Chapter 8.
Principal-Components Analysis

Neural Networks and Learning Machines
(Haykin)

Lecture Notes of
Self-learning Neural Algorithms

Byoung-Tak Zhang
School of Computer Science and Engineering
Seoul National University

Contents

8.1 Introduction

- Supervised learning
  - Learning from labeled examples
- Semisupervised learning
- Unsupervised learning
  - Learning from examples without a teacher
  - Self-organized learning
    - Neurobiological considerations
    - Locality of learning (immediate local behavior of neurons)
  - Statistical learning theory
    - Mathematical considerations
    - Less emphasis on locality of learning

8.2 Principles of Self-Organization (1/2)

- Principle 1: Self-amplification (self-reinforcement)
  - Synaptic modification self-amplifies by Hebb's postulate of learning
  1) If two neurons of a synapse are activated simultaneously, then synaptic strength is selectively increased.
  2) If two neurons of a synapse are activated asynchronously, then synaptic strength is selectively weakened or eliminated.

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

  - Four key mechanisms of Hebbian synapse
    - Time-dependent mechanism
    - Local mechanism
    - Interactive mechanism
    - Conjunctional or correlational mechanism

8.2 Principles of Self-Organization (2/2)

- Principle 2: Competition
  - Limitation of available resources
  - The most vigorously growing (fittest) synapses or neurons are selected at the expense of the others.
  - Synaptic plasticity (adjustability of a synaptic weight)
- Principle 3: Cooperation
  - Modifications in synaptic weights at the neural level and in neurons at the network level tend to cooperate with each other.
  - Lateral interaction among a group of excited neurons
- Principle 4: Structural information
  - The underlying structure (redundancy) in the input signal is acquired by a self-organizing system
  - Inherent characteristic of the input signal

8.3 Self-organized Feature Analysis

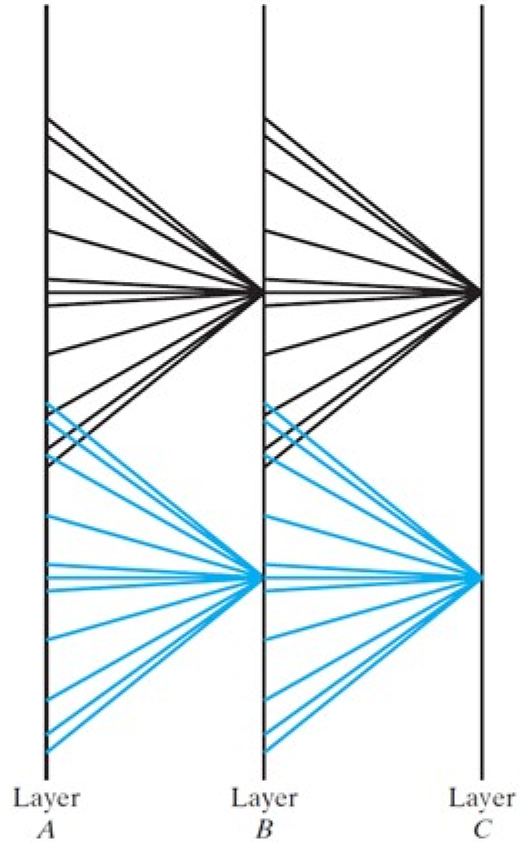


Figure 8.1 Layout of modular self-adaptive Linsker's model, with overlapping receptive fields. Mammalian visual system model.

8.4 Principal-Components Analysis (1/8)

Does there exist an invertible linear transformation **T** such that the truncation of **Tx** is optimum in the mean-square-error sense?

$\mathbf{x}: m$-dimentional vector

$\mathbf{X}: m$-dimentional random vector

$\mathbf{q}: m$-dimentional unit vector

Projection:

$$A = \mathbf{X}^T\mathbf{q} = \mathbf{q}^T\mathbf{X}$$

Variance of A:

$$\sigma^2 = \mathrm{E}[A^2] = \mathrm{E}[(\mathbf{q}^T\mathbf{X})(\mathbf{X}^T\mathbf{q})] = \mathbf{q}^T\mathrm{E}[\mathbf{X}\mathbf{X}^T]\mathbf{q} = \mathbf{q}^T\mathbf{R}\mathbf{q}$$

$\mathbf{R}: m$-by-m correlation matrix

$$\mathbf{R} = \mathrm{E}[\mathbf{X}\mathbf{X}^T]$$

8.4 Principal-Components Analysis (2/8)

$$\psi(\mathbf{q}) = \sigma^2 = \mathbf{q}^T\mathbf{R}\mathbf{q} \qquad **$$

For any small perturbation $\delta\mathbf{q}$:

$$\psi(\mathbf{q}+\delta\mathbf{q}) = \psi(\mathbf{q})$$

. . . .

Introduce a scalar factor λ:

$$\mathbf{R}\mathbf{q} = \lambda\mathbf{q} \qquad \text{(eigenvalue problem)}$$

$\lambda_1, \lambda_2, ..., \lambda_m$: Eigenvalues of $\mathbf{R}$

$\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_m$: Eigenvectors of $\mathbf{R}$

$$\mathbf{R}\mathbf{q}_j = \lambda_j\mathbf{q}_j \qquad j = 1,\ 2,\ ...,\ m$$

$$\lambda_1 > \lambda_2 > \cdots > \lambda_j > \cdots > \lambda_m$$

$$\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_j, ..., \mathbf{q}_m]$$

$$\mathbf{R}\mathbf{Q} = \mathbf{Q}\Lambda$$

$$\mathbf{R}\mathbf{Q} = \mathbf{Q}\Lambda$$

Eigen decomposition:

i) $\mathbf{Q}^T\mathbf{R}\mathbf{Q} = \Lambda$

$$\mathbf{q}_j^T\mathbf{R}\mathbf{q}_j = \begin{cases} \lambda_j, & k = j \\ 0, & k \neq j \end{cases} \qquad **$$

ii) $\mathbf{R} = \mathbf{Q}\Lambda\mathbf{Q}^T = \sum_{i=1}^{m} \lambda_i \mathbf{q}_i \mathbf{q}_i^T$

(spectral theorem)

From **, we see that

$$\psi(\mathbf{q}_j) = \lambda_j \qquad j = 1,2,...,m$$

8.4 Principal-Components Analysis (3/8)

- Summary of the eigenstructure of PCA

1) The eigenvectors of the correlation matrix **R** for the random vector **X** define the unit vectors $\mathbf{q}_j$, representing the principal directions along with the variance probes $\psi(\mathbf{q}_j)$ have their extremal values.
2) The associated eigenvalues define the extremal values of the variance probes $\psi(\mathbf{u}_j)$

8.4 Principal-Components Analysis (4/8)

Data vector $\mathbf{x}$: a realization of $\mathbf{X}$

a: a realization of A

$$a_j = \mathbf{q}_j^T \mathbf{x} = \mathbf{x}^T \mathbf{q}_j \qquad j = 1, 2, \ldots, m$$

a_j: the projections of $\mathbf{x}$ onto principal directions

(principal components)

Reconstruction (synthesis) of the original data $\mathbf{x}$:

$$\mathbf{a} = [a_1, a_2, \ldots, a_m]^T = [\mathbf{x}^T \mathbf{q}_1, \mathbf{x}^T \mathbf{q}_2, \ldots, \mathbf{x}^T \mathbf{q}_m]^T = \mathbf{Q}^T \mathbf{x}$$

$$\mathbf{Qa} = \mathbf{QQ}^T \mathbf{x} = \mathbf{Ix} = \mathbf{x}$$

$$\mathbf{x} = \mathbf{Qa} = \sum_{j=1}^{m} a_j \mathbf{q}_j$$

8.4 Principal-Components Analysis (5/8)

Dimensionality reduction

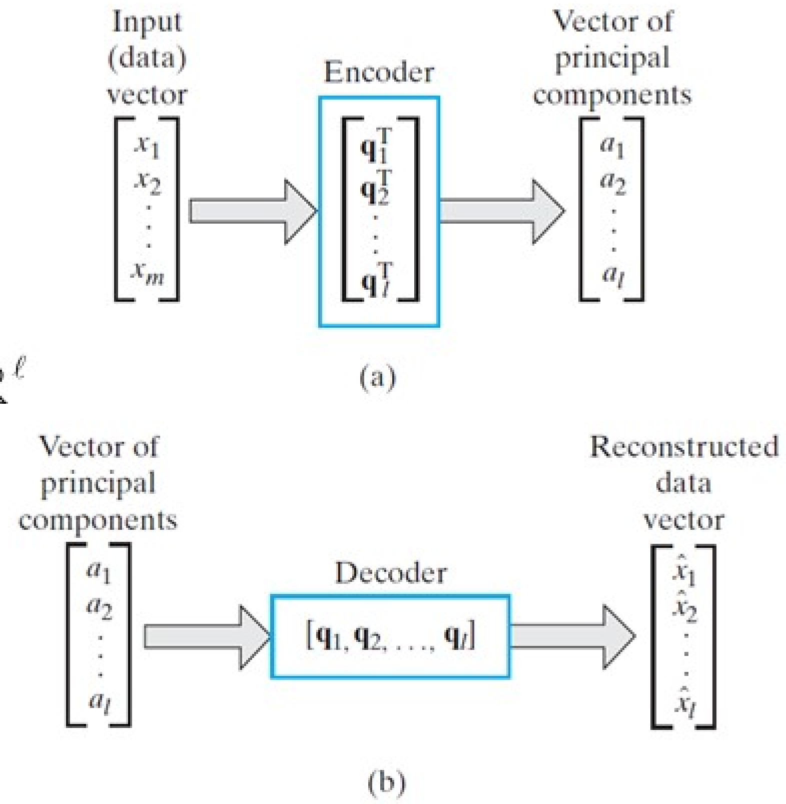
$\lambda_1, \lambda_2, \ldots, \lambda_\ell$: largest ℓ eigenvalues of $\mathbf{R}$

$$\hat{\mathbf{x}} = \sum_{j=1}^{\ell} a_j \mathbf{q}_j = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_\ell] \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_\ell \end{bmatrix}, \quad \ell \le m$$

Encoder for $\mathbf{x}$: linear projection from $\mathbb{R}^m$ to $\mathbb{R}^\ell$

$$\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_\ell \end{bmatrix} = \begin{bmatrix} \mathbf{q}_1^T \\ \mathbf{q}_2^T \\ \vdots \\ \mathbf{q}_\ell^T \end{bmatrix} \mathbf{x}, \qquad \ell \le m$$

Figure 8.2 Two phases of PCA
(a) Encoding, (b) Decoding



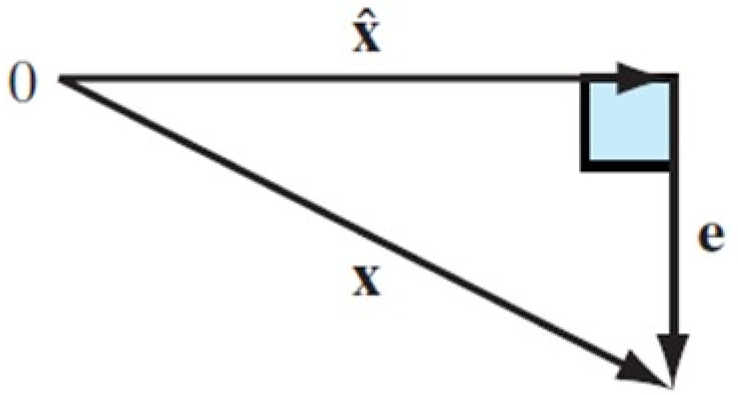
8.4 Principal-Components Analysis (6/8)

Approximation error vector:

$$\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}}$$

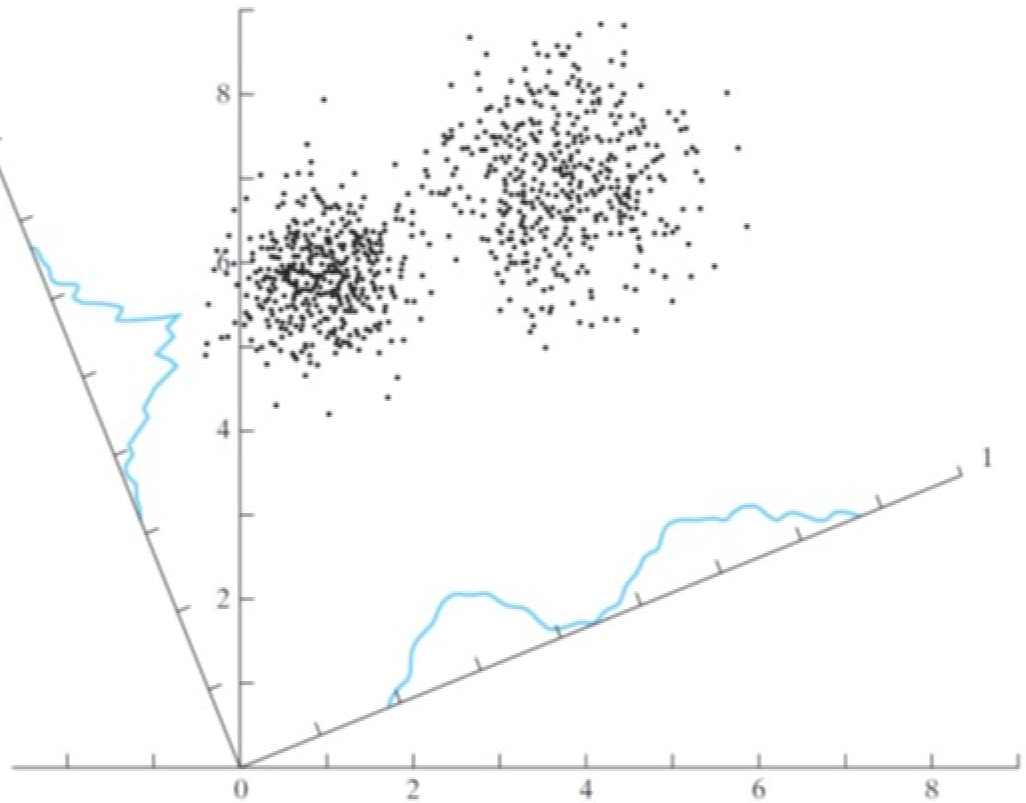
$$\mathbf{e} = \sum_{i=\ell+1}^{m} a_i \mathbf{q}_i$$

Figure 8.3: Relationship between data vector **x**, its reconstructed version $\hat{\mathbf{x}}$ and error vector **e**.



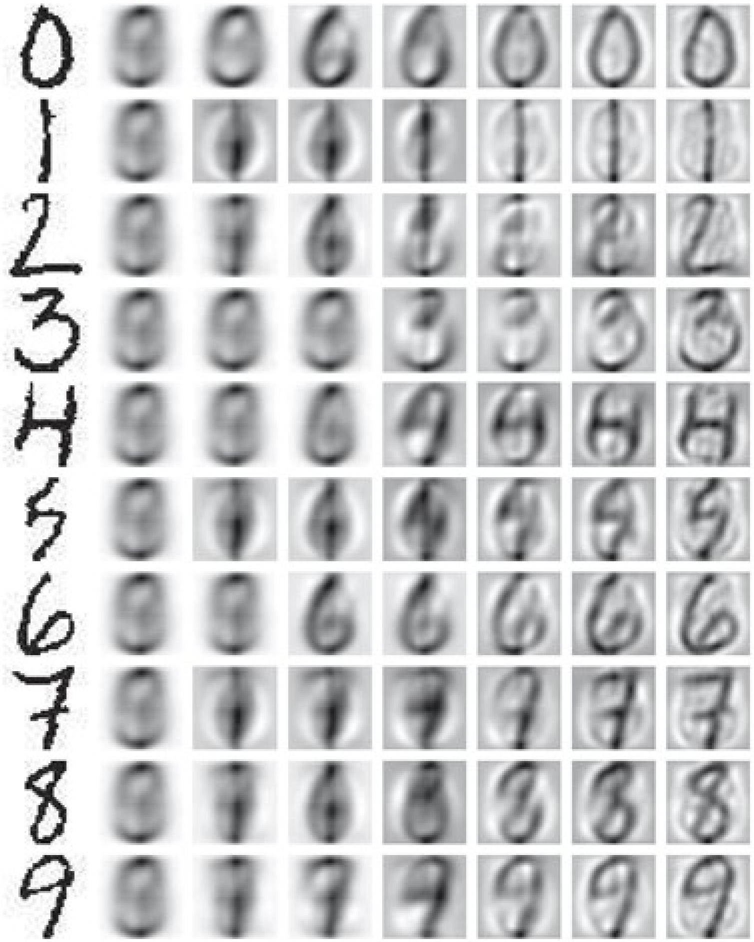
8.4 Principal-Components Analysis (7/8)

Figure 8.4: A cloud of data points. Projection onto Axis 1 has maximum variance and shows bimodal.



8.4 Principal-Components Analysis (8/8)

Figure 8.5: Digital compression of handwritten digits using PCA.



8.5 Hebbian-Based Maximum Eigenfilter (1/4)

Linear neuron with Hebbian adaptation

$$y = \sum_{i=\ell+1}^{m} w_i x_i$$

Synaptic weight w_i varies with time

$$w_i(n+1) = w_i(n) + \eta y(n) x_i(n), \quad i = 1,2,...,m$$

$$\cdots$$

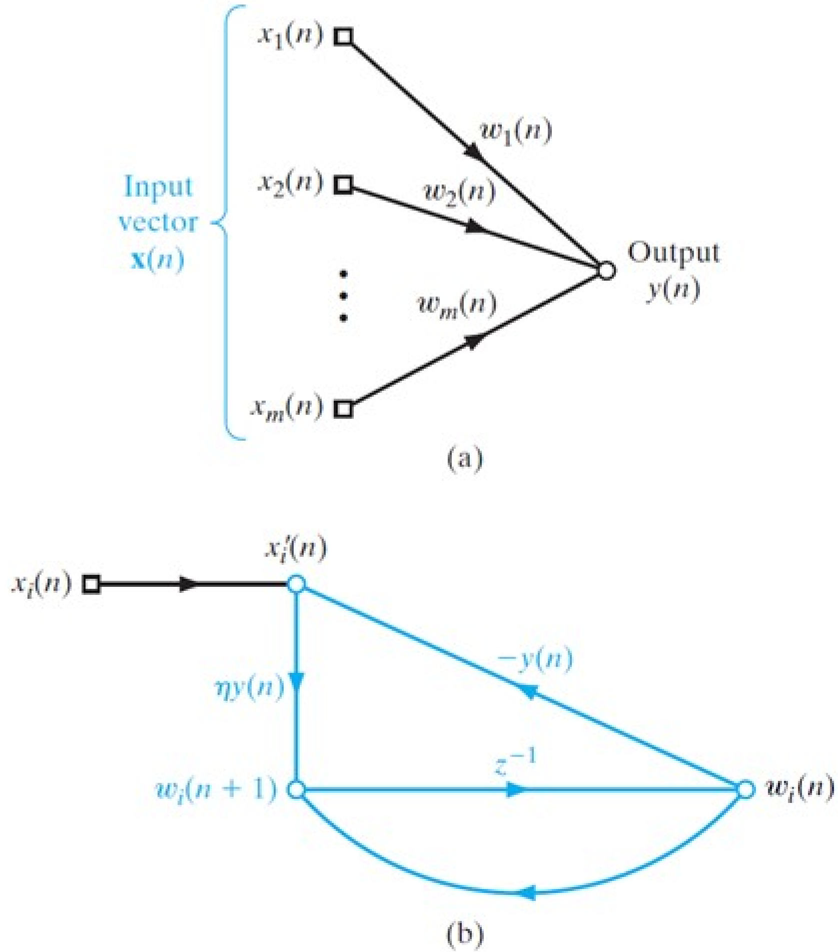
$$w_i(n+1) = w_i(n) + \eta y(n)\left(x_i(n) - y(n) w_i(n)\right)$$

$$x_i'(n) = x_i(n) - y(n) w_i(n)$$

$$w_i(n+1) = w_i(n) + \eta y(n) x_i'(n)$$

8.5 Hebbian-Based Maximum Eigenfilter (2/4)

Figure 8.6: Signal-flow graph representation of maximum eigenfilter



8.5 Hebbian-Based Maximum Eigenfilter (3/4)

Matrix formulation

$$\mathbf{x}(n) = [x_1(n), x_2(n), \ldots, x_m(n)]^T$$

$$\mathbf{w}(n) = [w_1(n), w_2(n), \ldots, w_m(n)]^T$$

$$y(n) = \mathbf{x}^T(n)\mathbf{w}(n) = \mathbf{w}^T(n)\mathbf{x}(n)$$

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta y(n)[\mathbf{x}(n) - y(n)\mathbf{w}(n)]$$

$$= \mathbf{w}(n) + \eta \mathbf{x}^T(n)\mathbf{w}(n)[\mathbf{x}(n) - \mathbf{w}^T(n)\mathbf{x}(n)\mathbf{w}(n)]$$

$$= \mathbf{w}(n) + \eta[\mathbf{x}^T(n)\mathbf{x}(n)\mathbf{w}(n) - \mathbf{w}^T(n)\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{w}(n)\mathbf{w}(n)]$$

8.5 Hebbian-Based Maximum Eigenfilter (4/4)

Aymptotic stability of maximum eigenfilter

$$\mathbf{w}(t) \to \mathbf{q}_1 \qquad \text{as } t \to \infty$$

A single linear neuron governed by the self-organizing learning rule adaptively extracts the first principal component of a stationary input.

$$\mathbf{x}(n) = y(n)\mathbf{q}_1 \qquad \text{for } n \to \infty$$

A Hebbian-based linear neuron with learning rule

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta y(n)[\mathbf{x}(n) - y(n)\mathbf{w}(n)]$$

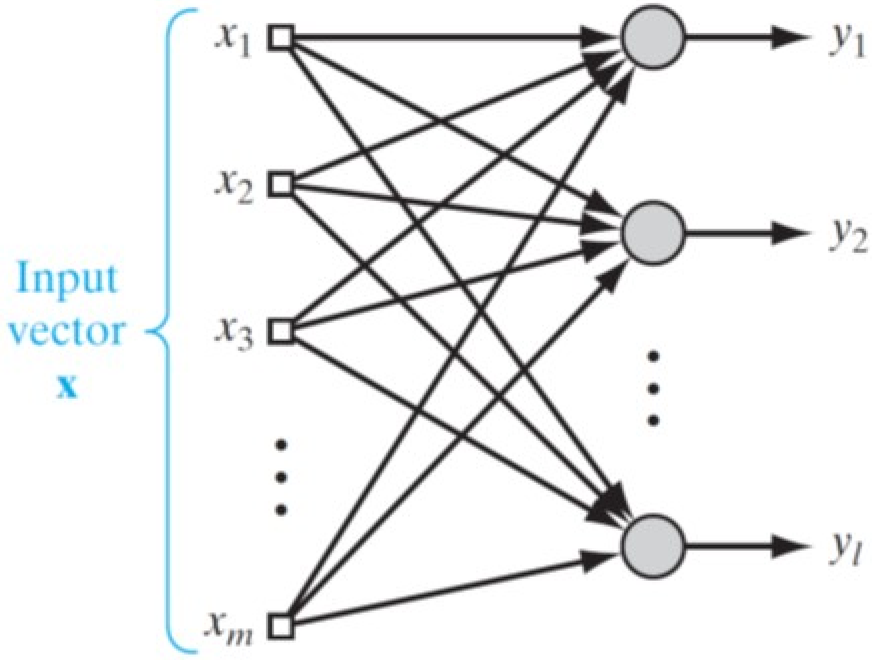
converges with probability 1 to a fixed point:

1) $\lim\limits_{n\to\infty} \sigma^2(n) = \lambda_1$

2) $\lim\limits_{n\to\infty} \mathbf{w}(n) = \mathbf{q}_1$ with $\lim\limits_{n\to\infty} ||\mathbf{w}(n)|| = 1$

8.6 Hebbian-Based PCA (1/3)

Figure 8.7: Feedforward network with a single layer of computational nodes



linear neuron

m inputs, ℓ outputs: $\ell < m$

w_{ji} : synaptic weight from i to j

$$y_j(n) = \sum_{i=1}^{m} w_{ji}(n) x_i(n) \qquad j = 1, 2, ..., \ell$$

8.6 Hebbian-Based PCA (2/3)

Generalized Hebbian Algorithm (GHA)

$$y_j(n) = \sum_{i=1}^{m} w_{ji}(n) x_i(n) \qquad j = 1, 2, \ldots, \ell$$

Weight update rule:

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

Rewriting as

$$\Delta w_{ji}(n) = \eta y_j(n) \left[x_i'(n) - w_{ji}(n) y_j(n) \right]$$

$$x_i'(n) = x_i(n) - \sum_{k=1}^{j-1} w_{ki}(n) y_k(n)$$

8.6 Hebbian-Based PCA (3/3)

$$\Delta w_{ji}(n) = \eta y_j(n) x_i''(n)$$

$$x_i''(n) = x_i'(n) - w_{ji}(n) y_j(n)$$

$$w_{ji}(n+1) = w_{ji}(n) + \Delta w_{ji}(n)$$

$$w_{ji}(n) = z^{-1}[w_{ji}(n+1)]$$

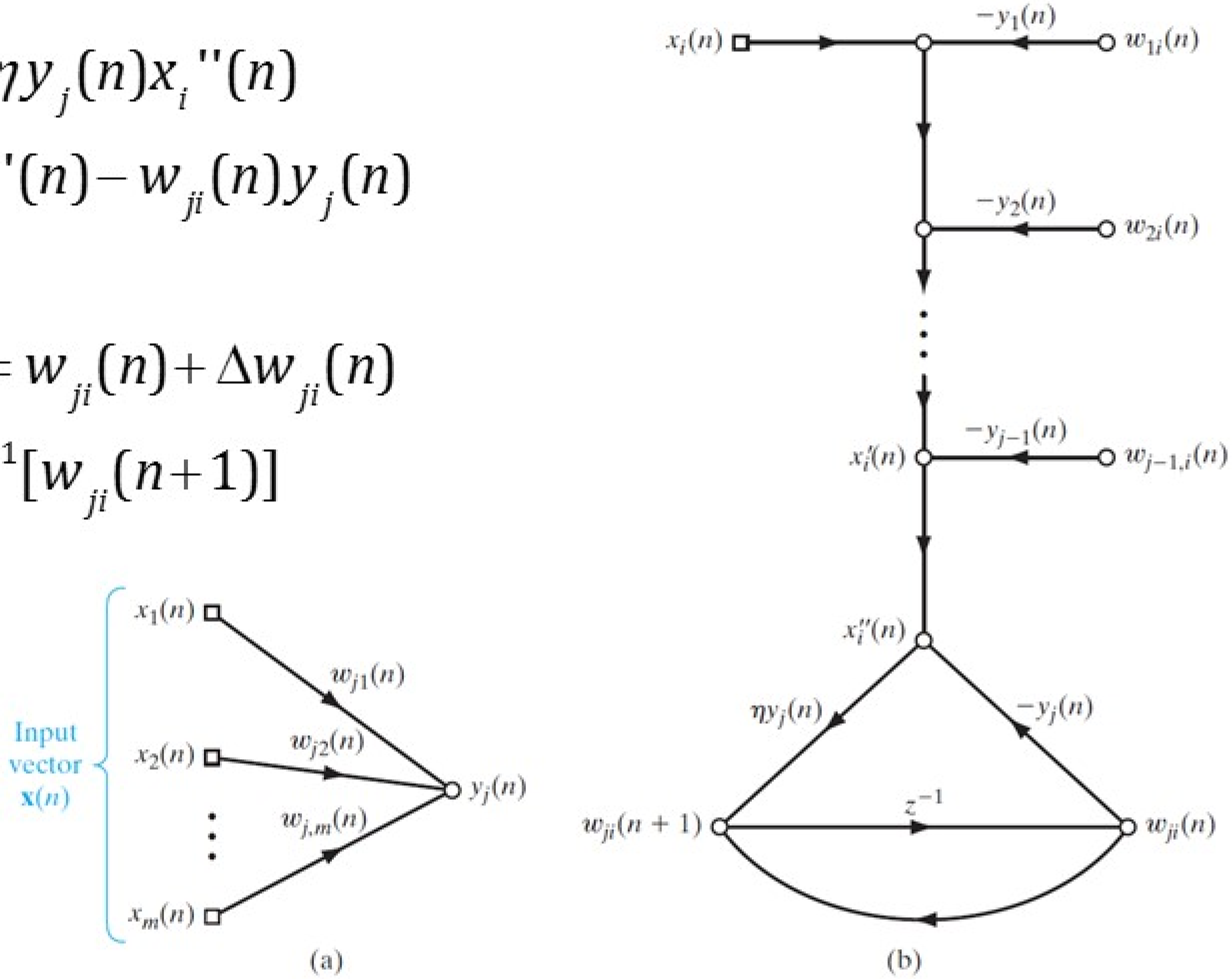
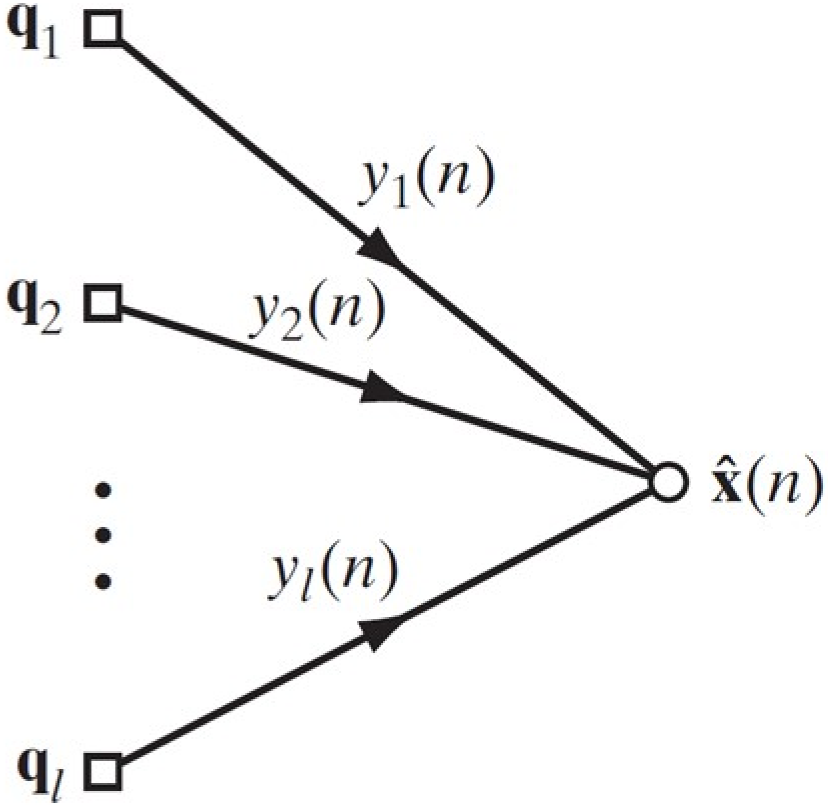


Figure 8.8: Signal-flow graph of GHA

8.7 Case Study: Image Decoding (1/3)

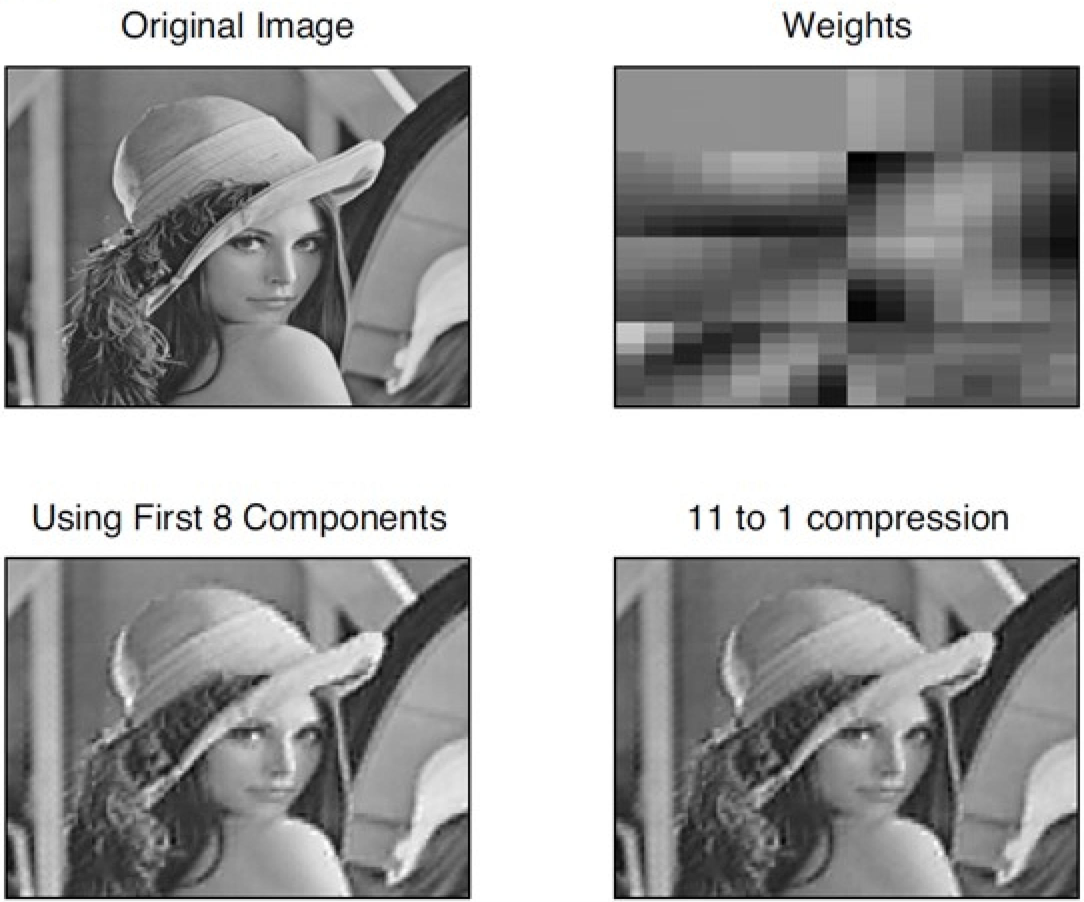
Figure 8.9: Signal-flow graph representation of how the reconstructed vector x^ is computed in the GHA.



$$\hat{\mathbf{x}}(n) = \sum_{k=1}^{\ell} y_k(n)\mathbf{q}_k$$

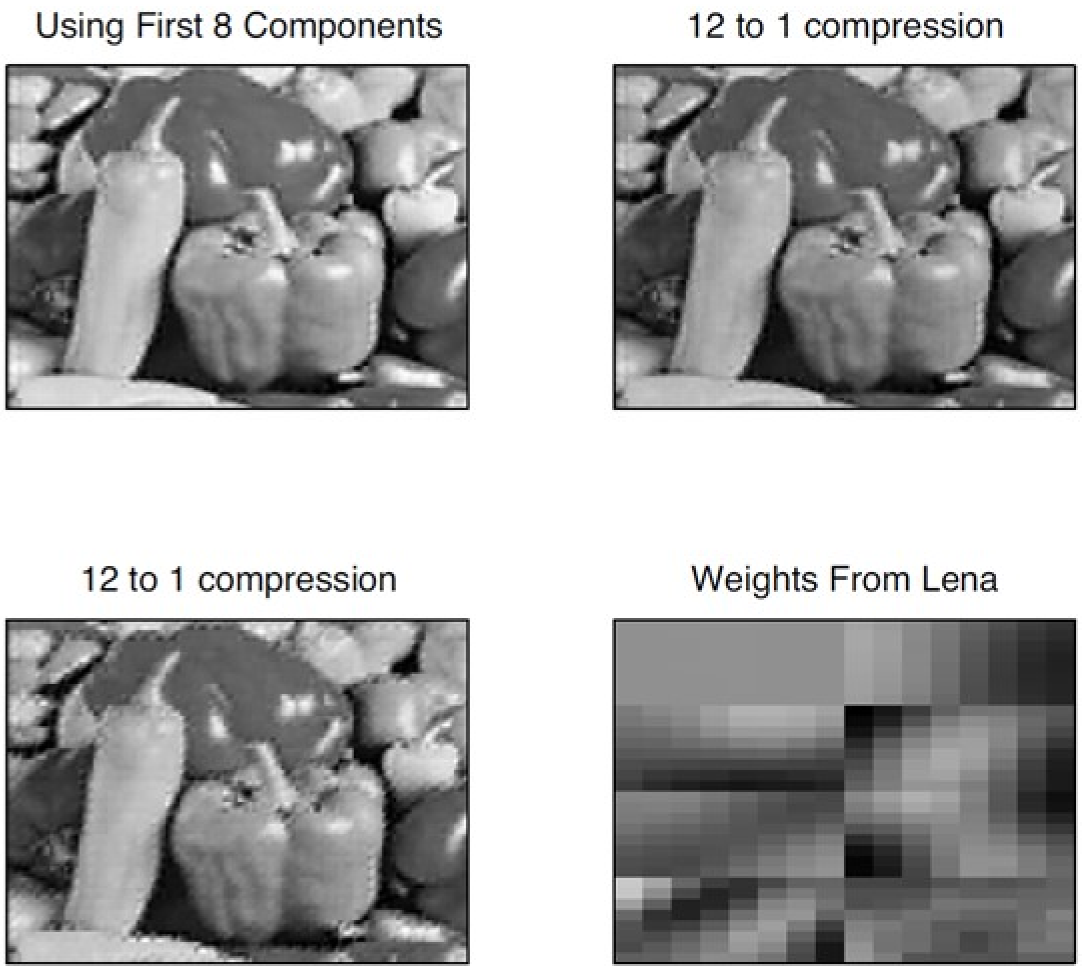
8.7 Case Study: Image Decoding (2/3)

Figure 8.10: (a) An image of Lena used in the image-coding experiment. (b) 8 x 8 masks representing the synaptic weights learned by the GHA. (c) Reconstructed image of Lena obtained using the dominant 8 principal components without quantization. (d) Reconstructed image of Lena with an 11-to-1 compression ratio using quantization



8.7 Case Study: Image Decoding (3/3)

Figure 8.11: Image of peppers



Summary and Discussion

- PCA = dimensionality reduction = compression
- Generalized Hebbian algorithm (GHA) = Neural algorithm for PCA
- Dimensionality reduction
  1) Representation of data
  2) Reconstruction of data
- Two views of unsupervised learning
  1) Bottom-up view
  2) Top-down view
- Nonlinear PCA methods
  1) Hebbian networks
  2) Replicator networks or autoencoders
  3) Principal curves
  4) Kernel PCA

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_\ell \end{bmatrix} = \begin{bmatrix} \mathbf{q}_1^T \\ \mathbf{q}_2^T \\ \vdots \\ \mathbf{q}_\ell^T \end{bmatrix} \mathbf{x}, \quad \ell \le m$$

$$\hat{\mathbf{x}} = \sum_{j=1}^{\ell} a_j \mathbf{q}_j = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_\ell] \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_\ell \end{bmatrix}, \quad \ell \le m$$

$$\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}}$$

$$\mathbf{e} = \sum_{i=\ell+1}^{m} a_i \mathbf{q}_i$$