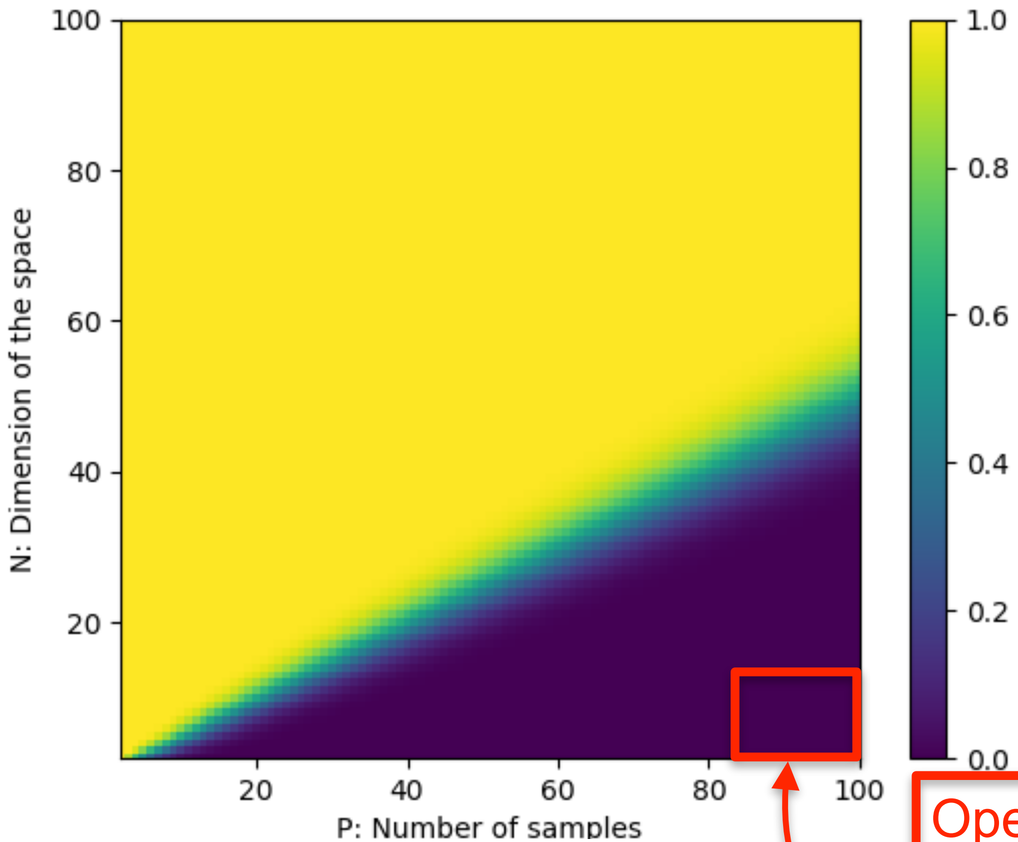
Linear Dimensionality Reduction

Pascal Fua
IC-CVLab

Reminder: Cover's Theorem

A complex pattern-classification problem, cast in a high-dimensional space nonlinearly, is more likely to be linearly separable than in a low-dimensional space, provided that the space is not densely populated.

Geometrical and Statistical properties of systems of linear inequalities with applications,1965



N : Dimension of space

p : Number of samples

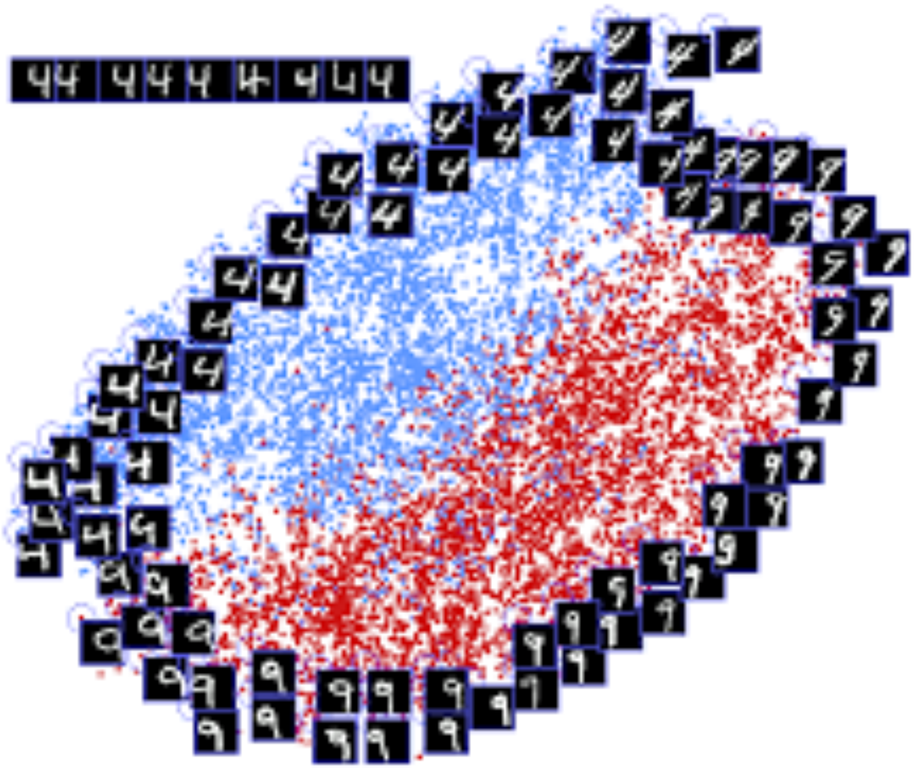
$\frac{C(p,N)}{2^p}$: Percentage of separable partitions

Operating range

- ML shouldn't work.
- Yet it does.

?

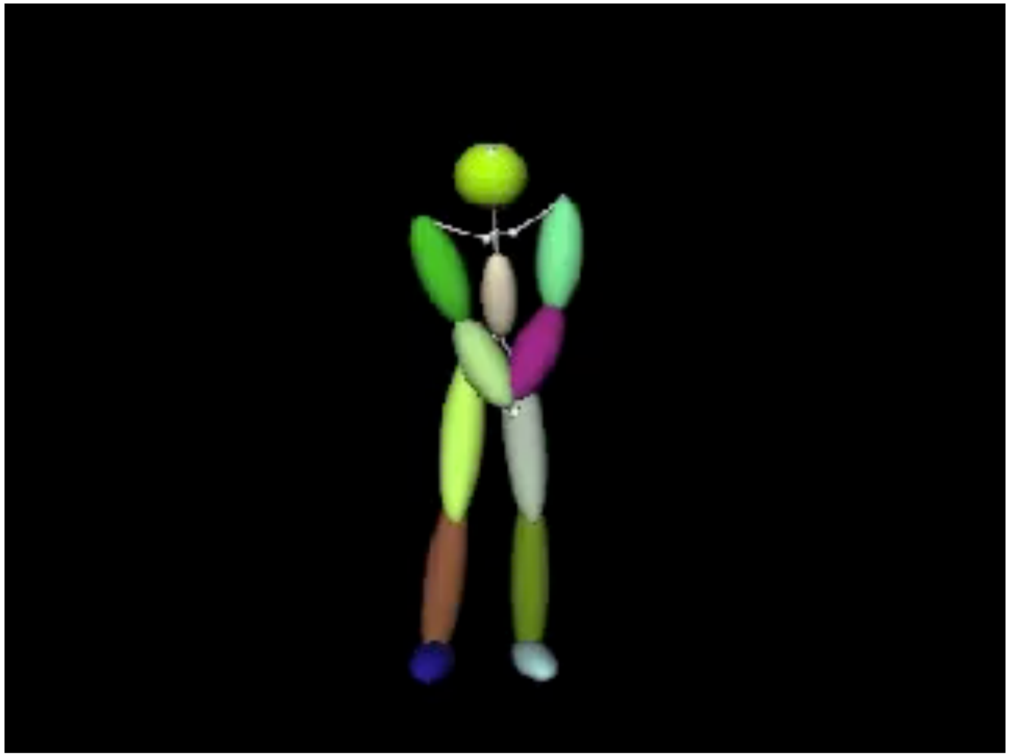
Example: MNIST Again



- The MNIST images are 28x28 arrays.
- They are **not** uniformly distributed in R^{784}.
- In fact they exist on a low dimensional manifold.

Example: Golf Swings

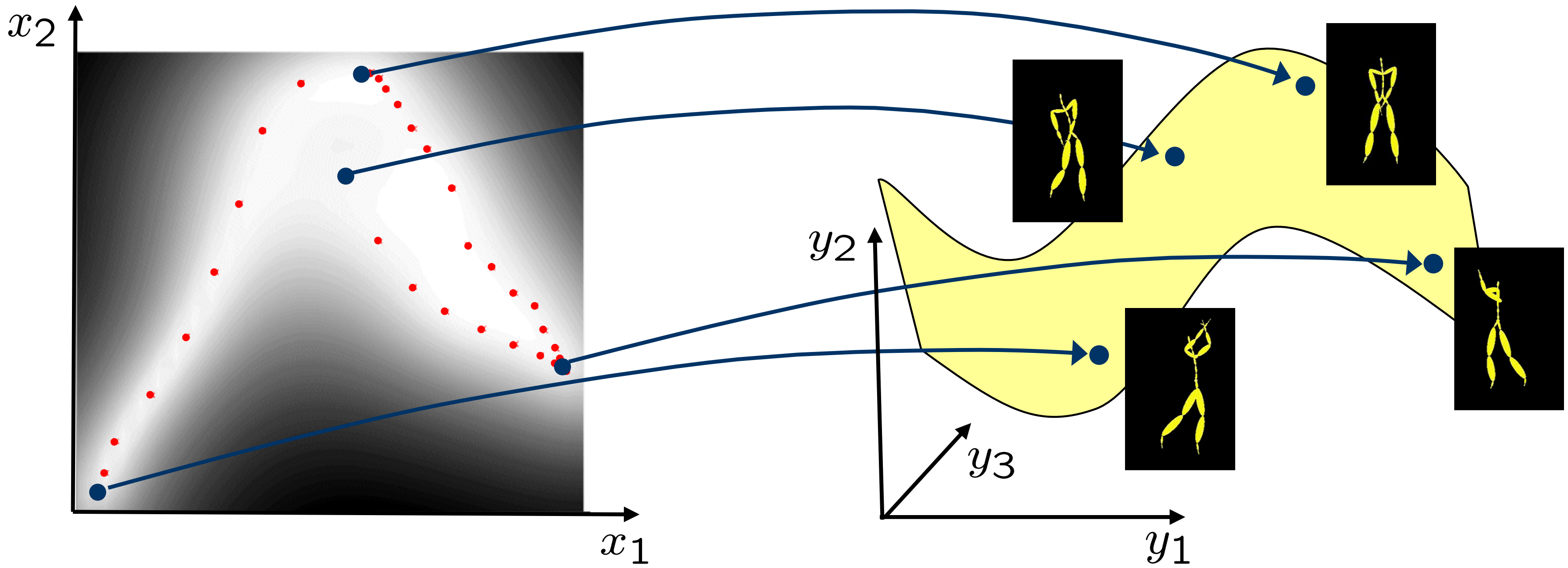
....



The skeleton used to describe the body pose has 51 degrees of freedom.

Urtasun et al. , CVPR'05

Example: Golf Latent Space



Latent Space ($\mathbf{X}$)

Pose Space ($\mathbf{Y}$)

- The golf swings exist on a 2D manifold in R^{51}.
- There is a mapping from a 2D space to this manifold.
- This can be said of MNIST images, golf swings, and many other things.

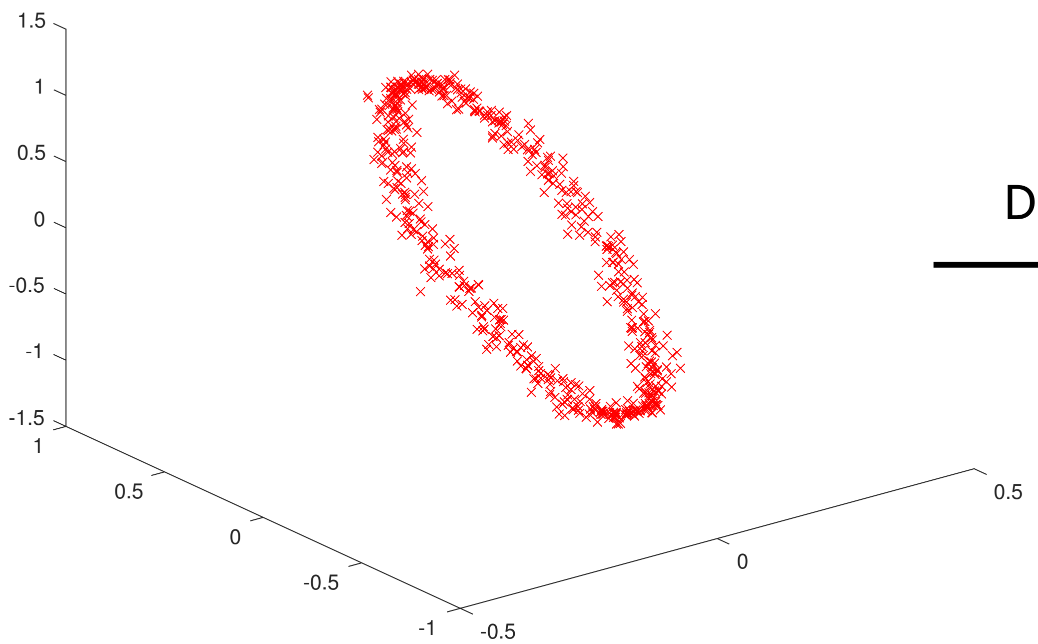
—> This is what makes many ML techniques viable.

Urtasun et al. , CVPR’05

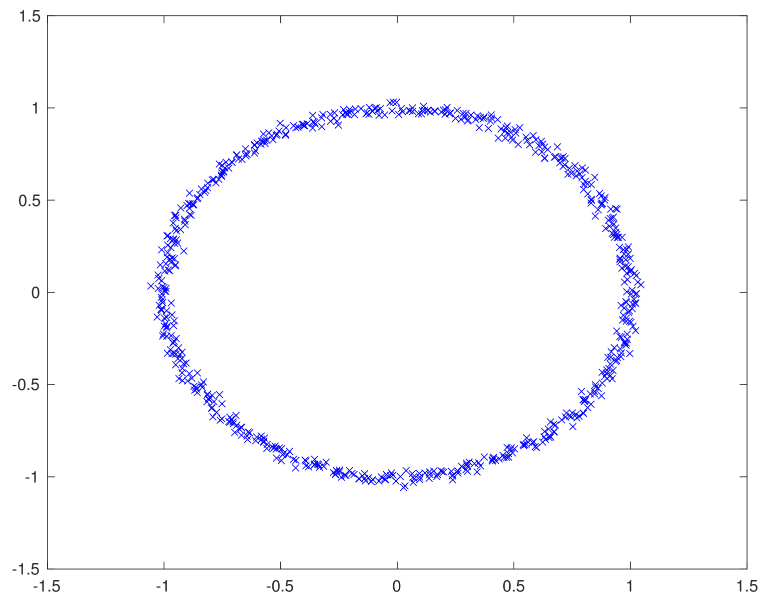
Dimensionality Reduction

It involves:

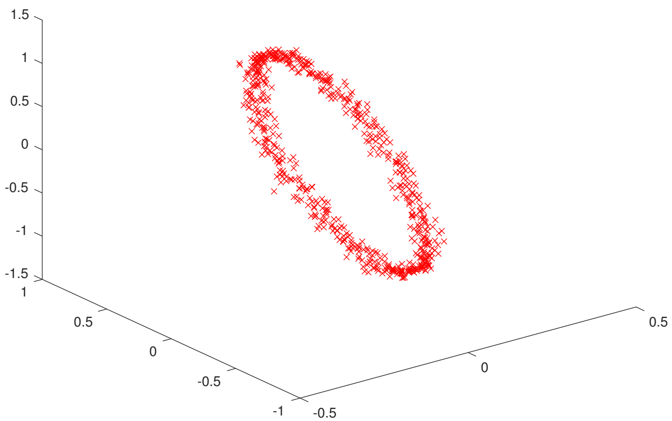
- discovering the data manifold,
- finding a low-dimensional representation of the data,
- some loss of information and hopefully noise reduction.



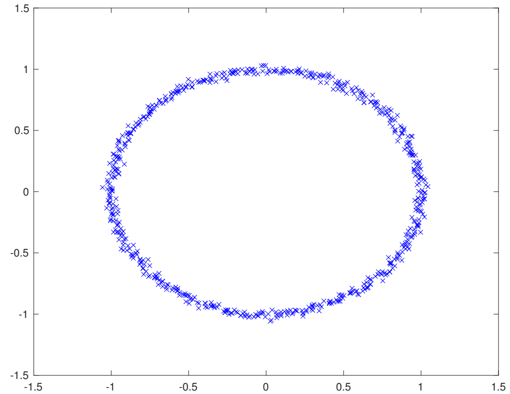
DR



Formalization



$\mathbf{x}$



$\mathbf{y}$

Our goal is to find a mapping $\mathbf{y}_i = f(\mathbf{x}_i)$

- $\mathbf{x}_i \in \mathbb{R}^D$: High-dimensional data sample
- $\mathbf{y}_i \in \mathbb{R}^d$: Low-dimensional representation

How about a linear one $\mathbf{y}_i = \underbrace{\mathbf{W}^T}_{D \times d} \mathbf{x}_i$?

We will talk about Non-Linear ones next week.

Principal Component Analysis (PCA)

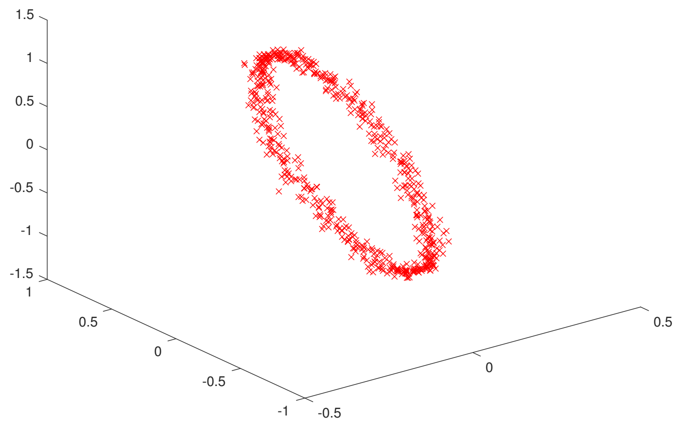
Given N samples $\{\mathbf{x}_i\}$, PCA yields a projection of the form

$$\mathbf{y}_i = \mathbf{W}^T(\mathbf{x}_i - \bar{\mathbf{x}}) \quad \text{s.t.} \quad \mathbf{W}^T\mathbf{W} = \mathbf{I}_d$$

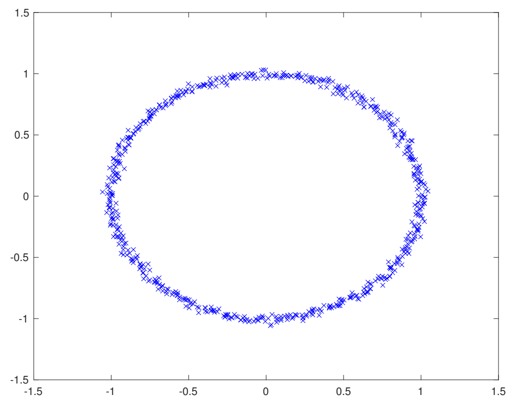
$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i$$

What do we want this projection to achieve?

PCA Objective



x



y

- We want to keep most of the "important" signal while removing the noise.
- This can be achieved by finding directions in which there is a large variance, that is, for the j^{th} output dimension, we want to maximize

$$\mathrm{var}(\{y_i^{(j)}\}) = \frac{1}{N}\sum_{i=1}^{N}(y_i^{(j)} - \bar{y}^{(j)})^2,$$

where $\bar{y}^{(j)}$ is the mean of the dimension of the j^{th} data point after projection.

Variance Maximization

Let us begin with the projection into a 1D space:

- We use a D-dimensional vector $\mathbf{w}_1$, s.t., $\mathbf{w}_1^T\mathbf{w}_1 = 1$, instead of a matrix $\mathbf{W} \in \mathbb{R}^{D\times d}$.
- In this case, the mean of the data after projection is

$$
\begin{aligned}
\bar{y} &= \frac{1}{N}\sum_{i=1}^{N} y_i \\
&= \frac{1}{N}\sum_{i=1}^{N} \mathbf{w}_1^T\mathbf{x}_i \\
&= \mathbf{w}_1^T\left(\frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i\right) \\
&= \mathbf{w}_1^T\bar{\mathbf{x}}
\end{aligned}
$$

Variance Maximization

Therefore, the variance of the data after projection is

$$\mathrm{var}(\{y_i\}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y})^2 = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{w}_1^T\mathbf{x}_i - \mathbf{w}_1^T\bar{\mathbf{x}})^2$$

$$= \frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{w}_1^T(\mathbf{x}_i - \bar{\mathbf{x}})\right)^2 = \frac{1}{N}\sum_{i=1}^{N}\mathbf{w}_1^T(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T\mathbf{w}_1$$

$$= \mathbf{w}_1^T\left(\frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T\right)\mathbf{w}_1 = \mathbf{w}_1^T\mathbf{C}\mathbf{w}_1$$

where $\mathbf{C}$ is the input data covariance matrix

$$\mathbf{C} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T$$

Variance Maximization

- Ultimately, we seek to solve

$$\max_{\mathbf{w}_1} \mathbf{w}_1^T \mathbf{C} \mathbf{w}_1 \text{ subject to } \mathbf{w}_1^T \mathbf{w}_1 = 1.$$

➡$\mathbf{w}_1$ should be the eigenvector associated to the larger eigenvalue of $\mathbf{C}$.

Back to d > 1

- To obtain an output representation that is more than 1D, i.e., $d > 1$, we can iterate:
  - ➡The second projection vector $\mathbf{w}_2$ corresponds to the eigenvector of $\mathbf{C}$ with the second largest eigenvalue
  - ➡ The third vector $\mathbf{w}_3$ to the eigenvector with the third largest eigenvalue
  - ➡…
- The matrix $\mathbf{W}$ is obtained by concatenating the resulting vectors

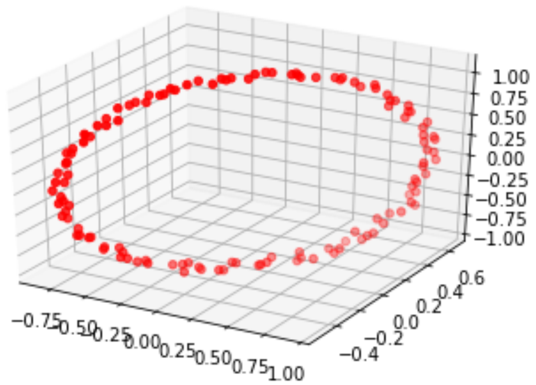
$$\mathbf{W} = \left[\mathbf{w}_1 \,|\, \mathbf{w}_2 \,|\, \cdots \,|\, \mathbf{w}_d\right] \in \mathbb{R}^{D\times d}$$

- This is guaranteed to satisfy the constraint $\mathbf{W}^T\mathbf{W} = \mathbf{I}_d$ because the eigenvectors of a matrix are orthogonal and of norm 1.

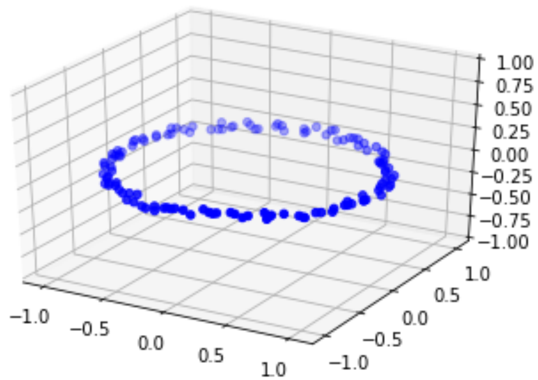
- The amount of explained variance is $\mathbf{W}^T\mathbf{C}\mathbf{W} = \sum_i \lambda_i$.

PCA without Dimensionality Reduction

- In the limit, one can use all dimensions, i.e., set $d = D$
  - –There is therefore no reduction of dimensionality
  - –In 3D, you can think of this as a rotation of the data
  - –This incurs no loss of information
  - –The $d = D$ dimensions in the new space are uncorrelated



x

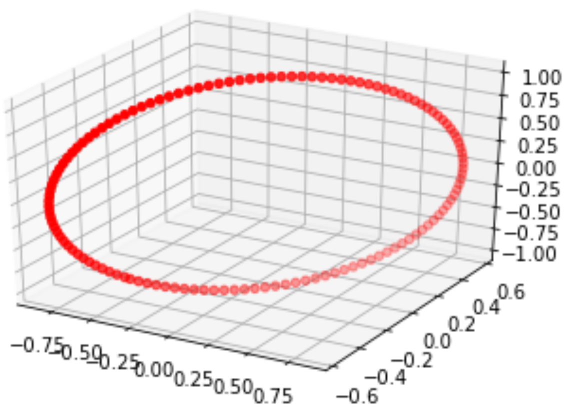


y

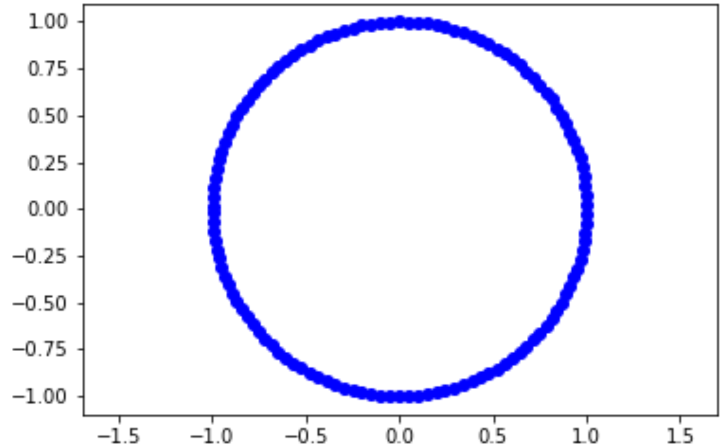
PCA without Loss of Information

Another option is to keep all the eigenvectors corresponding to non-zero eigenvalues:

- This works best the data is truly low-dimensional.
- The resulting $\{\mathbf{y}_i\}$ are lower dimensional $(d < D)$ without loss of information.
- This happens trivially when there are fewer samples than dimensions $(N < D)$.



$\mathbf{x}$



$\mathbf{y}$

EPFL

PCA with Loss of Information

- In practice, one typically truncates the eigenvalues so as to discard some that are non-zero.
  - This can be achieved by aiming to retain a pre-defined percentage of the data variance, measured as the sum of eigenvalues.
  - For example, to retain at least 90% of the variance, one can search for d such that

$$\sum_{j=1}^{d} \lambda_j \geq 0.9 \cdot \sum_{k=1}^{D} \lambda_k ,$$

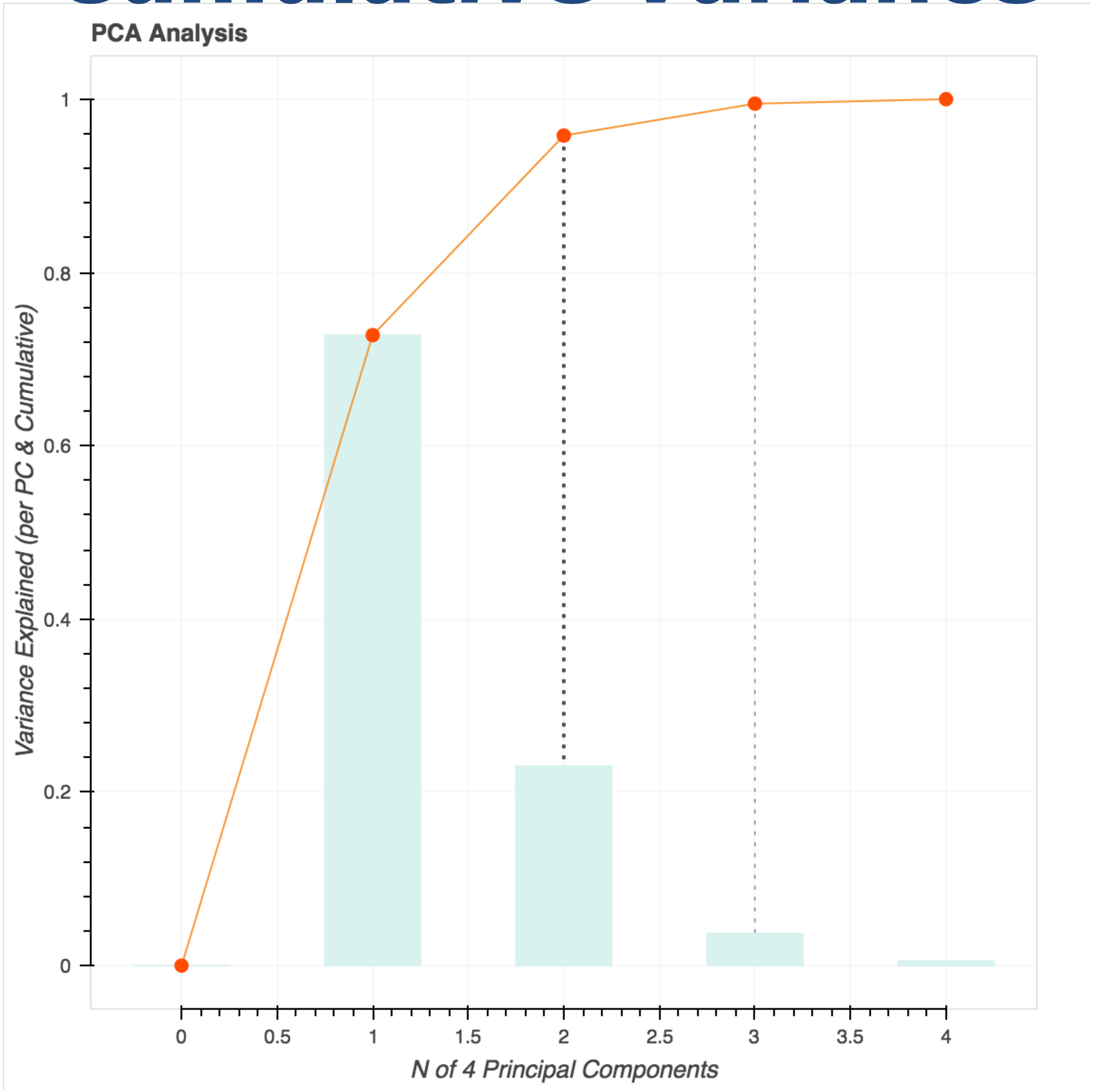
    assuming the eigenvalues to be sorted in decreasing order.
- The resulting $\{\mathbf{y}_i\}$ have an even lower dimension.

Classifying Irises

- UCI Iris dataset:
  - 3 different types of irises
  - 4 attributes
    - ✓ petal length
    - ✓ petal width
    - ✓ sepal length
    - ✓sepal width

- 4 attributes means $D = 4$, so d is at most 4.

Cumulative Variance



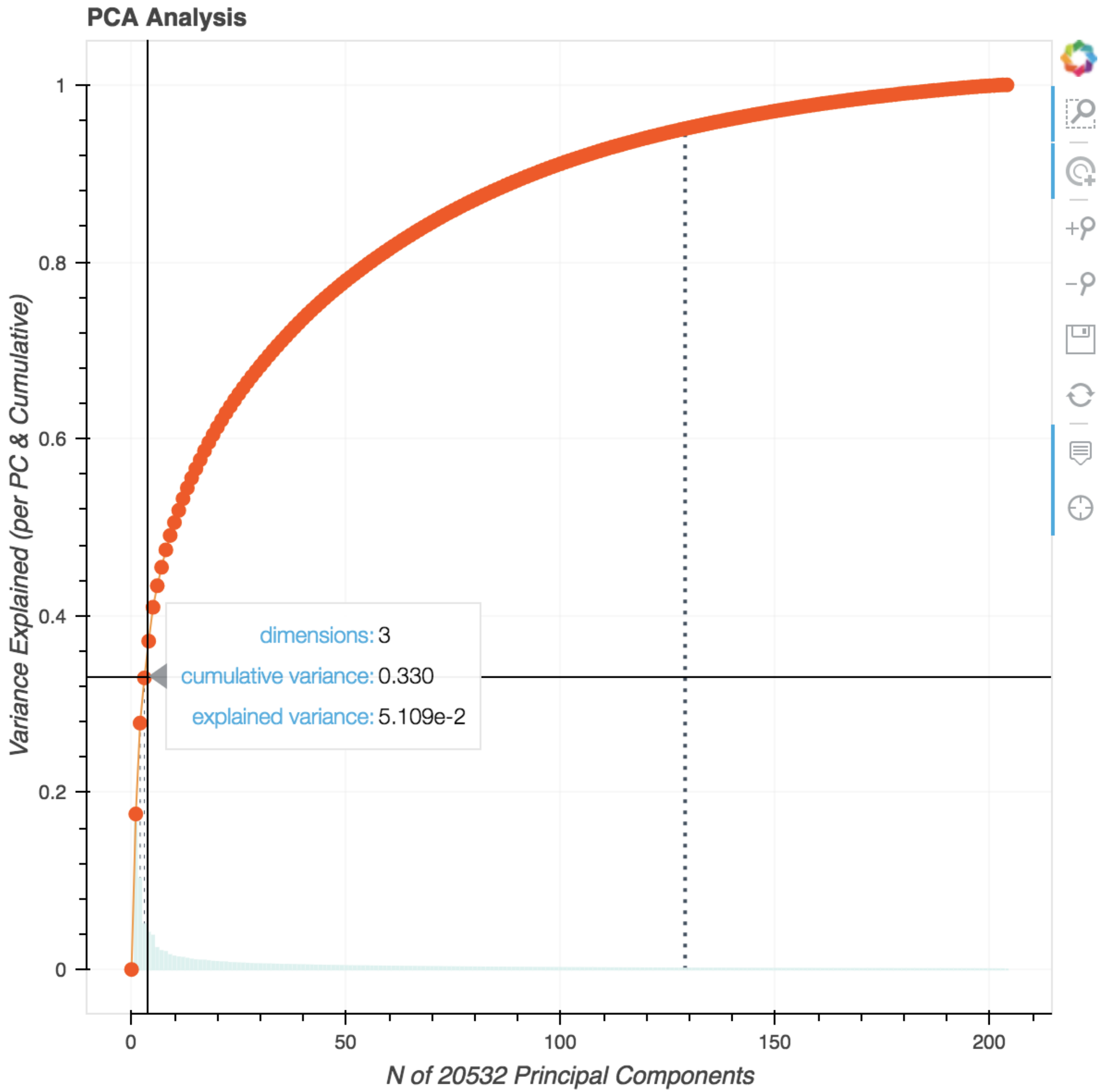
Cumulative variance explained by the principal components

Medical Application

- The Cancer Genome Atlas breast cancer RNA-Seq dataset:
  - Normal tissue vs primary tumor:
  - 20532 features, that is genes for which an expression is measured.
  - 204 samples.
- 20532 features means $D = 20532$, so d is at most 20532.
- However, because we only have $N = 204$ samples, d is at most 204.

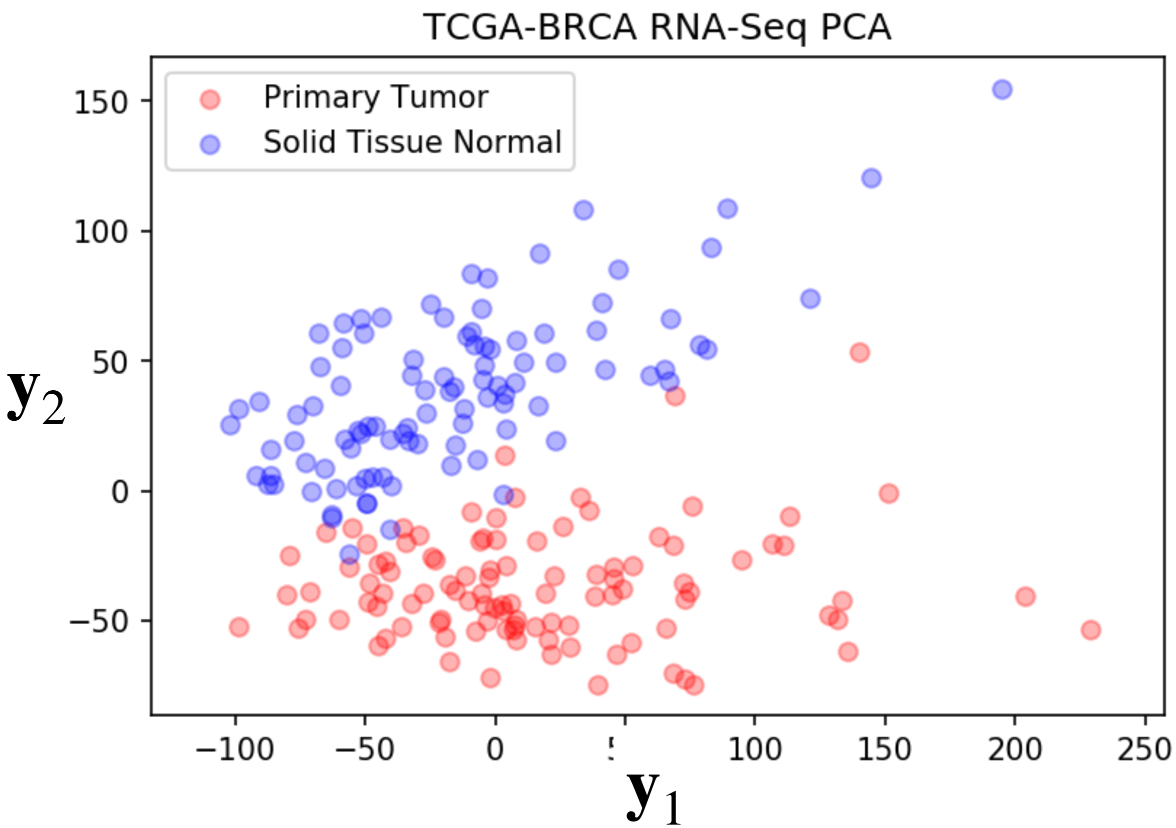
https://medium.com/cascade-bio-blog/creating-visualizations-to-better-understand-your-data-and-models-part-1-a51e7e5af9c0

Cumulative Variance



Cumulative variance explained by the principal components

Medical Application



Samples of the Cancer Genome Atlas breast cancer RNA-Seq dataset projected in 2D.

—> Relatively easy to classify.

PCA: Mapping

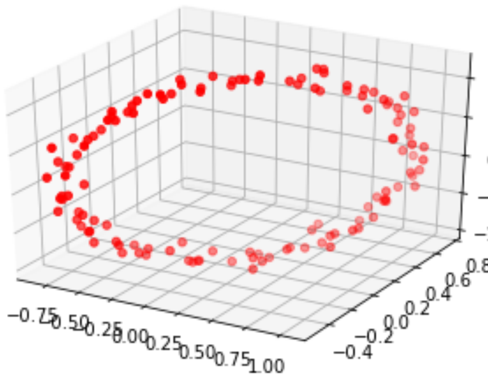
- PCA not only reduces the dimensionality of the original data. It provides a continuous mapping from the low-dimensional space to the high-dimensional one
- That is, for any $\mathbf{y} \in \mathbb{R}^d$, we can compute a point in the high-dimensional space as

$$\begin{aligned}\hat{\mathbf{x}} &= \bar{\mathbf{x}} + \mathbf{W}\mathbf{y} \\ &= \bar{\mathbf{x}} + \sum \alpha_i \mathbf{w_i} \text{ with } \mathbf{y} = [\alpha_{\mathbf{1}}, \ldots, \alpha_{\mathbf{d}}]^{\mathbf{T}}\end{aligned}$$

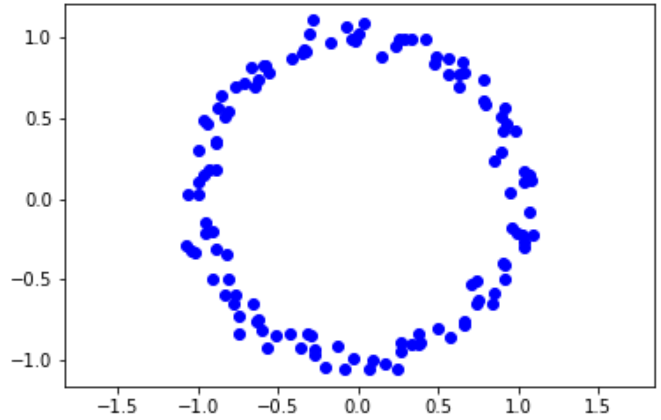
- This mapping constrains $\hat{\mathbf{x}}$ to lie in a subspace, and thus provides a form of regularization.

Toy Example

- Original data



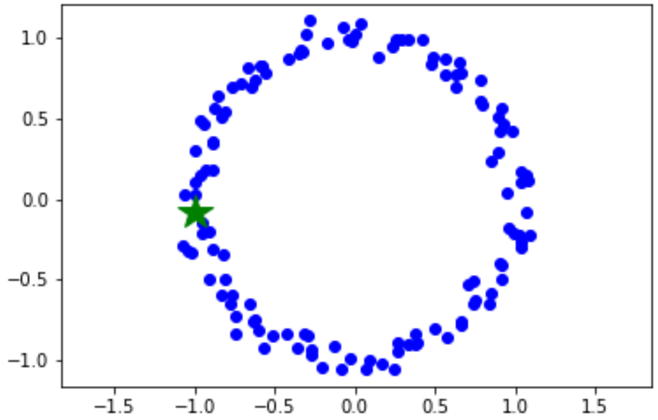
PCA



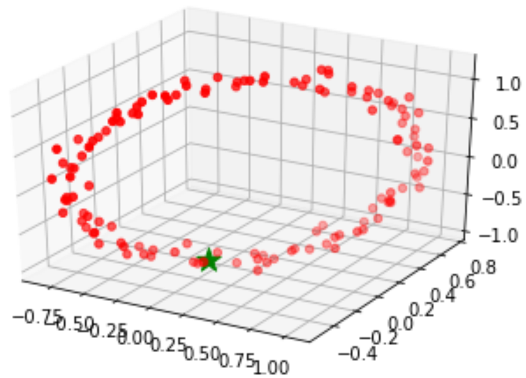
$\mathbf{x}$

$$\mathbf{y} = \mathbf{W}^{\mathbf{T}}(\mathbf{x} - \bar{\mathbf{x}})$$

- New point (green star)



Mapping



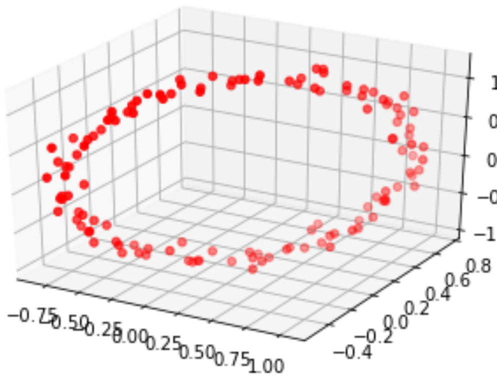
$\mathbf{y}$

$$\hat{\mathbf{x}} = \bar{\mathbf{x}} + \mathbf{W}\mathbf{y}$$

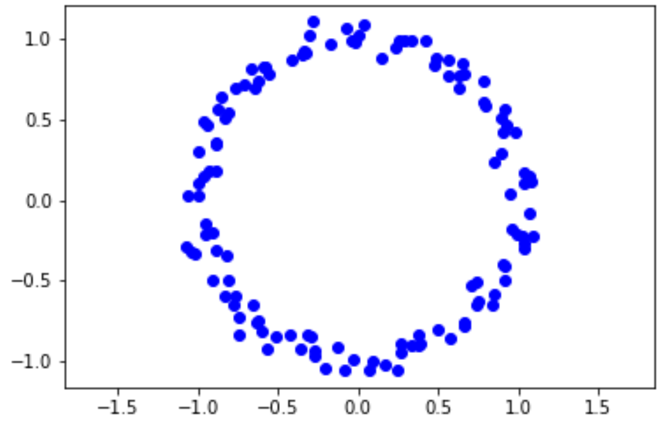
$$= \bar{\mathbf{x}} + \sum_i \alpha_i \mathbf{w_i}$$

Toy Example

- Original data



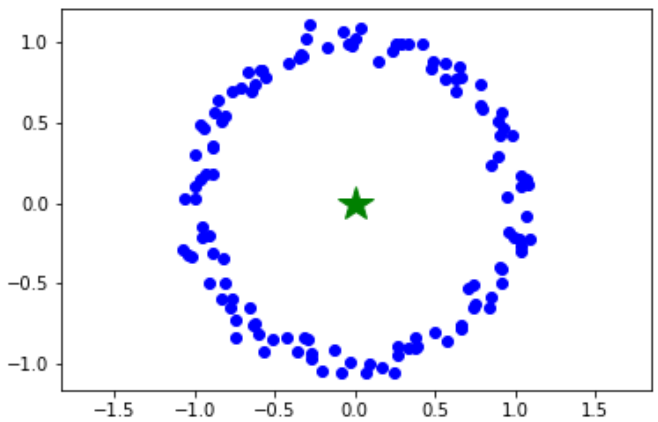
PCA



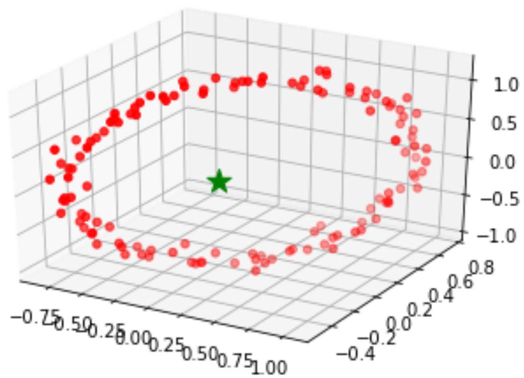
$\mathbf{x}$

$$\mathbf{y} = \mathbf{W}^{\mathbf{T}}(\mathbf{x} - \bar{\mathbf{x}})$$

- New point (green star)



Mapping

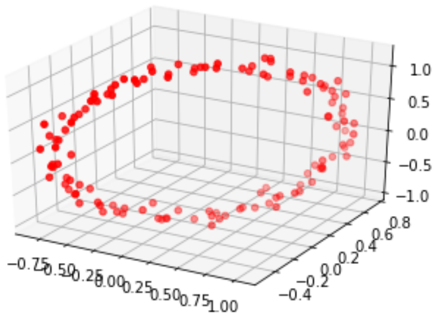


$\mathbf{y}$

$$\hat{\mathbf{x}} = \bar{\mathbf{x}} + \mathbf{W}\mathbf{y}$$

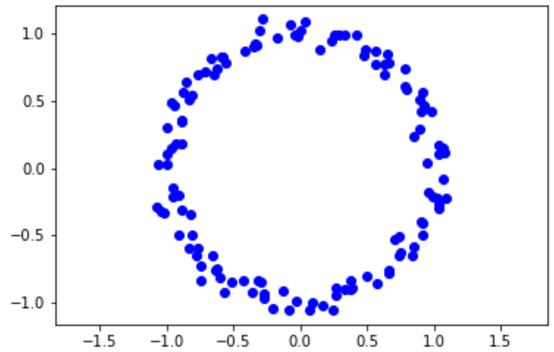
$$= \bar{\mathbf{x}} + \sum_i \alpha_i \mathbf{w_i}$$

Optimal Linear Mapping



PCA $\longrightarrow$

$\longleftarrow$ Mapping



$\mathbf{x}$

$\mathbf{y} = \mathbf{W}^{\mathbf{T}}(\mathbf{x} - \bar{\mathbf{x}})$

$\hat{\mathbf{x}} = \bar{\mathbf{x}} + \mathbf{W}\mathbf{y}$

- This mapping incurs some loss of information.
- However, the corresponding rectangular matrix $\mathbf{W}$ is the orthogonal matrix that minimizes the reconstruction error

$$e = \|\hat{\mathbf{x}} - \mathbf{x}\|^2$$

where

$$\hat{\mathbf{x}} = \bar{\mathbf{x}} + \mathbf{W}\mathbf{y} = \bar{\mathbf{x}} + \mathbf{W}\mathbf{W}^T(\mathbf{x} - \bar{\mathbf{x}})$$

EigenFaces

x

w

- The **x** are vectors representing the images. The **w** are the eigenvectors of the covariance matrix.
- Exact reconstruction:

$$\hat{\mathbf{x}} = \bar{x} + \sum_{n=1}^{N^2} \alpha_i \mathbf{w}_i$$

- Approximate reconstruction:

$$\hat{\mathbf{x}} = \bar{x} + \sum_{n=1}^{M} \alpha_i \mathbf{w}_i \text{ with } M \ll N^2$$

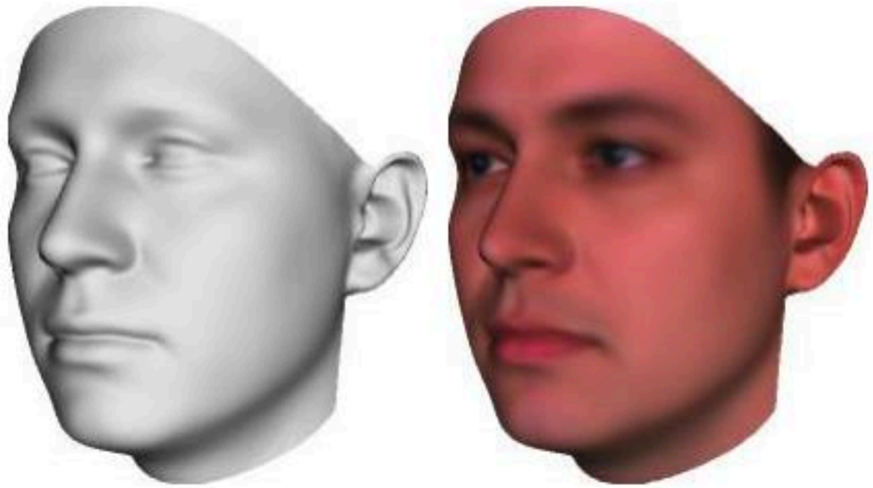
Turk and Pentland, CVPR'91

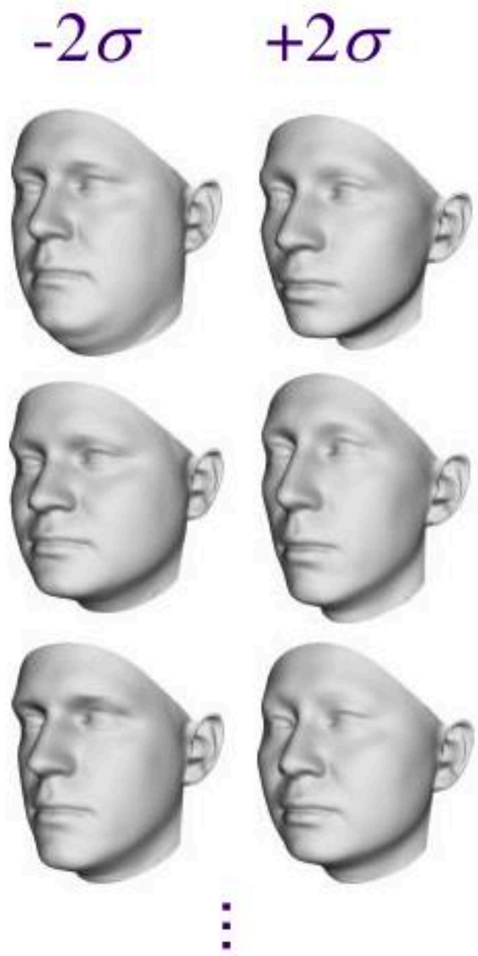
Reconstruction Using Eigenfaces

x

Project and reconstruct left image to produce the right one.

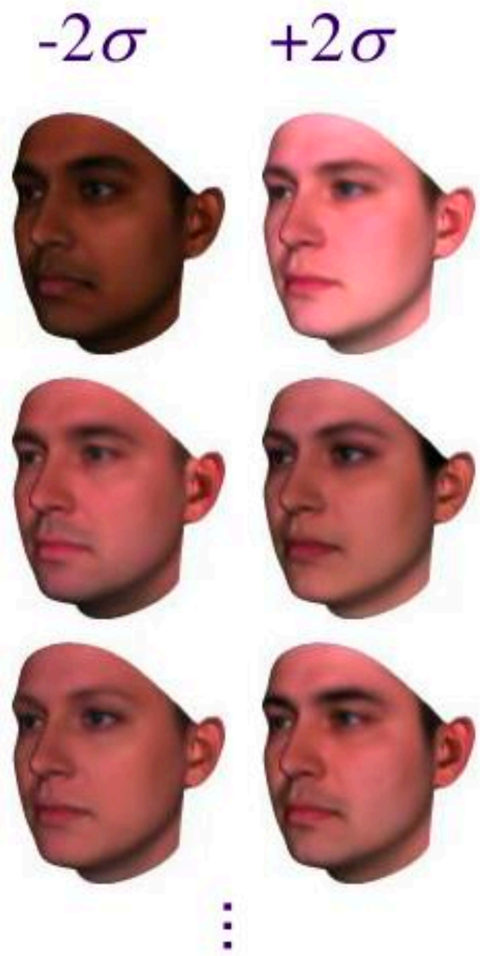
3D Face Modeling





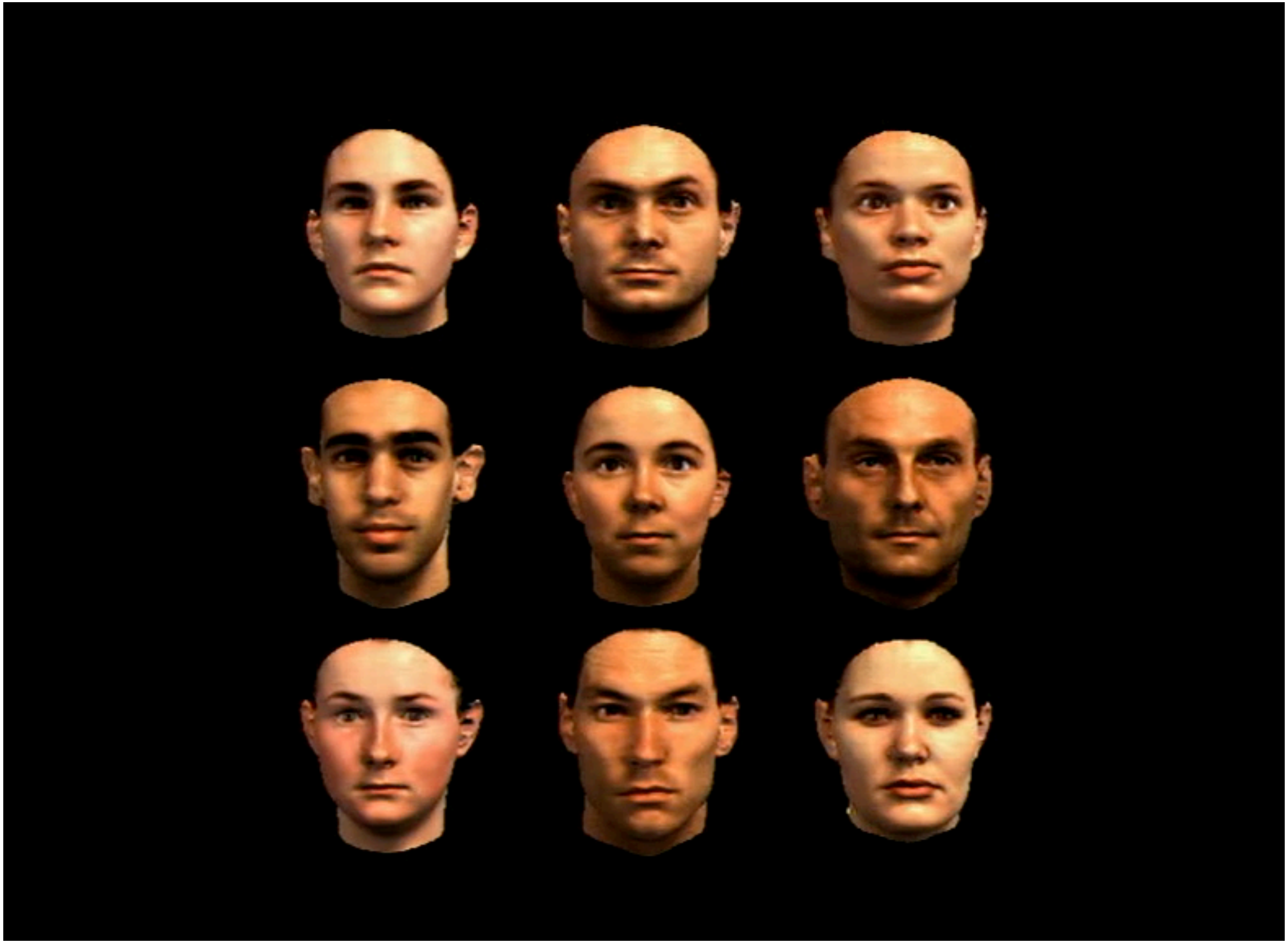
$$\mathbf{S} = \overline{\mathbf{S}} + \sum_{j=1}^{m} \alpha_j \mathbf{S}_j$$

$$\mathbf{T} = \overline{\mathbf{T}} + \sum_{j=1}^{m} \beta_j \mathbf{T}_j$$



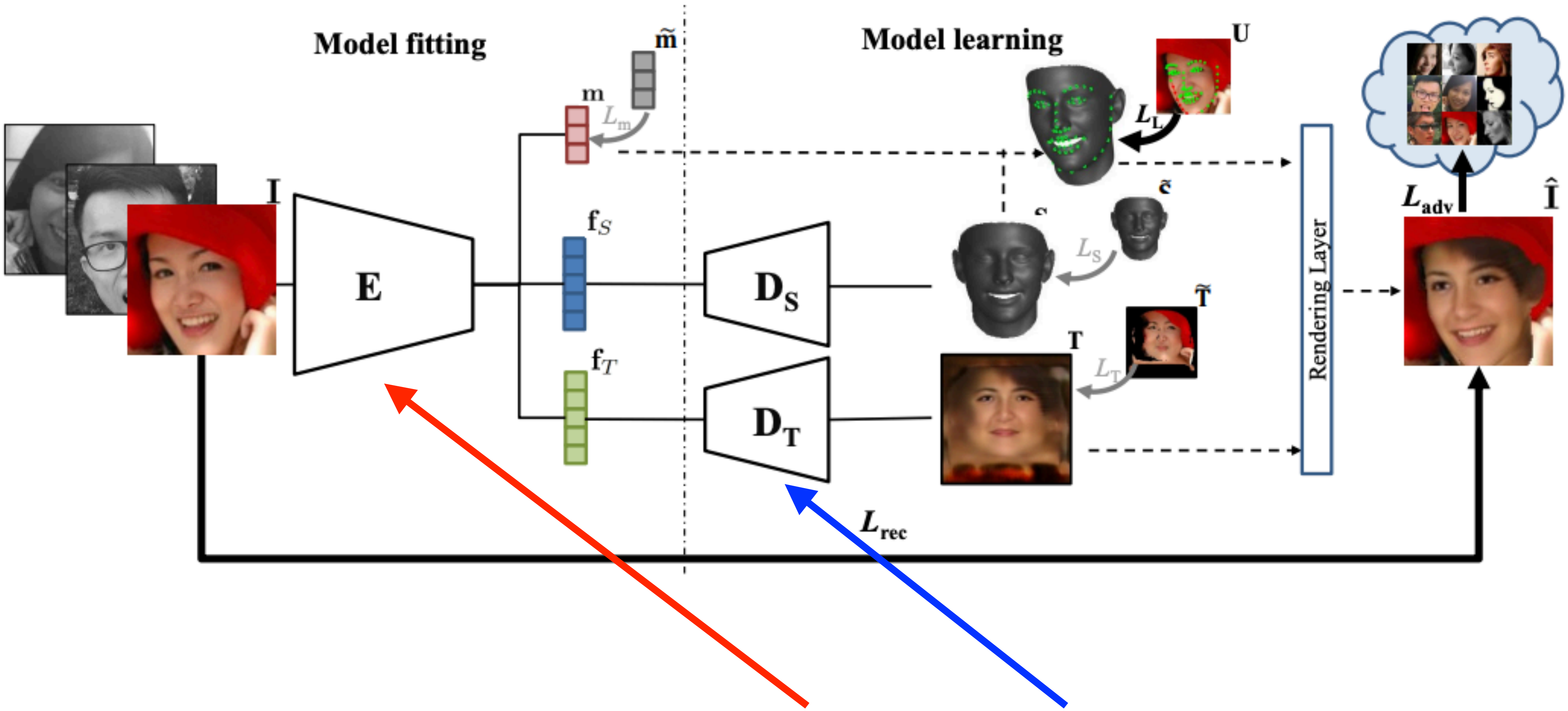
Blanz & Vetter, SIGGRAPH'99

3D Face Modeling



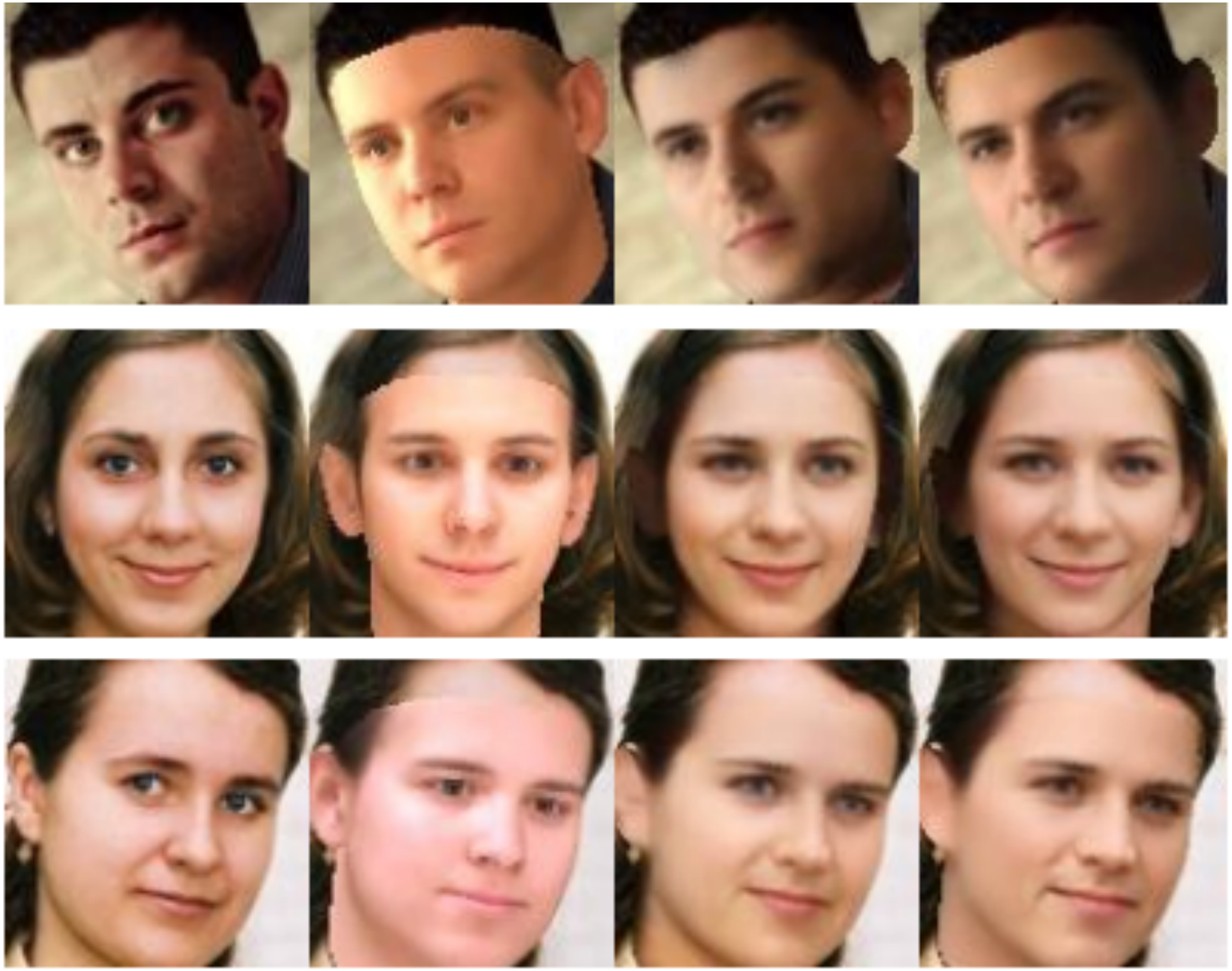
Blanz & Vetter, SIGGRAPH'99

Non Linear Face Models



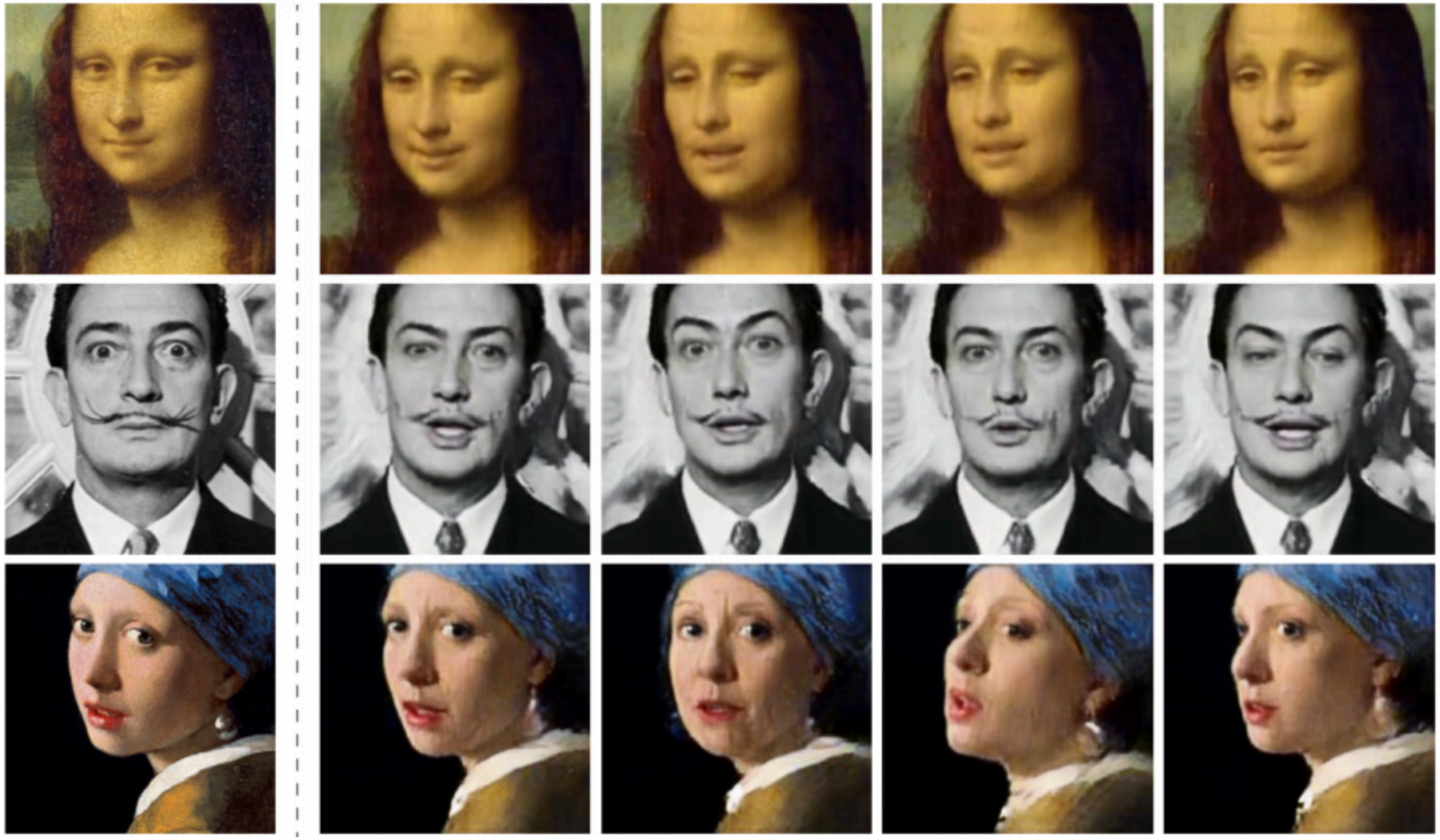
- PCA has been replaced by an encoder / decoder architecture.
- To be discussed next week.

Linear vs Non Linear

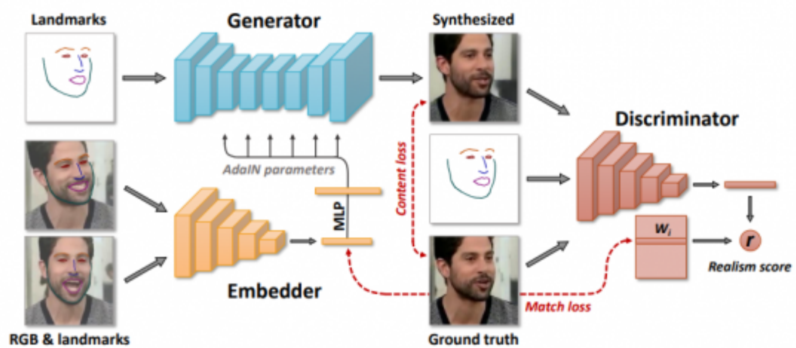


Original Linear Non Linear

20 Years Later: Deep Fakes

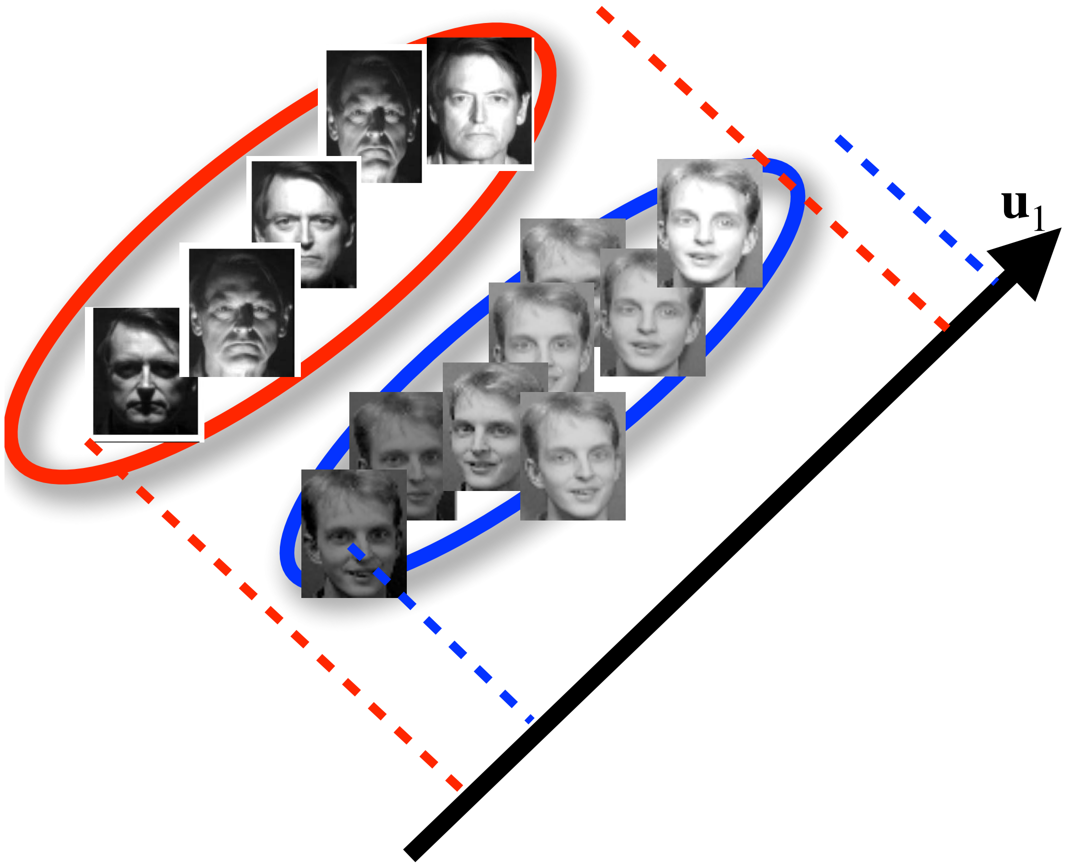


- Even better results using deep networks.
- But, much more complicated non-linear technique.
- We will talk return to this in the next lecture.



https://neurohive.io/en/news/deepfake-videos-gan-sythesizes-a-video-from-a-single-photo/

A Problem for EigenFaces

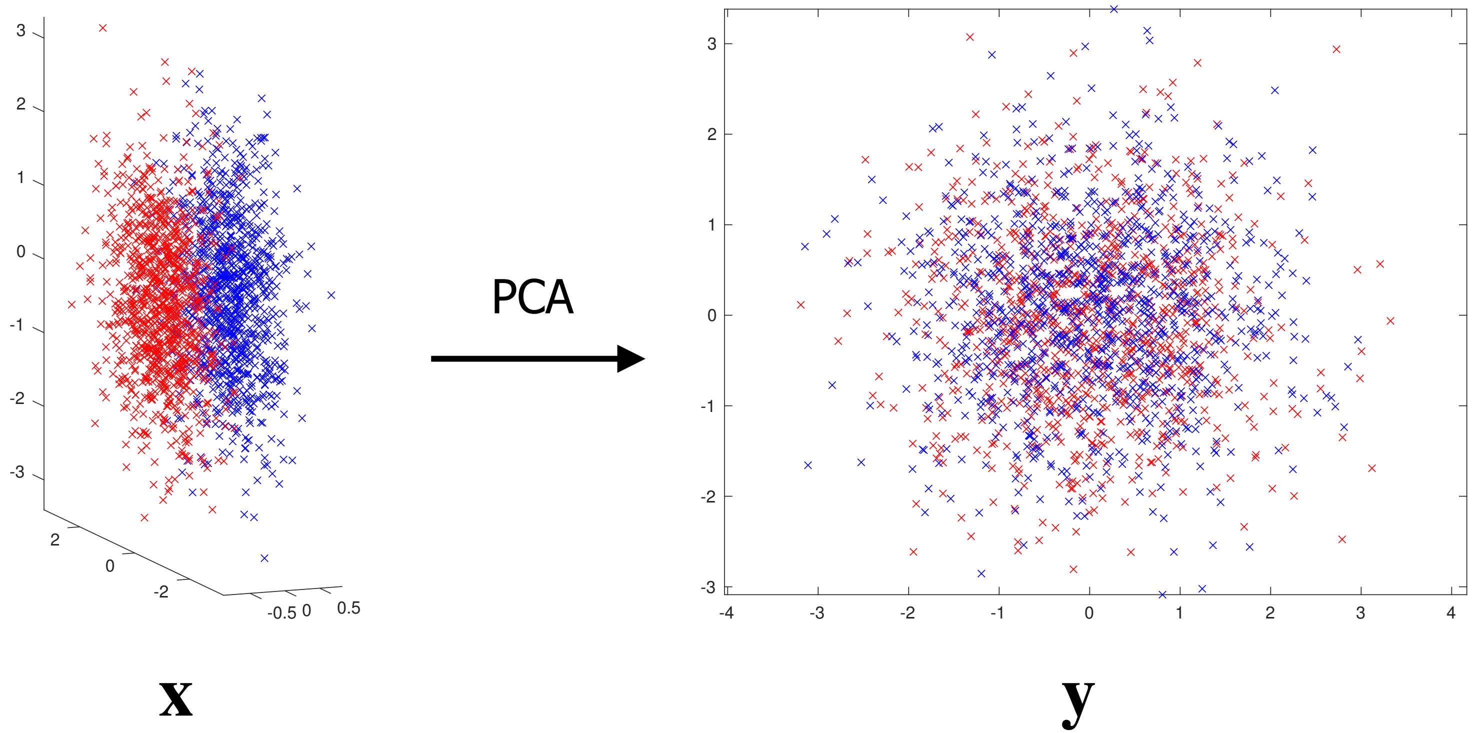


- Two different faces seen under very different illumination condition.
- The first eigenvector is very likely to capture differences in illumination.

—> Classes are not well separated.

Dimensionality Reduction for Classification

PCA is unsupervised and thus may not always preserve category information.



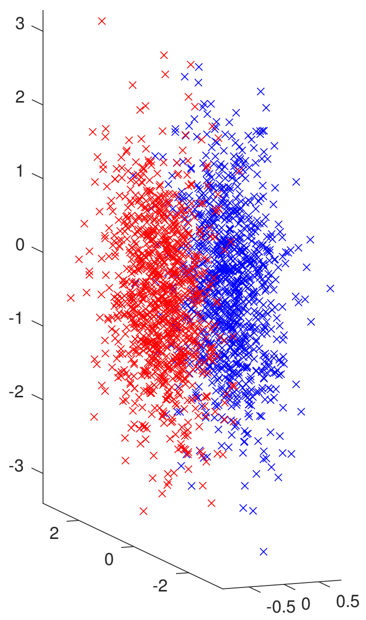
3D data from 2 classes (colors)

How about exploiting class labels during DR?

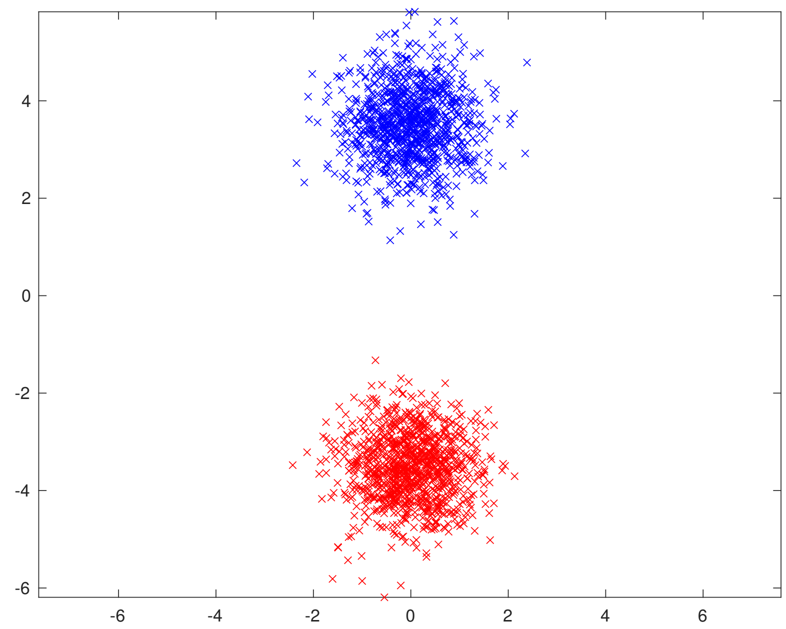
Fisher Linear Discriminant Analysis (LDA)

Ideally, we want:

- the samples from the same class to be clustered
- the different classes to be separated



x



y

Bishop, Chap. 4.1.6

Clustering Samples from the Same Class

- Mathematically, this means that we want a low variance within each class after projection
- For a 1D projection, encoded via a vector $\mathbf{w}_1$, and C classes, this can be expressed as aiming to minimize

$$E_W(\mathbf{w}_1) = \sum_{c=1}^{C} \sum_{i \in c} (y_i - \nu_c)^2$$

where ν_c is the mean of the samples in class c after projection, and $i \in c$ indicates that sample i belongs to class c.

Note that both the y_i and ν_c depend on $\mathbf{w}_1$.

Clustering Samples from the Same Class

- As in the PCA case, the variance after projection is equal to the projection of the covariance matrix
- This lets us rewrite the previous objective function as

$$E_W(\mathbf{w}_1) = \mathbf{w}_1^T \mathbf{S}_W \mathbf{w}_1,$$

where

$$\mathbf{S}_W = \sum_{c=1}^{C} \sum_{i \in c} (\mathbf{x}_i - \mu_c)(\mathbf{x}_i - \mu_c)^T,$$

and μ_c is the mean of the data in class c before projection.

- $\mathbf{S}_W$ is referred to as the within-class scatter matrix.

Separating the Different Classes

- In addition to clustering the samples according to the classes, we want to separate the different clusters
- This can be achieved by pushing the means of the clusters away from each other.
- Mathematically, this means maximizing

$$E_B(\mathbf{w}_1) = \sum_{c=1}^{C} N_c(\nu_c - \bar{y})^2,$$

where ν_c is defined as before, $\bar{y}$ is the mean of all samples after projection, and N_c is the number of samples in class c.

Separating the Different Classes

- Following the same reasoning as before, this can be re-written as

$$E_B(\mathbf{w}_1) = \mathbf{w}_1^T \mathbf{S}_B \mathbf{w}_1,$$

  where

$$\mathbf{S}_B = \sum_{c=1}^{C} N_c(\mu_c - \bar{\mathbf{x}})(\mu_c - \bar{\mathbf{x}})^T,$$

  $\bar{\mathbf{x}}$ is the mean of all the samples, and the $\{\mu_c\}$ are class-specific means.

- $\mathbf{S}_B$ is referred to as the between-class scatter matrix

Fisher LDA in Dimension 1

- We want to simultaneously
  - minimize $E_W(\mathbf{w}_1)$
  - maximize $E_B(\mathbf{w}_1)$
- This can be achieved by maximizing

$$J(\mathbf{w}_1) = \frac{E_B(\mathbf{w}_1)}{E_W(\mathbf{w}_1)} = \frac{\mathbf{w}_1^T \mathbf{S}_B \mathbf{w}_1}{\mathbf{w}_1^T \mathbf{S}_W \mathbf{w}_1},$$

because minimizing a function $f(\cdot)$ can be done by maximizing $1/f(\cdot)$, in general.

EPFL

Fisher LDA in Dimension 1

- The previous objective function is invariant to scaling:

$$J(\alpha \mathbf{w}_1) = J(\mathbf{w}_1)$$

- So we can fix the scale by constraining $\mathbf{w}_1$ to be such that

$$\mathbf{w}_1^T \mathbf{S}_W \mathbf{w}_1 = 1.$$

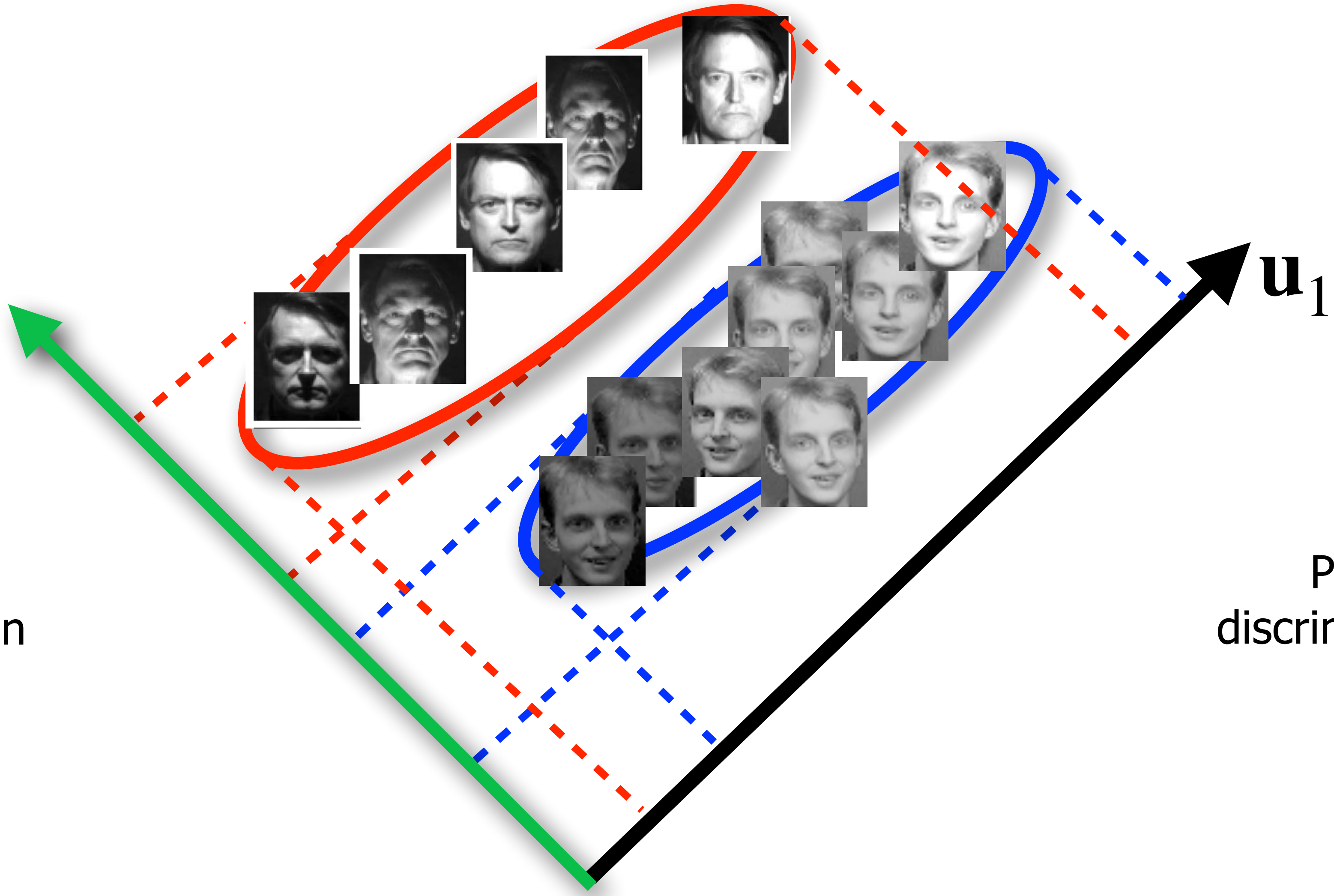
—> Fisher LDA formulation

$$\max_{\mathbf{w}_1} \mathbf{w}_1^T \mathbf{S}_B \mathbf{w}_1 \text{ subject to } \mathbf{w}_1^T \mathbf{S}_W \mathbf{w}_1 = 1.$$

—> $\mathbf{w}_1$ is the solution of a *generalized* eigenvalue problem.

PCA vs LDA

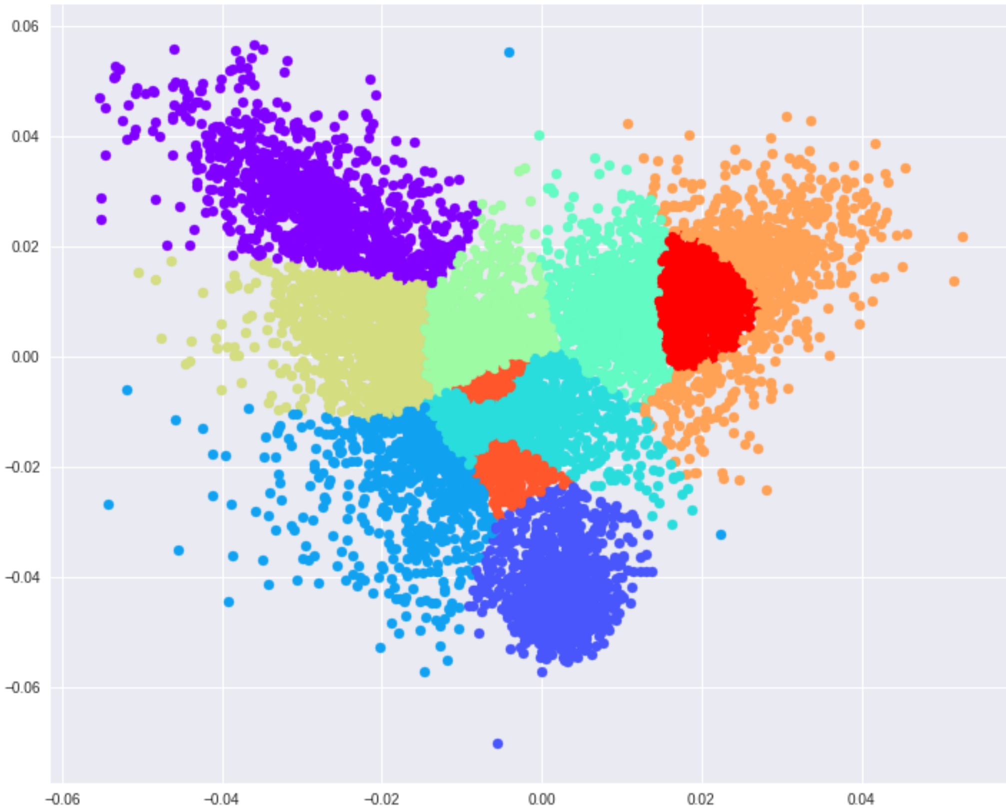
- PCA : Maximize projected variance.
- LDA : Maximise between class variance and minimize within class variance.



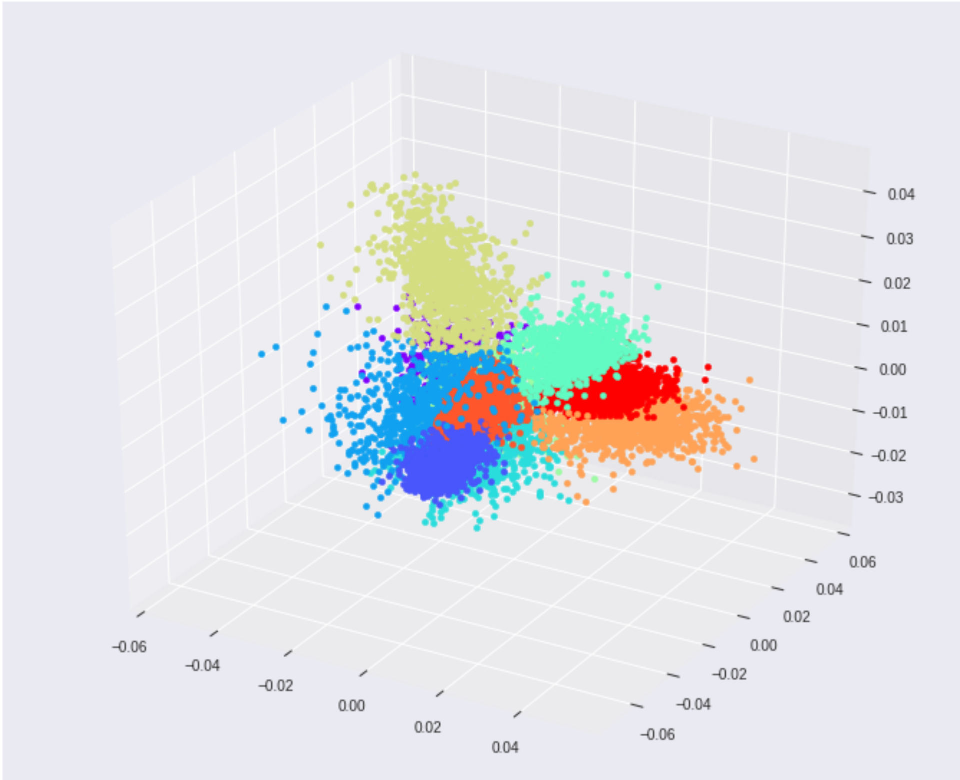
Good discrimination

Poor discrimination

Fisher LDA on MNIST



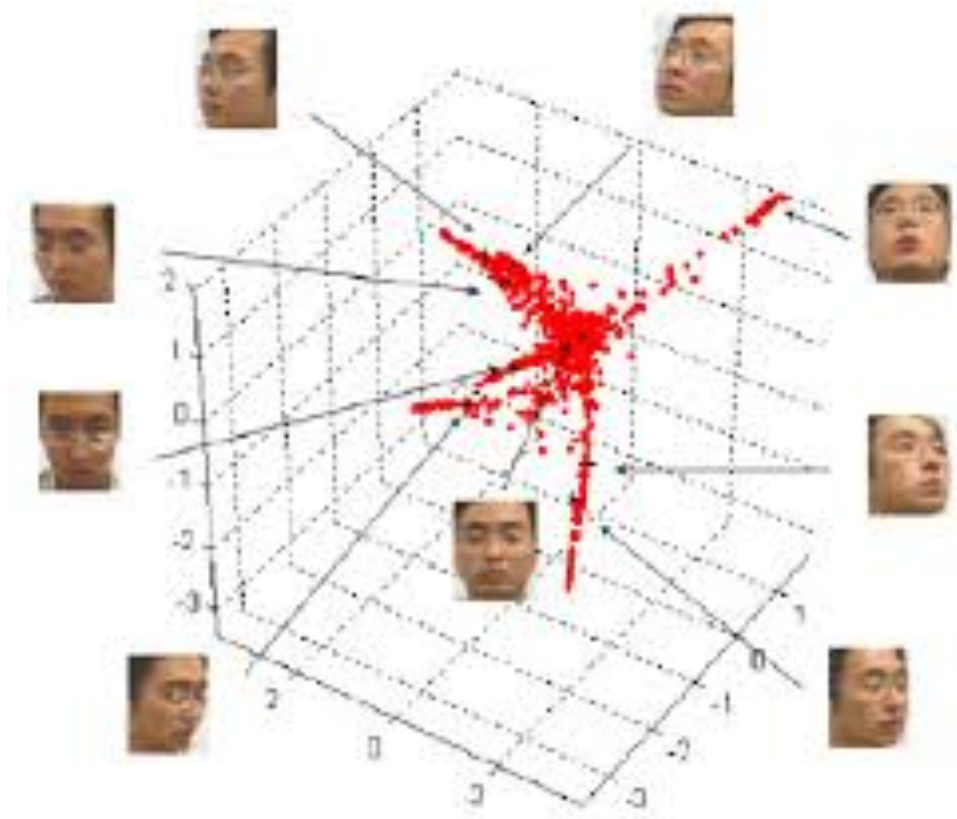
2D



3D

—> It only takes relatively low-dimensional spaces to yield decent clusters!

Reminder: Face Images



- The same can be said about face images.
- And of many other things.

—> Non linear classification is a practical proposition.

Fan et al. AVBPA'05

EigenFaces vs FisherFaces

- Consider a dataset of face images:
  - 2 different expressions.
  - several illumination conditions.

p1,ex1,il1 p1,ex1,il2 p1,ex1,il3 p1,ex2,il1 p1,ex2,il2 p1,ex2,il3

- One can apply either PCA or LDA to these images
  - The resulting eigenvectors can also be thought of as images.
  - They are called eigenfaces for PCA and fisherfaces for LDA.

Swets & Weng, PAMI'96

EigenFaces vs FisherFaces

EigenFaces

FisherFaces

- The EigenFaces contain information about the illumination and yield the best reconstructions.
- The FisherFaces discard the illumination information and are thus more useful for classification.

Linear vs NonLinear

- We could get better classification results with non-linear classifier.
- Is it also true of dimensionality reduction?

—> We will talk about this next.