DAM lecture 8:

Dimensionality reduction 2
03.03.2016

Aasa Feragen
aasa@diku.dk

Assignment 1 feedback

- Feedback is now available
- We have been strict on presentation – most of you will have an easy update
- Use the TA sessions!
- Questions?

After today's lecture you should

- be familiar with the equivalent definitions of PCA by least squares projection error minimization, projected variance maximization, low distortion embedding, and eigenvalue decomposition of the covariance matrix
- be able to interpret the different equivalent PCA definitions and use them to pinpoint strengths and weaknesses of PCA
- be able to use PCA for visualization of global dataset variation
- be familiar with the curse of dimensionality and the need for dimensionality reduction
- be familiar with partial derivatives, gradients, and their use for finding principal components (if time allows)

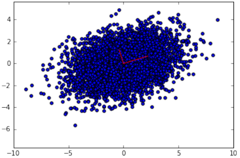
Literature for today's lecture

- Chapters 4 and 10
- **Shlens tutorial:**
  Optional; fantastic intro to PCA with Matlab code
  (find it on Absalon)

Recall from Lecture 6: Covariance matrix

- For a sampled dataset $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\} \subset \mathbb{R}^d$, we can define its $d \times d$ *covariance matrix* Σ by setting
$$\Sigma_{i,j} = cov(x_i, x_j).$$
- The variance of each coordinate is found along the diagonal: $s_{x_i}^2 = \lambda_i$

Recall from Lecture 6: Decomposing the covariance matrix



Theorem (Eigenvalue decomposition)

If Σ is a $d \times d$ matrix with linearly independent eigenvectors $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_d$, with corresponding eigenvalues $\lambda_1, \ldots, \lambda_d$, then Σ has a decomposition

$$\Sigma = Q \begin{pmatrix} \lambda_1 & 0 & \ldots & 0 \\ 0 & \lambda_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \lambda_d \end{pmatrix} Q^{-1},$$

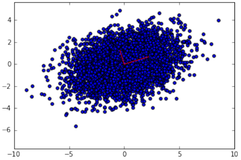
where the columns of Q are the eigenvectors $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_d$.

Recall from Lecture 6: What does the eigenvalue decomposition of the covariance matrix mean?

- What does

$$\Sigma = Q \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_d \end{pmatrix} Q^{-1}$$

  mean?
- Q is a change of bases, re-expressing the covariance matrix in the basis defined by the eigenvectors.

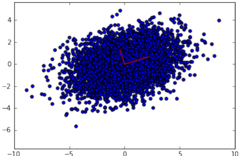


Recall from Lecture 6: What does the eigenvalue decomposition of the covariance matrix mean?

- The diagonal matrix

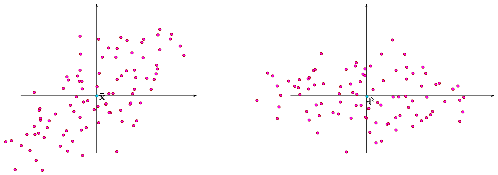
$$D = \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_d \end{pmatrix}$$

  is the covariance of the dataset $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\} \subset \mathbb{R}^d$, expressed in the new basis.
- What do you see?
  - The coordinates of the data points in the new basis are independent!
  - The variance of each coordinate is found along the diagonal!



Recall from Lecture 6: What does the change of basis do?

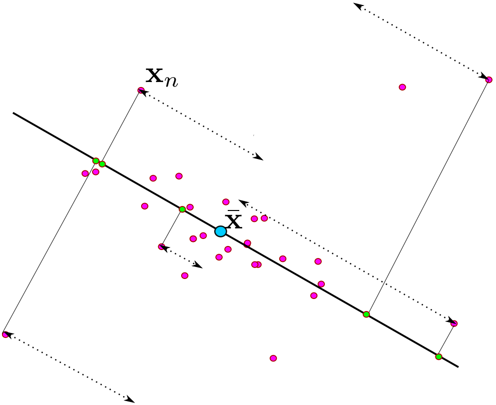
- Align principal components with axes in the new coordinate system
- The intrinsic geometry of the data is unchanged! Only rotation and reflection.
  (Because eigenvectors are orthonormal)



Recall from Lecture 6: Principal components analysis (PCA)

The k first *principal components* of the dataset $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$ span the k-dimensional linear subspace $V \subset \mathbb{R}^d$ that *maximizes the variance* of the projected dataset

- **Question:** Is this V unique?

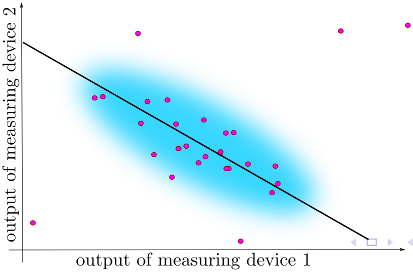


Dimensionality reduction

- PCA is an example of *dimensionality reduction*
- Dimensionality reduction refers to the process of reducing the dimensionality in your data representation.
- More precisely: Given a dataset $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\} \subset \mathbb{R}^{d_1}$, finding a representation of your dataset

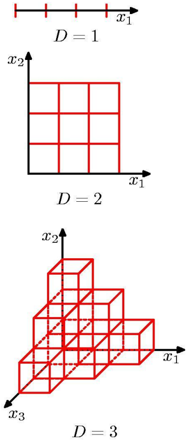
$$\{\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2), \ldots, \phi(\boldsymbol{x}_N)\} \subset \mathbb{R}^{d_2}$$

  where $d_2 < d_1$, and where you retain the properties of your dataset as well as possible.
- Why is this useful?



The curse of dimensionality[1]

- In order to sample the interval $[0, 1]$ with density 0.1, I need 10 points.
- In order to sample the cube $[0, 1] \times [0, 1]$ with the same density, I need 100 points.
- etc
- The more dimensions, the more data you need for drawing conclusions.



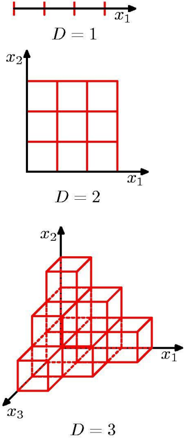
[1]Figure from Bishop: Pattern Recognition and Machine Learning

The curse of dimensionality[1]

- Consider the d-cube $[-1,1]^d$.
- The distance from the center to a corner is

$$\sqrt{d} \to \infty \text{ as } d \to \infty$$

- When d gets large, everything gets large – including noise effects!



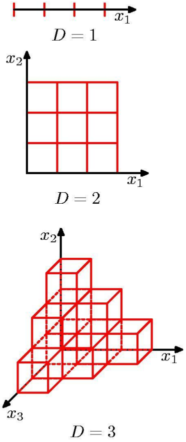
[1] Figure from Bishop: Pattern Recognition and Machine Learning

The curse of dimensionality[1]

- Consider the d-cube $[-1, 1]^d$.
- The distance from the center to a corner is

$$\sqrt{d} \to \infty \text{ as } d \to \infty$$

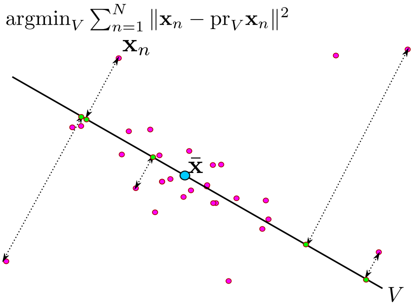
- When d gets large, everything gets large – including noise effects!
- What sort of problems could this give you?



[1] Figure from Bishop: Pattern Recognition and Machine Learning

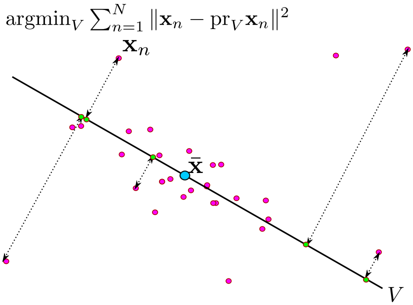
Equivalence of projection error minimization / projected variance maximation

- PCA equivalently formulated as minimizing squared projection error
- A least squares problem
- Equivalent to variance maximization *up to projection*



Equivalence of projection error minimization / projected variance maximation

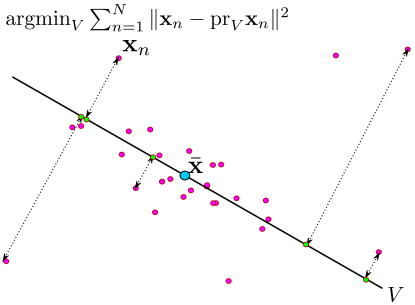
- PCA equivalently formulated as minimizing squared projection error
- A least squares problem
- Equivalent to variance maximization *up to projection* (**is it?**)



Equivalence of projection error minimization / projected variance maximation

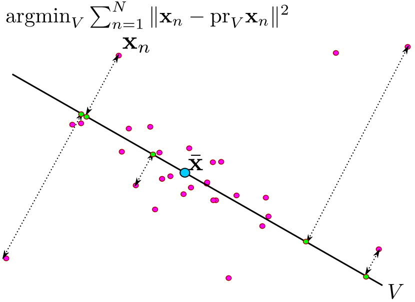
- PCA equivalently formulated as minimizing squared projection error
- A least squares problem
- Equivalent to variance maximization *up to projection* (**is it?**)

- For equivalence: Ask PCs to pass through the mean



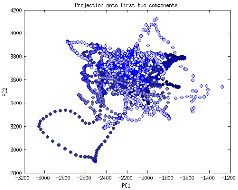
Equivalence of projection error minimization / projected variance maximation

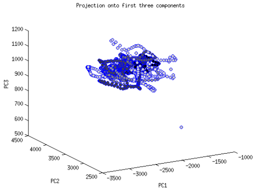
- PCA equivalently formulated as minimizing squared projection error
- A least squares problem
- Equivalent to variance maximization *up to projection* (**is it?**) (**why equivalent?**)
- For equivalence: Ask PCs to pass through the mean



PCA and low distortion embedding – Multidimensional Scaling

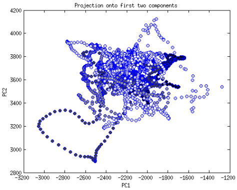
- Projection of data onto PCs is often used to visualize global dataset structure – below is the talking face dataset

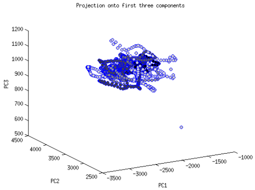




PCA and low distortion embedding – Multidimensional Scaling

- Projection of data onto PCs is often used to visualize global dataset structure – below is the talking face dataset





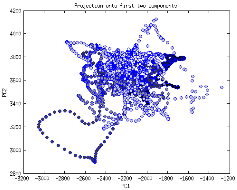
- PCA defines minimizing subspaces for the objective function

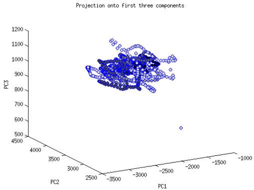
$$\sum_{i,j=1}^{N} \left(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2_{\mathbb{R}^d} - \| \mathrm{pr}_V(\boldsymbol{x}_i) - \mathrm{pr}_V(\boldsymbol{x}_j)\|^2_V \right)$$

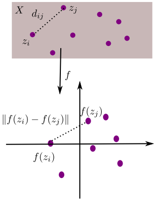
- Does PCA define a unique minimizing subspace?

PCA and low distortion embedding – Multidimensional Scaling

- Projection of data onto PCs is often used to visualize global dataset structure – below is the talking face dataset







- PCA defines minimizing subspaces for the objective function

$$\sum_{i,j=1}^{N} \left(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_{\mathbb{R}^d}^2 - \| \operatorname{pr}_V(\boldsymbol{x}_i) - \operatorname{pr}_V(\boldsymbol{x}_j) \|_V^2 \right)$$

- Does PCA define a unique minimizing subspace?
- The strategy of projecting onto subspaces while preserving distances is called *multidimensional scaling*, or MDS

Projection onto a subspace

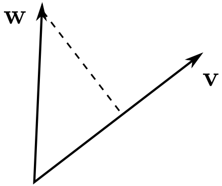
- **Task:** Project your datapoints $\boldsymbol{x}_i$ onto the linear subspace V spanned by $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_k$.

Projection onto a subspace

- **Task:** Project your datapoints $\boldsymbol{x}_i$ onto the linear subspace V spanned by $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_k$.
- **How do you do that? (You will be doing this in a future assignment)**

Projection onto a subspace

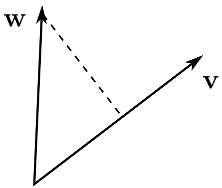
- **Task:** Project your datapoints $\boldsymbol{x}_i$ onto the linear subspace V spanned by $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_k$.
- **How do you do that? (You will be doing this in a future assignment)**
- $\mathrm{pr}_{\boldsymbol{v}}(\boldsymbol{w}) = \boldsymbol{w} \cdot \boldsymbol{v} = \boldsymbol{w}^T \boldsymbol{v}$ for a unit vector $\boldsymbol{v}$ – given by dot product



$$\mathrm{pr}_{\mathbf{v}}\mathbf{w} = \mathbf{v} \cdot \mathbf{w} = \mathbf{v}^{\mathbf{T}}\mathbf{w} = \mathbf{w}^{\mathbf{T}}\mathbf{v} \text{ if } \|\mathbf{v}\| = \mathbf{1}$$

Projection onto a subspace

- **Task:** Project your datapoints $\boldsymbol{x}_i$ onto the linear subspace V spanned by $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_k$.
- **How do you do that? (You will be doing this in a future assignment)**
- $\mathrm{pr}_{\boldsymbol{v}}(\boldsymbol{w}) = \boldsymbol{w} \cdot \boldsymbol{v} = \boldsymbol{w}^T \boldsymbol{v}$ for a unit vector $\boldsymbol{v}$ – given by dot product
- $\mathrm{pr}_V(\boldsymbol{w}) = (\boldsymbol{w}^T \boldsymbol{e}_1)\boldsymbol{e}_1 + \ldots + (\boldsymbol{w}^T \boldsymbol{e}_k)\boldsymbol{e}_k = (\boldsymbol{w}^T \boldsymbol{e}_1, \ldots, \boldsymbol{w}^T \boldsymbol{e}_k)$



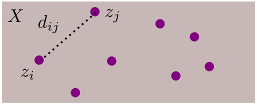
$\mathrm{pr}_{\mathbf{v}}\mathbf{w} = \mathbf{v} \cdot \mathbf{w} = \mathbf{v}^{\mathbf{T}}\mathbf{w} = \mathbf{w}^{\mathbf{T}}\mathbf{v}$ if $\|\mathbf{v}\| = \mathbf{1}$

Why does PCA give MDS?

Input: Distance matrix $D = (d_{ij})$

for distances $d_{ij} = d(z_i, z_j)$

dataset $\{z_n\}_{n=1}^N \subset X$ general data space



Why does PCA give MDS?

Input: Distance matrix $D = (d_{ij})$

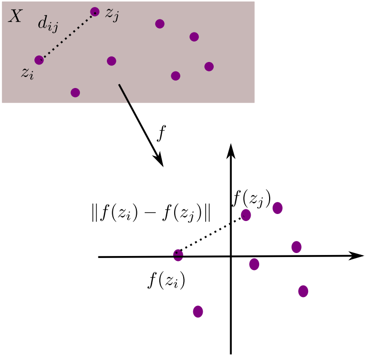
for distances $d_{ij} = d(z_i, z_j)$

dataset $\{z_n\}_{n=1}^N \subset X$ general data space

Goal: Find mapping $f\colon X \to \mathbb{R}^d$ for small d such that

$\Phi(Y) = \sum_{i=1}^N \sum_{j=1}^N (d_{ij}^2 - \|f(z_i) - f(z_j)\|^2)$

is minimized.



Why does PCA give MDS?

Input: Distance matrix $D = (d_{ij})$

for distances $d_{ij} = d(z_i, z_j)$

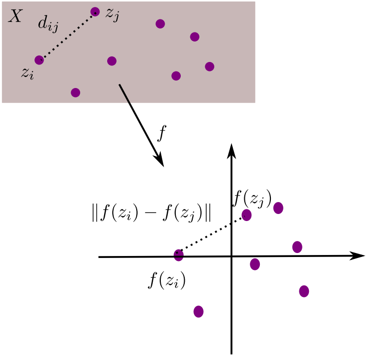
dataset $\{z_n\}_{n=1}^N \subset X$ general data space

Goal: Find mapping $f\colon X \to \mathbb{R}^d$ for small d such that

$$\Phi(Y) = \sum_{i=1}^N \sum_{j=1}^N (d_{ij}^2 - \|f(z_i) - f(z_j)\|^2)$$

is minimized.

That is, distances are preserved as well as possible.



Why does PCA give MDS?

Input: Distance matrix $D = (d_{ij})$

for distances $d_{ij} = d(z_i, z_j)$

dataset $\{z_n\}_{n=1}^N \subset X$ general data space

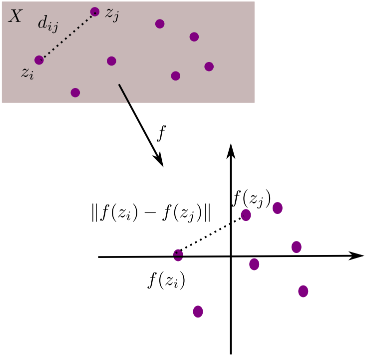
Goal: Find mapping $f\colon X \to \mathbb{R}^d$ for small d such that

$$\Phi(Y) = \sum_{i=1}^N \sum_{j=1}^N (d_{ij}^2 - \|f(z_i) - f(z_j)\|^2)$$

is minimized.

That is, distances are preserved as well as possible.

Assume $X = \mathbb{R}^k$ for $k >> 0$



Why does PCA give MDS?

Input: Distance matrix $D = (d_{ij})$

for distances $d_{ij} = d(z_i, z_j)$

dataset $\{z_n\}_{n=1}^N \subset X$ general data space

Goal: Find mapping $f\colon X \to \mathbb{R}^d$ for small d such that

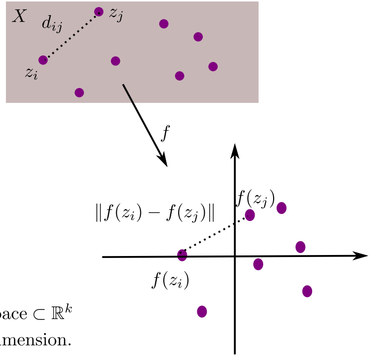
$$\Phi(Y) = \sum_{i=1}^N \sum_{j=1}^N (d_{ij}^2 - \|f(z_i) - f(z_j)\|^2)$$

is minimized.

That is, distances are preserved as well as possible.

Assume $X = \mathbb{R}^k$ for $k >> 0$

$f(z) = Mz$ linear projection onto linear subspace $\subset \mathbb{R}^k$ of low dimension.



Why does PCA give MDS?

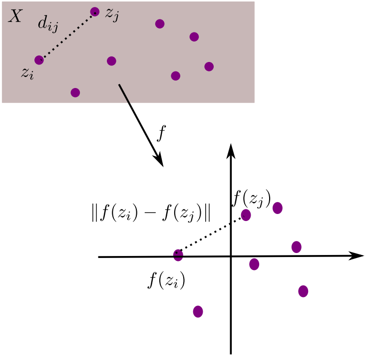
Goal: Find mapping $f\colon X \to \mathbb{R}^d$ for small d such that

$$\Phi(Y) = \sum_{i=1}^{N} \sum_{j=1}^{N} (d_{ij}^2 - \|f(z_i) - f(z_j)\|^2)$$

is minimized.

That is, we seek

$$\operatorname{argmin} \sum_{i,j} (d_{ij}^2 - \|Mz_i - Mz_j\|^2)$$



Why does PCA give MDS?

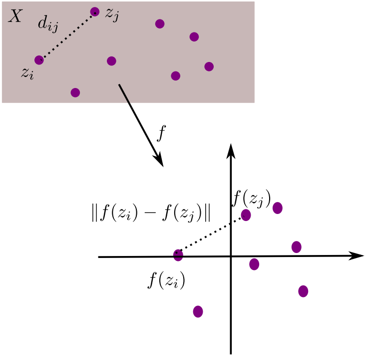
Goal: Find mapping $f\colon X \to \mathbb{R}^d$ for small d such that

$$\Phi(Y) = \sum_{i=1}^{N} \sum_{j=1}^{N} (d_{ij}^2 - \|f(z_i) - f(z_j)\|^2)$$

is minimized.

That is, we seek

$$\begin{aligned} &\operatorname{argmin} \textstyle\sum_{i,j} (d_{ij}^2 - \|Mz_i - Mz_j\|^2) \\ = &\operatorname{argmin} \left(\textstyle\sum_{i,j} d_{ij}^2 - \sum_{i,j} \|Mz_i - Mz_j\|^2 \right) \end{aligned}$$



Why does PCA give MDS?

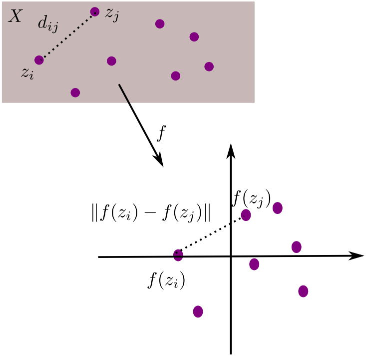
Goal: Find mapping $f\colon X \to \mathbb{R}^d$ for small d such that

$$\Phi(Y) = \sum_{i=1}^N \sum_{j=1}^N (d_{ij}^2 - \|f(z_i) - f(z_j)\|^2)$$

is minimized.

That is, we seek

$$\begin{aligned} &\operatorname{argmin} \textstyle\sum_{i,j} (d_{ij}^2 - \|Mz_i - Mz_j\|^2) \\ = &\operatorname{argmin} \left(\textstyle\sum_{i,j} d_{ij}^2 - \sum_{i,j} \|Mz_i - Mz_j\|^2 \right) \\ = &\operatorname{argmax} \textstyle\sum_{i,j} \|Mz_i - Mz_j\|^2 \end{aligned}$$



Why does PCA give MDS?

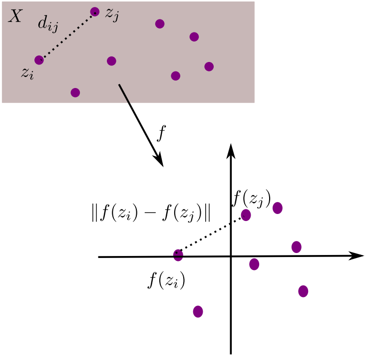
Goal: Find mapping $f\colon X \to \mathbb{R}^d$ for small d such that

$$\Phi(Y) = \sum_{i=1}^{N} \sum_{j=1}^{N} (d_{ij}^2 - \|f(z_i) - f(z_j)\|^2)$$

is minimized.

That is, we seek

$$\begin{aligned}
&\operatorname{argmin} \textstyle\sum_{i,j} (d_{ij}^2 - \|Mz_i - Mz_j\|^2) \\
&= \operatorname{argmin} \left(\textstyle\sum_{i,j} d_{ij}^2 - \sum_{i,j} \|Mz_i - Mz_j\|^2 \right) \\
&= \operatorname{argmax} \textstyle\sum_{i,j} \|Mz_i - Mz_j\|^2 \\
&= \operatorname{argmax} 2N \textstyle\sum_{i} \|Mz_i - \frac{1}{N} \sum_j Mz_j\|^2
\end{aligned}$$



Why does PCA give MDS?

Goal: Find mapping $f\colon X \to \mathbb{R}^d$ for small d such that

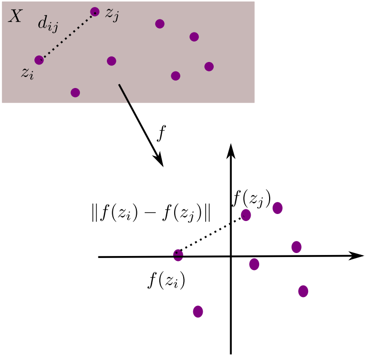
$$\Phi(Y) = \sum_{i=1}^{N} \sum_{j=1}^{N} (d_{ij}^2 - \|f(z_i) - f(z_j)\|^2)$$

is minimized.

That is, we seek

$$\begin{aligned}
&\operatorname{argmin} \textstyle\sum_{i,j} (d_{ij}^2 - \|Mz_i - Mz_j\|^2) \\
&= \operatorname{argmin} \left(\textstyle\sum_{i,j} d_{ij}^2 - \sum_{i,j} \|Mz_i - Mz_j\|^2 \right) \\
&= \operatorname{argmax} \textstyle\sum_{i,j} \|Mz_i - Mz_j\|^2 \\
&= \operatorname{argmax} 2N \textstyle\sum_{i} \|Mz_i - \frac{1}{N} \sum_j Mz_j\|^2
\end{aligned}$$

equivalent to maximizing projected variance.



Why does PCA give MDS?

Goal: Find mapping $f\colon X \to \mathbb{R}^d$ for small d such that

$$\Phi(Y) = \sum_{i=1}^{N} \sum_{j=1}^{N} (d_{ij}^2 - \|f(z_i) - f(z_j)\|^2)$$

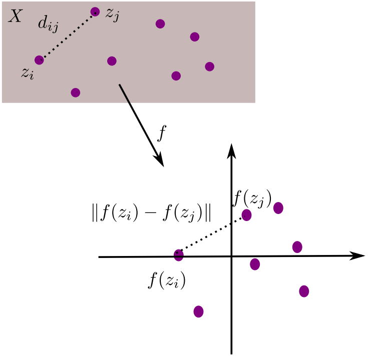
is minimized.

That is, we seek

$$\begin{aligned}
&\operatorname{argmin} \sum_{i,j} (d_{ij}^2 - \|Mz_i - Mz_j\|^2) \\
&= \operatorname{argmin} \left(\sum_{i,j} d_{ij}^2 - \sum_{i,j} \|Mz_i - Mz_j\|^2 \right) \\
&= \operatorname{argmax} \sum_{i,j} \|Mz_i - Mz_j\|^2 \\
&= \operatorname{argmax} 2N \sum_i \|Mz_i - \tfrac{1}{N} \sum_j Mz_j\|^2
\end{aligned}$$

equivalent to maximizing projected variance.

Familiar?



Why does PCA give MDS?

Input: Distance matrix $D = (d_{ij})$

for distances $d_{ij} = d(z_i, z_j)$

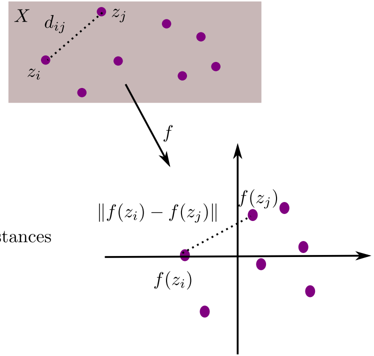
dataset $\{z_n\}_{n=1}^N \subset X$ general data space

Goal: Find mapping $f\colon X \to \mathbb{R}^d$ for small d such that

$\Phi(Y) = \sum_{i=1}^N \sum_{j=1}^N (d_{ij}^2 - \|f(z_i) - f(z_j)\|^2)$

is minimized.

New interpretation of PCA: Preserving squared distances



Why does PCA give MDS?

Input: Distance matrix $D = (d_{ij})$

for distances $d_{ij} = d(z_i, z_j)$

dataset $\{z_n\}_{n=1}^N \subset X$ general data space

Goal: Find mapping $f\colon X \to \mathbb{R}^d$ for small d such that

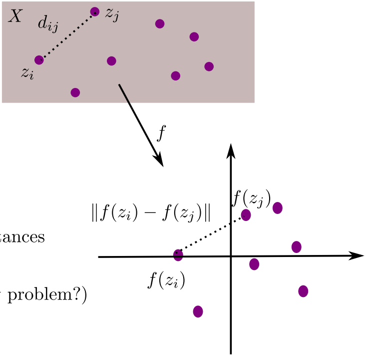
$\Phi(Y) = \sum_{i=1}^N \sum_{j=1}^N (d_{ij}^2 - \|f(z_i) - f(z_j)\|^2)$

is minimized.

New interpretation of PCA: Preserving squared distances

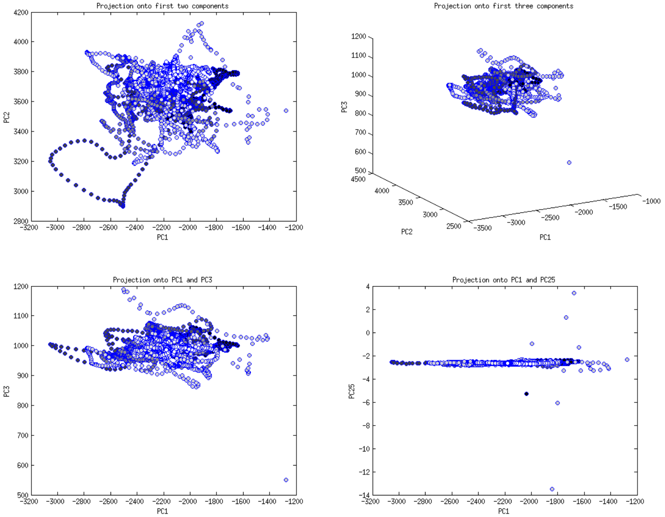
Revealing problem with PCA:

Preserves long distances better than short ones (why problem?)



PCA and visualization

- Low-dimensional representation of the data via projection onto first 2 or 3 principal components



Summary

We have seen two alternative ways of defining PCA:

- Minimizing projection error

$$\operatorname{argmin}_V \sum_{n=1}^{N} \|\boldsymbol{x}_n - \mathrm{pr}_V(\boldsymbol{x}_n)\|^2$$

- Minimizing distortion

$$\operatorname{argmin}_V \sum_{i,j=1}^{N} \left(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_{\mathbb{R}^d}^2 - \| \mathrm{pr}_V(\boldsymbol{x}_i) - \mathrm{pr}_V(\boldsymbol{x}_j)\|_V^2 \right)$$

- What do we learn from these?

Summary

We have seen two alternative ways of defining PCA:

- Minimizing projection error

$$\operatorname{argmin}_V \sum_{n=1}^{N} \|\boldsymbol{x}_n - \mathrm{pr}_V(\boldsymbol{x}_n)\|^2$$

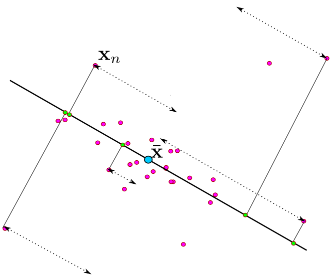
- Minimizing distortion

$$\operatorname{argmin}_V \sum_{i,j=1}^{N} \left(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_{\mathbb{R}^d}^2 - \|\mathrm{pr}_V(\boldsymbol{x}_i) - \mathrm{pr}_V(\boldsymbol{x}_j)\|_V^2\right)$$

- What do we learn from these? PCA is sensitive to outliers!

Alternate way of solving PCA: Optimization

The PC1 of the dataset $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$ spans the line $l \subset \mathbb{R}^d$ that *maximizes the variance* of the projected dataset



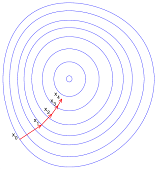
Assuming $\bar{\boldsymbol{x}} = 0$, this translates to the optimization problem

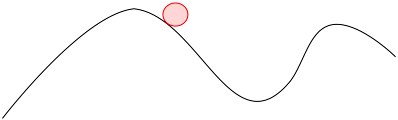
$$\operatorname{argmin}_{\boldsymbol{w}, \|\boldsymbol{w}=1\|} \sum_{n=1}^{N} \| \operatorname{pr}_{\boldsymbol{w}} \boldsymbol{x}_n - \overline{\operatorname{pr}_{\boldsymbol{w}}(\boldsymbol{x})} \|^2$$
$$= \operatorname{argmin}_{\boldsymbol{w}, \|\boldsymbol{w}=1\|} \sum_{n=1}^{N} \|\boldsymbol{w}^T \boldsymbol{x}_n\|^2,$$

where $\boldsymbol{w}$ is a unit vector parallel to the line l.

Optimization: Gradient descent

- We will be using gradient descent intensively in the next couple of weeks, so let's start out reviewing its basics
- Gradient descent is used to find *minimum values*
- Idea: Walk downhill until you hit bottom



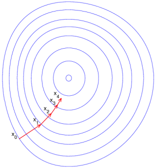


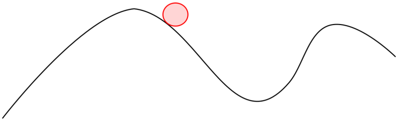
Issues that need your attention:

- Where do I start?
- How long steps?
- When have I hit bottom?
- Do you see any dangers?

Optimization: Gradient descent

- We will be using gradient descent intensively in the next couple of weeks, so let's start out reviewing its basics
- Gradient descent is used to find *minimum values*
- Idea: Walk downhill until you hit bottom



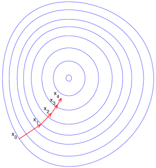


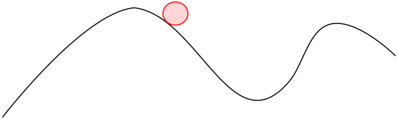
Issues that need your attention:

- Where do I start?
  Common: Initialize Random sample from normal distribution
- How long steps?
- When have I hit bottom?
- Do you see any dangers?

Optimization: Gradient descent

- We will be using gradient descent intensively in the next couple of weeks, so let's start out reviewing its basics
- Gradient descent is used to find *minimum values*
- Idea: Walk downhill until you hit bottom



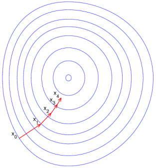


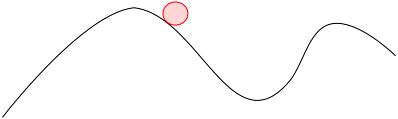
Issues that need your attention:

- Where do I start?
  Common: Initialize Random sample from normal distribution
- How long steps? Common: $\eta_t = \eta \|\nabla E\|$; typical $\nabla = 0.1$
- When have I hit bottom?
- Do you see any dangers?

Optimization: Gradient descent

- We will be using gradient descent intensively in the next couple of weeks, so let's start out reviewing its basics
- Gradient descent is used to find *minimum values*
- Idea: Walk downhill until you hit bottom





Issues that need your attention:

- Where do I start?
  Common: Initialize Random sample from normal distribution
- How long steps? Common: $\eta_t = \eta \|\nabla E\|$; typical $\nabla = 0.1$
- When have I hit bottom? Threshold on $\sharp$ steps or step size
- Do you see any dangers?

Before gradient descent: Derivatives

- Let $f : \mathbb{R} \to \mathbb{R}$, given by $y = f(x)$, be a function
- The *derivative* of f in the point x_0 is the limit

$$\frac{df}{dx}(x_0) = f'(x_0) = \lim_{h \to 0} \frac{f(x_0 + h) - f(x_0)}{h}.$$

- When the derivative exists in every x_0, the function f is *differentiable*.
- Using the theory of limits, one can derive formulas for common derivatives:

$$\tfrac{d}{dx}x^n = nx^{n-1}, \quad \tfrac{d}{dx}(f(g(x)) = f'(g(x))g'(x)$$
$$\tfrac{d}{dx}(f(x) + g(x)) = \tfrac{d}{dx}f(x) + \tfrac{d}{dx}g(x)$$

Before gradient descent: Derivatives

- Let $f : \mathbb{R} \to \mathbb{R}$, given by $y = f(x)$, be a function
- The *derivative* of f in the point x_0 is the limit

$$\frac{df}{dx}(x_0) = f'(x_0) = \lim_{h\to 0} \frac{f(x_0+h) - f(x_0)}{h}.$$

- When the derivative exists in every x_0, the function f is *differentiable*.
- Using the theory of limits, one can derive formulas for common derivatives:

$$\tfrac{d}{dx}x^n = nx^{n-1}, \qquad \tfrac{d}{dx}(f(g(x)) = f'(g(x))g'(x)$$
$$\tfrac{d}{dx}(f(x) + g(x)) = \tfrac{d}{dx}f(x) + \tfrac{d}{dx}g(x)$$

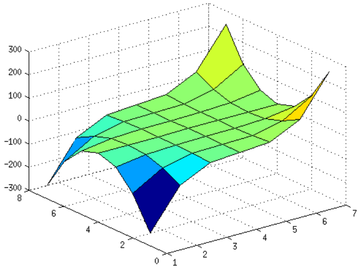
- What are the following derivatives?

$$\frac{d}{dx}(a-x)^2 \qquad \frac{d}{dx}\sum_{n=1}^{N}(x_n - x)^2$$

Partial derivatives – informally

Example

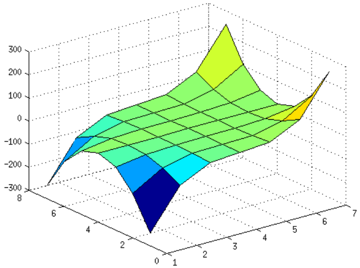
The function $f(x, y) = x^2y^3$.



Partial derivatives – informally

Example

The function $f(x, y) = x^2 y^3$.

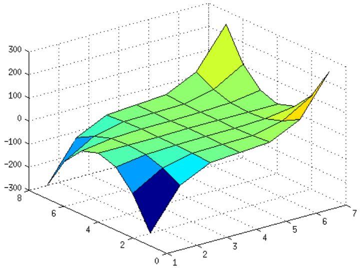


$$\frac{\partial}{\partial x} f(x, y) = 2xy^3 \quad \frac{\partial}{\partial y} f(x, y) = x^2 3y^2$$

Partial derivatives – informally

Example

The function $f(x, y) = x^2 y^3$.



$$\frac{\partial}{\partial x} f(x, y) = 2xy^3 \quad \frac{\partial}{\partial y} f(x, y) = x^2 3y^2$$

Exercise

What are $\frac{\partial}{\partial x}$ and $\frac{\partial}{\partial y}$ for the following functions?

- $f(x, y) = y^2 \sin^2(x)$
- $f(x, y) = \sin(x) e^{yx}$

Partial derivatives

Recall that the derivative of a function $f : \mathbb{R} \to \mathbb{R}$ at $x_0 \in \mathbb{R}$ is given by

$$\frac{\partial f}{\partial x}(x_0) = \lim_{h \to 0} \frac{f(x_0 + h) - f(x_0)}{h}$$

Let $f : \mathbb{R}^2 \to \mathbb{R}$, $z = f(x, y)$ be a function of two variables.

- The partial derivative of f with respect to x at (x_0, y_0) is

$$\frac{\partial f}{\partial x}(x_0, y_0) = \lim_{h \to 0} \frac{f(x_0 + h, y_0) - f(x_0, y_0)}{h}$$

Partial derivatives

Recall that the derivative of a function $f: \mathbb{R} \to \mathbb{R}$ at $x_0 \in \mathbb{R}$ is given by

$$\frac{\partial f}{\partial x}(x_0) = \lim_{h \to 0} \frac{f(x_0 + h) - f(x_0)}{h}$$

Let $f: \mathbb{R}^2 \to \mathbb{R}$, $z = f(x, y)$ be a function of two variables.

- The partial derivative of f with respect to x at (x_0, y_0) is

$$\frac{\partial f}{\partial x}(x_0, y_0) = \lim_{h \to 0} \frac{f(x_0 + h, y_0) - f(x_0, y_0)}{h}$$

- The partial derivative of f with respect to y at (x_0, y_0) is

$$\frac{\partial f}{\partial x}(x_0, y_0) = \lim_{h \to 0} \frac{f(x_0, y_0 + h) - f(x_0, y_0)}{h}$$

Partial derivatives

- f is *differentiable* at (x_0, y_0) if both partial derivatives exist there

Partial derivatives

- f is *differentiable* at (x_0, y_0) if both partial derivatives exist there
- f is *differentiable* if it is differentiable everywhere

Partial derivatives

- f is *differentiable* at (x_0, y_0) if both partial derivatives exist there
- f is *differentiable* if it is differentiable everywhere
- Concept carries over to functions $f : \mathbb{R}^n \to \mathbb{R}$ of any number of variables.

Gradients

The *gradient* of a function $f : \mathbb{R}^2 \to \mathbb{R}$, $z = f(x, y)$ in the point (x_0, y_0) is given by

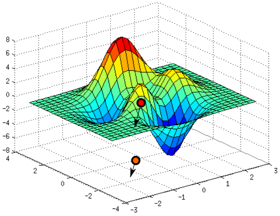
$$\begin{aligned} \nabla f(x_0, y_0) &= \left(\frac{\partial f}{\partial x}(x_0, y_0), \frac{\partial f}{\partial y}(x_0, y_0) \right) \\ &= \lim_{h \to 0} \left(\frac{f(x_0+h, y_0) - f(x_0, y_0)}{h}, \frac{f(x_0, y_0+h) - f(x_0, y_0)}{h} \right) \end{aligned}$$

Gradients

The *gradient* of a function $f : \mathbb{R}^2 \to \mathbb{R}$, $z = f(x, y)$ in the point (x_0, y_0) is given by

$$\begin{aligned} \nabla f(x_0, y_0) &= \left(\frac{\partial f}{\partial x}(x_0, y_0), \frac{\partial f}{\partial y}(x_0, y_0) \right) \\ &= \lim_{h \to 0} \left(\frac{f(x_0+h, y_0) - f(x_0, y_0)}{h}, \frac{f(x_0, y_0+h) - f(x_0, y_0)}{h} \right) \end{aligned}$$

The gradient points in the direction of the steepest descent of the function.

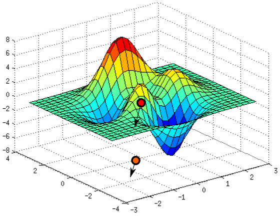


Gradients

The *gradient* of a function $f : \mathbb{R}^2 \to \mathbb{R}$, $z = f(x, y)$ in the point (x_0, y_0) is given by

$$\begin{aligned} \nabla f(x_0, y_0) &= \left(\frac{\partial f}{\partial x}(x_0, y_0), \frac{\partial f}{\partial y}(x_0, y_0)\right) \\ &= \lim_{h \to 0} \left(\frac{f(x_0+h, y_0) - f(x_0, y_0)}{h}, \frac{f(x_0, y_0+h) - f(x_0, y_0)}{h}\right) \end{aligned}$$

The gradient points in the direction of the steepest descent of the function.



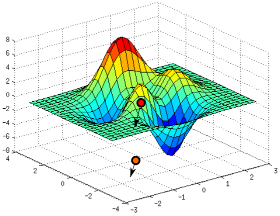
What can this be useful for?

Gradients

The *gradient* of a function $f : \mathbb{R}^2 \to \mathbb{R}$, $z = f(x, y)$ in the point (x_0, y_0) is given by

$$\begin{aligned} \nabla f(x_0, y_0) &= \left(\frac{\partial f}{\partial x}(x_0, y_0), \frac{\partial f}{\partial y}(x_0, y_0)\right) \\ &= \lim_{h\to 0}\left(\frac{f(x_0+h, y_0) - f(x_0, y_0)}{h}, \frac{f(x_0, y_0+h) - f(x_0, y_0)}{h}\right) \end{aligned}$$

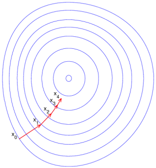
The gradient points in the direction of the steepest descent of the function.

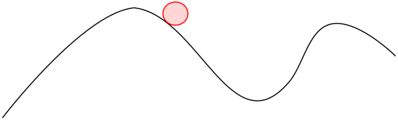


What can this be useful for? **Optimization – gradient descent!**

Optimization: Gradient descent

- We will be using gradient descent intensively in the next couple of weeks, so let's start out reviewing its basics
- Gradient descent is used to find *minimum values*
- Idea: Walk downhill until you hit bottom





Issues that need your attention:

- Where do I start?
- How long steps?
- When have I hit bottom?
- Do you see any dangers?

Basic gradient descent algorithm for PCA

Searching for

$$\operatorname{argmin}_{\boldsymbol{w},\|\boldsymbol{w}=1\|} f(\boldsymbol{w}) = \operatorname{argmin}_{\boldsymbol{w},\|\boldsymbol{w}=1\|} \sum_{n=1}^{N} \|\boldsymbol{w}^T \boldsymbol{x}_n\|^2$$

```
Initialize: w ← w_0
while Convergence = False do
    Compute ∇f(w_0)
    Set step size s
    w ← w + s∇f(w_0)
    w ← w / ||w||
    if Convergence criteria hold then
        Convergence = True
    end if
end while
```

Basic gradient descent algorithm for PCA

Searching for

$$\text{argmin}_{\boldsymbol{w},\|\boldsymbol{w}=1\|} f(\boldsymbol{w}) = \text{argmin}_{\boldsymbol{w},\|\boldsymbol{w}=1\|} \sum_{n=1}^{N} \|\boldsymbol{w}^T \boldsymbol{x}_n\|^2$$

```
Initialize: w ← w_0
while Convergence = False do
    Compute ∇f(w_0)
    Set step size s
    w ← w + s∇f(w_0)
    w ← w/‖w‖   Why do we need this?
    if Convergence criteria hold then
        Convergence = True
    end if
end while
```

When is gradient descent useful?

- For PCA? Large, high-dimensional datasets
- For other things? We shall see next week!

Next week:

- Tuesday: Regression with a single variable, and a bit of linear algrbra (Chapter 4 + 14)
- Thursday: Multivariate regression (Chapter 15)