Data Science
Principal Component Analysis

Linear latent decomposition

Stéphane Marchand-Maillet

Department of Computer Science

UNIVERSITÉ DE GENÈVE
FACULTÉ DES SCIENCES

Master en Sciences Informatiques - Autumn semester

Table of contents

What is this lecture about?

- ⋆ Base representations may not be optimal (to be defined)
- ⋆ Latent models promise to exhibit the underlying (latent) factors that drive the process in question

- ⋆ This initial (but fundamental) definition of latent factors uses statistical correlation

$\Rightarrow$ It exhibits linear latent factors

$\Rightarrow$ Enables "simplification" of the data by sound decimation

- ⋆ Also: we will study several interpretations

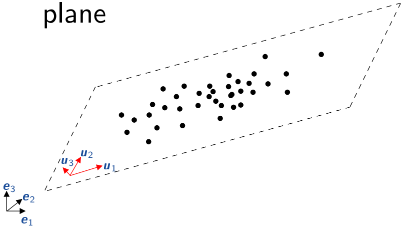
Reading: [1] (chap 12) and [3] (chap 3 and 10.3)

Intuition

Given $\mathcal{X} \subset \Omega$, we wish to decompose Ω into subspaces such that the projection of $\mathcal{X}$ onto these subspaces retains the most "information".

Q: What information should we consider?

- ⋆ Say $\mathcal{X}$ is almost "contained" into a 2D plane in a 3D space
- ⋆ A relevant choice for our subspace is to chose a basis $\{\boldsymbol{u}_1, \boldsymbol{u}_2\}$ for the plane



Q: What characterizes $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3\}$?

$\Rightarrow$ the fact that the data varies most along these directions

$\Rightarrow$ the fact that the data varies least orthogonally to these directions ($\boldsymbol{u}_3$)

Formalization

Given $\mathcal{X}$, $\{\boldsymbol{u}_i\}_{i\in[\![D]\!]}$ is a new orthonormal basis of $\mathbb{R}^D$. The Principal Component $\boldsymbol{u}_1$ is chosen such that the variance of the data projected over $\boldsymbol{u}_1$ is maximum. $\boldsymbol{u}_2$ is chosen using $\mathrm{Proj}_{\boldsymbol{u}_1^\perp}(\mathcal{X})$.

Model

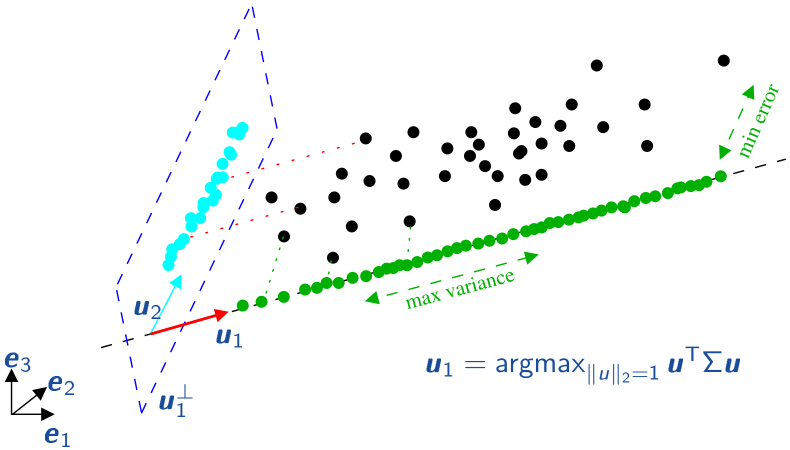
Given $\mathcal{X}$, the sample mean ($\overline{\boldsymbol{x}}$) and the variance of the data projected over $\boldsymbol{u}_1$ are

$$\overline{\boldsymbol{x}} = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{x}_i \quad \text{and} \quad v_{\boldsymbol{u}_1} = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{u}_1^\top\boldsymbol{x}_i - \boldsymbol{u}_1^\top\overline{\boldsymbol{x}})^2 = \boldsymbol{u}_1^\top\Sigma\boldsymbol{u}_1$$

where $\Sigma = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i - \overline{\boldsymbol{x}})(\boldsymbol{x}_i - \overline{\boldsymbol{x}})^\top$ is the data covariance matrix. Therefore

$$\boldsymbol{u}_1 = \underset{\boldsymbol{u}^\top\boldsymbol{u}=1}{\operatorname{argmax}}\, \boldsymbol{u}^\top\Sigma\boldsymbol{u}$$

Intuition



$$u_1 = \mathrm{argmax}_{\|u\|_2=1}\, u^\top \Sigma u$$

Formalization

$$\boldsymbol{u}_1 = \underset{\boldsymbol{u}^\top \boldsymbol{u}=1}{\operatorname{argmax}}\, \boldsymbol{u}^\top \Sigma \boldsymbol{u} \quad \Rightarrow \quad J(\boldsymbol{u}) = \boldsymbol{u}^\top \Sigma \boldsymbol{u} + \lambda(1 - \boldsymbol{u}^\top \boldsymbol{u})$$

So that

$$\left.\frac{\partial J(\boldsymbol{u})}{\partial \boldsymbol{u}}\right|_{\boldsymbol{u}=\boldsymbol{u}_1} = 0 \quad \text{and} \quad \left.\frac{\partial J(\boldsymbol{u})}{\partial \lambda}\right|_{\lambda=\lambda_1} = 0$$

Hence

$$\Sigma \boldsymbol{u}_1 = \lambda_1 \boldsymbol{u}_1 \quad \Rightarrow \quad v_{\boldsymbol{u}_1} = \boldsymbol{u}_1^\top \Sigma \boldsymbol{u}_1 = \lambda_1$$

$\Rightarrow$ $(\boldsymbol{u}_1, \lambda_1)$ is an eigenpair of the covariance matrix Σ
$\Rightarrow$ continuing with the decimation process, we obtain the set of Principal Components as the eigenpairs $\{(\boldsymbol{u}_i, \lambda_i)\}_{i \in [\![D]\!]}$ of the covariance matrix Σ of data $\mathcal{X}$

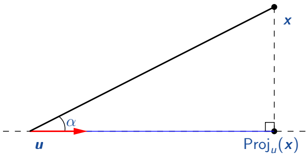
$$\Lambda = \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_D \end{pmatrix} = \begin{pmatrix} | & & | \\ \boldsymbol{u}_1 & \cdots & \boldsymbol{u}_D \\ | & & | \end{pmatrix}^\top \Sigma \begin{pmatrix} | & & | \\ \boldsymbol{u}_1 & \cdots & \boldsymbol{u}_D \\ | & & | \end{pmatrix} = \boldsymbol{U}^\top \Sigma \boldsymbol{U}$$

Formalization

- ⋆ The variance $v_{\boldsymbol{u}_1}$ is expressed as a sum of squares
  $v_{\boldsymbol{u}_1} = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{u}_1^\top(\boldsymbol{x}_i - \overline{\boldsymbol{x}}))^2$
- ⇒ To maximize $v_{\boldsymbol{u}_1}$, terms $\boldsymbol{u}_1^\top(\boldsymbol{x}_i - \overline{\boldsymbol{x}})$ should be collectively maximized
- ⇒ Since $(\boldsymbol{x}_i - \overline{\boldsymbol{x}})$ is fixed, Pythagoras tells us it is equivalent to minimize the distance to the axis of projection (approximation error)
- ⇒ A Principal Component is a quadratic regression over the data

$$\boldsymbol{u}_1 = \underset{\boldsymbol{u}^\top\boldsymbol{u}=1}{\operatorname{argmin}} \sum_{i=1}^{N} \|(\boldsymbol{x}_i - \overline{\boldsymbol{x}}) - [\boldsymbol{u}^\top(\boldsymbol{x}_i - \overline{\boldsymbol{x}})]\boldsymbol{u}\|_2^2 \quad \Rightarrow \quad \Sigma\boldsymbol{u}_1 = \lambda_1\boldsymbol{u}_1 \qquad \varphi$$

Maximize Projection Variance
⇔ Minimize Approximation Error



Physical interpretation

- ⋆ Consider a physical system $\mathcal{X}$ with masses $m_i = 1$ at positions $\boldsymbol{x}_i$
- ⋆ The inertia of the system w.r.t $\boldsymbol{a} \in \Omega$ is $I_{\boldsymbol{a}}(\mathcal{X}) = \sum_{i=1}^{N} d^2(a, x_i)$
- ⋆ Huygens theorem tells us that if $\boldsymbol{g} = \frac{1}{N} \sum_{i=1} \boldsymbol{x}_i$ then

$$I_{\boldsymbol{a}}(\mathcal{X}) = d^2(\boldsymbol{a}, \boldsymbol{g}) + I_{\boldsymbol{g}}(\mathcal{X}) \qquad (\mathrm{Var}(X) = \mathbb{E}[X^2] - (\mathbb{E}X)^2)$$

- ⋆ and if $\mathfrak{F}$ is a subspace of Ω going thru $\boldsymbol{g}$ then

$$I_{\mathfrak{F}}(\mathcal{X}) = \sum_{i=1}^{N} d^2(\mathfrak{F}, \boldsymbol{x}_i) \quad \text{where} \quad d(\mathfrak{F}, \boldsymbol{x}_i) = \|x_i - \mathrm{Proj}_{\mathfrak{F}}(\boldsymbol{x}_i)\|$$

$\Rightarrow$ A Principal Component is a subspace of least inertia w.r.t $\mathcal{X}$

Structure of the latent space

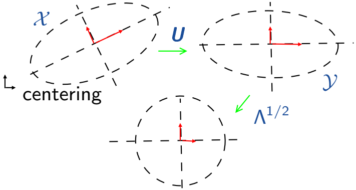
The latent space with basis $\{\boldsymbol{u}_i, \cdots, \boldsymbol{u}_D\}$ has the following properties:

- ⋆ By construction $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_D$ and $\lambda := \sum_{d=1}^{D} \lambda_d = \mathrm{Tr}(\Sigma)$
- ⋆ $v_{\boldsymbol{u}_d} = \lambda_d \Rightarrow \lambda$ represents the total variance $\Rightarrow$ latent features are decorrelated $\boldsymbol{u}_d^{\mathsf{T}} \boldsymbol{u}_{d'} = 0$ (Λ is the diagonal latent covariance matrix)
- ⋆ The basis $\{\boldsymbol{u}_i, \cdots, \boldsymbol{u}_D\}$ induces latent coordinates $\boldsymbol{y}_i$ for the data:

$$\boldsymbol{y}_i(d) = \langle \boldsymbol{u}_d, \boldsymbol{u}_i - \overline{\boldsymbol{x}} \rangle = \boldsymbol{u}_d^{\mathsf{T}}(\boldsymbol{x}_i - \overline{\boldsymbol{x}}) \qquad \text{so that} \qquad \boldsymbol{y}_i = \boldsymbol{U}^{\mathsf{T}}(\boldsymbol{x}_i - \overline{\boldsymbol{x}})$$

The transform is linear and composed of:

- ◦ Centering on $\overline{x}$
- ◦ Rotation using $\boldsymbol{U}$
- ◦ Scaling using $\Lambda^{1/2}$ (whitening)

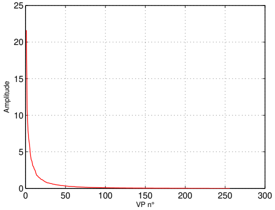


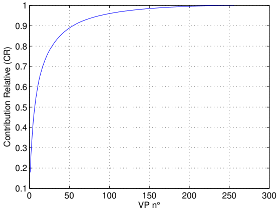
Structure of the model

- ⋆ The latent space preserves the variance (of the centered data)
- ⇒ the underlying data model is $X_i \sim \boldsymbol{f}_{\mathcal{X}} = \mathcal{N}(\overline{\boldsymbol{x}}, \Sigma)$
- ⇒ PCA will not be relevant for non-Gaussian data (e.g clustered)
- ⚠ Important: So far PCA is an exact transform

$$\boldsymbol{y}_i = \boldsymbol{U}^{\mathsf{T}}(\boldsymbol{x}_i - \overline{\boldsymbol{x}}) \qquad \text{so that} \qquad \boldsymbol{U}\boldsymbol{y}_i = \boldsymbol{U}\boldsymbol{U}^{\mathsf{T}}(\boldsymbol{x}_i - \overline{\boldsymbol{x}}) = \boldsymbol{x}_i - \overline{\boldsymbol{x}}$$

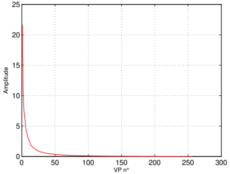
- ⇒ at this stage, one purpose is to study the spectrum $\{\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_D\}$ of the data

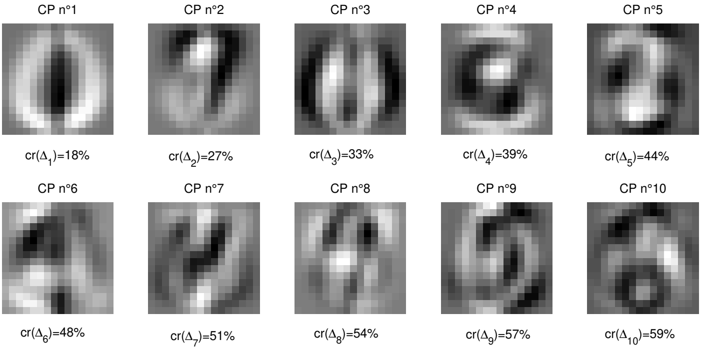




Decomposition via PCA

MNIST partial dataset: $N = 7291$ images 16×16 (8bits) $\Rightarrow D = 256$





Approximation via PCA

Low-rank approximation from the Eckart and Young theorem:

If $\Sigma = \boldsymbol{U}\Lambda\boldsymbol{U}^{\mathsf{T}}$ and for $K < D$ define $\Sigma_K := \sum_{d=1}^{K} \lambda_d \boldsymbol{u}_d \boldsymbol{u}_d^{\mathsf{T}}$ then

$$\underset{\text{rank}(\boldsymbol{S})=K}{\text{argmin}} \|\Sigma - \boldsymbol{S}\|_F^2 = \Sigma_K \qquad \text{and} \qquad \|\Sigma - \Sigma_K\|_F^2 = \sum_{d=K+1}^{D} \lambda_d$$

$\Rightarrow$ Σ_K is the closest K-rank matrix to Σ

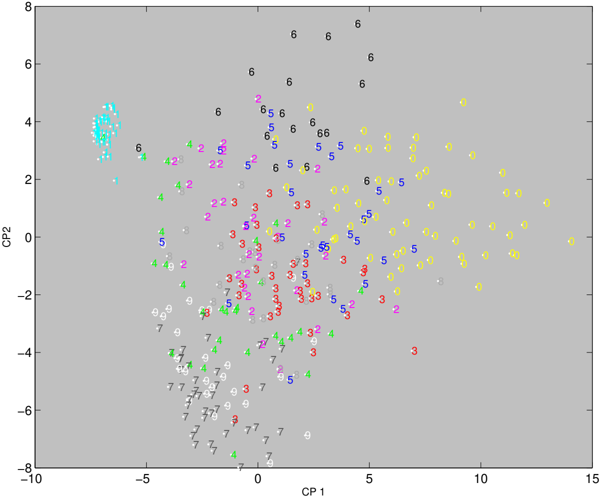
Truncation (of $\boldsymbol{U}$ and Λ)

$$\Sigma_K = \begin{pmatrix} | & & | \\ \boldsymbol{u}_1 & \cdots & \boldsymbol{u}_K \\ | & & | \end{pmatrix} \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_K \end{pmatrix} \begin{pmatrix} | & & | \\ \boldsymbol{u}_1 & \cdots & \boldsymbol{u}_K \\ | & & | \end{pmatrix}^{\mathsf{T}} = \boldsymbol{U}_K \Lambda_K \boldsymbol{U}_K^{\mathsf{T}}$$

⚠ $\Rightarrow \tilde{\boldsymbol{y}}_i = \boldsymbol{U}_K^{\mathsf{T}} \boldsymbol{x}_i \in \mathbb{R}^K$ and $\tilde{\boldsymbol{x}}_i = \boldsymbol{U}_K \boldsymbol{U}_K^{\mathsf{T}} \boldsymbol{x}_i + \overline{\boldsymbol{x}}$

Visualization via PCA

- MNIST partial dataset: $K = 2 \Rightarrow \tilde{\boldsymbol{y}}_i \in \mathbb{R}^2$



Geometry of PCA

The quality of reconstruction can be measured by

- ⋆ the relative contribution of each dimension to the variance $c_d = \frac{\lambda_d}{\sum_k \lambda_k}$

$\Rightarrow$ depends on the distribution of the spectrum

- ⋆ the projection ratio of each data $\boldsymbol{x}_i$ over a latent factor
  $\rho_d(\boldsymbol{x}_i) = \frac{\langle \boldsymbol{u}_d, \boldsymbol{x}_i \rangle^2}{\|\boldsymbol{x}\|^2} = \cos^2(\angle(\boldsymbol{u}_d, \boldsymbol{x}_i))$

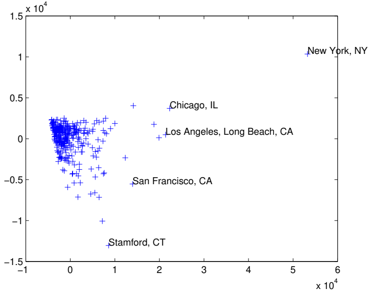
$\Rightarrow$ the closer $\rho_d(\boldsymbol{x}_i)$ is to 1, the more $\boldsymbol{x}_i$ lies on $\boldsymbol{u}_d$

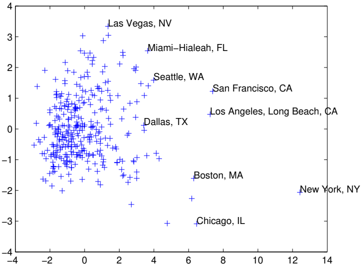
$\Rightarrow$ The above can be grouped (summed) to evaluate wrt a subspace $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots\}$

Geometry of PCA

Every original data comes with its unit (scale), that we can estimate via σ_d^2 the sample variance along original dimension d. PCA is more effective is all scales are similar.

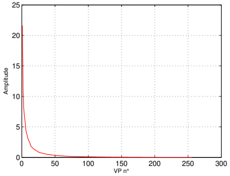
$\Rightarrow$ we create the scaling matrix $\boldsymbol{S} = \mathrm{diag}[\sigma_1^2, \cdots, \sigma_d^2]$ and we define the metric $d_{\boldsymbol{S}}^2(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x} - \boldsymbol{y})^\mathsf{T} \boldsymbol{S}^{-1} (\boldsymbol{x} - \boldsymbol{y}) \Rightarrow$ in that metric space, the covariance matrix $\Sigma_{\boldsymbol{S}}$ is also rescaled (into the correlation matrix) and used as a base for PCA.

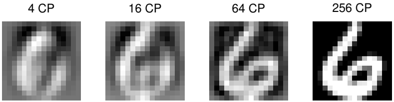


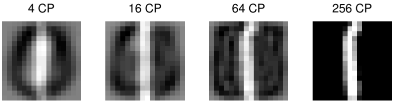


Approximation via PCA

MNIST partial dataset: $N = 7291$ images 16×16 (8bits) $\Rightarrow D = 256$





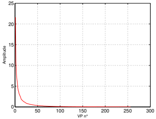


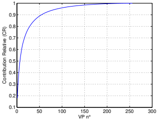
Practical PCA

Given $\mathcal{X} \in \Omega$

- ⋆ Compute the sample mean $\overline{\boldsymbol{x}}$
- ⋆ Center the data $\boldsymbol{x}_i \leftarrow (\boldsymbol{x}_i - \overline{\boldsymbol{x}})$ and form centered data matrix $\boldsymbol{X}$
- ⋆ $\Sigma = \frac{1}{N}\boldsymbol{X}\boldsymbol{X}^\top$ and $\Sigma = \boldsymbol{U}\Lambda\boldsymbol{U}^\top$ ← ⚠ Exact transform so far
- ⋆ Select the number of components K
- ⋆ Define $\boldsymbol{U}_K$, Λ_K and compute $\{\tilde{\boldsymbol{y}}_i\}_{i\in[\![N]\!]}$ and/or $\{\tilde{\boldsymbol{x}}_i\}_{i\in[\![N]\!]}$

Choice of K





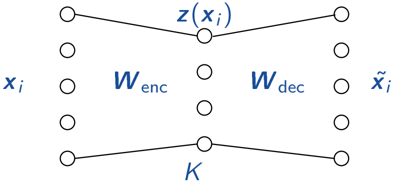
1. $K = d$, the target dimension $\Rightarrow \tilde{\boldsymbol{y}}_i \in \mathbb{R}^d$
2. Require $\mathrm{Var}(\mathcal{Y}) = \tau.\mathrm{Var}(\mathcal{X}) \quad \Rightarrow \quad K$ such that $\frac{\sum_{d=1}^{K} \lambda_d}{\sum_{d=1}^{D} \lambda_d} \geq \tau$
3. Train K such that $\mathcal{L}(\mathcal{X}) \leq \varepsilon$ (e.g $\mathcal{L}(\mathcal{X}) = \sum_i \|\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i\|^2$)

PCA and linear AutoEncoders

A linear AE is a simple structure

- ⋆ $\boldsymbol{W}_{\text{enc}}$ and $\boldsymbol{W}_{\text{dec}}$: encoder and decoder weights

$\boldsymbol{z}(\boldsymbol{x}_i) = \boldsymbol{W}_{\text{enc}}\boldsymbol{x}_i \qquad \tilde{\boldsymbol{x}}_i = \boldsymbol{W}_{\text{dec}}\boldsymbol{z}(\boldsymbol{x}_i)$



We optimize the weight matrices

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{W}_1,\boldsymbol{W}_2}{\operatorname{argmin}} \sum_{i=1}^{N} \frac{1}{2} \sum_{d=1}^{D} (\boldsymbol{x}_i(d) - \tilde{\boldsymbol{x}}_i(d))^2 = \underset{\text{rank}(\boldsymbol{W})=K}{\operatorname{argmin}} \|\boldsymbol{X} - \boldsymbol{W}\boldsymbol{Z}\|_F^2$$

Using again the Eckart and Young theorem with $\boldsymbol{X} = \boldsymbol{U}\Psi\boldsymbol{V}^\top$ then the solution is $\boldsymbol{W}_{\text{dec}}\boldsymbol{Z} = \boldsymbol{U}_K\Psi_K\boldsymbol{V}_K^\top$.

Setting $\boldsymbol{W}_{\text{dec}} = \boldsymbol{U}_K\Psi_K$, then clearly $\boldsymbol{W}_{\text{enc}} = \Psi_K^{-1}\boldsymbol{U}_K^\top$

$\boldsymbol{W}_{\text{enc}}\boldsymbol{X} = \boldsymbol{Z} = \boldsymbol{V}^\top = \boldsymbol{V}^\top(\boldsymbol{X}^\top\boldsymbol{X})^{-1}(\boldsymbol{X}^\top\boldsymbol{X}) = \boldsymbol{V}^\top(\boldsymbol{V}\Psi\Psi\boldsymbol{V}^\top)^{-1}(\boldsymbol{V}\Psi\boldsymbol{U}^\top)\boldsymbol{X} = \Psi_K^{-1}\boldsymbol{U}_K^\top\boldsymbol{X}$

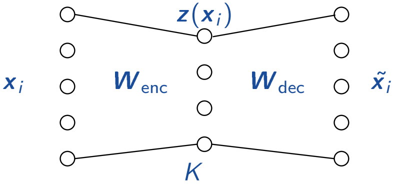
Note: $\boldsymbol{W}_{\text{dec}} = \boldsymbol{W}_{\text{dec}} = \mathbf{Id}_D$ cannot be a solution if $K < D$ and $\boldsymbol{W}_{\text{enc}}$ is not the inverse of $\boldsymbol{W}_{\text{dec}}$

PCA and linear AutoEncoders

A linear AE is a simple structure

- ⋆ $\boldsymbol{W}_{\text{enc}}$ and $\boldsymbol{W}_{\text{dec}}$: encoder and decoder weights

$\boldsymbol{z}(\boldsymbol{x}_i) = \boldsymbol{W}_{\text{enc}}\boldsymbol{x}_i \quad \tilde{\boldsymbol{x}}_i = \boldsymbol{W}_{\text{dec}}\boldsymbol{z}(\boldsymbol{x}_i)$



We optimize the weight matrices

$\boldsymbol{X} = \boldsymbol{U}\Psi\boldsymbol{V}^{\mathsf{T}}$ and $\boldsymbol{W}_{\text{dec}} = \boldsymbol{U}\Psi$ and $\boldsymbol{W}_{\text{enc}} = \Psi^{-1}\boldsymbol{U}^{\mathsf{T}} \Rightarrow \tilde{\boldsymbol{x}}_i = \boldsymbol{U}_K\boldsymbol{U}_K^{\mathsf{T}}\boldsymbol{x}_i$

Relation to PCA (centered data)

- ⋆ PCA: $\Sigma = \frac{1}{N}\boldsymbol{X}\boldsymbol{X}^{\mathsf{T}} = \boldsymbol{U}\Lambda\boldsymbol{U}^{\mathsf{T}}$ so that $\tilde{\boldsymbol{x}}_i = \boldsymbol{U}_K\boldsymbol{U}_K^{\mathsf{T}}\boldsymbol{x}_i$
- ⋆ AE: $\boldsymbol{X} = \boldsymbol{U}\Psi\boldsymbol{V}^{\mathsf{T}} \quad \Rightarrow \quad \Sigma = \frac{1}{N}\boldsymbol{X}\boldsymbol{X}^{\mathsf{T}} = \frac{1}{N}\boldsymbol{U}\Psi^2\boldsymbol{U}^{\mathsf{T}}$ and $\tilde{\boldsymbol{x}}_i = \boldsymbol{U}_K\boldsymbol{U}_K^{\mathsf{T}}\boldsymbol{x}_i$

$\Rightarrow \Lambda = \frac{1}{N}\Psi^2 = (\frac{1}{\sqrt{N}}\Psi)(\frac{1}{\sqrt{N}}\Psi)$ so that $\text{PCA}(\boldsymbol{X}) \leftrightarrow \text{AE}(\frac{1}{\sqrt{N}}\boldsymbol{X})$

$\Rightarrow$ A linear AE performs a PCA if the data is centered and scaled

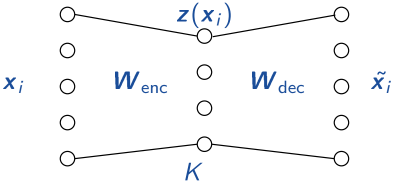
Note: $\boldsymbol{W}_{\text{dec}} = \boldsymbol{W}_{\text{dec}} = \mathbf{Id}_D$ cannot be a solution if $K < D$ and $\boldsymbol{W}_{\text{enc}}$ is not the inverse of $\boldsymbol{W}_{\text{dec}}$

PCA and linear AutoEncoders

A linear AE is a simple structure

- ⋆ $\boldsymbol{W}_{\text{enc}}$ and $\boldsymbol{W}_{\text{dec}}$: encoder and decoder weights

$\boldsymbol{z}(\boldsymbol{x}_i) = \boldsymbol{W}_{\text{enc}}\boldsymbol{x}_i \quad \tilde{\boldsymbol{x}}_i = \boldsymbol{W}_{\text{dec}}\boldsymbol{z}(\boldsymbol{x}_i)$



We optimize the weight matrices

$\Rightarrow$ A linear AE performs a PCA if the data is centered and scaled

- ⋆ If adding hidden layers, the latent space becomes non-linear

$\Rightarrow$ non-linear AutoEncoders

- ⋆ Changing the loss function (ELBO) enables sparsity in the latent space

$\Rightarrow$ Variational AutoEncoders (VAE)

Note: $\boldsymbol{W}_{\text{dec}} = \boldsymbol{W}_{\text{dec}} = \mathbf{Id}_D$ cannot be a solution if $K < D$ and $\boldsymbol{W}_{\text{enc}}$ is not the inverse of $\boldsymbol{W}_{\text{dec}}$

Alternative formulations

- ⋆ Probabilistic PCA reformulates PCA with an explicit latent distribution $\mathbb{P}(\boldsymbol{z}) = \mathcal{N}(0, \mathbf{Id}_K)$ and the conditional model is $\mathbb{P}(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{W}\boldsymbol{z} + \boldsymbol{\mu} + \sigma^2\mathbf{Id}_D)$ so that
  - the model accomodates "measurement noise" (with variance σ^2)
  - the model can run in a generative mode
  - parameters can be estimated via Maximum Likelihood
  - an EM algorithm (see later) can be derived for saving computations
  - Bayesian PCA reverts the conditional so as to find K by training
- ⋆ Kernel PCA embarks a nonlinear mapping $\phi(\boldsymbol{x}_i)$ via a kernel function $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^\mathsf{T}\phi(\boldsymbol{x}_j)$ to perform PCA within a more favorable space
- ⋆ Local PCA perform PCA on data neighborhoods to consider the local intrinsic dimensionality only

⚠ The limitation of PCA is often the decomposition of Σ in $O(D^3)$

Summary

- ⋆ PCA is part of the linear latent models
- ⋆ PCA applies on centered data and uses variance as a criterai for decomposition
- ⋆ PCA is an exact complete decomposition into decorrelated components
- ⋆ PCA assumes a Normal distribution of the data
- ⋆ PCA can be equivalently formulated as a regression
- ⋆ PCA offers a sound decimation strategy based on a low rank approximation
- ⋆ PCA can be used for denoising via a Gaussian noise model
- ⋆ PCA is equivalent to a linear AutoEncoder
- ⋆ PCA may be given a stochastic formulation
- ⋆ PCA may be generalized to the non-linear case via kernels

Example questions [mostly require formal – mathematical – answers]

- ⋆ Explain how PCA uses variance as a criterion
- ⋆ Show that variance maximization is equivalent to error minimization
- ⋆ Show how PCA uses the Eckart-Young theorem
- ⋆ Given some data, how do you apply PCA?
- ⋆ What information does it provide you with?
- ⋆ How do you reconstruct data with $K < D$ components?
- ⋆ How do you select the components to keep?
- ⋆ Can you apply PCA on any data?
- ⋆ Is it relevant to apply PCA on any data?
- ⋆ How can I apply PCA over clustered data?
- ⋆ Show that PCA is equivalent to a linear AE

Note: Make sure you can explain in detail what is: linear transform, orthogonal matrix, coordinate, rank, mean, variance, projection, eigen decomposition, trace, Frobenius norm, Lagrange Multiplier

References I

[1] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. (available online).

[2] Avrim Blum, John Hopcroft and Ravindran Kannan. *Foundations of Data Science*. Cambridge University Press, 2020. (available online).

[3] Richard O. Duda, Peter E. Hart and David G. Stork. *Pattern Classification*. Wiley, New York, 2 edition, 2001.