Fundaments of Machine learning for and with engineering applications

Enrico Riccardi[1]

Department of Mathematics and Physics, University of Stavanger (UiS).[1]

Sep 25, 2025

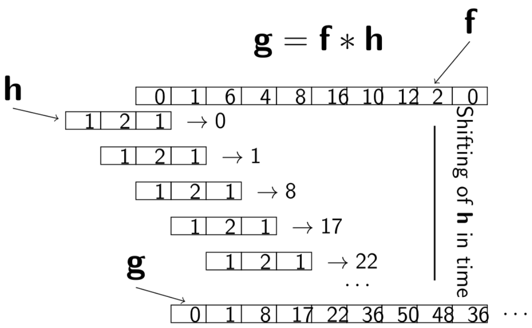
Filtering

Convolution

Convolution involves a window function that changes another function by *sliding* over it and performing local multiplications and additions. Depending on the shape of the convolution function we can perform

- Smoothings
- Deformations
- Differentiations

Convolution



Convolution

In general we can write the convolution between a function f and a convolving (deforming) function h as:

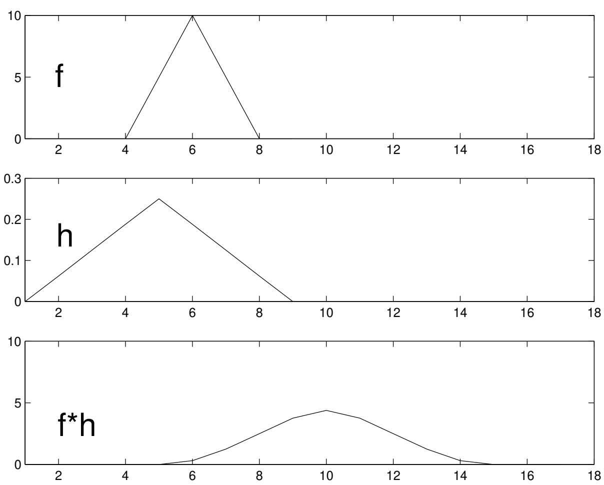
$$g(t) = \sum_{m=-\infty}^{\infty} f(m)h(m-t)$$

Often we use a more compact notation:

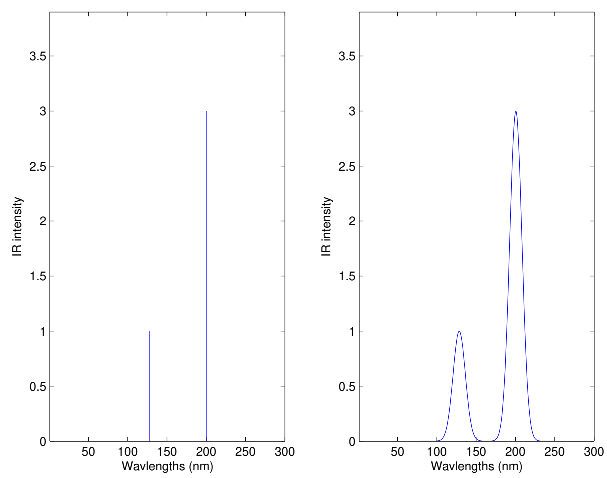
$$g(t) = f(t) * h(t) = h(t) * f(t)$$

where $*$ is the convolution operator

Convolution function



Peak broadening



Convolution operator properties

The convolution operator follows the distributive rule:

$$f_1(t) * [f_2(t) + f_3(t)] = f_1(t) * f_2(t) + f_1(t) * f_3(t)$$

It also follows the associative rule regarding order:

$$f_1(t) * [f_2(t) * f_3(t)] = [f_1(t) * f_2(t)] * f_3(t)$$

Convolution operator properties

Repeated convolutions

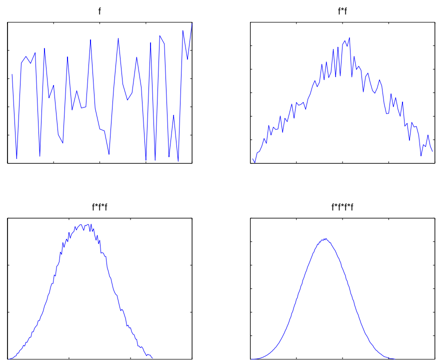
Take any function $g(t)$ and convolve it with any function $f(t)$ multiple times:

$$\begin{aligned} a_1(t) &= g(t) * f(t) \\ a_2(t) &= g(t) * a_1(t) \\ a_3(t) &= g(t) * a_2(t) \\ &\vdots \\ a_n(t) &\rightarrow \text{gaussian}(t) \end{aligned}$$

i.e. the result with always converge to a Gaussian function

Convolution properties

Any signal convolved with itself repeated many times will converge to a Gaussian function:



Mean Smooth operator

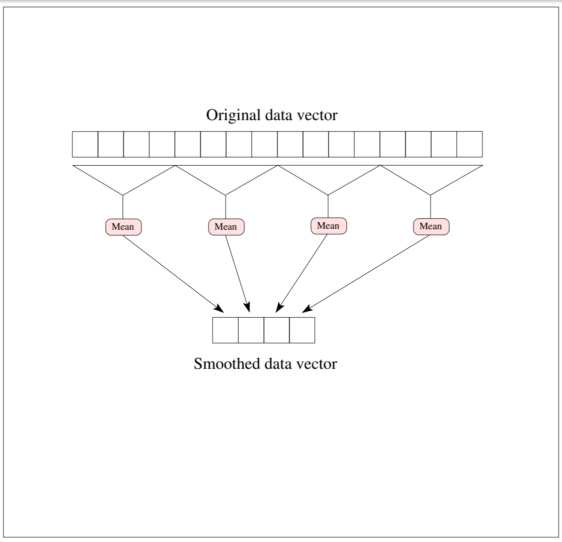
This is a very simple method which works as follows:

- Assume data vector x which contains n data points
- Start with the k first points, i.e. $[x_1, x_2, \cdots, x_k]$ and compute mean u_1 of these k points
- Take the next k points, $[x_{k+1}, x_{k+2}, \cdots, x_{2k}]$ and compute the mean u_2 of these k points
- Continue with this process until the data vector x is exhausted of points

There are two effects of this preprocessing:

- The new data vector u is of length approximately $1/k$'th of compared to the original
- Each element u_j has less noise due the cancelling effects of computing the mean

Mean Smooth operator



Running Average Smooth operator

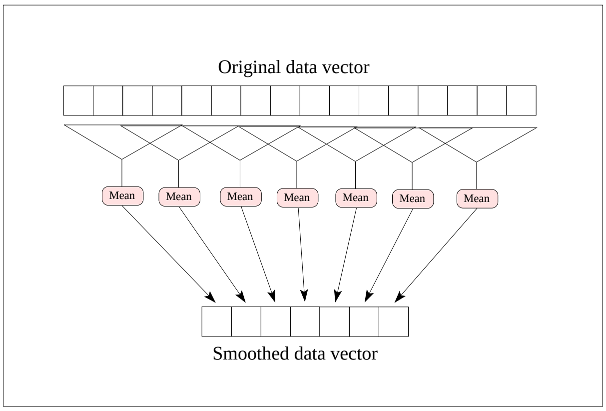
Let f be the original data profile and g the smooth version of this profile. Then we have:

$$g(i) = \sum_{j=-m}^{m} \frac{f(i+j)}{2m+1}$$

where m is the number of points in the window

Running Average Smooth operator

This is similar to the mean smoother but moves in shorter steps than the whole window length



Convolution or Moving average?

If we have a window with 3 points ($m = 1$) and we want to calculate the new value of point no. 5 in the original profile:

$$g(5) = [0 \cdot f(3) + 1 \cdot f(4) + 1 \cdot f(5) + 1 \cdot f(6) + 0 \cdot f(7)] \cdot \frac{1}{3}$$

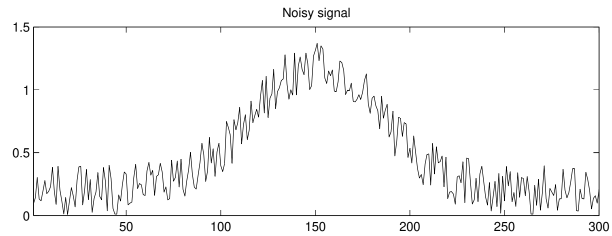
$$g(5) = \frac{1}{3}[0 \;\; 1 \;\; 1 \;\; 1 \;\; 0] \cdot [f(3) \;\; f(4) \;\; f(5) \;\; f(6) \;\; f(7)]^T$$

Convolution or Moving average?

A moving average IS the convolution between the vector **f** and a vector of ones (times a constant), i.e. :

$$\text{moving average} = \mathbf{f} * \mathbf{h} = \mathbf{f} * \frac{1}{n}[1 \;\; 1 \cdots 1 \;\; 1]$$

Moving average



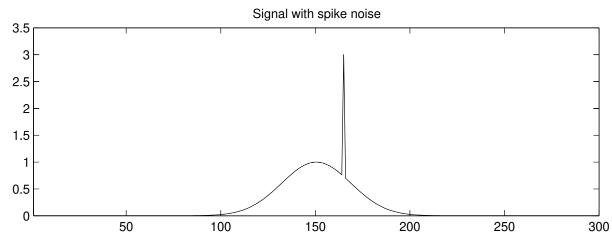
Moving average problems

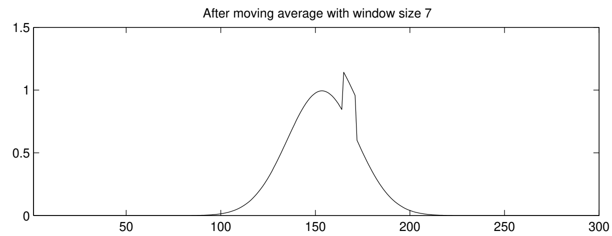
- Broadening of peaks
- *Spike-noise* affects the smoothed profile

A better alternative:

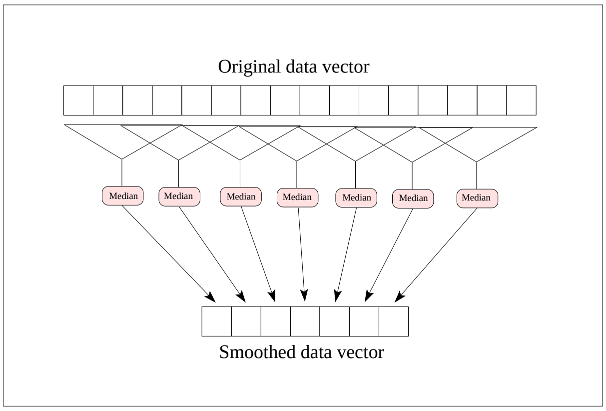
Use the median instead of the mean, then we don't have the problems with *spike-noise*. However, this filter is not linear

Running average example

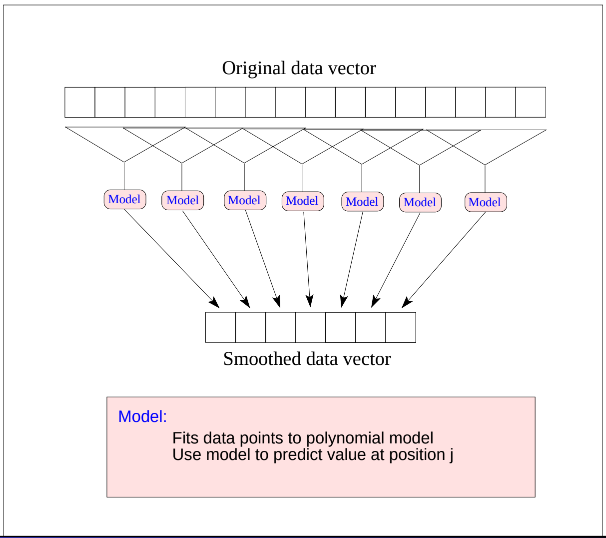




Running mediam

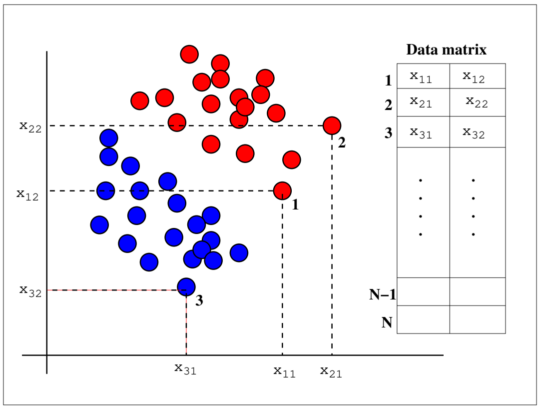


Polynomial smoothing



PCA

Central in any data analysis is the use of a *data matrix*



PCA base

Question:

How can we understand the information contained in such a data matrix?

- The geometry of the "object cloud" in N dimensions is used to understand relations

Geometrical insights

Plotting provides **geometrical insights** and observation of **hidden data structures** and **patterns**

Correlated variables

Problem

How can we use plotting if we have more than 3 variables?

Solution

1. Seek for **correlated variables** in the data matrix
2. Seek for latent variable
3. Give up (use AI)

Correlated variables

- Correlated variables contain approximately the same information.
- Several correlated variables suggests:
  - the same **phenomenon** is manifested in different way
  - an **underlying phenomenon** more fundamental exists

Let's assume the latter:

Linear combinations

Assuming the latent variables to be a **linear combinations** of the original variables, i.e.:

$$LV = a_1x_1 + a_2x_2 + \cdots + a_nx_n$$

Latent variables

A new coordinate system

1. Latent variables are based on creating a new coordinate system based on linear combination of the original variables
2. Objects are projected from a higher dimensional

data space onto this new (lower dimensional) coordinate system

1. The new coordinate system improves interpretation and prediction.

Principal component analysis (PCA) can automatically create useful latent variables

PCA

Originally invented in 1901 by Karl Pearson and re-invented several times. The PCA method is also referred to as:

1. Singular value decomposition (numerical analysis)
2. Karhunen-Loeve expansion (electric engineering)
3. Eigenvector analysis (physical sciences)
4. Hotelling transform (image analysis/statistics)
5. Correspondence analysis (double scaled version of PCA)

Karl Pearson

PCA

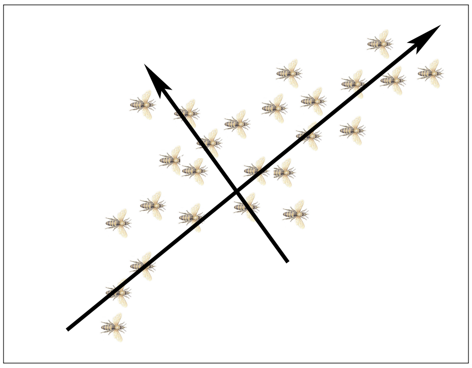
Goals of PCA:

1. Simplification.
2. Data reduction and data compression
3. Modeling
4. Outlier detection
5. Variable selection
6. Classification
7. Prediction
8. ... world peace ...

PCA

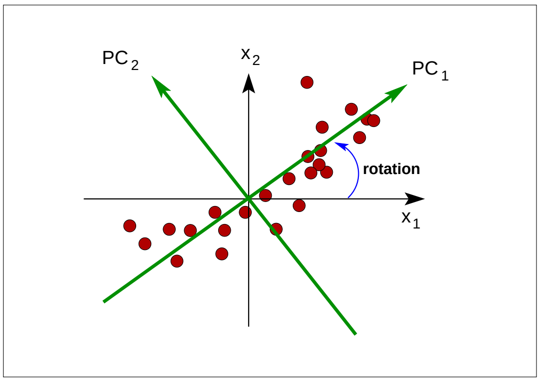
Rotation of the coordinate system

In PCA the original coordinate system is rotated such that the new latent variable axes point in the direction of max variance



PCA

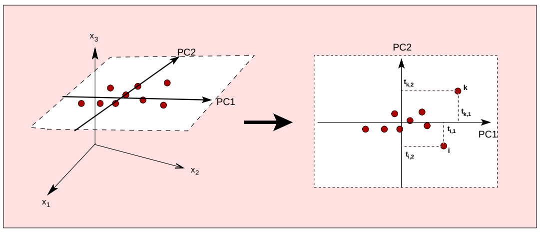
Rotation of the coordinate system



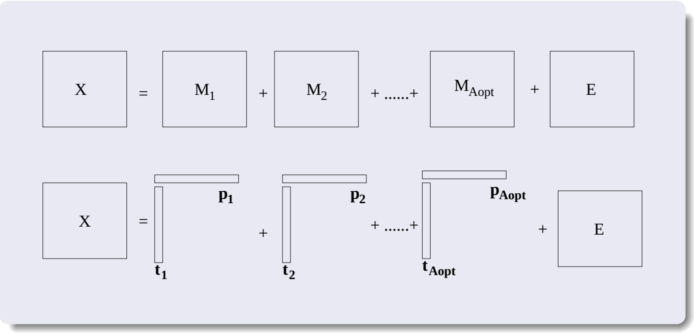
PCA

Scores are new coordinates

Scores are the coordinates of objects in the **new** coordinate system



PCA Model



Mapping the variables

- The loadings are the weights needed to define the latent variable
- The latent variable is a linear combination of the original variables:

$$t_i = \sum_{j=1}^{M} p_j x_j$$

where p_j is the loading associated with variable x_j and t_i is the score for object i for one principal component.

- The loading determines how much influence this variable has in the construction of the latent variable
- Loading plots shows the importance of variables

Model description

The PCA model

$$\mathbf{X} = \mathbf{T}\mathbf{P}^T + \mathbf{E}$$

where

- $\mathbf{X}$ is the data matrix
- $\mathbf{T}$ is the scores matrix
- $\mathbf{P}$ is the loadings matrix
- $\mathbf{E}$ is the residual matrix

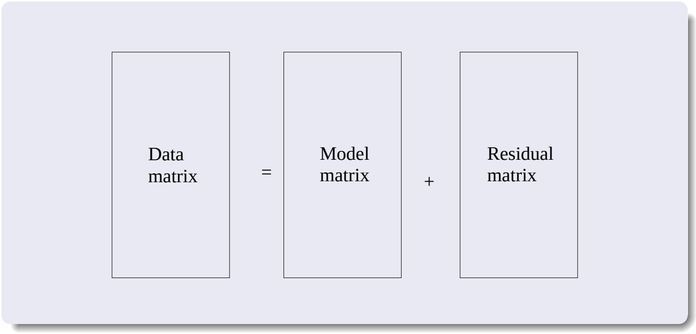
PCA is an example of a **bilinear model** where a matrix $\mathbf{Z}$ is written as a product of two others:

$\mathbf{Z} = \mathbf{AB}$

Data = Model + Noise

Using a PCA model of the data is very powerful, but there is a price to pay: Loss of information. As we do not use all the original dimensions there will usually be information that cannot be represented using a very small number of principal components.

The projection is an *approximation* of the original data and we talk about the *residuals* of the variance not explained by the model.



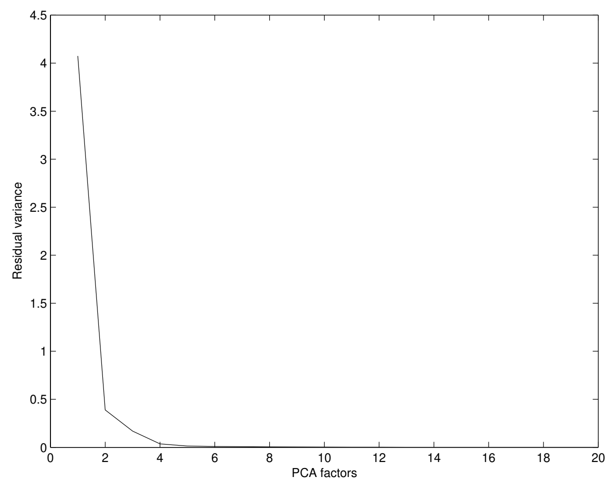
PCA sorting

The PC's are sorted according to variance

1. The new latent variables are **sorted with respect to how much variance they explain**. This means the first component explains the most, followed by no.2 etc.
2. Only the first set of components are actually used
3. The remaining last K components are related to noise (or are zero).

The contribution by each PC can be seen from the **residual variance plot**

Residual variance plot



Derication of PCA

Defining PCA latent variables

We find latent vectors in $\mathbf{X}$ which have maximum relevance for $\mathbf{X}$. We can formulate this as we find a vector $\mathbf{t}$ in column space of $\mathbf{X}$:

$$\mathbf{t} = \mathbf{X}\mathbf{p}$$

such that the squared variance of $\mathbf{t}$ is *maximized*:

$$\max\left[\mathbf{t}^T\mathbf{t}\right] = \max(\mathbf{p}^T\mathbf{X}^T\mathbf{X}\mathbf{p})$$

for $|\mathbf{p}| = 1$. $\mathbf{p}$ is the first principal component.

Derivation of PCA

Setting up the problem in terms of a Lagrange multiplier λ:

$$f(\mathbf{p}, \lambda) = \mathbf{p}^T\mathbf{X}^T\mathbf{X}\mathbf{p} - \lambda(\mathbf{p}^T\mathbf{p} - 1)$$

where $\mathbf{p}^T\mathbf{X}^T\mathbf{X}\mathbf{p} = \mathbf{t}^T\mathbf{t}$. Then we differentiate f with respect to $\mathbf{p}, \lambda$:

$$\begin{aligned}\frac{\partial f}{\partial \mathbf{p}} &= 2\mathbf{X}^T\mathbf{X}\mathbf{p} - 2\lambda\mathbf{p} = 0 \\ \frac{\partial f}{\partial \lambda} &= \mathbf{p}^T\mathbf{p} - 1 = 0\end{aligned}$$

Derivation of PCA

$$\mathbf{X}^T\mathbf{X}\mathbf{p} = \lambda\mathbf{p}$$

i.e the first principal component is the eigenvector to the covariance matrix $\mathbf{X}^T\mathbf{X}$. The next components will all be orthogonal to each other. We can also determine the value of λ:

$$\begin{aligned}\mathbf{p}^T\mathbf{X}^T\mathbf{X} &= \lambda\mathbf{p}^T \\ \mathbf{p}^T\mathbf{X}^T\mathbf{X}\mathbf{p} &= \lambda\mathbf{p}^T\mathbf{p} = \lambda \\ \mathbf{t}^T\mathbf{t} &= \lambda\end{aligned}$$

Derivation of PCA

To find the next loadings vector $\mathbf{p}_2$, assuming $\mathbf{p} = \mathbf{p}_1$ in previous slides and correspondingly $\mathbf{t} = \mathbf{t}_1$, we have the constraints that $\mathbf{t}_1^T\mathbf{t}_2 = 0$, $\mathbf{p}_1^T\mathbf{p}_2 = 0$ and $\mathbf{p}_2^T\mathbf{p}_2 = 1$. We have that

$$\begin{aligned}\mathbf{t}_1^T\mathbf{t}_2 &= \mathbf{p}_1^T\mathbf{X}^T\mathbf{X}\mathbf{p}_2 \\ &= \lambda_1\ \mathbf{p}_1^T\mathbf{p}_2 = 0\end{aligned}$$

This gives the Lagrange equation: $f(\mathbf{p}_2, \lambda_2, \phi) = \mathbf{p}_2\mathbf{X}^T\mathbf{X}\mathbf{p}_2 - \lambda_2(\mathbf{p}_2^T\mathbf{p}_2 - 1) - \phi\mathbf{p}_1^T\mathbf{p}_1$. The solution is that $\mathbf{p}_2$ is the **second largest** eigenvector of $\mathbf{X}^T\mathbf{X}$ and variance λ_2 which is the second largest eigen value. Similar results holds for the k'th loading vector.