ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Unsupervised and reinforcement learning in neural networks
1st mini project

Lecturer: Wulfram Gerstner

Assistants: Mohammadjavad Faraji
Felipe Gerhard
Friedemann Zenke

Independent Component Analysis

1 Find a single IC

Your task is to extract a single independent component (IC) from a mixed signal. To generate such a signal, take the function "miximages2" which will randomly mix two grayscale images to two data samples. This function is contained in the archive "miniproject1.zip" which you can download from the Moodle page.
Next, use the following algorithm to perform a "fastICA" and to extract the weights for a single independent component.

1.1 Algorithm

1. Center the data $\{\vec{z}^\mu\}$ so that the mean is zero.
2. Whiten the data $\{\vec{z}^\mu\}$.
3. Choose a random vector $\vec{w}$ of the size of your count of different input channels.
4. Compute: $\vec{w} = \frac{1}{N}\sum_\mu(\vec{z}^\mu g(\vec{w}^T\vec{z}^\mu)) - \vec{w}\frac{1}{N}\sum_\mu(g'(\vec{w}^T\vec{z}^\mu))$.
5. Normalize: $\vec{w} = \vec{w}/||\vec{w}||$.
6. Iterate 4-6 until converged.

Some points of the algorithm need clarification:

- For step 2: Whitening the data means to transform it linearly so that the components of the input vector are uncorrelated and have a variance of one. To do this use the function "whitendata" from the downloadable "miniproject1.zip".
- For step 4: Here a nonlinear function g and its derivative g' are required. You should try all of the following functions and compare the results you get with each one:

  $g_1(y) = \tanh(y)$
  $g_2(y) = y\,\exp(-y^2/2)$
  $g_3(y) = y^3$

- For step 6: Here you need to find a convergence measure to determine the quality of the independent component.

1.2 Report

Your report should include:

1. Your code for the fastICA finding a single IC. Which convergence measurement did you use?
2. Evaluations of the different nonlinear functions g. Use the data from "miximages2.m". Which measurement did you use to compare them?
3. Visualizations of the reconstructed images from the "miximages2" data. Reconstruct the first image using the appropriate weights and then think of a way to recover the second signal, too. For that purpose you should not yet use the method of the next part of the assignment.

2 Find all ICs

Your next task is to find all independent components at once. You will try your implementation on eight real images that are linearly mixed from eight sources. The data are provided in ".gif" format along with a function "miximages8" which will load and randomly mix these eight image sources. They are also part of the archive "miniproject1.zip". You will then have to find a way to assign each found signal to its original source.
To find all ICs you have to use a modified algorithm:

2.1 Algorithm

1. Center the data $\{\vec{z}^\mu\}$ so that the mean is zero.
2. Whiten the data $\{\vec{z}^\mu\}$.
3. Choose n random vectors $\vec{w}_i$ of the size the count of your different input channels. n is the number of desired independent components.
4. Compute for all vectors $\vec{w}_i$: $\vec{w}_i = \frac{1}{N}\sum_\mu(\vec{z}^\mu g(\vec{w}_i{}^T\vec{z}^\mu)) - \vec{w}_i\frac{1}{N}\sum_\mu(g'(\vec{w}_i{}^T\vec{z}^\mu))$.
5. Normalize all vectors: $\vec{w}_i = \vec{w}_i/||\vec{w}_i||$.
6. Do a symmetric orthogonalization on the matrix: $W = (\vec{w_1}, ..., \vec{w_n})^T$.
7. Iterate 4-6 until converged.

Again, here are some clarifications for the algorithm;

- For steps 1 to 5: These are the same as for a single IC. You just have to apply them to each IC you want to compute.
- For step 6: A symmetric orthogonalization is done like this:

$$W = (WW^T)^{-1/2}W$$

- For step 7: You should try to find a measure of convergence that works for multiple ICs as well as for single ICs.

2.2 Report

Your report should include:

- Your code for the fastICA with all ICs.
- An explanation on what the symmetric orthogonalization does and why it is necessary.
- A set of reconstructed images.
- A description and the code of your method to assign the restored signals to the corresponding sources.

3 Suggestions & Clarifications

3.1 Submission of your results

Submission takes place via the "moodle" web page. You should upload a zip file (named after your last name) containing:

- Your report in pdf format (maximum 4 pages).
- Your source code for all tasks in the original format.

The deadline is Sunday, 28 October 2012, 23:55. You may work in teams of two persons but you should both upload your report.

3.2 Programming environment

The functions mentioned above are given as Matlab and Python source code. All programming languages are accepted for the project.
To use the Python functions from the "miniproject1.zip" file, you need the Numpy and Scipy packages, which are anyway very strongly recommended for scientific computing in Python. Have a look at the functions in the "Scipy.stats" module for useful functions. As an open source alternative to Matlab you can use Octave which accepts the given functions.