Principal Component Analysis (PCA)

2IMW30 - Foundations of data mining
TU Eindhoven, Quartile 3, 2016-2017

Anne Driemel

Why reduce the dimension?

Representation of input data often is often high dimensional (images, documents, etc.)

There are two main reasons to reduce the dimension:

- some algorithms have **running time** exponential in the dimension
- we want to **visualize** inherent structure in the data

Why reduce the dimension?

Representation of input data often is often high dimensional (images, documents, etc.)

There are two main reasons to reduce the dimension:

- some algorithms have **running time** exponential in the dimension
- we want to **visualize** inherent structure in the data

Overview of this lecture

- Principal Component Analysis (PCA)
- Interpretation of Principle Components
- Computing Principal Components
- Singular-Value Decomposition (SVD)
- Power Method
- Eigenvectors of the Sample Covariance Matrix
- Multidimensional scaling
- Isomap

Principal Component Analysis (PCA)

Principal components provide a sequence of best linear approximations to a data set

Given a data set $P = \{\mathbf{p_1}, \ldots, \mathbf{p_n}\}$, we want to represent P using a k-dimensional linear model

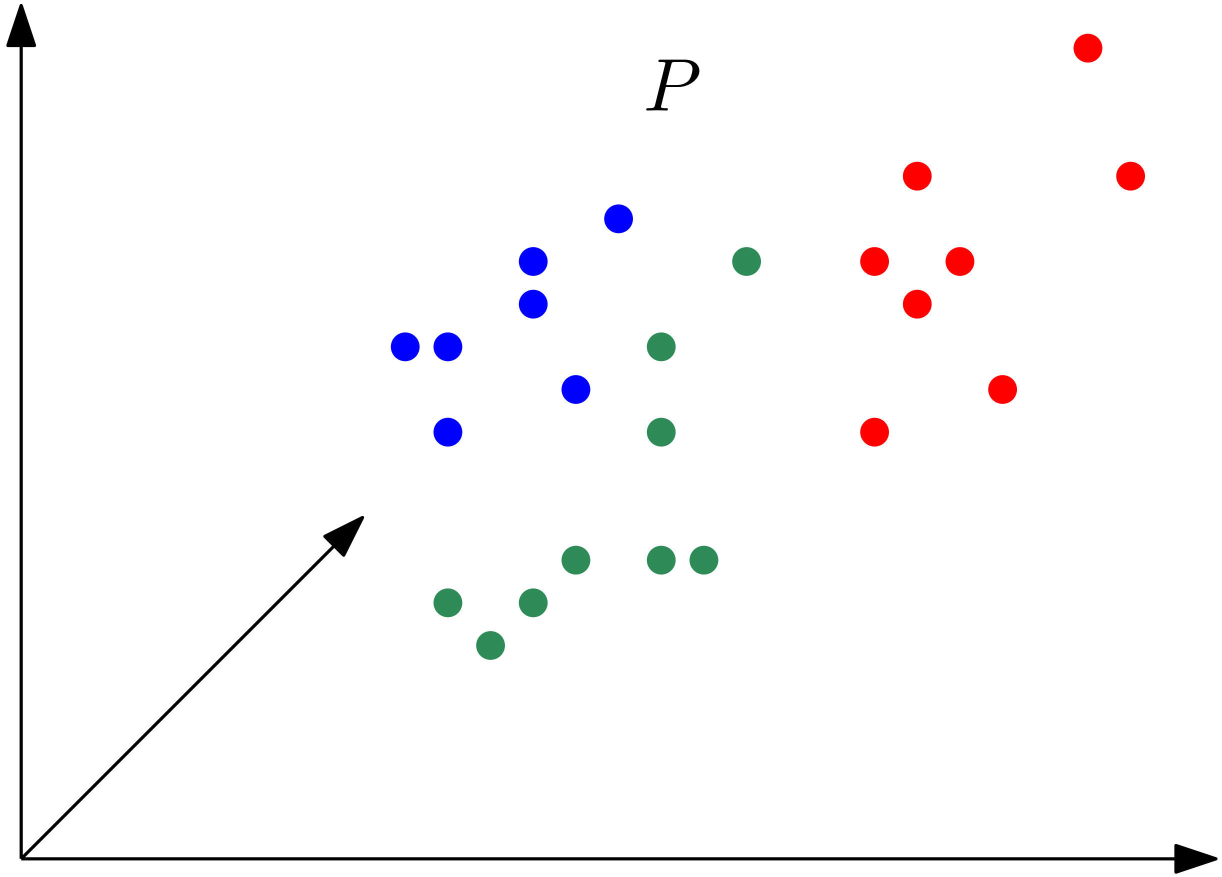
$$f(\lambda) = \mu + \mathbf{V}\lambda,$$

where

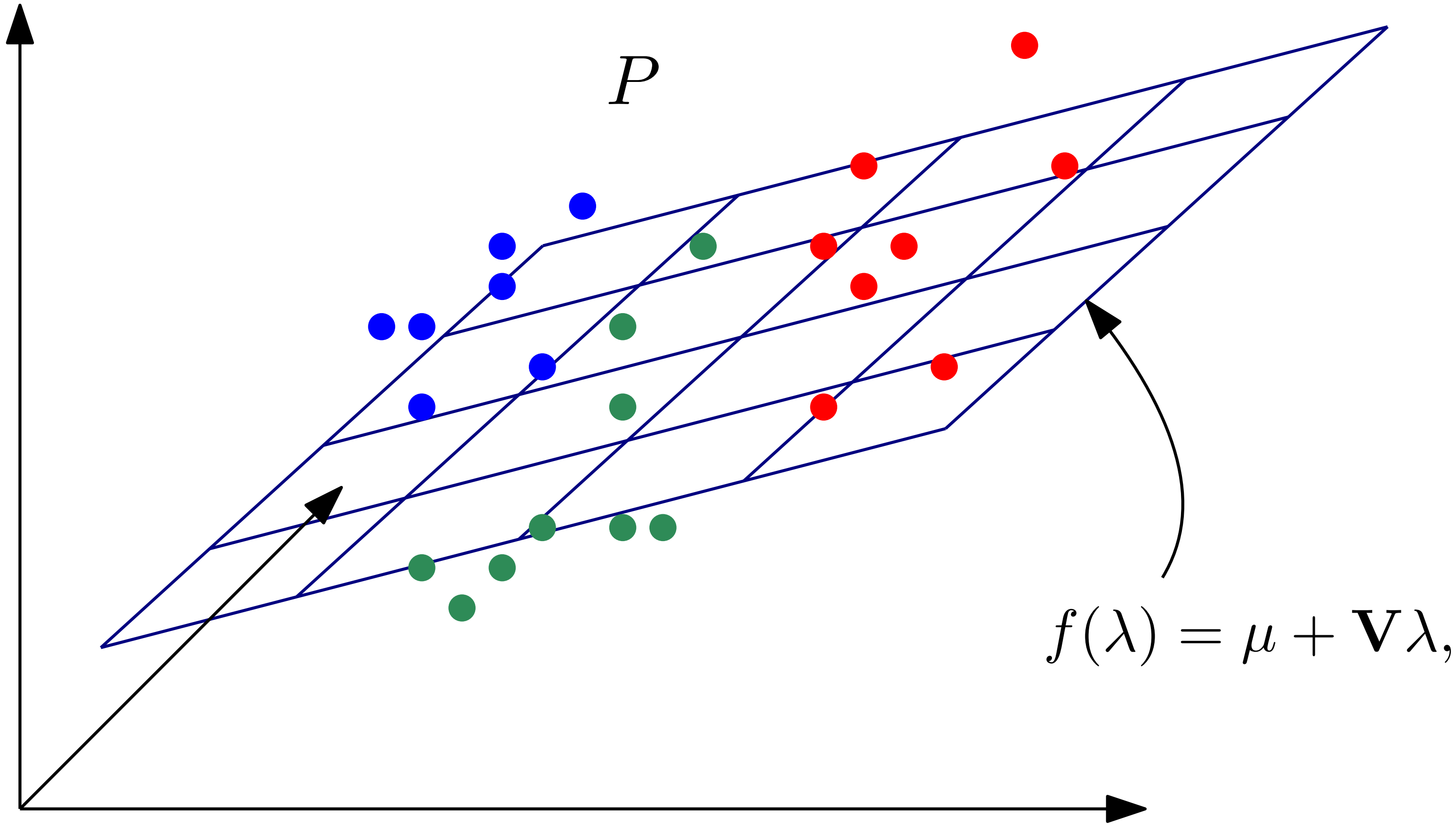
- μ is a location vector in $\mathbb{R}^d$
- $\mathbf{V}$ is a $d \times k$ orthonormal matrix
- λ is a k vector of parameters

The above is a parametric representation of an affine hyperplane of dimension k

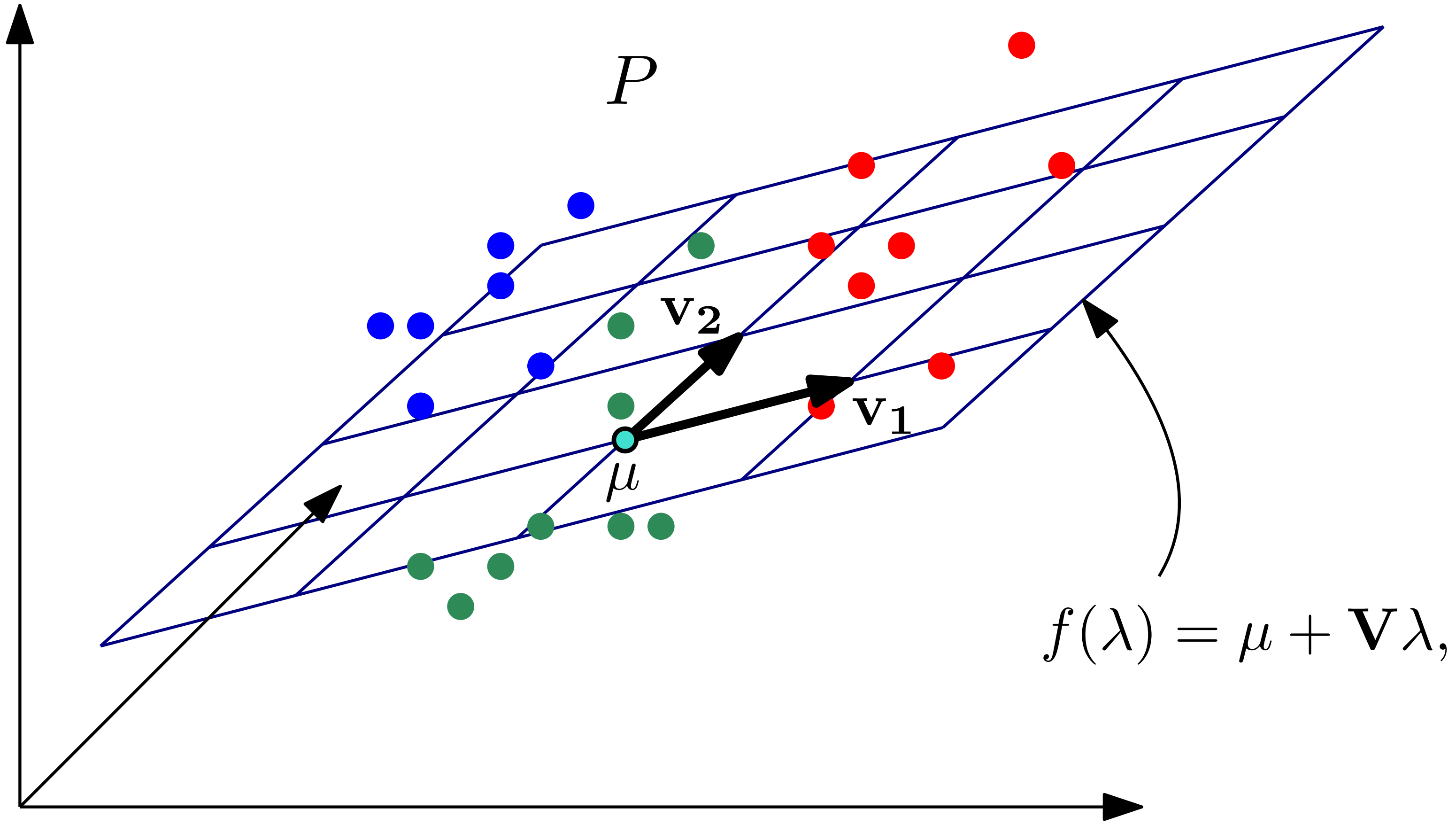
Principal Component Analysis (PCA)



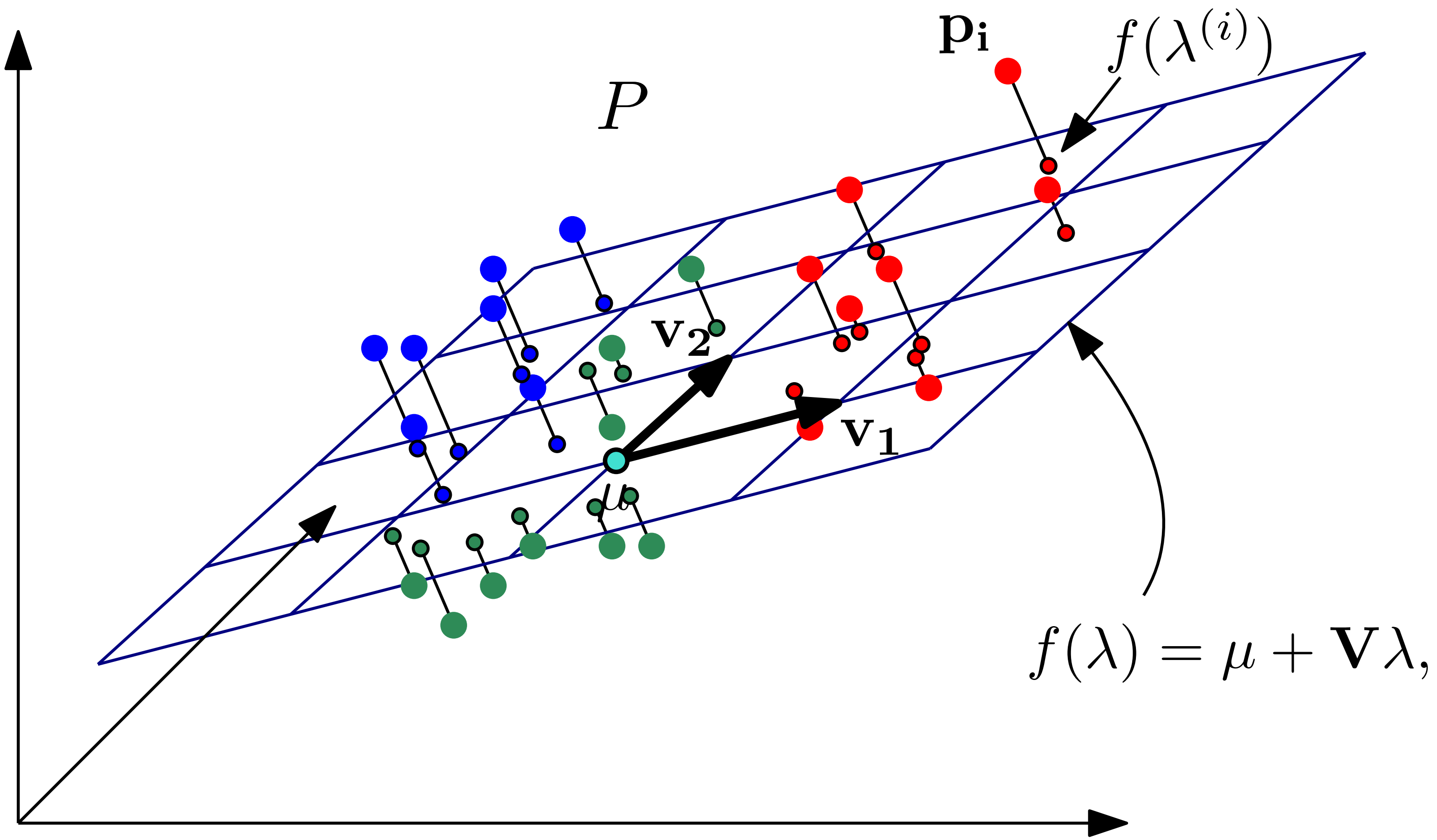
Principal Component Analysis (PCA)



Principal Component Analysis (PCA)



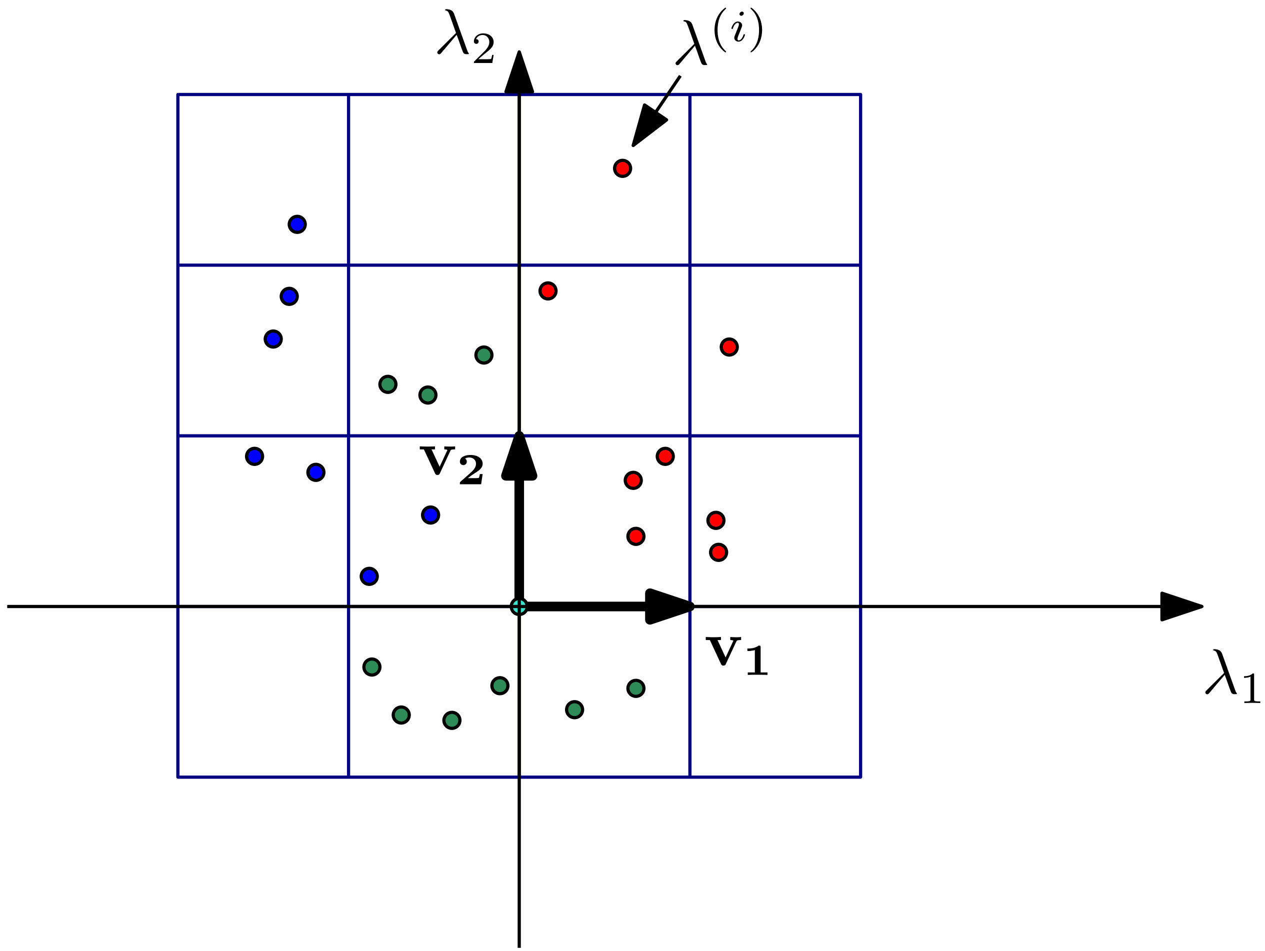
Principal Component Analysis (PCA)



Want to find the hyperplane which minimizes sum of squared distances ("best fitting") $\sum_{1 \leq i \leq n} \|\mathbf{p_i} - f(\lambda^{(i)})\|^2$

Principal Component Analysis (PCA)

We can visualize P in the subspace spanned by $\mathbf{v_1}$ and $\mathbf{v_2}$ by plotting the principle coordinates λ.



Principal Component Analysis (PCA)

We have our linear model

$$f(\lambda) = \mu + \mathbf{V}\lambda,$$

where

- μ is a location vector in $\mathbb{R}^d$
- $\mathbf{V}$ is a $d \times k$ matrix
- λ is a k vector of parameters

We have a function that defines "best fitting"

$$\min_{\mu, \mathbf{V_k}, \lambda} \sum_{1 \leq i \leq n} \|\mathbf{p_i} - f(\lambda^{(i)})\|^2$$

Principal Component Analysis (PCA)

We have our linear model

$$f(\lambda) = \mu + \mathbf{V}\lambda,$$

where
- μ is a location vector in $\mathbb{R}^d$
- $\mathbf{V}$ is a $d \times k$ matrix
- λ is a k vector of parameters

We have a function that defines "best fitting"

$$\min_{\mu, \mathbf{V_k}, \lambda} \sum_{1 \leq i \leq n} \|\mathbf{p_i} - f(\lambda^{(i)})\|^2$$

Optimizing for μ and λ gives

$$\mu = \frac{1}{n} \sum_{1 \leq i \leq n} \mathbf{p_i} \qquad \text{and} \qquad \lambda^{(i)} = \mathbf{V}^T(\mathbf{p_i} - \mu)$$

Principal Component Analysis (PCA)

We have our linear model

$$f(\lambda) = \mu + \mathbf{V}\lambda,$$

We can assume that μ is the mean of the data

where

- μ is a location vector in $\mathbb{R}^d$
- $\mathbf{V}$ is a $d \times k$ matrix
- λ is a k vector of parameters

We have a function that defines "best fitting"

$$\min_{\mu, \mathbf{V_k}, \lambda} \sum_{1 \leq i \leq n} \|\mathbf{p_i} - f(\lambda^{(i)})\|^2$$

Optimizing for μ and λ gives

$$\mu = \frac{1}{n} \sum_{1 \leq i \leq n} \mathbf{p_i} \quad \text{and} \quad \lambda^{(i)} = \mathbf{V}^T(\mathbf{p_i} - \mu)$$

Principal Component Analysis (PCA)

We have our linear model

$$f(\lambda) = \mu + \mathbf{V}\lambda,$$

where

- μ is a location vector in $\mathbb{R}^d$
- $\mathbf{V}$ is a $d \times k$ matrix
- λ is a k vector of parameters

We can assume that μ is the mean of the data

We have a function that defines "best fitting"

$$\min_{\mu, \mathbf{V_k}, \lambda} \sum_{1 \le i \le n} \|\mathbf{p_i} - f(\lambda^{(i)})\|^2$$

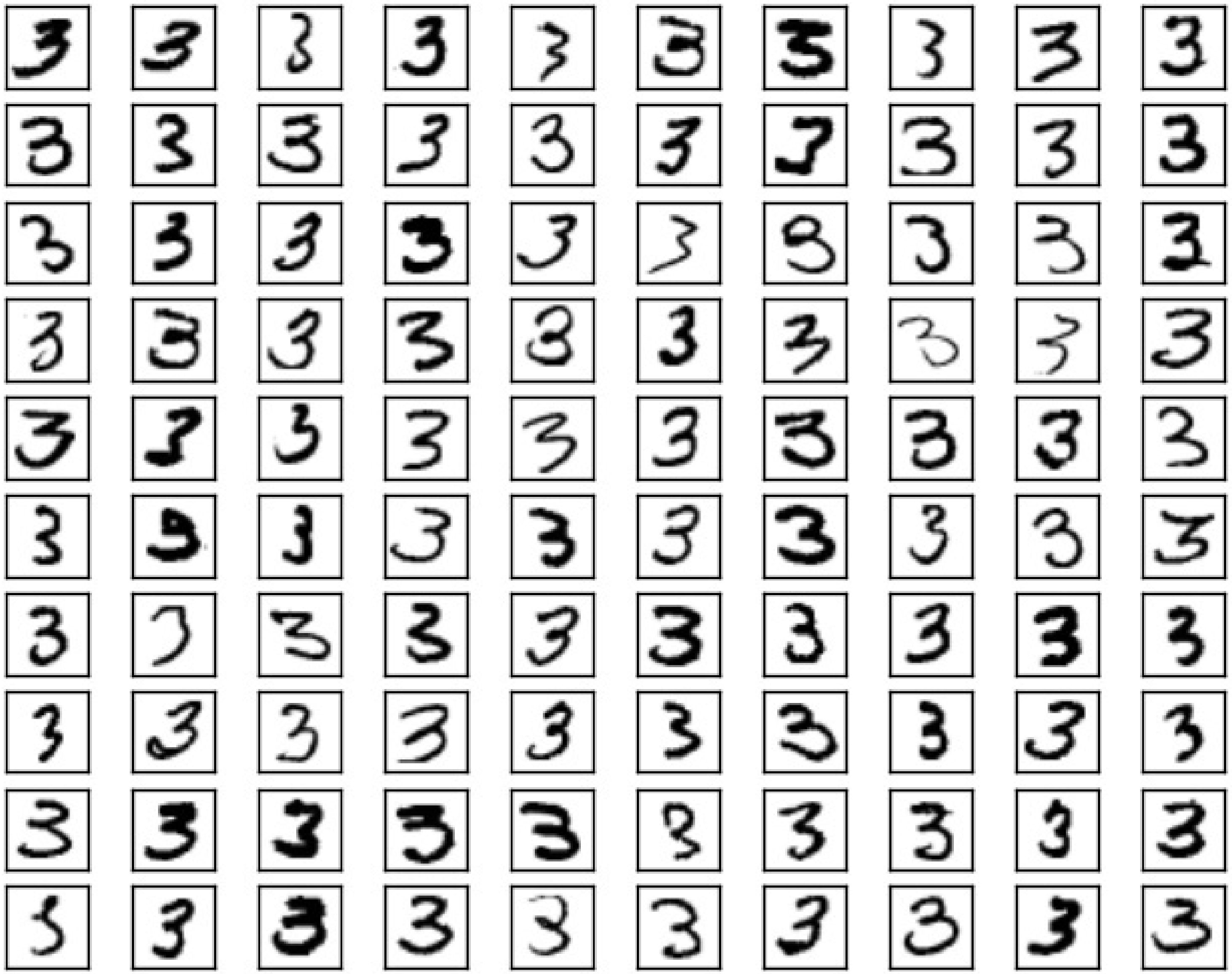
... and we use the projection onto $\mathbf{V}$ for λ

Optimizing for μ and λ gives

$$\mu = \frac{1}{n} \sum_{1 \le i \le n} \mathbf{p_i} \quad \text{and} \quad \lambda^{(i)} = \mathbf{V}^T(\mathbf{p_i} - \mu)$$

Principal Component Analysis (PCA)

Example: handwritten digits



Principal Component Analysis (PCA)

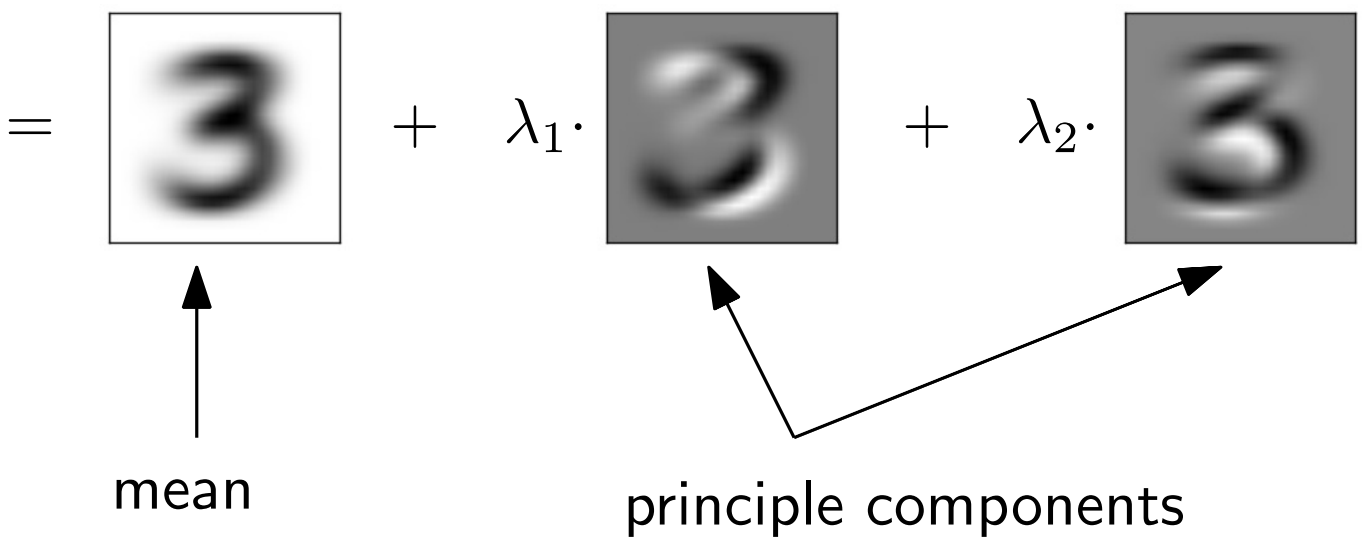
Example: handwritten digits

Assume we computed the first two principal components

We obtain an interpretable representation

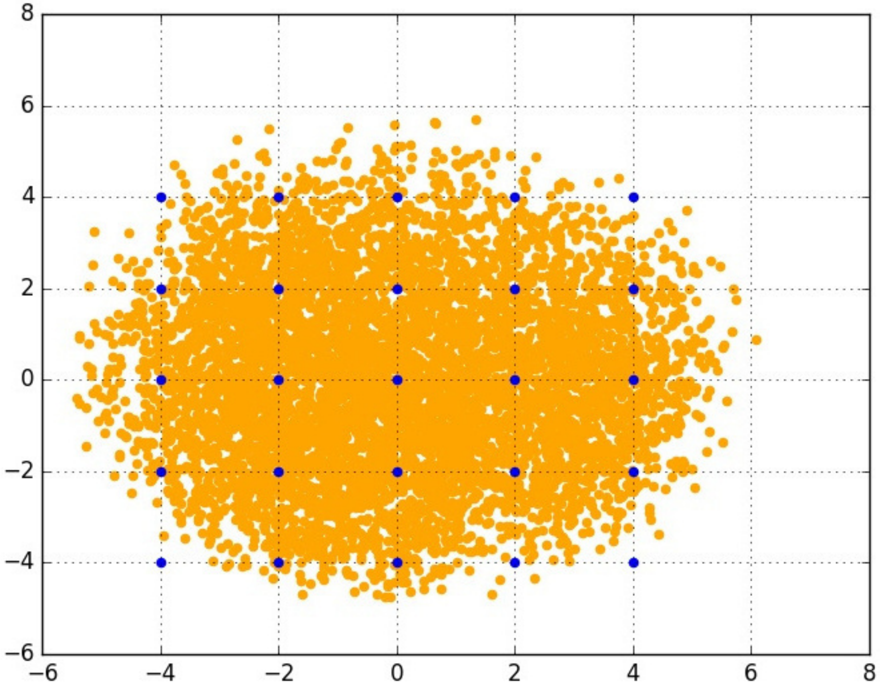
$$\widehat{f}(\lambda) = \mu + \mathbf{V}\lambda,$$

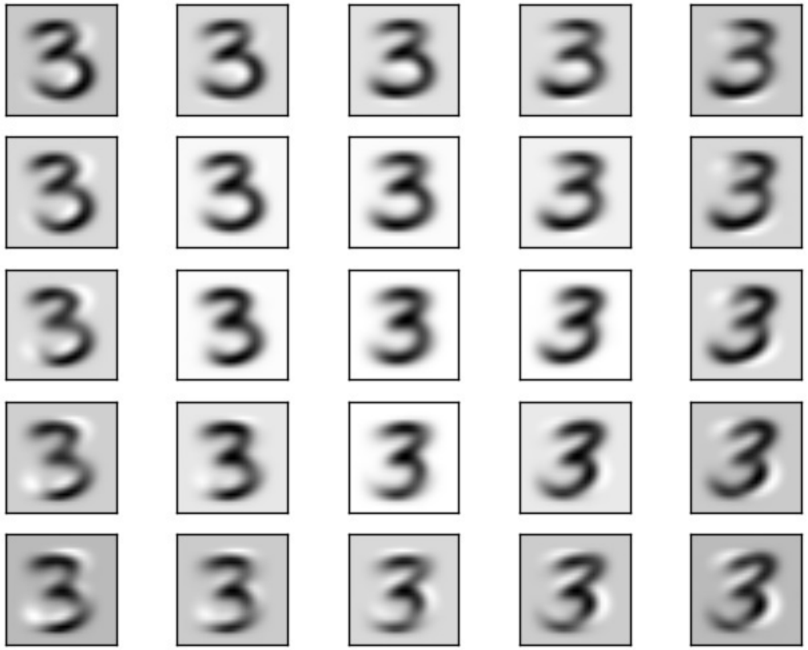
$$= \mu + \lambda_1 \mathbf{v_1} + \lambda_2 \mathbf{v_2}$$

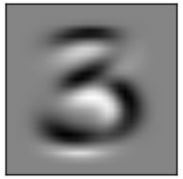


Principal Component Analysis (PCA)

Example: handwritten digits



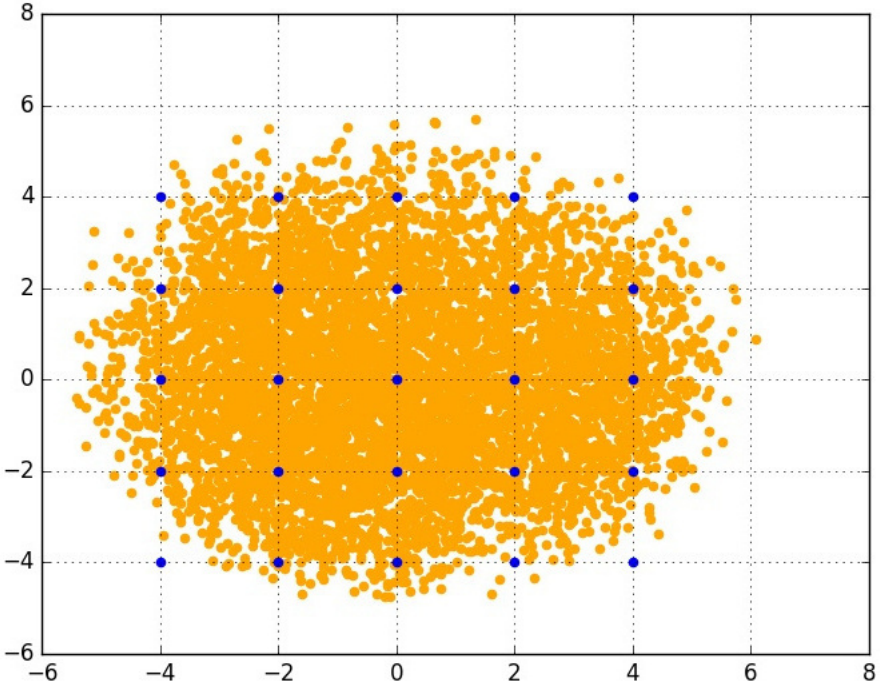


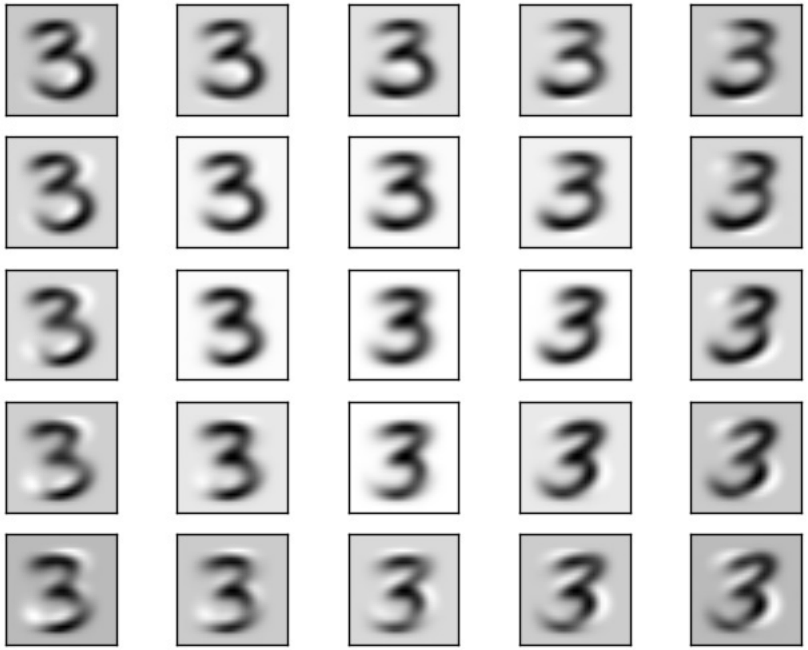
 $+ \quad \lambda_1 \cdot$  $+ \quad \lambda_2 \cdot$ 

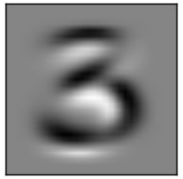
Interpretation?

Principal Component Analysis (PCA)

Example: handwritten digits





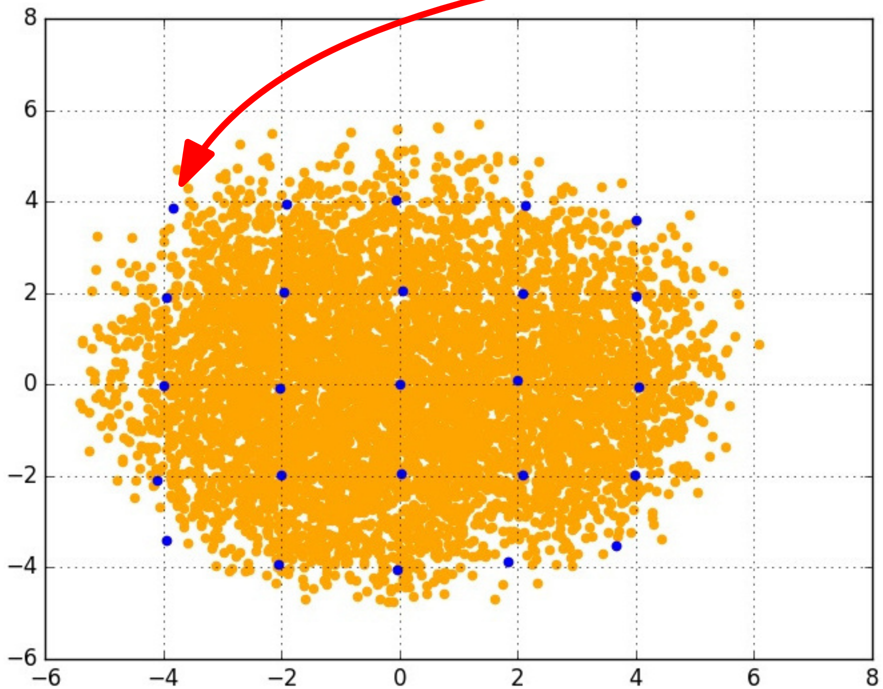
 $+ \quad \lambda_1 \cdot$  $+ \quad \lambda_2 \cdot$ 

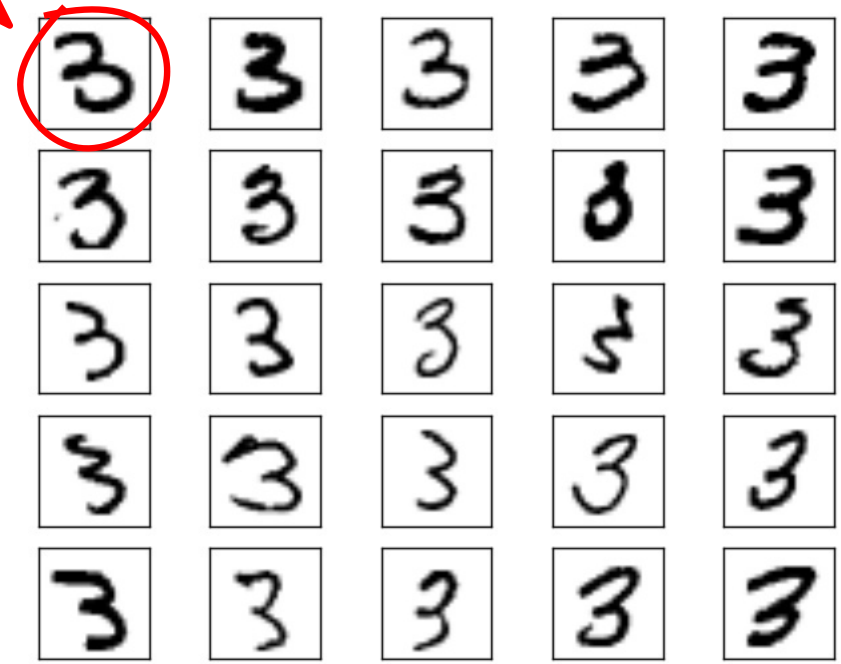
Interpretation? "slanting" "lengthening of lower tail"

Principal Component Analysis (PCA)

Example: handwritten digits

Instances of which the projections are closest to the grid points





 $+ \quad \lambda_1 \cdot$  $+ \quad \lambda_2 \cdot$ 

Interpretation? "slanting" "lengthening of lower tail"

Principal Component Analysis (PCA)

We have defined PCA as an optimization problem:
Fitting a k-dimensional hyperplane to the data

$$f(\lambda) = \mu + \mathbf{V}\lambda,$$

Principal Component Analysis (PCA)

We have defined PCA as an optimization problem:
Fitting a k-dimensional hyperplane to the data

$$f(\lambda) = \mu + \mathbf{V}\lambda,$$

How do we compute $\mathbf{V}$?

Principal Component Analysis (PCA)

We have defined PCA as an optimization problem:
Fitting a k-dimensional hyperplane to the data

$$f(\lambda) = \mu + \mathbf{V}\lambda,$$

How do we compute $\mathbf{V}$?

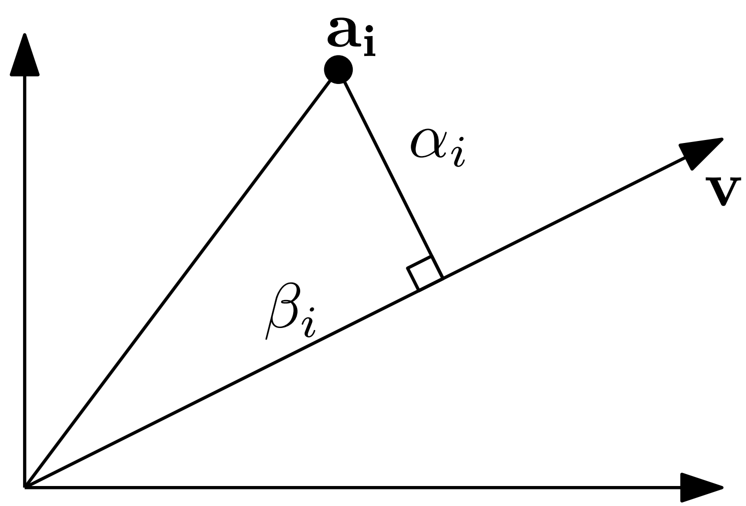
In the following, let $\mathbf{A}$ be a $n \times d$ matrix with row vectors $\mathbf{a_i}$ with

$$\mathbf{a_i} = \mathbf{p_i} - \mu$$

$\mathbf{A}$ is a **centered** version of P

Computing the principal components

Simplest case: fitting a line through the origin to $\mathbf{A}$

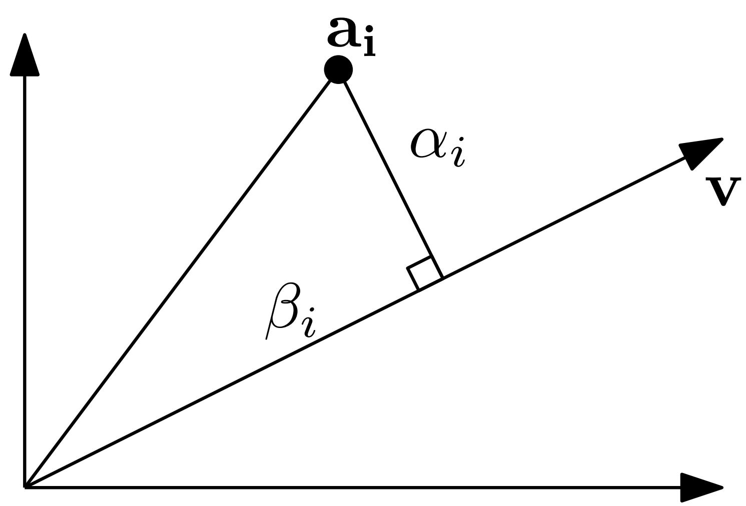


(Pythagoras)

$$\|\mathbf{a_i}\|^2 = \alpha_i^2 + \beta_i^2$$

Computing the principal components

Simplest case: fitting a line through the origin to $\mathbf{A}$



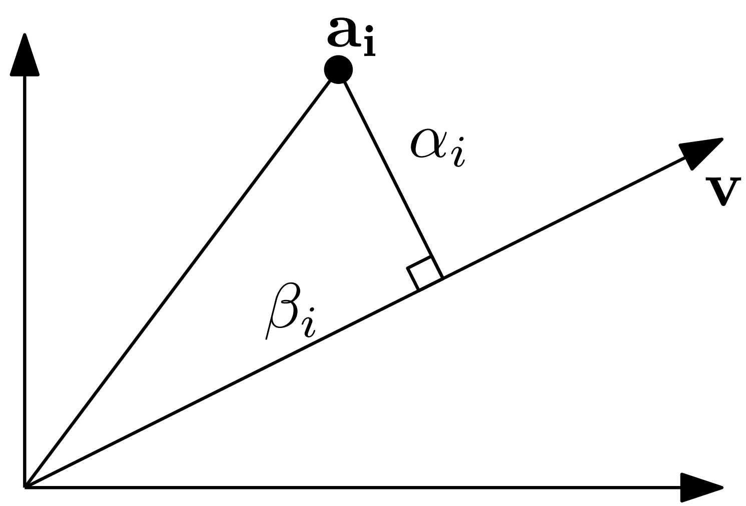
(Pythagoras)

$$\|\mathbf{a_i}\|^2 = \alpha_i^2 + \beta_i^2$$

$$\Leftrightarrow \quad \alpha_i^2 = \|\mathbf{a_i}\|^2 - \beta_i^2$$

Computing the principal components

Simplest case: fitting a line through the origin to $\mathbf{A}$



(Pythagoras)

$$\|\mathbf{a_i}\|^2 = \alpha_i^2 + \beta_i^2$$

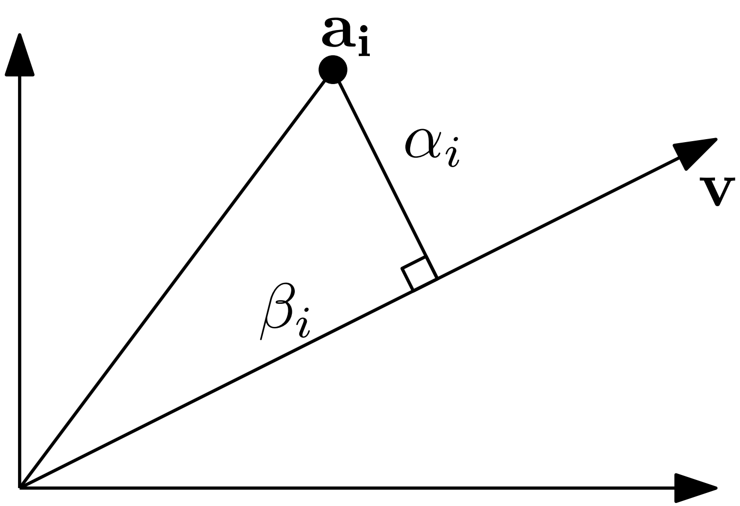
$$\Leftrightarrow \quad \alpha_i^2 = \|\mathbf{a_i}\|^2 - \beta_i^2$$

$$\underset{\|\mathbf{v}\|=1}{\operatorname{argmin}} \sum_{1 \le i \le n} \alpha_i^2 \quad = \quad \underset{\|\mathbf{v}\|=1}{\operatorname{argmin}} \sum_{1 \le i \le n} \|\mathbf{a_i}\|^2 - \alpha_i^2$$

"best fitting"

Computing the principal components

Simplest case: fitting a line through the origin to $\mathbf{A}$



(Pythagoras)

$$\|\mathbf{a_i}\|^2 = \alpha_i^2 + \beta_i^2$$

$$\Leftrightarrow \quad \alpha_i^2 = \|\mathbf{a_i}\|^2 - \beta_i^2$$

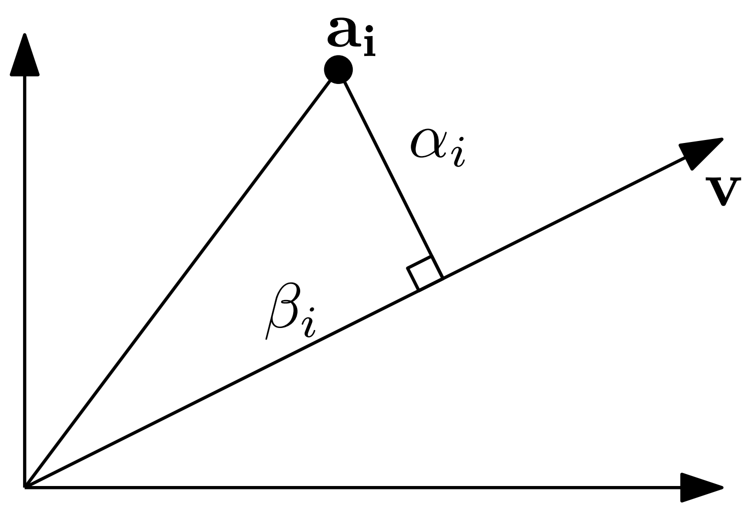
$$\underset{\|\mathbf{v}\|=1}{\operatorname{argmin}} \sum_{1 \le i \le n} \alpha_i^2 = \underset{\|\mathbf{v}\|=1}{\operatorname{argmin}} \sum_{1 \le i \le n} \|\mathbf{a_i}\|^2 - \alpha_i^2$$

"best fitting"

$$= \underset{\|\mathbf{v}\|=1}{\operatorname{argmax}} \sum_{1 \le i \le n} \beta_i^2$$

Computing the principal components

Simplest case: fitting a line through the origin to $\mathbf{A}$



(Pythagoras)

$$\|\mathbf{a_i}\|^2 = \alpha_i^2 + \beta_i^2$$

$$\Leftrightarrow \quad \alpha_i^2 = \|\mathbf{a_i}\|^2 - \beta_i^2$$

$$\operatorname*{argmin}_{\|\mathbf{v}\|=1} \sum_{1\leq i\leq n} \alpha_i^2 = \operatorname*{argmin}_{\|\mathbf{v}\|=1} \sum_{1\leq i\leq n} \|\mathbf{a_i}\|^2 - \alpha_i^2$$

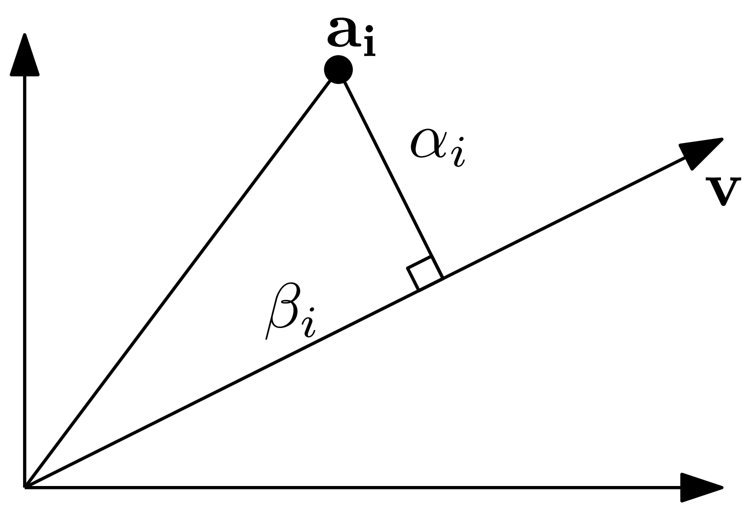
"best fitting"

$$= \operatorname*{argmax}_{\|\mathbf{v}\|=1} \sum_{1\leq i\leq n} \beta_i^2$$

$$= \operatorname*{argmax}_{\|\mathbf{v}\|=1} \sum_{1\leq i\leq n} (\mathbf{a_i}\mathbf{v})^2$$

Computing the principal components

Simplest case: fitting a line through the origin to $\mathbf{A}$



(Pythagoras)

$$\|\mathbf{a_i}\|^2 = \alpha_i^2 + \beta_i^2$$

$$\Leftrightarrow \quad \alpha_i^2 = \|\mathbf{a_i}\|^2 - \beta_i^2$$

$$\begin{aligned}
\operatorname*{argmin}_{\|\mathbf{v}\|=1} \sum_{1 \le i \le n} \alpha_i^2 &= \operatorname*{argmin}_{\|\mathbf{v}\|=1} \sum_{1 \le i \le n} \|\mathbf{a_i}\|^2 - \alpha_i^2 \\
&= \operatorname*{argmax}_{\|\mathbf{v}\|=1} \sum_{1 \le i \le n} \beta_i^2 \\
&= \operatorname*{argmax}_{\|\mathbf{v}\|=1} \sum_{1 \le i \le n} (\mathbf{a_i v})^2 = \operatorname*{argmax}_{\|\mathbf{v}\|=1} \|\mathbf{Av}\|^{\mathbf{2}}
\end{aligned}$$

"best fitting"

Computing the principal components

$\mathbf{A}$ is a $n \times d$ matrix with row vectors $\mathbf{a_i}$

The first singular vector of A is: $\mathbf{v_1} = \underset{\|\mathbf{v}\|=1}{\operatorname{argmax}} \|\mathbf{Av}\|$

The first singular value of A is: $\sigma_1 = \|\mathbf{Av_1}\|$

Computing the principal components

$\mathbf{A}$ is a $n \times d$ matrix with row vectors $\mathbf{a_i}$

The first singular vector of A is: $$\mathbf{v_1} = \operatorname*{argmax}_{\|\mathbf{v}\|=1} \|\mathbf{Av}\|$$

The first singular value of A is: $$\sigma_1 = \|\mathbf{Av_1}\|$$

The second singular vector of A is: $$\mathbf{v_2} = \operatorname*{argmax}_{\substack{\|\mathbf{v}\|=1 \\ \mathbf{v}\perp\mathbf{v_1}}} \|\mathbf{Av}\|$$

Computing the principal components

$\mathbf{A}$ is a $n \times d$ matrix with row vectors $\mathbf{a_i}$

The first singular vector of A is: $$\mathbf{v_1} = \underset{\|\mathbf{v}\|=1}{\operatorname{argmax}} \|\mathbf{Av}\|$$

The first singular value of A is: $$\sigma_1 = \|\mathbf{Av_1}\|$$

The second singular vector of A is: $$\mathbf{v_2} = \underset{\substack{\|\mathbf{v}\|=1 \\ \mathbf{v}\perp\mathbf{v_1}}}{\operatorname{argmax}} \|\mathbf{Av}\|$$

The second singular value of A is: $$\sigma_2 = \|\mathbf{Av_2}\|$$

Computing the principal components

$\mathbf{A}$ is a $n \times d$ matrix with row vectors $\mathbf{a_i}$

The first singular vector of A is: $$\mathbf{v_1} = \operatorname*{argmax}_{\|\mathbf{v}\|=1} \|\mathbf{Av}\|$$

The first singular value of A is: $$\sigma_1 = \|\mathbf{Av_1}\|$$

The second singular vector of A is: $$\mathbf{v_2} = \operatorname*{argmax}_{\substack{\|\mathbf{v}\|=1 \\ \mathbf{v}\perp\mathbf{v_1}}} \|\mathbf{Av}\|$$

The second singular value of A is: $$\sigma_2 = \|\mathbf{Av_2}\|$$

$\ldots$

Computing the principal components

$\mathbf{A}$ is a $n \times d$ matrix with row vectors $\mathbf{a_i}$

The first singular vector of A is: $$\mathbf{v_1} = \underset{\|\mathbf{v}\|=1}{\operatorname{argmax}} \|\mathbf{Av}\|$$

The first singular value of A is: $$\sigma_1 = \|\mathbf{Av_1}\|$$

The second singular vector of A is: $$\mathbf{v_2} = \underset{\substack{\|\mathbf{v}\|=1 \\ \mathbf{v}\perp\mathbf{v_1}}}{\operatorname{argmax}} \|\mathbf{Av}\|$$

The second singular value of A is: $$\sigma_2 = \|\mathbf{Av_2}\|$$

$\ldots$

The process stops when we have found singular vectors $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_r}$ and singular values $\sigma_1, \sigma_2, \ldots, \sigma_r$ and

$$\max_{\substack{\|\mathbf{v}\|=1 \\ \mathbf{v}\perp\mathbf{v_1},\mathbf{v_2},\ldots,\mathbf{v_r}}} \|\mathbf{Av}\| = \mathbf{0}$$

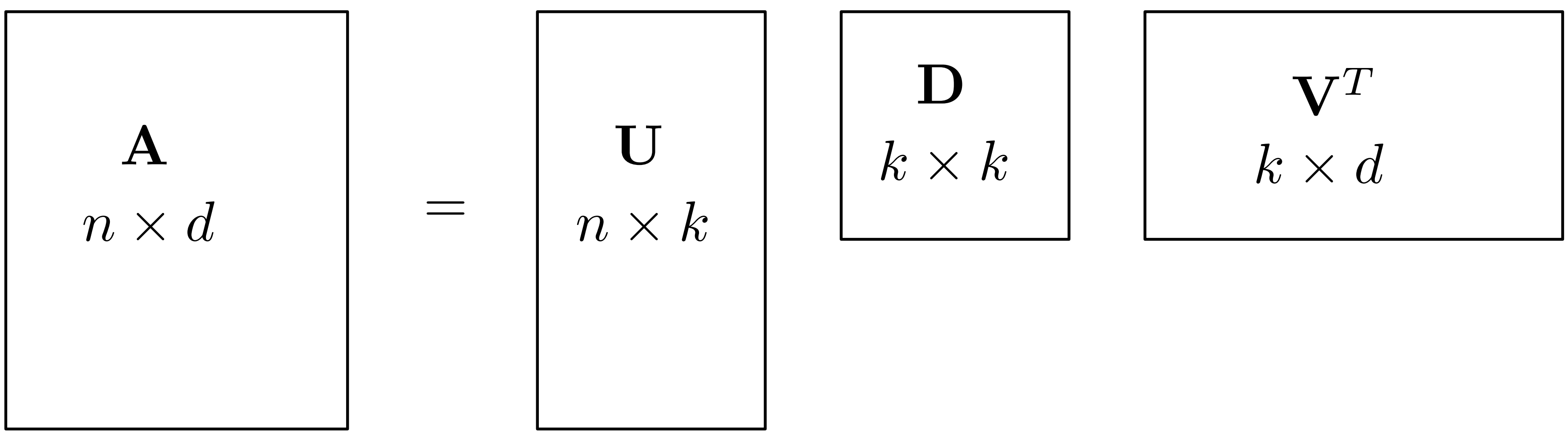
Singular Value Decomposition (SVD)

SVD is the factorization of a matrix A into three matrices

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T$$

where

- $\mathbf{U}$ and $\mathbf{V}$ are orthonormal
- $\mathbf{D}$ is diagonal with positive real entries σ_i
- σ_i are in descending order



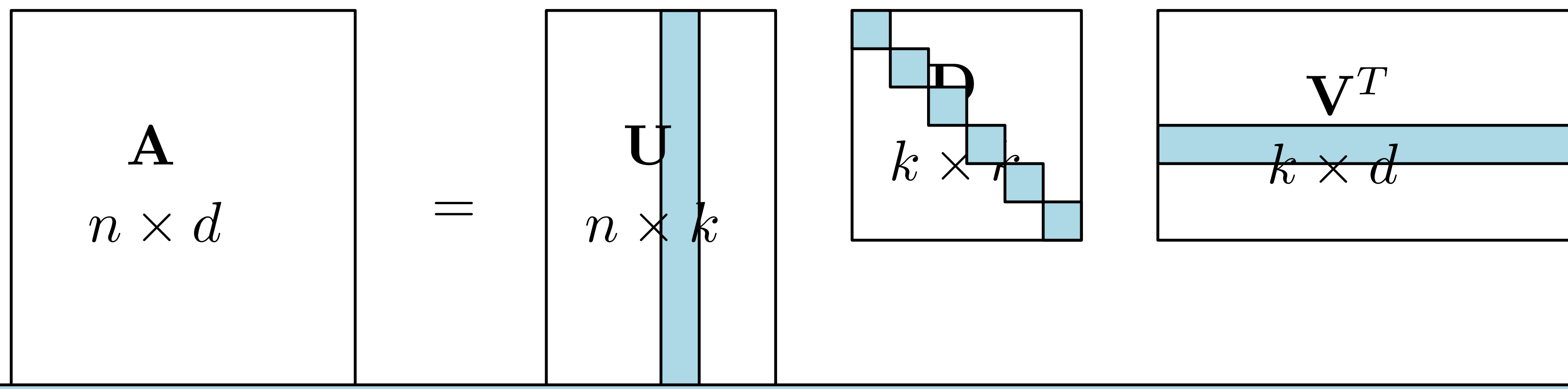
Singular Value Decomposition (SVD)

SVD is the factorization of a matrix A into three matrices

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T$$

where

- $\mathbf{U}$ and $\mathbf{V}$ are orthonormal
- $\mathbf{D}$ is diagonal with positive real entries σ_i
- σ_i are in descending order



Columns of $\mathbf{V}$ are called **singular vectors** $\mathbf{v_1}, \mathbf{v_2}, \ldots$
Diagonal entries of $\mathbf{D}$ are called **singular values** $\sigma_1, \sigma_2, \ldots$

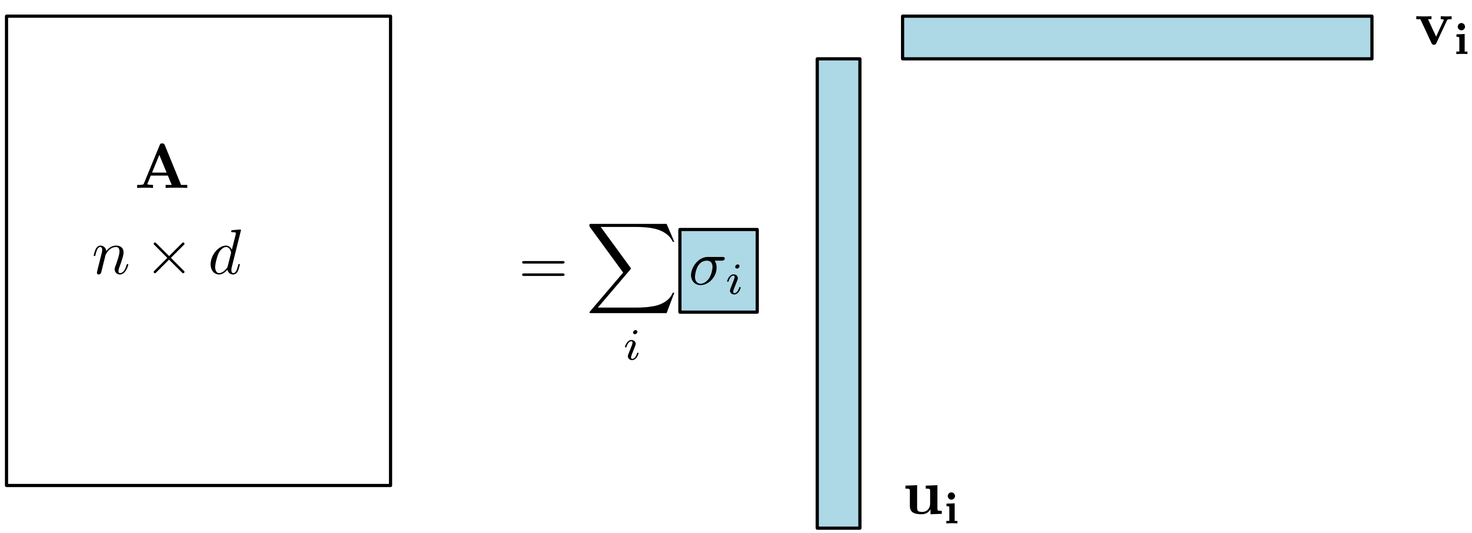
Singular Value Decomposition (SVD)

$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T$ can be rewritten using the sum of outer products

$$\mathbf{A} = \sum_i \sigma_i \mathbf{u_i} \mathbf{v_i}^T$$

where $\mathbf{u_i}$ and $\mathbf{v_i}$ are columns of $\mathbf{U}$ and $\mathbf{V}$

The i^{th} term in the above sum can be viewed as giving the components of the rows of $\mathbf{A}$ along $\mathbf{v_i}$



Power Method

The first principal component $\mathbf{v_1}$ can be computed using the **power method**:

$$\mathbf{B} = \mathbf{A}^T\mathbf{A} = \left(\sum_i \sigma_i \mathbf{u_i}\mathbf{v_i}^T\right)\left(\sum_j \sigma_j \mathbf{u_j}\mathbf{v_j}^T\right)$$

Power Method

The first principal component $\mathbf{v_1}$ can be computed using the **power method**:

$$\mathbf{B} = \mathbf{A}^T\mathbf{A} = \left(\sum_i \sigma_i \mathbf{u_i}\mathbf{v_i}^T\right)\left(\sum_j \sigma_j \mathbf{u_j}\mathbf{v_j}^T\right)$$

$$= \sum_i \sum_j \sigma_i \sigma_j \mathbf{v_i}(\mathbf{u_i}^T\mathbf{u_j})\mathbf{v_j}^T$$

orthogonal
for $i \neq j$

Power Method

The first principal component $\mathbf{v_1}$ can be computed using the **power method**:

$$\mathbf{B} = \mathbf{A}^T\mathbf{A} = \left(\sum_i \sigma_i \mathbf{u_i}\mathbf{v_i}^T\right)\left(\sum_j \sigma_j \mathbf{u_j}\mathbf{v_j}^T\right)$$

$$= \sum_i \sum_j \sigma_i \sigma_j \mathbf{v_i}(\mathbf{u_i}^T\mathbf{u_j})\mathbf{v_j}^T$$

orthogonal for $i \neq j$

$$= \sum_i \sigma_i^2 \mathbf{v_i}\mathbf{v_i}^T$$

Power Method

The first principal component $\mathbf{v_1}$ can be computed using the **power method**:

$$\mathbf{B} = \mathbf{A}^T\mathbf{A} = \left(\sum_i \sigma_i \mathbf{u_i}\mathbf{v_i}^T\right)\left(\sum_j \sigma_j \mathbf{u_j}\mathbf{v_j}^T\right)$$

$$= \sum_i \sum_j \sigma_i \sigma_j \mathbf{v_i}(\mathbf{u_i}^T\mathbf{u_j})\mathbf{v_j}^T$$

orthogonal for $i \neq j$

$$= \sum_i \sigma_i^2 \mathbf{v_i}\mathbf{v_i}^T$$

$$\mathbf{B}^2 = \sum_i \sum_j \sigma_i^2 \sigma_j^2 \mathbf{v_i}(\mathbf{v_i}^T\mathbf{v_j})\mathbf{v_j}^T = \sum_i \sigma_i^4 \mathbf{v_i}\mathbf{v_i}^T$$

Power Method

The first principal component $\mathbf{v_1}$ can be computed using the **power method**:

$$\mathbf{B} = \mathbf{A}^T\mathbf{A} = \left(\sum_i \sigma_i \mathbf{u_i}\mathbf{v_i}^T\right)\left(\sum_j \sigma_j \mathbf{u_j}\mathbf{v_j}^T\right)$$

$$= \sum_i \sum_j \sigma_i \sigma_j \mathbf{v_i}(\mathbf{u_i}^T\mathbf{u_j})\mathbf{v_j}^T$$

orthogonal for $i \neq j$

$$= \sum_i \sigma_i^2 \mathbf{v_i}\mathbf{v_i}^T$$

$$\mathbf{B}^2 = \sum_i \sum_j \sigma_i^2 \sigma_j^2 \mathbf{v_i}(\mathbf{v_i}^T\mathbf{v_j})\mathbf{v_j}^T = \sum_i \sigma_i^4 \mathbf{v_i}\mathbf{v_i}^T$$

$$\mathbf{B}^k = \sum_i \sigma_i^{2k} \mathbf{v_i}\mathbf{v_i}^T \to \sigma_1^{2k} \mathbf{v_1}\mathbf{v_1}^T$$

(using $\sigma_1 > \sigma_2$)

Power Method

The first principal component $\mathbf{v_1}$ can be computed using the **power method**:

$$\mathbf{B} = \mathbf{A}^T\mathbf{A} = \left(\sum_i \sigma_i \mathbf{u_i}\mathbf{v_i}^T\right)\left(\sum_j \sigma_j \mathbf{u_j}\mathbf{v_j}^T\right)$$

$$= \sum_i \sum_j \sigma_i \sigma_j \mathbf{v_i}(\mathbf{u_i}^T\mathbf{u_j})\mathbf{v_j}^T$$

orthogonal for $i \neq j$

$$= \sum_i \sigma_i^2 \mathbf{v_i}\mathbf{v_i}^T$$

$$\mathbf{B}^2 = \sum_i \sum_j \sigma_i^2 \sigma_j^2 \mathbf{v_i}(\mathbf{v_i}^T\mathbf{v_j})\mathbf{v_j}^T =$$

$$\mathbf{B}^k = \sum_i \sigma_i^{2k} \mathbf{v_i}\mathbf{v_i}^T \rightarrow \sigma_1^{2k}\mathbf{v_1}\mathbf{v_1}^T$$

(using $\sigma_1 > \sigma_2$)

We can estimate $\mathbf{v_1}$ using the first column of $\mathbf{B}^k$ normalized to unit length

Interpretation of principal components (again)

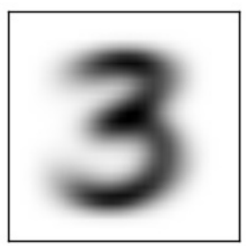
Example: handwritten digits

Assume we computed the first two principal components

We obtain an interpretable representation

$$\widehat{f}(\lambda) = \mu + \mathbf{V}\lambda,$$

$$= \mu + \lambda_1 \mathbf{v_1} + \lambda_2 \mathbf{v_2}$$

$=$  $+ \; \lambda_1 \cdot$  $+ \; \lambda_2 \cdot$ 

mean

principal components

An Alternative View

We can view $\mathbf{a_i}$ as an observation of a multivariate distribution

$\mathbf{A}$ contains n observations of d random variables $X_1, X_2, \ldots, X_d$

The **covariance** of two variables X_i, X_j is defined as

$$\mathrm{cov}(X_i, X_j) = \mathrm{E}\left[(X_i - \mu_i)(X_j - \mu_j)\right]$$

with $\mu_i = \mathrm{E}\left[X_i\right]$

The **sample covariance matrix** is defined as

$$\mathbf{M} = \frac{1}{n-1} \underbrace{\sum_{1 \leq i \leq n} (\mathbf{a_i} - \mu)^T (\mathbf{a_i} - \mu)}_{\mathbf{A}^T\mathbf{A}}$$

An Alternative View

A vector $\mathbf{v}$ such that

$$B\mathbf{v} = \gamma\ \mathbf{v}$$

is called an **eigenvector** of B and γ is called the **eigenvalue**

An Alternative View

A vector $\mathbf{v}$ such that

$$B\mathbf{v} = \gamma\ \mathbf{v}$$

is called an **eigenvector** of B and γ is called the **eigenvalue**

The following holds true since $\mathbf{V}^T = \mathbf{V}^{-1}$

$$\mathbf{A}\mathbf{v_i} = \sigma_i \mathbf{u_i} \qquad \text{and} \qquad \mathbf{A}^T\mathbf{u_i} = \sigma_i \mathbf{v_i}$$

together this implies

$$\mathbf{A}^T\mathbf{A}\mathbf{v_i} = \sigma_i^2 \mathbf{v_i}$$

Therefore, the **singular vectors** of $\mathbf{A}$ are the **eigenvectors** of the sample covariance matrix

Multidimensional scaling (Torgerson (1952))

Assume matrix $\mathbf{A}$ is not available, but instead we are given all squared pairwise distances as $n \times n$ matrix Δ

$$\Delta_{ij} = \|\mathbf{a_i} - \mathbf{a_j}\|^2$$

Multidimensional scaling (Torgerson (1952))

Assume matrix $\mathbf{A}$ is not available, but instead we are given all squared pairwise distances as $n \times n$ matrix Δ

$$\Delta_{ij} = \|\mathbf{a_i} - \mathbf{a_j}\|^2$$

We can recover inner products $\mathbf{a_i}\mathbf{a_j^T}$ of unknown $\mathbf{A}$ as follows

Multidimensional scaling (Torgerson (1952))

Assume matrix $\mathbf{A}$ is not available, but instead we are given all squared pairwise distances as $n \times n$ matrix Δ

$$\Delta_{ij} = \|\mathbf{a_i} - \mathbf{a_j}\|^2$$

We can recover inner products $\mathbf{a_i}\mathbf{a_j^T}$ of unknown $\mathbf{A}$ as follows

The following matrix is a **double-centering** of Δ

$$\mathbf{B} = \left(\mathbf{I} - \frac{\mathbf{J}}{n}\right) \Delta \left(\mathbf{I} - \frac{\mathbf{J}}{n}\right)$$

where

- $\mathbf{I}$ denotes the $n \times n$ identity matrix
- $\mathbf{J}$ be the $n \times n$ matrix of all $\mathbf{1}$'s

Multidimensional scaling (Torgerson (1952))

Assume matrix $\mathbf{A}$ is not available, but instead we are given all squared pairwise distances as $n \times n$ matrix Δ

$$\Delta_{ij} = \|\mathbf{a_i} - \mathbf{a_j}\|^2$$

We can recover inner products $\mathbf{a_i}\mathbf{a_j^T}$ of unknown $\mathbf{A}$ as follows

The following matrix is a **double-centering** of Δ

$$\mathbf{B} = \left(\mathbf{I} - \frac{\mathbf{J}}{n}\right) \Delta \left(\mathbf{I} - \frac{\mathbf{J}}{n}\right)$$

centering the rows of Δ

where

- $\mathbf{I}$ denotes the $n \times n$ identity matrix
- $\mathbf{J}$ be the $n \times n$ matrix of all $\mathbf{1}$'s

Multidimensional scaling (Torgerson (1952))

Assume matrix $\mathbf{A}$ is not available, but instead we are given all squared pairwise distances as $n \times n$ matrix Δ

$$\Delta_{ij} = \|\mathbf{a_i} - \mathbf{a_j}\|^2$$

We can recover inner products $\mathbf{a_i a_j^T}$ of unknown $\mathbf{A}$ as follows

The following matrix is a **double-centering** of Δ

$$\mathbf{B} = \left(\mathbf{I} - \frac{\mathbf{J}}{n}\right) \Delta \left(\mathbf{I} - \frac{\mathbf{J}}{n}\right)$$

centering the rows of Δ

where

- $\mathbf{I}$ denotes the $n \times n$ identity matrix
- $\mathbf{J}$ be the $n \times n$ matrix of all $\mathbf{1}$'s

centering the columns of Δ

Multidimensional scaling (Torgerson (1952))

Assume matrix $\mathbf{A}$ is not available, but instead we are given all squared pairwise distances as $n \times n$ matrix Δ

$$\Delta_{ij} = \|\mathbf{a_i} - \mathbf{a_j}\|^2$$

We can recover inner products $\mathbf{a_i a_j^T}$ of unknown $\mathbf{A}$ as follows

The following matrix is a **double-centering** of Δ

$$\mathbf{B} = \left(\mathbf{I} - \frac{\mathbf{J}}{n}\right)\Delta\left(\mathbf{I} - \frac{\mathbf{J}}{n}\right)$$

centering the rows of Δ

where

- $\mathbf{I}$ denotes the $n \times n$ identity matrix
- $\mathbf{J}$ be the $n \times n$ matrix of all $\mathbf{1}$'s

centering the columns of Δ

If $\mathbf{A}$ is mean-centered, one can show that $(-\frac{1}{2})\mathbf{B} = \mathbf{AA^T}$

Multidimensional scaling (Torgerson (1952))

Recall that from SVD we have

$$\mathbf{A}\mathbf{v_i} = \sigma_i \mathbf{u_i} \qquad \text{and} \qquad \mathbf{A}^T\mathbf{u_i} = \sigma_i \mathbf{v_i}$$

which implies

$$\mathbf{A}^T\mathbf{A}\mathbf{v_i} = \sigma_i^2 \mathbf{v_i}$$

Multidimensional scaling (Torgerson (1952))

Recall that from SVD we have

$$\mathbf{A}\mathbf{v_i} = \sigma_i \mathbf{u_i} \qquad \text{and} \qquad \mathbf{A}^T\mathbf{u_i} = \sigma_i \mathbf{v_i}$$

which implies

$$\mathbf{A}^T\mathbf{A}\mathbf{v_i} = \sigma_i^2 \mathbf{v_i}$$

symmetrically, this also implies

$$\mathbf{A}\mathbf{A}^T\mathbf{u_i} = \sigma_i^2 \mathbf{u_i}$$

Multidimensional scaling (Torgerson (1952))

Recall that from SVD we have

$$\mathbf{A}\mathbf{v_i} = \sigma_i \mathbf{u_i} \qquad \text{and} \qquad \mathbf{A}^T\mathbf{u_i} = \sigma_i \mathbf{v_i}$$

which implies

$$\mathbf{A}^T\mathbf{A}\mathbf{v_i} = \sigma_i^2 \mathbf{v_i}$$

symmetrically, this also implies

$$\mathbf{A}\mathbf{A}^T\mathbf{u_i} = \sigma_i^2 \mathbf{u_i}$$

Thus, the eigenvectors of $\mathbf{AA^T}$ are the vectors $\mathbf{u_i}$ of the SVD of $\mathbf{A}$

Multidimensional scaling (Torgerson (1952))

Recall that from SVD we have

$$\mathbf{A}\mathbf{v_i} = \sigma_i \mathbf{u_i} \qquad \text{and} \qquad \mathbf{A}^T\mathbf{u_i} = \sigma_i \mathbf{v_i}$$

which implies

$$\mathbf{A}^T\mathbf{A}\mathbf{v_i} = \sigma_i^2 \mathbf{v_i}$$

symmetrically, this also implies

$$\mathbf{A}\mathbf{A}^T\mathbf{u_i} = \sigma_i^2 \mathbf{u_i}$$

Thus, the eigenvectors of $\mathbf{A}\mathbf{A}^\mathbf{T}$ are the vectors $\mathbf{u_i}$ of the SVD of $\mathbf{A}$

Using this, we can compute the SVD of $\mathbf{A}$ and perfom PCA

Multidimensional scaling (Torgerson (1952))

Recall that from SVD we have

$$\mathbf{A}\mathbf{v_i} = \sigma_i \mathbf{u_i} \qquad \text{and} \qquad \mathbf{A}^T\mathbf{u_i} = \sigma_i \mathbf{v_i}$$

which implies

$$\mathbf{A}^T\mathbf{A}\mathbf{v_i} = \sigma_i^2 \mathbf{v_i}$$

symmetrically, this also implies

$$\mathbf{A}\mathbf{A}^T\mathbf{u_i} = \sigma_i^2 \mathbf{u_i}$$

Thus, the eigenvectors of $\mathbf{A}\mathbf{A^T}$ are the vectors $\mathbf{u_i}$ of the SVD of $\mathbf{A}$

Using this, we can compute the SVD of $\mathbf{A}$ and perfom PCA

The result is called an **embedding** of $\mathbf{A}$ and the process is called classical multidimensional scaling (MDS).

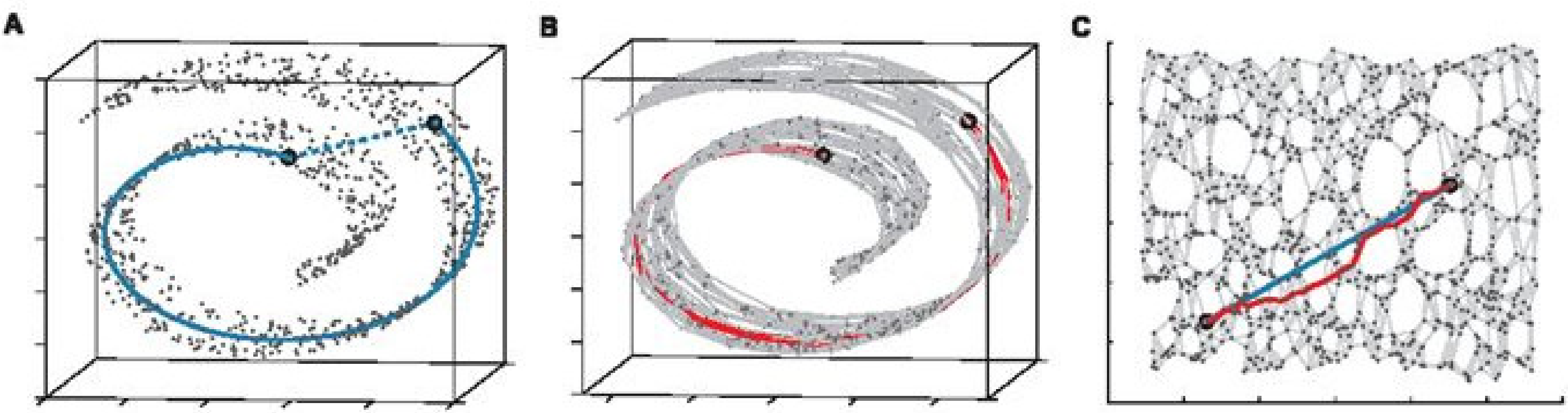
Isomap

Isomap is a non-linear embedding algorithm which assumes that the data lies on an Euclidean manifold

Isomap is due to Tenenbaum, Silva and Langford (2000)

Algorithm:

- Compute the k-nearest neighbor graph G
- Compute all pairwise shortest paths in G
- Use Multidimensional scaling on the obtained distances



Summary

- Principal Component Analysis (PCA)
- Interpretation of Principal Components
- Computing Principal Components
- Singular-Value Decomposition (SVD)
- Power Method
- Eigenvectors of the Sample Covariance Matrix
- Multidimensional scaling
- Isomap

References

- Avrim Blum, John Hopcroft, Ravindran Khannan: *Foundations of Data Science*
- Trevor Hastie, Robert Tibshirani, Jerome Friedman: *Elements of Statistical Learning*
- J. B. Tenenbaum, V. de Silva, J. C. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction", *Science* 290, (2000).