Lecture IV

ICA for linear static problems: algebraic approaches

Ivica Kopriva

Ruđer Bošković Institute

e-mail: ikopriva@irb.hr ikopriva@gmail.com
Web: http://www.lair.irb.hr/ikopriva/

Course outline

- Motivation with illustration of applications (lecture I)
- Mathematical preliminaries with principal component analysis (PCA)? (lecture II)
- Independent component analysis (ICA) for linear static problems: information-theoretic approaches (lecture III)
- ICA for linear static problems: algebraic approaches (lecture IV)
- ICA for linear static problems with noise (lecture V)
- Dependent component analysis (DCA) (lecture VI)

Course outline

- Underdetermined blind source separation (BSS) and sparse component analysis (SCA) (lecture VII/VIII)
- Nonnegative matrix factorization (NMF) for determined and underdetermined BSS problems (lecture VIII/IX)
- BSS from linear convolutive (dynamic) mixtures (lecture X/XI)
- Nonlinear BSS (lecture XI/XII)
- Tensor factorization (TF): BSS of multidimensional sources and feature extraction (lecture XIII/XIV)

Homework problems

1. Understanding natural (relative) gradient. Convergence analysis (comparison) of algorithms in adaptive minimum mean square error problem with matrix argument using Riemanian and Euclidean gradient. **Theory, MATLAB demonstration and comments of the results.**

2. Principal component analysis (PCA) based separation of two Gaussian signals.PCA based separation of two uniformly distributed signals. Scatter plots of true sources, mixtures and estimated sources. PCA based separation of two images (histograms). **Theory, MATLAB demonstration and comments of the results.**

3. Independent component analysis (ICA) based separation of two uniformly distributed signals (scatter plots of true sources, mixtures and estimated sources).ICA based separation of two images (histograms). **Theory, MATLAB demonstration and comments of the results.**

Seminar problems

1. Blind separation of two uniformly distributed signals with maximum likelihood (ML) and AMUSE/SOBI independent component analysis (ICA) algorithm.
   Blind separation of two speech signals with ML and AMUSE/SOBI ICA algorithm. **Theory, MATLAB demonstration and comments of the results.**

2. Blind decomposition/segmentation of multispectral (RGB) image using ICA, dependent component analysis (DCA) and nonnegative matrix factorization (NMF) algorithms. **Theory, MATLAB demonstration and comments of the results.**

3. Blind separation of acoustic (speech) signals from convolutive dynamic mixture. **Theory, MATLAB demonstration and comments of the results.**

Seminar problems

4. Blind separation of images of human faces using ICA and DCA algorithms (innovation transform and ICA, wavelet packets and ICA) **Theory, MATLAB demonstration and comments of the results.**

5. Blind decomposition of multispectral (RGB) image using sparse component analysis (SCA): clustering + L_p norm ($0<p\leq 1$) minimization . **Theory, MATLAB demonstration and comments of the results.**

6. Blind separation of four sinusoidal signals from two static mixtures (a computer generated example) using sparse component analysis (SCA): clustering + L_p norm ($0<p\leq 1$) minimization in frequency (Fourier) domain. **Theory, MATLAB demonstration and comments of the results.**

Seminar problems

7. Blind separation of three acoustic signals from two static mixtures (a computer generated example) using sparse component analysis (SCA): clustering + L_p norm ($0<p\leq 1$) minimization in time-frequency (short-time Fourier) domain. **Theory, MATLAB demonstration and comments of the results.**

8. Blind extraction of five pure components from mass spectra of two static mixtures of chemical compounds using sparse component analysis (SCA): clustering a set of single component points + L_p norm ($0<p\leq 1$) minimization in m/z domain. **Theory, MATLAB demonstration and comments of the results.**

9. Feature extraction from protein (mass) spectra by tensor factorization of disease and control samples in joint bases. Prediction of prostate/ovarian cancer. **Theory, MATLAB demonstration and comments of the results.**

ICA for linear instantaneous models

- Second order ICA methods (time-delayed correlations)
- Tensorial ICA methods (Fourth order cumulants)
- Kernel ICA algorithm

Second order statistics (SOS) based methods[1]

Problem. $\mathbf{x}=\mathbf{A}\mathbf{x}$, $\mathbf{x}\in R^n$, $\mathbf{s}\in R^m$, $\mathbf{A}\in R^{n\times m}$ $n>m$.

Assumptions:

- **A** is a full column rank.

- Sources are spatially uncorrelated with different autocorrelation functions but temporally correlated (colored) zero mean signals:

$$E\left[\mathbf{s}(t)\mathbf{s}^T(t-\tau)\right]=\mathbf{R}_{\mathbf{s}}(\tau)=diag\left\{E\left[s_i(t)s_i(t-\tau)\right]\right\}_{i=1}^{m}\ \forall\ \ \tau\in\{0,...,T\}$$

$$E\left[s_i(t)s_i(t-\tau)\right]\neq E\left[s_j(t)s_j(t-\tau)\right]\ \forall i\neq j$$

- Additive noise **V** if present is independent of source signals, it can be spatially correlated but it is assumed to be temporally uncorrelated (white):

$$E\left[\mathbf{v}(t)\mathbf{v}^T(t-\tau)\right]=\delta_{\tau 0}\mathbf{R}_{\mathbf{v}}(\tau)$$

[1]Blind Decorrelation and SOS for Robust Blind Identification," *Chapter 4* in: *Adaptive Blind Signal and Image Processing* by A. Chichocki and S.I. Amari, John Wiley, 2002.

Eigenvalue decomposition (EVD) approach

When source signals have time structure i.e. their correlations and cross-correlations are nonzero for different time lags:

$$E\left[\mathbf{s}(t)\mathbf{s}^T(t)\right]=\mathbf{R}_\mathbf{s}(0),\;\; E\left[\mathbf{s}(t)\mathbf{s}^T(t-\tau)\right]=\mathbf{R}_\mathbf{s}(\tau)\;\;\forall\tau\in\{0,\ldots T\}$$

$$E\left[\mathbf{x}(t)\mathbf{x}^T(t)\right]=\mathbf{R}_\mathbf{x}(0)=\mathbf{A}\mathbf{R}_\mathbf{s}(0)\mathbf{A}^T$$

$$E\left[\mathbf{x}(t)\mathbf{x}^T(t-\tau)\right]=\mathbf{R}_\mathbf{x}(\tau)=\mathbf{A}\mathbf{R}_\mathbf{s}(\tau)\mathbf{A}^T$$

it is possible to generate enough equations in order to solve the BSS problem without usage of the higher order statistics. If source signals have time structure (colored statistics) they are even allowed to be Gaussian. First, we want to whiten data with **z=Qx**:

$$\mathbf{z}(t)=\underbrace{\Lambda^{-1/2}\mathbf{V}_\mathbf{x}^T}_{\mathbf{Q}}\mathbf{x}(t):\;\;\mathbf{R}_\mathbf{x}(0)=\mathbf{V}_\mathbf{x}\Lambda_\mathbf{x}\mathbf{V}_\mathbf{x}^T$$

$$\mathbf{R}_\mathbf{z}(0)=E\left[\mathbf{z}(t)\mathbf{z}^T(t)\right]=\mathbf{Q}\mathbf{R}_\mathbf{x}(0)\mathbf{Q}^T=\mathbf{I}_n$$

$$\mathbf{R}_\mathbf{z}(\tau)=E\left[\mathbf{z}(t)\mathbf{z}^T(t-\tau)\right]=\mathbf{Q}\mathbf{R}_\mathbf{x}(\tau)\mathbf{Q}^T$$

EVD approach

We want to diagonalize $\mathbf{R}_z(\tau)$

$$\mathbf{R}_z(\tau) = \mathbf{V}_z \Lambda_z \mathbf{V}_z^T$$

$$\mathbf{R}_z(\tau) = \mathbf{Q}\mathbf{R}_x(\tau)\mathbf{Q}^T = \left| \mathbf{R}_x(\tau) = \mathbf{A}\mathbf{R}_s(\tau)\mathbf{A}^T \right|$$
$$= \mathbf{Q}\mathbf{A}\mathbf{R}_s(\tau)\mathbf{A}^T\mathbf{Q}^T \triangleq \mathbf{V}_z \Lambda_z \mathbf{V}_z^T$$

If Λ_z has distinctive eigenvalues then:

$$\mathbf{Q}\mathbf{A} = \mathbf{V}_z \Rightarrow \hat{\mathbf{A}} = \mathbf{Q}^{\dagger}\mathbf{V}_z$$

Two-steps procedure:

- Diagonalize $\mathbf{R}_x(0)$ to get $\mathbf{Q}$
- Diagonalize $\mathbf{R}_z(\tau)$ to get $\mathbf{V}_z$

Generalized EVD approach

$$\mathbf{R_x}(0) = \mathbf{A}\mathbf{R_s}(0)\mathbf{A}^T$$

$$\mathbf{R_x}(\tau) = \mathbf{A}\mathbf{R_s}(\tau)\mathbf{A}^T$$

$$\mathbf{R_x}^{-1}(0)\mathbf{R_x}(\tau) = \mathbf{A}^{-T}\mathbf{R_s}^{-1}(0)\mathbf{A}^{-1}\mathbf{A}\mathbf{R_s}(\tau)\mathbf{A}^T$$

$$= \mathbf{A}^{-T}\mathbf{R_s}^{-1}(0)\mathbf{R_s}(\tau)\mathbf{A}^T = \mathbf{V}\Lambda\mathbf{V}^{-1}$$

$$\mathbf{R_x}(\tau)\mathbf{V} = \mathbf{R_s}(0)\mathbf{V} \rightarrow GEVD : \mathbf{V}$$

If Λ has distinctive eigenvalues then:

$$\mathbf{A}^{-1} = \mathbf{V} \Rightarrow \hat{\mathbf{A}} = \mathbf{V}^{-T}$$

Robust second order blind identification (SOBI)[2]

To obtained robust estimate of the mixing matrix using SOS it is wise to make sources uncorrelated for several (many) time lags. This leads to joint approximate diagonalization (JAD) problem. It is again assumed that data are whitened: **z=Qx**.

We want to diagonalize the set of matrices

$$\mathbf{M}_i = \mathbf{U}\mathbf{D}_i\mathbf{U}^T \qquad (i = 1,\ldots,L)$$

$\mathbf{M}_i$ are data matrices (time delayed covariance matrices). Since data are prewhitened **U** is unitary matrix. Hence, perform JAD on a set of matrices: $\left\{\mathbf{R}_\mathbf{z}\left(\tau_i\right)\right\}_{i=1}^{L}$ to obtain **U**. Then from

[2]A. Belouchrami, K.A. Meraim, J.F. Cardoso, and E. Moulines, "A blind source separation technique based on second order statistics," *IEEE Trans. on Signal Processing,* 45(2), pp. 434-444, 1997.

Robust SOBI

From:

$$\mathbf{R_z}(\tau_i) = \mathbf{Q}\mathbf{R_x}(\tau_i)\mathbf{Q}^T = \mathbf{Q}\mathbf{A}\mathbf{R_s}(\tau_i)\mathbf{A}^T\mathbf{Q}^T = \mathbf{U}\mathbf{D}_i\mathbf{U}^T$$

it follows:

$$\mathbf{QA} = \mathbf{U} \Rightarrow \hat{\mathbf{A}} = \mathbf{Q}^{\dagger}\mathbf{U}$$

$$\hat{\mathbf{s}}(t) = \hat{\mathbf{A}}^{-1}\mathbf{x}(t) = \mathbf{U}^T\mathbf{Q}\mathbf{x}(t)$$

ICA by time-delayed correlations[3,4]

For whitened with **z=Qx**, it is possible to formulate symmetric one-lag covariance matrix as:

$$\bar{\mathbf{R}}_{\mathbf{z}}(\tau)=\frac{1}{2}\left[\mathbf{R}_{\mathbf{z}}(\tau)+\left(\mathbf{R}_{\mathbf{z}}(\tau)\right)^{\mathrm{T}}\right]$$

$$=\frac{1}{2}\mathbf{QA}\left[E\left\{\mathbf{s}(t)\mathbf{s}(t-\tau)^{\mathrm{T}}\right\}+E\left\{\mathbf{s}(t-\tau)\mathbf{s}(t)^{\mathrm{T}}\right\}\right]\mathbf{A}^{T}\mathbf{Q}^{T}$$

$$=\mathbf{QA}\bar{\mathbf{R}}_{\mathbf{s}}(\tau)\mathbf{A}^{T}\mathbf{Q}^{T}=\mathbf{V}_{\mathbf{z}}\Lambda_{\mathbf{z}}\mathbf{V}_{\mathbf{z}}^{T}$$

Because source signals are uncorrelated by assumption one-time lag covariance matrix $\bar{\mathbf{R}}_{\mathbf{s}}(\tau)$ is diagonal. Hence, $\mathbf{V}_{\mathbf{z}}$ is obtained by EVD of $\bar{\mathbf{R}}_{\mathbf{z}}(\tau)$ and it follows:

$$\mathbf{QA}=\mathbf{V}_{\mathbf{z}}\ \Rightarrow\ \hat{\mathbf{A}}=\mathbf{Q}^{\dagger}\mathbf{V}_{\mathbf{z}}$$

That is how SOS-based BSS problem is solved by ***AMUSE*** algorithm.

[3] L. Molgedey and H. G. Schuster, "Separation of mixture of independent signals using time delayed correlations," *Physical Review Letters,* vol. 72, pp. 3634-3636, 1994.

[4] L. Tong, R.W. Liu, V.C. Soon, and Y. F. Huang, "Indeterminacy and identifiability of blind identification," *IEEE Trans. on Circuits and Systems,* 38:499-509, 1991.

ICA by time-delayed correlations

Previous approach with symmetric one-lag covariance matrix can be extended by using multiple time lags. The ICA algorithm is formulated as joint diagonalization problem:

$$J(\mathbf{U}) = \sum_{\tau \in S} \text{off}\left(\mathbf{U}\bar{\mathbf{R}}_{\mathbf{z}}(\tau)\mathbf{U}^T\right)$$

Representative algorithms are SOBI (second order blind identification)[2] and TDSEP[5].

[2]A. Belouchrami, K.A. Meraim, J.F. Cardoso, and E. Moulines, “A blind source separation technique based on second order statistics,” *IEEE Trans. on Signal Processing,* 45(2), pp. 434-444, 1997.

[5] A. Ziehe, K.R. Muller, G. Nolte, B. M. Mackert, and G. Curio, “TDSEP-an efficient algorithm for blind separation using time structure,” *Proc. ICANN’98,* pp. 675-680, Skovde, Sweden, 1998.

Tensorial methods based ICA

- For i.i.d. random variables autocorrelation function is delta function. Thus, only one SOS (a zero lag covariance matrix) exists.

- If random variables are non-Gaussian higher-order-statistics (HOS) exist. It is possible to define higher-order generalizations of the covariance matrix. The fourth-order (FO) generalization is known as quadricovariance. If **x** is n-dimensional random vector than its quadricovariance is fourth-dimensional array i.e. FO-tensor: $\underline{\mathbf{R}}_{\mathbf{x}} \in \mathbb{R}^{n\times n\times n\times n}$. Its elements are FO-crosscumulants defined for zero mean variables as:

$$\left[\underline{\mathbf{R}}\right]_{ijkl} = \hat{C}_4(x_i, x_j, x_k, x_l)$$
$$= E\left[x_i x_j x_k x_l\right] - E\left[x_i x_j\right]E\left[x_k x_l\right] - E\left[x_i x_k\right]E\left[x_j x_l\right] - E\left[x_i x_l\right]E\left[x_j x_k\right]$$

- Assuming that random variables are distributed symmetrically and that random vector is prewhitened **z**=**Qx**, diagonalization of $\underline{\mathbf{R}}_{\mathbf{x}}$ ensures statistical independence between $\{x_i\}_{i=1}^{n}$ up to the FO.

Joint Approximate Diagonalization of Eigen-matrices (JADE)[6]

- Instead of diagonalizing FO-quadriconaviance tensor that demands estimation of the n^4 FO cross-cumulants a method proposed in ref.6 solves the problem by JAD of n FO-cross-cumulant matrices. Algorithm is known as JADE (Joint Approximate Diagonalization of Eigen-matrices) and can be downloaded from: http://www.tsi.enst.fr/~cardoso/Algo/Jade/jade.m

- Let us start with the linear memoryless model $\mathbf{x}=\mathbf{A}\mathbf{s}$, $\mathbf{x}\in R^n$, $\mathbf{s}\in R^m$, $\mathbf{A}\in R^{n\times m}$ $n>m$. Let us also assume that $\mathbf{x}$ is prewhitened: $\mathbf{z}(t)=\underbrace{\mathbf{\Lambda}_{\mathbf{x}}^{-1/2}\mathbf{V}_{\mathbf{x}}^{T}}_{\mathbf{Q}}\mathbf{x}(t)$. Then:

$$\mathbf{R}_{\mathbf{z}}(0)=E\left[\mathbf{z}(t)\mathbf{z}^{T}(t)\right]=\mathbf{Q}\mathbf{R}_{\mathbf{x}}(0)\mathbf{Q}^{T}=$$
$$=\mathbf{Q}\mathbf{A}\mathbf{R}_{\mathbf{s}}(0)\mathbf{A}^{T}\mathbf{Q}^{T}=\left|\mathbf{R}_{\mathbf{s}}(0)=\mathbf{I}_{m}\right|=$$
$$=\mathbf{Q}\mathbf{A}\mathbf{A}^{T}\mathbf{Q}^{T}=\mathbf{I}_{n}\Rightarrow\mathbf{Q}\mathbf{A}=\mathbf{U}\ (\text{since}\ \mathbf{U}\mathbf{U}^{T}=\mathbf{I}_{n}).$$

[6]J. F. Cardoso and A. Souloumiac, "Blind beamforming for non-Gaussian signals," *IEE-Proc. – F,* Vol. 140, pp. 1362-1370, 1993.

JADE

- **Q** follows from eigendecompositon of $\mathbf{R}_x(0)$, but **U** has to be estimated from FO statistics. Then, $\hat{\mathbf{A}} = \mathbf{Q}^{\dagger}\mathbf{U}$ i.e. $\hat{\mathbf{s}}(t) = \mathbf{U}\mathbf{Q}\mathbf{x}(t)$.

- Let us define $n \times n$ "cumulant matrix" $\mathbf{C}_z(\mathbf{M})$:

$$\mathbf{C}_z(\mathbf{M}) \triangleq \left[\mathbf{C}_z(\mathbf{M})\right]_{ij} = \sum_{k,l=1}^{n} cum\left(z_i, z_j, z_k, z_l\right) m_{kl} \quad 1 \le i, j \le n$$

- **Goal:** **M** should be eigen-matrix of $\mathbf{C}_z(\mathbf{M})$: $\mathbf{C}_z(\mathbf{M}) = \lambda\mathbf{M}$. Using multilinearity property of cumulants as operators as well as the assumption that sources $\{s_i\}_{i=1}^{n}$ are statistically independent we obtain:

$$\mathbf{C}_z(\mathbf{M}) = \left|\mathbf{z}(t) = \mathbf{Q}\mathbf{x}(t) = \mathbf{Q}\mathbf{A}\mathbf{s}(t) = \mathbf{U}\mathbf{s}(t)\right|$$

$$= \sum_{p=1}^{n} c_p \mathbf{u}_p^T \mathbf{M} \mathbf{u}_p \mathbf{u}_p \mathbf{u}_p^T \quad \forall \mathbf{M} \quad c_p \triangleq cum\left(s_p, s_p, s_p, s_p\right)$$

$$= \mathbf{U}\boldsymbol{\Lambda}_{\mathbf{M}}\mathbf{U}^T \quad \boldsymbol{\Lambda}_{\mathbf{M}} \triangleq diag\left\{c_1 \mathbf{u}_1^T \mathbf{M} \mathbf{u}_1, \ldots, c_n \mathbf{u}_n^T \mathbf{M} \mathbf{u}_n\right\}$$

JADE

- It follows that any cumulant matrix $\mathbf{C_z}(\mathbf{M})$ is diagonalized by $\mathbf{U}$, provided that sources have distinct kurtoses c_p i.e. $\mathbf{U}$ contains eigenvectors of $\mathbf{C_z}(\mathbf{M})$.

- How to choose $\mathbf{M}$? How many cumulant maticies to use?

- If we choose $\mathbf{M} = \mathbf{u}_p \mathbf{u}_q^T$ then:

$$\mathbf{C_z}(\mathbf{M} = \mathbf{u}_p \mathbf{u}_q^T) = \sum_{p=1}^{n} c_p \mathbf{u}_p^T \mathbf{u}_p \mathbf{u}_q^T \mathbf{u}_p \mathbf{u}_p \mathbf{u}_p^T =$$

$$= \begin{cases} c_p \mathbf{u}_p \mathbf{u}_p^T & p = q \\ 0 & p \neq q \end{cases}$$

- Hence, we choose: $\mathbf{M} = \mathbf{u}_p \mathbf{u}_p^T$.

JADE

Step 1. Estimate $\mathbf{R}_{\mathbf{x}}(0) = E\left[\mathbf{x}(t)\mathbf{x}(t)^T\right] \simeq \frac{1}{T}\sum_{t=1}^{T}\mathbf{x}(t)\mathbf{x}(t)^T$.

Compute whitening matrix $\mathbf{Q}$ from EVD of $\mathbf{R}_{\mathrm{x}}(0)$: $\mathbf{Q} = \boldsymbol{\Lambda}_{\mathbf{x}}^{-1/2}\mathbf{V}_{\mathbf{x}}^T$.

Step 2. Form set of FO cumulants of $\mathbf{z}$: $\mathbf{c}_{\mathbf{z}} = cum\left(z_i, z_j, z_k, z_l\right)$.

Form n "cumulant matrices"

$$\left[\mathbf{C}_{\mathbf{z}}(\mathbf{u}_p\mathbf{u}_p^T)\right]_{ij} = \sum_{k,l=1}^{n} cum\left(z_i, z_j, z_k, z_l\right)\left[\mathbf{u}_p\mathbf{u}_p^T\right]_{kl}$$

To compute $\left[\mathbf{C}_{\mathbf{z}}(\mathbf{u}_p\mathbf{u}_p^T)\right]$ we need initial value for $\mathbf{U}$ that is assumed to be obtained from JAD of $\left[\mathbf{C}_{\mathbf{z}}(\mathbf{u}_p\mathbf{u}_p^T)\right]$?

JADE

Initial estimate of **U** can be obtained by EVD of:

$$\left[\mathbf{C}_{\mathbf{z}}(\mathbf{I}_n)\right] = \underbrace{E\left[\mathbf{z}\mathbf{z}^T\mathbf{z}\mathbf{z}^T\right]}_{\text{This is obtained because } \{z_i\}_{i=1}^n \text{ are uncorrelated and zero mean.}} = \mathbf{U}\Lambda\mathbf{U}^T$$

Step 3. Estimate **U** as a solution of JAD of $\mathbf{C}_{\mathbf{z}}(\hat{\mathbf{u}}_p\hat{\mathbf{u}}_p^T) \quad p=1,\ldots,n$.

This amounts to minimize sum of squares of off-diagonal terms:

$$\mathbf{U}\mathbf{C}_{\mathbf{z}}(\hat{\mathbf{u}}_p\hat{\mathbf{u}}_p^T)\mathbf{U}^T \quad p=1,\ldots,n$$

That is equivalent to maximize: .

$$J(\mathbf{U}) = \sum_{p=1}^{n}\left\| diag\left(\mathbf{U}\mathbf{C}_{\mathbf{z}}(\hat{\mathbf{u}}_p\hat{\mathbf{u}}_p^T)\mathbf{U}^T\right)\right\|^2$$

JADE

It can be shown (after complex algebraic manipulations) that the final outcome of the JADE algorithm is:

$$J(\mathbf{U}) = \sum_{i,j,k,l \neq iikl}^{n} cum(y_i, y_j, y_k, y_l)^2$$

where:

$$\mathbf{y}(t) = \mathbf{U}\mathbf{z}(t) = \mathbf{U}\mathbf{Q}\mathbf{x}(t)$$

Kernel ICA algorithm[7]

Kernel ICA algorithm is obtained by minimizing canonical correlations in the reproducing kernel Hilbert space (RKHS). There the *kernel trick* is used to avoid explicit mappings from data to image space. Thus, kernel functions are evaluated in the input data space. This leads to the KCCA algorithm.

Second contrast function used in RKHS is approximation of mutual information based on relationship between mutual information and canonical correlation for Gaussian variable. This leads to KGV algorithm.

For two random variables x_1 and x_2 we want to maximize correlation defined on a function space $F : \mathbb{R} \to \mathbb{R}$ between $f_1(x_1)$ and $f_2(x_2)$ where f_1 and f_2 range over F:

$$\rho_F = \max_{f_1, f_2 \in F} corr\left(f_1(x_1), f_2(x_2) \right)$$

ρ_F is called F-correlation. When x_1 and x_2 are independent ρ_F is zero. The converse is also true if function space is rich enough.

[7]F. R. Bach and M. I. Jordan "Kernel Independent Component Analysis," *Journal of Machine Learning Research* 3, pp.1-48, 2002.

Kernel ICA algorithm

Thus, we want to minimize nonlinear correlation between x_1 and x_2 looking for optimal functions f_1 and f_2. Concept of RKHS makes this idea computationally tractable.

$$corr(f_1(x_1), f_2(x_2)) = E[f_1(x_1) f_2(x_2)] = E[f_1(x_1)] E[f_2(x_2)]$$

$$E[f_1(x_1)] E[f_2(x_2)] = E\left[\sum_i a_i x_1^i\right] E\left[\sum_j b_j x_2^j\right] = \sum_i \sum_j a_i b_j E[x_1^i x_2^j]$$

Let F is space of functions, called *feature space*, be RKHS on $\mathbb{R}$. Let $K(x,y)$ be associated kernel function that induces F (proper choice is Gaussian kernel). Let $\Phi(x)=K(.,x)$ be a feature map, where $K(.,x)$ is function in F for each x. The reproducing property implies:

$$f(x) = \langle \Phi(x), f \rangle, \quad \forall f \in F, \ \forall x \in \mathbb{R}$$

It follows

$$corr(f_1(x_1), f_2(x_2)) = corr(\langle \Phi(x_1), f_1 \rangle, \langle \Phi(x_2), f_2 \rangle)$$

Kernel ICA algorithm

Hence *F*-correlation is maximal possible correlation between one dimensional linear projections of $\Phi(x_1)$ and $\Phi(x_2)$. That is definition of the first canonical correlation between $\Phi(x_1)$ and $\Phi(x_2)$.

It follows that ICA-contrast function can be constructed by computing canonical correlation in function space *F*. That is important because canonical correlation analysis (CCA) is reduced to generalized eigen-value problem.

Use of RKHS enables not to work with functions $f \in F$ explicitly but implicitly through the use of kernel function $K(.,x)$ such that: $\forall x_i \in \mathbb{R}^p$ Gram matrix $K_{ij}(x_i,x_j)$ is positive definite.

For *K* there is associate feature map $\Phi : X \mapsto F$ such that

$$K(x,y) = \langle \Phi(x), \Phi(y) \rangle \; : \; \text{kernel trick}$$

Kernel trick enables calculations in the input data space $X \subset \mathbb{R}^p$ without activating feature map Φ that for some kernels (Gaussian) is infinite.

Kernel ICA algorithm

If we now assume that function space *F* is RKHS associated with $K:\{K(.,x):x\in X\}$ then the reproducing property $f(x)=\langle K(.,x),f\rangle$ for $\Phi(x)=K(.,x)$ yields:

$$\langle \Phi(x),\Phi(y)\rangle = \langle K(.,x),K(.,y)\rangle = K(x,y)$$

For translation invariant kernels (example: Gaussian kernel): $K(x,y)=k(x-y)$ where *k* is function $k:\mathbb{R}^p\mapsto\mathbb{R}$ (that must be real and positive). The function space *F* induced by kernel $k(x-y)$ has infinite dimension and is composed of functions $f\in L^2\left(\mathbb{R}^p\right)$ such that:

$$\int_{\mathbb{R}^p}\frac{\left|\hat{f}(\omega)\right|^2}{\nu(\omega)}d\omega<\infty$$

Where $\hat{f}(\omega)$ is Fourier transform of *f* and $\nu(\omega)$ is Fourier transform of $k(x-y)$. For Gaussian kernel in *p*-dimensional space:

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

Kernel ICA algorithm

The Fourier transform of G_σ is

$$\nu(\omega) = \left(2\pi\sigma^2\right)^{p/2} \exp\left(-\frac{\sigma^2}{2}\|\omega\|^2\right)$$

In this case F is a space of smooth functions whose Fourier transform that decays rapidly.

Thus, since the kernel function induces functional space, minimization of canonical correlations w.r.t. functions across F is replaced by minimization of *kernelized* canonical correlations w.r.t. to samples across X. Kernel trick enables numerical solution of the CCA problem in RKHS induced by the kernel.

Thus, minimizing canonical correlations in kernel induced function space F is equivalent to minimizing nonlinear correlations between random variables that are induced by the *optimal nonlinear* functions. This will minimize statistical independence between the random variables in the input data space X.

Kernel ICA algorithm

Minimization of canonical correlations in RKHS implies:

$$\max_{f_1, f_2 \in F} corr\left(f_1(x_1), f_2(x_2) \right) = \max_{f_1, f_2 \in F} corr\left(\left\langle \Phi_1(x_1), f_1 \right\rangle, \left\langle \Phi_2(x_2), f_2 \right\rangle \right)$$

$$\max_{f_1, f_2 \in F} \frac{\operatorname{cov}\left(\left\langle \Phi_1(x_1), f_1 \right\rangle, \left\langle \Phi_2(x_2), f_2 \right\rangle \right)}{\left(\operatorname{var} \left\langle \Phi_1(x_1), f_1 \right\rangle \right)^{1/2} \left(\operatorname{var} \left\langle \Phi_2(x_2), f_2 \right\rangle \right)^{1/2}}$$

In practice we need to work with empirical linear projections of feature maps on a finite set of samples: $x_1^n, x_2^n, \quad n = 1, \ldots, N$.

$$\operatorname{cov}\left(\left\langle \Phi_1(x_1), f_1 \right\rangle, \left\langle \Phi_2(x_2), f_2 \right\rangle \right) \approx \frac{1}{N} \sum_{k=1}^{N} \left\langle \Phi_1(x_1^k), f_1 \right\rangle, \left\langle \Phi_2(x_2^k), f_2 \right\rangle$$

Kernel ICA algorithm

If S_1 and S_2 are linear spaces spanned by images of data points we can express f_1 and f_2 as:

$$f_1 = \sum_{k=1}^{N} \alpha_1^k \Phi_1(x_1^k) + f_1^{\perp}$$

$$f_2 = \sum_{k=1}^{N} \alpha_2^k \Phi_2(x_2^k) + f_2^{\perp}$$

where $f_1^{\perp}$ and $f_2^{\perp}$ are orthognal to S_1 and S_2 and represent approximation errors.

Kernel ICA algorithm

Now, empirical covariance of the linear projections in feature space can be estimated as:

$$\operatorname{cov}\left(\left\langle \Phi_1(x_1), f_1 \right\rangle, \left\langle \Phi_2(x_2), f_2 \right\rangle\right)$$

$$\approx \frac{1}{N}\sum_{k=1}^{N}\left\langle \Phi_1(x_1^k), \sum_{k=1}^{N}\alpha_1^k \Phi_1(x_1^k) \right\rangle, \left\langle \Phi_2(x_2^k), \sum_{k=1}^{N}\alpha_2^k \Phi_2(x_2^k) \right\rangle$$

$$= \frac{1}{N}\sum_{k=1}^{N}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_1^i K_1\left(x_1^i, x_1^k\right) K_2\left(x_2^i, x_2^k\right)\alpha_2^j$$

$$= \frac{1}{N}\boldsymbol{\alpha}_1^T \mathbf{K}_1 \mathbf{K}_2 \boldsymbol{\alpha}_2$$

$\mathbf{K}_1$ and $\mathbf{K}_2$ are Gram matrices associated with data sets $x_1^n, x_2^n, \quad n = 1, \ldots, N$

Kernel ICA algorithm

Empirical variance are obtained as:

$$\operatorname{var}\left\langle \Phi_1(x_1), f_1 \right\rangle \approx \frac{1}{N} \boldsymbol{\alpha}_1^T \mathbf{K}_1 \mathbf{K}_1 \boldsymbol{\alpha}_1$$

$$\operatorname{var}\left\langle \Phi_2(x_2), f_2 \right\rangle \approx \frac{1}{N} \boldsymbol{\alpha}_2^T \mathbf{K}_2 \mathbf{K}_2 \boldsymbol{\alpha}_2$$

Putting everything together yields:

$$\hat{\rho}_F\left(\mathbf{K}_1, \mathbf{K}_2\right) = \max_{\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2 \in \mathbb{R}^N} \frac{\boldsymbol{\alpha}_1^T \mathbf{K}_1 \mathbf{K}_2 \boldsymbol{\alpha}_2}{\left(\boldsymbol{\alpha}_1^T \mathbf{K}_1 \mathbf{K}_1 \boldsymbol{\alpha}_1\right)^{1/2} \left(\boldsymbol{\alpha}_2^T \mathbf{K}_2 \mathbf{K}_2 \boldsymbol{\alpha}_2\right)^{1/2}}$$

Kernel ICA algorithm

This is transformed into following generalized eigen-value problem:

$$\begin{pmatrix} \mathbf{0} & \mathbf{K}_1\mathbf{K}_2 \\ \mathbf{K}_2\mathbf{K}_1 & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_1 \\ \boldsymbol{\alpha}_2 \end{pmatrix} = \rho \begin{pmatrix} \mathbf{K}_1^2 & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_2^2 \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_1 \\ \boldsymbol{\alpha}_2 \end{pmatrix}$$

Above problem is generalized to more than two variables which leads to KCCA algorithm.

MATLAB code for KCCA and KGV can be downloaded from:
http://www.cs.berkeley.edu/~fbach/kernel-ica/

Comparison between basic ICA methods

In Ref. [24] representative ICA algorithms were compared for different distributions. Performance measure was Amari's error which measures distance between **Q=WA** and **P** where **P** is general permutation matrix :

$$d(\mathbf{Q},\mathbf{P}) = \frac{1}{2N}\sum_{n=1}^{N}\left(\frac{\sum_{m=1}^{N}|q_{nm}|}{\max_m |q_{nm}|}-1\right)$$

$$+\frac{1}{2N}\sum_{m=1}^{N}\left(\frac{\sum_{n=1}^{N}|q_{nm}|}{\max_n |q_{nm}|}-1\right)$$

This measure is always between 0 and N-1.

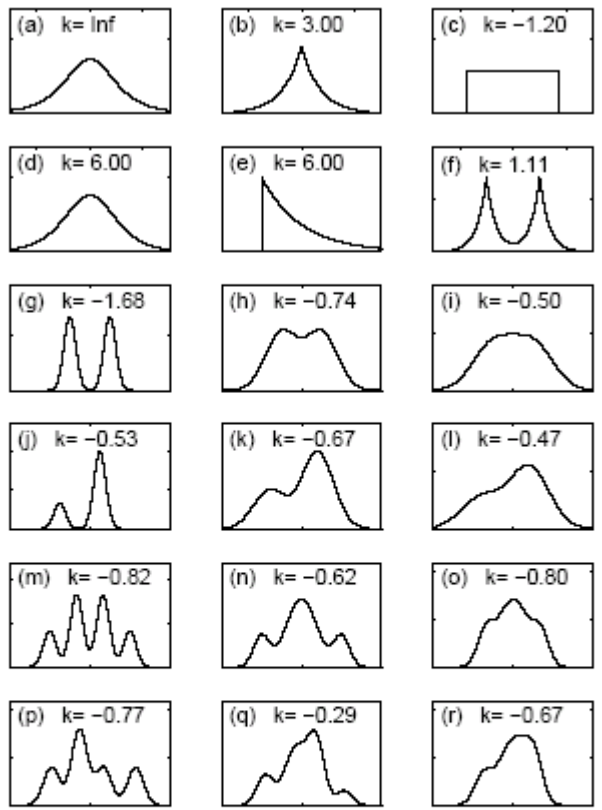

Figure 5: Probability density functions of sources with their kurtoses: (a) Student with 3 degrees of freedom; (b) double exponential; (c) uniform; (d) Student with 5 degrees of freedom; (e) exponential; (f) mixture of two double exponentials; (g)-(h)-(i) symmetric mixtures of two Gaussians: multimodal, transitional and unimodal; (j)-(k)-(l) nonsymmetric mixtures of two Gaussians, multimodal, transitional and unimodal; (m)-(n)-(o) symmetric mixtures of four Gaussians: multimodal, transitional and unimodal; (p)-(q)-(r) nonsymmetric mixtures of four Gaussians: multimodal, transitional and unimodal.

Comparison between basic ICA methods[7]

pdfs	F-ica	Jade	Imax	KCCA	KGV
a	7.2	6.4	50.0	7.7	6.5
b	13.1	11.0	58.4	7.9	7.1
c	4.7	3.6	15.2	5.5	4.4
d	12.8	10.9	54.3	13.8	11.8
e	8.8	8.0	70.3	3.7	3.3
f	7.1	5.2	9.5	3.4	3.2
g	3.4	2.7	19.2	2.7	2.5
h	13.4	9.1	29.7	9.8	8.3
i	24.9	18.4	36.2	23.5	21.6
j	20.6	16.5	52.1	3.2	3.1
k	13.4	8.1	28.4	6.0	5.0
l	27.7	20.0	35.0	12.6	10.2
m	8.6	6.2	25.6	13.6	10.6
n	12.9	9.1	34.3	15.1	9.2
o	9.6	6.9	24.6	11.9	9.6
p	9.2	6.0	27.4	8.1	6.2
q	41.2	34.4	40.6	11.8	8.2
r	14.3	9.2	33.9	9.0	8.0
mean	**14.1**	**10.6**	**35.8**	**9.4**	**7.7**
rand	**10.8**	**8.6**	**30.4**	**6.9**	**5.4**

pdfs	F-ica	Jade	Imax	KCCA	KGV
a	4.4	3.7	1.8	3.7	3.0
b	5.8	4.1	3.4	3.7	2.9
c	2.3	1.9	2.0	2.7	2.4
d	6.4	6.1	6.9	7.1	5.7
e	4.9	3.9	3.2	1.7	1.5
f	3.6	2.7	1.0	1.7	1.5
g	1.7	1.4	0.5	1.4	1.3
h	5.5	3.9	3.2	4.3	3.6
i	8.7	7.2	6.8	7.8	6.5
j	6.7	4.6	57.6	1.4	1.3
k	5.7	4.0	3.5	3.2	2.6
l	12.1	7.2	10.4	4.8	4.2
m	3.6	2.9	4.2	6.3	4.6
n	5.4	3.5	30.6	7.6	3.0
o	4.7	3.3	4.4	5.1	4.3
p	4.1	3.1	7.4	3.8	3.0
q	22.9	15.8	40.9	5.1	3.9
r	6.6	4.4	4.9	4.3	3.6
mean	**6.4**	**4.6**	**10.7**	**4.2**	**3.3**
rand	**5.3**	**4.3**	**6.9**	**3.0**	**2.4**

Table 1: The Amari errors (multiplied by 100) for two-component ICA with 250 samples (left) and 1000 samples (right). For each pdf (from a to r), averages over 100 replicates are presented. The overall mean is calculated in the row labeled **mean**. The **rand** row presents the average over 1000 replications when two (generally different) pdfs were chosen uniformly at random among the 18 possible pdfs.