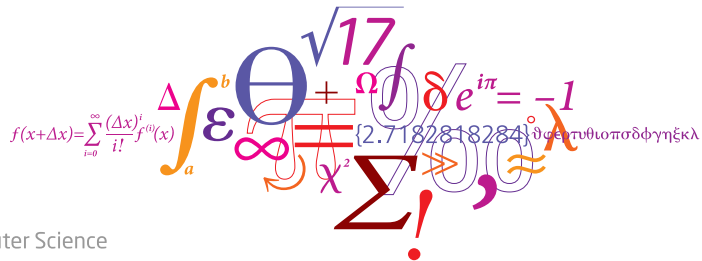
02471 Machine Learning for Signal Processing

Dictionary learning and source separation

Tommy Sonne Alstrøm

Cognitive Systems section



DTU Compute
Department of Applied Mathematics and Computer Science

Outline

- Last week review
- Dictionary learning – applications
- Source separation – applications
- Relation to earlier topics
- Dictionary learning
  - k-SVD
  - Non-negative Matrix Factorization (NMF)
- Independent Component Analysis (ICA)
- Next week

Material: 2.5, 19.1–19.3, 19.5–19.7

The story so far and what the future holds

- Problem set 2 re-submissions due Sunday 10/11 at 23.59.
- Problem set 3 is available and is due 15/12 23.59, and counts 20% towards the final grade.

What you have learned so far:

- Parameter estimation [L2 regularization, biased estimation, mean squared error minimization]. L1 regularization Todo: Bayesian parameter estimation (next week).
- Filtering signals [Stochastic processes, correlation functions, Wiener filter, linear prediction, adaptive filtering using stochastic gradient decent (LMS, APA/NLMS), adaptive filtering using regularization (RLS)].

Sparse signal representations and dictionary learning:

- Signal representations [Time frequency analysis, sparsity aware learning, Todo: factor models].
- Sparsity aware sensing (lasso, sparse priors), compressed sensing, Todo: dictionary learning [Independent component analysis, Non-negative matrix factorization, k-SVD].

Last week review

Linear signal representations

A linear signal representation model can be thought of as

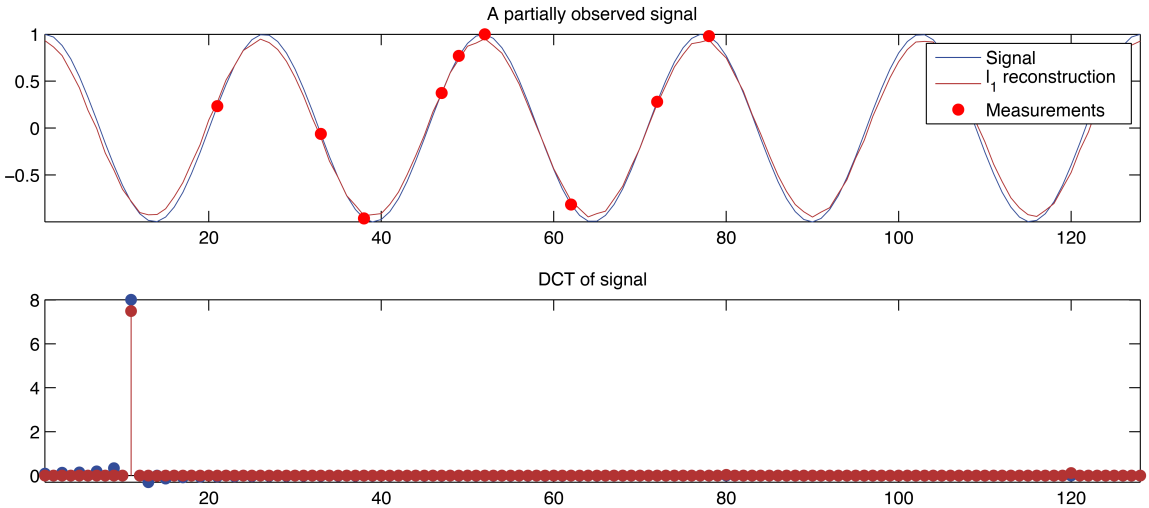
Linear signal representations

$$\tilde{\boldsymbol{s}} = \Phi^H \boldsymbol{s} \qquad \text{analysis}$$
$$\boldsymbol{s} = \Phi \tilde{\boldsymbol{s}} \qquad \text{synthetis}$$

- $\boldsymbol{s}$ is the vector of raw samples.
- $\tilde{\boldsymbol{s}}$ is the transformed vector.
- Φ is the unitary transformation matrix, $\Phi\Phi^H = I$.

There are many choices of matrices, we can choose Φ^H as a matrix of fourier coefficients (complex matrix), the DCT matrix (which is real), or a wavelet matrix.

Signal reconstruction using ℓ_1



Orthogonal Matching Pursuit (OMP)

The OMP algorithm – algorithm 10.1 in the book

- **Initialize**
  - $\boldsymbol{\theta}^{(0)} = \mathbf{0} \in \mathbb{R}^l$.
  - $S^{(0)} = \emptyset$.
  - $\boldsymbol{e}^{(0)} = \boldsymbol{y}$.
- **For** $i = 1, \cdots, k$ **Do**
  - Select the column in X that forms the smallest angle with the error.
  - Update the indices of active vectors, $S^{(i)}$.
  - Update the parameter vector $\boldsymbol{\theta}^{(i)}$ using least squares using the columns in X indexed by $S^{(i)}$.
  - Update the error vector.
- **End For**

Parameters:

k is the number of non–zero components (must be smaller than the number of observations)

Iterative Shrinkage/thresholding (IST)

The naive IST formula (10.3)–(10.7) in the book (estimates the LASSO solution)

- **Initialize**
  - $\boldsymbol{\theta}^{(0)} = \mathbf{0} \in \mathbb{R}^l$.
  - Select the value of μ
  - Select the value of λ
- **For** $i = 1, \cdots$ **Do**
  - $\boldsymbol{e}^{(i-1)} = \boldsymbol{y} - X\boldsymbol{\theta}^{(i-1)}$
  - $\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta}^{(i-1)} + \mu X^T \boldsymbol{e}^{(i-1)}$
  - $\boldsymbol{\theta}^{(i)} = \text{sign}\left(\tilde{\boldsymbol{\theta}}\right) \max\left(\left|\tilde{\boldsymbol{\theta}}\right| - \lambda\mu, 0\right)$
- **End For**

Parameters:

μ is still the step size, but also affects the shrinkage.

λ is the regularization parameter.

The spectrogram

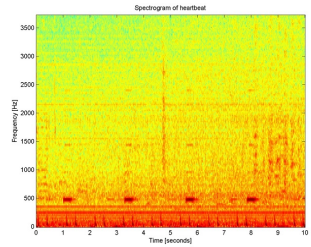
STFT

$$X(n,k) = \sum_{m=-\infty}^{\infty} x(m)w(m-n)e^{-j\frac{2\pi}{N}kn}$$

The spectrogram

The magnitude spectrum computed using STFT, ie $|X(n,k)|$.

Two important parameters; the block size B, and the hop size S.

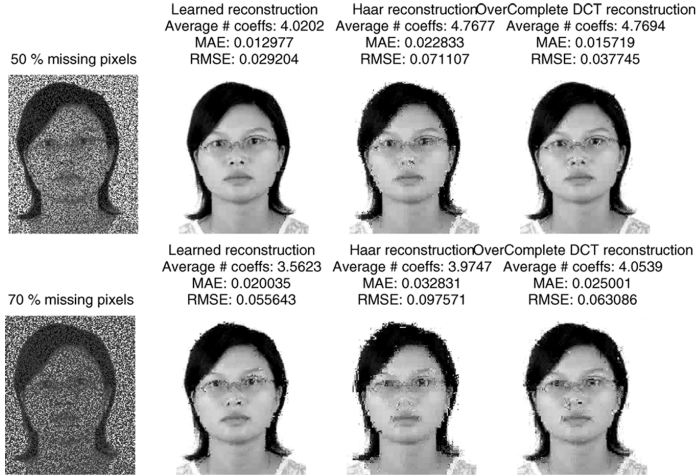


Lecture summary

- Two algorithms were presented, OMP and IST. And they solve ℓ_0 and ℓ_1 respectively.
  - OMP is used e.g. in k-SVD (today).
- This lead to linear signal representation models.
- If the signal is not stationary, the representation models can be applied on smaller chunks of the signal. This approach is called "time-frequency analysis".
- The models are designed using domain knowledge. If we want to learn the models from data, in machine learning, we call it dictionary learning, or source separation.

Dictionary learning – applications

Image denoising

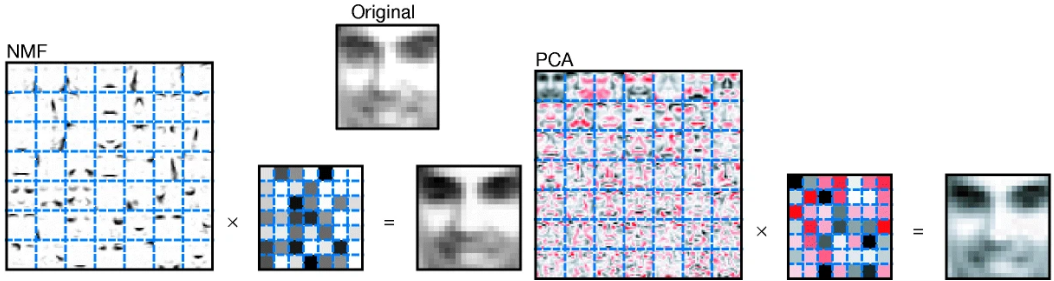


Michal Aharon, Michael Elad, and Alfred Bruckstein, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation", 2006.

Dictionary learning – applications

Prof. Saprio, Duke University

Image decomposition



Learning the parts of objects by non-negative matrix factorization, 1999 Daniel D. Lee and H. Sebastian Seung

Spectral denoising

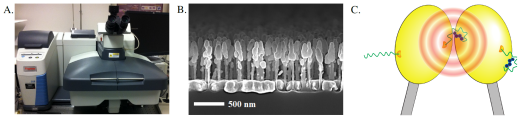


Fig. 1. A. The DXR™ Raman Microscope used to collect data. B. A side-view up of a Raman substrate depicting the nanopillars, courtesy of Kaiyu Wu, DTU. C. Illustration of the principle behind the SERS substrates. The two left pillars have molecules on them but in order to get the improved SNR the molecule needs to be captured in the hot spot as shown on the right. This is achieved by leaning the pillars through solvent evaporation.

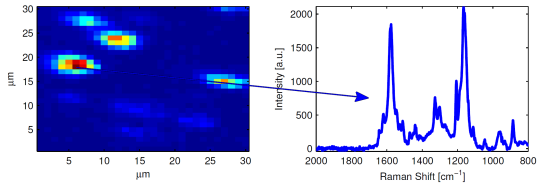
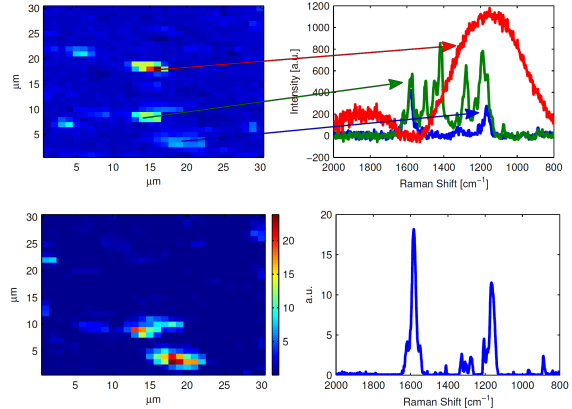
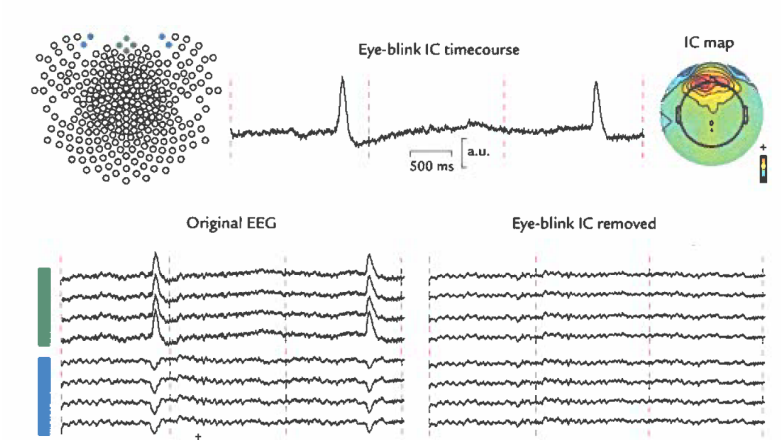


Fig. 2. Illustration of an uncontaminated SERS measurement using a Raman map at 1166 cm^{-1} (left) and an example spectrum for Estradiol Glow (right). Hot spots containing molecules are readily identified on the Raman map as the red areas. On the spectrum the peaks at 1166 cm^{-1} and 1580 cm^{-1} are considered the major discriminative.



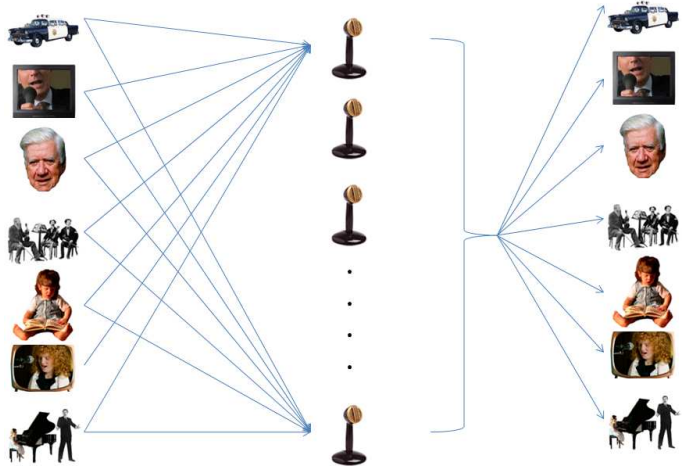
Alstrøm et al. Improving the robustness of surface enhanced Raman spectroscopy based sensors by Bayesian non-negative matrix factorization, 2014.

Source separation in EEG



"Combining EEG and eye tracking: identification, characterization, and correction of eye movement artifacts in electroencephalographic data", 2012, Michael Plöchl, José P. Ossandón, and Peter König.

Source separation in audio



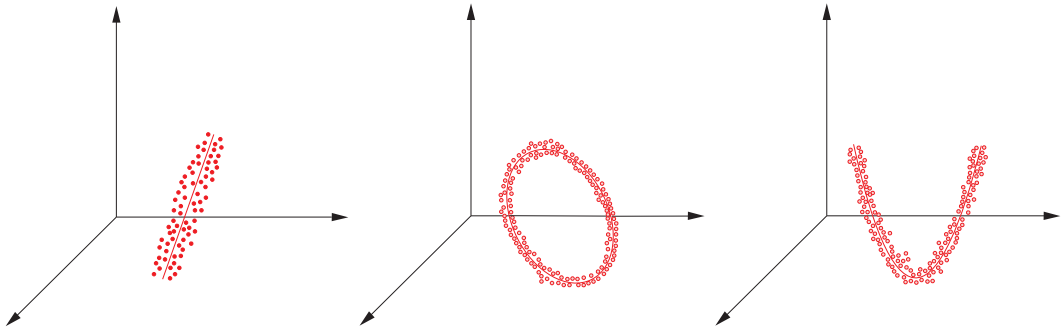
Summary

- Blind source separation and dictionary learning are closely related, it only depends on the constraints enforced on the model.
- Dictionary learning is used to learn a dictionary of the data, that can later be used to encode the data.
- Blind source separation seeks to unmix the data, because the source recovery is the task.
- Has many applications in both machine learning and signal processing.
- An active research topic.

Relation to earlier topics

Relation to sparsity aware–learning

We want to identify good representations for objects



A good representation for a circle

$$\mathbf{x} = [r\cos\theta, r\sin\theta]^T$$

The data is one-dimensional under this representation (intrinsic dimensionality).

The linear factor model

Factor model

$$\mathbf{x} = A\mathbf{z}$$

if we observe N measurements, we get
$X := [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N]$ $(l \times n)$, A $(l \times m)$, $Z := [\boldsymbol{z}_1, \cdots \boldsymbol{z}_n]$ $(m \times n)$:

$$X = AZ$$

We have only observed X, so to estimate A and Z, we need additional constraints.

ICA, NMF, PCA and k-SVD are methods that use the above model, but impose different constraints in order to enforce the desired structure on A and Z.

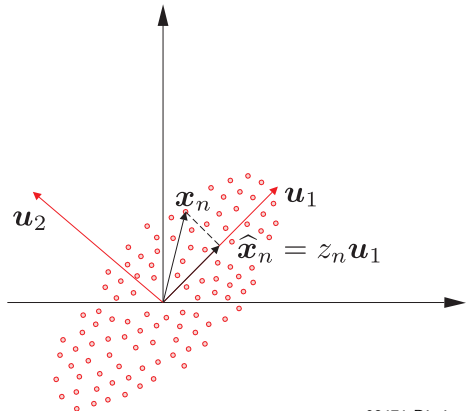
Traditionally, digital signal processing seeks to design A (e.g. DCT), machine learning seeks to learn A.

PCA

PCA will scale and rotate the coordinate system. We have the model

$$\mathbf{z} = U^T \boldsymbol{x}$$

Where U^T is the projection matrix, i.e $\mathbf{x}$ is projected unto the space spanned by U, and $\mathbf{z}$ is the coordinate in that new subspace.



Dictionary learning

Towards dictionary learning

The k-SVD model

$$X = AZ$$
$$X \in \mathbb{R}^{l \times n}$$
$$A \in \mathbb{R}^{l \times m}$$
$$Z \in \mathbb{R}^{m \times n}$$
$$m \gg l$$

The k-SVD optimization problem

$$\hat{A}, \hat{Z} = \arg\min_{A,Z} \|X - AZ\|_F^2$$

s.t. $\|\boldsymbol{z}_n\|_0 \leq T_o$

$$\|E\|_F = \sqrt{\sum_{i=1}^{l} \sum_{j=1}^{n} |e_{ij}|^2}$$ Frobenius norm of a matrix

k-SVD update rules

Step 1, sparse coding (membership update)

$$\hat{z}_n = \arg\min_{\boldsymbol{z}_n} \|\boldsymbol{x}_n - A\boldsymbol{z}_n\|^2, \quad \forall n$$
$$\text{s.t.} \quad \|\boldsymbol{z}_n\|_0 \leq T_o$$

Step 2, cookbook/dictionary update (update vectors)

For the k'th column in A, $\forall k$,

1. Identify non-zeros in row $Z_{k,:}$ and denote that set K.
2. Perform PCA/SVD on $X_{K,:}$
3. Set the k'th column in A to the principal component 1.

Is this similar to something you know?

The non-negative matrix factorization

Non-negative matrix factorization model

$$X = AZ$$
$$X \in \mathbb{R}^{l \times n}$$
$$A \in \mathbb{R}_{+}^{l \times m}$$
$$Z \in \mathbb{R}_{+}^{m \times n}$$

Model implications: models the superposition principle well.

X can be allowed to be "a little" negative, since we usually allow white noise residuals.

Extension:

- Put sparsity constraints on A and/or D.
- Put smoothness constraints on A, e.g. if A contains a smooth spectrum.

Good overview: The Why and How of Non-negative Matrix Factorization, 2014, Nicolas Gillis.

The ICA model

The ICA model

$$\mathbf{x} = A\mathbf{z}$$
$$X \in \mathbb{R}^{l \times n}$$
$$A \in \mathbb{R}^{l \times l}$$
$$Z \in \mathbb{R}^{l \times n}$$

The ICA optimization problem

Identify A, but with respect to maximize statistical independence on the random variables in $\mathbf{z}$

But is this the task of PCA?

PCA vs ICA

PCA identifies components such that z_i and z_j are uncorrelated ($i \neq j$), i.e (if zero-mean):

$$\mathbb{E}[\mathrm{z}_i \mathrm{z}_j] = 0$$

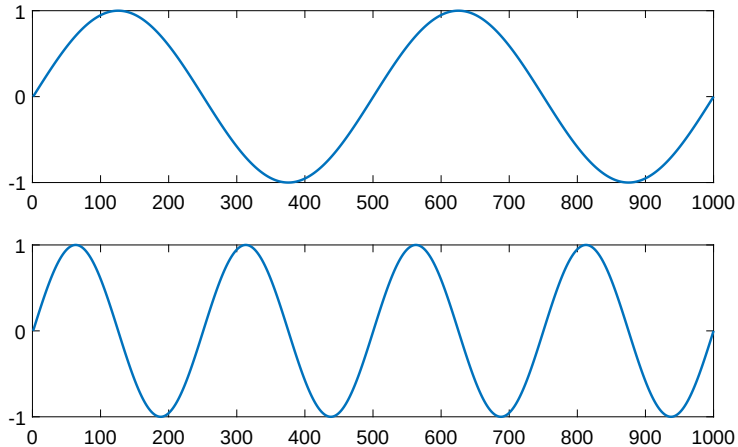
ICA identifies component such that z_i and z_j are statistical independent ($i \neq j$) , i.e

$$p(\mathrm{z}_i, \mathrm{z}_j) = p(\mathrm{z}_i) p(\mathrm{z}_j)$$

For Gaussian variables (if z is Gaussian), these statements are identical, see eq. (2.79)–(2.80).

Example

Consider the following problem:

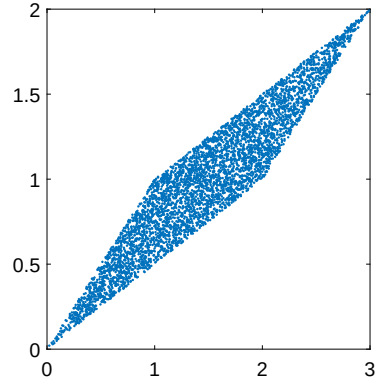


Signals are uncorrelated (covariance is zero), but not independent.

Example 2

Consider the model

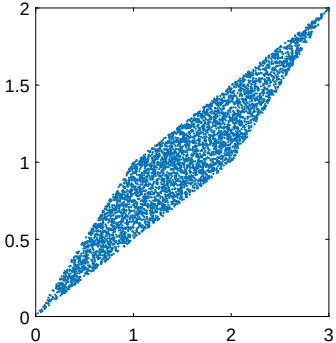
$$
\begin{aligned}
\mathrm{s}_1 &\sim U(0,1) \\
\mathrm{s}_2 &\sim U(0,1) \\
\mathbf{x} &= \begin{bmatrix} 1 & 1 \\ 2 & 1 \end{bmatrix} \cdot [\mathrm{s}_1 \ \mathrm{s}_2]^T
\end{aligned}
$$



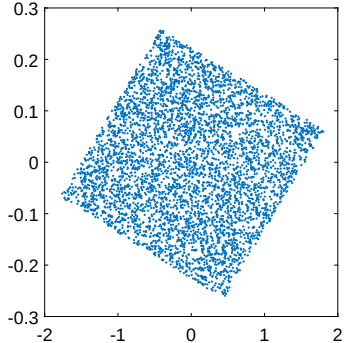
What would PCA produce?

Example 2

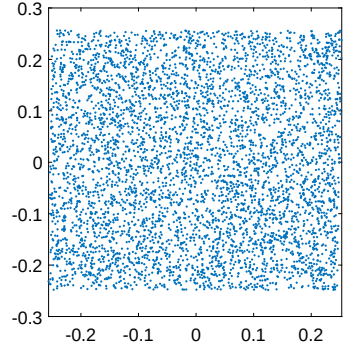
Observations



PCA



ICA



ICA disclaimer

ICA is a huge topic, there are books dedicated to only this. E.g "Independent Component Analysis by Hyvärinen, Aapo; Karhunen, Juha; Oja, Erkki", free at `https://findit.dtu.dk/en/catalog/2304962533`

For real applications, use FastICA library to perform ICA.

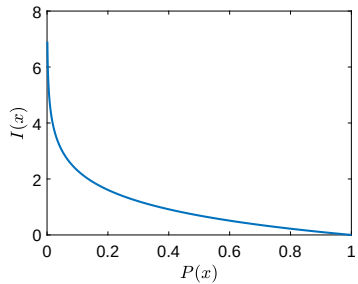
Information

Information of event (discrete variable)

The information for a particular value x for a discrete random variable x is

$$I(x) := -\log P(x)$$

The information measures the amount of "surprise", e.g. sun in Sahara is expected thus this message has very low information, but water in Sahara is unexpected, thus this message has high information.



Mutual information

The information provided by the occurrence of event y about event x is called mutual information

Mutual information of events (discrete variable)

$$I(x;y) := \log \frac{P(x|y)}{P(x)}$$

Work through example 2.6 on your own.

Average mutual information

Average mutual information of random variables (discrete case)

$$I(\mathrm{x};\mathrm{y}) := \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} P(x,y)\log\frac{P(x,y)}{P(x)P(y)}$$

The mutual information is a rewrite of $I(\mathrm{x};\mathrm{y}) := \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} P(x,y)I(x;y)$.

Average mutual information of random variables (continuous case)

$$I(\mathrm{x};\mathrm{y}) := \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p(x,y)\log\frac{p(x,y)}{p(x)p(y)}dxdy$$

How?

One approach is to minimize the mutual information

$$I(\mathrm{x};\mathrm{y}) := \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p(x,y)\log\frac{p(x,y)}{p(x)p(y)}dxdy$$

It can be shown that $I(\mathrm{x},\mathrm{y}) \geq 0$ and $I(\mathrm{x},\mathrm{y}) = 0$ implies that x and y are statistically independent.

If $\mathbf{s} = [\mathrm{x}\,\mathrm{y}]^{\mathbf{T}}$ is our unobserved sources, estimate W such that we get sources that has minimum mutual information.

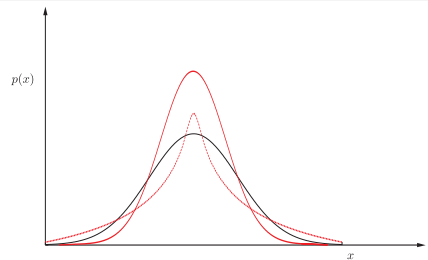
ICA update

ICA based on mutual information and natural gradient descent

1. Randomly initialize W (W is the unmixing matrix and estimates A^{-1}; model is $X = AZ$).
2. Update rule: $W^{(i)} = W^{(i-1)} + \mu_i \left(I - \mathbb{E}\left[\boldsymbol{\phi}(\mathbf{z})\mathbf{z}^T\right]\right) W^{(i-1)}$

Choose $\phi(\mathbf{z})$ as

- $\phi(\mathbf{z}) = 2\tanh(\mathbf{z})$ if $\mathbf{z}$ is super-Gaussian (more spiky, longer tails).
- $\phi(\mathbf{z}) = \mathbf{z} - \tanh(\mathbf{z})$ if $\mathbf{z}$ is sub-Gaussian (faster tail decay).



Lecture summary

PCA, ICA, k-SVD, NMF - which should I choose ??? Depends on what you want to do:

- PCA obtains uncorrelated features, and great for feature extraction and dimensionality reduction.
- ICA obtains maximal independence but does not reduce dimensionality. Provides a sparser output and is perceptually more relevant.
    - There are more than one way to arrive at the ICA solution. We choose mutual information.
- NMF is best for analysis of non-negative data, e.g. pixels, energies, count data, etc, and often provides interpretable results (for non-negative data).
- k-SVD is best for sparse coding if a compact sparse dictionary is wanted with direct control on sparsity, but often is not very interpretable.

Next week

Week 46 material; ML 11.2, 12.1–12.2, 12.4–12.5, 12.10 (online).

- Bayesian linear regression.
- Expectation Maximization.