IEEE Transactions on Neural Networks, Vol.9, No.4, pp.613-628, (1998)

Noise Suppressing Sensor Encoding and Neural Signal Orthonormalization

Rüdiger W. Brause
Michael Rippl[+)]
J. W. Goethe-University, Computer Science Dep., Frankfurt a.M.
brause@informatik.uni-frankfurt.de
[+)] *now at Microsoft Corporation, mike@microsoft.com*

Abstract

In this paper we regard first the situation where parallel channels are disturbed by noise. With the goal of maximal information conservation we deduce the conditions for a transform which „immunizes" the channels against noise influence before the signals are used in later operations. It shows up that the signals have to be decorrelated and normalized by the filter which corresponds for the case of one channel to the classical result of Shannon. Additional simulations for image encoding and decoding show that this constitutes an efficient approach for noise suppression.

Furthermore, by a corresponding objective function we deduce the stochastic and deterministic learning rules for a neural network that implements the data orthonormalization. In comparison with other already existing normalization networks our network shows approximately the same in the stochastic case but, by its generic deduction ensures the convergence and enables the use as independent building block in other contexts, e.g. whitening for independent component analysis.

Keywords: information conservation, whitening filter, data orthonormalization network, image encoding, noise suppression.

1 Introduction

In many sensor encoding tasks the ability to deal with the noisy environment is of crucial importance. More specifically, we regard the situation when a signal **x** has been encoded and has to be transmitted through a noisy environment and has to be reconstructed afterwards. In Fig. 1.1 this situation is shown.

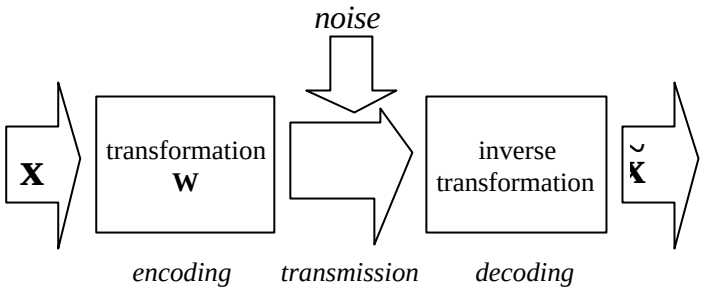

Fig. 1.1 *The signal encoding situation*

This situation is met for instance in image encoding and transmission systems as well as (replacing the reconstruction of a pattern by a stage which needs to distinguish between different patterns, e.g. classification or memorization) in the noisy environment of nervous brain tissue.

For the scalar signal x, this situation has been treated by Shannon [15] with the performance criterion of information maximization for the reconstructed $\tilde{x}$. He assumed a stationary signal of finite power and, after a spectral decomposition, treated it as being composed of independent frequency channels. He got as optimality condition for maximal information transmission that all frequency components should be transformed to equal variance by the transformation W(x). The necessary linear transformation W is called a *whitening filter*.

The idea of the whitening filter can be extended in our case from one channel to several parallel signal channels. The next section will show us how we have to treat the data to obtain maximal noise immunity on several, parallel channels by data orthonormalization. Analog to the classical result of Shannon, conditions can be obtained for the actual case of noise immunity for many parallel signals, **x** being a vector. This was done for instance by Plumbley [14].

In the next section an alternative mathematical treatment for the solution for parallel, disturbed channels is provided where we obtain the same conditions for the sensor encoding which are multi-channel generalizations of the whitening filter.

After this, we show by simulations for a real image that the analytically obtained noise suppression conditions are valid, especially in the case of image encoding.

In section 3 it is shown that the orthonormalization conditions derived for parallel channels correspond well to the conditions which are often used intuitively in a preprocessing stage of data fusion, prior to some operations like classification and decision implemented by neural networks. New approaches for data orthonormalization are also welcome for neural networks which implement blind source seperation.

To reflect this needs, we construct a new neural network in section 4 for implementing the information-optimal filtering and orthonormalization and derive mathematically its error and its convergence properties.

In section 5, the learning performance of the new network is compared with the well-known networks of Silva and Almeida.

A discussion concludes the paper.

2 Information transfer in the presence of noise

Now, let us consider the situation of Fig. 1.1, specified in more detail by the following Fig. 2.1.

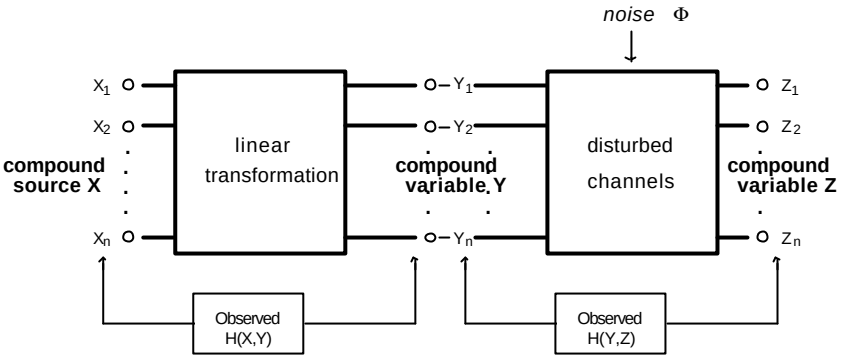

Fig. 2.1 *The information situation*

Let us assume that we have a linear transformation W which prepares the multi-channel signal $\mathbf{x} = (x_1,..,x_n)$, i.e. an instance of the compound random variable X, against noise Φ by

$$\mathbf{y} = W(\mathbf{x}) = \mathbf{W}\mathbf{x} \qquad (2.1)$$

How should **W** be designed to obtain a maximal information transfer from **y** to **z** through the noisy channel for a limited signal power P_y ?

2.1 Maximal transinformation for noisy parallel channels

To resolve this question, let us define the problem more formally. First, we model the variables **x**, **y** and **z** as compound stochastic variables X, Y and Z. For these variables, we can define the *mutual information* or *transinformation* by

$$H(X;Y) := H(X) + H(Y) - H(X,Y)\,. \qquad (2.2)$$

which can be generalized (see [12]) to

$$H\left(X^1; X^2; \ldots; X^n\right) := H\left(X^1\right) + H\left(X^2\right) + \ldots + H\left(X^n\right) - H\left(X^1, X^2, \ldots, X^n\right)$$

which is equal to zero iff the random variables X^i are independent.

Now, let us assume that the signal x_i has Gaussian properties and is centered. This is often the case in real world signals, for instance in some image and short time speech statistics especially when they are mixtures of several independent sources. Additionally, the noise Φ is also normally distributed.
This means e.g. for the signal **x**

$$p(x_1, x_2, \ldots x_n) = \frac{1}{\sqrt{(2\pi)^n}\sqrt{|\det C_{XX}|}}\, e^{\underbrace{-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} C_{XX}^{-1}[i,j]\, x_i\, x_j}_{=x^T C_{XX}^{-1} x}} \qquad (2.3)$$

with the covariance matrix $\mathbf{C}_{XX}$ of the compound random variable X.
Additionally, we know that a continuous probability density is transformed by the condition

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\ldots\int_{-\infty}^{\infty} p(x_1, x_2, \ldots, x_n)\, dx_1\, dx_2 \ldots dx_n = 1$$
$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\ldots\int_{-\infty}^{\infty} p(y_1, y_2, \ldots, y_n)\, dy_1\, dy_2 \ldots dy_n$$
$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\ldots\int_{-\infty}^{\infty} p(\mathbf{y}(\mathbf{x}))\det(\partial y_i / \partial x_j)\, dx_1 dx_2 \ldots dx_n$$

which means

$$p(y_1, y_2, \ldots y_n) = p(x_1, x_2, \ldots x_n)\det(\partial y_i / \partial x_j)^{-1}$$

or, for linear, neutral transforms with det(**W**)=1,

$$p(y_1, y_2, \ldots, y_n) = p(x_1, x_2, \ldots, x_n)$$

This means that our random variables X and Y are both normally distributed. Since the sum of normally distributed variables is also normally distributed, this is the case for all three random variables X, Y and Φ. In order to maximize the transinformation H(Y;Z) we have to realize that

$$H(Y,Z) = H(Y,\Phi) \qquad (2.4a)$$

and

$$H(Y,\Phi) = H(Y) + H(\Phi) \qquad (2.4b)$$

The first equation can be results by the fact that the compound random variable $\mathbf{a} \equiv (Y,Z)$ is a linear transform of $\mathbf{b} \equiv (Y,\Phi)$ by

$$\begin{bmatrix} Y \\ Z \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} Y \\ \Phi \end{bmatrix} \text{ or } \mathbf{a} = \mathbf{W}\mathbf{b}$$

Since we have det(**W**) = 1 we get p(**a**)=p(**b**) and therefore (2.4a).The second equality (2.4b) is due to $p(Y,\Phi) = p(Y)p(\Phi)$ for independent signals Y and noise Φ.
Thus, with eqs.(2.2) and (2.4a,b) we get for the mutual information between the channels

$$H(Y;Z) = H(Y) + H(Z) - H(Y) - H(\Phi) = H(Z) - H(\Phi) \qquad (2.5)$$

With the average information or entropy H(X) of a normally distributed, centered X with equivariant noise we get for eq.(2.5), cf. appendix A, eqs.(A.1) and (A.2):

$$H(Y;Z) = \frac{1}{2} \ln \det\left(\frac{1}{P_\Phi} \mathbf{C}_{YY} + I \right) \qquad (2.6)$$

To maximize the monotone increasing function H(Y;Z), it is sufficient to maximize an objective function defined by its argument

$$R_1(Y,Z) = \det\left(\frac{1}{P_\Phi} \mathbf{C}_{YY} + I \right)$$

We know that for the determinant of a matrix $\mathbf{A} = (A_{ij})$ the property

$$\det \mathbf{A} \leq A_{11}A_{22}\cdots A_{nn}$$

holds (Wegners theorem, see [2] or [12]) with the equality iff $\mathbf{A}$ is a diagonal matrix. Therefore, our objective function R_1 takes its maximum when

$$\max_{A_{ij}} (\det \mathbf{A}) = \max_{A_{ii}} \left(\prod_i A_{ii} \right)$$

i.e. the objective function R_2 defined by

$$R_2(c_{11}, ..,c_{nn}) = \prod_i \left(\frac{1}{P_\Phi} c_{ii} + 1 \right), \quad c_{ij}=0 \ \forall j \neq i \qquad (2.7)$$

becomes a maximum under the constraint of finite, e.g. constant signal power

$$P_Y = \sum_i c_{ii} = \text{const} \qquad (2.8)$$

This goal is achieved by using a Lagrange function

$$L(c_{11}, ..,c_{nn},\lambda) = R_2(c_{11}, ..,c_{nn}) + \lambda(P_Y - \sum_i c_{ii})$$

The conditions $\partial L/\partial c_{ii} = 0$ are necessary for a multi-dimensional extremum, a maximum in our case. These conditions are satisfied when e.g. for the elements c_{kk} and c_{ss} we have

$$\frac{\partial L}{\partial c_{kk}} = \frac{1}{P_\Phi}\left(\frac{1}{P_\Phi} c_{ss} + 1\right) \prod_{\substack{i \neq k \\ i \neq s}} \frac{1}{P_\Phi}\left(c_{ii} + 1\right) + \lambda = 0$$

$$= \frac{1}{P_\Phi}\left(\frac{1}{P_\Phi} c_{kk} + 1\right) \prod_{\substack{i \neq k \\ i \neq s}} \frac{1}{P_\Phi}\left(c_{ii} + 1\right) + \lambda = \frac{\partial L}{\partial c_{ss}}$$

Thus, for each pair of diagonal elements we can conclude

$$\frac{1}{P_\Phi}\left(\frac{1}{P_\Phi} c_{ss} + 1\right) = \frac{1}{P_\Phi}\left(\frac{1}{P_\Phi} c_{kk} + 1\right) \qquad \text{or} \quad c_{ss} = c_{kk}$$

which means with condition (2.8)

$$c_{ii} = P_Y/n, \quad c_{ij}=0 \quad \forall j \neq i \tag{2.9}$$

i.e. equal variance for all neural outputs.
What does this mean for the basis vectors of the transform ?
With

$$c_{ij} = \left\langle y_i\, y_j \right\rangle = \left\langle \mathbf{w}_i^T x x^T \mathbf{w}_j \right\rangle = \mathbf{w}_i^T \left\langle \mathbf{x}\mathbf{x}^T \right\rangle \mathbf{w}_j = \mathbf{w}_i^T\, \mathbf{C}_{XX}\, \mathbf{w}_j \tag{2.10}$$

and expanding the row vectors $\mathbf{w}_i^T$ of $\mathbf{W}$ in the base of eigenvectors $\{\mathbf{e}_r\}$ of $\mathbf{C}_{XX}$ we get with

$\mathbf{w}_i^T = (w_{1i}, .., w_{ni})$ in Cartesian $\equiv (a_{1i}, .., a_{ni})$ in eigenvector coordinates

and $a_{ij} = \mathbf{e}_i^T \mathbf{w}_j$ we get

$$c_{ij} = \left(\sum_{r=1}^{n} a_{ri}\, \mathbf{e}_r^T\right) \mathbf{C}_{XX} \left(\sum_{s=1}^{n} a_{sj}\, \mathbf{e}_s\right) = \left(\sum_{r=1}^{n} a_{ri}\, \mathbf{e}_r^T\right)\left(\sum_{s=1}^{n} a_{sj}\, \mathbf{C}_{XX}\, \mathbf{e}_s\right)$$

$$= \sum_{r=1}^{n}\sum_{s=1}^{n} a_{ri}\, a_{sj}\, \lambda_s \underbrace{\mathbf{e}_r^T \mathbf{e}_s}_{=0\ r \neq s} = \sum_{r=1}^{n} a_{ri}\, a_{rj}\, \lambda_r \left|\mathbf{e}_r\right|^2 = 0$$

We can see that the new base vectors $\mathbf{w}_i$ are orthogonal

$$\mathbf{w}_i^T\, \mathbf{w}_j = \sum_{r=1}^{n} a_{ri}\, a_{rj} = 0 \qquad i \neq j \tag{2.11}$$

if we scale the eigenvector base system by $|\mathbf{e}_r|^2 = \lambda_r^{-1}$. Please note that generally this is not the case in Cartesian coordinates.
The length of the new base vectors in the eigenvector base coordinates are computed by

$$c_{ii} = \left\langle y_i^2 \right\rangle = \mathbf{w}_i^T \mathbf{C}_{XX} \mathbf{w}_i = \left(\sum_{r=1}^{n} a_{ri} \mathbf{e}_r^T \right) \left(\sum_{s=1}^{n} a_{si} \lambda_s \mathbf{e}_s \right)$$

$$= \sum_{r=1}^{n} a_{ri}^2 \lambda_r \underbrace{|\mathbf{e}_r|^2}_{=1/\lambda_r} = \sum_{r=1}^{n} a_{ri}^2 = P_Y/n \equiv P \qquad i = 1, 2, \ldots, m.$$

to $$\mathbf{w}_i^T \mathbf{w}_i = \sum_{r=1}^{n} a_{ri}^2 = P \qquad (2.12)$$

Our result indicates that we should use a linear transform which decorrelates the output and normalizes the output variance to equal signal power P.

In the next section, we will use the orthonormalization approach for the noise resistant encoding of images. For this reason, let us first shortly introduce the *transform coding* approach for image encoding and modify it for our situation. Then, some simulations show the performance of the conditions deduced so far.

2.2 Orthonormalized transform coding

The standard transform coding approach for pictures, which is the base for several actual encoding schemes like JPEG or MPEG (cf. [17]) sees the pixels of an image as parallel signals which have to be encoded. For this purpose, the picture is subdivided into subimages (e.g. 8×8 pixels) and transformed by a linear transform into coefficients. Afterwards, the code coefficients are quantized according to a quantization table. For the reconstruction process, these procedures are inversely done. In Fig. 2.2 the encoding and decoding situation is visualized.

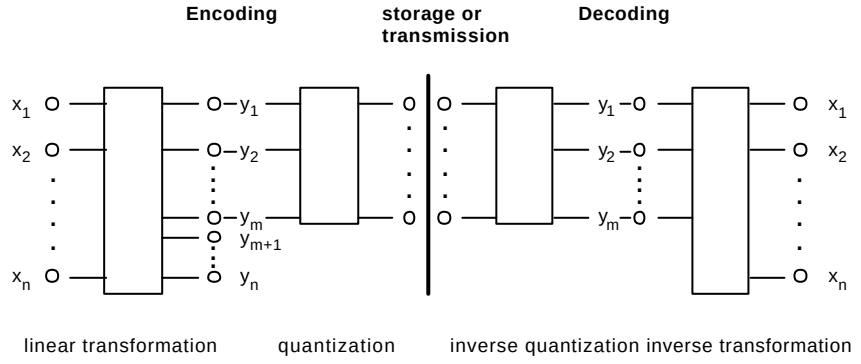

Fig. 2.2 *The transform coding approach*

What do our results of the previous section 2.1 mean for transform coding?

It is known that transform coding minimizing the least mean squared error (LMSE) for the reproduced images can be obtained by lateral inhibited neural networks implementing a principle component analysis (PCA), see [4]. For the linear transformation conditions (2.9), we have to take into account the conditions (2.11) and (2.12) for the weights. This means that we have no longer to implement a PCA, but to decorrelate and normalize the output data. This can be done by an infinity of base vector systems that satisfy the conditions above. Among them, a PCA with scaled eigenvectors according to the conditions of (2.11, 2.12) is just one sufficient solution, not a necessary one.

The reproduction is obtained by the inverse transform matrix $\mathbf{W}^{-1}=(\mathbf{b}_1 \ldots \mathbf{b}_n)$ which contains as base vectors

$$\mathbf{b}_i = 1/P\ \mathbf{C}_{XX}\mathbf{w}_i \qquad (2.13)$$

This can be easily verified, based on eqs.(2.10) and (2.11), because with

$$\mathbf{w}_i\mathbf{b}_j = 1/P\ \mathbf{w}_i\ \mathbf{C}_{XX}\ \mathbf{w}_j = 1/P\ c_{ij}$$

we get the equation

$$\mathbf{W}\ \mathbf{W}^{-1} = (\mathbf{w}_1 \ldots \mathbf{w}_n)^T\ (\mathbf{b}_1 \ldots \mathbf{b}_n) = 1/P\ \mathbf{C}_{YY} = \mathbf{E}$$

with the unity matrix $\mathbf{E}$. Thus, if we have the transform matrix $\mathbf{W}$ we easily get the inverse transform for calculating the reproduced signal by

$$\mathbf{x} = \mathbf{W}^{-1}\mathbf{y} = \sum_{i=1}^{n} y_i \mathbf{b}_i = \sum_{i=1}^{n} y_i \left((1/P)\mathbf{C}_{XX}\mathbf{w}_i\right) \qquad (2.14)$$

Now, let us regard the case when we reduce the number of coefficients from *n* to *m*, dropping *n-m* ones. This kind of compression is typical for the transform coding approach, see Fig. 2.2.
It is shown in appendix C that the resulting reproduction error is given by

$$\varepsilon^2 = (1/P)\sum_{r=1}^{n}\lambda_r \sum_{i=m+1}^{n} a_{ri}^2 \qquad (C.1)$$

This is minimized if the *n-m* weight vectors are the *n-m* eigenvectors with the least eigenvalues and the error becomes

$$\varepsilon_s^2 = \sum_{r=m+1}^{n}\lambda_r \qquad (C.2)$$

Thus, the base $\{\mathbf{w}\}$ of the transform consists of the *m* most significant eigenvectors. This result is in good correspondence to the classical results of PCA, whereas the scaled eigenvectors constitute only a special solution for the orthonormalization problem.

2.3 Orthonormalized image encoding

As the last and most important issue, let us investigate the benefits of our model: the noise immunity which it shares with all data orthonormalizing models. Let us regard the case where the parallel channels constitute the set of pixels of a discretized image. In the image encoding case, let us regard the unfavorite case of a picture with not many regularities, a face, shown in Fig. 2.3.

Fig. 2.3 *The sample picture "Zoe"*

We divided the whole picture in subpicture blocks of the size of 8x8 pixels, where each block might form the 64-dim input vector for neural networks which are discussed in more detail in the next section. The ensemble of all blocks of this picture gives us the image statistic. To show the pure characteristics of the transform, we excluded the problem of network convergence which depends on the neural model and simulation time (which is quite important for a net of 32 or 64 neurons) and assumed perfect convergence.

Therefore, we determined the eigenvectors and eigenvalues of the 64x64 covariance matrix with conventional approximation methods. We used for data compression the eigenvectors with the most significant eigenvalues. They formed the set of base vectors of the transformation. The eigenvectors were all scaled with the same scaling coefficient, the biggest eigenvalue λ_0, in order to obtain unit variance for the first component. For orthonormalization, the eigenvectors were additionally scaled $|\mathbf{e}_r|^2 = \lambda_r^{-1}$ by their eigenvalues, see eq. (2.11). This gives unit variance in all the components.

In the simulation, each image block was transformed to the encoding coefficients, superimposed by Gaussian noise with different variance and then transformed back again to the image. The squared difference between the original image and the reconstructed one is the error of the disturbed encoding.

If we drop some of the encoding coefficients additionally, we get an additional reconstruction error. To show the interdependence of the two different error sources, let us consider the two cases of on one hand fully using the 64 encoding coefficients and on the other hand only half of it, i.e. 32 coefficients.

Each situation is tried with three variances, σ^2=0, 0.001 and 0.01. The following table shows the mean squared error per pixel for using either the simple eigenvectors of the PCA analysis or the scaled eigenvectors of the normalization (called NPCA)

Transform	**m**	s^2**= 0.0**	**0.001**	**0.01**
PCA	64	0.0	59	591
NPCA	64	0.0	1	11
PCA	32	12	41	308
NPCA	32	12	13	22
PCA	16	28	42	175
NPCA	16	28	29	38

The table shows us the overlap effect of the two error sources. For a complete decomposition and reconstruction (m=64) with no noise present, the error is restricted on the pure computing error of the computer which is very small. Nevertheless, when we add noise, the NPCA scheme shows its strength by a smaller error of a factor about 60. The situation is illustrated in Fig. 2.4. On the left hand side, the reconstructed image of the 64 PCA components, corrupted heavily by noise, is shown. On the right hand side, benefits of the orthonomalization approach are well demonstrated. The visual error is dominated by the error in the first component (the average tile gray level) which does not profit by the variance downscaling process because its variance remains one.

a) PCA reconstruction b) NPCA reconstruction

Fig. 2.4 *The reconstructed image (m=64) of code corrupted by Gaußian noise of* s^2=0.01

For the case of dropped components (m=16), the noise influences the result only remarkably if it is larger than the error due to the neglected components as it is the case for σ^2=0.01. Here again, the normalization scheme works well.

a) PCA reconstruction b) NPCA reconstruction

Fig. 2.5 *The reconstructed image (m=16) of code corrupted by Gaußian noise of* s^2=0.01

There is an additional effect which can be seen in the table above: the error for using 64 noise-corrupted PCA components is much higher than the error when using only 16 components. This behavior is due to the fact that in the high components the image variance (σ^2 =0.00028 for k=64) is remarkably smaller than the noise variance (σ^2 =0.01 for k=64). Thus, the additional components do not reduce the error but are the source of additional error. This behavior is shown generally in Fig. 2.6. Here, the mean squared error per pixel is shown as a function of the number of reconstruction components for PCA and NPCA. Additional parameter are the three noise levels of σ^2 = 0, 0.001 and 0.01.

The behavior of the error can be better understood by the following arguments. In the PCA case, we have a linear, non-scaling transform by $|\mathbf{w}|$=1. Since the noise is not correlated to the image, we find as resulting reconstruction error ε^2 the sum of the *n-m* eigenvalues λ_i (the variance of the neglected components, see eq. (C.2)) and the noise of the *m* components used

$$\varepsilon^2 = \sum_{i=m+1}^{n} \lambda_i + m\sigma^2 \qquad (2.15)$$

The first term is a non-linear monotone decreasing function of *m* while the second term adds linearly. For $m \to \infty$ the error becomes proportional to the number of available components. This can be observed in Fig. 2.6 where the error function approaches a line with positive slope.

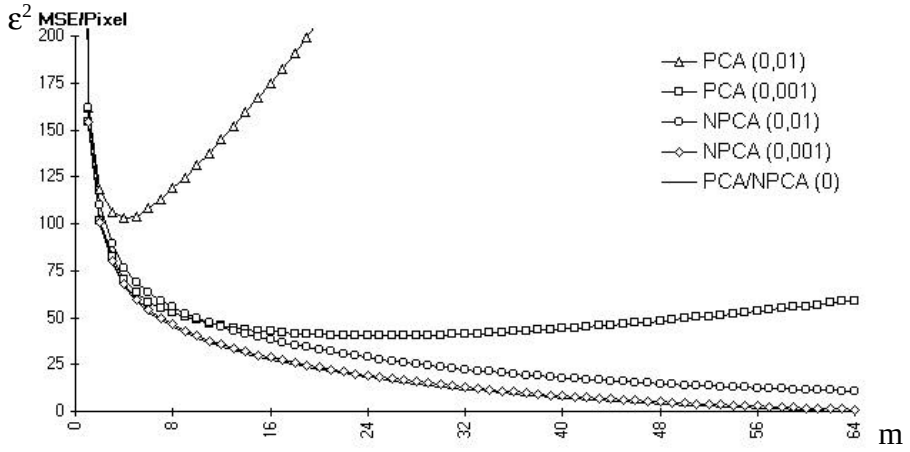

Fig. 2.6 *The mean square error as a function of the number of components and the noise level*

In the NPCA case, this is not the case. Since each coefficient *i* is transformed by the scaling factor of λ_i, the noise is also suppressed by a factor of λ_i.

$$\varepsilon^2 = \sum_{i=m+1}^{n} \lambda_i + \sum_{i=1}^{m} \lambda_i \sigma^2 = \sum_{i=m+1}^{n} \lambda_i + \sigma^2 \sum_{i=1}^{m} \lambda_i \qquad (2.16)$$

$$= \sum_{i=1}^{n} \lambda_i - \sum_{i=1}^{m} \lambda_i + \sigma^2 \sum_{i=1}^{m} \lambda_i = P - (1-\sigma^2) \sum_{i=1}^{m} \lambda_i$$

In this case the resulting reconstruction error is determined by the noise level as a fixed fraction of the reconstructing components. Here, the error becomes a decreasing function of *m*: the more components we have the smaller the error will be.

Now, before we present a new neural network model to implement the transformation above by massively parallel means, let us show that the demand for orthonormalization is also important in other situations than for the reconstruction of noisy images.

3 Other neural network applications of data orthonormalization

Beside the neural network implementation of noise resistant image transform coding by orthonormalization as we discussed it above there are a lot of other applications of data orthonormalization by neural networks.

3.1 Sensor fusion and data orthonormalization

In neural network modeling, especially for networks which deal with a mixture of different kind of real-world data, e.g. medical diagnosis which has to combine blood pressure, heart beat data and questionnaire data, a typical problem of data fusion arises: different kind of data sources give also rise to different amplitudes. Also if we combine two signals with the same analog values, e.g. an auditorial signal and a tactile signal, due to the different nature of the two sensors the scaling of the two signals are not the same. If we process them equally, the bigger one may dominate the learning process without any rational justification.

Thus, to obtain valuable results and to speed up learning, all data have to be transformed up to one unique scale. Here, the objective to have the same, normalized variance in each variable is a widely accepted scaling criterion. To obtain the normalized variance on each sensor channel, a linear normalization procedure is commonplace. This situation is shown in Fig. 3.1.

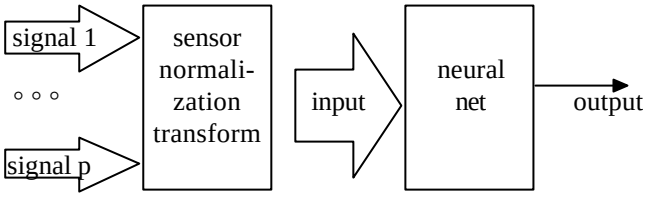

Fig. 3.1 *Sensor fusion and signal normalization*

Additionally, error correction procedures in the neural network (which has the task e.g. to classify the input or to approximate a given function of the input) encounter problems if the input channels are interdependent. Then, the error correction in one direction might counteract the error correction in another direction. A good procedure to circumvent this kind of problems consists of decorrelating the input channels before feeding them to the network. This concludes the specification of the sensor fusion network: it scales the signals to the same variance and additionally decorrelates them. This is done by a data orthonormalization procedure as it was deduced for the necessity of noise-immune sensor encoding.

3.2 PCA, independent component analysis (ICA) and data orthonormalization

The linear transformation of the input space to the base of principal components, which minimizes the mean squared error when dropping some of the output channels, is called Principal Component Analysis (PCA) and is obtained by aligning the base vectors to the directions of maximal variance. This is identical to a discrete Karhunen-Loève or Hotelling transformation.

The approach of PCA is only optimal for the performance measure of the mean squared error and assumes no specific information about the statistical properties of the observed signals. If we want to maximize other measures of information processing, for instance the information capacity of the encoding coefficients (i.e. the output signals of the transforming system), we have to obtain other properties.

Here, the mutual information $H(y_1;y_2;..;y_n)$ between the output channels is a good measure for an efficient output coding. The output information $H(y_1,y_2)$ of two channels y_1 and y_2

$$H(y_1,y_2) = H(y_1) + H(y_2) - H(y_1;y_2)$$

becomes maximal if for constant channel information $H(y_i)$ the mutual information becomes minimal. This is the case if

$$H(y_1,y_2) = H(y_1) + H(y_2)$$

which means

$$p(y_1,y_2) = p(y_1)p(y_2)$$

Thus, the demand for minimal transinformation is identical with the demand for independent channel probability distributions ("factorial code"). For *n* channels this means

$$p(\mathbf{x}) = p(x_1)p(x_2)\cdots p(x_n)$$

This demand for minimal transinformation can be used for a special situation. Let us assume that all observed signals $\mathbf{x}=(x_1,..,x_n)$ are composed by a linear mixture of independent source signals $\mathbf{s}=(s_1,..,s_n)$

$$\mathbf{x} = \mathbf{M}\mathbf{s}$$

How can the original source signals be reconstituted? Another linear transformation

$$y = \mathbf{B}\mathbf{x} = \mathbf{B}\mathbf{M}\mathbf{s}$$

might obtain the sources if

$$\mathbf{y} = \mathbf{s} \quad \Leftrightarrow \quad \mathbf{B}\mathbf{M} = \mathbf{I}$$

the demixing matrix **B** becomes the inverse of **M.**

The problem of finding this matrix **B** is known as the problem of "blind separation of sources" or "Independent Component Analysis" ICA and is a fast growing topic in neural network research, see e.g. [1], [7],[8], [10],..

The standard ICA procedure consists mainly of the following stages, shown in Fig. 3.2.

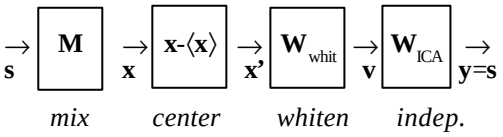

Fig. 3.2 *The processing stages in ICA*

After mixing the sources, the observed signals **x** are diminished by their first and second moments: They are centered, decorrelated and whitened to unit variance (orthonormalized) by a linear transform with a matrix $\mathbf{W}_{whit}$, and then separated by their higher moments in the last stage by a linear transform $\mathbf{W}_{ICA}$. The latter which uses the preprocessed input is often referred as "the ICA matrix".

The whitening is often performed by computing the PCA matrix using conventional methods and then rescaling the basis vectors with their corresponding eigenvalues by $|\mathbf{e}_r|^2 = \lambda_r^{-1}$. This is acceptable for quick data preprocessing, but it takes only one solution, impeding all other possible solutions which might also reflect other demands. For instance, the whole ICA process formally can also be obtained by one linear matrix only, combining the centering, whitening and ICA into one transform. Certainly, the resulting base vectors of this transform are not the eigenvectors; the PCA approach does not help here.

To obtain learning algorithms for this, more general neural network learning models for data orthonormalization are necessary. In the following section, we will therefore further investigate learning the data orthonormalization.

4 A new network model for data orthonormalization

There are several neural network models mentioned in the literature that implement the demands of eq.(2.9). Since the power is assumed to be normalized, they are termed *data orthonormalization* networks. The most well known ones are the nets of Silva and Almeida [16] and the one of Plumbley [14]. There are problems associated with these networks: The former one is purely heuristic which prohibits the combination with other constraints or conditions. Additionally, the signals are routed backwards through the weights which is biologically implausible. The latter one uses only constant feedforward weights which is not plausible either. Therefore, let us introduce a new network model in this section which avoids these problems.

It is well known that *n* linear neurons each implementing a scalar product

$$y = \sum_{j=1}^{n} w_j x_j = \mathbf{w}^T \mathbf{x}$$

as a network they implement a linear transform of eq.(2.1) by their parallel action

$$\left.\begin{aligned} y_1 &= \mathbf{w}_1^T \mathbf{x} \\ &\cdots \\ y_n &= \mathbf{w}_n^T \mathbf{x} \end{aligned}\right\} = \mathbf{W}\mathbf{x} \qquad (4.1)$$

This is shown in Fig. 4.1.

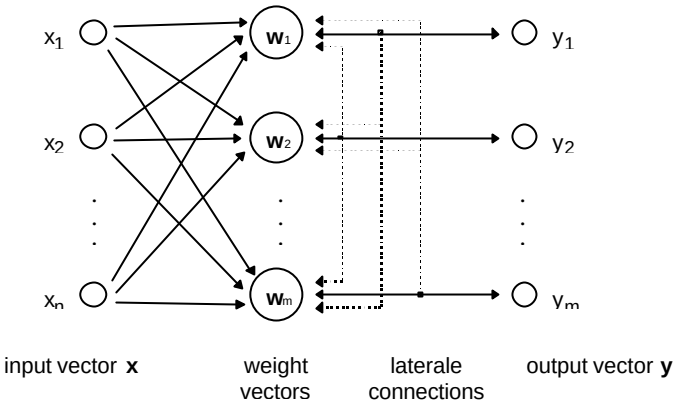

Fig. 4.1 *The neural network model*

Now, let us introduce an objective function to obtain the learning rules to implement the conditions (2.9) for the covariance coefficients

$$c_{ij} = \langle y_i\, y_j \rangle = \begin{cases} 0 & i \neq j \quad (\textit{decorrelation}) \\ \dfrac{P_Y}{m} & i = j \quad (\textit{normalization}) \end{cases}$$

For making a neural network learn the desired optimum, one could think of using the Lagrange function we used in section 2.1 to design the rules for the weights as a gradient descent on it, see e.g. [14]. This approach is not valid. First, it assumes that a Lagrange function has a maximum in all parameters. This is generally not true: consider for instance a Lagrange function $L(a,\lambda)=R(a)+\lambda c(a)$. Apparently, when λ is increased $L(a,\lambda)$ tends to infinity and has no maximum. Further on, the condition $\partial L/\partial a = 0$ indicates an extremum and not necessarily a maximum.

Therefore, we chose here another approach by designing an objective function for the network performance which is minimized by the learning process, not using the Lagrange function.

Using the abbreviation $P = P_Y/m$ we choose the following objective function

$$R(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m) = \underbrace{\frac{1}{4}\sum_{i=1}^{m}\sum_{j=1,\, j\neq i}^{m}\langle y_i\, y_j\rangle^2}_{R_1} + \beta\underbrace{\frac{1}{4}\sum_{i=1}^{m}\left(P - \langle y_i^2\rangle\right)^2}_{R_2} \qquad (4.2)$$

which is composed by two terms R_1 and R_2. The first term becomes zero only when all crosscorrelation terms c_{ij} are zero while the second term only becomes zero when all variances c_{ii} of the neurons become equal.

Now we let the weights of this feedforward network learn by the simple gradient descent learning rule

$$\mathbf{w}_k(t) = \mathbf{w}_k(t-1) - \gamma(t)\,\nabla_{w^k}\, R\left(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m\right) \qquad k = 1, 2, \ldots, m$$

with the learning rate γ and the Nabla-operator ∇ for the gradient.

With the gradient we can directly compute the deterministic learning rule for the k-th neuron

$$\mathbf{w}_k(t) = \mathbf{w}_k(t-1) - \gamma(t)\left[\underbrace{\sum_{i=1,i\neq k}^{m}\langle y_k\, y_i\rangle\langle x\, y_i\rangle}_{\textit{decorrelation}} + \underbrace{\beta\left(\langle y_k^2\rangle - P\right)\langle x\, y_k\rangle}_{\textit{normalization}}\right] \quad (4.3)$$

Introducing *lateral coupling weights* which are often observed in biological nervous circuitry

$$u_{ij} = -\langle y_i y_j\rangle \quad \textit{lateral inhibition} \quad (4.4)$$

between the neurons for the learning process (see e.g. [3]) we finally get as the deterministic learning rule

$$\mathbf{w}_k(t) = \mathbf{w}_k(t-1) + \gamma(t)\left[\sum_{i=1,i\neq k}^{m} u_{ki}\langle \mathbf{x}\, y_i\rangle + \beta(P - u_{kk})\langle \mathbf{x}\, y_k\rangle\right] \quad (4.5)$$

and the corresponding stochastic rule

$$\mathbf{w}_k(t) = \mathbf{w}_k(t-1) + \gamma(t)\,\mathbf{x}\left[\sum_{i=1,i\neq k}^{m} u_{ki} y_i + \beta(P - u_{kk})\, y_k\right] \quad (4.6)$$

The lateral inhibition weights should also be updated and reflect an average of the most recent patterns. Please note that the standard stochastic approximation approach yields some problems in this case because the distribution of the y is not stationary; they are subject for change of the weights. Therefore, the learning rate $\gamma(t)$ should not be chosen as $1/t$ which is normally a good compromise between the influence of the sample history and the present samples (see [13]), but otherwise, e.g. by decreasing it each time the sign of the gradient changes, indicating an overshooting step.

It can be shown that all sequential gradient learning rules which use limited objective functions principally lead to convergence, see appendix B-1. However, for discrete time steps in the stochastic case the convergence can be disturbed and limits for the parameters $\gamma(t)$ and β have to be established additionally, see e.g. [5]. However, this is not done here.

Now, assuming convergence we still have to ask whether the system will converge to the desired state implementing the two objectives at the same time. Intentionally, when the learning process stops the leaming goal is reached and the value of the objective function should be zero. For given learning rules, this is neither obvious nor trivial. Therefore, in appendix B-2 it is shown that for the learning equations (4.5) above the goal of the learning process, the state where the gradient is zero, is reached when the weight vectors become basis vectors which satisfy our demands for noise immunity of eqs. (2.10) and (2.11) i.e. data orthonormalization.

The data orthonormalization network defined by Fig. 4.1 and the activation and learning rule differs greatly from the ones introduced by Silva and Almeida [16] and Plumbley [14]. Contrary to Silva and Almeida, we have no biologically implausible activation backwards through the weights, changing the input signals according to the neuronal processing, but here we have a feedforward network, complemented by lateral inhibition lines and directly deduced by the theoretical demands.

The difference to the network of Plumbley is more subtle: his network is a pure lateral feedback network and does not contain feedforward weights. Thus, after convergence the lateral inhibition weights of our network are decreasing to zero, the main information is held in the feedforward weights. In his network, the feedforward weights are weighted equally by one and the inhibition weights will contain all the information after convergence.

Depending on the application, other demands (cf. section 3) can be combined with one of the two networks, implementing these demands by either feedforward or feedback activity lines. Thus, the two networks are suitable for different situations and purposes.

5 Comparative simulation results for image encoding

First, let us compare the performance of our orthonormalization algorithm for image encoding with the one of Silva and Almeida which is known for their fast convergence. As input we chose an synthetic image according to the statistic model proposed by Habibi and Wintz [11] by

$$C(x_1,x_1',x_2,x_2') = e^{-\alpha|x_1-x_1'|-\beta|x_2-x_2'|}$$

which describes the 2-dim image pixel correlation between two pixels at the image coordinates **x** and **x'.** For α=0.125 and β=0.249 (which corresponds to a moderate contrastive image) we constructed the correlation matrix for the pixel input vector, formed by concatenating all N rows of N pixels to an n = N×N dimensional vector.

As algorithms, we compare the deterministic learning rule of Silva and Almeida, called here "SAnet", which turned out to be faster in our simulations than the algorithm of Plumbley,

$$\mathbf{w}_k(t) = \big(1+\gamma(t)\big)\mathbf{w}_k(t-1) - \gamma(t)\sum_{i=1}^{n}\big(\mathbf{w}_k^T\mathbf{C}_{XX}\mathbf{w}_i\big)\mathbf{w}_i$$

$$= \mathbf{w}_k(t-1) - \gamma(t)\left[\sum_{i=1,i\neq k}^{n}\big(\mathbf{w}_k^T\mathbf{C}_{XX}\mathbf{w}_i\big)\mathbf{w}_i + \big(\mathbf{w}_k^T\mathbf{C}_{XX}\mathbf{w}_k - 1\big)\mathbf{w}_k\right] \qquad (5.1)$$

with the learning rule of our model ("RipNet")

$$\mathbf{w}_k(t) = \mathbf{w}_k(t\text{-}1) - \gamma(t)\left[\sum_{i=1,i\neq k}^{n}\big(\mathbf{w}_k^T\mathbf{C}_{XX}\mathbf{w}_i\big)\mathbf{C}_{XX}\mathbf{w}_i + \big(\mathbf{w}_k^T\mathbf{C}_{XX}\mathbf{w}_k - 1\big)\mathbf{C}_{XX}\mathbf{w}_k\right] \qquad (5.2)$$

Let us compare the expected performance of the two models for the input statistics specified above for P = 1 and k = 1, 2, …, n. Initially, the weight matrix was set to the unity matrix **W**=**I**. As performance measure we chose the cost function defined in eq.(4.2). Since the „best choice" for the initial parameter of the two algorithms depends heavily on the input statistics, we chose the initial learning rate as γ=0.05 for both models. The weight vector update was done sequentially to ensure a proper gradient descend. In Fig. 5.1 a objective function time course in a typical simulation run is shown for the SAnet.

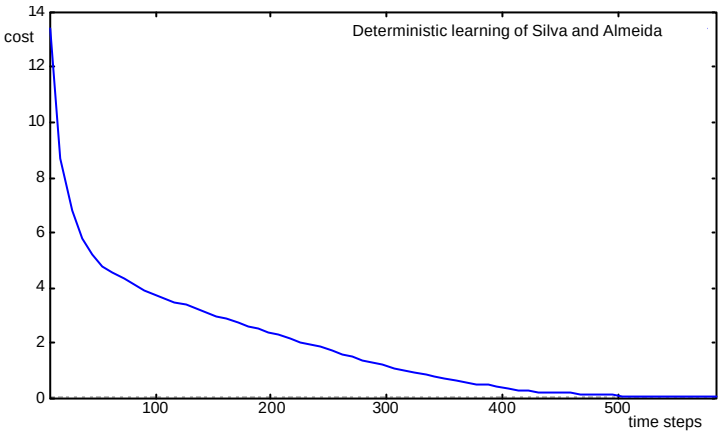

Fig. 5.1 *The objective function development of the SAnet*

Conversely, in Fig. 5.2 the convergence of the RipNet model is shown.

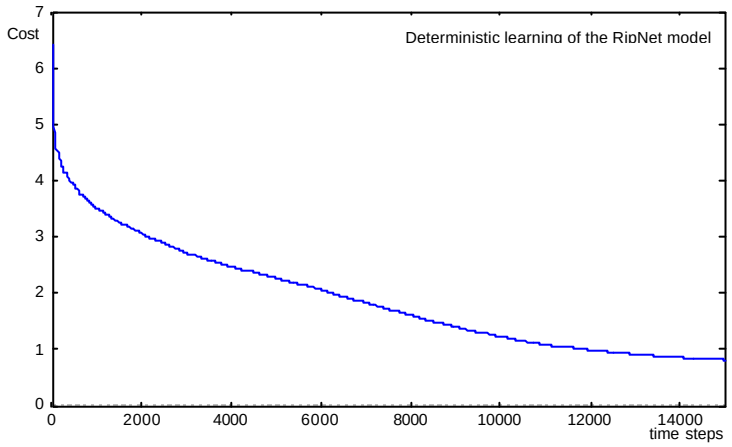

Fig. 5.2 *The objective function development of the RipNet model.*

We can see that under the same conditions for the example input statistics the model of Silva and Almeida performs better than ours. The two models differ in the learning rules just by an factor $\mathbf{C}_{xx}$ in the decorrelation and the normalization term. Additional simulations confirmed that the expectation term $\mathbf{C}_{xx}$ slows down the convergence a lot. So, by dropping the $\mathbf{C}_{xx}$ term in the RipNet model at the normalization only, let us define another model (”RipNet2”) which contains an additional term $\mathbf{C}_{XX}$ at the decorrelation component compared to the SAnet.

$$\mathbf{w}_k(t) = \mathbf{w}_k(t-1) - \gamma(t)\left[\sum_{i=1, i\neq k}^{n}\left(\mathbf{w}_k^T\mathbf{C}_{XX}\mathbf{w}_i\right)\mathbf{C}_{XX}\mathbf{w}_i + \left(\mathbf{w}_k^T\mathbf{C}_{XX}\mathbf{w}_k - 1\right)\mathbf{w}_k\right] \quad (5.3)$$

In Fig. 5.3 we see the corresponding performance of this algorithm for the same parameters.

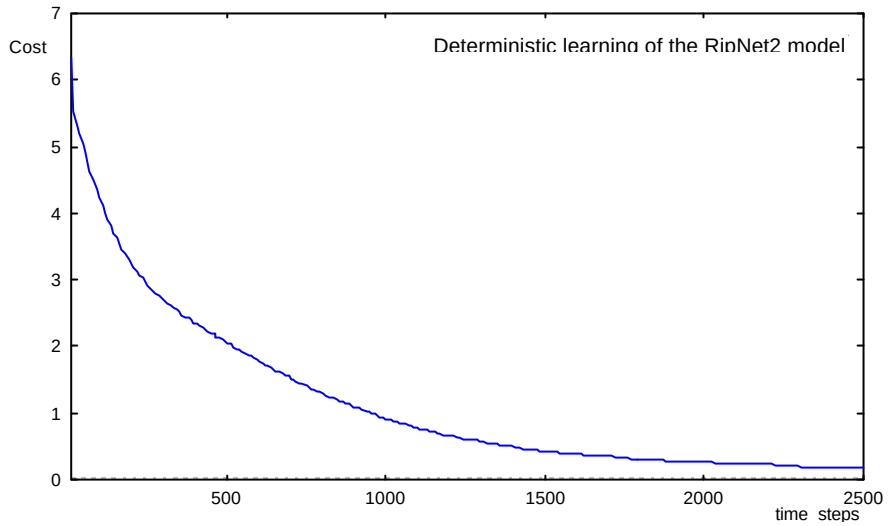

Fig. 5.3 *The objective function development of the RipNet2 Model*

As we expected, under these conditions the RipNet2 model has a convergence performance between the two others. Now, does this mean that our model that is directly derived from theoretic considerations is generally slower for the same input and learning rate than the heuristic model of Silva and Almeida?

First, we did not used specially designed, application-dependent „optimal“ parameters for each algorithm and each input pattern set. So, our simulations gives us only trends but no absolute rankings. Second, let us regard the stochastic versions of the deterministic algorithms which are much more important in an unknown environment. Here, we replace the deterministic term in the learning rules by the averaged, measured output correlations

$$\mathbf{w}_i^T \mathbf{C}_{XX} \mathbf{w}_j = \mathbf{w}_i^T \left\langle \mathbf{x}\mathbf{x}^T \right\rangle \mathbf{w}_j = \left\langle y_i\, y_j \right\rangle$$

and the averaged input term by the non-averaged stochastic terms

$$\mathbf{C}_{XX}\mathbf{w}_i = \left\langle \mathbf{x}\mathbf{x}^T \right\rangle \mathbf{w}_i = \left\langle y_i \mathbf{x} \right\rangle \rightarrow y_i \mathbf{x}$$

By this, the expectation value of the learning rules, i.e. the learning goal, remains the same.

For the simulation, we initialized again the weight matrix by $\mathbf{W}=\mathbf{I}$ and chose $\gamma=0.08$. To ensure the convergence of the stochastic case, we decreased the learning rate by 0.004 after each 20 steps. For n=5 units we generated 20 stochastic input vectors containing 5 independent, Gaussian distributed, centered components with different variance each.

Now, we compare the convergence of the resulting stochastic learning algorithms by typical sample runs of the cost developments. First, the performance of the stochastic learning rule (5.1) is shown in Fig.

5.4. We can see that contrary to the „experience gathering“ the expectation terms do not speed up the convergence. In reverse, the convergence is much faster now.

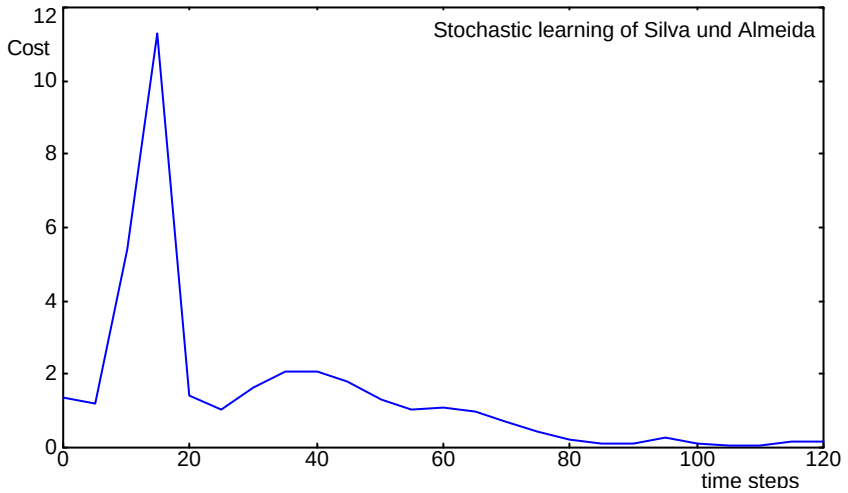

Fig. 5.4 *The objective function of a typical simulation run of the stochastic SAnet*

The corresponding time course of the stochastic RipNet model of eq. (5.2) shows the same proportions for the same input:

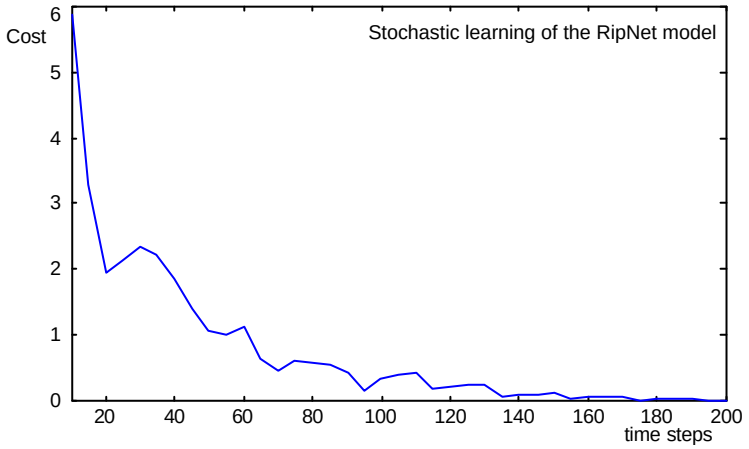

Fig. 5.5 *The objective function of a typical simulation run of the stochastic RipNet*

Now, the difference between the two models is no longer of practical importance: the convergence of both is much faster than in the deterministic case and is nearly the same. This includes also the case of the RipNet2 of eq. (5.3) model that is not shown here.

There is another interesting fact to notice: Comparing Fig. 5.1 and Fig. 5.2 with Fig. 5.4 and Fig. 5.5 we see that the cost, i.e. the objective function does not decrease monotonically any more. This is due to the

stochastic nature of the algorithm; the stochastic approximation properties of the heuristic network are very irregular while our network performs in the average just the sequential gradient descent.

6 Discussion and conclusion

In this paper we deduced the conditions for a transform which „immunizes“ parallel channels against noise influence before the signals are used in later stages and which conserves the maximal information. It shows up that the signals have to be decorrelated and normalized by the filter which corresponds for the case of one channel to the classical result of Shannon.

In the application of image encoding the proposed methodology shows good results and constitutes an efficient approach for noise suppression.

Furthermore, by a corresponding objective function we deduced the deterministic and stochastic learning rules for a neural network that implements data orthonormalization. In comparison with other already existing normalization networks it complements the one of Plumbley by a complementary architecture: it stores the information in the feedforward lines and not in the lateral inhibition ones. Compared to the heuristic network of Silva and Almeida our network shows a slower convergence speed in the deterministic case, but approximately the same in sample stochastic cases.

Additionally, by our systematic canonical derivation its convergence proportions are very regular. Contrary to the one of Silva and Almeida it is based on an objective function and can therefore serve as a building block for further enlarged objectives, i.e. objective functions containing additional terms, which can not be handled by scaled PCA network solutions for orthonormalization.

Certainly, there are still many questions open for the algorithms presented here. For instance, the optimal initial setting of the parameters depending on the input statistics, the global convergence behavior of the different algorithms and their restrictions on the input pattern range, the optimal parameter regime to be used for the discrete algorithm in the case of sequential and parallel update. All this is left for future research.

Acknowledgment

We want to thank Björn Arlt for providing us with the image normalization simulation data.

References

[1] S.Amari, A. Cichocki, H. Yang: *A New Learning Algorithm for Blind Signal Separation*; in: Advances in Neural Information Processing Systems 8, Touretzky, Mozer, Hasselmo (Eds.), MIT Press, pp.757-763, 1996
and available by http://www.bip.riken.go.jp/irl/hhy/hhy/acyNIPS95.ps.Z

[2] Bodewig: *Matrix Calculus*; North Holland, Amsterdam 1956

[3] R. Brause: *A Symmetrical Lateral Inhibited Network for PCA and Feature Decorrelation*; Proc. Int. Conf. Art. Neural Networks ICANN-93, Springer Verlag, pp. 486-489

[4] R. Brause: *Transform Coding by Lateral Inhibited Neural Nets;* Proc. IEEE Tools for Art. Intell. TAI-93

[5] R. Brause: *Sensor encoding with Cellular Neural Networks*; Neural Networks, Vol.9, No.1, pp.99-120, (1996)

[6] I. N. Bronstejn, K. A. Semendyayev: *Handbook of Mathematics*; Springer-Verlag, New York 1997

[7] Burel, *Blind Separation of Sources: A Nonlinear Neural Algorithm;* Neural Networks, Vol. 5, pp.937-947 (1992)

[8] P. Comon: *Independent Component Analysis - a new concept?*, Signal Processing, vol.36, pp. 287-314, 1994

[9] Th. Cover, J. Thomas: *Elements of Information Theory*; Wiley 1991

[10]G. Deco, D. Obradovic: *An Information-Theoretic Approach to Neural Computing*; Springer Verlag 1996

[11]A. Habibi, P. A. Wintz: *Image Coding by Linear Transformation and Block Quantization*; IEEE Transactions on Communication Technology, Vol. COM-19, No. 1, pp. 50-62, February 1971

[12]A. Papoulis: *Probability, Random Variables, and Stochastic Processes*; McGraw-Hill, International Edition 1991

[13]E. Pfaffelhuber, P. S. Damle: Learning and Imprinting in Stationary and Non-stationary Environments; Kybernetik Vol.13, pp.229-237 (1973)

[14]M. D. Plumbley: *Efficient Information Transfer and Anti-Hebbian Neural Networks*; Neural Networks, Vol. 6, pp. 823-833, Pergamon Press Ltd. 1993

[15]C. E. Shannon: *Communication in the Presence of Noise*; Proceedings of the IRE, Vol. 37, pp. 10-21, January 1949

[16]F. M. Silva, L. Almeida: *A distributed solution for data orthonormalization*; Artificial Neural Networks, T. Kohonen, K. Mäkisara, O. Simula, J. Kangas (Editors), Elsevier Science Publishers B.V. (North-Holland), pp. 943-948, 1991

[17]G. Wallace: The JPEG Still Picture Compression Standard; Comm. of the ACM, Vol.34, No.4, pp.31-44, April 1991

Appendix A
The information of multichannel gaussian sources

For the convenience of the reader we present here the multidimensional extension of the computations presented in [15].

Theorem A: Let X be a n-dimensional, normally distribute random variable with the probability density function

$$p(x_1, x_2, \ldots x_n) = \frac{1}{\sqrt{(2\pi)^n}\sqrt{\det \mathbf{C}_{XX}}}\; e^{\underbrace{-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} C_{XX}^{-1}[i,j]\, x_i\, x_j}_{=\mathbf{x}^T \mathbf{C}_{XX}^{-1}\,\mathbf{x}}},$$

with the correlation matrix $\mathbf{C}_{XX}$

$$\mathbf{C}_{XX}[i,j] = \langle x_i\, x_j \rangle = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\ldots\int_{-\infty}^{\infty} x_i\, x_j\, p(x_1, x_2, \ldots, x_n)\, dx_1\, dx_2 \ldots dx_n\ .$$

Then, the average information or entropy of X is given by

$$H(X) = \ln\sqrt{(2\pi e)^n \det \mathbf{C}_{XX}}\ . \qquad \text{(A.1)}$$

and the transinformation for the noisy channels of eq. (2.5) becomes

$$H(Y;Z) = \frac{1}{2}\ln\det\left(\frac{1}{P_\Phi} C_{YY} + I\right) \qquad \text{(A.2)}$$

Proof:

Equation (A.1) is a standard result of information theory, see e.g. [9] Theorem 9.6.5.
Therefore, we get for eq. (2.5)

$$\begin{aligned} H(Y;Z) &= H(Z) - H(\Phi) \\ &= \ln\sqrt{(2\pi e)^n \det C_{ZZ}} - \ln\sqrt{(2\pi e)^n \det C_{\Phi\Phi}} \\ &= \ln\sqrt{\det C_{ZZ}} - \ln\sqrt{\det C_{\Phi\Phi}} = \frac{1}{2}\left[\ln\det C_{ZZ} + \ln\det C_{\Phi\Phi}^{-1}\right] \\ &= \frac{1}{2}\ln\left(\det C_{ZZ}\, \det C_{\Phi\Phi}^{-1}\right) = \frac{1}{2}\ln\det\left(\left[C_{YY} + C_{\Phi\Phi}\right] C_{\Phi\Phi}^{-1}\right) \\ &= \frac{1}{2}\ln\det\left(C_{YY}\, C_{\Phi\Phi}^{-1} + I\right) \end{aligned}$$

and for equivariant noise on all channels with $C_{\Phi\Phi}^{-1} = \frac{1}{P_\Phi} I$

$$= \frac{1}{2} \ln \det \left(C_{YY} \left[\frac{1}{P_\Phi} I \right] + I \right)$$

$$= \frac{1}{2} \ln \det \left(\frac{1}{P_\Phi} C_{YY} + I \right)$$

q.e.d.

Appendix B The convergence

Theorem B-1 *The convergence proof*

If we change for the objective function

$$R(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m) = \underbrace{\frac{1}{4}\sum_{i=1}^{m}\sum_{j=1, j\neq i}^{m}\langle y_i\, y_j\rangle^2}_{R_1} + \beta\underbrace{\frac{1}{4}\sum_{i=1}^{m}\left(P - \langle y_i^2\rangle\right)^2}_{R_2}$$

the weight parameters sequentially that for each time step the learning equation is of the form of a gradient descend, i.e.

$$\mathbf{w}_k(t) = \mathbf{w}_k(t-1) - \gamma(t)\nabla_{\mathbf{w}_k} R(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m)$$

then the weights will be changed such that the objective function R(**w**) becomes the global minimum.

Proof

If we can show that the function R decreases at each time step and has a global lower limit it has the properties of a Ljapunov function. According to [6] then the convergence to a minimum is given.

For the time development of the objective function R we know that

$$\frac{dR}{dt} = \sum_{i=1}^{m}\frac{\partial R}{\partial \mathbf{w}_i}\frac{\partial \mathbf{w}_i}{\partial t}$$

With the gradient descend we have

$$\frac{\partial \mathbf{w}_k}{\partial t} = -\gamma(t)\frac{\partial R}{\partial \mathbf{w}_k} \qquad k = 1, 2, \ldots, m.$$

Combining the two equations gives us

$$\frac{dR}{dt} = \sum_{i=1}^{m}\frac{\partial R}{\partial \mathbf{w}_i}\left(-\gamma(t)\frac{\partial R}{\partial \mathbf{w}_i}\right) = -\gamma(t)\sum_{i=1}^{m}\left(\frac{\partial R}{\partial \mathbf{w}_i}\right)^2 \leq 0\,.$$

This means that the objective function R decreases monotonously at each time step t. Additionally, R has a minimal limit, because R_1 and R_2 are sums of squares which has the lower limit 0. Therefore, all conditions for a Ljapunov function are fulfilled and the learning equation supplies us with the solutions the minimum of the objective function.

q.e.d.

Theorem B-2 *The convergence goal*

With the objective function

$$R(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m) = \underbrace{\frac{1}{4}\sum_{i=1}^{m}\sum_{j=1, j\neq i}^{m}\langle y_i\, y_j\rangle^2}_{R_1} + \beta\underbrace{\frac{1}{4}\sum_{i=1}^{m}\left(P - \langle y_i^2\rangle\right)^2}_{R_2}$$

we get as all solutions of the equation system

$$\begin{cases} \nabla_{w_1} R(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m) = 0 \\ \nabla_{w_2} R(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m) = 0 \\ \quad\vdots \\ \nabla_{w_m} R(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m) = 0 \end{cases} \quad m \le n \qquad \text{(B.1)}$$

the weight vectors $\mathbf{w}_i$ in the eigenvector base $\mathbf{e}_r$ of the correlation matrix $\mathbf{C}_{XX}$

$$\mathbf{w}_i = \sum_{r=1}^{n} a_{ri}\, \mathbf{e}_r \qquad i = 1, 2, \ldots, m \qquad \text{(B.2)}$$

with the conditions

$$\sum_{r=1}^{n} a_{ri}^2 = P \quad \text{and} \quad \sum_{r=1}^{n} a_{ri}\, a_{rj} = 0 \qquad i, j = 1, 2, \ldots, m;\ i \neq j \qquad \text{(B.3)}$$

for the coordinates in the orthogonal eigenvector base $\{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n\}$ of the input correlation matrix $\mathbf{C}_{XX}$ with different eigenvalues $\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$.
Additionally, for the norm of each eigenvector we have

$$|\mathbf{e}_r|^2 = 1/\lambda_r \qquad r = 1, 2, \ldots, n. \qquad \text{(B.4)}$$

Proof

Let us prove the theorem by showing that the conditions (B.3) for the coordinates in the eigenvector base (B.2) above satisfy the equation (B.1) and is therefore a valid solution for the minimization of the objective function.

The gradient of the objective function gives us (see eq. 3.3)

$$\nabla_{w_k} R(w^1, w^2, \ldots, w^m) = \sum_{i=1, i\neq k}^{m}\left(\mathbf{w}_k^T \mathbf{C}_{XX}\, \mathbf{w}_i\right)\left(\mathbf{C}_{XX}\, \mathbf{w}_i\right) - P\,\mathbf{C}_{XX}\, \mathbf{w}_k + \left(\mathbf{w}_k^T \mathbf{C}_{XX}\, \mathbf{w}_k\right)\left(\mathbf{C}_{XX}\, \mathbf{w}_k\right)$$

$$= \sum_{i=1, i\neq k}^{m}\left(\left(\sum_{r=1}^{n} a_{rk}\, \mathbf{e}_r^T\right)\mathbf{C}_{XX}\left(\sum_{s=1}^{n} a_{si}\, \mathbf{e}_s\right)\right)\left(\mathbf{C}_{XX}\left(\sum_{s=1}^{n} a_{si}\, \mathbf{e}_s\right)\right)$$

$$-P\,C_{XX}\left(\sum_{r=1}^{n} a_{rk}\,\mathbf{e}_r\right)+\left(\left(\sum_{q=1}^{n} a_{qk}\,\mathbf{e}_q^T\right)C_{XX}\left(\sum_{r=1}^{n} a_{rk}\,\mathbf{e}_r\right)\right)\left(C_{XX}\left(\sum_{r=1}^{n} a_{rk}\,\mathbf{e}_r\right)\right)$$

$$=\sum_{i=1,i\neq k}^{m}\left(\left(\sum_{r=1}^{n} a_{rk}\,\mathbf{e}_r^T\right)\left(\sum_{s=1}^{n} a_{si}\,\lambda_s\,\mathbf{e}_s\right)\right)\left(\sum_{s=1}^{n} a_{si}\,\lambda_s\,\mathbf{e}_s\right)$$

$$-P\left(\sum_{r=1}^{n} a_{rk}\,\lambda_r\,\mathbf{e}_r\right)+\left(\left(\sum_{q=1}^{n} a_{qk}\,\mathbf{e}_q^T\right)\left(\sum_{r=1}^{n} a_{rk}\,\lambda_r\,\mathbf{e}_r\right)\right)\left(\sum_{r=1}^{n} a_{rk}\,\lambda_r\mathbf{e}_r\right)$$

$$=\sum_{i=1,i\neq k}^{m}\left(\sum_{r=1}^{n}\sum_{s=1}^{n} a_{rk}\,a_{si}\,\lambda_s\underbrace{\mathbf{e}_r^T\mathbf{e}_s}_{=0\;r\neq s}\right)\left(\sum_{s=1}^{n} a_{si}\,\lambda_s\,\mathbf{e}_s\right)$$

$$-P\left(\sum_{r=1}^{n} a_{rk}\,\lambda_r\,\mathbf{e}_r\right)+\left(\sum_{q=1}^{n}\sum_{r=1}^{n} a_{qk}\,a_{rk}\,\lambda_r\underbrace{\mathbf{e}_q^T\mathbf{e}_r}_{=0\;q\neq r}\right)\left(\sum_{r=1}^{n} a_{rk}\,\lambda_r\mathbf{e}_r\right)$$

which becomes under the condition (B.4)

$$=\sum_{i=1,i\neq k}^{m}\left(\sum_{r=1}^{n} a_{rk}\,a_{ri}\,\lambda_r\underbrace{|\mathbf{e}_r|^2}_{=1/\lambda_r}\right)\left(\sum_{s=1}^{n} a_{si}\,\lambda_s\,\mathbf{e}_s\right)$$

$$-P\left(\sum_{r=1}^{n} a_{rk}\,\lambda_r\,\mathbf{e}_r\right)+\left(\sum_{q=1}^{n} a_{qk}^2\,\lambda_q\underbrace{|\mathbf{e}_q|^2}_{=1/\lambda_q}\right)\left(\sum_{r=1}^{n} a_{rk}\,\lambda_r\,\mathbf{e}_r\right)$$

$$=\sum_{i=1,i\neq k}^{m}\underbrace{\left(\sum_{r=1}^{n} a_{rk}\,a_{ri}\right)}_{=0\;i\neq k}\left(\sum_{s=1}^{n} a_{si}\,\lambda_s\,\mathbf{e}_s\right)-P\left(\sum_{r=1}^{n} a_{rk}\,\lambda_r\,\mathbf{e}_r\right)+\underbrace{\left(\sum_{q=1}^{n} a_{qk}^2\right)}_{=P}\left(\sum_{r=1}^{n} a_{rk}\,\lambda_r\,\mathbf{e}_r\right)$$

$$=0\qquad k=1,2,\ldots,m.$$

For the *m* different equations we have at most *m* different solutions. Since the *m* linear independent weight vectors $\mathbf{w}_i$ with conditions (B.2) and (B.3) are valid solutions, they form a *m*-dim. solution space which is sufficient for the maximal *m*-dim. space span by the *m* equations with *m* variables of (B.1). There can be no more valid solutions which means that we have found them all.

q.e.d.

Appendix C The minimal reconstruction error

Theorem C-1 *The reconstruction error*

Let $\mathbf{y} = \mathbf{W}\mathbf{x}$ be a linear transformation by a matrix $\mathbf{W}$ such that the transformation performs a data orthonormalization. Since we have dim(**x**)=rank(**W**)=n there exists an inverse transformation $\mathbf{W}^{-1}$ and each input vector $\mathbf{x} \in X$ can be decomposed by the inverse transformation (2.14).
Then, to each **x** there exists an approximate vector **x'** defined by

$$\mathbf{x}' = \sum_{i=1}^{m} y_i \left((1/P)\, \mathbf{C}_{XX} \mathbf{w}_i \right) \qquad m \le n$$

For the approximation, all terms in the sum for $m+1 \le i \le n$ are neglected.
The mean squared error of this approximation is given by

$$\varepsilon_s^2 = \left\langle |x - x'|^2 \right\rangle = (1/P) \sum_{r=1}^{n} \lambda_r \sum_{i=m+1}^{n} a_{ri}^2$$

.

Proof

We have

$$\varepsilon_s^2 = \left\langle |x - x'|^2 \right\rangle = \left\langle \left| \sum_{i=1}^{n} y_i \left((1/P)\, \mathbf{C}_{XX}\, \mathbf{w}_i \right) - \sum_{i=1}^{m} y_i \left((1/P) \mathbf{C}_{XX}\, \mathbf{w}_i \right) \right|^2 \right\rangle$$

$$= \left\langle \left| \sum_{i=m+1}^{n} y_i \left((1/P) \mathbf{C}_{XX}\, \mathbf{w}_i \right) \right|^2 \right\rangle$$

$$= \left\langle \left(\sum_{i=m+1}^{n} y_i\, (1/P)\, \mathbf{C}_{XX}\, \mathbf{w}_i \right)^T \left(\sum_{j=m+1}^{n} y_j (1/P) \mathbf{C}_{XX}\, \mathbf{w}_j \right) \right\rangle$$

$$= \sum_{i=m+1}^{n} \sum_{j=m+1}^{n} \underbrace{\left\langle y_i\, y_j \right\rangle}_{=0 \; i \ne j} \left(1/P^2 \right) \left(\mathbf{C}_{XX}\, \mathbf{w}_i \right)^T \left(\mathbf{C}_{XX}\, \mathbf{w}_j \right)$$

With $\left\langle y_i\, y_i \right\rangle = P$ and the eigenvector decomposition (B.2) of $\mathbf{w}_i$ this becomes

$$= (1/P) \sum_{i=m+1}^{n} \left(\mathbf{C}_{XX}\, \mathbf{w}_i \right)^T \left(\mathbf{C}_{XX}\, \mathbf{w}_i \right) = (1/P) \sum_{i=m+1}^{n} \left(\mathbf{C}_{XX} \sum_{r=1}^{n} a_{ri}\, \mathbf{e}_r \right)^T \left(\mathbf{C}_{XX} \sum_{s=1}^{n} a_{si}\, \mathbf{e}_s \right)$$

$$= (1/P) \sum_{i=m+1}^{n} \left(\sum_{r=1}^{n} a_{ri}\, \lambda_r\, \mathbf{e}_r \right)^T \left(\sum_{s=1}^{n} a_{si}\, \lambda_s \mathbf{e}_s \right)$$

$$= (1/P) \sum_{i=m+1}^{n} \left(\sum_{r=1}^{n} \sum_{s=1}^{n} a_{ri}\, a_{si}\, \lambda_r\, \lambda_s \underbrace{\left(e^r, e^s\right)}_{\delta_{rs}/\lambda_s} \right) = (1/P) \sum_{i=m+1}^{n} \sum_{r=1}^{n} a_{ri}^2\, \lambda_r$$

$$= (1/P) \sum_{r=1}^{n} \lambda_r \sum_{i=m+1}^{n} a_{ri}^2 \qquad \text{(C.1)}$$

q.e.d.

Theorem C-2 *The minimum of the error*

Let $\mathbf{y} = \mathbf{W}\,\mathbf{x}$ be a linear transformation with a matrix $\mathbf{W}$ such that the transformation performs a data orthonormalization. The eigenvectors should be ordered according to their size and be scaled such that their norm fulfills the following condition

$$\left|\mathbf{e}_i\right|^2 = \frac{1}{\lambda_i} \qquad i = 1, 2, \ldots, n.$$

Then the mean squared reconstruction error

$$\varepsilon_s^2 = \left\langle \left|\mathbf{x} - \mathbf{x}'\right|^2 \right\rangle$$

becomes minimal when the rows of $\mathbf{W}$ become

$$\mathbf{w}_i^T = \sqrt{P}\, \mathbf{e}_i^T \qquad i = 1, 2, \ldots, n; \quad P > 0$$

with $\{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n\}$ being the eigenvectors of the Correlation matrix $\mathbf{C}_{XX}$ with the different eigenvalues $\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ and the error has the form

$$\varepsilon_s^2 = \sum_{r=m+1}^{n} \lambda_r \, .$$

Proof

Let $\mathbf{W}$ be an arbitrary transformation matrix such that it performs a linear transformation $\mathbf{y} = \mathbf{W}\mathbf{x}$ implementing a data orthonormalization. By theorem C-1 the mean squared error of the reconstruction is given by eq. (C.1)

$$\varepsilon_s^2 = (1/P) \sum_{r=1}^{n} \lambda_r \sum_{i=m+1}^{n} a_{ri}^2 \, .$$

To minimize this error, we are looking for the relative extrema of this function under the constraint of constant signal power

$$\sum_{s=1}^{n} a_{si}^2 = P \qquad i = m+1, \ldots, n$$

In order to solve this problem, we use a Lagrange function. The constraint becomes

$$g_i(a_{1i}, \ldots, a_{ni}) = \sum_{s=1}^{n} a_{si}^2 - P = 0 \qquad i = m+1, \ldots, n$$

and the function is

$$f\left(a_{1(m+1)}, \ldots, a_{nn}\right) = (1/P) \sum_{r=1}^{n} \lambda_r \sum_{i=m+1}^{n} a_{ri}^2$$

Using the (n-m) unknown Lagrange multipliers $\mu_{m+1}, \ldots, \mu_n$ the Lagrange-Funktion L becomes

$$L\left(a_{1(m+1)}, \ldots, a_{nn}, \mu_{m+1}, \ldots, \mu_n\right) = f\left(a_{1(m+1)}, \ldots, a_{nn}\right) + \sum_{i=m+1}^{n} \mu_i \, g_i(a_{1i}, \ldots, a_{ni})$$

For the unknown coefficients $\{a_{ri}\}$ *r = 1, 2, ¼, n; i = m+1, ¼, n* and $\{\mu_i\}$ *i = m+1, ¼, n* we have to solve the equation system

$$\begin{cases} g_i(a_{1i}, \ldots, a_{ni}) = 0 & i = m+1, \ldots, n \\ \dfrac{\partial L}{\partial a_{jk}}\left(a_{1(m+1)}, \ldots, a_{nn}, \mu_{m+1}, \ldots, \mu_n\right) = 0 & j = 1,2, \ldots, n;\ k = m+1, \ldots n \end{cases}$$

The equations $g_i(a_{1i}, \ldots, a_{ni}) = 0$ for i = m+1, …, n are fulfilled when the constraints are satisfied. What we really are interested in are the remaining equations to be solved

$$\begin{aligned} &\frac{\partial L}{\partial a_{jk}}\left(a_{1(m+1)}, \ldots, a_{nn}, \mu_{m+1}, \ldots, \mu_n\right) \\ &= \frac{\partial}{\partial a_{jk}}\left(\left((1/P)\sum_{r=1}^{n}\lambda_r \sum_{i=m+1}^{n} a_{ri}^2\right) + \left(\sum_{i=m+1}^{n}\mu_i \sum_{s=1}^{n} a_{si}^2 - P\right)\right) \\ &= \frac{\partial}{\partial a_{jk}}\left((1/P)\sum_{r=1}^{n}\lambda_r \sum_{i=m+1}^{n} a_{ri}^2\right) + \frac{\partial}{\partial a_{jk}}\left(\sum_{i=m+1}^{n}\mu_i \sum_{s=1}^{n} a_{si}^2 - P\right) \\ &= (1/P)\sum_{r=1}^{n}\lambda_r \underbrace{\left(\frac{\partial}{\partial a_{jk}} \sum_{i=m+1}^{n} a_{ri}^2\right)}_{=0\ a_{ri} \neq a_{jk}} + \sum_{i=m+1}^{n}\mu_i \underbrace{\left(\frac{\partial}{\partial a_{jk}} \sum_{s=1}^{n} a_{si}^2 - P\right)}_{=0\ a_{si} \neq a_{jk}} \end{aligned}$$

$$= (1/P)\lambda_j 2a_{jk} + \mu_k 2a_{jk} = 2a_{jk}\left((1/P)\lambda_j + \mu_k\right)$$

$= 0$ for j = 1,2,…,n; k = m+1,…, n.

This means that

$$a_{jk}\left((1/P)\lambda_j + \mu_k\right) = 0 \qquad j = 1,2,\ldots,n;\ k = m+1,\ldots, n$$

which is fulfilled by the conditions

$$a_{jk}=0 \text{ or } \left((1/P)\lambda_j + \mu_k\right) = 0$$

For a non-zero component $a_{jk} \neq 0$ we conclude $\mu_k = -(1/P)\lambda_j$.
Now, let us assume that there are *two* non-zero components

$$a_{rk} \neq 0 \quad \wedge \quad a_{sk} \neq 0 \qquad r, s = 1, 2, \ldots, n;\ r \neq s$$

We can conclude

$$\left.\begin{array}{l} a_{rk} \neq 0 \quad \Rightarrow \quad \mu_k = -(1/P)\lambda_r \\ a_{sk} \neq 0 \quad \Rightarrow \quad \mu_k = -(1/P)\lambda_s \end{array}\right\} \Rightarrow \lambda_r = \lambda_s \qquad r, s = 1, 2, \ldots, n;\ r \neq s$$

which contradicts the assumption that different eigenvectors of $\mathbf{C}_{XX}$ have different eigenvalues. Thus, the assumption is wrong: if a component a_{jk} of $\mathbf{w}_k$ is not zero all the other ones in $\mathbf{w}_k$ must be zero

$$\text{If} \quad a_{jk} \neq 0 \text{ for each } j = 1,2,\ldots, n \ \Rightarrow \ a_{rk} = 0 \ \text{ for } \ r = 1,2,\ldots, n;\ r \neq j.$$

i.e. for the extremum of the error the components in the eigenvector base

$$\left(a_{1k}, a_{2k}, \ldots, a_{nk}\right) \qquad k = m+1, \ldots, n$$

of the vector $\mathbf{w}_k$ must be zero except just one element. With the constraint for $g_k(.)$ we know that then this element has the value $\sqrt{P}$.
Additionally, with eq.(2.11) we know that for a complete orthonormal transform with *n* basis vectors we have

$$\sum_{r=1}^{n} a_{ri}\, a_{rj} = 0 \qquad i, j = 1, 2, \ldots, n;\ i \neq j$$

Since a_{ri} is zero (r=1..n, i=m+1,..n) except for one element (denoted by k) we know that

$$\sum_{r=1}^{n} a_{ri}\, a_{rj} = a_{rk}\, a_{rj} = \sqrt{P} a_{rj} = a_{rj} = 0 \qquad r, j = 1, 2, \ldots, n;\ i \neq j$$

Now, we know more: The coefficient a_{rj} (r=1..n, j=1,..n) is always zero except for one index *i* where it is non-specified. With (2.12) we also have for the complete orthonormal transformation

$$\sum_{r=1}^{n} a_{rj}^2 = P = a_{ri}^2 \qquad j = 1, 2, \ldots, n;$$

which gives us the complete picture: For a minimum error the coefficients in the scaled eigenvector system of all basis vectors $\mathbf{w}_i$ are zero except for just one component in each basis vector. This means that each base vector is completely aligned to just one eigenvector direction. Since the complete transform with m=n is invertible, the base vectors $\mathbf{w}_i$ are linear independent which means that the non-zero components have different indices.

Now we order the vectors $\mathbf{w}_i$ by the index of the non-zero component in the eigenvector base, i.e. re-number the indices such that a_{ii}^2=P is the non-zero element. Then each base vector is aligned to the eigenvector with the same index, i.e.

$$\mathbf{w}_i^T = \sqrt{P}\, \mathbf{e}_i^T \qquad i = 1, \ldots, n. \tag{C.2}$$

and we get

$$\varepsilon_s^2 = (1/P) \sum_{r=1}^{n} \lambda_r \sum_{i=m+1}^{n} a_{ri}^2 = (1/P) \sum_{r=1}^{n} \sum_{i=m+1}^{n} \lambda_r a_{ri}^2 \delta_{ri} = \sum_{r=m+1}^{n} \lambda_r \tag{C.3}$$

with the Kronecker symbol δ_{ij}.

The sum becomes minimal if we use only the *m-n* smallest eigenvalues i.e. neglect the eigenvectors with the smallest eigenvalues. This means that the transformation base vectors $\{\mathbf{w}_k\}$ should be the *m* eigenvectors with the biggest eigenvalues.

This concludes the proof.

7 The authors

Michael Rippl is European Managing Consultant at Microsoft Corp. where he heads the application development group. He received his diploma in Computer Science at the J.W.Goethe-University at Frankfort in 1995.

His current interests include (beside music, cats and sports cars) multi-media architectures, internet and middle ware technologies and application-oriented software architectures.

Rüdiger W. Brause studied physics and cybernetics at Tübingen where he received his diploma in physics. His PhD thesis treated the adaptive diagnosis of wafer-scale multiprocessor systems. After joining the fault-tolerant multiprocessor system project ATTEMPTO he changed to Frankfurt where he heads now a working group for adaptive information processing systems.

His current research interests are concentrated on encoding and adaptive, content-oriented information extraction in images, medical and financial applications.

Readers my write to

PD Dr. R. Brause,
J. W. Goethe-University
FB Informatik
60054 Frankfurt
Germany