Density Shaping by Neural Networks with Application to Classification, Estimation and Forecasting

by

Y. Baram and Z. Roth

CIS Report #9420
December 1994
Density Shaping by Neural Networks with Application to Classification, Estimation and Forecasting

Yoram Baram* and Ze'ev Roth†

December 6, 1994

Abstract

An estimate of the probability density function of a random vector is obtained by maximizing the output entropy of a feed-forward network of sigmoidal units with respect to the input weights. Classification problems can be solved by selecting the class associated with the maximal estimated density. Newton's method, applied to the estimated density, yields a recursive estimator for a random variable or a random sequence. A constrained connectivity structure yields a linear estimator, which is particularly suitable for "real time" prediction. Applications to real classification and forecasting problems are demonstrated. The convergence of the training algorithm is analyzed in the appendix.

*Y. Baram is with the Department of Computer Science, Technion, Israel Institute of Technology, Haifa 32000, Israel. He is also associated with the NASA Ames Research Center, Moffett Field, CA 94035.
†Z. Roth is with Qualcomm, P.O.B. 719, Advanced Technology Center, Haifa 31000, Israel.
This work was supported in part by the Director's Discretionary Fund, NASA Ames Research Center and in part by the Fund for the Promotion of Research at the Technion.
1 Introduction

Neural networks are being applied to a wide variety of pattern recognition and signal processing problems. Statistical and information theoretic methods are playing an increasing role in the design and analysis of such networks ([1] - [5]). The representation of probability density functions by neural networks has been of particular interest. The proposed techniques were largely based on parametric function approximation by an arbitrary number of neurons (e.g., [6],[7]) or on non-parametric representation, in the spirit of the Parzen estimator, by a network whose size equals that of the sample [8]. Employing well established statistical performance criteria in neural network design leads not only to the development of useful new tools, but to a more profound understanding of the role of neural networks in solving statistical inference problems.

In this work we suggest estimating the probability density function (pdf) of a random vector by passing the latter through a feed-forward network consisting of a layer of sigmoidal units. The number of units is equal to the input dimension. Values for the network’s parameters are derived by maximizing the output entropy. This approach is reminiscent of the "shaping filter" or the "inverse filter" approach used in digital signal processing [9] to derive the spectral density of a given signal. Approximating the output spectral density of the shaping filter by that of white noise, the filter’s inverse transfer function approximates the input’s spectral density. In our approach, maximizing the output entropy implies approximating the output pdf by a uniform density. This analogy suggest the term "shaping network" for the sigmoidal network involved. We derive the parameter optimization algorithm for the general case, then specialize to specific neuron functions. In particular, the hyperbolic tangent and the gaussian functions, which are suitable for approximating unimodal distributions, are considered. The gaussian function, which produces a closed form solution for the weights matrix, is useful in initializing the algorithm. The convergence of the training algorithm is analyzed in the appendix.

Classification problems can be solved by first estimating the pdf corresponding to each of the classes, then selecting the class corresponding to the largest estimated pdf for a given input. We also derive a recursive estimator for a random variable given a random vector. This estimator maximizes the conditional density calculated by the pdf estimator. Employing a particular sigmoidal nonlinearity produces explicit expressions for the gradient and the Hessian in the optimization algorithms. A certain restriction on the connectivity structure of the density shaping network results in a linear estimator, based on the conditional expectation under the estimated density. The linear structure is particularly useful for predicting the future values of a random sequence, given past and present values, in "real time" (which stands in contrast to the recursive estimator). Applications of the proposed techniques to diamond classification and to the prediction of the sun-spot process are described.
2 Density Shaping

Let \( x \in \mathbb{R}^n \) be a random vector, having a probability density function \( p_X(x) \). Let \( y \in \mathbb{R}^n \) be defined by its \( i \)'th component

\[
y_i = G(u_i)
\]

where \( G(\cdot) \) is a monotone increasing, continuous function, satisfying \( \lim_{v \to \infty} G(v) = 1 \) and \( \lim_{v \to -\infty} G(v) = 0 \), a “sigmoid”, and

\[
u_i = w_i^T x + t_i
\]

where \( w_i \) is the \( i \)th row of \( W \in \mathbb{R}^2 \), a real nonsingular matrix, \( w^T \) denotes the transpose of \( w \) and \( t_i \) is a scalar “threshold”. In vector form

\[
y = G(u)
\]

where \( y = [y_1, \ldots, y_n]^T \), \( u = [u_1, \ldots, u_n]^T \), and \( G(u) = [G(u_1), \ldots, G(u_n)]^T \). It follows that

\[
x = F(y) \equiv W^{-1} \left[ G^{-1}(y) - t \right]
\]

where \( W^{-1} \) is the inverse of the matrix \( W \), \( G^{-1}(y) \) is a vector whose \( i \)'th component is the inverse function \( u_i = G^{-1}(y_i) \), and \( t = [t_1, \ldots, t_n]^T \).

The probability density function of \( y \) satisfies [10]

\[
p_Y(y) = \frac{p_X(x)}{|\text{det } J(x)|} \bigg|_{x=F(y)}
\]

where \( \text{det } J(x) \) is the determinant of the Jacobian of \( y \) with respect to \( x \), whose \( i, j \)'th component is

\[
J_{i,j}(x) = \frac{\partial y_i}{\partial x_j}
\]

Clearly, \( J(x) \) is a square matrix. The entropy of \( x \) is defined by

\[
h(x) = -E\{\log p_X(x)\}
\]

hence,

\[
h(y) = h(x) + E\{\log |\text{det } J(x)|\}
\]

Supposed that \( |\text{det } J(x)| \) is a density function for \( x \) (that is, a nonnegative scalar function, integrating to unity). Denoting by

\[
D[p_X(x), q(x)] = E\left\{ \log \frac{p_X(x)}{|\text{det } J(x)|} \right\}
\]

the divergence between the true density of \( x, p_X(x) \), and an arbitrary density \( q(x) \), we have

\[
-h(y) = E\{\log p_X(x)\} - E\{\log |\text{det } J(x)|\} = D[p_X(x), |\text{det } J(x)|]
\]
Figure 1: Density shaping: (a) The optimal weights are attained by maximizing the joint entropy of the output vector. (b) Density calculation.

It is not difficult to show (see, e.g., [11], p. 535) that $D[p_X(x), |\det J(x)|] \geq 0$. It follows that the divergence between $p_X(x)$ and $|\det J(x)|$ is minimized when $h(y)$ is maximized. The input pdf will be approximated then by $|\det J(x)|$. Note that when $G(\cdot)$ is bounded within the interval $[0, 1]$, the maximum possible entropy of $y$ is 0, the entropy of a uniform (unity) density over the cube $[0, 1]^n$. Maximizing the output entropy then implies approximating the output density, in the sense of minimum divergence, by a uniform density. This may be viewed as an analogy to producing white noise at the output of the “shaping filter” in spectral estimation. Since $h(x)$ does not depend on $W$ or $t$, $h(y)$ is maximized by finding

$$\max_{W,t} E\{\log |\det J(x)|\}$$

(5)

this implies that the input entropy according to the estimated density $|\det J(x)|$ is maximized. Our approach may be viewed, then, as maximum entropy estimation of the input density under the parameterization of $|\det J(x)|$.

Noting that

$$\frac{\partial y_i(x)}{\partial x_j} = \sum_{k=1}^{n} \frac{\partial y_i(x)}{\partial u_k} \frac{\partial u_k(x)}{\partial x_j}$$

it can be seen that

$$|\det J(x)| = |\det W| \prod_{i=1}^{n} g(u_i)$$

(6)

where

$$g(u_i) = \frac{\partial G(u_i)}{\partial u_i}$$

(7)

is non-negative. The proposed pdf estimate is then

$$\hat{p}_X(x) = |\det W| \prod_{i=1}^{n} g(u_i)$$

(8)

The “density shaping” stages are illustrated by Fig. 1.
We have assumed that $|\det J(x)|$ is a density for $x$. Let us show that this is true under the assumptions made above.

**Lemma** Suppose that $G(\cdot)$ is a monotone increasing and differentiable function satisfying $\lim_{v \to -\infty} G(v) = 1$ and $\lim_{v \to -\infty} G(v) = 0$. Let $w_i$, $i = 1, \ldots, n$ be parameter vectors and let $u_i = r(w_i, x)$ be a differentiable function of $x$. Further let $y_i = G(u_i), u = [u_1, \ldots, u_n]^T$ and $y = [y_1, \ldots, n]^T$ and let $J(x) \equiv J_{xy}$ denote the Jacobian of $y$ with respect to $x$. Then $|\det J(x)|$ is a density of $x$.

**Proof** Clearly, $|\det J(x)| > 0$. It remains to show that

$$\int |\det J(x)| \, dx = 1$$

It follows from the chain rule of differentiation that

$$J_{xy} = J_{uy}J_{xu}$$

hence

$$\det J_{xy} = \det J_{uy} \det J_{xu}$$

Noting that $du = |\det J_{xu}| \, dx$, we have

$$\int |\det J(x)| \, dx = \int \cdots \int |\det J_{uy}| \, |\det J_{xu}| \, dx = \int \cdots \int |\det J_{uy}| \, du$$

Substituting

$$\det J_{uy} = \prod_{i=1}^{n} g(u_i)$$

where $g(u_i) = \partial G(u_i)/\partial u_i \geq 0$, we obtain

$$\int |\det J(x)| \, dx = \int \cdots \int \prod_{i=1}^{n} g(u_i) \, du = \prod_{i=1}^{n} \int g(u_i) = 1^n = 1$$

which completes the proof. In the case $r(w_i, x) = w_i^T x$, we have, of course, $J_{xu} = W$.

**Training Algorithm**

To find the optimal parameters, we need to maximize

$$Q = E\{\log |\det J(x)|\} = \log |\det W| + \sum_{i=1}^{n} E\{\log g(w_i^T x + t_i)\}$$

Suppose that $\det W > 0$. Let us denote

$$s(u_i) = \frac{\partial}{\partial u_i} \log g(u_i) = \frac{\partial g(u_i)/\partial u_i}{g(u_i)}$$

and

$$S(u) = [s(u_1), \ldots, s(u_n)]^T$$

5
Let the \(i,j\)'th minor of \(W\) (that is, the matrix obtained from \(W\) by deleting the \(i\)'th row and the \(j\)'th column) be denoted \(A_{i,j}\). The Laplace decomposition of \(\det W\) is given by [13]

\[
\det W = \sum_{j=1}^{n} (-1)^{i+j} w_{i,j} \det A_{i,j}
\]

Hence,

\[
\frac{\partial}{\partial w_{i,j}} \det W = (-1)^{i+j} \det A_{i,j}
\]

and since

\[
[W^{-1}]_{i,j} = \frac{(-1)^{i+j} \det A_{i,j}}{\det W}
\]

we have (assuming that the order of differentiation and integration can be properly interchanged)

\[
\frac{\partial Q}{\partial w_{i,j}} = [W^{-1}]_{i,j} + E\{s(u_i)x_j\}
\]

It follows that the gradients of \(Q\) with respect to \(W\) and \(t\) are

\[
\nabla_W Q = W^{-T} + E\{S(u)x^T\} \quad (9)
\]

where \(W^{-T} = W^{-1T}\) and

\[
\nabla_t Q = E\{S(u)\} \quad (10)
\]

The optimal solutions for \(W\) and \(t\) satisfy the equations

\[
W^{-T} + E\{S(u)x^T\} = 0 \quad \text{and} \quad E\{S(u)\} = 0
\]

It should be noted that, when \(p_X(x)\) is symmetric with respect to \(E\{x\}\), and \(g(Wx + t)\) is symmetric with respect to \(t\), replacing \(x\) by its centered version (that is, subtracting \(E\{x\}\) from \(x\)) results in an optimal threshold \(t = 0\).

In the case of a gaussian nonlinearity

\[
G(u_i) = \text{erf}(u_i) = \frac{1}{2\pi} \int_{-\infty}^{u_i} \exp \left( -\frac{\eta^2}{2} \right) d\eta
\]

we obtain

\[
\nabla_W Q = W^{-T} + E\{xx^T\}
\]

and

\[
\nabla_t Q = -E\{Wx + t\}
\]

The optimal solutions for \(W\) and \(t\) are in this case

\[
W = \left[ E\{xx^T\} - E\{x\}E\{x^T\} \right]^{-1/2} = [\text{cov}(x)]^{-1/2} \quad (11)
\]
which is in agreement with our previous observation that the optimal \( t \) for centered \( x \) in the case of symmetric density is 0. The problem has in this case a closed solution. In practice, the expectations in (11) and (12) will be replaced by the sample averages. We note that if \( W \) is a solution satisfying

\[
W^T W = [\text{cov}(x)]^{-1}
\]

then, for any orthogonal matrix \( V \), the matrix \( W = V W \) is also a solution. This means that the optimal solution for \( W \) is not unique.

For non-gaussian linearities, iterative algorithms of the form

\[
W(k + 1) = W(k) + \mu(k) \nabla Q(k)
\]

where \( \mu(k) \) is a step size control parameter and \( \nabla Q(k) \) is an empirical version of \( \nabla Q \), can be used in searching for the optimal parameters. One possibility is replacing the expectations in (9) and (10) by empirical averages over the input samples. Another is replacing them by the samples themselves. For instance, \( E\{S[W(k) x + t] x^T \} \) would be replaced by

\[
\frac{1}{M} \sum_{i=1}^{M} S[W(k)x^{(i)} + t(k)x^{(i)^T}],
\]

where \( x^{(i)} \) is the \( i \) th training input vector, or simply by

\[
S[W(k)x^{(k)} + t(k)x^{(k)^T}].
\]

For the steepest descent algorithm [12], \( \mu(k) \) is chosen so as to maximize \( Q(k) \) on each iteration. The convergence of the algorithm is analyzed in the appendix.

In the case

\[
G(u_i) = 0.5[1 + \tanh(u_i)]
\]

we have

\[
g(u_i) = \frac{0.5}{\cosh^2(u_i)}
\]

for which

\[
s(u_i) = -2 \tanh(u_i)
\]

The algorithm will be initialized at some \( W > 0 \). Positiveness of \( \det W \) then must be maintained throughout the search procedure. In all cases tested in actual application, we found that \( \det W \) remained positive without invoking constraints. Replacing the expectation by the sample mean, the closed form of \( W \) obtained for the gaussian nonlinearity make the use of such neuronal function attractive. It would be particularly suitable for cases where a gaussian density may be assumed for the input vector. In other cases, when, say, the tail density cannot be assumed to be as thick as that for the gaussian one, and another nonlinearity, such as the hyperbolic tangent, is more appropriate, the gaussian solution may serve as an initial condition. The gain in using a non-gaussian nonlinearity beyond initialization can be measured by an increase in the value of the output entropy coupled with a decrease in the gradient norm during the optimization process. Such trends were observed in each of the cases where we applied the proposed method to real problems.
Application to classification

An immediate application of pdf estimation is in solving classification problems. In training, the pdf corresponding to each of the classes is learnt by a different set of parameters. In operation, the class corresponding to the largest pdf is selected. This is demonstrated by the following example.

Example 1: Diamond evaluation can be formulated as a classification problem in which a diamond is to be assigned one of 36 of color grades \([14]\). The intensities of the red, green and blue components of the reflection light and the same components of the illumination light constitute a six-dimensional real-valued input vector. In \([14]\), a set of 225 diamonds, pre-classified by a certain widely recognized institute, were randomly divided into several pairs of training and testing sets. Several network architectures and training methods were tried. A single-cell perceptron assigned to each of the grades performed poorly, indicating that the data is not linearly separable. The back-propagation method applied to several network architecture also failed to produce satisfactory results. Applying a multi-cell perceptron, consisting of a single internal layer of 2400 spherical cells per class, produced a success rate of 81 percent (by the most severe measure, which does not forgive misclassification to neighboring grades on the colour scale). The number of internal cells and their threshold value used in \([14]\) were chosen so as to satisfy a minimal covering requirement on the input space.

Applying the classifier proposed in the present work to the diamond evaluation problem, we achieved a success rate of 81 percent with 6 neurons per class (using \(\mu = 0.1\) and allowing the algorithm run until the norm of the gradient of \(W\) became smaller than 5 percent of the norm of \(W\)). There was a considerable saving in both the execution time and the memory requirement with respect to those of the previous design, with the same rate of success.

3 Estimating Random Variables and Sequences

Maximum a posteriori probability

Let \(\alpha\) be a random variable, let \(\hat{x} = [x_1, \ldots, x_{n-1}]^T\) be a vector of \(n\) random variables having a joint pdf \(p_x(\hat{x})\), and let the joint pdf of \(\hat{x}\) and \(\alpha\) be denoted \(p_{\hat{x},\alpha}(\hat{x}, \alpha)\). The maximum density estimate of \(\alpha\) given \(\hat{x}\) is obtained by maximizing the corresponding conditional pdf

\[
p(\alpha \mid \hat{x}) = \frac{p_{\hat{x},\alpha}(\hat{x}, \alpha)}{p_{\hat{x}}(\hat{x})}
\]

with respect to \(\alpha\), which is the same as maximizing \(p_{\hat{x},\alpha}(\hat{x}, \alpha)\) with respect to \(\alpha\).

In real problems, \(p_{\alpha,\hat{x}}(\alpha, \hat{x})\) is not available and must be estimated from the data. Defin-
\[ x = (\hat{x}^T, \alpha)^T \]

the proposed estimate of the pdf of \( x \) is

\[ \hat{p}_x(x) = (\det W) \prod_{i=1}^{n} g(w_i x + t_i) \]

or, using the normalized neuron

\[ \hat{p}_x(x) = \prod_{i=1}^{n} g(w_i^T x + t_i) \]

where, as before, \( w_i \) is the \( i \)’th row of the weights matrix obtained in estimating \( p_x(x) \) by the algorithm described in the previous section.

Maximizing \( \hat{p}_x(x) \) is equivalent to maximizing \( \log p_x(x) \), which is, in turn, equivalent to maximizing

\[ f(x) = \sum_{i=1}^{n} \log g(w_i x + t_i) \]

This applies to both the normalized and the unnormalized versions of the neuron functions and, consequently, the algorithm developed next will apply to both.

Newton’s iterative optimization algorithm for maximizing \( f(x) \) with respect to the estimated variable \( \alpha \) is [12]

\[ \alpha(k) = \alpha(k - 1) - \left[ \frac{\partial^2 f(x)}{\partial x^2} \right]^{-1} \frac{\partial f(x)}{\partial x} \bigg|_{x=x(k-1)} \]

(14)

In our case

\[ \frac{\partial f(x)}{\partial \alpha} = \sum_{i=1}^{n+1} \frac{\partial}{\partial \alpha} \log g(w_i x + t_i) \]

and

\[ \frac{\partial^2 f(x)}{\partial \alpha^2} = \sum_{i=1}^{n} \frac{\partial^2}{\partial \alpha^2} \log g(w_i x + t_i) \]

Using \( G(u_i) = 0.5[1 + \tanh(w_i x + t)] \), we have

\[ g(w_i x + t) = \frac{0.5}{\cosh^2(w_i x + t_i)} \]

It follows that

\[ \frac{\partial}{\partial \alpha} \log g(w_i x + t_i) = -2w_i \tanh(w_i x + t_i) \]

and

\[ \frac{\partial^2}{\partial \alpha^2} \log g(w_i x + t) = \frac{-2w_i^2}{\cosh^2(w_i x + t_i)} \]
The iterative algorithm (14) becomes
\[
\alpha(k) = \alpha(k-1) - \mu(k-1)w^{(n)T}z(k-1)
\]  
(15)
where
\[
\mu(k-1) = \left[ \sum_{i=1}^{n} \frac{w_{i,n}^2}{\cosh^2(w_i x + t_i)} \right]^{-1}
\]
\(w^{(n)}(k)\) is the last column of the weights matrix \(W(k)\) and \(z(k-1)\) is the vector whose \(i\)'th component is
\[
z_i(k-1) = \tanh\left( u_i(k-1) \right)
\]
where \(u_i(k) = w_i x(k) + t\).

**A linear estimator**

The estimator proposed before requires performing several optimization steps for finding \(\alpha\) which maximizes the estimated joint density of \(\alpha\) and \(\hat{x}\). Although we have found that the number of iterations for satisfactory convergence is relatively small (compared to that required for optimizing the network parameters), this may make the algorithm unsuitable for predicting a fast evolving process in “real-time”. An approximate implementation of the conditional expectation estimator, which facilitates a real-time implementation is described next.

We restrict the connectivity of the density shaping network (that is, the virtual network used for optimizing the estimator’s parameters) to be the one described by Fig. 4. The given vector \(\tilde{x}\) is fully connected to a group of \(n - 1\) cells through a connection weights matrix \(\tilde{W}\). In training for estimating \(\alpha\), the value of \(\alpha\) is connected only to the \(n\)'th cell, through a connection weight \(w_{n,n}\). The latter cell is also connected to the elements of \(\tilde{x}\), as shown in the figure. Denoting, as before, \(x = [\tilde{x}, \alpha]^T\), and the connectivity matrix between \(x\) and the
internal layer of the density shaping network \( W \), it follows readily that

\[
\det W = w_{n,n} \det \tilde{W}
\]

where \( \tilde{W} \) is the submatrix of the weights matrix \( W \), corresponding to the connections between the inputs \( x_1, \ldots, x_n \) and the outputs \( y_1, \ldots, y_n \) of the density shaping network. Let \( \hat{u}_i = \tilde{W}_i \hat{x} + \hat{t} \), with \( \hat{t} = [\hat{t}_1, \ldots, \hat{t}_{n-1}]^T \). Then the estimated densities are

\[
\hat{p}_x(\hat{x}) = \det \tilde{W} \prod_{i=1}^{n-1} g(\hat{u}_i) \tag{16}
\]

and

\[
\hat{p}_x(x) = \det W \prod_{i=1}^{n} g(u_i) = w_{n,n} g(w_n^T x) p_x(x) \tag{17}
\]

where \( \hat{w}_i \) and \( w_i \) are the input weights vectors of the \( i \)th cell, connecting it to \( \hat{x} \) and \( x \), respectively.

The estimated conditional density function of \( \alpha \) given \( \hat{x} \) is

\[
\hat{p}(\alpha \mid \hat{x}) = \frac{\hat{p}_x(x)}{\hat{p}_x(\hat{x})} = w_{n,n} g(w_n^T x) \tag{18}
\]

It can be readily verified that the same expression would be obtained when the neuron function is normalized as described in the previous section. The conditional expectation of \( \alpha \) given \( \hat{x} \) is

\[
\hat{\alpha} = E\{\alpha \mid \hat{x}\} = \int_{-\infty}^{\infty} \alpha \hat{p}(\alpha \mid \hat{x}) d\alpha \tag{19}
\]

An approximation of this estimator is

\[
\hat{\alpha} = \int_{-\infty}^{\infty} \alpha \hat{p}(\alpha \mid \hat{x}) d\alpha \tag{20}
\]

Denoting

\[
\beta = \sum_{i=1}^{n-1} w_{n,i} x_i \quad \text{and} \quad \nu = w_n^T x = w_{n,n} \alpha + \beta
\]

we have

\[
\hat{\alpha} = \frac{1}{w_{n,n}} \int_{-\infty}^{\infty} (\nu - \beta) \hat{p}(\alpha \mid \hat{x}) d\alpha = \frac{1}{w_{n,n}} \int_{-\infty}^{\infty} \nu \hat{p}(\alpha \mid \hat{x}) d\alpha - \frac{\beta}{w_{n,n}}
\]

Assuming that \( g(\cdot) \) is symmetrical with respect to the origin (as is the case with the functions considered in this work), we have

\[
\frac{1}{w_{n,n}} \int_{-\infty}^{\infty} \nu \hat{p}(\alpha \mid \hat{x}) d\alpha = \int_{-\infty}^{\infty} \nu g(\nu) d\nu = 0
\]

hence

\[
\hat{\alpha} = -\frac{\beta}{w_{n,n}} = \frac{1}{w_{n,n}} \sum_{i=1}^{n-1} w_{n,i} x_i
\]
Figure 3: Predicting the sunspot process. The broken line represents the process, while the solid one corresponds to the (10 steps ahead) predictions.

The resulting estimator is, then, linear. This form is particularly useful for the prediction of a process, which is discussed next.

**Application to forecasting**

The problem of predicting a future value, $x_{j+k}$ of a stationary random sequence $x_1, x_2, \ldots$, at instance $j$ given the last $n-1$ values is naturally addressed by both the maximum density and the conditional expectation estimation methods described before. Simply define $\hat{x} = (x_{j-n+2}, \ldots, x_j)^T$ and $\alpha = x_{j+k}$. The stationarity assumption cannot be strictly verified in practice and certain sequences will qualify better than others. A popular test case is examined next.

**Example 2:** Monthly mean sunspot data, presented in [15], were normalized into the interval [0 1] and centered by subtracting the mean, so as to achieve a good numerical behavior of the algorithm and to eliminate the need for calculating a threshold for the sigmoidal units. First, a maximum density estimator based on the proposed method was used to predict, at time $j$, the value of $x_{j+10}$ given $x_j, x_{j-1}, \ldots, x_{j-9}$. For each time step the algorithm (15) converged (the correction term becoming smaller than $10^{-6}$) in less than 5 iterations. The first 1500 points were used for training. Performance on the next 200 points and the next 1200 points is shown in Fig. 5. The average prediction error on 1200 points was 0.0790. This compares with an average error of 0.0837 achieved by a predictor which uses the present value of the process as the prediction and to an average error of 0.0947 achieved by a predictor which uses the average of the last ten values as the prediction. We also note that the average error
of a predictor based on the sample mean of the entire data set, as suggested by Stigler for a large variety of estimation problems on real data [16], was 0.1309. The performance of the proposed estimator compares favorably, then, with these simpler estimators.

Next, we used the linear predictor to perform the same task. The average error was 0.0757 – smaller than the one produced by the maximum density estimator. While the advantages in the implementation simplicity and in the execution time of the predictions themselves are obvious, the training time of this predictor is considerably longer than that for the maximum density predictor due to the connectivity constraints which burden the optimization process.

4 Conclusion

We have presented a method for estimating the probability density function of a random vector. It is based on finding the parameter values of a “shaping network”, consisting of a single internal layer of sigmoidal unit, so that the output entropy is maximized. A recursive maximum density estimator for a random variable, based on the estimated pdf, has been derived, along with a constrained conditional expectation estimator, having a linear structure. While we have considered, in particular, gaussian and hyperbolic tangent nonlinearities which are particularly suitable for modeling unimodal densities, other sigmoidal transfer functions may be attempted so as to better fit the data.

Appendix: Convergence of the Training Algorithms

We analyze the convergence of the recursive algorithm (13). For simplicity, we consider the case of scalar independent identically distributed (iid) inputs, but the analysis is extendable to vector-valued inputs and to stationary sequences. We also assume for simplicity that the optimal threshold in the neuron function is 0, which, as explained before, corresponds to a symmetric input density, a symmetric nonlinearity and a centered input. This assumption, too, can be readily removed. We prove convergence under certain conditions, and show that the conditions hold for a certain case of interest (hyperbolic tangent nonlinearity).

Let \( x_i, i = 1, \ldots, N \) be independent identically distributed input variables with zero mean and variance \( \sigma^2 \). The optimal weight \( w \) satisfies

\[
0 = \frac{1}{w} + E \{b(wx)x\}
\]

(21)

We define

\[
r(w, x) = s(wx)(wx) \quad \text{and} \quad R(w) = E\{r(w, x)\}
\]

13
Assuming that \( w \neq 0 \), (21) can be written as
\[
0 = 1 + R(w)
\]  
(22)

The \( k \)'th iteration of steepest descent algorithm for the weight \( w \), is given by
\[
w_{k+1} = w_k + \alpha_k (1 + E \{ b(w_k x)(w_k x) \}) (w_k)^{-1}
\]  
(23)

which assumes that \( w_k \neq 0 \) for all \( k \).

Given a training set \( x_i, \ i = 1, \ldots, N \), (22) is replaced by
\[
0 = 1 + R_N(w)
\]  
(24)

where
\[
R_N(w) = \frac{1}{N} \sum_{i=1}^{N} \{ s(w x_i) w x_i \}
\]  
(25)

Denoting by \( w^* \) the solution of (22) and by \( w^\dagger \) the solution of (24), we have the following:

**Theorem** The iterative solution of (24) converges to the solution of (22) in the sense that, for every \( \delta > 0 \), \( \lim_{N \to \infty} P\{|w^\dagger - w^*| < \delta\} = 1 \), if the following conditions are met:

1. \( E\{R(w)\} \) is twice continuously differentiable with respect to \( w \), and the order of differentiation and expectation can be interchanged.
2. \( \frac{\partial R(w)}{\partial w} \) is continuous in a neighborhood of \( w^* \).
3. \( \left| \frac{\partial R(w)}{\partial w} \right|_{w^*} \geq K_1 \).
4. \( \frac{\partial^2 R(w)}{\partial w^2} \) is continuous in a neighborhood of \( w^* \).
5. \( E\{r^2(w, x)\} \) is \( K_3 w^\dagger \).
6. \( K_4 \leq w_k \leq K_5 \) for all \( k \).

where \( K_1, \ldots, K_5 \) are positive scalars and \( l \) is a positive integer.

**Proof** First we show that for a given \( N \) the iterative solution of (24) converges. Then we show that for \( N \to \infty \) the solution of (24) tends to the solution of (21) with probability 1.

The function to be maximized is \( Q(w) = E \{ \log \left| \frac{\partial G(w)}{\partial w} \right| \} = E \{ \log |w| + \log \det(g(w x)) \} \). If \( w_k > K_6 > 0 \) for all \( k \), then the weight update rule (23) guarantees that \( w_k \) is bounded (see [12] Section 7.6). Also, by the assumptions of the theorem, \( Q(w) \) is a continuous function, hence, the global convergence of the steepest descent method holds (see [12], Section 7.6 and references therein).
The central limit theorem [17 Theorem 5.1.3] implies convergence in probability

$$
\left( \frac{1}{N} \sum_{i=1}^{N} r(w, x_i) - R(w) \right) \xrightarrow{P} \mathcal{N} \left( 0, \frac{\sigma^2}{N} \right)
$$

(26)

where \( \xrightarrow{P} \) denotes convergence in probability and the right hand side of (26) represents a normal distribution with mean 0 and variance \( \sigma^2/N \). Denoting

$$
\Delta(w) = \left( \frac{1}{N} \sum_{i=1}^{N} r(w, x_i) - R(w) \right) = R_N(w) - R(w)
$$

(27)

the equation to be solved is

$$
0 = 1 + R(w) + \Delta(w)
$$

(28)

Since, by the assumptions of the theorem, \( \sigma^2 \leq Mw^t \), we have by Corollary (5.1.1) of [17]

$$
\lim_{N \to \infty} P(|\Delta(w)| < \epsilon) = 1
$$

Denoting the solution of (22) by \( w^* \) and the solution of (28) by \( w^t \) we now show that

$$
\lim_{N \to \infty} P(|w^t - w^*| > \delta) = 0
$$

To this end we define the function \( z = R(w) \) and the inverse function \( w = D(z) = R^{-1}(z) \). By (22) we have \( z^* = R(w^*) = -1 \). By Taylor’s theorem ([18] Theorem 5.15), if \( \frac{\partial D}{\partial z} \) is continuous on an interval \([a, b]\) such that \(-1 \in [a, b]\) and \( \frac{\partial^2 D}{\partial z^2}(t) \) exists for every \( t \in (a, b) \) then

$$
D(-1 - \eta) = D(-1) + \frac{\partial D(z)}{\partial z} \bigg|_{z=-1} (-\eta) + \frac{1}{2} \frac{\partial^2 D}{\partial z^2} \bigg|_{z=z_0} \eta^2
$$

(29)

for some \( z_0 \in [-1, -1 - \eta] \). Noting that

$$
\frac{\partial D(z)}{\partial z} \bigg|_{z=-1} = \frac{1}{\frac{\partial R(w)}{\partial w} \bigg|_{w=w^*}}
$$

and that

$$
\frac{\partial^2 D}{\partial z^2} \bigg|_{z=z_0} = -\left( \frac{\partial R(w)}{\partial w} \bigg|_{w=w^*} \right)^3 \frac{\partial^2 R(w)}{\partial w^2} \bigg|_{w=w_0}
$$

where \( z_0 = R(w_0) \), (29) can be rewritten as

$$
R^{-1}(-1 - \eta) - R^{-1}(-1) = (-\eta) \left( \frac{\partial R}{\partial w} \bigg|_{w=w^*} \right)^{-1} - \frac{\eta^2}{2} \left( \frac{\partial R}{\partial w} \bigg|_{w=w_0} \right)^{-3} \left( \frac{\partial^2 R}{\partial w^2} \bigg|_{w=w_0} \right)
$$

(30)

Taking the absolute value of both sides of (30) and using the assumptions of the theorem, we have

$$
|R^{-1}(-1 - \eta) - R^{-1}(-1)| \leq |\eta| \left( \frac{1}{K_1} + \frac{\eta}{2(K_1)^3} \right)
$$

15
For every $\delta$ we can find an $\epsilon$ such that

$$\epsilon \left( \frac{1}{K_1} + \epsilon \frac{K_2}{2(K_1)^3} \right) \leq \delta$$

Therefore,

$$\lim_{N \rightarrow \infty} P(|w^i - w^*| < \delta) \geq P(|R_N(w) - R(w)| < \epsilon) = \lim_{N \rightarrow \infty} Q \left( \frac{\sqrt{N \epsilon}}{\sqrt{Var(R(w))}} \right) = 1$$

where $Q(x) = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}} \exp \left( -\frac{t^2}{2} \right) dt$.

**Example** We now show that the neuron transfer function $G(x) = \tanh(wx)$ satisfies the assumptions of the convergence theorem. It follows from Theorem 9.42 of [18] (p 236) that differentiation and expectation can be interchanged in this case. In this case

$$|R(w)| = E\{\tanh(wx)(wx)\}$$

$$\frac{\partial R}{\partial w} = wE\left\{ \frac{x^2}{\cosh^2(wx)} \right\} + E\{x \tanh(wx)\}$$

Since the the arguments of the expectations are nonnegative, the condition on $R'$ is satisfied. Taking the second derivative we obtain

$$\frac{\partial^2 R}{\partial w^2} = 2E\left\{ \frac{x^2}{\cosh^2(wx)} \right\} - wE\left\{ \frac{x^2}{\cosh^2(wx)} x \tanh(wx) \right\} \leq wE\{x^2\} + E\{|x|^3\}$$

hence, the conditions on the second derivative are met if $E\{|x|^3\}$ is bounded. The variance of $r(w, x)$ is finite in this case as well since

$$E\{[\tanh(wx)(wx)]^2\} \leq w\sigma^2$$

**References**


Technion - Israel Institute of Technology
Computer Science Department
LIST OF CIS REPORTS

CIS 9001
Guy Foux, Michael Heymann, and Alfred M. Bruckstein: “Two-Dimensional Polygon Navigation Among Unknown Stationary Polygonal Obstacles”.

CIS 9002

CIS 9003

CIS 9004

CIS 9005

CIS 9006

CIS 9007

CIS 9008

CIS 9009

CIS 9010

ANNUAL REPORT, 1990.

CIS 9011
CIS 9012

CIS 9101

CIS 9102

CIS 9103

CIS 9104

CIS 9105

CIS 9106

CIS 9107

CIS 9108

CIS 9109

CIS 9110

CIS 9111
CIS 9201

CIS 9202

CIS 9202 (Revised, June 1992)

CIS 9202 (2nd Revision, April 1993)

CIS 9203

CIS 9204

CIS 9205

CIS 9206

CIS 9207

CIS 9208

CIS 9209

CIS 9210

CIS 9211

CIS 9212
CIS 9213

CIS 9214

CIS 9215

CIS 9216

CIS 9217

CIS 9218

CIS 9219

CIS 9220

CIS 9221

CIS 9222

CIS 9301

CIS 9302
CIS 9303

CIS 9304

CIS 9305

CIS 9306

CIS 9307

CIS 9308

CIS 9309

CIS 9310

CIS 9311

CIS 9312

CIS 9313
Y. Barak: “Pattern Correction and Association and Sequence Regeneration by Correlative Memory”, March 1993.

CIS 9314
R. Kimmel and A.M. Bruckstein: “Sub-Pixel Distance Maps and Weighted Distance Transforms”, April 1993.

CIS 9315
CIS 9316

CIS 9317

CIS 9215

CIS 9318

CIS 9319


CIS 9320

CIS 9321

CIS 9322

CIS 9323

CIS 9324

CIS 9325

CIS 9326

CIS 9327
CIS 9328

CIS 9329

CIS 9330

CIS 9331

CIS 9332

CIS 9401

CIS 9402

CIS 9403

CIS 9404

CIS 9405

CIS 9406

CIS 9407

CIS 9408
CIS 9409

CIS 9410

CIS 9411

CIS 9412

CIS 9413

CIS 9414

CIS 9415

CIS 9416

CIS 9417

CIS 9419

CIS 9420
Y. Baram and Z. Roth: “Density Shaping by Neural Networks with Application to Classification, Estimation and Forecasting”, December 1994.