ENGR-E 511; ENGR-E 399

“Machine Learning for Signal Processing”

Module 04:

Dimension Reduction

Minje Kim

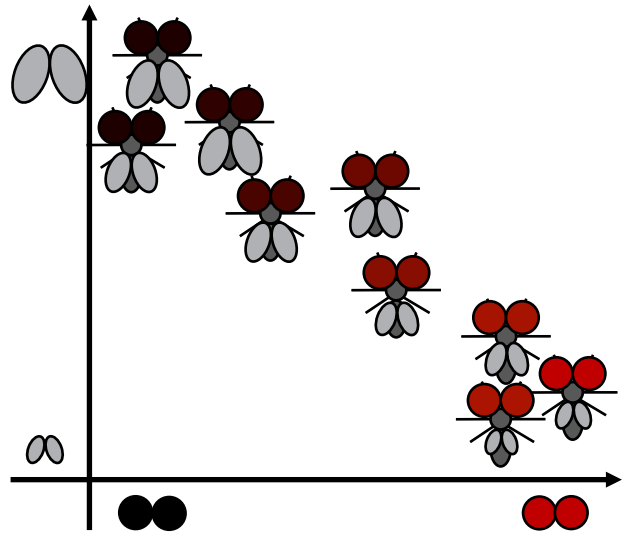
Department of Intelligent Systems Engineering

Email: minje@indiana.edu

Website: http://minjekim.com

Research Group: http://saige.sice.indiana.edu

Meeting Request: http://doodle.com/minje



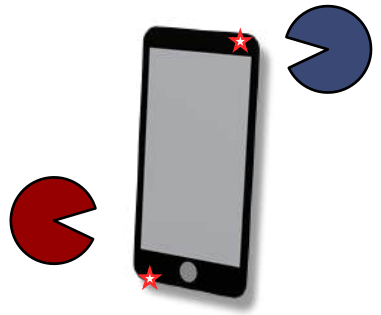
INDIANA UNIVERSITY

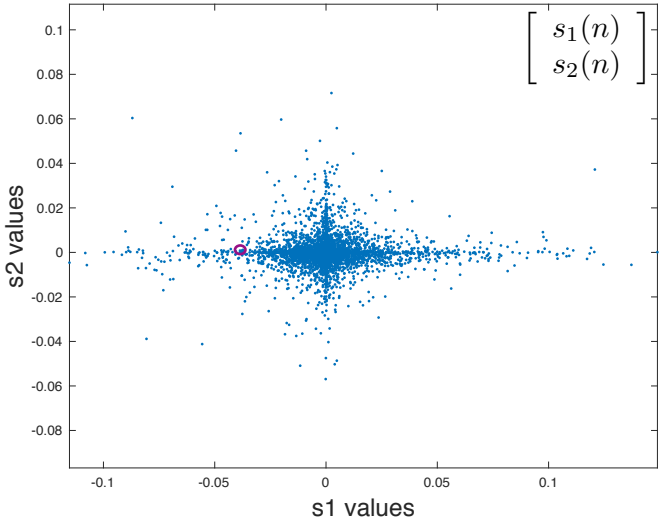
SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

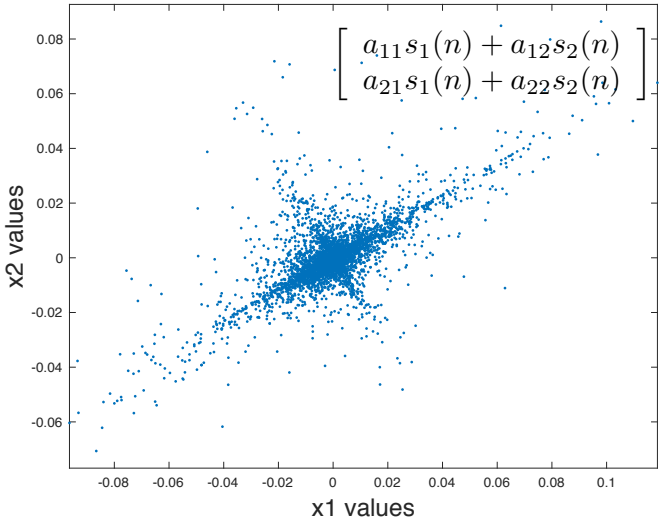
A Toy Source Separation Problem

- Two sensors and two sources

- Two people are saying at the same time and two mics are recording it
- I'm ignoring a lot of important things
  - e.g. delay, reverberation, geometry, etc.
- The sources
  - s1:
  - s2:
  - We can scatter plot these samples on a 2D space (a 2XN matrix)
    - The projection onto each axis recovers sources
- Each recording is a linear mixture of the sources
  - x1:
  - x2:
  - Now, the projection onto each axis doesn't recover the sources
  - We need to project the data points more carefully
  - What would that correct projections be?



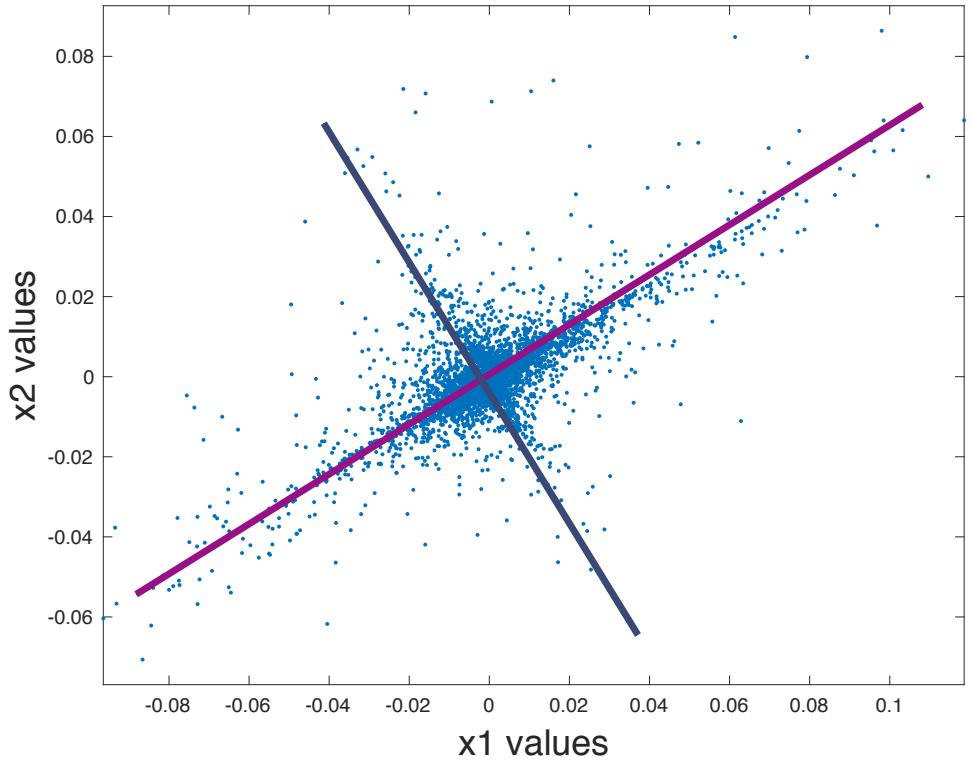




A Toy Source Separation Problem

- Two sensors and two sources

- For this overly simplified example, we know the answer
  - Because the linear combination for mixing is nothing but some rotation



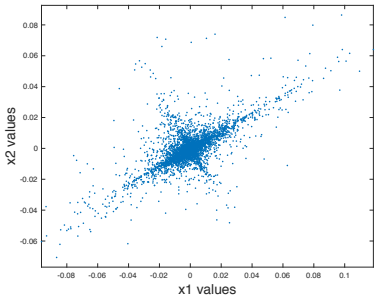
  - How do we find these vectors?

Principal Component Analysis

- a.k.a. Karhunen-Loève Transform

- First, let's define the variables $\boldsymbol{x}_n = \begin{bmatrix} x_1(n) \\ x_2(n) \end{bmatrix} = \begin{bmatrix} A_{11}s_1(n) + A_{12}s_2(n) \\ A_{21}s_1(n) + A_{22}s_2(n) \end{bmatrix} = \boldsymbol{A}\boldsymbol{s}_n$

$$\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N]$$



- For this data I've got a feeling that the best projection would be
  - The one that makes the sample variance maximal
- If we assume zero mean variables, the variance after a projection is

$$var(\boldsymbol{w}^\top \boldsymbol{x}_n) = \frac{1}{N}\sum_n (\boldsymbol{w}^\top \boldsymbol{x}_n)^2 = \frac{1}{N}\sum_n \left([w_1, w_2] \begin{bmatrix} x_1(n) \\ x_2(n) \end{bmatrix} \right)^2$$

  - Or in the matrix form $= \frac{1}{N}(\boldsymbol{w}^\top \boldsymbol{X})(\boldsymbol{w}^\top \boldsymbol{X})^\top = \frac{1}{N}\boldsymbol{w}^\top \boldsymbol{X}\boldsymbol{X}^\top \boldsymbol{w}$

- The objective function is $\arg\max_{\boldsymbol{w}} \frac{1}{N}\boldsymbol{w}^\top \boldsymbol{X}\boldsymbol{X}^\top \boldsymbol{w}$
  - What am I missing? $\arg\max_{\boldsymbol{w}} \boldsymbol{w}^\top \boldsymbol{\Sigma}_x \boldsymbol{w} + \lambda(1 - \boldsymbol{w}^\top \boldsymbol{w})$
  - What happens if it were not for the constraints?

Principal Component Analysis

- a.k.a. Karhunen-Loève Transform

- Partial differentiation $\arg\max_{\boldsymbol{w}} \boldsymbol{w}^\top \boldsymbol{\Sigma}_x \boldsymbol{w} + \lambda\left(1 - \boldsymbol{w}^\top \boldsymbol{w}\right)$

$$\frac{\partial \mathcal{J}}{\partial \boldsymbol{w}} = 2\boldsymbol{\Sigma}_x \boldsymbol{w} - 2\lambda \boldsymbol{w} = 0 \qquad \frac{\partial \mathcal{J}}{\partial \lambda} = 1 - \boldsymbol{w}^\top \boldsymbol{w} = 0 \qquad \boldsymbol{\Sigma}_x \boldsymbol{w} = \lambda \boldsymbol{w}$$

  - It's an eigendecomposition problem
  - The variance after projection equals the eigenvalue

$$\boldsymbol{w}^\top \boldsymbol{\Sigma}_x \boldsymbol{w} = \lambda \Leftrightarrow \boldsymbol{w}^\top \boldsymbol{X}\boldsymbol{X}^\top \boldsymbol{w} = \lambda$$

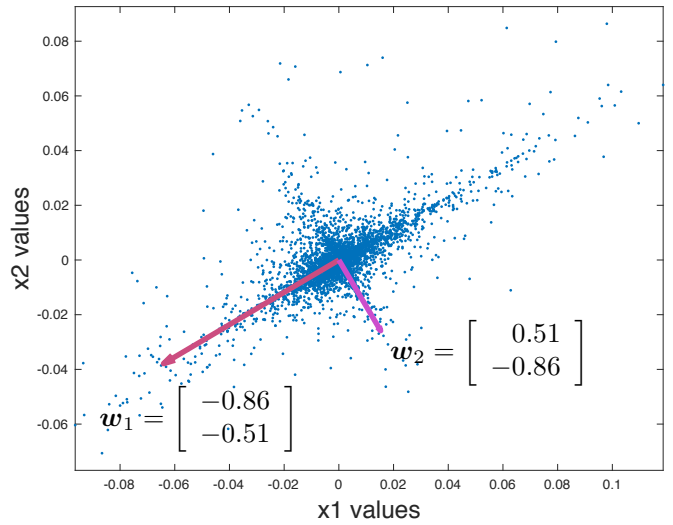
$$\boldsymbol{W}^\top \boldsymbol{\Sigma}_x \boldsymbol{W} = \Lambda$$

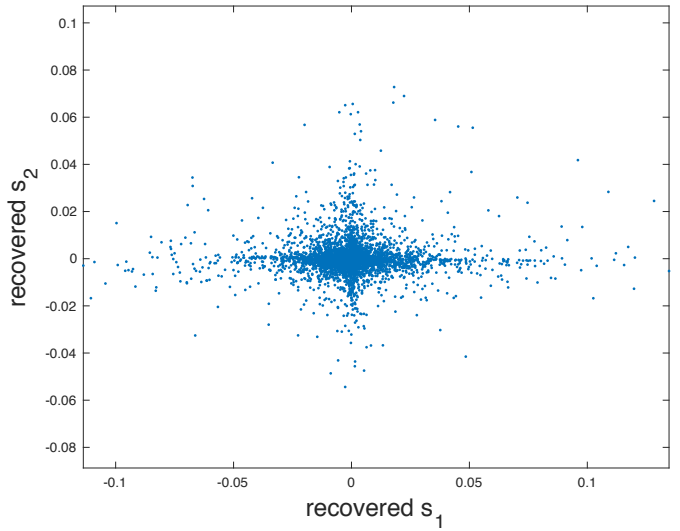
- Principal Components (PC)
  - They are projection vectors
  - Can be found using eigendecomposition on the covariance matrix (eigenvectors)
  - They are ordered based on the eigenvalues
    - $\boldsymbol{w}_1$ is the 1st PC if $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_D$

Principal Component Analysis

- a.k.a. Karhunen-Loève Transform

- The estimated PCs
  - There's a sign ambiguity
- Variances after projection
  - $\lambda_1 = 3.48 \times 10^{-4},\ \lambda_2 = 5.65 \times 10^{-5}$
- How about source separation?
  - We're trying to eliminate the mixing effect by rotating the data points back to the original space
  - Therefore: $\boldsymbol{W}^\top \boldsymbol{A} \approx \boldsymbol{I} \qquad \because \boldsymbol{x}_n = \boldsymbol{A}\boldsymbol{s}_n$
  - $$\begin{bmatrix} -0.86 & -0.51 \\ 0.51 & -0.86 \end{bmatrix} \begin{bmatrix} 0.87 & -0.50 \\ 0.50 & 0.86 \end{bmatrix} = \begin{bmatrix} -1 & -0.01 \\ 0.01 & -1 \end{bmatrix}$$
    - Almost correct except the flipped signs
- Separation (projection) results $\begin{bmatrix} \boldsymbol{s}_1 \\ \boldsymbol{s}_2 \end{bmatrix} \approx \boldsymbol{W}^\top \boldsymbol{X}$

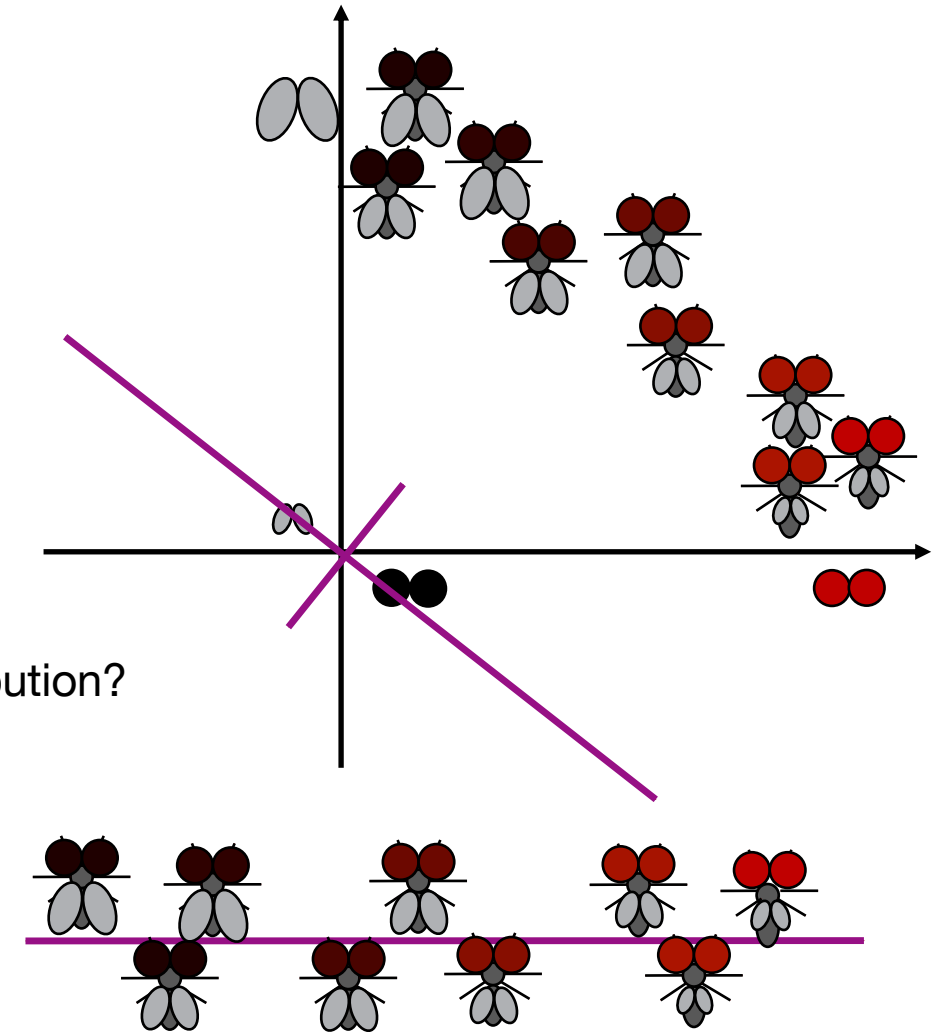




PCA for Dimension Reduction

- The Mating Flies and Lazy RA Example

- One day as he takes off Prof. K ordered his RA to group the flies into two clusters
  - The RA forgot about it and went home..
- Originally it was a clustering problem
  - GMM could have described the sample distribution
  - But, not anymore, b/c over the night the flies mated
- Is there another way to describe this sample distribution?
  - Rather than clustering?
  - The 1st PC
- We lost some information by reducing the dimension?



Clustering vs PCA

- In terms of compression

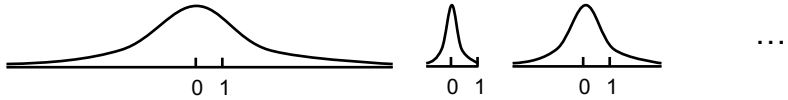
- Clustering (GMM, k-means, vector quantization, etc)
  - It reduces the levels of representation by allowing “collision”
  - Multiple data samples are allowed (or forced) to have the same value
  - Number of clusters will define the amount of compression

- PCA (or the other dimension reduction methods)
  - It transforms the data into a new coordinate system
  - Which can be defined with a less number of axis
  - Each data sample has its own unique combination of coefficients in the new coordinate system (no collision)
  - Still need to encode the coefficients

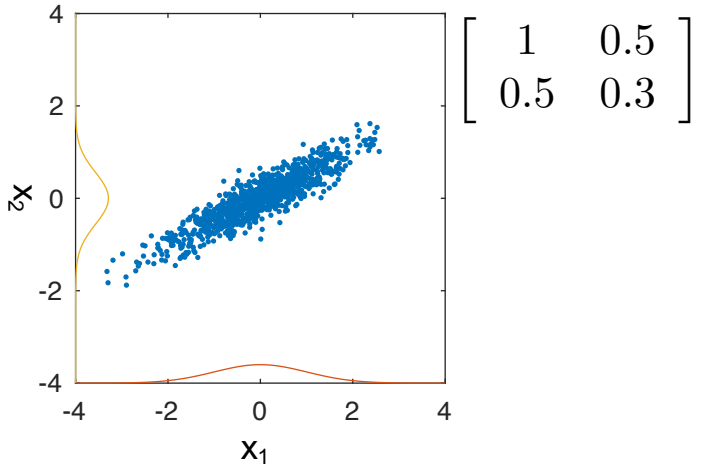
PCA for Whitening

- Feature extraction and normalization in one shot

- Say that there are 10 original dimensions with unique dimension-wise distributions

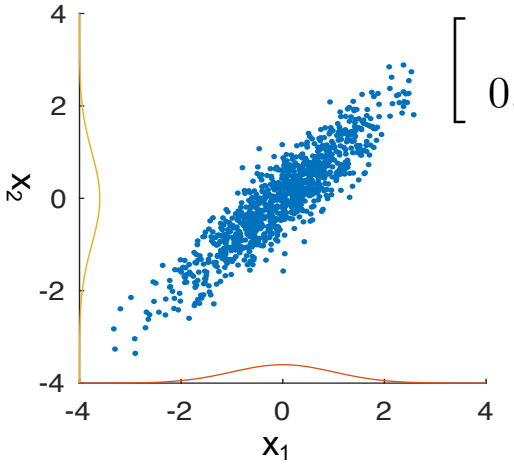


- Our job is
  - To figure out a smaller set of features
  - To lessen the difference in dynamic ranges
- As for the DR, maybe just a standardization could be enough
  - Zero mean and unit variance
- But, some of them are correlated



$$\begin{bmatrix} 1 & 0.5 \\ 0.5 & 0.3 \end{bmatrix}$$

After coordinate-wise standardization



$$\begin{bmatrix} 1 & 0.91 \\ 0.91 & 1 \end{bmatrix}$$

PCA for Whitening

- Feature extraction and normalization in one shot

- Whitening is not only for ensuring unit variances (1's in the diagonal)
  - But for making off-diagonals zero
- We can use PCA for whitening
  - Eigendecomposition of the covariance yields a diagonal matrix: $\boldsymbol{W}^\top \boldsymbol{\Sigma}_x \boldsymbol{W} = \Lambda$ ← So far, just diagonalization, not whitening
  - By the way,

$$\Lambda = \Lambda^{\frac{1}{2}} \Lambda^{\frac{1}{2}} = \begin{bmatrix} \lambda_1^{\frac{1}{2}} & 0 & \cdots & 0 \\ 0 & \lambda_2^{\frac{1}{2}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_D^{\frac{1}{2}} \end{bmatrix} \begin{bmatrix} \lambda_1^{\frac{1}{2}} & 0 & \cdots & 0 \\ 0 & \lambda_2^{\frac{1}{2}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_D^{\frac{1}{2}} \end{bmatrix}$$

  - Therefore,

$$\Lambda^{(-\frac{1}{2})} \boldsymbol{W}^\top \boldsymbol{\Sigma}_x \boldsymbol{W} = \Lambda^{\frac{1}{2}} \qquad \Lambda^{(-\frac{1}{2})} \boldsymbol{W}^\top \boldsymbol{\Sigma}_x \boldsymbol{W} \Lambda^{(-\frac{1}{2})} = \boldsymbol{I}$$

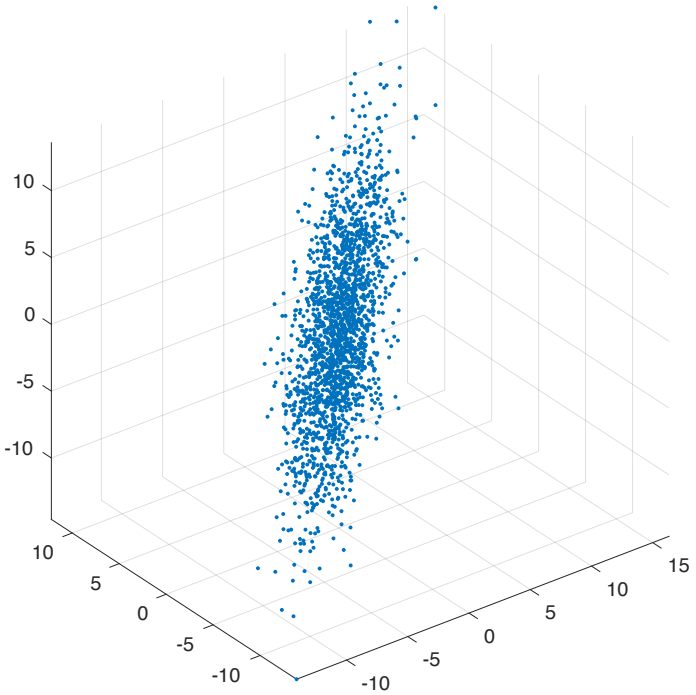
- The projection for whitening:

$$\Lambda^{(-\frac{1}{2})} \boldsymbol{W}^\top \boldsymbol{X} = \begin{bmatrix} \boldsymbol{w}_1^\top / \sqrt{\lambda_1} \\ \boldsymbol{w}_2^\top / \sqrt{\lambda_2} \\ \vdots \\ \boldsymbol{w}_D^\top / \sqrt{\lambda_D} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_D \end{bmatrix}$$

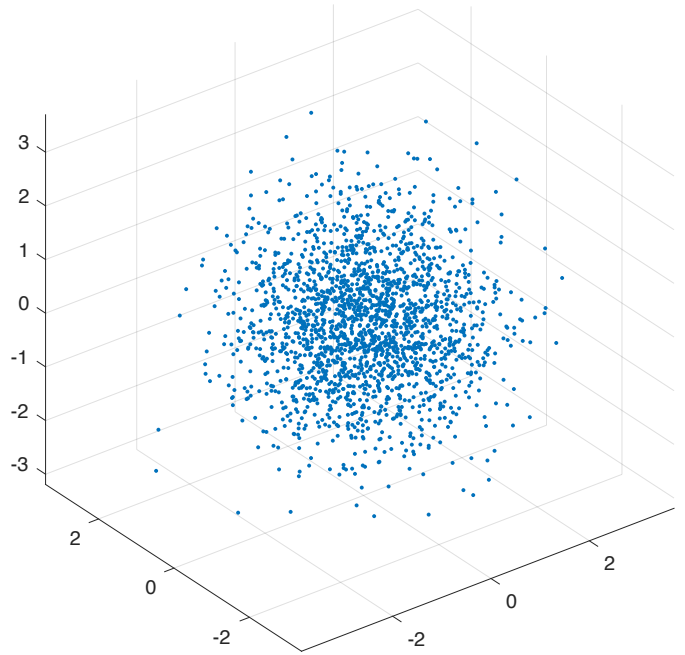
PCA for Whitening

- Feature extraction and normalization in one shot

- A whitening example (MATLAB fig)



Before whitening



After whitening

Electroencephalograpy (EEG) Signals

- A Higher-Dimensional Data

- Our first audio example in this lecture wasn't really a dimension reduction case
  - It's more likely a rotating problem for unmixing
- An EEG data with 14 channels and a label for the eye status {open, close}

```
@RELATION EEG_DATA
@ATTRIBUTE AF3 NUMERIC
@ATTRIBUTE F7 NUMERIC
@ATTRIBUTE F3 NUMERIC
@ATTRIBUTE FC5 NUMERIC
@ATTRIBUTE T7 NUMERIC
@ATTRIBUTE P7 NUMERIC
@ATTRIBUTE O1 NUMERIC
@ATTRIBUTE O2 NUMERIC
@ATTRIBUTE P8 NUMERIC
@ATTRIBUTE T8 NUMERIC
@ATTRIBUTE FC6 NUMERIC
@ATTRIBUTE F4 NUMERIC
@ATTRIBUTE F8 NUMERIC
@ATTRIBUTE AF4 NUMERIC
@ATTRIBUTE eyeDetection {0,1}
@DATA
4329.23,4009.23,4289.23,4148.21,4350.26,4586.15,4096.92,4641.03,4222.05,4238.46,4211.28,4280.51,4635.90,4393.85,0
4324.62,4004.62,4293.85,4148.72,4342.05,4586.67,4097.44,4638.97,4210.77,4226.67,4207.69,4279.49,4632.82,4384.10,0
4327.69,4006.67,4295.38,4156.41,4336.92,4583.59,4096.92,4630.26,4207.69,4222.05,4206.67,4282.05,4628.72,4389.23,0
4328.72,4011.79,4296.41,4155.90,4343.59,4582.56,4097.44,4630.77,4217.44,4235.38,4210.77,4287.69,4632.31,4396.41,0
4362.56,3983.08,4273.33,4115.38,4336.41,4624.62,4104.62,4627.18,4212.82,4233.85,4245.13,4324.62,4663.08,4421.03,1
4358.46,3978.97,4268.21,4104.62,4330.26,4615.90,4103.08,4623.08,4205.13,4235.90,4244.62,4322.05,4663.08,4411.79,1
4363.08,3981.03,4276.92,4106.67,4333.85,4608.21,4097.44,4614.36,4194.36,4230.26,4247.69,4323.59,4667.18,4418.46,1
4366.67,3981.54,4274.36,4112.82,4334.87,4612.82,4093.85,4608.72,4192.82,4226.67,4247.18,4318.97,4671.28,4426.67,1
...
```

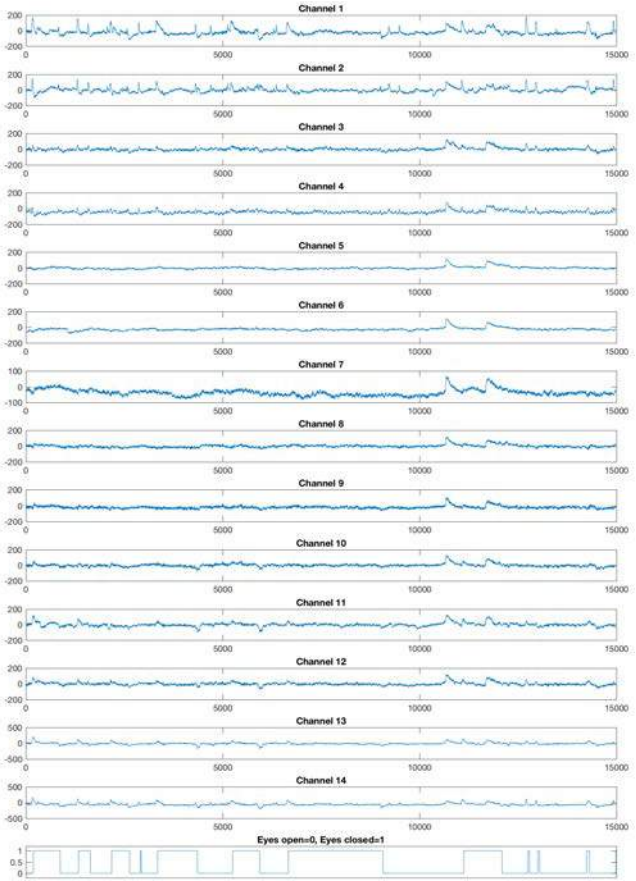
https://en.wikipedia.org/wiki/Electroencephalography

https://en.wikipedia.org/wiki/Emotiv

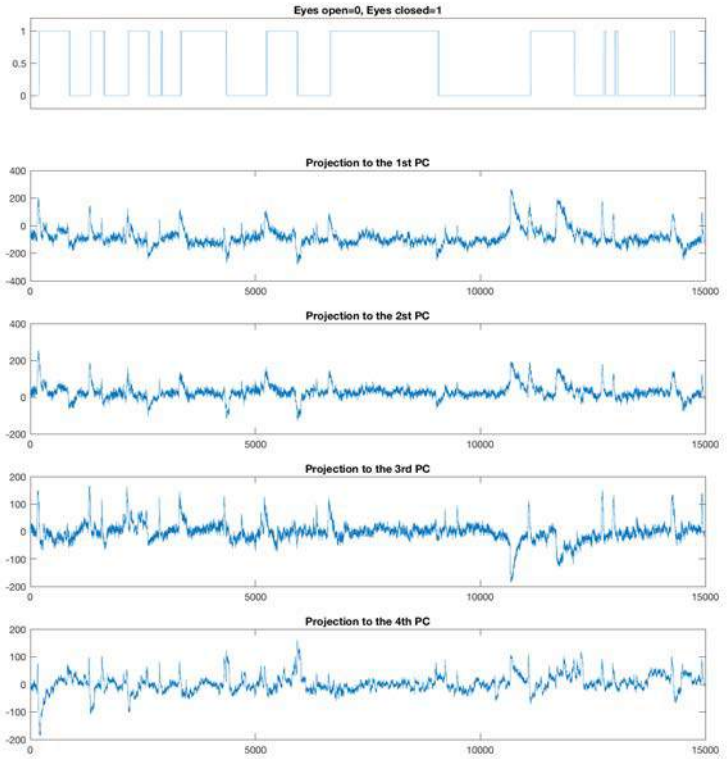
https://archive.ics.uci.edu/ml/datasets/EEG+Eye+State

Electroencephalograpy (EEG) Signals

- A Higher-Dimensional Data



Raw signals (after outlier detection)

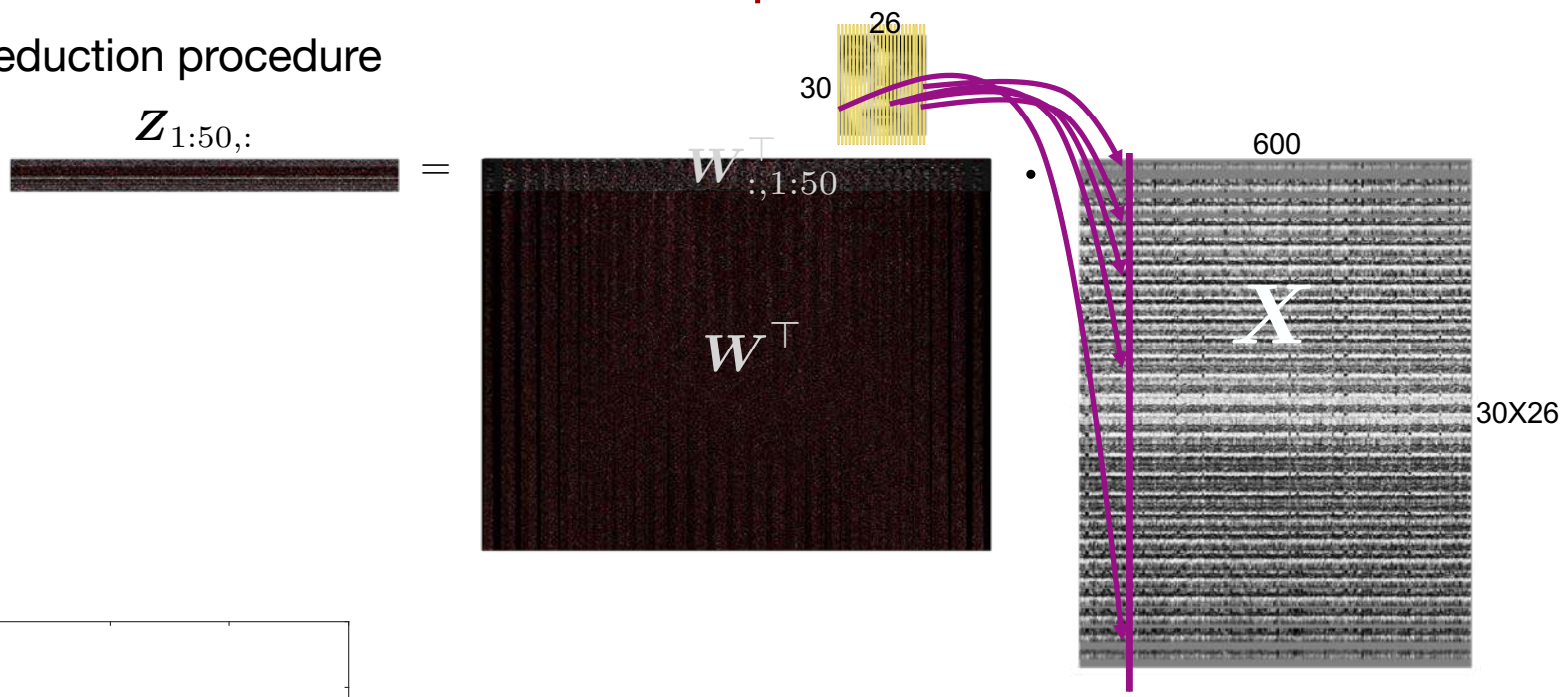


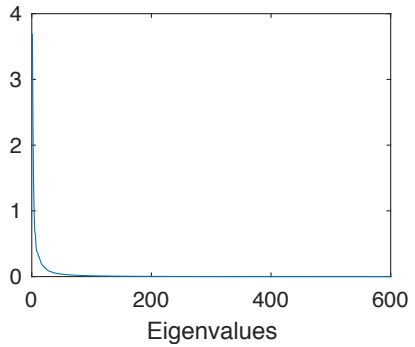
After PCA (4 PCs)

Eigenfaces

- Just another dimension reduction example

- Dimension reduction procedure



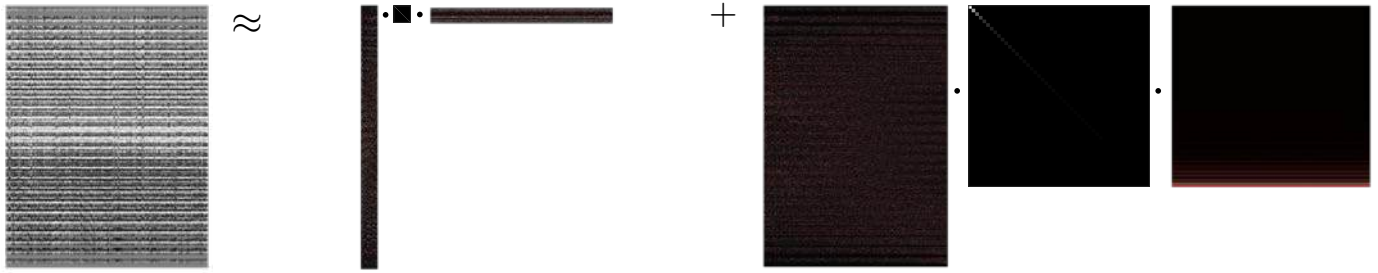


Eigenfaces

- Just another dimension reduction example

- Why does it work?
- Remember SVD
  - whose singular vectors are eigenvectors of the covariance matrix

$$\boldsymbol{X} \approx \boldsymbol{V}\boldsymbol{S}\boldsymbol{U}^\top = \underbrace{\boldsymbol{V}_{:,1:50}\boldsymbol{S}_{1:50,1:50}\boldsymbol{U}^\top_{:,1:50}}_{\text{More significant part}} + \underbrace{\boldsymbol{V}_{:,51:D}\boldsymbol{S}_{51:D,51:D}\boldsymbol{U}^\top_{:,51:D}}_{\text{Less significant part}}$$



  - So, the dimension reduction projections are

$$\boldsymbol{W}^\top_{:,1:50}\boldsymbol{X} = \boldsymbol{V}^\top_{:,1:50}\boldsymbol{X} \approx \boldsymbol{V}^\top_{:,1:50}\boldsymbol{V}_{:,1:50}\boldsymbol{S}_{1:50,1:50}\boldsymbol{U}^\top_{:,1:50} + 0 \leftarrow \text{Orthogonality}$$

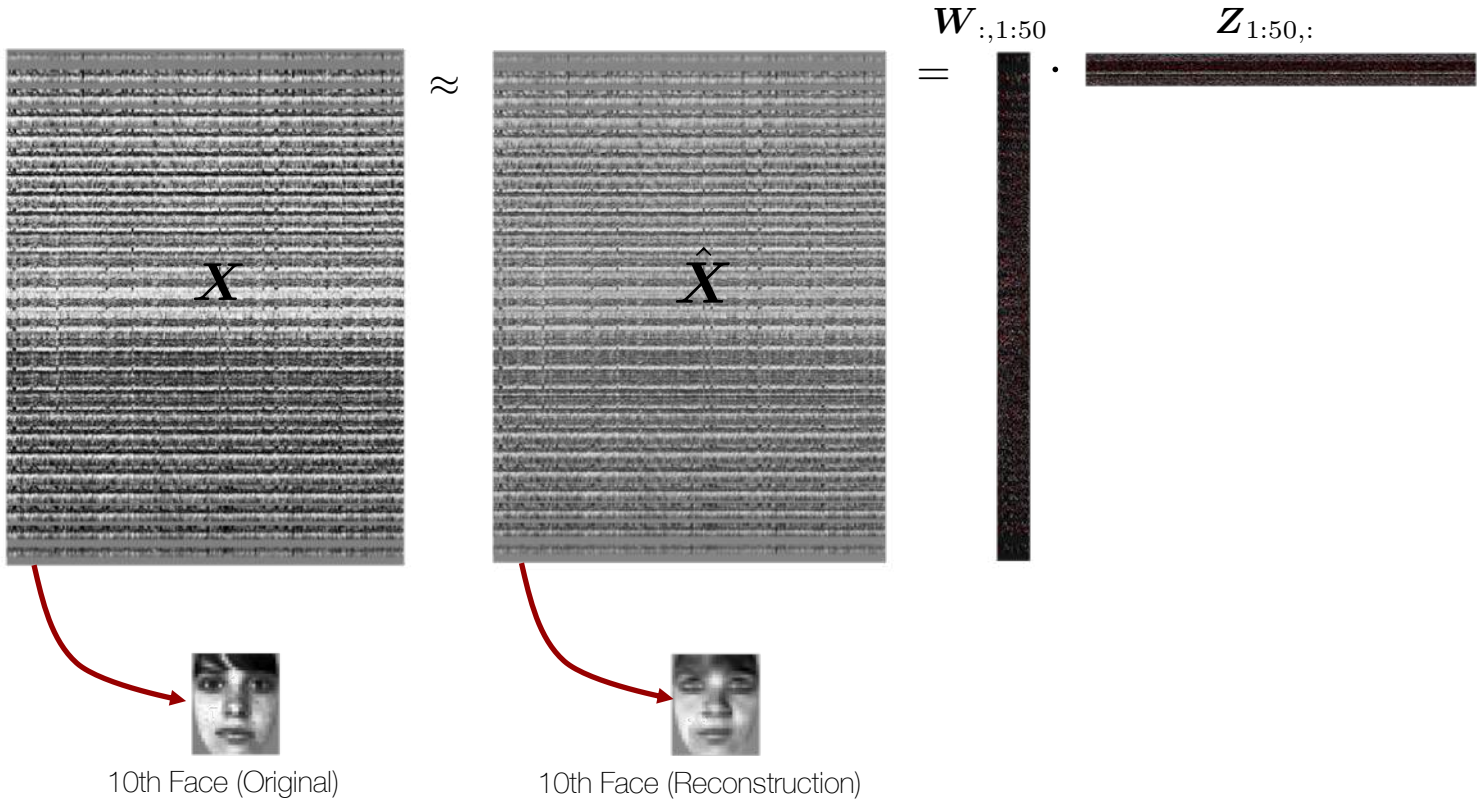
$$\boldsymbol{W}^\top_{:,1:50}\boldsymbol{X} = \boldsymbol{S}_{1:50,1:50}\boldsymbol{U}^\top_{:,1:50} = \boldsymbol{Z}_{1:50,:} \leftarrow \text{Unnormalized right singular vectors}$$

  - Then, the reconstruction procedure is

$$\boldsymbol{W}_{:,1:50}\boldsymbol{Z}_{1:50,:} = \boldsymbol{W}_{:,1:50}\boldsymbol{W}^\top_{:,1:50}\boldsymbol{X} = \boldsymbol{V}_{:,1:50}\boldsymbol{S}_{1:50,1:50}\boldsymbol{U}^\top_{:,1:50} \leftarrow \text{SVD with first 50 components}$$

Eigenfaces

- Reconstruction



- We see that only 50 coefficients can carry a lot of information

Eigenfaces

- The learned eigenfaces (eigenvectors in the matrix form)

Another Toy Source Separation Problem

- A little bit more realistic example

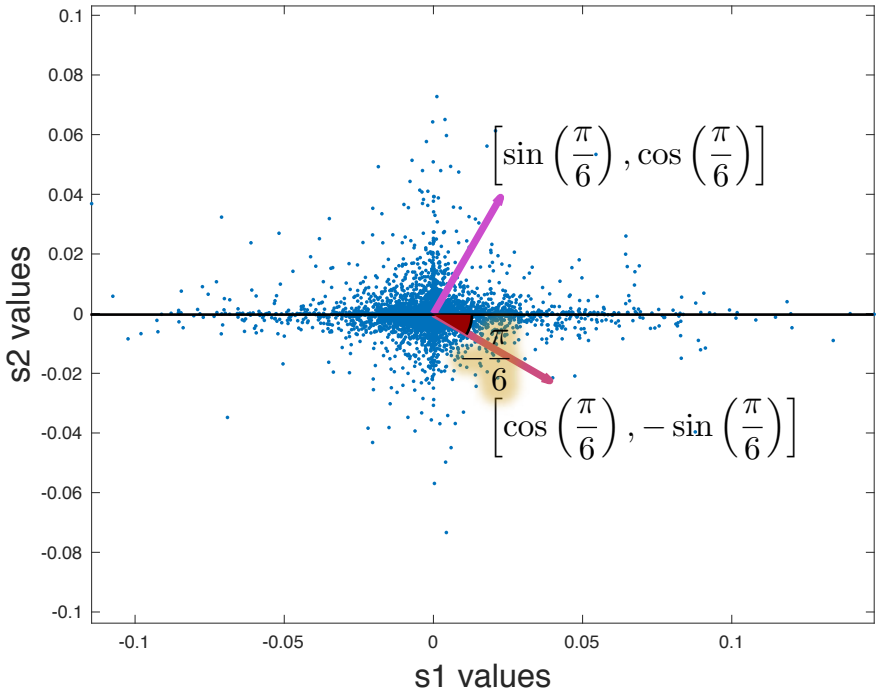
- In the previous separation example, the mixing matrix I used was

$$\boldsymbol{A} = \begin{bmatrix} \cos\left(\frac{\pi}{6}\right) & -\sin\left(\frac{\pi}{6}\right) \\ \sin\left(\frac{\pi}{6}\right) & \cos\left(\frac{\pi}{6}\right) \end{bmatrix}$$

- It's a special matrix, because the two rotations are orthogonal

$$\left[\cos\left(\frac{\pi}{6}\right), -\sin\left(\frac{\pi}{6}\right)\right] \cdot \begin{bmatrix} \sin\left(\frac{\pi}{6}\right) \\ \cos\left(\frac{\pi}{6}\right) \end{bmatrix} = 0$$

  - So, the separation was exceptionally easy
    - Just rotating the data back to normal
  - What if the mixing matrix was more realistic?



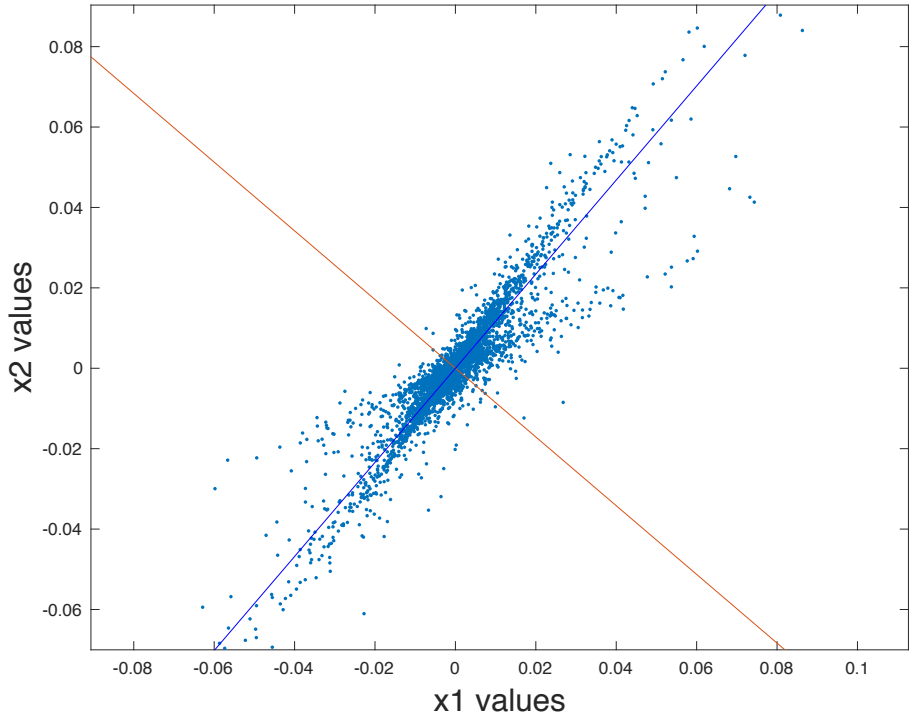
Another Toy Source Separation Problem

- A little bit more realistic example

- A more realistic mixing matrix

$$\boldsymbol{A} = \begin{bmatrix} 0.5 & 0.9 \\ 0.7 & 0.4 \end{bmatrix}, \qquad \begin{bmatrix} x_1(n) \\ x_2(n) \end{bmatrix} = \begin{bmatrix} 0.5 & 0.9 \\ 0.7 & 0.4 \end{bmatrix} \cdot \begin{bmatrix} s_1(n) \\ s_2(n) \end{bmatrix}$$

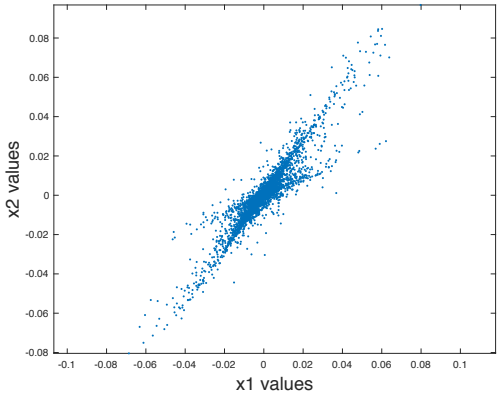
- What does it mean?
  - In the first channel the female speaker is attenuated more (0.5 vs 0.9)
  - In the second channel the male speaker is attenuated more (0.4 vs 0.7)
- PCA?
  - Doesn't seem to work



Another Toy Source Separation Problem

- A little bit more realistic example

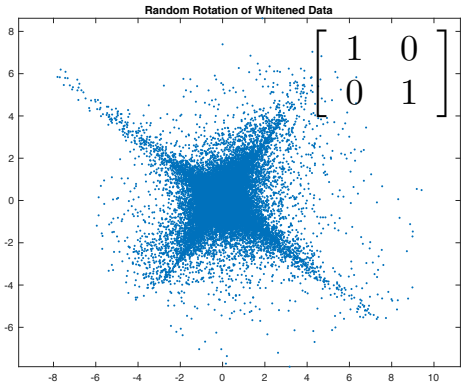
- Can PCA whitening solve this?

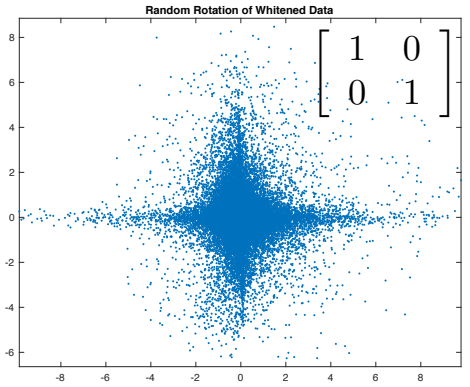


PCA Whitening Results

$$cov(\boldsymbol{Z}) = \boldsymbol{I}$$

  - No, because after whitening all rotations give unit covariance



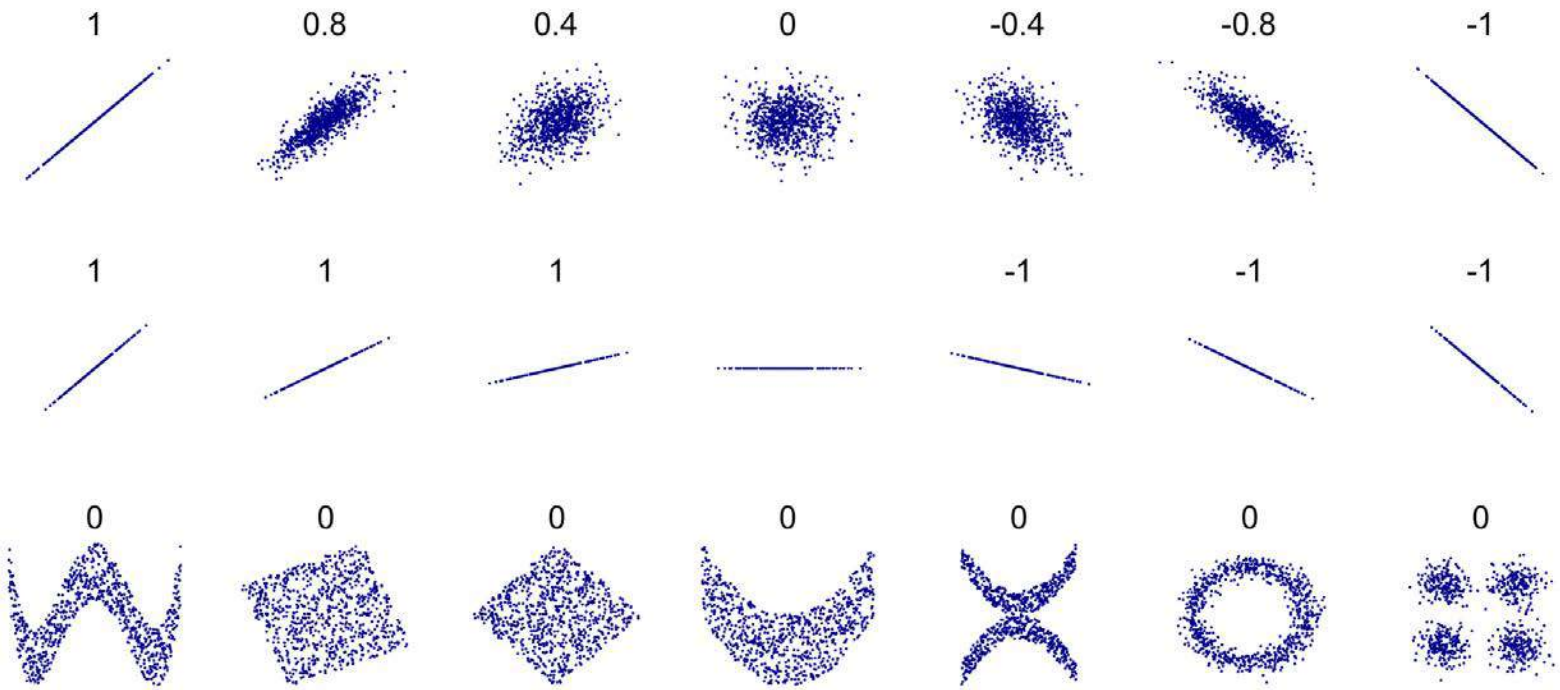


Correlation

- Whitening is not enough

- Correlation values of various joint distributions

$$\frac{cov(x,y)}{\sigma_x \sigma_y}$$

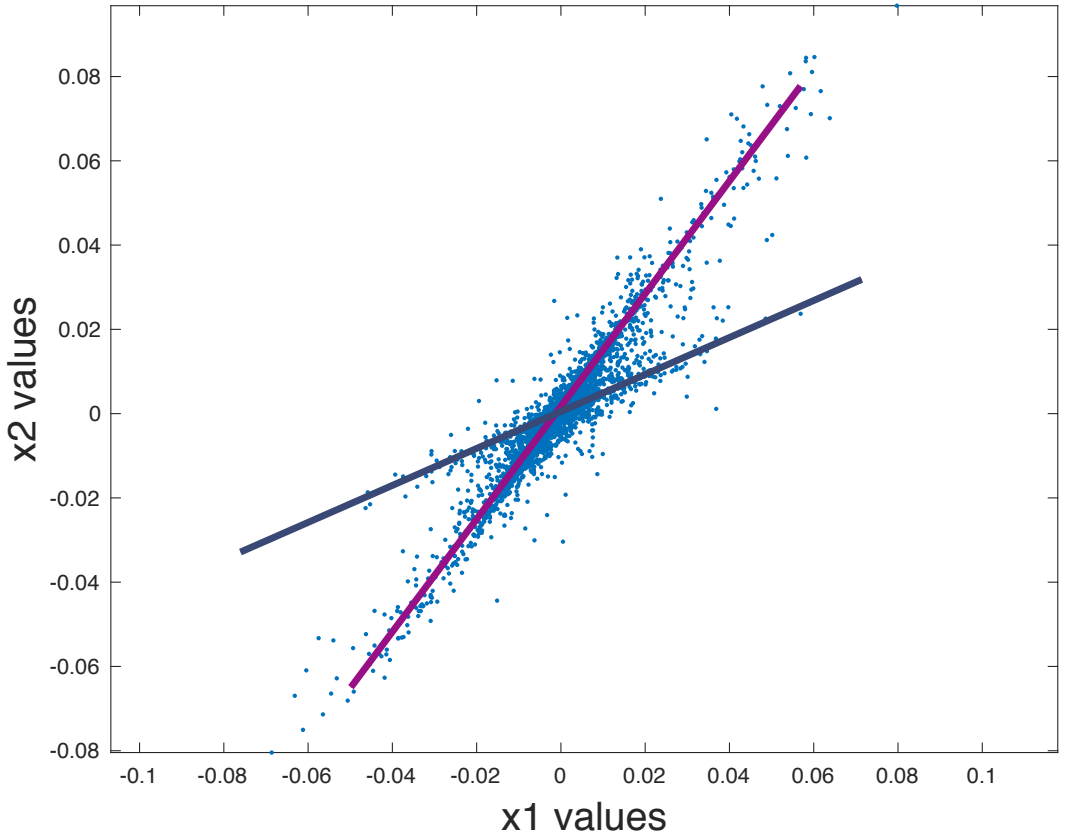


https://en.wikipedia.org/wiki/Correlation_and_dependence

Independent Component Analysis

- An algorithm beyond whitening

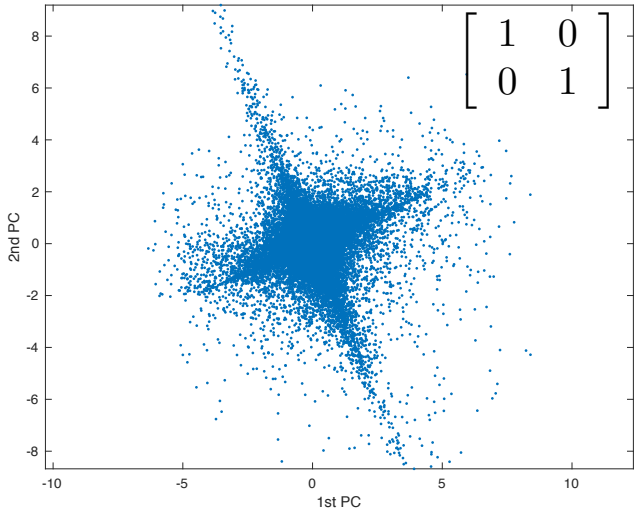
- We need another criteria that can estimate the non-orthogonal basis vectors

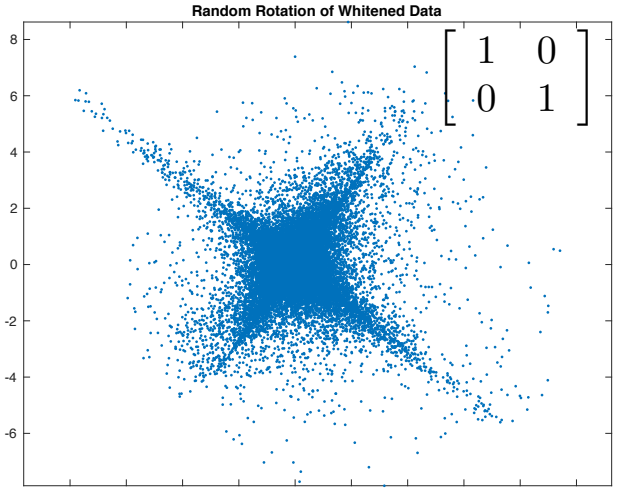


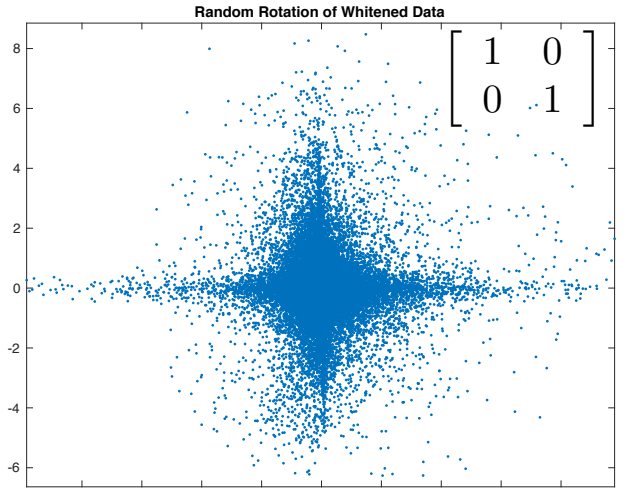
Independent Component Analysis

- An algorithm beyond whitening

- All these whitened data and their rotations are already uncorrelated $E[x, y] = E[x]E[y]$







- How about their independence? $P(x, y) = P(x)P(y)$

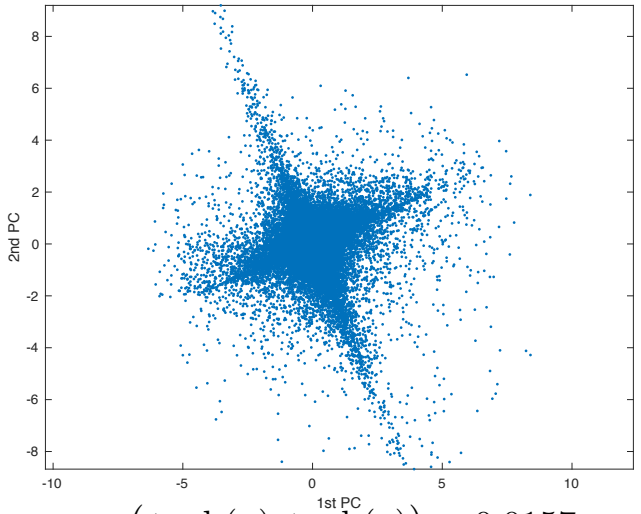
$$E\left[f(x)g(y)\right] = \int_x \int_y f(x)g(y)P(x)P(y)dxdy = \int_x f(x)P(x)dx \int_y g(y)P(y)dy = E\left[f(x)\right]E\left[g(y)\right]$$

  - In order to be independent, for any functions f and g, $f(x)$ and $g(y)$ should be uncorrelated
  - This is a much stronger condition than whitening

Independent Component Analysis

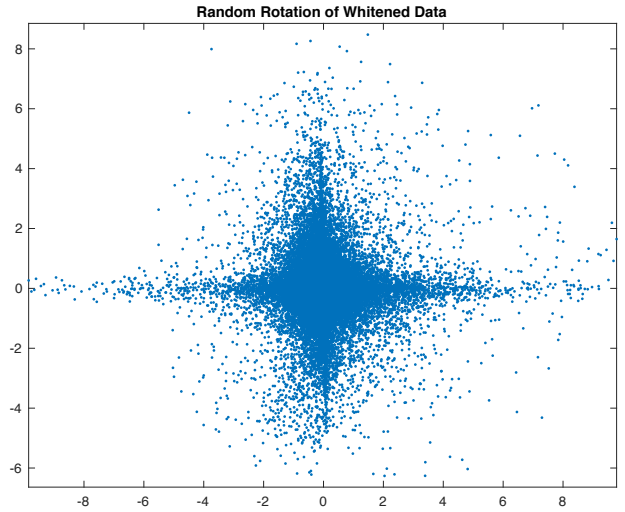
- An algorithm beyond whitening

- Let's check the covariance matrix of after applying $f(x)$ and $g(y)$



$$cov\big(\tanh(x), \tanh(y)\big) = 0.0157$$
$$cov\big(x^3, y^3\big) = -42.63$$

$$0.0154$$
$$-69.68$$



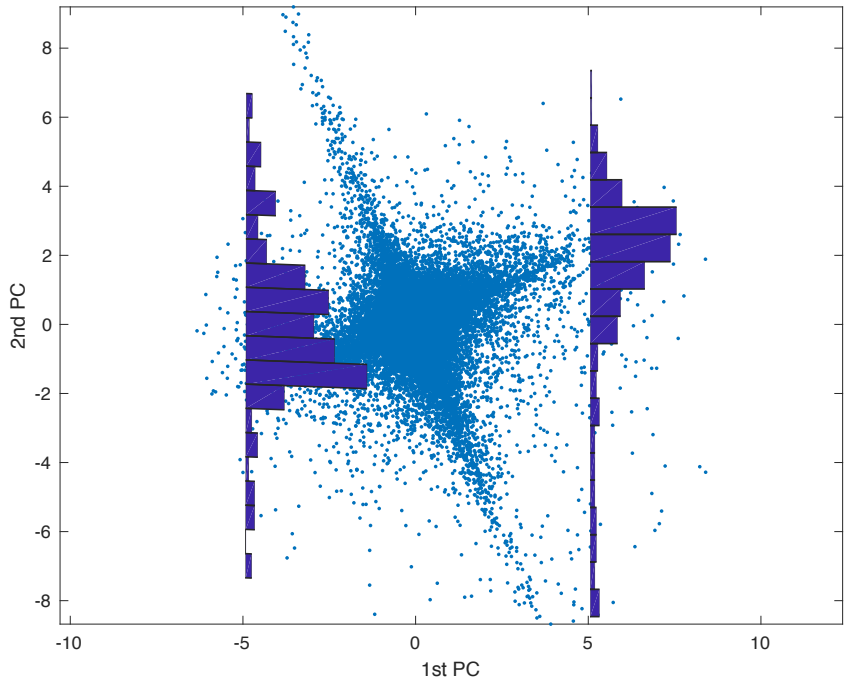
$$-0.0059$$
$$8.55$$

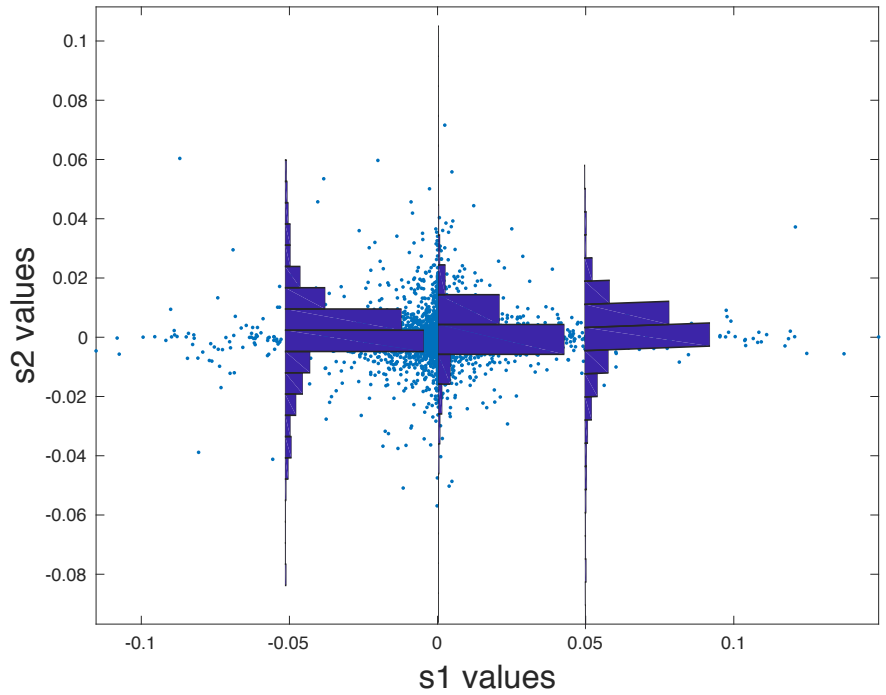
- There's a rotation that can maximize the independence
- A Usual pipeline for ICA
  1. Whiten the data using PCA
     1. Drop some eigenvectors if dimension reduction is needed
  2. Do ICA

Independent Component Analysis

- An algorithm beyond whitening

- Which one is more independent? $P(\boldsymbol{x}_2|\boldsymbol{x}_1) = P(\boldsymbol{x}_2)?$





Independent Component Analysis

- The optimization

- There are a lot of ICA algorithms. A successful ICA can produce
  - The components that are less Gaussian-like (Central Limit Theorem)
    - Higher-order statistics can measure the non-Gaussianity
  - The components whose **joint dist** and **product of marginal distributions** are with minimal KL-div
    - According to the definition of independence
- One update rule derived from the mutual information

$$\Delta \boldsymbol{W} \propto (D\boldsymbol{I} - f(\boldsymbol{y}) \cdot g(\boldsymbol{y})^{\top})\boldsymbol{W}$$

$$\boldsymbol{W} \leftarrow \boldsymbol{W} + \rho \Delta \boldsymbol{W}$$

$\boldsymbol{W}$: ICA unmixing matrix

$\boldsymbol{y}$: Estimated sources

D : The number of data samples

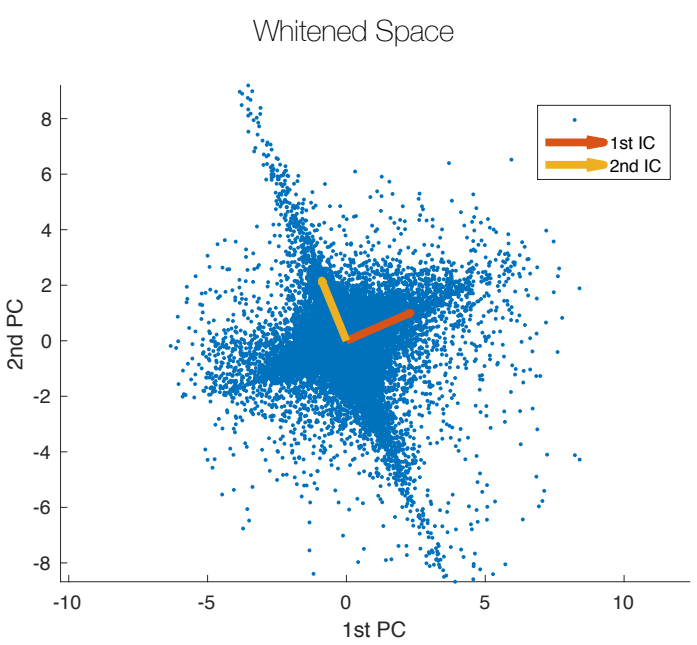
$f(\cdot), g(\cdot)$: Nonlinear functions

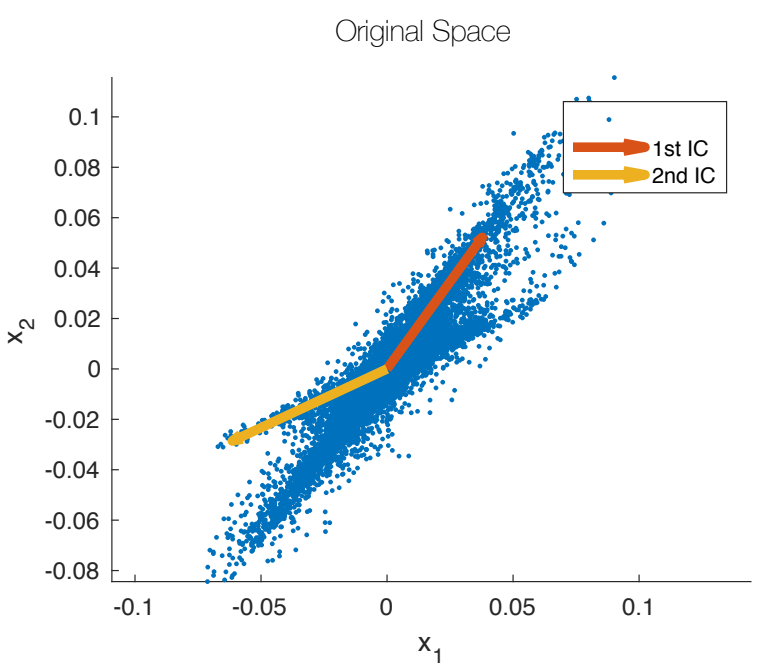
ρ : learning rate

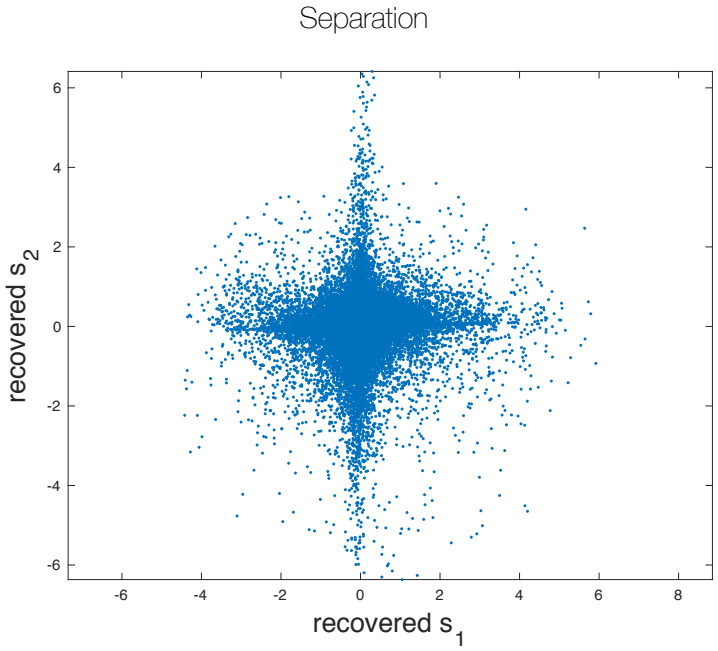
- In what condition $\Delta \boldsymbol{W} = 0$?

ICA for Separation

- Distribution of the recovered sources







$\boldsymbol{x}_1$ $\boldsymbol{x}_2$ $\hat{\boldsymbol{s}}_1$ $\hat{\boldsymbol{s}}_2$

Properties of ICA

- Identifiability Condition
  - ICA wants to find an inverse of the mixing matrix $\boldsymbol{A}$ to recover the sources (kind of indirectly)

$$\boldsymbol{X} = \boldsymbol{A}\boldsymbol{S}$$

$$\boldsymbol{Z} = \underbrace{\left(\boldsymbol{\Lambda}^{-1/2}\boldsymbol{W}_{pca}^{\top}\right)}_{\text{Whitening}}\boldsymbol{X} = \underbrace{\left(\boldsymbol{\Lambda}^{-1/2}\boldsymbol{W}_{pca}^{\top}\right)}_{\text{Whitening}}\boldsymbol{A}\boldsymbol{S}$$

$$\boldsymbol{W}_{ica}\boldsymbol{Z} = \boldsymbol{W}_{ica}\left(\boldsymbol{\Lambda}^{-1/2}\boldsymbol{W}_{pca}^{\top}\right)\boldsymbol{X} = \underbrace{\boldsymbol{W}_{ica}\left(\boldsymbol{\Lambda}^{-1/2}\boldsymbol{W}_{pca}^{\top}\right)}_{\text{Left Inverse of } \boldsymbol{A}}\boldsymbol{A}\boldsymbol{S}$$

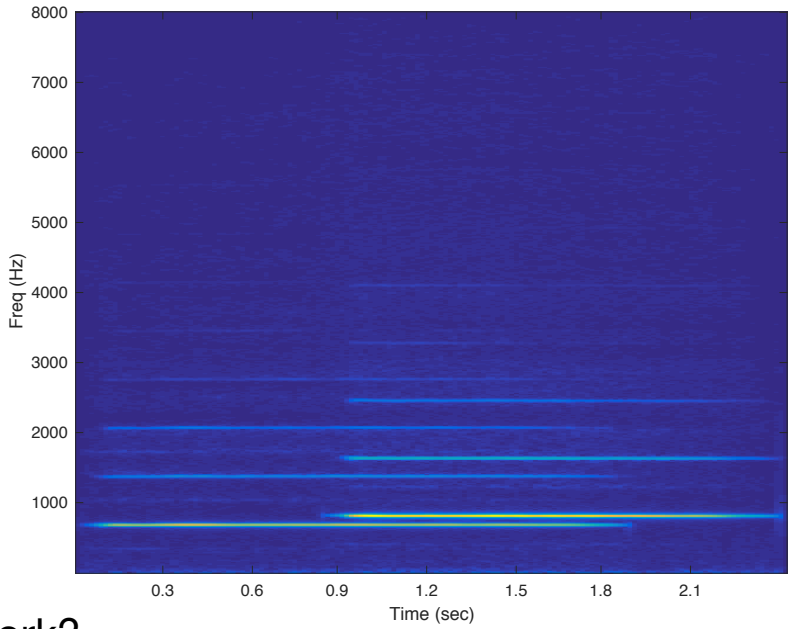
  - $\boldsymbol{A}$ needs to have more rows than columns (or at least square) and be full column rank
  - Sources need to be non-Gaussians
- ICA solutions are subject to ambiguities
  - A permuted and scaled version of the sources: $\begin{bmatrix} 0 & .3 \\ 2 & 0 \end{bmatrix}\boldsymbol{S}$
  - We can build a new mixing scenario, from which ICA starts: $\boldsymbol{X} = \underbrace{\boldsymbol{A}\begin{bmatrix} 0 & .5 \\ 1/0.3 & 0 \end{bmatrix}}_{\text{The new mixing matrix}}\begin{bmatrix} 0 & .3 \\ 2 & 0 \end{bmatrix}\boldsymbol{S}$
  - There are infinitely many permuted/scaled versions of the mixing matrix

Underdetermined Separation

- What if we have only one mixture

- What if the mixing matrix *A* is just a row vector?
- Or, what if the mixing procedure is not linear?
- A single-channel recording with two sources

$$x(n) = [a_1, a_2] \begin{bmatrix} s_1(n) \\ s_2(n) \end{bmatrix}$$



- Would a clustering technique work?
  - No (overlap of sources from 0.9 to 1.8 sec)

Underdetermined Separation

- What if we have only one mixture

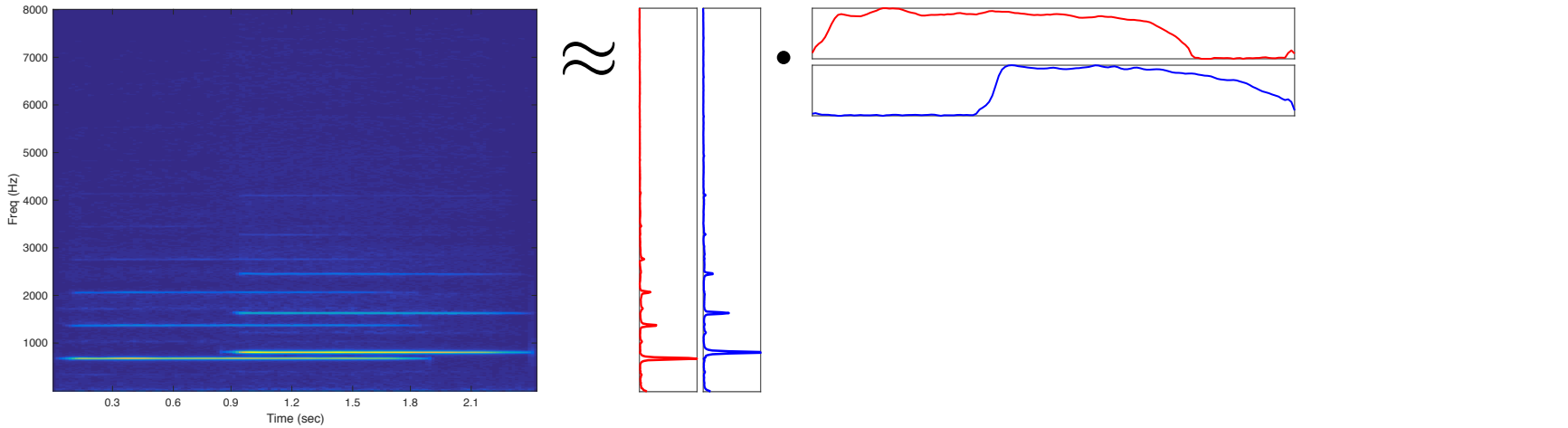
- Can we still separate? How? (Hint: previous slide)
  - Increase the dimension via STFT on the 1D signal
    - And take the magnitudes
- Then, the “mixing” procedure is (we’re not mixing the actual sources, it’s just a concept)

$$X \approx W H$$

F X T matrix of STFT magnitudes
(corresponds to the mixture)

K F-d basis vectors
(corresponds to the ICA mixing matrix)

K X T weight matrix
(corresponds to the ICA sources)



Underdetermined Separation

- What if we have only one mixture

- How can this procedure be source separation?
- We can decompose the matrix into two components

$$\boldsymbol{X} \approx \boldsymbol{W}\boldsymbol{H}$$

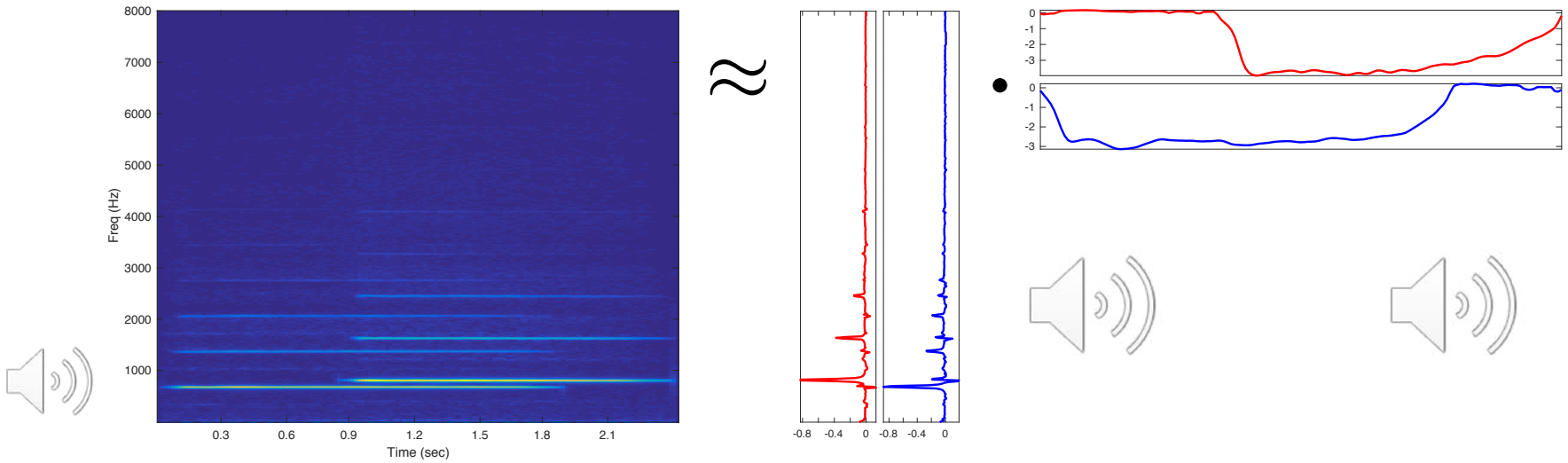
$$\boldsymbol{X} \approx [\boldsymbol{w}_1, \boldsymbol{w}_2] \begin{bmatrix} \boldsymbol{h}_1 \\ \boldsymbol{h}_2 \end{bmatrix} = \boldsymbol{w}_1\boldsymbol{h}_1 + \boldsymbol{w}_2\boldsymbol{h}_2$$

- How do we find the best decomposition?
  - Well, PCA and ICA can actually do it to some degree

Underdetermined Separation

- What if we have only one mixture

- The PCA results on the spectrogram of one mixture of two notes

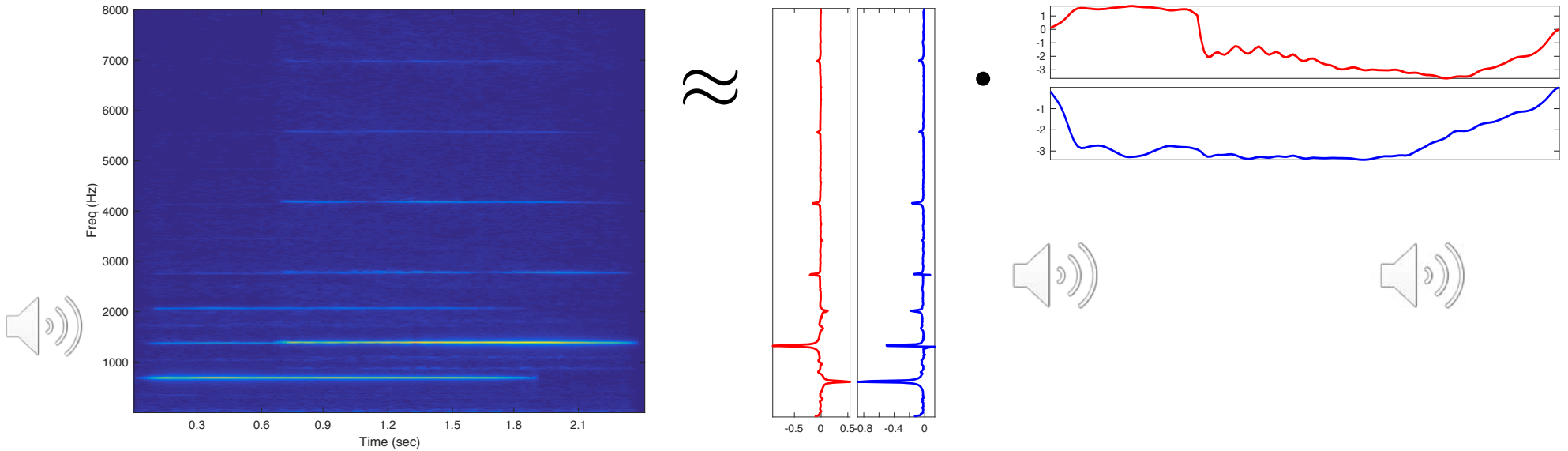


- This seems to work, but..
  - It's a corner case where the source spectra are almost orthogonal (their inner product is near zero)
    - There's no guarantee
  - I don't like all the combination of positive and negative values to recover positive values

Underdetermined Separation

- What if we have only one mixture

- Well, let's see what happens if the spectra are not orthogonal



- There are a lot of overlapping harmonics
  - Furthermore, there's no guarantee that each reconstruction is nonnegative

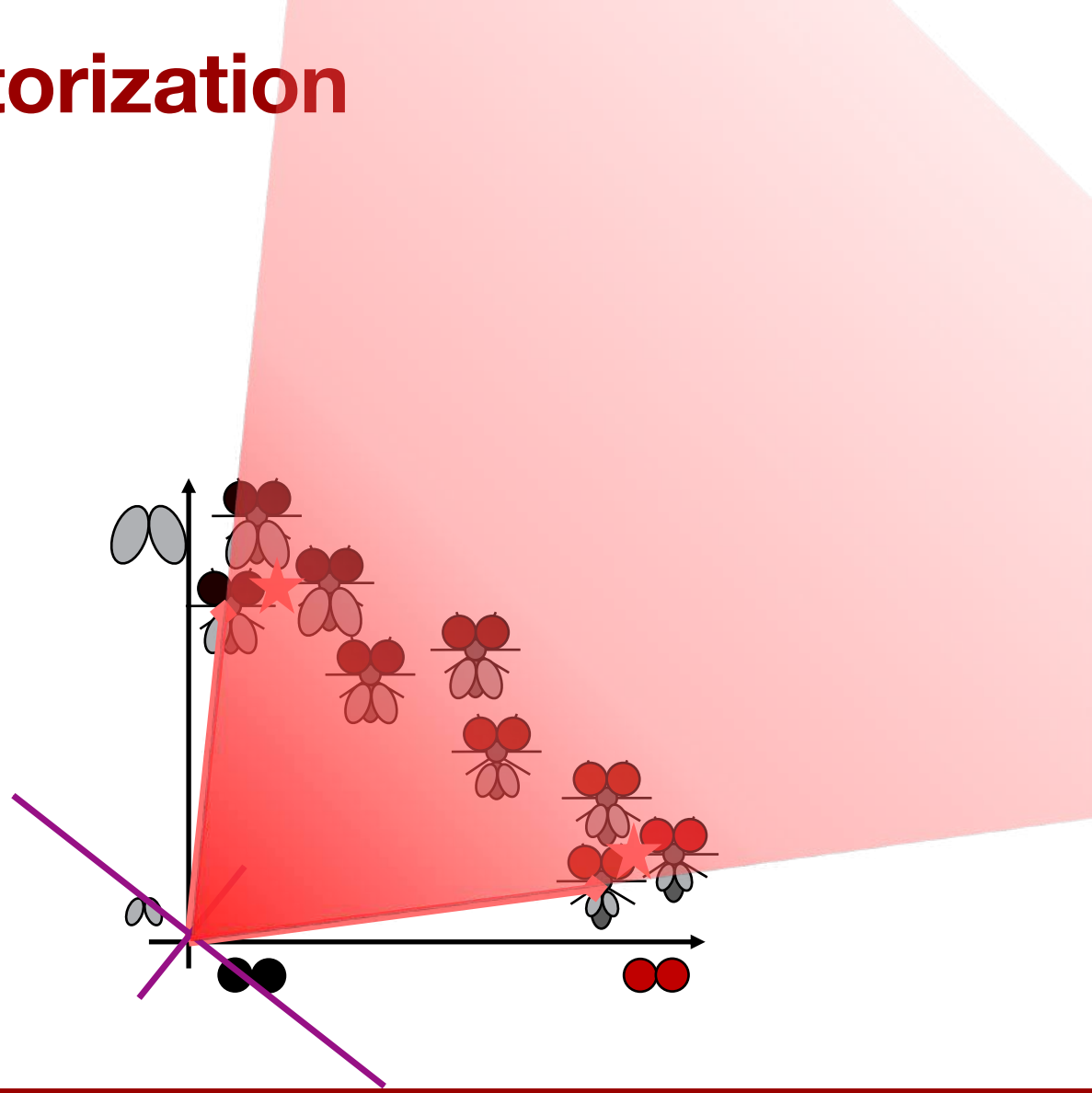
Nonnegative Matrix Factorization

- Parts-based representation

- The mating flies example once again
  - Originally it was a clustering problem

- After mating
  - PCA can find a new subspace
  - With NMF we will find yet another one

- NMF can find basis vectors that are
  - Nonnegative
    - So are their weights
  - Not required to be orthogonal
  - The convex cone defined by the basis is the subspace



Nonnegative Matrix Factorization

- Parts-based representation

- Objective function $$\arg\min_{\boldsymbol{W},\boldsymbol{H}\geq 0} \mathcal{J} = \frac{1}{2}||\boldsymbol{X}-\boldsymbol{W}\boldsymbol{H}||_F^2 = \frac{1}{2}\mathrm{tr}\,(\boldsymbol{X}-\boldsymbol{W}\boldsymbol{H})^\top(\boldsymbol{X}-\boldsymbol{W}\boldsymbol{H})$$

- Derivatives $$\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}} = \boldsymbol{W}\boldsymbol{H}\boldsymbol{H}^\top - \boldsymbol{X}\boldsymbol{H}^\top$$
$$\frac{\partial \mathcal{J}}{\partial \boldsymbol{H}} = \boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{H} - \boldsymbol{W}^\top\boldsymbol{X}$$

- Gradients descent $$\boldsymbol{W} \leftarrow \boldsymbol{W} - \gamma(\boldsymbol{W}\boldsymbol{H}\boldsymbol{H}^\top - \boldsymbol{X}\boldsymbol{H}^\top)$$
$$= \boldsymbol{W} - \frac{\boldsymbol{W}}{\boldsymbol{W}\boldsymbol{H}\boldsymbol{H}^\top} \odot (\boldsymbol{W}\boldsymbol{H}\boldsymbol{H}^\top - \boldsymbol{X}\boldsymbol{H}^\top)$$
← Choose the step size wisely

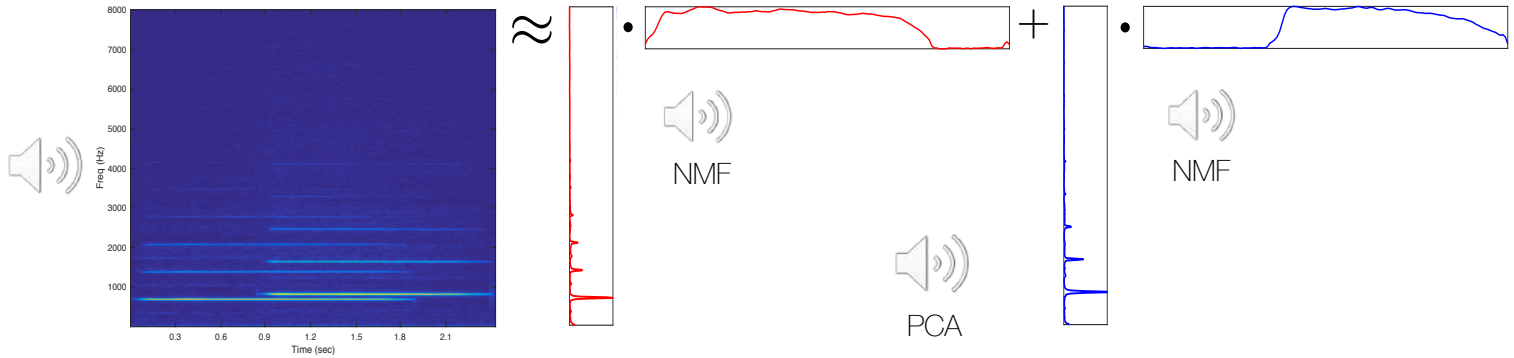
- Multiplicative update rules $$\boldsymbol{W} \leftarrow \boldsymbol{W} \odot \frac{\boldsymbol{X}\boldsymbol{H}^\top}{\boldsymbol{W}\boldsymbol{H}\boldsymbol{H}^\top}$$
$$\boldsymbol{H} \leftarrow \boldsymbol{H} \odot \frac{\boldsymbol{W}^\top\boldsymbol{X}}{\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{H}}$$

$$\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}} = \left[\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}}\right]^+ - \left[\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}}\right]^-$$
$$= \boldsymbol{W}\boldsymbol{H}\boldsymbol{H}^\top - \boldsymbol{X}\boldsymbol{H}^\top$$
← An alternative way to derive update rules

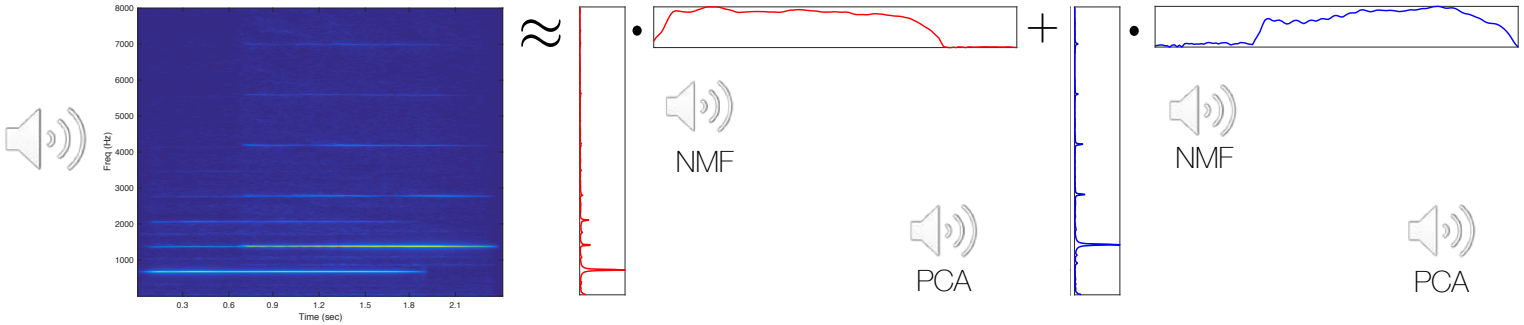
Nonnegative Matrix Factorization

- Parts-based representation

- So, does it work?
  - Below are from actual NMF runs
    - The almost orthogonal case

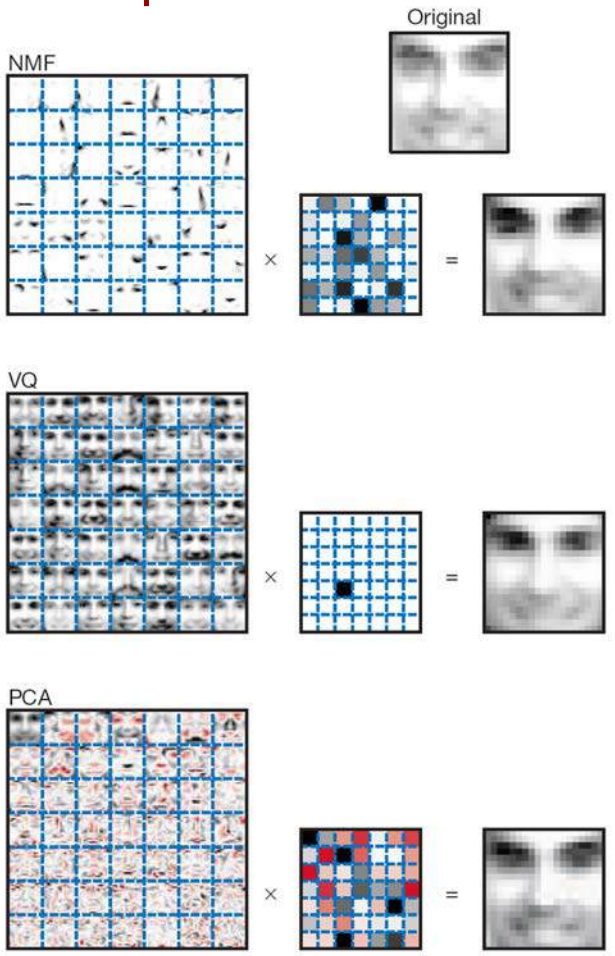


    - The non-orthogonal case



Nonnegative Matrix Factorization

- Parts-based representation



NMF estimates parts-based representations,
something like Lego blocks.
Reconstruction is a linear combination of them,
but subtraction is not allowed.

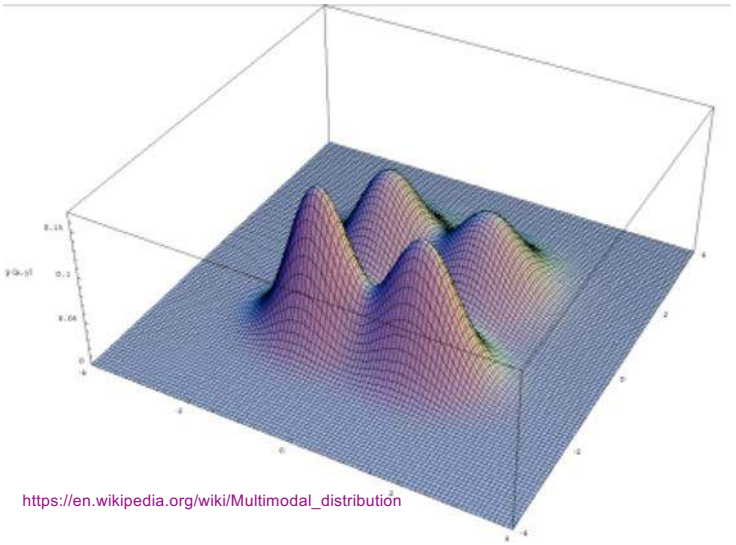
VQ finds a bunch of cluster means.
Reconstruction is choosing the most similar mean.

PCA finds the holistic eigenfaces.
From the important one down to the subtle ones.

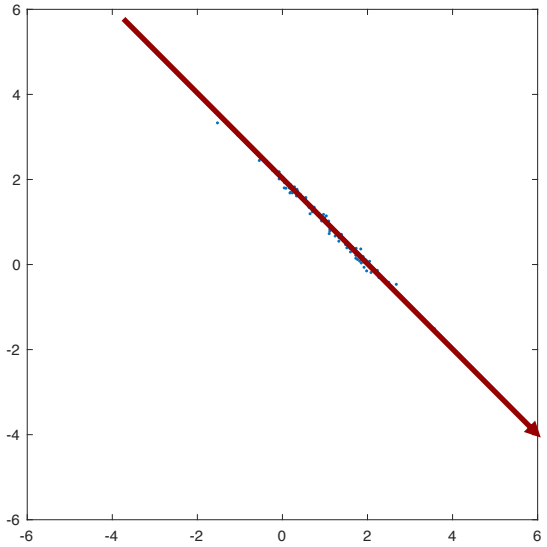
Lee&Seung Nature 1999, Lee&Seung NIPS 2001

Recap

- Mixture models (clustering)
  - Assume less modes than the samples
  - Each sample belongs to one cluster, though there's uncertainty
  - Compression is done by keeping the cluster means as representatives
- Dimension reduction
  - Assumes the samples lie on a lower dimensional subspace
  - The subspace is defined with K basis vectors (so called dictionary)
    - $K < D$ (not always)
  - Various kinds of basis vectors depending on the constraints
    - Orthogonality, variance maximization, independence, nonnegativity, etc.

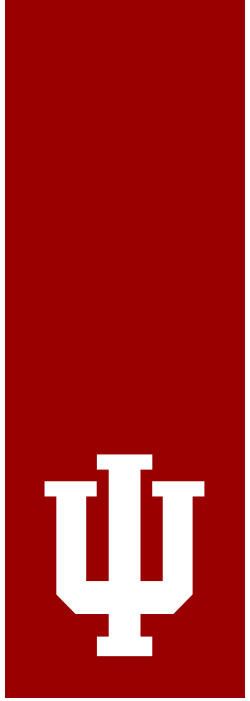


https://en.wikipedia.org/wiki/Multimodal_distribution



Reading

- 6.1-6.7



Thank You!