Harri Lappalainen and Antti Honkela:

Bayesian Non-Linear ICA by Multi-Layer Perceptrons

Presented by Antti Honkela

November 16, 2000

Contents

- Background
- The model
- Experiments
- Summary

Background: Nonlinear ICA

- For given n-dimensional random vector $\boldsymbol{x}$, there exists a continuous nonlinear transformation $\boldsymbol{g}$ such that the components of $\boldsymbol{y} = \boldsymbol{g}(\boldsymbol{x})$ are independent
- Without additional constraints, there are infinitely many non-trivially related solutions[a]
- Other attempts to make the problem well-posed: post-nonlinear ICA, SOM, GTM, ...

[a] A. Hyvärinen and P. Pajunen, "Nonlinear ICA: Existence and Uniqueness Results," in *Neural Networks*, **12**(3), 1999.

Background: Generative models

Latent variables s(t) (explanations, factors, causes, sources, ...)

$\Downarrow$

Mapping f(,W)

$d(t) = f(s(t), W)$

$\Downarrow$

Observations d(t)

Background: Generative models (continued)

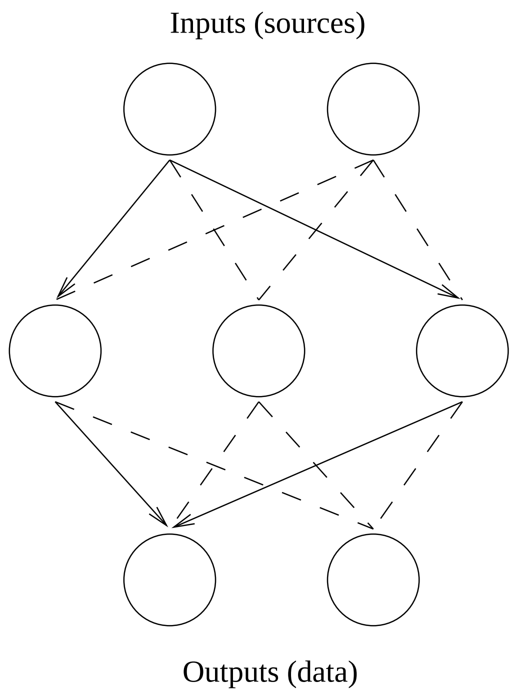
- (Linear) factor analysis (FA): $\boldsymbol{x} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{n}$
  - Sources $\boldsymbol{s}$ are Gaussian and white, $\boldsymbol{n}$ Gaussian noise
  - "Noisy PCA"
  - Rotation indeterminacy because of Gaussian sources
- (Linear) independent factor analysis (IFA)
  - Sources assumed to be independent, modeled with mixtures-of-Gaussians

The model

- Generative model for the data:

$$\boldsymbol{x}(t) = \boldsymbol{f}(\boldsymbol{s}(t)) + \boldsymbol{n}(t)$$

- Function $\boldsymbol{f}$ is modeled with a Multi-Layer Perceptron (MLP) network
  - $\boldsymbol{f}(\boldsymbol{s}) = \boldsymbol{B}\boldsymbol{\phi}(\boldsymbol{A}\boldsymbol{s} + \boldsymbol{a}) + \boldsymbol{b}$
  - Universal approximator
  - Efficient parameterization for high dimensional spaces



The model, continued

- Fully Bayesian treatment
  - Distributions for everything, not point estimates
- Noise and source variances estimated for all channels separately
- Common hyperparameter controls the priors of similar variables
- Activation function ϕ of the MLP network is approximated with Taylor series around the posterior mean of the input
- Propagation of variance through multiple paths makes things more difficult to compute

The cost function

- The cost function used is Kullback-Leibler information between the true posterior $p(\boldsymbol{S}, \boldsymbol{\theta}|\boldsymbol{X})$ and its parametric approximation $q(\boldsymbol{S}, \boldsymbol{\theta}|\boldsymbol{X}) = q(\boldsymbol{S}|\boldsymbol{X})q(\boldsymbol{\theta}|\boldsymbol{X})$
- The approximate posterior of model parameters $\boldsymbol{\theta}$ is a product of independent Gaussians
- The approximate posterior of sources is a product of independent distributions $q(\boldsymbol{S}|\boldsymbol{X}) = \prod_t \prod_i q_{t,i}(s_i(t)|\boldsymbol{X})$
- Source distributions $q_{t,i}(s_i(t)|\boldsymbol{X})$ are Gaussians or mixtures-of-Gaussians (MoG)
- Variances are parameterized with log-std, i.e. $\mathrm{Var}[\boldsymbol{x}] = e^{2\boldsymbol{v}_{\boldsymbol{x}}}$, $\boldsymbol{v}_{\boldsymbol{x}} \sim N(\alpha, \beta)$

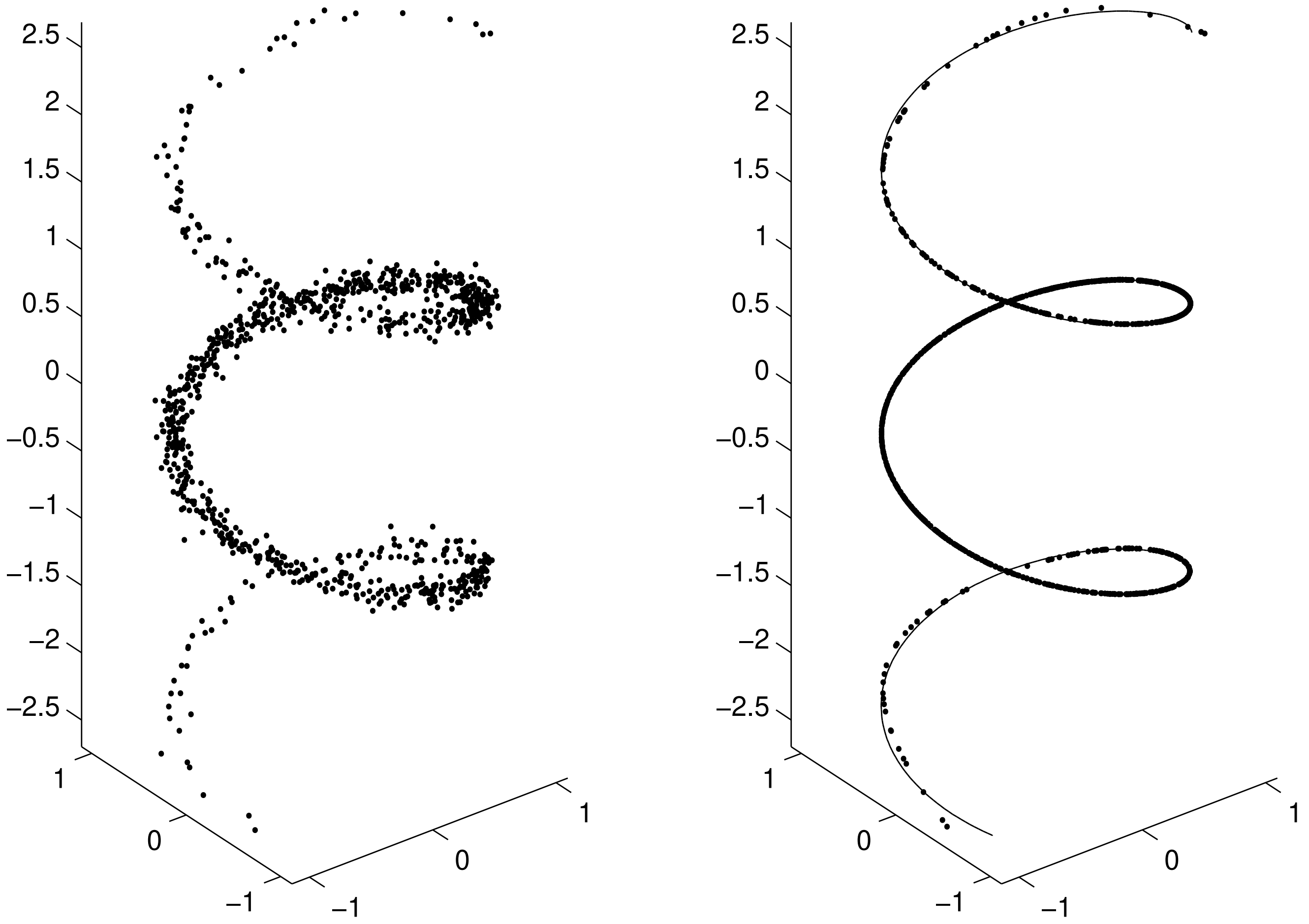
Learning procedure

- Only the outputs of the MLP are known
- Sources and network parameters are adapted with a backpropagation-like algorithm
- One iteration consists of going through all the samples and updating all the sources, weights and parameters
- Optimal variances of all components can be solved analytically
- Parameter means are updated with a diagonal Newton's iteration
- Every variable has an own dampening coefficient
- Source adaptation has additional coefficient to compensate cumulative effects of changing different sources

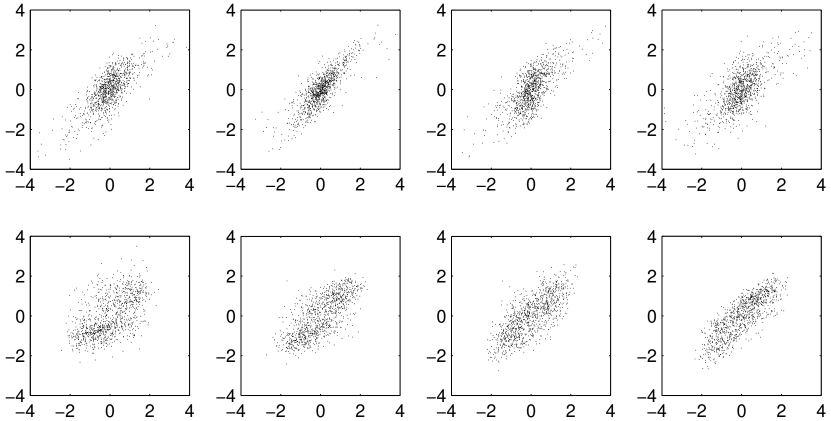
Learning procedure, continued

Iteration	Function
0-50	Sources fixed to PCA values, only network is adapted
50-100	Gaussian sources and network are adapted
100-2000	Gaussian sources, network and other parameters (noise, hyperparameters, ...) are adapted
2000	Sources are rotated with e.g. FastICA and modeled with mixtures-of-Gaussians
2000-(7500)	Adapt everything, MoG sources

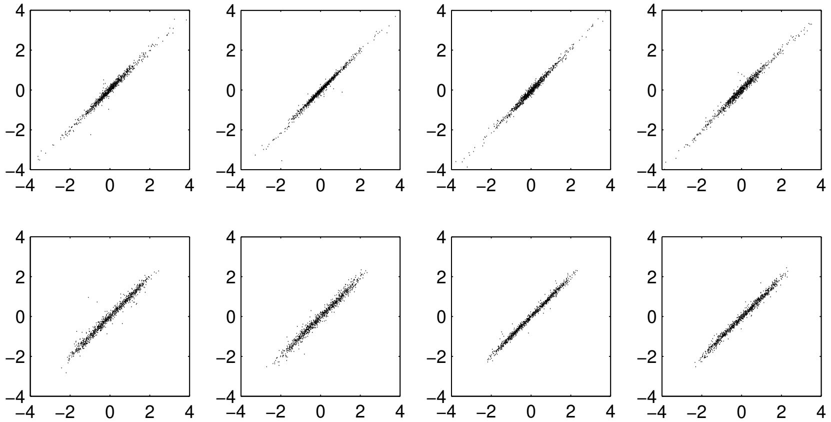
Example: Helix



Example: Generated MLP-data

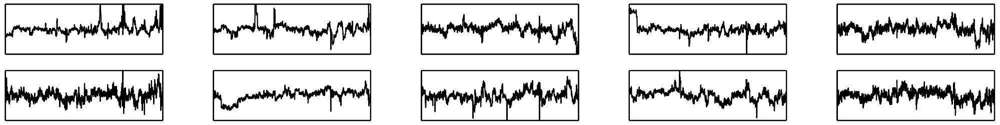


True sources vs. sources found with linear ICA

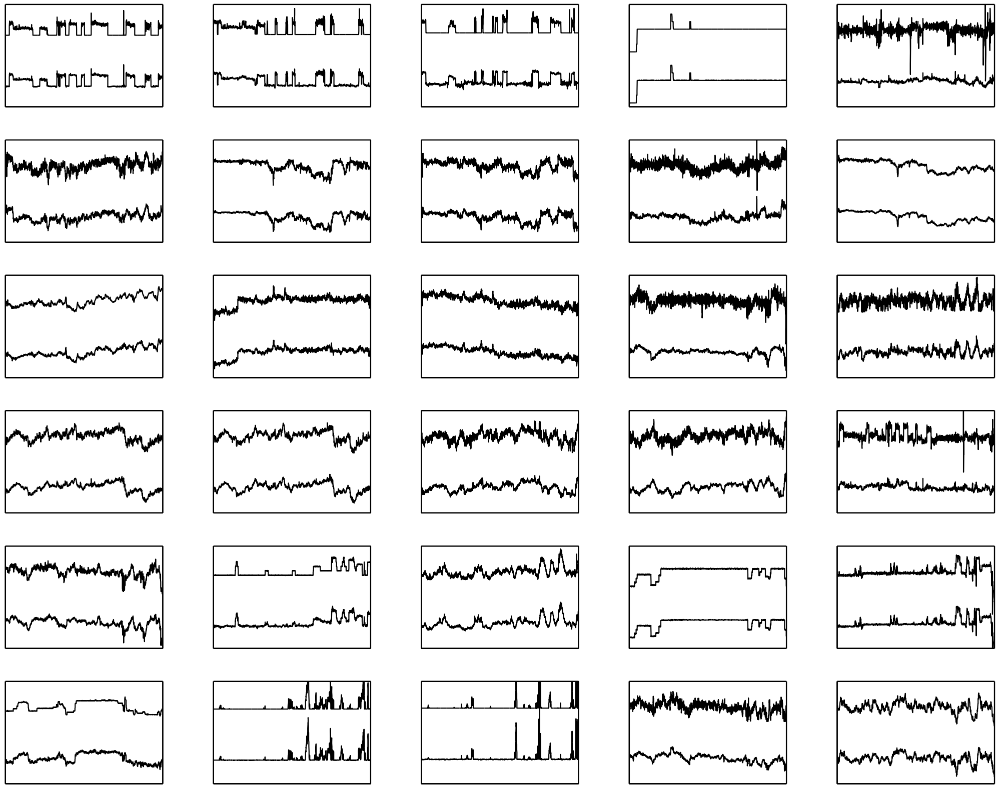


True sources vs. sources found with nonlinear IFA

Example: Process data



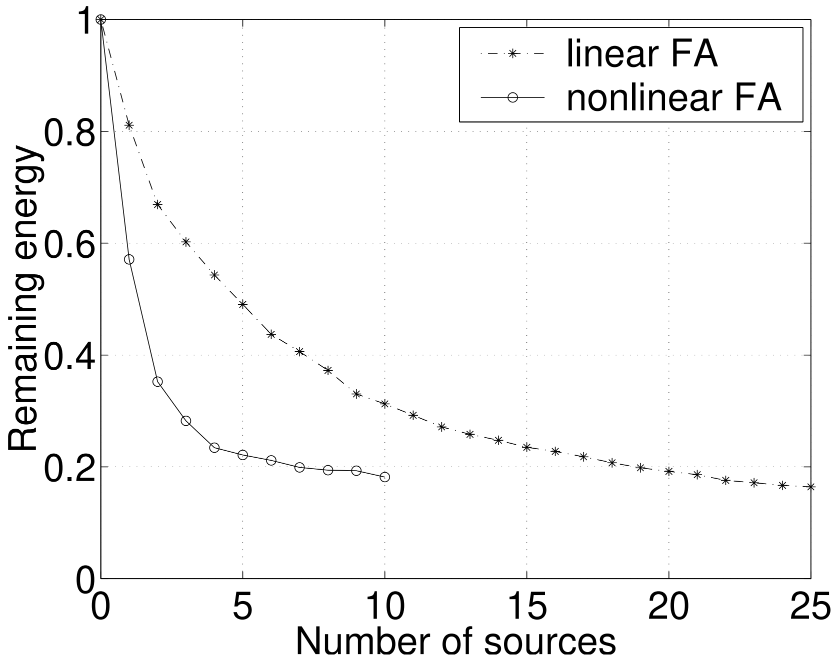
Sources found with non-linear IFA



The original data and reconstructions of the model

Summary of the examples

- Optimal number of hidden neurons and sources can be estimated
- Network can prune extra sources and hidden neurons
- Nonlinear FA can account for much more data with same number of sources than linear FA



Conclusions

- Generative model for nonlinear ICA
- Uses Bayesian ensemble learning
- Computationally efficient for high dimensional data
- Works well for artificial and real world data
- Matlab implementation available at
  `http://www.cis.hut.fi/projects/ica/bayes/`