Machine Learning (part II)

Hebbian Learning and Component Analysis

Angelo Ciaramella

Università Parthenope
DiST Dipartimento di Scienze e Tecnologie

Hebbian learning and NNs

- NNs based on the Hebb's rule
  - Oja's rule
    - computer scientist Erkki Oja
    - Unsupervised learning
    - Symmetric Oja Space
  - Sanger's rule
    - scientist Terence D. Sanger
    - Unsupervised learning
    - Selective Principal Components
  - Generates an algorithm for
    - Principal Component Analysis (PCA)
    - non-linear PCA
    - Independent Component Analaysis (ICA)

Principal Component Analysis

- Principal Component Analysis (PCA) is a statistical technique
  - Dimensionality reduction
  - Lossy data compression
  - Feature extraction
  - Data visualization

- It is also known as the *Karhunen-Loeve* transform

- PCA can be defined as the principal subspace such that the variance of the projected data is maximized

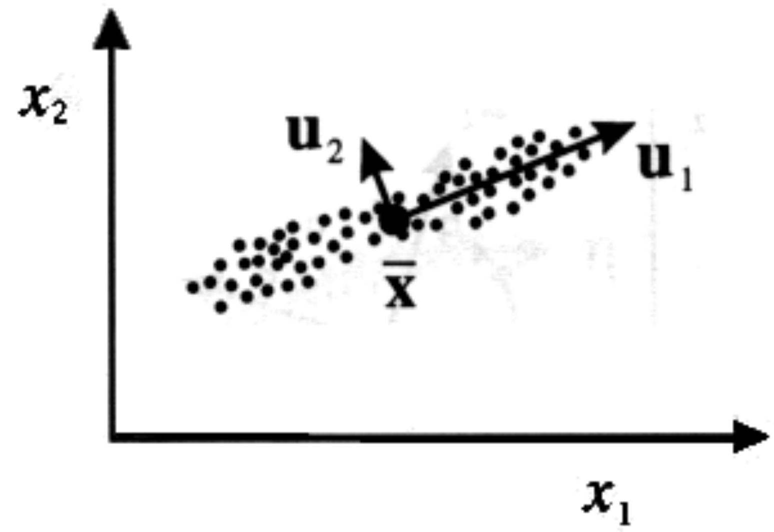
Second-Order methods

- The second-order methods are the most popular methods to find a linear transformation

- This methods find the representation using only the information contained in the covariance matrix of the data vector **x**

- PCA is widely used in signal processing, statistics, and neural computing

Principal Components



In a linear projection down to one dimension, the optimum choice of projection, in the sense of minimizing the sum-of-squares error, is obtained first subtracting off the mean of the data set, and then projecting onto the first eigenvector $\mathbf{u}_1$ of the covariance matrix.

Projection error minimization

- We introduce a complete orthonormal set of *D*-dimensional basis vectors (i=1,…,*D*)

$$\boxed{\mathbf{u}_i^T \mathbf{u}_j = \delta_{ij}}$$

- Because this basis is complete, each data point can be represented by a linear combination of the basis vectors

$$\boxed{\mathbf{x}_n = \sum_{i=1}^{D} \alpha_{ni} \mathbf{u}_i}$$

Projection error minimization

- We can write also that

$$\boxed{\mathbf{x}_n = \sum_{i=1}^{D} \left(\mathbf{x}_n^T \mathbf{u}_i\right) \mathbf{u}_i} \qquad \boxed{\alpha_{nj} = \mathbf{x}_n^T \mathbf{u}_j}$$

- Our goal is to approximate this data point using a representation involving a restricted number $M < D$ of variables corresponding to a projection onto a lower-dimensional subspace

$$\boxed{\widetilde{\mathbf{x}}_n = \sum_{i=1}^{M} z_{ni} \mathbf{u}_i + \sum_{i=M+1}^{D} b_i \mathbf{u}_i}$$

Projection error minimization

- As our distortion measure we shall use the squared distance between the original point and its approximation averaged over the data set so that our goal is to minimize

$$\boxed{J = \frac{1}{N}\sum_{n=1}^{N}\left\|\mathbf{x}_n - \tilde{\mathbf{x}}_n\right\|^2}$$

- The general solution is obtained by choosing the basis to be eigenvectors of the covariance matrix given by

$$\boxed{\mathbf{S}\mathbf{u}_i = \lambda_i \mathbf{u}_i}$$

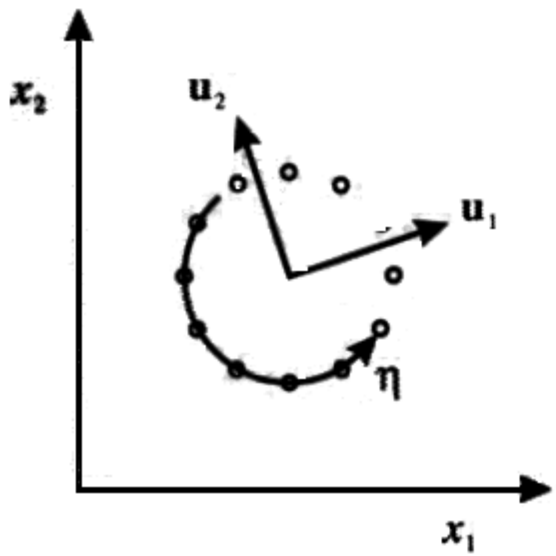
Projection error minimization

- The corresponding value of the distortion measure is then given by

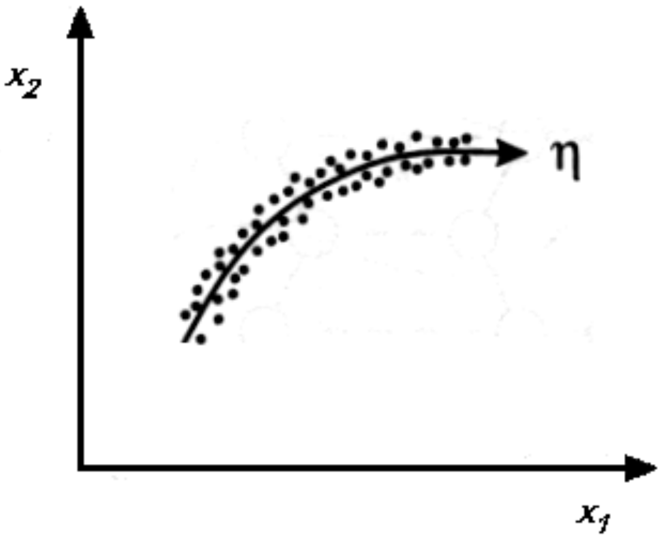
$$\boxed{J = \sum_{i=M+1}^{D} \lambda_i}$$

- We minimize this error selecting the eigenvectors defining the principal subspace are those corresponding to the M largest eigenvalues

Complex distributions



A linear dimensionality reduce technique, such as PCA, is unable to detect the lower dimensionality. In this case PCA gives two eigenvectors with equal eigenvalues. The data can described by a single eigenvalue



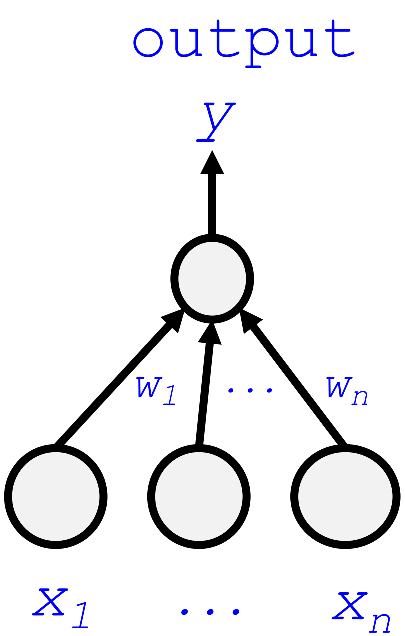
Addition of a small level of noise to data having an intrinsic. Dimensionality to 1 can increase its intrinsic dimensionality to 2. The data can be represented to a good approximation by a single variable η and can be regarded as having an intrinsic dimensionality of 1.

Unsupervised Neural Networks

- Typically Hebbian type learning rules are used
- There are two type of NN able to extract the Principal Components:
  - Symmetric (Oja, 1989)
  - Hierarchical (Sanger, 1989)

Information and Hebbian Learning

- Information extraction



$$y = \sum_i w_i x_i$$

Hebbian learning - self-amplification

$$\Delta w_i = \eta y x_i$$

the net learns to respond the patterns that present the most frequent samples

Principal Component

- Weights can grow to infinity
  - Solution – normalization (no - local)

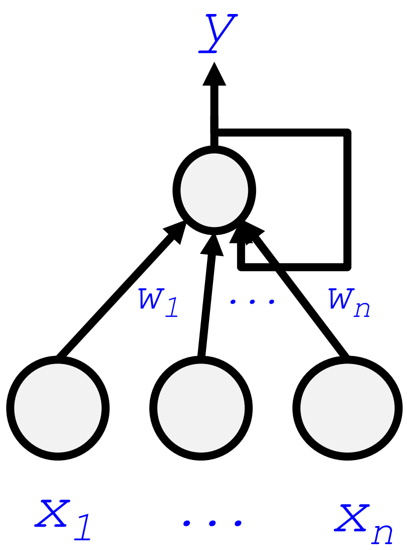
$$w_i = \frac{w_i}{||\mathbf{w}||}$$

  - Competition mechanism for a stable solution
    - weights in the direction of maximum variance of the distribution
    - Maximization of the variance on the oputput
    - weights in the direction of the eigenvector corresponding to the maximum eigenvalue of the correlation matrix

$$C = \langle \mathbf{x}\mathbf{x}^{\mathrm{T}} \rangle_\mu$$

Oja's rule

- Idea



Information feedback

Oja's rule

- Normalization is not local

- Oia's rule

$$\Delta w_j = \eta\left(x_j - w_j y\right)$$

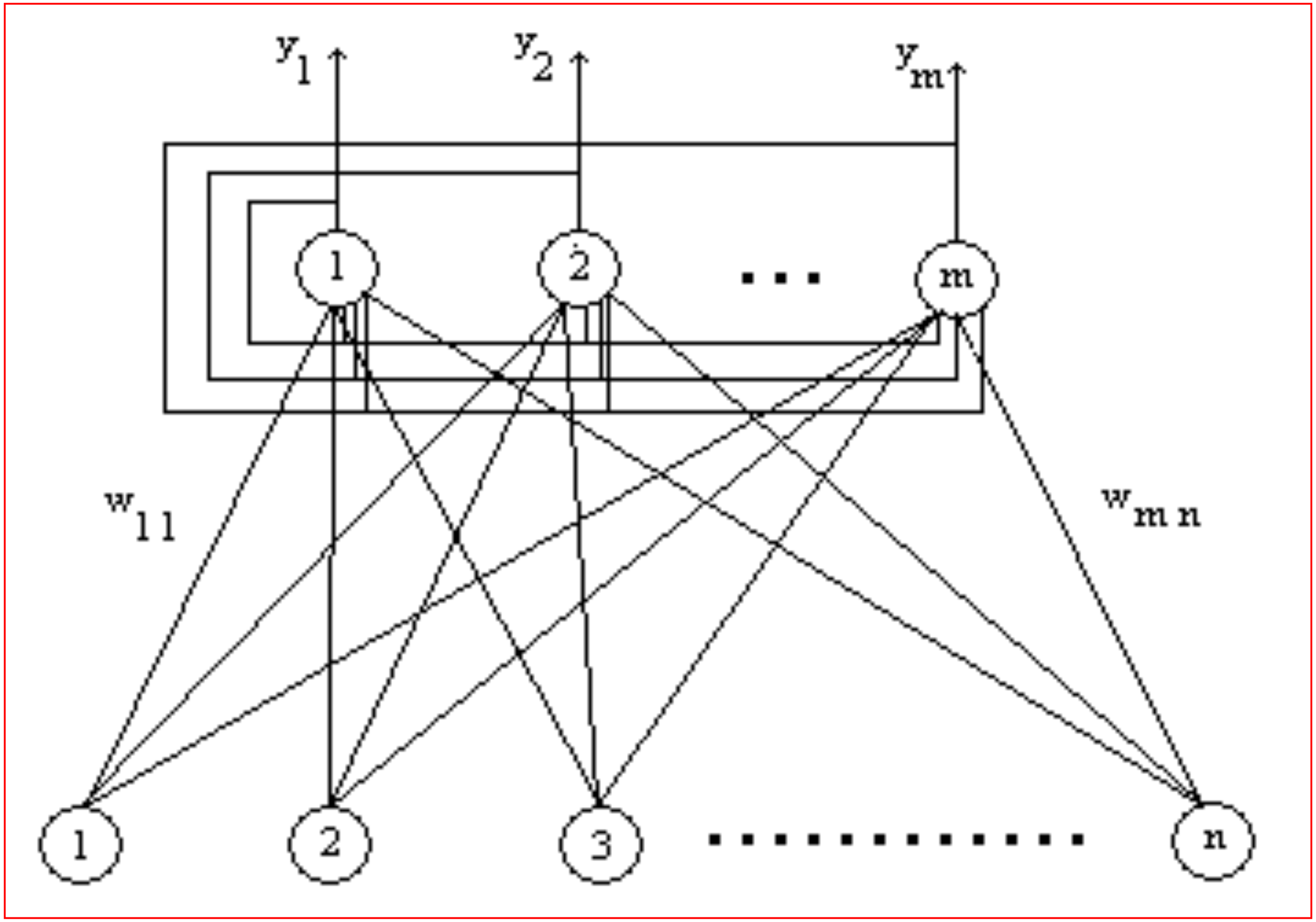
Forgetting factor

- More outputs

$$\Delta w_{ij} = \eta y_i\left(x_j - \sum_{k=1}^{n} w_{kj} y_k\right)$$

Syemmetric NN

Single layer Neural Network



$$\boxed{E\left[\mathbf{y}^2\right] = E\left[\left(\mathbf{w}^T\mathbf{x}\right)^2\right]}$$

Objective function

Symmetric PCA NN

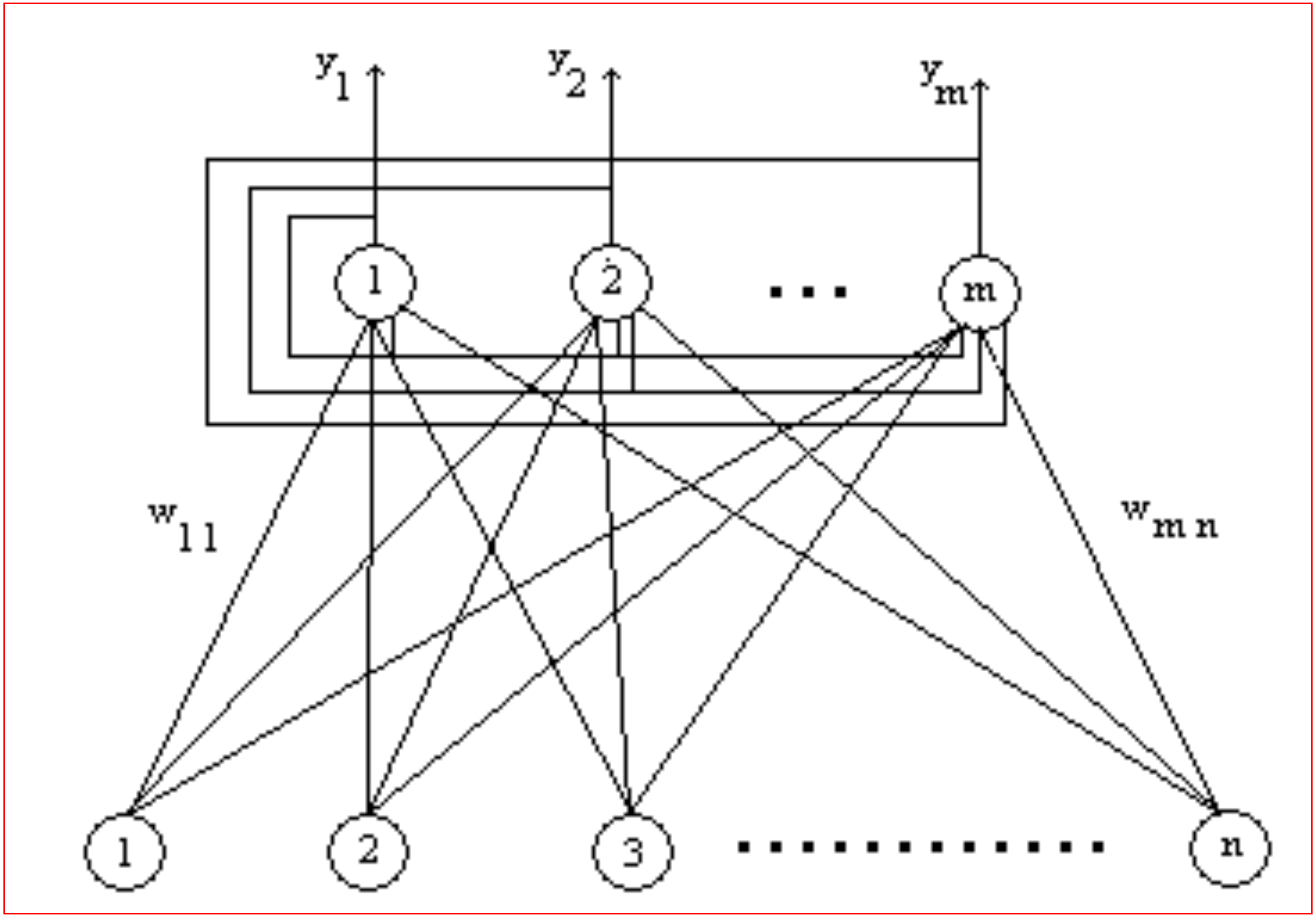
Sanger's rule

- Sanger's learning rule

$$\Delta w_{ij} = \eta y_i \left(x_j - \sum_{k=1}^{i} w_{kj} y_k \right)$$

Hierarchical NN

Single layer Neural Network

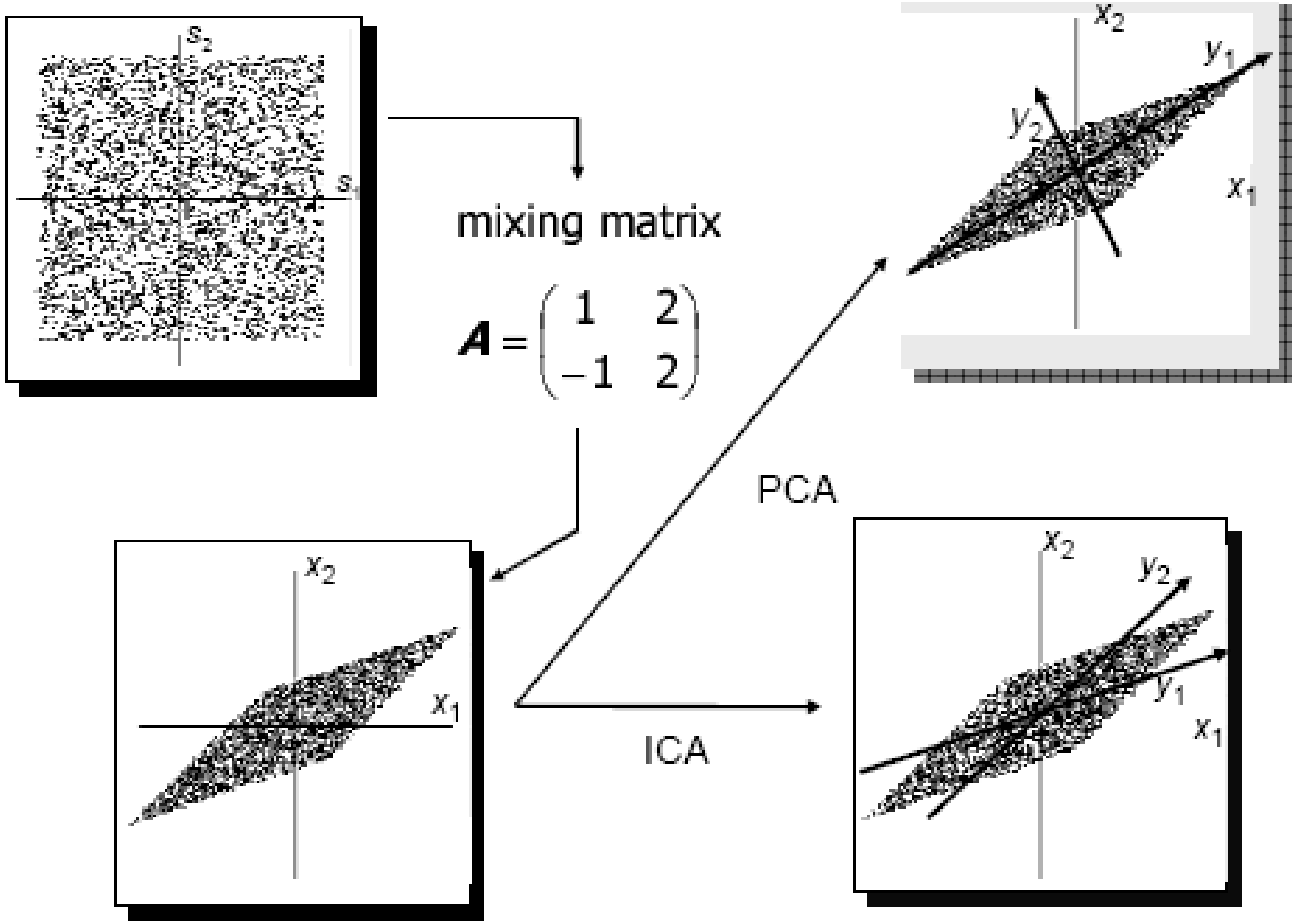


Hierarchical PCA NN

Oja's rule vs. Sanger's rule

- Oja's rule
  - Symmetric Space
  - Principal Components without a specific order
- Sanger's rule
  - Hierarchical space
  - Principal Components without a specific order
    - weights of the first output neuron corresponding to the first component, weights of the second neuron to the second residual component, and so on

Mixing matrix



Non-linear objective function

Maximization

$$\mathbf{x} \xrightarrow{\mathbf{w}\ \text{(weights)}} \boxed{E\left\{f\left(\mathbf{w}^T\mathbf{x}\right)\right\}}$$

L-dimensional vector

where E is the expectation with respect to the (unknown) density of **x** and $f(.)$ is a continue function (e.g. ln cosh(.))

Taylor series

$$\ln\cosh(y) = \frac{1}{2}y^2 - \frac{1}{12}y^4 + \frac{1}{45}y^6 + O(y^8)$$

$$E\{\ln\cosh(y)\} = \tfrac{1}{2}E\left\{(w^Tx)^2\right\} - \tfrac{1}{12}E\left\{(w^Tx)^4\right\} + \tfrac{1}{45}E\left\{(w^Tx)^6\right\} + E\left\{O((w^Tx)^2)\right\}$$

$$C = I \quad \text{and} \quad \tfrac{1}{2}E\left\{(w^Tx)^2\right\} = \tfrac{1}{2}$$

$$-\frac{1}{12}E\left\{(w^Tx)^4\right\}$$

That is dominating, and the kurtosis is optimized

Robust and non-linear PCA

Standard PCA

$$E\left[\mathbf{y}^2\right] = E\left[\left(\mathbf{w}^T\mathbf{x}\right)^2\right]$$

Robust PCA

$$J_1(\mathbf{w}) = E\left[f\left(\mathbf{x}^T\mathbf{w}\right)\right] + \sum_{j=1}^{I(i)} \lambda_{ij}\left[\mathbf{w}_i^T\mathbf{w}_j - \delta_{ij}\right]$$

$$J_2(e_i) = 1^T E\left[f(\mathbf{x} - \hat{\mathbf{x}}_i)\right]$$

Nonlinear PCA

$$E\left[\left|b_i(k)\right|^2\right]$$

$$b_i(k) = \mathbf{x}_k - \sum_{j=1}^{I(i)} g\left(\mathbf{y}_j(k)\right)\mathbf{w}_j(k)$$

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu_k g(\mathbf{y}_i(k)) e_i(k)$$

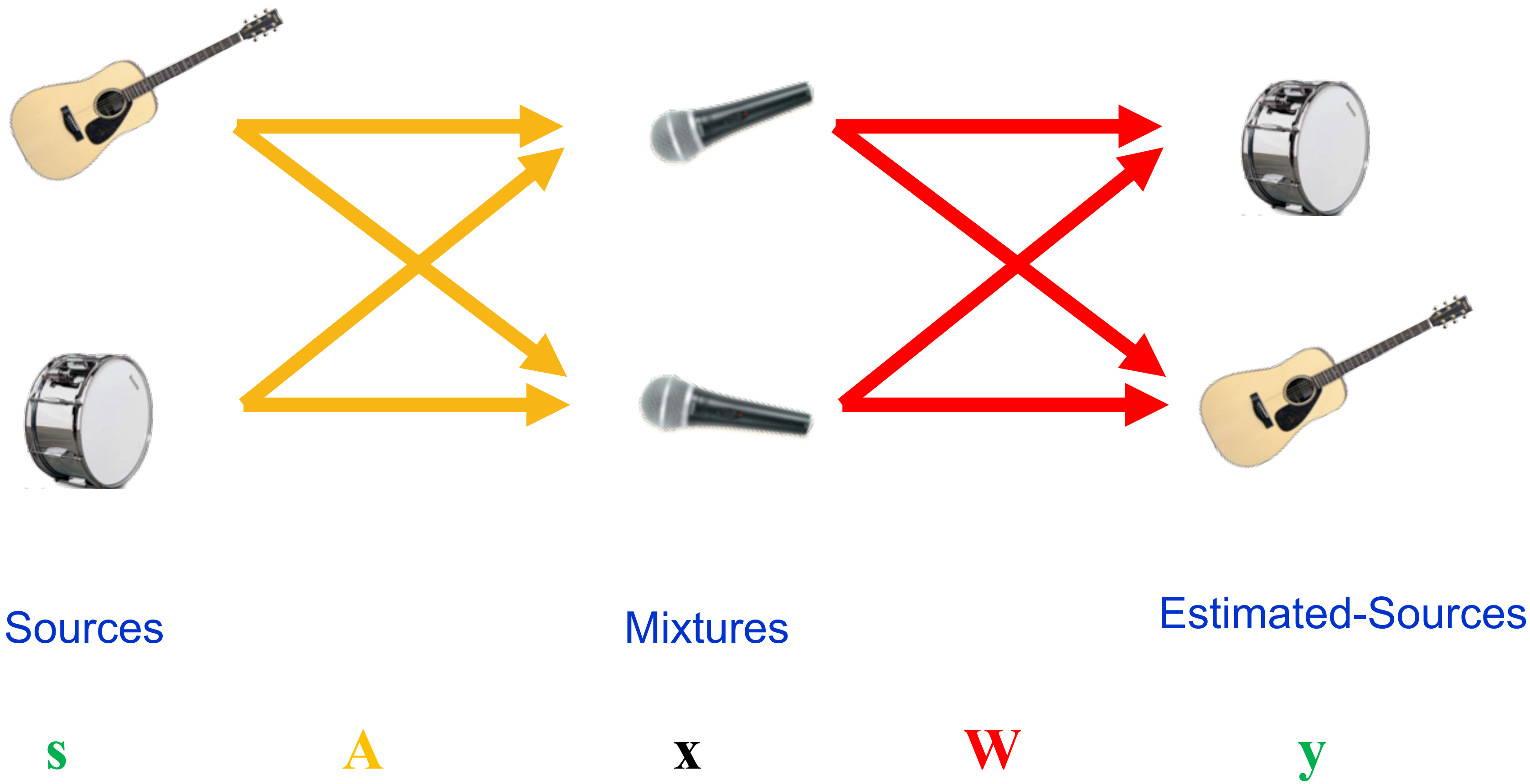
$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu_k \begin{pmatrix} \mathbf{w}_i(k)^T g(e_i(k))\mathbf{x}_k + \\ + \mathbf{x}_k^T \mathbf{w}_i(k) g(e_k(i)) \end{pmatrix}$$

$$\mathbf{w}_i(k+1) = \mathbf{w}_i(k) + \mu g\left(\mathbf{y}_i(k)\right) b_i(k)$$

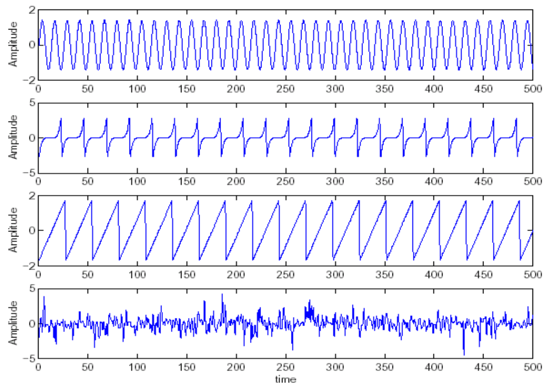
Descendent gradient algorithm

$$e(k) = \mathbf{x}_k - \sum_{j=1}^{I(i)} \mathbf{y}_j(k)\mathbf{w}_j(k)$$

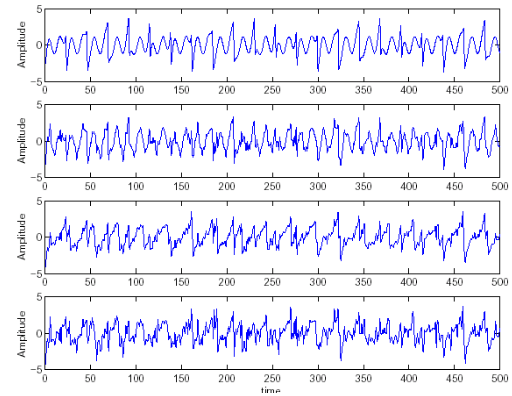
Cocktail party



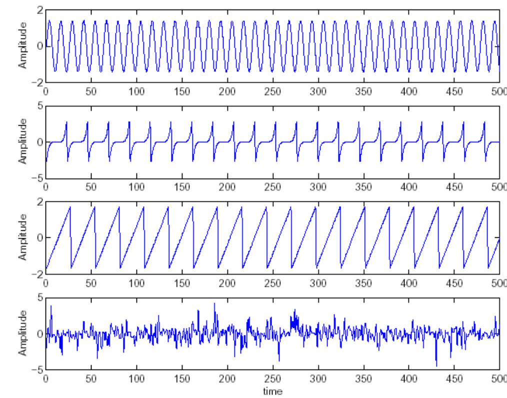
Source estimation



Source signals



Mixed signals



Estimated signals

$$
\begin{aligned}
x_1(t) &= a_{11}s_1(t) + a_{12}s_2(t) + a_{13}s_3(t)\\
x_2(t) &= a_{21}s_1(t) + a_{22}s_2(t) + a_{23}s_3(t)\\
x_3(t) &= a_{31}s_1(t) + a_{32}s_2(t) + a_{33}s_3(t)
\end{aligned}
$$

$$
\begin{aligned}
y_1(t) &= w_{11}x_1(t) + w_{12}x_2(t) + w_{13}x_3(t)\\
y_2(t) &= w_{21}x_1(t) + w_{22}x_2(t) + w_{23}x_3(t)\\
y_3(t) &= w_{31}x_1(t) + w_{32}x_2(t) + w_{33}x_3(t)
\end{aligned}
$$

x_1(t), $x_2(t)$,$x_3(t)$ are the observed signals, $s_1(t)$, $s_2(t)$, $s_3(t)$ the source signals

y_1(t), $y_2(t)$,$y_3(t)$ are the separated signals