Machine Intelligence II | **Exercise Sheet 7**
SS2012, Prof. Obermayer | due: 07.06.2012

Kurtosis, negentropy, and the independent components of image patches

In this exercise we will search for independent components by maximizing *"non-Gaussianity"*. First, we will use the *kurtosis* as a measure and apply it to toy data from different distributions. Next, approximations of *negentropy* (as implemented in the *FastICA* algorithm) shall be applied to separate mixed toy signals. Finally we will use this to find the independent features of images.

Additional Material: Download the archive `datafilesICA.zip` from the ISIS platform. For the second and the third problem you are asked to apply the *FastICA* algorithm. Links to toolboxes for *Matlab* and *Python* implementing the algorithm be downloaded from the links given on ISIS. `.mat` files can be read with `loadmat` from `scipy.io` and will be stored as a dictionary containing the variables.

7.1 Kurtosis of Toy Data (4 points)

Load the file `distrib.mat`, which contains three toy datasets (`uniform`, `normal`, `laplacian`), each 10000 samples of 2 sources. Do the following for each dataset:

1. Apply mixing matrix $\mathbf{A}$ to the original data $\mathbf{s}$:

$$\begin{aligned} \mathbf{A} &= \begin{pmatrix} 4 & 3 \\ 2 & 1 \end{pmatrix} \\ \mathbf{x} &= \mathbf{A}\mathbf{s}. \end{aligned}$$

2. Center the mixed data to zero mean.
3. Decorrelate the data by applying principal component analysis (PCA) and project them onto the principal components (PCs).
4. Scale the data to unit variance in each PC direction (now the data is *whitened* or *sphered*).
5. Rotate the data by different angles θ

$$\begin{aligned} \mathbf{x}_\theta &= \mathbf{R}_\theta \mathbf{x} = \begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{pmatrix} \mathbf{x} \\ \theta &= 0, \frac{\pi}{50}, \ldots, 2\pi, \end{aligned}$$

   and calculate the kurtosis for each dimension:

$$kurt(x_\theta) = \langle x_\theta^4 \rangle - 3\underbrace{\langle x_\theta^2 \rangle}_{=1}^2.$$

6. Find the minimum and maximum kurtosis value for the first dimension and rotate the data accordingly.

- Plot the original dataset (sources) and the mixed dataset after the steps 1, 2, 3, 4, and 6 as a scatter plot and display the respective marginal histograms. After step 5 plot the kurtosis value as a function of angle for each dimension.
- Compare the histograms for θ_{min} and θ_{max} for the different distributions.

7.2 Toy Signal Separation (2 points)

Generate three signals as row vectors given at time points $t = 0,\ 0.05, \ldots, 50$ by

$$\begin{aligned} s_1(t) &= 4\,sin(t-3) \\ s_2(t) &= t+5 \mod 10 \\ s_3(t) &= \begin{cases} -14 & \text{if } cos(2t) > 0 \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

1. Mix the signals to get $\mathbf{x} = \mathbf{A}\mathbf{s}$ where

$$\mathbf{A} = \begin{pmatrix} 2 & -3 & -4 \\ 7 & 5 & 1 \\ -4 & 7 & 5 \end{pmatrix}$$

2. Whiten the mixed (observed) signals $\mathbf{x}$ using the function `fastica` and separate the signals using the same function.

Plot the original source signals, the mixtures, the whitened mixtures, and the unmixed signals.

7.3 ICA on Image Patches (4 points)

Use the files in directory `images`, which contains three categories of images: *nature*, *buildings*, and *text* (prefixes `n,b,t`). Proceed as follows for each category:

1. Sample P patches of $\sqrt{N} \times \sqrt{N}$ pixels from all images of one category and rearrange each sample to a column vector. Choose number and size of the patches depending on your memory and CPU power. Recommended are $P \geq 20000$ and $N \geq 144$.
2. Calculate the independent features of the image patches (these are the columns of mixing matrix $\mathbf{A}$). Use the function `fastica` from the toolbox to compute this matrix:
   - Let `fastica` perform PCA and whitening of the data (default setting).
   - Use the symmetric approach (parameter '`approach`') and tanh as the non-linearity (parameter '`g`').
3. Show the first 20 independent features as (grayscale) image patches by rearranging the vectors into $\sqrt{N} \times \sqrt{N}$ matrices. In *Matlab* use `imagesc` for visualizing.

Compare the results for the different categories.

Total points: 10