Machine Intelligence 2

2.4 Fast ICA

Prof. Dr. Klaus Obermayer

Fachgebiet Neuronale Informationsverarbeitung (NI)

SS 2018

FastICA

ICA and Projection Pursuit

ICA and Projection Pursuit

PCA:

- $\rightsquigarrow$ variance criterion
  - useful preprocessing method: decorrelation, but scale sensitive solutions
  - fails in some cases (see previous problem sheets)

Projection Pursuit

- general family of methods
- exploratory data analysis (no explicit data model $\rightarrow$ visualization)
- techniques to find interesting projections of multidimensional data for criteria beyond variance
- PCA belongs to this family
- ICA

Why not maximize non-Gaussianity?

ICA and non-Gaussianity

Mixing model and recovery of sources

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

mixing matrix $\underline{\mathbf{A}}$, statistically independent sources $\underline{\mathbf{s}}$

$$\widehat{s}_i = \underline{\mathbf{w}}_i^T \underline{\mathbf{x}}$$

estimation of one source through one row vector of an unmixing matrix W

$$\widehat{s}_i = \underline{\mathbf{w}}_i^T \underline{\mathbf{A}}\underline{\mathbf{s}} = \underline{\mathbf{z}}_i^T \underline{\mathbf{s}}$$

linear combination of statistically independent variables

Intuition from the Central Limit Theorem

The distribution of the sum of independent random variables is "more Gaussian" than the original distributions of the random variables.

Searching for the direction of maximum deviation from a Gaussian distribution may recover the original sources.

Preliminaries (repeated)

- Assumption: sources have unit variance (w.l.o.g. since amplitude recovery impossible)

  $\rightsquigarrow$ $\langle \underline{\mathbf{s}}\underline{\mathbf{s}}^T \rangle = \underline{\mathbf{I}}$ or equivalently $\langle \mathrm{s}_i \mathrm{s}_j \rangle = \delta_{ij}$

- Preprocessing step: PCA & whitening

  $\underline{\mathbf{u}}^{(\alpha)} = \underline{\mathbf{\Lambda}}^{-\frac{1}{2}} \underline{\mathbf{E}}^T \underline{\mathbf{x}}^{(\alpha)}, \quad \alpha = 1, \ldots, p$

  i.e., instead of $\underline{\mathbf{x}} = \underline{\mathbf{A}}\underline{\mathbf{s}}$ consider equivalently $\underline{\mathbf{u}} = \underline{\tilde{\mathbf{A}}}\underline{\mathbf{s}} \Leftrightarrow \underline{\mathbf{x}} = \underbrace{\underline{\mathbf{E}}\underline{\mathbf{\Lambda}}^{\frac{1}{2}}\underline{\tilde{\mathbf{A}}}}_{=\underline{\mathbf{A}}}\underline{\mathbf{s}}$

Consequence:

$$\langle \underline{\mathbf{s}}\underline{\mathbf{s}}^T \rangle = \langle \underline{\tilde{\mathbf{A}}}^T \underline{\mathbf{u}}\underline{\mathbf{u}}^T \underline{\tilde{\mathbf{A}}} \rangle = \underline{\tilde{\mathbf{A}}}^T \underbrace{\langle \underline{\mathbf{u}}\underline{\mathbf{u}}^T \rangle}_{=\underline{\mathbf{I}}} \underline{\tilde{\mathbf{A}}} = \underline{\mathbf{I}}$$

$\rightsquigarrow$ The mixing matrix $\underline{\tilde{\mathbf{A}}}$ is orthogonal for whitened data!

$\rightsquigarrow$ Number of free parameters for $\underline{\mathbf{W}}$ is reduced from N^2 to $\frac{(\mathrm{N}-1)\mathrm{N}}{2}$

Kurtosis: Definition & properties

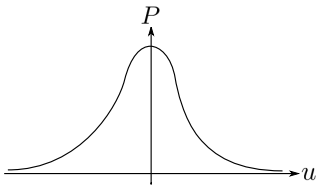
Definition

$$\mathrm{kurt}(x) = \langle x^4 \rangle_{P_x} - 3\Big(\langle x^2 \rangle_{P_x}\Big)^2 \quad \overset{\text{sphered data}}{=} \quad \langle x^4 \rangle_{P_x} - 3$$

Let x_1 and x_2 be two independent random variables; then

$$\begin{aligned} \mathrm{kurt}(x_1 + x_2) &= \mathrm{kurt}(x_1) + \mathrm{kurt}(x_2) \\ \mathrm{kurt}(z_1 x_1) &= z_1^4 \, \mathrm{kurt}(x_1) \end{aligned}$$

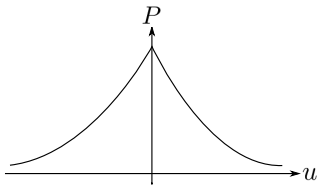
Kurtosis: Definition & properties



$\text{kurt}(x) = 0$

Gaussian PDF
bell shaped

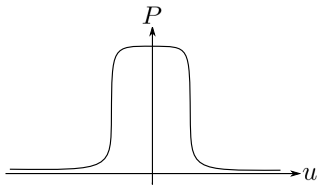
normal



$\text{kurt}(x) > 0$

super-Gaussian
peaky, long tails ("outliers")

Laplace



$\text{kurt}(x) < 0$

sub-Gaussian
bulky, no "outliers"

uniform

Kurtosis: Definition & properties

Two independent sources with $\langle s_i s_j \rangle = \delta_{ij}$

$$\widehat{s} \quad = \quad \underline{\mathbf{W}}^T \underline{\mathbf{x}} \quad = \quad \underline{\mathbf{W}}^T \underline{\mathbf{A}} \cdot \underline{\mathbf{s}} \quad = \quad \underline{\mathbf{z}}^T \underline{\mathbf{s}} \quad = \quad z_1 s_1 + z_2 s_2$$

$$\mathrm{var}(\widehat{s}) \quad = \quad \langle \left(z_1 s_1 + z_2 s_2 \right)^2 \rangle_{P_s} \quad = \quad z_1^2 \langle s_1^2 \rangle + z_2^2 \langle s_2^2 \rangle \quad = \quad z_1^2 + z_2^2$$

$$\begin{aligned} \mathrm{kurt}(\widehat{s}) \quad &= \quad \mathrm{kurt}(z_1 s_1 + z_2 s_2) \\ &= \quad \mathrm{kurt}(z_1 s_1) + \mathrm{kurt}(z_2 s_2) \quad = \quad z_1^4 \, \mathrm{kurt}(s_1) + z_2^4 \, \mathrm{kurt}(s_2) \end{aligned}$$

Kurtosis: search for "interesting" directions

Problem statement

$$|\mathrm{kurt}(\hat{s})| \stackrel{!}{=} \max_{\underline{\mathbf{z}}} \quad \leftarrow \text{search for the direction with extreme kurtosis}$$

$$\mathrm{var}(\hat{s}) = z_1^2 + z_2^2 \stackrel{!}{=} 1 \quad \leftarrow \text{such that data remains sphered}$$

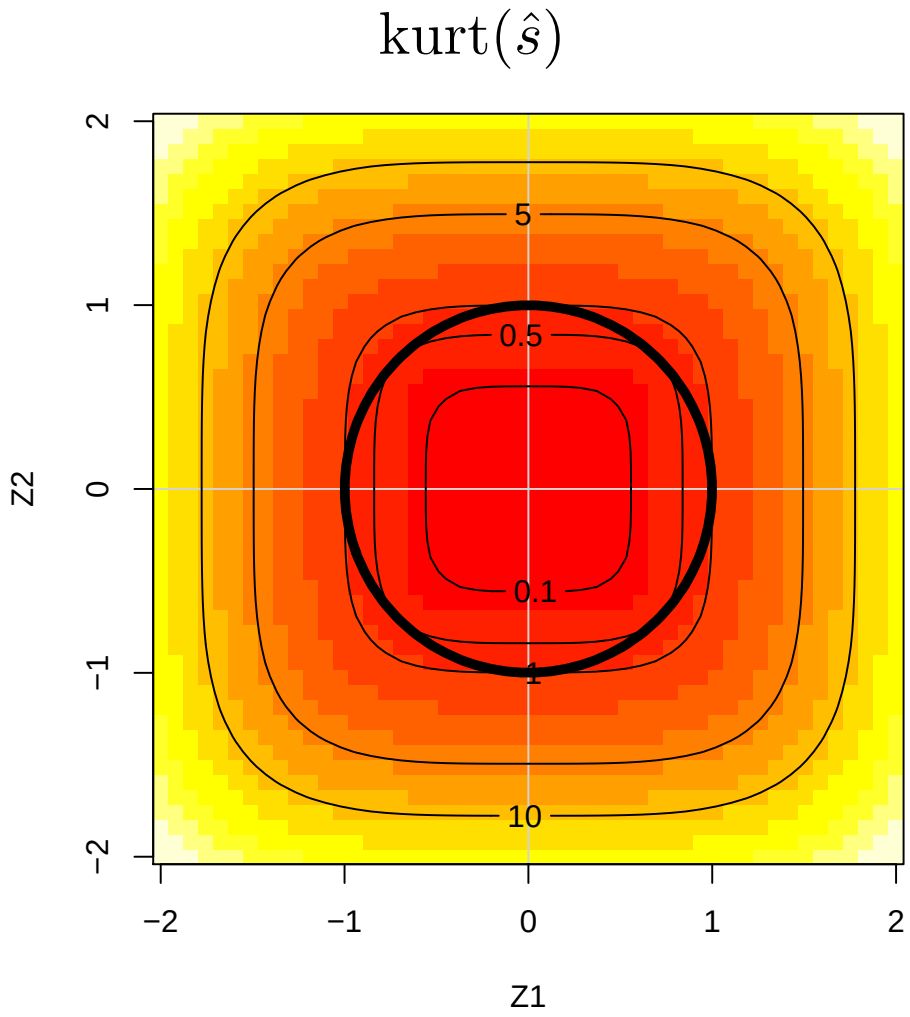
Using a proper Lagrangian, one can show that (*see supplementary material*)

$$\underline{\mathbf{z}}_{\text{opt.}} = \begin{pmatrix} 0 \\ \pm 1 \end{pmatrix} \text{ or } \underline{\mathbf{z}}_{\text{opt.}} = \begin{pmatrix} \pm 1 \\ 0 \end{pmatrix}$$

(there are other local optima, but with lower values for $|\mathrm{kurt}(\hat{s})|$)

- Extrema of the kurtosis correspond to independent sources - if model assumptions are correct.

Illustration of the optimization problem



Toy example $\hat{s} = \underline{\mathbf{z}}^T \underline{\mathbf{s}}$

- $\mathrm{var}(s_1) = \mathrm{var}(s_2) = 1$

$\Rightarrow$ search space: $z_1^2 + z_2^2 = 1$

- $\mathrm{kurt}(s_1) = \mathrm{kurt}(s_2) = 1$

$\Rightarrow$ $\mathrm{kurt}(\hat{s}) = z_1^4 + z_2^4$

$\rightsquigarrow \mathrm{argmax}_z \, \mathrm{kurt}(\hat{s})$:
$z \in \{(0, \pm 1), (\pm 1, 0)\}$

Equivalent optimization problems

$$\hat{\underline{\mathbf{s}}} = \underline{\mathbf{z}}^T \underline{\mathbf{s}} = \underline{\mathbf{w}}^T \underbrace{\underline{\mathbf{A}}\underline{\mathbf{s}}}_{\underline{\mathbf{x}}} = \underline{\mathbf{b}}^T \underbrace{\tilde{\underline{\mathbf{A}}}\underline{\mathbf{s}}}_{\underline{\mathbf{u}}} \qquad \text{kurt}(s_i) \neq \sigma$$

1. $\max_{\underline{\mathbf{z}}} |\,\text{kurt}\,(\underline{\mathbf{z}}^T \underline{\mathbf{s}})| \quad s.t. \quad |\underline{\mathbf{z}}| = 1$
2. $\max_{\underline{\mathbf{w}}} |\,\text{kurt}\,(\underline{\mathbf{w}}^T \underline{\mathbf{x}})| \quad s.t. \quad |\underline{\mathbf{A}}^T \underline{\mathbf{w}}| = 1$
3. $\max_{\underline{\mathbf{b}}} |\,\text{kurt}\,(\underline{\mathbf{b}}^T \underline{\mathbf{u}})| \quad s.t. \quad \underbrace{|\tilde{\underline{\mathbf{A}}}^T \underline{\mathbf{b}}|}_{=|\underline{\mathbf{b}}| \text{ since } \tilde{\underline{\mathbf{A}}} \text{ orthogonal}} = 1$

Methods for kurtosis optimization

$$\frac{\partial |\text{kurt}(\underline{\mathbf{b}}^T \underline{\mathbf{u}})|}{\partial \underline{\mathbf{b}}} = \textbf{see blackboard}$$

$$= 4 \operatorname{sgn}\left[\text{kurt}\left(\underline{\mathbf{b}}^T \underline{\mathbf{u}}\right)\right] \left(\langle \underline{\mathbf{u}} (\underline{\mathbf{b}}^T \underline{\mathbf{u}})^3 \rangle - 3 \underline{\mathbf{b}} \langle (\underline{\mathbf{b}}^T \underline{\mathbf{u}})^2 \rangle \right)$$

$$\stackrel{!}{=} \underline{\mathbf{0}} \quad \text{s.t. } |\underline{\mathbf{b}}| = 1$$

$$\underline{\mathbf{b}}^T \langle \underline{\mathbf{u}} \underline{\mathbf{u}}^T \rangle \underline{\mathbf{b}} = \underline{\mathbf{b}}^T \underline{\mathbf{I}} \underline{\mathbf{b}} = |\underline{\mathbf{b}}|^2$$

last term: changes only length of $\underline{\mathbf{b}} \rightsquigarrow$ can be removed due to constraint $|\underline{\mathbf{b}}| = 1$

$$4 \operatorname{sgn}\left[\text{kurt}\left(\underline{\mathbf{b}}^T \underline{\mathbf{u}}\right)\right] \langle \underline{\mathbf{u}} (\underline{\mathbf{b}}^T \underline{\mathbf{u}})^3 \rangle = \underline{\mathbf{0}} \quad \text{s.t. } |\underline{\mathbf{b}}| = 1$$

Methods for kurtosis optimization

I. batch learning:

Initialization: random vector $\underline{\mathbf{b}}$ of unit length

$$\begin{aligned} \Delta\underline{\mathbf{b}} &= \varepsilon \operatorname{sgn}\left[\operatorname{kurt}\left(\underline{\mathbf{b}}^T\underline{\mathbf{u}}\right)\right]\left\langle \underline{\mathbf{u}}(\underline{\mathbf{b}}^T\underline{\mathbf{u}})^3\right\rangle \\ \underline{\mathbf{b}} &\leftarrow \underline{\mathbf{b}}/|\underline{\mathbf{b}}| \text{ (normalization to fulfill constraint } |\underline{\mathbf{b}}| = 1) \end{aligned}$$

ERM: replace expectations (kurt and $\langle\cdot\rangle$) by the respective empirical averages

Methods for kurtosis optimization

II. online learning:

Initialization: random vector $\underline{\mathbf{b}}$ of unit length, $\gamma = 0$

choose a data point $\underline{\mathbf{u}}$

$$\Delta\underline{\mathbf{b}} = \varepsilon \operatorname{sgn}(\gamma)\underline{\mathbf{u}}(\underline{\mathbf{b}}^T\underline{\mathbf{u}})^3 \quad \text{(weight update per data point)}$$

$$\Delta\gamma = \eta\left[(\underline{\mathbf{b}}^T\underline{\mathbf{u}})^4 - 3 - \gamma\right] \quad \text{(running average of the kurtosis with learning rate } \eta\text{)}$$

$$\underline{\mathbf{b}} \leftarrow \underline{\mathbf{b}}/|\underline{\mathbf{b}}|$$

Method for kurtosis optimization

III. fixed-point algorithm (Kurtosis-based fastICA)

equilibrium point of gradient descent: $\underline{\mathbf{b}} \propto \Delta\underline{\mathbf{b}}$
$\rightsquigarrow \underline{\mathbf{b}} \propto \langle \underline{\mathbf{u}}(\underline{\mathbf{b}}^T\underline{\mathbf{u}})^3\rangle - 3|\underline{\mathbf{b}}|^2\underline{\mathbf{b}}$

in combination with normalization we have

$$\begin{aligned} \underline{\mathbf{b}} &\leftarrow \langle \underline{\mathbf{u}}(\underline{\mathbf{b}}^T\underline{\mathbf{u}})^3\rangle - 3\underline{\mathbf{b}} \\ \underline{\mathbf{b}} &\leftarrow \underline{\mathbf{b}}/|\underline{\mathbf{b}}| \end{aligned}$$

yielding the Kurtosis-fastICA-algorithm for whitened data $\underline{\mathbf{u}}^{(\alpha)}, \quad \alpha = 1, \ldots, p$

Initialization: random vector $\underline{\mathbf{b}}$ of unit length

$$\begin{aligned} \underline{\mathbf{b}} &\leftarrow \frac{1}{p}\sum_{\alpha=1}^{p} \underline{\mathbf{u}}^{(\alpha)}(\underline{\mathbf{b}}^T\underline{\mathbf{u}}^{(\alpha)})^3 - 3\underline{\mathbf{b}} \\ \underline{\mathbf{b}} &\leftarrow \underline{\mathbf{b}}/|\underline{\mathbf{b}}| \end{aligned}$$

remarks: 1.) no learning rate parameter!
2.) fast & robust convergence

Problems with kurtosis

Definition

$$\mathrm{kurt}(x) = \langle x^4 \rangle_{P_x} - 3\left(\langle x^2 \rangle_{P_x}\right)^2 \overset{\text{sphered data}}{=} \langle x^4 \rangle_{P_x} - 3$$

Let x_1 and x_2 be two independent random variables; then

$$\begin{aligned} \mathrm{kurt}(x_1 + x_2) &= \mathrm{kurt}(x_1) + \mathrm{kurt}(x_2) \\ \mathrm{kurt}(z_1 x_1) &= z_1^4 \,\mathrm{kurt}(x_1) \end{aligned}$$

Kurtosis is not ideal because it can be sensitive to outliers:

- easy to compute, but not "robust"

Example

- Sample of 1000 values from a distribution with mean = 0 and std=1
- One observation $x = 10$

$\rightsquigarrow$ $\mathrm{kurt}(x) \geq 10^4/1000 - 3 = 7$

Negentropy

$$H(\hat{s}) := -\int p(\hat{s}) \log p(\hat{s}) d\hat{s} \qquad \text{(differential entropy)}$$

Definition of the negentropy

$$J(\hat{s}) := \underbrace{H(\hat{s})_{\mathcal{N}}}_{\substack{\text{entropy of a Gaussian} \\ \text{distribution with} \\ \text{same variance}}} - \underbrace{H(\hat{s})}_{\substack{\text{entropy of true} \\ \text{distribution} \\ \text{(variance } \sigma^2\text{)}}}$$

- $\Rightarrow$ theoretically well motivated measure
- $\Rightarrow$ non-negative
- $\Rightarrow$ scale-invariant: $J(\alpha\hat{s}) = J(\hat{s}), \quad \forall \alpha \neq 0$ (cf. exercise sheet)
- $\Rightarrow$ **Problem:** requires estimation of density $p(\hat{s})$

Approximations to negentropy

Polynomial density expansion (cumulants):

$$J(\hat{s}) \approx \frac{1}{12}\langle(\hat{s})^3\rangle^2 + \frac{1}{48}(\mathrm{kurt}\,(\hat{s}))^2 + \text{higher order terms}$$

$\rightarrow$ for symmetric distributions optimizing this is equivalent with optimizing $|\,\mathrm{kurt}\,(\hat{s})|$ sharing the outlier sensitivity

"Nonpolynomial moments" contrast functions G:

$$J(\hat{s}) \approx \left(\langle G(\hat{s})\rangle - \langle G(u_{\text{Gauss}})\rangle\right)^2$$

Common contrast functions:

$$G_1(\hat{s}) = \frac{1}{a}\log\cosh(a \cdot \hat{s}) \qquad G_1'(\hat{s}) = \tanh(a\hat{s}) \qquad G_1''(\hat{s}) = a(1 - \tanh^2(a\hat{s}))$$

general purpose

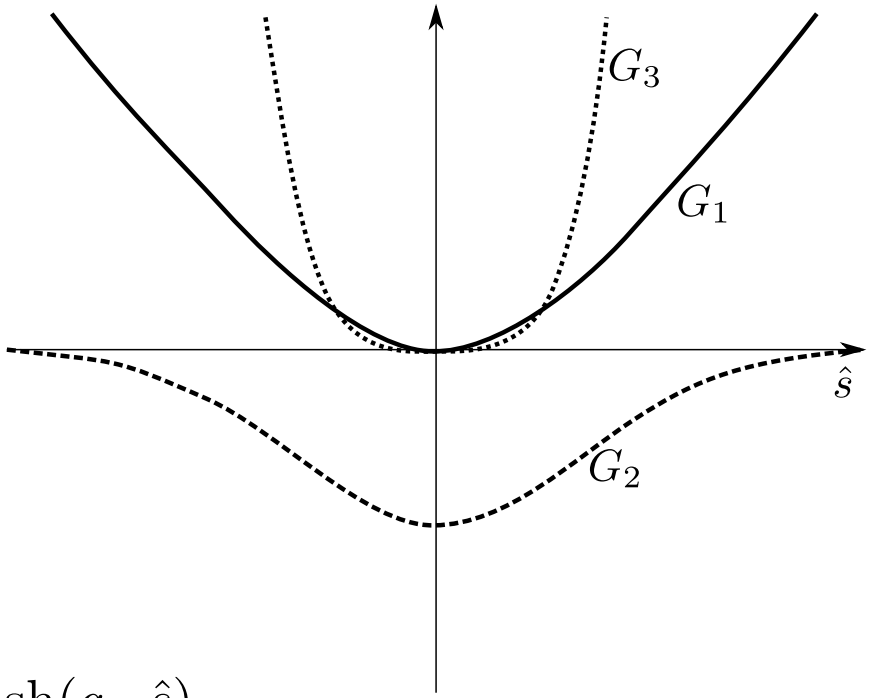
$$G_2(\hat{s}) = -\exp\left(-\frac{(\hat{s})^2}{2}\right) \qquad G_2'(\hat{s}) = \hat{s}\exp(-\frac{(\hat{s})^2}{2}) \qquad G_2''(\hat{s}) = (1 - (\hat{s})^2)\exp(-\frac{(\hat{s})^2}{2})$$

good for "super"-Gaussian sources with many "outliers"

$$G_3(\hat{s}) = \frac{1}{4}(\hat{s})^4 \qquad G_3'(\hat{s}) = (\hat{s})^3 \qquad G_3''(\hat{s}) = 3(\hat{s})^2$$

kurtosis good for "sub"-Gaussian sources with few "outliers"

Common contrast functions



- $G_1(\hat{s}) = \frac{1}{a} \log \cosh(a \cdot \hat{s})$
- $G_2(\hat{s}) = -\exp\left(-\frac{(\hat{s})^2}{2}\right)$
- $G_3(\hat{s}) = \frac{1}{4}(\hat{s})^4$

Source: Hyvärinen, 2001

$\rightsquigarrow$ less sensitive to tail of data distribution than kurtosis

General result: any even, non-constant and non-quadratic (contrast) function G can be used for ICA

Methods for optimizing contrast functions

$$\max_{\underline{\mathbf{b}}} J(\underline{\mathbf{b}}^T \underline{\mathbf{u}}) = \max_{\underline{\mathbf{b}}} \left(\langle G(\underline{\mathbf{b}}^T \underline{\mathbf{u}}) \rangle_{P_{(u)}} - \langle G(u_{\text{Gauss.}}) \rangle_{\mathcal{N}_{(0,1)}} \right)^2$$

I. batch learning:

Initialization: random vector $\underline{\mathbf{b}}$ of unit length

$$\overset{\frac{\partial}{\partial \underline{\mathbf{b}}}}{\rightsquigarrow} \Delta \underline{\mathbf{b}} = \varepsilon \overbrace{\left\{ \langle G(\underline{\mathbf{b}}^T \underline{\mathbf{u}}) \rangle_{P_{(u)}} - \langle G(u_{\text{Gauss.}}) \rangle_{\mathcal{N}_{(0,1)}} \right\}}^{=:\gamma} \langle \underline{\mathbf{u}} \cdot G'(\underline{\mathbf{b}}^T \underline{\mathbf{u}}) \rangle_{P_{(u)}}$$

$$\underline{\mathbf{b}} \leftarrow \underline{\mathbf{b}} / |\underline{\mathbf{b}}|$$

remarks:

1.) for $G = \frac{1}{4}(\hat{s})^4$ the kurtosis optimization (gradient-based) is recovered

2.) ERM: mathematical expectations $\rightarrow$ empirical average

Methods for optimizing contrast functions

$$\max_{\underline{\mathbf{b}}} J(\underline{\mathbf{b}}^T \underline{\mathbf{u}}) = \max_{\underline{\mathbf{b}}} \left(\langle G(\underline{\mathbf{b}}^T \underline{\mathbf{u}}) \rangle_{P_{(u)}} - \langle G(u_{\text{Gauss.}}) \rangle_{\mathcal{N}_{(0,1)}} \right)^2$$

II. online learning:

Initialization: random vector $\underline{\mathbf{b}}$ of unit length, $\gamma = 0$

choose a data point $\underline{\mathbf{u}}$

$$\begin{aligned} \Delta\underline{\mathbf{b}} &= \varepsilon \cdot \gamma \cdot \underline{\mathbf{u}} \cdot G'(\underline{\mathbf{b}}^T \underline{\mathbf{u}}) \\ \underline{\mathbf{b}} &\leftarrow \underline{\mathbf{b}}/|\underline{\mathbf{b}}| \\ \Delta\gamma &= \eta(G(\underline{\mathbf{b}}^T \underline{\mathbf{u}}) - \langle G(u_{\text{Gauss.}}) \rangle_{\mathcal{N}_{(0,1)}} - \gamma) \text{ running estimate of } \gamma \end{aligned}$$

remarks:

1.) often simple expression for $\langle G(u_{\text{Gauss.}}) \rangle_{\mathcal{N}_{(0,1)}}$

2.) γ can be replaced by $\text{sgn}(\gamma)$ that is sometimes known a priori (e.g. speech: highly super-Gaussian)

Methods for optimizing contrast functions

$$\max_{\underline{\mathbf{b}}} J(\underline{\mathbf{b}}^T \underline{\mathbf{u}}) = \max_{\underline{\mathbf{b}}} \left(\langle G(\underline{\mathbf{b}}^T \underline{\mathbf{u}}) \rangle_{P_{(u)}} - \langle G(u_{\text{Gauss.}}) \rangle_{\mathcal{N}_{(0,1)}} \right)^2$$

III. fixed point algorithm ("fastICA"):

- at equilibrium of gradient descent
- suitable manipulation ("adding a zero") to increase speed of convergence

$\rightsquigarrow$ equivalent fixed point iteration

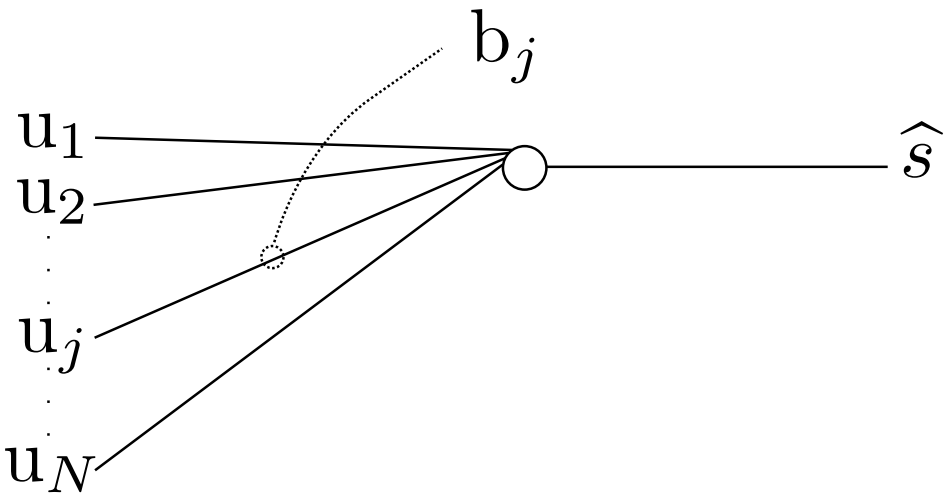
Initialization: random vector $\underline{\mathbf{b}}$ of unit length

$$\begin{aligned} \underline{\mathbf{b}} &\leftarrow \langle \underline{\mathbf{u}} \cdot G'(\underline{\mathbf{b}}^T \underline{\mathbf{u}}) \rangle - \langle G''(\underline{\mathbf{b}}^T \underline{\mathbf{u}}) \rangle \underline{\mathbf{b}} \\ \underline{\mathbf{b}} &\leftarrow \underline{\mathbf{b}} / |\underline{\mathbf{b}}| \end{aligned}$$

remarks:
1.) no learning rate
2.) fast & robust convergence

for details, see *Hyvärinen et.al.(2001); Independent Component Analysis; ch. 8.3.5*

Neural network implementation: One unit



$$\widehat{s} = \underline{\mathbf{b}}^T \underline{\mathbf{u}} = \underline{\mathbf{w}}^T \underline{\mathbf{x}}$$

Goal: weights should converge to direction of maximal "non-Gaussianity"

Fixed point algorithm:

- choose contrast function G
- take gradient of negentropy approximation

$$J(\hat{s}) \approx \left[\left\langle G(\underline{\mathbf{b}}^T \underline{\mathbf{u}})\right\rangle_{P_{(u)}} - \left\langle G(u_{\text{Gauss.}})\right\rangle_{\mathcal{N}_{(0,1)}}\right]^2$$

- at the fixed point, gradient $\nabla_{\underline{\mathbf{b}}} J(\hat{s})$ points in the same direction as $\underline{\mathbf{b}}$

$\rightsquigarrow$ fast approximate Newton iteration scheme to update $\underline{\mathbf{b}}$

for details, see *Hyvärinen et.al.(2001); Independent Component Analysis; ch. 8.3.5*

Neural network implementation: One unit

1. center & whiten data $\underline{\mathbf{u}}^{(\alpha)} = \underline{\mathbf{\Lambda}}^{-\frac{1}{2}} \underline{\mathbf{E}}^T \underline{\mathbf{x}}^{(\alpha)}_{\text{centered}}, \quad \alpha = 1, \ldots, p$
2. choose a random vector $\underline{\mathbf{b}}$ of unit length

   BEGIN loop

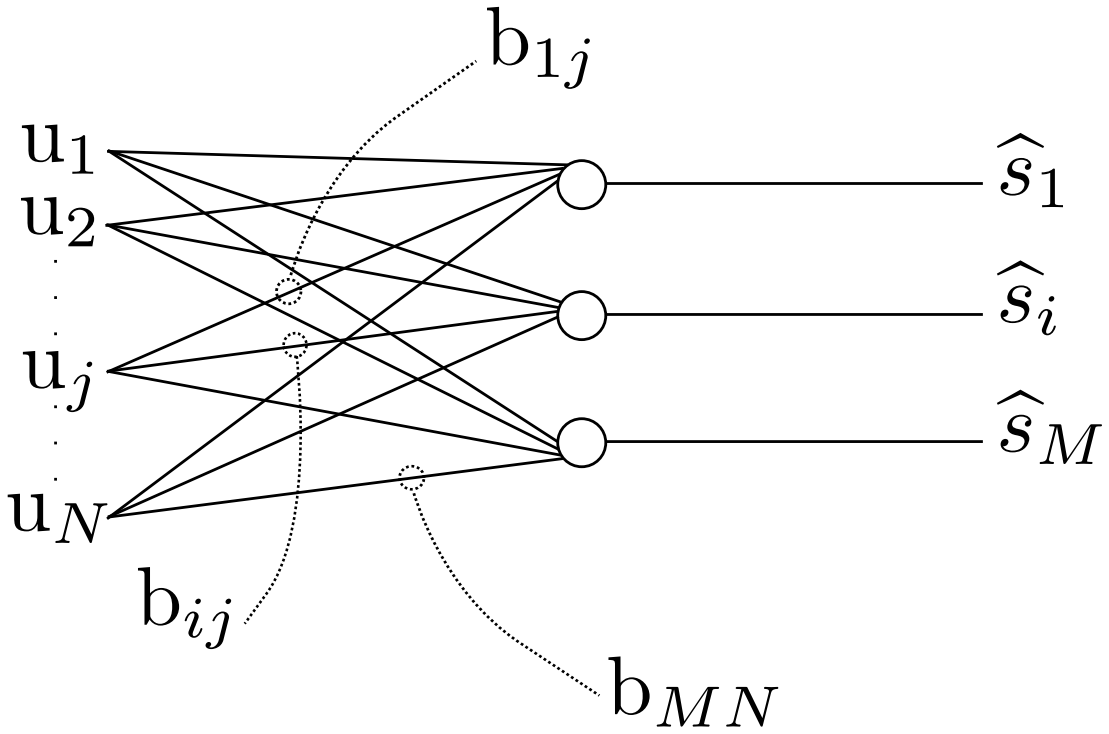
   $$\underline{\mathbf{b}} \leftarrow \frac{1}{p} \left\{ \sum_{\alpha=1}^{p} \underline{\mathbf{u}}^{(\alpha)} G'(\underline{\mathbf{b}}^T \underline{\mathbf{u}}^{(\alpha)}) - \underline{\mathbf{b}} \sum_{\alpha=1}^{p} G''(\underline{\mathbf{b}}^T \underline{\mathbf{u}}^{(\alpha)}) \right\}$$
   $$\underline{\mathbf{b}} \leftarrow \underline{\mathbf{b}} / |\underline{\mathbf{b}}|$$

   END loop

3. if required: obtain weights for original (but centered) data: $\underline{\mathbf{w}} = \underline{\mathbf{E}} \underline{\mathbf{\Lambda}}^{-\frac{1}{2}} \underline{\mathbf{b}}$

$\rightsquigarrow$ faster & more robust than **gradient based learning**, no learning rate

Neural network implementation: Multiple units



$$\underline{\widehat{\mathbf{s}}} = \underline{\mathbf{Bu}} = \underline{\mathbf{Wx}}$$

Generalization to multiple units:

- here: $M = N$
- $\underline{\mathbf{B}} = (\underline{\mathbf{b}}_1, \ldots, \underline{\mathbf{b}}_N)$ orthogonal $\rightsquigarrow$ calculate $\underline{\mathbf{b}}_i, \quad i = 1, \ldots, N,$ independently

$\rightsquigarrow$ (Gram-Schmidt) orthogonalization

Neural network implementation: Multiple units

1. center & whiten data $\underline{\mathbf{u}}^{(\alpha)} = \underline{\mathbf{\Lambda}}^{-\frac{1}{2}} \underline{\mathbf{E}}^T \underline{\mathbf{x}}^{(\alpha)}_{\text{centered}}, \quad \alpha = 1, \ldots$
2. choose random orthogonal matrix $\underline{\mathbf{B}} = (\underline{\mathbf{b}}_1, \ldots, \underline{\mathbf{b}}_N)$

BEGIN loop

DO for all $i = 1, \ldots, N$:

$$\underline{\mathbf{b}}_i \leftarrow \frac{1}{p} \left\{ \sum_{\alpha=1}^{p} \underline{\mathbf{u}}^{(\alpha)} G'(\underline{\mathbf{b}}_i^T \underline{\mathbf{u}}^{(\alpha)}) - \underline{\mathbf{b}}_i \sum_{\alpha=1}^{p} G''(\underline{\mathbf{b}}_i^T \underline{\mathbf{u}}^{(\alpha)}) \right\}$$

$$\left.\begin{array}{l} \underline{\mathbf{B}} \leftarrow \underline{\mathbf{B}} / \max_{i=1,\ldots,N} |\underline{\mathbf{b}}_i| \\ \underline{\mathbf{B}} \leftarrow (\underline{\mathbf{B}}\underline{\mathbf{B}}^T)^{-\frac{1}{2}} \underline{\mathbf{B}} \end{array}\right\} \text{symmetric orthogonalization}$$

END loop

3. if required $\underline{\mathbf{W}} = \underline{\mathbf{E}}\underline{\mathbf{\Lambda}}^{-\frac{1}{2}} \underline{\mathbf{B}}$ in original data space

remark:

1. inverse can be replaced by an iterative procedure
2. algorithm can then also be applied if $M < N$ independent components are required $\rightarrow \underline{\mathbf{B}} = (\underline{\mathbf{b}}_1, \ldots, \underline{\mathbf{b}}_M) \in \mathbb{R}^{N \times M}$

Code and documentation

`http://research.ics.aalto.fi/ica`

- *Hyvärinen, A. et. al. (2001); Independent Component Analysis; ch 8*
- *Hyvaerinen, A. (1999); IEEE Trans. Neural Netw. 10; 626 ff.*

Applications in

- Data Analysis (EEG, fMRI)
- Theoretical Neuroscience (Sparse Coding)

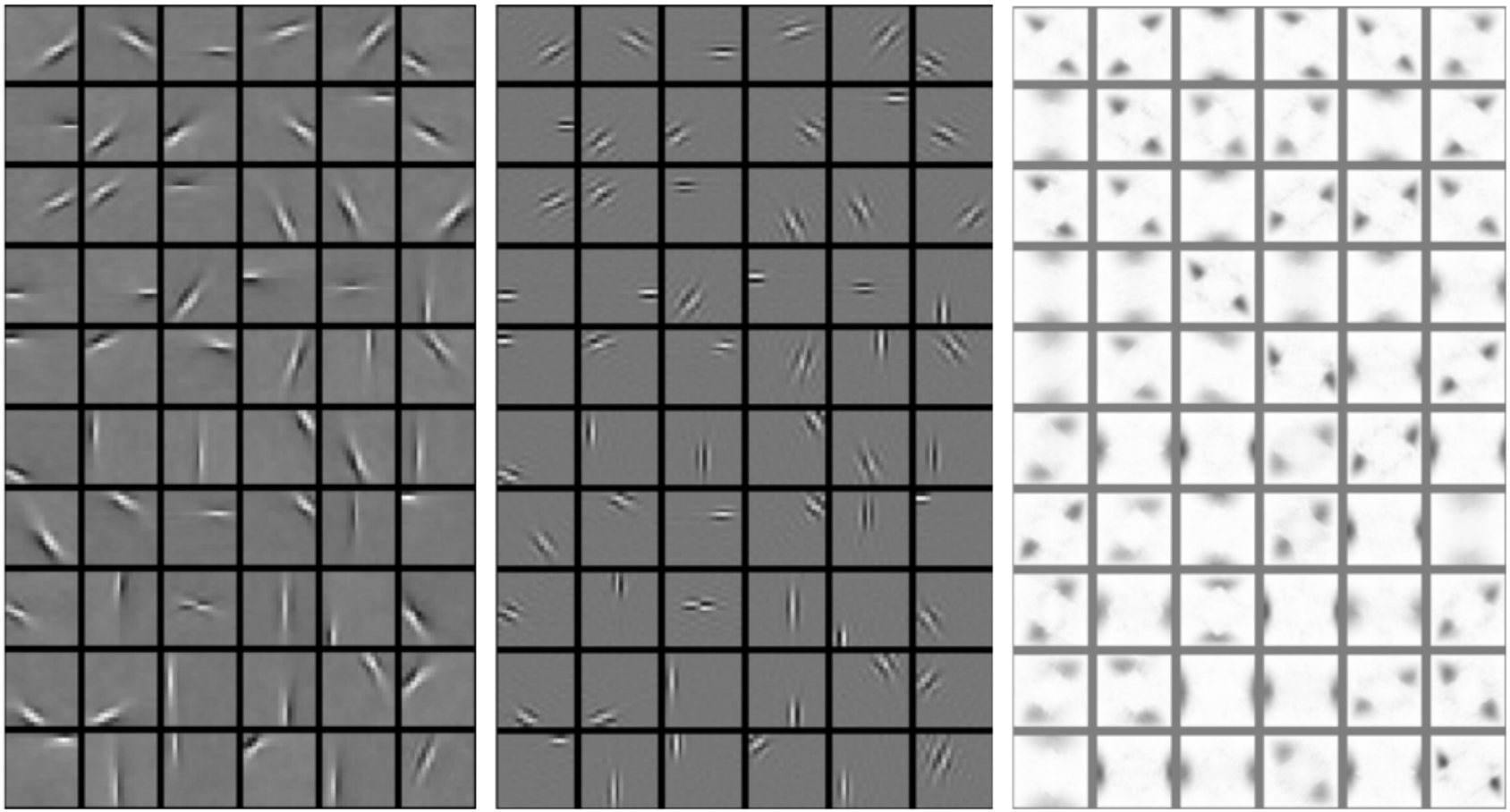
Example: Application to natural images

(van Hateren & van der Schaaf)

- 100 - 120 images
- 120,000 samples, 18 x 18 pixels
- log intensities
- centering, PCA-whitening & kurtosis optimization

$$\underline{\hat{\mathbf{s}}} = \underline{\mathbf{W}} \cdot \underline{\mathbf{x}}$$

Example: Application to natural images

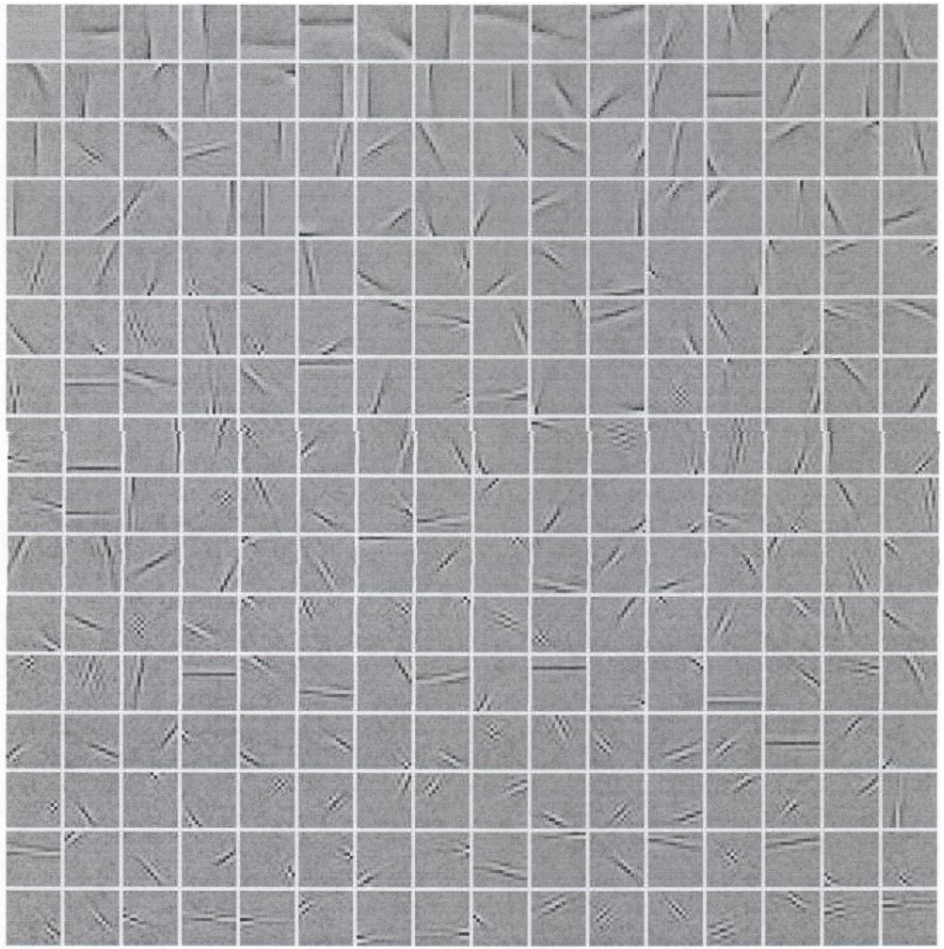


basis vectors $\hat{=}$ "sources"

filters $\hat{=}$ "unmixing weights"

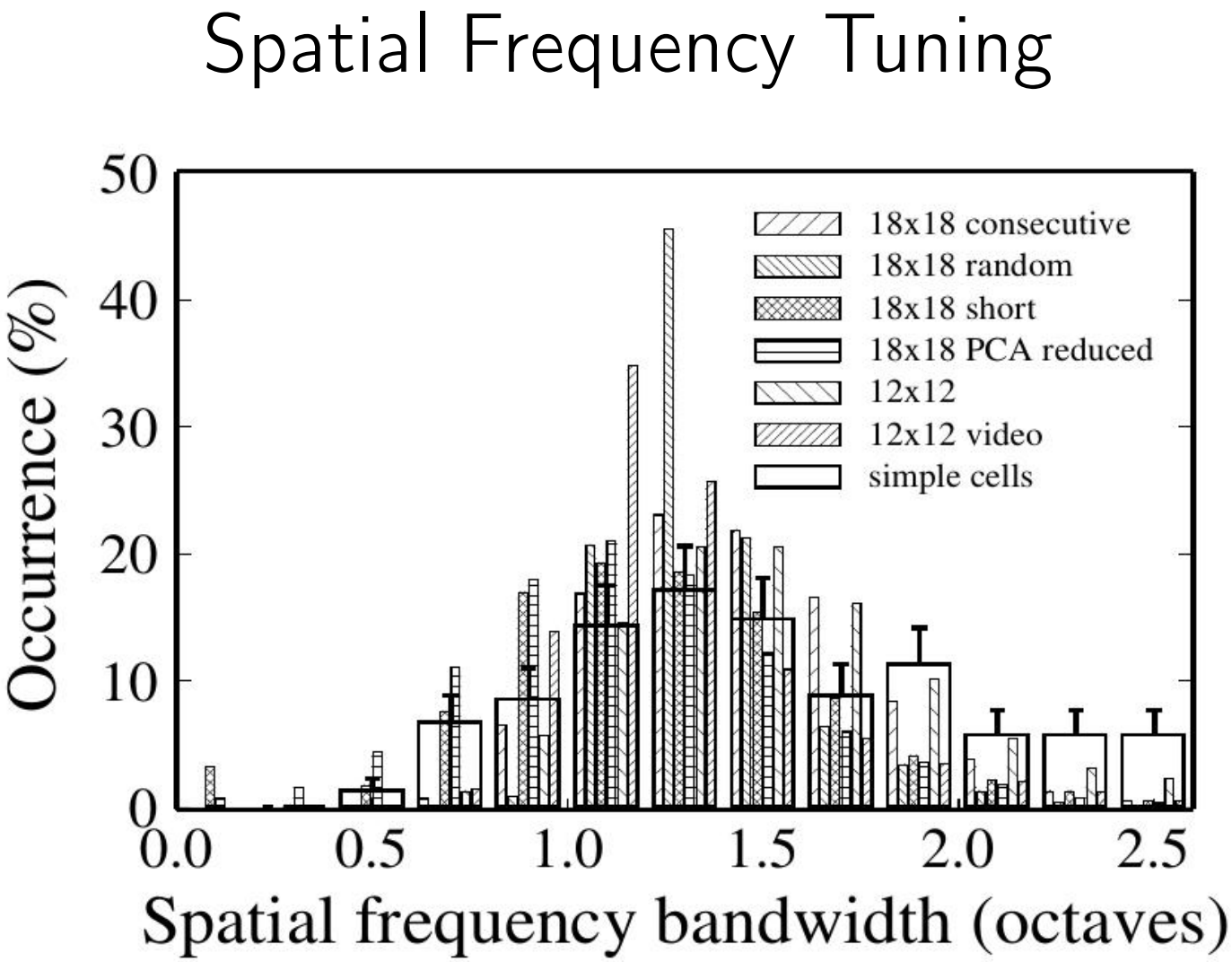
amplitude spectra

Example: Application to natural images



basis vectors

Example: Application to natural images



Sound demo: Blind Source Separation

mixing matrix:

$$\begin{bmatrix} 0.99 & 1.00 & 1.00 & 1.00 & 1.00 & 1.00 \\ 1.00 & 0.99 & 1.00 & 1.00 & 1.00 & 1.00 \\ 1.00 & 1.00 & 0.99 & 1.00 & 1.00 & 1.00 \\ 1.00 & 1.00 & 1.00 & 0.99 & 1.00 & 1.00 \\ 1.00 & 1.00 & 1.00 & 1.00 & 0.99 & 1.00 \\ 1.00 & 1.00 & 1.00 & 1.00 & 1.00 & 0.99 \end{bmatrix}$$

determinant of the mixing matrix: 5.99×10^{-10}

Sound demo: Blind Source Separation

unmixing matrix:

$$\begin{bmatrix} 29.9 & 31.0 & -141.2 & 27.9 & 30.9 & 21.8 \\ -52.1 & 259.3 & -52.1 & -50.2 & -52.5 & -52.9 \\ -82.8 & 15.5 & 18.3 & 23.6 & 17.1 & 8.5 \\ -52.4 & -53.2 & -53.2 & 260.4 & -53.5 & -48.6 \\ -49.5 & -48.7 & -50.6 & -48.4 & 249.1 & -52.3 \\ 62.8 & -59.0 & -62.3 & -65.0 & -58.4 & 307.0 \end{bmatrix}$$

Sound demo: Blind Source Separation

product of mixing and unmixing matrix:

$$\begin{bmatrix} 0.01 & 0.00 & \mathbf{1.72} & 0.03 & 0.00 & 0.09 \\ 0.00 & \mathbf{-3.11} & 0.00 & -0.02 & 0.01 & 0.01 \\ \mathbf{0.99} & 0.01 & -0.02 & -0.07 & -0.01 & 0.09 \\ -0.01 & 0.00 & 0.00 & \mathbf{-3.13} & 0.01 & -0.04 \\ -0.01 & -0.02 & 0.00 & -0.02 & \mathbf{-2.99} & 0.02 \\ 0.02 & -0.02 & 0.01 & 0.04 & -0.03 & \mathbf{-3.68} \end{bmatrix}$$