Corrections: Independent Component Analysis

Exercise 1

1.1. Let’s calculate the second and fourth moments of $y$:

$$E[y^2] = E[(x_1 + x_2)^2] = E[x_1^2 + 2x_1x_2 + x_2^2] = E[x_1^2] + E[2x_1x_2] + E[x_2^2]$$

$$= E[x_1^2] + 2E[x_1]E[x_2] + E[x_2^2] = E[x_1^2] + E[x_2^2],$$

and

$$E[y^4] = E[(x_1 + x_2)^4] = E[x_1^4 + 4x_1^3x_2 + 6x_1^2x_2^2 + 4x_1x_2^3 + x_2^4]$$


$$= E[x_1^4] + 6E[x_1^2]E[x_2^2] + E[x_2^4].$$

Thus the kurtosis of $y$ is:

$$\kappa(y) = E[y^4] - 3E[y^2]^2 = E[x_1^4] + 6E[x_1^2]E[x_2^2] + E[x_2^4] - 3(E[x_1^2] + E[x_2^2])^2$$

$$= \underbrace{E[x_1^4]}_{\kappa(x_1)} - 3\underbrace{E[x_1^2]^2}_{\kappa(x_2)} + 6\underbrace{E[x_1^2]E[x_2^2]}_{0} - 6\underbrace{E[x_1^2]E[x_2^3]}_{0} + \underbrace{E[x_2^4]}_{\kappa(x_2)} - 3\underbrace{E[x_2^4]}_{\kappa(x_2)}$$

$$= \kappa(x_1) + \kappa(x_2).$$

1.2. We simply calculate:

$$\kappa(y) = E[\alpha^4x_1^4] - 3E[\alpha^2x_1^2]^2 = \alpha^4(E[x_1^4] - 3E[x_1^2]^2) = \alpha^4\kappa(x).$$

1.3. Thanks to 1.1 and 1.2 above, this is simple:

$$\kappa(y) = \kappa(\sqrt{a}x_1 + \sqrt{1-a}x_2) = \kappa(\sqrt{a}x_1) + \kappa(\sqrt{1-a}x_2) = a^2\kappa(x_1) + (1-a)^2\kappa(x_2).$$

1.4. Since we want to find maxima and minima, let’s compute the derivative $dk(y)/da$:

$$\frac{dk(y)}{da} = \frac{d}{da} \left[ a^2\kappa(x_1) + (1-a)^2\kappa(x_2) \right] = 2a\kappa(x_1) + 2(1-a)\kappa(x_2) = 2ac + 2(a-1)d.$$
This is 0 only if \( a = \frac{d}{c+d} \). To find whether this corresponds to a minimum or a maximum, we compute the second derivative

\[
\frac{d^2 \kappa(y)}{da^2} = \frac{d}{da} \left[ 2a \kappa(x_1) + 2(a-1) \kappa(x_2) \right] = 2\kappa(x_1) + 2\kappa(x_2) = 2c + 2d.
\]

This is positive for all allowed values of \( a \), so the point \( a^* = \frac{d}{c+d} \) is a minimum. Thus the maxima happen at the boundaries of the domain, \( a = 0 \) and \( a = 1 \). Since \( \kappa(x_2) > \kappa(x_1) \), the global maximum happens for \( a = 0, y = \kappa(x_2) \), see Figure 1.

This means that for linear, whitened mixtures of super-gaussian signals, maximizing the kurtosis leads to the recovery of one of the two signal. Using an analytical method, as we did, we are guaranteed to find the global minimum, but usually one would use a numerical gradient ascent method on the data. Even then, the graph shows clearly that one would recover one of the two statistically independent signals, although not necessarily the one with highest kurtosis.

**Figure 1:** The kurtosis of the mixture \( y \) as a function of \( a \), in exercises 1.4 and 1.5.

1.5. Similarly as above, there is a fixed point in \( a^* = \frac{d}{c+d} \), and the second derivative is \( \frac{d^2 \kappa(y)}{da^2} = 2\kappa(x_1) + 2\kappa(x_2) \). Since this time the second derivative is negative, this point is a local (and global) maximum, see Figure 1.

In contrast to the situation of exercise 1.4, maximizing the kurtosis for a mixture of sub-gaussian signals is a bad idea, since it finds the linear combination that is most gaussian! In that case, the correct approach would be to minimize the kurtosis.

**Exercise 2**

2.1. We just derive \( F'(y) \), remembering that \( (f(g(x)))' = f'(g(x)) g'(x) \), and that \( \cosh' = \sinh \):

\[
\frac{d}{dy} \left[ \frac{1}{a} \log \cosh(ay) \right] = \frac{1}{a} \cdot \frac{1}{\cosh(ay)} \cdot \sinh(ay) \cdot a = \tanh(ay).
\]

2.2. This time we use the chain rule: since \( F \) depends on \( w_j \) only through \( y \), we can write

\[
\frac{dF}{dw_j} = \frac{dF}{dy} \frac{dy}{dw_j} = \tanh(ay) x_j.
\]
2.3. The gradient ascent rule suggested above is:

\[ \Delta w_j = \eta \frac{d}{dw_j} J(\vec{w}) = \eta \frac{d}{dw_j} \langle F(\vec{w}^T \vec{x}) \rangle = \eta \frac{d}{dw_j} F(\vec{w}^T \vec{x}) = \frac{\eta}{M} \sum_{\mu=1}^{M} \tanh(ay^\mu) x_j^\mu, \]

where \( \eta > 0 \) is a learning rate. To convert this into a neural Hebbian rule, we make two hypotheses:

1. The learning rate \( \eta \) is small enough, so that we can approximate the current \textit{batch} mode (learning for all samples to be presented and update the weights only then) with an \textit{online} mode (weights are updated at each sample presentation) with learning rate \( \eta/M \).

2. We define our output neuron rate to be \( x_{out} = \tanh(a\vec{w}^T \vec{x}) \).

In that case the online version of the rule above becomes

\[ \Delta w_j^\mu = \frac{\eta}{M} x_{out} x^\mu_j, \]

which is indeed a Hebbian learning rule.

**Exercise 3**

3.1. Using the general formula for the Taylor expansion in multiple dimensions,

\[ J(\vec{w}) = \sum_{j=0}^{\infty} \frac{1}{j!}((\vec{w} - \vec{w}_0)^T \cdot \frac{\partial}{\partial \vec{w}'} f(\vec{w}')) \bigg|_{\vec{w}'=\vec{w}_0}, \]

we get the expression for approximations up to second order as:

\[ J^*(\vec{w}) = \langle F(\vec{w}_0^T \vec{x}) \rangle + (\vec{w} - \vec{w}_0)^T (g(\vec{w}_0^T \vec{x}) \vec{x}) + \frac{1}{2} (\vec{w} - \vec{w}_0)^T H (\vec{w} - \vec{w}_0), \]

with \( H \) being the Jacobian matrix with entries \( H_{ij} = \frac{\partial^2 J}{\partial w_i \partial w_j} \), and \( g(y) = \frac{dF(y)}{dy} \). \( H \) is given by:

\[ H = \frac{\partial}{\partial \vec{w}} (g(\vec{w}^T \vec{x}) \vec{x}). \]

3.2. Let us compute the gradient of \( J^* \) with respect to \( \vec{w} \):

\[ \langle g(\vec{w}_0^T \vec{x}) \vec{x} \rangle + H(\vec{w}_{new} - \vec{w}_0). \]

The gradient is zero for the extreme point \( \vec{w}_{new} \):

\[ 0 = \langle g(\vec{w}_0^T \vec{x}) \vec{x} \rangle + H(\vec{w}_{new} - \vec{w}_0). \]

Let us have a closer look at the Jacobian matrix \( H \):

\[ H = \frac{\partial}{\partial \vec{w}} (g(\vec{w}^T \vec{x}) \vec{x})|_{\vec{w}=\vec{w}_0} = \langle g'(\vec{w}_0^T \vec{x}) \vec{x} \vec{x}^T \rangle \approx \langle g'(\vec{w}_0^T \vec{x}) \rangle \langle \vec{x} \vec{x}^T \rangle = \langle g'(\vec{w}_0^T \vec{x}) \rangle E, \]

with the identity matrix \( E \), since \( \vec{x} \) is pre-whitened \( (\langle \vec{x} \vec{x}^T \rangle = E) \).

Plugging this into the expression, we get:

\[ 0 = \langle g(\vec{w}_0^T \vec{x}) \vec{x} \rangle + \langle g'(\vec{w}_0^T \vec{x}) \rangle (\vec{w}_{new} - \vec{w}_0). \]
Solving for \( \vec{w}_{\text{new}} \) yields:

\[
-c \langle g'(\vec{w}_0^T \vec{x}) \rangle \vec{w}_{\text{new}} = \langle g(\vec{w}_0^T \vec{x}) \rangle - \langle g'(\vec{w}_0^T \vec{x}) \rangle \vec{w}_0.
\]

Since \( c = -\langle g'(\vec{w}_0^T \vec{x}) \rangle \) is just a scalar quantity, we can neglect it in the update as long as we properly normalize the new vector to satisfy \( ||\vec{x}_{\text{new}}|| = 1 \). We end up with:

\[
\vec{w}_{\text{new}} = \langle g(\vec{w}_0^T \vec{x}) \rangle - \langle g'(\vec{w}_0^T \vec{x}) \rangle \vec{w}_0,
\]

which corresponds to the fastICA update rule.

**Exercise 4**

Since \( D \) is assumed to be zero-mean, \( E_D(y) = w^T E_D(x) = 0 \) and so

\[
\text{var}(y) = \langle y^2 \rangle = \langle (w^T x)^2 \rangle = w^T \langle xx^T \rangle w = w^T C w
\]

where \( C \) is the covariance matrix of dataset \( D \). So, maximizing the variance \( \text{var}(y) \) subject to \( ||w|| = w^T w = 1 \) can be written as

\[
\max_w [L(w, \lambda) = w^T C w - \lambda (w^T w - 1)]
\]

Taking the derivative of \( L \) wrt \( w \) and setting it to zero to find the stationary points, we have

\[
\frac{dL}{dw} = 2Cw - 2\lambda w = 0 \Rightarrow cW = \lambda w
\]

So, the optimal \( w \) have to form an eigenvector of the covariance matrix \( C \). Among all the eigenvectors, \( w^* \), which is the eigenvector corresponding to the largest eigenvalue, is the optimal solution we look for because \( \text{var}(y) = w^T C w \) is maximized for such an eigenvector. Remember that \( w_n^T C w_n = \lambda_n \) where \( w_n \) is the eigenvector corresponding to the eigenvalue \( \lambda_n \) of the covariance matrix \( C \).

**Exercise 5**

Let’s calculate the time-delayed covariance matrix of signals \( \vec{x}^\tau \):

\[
C^* = \langle \vec{x}^\tau \vec{x}^\tau \rangle = \langle R^T \vec{y} \vec{y}^T \rangle = \langle R^T (\vec{y}^\tau)^T \rangle R = R^T C R,
\]

where \( C \) is the original covariance matrix. We verify that \( C^* \) is indeed symmetric:

\[
C^{*T} = (R^T C R)^T = R^T (R^T C)^T = R^T \underbrace{C^T}_C \underbrace{R^T}_R = R^T C R = C^*.
\]