Feature Discovery and Hebbian Learning

Todd K. Leen

Technical Report No. CSE 90-005
March, 1990

Department of Computer Science and Engineering
Oregon Graduate Institute of Science & Technology
19600 N.W. Von Neumann Drive
Beaverton, OR 97006

Feature Discovery and Hebbian Learning

Todd Leen
Department of Computer Science & Engineering
Oregon Graduate Institute of Science & Technology
19600 N.W. von Neumann Drive
Beaverton, OR 97006

March 20, 1990

Preface

This report is primarily pedagogical in purpose and content (there are even a few exercises included). Although a few novel derivations and points of view are developed, the report is essentially an exposition of ideas which have already appeared in the literature. The notes contained here first appeared as a set of lectures on feature detection given as part of a graduate course on the mathematics of neural, networks taught at OGI during the fall of 1989.

I offer thanks to my students for their questions, which helped to clarify the exposition, and to Mr. Vince Weatherill for turning my handwritten lecture notes and diagrams into a readable monograph.

Abstract

Real-world pattern recognition problems often involve data spaces of exceedingly large dimension. In order to ease the computational burden on pattern-classifier algorithms, the naive, high-dimensional features can be encoded in a lower dimensional set.

Principal component analysis (PCA) provides a means for this encoding. More importantly, PCA can be implemented in neural networks that use local, Hebbian learning rules to change synaptic strengths. Thus, data encoding can be accomplished in the same environment that serves the computational needs of the classifier. This report discusses PCA from statistical and geometric points of view, and its implementation in neural networks.

1 Feature Discovery by Hebbian Learning

1.1 Introduction

Data spaces can often be described by *feature spaces* which are more compact than a naive representation. Put another way, feature spaces can often be more compact than the data that they describe. For example, in speech recognition one may calculate, from a digitized signal, 256 Fourier coefficients for a signal window of 10 ms. Retaining only the power coefficients leaves over 12000 spectral features for one second of speech. Examining a much smaller problem — three spectral slices of 64 coefficients each may serve to describe a vowel extracted from continuous speech. A backpropagation network trained to distinguish 12 different vowel sounds using these 192 spectral coefficients may reasonably require on the order of a dozen hidden nodes. This results in a network with over 2400 weights to be trained. Since supervised learning algorithms can be slow to train it is worthwhile to consider *compact* data representations.

Training time is only one of the motivations. Hardware implementation costs are known to scale cubically with node fan-in [1], and recent results in the theory of learning indicate that the number of training examples required for accurate generalization can scale approximately linearly in the number of connections [2]. Thus learning time, hardware implementation costs and the number of training examples required for accurate generalization, increase polynomially with the number of network connections. This constitutes a scaling catastrophe which prohibits the solution of large, complex perceptual and cognitive problems with homogeneous networks driven with raw data.

All three of these problems can be alleviated by reducing the size of the input layer i.e., the dimension of the input space. One would like to develop representations that capture the information in the original data in an economical fashion. These lectures describe one class of techniques which map naturally to neural networks, have a strong theoretical foundation, and may be related to biological feature encoding.

1.2 Dimension Reduction and Linear Mappings

Our discussion will be limited to *linear* networks. We consider a single layer of feedforward weights with N inputs x, M outputs y and a weight matrix w as shown in Fig. 1. The input and output, are vectors in R^N and R^M respectively, with $N > M$. The weight matrix w is an M row by N column matrix. The input vectors are taken to be random variables drawn from

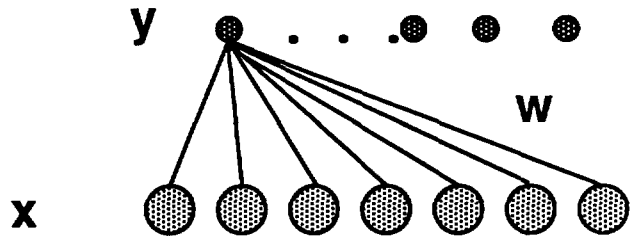


Figure 1: Single layer of feedforward weights.

some probability distribution which is presumed to be stationary [3]. (Each input vector is considered to be a pattern in the input space, drawn from an ensemble of patterns arising from the probability distribution). The output vector is given by

$$y = wx$$

or in component notation

$$y_i = \sum_j w_{ij}\ x_j\ .$$

We want to construct w to build up as faithful a representation as possible, given the constraints imposed by the dimension reduction. Specifically, $w : R^N \rightarrow R^M$ maps an N dimensional space into an M dimensional space with $N > M$. Thus the rowspace of w spans *at most* R^M, and w has a null space or kernal. The dimension of this null space is at least $N - M$. This is depicted schematically in Fig. 2. The trick then is to design the null space of w so that the directions spanning $NullSp(w)$ are *unimportant* for distinguishing between vectors in the input ensemble. Suppose that the input space is R^3 and the output space is R^2 as in Fig. 2. If the data lie on a plane in R^3 (Fig. 3), then it is clear that we really need only 2 numbers to specify each point — the third coordinate is highly correlated with the 1st two. Given the cloud of data points, we want to find this plane, and choose a set of coordinate axes on it. With a properly chosen w, the components along the axes in the plane will be given by the output nodes in the network $y = wx$.

This geometric construction provides a hint: design w so that its null space is spanned by directions in R^N along which the scatter of the data is

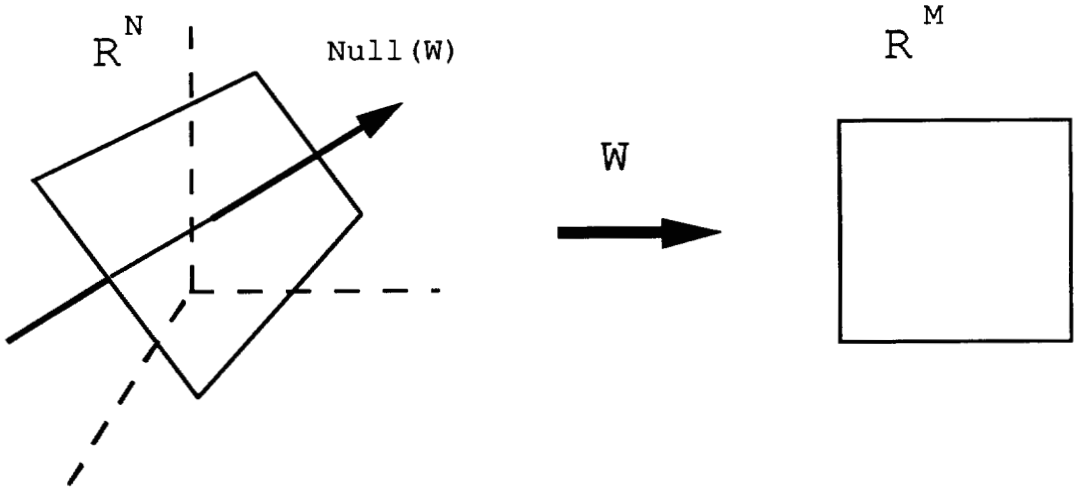


Figure 2: Dimension reduction

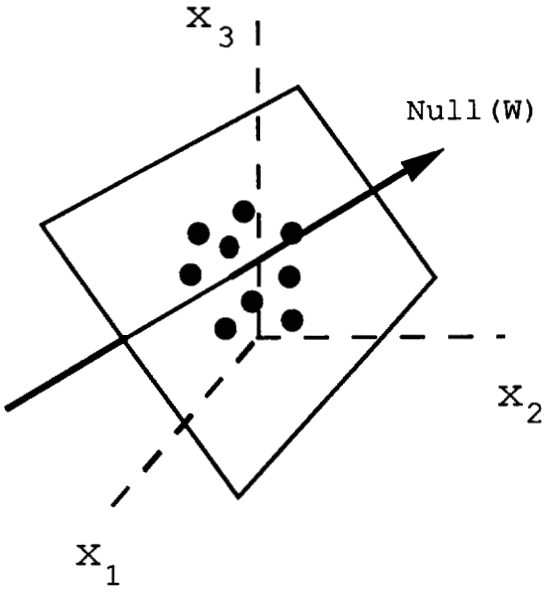


Figure 3: Data constrained to a plane in R^3

minimal. We need some statistical apparatus to develop this notion more completely.

2 Continuous Probability Distributions

2.1 Moments

Let $x \epsilon R^N$ be a random variable with the probability density $\rho(x)$. This density is taken to be normalized

$$\int d^N x \;\; \rho(x) = \int dx_1 dx_2 \cdots \rho(x_1, x_2, \ldots) = 1$$

corresponding to the certainty that x takes on some value in R^N.

Associated wih $\rho(x)$ are a set of *moments*:

$$\begin{aligned} 0^{th} \text{ moment} \;\; 1 &= \int dx^N \rho(x) \\ 1^{st} \text{ moment} \;\; E(x) &= < x > = \int dx^N \;\; x\rho(x) \\ 2^{nd} \text{ moment} \;\; Q_{ij} &= E(x_i x_j) =< x_i x_j >= \int dx^N \;\; x_i x_j \rho(x) \end{aligned}$$

The first moment is called the mean, the second is the correlation matrix (or the auto-correlation matrix) [3]. Notice that the auto-correlation is a symmetric, real $N \times N$ matrix. Consequently, its eigenvalues are all real, and the eigenvectors form an orthonormal basis spanning R^N . Higher order moments are similarly defined, however we will only require up to the 2nd moment in what follows. Associated with the correlation matrix, is the covariance matrix, defined as

$$\begin{aligned} \Sigma_{ij} &= < (x_i - < x_i >)(x_j - < x_j >) > \\ &= \int dx^N \;\; \rho(x)(x_i - < x_i >)(x_j - < x_j >) \end{aligned}$$

The correlation and covariance matrices are related by:

$$\begin{aligned} \Sigma_{ij} &= < x_i x_j > - < x_i >< x_j > \\ &\equiv Q_{ij} - < x_i >< x_j > \end{aligned}$$

Finally, the correlation matrix can be represented as an outer product, in the form

$$Q = \int dx^N \rho(x) x x^T$$

where x^T denotes the transpose of the column vector x. The covariance matrix Σ can be similarly represented.

Definition 2.1 *Two random variables, x_1 and x_2, are said to be* independent *if their joint probability density factors into a product of two probability density functions, one for each of the two variables;*

$$\rho(x_1 x_2) = \rho_1(x_1)\ \rho_1(x_2)\ .$$

Claim: Let $x_1, \ldots, x_N$ be independent, *zero-mean* random variables. Then the correlation, Q, is diagonal with the variances of each of the x_i along the diagonal.

Proof 2.1

$$\begin{aligned} Q_{ij} &\equiv \int dx_1, \ldots, dx_N\ x_i x_j\ \rho(x_1, \ldots, x_N) \\ &= \int dx_1\ \rho(x_1) \int dx_2\ \rho(x_2) \cdots \int dx_i\ x_i\ \rho(x_i) \cdots \int dx_j\ x_j\ \rho(x_j) \cdots \\ &= \int dx_i\ x_i\ \rho(x_i) \int dx_j\ x_j\ \rho(x_j) \\ &= < x_i >< x_j >= 0 \quad for\ i \neq j \end{aligned}$$

While for $i = j$ we have:

$$\begin{aligned} Q_{ii} &= \int dx_1, \cdots, dx_N\ x_i^2\ \rho(x_i)\ \rho(x_1, \cdots, x_{i-1}, x_{i+1}, \ldots, x_N) \\ &= \int dx_i\ x_i^2\ \rho(x_i) \equiv \sigma_{x_i}^2 \end{aligned}$$

If the random variables do not have zero mean, then a similar statement holds with the covariance matrix substituted for the correlation (the reader should verify this). Notice that the converse is *not true* i.e., diagonal Q does not imply statistical independence. Random variables for which the correlation matrix is diagonal are called *un-correlated.*

2.2 Multivariate Gaussian Distribution

For N-dimensional data with zero mean [1], the Gaussian probability density function is

$$\rho(x) = \frac{1}{(2\pi)^{N/2}\,(\det Q)^{1/2}}\ \exp\ -\frac{1}{2}(x^T Q^{-1} x)\,. \tag{1}$$

(See 2.2 which represents graphically the contours and $\rho(x)$ for Q^{-1} in 1).

The argument of the exponential

$$x^T Q^{-1} x$$

is a non-negative quadratic form.

Example 2.1 *For*

$$Q^{-1} = \begin{pmatrix} \alpha+\beta & \beta-\alpha \\ \beta-\alpha & \alpha+\beta \end{pmatrix} \quad \alpha,\beta > 0$$

the argument of the exponential is

$$\begin{aligned} x^T Q^{-1} x &= (x_1 x_2)\begin{bmatrix} \alpha+\beta & \beta-\alpha \\ \beta-\alpha & \alpha+\beta \end{bmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\ &= (x_1 x_2)\begin{pmatrix} (\alpha+\beta)x_1 + (\alpha+\beta)x_2 \\ (\beta-\alpha)x_1 + (\alpha+\beta)x_2 \end{pmatrix} \\ &= (\alpha+\beta)(x_1^2+x_2^2) + 2(\beta-\alpha)x_1 x_2 \\ &= \alpha(x_1-x_2)^2 + \beta(x_1+x_2)^2 \ \geq 0 \end{aligned}$$

The eigenvalues of Q are given by solving the characteristic equation

$$\det\,(Q^{-1} - \lambda 1) = 0 \quad \Rightarrow \quad \lambda = 2\alpha, 2\beta > 0\,.$$

The density $\rho(x)$ is constant along surfaces for which

$$x^T Q^{-1} x = C\ \ (\text{const.}>0) \tag{2}$$

Since any real quadratic form can be diagonalized by an orthogonal transformation, there is a set of coordinates for which (2) reads:

$$\tilde{x}^T\ \tilde{Q}^{-1}\ \tilde{x} = \lambda_1^{-1}\ \tilde{x}_1^2 + \lambda_2^{-1}\ \tilde{x}_2^2, \ldots, \lambda_N^{-1}\ \tilde{x}_N^2 = C$$

[1] If the data has mean x_0, the analagous form is
$\rho(x) = \frac{1}{(2\pi)^{N/2}\,(\det\Sigma)^{1/2}}\ \exp\ -\frac{1}{2}(x-x_0)^T\,\Sigma^{-1}(x-x_0)$

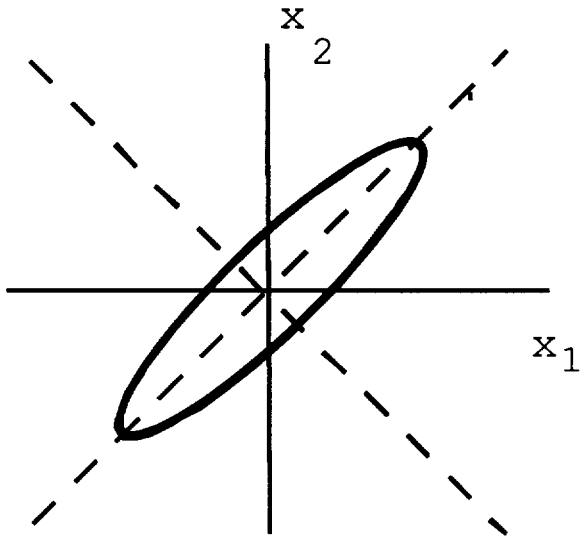


Figure 4: Constant-density ellipsoid.

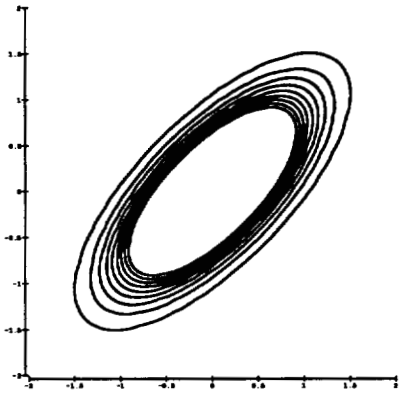
where $\lambda_1^{-1}, \ldots, \lambda_N^{-1}$ are the (positive) eigenvalues of Q^{-1}. This last equation describes an *ellipsoid* (shown in Fig. 4) on which the density $\rho(x)$ is constant. The eigenvalues λ_i are proportional to the square of the lengths of the semi-major axes of this ellipsoid.

> *Note:* There is a family of such ellipsoids, one for each value of C, in (2). Furthermore, the eigenvectors of Q^{-1} are along the principal axes of these ellipsoids.

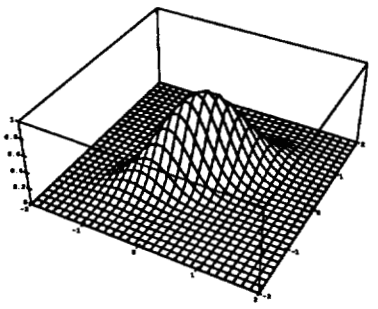
Exercise 2.1 *Show that if $\overline{v}$ is an eigenvector of Q^{-1} with eigenvalue λ^{-1}, then $\overline{v}$ is an eigenvector of Q with eigenvalue λ.*

Summary:

1. The surfaces of constant density of (1) are ellipsoids.
2. The eigenvectors of Q^{-1} (and of Q) are along the principal axes of the constant-density ellipsoids.
3. The corresponding eigenvalues of Q, $\lambda_1, \lambda_2, \ldots, \lambda_N > 0$ are proportional to the squared lengths of the principal axes of the constant-density ellipsoids.



(A)



(B)

Figure 5: Contours (A) and 3D plot (B) of $\rho(x)$ (not properly normalized here) for Q^{-1} of example 1.

What follows is independent of the underlying probability density (p.d.f.) — but we will use Q, Σ and their estimates based on samples from the population.

Suppose we have a p.d.f., $\rho(x)$ with zero mean and *known* correlation Q. What is the variance of x along a *particular unit direction* v ?

To find the variance of x along v_1, project x onto v and and compute the variance of the resulting scalar:

$$\begin{aligned}\sigma_v^2 &\equiv E[(v\cdot x)(v\cdot x)] = E[(v^T x)(x^T v)] \\ &= E[v^T(xx^T)v] = v^T E(xx^T)v\end{aligned}$$

where the last equality follows because v is not a random vector. Since $x_0 = 0$, by assumption:

$$\sigma_v^2 = v^T E[(x-x_0)(x-x_0)^T]v \equiv v^T\Sigma v$$

If $v^T v \neq 1$, then:

$$\sigma_v^2 = \frac{v^T\Sigma v}{v^T v} \tag{3}$$

If v is a unit eigenvector of Σ with eigenvalues $\lambda\bar{v}$, then $v^T\Sigma v = \lambda v$. Thus, the eigenvalues of Σ give the variance along the eigenvectors.

2.3 Estimating Σ and Q from a Sample

The sample mean is given by

$$\mu = \frac{1}{M}\sum_{I=1}^{M} x^I \quad (M \text{ sample vectors } x^I)\,. \tag{4}$$

As $M \to \infty$, $\mu \to x_o$ the population mean.

The sample covariance is given by

$$R = \frac{1}{M}\sum_{I=1}^{M}(x^I-\mu)(x^I-\mu)^T$$

or in components

$$R_{ij} = \frac{1}{M}\sum_{I=1}^{M}(x_i^I-\mu_i)(x_j^I-\mu_j)\,. \tag{5}$$

As $M \to \infty$, $R_{ij} \to \Sigma_{ij} = E([x_i - x_{io}][x_j - x_{jo}])$.
The sample correlation matrix is given by

$$Q = \frac{1}{M} \sum_{I=1}^{M} x^I x^{I\,T} ,$$

or in components

$$Q_{ij} = \frac{1}{M} \sum_{I=1}^{M} x_i^I x_j^I . \tag{6}$$

As $M \to \infty$, the sample correlation approaches the population correlation $M \to \infty$, $Q \to E(xx^T)$.

Notice that the sample covariance and correlation are formed by a sum of projection operators — thus for $x \in R^N$ with $M < N$ samples, Q or R will be *singular* (this is likely to occur in image processing). Similarly, if the sample consists of very many vectors, but only $M < N$ are linearly independent, then Q or R will be singular.

Exercise 2.2 *Show that for $x \epsilon R^N$ with $M < N$ linearly independent samples, Q as estimated by 6 has an $(N - M)$ dimensional null space.*

3 Principal Components

3.1 Maximum Variance Directions

As per our geometric intuition from 1.2, let us construct
$w : R^N \rightarrow R^M$, so that its null space is spanned by vectors in R^N along which the scatter of the data is minimal. We need to locate these directions.

We start by finding the directions $\bar{v}$, along which σ_v^2 has extrema. To find these directions, expand $\bar{v}$ in the orthonormal basis of eigenvectors $\bar{e}_i$ of Σ $(\Sigma\bar{e}_i = \lambda_i, \quad \bar{e}_i \cdot \bar{e}_j = \delta_{ij})$.

$$\bar{v} = \sum_{i=1}^{N} \alpha_i \bar{e}_i$$

Then, from 3,

$$\begin{aligned} \sigma_v^2 &\equiv \frac{\bar{v} \cdot \Sigma\bar{v}}{(\bar{v} \cdot \bar{v})} = \frac{(\sum_i \alpha_i \bar{e}_i) \cdot (\sum_j \alpha_j (\Sigma \bar{e}_j))}{(\sum_k \alpha_k \bar{e}_k) \cdot (\sum_k \alpha_k \bar{e}_k)} \\ &= \frac{\sum_j \alpha_j^2 \lambda_j}{\sum_k \alpha_k^2} \end{aligned} \tag{7}$$

To find the critical points of $\sigma_{\bar{v}}^2$ set all of its partial derivatives with respect to α_i equal to zero,

$$\begin{aligned} \frac{\partial \sigma_v^2}{\partial \alpha_i} &= \frac{\partial}{\partial \alpha_i} \left[\frac{\sum_j \alpha_j^2 \lambda_j}{\sum_k \alpha_k^2} \right] \\ &= \left(\frac{1}{\sum_k \alpha_k^2} \right)^2 \left\{ 2\alpha_i \lambda_i (\sum_j \alpha_j^2) - 2\alpha_i (\sum_j \alpha_j^2 \lambda_j) \right\} \\ &= \frac{2\alpha_i}{(\sum_j \alpha_j^2)^2} \left[\sum_j \alpha_j^2 (\lambda_i - \lambda_j) \right] = 0 \end{aligned} \tag{8}$$

The solutions of (8) are given by

$$\alpha_i = 0$$

or

$$\alpha_i = C, \quad \alpha_j = 0 \ \ \forall j \neq i$$

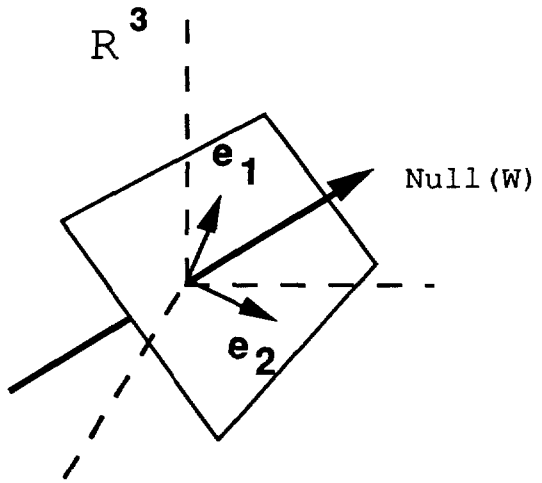


Figure 6: The new coordinates on R^3

.

where C is a nonzero constant. Since we need to satisfy (8) for *all* possible values of i, it is clear that the solutions are obtained for exactly one of the α_i nonzero. Consequently, σ_v^2 has its critical points at

$$\bar{v}_0 = C\bar{e}_k \quad k = 1, \ldots, N. \tag{9}$$

In these critical directions, the variance is given by

$$\sigma_{v_0}^2 = \frac{\bar{v}_0 \cdot \Sigma\, \bar{v}_0}{(\bar{v}_0 \cdot \bar{v}_0)} = \bar{e}_k \cdot \Sigma\, \bar{e}_k = \lambda_k \,. \tag{10}$$

Thus, the variance of the data along a critical direction is given by the corresponding eigenvalue of the covariance.

According to (10), the variance will be least along the eigenvectors of the covariance corresponding to the smallest eigenvalues. Thus, we should choose w so that its null space is spanned by the eigenvectors corresponding to the smallest eigenvalues. This is shown schematically in Fig. 6.

The simplest construction is to take the M rows of w to be the eigenvectors of Σ corresponding to M largest eigenvalaues. If we order the eigenvalues

$$\lambda_1 > \lambda_2 > \lambda_3, \ldots, > \lambda_N$$

and pick

$$w = \begin{pmatrix} \bar{e_1} \\ \bar{e_2} \\ \vdots \\ \bar{e_m} \end{pmatrix} \tag{11}$$

Then the components of $y = wx$, i.e.

$$\begin{aligned} y_1 &= \bar{x} \cdot \bar{e_1} \\ y_2 &= \bar{x} \cdot \bar{e_2} \\ &\vdots \\ y_m &= \bar{x} \cdot \bar{e_m} \end{aligned}$$

are the first m *principal components* of x.

Claim: The principal components are uncorrelated.

Proof 3.1

$$\begin{aligned} E\left[(y_i - y_{0i})(y_j - y_{0j})\right] &= E\left[(\bar{e_i} \cdot (\bar{x} - \bar{x_0}))((\bar{x} - \bar{x_0}) \cdot \bar{e_j})\right] \\ &= E\left[e_i^T (x - x_0)(x - x_0)^T e_j\right] \\ &= e_i^T \; E[(x - x_0)(x - x_0)^T] e_j \\ &= e_i^T \Sigma \, e_j = \lambda_j \delta_{ij} \end{aligned}$$

Note: *Regarding y as a vector given by $y = wx$, the covariance matrix for y is*

$$\begin{aligned} \Sigma^y &\equiv E\left[((y - y_0)(y - y_0)^T\right] = E\left[(w(x - x_0))((x - x_0)^T w)^T\right] \\ &= w \Sigma_x w^T \, . \end{aligned}$$

Thus, under a change of basis, one has

$$x' \equiv Ax$$

and the covariance matrix transforms as

$$\begin{aligned} \Sigma' &\equiv E\left[(x' - x'_0)(x' - x'_0)^T\right] \\ &= E\left[A(x - x_0)(x - x_0)^T A^T\right] \\ &= A \Sigma A^T \, . \end{aligned} \tag{12}$$

This is *not* a similarity transformation! The reader can verify that the correlation matrix $E(xx^T)$ transforms the same way.

3.2 Least Mean Square Error

Principal components calculated with the correlation matrix have the property of minimizing the mean square error of the representation. For $x \epsilon R^N$ a random vector, which M dimensional approximation to x $(M < N)$ minimizes the mean square error of estimating x ? Again, pick a basis of eigenvectors of Q for R^N. Then, an M dimensional estimation of $\bar{x}$ is

$$\tilde{x} = \sum_{i=1}^{M} (\bar{x} \cdot \bar{e}_i) \bar{e}_i$$

while the original N dimensional vector is given by

$$\bar{x} = \sum_{i=1}^{N} (\bar{x} \cdot \bar{e}_i) \bar{e}_i \ .$$

The square error in the estimate $\tilde{x}$ of $\bar{x}$ is

$$\begin{aligned} \epsilon^2 &= (\bar{x} - \tilde{x}) \cdot (\bar{x} - \tilde{x}) \\ &= \left[\sum_{i=1}^{N} (\bar{x} \cdot \bar{e}_i) \bar{e}_i - \sum_{i=1}^{M} (\bar{x} \cdot \bar{e}_i) \bar{e}_i \right]^2 \\ &= \left(\sum_{i=M+1}^{N} (\bar{x} \cdot \bar{e}_i) \bar{e}_i \right) \cdot \left(\sum_{j=M+1}^{N} (\bar{x} \cdot \bar{e}_j) \bar{e}_j \right) \\ &= \sum_{i=M+1}^{N} (\bar{x} \cdot \bar{e}_i)^2 \end{aligned}$$

The expected value (ensemble average) of ϵ^2 is given by

$$\begin{aligned} E(\epsilon^2) &= E \left[\sum_{i=M+1}^{N} (\bar{x} \cdot \bar{e}_i)^2 \right] = \sum_{i=M+1}^{N} E \left((\bar{x} \cdot \bar{e}_i)^2 \right) \\ &= \sum_{i=M+1}^{N} \bar{e}_i \cdot E(x x^T) \bar{e}_i = \sum_{i=M+1}^{N} \bar{e}_i \cdot Q \bar{e}_i \ . \end{aligned}$$

Since $Q \bar{e}_i = \lambda_i \bar{e}_i$, and $\bar{e}_i \cdot \bar{e}_j = \delta_{ij}$,

$$E(\epsilon^2) = \sum_{i=M+1}^{N} \lambda_i \tag{13}$$

Clearly $E(\epsilon^2)$ will be minimized if the terms summed in 13 are the $N - M$ *smallest* eigenvalues of Σ.

Thus, we come to the following conclusion — we will minimize the mean square error in estimating $\bar{x}\epsilon R^M$ by $\tilde{x}\epsilon R^N$ if we form $\tilde{x}$ by projecting $\bar{x}$ onto the M eigenvectors of Q corresponding to the M largest eigenvalues λ. Equation 13 gives the MSE resulting from estimating $\bar{x}$ by $\tilde{x}$.

Consequently, if we know the maximum tolerable error in the estimate $\tilde{x}$, and the eigenvalue spectrum of Q, we can determine how large a representation (M) is needed (i.e. pick M such that $\sum_{i=M+1}^{N} \lambda_i <$ tolerable error where $\lambda_1 > \lambda_2 >, \ldots, \lambda_N$).

Claim: It is not necessary to choose the basis vectors for R^M *along* the M leading eigenvectors — any orthogonal set of basis vectors *spanned by* $(\bar{e_1}, \ldots, \bar{e_m})$ will do as well.

Proof 3.2 *Let $\bar{n}_i = \sum_{j=1}^{M} R_{ji}\bar{e}_j$ be a basis for R^M formed from the first M eigenvectors of Q (those with the largest eigenvalues). Then,*

$$\begin{aligned} \bar{n}_i \cdot \bar{n}_j &= \left(\sum_{k=1}^{M} R_{ki}\bar{e_k}\right) \cdot \left(\sum_{l=1}^{M} R_{lj}\bar{e}_l\right) \\ &= \sum_{k=1}^{M} R_{ki}R_{kj} = (R_T R)_{ij} \end{aligned}$$

so $\bar{n}_i \cdot \bar{n}_j = \delta_{ij} \Rightarrow R^T R = 1$ ($\bar{n_i}$ an orthonormal basis).

Let the M-dim estimation of $\bar{x}\epsilon R^N$ be

$$\tilde{x} \equiv \sum_{i=1}^{M} (\bar{x} \cdot \bar{n_i})\bar{n_i}$$

Then for $\bar{x}$ write

$$\bar{x} = \sum_{i=1}^{M} (\bar{x} \cdot \bar{n_i})\bar{n_i} + \sum_{i=M+1}^{N} (\bar{x} \cdot \bar{e_i})\bar{e_i}$$

[i.e. $\bar{n_i} \cdot \bar{e_k} = 0$]

Then, the square error is

$$\epsilon^2 = (\bar{x} - \tilde{x})^2$$

$$= \left[\sum_{i=1}^{M}(\bar{x}\cdot\bar{n}_i)\bar{n}_i + \sum_{i=M+1}^{N}(x\cdot\bar{e}_i)\bar{e}_i - \sum_{i=1}^{M}(\bar{x}\cdot\bar{n}_i)\bar{n}_i\right]^2$$

$$= \left[\sum_{i=M+1}^{N}(\bar{x}\cdot\bar{e}_i)\bar{e}_i\right]^2 = \sum_{i=M+1}^{N}(\bar{x}\cdot\bar{e}_i)^2$$

And the mean square error is

$$E(\epsilon^2) = E\left[\sum_{i=M+1}^{N}(\bar{x}\cdot\bar{e}_i)^2\right]$$

$$= \sum_{i=M+1}^{N}\bar{e}_i\cdot(Q\bar{e}_i) = \sum_{i=M+1}^{N}\lambda_i$$

in agreement with (13).

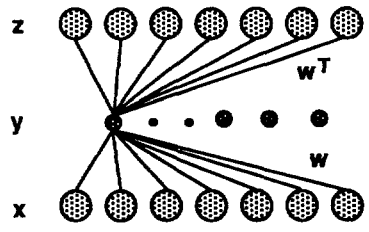


Figure 7: Two-layer, feed-forward network.

4 Neural Network Implementations

4.1 Self-supervised Back-propagation

We saw in the last section that to minimize the mean square error in an M-dimensional estimate of an N-dimensional vector, we should choose the basis on R^M to span the subspace spanned by the first M *principal eigenvectors* (those with the largest eigenvalues) of the data's correlation matrix.

Backpropagation is an algorithm designed to minimize mean square error in the input-output pairs x^I, z^I for a network. There is a variant of standard backpropagation which accomplishes exactly the principal subspace projection we have been discussing.

Suppose we configure a network as in Fig. 7 and train the network to perform the *IDENTITY* transformation. That is, train the network with input/output patterns

$$\{x^I, z^I = x^I\}$$

over the ensemble of data points $x^I \epsilon R^N$.

The network is trained to minimize the mean square error

$$E[(\bar{z} - \bar{x})^2]$$

and will produce an *encoding* of the data $\{x^I\}$ on its hidden layer. In general, with non-linear activation functions on the hidden and output layers, the encoding will not be a simple projection onto the M-dim principal subspace of Q. However, if we constrain the weights, and use *linear* activation functions then we *do* get a PCA out of the network.

In Fig. 7, we take $y = wx$ and $z = w^T y$. Suppose we train the net to perform the identity map on some ensemble of zero mean vectors $\{x^I\}$ using the LMS algorithm [back-prop].

The mean square error

$$E(\epsilon^2) = E\left[(z-x)^2\right]$$

will be minimized by the back-prop (there are no local minima for this problem) [4]. This mean square error can be written as

$$E(\epsilon^2) = \text{Trace}\left[w^T w Q w^T w + Q - 2Q w^T w\right]$$

where Q is the input ensemble's correlation matrix. To see this, first notice that the inner product can be represented as

$$\bar{x} \cdot \bar{z} = x^T z = \text{Trace}(x z^T) .$$

Expand the outer product and compute the trace,

$$\begin{aligned} \text{Trace}[x z^T] &= \text{Trace}\left(\begin{pmatrix} x_1 \\ x_2 \\ \vdots \end{pmatrix} (z_1 z_2 \ldots)\right) \\ &= \text{Trace}\begin{bmatrix} x_1 z_1 & x_1 z_2 \ldots \\ x_2 z_1 & x_2 z_2 \ldots \\ \vdots & \end{bmatrix} = x_1 z_1 + x_2 z_2 + \ldots \equiv x \cdot z . \end{aligned}$$

Then the square error for one pattern pair is given by

$$\begin{aligned} \epsilon^2 = (z-x)\cdot(z-x) &= \text{Trace}[(z-x)(z-x)^T] \\ &= \text{Trace}\left[\{(w^T w - 1)x\}\{(w^T w - 1)x\}^T\right] \\ &= \text{Trace}[\{(w^T w - 1)x\}\{x^T(w^T w - 1)\} . \end{aligned}$$

The expectation of value ϵ^2 is

$$E(\epsilon^2) = \text{Trace}\left[(w^T w - 1)Q(w^T w - 1)\right]$$

Thus

$$E(\epsilon^2) = \text{Trace}\left[w^T w Q w^T w - 2Q w^T w + Q\right] . \qquad (14)$$

First, we will show that $E(\epsilon^2)$ is minimized when the rows of w are the principal M eigenvectors of Σ, and then show that orthogonal combinations also minimize $E(\epsilon^2)$.

Let

$$w = \begin{bmatrix} (e_1) \\ (e_2) \\ \vdots \\ (e_m) \end{bmatrix}$$

where $Q\bar{e}_i = \lambda_i \bar{e}_i$ and

$$\lambda_1 > \lambda_2 >, \ldots, \lambda_N \, .$$

Then, since the $\bar{e}_i$ are eigenvectors of Q

$$Qw^T = Q\left[(e_1)(e_2)\ldots(e_m)\right] = \left[(\lambda_1 e_1)(\lambda_2 e_2)\ldots\right] ,$$

and

$$wQw^T = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \ddots \lambda_m \end{pmatrix} \equiv \Lambda \ .$$

Finally, completing the first term in 14,

$$\text{Trace}[w^T w Q w^T w] = \text{Trace}[w^T \Lambda w] = \text{Trace}[\Lambda w w^T]$$

$$= \text{Trace}\left[\begin{pmatrix} \lambda_1 & 0 \\ 0 & \ddots \lambda_m \end{pmatrix} \begin{pmatrix} (\bar{e_1}) \\ (\bar{e_2}) \\ \vdots \end{pmatrix} ((\bar{e_1})(\bar{e_2})\cdots) \right]$$

$$= \text{Trace}(\Lambda) = \sum_{i=1}^{M} \lambda_i \, .$$

The second term in 14 is

$$\begin{aligned} -2 \text{ Trace}[Qw^T w] &= \text{Trace}[wQw^T] = \text{Trace}(\Lambda) \\ &= \sum_{i=1}^{M} \lambda_i \, . \end{aligned}$$

And the last term is

$$\text{Trace}(Q) = \sum_{i=1}^{N} \lambda_i \, .$$

Putting them together leaves in place of 14 ,

$$E(\epsilon^2) = \sum_{j=1}^{N} \lambda_i - \sum_{i=1}^{M} \lambda_i = \sum_{i=M+1}^{N} \lambda_i \, . \tag{15}$$

Thus $E(\epsilon^2)$ will be minimized if the sum in 15 is over the smallest eigenvalues. Thus, $E(\epsilon^2)$ is minimized when the M rows of w are the principal M eigenvectors of Q .

Next, we show that $E(\epsilon^2)$ is unchanged by rotating the rows of w within the M-dimensional PCA subspace. To see this, make an orthogonal transformation of the columns of w^T

$$w'^T = Rw^T = ((R\bar{e_1})(R\bar{e_2})\ldots)$$

where $R : R^M \rightarrow R^M$, $RR^T = 1$, then

$$\begin{aligned} \text{Trace}\,[w'^T w' Q w'^T w'] &= \text{Trace}\,[Rw^T w R^T Q R w^T w R^T] \\ &= \text{Trace}\,[(RR^T) w^T w Q w^T w (RR^T)] \\ &= \text{Trace}\,[w^T w Q w^T w] \end{aligned}$$

and

$$\begin{aligned} \text{Trace}\,[w' Q w'^T] &= \text{Trace}\,[w R^T Q R w^T] \\ &= \text{Trace}\,[RR^T w Q w^T] \\ &= \text{Trace}\,[w Q w^T] \end{aligned}$$

Thus, $E(\epsilon^2)$ is invariant under orthogonal transformations of the rows of w that leave the M-dim principal subspace unchanged.

> $\Rightarrow$ $E(\epsilon^2)$ is minimized when the rows of w form an orthonormal basis for the M-dim principal subspace.

4.2 Feature Discovery by Hebbian Learning

In a classic paper, Oja [5] draws the connection between Hebbian learning and principal component analysis. We will review both Oja's original formulation and extensions that perform complete principal component analysis.

Consider a single linear neuron with output activity $y = \bar{w} \cdot \bar{x}$. The input patterns $\bar{x}$ are assumed to be random vectors drawn from an unknown, but stationary, probability distribution. As each new pattern is presented, the vector of synaptic weights $\bar{w}$ is updated according to the Hebbian learning rule

$$\delta \bar{w} = \alpha y \bar{x} \tag{16}$$

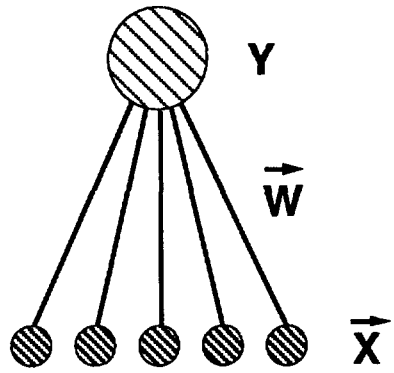


Figure 8: Oja's single, linear neuron.

where α is the learning rate.

We will assume that α is small enough so that the change in $\bar{w}$ is adiabatic (i.e., slow) with respect to the changes in $\bar{x}$. Given this adiabatic assumption, we can average $\delta\bar{w}$ over the entire ensemble of input patterns

$$\begin{aligned} \delta\bar{w} \simeq \alpha E(y\bar{x}) &= \alpha E[(\bar{w}\cdot\bar{x})\bar{x}] \\ &= \alpha E(\bar{x}(\bar{x}^T w)) = \alpha Q\bar{w} \end{aligned}$$

Thus, on the average, the weight changes are driven by

$$\delta\bar{w} \simeq \alpha Q\bar{w}. \tag{17}$$

What does (17) produce? Expand $\bar{w}$ in eigenvectors of Q

$$\bar{w} = \sum_{i=1}^{N} w_i\bar{e_i} \ , \ \ Qe_i = \lambda_i e_i \ , \ \ \lambda_1 > \lambda_2 > \ldots > \lambda_N.$$

Then the weight change is given by

$$\begin{aligned} \delta\sum_i w_i\bar{e_i} &= \alpha Q\sum_i (w_i\bar{e_i}) \\ &= \alpha\sum_i w_i\lambda_i\bar{e_i} \ . \end{aligned} \tag{18}$$

Or, for the change in the i^{th} component of $\bar{w}$,

$$\delta w_i = \alpha\lambda_i w_i \ . \tag{19}$$

If we apply 19 iteratively, then after the n^{th} iteration we find

$$w_i(n) = w_i(0)[1 + \alpha\lambda i]^n$$

which diverges as $n \longrightarrow \infty$. However, examine the ratio

$$\frac{w_i(n)}{w_j(n)} = \left(\frac{1 + \alpha\lambda_i}{1 + \alpha\lambda_j}\right)^n .$$

Clearly if $\lambda_i > \lambda_j$ then $w_i(n)$ / $w_j(n) \rightarrow \infty$ as $n \rightarrow \infty$. But if $\lambda_j > \lambda_i$,then $w_i(n)$ / $w_j(n) \rightarrow 0$ as $n \rightarrow \infty$. Thus, after many applications of (19), the projection of $\bar{w}$ onto the eigenvector of Q with largest eigenvalue dominates, and $\bar{w}$ diverges in the direction of $\bar{e}_i$. That is

$$\bar{w} \rightarrow C^n \bar{e}_1$$

where $C^N \rightarrow \infty$ *as* $n \rightarrow \infty$.

This divergent behavior for Hebbian learning laws has been known for a long time, and the learning rule in (16) needs to be augmented with a "forgetting" term to bound w.

If suitable forgetting terms can be devised, then the neuron is useful as a *principal component* analyzer. Various bounding terms are appropriate. We will consider two.

To summarize, if we average the Hebbian learning rule

$$\delta \bar{w} = [y\bar{x}] \tag{20}$$

over the ensemble of input patterns, the weight changes are driven by the correlation matrix,

$$\delta \bar{w} = \alpha Q \bar{w} \,, \tag{21}$$

where $Q = E(xx^T)$. Repeated application of (21) gives

$$\bar{w}(n) = (1 + \alpha Q)^n \bar{w}(0) \tag{22}$$

and leads to a weight vector which diverges in magnitude, but points in the direction of the *principal eigenvector* of Q.

Exercise 4.1 *Let the input ensemble be formed from a fixed N-Dim vector* $\bar{x}^0$ *by adding independently distributed zero mean noise to each component. Thus*

$$x_i = (x_1^0 + \xi_1, \;\; x_2^0 + \xi_2, \; \ldots, x_N^0 + \xi_N)$$

where

$$E(\xi_i) = 0 \quad \textit{(zero mean)}$$

and $E\,(\xi_i \xi_j) = \sigma^2 \delta_{ij}$ *(independently, identically distributed noise with variance* σ^2 *):*

(a) Show that the correlation matrix $Q_{ij} = (x_i x_j) = (xx^T)_{ij}$ *is given by*

$$Q = x^0 x^{0T} + 1\sigma^2$$

(b) Find the eigenvectors of Q *and their associated eigenvalues.*

In order to bound the weight vectors, a suitable term must be added to (20). The form suggested by Oja reads

$$\delta \bar{w} = \alpha[\bar{x}y - y^2 \bar{w}]. \tag{23}$$

When averaged over an ensemble of input vectors, (23) becomes

$$\begin{aligned} \delta \bar{w} &= \alpha E(\bar{x}\bar{y} - y^2 \bar{w}) \\ &= \alpha E[\bar{x}(\bar{x} \cdot \bar{w}) - (\bar{x} \cdot \bar{w})(\bar{x} \cdot \bar{w})\bar{w}] \end{aligned}$$

or, in matrix notation

$$\begin{aligned} \delta w &= \alpha E[(xx^T)w - (w^T x x^T w)w] \\ &= \alpha[Qw - (w^T Q w)w] \,. \end{aligned} \tag{24}$$

Under suitable conditions (the subject of stochastic approximation theory) the discrete update rule of (24) is equivalent to the differential equation

$$\frac{d}{dt}w = Qw - (w^T Q w)w. \tag{25}$$

The behavior of the solutions to (25) can be examined by expanding w in the basis of eigenvectors of Q,

$$w \equiv \sum_{i=1}^{N} a_i \bar{e}_i$$

and substituting in (25) to obtain

$$\sum_i \left(\frac{da_i}{dt}\right) \bar{e}_i = \sum_i \lambda_i a_i \bar{e}_i - \left(\sum_j \lambda_j a_j^2\right) \sum_i a_i \bar{e}_i \,.$$

Take the inner product of this last expression with $\bar{e}_k$ to recover

$$\frac{d}{dt}a_k = \lambda_k a_k - \left(\sum_j \lambda_j a_j^2\right) a_k = \left[\lambda_k - \sum_j \lambda_j a_j^2\right] a_k \,. \tag{26}$$

Now, if (26) is to converge to a static equilibrium, we must have $\frac{da_k}{dt} = 0$. So as $t \to \infty$, we must have

$$\begin{aligned} [\lambda_k - \sum_j \lambda_j a_j^2] &= 0 \\ \text{or } \; a_k &= 0. \end{aligned} \tag{27}$$

One *possible* solution to (27) is

$$\begin{aligned} a_1 &= \pm 1 \\ a_k &= 0 \quad k \neq 1 \end{aligned}$$

in which case $\bar{w} = \pm \bar{e}_1$. We need to show that this is, in fact, the solution that evolves from (26).

The trick to solving (26) is to examine the time course of the ratio of coefficients $\frac{a_k}{a_l}$. Examine

$$\begin{aligned}\left(\frac{a_l}{a_k}\right)\frac{d}{dt}\left(\frac{a_k}{a_l}\right) &= \left(\frac{a_l}{a_k}\right)\left[\frac{\dot{a}_k}{a_l} - \frac{a_k\dot{a}_e}{a_l^2}\right] = \frac{\dot{a}_k}{a_k} - \frac{\dot{a}_e}{a_l} \\ &= \frac{(\lambda_k - \sum_j \lambda_j a_j^2)a_k}{a_k} - \frac{(\lambda_l - \sum_j \lambda_j a_j^2)a_l}{a_l} \\ &= (\lambda_k - \lambda_l)\,.\end{aligned}$$

Thus the ratio evolves according to —

$$\frac{d}{dt}\left(\frac{a_k}{a_l}\right) = (\lambda_k - \lambda_l)\left(\frac{a_k}{a_l}\right) \tag{28}$$

which is a simple *linear differential equation* with constant coefficients. The solution of (28) is

$$\left(\frac{a_k}{a_l}\right) = \left(\frac{a_k}{a_l}\right)_{t=0} \exp\,(\lambda_k - \lambda_l)t. \tag{29}$$

Clearly, if $k > l$ then a_k / a_l diverges at $t \to \infty$ (recall $\lambda_1 > \lambda_2 > \ldots > \lambda_N$). Thus a_1 dominates as $t \to \infty$. We need to show that $|a_1|$ is bounded for $t \to \infty$. Returning to (26) with $k = 1$,

$$\frac{d}{dt}a_1 = (\lambda_1 - \sum_j \lambda_j a_j^2)a_1 \tag{30}$$

Since for large t, a_1 dominates, we can approximate (30) by

$$\frac{d}{dt}a_1 \simeq \lambda_1(1 - a_1^2)a_1 \quad \text{for } t \gg 0. \tag{31}$$

Now we're *really* interested in $\frac{d}{dt}(a_1)^2 = 2\,a_1\,\frac{da_1}{dt}$, so using (31)

$$\frac{d}{dt}(a_1)^2 = 2a_1\frac{da_1}{dt} = 2\lambda_1(1 - a_1^2)a_1^2\,. \tag{32}$$

Equation (32) is seperable so we can integrate it in closed form

$$\int \frac{d(a_1^2)}{a_1^2(1 - a_1^2)} = \int 2\lambda_1 dt\,. \tag{33}$$

Performing the integration leaves

$$\ln(\frac{a_1^2}{a_1^2-1}) = 2\lambda_1 t + \text{const.}$$

or

$$\left(\frac{a_1^2}{a_1^2-1}\right) = C\ \exp 2\lambda_1 t\ .$$

The last form can be rewritten as

$$a_1^2(t) = \left[1 - C^{-1}\ \exp(-2\lambda_1 t)\right]^{-1}\ . \tag{34}$$

Thus as $t \to \infty$

$$\begin{aligned} a_1 &\to \pm 1 \\ a_i &\to 0, \quad i \neq 1 \end{aligned}$$

and so $\bar{w} \to \pm\bar{e}_1$ the *principal* eigenvector of Q.

Exercise 4.2 *The form of the update rule in (24) is not unique for our purpose (bounding $|w|^2$). A similar form*

$$\begin{aligned} \delta\bar{w} &= \alpha E[\bar{x}y - (\bar{w}\cdot\bar{w})\bar{w}] \\ &= \alpha E[xx^T w - (w^T w)w] \\ &= \alpha(Qw - (w^T w)w) \end{aligned} \tag{35}$$

also bounds $|w|^2$. Following the derivation from (25) - (34), show that the continuous limit of (35), namely $\frac{d}{dt}\bar{w} = Q\bar{w} - (\bar{w}\cdot\bar{w})\bar{w}$, converges to $\bar{w} = \pm\sqrt{\lambda_1}\bar{e}_1$.

The form of (35) is also interesting since it can be derived from a *potential.* Let

$$\begin{aligned} V &\equiv \frac{1}{2}\{-\bar{w}\cdot Q\bar{w} + \frac{1}{2}(\bar{w}\cdot\bar{w})^2\} \\ &= \frac{1}{2}\{-\sum_{i,j} w_i Q_{ij} w_j + \frac{1}{2}(\sum_j w_j w_j)^2\} \end{aligned} \tag{36}$$

Then the gradient of V with respect to w is given by

$$\frac{\partial V}{\partial w_k} = -\sum_j Q_{kj} w_j + (\sum_j w_j w_j) w_k$$

or in matrix form

$$\frac{\partial V}{\partial w} = -Qw + (w^T w)w.$$

Thus (35) can be cast in the form

$$\delta \bar{w} = -\alpha \frac{\partial V}{\partial w} \tag{37}$$

the continuous form of which is

$$\frac{d}{dt}w = -\frac{\partial V}{\partial w}\,. \tag{38}$$

Equations (37) or (38 tell us that w evolves by *gradient descent* on the potential surface of (36).

Exercise 4.3 *Show that the potential function*

$$V = \frac{1}{2}\{-\bar{w}\cdot Q\bar{w} + \frac{1}{2}(\bar{w}\cdot\bar{w})^2\}$$

(a) *has critical points at* $\bar{w} = \pm\sqrt{\lambda_i}\bar{e}_i$, $\bar{e}_i$ *an eigenvector of* Q,

(b) *has a minimum at* $\bar{w} = \pm\sqrt{\lambda_1}\bar{e}_1 \quad \lambda_1 > \lambda_2 > \ldots > \lambda_N$ *and* saddle points *at all other critical points in (a),*

(c) *has a local maximum at* $w = 0$.

(d) *Sketch* $V(w)$ *for* $w \epsilon R^2$.

4.3 Capturing Higher-Order Eigenvectors

An array of cells developing according to (23) or (35) would all converge to the principal eigenvector of $Q = E(xx^T)$. In order to retreive more of the eigenvectors, some additional interactions need to be added. There are at least two ways of accomplishing this:

1. Feedback from the array to the input;

2. Lateral connections between the output cells of the array.

Lateral connections are quite involved, so we will look at a feedback-like scheme. The scheme was developed by Sanger [6]. The idea is to use the basic scheme of (23) to train the weight vector to the first node

$$\delta \bar{w}_1 = \alpha[y_1\bar{x} - y_1^2\bar{w}_1] \tag{39}$$

where $y_1 = \bar{w}_1 \cdot \bar{x}$. The first output node is trained according to (39) and converges to the leading eigenvector, $\bar{e}_1$. The weight vector to the second node is trained according to

$$\delta \bar{w}_2 = \alpha[y_2(\bar{x} - y_1 \bar{w}_1) - y_2^2 \bar{w}_2] \tag{40}$$

where $y_2 = \bar{w}_2 \cdot \bar{x}$. Notice that in the first term on the right-hand side of (40), the usual Hebbian rule has been modified. Rather than updating the weight vector in proportion to $y_2 \bar{x}$, the update is proportional to y_2 times

$$\bar{x}_2 \equiv \bar{x} - y_1 \bar{w}_1 = \bar{x} - (\bar{w}_1 \cdot \bar{x}) \bar{w}_1 \tag{41}$$

We can interprete (41) in several ways. The first equality tells us to calculate $\bar{x}_2$ by subtracting from the input $\bar{x}$, the output of the first node *fed back* to the input through the weights $\bar{w}_1$, i.e. $y_1 \bar{w}_1$. This may be interpreted as an inhibitory projection back to the input plane.

Suppose that $\bar{w}_1$ has converged to $\bar{e}_1$. Then the output of the first node is $y_1 = \bar{e}_1 \cdot \bar{x}$ and (40) becomes

$$\delta \bar{w}_2 = \alpha[y_2(\bar{x} - \bar{e}_1(\bar{e}_1 \cdot \bar{x})) - y_2^2 \bar{w}_2] \tag{42}$$

Averaging (42) and passing to the continuous time limit leaves the differential equation

$$\frac{d}{dt} \bar{w}_2 = Q \bar{w}_2 - (\bar{e}_1 \cdot Q \bar{w}_2) \bar{e}_1 - (\bar{w}_2 \cdot Q \bar{w}_2) \bar{w}_2 \tag{43}$$

Now the evolution of the component of $\bar{w}_2$ along $\bar{e}_1$ is given by taking the inner product of (43) with $\bar{e}_1$,

$$\frac{d}{dt}(\bar{w}_2 \cdot \bar{e}_1) = -(\bar{w}_2 \cdot Q \bar{w}_2)(\bar{w}_2 \cdot \bar{e}_1). \tag{44}$$

Recalling that Q is positive definite, (44) says that $\bar{w}_2 \cdot \bar{e}_1$ decays exponentially. We can determine the behavior of the solution of (43) by taking the inner product with $\bar{e}_k$ and using the same argument used to discuss Oja's formulation in 4.2. Using a more direct approach, we define the *projection operator*

$$\Pi_1 \equiv 1 - e_1 e_1^T \ . \tag{45}$$

This operator satisfies

$$\begin{aligned} &\text{a.} \quad \Pi_1^T = \Pi_1 \\ &\text{b.} \quad \Pi_1 e_1 = 0 \\ &\text{c.} \quad \Pi_1^2 \bar{v} = \Pi_1 \bar{v} \text{ for any } \bar{v} \in R^N \\ &\text{d.} \quad \bar{e}_1 \cdot (\Pi_1 \bar{v}) = 0 \end{aligned} \tag{46}$$

and acts to project vectors *orthogonal* to $\bar{e}_1$.

Apply Π_1 to both sides of (43) and use (46) to obtain

$$\frac{d}{dt}(\Pi_1 \bar{w}_2) = \Pi_1 Q \bar{w}_2 - (\bar{w}_2 \cdot Q \bar{w}_2)(\Pi_1 w_2) \,. \tag{47}$$

We know from (44) that at *late times* $\bar{w}_2 \cdot \bar{e}_1 \rightarrow 0$, and so by the definition in (45), $\Pi_1 \bar{w}_2 = \bar{w}_2$. Furthermore, by (46a) and (46c), $\Pi_1^T \Pi_1 \bar{w}_2 = \bar{w}_2$. Using these last results, we can rewrite (47) as

$$\begin{aligned} \frac{d}{dt}\bar{w}_2 &= \Pi_1 Q \bar{w}_2 - (\bar{w}_2 \cdot Q \bar{w}_2)(\Pi_1 \bar{w}_2) \\ &= (\Pi_1 Q \Pi_1^T)\bar{w}_2 - [\bar{w}_2 \cdot (\Pi_1 Q \Pi_1^T)\bar{w}_2]\bar{w}_2 \\ &\quad \text{for } t \gg 0. \end{aligned} \tag{48}$$

Equation (48) says that at late times, changes in $\bar{w}_2$ are driven by

$$\tilde{Q} \equiv \Pi_1 Q \Pi_1^T \,. \tag{49}$$

In analogy with the development in Section 4.2, $\bar{w}_2$ will converge to the leading eigenvector of $\tilde{Q}$. It is straightforward to show that

$$\tilde{Q}\bar{e}_1 = 0 \tag{50}$$

and

$$\tilde{Q}\bar{e}_i = \lambda_i \bar{e}_i \,, \quad i \neq 1 \,.$$

Thus, $\bar{e}_2$ is the eigenvector of $\tilde{Q}$ with the largest eigenvalue. Hence, $\bar{w}_2$ converges to $\pm\bar{e}_2$.

The scheme extends to arbitrarily many output nodes (with the restriction that the number of nodes M is less than or equal to the dimension of the input space). The weight vector to the i^{th} output node is adapted according to

$$\begin{aligned} \delta \bar{w}_i &= \alpha \left[y_i (\bar{x} - \sum_{j=1}^{i-1} y_j \bar{w}_j) - y_i^2 \bar{w}_i \right] \\ &= \alpha \left[y_i \bar{x} - y_i \sum_{j=1}^{i} y_j \bar{w}_j \right] \end{aligned} \tag{51}$$

This scheme is very robust with respect to convergence, however the complexity involved in the feedback projections is high. Properly designed lateral connections between the output cells can achieve the same results, although such schemes are inherently less stable.

References

[1] D. Hammerstrom. A connectivity analysis of recursive, auto-associative connection networks. Technical Report CSE-86-009, Oregon Graduate Institute, August 1986.

[2] Eric Baum and David Haussler. What size net gives valid generalization? *Neural Computation*, 1:151–160, 1989.

[3] A. Papoulis:. *Probability, Random Variables, and Stochastic Processes.* McGraw-Hill Book Company, 1984.

[4] P. Baldi and K. Hornick. Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, 2:53–58, 1989.

[5] E. Oja. A simplified neuron model as a principal component analyzer. *Jour. Math. Bio.*, 15:267–273, 1982.

[6] T. Sanger. An optimality principle for unsupervised learning. In D.S. Touretzky, editor, *Advances in Neural Information Processing Systems 1.* Morgan Kauffmann, 1989.