Submitted to: *IEEE Transactions on Neural Networks.*
MIT Media Laboratory Vision and Modeling Group Technical Report #188

On Training Gaussian Radial Basis Functions for Image Coding

Alex Sherstinsky[†‡]
Rosalind W. Picard[‡]

[†]Department of Electrical Engineering and Computer Science
[‡]The Media Laboratory

February, 1992

Abstract

The efficiency of the Orthogonal Least Squares (OLS) method for training approximation networks is examined using the criterion of energy compaction. We show that the selection of basis vectors produced by the procedure is not the most compact when the approximation is performed using a non-orthogonal basis. Hence, the algorithm does not produce the smallest possible networks for a given approximation error. Specific examples are given using the Gaussian Radial Basis Functions (RBF) type of approximation networks. A new procedure that finds the most compact subset of non-orthogonal basis vectors is described and used to evaluate the performance of OLS in image coding. The new procedure also permits a comparison of the Gaussian RBFs to the Discrete Cosine Transform (DCT), an orthogonal basis commonly used in image coding. This comparison shows that in terms of efficiency, the Gaussian RBFs can perform close to the DCT. Differences in perceptual distortion produced by the two coding techniques are also discussed.

Keywords: approximation networks, Radial Basis Functions, interpolation, approximation, Linear Least Squares Estimation, Gram-Schmidt, Orthogonal Least Squares, Karhunen-Loève Transform, Gaussians, Discrete Cosine Transform, energy compaction, image coding, blocking effect, point defects.

Inquiries should be addressed to: Alex Sherstinsky, MIT Media Laboratory, E15-383, 20 Ames Street, Cambridge, MA 02139. E-mail: shers@media.mit.edu

1 Introduction

Recently, a number of feed-forward networks with one hidden layer of processing units have been proven to possess the ability to approximate any continuous function arbitrarily well [1], [2]. Even though these networks differ in the type of the processing unit they use (e.g., sigmoids, Gaussians, etc.), they all attain any desired accuracy by interconnecting a sufficiently large number of nodes. However, a proof of the best approximation property does not automatically yield a practical algorithm for training the connection weights.

One such approximation scheme, the Radial Basis Function (RBF) network, has been used as a classifier [3] – [7]. The training problem for an RBF network can be viewed as interpolation and solved by inverting a matrix. But this approach often causes numerical problems, because the matrices involved are typically large. This problem has led to several alternatives aimed at reducing the training complexity without significant losses in approximation accuracy [8] – [10].

This report analyzes the efficiency and applications to image coding of one such method. The method analyzed is the Orthogonal Least Squares (OLS), proposed by Chen *et al* [10]. Section 2 reviews the RBF approximation problem and the OLS algorithm for solving it. Section 3 presents the compaction criteria, which are subsequently used in Section 4 to analyze the efficiency of the OLS method. Examples using Gaussian RBFs are also given in Section 4. Section 5 introduces an energy efficient training procedure, which is used to evaluate the performance of the OLS method in image coding. The Gaussian RBFs are compared to the Discrete Cosine Transform (DCT) in terms of the energy efficiency and the attendant visual artifacts. Finally, Section 6 summarizes the present study.

2 Background

2.1 Radial Basis Functions

A non-linear function $h(\vec{x},\ \vec{c})$, where $\vec{x}$ is the independent variable and $\vec{c}$ is the constant parameter, is called a Radial Basis Function (RBF) when it depends only on the radial distance $r = ||\vec{x} - \vec{c}||$, where $\vec{c}$ is its "center". The RBF method is one of the possible solutions to the real multivariate interpolation problem, stated as follows [11] – [14]:

Interpolation Problem: *Given N different points $\{\vec{x_i} \in \mathcal{R}^d \mid i = 1,\ \ldots,\ N\}$, where d is the number of dimensions, and N real numbers $\{y_i \in \mathcal{R} \mid i = 1,\ \ldots,\ N\}$, find a function F from $\mathcal{R}^d$ to $\mathcal{R}$ satisfying the interpolation conditions:*

$$F(\vec{x_i}) = y_i, \quad i = 1,\ \ldots,\ N. \tag{1}$$

The RBF approach consists of choosing the function F to be an expansion of the form

$$F(\vec{x}) = \sum_{j=1}^{N} w_j h(||\vec{x} - \vec{c_j}||), \tag{2}$$

where the centers of the expansion $\vec{c_j} = \vec{x_j}$ must be the the known data points, and $\{w_j \in \mathcal{R} \mid j = 1,\ \ldots,\ N\}$ are the corresponding weights.

The unknown weights can be recovered by imposing the interpolation conditions. An RBF matrix $H \in \mathcal{R}^{N \times N}$ is constructed by evaluating $h(||\vec{x_i} - \vec{c_j}||)$ at each x_i and c_j; $i,\ j = 1,\ \ldots,\ N$:

$$\begin{aligned} H &= [h_{ij}], \\ h_{ij} &= h(||\vec{x_i} - \vec{c_j}||). \end{aligned} \tag{3}$$

In other words, each column of H is a basis vector corresponding to a particular center. The resulting linear system

$$H\vec{w} = \vec{y} \tag{4}$$

can be solved if H^{-1} exists:

$$\vec{w} = H^{-1}\vec{y}. \tag{5}$$

From (5), a necessary and sufficient condition for the existence of a unique solution to the interpolation problem is the invertibility of the matrix H. The RBF matrix will be invertible if the column vectors of H form a basis in $\mathcal{R}^N$. This condition is satisfied for a number of RBFs, for example [12]:

$$\begin{aligned} h(r) &= \exp\{-(r/a)^2\} && \text{(Gaussian)} \\ h(r) &= (a^2 + r^2)^\beta, \quad \beta < 1 \\ h(r) &= r && \text{(linear)} \\ h(r) &= r^2 \log r && \text{(thin plate splines).} \end{aligned} \tag{6}$$

Figure 1 shows a realization of (2) in the form of a network with one layer of hidden units. Each hidden unit implements the same radial function, but with its own center as a parameter. According to the interpolation conditions, which led to (4), centers are the coordinates of data points, and the number of centers equals the number of data points [1]. Each hidden unit accepts inputs from all components of $\vec{x}$, computes the norm, and

[1] Subsequent sections address the issue of reducing N to M.

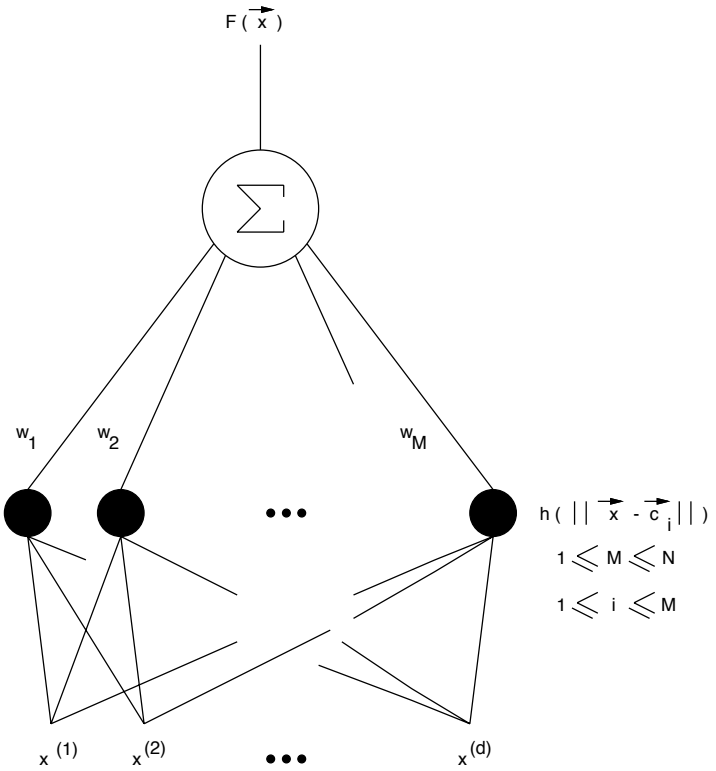


Figure 1: Schematic of an RBF network. The subscripts denote the indices of RBF centers; the superscripts denote the components of the input vector.

produces an output in accordance with the given RBF behavior. The outputs of all hidden units are weighted by the coefficients $\vec{w}$ and summed to give $F(\vec{x})$. Since each radial hidden unit defines a $(d+1)$-dimensional hypersurface, the RBF network interpolates by reconstructing the data with scaled hypersurfaces. The examples in this report employ a special case of RBFs, multivariate Gaussians of constant variance, i.e., (6) with variance $\sigma^2 = \frac{a^2}{2}$, in which case the level sets of the reconstructing hypersurfaces are the d-dimensional hyperspheres.

2.2 Training RBF Networks

In most applications, N is large, deeming the direct use of (5) impractical. However, a well-known result allows dimensionality reduction to $M < N$. Starting with $H \in \mathcal{R}^{N\times N}$, which is a basis in $\mathcal{R}^N$, $\hat{H}$ is obtained by selecting $M = N - k$, $k = 1, \ldots, N$ basis vectors from H, such that $\hat{H} \in \mathcal{R}^{N\times M}$. Then the product $(\hat{H}^T\hat{H}) \in \mathcal{R}^{M\times M}$ is an invertible matrix and thus a basis in $\mathcal{R}^M$ [15]. Using this result, an approximation to (4) can be formulated and solved by the method of Linear Least Squares Estimation (LLSE) [8]:

Approximation Problem: *Given $\hat{H} \in \mathcal{R}^{N\times M}$ and $\vec{y} \in \mathcal{R}^N$, related by*

$$\vec{y} = \hat{H}\vec{w} + \vec{e}, \tag{7}$$

find an optimal coefficient vector $\vec{w} \in \mathcal{R}^M$ such that the error energy $\vec{e}^T\vec{e}$ is minimized. This can be equivalently stated as: Find $\{w_j^\star \in \mathcal{R} \mid j = 1, \ldots, M\}$ such that $w_j = w_j^\star$ solves

$$\min_{w_j} (y_i - \sum_{j=1}^{M} w_j h(||\vec{x_i} - \vec{c_j}||))^2, \qquad i = 1, \ldots, N. \tag{8}$$

In contrast with the interpolation problem, the approximation problem does not require the centers $\vec{c_j}$ to coincide with $\vec{x_j}$, so one may choose any $\vec{c_j} \in \mathcal{R}^d$. However, the centers are commonly chosen to be a subset of data points.

The data is subsequently approximated using

$$\vec{y_a} = \hat{H}\vec{w}^\star \tag{9}$$

or, equivalently, using

$$F_a(\vec{x}) = \sum_{i=1}^{M} w_i^\star h(||\vec{x} - \vec{c_i}||), \qquad M \leq N, \tag{10}$$

where $\vec{y_a}$ and $F_a(\vec{x})$ are the approximated values of the data samples and the generalizing function, respectively.

The well-known LLSE optimal solution is in the form of (5):

$$\vec{w}^\star = \hat{H}^+\vec{y}, \tag{11}$$

where $\hat{H}^+$ is the pseudoinverse of $\hat{H}$:

$$\hat{H}^+ = (\hat{H}^T\hat{H})^{-1}\hat{H}^T. \tag{12}$$

If an RBF network with $M << N$ centers adequately approximates the data, then the above approach provides a computationally efficient procedure for determining the weights. However, arbitrarily selecting the centers from data points often results in poor performance in a sense that the networks end up with more nodes than necessary for a desired accuracy of approximation [10].

2.3 Orthogonal Least Squares

In order to improve the performance of an RBF network trained by solving the approximation problem, a judicious selection of centers is needed. It has been reported in [10] that the approximation problem, stated in (7), lends itself to the Orthogonal Least Squares (OLS) method, which is a recursive algorithm for selecting a suitable subset of data points as centers. A basis vector produced at each step of the procedure maximizes the increment of the explained energy of the desired output.

We now review the process of center selection performed by OLS using the concept of permutation matrices.

Definition 2.1 *A permutation of $H \in \mathcal{R}^{N\times N}$ is $H' \in \mathcal{R}^{N\times N}$ such that each column vector of H' is identical to exactly one column vector of H. Formally,*

$$H' = HP,$$

where $P \in \mathcal{R}^{N\times N}$ is a permutation matrix comprised of the column vectors of the identity matrix, whose positions are arranged in one of $N!$ possibilities.

Definition 2.2 *A selection matrix $S \in \mathcal{R}^{N\times M}$ is obtained by selecting*
$M = N - k$, $k = 1, \ldots, N$ column vectors of a particular permutation matrix
$P \in \mathcal{R}^{N\times N}$.

In OLS, a selection of the original RBF matrix H is obtained and orthonormalized using the classical Gram-Schmidt process (GS), summarized here for reference [2].

[2] For a detailed treatment, consult a standard linear algebra text, such as [15], [16], etc.

Let $\vec{a_i}$ be the column vectors of $A \equiv HS$. The GS process finds $A = \hat{Q}R$, where the matrix $\hat{Q} \in \mathcal{R}^{N \times M}$ consists of orthonormal column vectors, and the right-triangular matrix $R \in \mathcal{R}^{M \times M}$ contains projection and normalization coefficients computed by GS:

$$\left.\begin{array}{rcl} \vec{v_i} &=& \vec{a_i} - \sum_{l=1}^{i-1} r_{li}\vec{q_l}, \\ r_{li} &=& \vec{q_l}^T \vec{a_i}, \\ r_{ii} &=& ||\vec{v_i}||, \\ \vec{q_i} &=& \frac{\vec{v_i}}{r_{ii}}. \end{array}\right\} \begin{array}{rcl} i &=& 1, \ldots, N; \\ l &=& 1, \ldots, i-1; \\ M &\leq& N \end{array} \quad (13)$$

Using the selection matrix notation, the approximation problem, stated in (7), takes the following form:

$$\vec{y} = HS\vec{w} + \vec{e}, \quad (14)$$

$$HS = \hat{Q}R, \quad (15)$$

$$\vec{y} = \hat{Q}R\vec{w} + \vec{e}, \quad (16)$$

where $\vec{w} \in \mathcal{R}^M$ is the coefficient vector and $\vec{e} \in \mathcal{R}^N$ is the error vector. By defining

$$\vec{g} = R\vec{w}, \quad (17)$$

we obtain an orthonormal expansion of the data vector:

$$\vec{y} = \hat{Q}\vec{g} + \vec{e}. \quad (18)$$

Since (18) is a special case of the approximation problem, due to the orthonormality, its LLSE solution is particularly simple (and well-known):

$$\vec{g} = \hat{Q}^T \vec{y}, \quad (19)$$

from which $\vec{w}$ can be recovered by back substitution:

$$R\vec{w} = \vec{g}. \quad (20)$$

Since $\hat{Q}^T\hat{Q} = I$, the M-dimensional identity matrix, then

$$\begin{aligned} \vec{y}^T\vec{y} &= \vec{g}^T\vec{g} + \vec{e}^T\vec{e} \\ &= \sum_{j=1}^{M} g_j^2 + \vec{e}^T\vec{e} \\ &= \sum_{j=1}^{M} (\vec{q_j}^T\vec{y})^2 + \vec{e}^T\vec{e}. \end{aligned} \quad (21)$$

The OLS algorithm begins with H consisting of $M = N$ RBF vectors $\vec{h_j}$, $j = 1, \ldots, N$ and produces $\hat{Q}$ consisting of $M \leq N$ orthonormal regressors [3] $\vec{q_i}$ as well as the selection matrix S. In fact, the key difference between OLS and GS is the computation of the selection matrix S. The OLS method finds S so that GS maximizes $g_i^2 = (\vec{q_i}^T\vec{y})^2$ at each step. In other words, on each iteration $i = 1, \ldots, M$ of the GS orthonormalization procedure, the OLS method selects from the remaining $N - i + 1$ choices the values j and $\vec{h_j}$ such that the resulting regressor $\vec{q_i}$ will give the largest possible energy g_i^2. The algorithm keeps track of the order in which the original basis vectors are selected to form HS by setting $s_{ji} = 1$. The selection procedure is terminated when the error energy has been reduced to the specified tolerance.

3 Efficiency Using the Energy Compaction Criterion

It is helpful to distinguish two approaches aimed at finding efficient bases: one is "variational", while the other is not. The variational approach allows the components of the basis vectors to depend on the data, and finds the optimal set of basis vectors, corresponding to some criterion and constraint. In contrast, the non-variational approach starts with fixed basis vectors and searches for a combination that best approximates the data.

In the context of the approximation problem, the criterion is typically the minimization of the mean-squared error, and "smoothness" of the solution is a possible choice for the constraint [14]. Alternatively, using the same criterion, but constraining the basis matrix to be orthogonal, and applying the variational approach leads to the method of "principal components".

3.1 Principal Components is Variational

It is well-known that the eigenvectors of the covariance matrix of the data are the "principal components", which form the basis that possesses the best energy compaction properties [17], [18]. This basis constitutes the Karhunen-Loève Transform (KLT), which decorrelates the data and maximizes the incremental energy (or variance, in the statistical sense) explained by each regressor. The KLT basis vectors are orthonormal, allowing the approximation problem, (7), to take the form of (18):

$$\vec{y} = \hat{Q}\hat{\vec{g}} + \vec{e}. \quad (22)$$

[3] We follow the terminology in which "regressor" denotes the orthonormal columns of Q, and "basis vector" is reserved for the columns of H.

Let $E[\vec{y}]$ be the expected value of a random vector $\vec{y}$. Then for a general stochastic vector, the principal components are the eigenvectors of the covariance matrix, $C_{\vec{y}}$, sorted in the order of decreasing eigenvalues λ_j (variance or energy):

$$\begin{aligned} C_{\vec{y}} &= E[(\vec{y}-E[\vec{y}])(\vec{y}-E[\vec{y}])^T] \\ &= Q\Lambda Q^T, \\ \Lambda &= \mathrm{diag}(\lambda_1 \ \ldots \ \lambda_N). \\ \text{Since } \vec{g} &= Q^T\vec{y}, \\ C_{\vec{g}} &= E[(\vec{g}-E[\vec{g}])(\vec{g}-E[\vec{g}])^T] \\ &= Q^T C_{\vec{y}} Q = \Lambda. \end{aligned}$$

Even though $\mathrm{trace}(C_{\vec{g}}) = \mathrm{trace}(C_{\vec{y}})$, meaning that the total energy is preserved by Q, the distribution of energy in $C_{\vec{g}}$ is more skewed towards the first few eigenvalues. This is a direct consequence of the fact that the KLT expansion is the solution of the variational problem with the mean-squared error criterion and the orthogonality constraint. Thus, the KLT is the most compact orthogonal basis, because it produces the most skewed $C_{\vec{g}}$.

The significance of the KLT is in its energy efficiency, and networks that "learn" the principal components of the data need the smallest number of processing units for a given amount of error [19]. However, if the data exhibits smoothness in a certain sense, a sinusoidal transform, such as the Discrete Cosine Transform (DCT) becomes the KLT [20], [18], and, since the DCT is non-variational, the burdensome task of computing the principal components can be alleviated. The DCT involves an expansion in M of the following (non-radial) basis functions:

$$\begin{aligned} h_{\vec{c}}(\vec{x}) &= \Pi_{i=1}^{d} b(c^{(i)}) \cos\left(\frac{\pi}{2N^{(i)}} c^{(i)}(2x^{(i)}-1)\right), \\ b(c^{(i)}) &= 1 - \tfrac{1}{2}\delta(c^{(i)}), \end{aligned} \tag{23}$$

where $x^{(i)}$, $c^{(i)} = 1, \ldots, N^{(i)}$; $i = 1, \ldots, d$, and $\delta(\cdot)$ is the Kronecker delta. Similarly to the RBF expansion, the DCT basis expansion also fits the network notation of Figure 1, but uses a different type of the processing node (not an RBF). The DCT is also the accepted standard in image coding. Hence, in Section 5, we will assume that the DCT is the KLT and use it to judge the overall efficiency of the RBF expansion in image coding.

3.2 OLS is Non-Variational

The objective of the OLS method is to find the smallest subset of a fixed original basis (while not exceeding the allowed approximation error); therefore, the choices available to the procedure are restricted to various combinations of the original basis vectors. Since the number of candidate subsets is finite, it is natural to view efficiency as a relative measure. Thus, we will adopt the following definition of energy compactness in order to evaluate the efficiency of the OLS method:

Definition 3.1 *Consider the following two schemes for approximating the same data:*

$$\begin{aligned} \vec{y} &= B_1 S_1 \vec{w_1} + \vec{e_1} \quad \textit{and} \\ \vec{y} &= B_2 S_2 \vec{w_2} + \vec{e_2}, \end{aligned}$$

where B_1 and B_2 are bases (i.e., each has an inverse and is capable of interpolating the data). Let S_1 and S_2 be the selection matrices (according to Definition 2.2) with M_1 and M_2 columns, respectively. Assume

$$\vec{e_1}^T\vec{e_1} = \vec{e_2}^T\vec{e_2}.$$

Then B_1 is more compact than B_2 if $M_1 < M_2$.

3.3 Deterministic KLT

In order to judge the energy compaction properties of the OLS method, it is helpful to consider the degenerate or "deterministic" case of the KLT. In the deterministic case, the "covariance matrix" of the data (after the sample mean has been removed) is $C_{\vec{y}} = \vec{y}\vec{y}^T$ and is of rank 1. The entire KLT basis is reduced to only one principal component, which becomes the normalized version of the data vector itself. Thus the energy compaction properties of any orthogonal basis can be judged by how well its vectors align with the data vector. The i-th regressor's energy, g_i^2, is related to the alignment via $g_i^2 = (\vec{q_i}^T\vec{y})^2$. Using this measure, a basis with good energy compaction properties will need only a small number of its vectors to be retained in order to explain the required percentage of the data energy. The remaining basis vectors, which align poorly with the data, can be discarded.

3.4 Orthogonality

A convenient property of an orthogonal basis is that the energy contributions of the component vectors are decoupled. A maximally compact permutation of an orthogonal basis matrix can be formed by computing the projections of the basis vectors onto the data and rearranging the column vectors in the order of decreasing energy. In this new matrix, the energy, g_i^2, of a basis vector (which, due to orthogonality, is also a regressor) as a function of its index, $i = 1, \ldots, M$ becomes monotonically decreasing. As $\vec{e}^T\vec{e}$ in (14) decreases, the basis

vectors of progressively smaller energy become involved in the approximation process as needed. It follows that a permutation of an orthogonal basis is the most compact if and only if g_i^2 is monotonically decreasing; no other permutation of the original basis matrix can yield the same error with a smaller M.

In the case of both GS and OLS, determining the energy efficiency is more complicated, because the starting basis is non-orthogonal and the basis vectors cannot be treated separately. A permutation of the basis matrix, whose regressors have monotonically decreasing g_i^2, no longer assures maximal energy compaction. As a consequence, for different error allowances different permutations of the original basis matrix will be the most compact. This will be illustrated with examples in Section 4.3.

4 Energy Compaction Provided by OLS

The significance of the most compact permutation is that it produces the smallest possible RBF network for a given error tolerance. Therefore, it is interesting to find out whether or not the selection performed by the OLS procedure is the best in terms of energy packing. It has been claimed in [10] that the OLS method is "parsimonious" in a sense that, given a required level of unexplained energy, it will pick no more basis vectors than needed, and thereby produce an RBF network with fewer nodes than one with randomly selected centers. However, we show that the selection made by OLS is not guaranteed to contain the smallest number of centers.

4.1 OLS is Always Efficient for Orthogonal Bases

It is unlikely that the OLS procedure will be used in practice on an orthogonal basis, because dealing with such a convenient basis requires only sorting its vectors according to the alignment criterion (see Section 3), with no need for GS. However, it is illuminating to examine this trivial case first, because it is the only one for which the OLS method is always energy efficient.

Clearly, since the basis is orthogonal, the GS part of OLS does not alter the components of the basis vectors, and $R = I$. The only action of OLS is computing the S matrix. The first step of the algorithm locates the basis vector with the largest alignment to the data vector. The second step locates the next largest, and so on. At the end, all basis vectors are sorted in the order of decreasing alignment, making the algorithm both parsimonious and energy efficient, simultaneously.

4.2 OLS is not Always Efficient for Non-Orthogonal Bases

If the RBF basis is non-orthogonal, the energy contributions of different basis vectors are mixed (i.e., not independent). Hence, for a general data vector, every step of the OLS procedure is unable to locate the regressor with maximal alignment in the global sense. In other words, even though every step yields a regressor with the largest possible alignment, "the largest" may not be large enough. Since the data vector is the principal component, the OLS algorithm is effectively trying to approximate this principal component as closely as possible at each local step, with no regard for the global energy distribution properties. In a sense, this method is analogous to pursuing "short term profits" as opposed to "long term profits".

Evidently, as the examples below indicate, it is possible to benefit from relaxing the restriction of maximal alignment between the regressor and the data at each step of GS. Admitting some basis vectors that produce regressors with poor alignment may steer GS toward future basis vectors that produce regressors with excellent alignment, such that the overall energy compaction is improved. However, there is no mechanism in OLS to decide ahead of time what the optimal permutation of H should be for a specified error value.

4.3 Examples Using Gaussian RBFs

The goal of this section is to illustrate some cases, where the OLS procedure does not select the optimal subset of basis vectors in the energy compaction sense. For clarity, the following examples use one-dimensional data and a 3×3 Gaussian RBF matrix with variance $\sigma = 1$:

$$H = \begin{bmatrix} 1 & 0.606531 & 0.135335 \\ 0.606531 & 1 & 0.606531 \\ 0.135335 & 0.606531 & 1 \end{bmatrix}.$$

4.3.1 Example 1

Let

$$\vec{y} = \begin{bmatrix} 190 \\ 80 \\ 200 \end{bmatrix},$$

which gives the total energy $\vec{y}^T\vec{y} = 82500$.

In the first step, the OLS procedure selects the second column of H, because it gives the largest projection onto $\vec{y}$. After one step of GS, performed on the remaining columns of H, the third column produces a regressor with the largest alignment. In the third step, the first column must be selected. Combining these steps yields the following permutation of H:

$$HP_{OLS} = \begin{bmatrix} 0.606531 & 0.135335 & 1 \\ 1 & 0.606531 & 0.606531 \\ 0.606531 & 1 & 0.135335 \end{bmatrix},$$

which produces regressors with the energies:

$$g_1^2 = 57728.1,\ g_2^2 = 3447.38,\ \text{and}\ g_3^2 = 21324.6,$$

respectively, and

$$\frac{g_1^2 + g_2^2}{\vec{y}^T\vec{y}} \approx 0.74.$$

For a given data vector, the OLS method always selects the same sequence of regressors, regardless of the desired error. In this case, all three basis vectors are needed in order to approximate as much as 75% of the total energy. However, if OLS were able to select the following permutation of H for this data:

$$HP_{opt} = \begin{bmatrix} 0.135335 & 1 & 0.606531 \\ 0.606531 & 0.606531 & 1 \\ 1 & 0.135335 & 0.606531 \end{bmatrix},$$

which produces regressors with the energies:

$$g_1^2 = 54253.2,\ g_2^2 = 17759.4,\ \text{and}\ g_3^2 = 10487.4,$$

respectively, and

$$\frac{g_1^2 + g_2^2}{\vec{y}^T\vec{y}} \approx 0.87,$$

only two basis vectors would be needed in order to approximate up to 87% of the total energy. Note that even though this permutation of H produces regressors with monotonically decreasing energies g_i^2, it still may not be optimal if it is desired to satisfy (i.e. just exceed) a different error value.

4.3.2 Example 2

In the following example, the first column vector of H is nearly orthogonal to $\vec{y}$. Therefore, one of the other two basis vectors (in this case, the second) must produce a regressor that is nearly parallel to $\vec{y}$, resulting in a large value of the energy. The OLS misses this opportunity, because it searches for the largest alignment at each step. Let

$$\vec{y} = \begin{bmatrix} -100 \\ 100 \\ 100 \end{bmatrix},$$

which gives the energy $\vec{y}^T\vec{y} = 30000$.

The OLS procedure selects the following permutation of H:

$$HP_{OLS} = \begin{bmatrix} 0.135335 & 1 & 0.606531 \\ 0.606531 & 0.606531 & 1 \\ 1 & 0.135335 & 0.606531 \end{bmatrix},$$

which produces regressors with the energies:

$$g_1^2 = 15614.1,\ g_2^2 = 8019.76,\ \text{and}\ g_3^2 = 6366.17,$$

respectively, and

$$\frac{g_1^2 + g_2^2}{\vec{y}^T\vec{y}} \approx 0.79.$$

However, the following (identity) permutation of H:

$$HP_{opt} = \begin{bmatrix} 1 & 0.606531 & 0.135335 \\ 0.606531 & 1 & 0.606531 \\ 0.135335 & 0.606531 & 1 \end{bmatrix},$$

which produces regressors with the energies:

$$g_1^2 = 480.691,\ g_2^2 = 29305.3,\ \text{and}\ g_3^2 = 213.976,$$

respectively, and

$$\frac{g_1^2 + g_2^2}{\vec{y}^T\vec{y}} \approx 0.99,$$

is clearly superior in case 80% or more of the energy is needed. The OLS will require all three basis vectors, while the other permutation will need only two. Again, note that the optimality is error dependent. If we only needed a 52% accuracy, the one-vector subset chosen by OLS would be optimal.

4.4 Sorting Regressors does not Improve Efficiency

Since the energy function of the orthonormal regressors produced by OLS is not monotonic in general, a seemingly obvious next step would be to sort the regressors in the order of decreasing energy and delete the ones with the smallest g_i^2 contributions, without exceeding the allowed error. However, such a parsimonious

sorting of the regressors alters the specific permutation adhered to by the GS process.

The premise of OLS is that the chosen selection of the original RBF vectors can be recovered by reversing GS, a forward recursion procedure. Using (13) gives:

$$\vec{a_i} = r_{ii}\vec{q_i} + \sum_{l=1}^{i-1} r_{li}\vec{q_l}, \tag{24}$$

where r_{li} and $\vec{q_l}$ depend on the order in which the original basis vectors $\vec{h_j}$ were selected as $\vec{a_i}$. If the columns of $\hat{Q}$ and R are now sorted without deleting any $\vec{q_i}$, A can still be recovered. However, deleting any regressors, no matter how insignificant their g_i^2 may be, upsets the consistency of the GS process, thereby precluding the recovery of A and destroying the connection to the original RBF weights. In other words, the sorted regressors cannot be used to train the RBF approximation network.

Since every permutation of the original RBF matrix leads via GS to a different set of orthonormal regressors, their energies are meaningful only in the context of a particular permutation selected by OLS. Both the order of selection and the intermediate regressors themselves, even with low energies, are the key to a possibly high degree of alignment of a regressor selected at a later step of the procedure. On the other hand, deleting a GS regressor shrinks the number of basis vectors in the selection. And since the original RBF matrix is non-orthogonal, removing a regressor with a relatively low energy is not equivalent to disposing of an insignificant basis vector.

5 Performance of OLS in Image Coding

This section presents a new algorithm for selecting RBF centers, which is optimal in the sense of energy efficiency. This algorithm is used to evaluate the performance of the OLS method in a practical signal processing application, image coding. Specifically, the goal is to compare the size of an RBF network trained by OLS to the size of the smallest network achievable for a given error tolerance.

5.1 A New Optimal Selection Procedure

The OLS algorithm preserves the order of selection for a given data vector and attains higher accuracy by enlarging the selection matrix. In the limit of zero error, the selection matrix becomes a particular permutation matrix. It follows that the best selection of a non-orthogonal basis matrix H can be found by orthonormalizing all of its $N!$ permutations with GS and choosing the selection that accomplishes the specified approximation accuracy with the smallest number of regressors. Although the permutations can be processed concurrently, having to maintain the order of basis vectors, i.e., keeping $N!$ of them in storage, leads to an enormous amount of hardware, even for modest values of N.

Regardless of the order in which the basis vectors are chosen, every M-member combination of N original basis vectors has an error value associated with it. For instance, any one of $N!$ permutations of H can approximate the data with zero error. Therefore, M and the particular basis vectors comprising the subset must be chosen depending on both the data vector and the required error. This suggests dispensing with the order and performing LLSE on all $\sum_{M=1}^{M_{opt}} \binom{N}{M}$ possible combinations of the original basis vectors until the error tolerance is met at $M = M_{opt}$. This can also be implemented in parallel, requiring less hardware than the previous approach.

We combined this reasoning with OLS to arrive at the following procedure for determining the most efficient RBF network:

1. Run OLS to determine M_{OLS} – an upper bound on M.

2. For each of the $\binom{N}{M-1}$ possible combinations of the basis vectors of H, construct $\hat{H}$ and evaluate

$$\begin{aligned} \vec{e}^T\vec{e} &= \vec{y}^T\vec{y} - \vec{y}_a^T\vec{y}_a \\ &= \vec{y}^T\vec{y} - \vec{w}^{\star T}\hat{H}^T\hat{H}\vec{w}^\star \end{aligned}$$

   using (9) and (11) [4]. Record the combination that gives the lowest $\vec{e}^T\vec{e}$.

3. If $\vec{e}^T\vec{e}$ in Step 2 is less than or equal to the desired error allowance,
   set $M \leftarrow M - 1$ and go back to Step 2. Otherwise, $M = M_{opt}$, which cannot be further reduced without violating the allowed error specifications.

The following experiment examines energy efficiency only. Clearly, the computational efficiency of OLS will always be better than that of the new optimal method.

5.2 Experiment and Results

[4] The Singular Value Decomposition method [21] for computing the pseudoinverse matrices was used in the actual implementation.

Even one iteration of the exhaustive search in Step 2 is a tremendous computational task if $N > 10$. This is not a problem in image coding, because the processing is commonly performed on small blocks of data [22]. Two 256×256-pixel images (a face [5] and a texture), shown in Figure 2, were each segmented into 1024 square blocks of 8×8 pixels. To each block, we applied the Gaussian RBF transform (with variance $\sigma = 1$). Treating pixel locations as the coordinates of RBF centers and noting that the multivariate Gaussian RBFs are separable [6], assigns the network parameters to $d = 1$ and $N = 8$ (see Figure 1). The OLS and the optimal algorithms were applied to every block of an image, and the total counts of RBF nodes were recorded. Coding a 256×256-pixel image with the optimal algorithm was found to be 20 times slower than with OLS. In order to juxtapose the overall efficiency of the Gaussian RBF approximation scheme against the KLT, the image was also coded with the DCT in the same fashion. The DCT, (23), is an orthonormal sinusoidal transform. Because of these properties, it has a fast algorithm for its implementation, just like the Discrete Fourier Transform (DFT) has the Fast Fourier Transform (FFT) [20]. Therefore, DCT is faster than OLS.

The comparison of the Gaussian RBFs with the DCT basis functions is justified by the fact that the images are sampled on a dense regular grid, as opposed to using the typically sparse data. We are interested in approximating the image intensities at these grid positions with as few basis vectors as possible for a given value of the error. Note that, depending on the assumptions regarding the source of the data samples, approximating with either the Gaussian RBFs or the DCT basis may be understood as generalizing between the data points [23], [14]. However, this issue is tangential to that of efficiency, which is the main focus of the present experiment.

The results plotted in Figure 3 demonstrate the image coding efficiencies achieved by the OLS and the optimal methods of training an RBF network as a function of the signal-to-noise ratio (SNR), defined as

$$\text{SNR in dB} = 10 \log_{10} \left(\frac{\vec{y}^T \vec{y}}{\vec{e}^T \vec{e}} \right).$$

In addition, Figure 4 graphs the percentage oversize of the OLS-trained RBF network as a function of SNR.

All three methods exhibit the same poor efficiency at high values of SNR, because a high approximation accuracy requires a large number of RBF centers (or spatial frequencies in the case of the DCT). However, the relative efficiency of the OLS method drops rapidly as more error is allowed. For example, at SNR = 35dB, OLS requires 15% more RBF centers than the optimal technique.

For applications that use large training sets and require extremely high SNR, the computational efficiency of the OLS method warrants its use, despite its suboptimal energy compaction characteristics. On the other hand, for applications which allow lots of "encoding" time, the new selection procedure will give a more efficient set of RBF coefficients. Note that the optimal method can be helpful in comparing the compaction efficiencies of various, possibly non-orthogonal, bases. In fact, the plots reveal that the optimal method, applied to the Gaussian RBFs with $\sigma = 1$ achieves coding rates that are close to those attained by the DCT.

Having established the potential of Gaussian RBFs for efficient representation, other relevant properties of the basis become of interest. For example, Figure 5 offers the comparison of visual artifacts associated with reconstructing the image in Figure 2(a) using the DCT and the Gaussian bases. A typical in compression value of SNR (35dB) is chosen in order to emphasize the distinctions. The well-known "blocking effect" is prominent in the reconstructed image for the DCT case. This type of distortion is caused by differences in spatial frequencies required for reconstructing each block with the given SNR. Since the human eye is sensitive to spatial frequencies, it notices these discontinuities at the block boundaries.

However, in the case of the Gaussian RBFs, the parameter of the expansion is not the spatial frequency, but the actual coordinate of a center. Eliminating a center from the RBF set causes a different type of distortion than eliminating a wave number from the DCT expansion. While in the DCT expansion each spatial frequency affects all pixels of a block, in the Gaussian RBF expansion each center affects only a narrow neighborhood around the corresponding pixel. This locality property of the Gaussians leads to the "point defects" seen in the reconstructed image for the RBF case. The point defects occur when the pixel intensity at the deleted center is not adequately approximated by the preserved basis vectors.

An informal study of these perceptual defects was conducted by asking each of ten research colleagues to stand ten feet away from the 1152 pixel $\times$ 900 pixel display monitor and rank three 3 in $\times$ 3 in (or 256×256-pixel) versions of "Lena" based on the overall quality. The images were placed in the center of the screen with the original in the middle and the versions reconstructed at 35dB SNR using the Gaussian RBFs and the DCT

[5] Actually, a second face image ("Einstein") was coded as well, but because the results differed by only a few tenths of a percent, only one is presented here.

[6] In other words, rows and columns of an image block can be processed separately.

abutting its left and right hand sides, respectively. The image reconstructed with the Gaussian RBFs was unanimously found to look more like the original than the one reconstructed with the DCT basis. This stems from the fact that the point defects occur in random places within a block, while the blocking effect produces correlated patterns with periods commensurable to the block dimensions [24]. Figure 6 illustrates this point by displaying the error images corresponding to approximating the image in Figure 2(a) using the DCT and the Gaussian bases at 35dB SNR. When people stood closer than ten feet away from the screen, the preferences varied. This may be related to the fact that some of the subjects work in image coding, while others in vision, and bring in with them different biases.

This effect is even more pronounced when coding a textured image, such as "Treebark". For this texture, the results of the same experiment appear in Figure 7 and Figure 8. Since textures generally contain many spatial frequencies, coding with the DCT produces extremely correlated error images. On the other hand, the error images resulting from coding with the Gaussian RBFs look much more random.

6 Conclusions

While the OLS method has been believed to find a more efficient selection of RBF centers than a random-based approach [10], it does not produce the smallest RBF network for a given approximation accuracy. Simple examples were constructed to provide intuition about the sources of inefficiency.

To determine its performance in a practical signal processing application, OLS was used to train a Gaussian RBF network for image coding. In addition, a new optimal training method was described and also applied to image coding. The new method requires significantly more computation than OLS, but is optimal in the energy compaction sense, since it effectively searches all possible subsets of the basis. Simulations revealed that at low SNR, this optimal technique gives higher data compression than OLS.

Both algorithms performed poorly at high SNR, indicating that the efficiency depends not only on the training method, but also on the choice of the basis. The new optimal method allows a fair comparison among different bases and becomes a useful tool when non-orthogonal representations are considered. Thus, the Gaussian RBF network was evaluated with respect to the DCT expansion, a standard image coding technique. Relative efficiencies of various bases are always data-dependent. For the three test images studied, the efficiencies of the Gaussian RBF network and the DCT were found to be comparable, judged by the number of basis functions necessary for a desired SNR.

In addition to coding efficiency, the two bases were contrasted from the standpoint of visual perception, another criterion relevant in image coding. An informal study involving ten human observers found the "point defects", resulting from approximating the images with the Gaussians, to be perceptually less objectionable under certain conditions than the "blocking effect", caused by the DCT.

Acknowledgements

The authors would like to thank Federico Girosi and Terence Sanger for their comments on the manuscript.

(a) (b)

Figure 2: Test images: (a) "Lena"; (b) "Tree-bark".

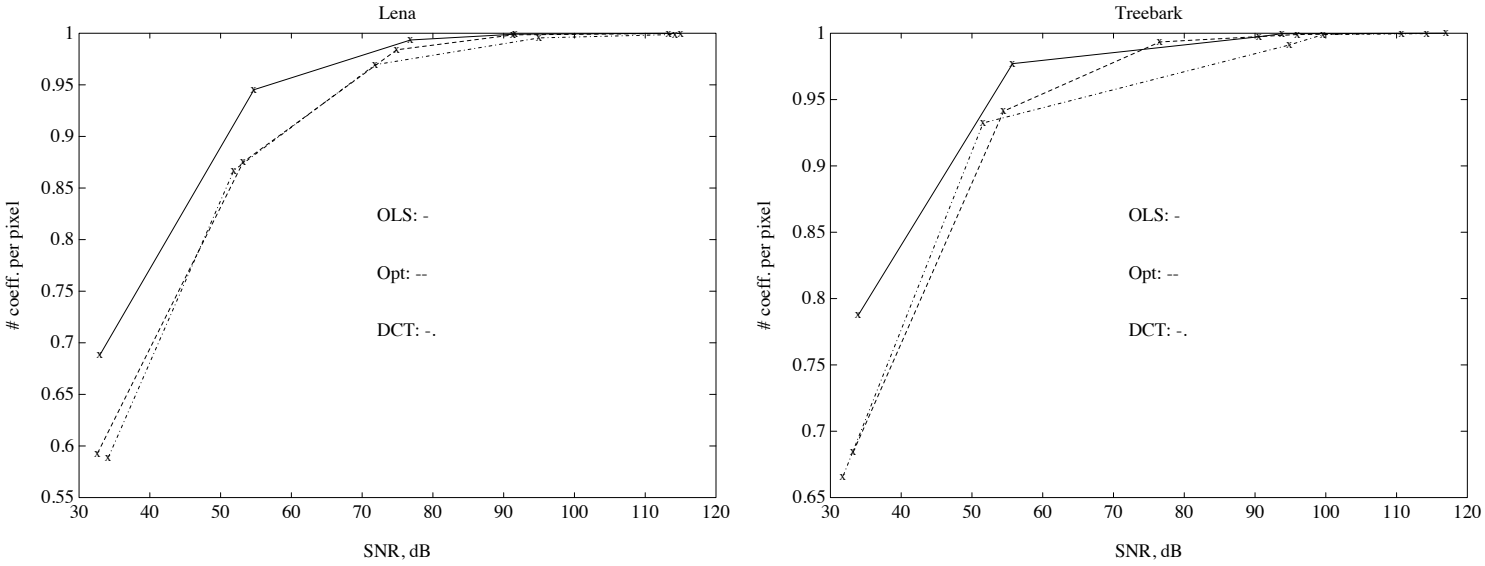


Figure 3: The number of RBF nodes per pixel selected by the OLS and the optimal training methods as a function of SNR. A similar plot for DCT is included for comparing the efficiency of the RBF and the KLT expansions.

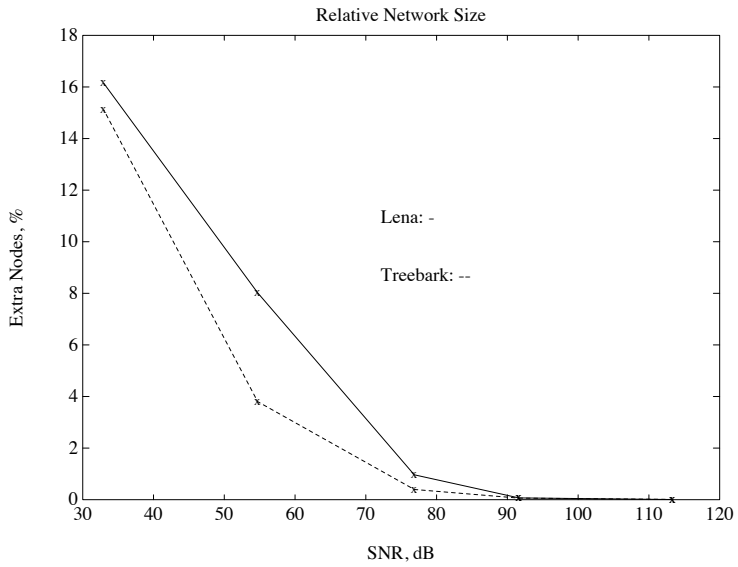
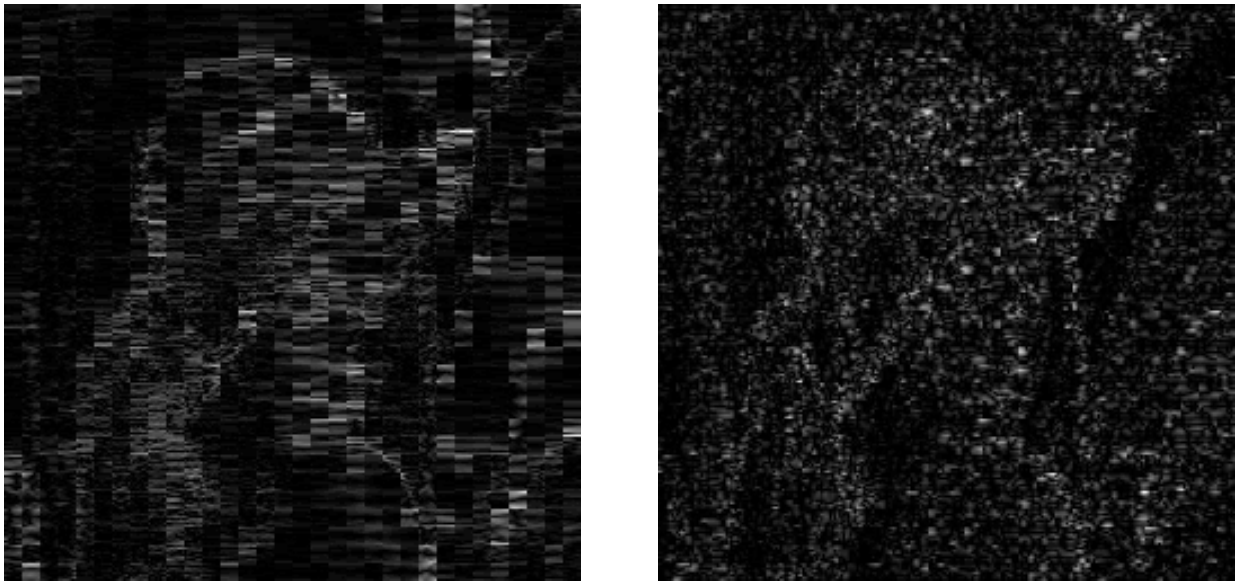


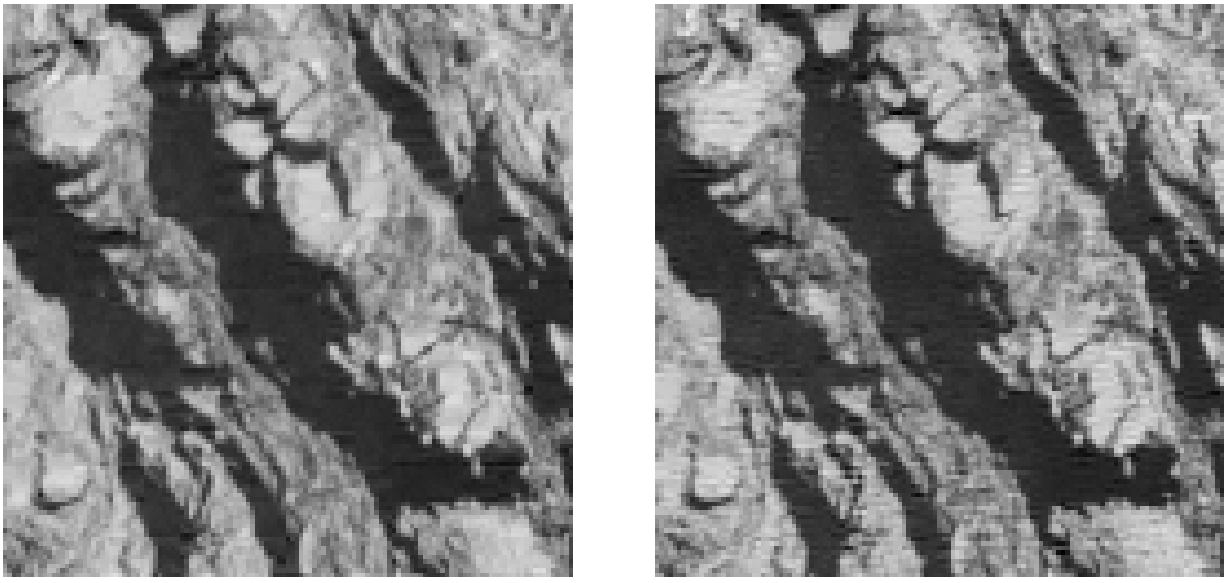
Figure 4: Percentage of extra nodes produced by the OLS training method as a function of SNR.

Figure 5: Reconstruction of the image in Figure 2(a) at 35dB SNR; the magnification is $\times 2$: (a) using the DCT basis; (b) using the Gaussian RBFs with the new optimal selection procedure.



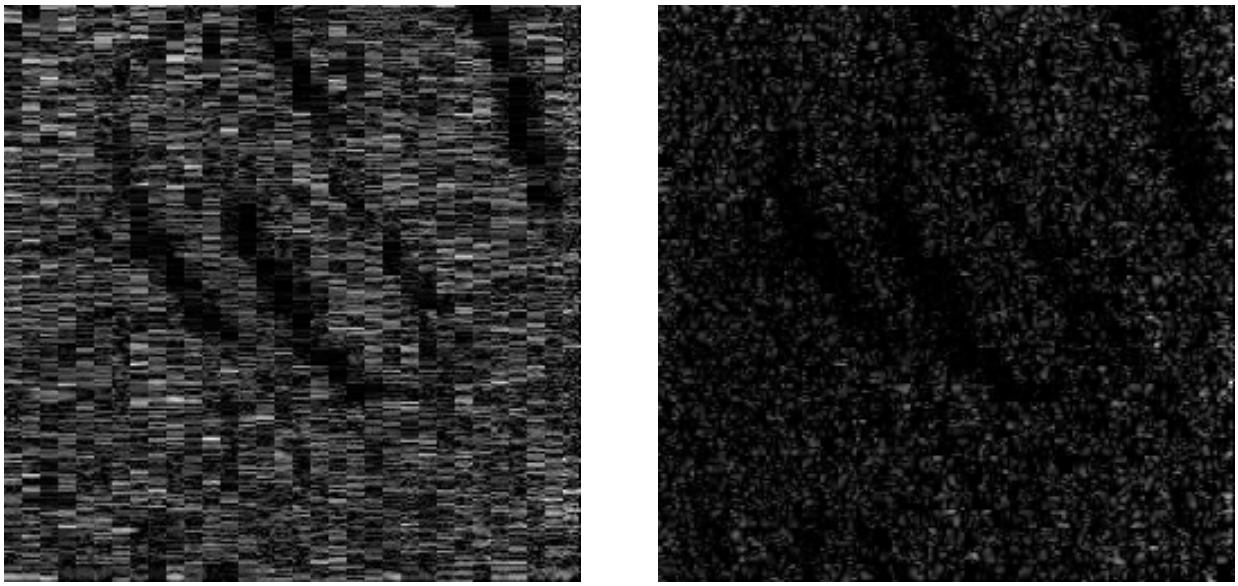
(a) (b)

Figure 6: Error images corresponding to approximating the image in Figure 2(a) at 35dB SNR: (a) using the DCT basis: the range of values was modified from $\{-44, 38\}$ to $\{0, 255\}$ for displaying purposes; (b) using the Gaussian RBFs with the new optimal selection procedure: the range of values was modified from $\{-30, 47\}$ to $\{0, 255\}$ for displaying purposes. Notice that (a) is more correlated than (b).



(a) (b)

Figure 7: Reconstruction of the image in Figure 2(b) at 35dB SNR; the magnification is $\times 2$: (a) using the DCT basis; (b) using the Gaussian RBFs with the new optimal selection procedure.



(a) (b)

Figure 8: Error images corresponding to approximating the image in Figure 2(b) at 35dB SNR: (a) using the DCT basis: the range of values was modified from $\{-43,\ 38\}$ to $\{0,\ 255\}$ for displaying purposes; (b) using the Gaussian RBFs with the new optimal selection procedure: the range of values was modified from $\{-66,\ 111\}$ to $\{0,\ 255\}$ for displaying purposes. Notice that (a) is more correlated than (b).

References

[1] G. Cybenko, "Approximations by superpositions of a sigmoidal function," *Math. Control Signals Systems*, vol. 2, no. 4, pp. 303–314, 1989.

[2] T. Poggio and F. Girosi, "Networks and the best approximation property," A.I. Memo #1164, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1989.

[3] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[4] S. Renals and R. Rohwer, "Phoneme classification experiments using radial basis functions," in *Proceedings Of IEEE International Joint Conference on Neural Networks*, (Washington DC), pp. I:461–467, June 1989.

[5] K. Ng, "A comparative study of the practical characteristics of neural network and conventional pattern classifiers," Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, 1990.

[6] R. P. Lippmann, "An introduction to computing with neural nets," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 4, pp. 4–22, Apr. 1987.

[7] R. P. Lippmann, "Pattern classification using neural networks," *IEEE Transactions on Communications*, vol. 27, no. 11, pp. 47–64, 1989.

[8] D. S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks," *Complex Systems*, vol. 2, pp. 321–355, 1988.

[9] J. Moody and C. Darken, "Fast-learning in networks of locally-tuned processing units," *Neural Computation*, vol. 1, no. 2, pp. 281–294, 1989.

[10] S. Chen, C. F. N. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Transactions on Neural Networks*, vol. 2, pp. 302–309, Mar. 1991.

[11] M. J. D. Powell, "Radial basis functions for multivariable interpolation: A review," in *Algorithms for Approximation* (J. C. Mason and M. G. Cox, eds.), (Oxford), Clarendon Press, 1987.

[12] T. Poggio and F. Girosi, "A theory of networks for approximation and learning," A.I. Memo #1140, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1989.

[13] T. Poggio and F. Girosi, "Extension of a theory of networks for approximation and learning: Dimensionality reduction and clustering," A.I. Memo #1167, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1990.

[14] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proceedings of the IEEE*, vol. 78, pp. 1481–1497, Sept. 1990.

[15] G. Strang, *Linear Algebra and Its Applications*. Academic Press, 1980.

[16] G. Strang, *Introduction to Applied Mathematics*. Cambridge, MA: Wellesley-Cambridge Press, 1986.

[17] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, 1965.

[18] R. J. Clarke, *Transform Coding of Images*. Orlando: Academic Press, Inc., 1985.

[19] T. D. Sanger, "Optimal unsupervised learning in a single-layer linear feedforward neural network," *Neural Networks*, vol. 2, pp. 459–473, 1989.

[20] A. K. Jain, "A sinusoidal family of unitary transforms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-1, no. 4, pp. 356–365, 1979.

[21] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling, *Numerical Recipes in C*. New York, NY: Cambridge University Press, 1988.

[22] J. S. Lim, *Two-Dimensional Signal and Image Processing.* New Jersey: Prentice-Hall, 1990.

[23] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing.* London: Prentice-Hall, Inc., 1975.

[24] R. A. Ulichney, *Digital Halftoning.* Cambridge, MA: MIT Press, 1987.