Lecture 12: Principal Components and Eigenfaces

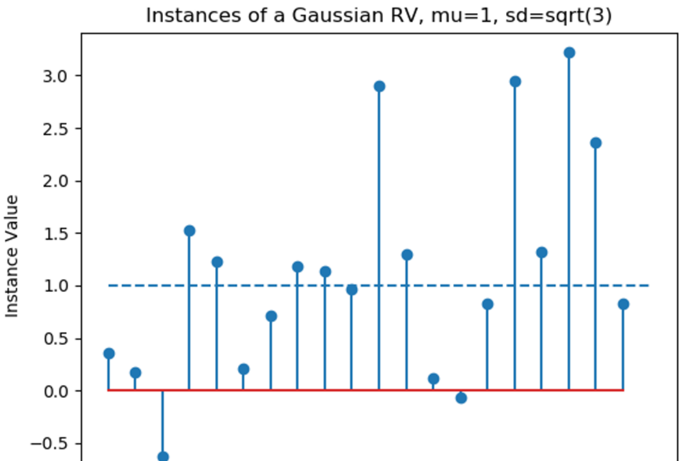
Mark Hasegawa-Johnson

ECE 417: Multimedia Signal Processing, Fall 2021

Outline

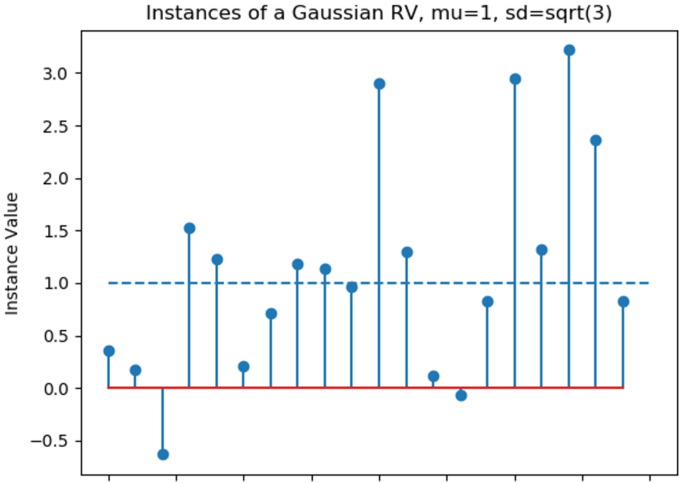
Scalar Gaussian random variables

$$p_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$



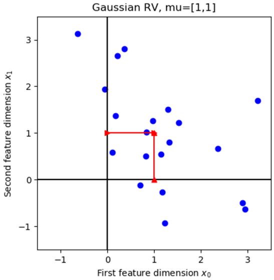
Scalar Gaussian random variables

$$\mu = E[X], \quad \sigma^2 = E[(X-\mu)^2]$$



Gaussian random vector

$$p_{\vec{X}}(\vec{x}) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(\vec{x}-\vec{\mu})^T\Sigma^{-1}(\vec{x}-\vec{\mu})}$$

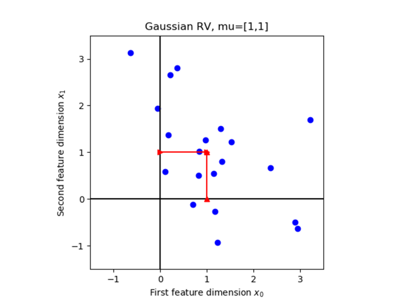


Gaussian random vector

$$\vec{x} = \begin{bmatrix} x_0 \\ \cdots \\ x_{D-1} \end{bmatrix}$$

$$\vec{\mu} = E[\vec{x}] = \begin{bmatrix} \mu_0 \\ \cdots \\ \mu_{D-1} \end{bmatrix}$$

Example: Instances of a Gaussian random vector



Gaussian random vector

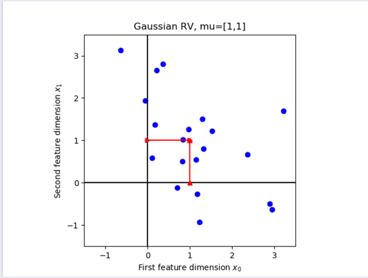
$$\Sigma = \begin{bmatrix} \sigma_0^2 & \rho_{0,1} & \ddots \\ \rho_{1,0} & \ddots & \rho_{D-2,D-1} \\ \ddots & \rho_{D-1,D-2} & \sigma_{D-1}^2 \end{bmatrix}$$

where

$$\rho_{ij} = E[(x_i - \mu_i)(x_j - \mu_j)]$$

$$\sigma_i^2 = E[(x_i - \mu_i)^2]$$

Example: Instances of a Gaussian random vector



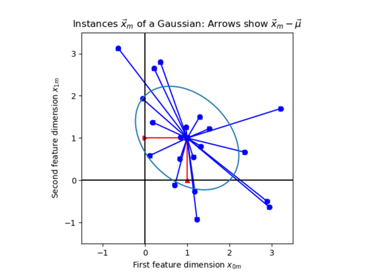
Maximum Likelihood Parameter Estimation

In the real world, we don't know $\vec{\mu}$ and Σ!
If we have a training database $\mathcal{D} = \{\vec{x}_0, \ldots, \vec{x}_{M-1}\}$, we can estimate $\vec{\mu}$ and Σ according to

$$\left\{\hat{\mu}_{ML}, \hat{\Sigma}_{ML}\right\} = \text{argmax} \prod_{m=0}^{M-1} p(\vec{x}_m | \vec{\mu}, \Sigma)$$

$$= \text{argmax} \sum_{m=0}^{M-1} \ln p(\vec{x}_m | \vec{\mu}, \Sigma)$$

Examples of $\vec{x}_m - \vec{\mu}$



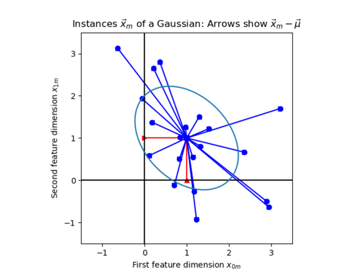
Maximum Likelihood Parameter Estimation

If you differentiate the RHS on the previous slide, and set it to zero, you find that the maximum likelihood solution is

$$\hat{\mu}_{ML} = \frac{1}{M}\sum_{m=0}^{M-1}\vec{x}_m$$

$$\hat{\Sigma}_{ML} = \frac{1}{M}\sum_{m=0}^{M-1}(\vec{x}_m-\vec{\mu})(\vec{x}_m-\vec{\mu})^T$$

Examples of $\vec{x}_m - \vec{\mu}$



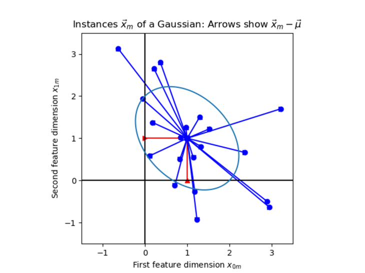
Sample Mean, Sample Covariance

The ML estimate of Σ is usually too small. It is better to adjust it slightly. The following are the **unbiased estimators** of $\vec{\mu}$ and Σ, also called the **sample mean** and **sample covariance**:

$$\vec{\mu} = \frac{1}{M}\sum_{m=0}^{M-1}\vec{x}_m$$

$$\Sigma = \frac{1}{M-1}\sum_{m=0}^{M-1}(\vec{x}_m - \vec{\mu})(\vec{x}_m - \vec{\mu})^T$$

Examples of $\vec{x}_m - \vec{\mu}$



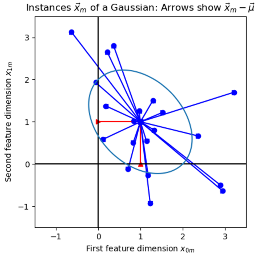
Sample Mean, Sample Covariance

$$\vec{\mu} = \frac{1}{M}\sum_{m=0}^{M-1}\vec{x}_m$$

$$\Sigma = \frac{1}{M-1}\sum_{m=0}^{M-1}(\vec{x}_m-\vec{\mu})(\vec{x}_m-\vec{\mu})^T$$

Sample mean and sample covariance are not the same as real mean and real covariance, but we'll use the same letters ($\vec{\mu}$ and Σ) unless the problem requires us to distinguish.

Examples of $\vec{x}_m - \vec{\mu}$



Outline

Review: Eigenvalues and eigenvectors

The right eigenvectors of a $D \times D$ square matrix, A, are the vectors $\vec{v}$ such that

$$A\vec{v} = \lambda\vec{v} \tag{1}$$

The scalar, λ, is called the eigenvalue. It's only possible for Eq. (1) to have a solution if

$$|A - \lambda I| = 0 \tag{2}$$

Left and right eigenvectors

We've been working with right eigenvectors and right eigenvalues:

$$A\vec{v}_d = \lambda_d \vec{v}_d$$

There may also be left eigenvectors, which are row vectors $\vec{u}_d$ and corresponding left eigenvalues κ_d:

$$\vec{u}_d^T A = \kappa_d \vec{u}_d^T$$

Eigenvectors on both sides of the matrix

You can do an interesting thing if you multiply the matrix by its eigenvectors both before and after:

$$\vec{u}_i^T(A\vec{v}_j) = \vec{u}_i^T(\lambda_j\vec{v}_j) = \lambda_j\vec{u}_i^T\vec{v}_j$$

...but...

$$(\vec{u}_i^T A)\vec{v}_j = (\kappa_i\vec{u}_i^T)\vec{v}_j = \kappa_i\vec{u}_i^T\vec{v}_j$$

There are only two ways that both of these things can be true. Either

$$\kappa_i = \lambda_j \quad \text{or} \quad \vec{u}_i^T\vec{v}_j = 0$$

Left and right eigenvectors must be paired!!

There are only two ways that both of these things can be true. Either

$$\kappa_i = \lambda_j \quad \text{or} \quad \vec{u}_i^T \vec{v}_j = 0$$

That means, if the eigenvalues are **distinct**, then there is **at most one** λ_i that can equal each κ_i:

$$\begin{cases} i \neq j & \vec{u}_i^T \vec{v}_j = 0 \\ i = j & \kappa_i = \lambda_i \end{cases}$$

Symmetric matrices: left=right

If A is symmetric ($A = A^T$), then the left and right eigenvectors and eigenvalues are the same, because

$$\lambda_i \vec{u}_i^T = \vec{u}_i^T A = (A^T \vec{u}_i)^T = (A\vec{u}_i)^T$$

... and that last term is equal to $\lambda_i \vec{u}_i^T$ if and only if $\vec{u}_i = \vec{v}_i$.

Symmetric matrices: eigenvectors are orthonormal

Let's combine the following facts:

- $\vec{u}_i^T \vec{v}_j = 0$ for $i \neq j$ — any square matrix with distinct eigenvalues
- $\vec{u}_i = \vec{v}_i$ — symmetric matrix
- $\vec{v}_i^T \vec{v}_i = 1$ — standard normalization of eigenvectors for any matrix (this is what $\|\vec{v}_i\| = 1$ means).

Putting it all together, we get that

$$\vec{v}_i^T \vec{v}_j = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

The eigenvector matrix

So if A is symmetric with distinct eigenvalues, then its eigenvectors are orthonormal:

$$\vec{v}_i^T \vec{v}_j = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

We can write this as

$$V^T V = I$$

where

$$V = [\vec{v}_0, \ldots, \vec{v}_{D-1}]$$

The eigenvector matrix is orthonormal

$$V^T V = I$$

... and it also turns out that

$$VV^T = I$$

Proof: $VV^T = VIV^T = V(V^T V)V^T = (VV^T)^2$, but the only matrix that satisfies $VV^T = (VV^T)^2$ is $VV^T = I$.

Eigenvectors orthogonalize a symmetric matrix

So now, suppose A is symmetric:

$$\vec{v}_i^T A \vec{v}_j = \vec{v}_i^T(\lambda_j \vec{v}_j) = \lambda_j \vec{v}_i^T \vec{v}_j = \begin{cases} \lambda_j, & i = j \\ 0, & i \neq j \end{cases}$$

In other words, if a symmetric matrix has D eigenvectors with distinct eigenvalues, then its eigenvectors orthogonalize A:

$$V^T A V = \Lambda$$

$$\Lambda = \begin{bmatrix} \lambda_0 & 0 & 0 \\ 0 & \ldots & 0 \\ 0 & 0 & \lambda_{D-1} \end{bmatrix}$$

A symmetric matrix is the weighted sum of its eigenvectors:

One more thing. Notice that

$$A = VV^TAVV^T = V\Lambda V^T$$

The last term is

$$[\vec{v}_0, \ldots, \vec{v}_{D-1}] \begin{bmatrix} \lambda_0 & 0 & 0 \\ 0 & \ldots & 0 \\ 0 & 0 & \lambda_{D-1} \end{bmatrix} \begin{bmatrix} \vec{v}_0^T \\ \vdots \\ \vec{v}_{D-1}^T \end{bmatrix} = \sum_{d=0}^{D-1} \lambda_d \vec{v}_d \vec{v}_d^T$$

Summary: properties of symmetric matrices

If A is symmetric with D eigenvectors, and D distinct eigenvalues, then

$$A = V\Lambda V^T$$

$$\Lambda = V^T A V$$

$$VV^T = V^T V = I$$

Outline

How do you classify an image?

Suppose we have a test image, $\vec{x}_{\text{test}}$. We want to figure out: who is this person?

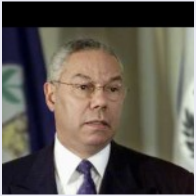
Test Datum $\vec{x}_{\text{test}}$:

Training Data?

In order to classify the test image, we need some training data. For example, suppose we have the following four images in our training data. Each image, $\vec{x}_m$, comes with a label, y_m, which is just a string giving the name of the individual.

Training Datum: y_0 =Colin Powell:

$\vec{x}_0 =$



Training Datum y_1 =Gloria Arroyo:

$\vec{x}_1 =$

Training Datum y_2 =Megawati Sukarnoputri:

$\vec{x}_2 =$

Training Datum y_3 =Tony Blair:

$\vec{x}_3 =$

Nearest Neighbors Classifier

A "nearest neighbors classifier" makes the following guess: the test vector is an image of the same person as the closest training vector:

$$\hat{y}_{\text{test}} = y_{m^*}, \quad m^* = \underset{m=0}{\overset{M-1}{\text{argmin}}} \|\vec{x}_m - \vec{x}_{\text{test}}\|$$

where "closest," here, means Euclidean distance:

$$\|\vec{x}_m - \vec{x}_{\text{test}}\| = \sqrt{\sum_{d=0}^{D-1} (x_{md} - x_{\text{test},d})^2}$$

Improved Nearest Neighbors: Eigenface

- The problem with nearest-neighbors is that subtracting one image from another, pixel-by-pixel, results in a measurement that is dominated by noise.
- We need a better measurement.
- The solution is to find a signal representation, $\vec{y}_m$, such that $\vec{y}_m$ summarizes the way in which $\vec{x}_m$ differs from other faces.
- If we find $\vec{y}_m$ using principal components analysis, then $\vec{y}_m$ is called an "eigenface" representation.

Outline

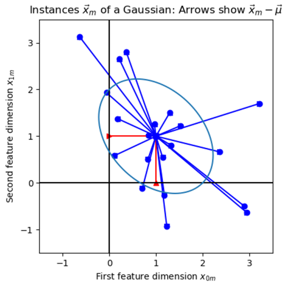
Sample covariance

$$\Sigma = \frac{1}{M-1}\sum_{m=0}^{M-1}(\vec{x}_m-\vec{\mu})(\vec{x}_m-\vec{\mu})^T$$
$$= \frac{1}{M-1}X^TX$$

...where X is the centered data matrix,

$$X = \begin{bmatrix} (\vec{x}_0-\vec{\mu})^T \\ \vdots \\ (\vec{x}_{M-1}-\vec{\mu})^T \end{bmatrix}$$

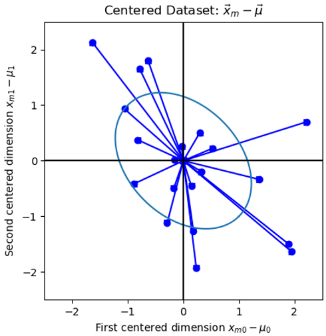
Examples of $\vec{x}_m - \vec{\mu}$



Centered data matrix

$$X = \begin{bmatrix} (\vec{x}_0 - \vec{\mu})^T \\ \vdots \\ (\vec{x}_{M-1} - \vec{\mu})^T \end{bmatrix}$$

Examples of $\vec{x}_m - \vec{\mu}$

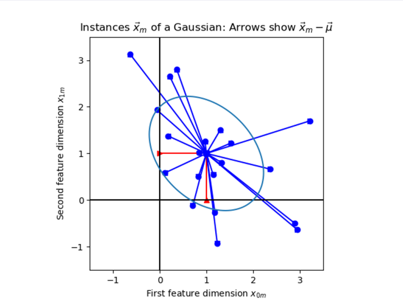


Sample covariance

$$\Sigma = \frac{1}{M-1} X^T X$$

The matrix $X^T X$ is called the **sum-of-squares (SS)** matrix. It is related to the **sample covariance** matrix by a scalar multiplier $(M-1)$.

Examples of $\vec{x}_m - \vec{\mu}$



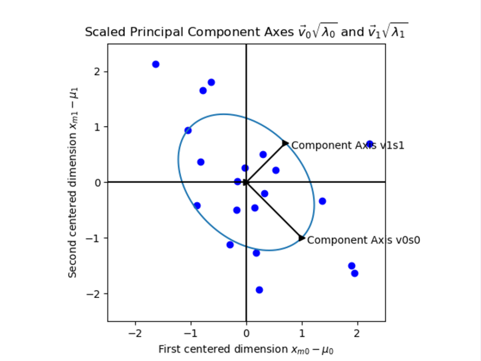
Principal component axes

X^TX is symmetric! Therefore,

$$X^TX = V\Lambda V^T$$

$V = [\vec{v}_0, \ldots, \vec{v}_{D-1}]$, the eigenvectors of X^TX, are called the principal component axes, or principal component directions.

Principal component axes



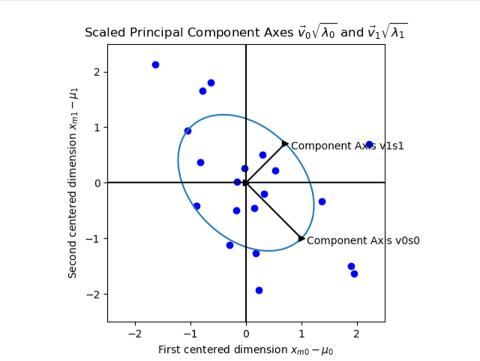
Principal component axes

$\Sigma = \frac{1}{M-1} X^T X$, therefore

$$\Sigma = V\left(\frac{1}{M-1}\Lambda\right)V^T$$

$V = [\vec{v}_0, \ldots, \vec{v}_{D-1}]$ are the eigenvectors of both the **sum-of-squares matrix** and the **covariance matrix**. Λ are the eigenvalues of the sum-of-squares matrix, equal to $M-1$ times the eigenvalues of the covariance matrix.

Principal component axes



Principal components

Remember that the eigenvectors of a matrix diagonalize it. So if V are the eigenvectors of X^TX, then

$$V^TX^TXV = \Lambda$$

$$\vec{v}_i^T X^T X \vec{v}_j = \begin{cases} \lambda_j & \lambda_i = \lambda_j \\ 0 & \lambda_i \neq \lambda_j \end{cases}$$

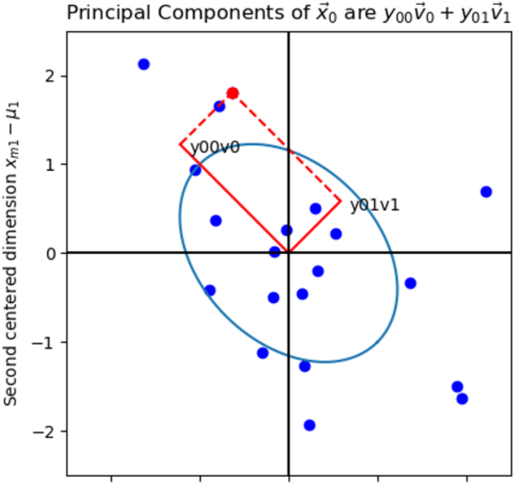
Remember that the rows of X are $(\vec{x}_m - \vec{\mu})^T$, so if we define

$$(\vec{x}_m - \vec{\mu})^T V = \vec{y}_m^T, \qquad Y = \begin{bmatrix} \vec{y}_0^T \\ \vdots \\ \vec{y}_{M-1}^T \end{bmatrix}$$

Then we have that $Y^TY = \Lambda$. In other words, the covariance matrix of the $\vec{y}$ vectors is a diagonal!

Principal components

$$\vec{y}_m = V^T(\vec{x}_m - \vec{\mu})$$



Principal components

Let's write $Y = XV$, and $Y^T = V^T X^T$. In other words,

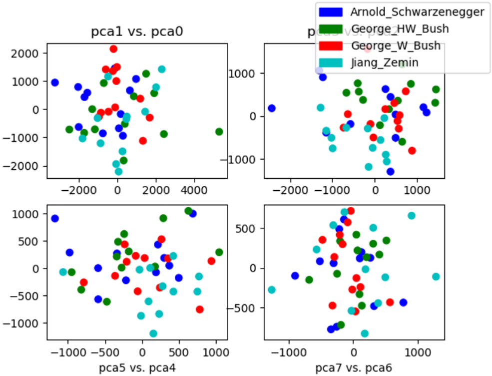
$$(\vec{x}_m - \vec{\mu})^T V = \vec{y}_m^T$$
$$XV = Y$$
$$V^T X^T X V = Y^T Y = \Lambda$$

$\vec{y}_m = [y_{m,0}, \ldots, y_{m,D-1}]^T$ is the vector of principal components of $\vec{x}_m$. Expanding the formula $Y^T Y = \Lambda$, we discover that PCA orthogonalizes the dataset:

$$\sum_{m=0}^{M-1} y_{im} y_{jm} = \begin{cases} \lambda_i & i = j \\ 0 & i \neq j \end{cases}$$

Principal components are uncorrelated, and PC with larger eigenvalues have more energy

In the following figure, notice that (1) the principal components are uncorrelated with one another, (2) the eigenvalues have been sorted so that $\lambda_0 > \lambda_1 > \lambda_2$ and so on. With this sorting, you see that the the first PCA, $y_{m,0}$, has the biggest variance:



Eigenvalue=Energy of the Principal Component

The total dataset energy, along the i^{th} principal component direction, is

$$\sum_{m=0}^{M-1} y_{mi}^2 = \lambda_i$$

But remember that $V^T V = I$. Therefore, the total dataset energy is the same, whether you calculate it in the original image domain, or in the PCA domain:

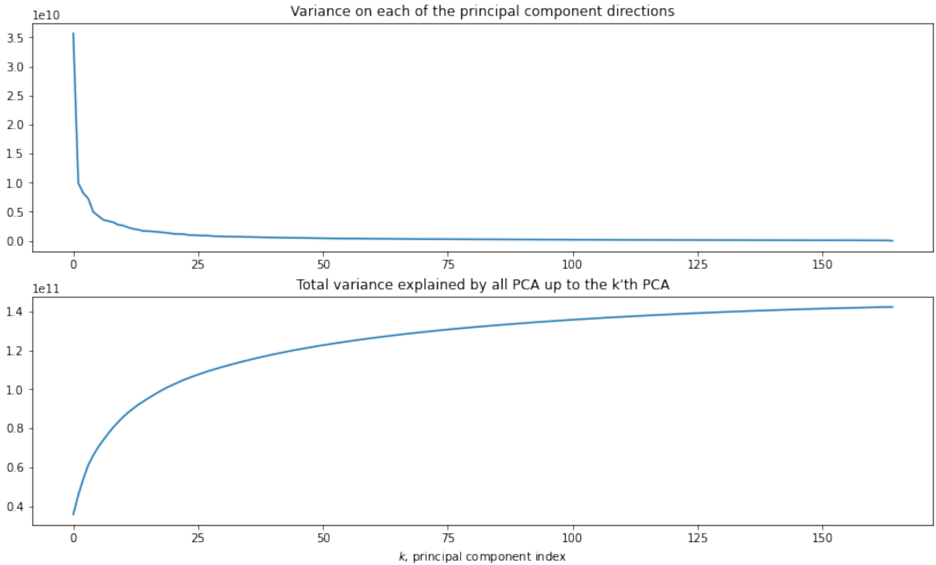
$$\sum_{m=0}^{M-1}\sum_{d=0}^{D-1}(x_{md}-\mu_d)^2 = \sum_{m=0}^{M-1}\sum_{i=0}^{D-1} y_{mi}^2 = \sum_{i=0}^{D-1}\lambda_i$$

Energy spectrum=Fraction of energy explained

The "energy spectrum" is energy as a function of basis vector index. There are a few ways we could define it, but one useful definition is:

$$E[k] = \frac{\sum_{m=0}^{M-1} \sum_{i=0}^{k-1} y_{mi}^2}{\sum_{m=0}^{M-1} \sum_{i=0}^{D-1} y_{mi}^2} = \frac{\sum_{i=0}^{k-1} \lambda_i}{\sum_{i=0}^{D-1} \lambda_i}$$

Energy spectrum=Fraction of energy explained



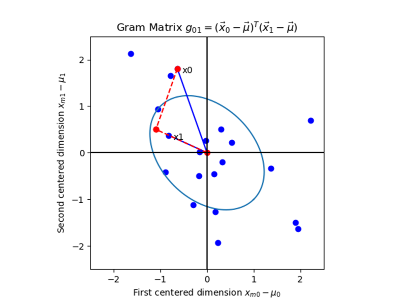
Outline

Gram matrix

- X^TX is usually called the sum-of-squares matrix. $\frac{1}{M-1}X^TX$ is the sample covariance.
- $G = XX^T$ is called the gram matrix. Its $(i,j)^{\text{th}}$ element is the dot product between the i^{th} and j^{th} data samples:

$$g_{ij} = (\vec{x}_i - \vec{\mu})^T(\vec{x}_j - \vec{\mu})$$

Gram matrix $g_{01} = (\vec{x}_0 - \vec{\mu})^T(\vec{x}_1 - \vec{\mu})$



Eigenvectors of the Gram matrix

XX^T is also symmetric! So it has orthonormal eigenvectors:

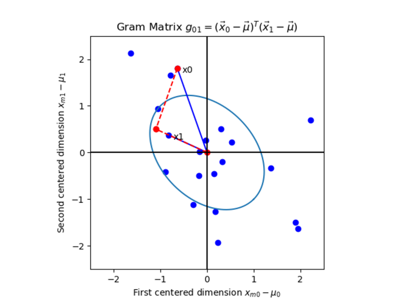
$$XX^T = U\Lambda U^T$$

$$UU^T = U^T U = I$$

Surprising Fact: $X^T X$ and XX^T have the same eigenvalues (Λ), but different eigenvectors (V vs. U).

Gram matrix

$g_{01} = (\vec{x}_0 - \vec{\mu})^T (\vec{x}_1 - \vec{\mu})$



Why the Gram matrix is useful:

Suppose that $D \sim 240000$ pixels per image, but $M \sim 240$ different images. Then, in order to perform this eigenvalue analysis:

$$X^T X = V \Lambda V^T$$

...requires factoring a 240000^{th}-order polynomial ($|X^T X - \lambda I| = 0$), then solving 240000 simultaneous linear equations in 240000 unknowns to find each eigenvector ($X^T X \vec{v}_d = \lambda_d \vec{v}_d$). If you try doing that using `np.linalg.eig`, your PC will be running all day. On the other hand,

$$X X^T = U \Lambda U^T$$

requires only 240 equations in 240 unknowns. Educated experts agree: $240^2 \ll 240000^2$.

Singular Values

- Both $X^T X$ and XX^T are positive semi-definite, meaning that their eigenvalues are non-negative, $\lambda_d \geq 0$.
- The **singular values** of X are defined to be the square roots of the eigenvalues of $X^T X$ and XX^T:

$$S = \begin{bmatrix} s_0 & 0 & 0 \\ 0 & \ldots & 0 \\ 0 & 0 & s_{D-1} \end{bmatrix}, \quad \Lambda = S^2 = \begin{bmatrix} s_0^2 & 0 & 0 \\ 0 & \ldots & 0 \\ 0 & 0 & s_{D-1}^2 \end{bmatrix}$$

Singular Value Decomposition

Let's use the equation $\Lambda = SS$ in the PCA decomposition formula:

$$X^T X = V \Lambda V^T$$
$$= VSSV^T$$
$$= VSISV^T$$

...where the last equation just inserted an identity matrix. But remember, since U is orthonormal, we can write $I = U^T U$, or

$$X^T X = VSU^T USV^T$$
$$= (USV^T)^T (USV^T)$$

Singular Value Decomposition

Let's try the same thing, but starting with the Gram matrix instead: formula:

$$
\begin{aligned}
XX^T &= U\Lambda U^T \\
&= USSU^T \\
&= USISU^T \\
&= USV^TVSU^T
\end{aligned}
$$

$$
= (USV^T)(USV^T)^T
$$

Singular Value Decomposition

ANY $M \times D$ **MATRIX**, X, can be written as $X = USV^T$.

- $U = [\vec{u}_0, \ldots, \vec{u}_{M-1}]$ are the eigenvectors of XX^T.
- $V = [\vec{v}_0, \ldots, \vec{v}_{D-1}]$ are the eigenvectors of X^TX.
- $S = \begin{bmatrix} s_0 & 0 & 0 & 0 & 0 \\ 0 & \ldots & 0 & 0 & 0 \\ 0 & 0 & s_{\min(D,M)-1} & 0 & 0 \end{bmatrix}$ are the singular values.

S has some all-zero columns if $M > D$, or all-zero rows if $M < D$.

What np.linalg.svd does

First, `np.linalg.svd` decides whether it wants to find the eigenvectors of X^TX or XX^T: it just checks to see whether $M > D$ or vice versa. If it discovers that $M < D$, then:

1. Compute $XX^T = U\Lambda U^T$, and $S = \sqrt{\Lambda}$. Now we have U and S, we just need to find V.
2. Since $X^T = VSU^T$, we can get V by just multiplying:

$$\tilde{V} = X^T U$$

...where $\tilde{V} = VS$ is exactly equal to V, but with each column scaled by a different singular value. So we just need to normalize:

$$\|\vec{v}_i\| = 1, \quad v_{i0} > 0$$

Methods that solve PCA when # faces // # features

- The **covariance matrix** method: (eigenvector analysis of $X^T X$) gives the right answer, but takes **a very long time**.
- The **gram matrix** method is **much faster**: Apply `np.linalg.eig` to get U from XX^T. Multiply $\tilde{V} = X^T U$, Tricky point: normalize so that $\|\vec{v}_k\| = 1$, $v_{k,1} \geq 0$.
- The **SVD** method: Applying `np.linalg.svd(X)`. Speed = `min(speed(covariance),speech(gram))`. Tricky point: $\lambda_m = s_m^2$.

Whatever you do, be sure to sort the eigenvalues so $|\lambda_k| \geq |\lambda_{k+1}|$.

Outline

Summary

- Symmetric matrices:

$$A = V\Lambda V^T, \quad V^T A V = \Lambda, \quad V^T V = VV^T = I$$

- Centered dataset:

$$X = \begin{bmatrix} (\vec{x}_0 - \vec{\mu})^T \\ \vdots \\ (\vec{x}_{M-1} - \vec{\mu})^T \end{bmatrix}$$

- Singular value decomposition:

$$X = USV^T$$

where V are eigenvectors of the sum-of-squares matrix, U are eigenvectors of the gram matrix, and $\Lambda = S^2$ are their shared eigenvalues.