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}}\end{bmatrix}^T\end{aligned}$$

The eigenvectors are unaffected by α and β, the direction of the data's spread remains the same regardless of their value.The size of the eigenvalues is proportional α^2 and β^2. This is due to the increase of the spread of the data.

3. Each data point is projected by multiplying its vector with the projection matrix A, which is built by pilling up the eigenvectors:

$$\boldsymbol{y^i} = A\boldsymbol{x}^i = \begin{bmatrix}\boldsymbol{e^1}^T\\ \boldsymbol{e^2}^T\end{bmatrix}\boldsymbol{x}^i$$

For the first point (α, α):

$$\boldsymbol{y^1} = \begin{bmatrix}\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}}\end{bmatrix}\begin{bmatrix}\alpha\\ \alpha\end{bmatrix} = \begin{bmatrix}\sqrt{2}\alpha\\ 0\end{bmatrix}$$

The same procedure applied to the other points yields:

$$\boldsymbol{y^2} = \begin{bmatrix}-\sqrt{2}\alpha\\ 0\end{bmatrix} \quad \boldsymbol{y^3} = \begin{bmatrix}0\\ -\sqrt{2}\beta\end{bmatrix} \quad \boldsymbol{y^4} = \begin{bmatrix}0\\ \sqrt{2}\beta\end{bmatrix}$$

The less relevant dimension is the dimension given by the eigenvector with the smallest eigenvalue, which in our case is $\boldsymbol{e}^2$:

This yields:

$$\boldsymbol{y^1} = \begin{bmatrix}\sqrt{2}\alpha\end{bmatrix} \quad \boldsymbol{y^2} = \begin{bmatrix}-\sqrt{2}\alpha\end{bmatrix} \quad \boldsymbol{y^3} = \begin{bmatrix}0\end{bmatrix} \quad \boldsymbol{y^4} = \begin{bmatrix}0\end{bmatrix}$$

4. We use the transpose of the reduced projection matrix to project the reduced dataset back to its original space

$$\boldsymbol{\hat{x}^i} = A_{red}^T\boldsymbol{y}^i = \left[\boldsymbol{e^1}^T\right]^T\boldsymbol{y^i}$$

For the first data point this is :

$$\boldsymbol{\hat{x}^1} = \begin{bmatrix}\frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\end{bmatrix}\begin{bmatrix}\sqrt{2}\alpha\end{bmatrix} = \begin{bmatrix}\alpha\\ \alpha\end{bmatrix}$$

The other reconstructed points are:

$$\boldsymbol{\hat{x}^2} = \begin{bmatrix}-\alpha\\ -\alpha\end{bmatrix} \quad \boldsymbol{\hat{x}^3} = \begin{bmatrix}0\\ 0\end{bmatrix} \quad \boldsymbol{\hat{x}^4} = \begin{bmatrix}0\\ 0\end{bmatrix}$$

The average reconstruction error is $\frac{1}{M}\sum_{i=1}^{M}||\boldsymbol{x}^i - \boldsymbol{\hat{x}^i}||^2 = \beta^2$, which is equal to the eigenvalue of the discarded projection dimension.

5. The covariance matrix is diagonal, it is aligned with the axis and as all covariance matrices are, it is symmetric. Its form would be diagonal with unequal values at the diagonal.

B)

$$\begin{aligned} YY^{\mathrm{T}} &= V^{\mathrm{T}}X\left(V^{\mathrm{T}}X\right)^{\mathrm{T}} \\ &= V^{\mathrm{T}}XX^{\mathrm{T}}V \\ &= V^{\mathrm{T}}V\Lambda V^{\mathrm{T}}V \\ &= I\Lambda I \\ &= \Lambda \end{aligned}$$

Λ is a diagonal matrix composed of the eigenvalue of XX^{T}.