Faster Training in Nonlinear ICA using MISEP

(with a simple introduction to nonlinear ICA and to MISEP)

Luis B. Almeida – IST and INESC-ID, Lisbon, Portugal

luis.almeida@inesc-id.pt

Summary

- Mutual information as a dependence measure
- Mutual information as output entropy
- Minimizing the mutual information
- Examples
- Learning speed
- Conclusions

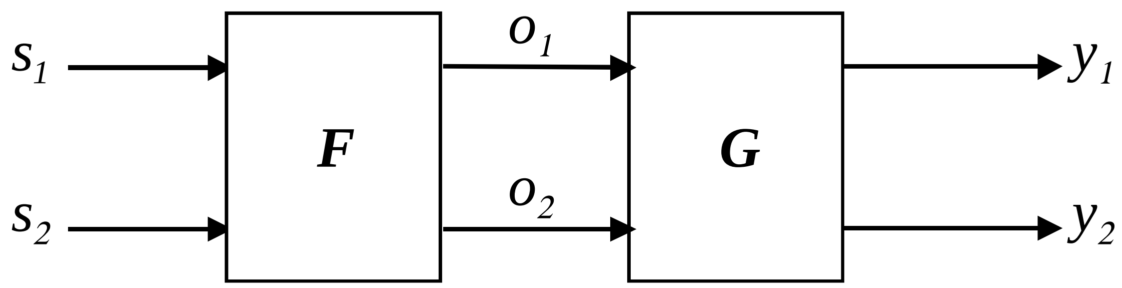
What is MISEP? How does it perform nonlinear ICA/BSS?

MISEP is an extension of INFOMAX to nonlinear ICA/BSS

But How?

- We wish to use mutual information (MI) as the dependence measure to be minimized.
- INFOMAX minimizes the mutual information, but
    - it is limited to linear ICA,
    - it needs a priori knowledge of the sources' distributions (at least approximately).
- We shall extend INFOMAX in two directions:
    - Extending it to nonlinear ICA.
    - Using adaptive estimation of the components' distributions.
- This results in an extension of INFOMAX, that we call MISEP.

Setting



s – source vector

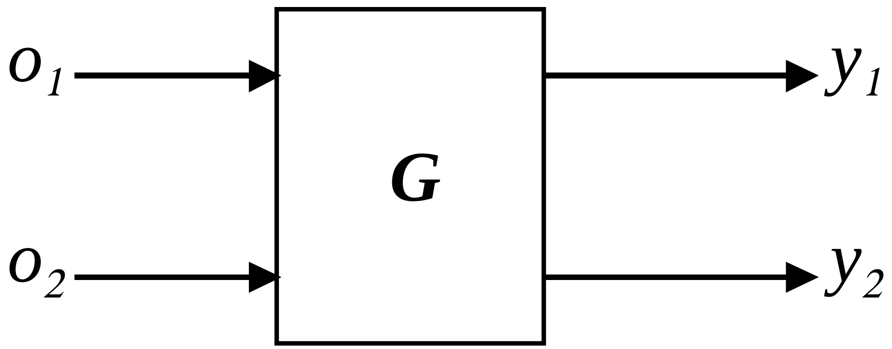
o – observation vector

y – vector of estimated components

F – mixture (linear or nonlinear)

G – ICA system (linear or nonlinear)

Mutual information as a dependence measure



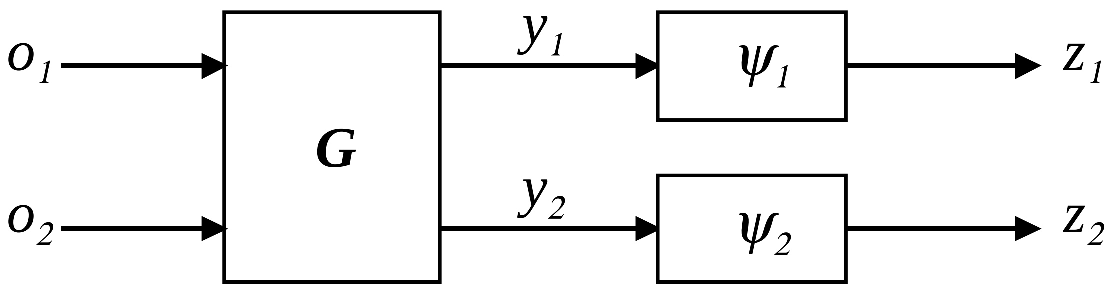
o_i - observations

y_i - estimated components

Mutual information:

- $I(\mathbf{Y}) = \sum_i H(Y_i) - H(\mathbf{Y})$ – sum of marginal entropies minus joint entropy
- $I(\mathbf{Y})$ is also the Kullback-Leibler divergence between $\prod_i p_{Y_i}(Y_i)$ and the true distribution $p_{\mathbf{Y}}(\mathbf{Y})$.
- $I(\mathbf{y})$ is non-negative, and is zero only if the Y_i are independent from one another. It is a good measure of the dependence of the Y_i.

Expressing the mutual information as output entropy

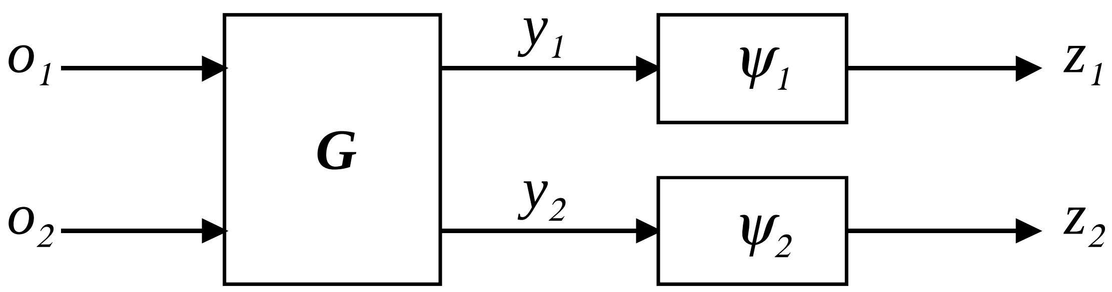


- The mutual information is hard to minimize directly. But…
- If the transformations ψ_i are invertible, the mutual information is not affected: $I(\boldsymbol{Z}) = I(\boldsymbol{Y})$.
- If ψ_i is the cumulative probability function of Y_i, then Z_i is uniformly distributed in [0,1], and $H(Z_i) = 0$.

$$I(\boldsymbol{Y}) = I(\boldsymbol{Z}) = \sum_i H(Z_i) - H(\boldsymbol{Z}) = -H(\boldsymbol{Z})$$

- Maximizing the output entropy is equivalent to minimizing $I(\boldsymbol{Y})$.

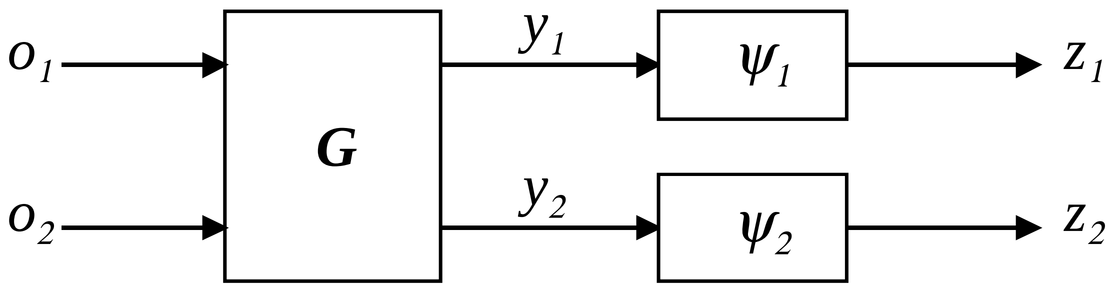
How do we find the cumulative functions?



- INFOMAX (Bell & Sejnowski, 95) – Cumulative functions known *a priori* (at least approximately).

- MISEP – Estimate the CPFs adaptively, by maximum entropy.

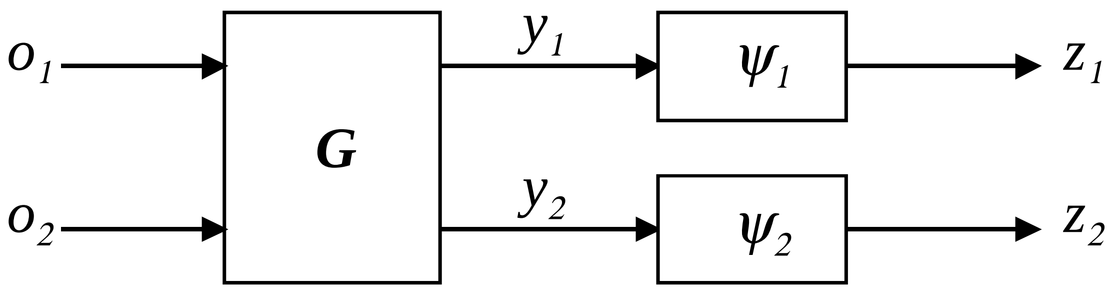
Estimating the cumulative functions (continued)



$$I(\boldsymbol{Y}) = \sum_i H(Z_i) - H(\boldsymbol{Z}) \qquad\qquad H(\boldsymbol{Z}) = \sum_i H(Z_i) - I(\boldsymbol{Y})$$

- If the distributions of the Y_i components were kept fixed, maximizing the $H(\boldsymbol{Z})$ would be equivalent to maximizing each of the marginal entropies…
- … but at the end of training (at convergence) the Y_i *are* fixed!
- If each of the outputs Z_i is bounded in [0,1], it will become uniform in that interval, and each ψ_i will be the CPF of Y_i as desired.
  - ψ_i will have to be constrained to be an increasing function.

Estimating the cumulative functions (continued)



By maximizing the output entropy we will:

- adapt the output MLPs to yield the CPFs of the Y_i;
- minimize the mutual information $I(\boldsymbol{Y})$.

The output MLPs are restricted to yield monotonically increasing functions, bounded to [0,1].

Maximizing the output entropy

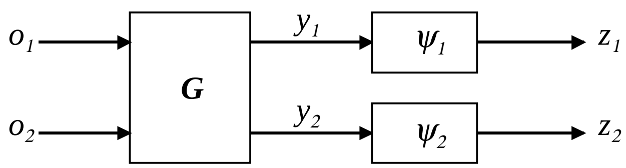
$$H(\boldsymbol{Z}) = H(\boldsymbol{O}) + \left\langle \log\left|\det \boldsymbol{J}\right| \right\rangle$$

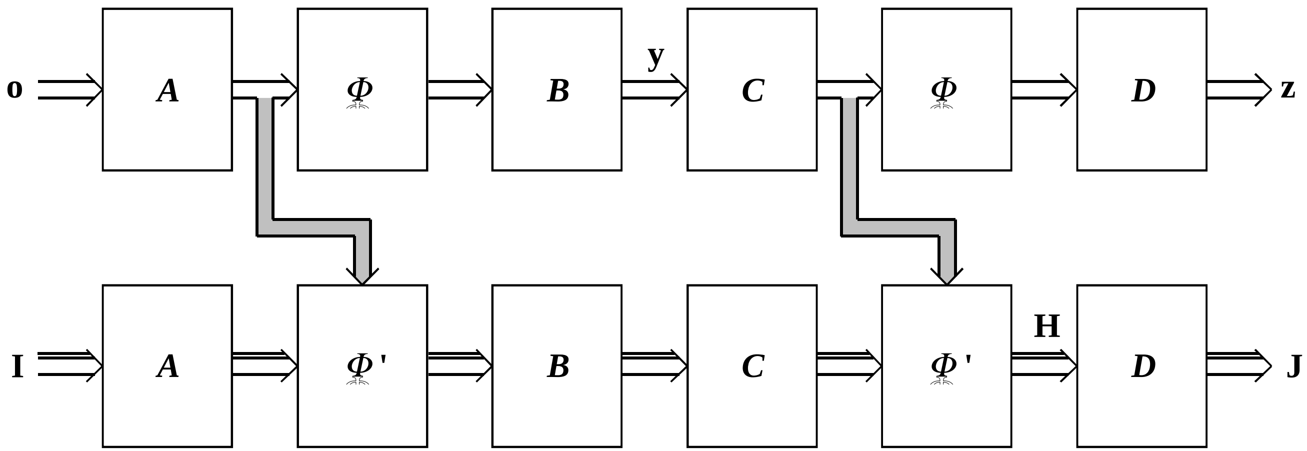
with $\boldsymbol{J} = \dfrac{\partial \boldsymbol{Z}}{\partial \boldsymbol{O}}$ (Jacobian of the transformation).

$H(\boldsymbol{O})$ is fixed. We need to maximize $\left\langle \log\left|\det \boldsymbol{J}\right| \right\rangle$.

But how do we do that?

Network that computes *J*:

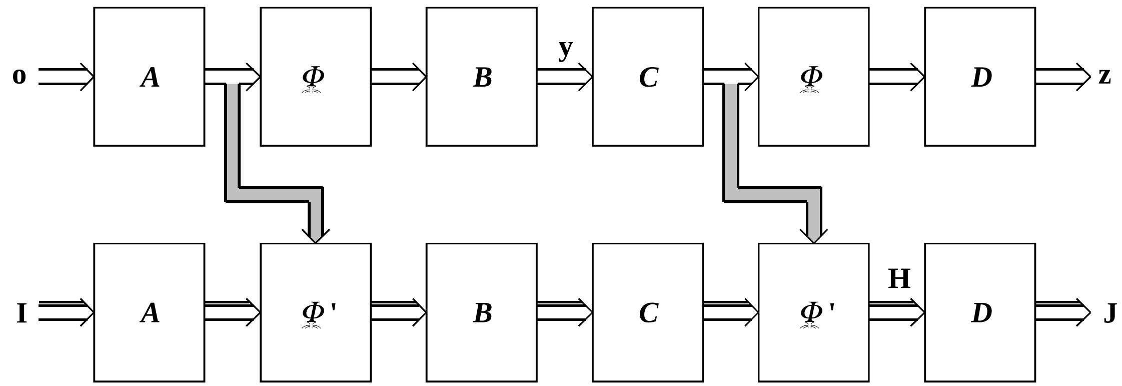




The upper part of the figure is the separating network. The lower part computes the Jacobian.

The lower part is essentially a linearized version of the upper part. Its input is the identity matrix.

Maximization of the entropy (continued)



- We have to backpropagate through the lower network (and through the shaded arrows, into the upper network).

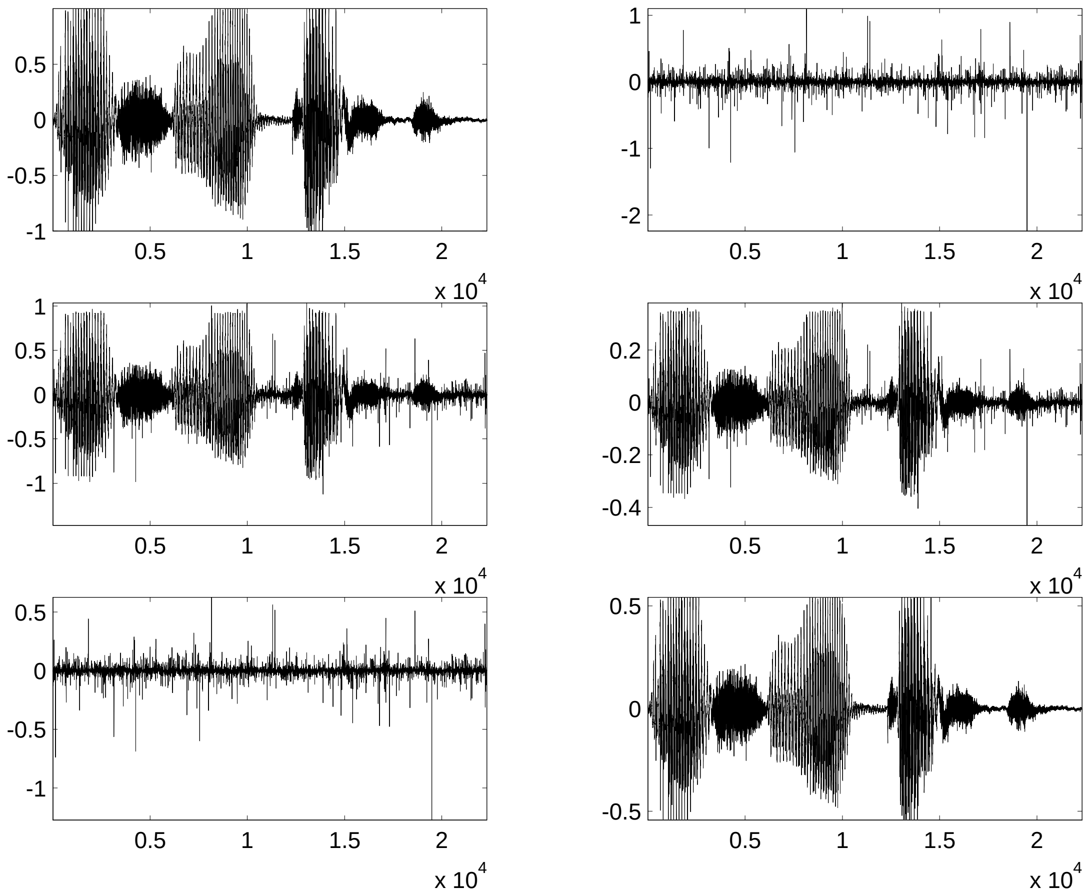
- Input to the backpropagation network:

$$\frac{\partial \log|\det \boldsymbol{J}|}{\partial \boldsymbol{J}} = (\boldsymbol{J}^{-1})^T$$

Examples

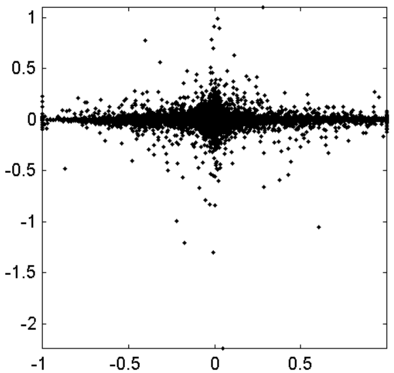
1. Linear ICA

Two supergaussians

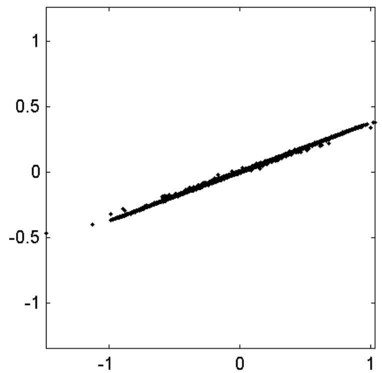


Scatter plots

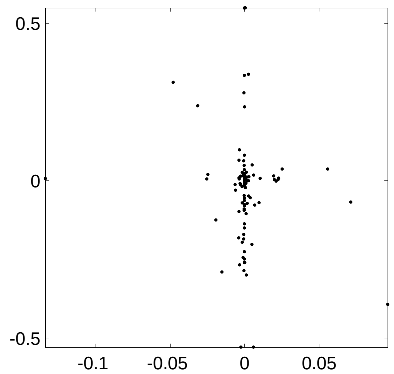
Original

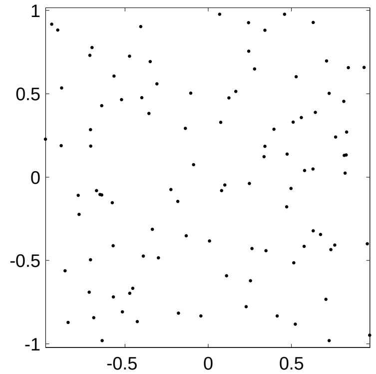


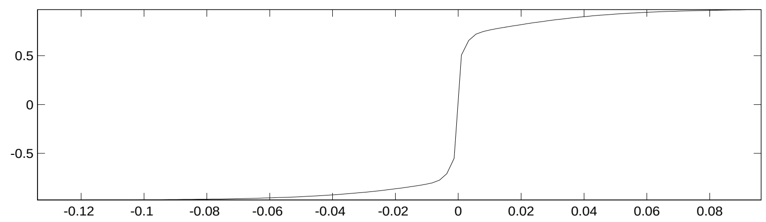
mixture

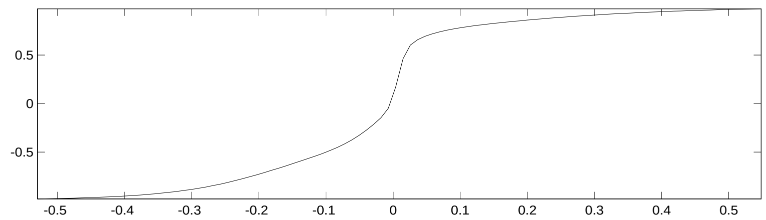


Just the 100 training points

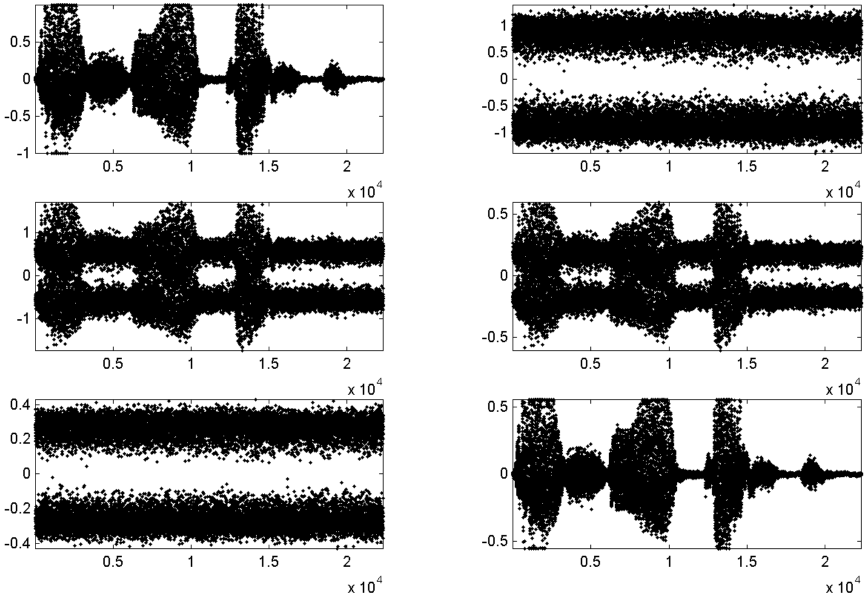


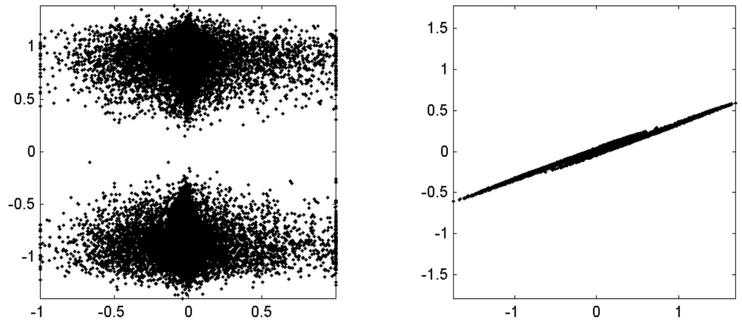


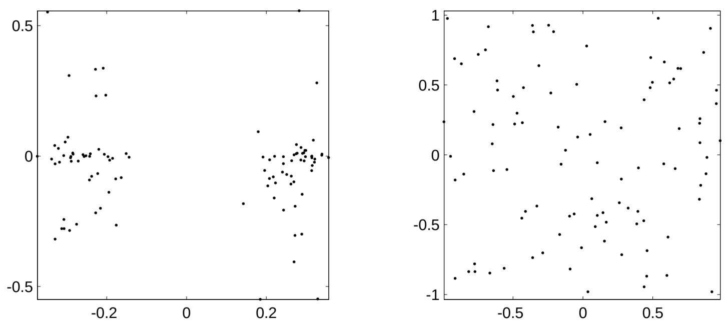


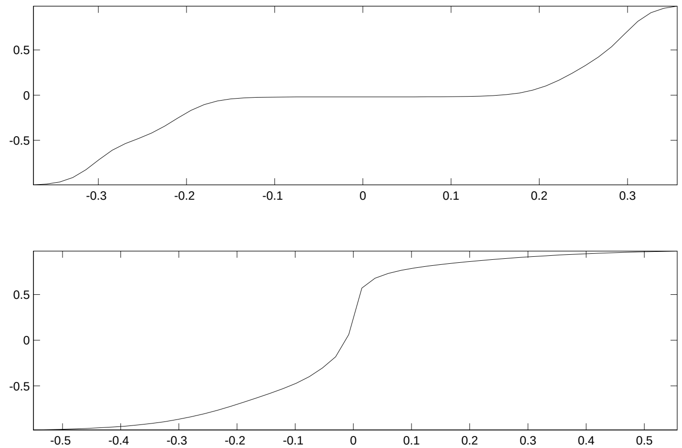


A supergaussian and a subgaussian

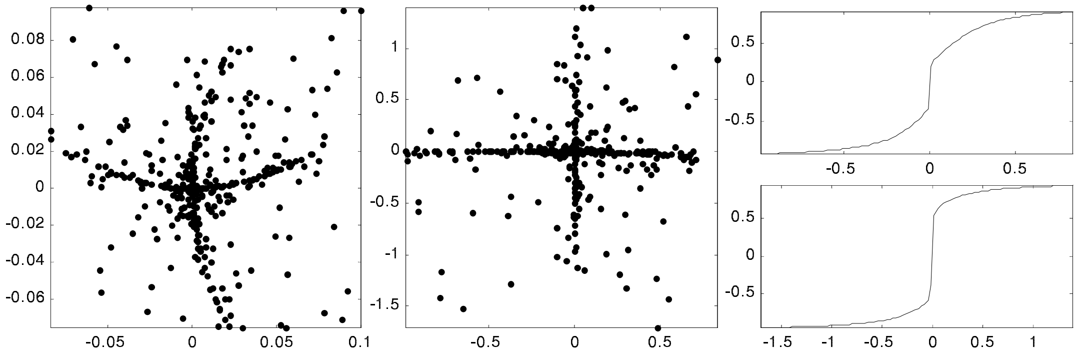




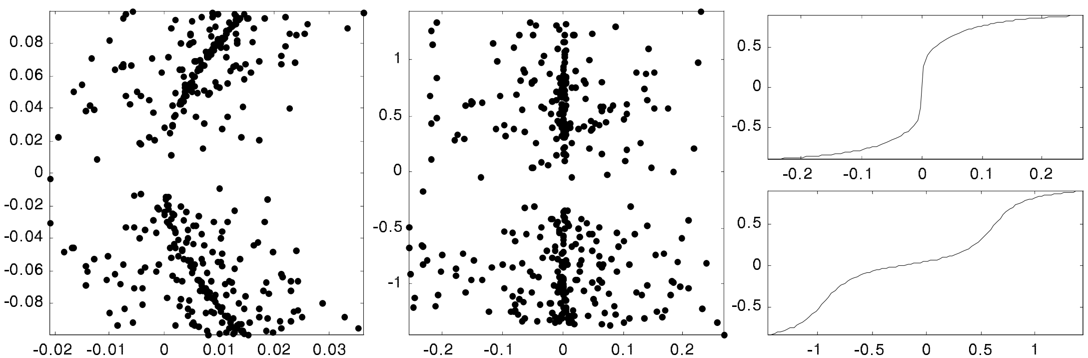




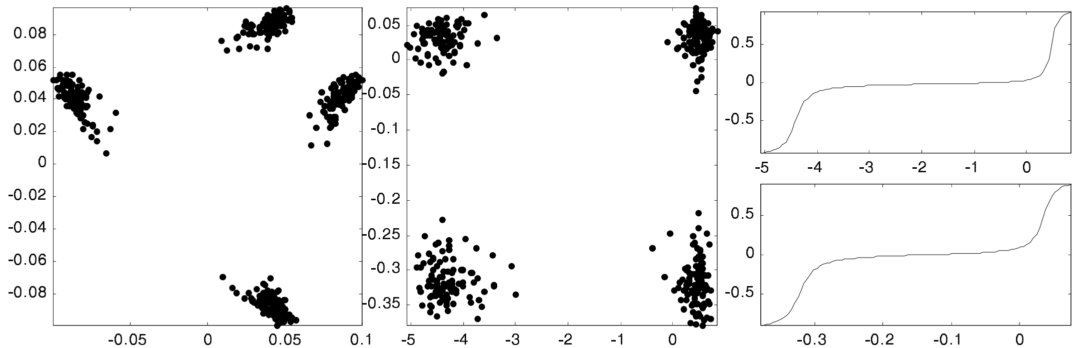
Nonlinear ICA, two supergaussians



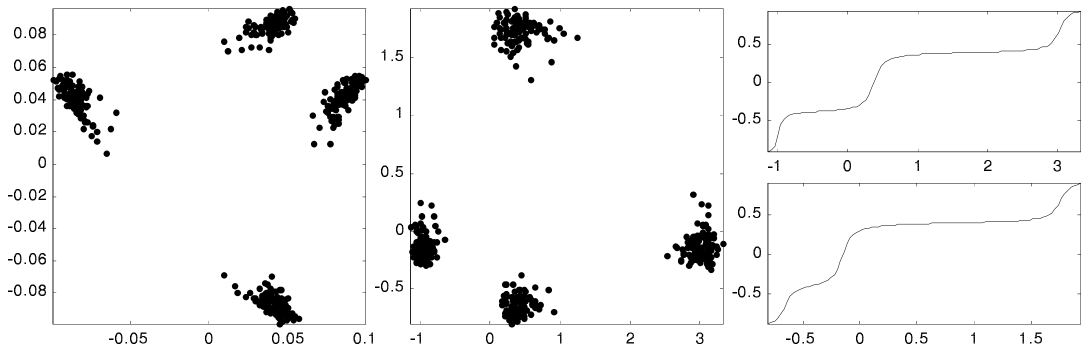
A supergaussian and a subgaussian



Two subgaussians



A local minimum of the mutual information



Nonlinear mixture of two speech signals

(listen to the demo)

Mixture:

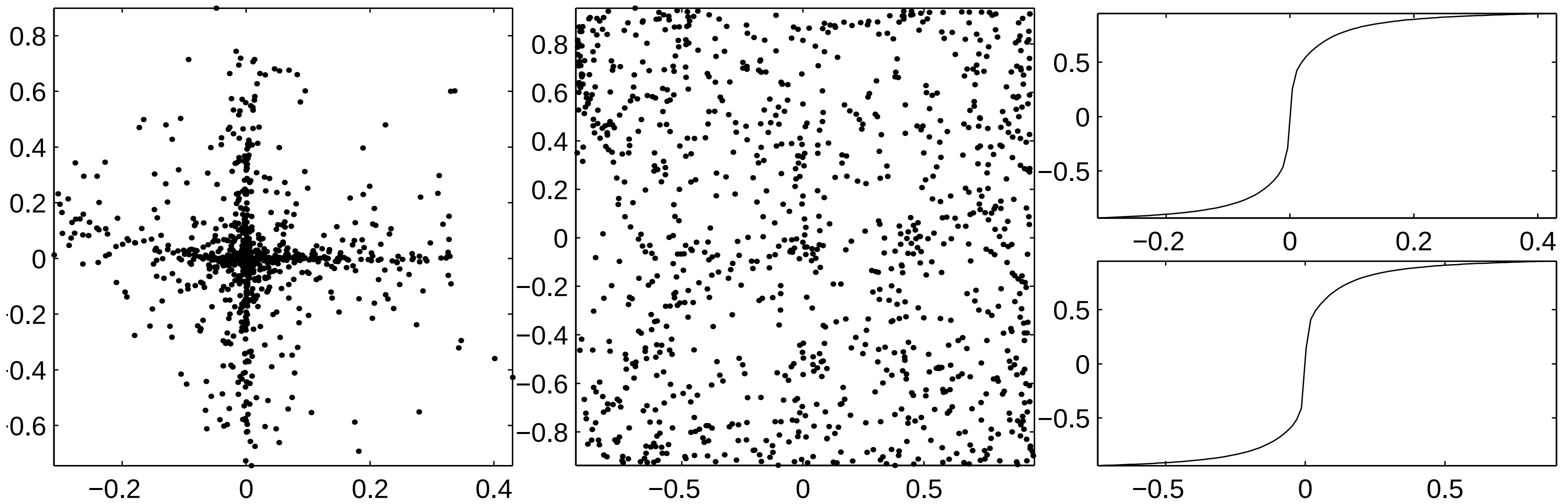
$$o_1 = s_1 + a(s_2)^2$$
$$o_2 = s_2 + a(s_1)^2$$

Signal to interference ratios:

Mixture:	9.1 dB
Separated:	16.9 dB
Improvement:	7.8 dB

Problem

Learning is often slower than one would expect



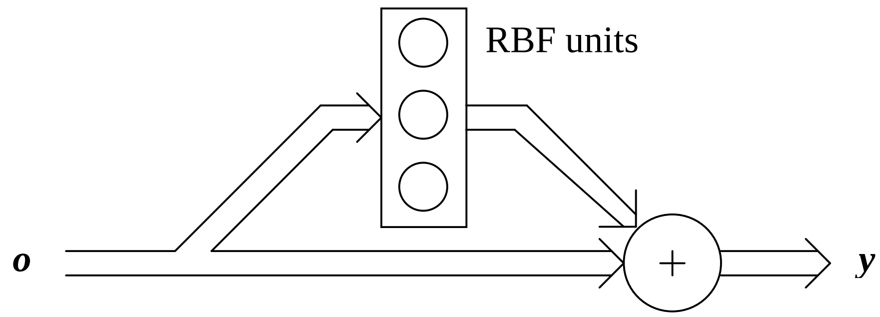
Separation after 350 epochs. Improves very slowly over the next 600 epochs

Why can't the system "spread" the high-density region into the low-density one?

Possible cause: The units of the MLP are non-local. Moving a unit to improve a part of the space would harm some other part of the space.

Solution

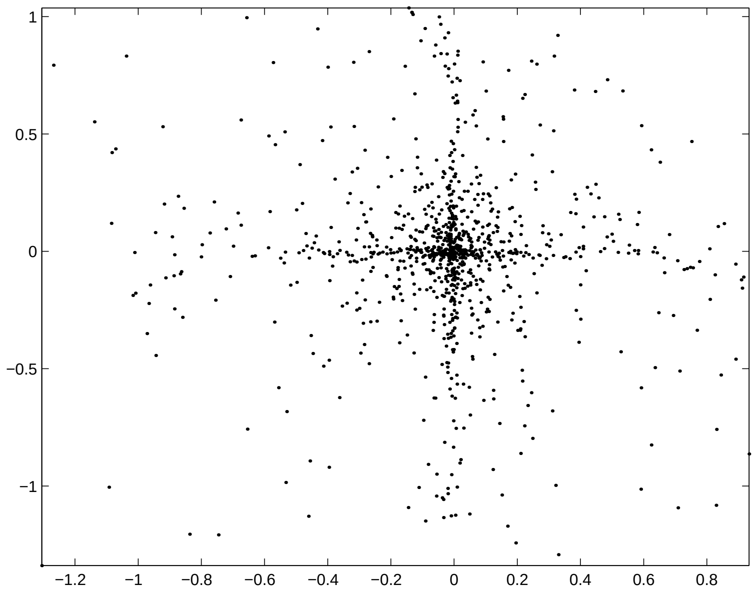
Use local units (e.g. Radial Basis Function units)



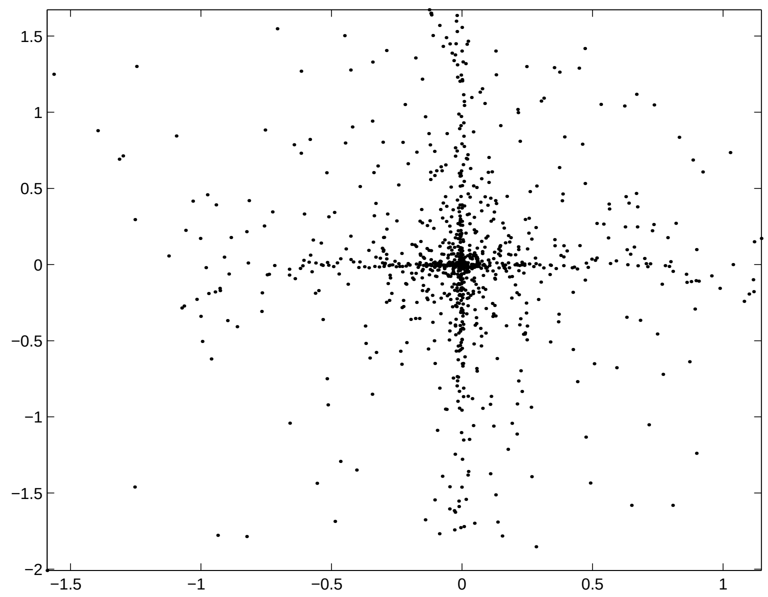
Nonlinear ICA block

- The direct connections yield a linear mapping, which is then modified by the RBF units.
- The RBF units' centers are trained by K-means. Radiuses are computed by a simple heuristic.
- Only the output weights are trained by the gradient of the objective function.

Results



MLP



RBF

Number of epochs	Two supergaussians		Supergaussian & subgaussian	
	MLP	RBF	MLP	RBF
Mean	500	68	610	233
St. deviation	152	10	266	87

But more recent results

(which didn't make it into this paper)

- The speed advantage of RBFs may be due more to initialization than to locality.
- MLPs with hidden units initialized "crisscrossing" the whole observation space have shown learning speeds comparable to those of RBFs.
- MLPs don't usually need explicit regularization, but RBFs do – and the amount of regularization has to be adjusted by hand in each case.
- Download the most recent preprint (see last page).

Conclusions

- Extension of INFOMAX
- ICA performed by minimizing the mutual information of the extracted components.
- Estimation of the independent components and of their distributions performed by a single network, with a single objective function.
- Can handle a wide variety of components' distributions.
- Able to perform linear and nonlinear ICA and nonlinear source separation.
- Networks of local units yield better performance
    - But this may have more to do with a good initialization than with the local units (see preprint of new paper submitted to Signal Processing)

A related issue

Is nonlinear source separation really possible?

- *Purely blind* nonlinear source separation is an ill-posed problem.

  It has an infinite number of solutions, not trivially related to one another.

- But we often solve ill-posed problems (e.g. the training of multiplayer perceptrons).

- What we need is some extra information, that is often available (e.g. smoothness).

- We can then use regularization to find an essentially unique solution.

- In our MLP-based examples, the regularization inherent to the MLP sufficed.

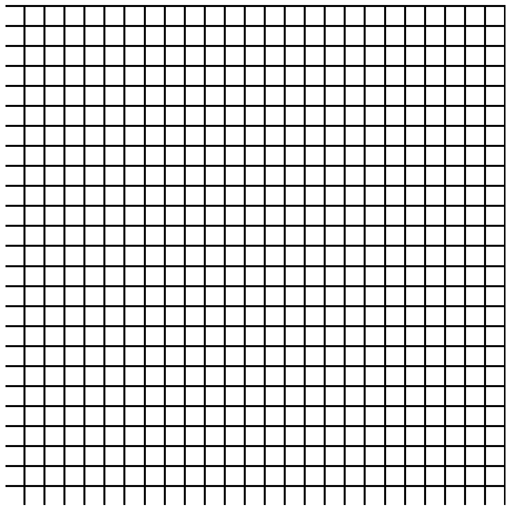
- In the RBF-based examples we needed explicit regularization – weight decay.

- **But in all our test cases we were able to perform nonlinear source separation.**

Christian Jutten's counter-example

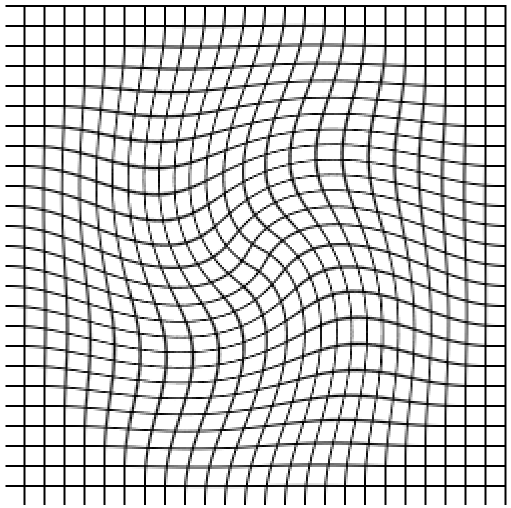
(NIPS 2002 workshop)

Identity mapping (uniform)



This is the smoothest possible mapping

Twisted mapping (still uniform)



This is less smooth…

A smoothing regularizer would select the first mapping, and not the second one.

Most recent and most comprehensive preprint

Luis B. Almeida, "MISEP – Linear and Nonlinear ICA Based on Mutual Information", submitted to *Signal Processing*, special issue on ICA.

Download at http://neural.inesc-id.pt/~lba/papers/AlmeidaSigProc2003.pdf

Probably also already available at the COGPRINTS archive, http://cogprints.ecs.soton.ac.uk/ (search for 'MISEP' in the title)

MATLAB – compatible toolkit

http://neural.inesc-id.pt/~lba/ica/mitoolbox.html