Principal Component Analysis

based on: Neural networks: a comprehensive foundation
by Simon Haykin

1 - The Hebbian Learning

- Hebb's postulate of learning is the oldest and most famous of all learning rules; it is named in honor of neuropsychologist Hebb (1949).
- The Organization of Behaviour (1949): When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic changes take place in one or both cells such that A's efficiency as one of the cells firing B, is increased.
- Hebb proposed this change as a basis of associative learning (at the cellular level), which would result in an enduring modification in the activity pattern of a spatially distributed assembly of nerve cells.
- This statement is made in a neurobiological context. We can restate it as:

1. If two neurons on either side of a synapse are activated simultaneously (i.e. synchronously), then the strength of that synapse is selectively increased.
2. If two neurons on either side of a synapse are activated asynchronously, then that synapse is selectively weakened or eliminated.

- The Hebb's hypothesis:

$$\Delta w_{kj}(n) = \eta y_k(n) x_j(n), \tag{1}$$

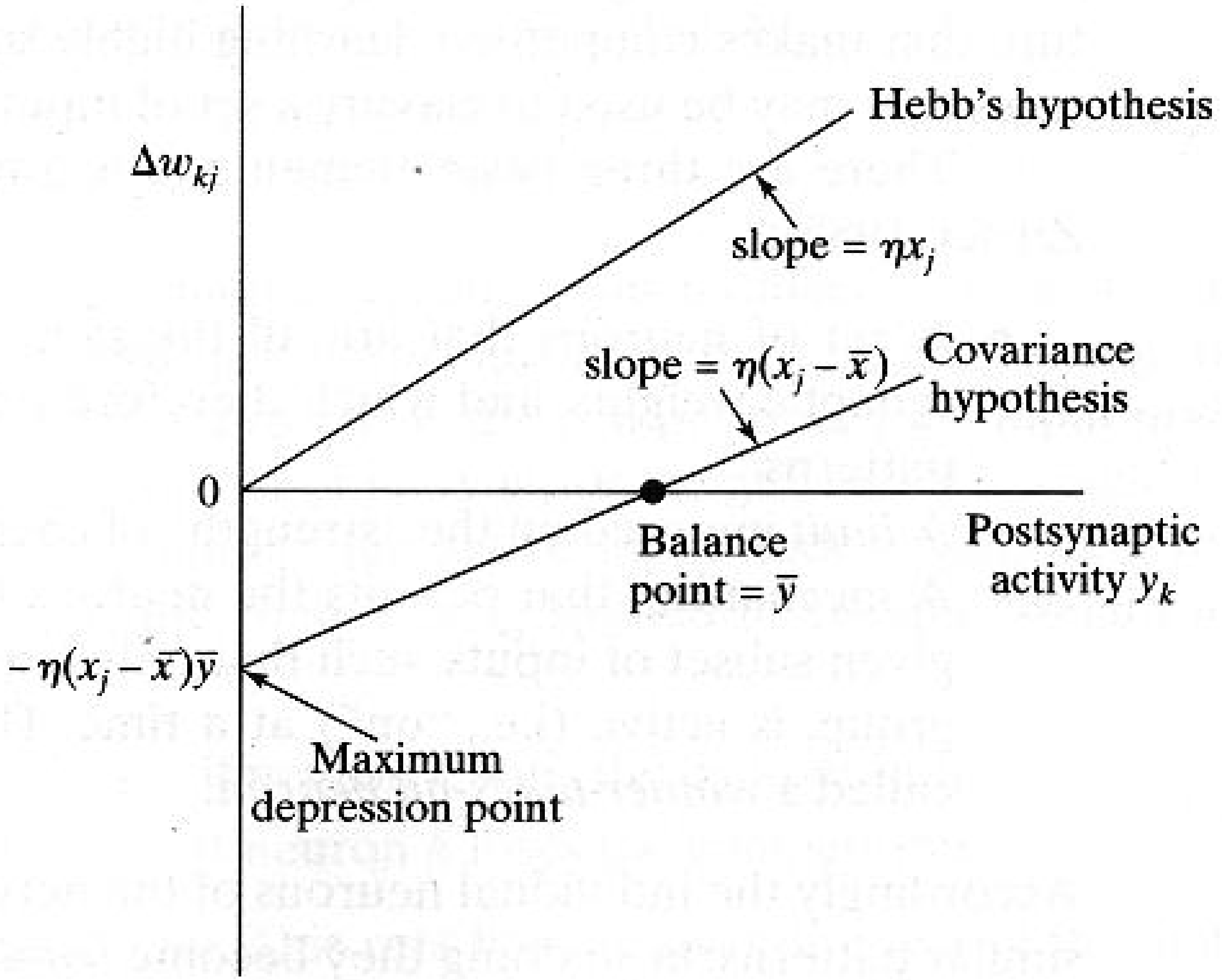
where η is the learning rate. However, this leads to exponential growth which is physically unacceptable.

- In order, to overcome this Sejnowsky (1977) introduced the covariance hypothesis:

$$\Delta wkj = \eta(x_j - \bar{x})(y_k - \bar{y}). \tag{2}$$

In this situation, w_{kj} can increase or decrease.

- Both learning rules are ilustrated in the following Figure.


Δw_{kj}
Hebb's hypothesis
slope = ηx_j
slope = $\eta(x_j - \bar{x})$
Covariance hypothesis
0
Balance point = $\bar{y}$
Postsynaptic activity y_k
$-\eta(x_j - \bar{x})\bar{y}$
Maximum depression point

2 - Principal Component Analysis (Karhunen-Loève Transformation)

- $X \in \mathbb{R}^m$.
- X has zero mean: $E[\mathbf{X}] = \mathbf{0}$. If X has a nonzero mean, first it is necessary to subtract it.
- Let $\mathbf{q}$ denote a unit vector of dimension m.
- The projection of $\mathbf{X}$ onto $\mathbf{q}$:

$$A = \mathbf{X}^T \mathbf{q} = \mathbf{q}^T \mathbf{X}, \tag{3}$$

subject to the constraint:

$$\|\mathbf{q}\| = (\mathbf{q}^T \mathbf{q})^{1/2} = 1. \tag{4}$$

- The projection A is a *random variable* with a mean and variance related to the mean and variance of the random vector $\mathbf{X}$.

- If random vector $\mathbf{X}$ has zero mean then the mean value of $\mathbf{A}$ is zero too:
$$E[A] = \mathbf{q}^T E[\mathbf{X}]. \tag{5}$$

- The variance of A is:
$$\begin{aligned} \sigma^2 &= E[A^2], & (6)\\ &= E[(\mathbf{q}^T\mathbf{X})(\mathbf{X}^T\mathbf{q})], & (7)\\ &= \mathbf{q}^T E[\mathbf{X}\mathbf{X}^T]\mathbf{q}, & (8)\\ &= \mathbf{q}^T\mathbf{R}\mathbf{q}. & (9)\end{aligned}$$

  where $\mathbf{R}^{m\times m}$ is the correlation matrix $R = E[\mathbf{X}\mathbf{X}^T]$.

- $\mathbf{R}$ is symmetric: $\mathbf{R}^T = \mathbf{R}$.

- From Eq. 9 we see that the variance σ^2 of the projection A is a function of the unit vector $\mathbf{q}$:

$$\begin{aligned} \psi(\mathbf{q}) &= \sigma^2, && (10) \\ &= \mathbf{q}^T\mathbf{R}\mathbf{q}. && (11) \end{aligned}$$

- Therefore $\psi(\mathbf{q})$ can be think as a *variance probe*.

3 - Eigenstructure of Principal Component Analysis

- We want to find those unit vectors $\mathbf{q}$ along which $\psi(\mathbf{q})$ has extremal values (local maxima or minima), subject to a constraint on the Euclidean norm of $\mathbf{q}$.
- The solution of this problem lies can be expressed by:

$$\mathbf{R}\mathbf{q} = \lambda\mathbf{q}, \tag{12}$$

  which is the equation of the eigenstructure of the correlation matrix $\mathbf{R}$.

- Let the eigenvalues of $\mathbb{R}^{m\times m}$ be denoted by $\lambda_1, \lambda_2, \ldots, \lambda_m$ and the associated eigenvectors be denoted by $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_m$:

$$\mathbf{R}\mathbf{q}_j = \lambda_j\mathbf{q}_j \qquad j = 1, \ldots, m. \tag{13}$$

- Let the corresponding eigenvalues be arranged in decreasing order:

$$\lambda_1 > \lambda_2 > \ldots > \lambda_j > \ldots > \lambda_m,$$

so that $\lambda_1 = \lambda_{\max}$.

- Let the associated eigenvectors be used to construct the following matrix:

$$\mathbf{Q} = \left[\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_j, \ldots, \mathbf{q}_m\right]. \tag{14}$$

- We may then combine the set of m equations of Eq. 13 into a single equation:

$$\mathbf{RQ} = \mathbf{Q\Lambda}, \tag{15}$$

  where $\mathbf{\Lambda}$ is a diagonal matrix defined by:

$$\mathbf{\Lambda} = \operatorname{diag}\left[\lambda_1, \lambda_2, \ldots, \lambda_j, \ldots, \lambda_m\right] \tag{16}$$

- $\mathbf{Q}$ is an orthogonal matrix:

$$\mathbf{q}_i^T \mathbf{q}_j = \begin{cases} 1, & j = i \\ 0, & j \neq i \end{cases}. \tag{17}$$

- we may write:

$$\mathbf{Q}^T\mathbf{Q} = \mathbf{I}, \tag{18}$$

or:

$$\mathbf{Q}^T = \mathbf{Q}^{-1}. \tag{19}$$

- We may rewrite Eq. 15 as the *orthogonal Similarity transformation*:

$$\mathbf{Q}^T\mathbf{R}\mathbf{Q} = \mathbf{\Lambda}. \tag{20}$$

Or in a expanded form:

$$\mathbf{q}_j^T\mathbf{R}\mathbf{q}_k = \begin{cases} \lambda_j, & k = j \\ 0, & k \neq j. \end{cases} \tag{21}$$

- The correlation matrix $\mathbf{R}$ may be itself be expressed in terms of its eigenvalues and eigenvectors as:

$$\mathbf{R} = \sum_{i=1}^{m} \lambda_i \mathbf{q}_i \mathbf{q}_i^T. \tag{22}$$

- Principal component analysis and eigendecomposition of matrix $\mathbf{R}$ are basically the same thing.
- Variance probes and eigenvalues are equal:

$$\psi(\mathbf{q}_j) = \lambda_j, \quad j = 1, 2, \ldots, m. \tag{23}$$

- Let the data vector $\mathbf{x}$ denote a realisation of the random vector $\mathbf{X}$.
- The projection of $\mathbf{x}$ onto $\mathbf{q}_j$ can be written like:

$$a_j = \mathbf{q}_j^T \mathbf{x} = \mathbf{x}^T \mathbf{q}_j, \quad j = 1, 2, \ldots, m. \tag{24}$$

- The a_j are called the principal components.
- The reconstruction:

$$\begin{aligned} \mathbf{a} &= [a_1, a_2, \ldots, a_m]^T, & (25) \\ &= \left[\mathbf{x}^T \mathbf{q}_2, \mathbf{x}^T \mathbf{q}_2, \mathbf{x}^T \mathbf{q}_m\right]^T, & (26) \\ &= \mathbf{Q}^T \mathbf{x}. & (27) \end{aligned}$$

$$\begin{aligned} \mathbf{x} &= \mathbf{Q}\mathbf{a} & (28) \\ &= \sum_{j=1}^{m} a_j \mathbf{q}_j. & (29) \end{aligned}$$

4 - Dimensionality Reduction

- PCA provides an effective technique for dimensionality reduction.
- Let $\lambda_1, \lambda_2, \ldots, \lambda_l$ denote the largest eigenvalues of the correlation matrix $\mathbf{R}$.
- The data vector $\mathbf{x}$ can be aproximated by truncating the expansion of:

$$\mathbf{x} = \mathbf{Q}^T \mathbf{a} = \sum_{j=1}^{m} a_j \mathbf{q}_j, \tag{30}$$

after l terms as follows:

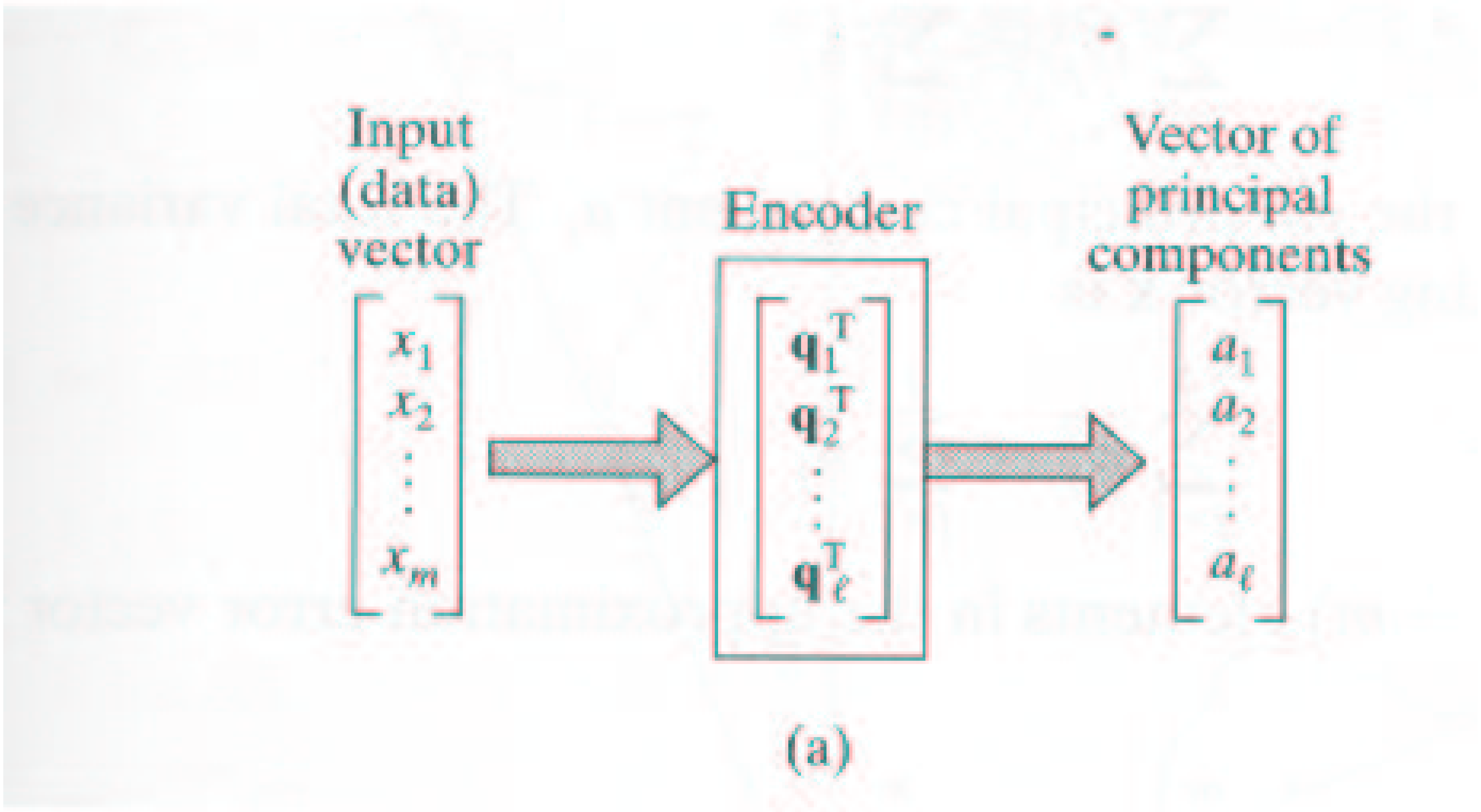
$$\hat{\mathbf{x}} = \sum_{j=1}^{l} a_j \mathbf{q}_j , \tag{31}$$

$$= [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_l] \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_l \end{bmatrix} , \quad l \leq m \tag{32}$$

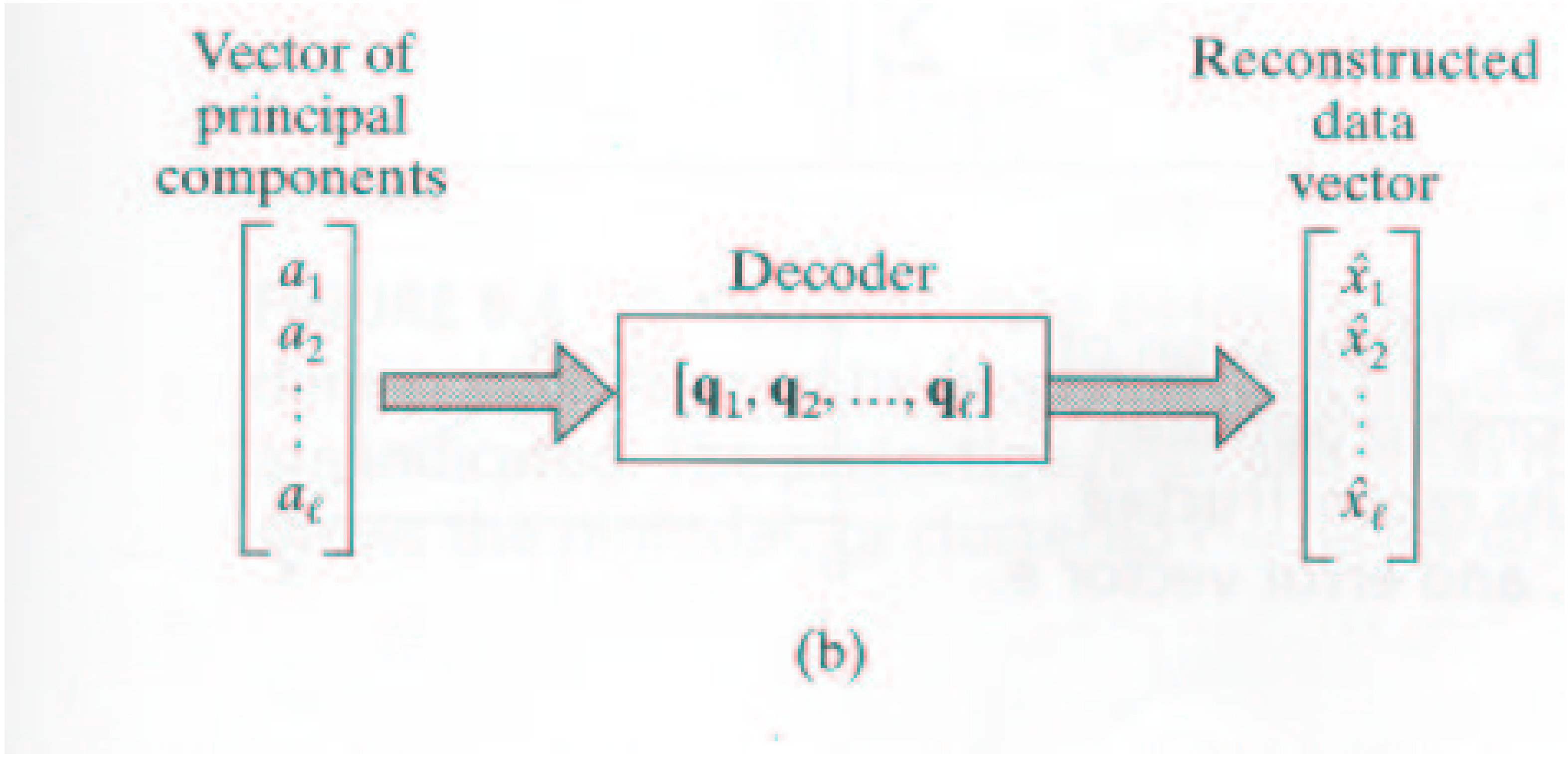
- Given the original data vector $\mathbf{x}$ it is possible to calculate the set of principal components $\mathbf{a}$ as follows:

$$\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_l \end{bmatrix} = \begin{bmatrix} \mathbf{q}_1^T \\ \mathbf{q}_2^T \\ \vdots \\ \mathbf{q}_l^T \end{bmatrix} \mathbf{x}, \quad l \leq m \tag{33}$$

- The linear projection from $\mathbb{R}^m \to \mathbb{R}^l$ (i.e., the mapping from the data space to the feature space) represents an encoder for the approximate representation of the data vector $\mathbf{x}$.

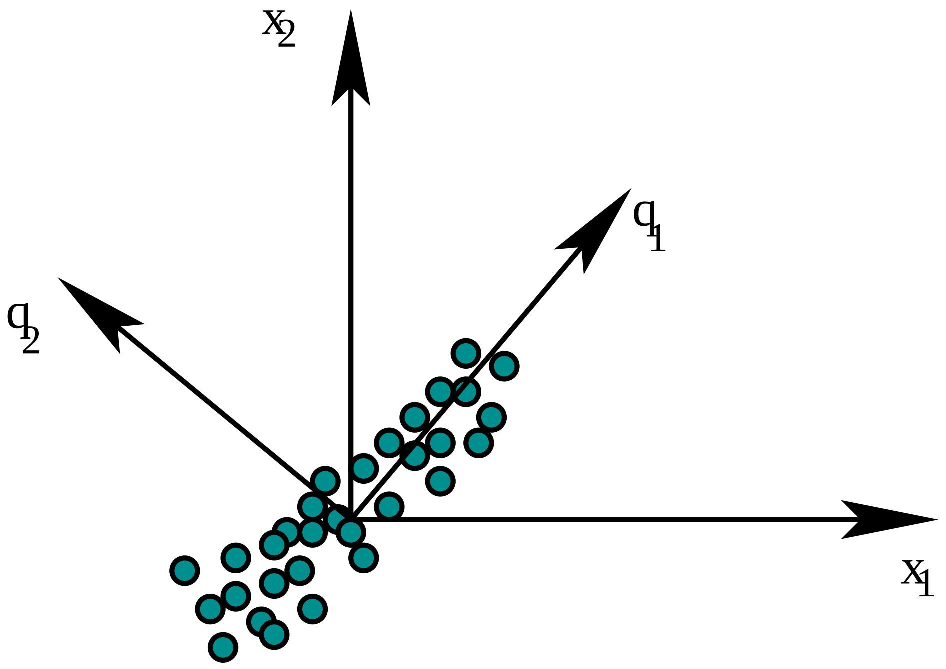

Input (data) vector
x_1 x_2 ⋮ x_m
Encoder
$\mathbf{q}_1^T$ $\mathbf{q}_2^T$ ⋮ $\mathbf{q}_\ell^T$
Vector of principal components
a_1 a_2 ⋮ a_ℓ
(a)

- The linear projection from $\mathbb{R}^l \to \mathbb{R}^m$ represents a decoder for the approximate reconstruction of the original data vector $\mathbf{x}$.


Vector of principal components
a_1
a_2
a_ℓ
Decoder
$[\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_\ell]$
Reconstructed data vector
$\hat{x}_1$
$\hat{x}_2$
$\hat{x}_\ell$
(b)

5 - An example

- Let $\mathbf{X}$ be an 2-dimensional distribution like in the following figure.



- The eigenvectors $\mathbf{q}_1$,$\mathbf{q}_2$ of the correlation matrix $\mathbf{R} = E[\mathbf{X}\mathbf{X}^T]$ will be at the direction of the largest variances.

- If we are interested only in the first principal component:

$$a_1 = \mathbf{q}_1^T \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}. \tag{34}$$

- To aproximate reconstruct the data vector $\mathbf{x}$:

$$\begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix} = a_1 \mathbf{q}_1. \tag{35}$$

6 - Hebbian-based maximum eigenfilter

- There is a close correspondence between the behaviour of self-organized networks and the statistical method of principal component analysis.
- Here we show that a single linear neuron with a Hebbian-type adaptation rule for its synaptic weights can evolve into a filter for the first principal component of the input distribution.
- Consider the neuron (as depicted in the figure) with a set of m input signals $x_1, x_2, \ldots, x_m$ with a corresponding set of m synapses with weights $w_1, \ldots, w_m$ respectivamente.

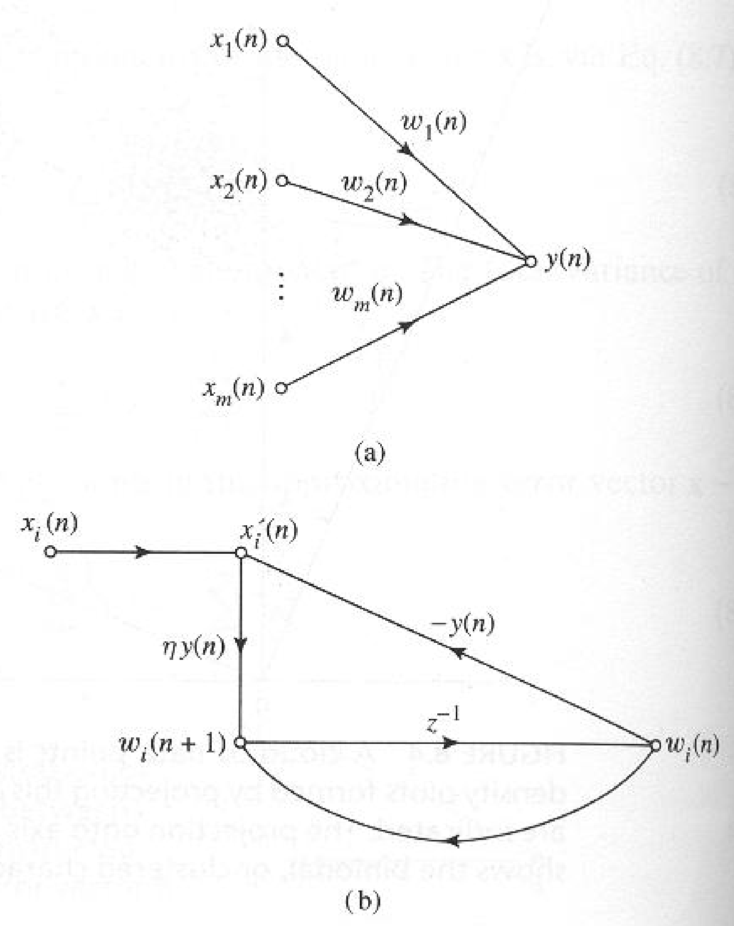


FIGURE 8.5 Signal-flow graph representation of maximum eigenfilter. (a) Graph of Eq. (8.36). (b) Graph of Eqs. (8.41) and (8.42).

- O modelo resultante pode ser definido como:

$$y = \sum_{i=1}^{m} w_i x_i \tag{36}$$

- According to the Hebb's postulate of learning we may write:

$$w_i(n+1) = w_i(n) + \eta y(n) x_i(n), \quad i = 1, 2, \ldots, m, \tag{37}$$

whre n denotes discrete time and η is the learning rate parameter.

- However, this learning rule in its basic form leads to the unlimited growth of the synaptic weight w_i which is unacceptable.

- We can overcome this by using saturation or normalisation:

$$w_i(n+1) = \frac{w_i(n) + \eta y(n) x_i(n)}{\left(\sum_{i=1}^{m} [w_i(n) + \eta y(n) x_i(n)]^2\right)^{1/2}}. \tag{38}$$

- Assuming that the learning rate η is small, we may expand this equation as a power series in η:

$$w_i(n+1) = w_i(n) + \eta y(n)[x_i(n) - y(n) w_i(n)] + O(\eta^2), \tag{39}$$

where the term $O(\eta^2)$ represents second and highest order effects in η. For small η we can ignore it and write:

$$w_i(n+1) = w_i(n) + \eta y(n)[x_i(n) - y(n)w_i(n)]. \tag{40}$$

- $y(n)x_i(n)$ is the usual Hebbian rule and the term $-y(n)w_i(n)$ is the negative feedback.
- We might rewrite the learning rule as:

$$w_i(n+1) = w_i(n) + \eta y(n)x_i'(n), \tag{41}$$

where:

$$x_i'(n) = x_i(n) - y(n)w_i(n). \tag{42}$$

- A matrix formulation might be written as:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta\left[\mathbf{x}(n)\mathbf{x}^T\mathbf{w}(n) - \mathbf{w}^T(n)\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{w}(n)\right] \tag{43}$$

7 - Properties of the Hebbian-Based Maximum Eigenfilter

1. The variance of the model output approches the largest eigenvalue of the correlation matrix:

$$\lim_{n \to \infty} \sigma^2(n) = \lambda_1. \tag{44}$$

2. The synaptic weight vector of the model approaches the associated eigenvector, as shown by:

$$\lim_{n \to \infty} \mathbf{w}(n) = \mathbf{q}_1, \tag{45}$$

with

$$\lim_{n \to \infty} \|\mathbf{w}(n)\| = 1. \tag{46}$$

8 - Hebbian-Based Principal Component Analysis

- The Hebbian based maximum eigenfilter extracts only the first principal component of the input.
- However this can be expanded to extract the first l principal components as ilustrated in the following Figure.

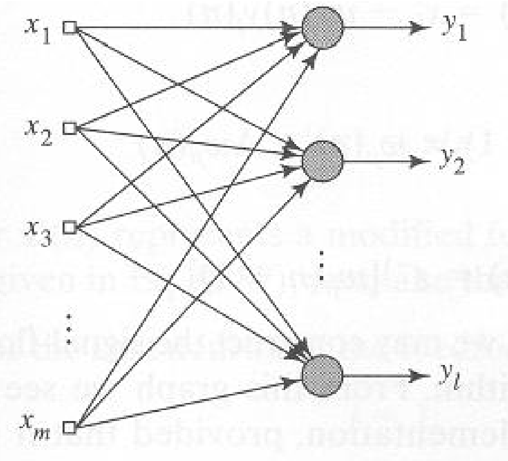


FIGURE 8.6 Feedforward network with a single layer of computation nodes.

- The learning algorithm could be summarised as:
  1. Initialise the synaptic weights of the network, w_{ji}, to small random values at time $n = 1$. Assign a small learning rate parameter η.

2. For $n = 1$, $j = 1, 2, \ldots, l$ and $i = 1, 2, \ldots, m$ compute:

$$y_j(n) = \sum_{i=1}^{m} w_{ji}(n) x_i(n), \tag{47}$$

$$\Delta w_{ji}(n) == \eta \left[y_j(n) x_i(n) - y_j(n) \sum_{k=1}^{j} w_{ki}(n) y_k(n) \right], \tag{48}$$

   where $x_i(n)$ is the i-th component of the m-by-1 input vector $\mathbf{x}(n)$ and l is the desired number of components.

3. Increment n by 1, go to Step 2, and continue until the synaptic weights w_{ji} reach their steady state values.

9 - An example: image coding (1)

1. Fig. a-) Original image with 256x256 pixels, 256 gray levels.
2. The image was coded using a linear feedforward network with a single layer of 8 neurons each with 64 inputs.
3. To train the network 8x8 nonoverlapping blocks of the image were used.
4. The experiment was performed with 2000 scans of the picture and a small learning rate $\eta = 10^{-4}$.
5. Fig. b-) shows the 8x8 masks representing the synaptic weights learned by the network. Excitatory (positive) synapses are white, inhibitory (negative) synapses are black, gray indicates zero weights.
6. Fig. c-) shows the reconstruction using the dominant 8 principal components.

7. Fig. d-) Each coefficient was uniformly quantized with a number of bits approximately proportionally to the logarithm of the variance of that coefficient over the image. Thus the first three masks were assigned 6 bits, th next two masks 4 bits each, the next two masks 3 bits each, and the last mask 2bits. Based on this representation a total of 34 bits were needed to code each 8x8 block of pixels, resulting in a data rate of 0.53 bits per pixel. This results in a compression of 15:1.

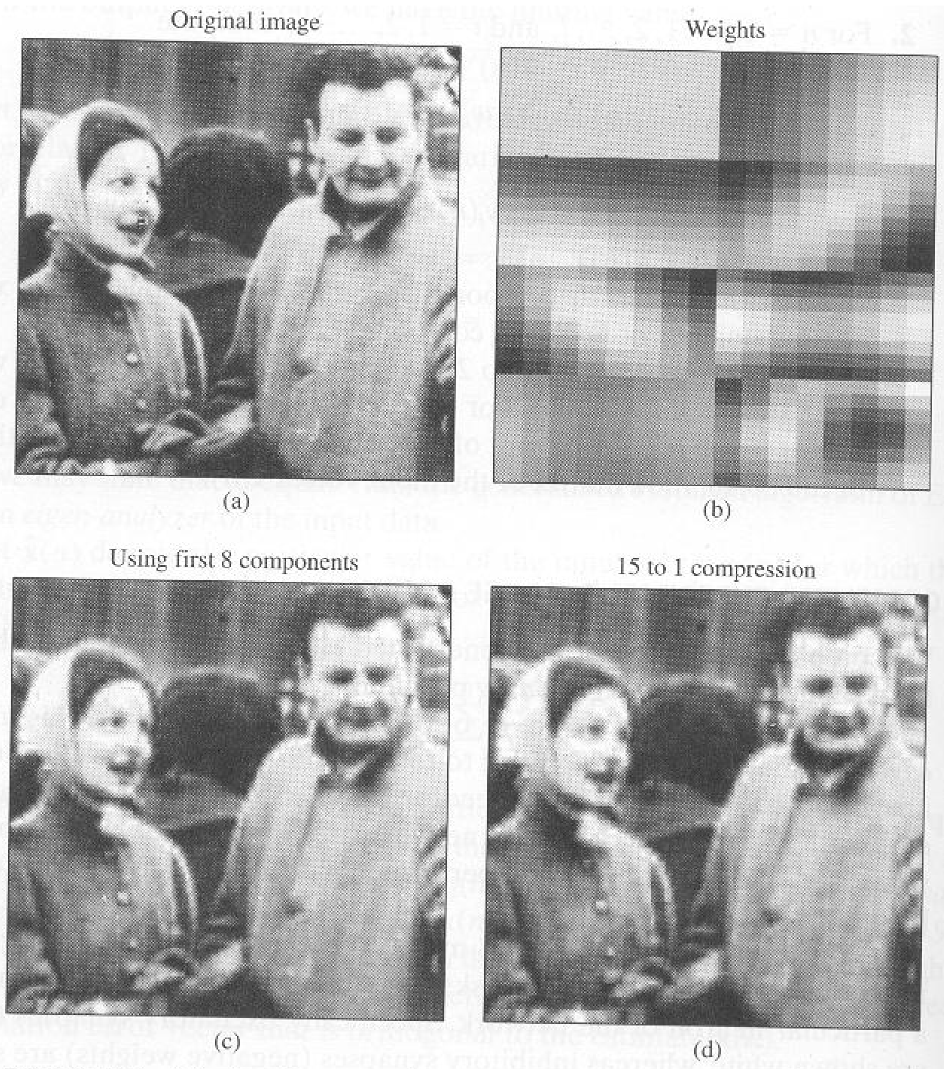


FIGURE 8.9 (a) An image of parents used in the image coding experiment. (b) 8 $\times$ 8 masks representing the synaptic weights learned by the GHA. (c) Reconstructed image of parents obtained using the dominant 8 principal components without quantization. (d) Reconstructed image of parents with 15 to 1 compression ratio using quantization.

10 - An example: image coding (2)

1. Fig. a-) Original image with 256x256 pixels, 256 gray levels.
2. The image was coded using a linear feedforward network with a single layer of 8 neurons each with 64 inputs.
3. Fig. b-) shows the 8x8 masks representing the synaptic weights learned by the network. Excitatory (positive) synapses are white, inhibitory (negative) synapses are black, gray indicates zero weights.
4. Fig. c-) shows the reconstruction using the dominant 8 principal components.
5. Fig. d-) Reconstruction with compression ratio of 22:1.
6. Fig. e-) Reconstruction using principal components of example 1 ans 22:1 compression ratio.

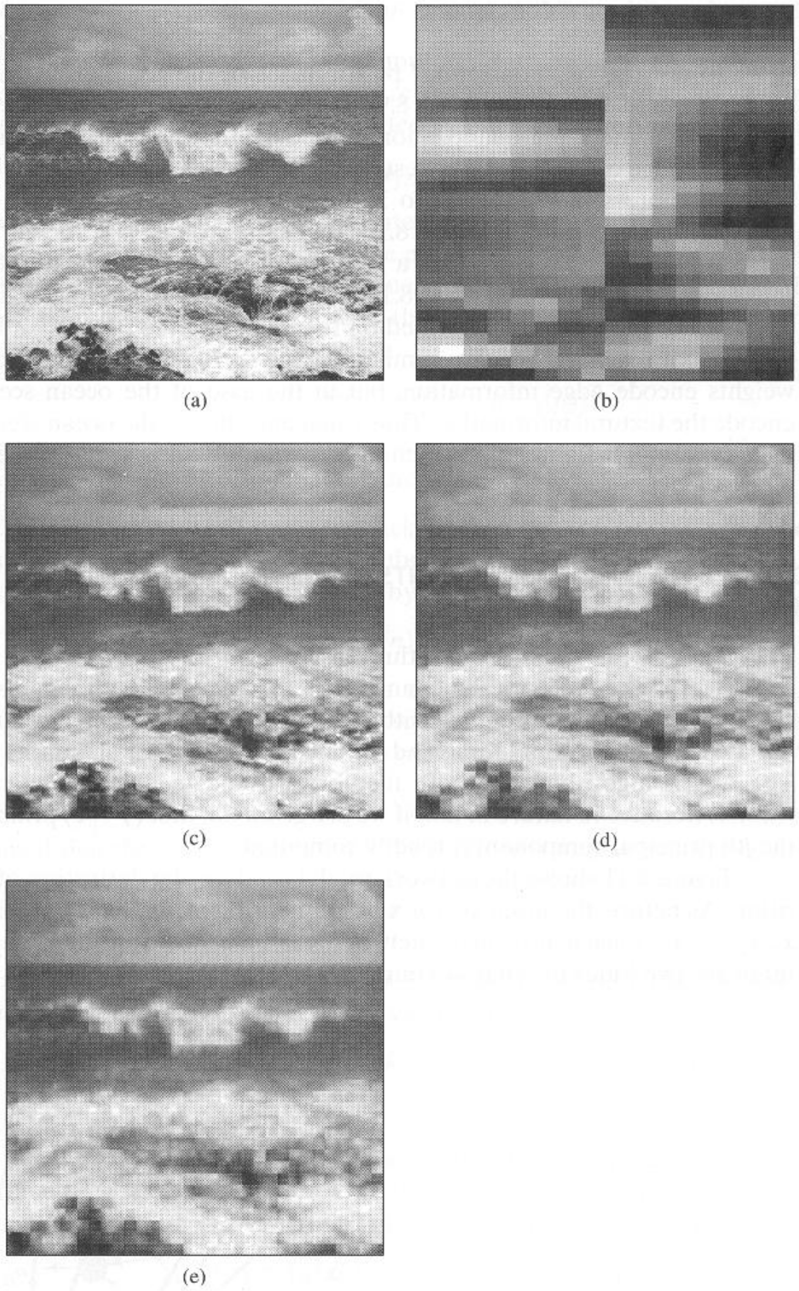


FIGURE 8.10 (a) Image of ocean scene. (b) 8 × 8 masks representing the synpatic weights learned by the GHA algorithm applied to the ocean scene. (c) Reconstructed image of ocean scene using 8 dominant principal components. (d) Reconstructed image of ocean

11 - Adaptive Principal Component Analysis using lateral Inhibition

- The generalised Hebbian algorithm relies on feedforward connections. Diamantaras and Kung (1996) proposed and algorithm named APEX - Adaptive Principal Components Extraction which uses both feedforward and feedback connections.

- The network is ilustrated in the following figure.

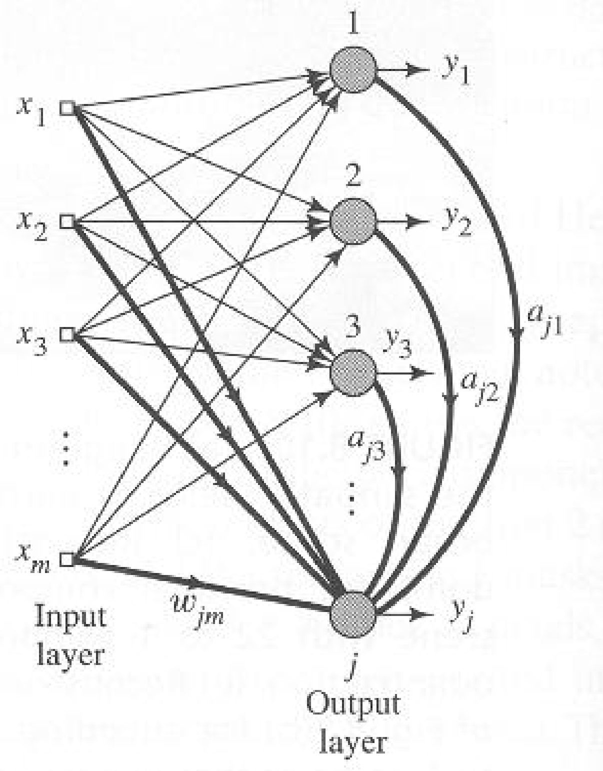


FIGURE 8.11 Network with feedforward and lateral connections for deriving the APEX algorithm.

where the feedforward connections are:

$$\mathbf{w}_j = [w_{j1}, w_{j2}, \ldots, w_{jm}] \,. \tag{49}$$

and the lateral connections are:

$$\mathbf{a}_j = [a_{j1}, a_{j2}, \ldots, a_{jj-1}]^T \tag{50}$$

- The algorithm can be summarised as:

  1. Initialise $\mathbf{w}_j$ and $\mathbf{a}_j$ to small random values at time $n = 1$ where $j = 1, 2, \ldots, m$. assign a small positive value to η.
  2. set $j = 1$ and for $n = 1, 2, \ldots$ compute

$$y_1(n) = \mathbf{w}_1^T(n)\mathbf{x}(n), \tag{51}$$

$$\mathbf{w}_1 = \mathbf{w}_1(n) = \eta[y_1(n)\mathbf{x}(n) - y_1^2(n)\mathbf{w}_1(n)]. \tag{52}$$

  3. Set $j = 2$ and for $n = 1, 2, \ldots$ compute:

$$\mathbf{y}_{j-1}(n) = [y_1(n), \ldots, y_{j-1}(n)]^T \tag{53}$$

$$y_j(n) = \mathbf{w}_j^T(n)\mathbf{x}(n) + \mathbf{a}_j^T(n)\mathbf{y}_{j-1}(n) \tag{54}$$

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \eta[y_j(n)\mathbf{x}(n) - y_j^2(n)\mathbf{w}_j(n)] \tag{55}$$

$$\mathbf{a}_j(n+1) = \mathbf{a}_j(n) - \eta[y_j(n)y_{j-1}(n) + y_j^2(n)\mathbf{a}_j(n)] \tag{56}$$

4. Increment j by 1 go to Step 3 and contiue until $j = m$, where m is the desired number of principal components.