Independent Component Analysis
and
Kernel Independent Component Analysis

Advanced Signal Processing 1, 2009

Lukas Pfeifenberger

l.pfeifenberger@commend.com

Motivation

- Feature extraction
  - Find a common structure in data
- Dimensionality reduction
  - Represent the most important part of the data in a lower dim
- Blind source separation
  - Separate data without prior knowledge of the mixing process
- Signal processing in feature space
  - Artifact removal from data

Outline

- Unsupervised Learning
- Basic ICA
  - Matlab examples
- Applications
  - Matlab examples
- Kernel ICA
  - Matlab examples
- Conclusion
- Literature

Unsupervised learning

- Is a descriptive model
  - Goal:
    - Finds structure in the data (feature extraction)
    - Dimensionality reduction with minimal information loss
  - Problems:
    - Separability should be maximized
    - Number of necessary features should be minimized
    - No means of a perfect solution (unsupervised)

Unsupervised learning

- Represents the data in feature space
  - Feature space can be
    - Higher-dimensional to find a hyperplane for separation
    - Lower-dimensional to omit unnecesary information
  - Different properties can be optimized:
    - PCA (maximal variance, uncorrelated, orthogonal data)
    - CCA (maximal correlation)
    - ICA (maximal non-gaussianity, independent data)

PCA vs. ICA

- Coordinate transform to an orthogonal basis where the data is uncorrelated
- Dimensionality reduction that preserves maximum variance and minimizes the mse

$$y_1 = \sum_{k=1}^{n} w_{k1} x_k = \mathbf{w}_1^T \mathbf{x}$$

- y1 is a pc of x, if its variance $\mathrm{E}\{y_1^2\}$ is maximal
- → find a weight vector w that maximizes the variance
  - Constraint: $\|\mathbf{w}_1\| = 1$

PCA vs. ICA

- Maximizing variance:

$$\sigma_{\hat{\mathbf{w}}}^2 = E[(\mathbf{x}^T\hat{\mathbf{w}})^2] = \hat{\mathbf{w}}^T E[\mathbf{x}\mathbf{x}^T]\hat{\mathbf{w}} = \frac{\mathbf{w}^T\mathbf{C}\mathbf{w}}{\mathbf{w}^T\mathbf{w}}$$

$$\frac{\partial\sigma_{\hat{\mathbf{w}}}^2}{\partial\mathbf{w}} = \frac{2}{\mathbf{w}^T\mathbf{w}}(\mathbf{C}\mathbf{w} - \sigma_{\hat{\mathbf{w}}}^2\mathbf{w}) = 0 \quad \Rightarrow \quad \mathbf{C}\mathbf{w} = \sigma_{\hat{\mathbf{w}}}^2\mathbf{w}$$

- Problem of finding the eigenvalues of C:

$$\mathbf{C} = \sum_i \lambda_i \mathbf{e}_i \mathbf{e}_i^T$$

PCA vs. ICA

- Maximizing variance:
  - C has n orthogonal eigenvectors E:

$$E\{\mathbf{x}\mathbf{x}^T\} = \mathbf{E}\mathbf{D}\mathbf{E}^T$$

  - Eigenvalues as diagonal matrix D:

$$\mathbf{D} = \mathbf{E}^T E\{\mathbf{x}\mathbf{x}^T\}\mathbf{E} = E\{\mathbf{E}^T\mathbf{x}\mathbf{x}^T\mathbf{E}\} = E\{\mathbf{z}\mathbf{z}^T\}$$

  - Eigenvectors are sorted after descending eigenvalues D
  - This also minimizes the mse when discarding the PCs with the smallest eigenvalues:

$$J_{MSE}^{PCA} = \mathrm{E}\{\|\mathbf{x}\|^2\} - \mathrm{E}\{\sum_{j=1}^{m}(\mathbf{w}_j^T\mathbf{x})^2\}$$

PCA vs. ICA

- PCA algorithms:
  - Closed-form: decomposition of the cov matrix C in eigenvectors E and eigenvalues D
  - on-line: calculates a PC at a time
    - Stochastic gradient descent
    - Recursive least squares

PCA vs. ICA

- Properties:
  - Dimensionality reduction of input vector x
    - Preprocessing step to ICA if the mixing matrix A is rectangular or noisy components need to be removed
  - The eigenvalues of C are an orthogonal basis for the input x
    - z is uncorrelated, scaling it to unit variance makes it white. Whitening is an important preprocessing step to ICA.
    - Whitening means: $\mathrm{cov}(z) = I$

Whitening

- Whitening of x
  - Find a whitening matrix V that spheres x:

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

$$\mathrm{E}\{\mathbf{z}\mathbf{z}^T\} = \mathbf{I}$$

  - V can be obtained from the eigenvalues and –vectors of C

$$\mathbf{V} = \mathbf{D}^{-1/2}\mathbf{E}^T$$

  - (blackboard)

Whitening

- Any arbitrary matrix UV is can be a whitening matrix, as long as U is orthogonal:

$$\mathrm{E}\{\mathbf{z}\mathbf{z}^T\} = \mathbf{U}\mathbf{V}\mathrm{E}\{\mathbf{x}\mathbf{x}^T\}\mathbf{V}^T\mathbf{U}^T = \mathbf{U}\mathbf{I}\mathbf{U}^T = \mathbf{I}$$

- Another example is the inverse square root of the cov matrix:

$$\mathbf{C}_\mathbf{x}^{-1/2} = \mathbf{E}\mathbf{D}^{-1/2}\mathbf{E}^T$$

- Whitening = decorrelation + scaling

Basic ICA

- PCA does only half the trick:
  - PCA yields uncorrelated components
- ICA extends PCA:
  - ICA yields uncorrelated and independent components
  - PCA can be used as preprocessing for whitening and dimensionality reduction

Basic ICA

- Basic ICA model
  - Unknown sources s
  - Unknown mixing matrix A
  - Observed mixtures x

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

  - Goal: estimate A and its inverted matrix only from the PDF of x

Basic ICA

- Model restrictions
  - s is assumed to be statistically independent

$$p(y_1, y_2, ..., y_n) = p_1(y_1)p_2(y_2)...p_n(y_n)$$

    - This holds for most BSS situations
  - The independent components s must not have a gaussian distribution up to one (see later)

Basic ICA

- Model restrictions
  - A must be invertible so that its inverse B exists:

$$\mathbf{s} = \mathbf{B}\mathbf{x}$$

  - The mixing matrix A cannot model:
    - uncorrelated noise:

$$\mathbf{x} = \mathbf{A}\mathbf{s} + \mathbf{n}$$

    - time lags in the observations x
    - convolved mixtures of x
    - Nonlinear mixtures
  - extensions to address these problems do exist

Basic ICA

- Ambiguities
  - The variances (and hence energies and amplitudes) of s cannot be estimated, because:

$$\mathbf{x} = \sum_i (\frac{1}{\alpha_i}\mathbf{a}_i)(s_i\alpha_i)$$

  - therefore, the variance of s is initially set to 1 by whitening

Basic ICA

- Ambiguities
  - The order of the independent components in s cannot be determined, because a permutation matrix P:

$$\mathbf{x} = \mathbf{A}\mathbf{P}^{-1}\mathbf{P}\mathbf{s}$$

  - …would simply result in an unknown, new mixing matrix:

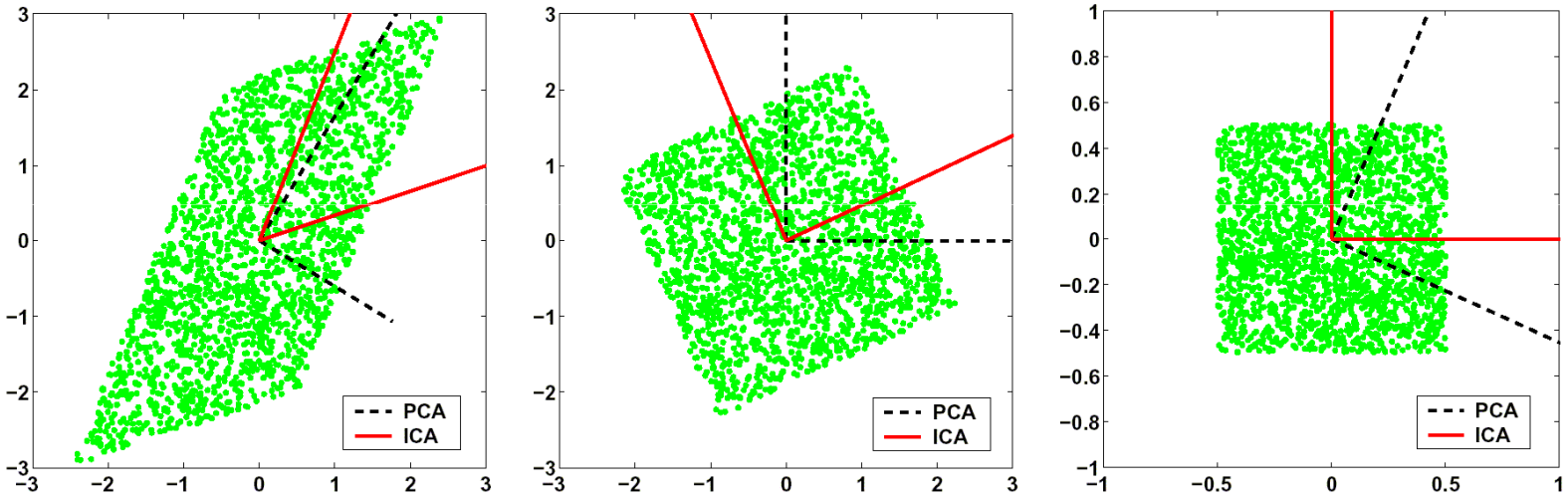
$$\mathbf{A}\mathbf{P}^{-1}$$

Basic ICA

- Example:
  - Generate two uniformly distributed variables and a mixing matrix A
  - Subtract the sample mean from observations x
  - Sphere the data with the whitening matrix V
  - Do a PCA and ICA and compare the results
  - → matlab example pca_vs_ica.m

Basic ICA

- Main difference between PCA and ICA:



(a) Original (b) After PCA pre-whitening (c) After ICA projection

- Taken from: http://ict.ewi.tudelft.nl/~dick/cvonline/ica/

Basic ICA

- uncorrelatedness:

$$\mathrm{cov}(y_1, y_2) = E\{y_1 y_2\} - E\{y_1\}E\{y_2\} = 0$$

$$E\{yy^T\} = \mathrm{cov}(y) \neq I$$

  - is a property of the PCA, but not sufficient for the ICA
  - Independence implies uncorrelatedness, but not vice versa

$$\mathrm{cov}(y) = I$$

Basic ICA

- Sphering
  - Solves the ICA problem up to an orthogonal transformation
  - Can be done by PCA with a whitening matrix:

$$\mathbf{V} = \mathbf{E}\mathbf{D}^{-1/2}\mathbf{E}^{T}$$

Basic ICA

- Problem with gaussian inputs
  - Whitening makes the mixing matrix A orthogonal:

$$z = Vx = VAs = \tilde{A}s$$

$$\tilde{A}\tilde{A}^T = I$$

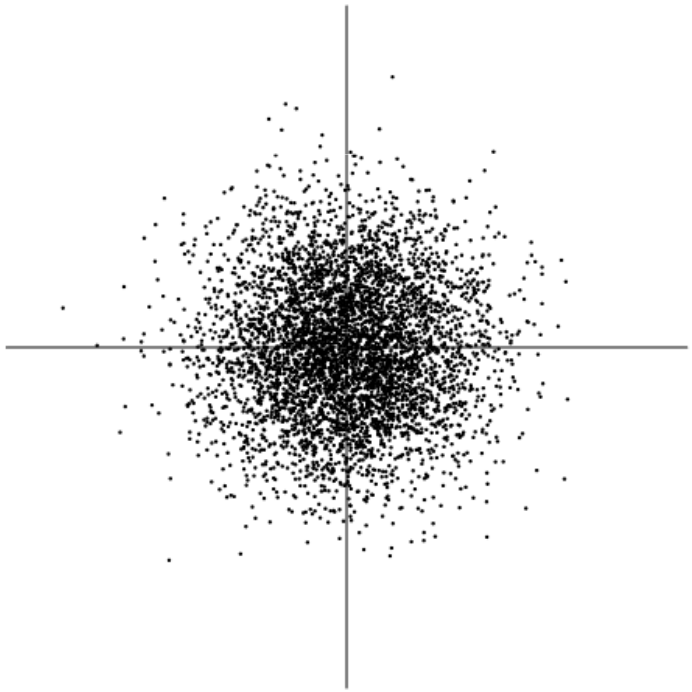
  - Therefore, $\tilde{A}$ does not change the joint pdf of z1 and z2:

$$p(z1, z2) = \frac{1}{2\pi}\exp\left(-\frac{\|s\|^2}{2}\right)$$

  - (blackboard)

Basic ICA

- Problem with gaussian inputs
  - Density is rotationally symmetric:



Basic ICA

- Solving the ICA problem
  - Measures of mutual independence:
  - Maximum likelihood
  - Information maximization
  - Higher-order statistics:
    - Kurtosis
    - Negentropy

Maximum likelihood

- Basic log likelihood function with the parameter $\vec{\Gamma}$:

$$l = \frac{1}{N}\log\{\mathcal{L}\} = \frac{1}{N}\sum_{j=0}^{N-1}\log\left\{p(\vec{x}^{(j)};\vec{\Gamma})\right\}$$

- Maximum likelihood estimate for $\vec{\Gamma}$:

$$\frac{\partial l}{\partial \gamma_i} = \sum_{j=0}^{N-1}\frac{d}{d\gamma_i}\log\left\{p(\vec{x}^{(j)};\vec{\Gamma})\right\} = -\sum_{j=0}^{N-1}\Phi(\vec{x}^{(j)};\vec{\Gamma})$$

- With the score function Φ as an prior assumption of the real pdf of x

Maximum likelihood

- Density of the ICA mixture model:

$$p_x(\mathbf{x}) = |\det \mathbf{B}| \prod_i p_i(\mathbf{b}_i^T \mathbf{x}) \qquad \mathbf{B} = \mathbf{A}^{-1} \qquad \mathbf{x} = \mathbf{A}\mathbf{s}$$

- Log likelihood:

$$\frac{1}{T} \log L(\mathbf{B}) = E\{\sum_{i=1}^{n} \log p_i(\mathbf{b}_i^T \mathbf{x})\} + \log |\det \mathbf{B}|$$

- Gradient of the log-likelihood with the score function g:

$$\frac{1}{T} \frac{\partial \log L}{\partial \mathbf{B}} = [\mathbf{B}^T]^{-1} + E\{\mathbf{g}(\mathbf{B}\mathbf{x})\mathbf{x}^T\} \qquad g_i = (\log p_i)' = \frac{p_i'}{p_i}$$

Maximum likelihood

- Natural gradient algorithm:

$$\Delta \mathbf{B} \propto (\mathbf{I} + E\{\mathbf{g}(\mathbf{y})\mathbf{y}^T\})\mathbf{B}$$

  - The pdf of x is not known exactly, the model distinguishes only the super- and subgaussian case:

$$\log \tilde{p}_i^+(s) = \alpha_1 - 2\log\cosh(s)$$
$$\log \tilde{p}_i^-(s) = \alpha_2 - [s^2/2 - \log\cosh(s)]$$

    - Which results in a score function for supergaussian ICs:

$$g^+(y) = -2\tanh(y)$$

    - And another for the subgaussian case:

$$g^-(y) = \tanh(y) - y$$

Maximum likelihood

- Algorithm:
  - Initialize the separation matrix B with random values
  - Compute y = Bx
  - Choose a learning rate μ and μ_γ
  - Update $\gamma_i = (1 - \mu_\gamma)\gamma_i + \mu_\gamma E\{-\tanh(y_i)y_i + (1 - \tanh(y_i)^2)\}$
  - coose g+ if $\gamma_i > 0$, else g-
  - Update the seperating matrix: $\mathbf{B} \leftarrow \mathbf{B} + \mu[\mathbf{I} + \mathbf{g}(\mathbf{y})\mathbf{y}^T]\mathbf{B}$
  - Repeat until the convergence criterion: $E\{\mathbf{g}(\mathbf{y})\mathbf{y}^T\} = \mathbf{I}$ is met.

- This algorithm is also used by FastICA

Information maximization

- Infomax principle:
  - The entropy from the outputs $y = \tilde{s} = Bx$ is maximized:

$$H(\mathbf{y}) = H(\phi_1(\mathbf{b}_1^T\mathbf{x}), ..., \phi_n(\mathbf{b}_n^T\mathbf{x}))$$

  - The entropy of the outputs can be expressed as the entropy of the inputs x plus the entropy of the transformation $x \rightarrow s$:

$$H(\mathbf{y}) = H(\mathbf{x}) + E\{\log|\det\frac{\partial\mathbf{F}}{\partial\mathbf{B}}(\mathbf{x})|\}$$

  - With a set $\mathbf{F}(\mathbf{x}) = (\phi_1(\mathbf{w}_1^T\mathbf{x}), ..., \phi_n(\mathbf{w}_n^T\mathbf{x}))$ of nonlinear output functions ϕ_i

Information maximization

- Infomax principle:
  - Taking a closer look at the derivative of F(x)

$$E\{\log|\det\frac{\partial\mathbf{F}}{\partial\mathbf{B}}(\mathbf{x})|\} = \sum_i E\{\log\phi_i'(\mathbf{b}_i^T\mathbf{x})\} + \log|\det\mathbf{B}|$$

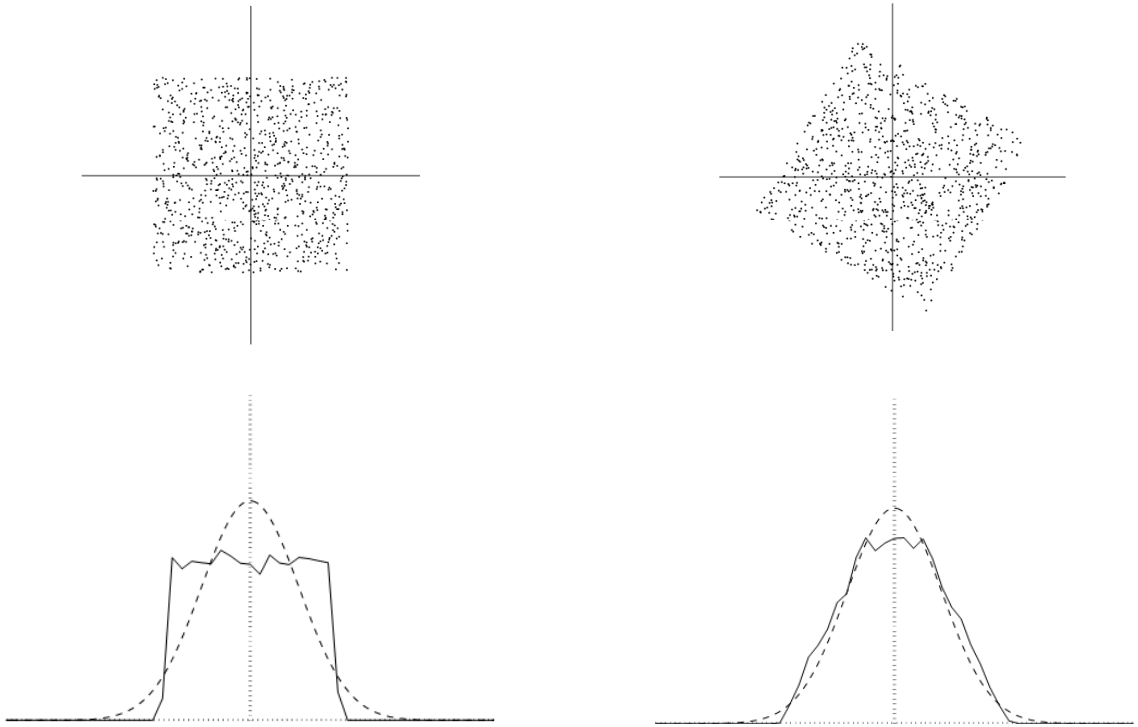
  - The output entropy is of the same form as the log likelihood on slide #27
  - The pdfs of the outputs y are replaced by the nonlinear output functions ϕ_i' , these would be the same as the score functions g
  - → infomax and mle algorithm are identical

Maximization of nongaussianity

- ICA by maximization of nongaussianity
  - Gaussian PDFs cannot be separated
  - According to the CLT, the sum of two independent variables tends towards a gaussian distribution
  - Idea: maximization of non-gaussianity also maximizes independence

Maximization of nongaussianity

- Linear mixtures tend towards a gaussian density:



Kurtosis

- Measure of non-gaussianity:
  - Kurtosis or curvature of the density of y:

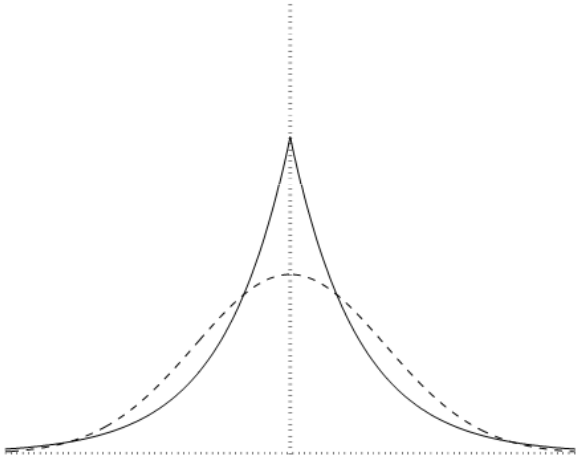
$$\text{kurt}(y) = E\{y^4\} - 3(E\{y^2\})^2$$

  - Assumptions: y has already zero mean is whitened, so that the variance = 1:

$$\text{kurt}(y) = E\{y^4\} - 3$$

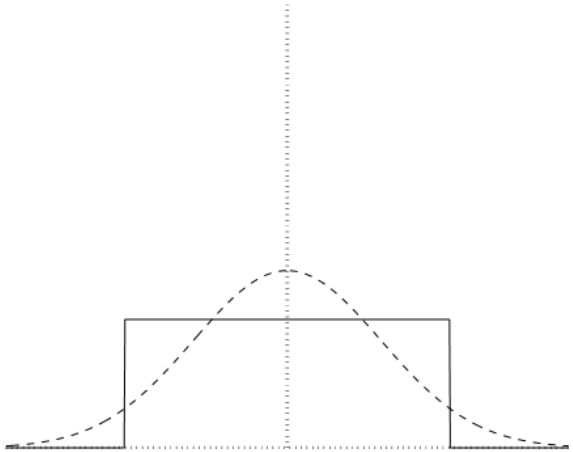
Kurtosis

- Kurtosis for a laplacian and an uniform distribution, both with unit variance:



supergaussian:

kurt(y) > 0



subgaussian:

kurt(y) < 0

Kurtosis

- Kurtosis is zero for a gaussian density
  - Positive for a supergaussian
  - Negative for a subgaussian

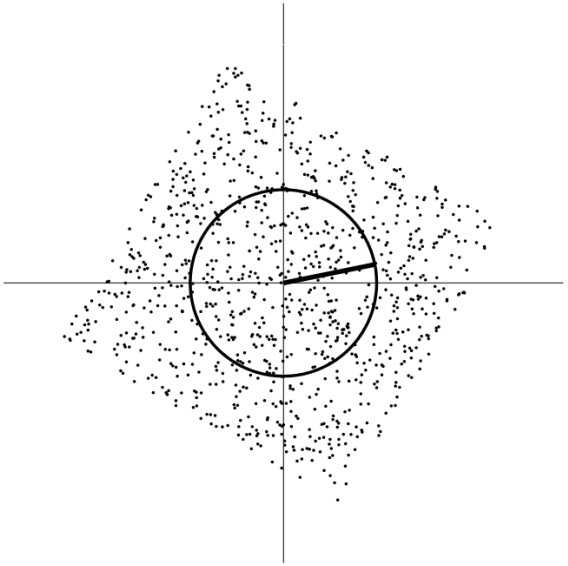
- Maximizing $|kurt(y)|$ equals maximizing the non-gaussianity of y.
- By whitening: $\mathbf{z} = \mathbf{V}\mathbf{x}$
  - z was defined to unit variance
  - and because of the 2nd ambiguity of the ICA, y also has unit variance

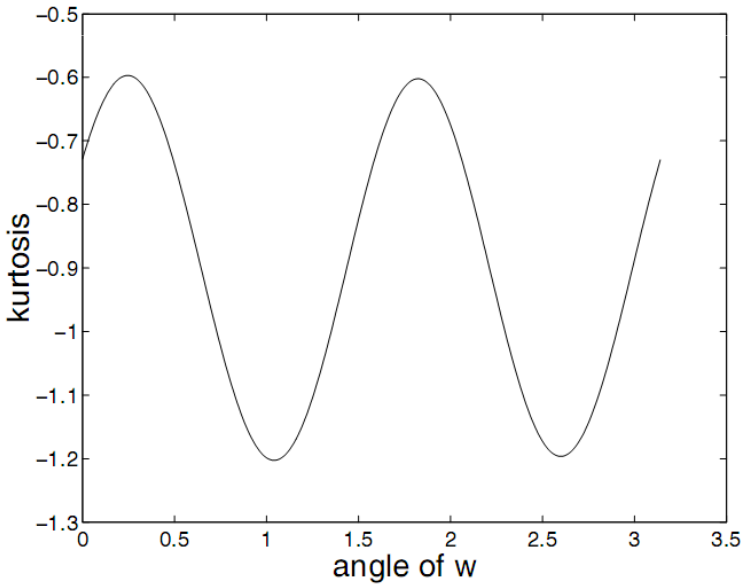
  - therefore: $\|\mathbf{w}\| = 1$ because of: $y = \mathbf{w}^T\mathbf{z}$

- → search for the angle in w that maximizes $|kurt(y)|$

Kurtosis

- The whitening step already made the data uncorrelated and normalized it to unit variance.
- The only step left is to find the angle in the demixing matrix w that makes the output y- independent:





Kurtosis

- The extrema of kurtosis occur at the correct angle of w
  - Calculating the derivative with respect to w

$$\frac{\partial |\mathrm{kurt}(\mathbf{w}^T\mathbf{z})|}{\partial \mathbf{w}} = 4\,\mathrm{sign}(\mathrm{kurt}(\mathbf{w}^T\mathbf{z}))[E\{\mathbf{z}(\mathbf{w}^T\mathbf{z})^3\} - 3\mathbf{w}\|\mathbf{w}\|^2]$$

  - results in a simple gradient algorithm:

$$\Delta\mathbf{w} \propto \mathrm{sign}(\mathrm{kurt}(\mathbf{w}^T\mathbf{z}))E\{\mathbf{z}(\mathbf{w}^T\mathbf{z})^3\}$$

$$\mathbf{w} \leftarrow \mathbf{w}/\|\mathbf{w}\|$$

  - dividing by the norm of w keeps the variance to 1
  - The term $3\mathbf{w}\|\mathbf{w}\|^2$ does only change the norm, and can thus be omitted.

- This algorithm is also used by FastICA

Negentropy

- Problems with kurtosis:
  - Sample outliers get massively amplified by the fourth moment

$$E\{y^4\} - 3(E\{y^2\})^2$$

  - Better: estimate the non-quadratic moment from kurtosis by nonlinear functions G
  - Computationally more complex, but also more stable

Negentropy

- Negentropy
  - A gaussian RV has the largest entropy amongst all RVs with equal variance → The normalized entropy, or negentropy J can be used as a measure of non-gaussianity:

$$H(\mathbf{y}) = -\int p_y(\boldsymbol{\eta}) \log p_y(\boldsymbol{\eta}) \mathrm{d}\boldsymbol{\eta}$$

$$J(\mathbf{y}) = H(\mathbf{y}_{gauss}) - H(\mathbf{y})$$

Negentropy

- Negentropy
  - Approximation:

$$J(y) \approx \frac{1}{12} E\{y^3\}^2 + \frac{1}{48} \mathrm{kurt}(y)^2$$

  - Approximation by a nonlinear function G:

$$J(y) \propto [E\{G(y)\} - E\{G(\nu)\}]^2$$

  - ν is a gaussian with zero mean and unit variance.
  - G is basically a compromise between approximating kurtosis and avoiding the fast-growing fourth power. Usually it is set to:

$$G_1(y) = \frac{1}{a_1} \log \cosh a_1 y \qquad \text{or} \qquad G_2(y) = -\exp(-y^2/2)$$

    based on heuristic search

Negentropy

- Algorithm:
  - Whiten the input data x and remove its mean
  - Choose an initial value for w and γ
  - Update $\Delta\gamma \propto (G(\mathbf{w}^T\mathbf{z}) - E\{G(\nu)\}) - \gamma$
  - Update $\Delta\mathbf{w} \propto \gamma\mathbf{z}g(\mathbf{w}^T\mathbf{z})$
    - g is the derivative of G
  - Normalize $\mathbf{w} \leftarrow \mathbf{w}/||\mathbf{w}||$ to attain unit variance
  - Repeat until the convergence criterion: $E\{\mathbf{g}(\mathbf{y})\mathbf{y}^T\} = \mathbf{I}$ is met.

  - This algorithm is also used by FastICA

Basic ICA

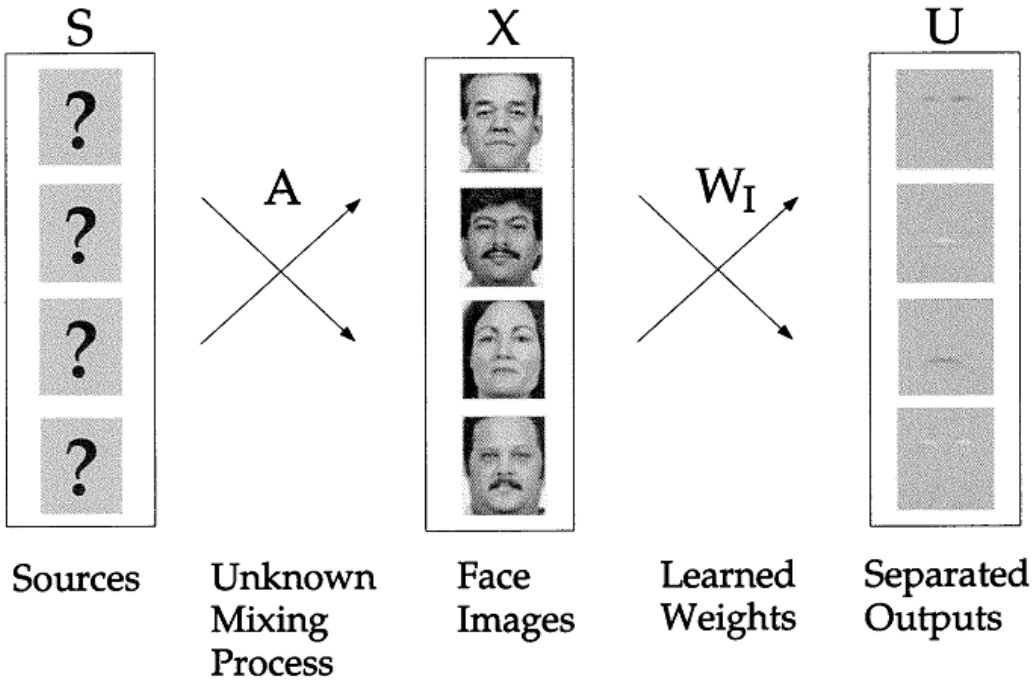
- Efficient fixed-point implementation of maximum likelihood estimation, kurtosis and negentropy:
  - FastICA toolbox
  - Matlab demo using kurtosis: fastica_test.m

ICA Applications

- Popular applications:
  - Feature extraction of natural images
  - Blind source separation (cocktail party phenomenon)
  - Image denoising
  - EEG signal separation

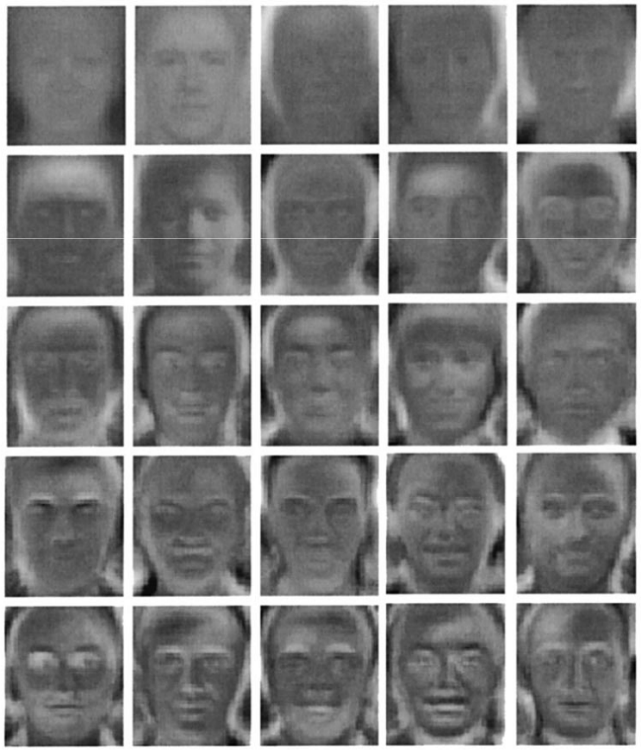
Feature extraction of natural images

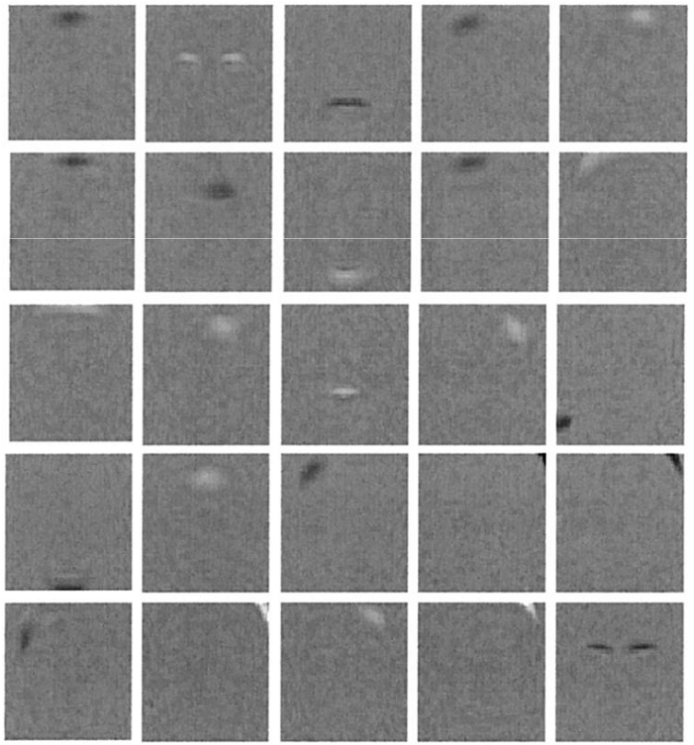
- Feature extraction of natural images done by Marian Stewart Bartlett et. al. on examples of the FERET database:



Feature extraction of natural images

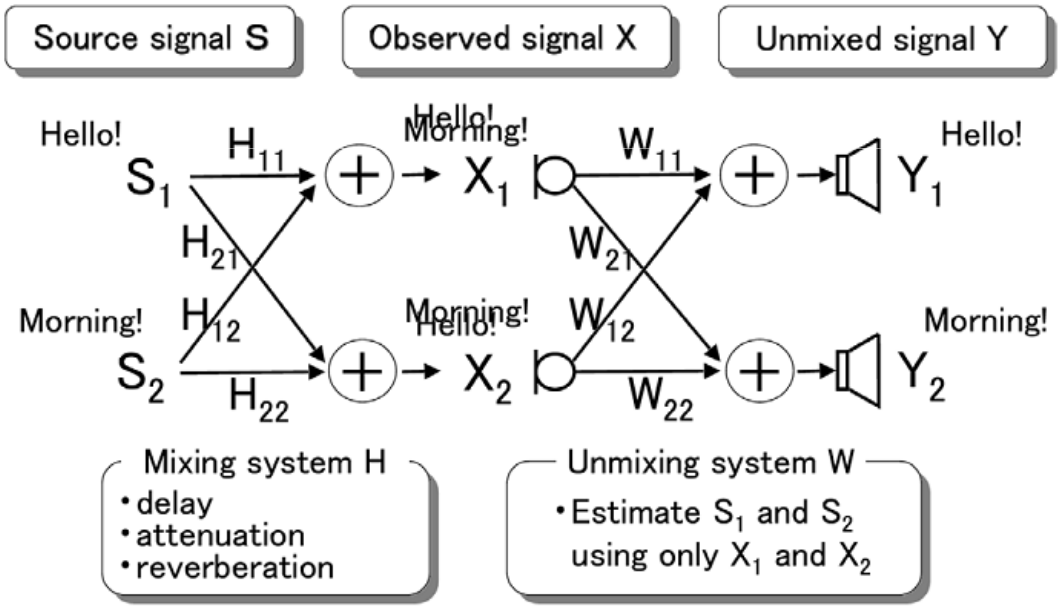
- PCA detects global features. ICA local ones.





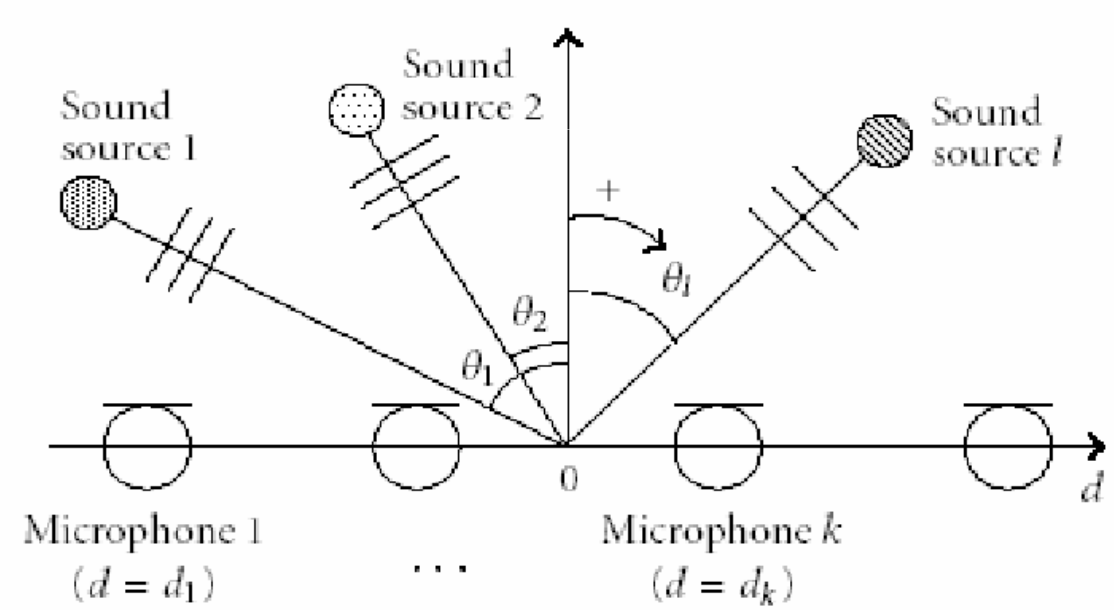
Blind source separation by ICA and ABF

- Quite realistic model done by Hiroshi Saruwatari et. al. for real-world speech signals:



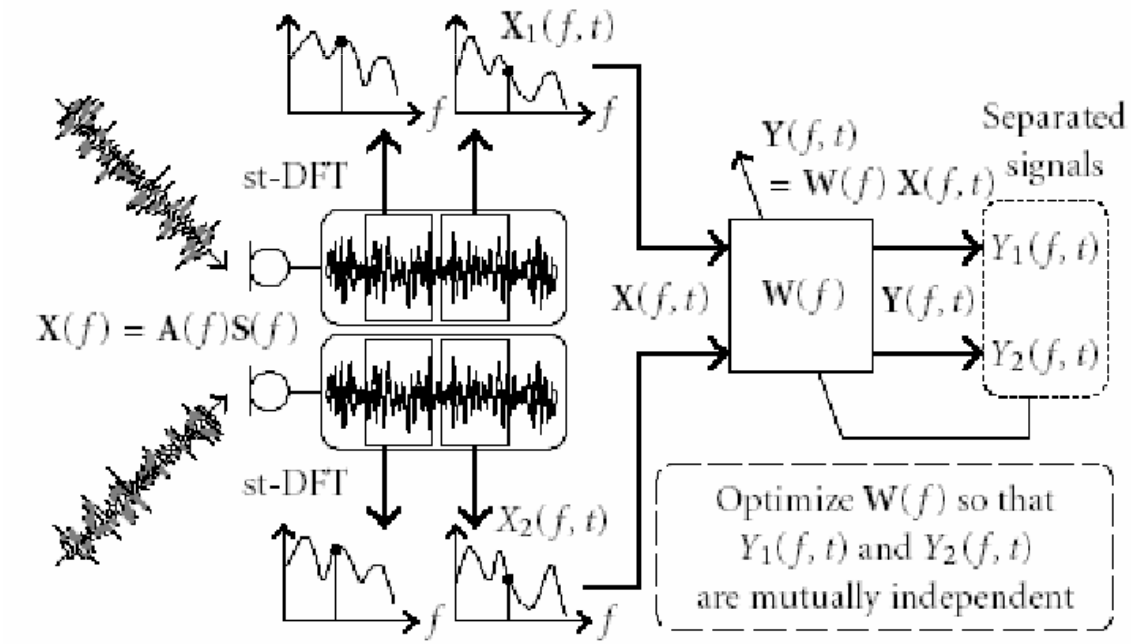
Blind source separation by ICA and ABF

- Their idea: beamforming in the FD with a subband ICA
  - Straight-line ABF:



Blind source separation by ICA and ABF

- The mixing matrix $\mathbf{A}(f) = \begin{bmatrix} A_{11}(f) & \cdots & A_{1L}(f) \\ \vdots & & \vdots \\ A_{K1}(f) & \cdots & A_{KL}(f) \end{bmatrix}$
  - consists of convolution kernels instead of mixing ratios
  - models echo paths and time delays and their change with time



Blind source separation

- The delay-and-sum beamformer is realized in the FD.
- The elements of the mixing matrix turn from convolutive mixtures in TD into a cross-correlation index per subband in FD.
- Hence the ICA can be done on each frequency bin over multiple st-DFTs to obtain an unmixing matrix for each bin.

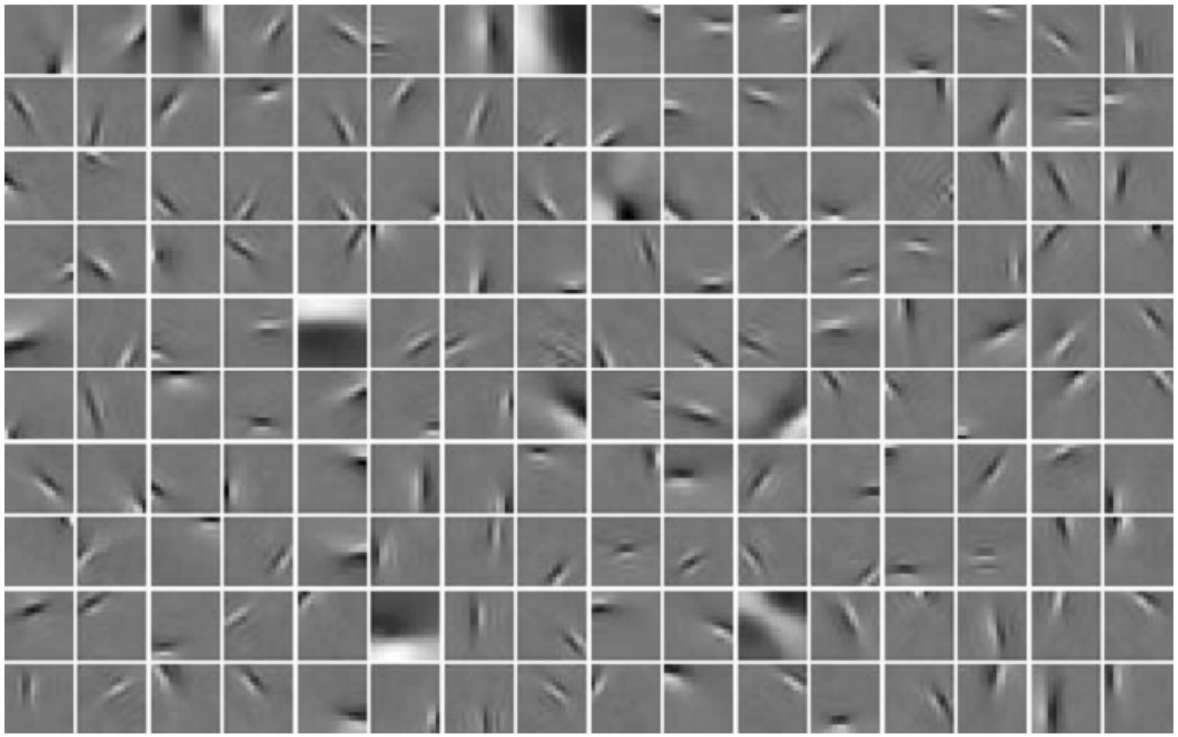
- Problems:
  - Permutation ambiguity must be solved for each frame.
  - Original volume is lost (ICA assumes unit variance)

- matlab demo: bss_test.m

Image denoising with sparse code shrinkage

- An Image is represented by a minimal number of basis vectors
  - Similar to wavelet decomposition or Gabor analysis
- Image x is modeled as clean image s with noise: $x = s + \nu$
  - ν is a gaussian with zero mean and variance σ^2
- A set of basis vectors W is obtained by a noise (and very large) free training set of patches of natural images with the ICA.
  - Typically 50,000 patches with 16x16 pixels [1][12].

Image denoising with sparse code shrinkage



- Advantage of ICA:
  - Gabor functions or the mother wavelet have to be chosen carefully.
  - ICA finds the best basis vectors from small patches of natural images.

Image denoising with sparse code shrinkage

- With the basis vectors W, the noisy image x can now be represented as:

$$y = Wx$$

- For optimal W, the components y_i will have a sparse distribution.
- A shrinkage nonlinearity g_i is now applied to y:

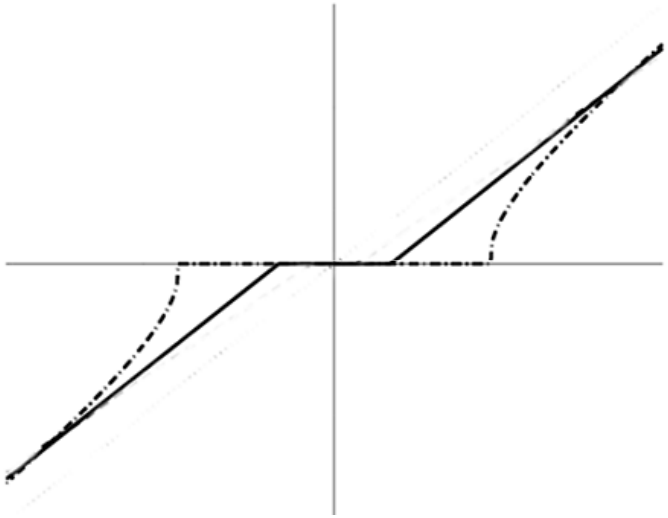
$$u_i = g_i(y_i)$$

  - The shrinkage nonlinearity is defined with the noise variance σ^2 :

$$g(u) = \frac{1}{1+\sigma^2 a}\operatorname{sign}(u)\max(0, |u| - b\sigma^2)$$

Image denoising with sparse code shrinkage

- The shrinkage nonlinearity is defined with the noise variance σ^2 :



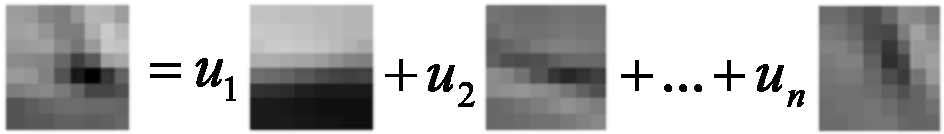
$$g(u) = \frac{1}{1+\sigma^2 a}\text{sign}(u)\max(0, |u| - b\sigma^2)$$

- The components u_i which only represent noise are wiped out from the sparse representation

Image denoising with sparse code shrinkage

- The noise-free image is reconstructed by transforming the sparse, noise-free representation u_i back:

$$\tilde{s}_i = W^T u_i$$

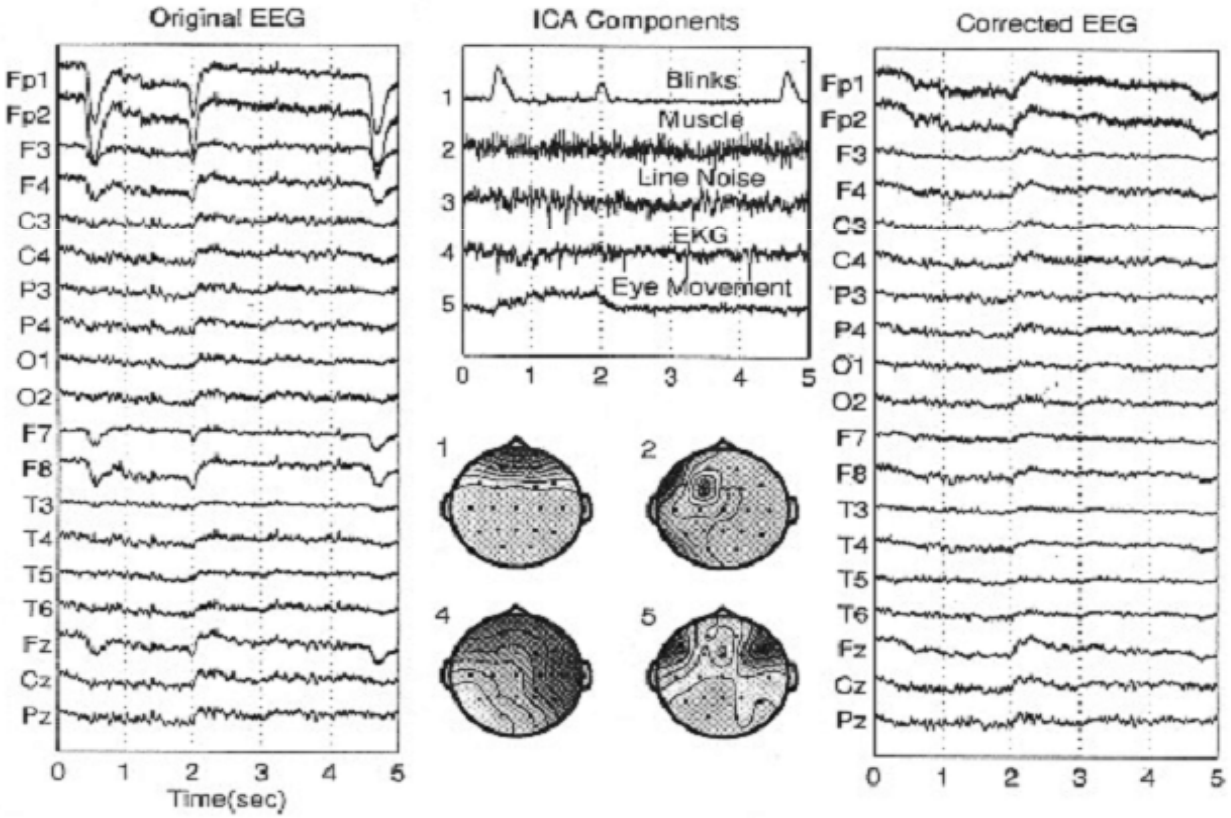


EEG signal separation

- Artifact identification and removal from EEG:
  - EEG data is corrupted by muscle activity, like eye movement.
  - Instantaneous mixing mode of the ICA is valid:
    - Source signals are statistically independent
    - No considerable reflections or time delays since all measured signals lie well below 1kHz
    - But the mixing matrix A is non-stationary → use small time frames and batch algorithms
    - FastICA achieves good results for small data sets

EEG signal separation

- EEG artifacts can be removed from relevant data:



Kernel ICA

- Problems with standard ICA algorithms:
  - Common approaches using Kurtosis or Negentropy have difficulties for near-gaussian distributions.
  - The pdf of the inputs and therefore the global optimum for independence is unknown, only local convergence is ensured.
  - Higher-order moments are approximated by nonlinearities.
  - Only for 2 cases are differentiated:
    - Subgaussian density
    - Supergaussian density
  - Idea: use Kernels to search over a richer set of nonlinearities and to define new independence measures

Kernel ICA

- Properties:
  - Kernel ICA does not project the input data to a nonlinear feature space for linear separation like kernel PCA
  - We are still interested in a linear separation in input space by the basic ICA model:

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

  - Kernels are used for finding nonlinear independence measures of x

Kernel ICA

- Properties:
  - Kernel methods measure independence in the spectrum of the covariance for functions defined in reproducing kernel hilbert spaces. Chen and Bickel (2005), (2002); Gretton et al. (2005)
  - Independence measures in Hilbert space:
    - CCA in feature space [2]
      - Maximizes the correlation between projections of the data in feature space
    - Constrained Covariance (COCO) [10]
      - Spectral norm of the covariance operator.
    - Hilbert-Schmidt Independence Criterion (HSIC) [10]
      - Hilbert-Schmidt norm of the covariance operator

Kernel ICA

- ICA model used in [2]: $y = Ax$
- Correlation in RKHS:
  - $\Phi(x)$ maps x into the feature space $\mathcal{F}$
  - The reproducing property for the RKHS $\mathcal{F}$ is defined as:

    $f(x) = \langle \Phi(x), f \rangle, \quad \forall f \in \mathcal{F}, \ \forall x \in \mathbb{R}$ (Saitoh, 1988):
  - [2] defines the correlation:

    $\mathrm{corr}(f_1(x_1), f_2(x_2)) = \mathrm{corr}\left(\langle \Phi(x_1), f_1 \rangle, \langle \Phi(x_2), f_2 \rangle\right)$

Kernel ICA

- Kernel
  - The Gram matrix $K_{ij} = K(x_i, x_j)$ contains every possible inner product of $\Phi(x)$ in feature space, and is postive semi-definite for any x.
  - The kernel can be used to evaluate an inner product in feature space $\mathcal{F}$:

    $\langle \Phi(x), \Phi(y) \rangle = \langle K(\cdot, x), K(\cdot, y) \rangle = K(x, y)$ "kernel trick"
  - Example of an isotropic gaussian kernel:

$$K(x, y) = G_\sigma(x - y) = \exp\left(-\frac{1}{2\sigma^2}||x - y||^2\right)$$

Kernel ICA

- Independence measures:
  - The independence measure presented in [2] is the maximization of the correlation between projections of the output data x projected to the feature space $\mathcal{F}$:

$$\rho_{\mathcal{F}} = \max_{f_1, f_2 \in \mathcal{F}} \operatorname{corr}(f_1(x_1), f_2(x_2)) = \max_{f_1, f_2 \in \mathcal{F}} \frac{\operatorname{cov}(f_1(x_1), f_2(x_2))}{(\operatorname{var} f_1(x_1))^{1/2} (\operatorname{var} f_2(x_2))^{1/2}}$$

$$\hat{\rho}_{\mathcal{F}}(K_1, K_2) = \max_{\alpha_1,\ \alpha_2 \in \mathbb{R}^N} \frac{\alpha_1^\top K_1 K_2 \alpha_2}{(\alpha_1^\top K_1^2 \alpha_1)^{1/2} (\alpha_2^\top K_2^2 \alpha_2)^{1/2}}$$

(blackboard)

  - Basically a CCA eigenvalue problem in feature space:

$$\begin{pmatrix} 0 & K_1 K_2 \\ K_2 K_1 & 0 \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix} = \rho \begin{pmatrix} K_1^2 & 0 \\ 0 & K_2^2 \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix}$$

Kernel ICA

- Algorithm:
  - Whiten the input data y
  - Define a Kernel K(x,y)
  - Compute the Gram Matrices for the kernels $K_1..K_m$ for the estimated outputs $x = Wy$
  - Minimze the contrast function with respect to W

$$\hat{I}_{\delta_{\mathcal{F}}}(K_1, \ldots, K_m) = -\frac{1}{2} \log \hat{\delta}_{\mathcal{F}}^{\kappa}(K_1, \ldots, K_m)$$

  - This algorithm is implemented in the KernelICA toolbox
    → matlab: kernel_ica_test.m

Kernel ICA

- Problems:
  - Difficulties with large data sets, the Gram matrices of the Kernels have to be computed and stored for every optimization step of the unmixing matrix W.
  - Gradient calculation on the independence measures with respect to W is of requires $O(m^4)$ operations for m independent sources [10].

Kernel ICA

- Conclusion:
  - ICA addresses a vast field of application
    - BSS
    - Deconvolution
    - Denoising
  - Though its model has some tough constraints
    - Additive noise must be gaussian, etc.
  - No algorithm that guarantees a global optimum
  - Kernel extension is computationally expensive

Literature

[1] Sparse Code Shrinkage: Denoising by Nonlinear Maximum Likelihood Estimation - Aapo Hyvärinen, Patrik Hoyer and Erkki Oja Helsinki University of Technology

[2] Kernel Independent Component Analysis - Francis R. Bach and Michael I. Jordan, University of California

[3] ICA-BASED BLIND SOURCE SEPARATION OF SOUNDS - Shoji Makino et. al. NTT Communication Science Laboratories, Kyoto Japan

[4] http://www.tsi.enst.fr/icacentral/ - general information on ICA

[5] http://www.jim-stone.staff.shef.ac.uk/ - BSS examples

[6] Feature Extraction by Independent Component Analysis - Yen-Wei CHEN, Ritsumeikan Univ., Japan

[7] Face Recognition by Independent Component Analysis - Marian Stewart Bartlett, Institute for Neural Computation, University of California-SanDiego

[8] Canonical Correlation Analysis - Pankratov Pawlo, TU Munich

Literature

[9] Blind Source Separation Combining Independent Component Analysis and Beamforming, Hiroshi Saruwatari et. al. EURASIP Journal on Applied Signal Processing 2003

[10] Brisk Kernel ICA - Stefanie Jegelka, Arthur Gretton

[11] Blind Speech Separation - Shoji Makino et. al., NTT Communication Science Labs. Japan

[12] Blind Source Separation: the Sparsity Revolution - Bobin J. et. al., Laboratoire AIM, CEA/DSM-CNRS-Universite Paris Diderot

[13] A Regularized Kernel CCA Contrast Function for ICA - Carlos Alzate, Johan A. K. Suykens, Department of Electrical Engineering, Katholieke Universiteit Leuven, Belgium

[14] http://www.cis.hut.fi/projects/ica/fastica/ - FastICA matlab demo

[15] http://www.di.ens.fr/~fbach/kernel-ica/index.htm - KernelICA matlab demo