The Minimum Entropy Network

Rüdiger W. Brause

Internal Report 1/92

Abstract

One of the most interesting domains of feedforward networks is the processing of sensor signals. There do exist some networks which extract most of the information by implementing the maximum entropy principle for Gaussian sources. This is done by transforming input patterns to the base of the eigenvectors of the input autocorrelation matrix with the biggest eigenvalues. The basic building block of such a transformation is the linear neuron, learning with the Oja learning rule.

Nevertheless, some researchers in pattern recognition theory claim that for pattern recognition and classification clustering transformations are needed which reduce the intra-class entropy. This leads to stable, reliable features and is implemented for Gaussian sources by a linear transformation using the eigenvectors with the *smallest* eigenvalues.

This paper states the problem and shows that the basic building block for this transformation can be implemented by a linear neuron using an Anti-Hebb rule and restricted weights even for non-centered input. The fixpoints of the transformation are computed and the stability of the desired solution is shown.

Additionally, the algorithm is given for an asymmetric network which computes the eigenvectors in the ascending order of their corresponding eigenvalues, the conditions for the convergence are computed and demonstrated by simulations.

The Minimum Entropy Network

Rüdiger W. Brause, J.W. Goethe-University, Germany

1. Introduction

For many purposes the necessary processing of sensor input signals is realized by using a system which implements the maximization of the transinformation from the input to the output of the system. For deterministic systems, this corresponds to the maximization of the output entropy (maximum entropy principle). In pattern recognition theory, it is well known that for Gaussian distributed sources this corresponds to the minimization of the mean square error of the output. For linear systems, this is done by a linear transformation to base of the eigenvectors of the autocorrelation matrix [Fuk72]. Furthermore, we can compress (encode) the input information by using only the base vectors (eigenvectors) with the biggest eigenvalues. Neglecting the ones with the smallest eigenvalues results in the smallest reconstruction error of the encoded input [Fuk72]. Generally, this approach can be used for sensor signal coding such as picture encoding, see e.g. [Jay84].

The neural network implementations of this approach use linear neurons, where each neural weight vector corresponds to one eigenvector. Examples of those architectures are the Oja subspace network [Oja89], the Sanger decomposition network [San89] and the lateral inhibition network of Rubner and Tavan [Rub89]. The two last mentioned networks decompose sequentially the input vector x, see figure 1.

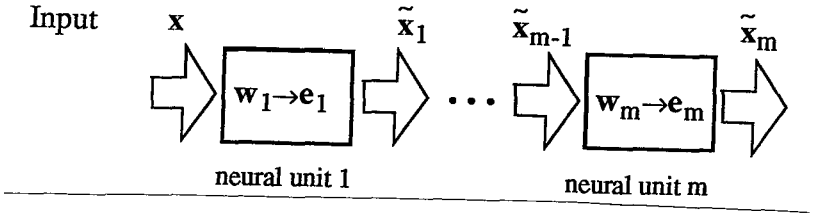

Fig.1 The sequential eigenvector decomposition by decomposition units

They use as a basic building block the linear correlation neuron which learns the input weights by a Hebb-rule, restricting the weights $w_1,..,w_n$. As Oja showed[Oja82], this learning rule let the weight vector of the neuron converge to the eigenvector of the expected autocorrelation matrix C of the input patterns x with the biggest eigenvalue λ_{max}:

$$\mathbf{w} \rightarrow \mathbf{e}^k \quad \text{with} \quad \lambda_k = \max_i \lambda_i \qquad \text{and} \quad \mathbf{C}\mathbf{e}^i = \lambda_i \mathbf{e}^i \qquad \mathbf{C} := \langle \mathbf{x}\mathbf{x}^T \rangle$$

2. The minimum entropy principle

The maximum entropy principle maximizes the entropy, i.e. for Gaussian sources it minimizes the quadratic error of the output coding. This aimes to minimize the reconstruction error for the input data from the encoded output.

In many applications, this is not the appropriate goal. If we want just to identify an object, we are not interested in the noisy representation of the object but in the code for the pure prototype of the object neglecting all variances. In the language of pattern recognition, all noisy instances of the object form a data point cloud (a cluster) around the prototype in the *n*-dimensional feature space. Here the goal of the transformation consists of projecting the cloud of data points onto the

prototype. This is done by removing some uncertainty from the data points: the entropy of the cluster is reduced. It was shown by Tou and Gonzales [Tou74], that for Gaussian distributed clusters with uniform variance the cluster entropy is maximally reduced by the linear transformation on the basis of the eigenvectors of the covariance matrix. Here the most reliable feature is given by the projection of the input to the eigenvector with the *smallest* eigenvalue. This necessity for clustering transformations motivates the question: Can we implement such a transformation also by a neural network ?

3.1 The minimum entropy neuron

The base of all three cited eigenvector decomposition networks consists of a neural unit learning the eigenvector of the input autocorrelation matrix with the biggest eigenvalue. In analyzing this approach we can derive the proper learning rule for the eigenvector with the *smallest* eigenvalue and prove the stability of the solution.

Let us assume an input $\mathbf{x}=(x_1,..,x_n)$ for one neuron. Traditionally, the input is weighted by the weights $\mathbf{w}=(w_1,...,w_n)$ and summed up to the activation z of the neuron

$$z(t) = \Sigma_i w_i x_i = \mathbf{w}^T\mathbf{x} \qquad\qquad y(t)=S(z) \tag{3.1}$$

which is expressed as the scalar product of $\mathbf{x}$ and the transpose of $\mathbf{w}$. Since we assume linear neurons, with the linear output function S(z) = z the output y(t) becomes z(t).
When we use m<n neurons, the resulting mean coding error is the mean output variance $\langle(y - \bar{y})^2\rangle$ [Fuk72] which becomes for centralized input $\bar{\mathbf{x}}:=\langle\mathbf{x}\rangle=\mathbf{0}$ (and therefore $\bar{y}:=\langle y\rangle=0$) the output intensity

$$f(\mathbf{w}) = \langle(y - \bar{y})^2\rangle = \langle y^2\rangle = \langle \mathbf{w}^T\mathbf{x}\mathbf{x}^T\mathbf{w}\rangle = \mathbf{w}^T\mathbf{C}\mathbf{w} \tag{3.2}$$

Since we are not interested in uniformly squeezing or expanding the pattern space, the volume should be conserved by the linear transformation defined in (3.1). Thus, we assume $\det(\mathbf{W})=1$ which is confirmed by the demand $|\mathbf{w}|=1$. This restriction of the weights is often used in learning systems to prevent the Hebbian learning rule from "blowing up" the weights.

Let us now investigate the necessary conditions for the local extrema of the objective function (3.2) with respect to the constrain $|\mathbf{w}|^2-1=0$. It is well known that the necessary conditions for the local extrema of a function using the Lagrange multiplier μ

$$L(w_1,...,w_n,\mu) := f(\mathbf{w}) + \mu(|\mathbf{w}|^2 - 1) = \mathbf{w}^T\mathbf{C}\mathbf{w} + \mu(\mathbf{w}^T\mathbf{w} - 1) \tag{3.3}$$

represent the desired conditions for the corresponding restricted objective function f(**w**)

$$\nabla_{\mathbf{w}} L(\mathbf{w},\mu) = 2\mathbf{C}\mathbf{w} + 2\mu\mathbf{w} = 0 \tag{3.4a}$$

$$\partial L(\mathbf{w},\mu)/\partial\mu = \Sigma_i w_i^2 - 1 = 0 \tag{3.4b}$$

The necessary extremum conditions (3.4a) provide as solutions the eigenvectors $\mathbf{e}^k$ of the expected autocorrelation matrix **C**

$$\mathbf{w}^* = \mathbf{e}^k \qquad \text{with} \qquad \mathbf{C}\mathbf{e}^k = \lambda_k\mathbf{e}^k, \quad \lambda_k:=-\mu_k \tag{3.5}$$

with the corresponding eigenvalues λ_k. In this case we have

$$f(\mathbf{w}^*) = \langle y^2\rangle = \mathbf{w}^{*T}\mathbf{C}\mathbf{w}^* = \lambda_k\mathbf{w}^{*2} = \lambda_k \tag{3.6}$$

Unfortunately, the approach with Lagrangian multipliers does not determine what kind of extrema

we do have. In appendix A it is shown by a different, more detailed approach that the fixpoint of the eigenvector with the maximal eigenvalue λ_{max} is a maximum, the eigenvector with the minimal eigenvalue λ_{min} a minimum. Beside these two unique all other fixpoints are unstable saddle-points. Thus, to reach the minimum we can use a simple gradient descend algorithm

$$\bar{\mathbf{w}}(t+1) = \mathbf{w}(t) - \gamma \operatorname{grad} f(\mathbf{w}) = \mathbf{w}(t) - \gamma \mathbf{C}\mathbf{w}(t) \qquad \textit{gradient descend} \qquad (3.7)$$

and $$\mathbf{w}(t+1) = \bar{\mathbf{w}}(t+1) / |\bar{\mathbf{w}}(t+1)| \qquad \textit{normalization}$$

The stochastic version of this algorithm is with $\mathbf{Cw}=\langle \mathbf{x}\mathbf{x}^T\mathbf{w}\rangle=\langle \mathbf{x}y\rangle$

$$\bar{\mathbf{w}}(t+1) = \mathbf{w}(t) - \gamma(t)\, \mathbf{x}(t)\, y(t) \qquad \textit{Anti-Hebb-rule} \qquad (3.8)$$

and $$\mathbf{w}(t+1) = \bar{\mathbf{w}}(t+1) / |\bar{\mathbf{w}}(t+1)| \qquad \textit{normalization}$$

If the learning rate $\gamma(t)$ satisfies all the convenient conditions for the stochastic approximation process (e.g. $\gamma(t):=1/t$), the convergence of the approximation process is guaranteed, see e.g. [Oja82]. If we replace the negative sign by the positive sign at (3.5) and (3.6), the gradient uphill climbing will provide us with the familiar Hebb-Rule for the maximal eigenvalue.

3.2. Convergence visualization example

For the visualization of the convergence process we choose an example which is not too low-dimensional (and therefore trivial) and can be shown satisfactory on a 2-dim sheet of paper.

Thus, we choose first the example of the 2-dim input patterns $\mathbf{x}^1=(1,1)$ and $\mathbf{x}^2=(0,1)$. These patterns have a autocorrelation matrix $\mathbf{C}$. Analytically, we can compute the eigenvectors $\mathbf{e}^1$ and $\mathbf{e}^2$ and the eigenvalues λ_1,λ_2

$$\mathbf{C}=\begin{pmatrix} 1 & 0.5 \\ 0.5 & 0.5 \end{pmatrix} \qquad \lambda_1=1.309, \quad \lambda_2=0.191 \qquad (3.9)$$

$$\mathbf{e}^1 = (0.851, 0.526), \quad \mathbf{e}^2 = (-0.526, 0.851)$$

Instead of representing $\mathbf{w}$ in the coordinates relative to the eigenvectors as in appendix A which implies the *a priori* knowledge of the eigenvectors and eigenvalues, let us transform $\mathbf{w}$ directly into its polar coordinates

$$\mathbf{w}=(w_1,w_2)^T = (|\mathbf{w}|\cos\alpha, |\mathbf{w}|\sin\alpha)^T = |\mathbf{w}|\,(\cos\alpha, \sin\alpha)^T$$

The objective function becomes with the constrain $|\mathbf{w}|^2=1$

$$f(\mathbf{w}) = \mathbf{w}^T\mathbf{C}\mathbf{w} = (\cos\alpha, \sin\alpha)\,\mathbf{C}\,(\cos\alpha, \sin\alpha)^T = 0.5 + 0.5\cos^2\alpha + \cos\alpha\sin\alpha \qquad (3.10)$$

In figure 2 the objective function is plotted as argument of the angle α. As we can see, the maximum of $f(\mathbf{w}^*)=\lambda_1$ is taken at $\alpha_1^*=0.553$ and $\alpha_1^*=3.69$, the minimum of $f(\mathbf{w}^*)=\lambda_2$ at

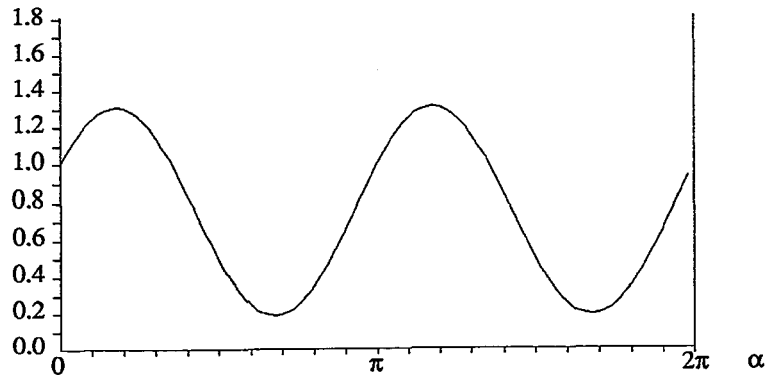

Fig. 2 The extrema of the objective function

$\alpha_2^*=2.12=\alpha_1^*+\pi/2$ and at $\alpha_2^*=5.26$.
The convergence process can be visualized by a needle-field picture. Here we plot for the field of 20x20=400 possible values of **w** (sall dots in figures 3 and 4) the change in **w** by a small needle which is proportional to the length $|\Delta \mathbf{w}|=|\mathbf{w}(t+1)-\mathbf{w}(t)|$ and points in the direction of $\Delta \mathbf{w}$.
In figure 3 the effect of the deterministic algorithm (3.7) is shown.

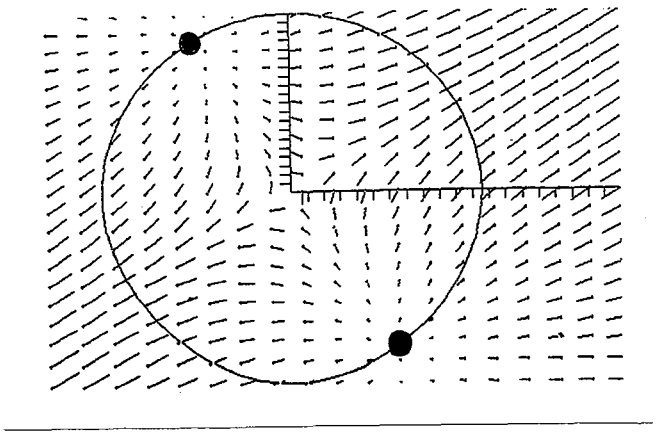

Fig.3 The convergence to the eigenvector fixpoints (•) with the smallest eigenvalue

Here we have two stable fixpoints on the unit circle, the eigenvectors $\mathbf{e}^2$ and $-\mathbf{e}^2$ with the smallest eigenvalue λ_2. The two eigenvectors with the biggest eigenvalue λ_1 are unstable. If we use instead the maximum searching gradient algorithm, the two stable fixpoints become unstable and the unstable ones with the biggest eigenvalue become stable. This is shown in figure 4.

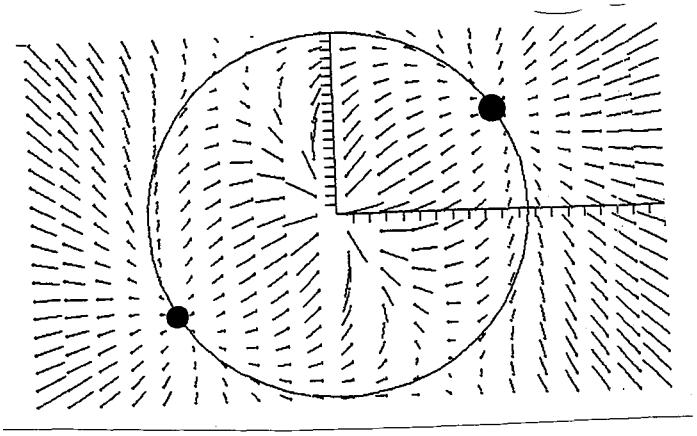

Fig.4 The convergence to the eigenvector fixpoints (•) with the biggest eigenvalue

Nevertheless, because in this example there exist only two eigenvectors the *n-2* unstable fixpoints common to both algorithms have to be demonstrated by other means.

3.3 Fixpoints and saddle points

For this purpose, let us regard a system with at least one saddle point which can be visualized by a 3D-plot. This is best done by a three-dimensional system of n=3. By (A.1),(A2),(A3) we know that, relative to a base system of eigenvectors, we have $f(\mathbf{w}) = \Sigma_i a_i^2\lambda_i$ with the components $a_1=|\tilde{\mathbf{w}}|^2\cos^2\beta$, $a_2 =|\tilde{\mathbf{w}}|^2\sin^2\beta$ and $a_3 = \cos^2\alpha$. Replacing $|\tilde{\mathbf{w}}|^2 = a_1^2+a_2^2 = 1-a_3^2$ gives us $f(\mathbf{w}) = \lambda_1(1-\cos^2\alpha)\cos^2\beta + \lambda_2(1-\cos^2\alpha)\sin^2\beta + \lambda_3\cos^2\alpha$. For $\lambda_1=\lambda_2=\lambda_3=1.0$ The objective function f(w) is constant, because the variance in all directions of the space is equal; there is neither an unique maximum nor minimum. If one eigenvalue becomes smaller the situation slightly changes. In figure 5 the corresponding objective function is plotted.

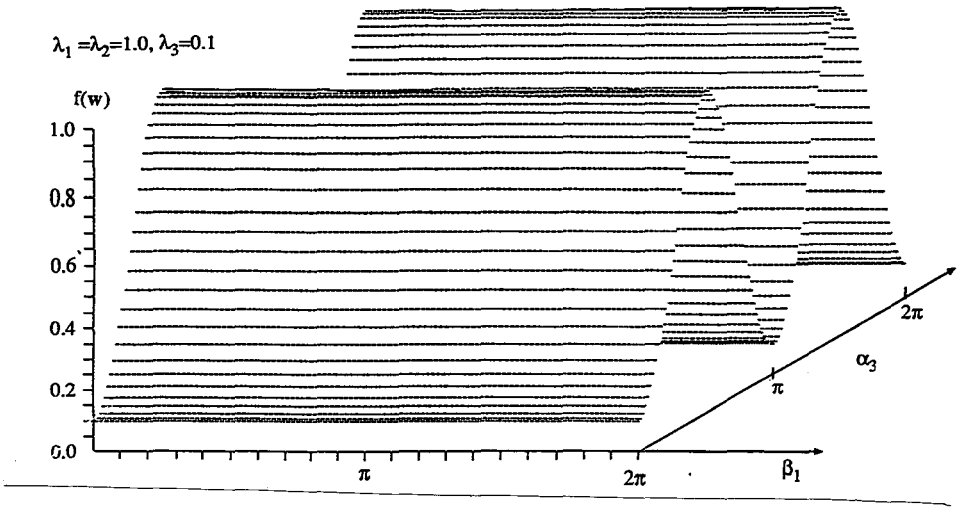

Fig. 5 The objective function with one small eigenvalue

The objective function becomes minimal at $\alpha=0,\pi$, i.e. at the eigenvector with the smallest eigenvalue. Here, the other angle β has no meaning. For the maximimum, the situation changes and with $\alpha=\pi/2$, $3\pi/2$ all possible values of β are solutions. This is quite instructive: The input space variance forms a disk where the direction of the smallest diameter is determined, but not the biggest one.

If we choose all the three eigenvalues different, figure 5 becomes figure 6. Here, the two fixpoints for a maximum and the two for a minimum (each for β and α) mark the eight fixpoints of the two directions of maximum and miunimum variance. Additionally, between the "hills" of figure 6 at $\beta=\pi/2$ and $\alpha=\pi/2$ we have the third, unstable fixpoint: in the direction of β it is a minimum but in the direction α it is a maximum. To reach this fixpoint, all algorithms which are uphill gradient ascends (maximum search) have to balance the input patterns such that the components in β1-direction are expected to have a constant value of $\beta=\pi/2$. On the other hand, all downhill algorithm (minimum search) have to maintain $\alpha=\pi/2$ to converge to the unstable point. This is the basis for all eigenvector decomposition networks.

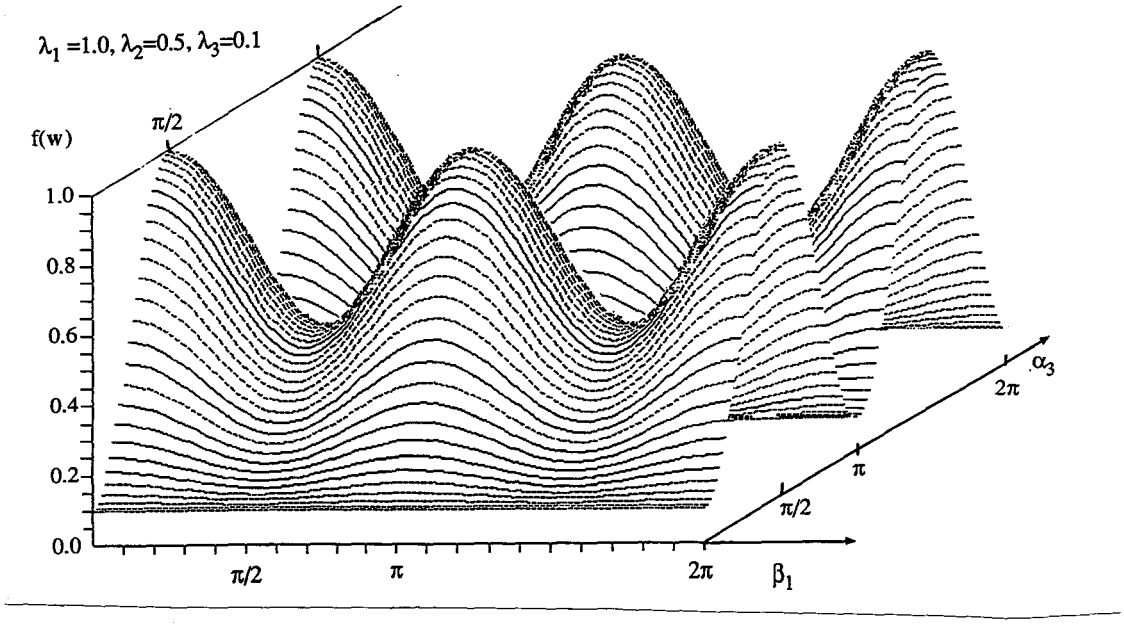

Fig.6 The objective function for three different eigenvalues

3.4 Non-centered input

All the preceeding networks assume that the pattern statistics are centered, i.e. the expected input $\langle \mathbf{x} \rangle$ is zero. Then the covariance matrix $\langle (\mathbf{x}-\langle \mathbf{x} \rangle)(\mathbf{x}-\langle \mathbf{x} \rangle)^T \rangle$ becomes the autocorrolation matrix $\mathbf{C}_{xx}=\langle \mathbf{x}\mathbf{x}^T \rangle$. For the latter case, the normalized Hebbian (or Anti-Hebbian) rule let the weight vector converge to an eigenvector of the autocorrelation matrix. In the former case, we are in trouble - how can we learn the eigenvectors of the covariance matrix ?

This can be overcome by the following approach. Let us redefine the input $\mathbf{x}^T=(x_1,..,x_n) \rightarrow \tilde{\mathbf{x}}^T:=(x_1,..,x_n,1)$ by an additional, constant line. Then the corresponding input weight w_{n+1} of $\tilde{\mathbf{w}}^T = (\mathbf{w}^T,w_{n+1}) = (w_1,..,w_n,w_{n+1})$ is learned by the Anti-Hebbian rule (3.8)

$$w_{n+1}(t+1) = w_{n+1}(t) - \gamma(t+1)\, x_{n+1}(t+1)\, y(t+1) \tag{3.11}$$

For the decreasing learning rate $\gamma(t):=1/t$ and the output $y(t+1)= \tilde{\mathbf{w}}^T(t)\tilde{\mathbf{x}}(t+1) = \mathbf{w}^T(t)\mathbf{x}(t+1) + x_{n+1}w_{n+1}$ this becomes with the activity (3.1)

$$w_{n+1}(t+1) = w_{n+1}(t) - 1/(t+1)\ (z(t+1)+w_{n+1}(t)) = w_{n+1}(t)(1-1/(t+1)) - z(t+1)/(t+1) \tag{3.12}$$

At the 2-th iteration with $w_{n+1}(1):=0$ this is

$$w_{n+1}(2) = w_{n+1}(1) - 1/2\, z(2) \ = - 1/2 \sum_{i=1}^{2} z(i)$$

Thus, by induction we have for (3.12) at the t+1-th iteration step with $(1-1/(t+1)) = t/(t+1)$

$$w_{n+1}(t+1) = t/(t+1)\, w_{n+1}(t) - z(t+1)/(t+1) = - 1/(t+1) \sum_{i=1}^{t+1} z(i) = - \langle z(t+1) \rangle \tag{3.13}$$

Thus, by the additional weight the output is

$$y = z - \langle z \rangle \qquad \text{with the mean value } \langle y \rangle = \langle z - \langle z \rangle \rangle = 0$$

The output becomes centered as if the input was centered; the objective function (3.2) remains valid and the weight vector converges to the eigenvector of the augmented input correlation matrix

$$\mathbf{C}_{\tilde{x}\tilde{x}}\tilde{\mathbf{w}}^* = \langle\tilde{\mathbf{x}}\tilde{\mathbf{x}}^T\rangle\,\tilde{\mathbf{w}}^* = \begin{bmatrix} \langle\mathbf{x}\mathbf{x}^T\rangle, \langle\mathbf{x}\rangle \\ \langle\mathbf{x}^T\rangle\ ,\ 1 \end{bmatrix}\begin{pmatrix}\mathbf{w}^* \\ -\langle z\rangle\end{pmatrix} = \begin{pmatrix}\langle\mathbf{x}\mathbf{x}^T\rangle\,\mathbf{w}^* - \langle\mathbf{x}\rangle\langle z\rangle \\ \langle\mathbf{x}^T\mathbf{w}^*\rangle - \langle z\rangle\end{pmatrix}$$

$$= \begin{bmatrix} \langle\mathbf{x}\mathbf{x}^T\rangle - \langle\mathbf{x}\rangle\langle\mathbf{x}^T\rangle, 0 \\ 0\ ,\ 0 \end{bmatrix}\begin{pmatrix}\mathbf{w}^* \\ w_{n+1}^*\end{pmatrix} = \lambda\,\tilde{\mathbf{w}}^*$$

Since

$$\langle\mathbf{x}\mathbf{x}^T\rangle - \langle\mathbf{x}\rangle\langle\mathbf{x}^T\rangle = \langle\mathbf{x}\mathbf{x}^T\rangle - 2\langle\mathbf{x}\langle\mathbf{x}^T\rangle\rangle + \langle\mathbf{x}\rangle\langle\mathbf{x}^T\rangle = \langle(\mathbf{x}-\langle\mathbf{x}\rangle)(\mathbf{x}-\langle\mathbf{x}\rangle)^T\rangle$$

is the covariance matrix, the part $\mathbf{w}^*$ of the eigenvector $\tilde{\mathbf{w}}^*$ of $\mathbf{C}_{\tilde{x}\tilde{x}}$ is the eigenvector of the covariance matrix which we looked for. Thus, the weights (except the threshold) will converge to the eigenvectors of the covariance matrix.

Nevertheless, since the additional constant input has no variance, the eigenvalue remains zero: it is the most stable feature.

4. The minimum entropy network

The base unit can be used in several ways. The direct approach replaces the Oja-unit of the cited eigenvector decomposition networks. Thus, the network of Sanger [San89] will first find the eigenvector with the minimal eigenvalue and subtract all its components from the input space, cf. figure 1. In the remaining space the second neuron will find the eigenvector with the smallest eigenvalue again which is the next one of the eigenvectors in ascending order of their eigenvalues. The same mechanism can work in the lateral inhibition network of Ruben and Tavan [Rub89]. In both networks, basically the Gram-Schmidt orthogonalization procedure is involved which depends only on the first base vector, the input statistics and the convergence goal (objective function) of each additional neuron.

The idea above sound reasonable, but it does not work: The maximum entropy and minimum entropy objectives are not symmetrical! The following section analyze this more deeply and shows, how the equations must be changed to reflect the proper objective.

Let us consider a simple, one-layer network of minimum entropy neurons as it is shown in figure 7. The *activity* of the network at time step t is determined by the linear equation

$$\mathbf{y}(t) = \mathbf{W}\,\mathbf{x}(t) \qquad \text{with } \mathbf{W} := (\mathbf{w}_1,\ldots,\mathbf{w}_m)^T \tag{4.1}$$

as it is associated with a classical, linear feed-forward network.

Nervertheless, for the procedure of *learning* the eigenvectors involves a completely different network. Starting with the first neuron each neuron gets its activity y_i, passes the reduced input to the next one, and correct and normalizes its weights (see fig.1). Since the idea is similiar to the General-Hebbian-network of Sanger [San89], the network is called "General-Anti-Hebbian-network" (GAH).

For the first neuron Eqs. (3.8) are valid and, as we know by section 3.1, the neuron will converge to the eigenvector with the smallest eigenvalue. To show the general step from s to $s+1$ in the induction, we have to show that neuron will converge to the eigenvector with the smallest eigenvalue of the n-s ones which rest, provided that all other s neurons have already converged to the eigenvectors with the s smallest eigenvalues.

Fig. 7 The General-Anti-Hebbian activity network

Now the neuron *s+1* will see as input

$$\tilde{\mathbf{x}}_s = \mathbf{x} + a\sum_i^s \tilde{y}_i \mathbf{w}_i \tag{4.2}$$

and gives as output

$$\tilde{y}_{s+1}(t) = \mathbf{w}_{s+1}{}^T \tilde{\mathbf{x}}_s \tag{4.3}$$

Thus, the objective function (3.2) of the neuron becomes

$$f(\mathbf{w}_{s+1}) = \mathbf{w}_{s+1}{}^T \mathbf{C}_{\tilde{x}\tilde{x}} \mathbf{w}_{s+1} \tag{4.4}$$

and the weights $\mathbf{w}_s$ will converge to the eigenvector of $\mathbf{C}_{\tilde{x}\tilde{x}}$ with the smallest eigenvalue by the gradient descend as in (3.8)

$$\bar{\mathbf{w}}_i(t+1) = \mathbf{w}_i(t) - \gamma(t)\, \tilde{\mathbf{x}}_{i-1}(t)\, y_i(t) \qquad 1 \le i \le m \quad \textit{Anti-Hebb-Rule} \tag{4.5a}$$

$$\tilde{\mathbf{x}}_0 := \mathbf{x},\ \tilde{\mathbf{x}}_i := \tilde{\mathbf{x}}_{i-1} + a\, \tilde{y}_i \mathbf{w}_i \qquad \textit{space correction} \tag{4.5b}$$

and

$$\mathbf{w}_i(t+1) = \bar{\mathbf{w}}_i(t+1) / |\bar{\mathbf{w}}_i(t+1)| \qquad \textit{normalization} \tag{4.5c}$$

The eigenvector equation is

$$\begin{aligned} \mathbf{C}_{\tilde{x}\tilde{x}}\mathbf{w}^* &= \langle \tilde{\mathbf{x}}_s \tilde{\mathbf{x}}_s{}^T \rangle \mathbf{w}^* = \langle (\mathbf{x} + a\textstyle\sum_i^s \tilde{y}_i \mathbf{w}_i)\,(\mathbf{x} + a\sum_j^s \tilde{y}_j \mathbf{w}_j)^T \rangle \mathbf{w}^* \\ &= \langle \mathbf{x}\mathbf{x}^T \rangle \mathbf{w}^* + a^2 \textstyle\sum_i^s \sum_j^s \langle \tilde{y}_i \tilde{y}_j \rangle \mathbf{w}_i \mathbf{w}_j{}^T \mathbf{w}^* + 2\, a\sum_i^s \langle \tilde{y}_i \mathbf{x} \mathbf{w}_i{}^T \rangle \mathbf{w}^* = \lambda \mathbf{w}^* \end{aligned} \tag{4.6}$$

Since we assume that the *s* weights have already converged to the eigenvectors $\mathbf{e}_i$, we know that $\mathbf{w}_i{}^T \mathbf{w}_j = 1$ only for i=j, otherwise it is zero. Therefore, we have for all i,j <s+1

$$\tilde{y}_i = \mathbf{w}_i{}^T \tilde{\mathbf{x}}_{i-1} = \mathbf{w}_i{}^T (\mathbf{x} + a\textstyle\sum_k^{i-1} \tilde{y}_k \mathbf{w}_k) = y_i + a\sum_k^{i-1} \tilde{y}_k \mathbf{w}_i{}^T \mathbf{w}_k = y_i$$

and therefore (4.6) becomes with (3.6)

$$\mathbf{C}_{\tilde{x}\tilde{x}}\mathbf{w}^* = \mathbf{C}\mathbf{w}^* + a^2 \textstyle\sum_i^s \sum_j^s \langle y_i y_j \rangle \mathbf{e}_i \mathbf{e}_j{}^T \mathbf{w}^* + 2\, a\sum_i^s \langle y_i \mathbf{x} \mathbf{e}_i{}^T \rangle \mathbf{w}^* = \lambda \mathbf{w}^* \tag{4.7}$$

As solution for $\mathbf{w}^*$ all *n* eigenvectors $\mathbf{e}_k$ of $\mathbf{C}$ are valid: if $\mathbf{w}^*$ is one of the *s* already obtained, the equation (4.7) will become

$$\mathbf{C}_{\tilde{x}\tilde{x}}\mathbf{e}_k = \lambda_k\mathbf{e}_k + a^2\lambda_k\mathbf{e}_k + 2a\langle y_k\mathbf{x}\rangle = \lambda_k\mathbf{e}_k + a^2\lambda_k\mathbf{e}_k + 2a\mathbf{C}\mathbf{e}_k = \lambda_k(1+a^2+2a)\mathbf{e}_k = \lambda\mathbf{e}_k$$

with the eigenvalue

$$\lambda = \lambda_k(1+a^2+2a) \tag{4.8}$$

If the eigenvector is new, the last two terms of (4.7) will become zero and the eigenvalue becomes

$$\lambda = \lambda_k \tag{4.9}$$

Therefore, if we would like to obtain a descending order of eigenvalues as Sanger [San89] does it, we just have to choose a:=-1. Then all old eigenvectors have an eigenvalue of zero and a gradient ascend (Eq. (4.5a) with positive sign) will find of the remaining ones the eigenvector with the biggest eigenvalue. This is basically the General Hebbian decomposition network.

Nevertheless, the problem to find the eigenvectors with the minimal eigenvalues is *not* symmetric. If we would use the gradient descend by Eqs. (4.5), the choice of a=-1 will make us find one of the eigenvectors already found which have the eigenvalue of zero: there is no other smaller eigenvalue! The eigenvalue in (4.8) of every eigenvector already found is only bigger than λ_{s+1}, the next one in the ascending order, if

$$\lambda_{s+1} < \lambda_k(1+a^2+2a) \qquad \text{for all } k < s+1$$

This must be true, even for the eigenvalues $\lambda_{s+1}=\lambda_{max}$ and $\lambda_k=\lambda_{min}$ of C.
So we get

$$\lambda_{max}/\lambda_{min} < (1+a^2+2a)=(a+1)^2$$

or $$a > (\lambda_{max}/\lambda_{min})^{1/2} - 1 \qquad \text{and} \qquad a < -(\lambda_{max}/\lambda_{min})^{1/2} - 1 \tag{4.10}$$

The following figure 8 shows the convergence of all the weights to the appropriate eigenvectors. As an error measure the absolute cosinus $g(\mathbf{e}_k,\mathbf{w}_i) := |\mathbf{e}_k^T\mathbf{w}_i|/|\mathbf{w}_i|$ is plotted against the number t of iterations for the example of a cyclic three pattern input $\mathbf{x}^{(i)}$. As you can observe, the error decreases very fast for the first neuron, whereas the convergence of the other two weights depend on the first one.

$\mathbf{x}^{(1)}=(1,0,0)$
$\mathbf{x}^{(2)}=(1,1,0)$
$\mathbf{x}^{(3)}=(1,1,1)$
$\lambda_1 = 0.1026$
$\lambda_2 = 0.2143$
$\lambda_3 = 1.6823$
$\gamma(t)=\gamma=0.01$
a= 5
$\mathbf{w}_i(0):=(1,1,1)$

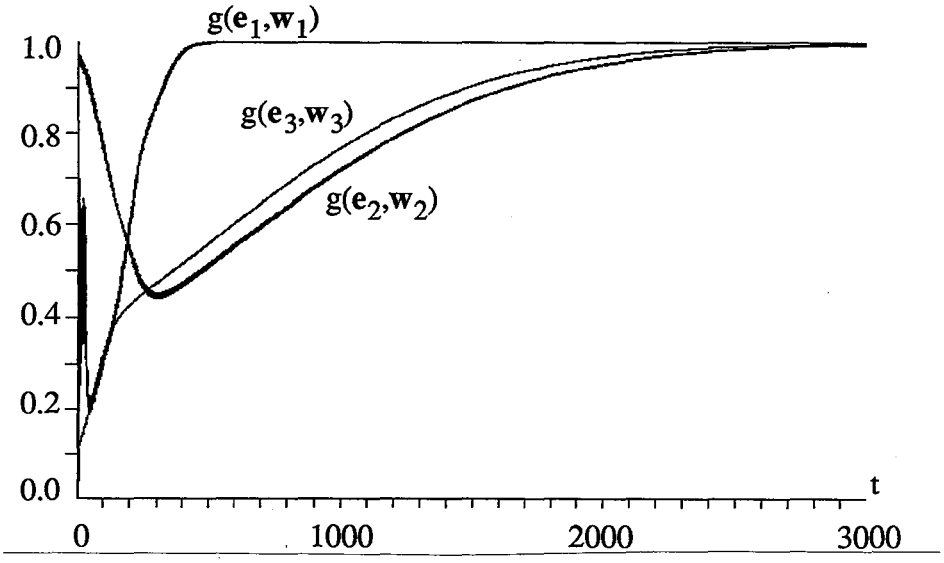

Fig.8 The convergence of the minimum entropy GAH network

In Figure 9 the case is shown when a=4 is too small. Then condition (4.8) is not met for the last eigenvalue $\lambda_3 = \lambda_{max}$ and the variance of the deterministic input disturb the convergence.

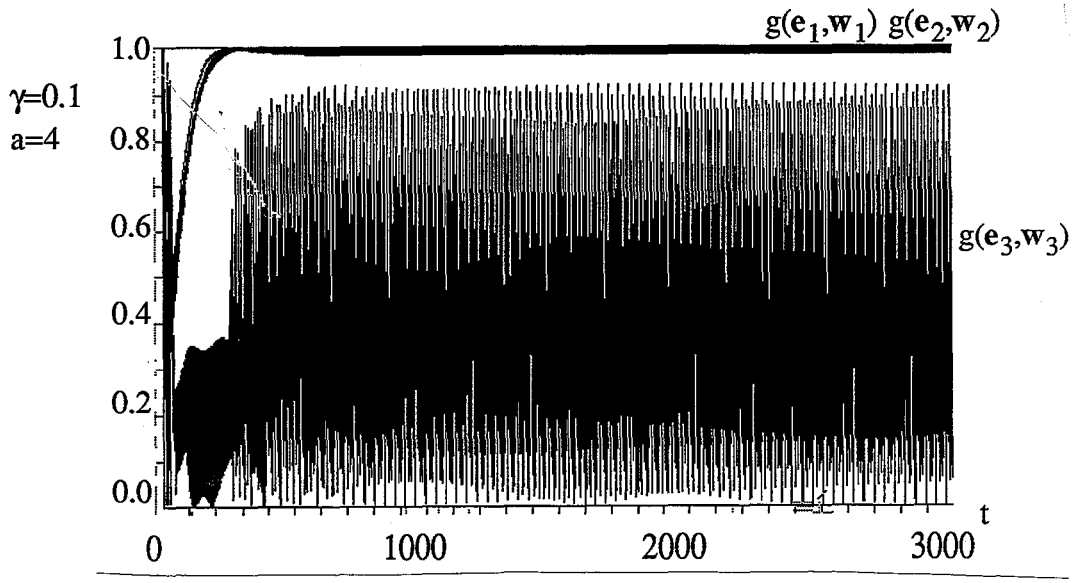

Fig.9 Partially non-converging GAH network

5. Discussion and conclusion

The paper showed how cluster transformation can be implemented by the base unit of a linear neuron where the weight vector converges to the eigenvector of the input pattern autocorrelation matrix with the smallest eigenvalue.

For m=n all the already known networks and the newer proposed ones decompose the input space into the complete set of eigenvectors. So, what is the difference of the proposed networks to the already existing ones? The main difference becomes evident for the case m<n when not all, but only a few eigenvectors are selected as target base. The maximum entropy networks choose first the eigenvectors with the *biggest* eigenvalues i.e. the features containing most of the information, neclecting all the rest. In contrast, the proposed ones will first select the eigenvectors with the *smallest* eigenvalues, thus choosing those features which are the most stable ones.

It should be noted that the proposed mechanism involves only linear neurons. Additional non-linearities in the neural output function S(z) (squashing function) will lead to further reduction of the cluster entropy, but do not provide directly the eigenvector decomposition [Oja91]. In the binary version it becomes the vector quantization which can directly be used for symbolic postprocessing of an object recognition system.

References

[Fuk72] K. Fukunaga: Introduction to Statistical Pattern Recognition; Academic Press, New York 1972.

[Jay84] N.S. Jayant, Peter Noll: Digital Coding of waveforms, Prentice Hall 1984.

[Oja82] Erkki Oja: A Simplified Neuron Model as a Principal Component Analyzer J. Math. Biol. 13: 267-273 (1982)

[Oja89] Erkki Oja: Neural Networks, Principal Components, and subspaces Int. J. Neural Systems, Vol 1/1 pp. 61-68 (1989)

[Oja91] E. Oja: Learning in non-linear Constrained Hebbian Networks; Proc. ICANN91, T.Kohonen et al. (Eds.), Artif. Neural Netw., Elsevier Sc. Publ. 1991, pp. 385-390

[Rub89] J. Rubner, P. Tavan: A Self-Organizing Network for Principal-Component Analysis, Europhys.Lett., 10(7), pp. 693-698 (1989).

[San89] Sanger: Optimal unsupervised Learning in a Single-Layer Linear Feedforward Neural Network; Neural Networks Vol 2, pp.459-473 (1989)

[Tou74] J.T. Tou, R.C. Gonzales: Pattern Recognition Principles; Addison-Wesley Publ. Comp., 1974

Appendix A The extrema of the objective function

The objective function is defined as

$$f(\mathbf{w}) = \langle y^2 \rangle = \langle \mathbf{w}^T \mathbf{x}\mathbf{x}^T \mathbf{w} \rangle = \mathbf{w}^T \mathbf{C}\mathbf{w} \tag{3.2}$$

Suppose that the symmetric **C** is of full rank. Then there exist a base vector system of orthogonal (and orthonormal) eigenvectors $\mathbf{e}^k$ of **C** with $\mathbf{C}\mathbf{e}^k = \lambda_k \mathbf{e}^k$ such that each **w** can be represented in this base by

$$\mathbf{w} = \sum_i a_i \mathbf{e}^i \qquad \text{and } a_i = \mathbf{w}^T \mathbf{e}^i = |\mathbf{w}|\,|\mathbf{e}^i| \cos(\mathbf{w}, \mathbf{e}^i)$$

with the projection a_i of **w** on the eigenvector $\mathbf{e}^i$. Due to the orthonormality of $\mathbf{e}^i$ and the constrain of **w** the coefficients depend only on the angle α_i between **w** and the eigenvector and we have

$$a_i = \cos(\mathbf{w}, \mathbf{e}^i) = \cos \alpha_i$$

By this condition we change our coordinates from Kartesian to polar based description. Nevertheless, by the constrain the *n* coordinates remain implicitely dependend from each other. Therefore, to eliminate the dependence we choose the first two variables α_i and α_2 and replace them by an independant variable β. For this purpose, let us regard the projection of **w** on the plane between $\mathbf{e}^1$ and $\mathbf{e}^2$, see figure A.1.

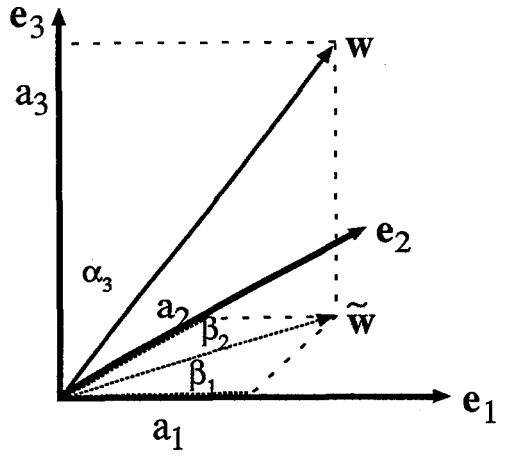

Figure A.1 The projection on a plane

The projection of **w** on one plane has the form $\tilde{\mathbf{w}} = a_1 \mathbf{e}^1 + a_2 \mathbf{e}^2$ because the difference vector $(\mathbf{w} - \tilde{\mathbf{w}})$ is orthogonal on the plane: $(\mathbf{w} - \tilde{\mathbf{w}})^T \mathbf{e}^1 = 0 = (\mathbf{w} - \tilde{\mathbf{w}})^T \mathbf{e}^2$. For the projection $\tilde{\mathbf{w}}$ we can replace the angle β_2 by its complemental counterpart β_1

$$\cos \beta_2 = \cos(\pi/2 - \beta_1) = \sin \beta_1$$

Thus, the objective function (3.2) becomes

$$f(\mathbf{w}) = \mathbf{w}^T \mathbf{C}\mathbf{w} = (\sum_i a_i \mathbf{e}^i)\, \mathbf{C}\, (\sum_j a_j \mathbf{e}^j) = \sum_i \sum_j a_i a_j \mathbf{e}^i \mathbf{C}\, \mathbf{e}^j = \sum_i a_i^2 \lambda_i \tag{A.1}$$

with the components

$$a_1 = \tilde{\mathbf{w}}^T \mathbf{e}^1 = |\tilde{\mathbf{w}}|\,|\mathbf{e}^1| \cos \beta = |\tilde{\mathbf{w}}| \cos \beta \qquad \beta := \beta_1 \tag{A.2}$$

$$a_2 = \tilde{\mathbf{w}}^T \mathbf{e}^2 = |\tilde{\mathbf{w}}|\,|\mathbf{e}^2| \sin \beta = |\tilde{\mathbf{w}}| \sin \beta \tag{A.3}$$

The length of the projection $|\tilde{w}|^2=a_1^2+a_2^2$ depends on all the other angles α_i but not on β

$$|w|^2 = 1 = a_1^2+a_2^2 + \Sigma_{k=3}^n a_k^2 \;\Rightarrow\; |\tilde{w}|^2 = 1 - \Sigma_{k=3}^n a_k^2 \qquad (A.3)$$

Now, the objective function depends only on the *n-1* independant variables β, $\alpha_3,...,\alpha_n$. The necessary conditions for the extrema are

$$\text{grad } f(\mathbf{w}) = \text{grad } f(\alpha) = \mathbf{0}$$

Let us evaluate this for all variables β, $\alpha_3,...,\alpha_n$.

$$\frac{\partial}{\partial\beta} f(\mathbf{w}) = \frac{\partial}{\partial\beta} \Sigma_k a_k^2\lambda_k = \frac{\partial}{\partial\beta} a_1^2\lambda_1 + \frac{\partial}{\partial\beta} a_2^2\lambda_2$$
$$= -\lambda_1 2|\tilde{w}|^2\cos\beta\sin\beta + \lambda_2 2|\tilde{w}|^2\sin\beta\cos\beta = (\lambda_2-\lambda_1)2|\tilde{w}|^2\cos\beta\sin\beta \qquad (A.4)$$

because the length $|\tilde{w}|$ of the projection does not change when the angle β is changed.
For $(\lambda_2-\lambda_1)\neq 0$ and $|\tilde{w}|\neq 0$ the conditions (A.4) become zero when $\mathbf{w}^*$ or β^* is given by

$$\sin\beta^* = 0 \;<=>\; \beta^*=0,\pi \qquad (A.5)$$
$$\cos\beta^* = 0 \;<=>\; \beta^*=\pi/2,\, 3\pi/2$$

For all other variables α_i, i=3..n we have

$$\frac{\partial}{\partial\alpha_i} f(\mathbf{w}) = \frac{\partial}{\partial\alpha_i} \Sigma_k a_k^2\lambda_k = \frac{\partial}{\partial\alpha_i} a_1^2\lambda_1 + \frac{\partial}{\partial\alpha_i} a_2^2\lambda_2 + \frac{\partial}{\partial\alpha_i} a_i^2\lambda_i$$

$$= (\lambda_1\cos^2\beta + \lambda_2\sin^2\beta - \lambda_i)\, 2\cos\alpha_i\sin\alpha_i \qquad (A.6)$$

$$\text{with } \frac{\partial}{\partial\alpha_i}|\tilde{w}|^2 = \frac{\partial}{\partial\alpha_i}(1 - \Sigma_{k=3}^n a_k^2) = -\frac{\partial}{\partial\alpha_i} a_i^2 = 2\cos\alpha_i\sin\alpha_i \qquad (A.6b)$$

and therefore with $(\lambda_1\cos^2\beta + \lambda_2\sin^2\beta - \lambda_i) \neq 0$ we have

$$\sin\alpha_i^* = 0 \quad <=> \alpha_i^*=0,\pi \qquad \text{for all i=3, ..., n} \qquad (A.7)$$
$$\cos\alpha_i^* = 0 \quad <=> \alpha_i^*=\pi/2,\, 3\pi/2$$

The solutions (A.5) and (A.7) correspond to the solutions obtained earlier in (3.5): the extrema occur for all $\mathbf{w}$ parallel ($\beta^*=0$, $\alpha_i^*=0$) or antiparallel ($\beta^*=\pi$, $\alpha_i^*=\pi$) to the eigenvector $\mathbf{e}^1$ or $\mathbf{e}^i$ which is orthogonal ($\beta^*=\pi/2$, $3\pi/2$ and $\alpha_i^*=\pi/2$, $3\pi/2$) to the other eigenvectors.

In the formulation with *n-1* independant angles we can discuss the nature of the extrema (and thus the nature of the fixpoints of the corresponding gradient algorithm) by the use of the second derivatives in the Hesse matrix $A = (f_{ij}) = (\partial^2 f(\alpha) / \partial\alpha_i\partial\alpha_j)$ at the extrema

$$\mathbf{w}^*=\mathbf{e}^1 \;<=>\; \beta^*=0,\pi\,, \qquad \alpha_i^*=\pi/2,\, 3\pi/2 \qquad (A.8)$$
$$\mathbf{w}^*=\mathbf{e}^2 \;<=>\; \beta^*=\pi/2,\, 3\pi/2\,, \qquad \alpha_i^*=\pi/2,\, 3\pi/2$$
$$\mathbf{w}^*=\mathbf{e}^i \;<=>\; \beta^*=\pi/2,\, 3\pi/2\,, \qquad \alpha_i^*=0,\pi$$

The mixed terms with $i\neq 1$ are by (A.4) and (A.6b)

$$\frac{\partial^2 f(\mathbf{w})}{\partial\beta\partial\alpha_j} = (\lambda_2-\lambda_1)2\frac{\partial}{\partial\alpha_j}|\tilde{w}|^2\cos\beta\sin\beta = (\lambda_2-\lambda_1)4\cos\alpha_i\sin\alpha_i\cos\beta\sin\beta \qquad (A.9)$$

which is identical to $\partial^2 f(\mathbf{w})/\partial\alpha_j\partial\beta$. For all extrema of (A.8) the mixed terms (A.9) become zero.

The other terms for all i,j=3, ..., n are with (A.6)

$$\frac{\partial^2 f(\mathbf{w})}{\partial\alpha_i \partial\alpha_j} = (\lambda_1 \cos^2\beta + \lambda_2 \sin^2\beta - \lambda_i)\, 2\frac{\partial}{\partial\alpha_j} \cos\alpha_i \sin\alpha_i$$

which becomes zero for all i≠j, otherwise by $\cos^2\alpha + \sin^2\alpha = 1$ we get

$$\frac{\partial^2 f(\mathbf{w})}{\partial^2 \alpha_i} = (\lambda_1 \cos^2\beta + \lambda_2 \sin^2\beta - \lambda_i)\, 2(1-2\sin^2\alpha_i) \qquad \text{(A.10)}$$

Finally, for i=j=1 we have

$$\frac{\partial^2 f(\beta)}{\partial^2 \beta} = \frac{\partial}{\partial\beta} (\lambda_2-\lambda_1) 2|\tilde{w}|^2 \cos\beta \sin\beta = (\lambda_2-\lambda_1) 2|\tilde{w}|^2 (1- 2\sin^2\beta) \qquad \text{(A.11)}$$

Since all mixed terms are zero, the *n-1* dimensional Hesse matrix becomes a diagonal matrix; its eigenvalues are just the components (A.10) and (A.11). Thus, for a minimum of the objective function at $\mathbf{e}^1$ all the second derivatives must be greater than zero. This is the case at ($\beta^*=0,\pi$, $\alpha_i^*=\pi/2, 3\pi/2$) when by (A.10) $(\lambda_i-\lambda_1)>0$ and by (A.11) $(\lambda_2-\lambda_1)>0$, i.e. the eigenvalue λ_1 is smaller than λ_2 and any other λ_i, it must be the smallest eigenvalue. Since the choice of $\mathbf{e}^1$ was arbitrary, the same arguments hold for any other eigenvector $\mathbf{e}^i$ with the smallest eigenvalue: it is the unique minimum. This can be verified by the interested reader by the application of the other extrema at $\mathbf{e}^i$ in (A.8) to Eqs. (A.10) and (A.11).

The equivalent argumentation holds for the unique maximum: (A.10) and (A.11) are negative for an extremum (a maximum) only iff λ_i is the maximal eigenvalue.

Now, let us denote by $\mathbf{w}^{max}$ the point with the biggest eigenvalue, by $\mathbf{w}^{min}$ the point with the smallest eigenvalue. Then, by the arguments above, all other eigenvectors correspond to extrema which fulfill both maximum and minimum conditions. The nature of the extrema depend on the direction of approaching them: they are saddle points which correspond to unstable fixpoints.

INTERNE BERICHTE AM FACHBEREICH INFORMATIK, UNIV. FRANKFURT

1/87 Risse, Thomas: On the number of multiplications needed to evaluate the reliability of k-out-of-n systems

Modelling interrupt based interprocessor communication by Time Petri Nets

2/87 Roll, Georg u.a.: Ein Assoziativprozessor auf der Basis eines modularen vollparallelen Assoziativspeicherfeldes

3/87 Waldschmidt, Klaus; Roll, Georg: Entwicklung von modularen Betriebssystemkernen für das ASSKO-Multi-Mikroprozessorsystem

4/87 Workshop über Komplexitätstheorie, effiziente Algorithmen und Datenstrukturen ; 3.2.1987, Universität Frankfurt/Main

5/87 Seidl, Helmut: Parameter-reduction of higher level grammars

6/87 Kemp, Rainer: On systems of additive weights of trees

7/87 Kemp, Rainer: Further results on leftist trees

8/87 Seidl, Helmut: The construction of minimal models

9/87 Weber, Andreas; Seidl, Helmut: On finitely generated monoids of matrices with entries in N

10/87 Seidl, Helmut: Ambiguity for finite tree automata

1/88 Weber, Andreas: A decomposition theorem for finite-valued transducers and an application to the equivalence problem

2/88 Roth, Peter: A note on word chains and regular languages

3/88 Kemp, Rainer: Binary search trees for d-dimensional keys

4/88 Dal Cin, Mario: On explicit fault-tolerant, parallel programming

5/88 Mayr, Ernst W.: Parallel approximation algorithms

6/88 Mayr, Ernst W.: Membership in polynomial ideals over Q is exponential space complete

1/89 Lutz, Joachim [u.a.]: Parallelisierungskonzept für ATTEMPTO-2

2/89 Lutz, Joachim [u.a.]: Die Erweiterungen der ATTEMPTO-2 Laufzeitbibliothek

3/89 Kemp, Rainer: A One-to-one Correspondence between Two Classes of Ordered Trees

4/89 Mayr, Ernst W.; Plaxton, C. Greg: Pipelined Parallel Prefix Computations, and Sorting on a Pipelined Hypercube

5/89 Brause, Rüdiger: Performance and Storage Requirements of Topology-conserving Maps for Robot Manipulator Control

6/89 Roth, Peter: Every Binary Pattern of Length Six is Avoidable on the Two-Letter Alphabet

7/89 Mayr, Ernst W.: Basic Parallel Algorithms in Graph Theory

8/89 Brauer, Johannes: A Memory Device for Sorting

1/90 Vollmer, Heribert: Subpolynomial Degrees in P and Minimal Pairs for L

2/90 Lenz, Katja: The Complexity of Boolean Functions in Bounded Depth Circuits over the Basis $\{\wedge, \oplus\}$

3/90 Becker, Bernd; Hahn, R.; Krieger, R.; Sparmann, U.: Structure Based Methods for Parallel Pattern Fault Simulation in Combinational Circuits

4/90 Goldstine, J.; Kintala, C.M.R.; Wotschke, D.: On Measuring Nondeterminism in Regular Languages

5/90 Goldstine, J.; Leung, H.; Wotschke, D.: On the Relation between Ambiguity and Nondeterminism in Finite Automata

1/91 Brause, Rüdiger: Approximator Networks and the Principles of Optimal Information Distribution

2/91 Brauer, Johannes; Stuchly, Jürgen: HyperEDIF: Ein Hypertext-System für VLSI Entwurfsdaten

3/91 Brauer, Johannes: Repräsentation von Entwurfsdaten als symbolische Ausdrücke

4/91 Trier, Uwe: Additive Weights of a Special Class of Nonuniformly Distributed Backtrack Trees

5/91 Dömel, P. [u.a.]: Concepts for the Reuse of Communication Software

6/91 Heistermann, Jochen: Zur Theorie genetischer Algorithmen

7/91 Wang, Alexander [u.a.]: Embedding complete binary trees in faulty hypercubes

1/92 Brause, Rüdiger: The Minimum Entropy Network

2/92 Trier, Uwe: Additive Weights Under the Balanced Probability Model