NEURAL COMPUTING
NCRG
RESEARCH GROUP

Neural Computing Research Group
Dept of Computer Science & Applied Mathematics
Aston University
Birmingham B4 7ET
United Kingdom
Tel: +44 (0)121 333 4631
Fax: +44 (0)121 333 4586
http://www.ncrg.aston.ac.uk/

GTM: The Generative Topographic Mapping

Christopher M. Bishop
C.M.Bishop@aston.ac.uk

Markus Svensén
svensjfm@aston.ac.uk

Christopher K. I. Williams
C.K.I.Williams@aston.ac.uk

Technical Report NCRG/96/015 April 16, 1997

Accepted for publication in *Neural Computation*.

Abstract

Latent variable models represent the probability density of data in a space of several dimensions in terms of a smaller number of latent, or hidden, variables. A familiar example is factor analysis which is based on a linear transformations between the latent space and the data space. In this paper we introduce a form of non-linear latent variable model called the *Generative Topographic Mapping* for which the parameters of the model can be determined using the EM algorithm. GTM provides a principled alternative to the widely used Self-Organizing Map (SOM) of Kohonen (1982), and overcomes most of the significant limitations of the SOM. We demonstrate the performance of the GTM algorithm on a toy problem and on simulated data from flow diagnostics for a multi-phase oil pipeline.

1 Introduction

Many data sets exhibit significant correlations between the variables. One way to capture such structure is to model the distribution of the data in terms of latent, or hidden, variables. A familiar example of this approach is factor analysis, which is based on a linear transformation from latent space to data space. In this paper we show how the latent variable framework can be extended to allow non-linear transformations while remaining computationally tractable. This leads to the GTM (*Generative Topographic Mapping*) algorithm, which is based on a constrained mixture of Gaussians whose parameters can be optimized using the EM (expectation-maximization) algorithm.

One of the motivations for this work is to provide a principled alternative to the widely used 'self-organizing map' (SOM) algorithm (Kohonen 1982) in which a set of unlabelled data vectors $\mathbf{t}_n$ $(n = 1, \ldots, N)$ in a D-dimensional data space is summarized in terms of a set of reference vectors having a spatial organization corresponding to a (generally) two-dimensional sheet. While this algorithm has achieved many successes in practical applications, it also suffers from some significant deficiencies, many of which are highlighted in Kohonen (1995). They include: the absence of a cost function, the lack of a theoretical basis for choosing learning rate parameter schedules and neighbourhood parameters to ensure topographic ordering, the absence of any general proofs of convergence, and the fact that the model does not define a probability density. These problems can all be traced to the heuristic origins of the SOM algorithm[1]. We show that the GTM algorithm overcomes most of the limitations of the SOM while introducing no significant disadvantages.

An important application of latent variable models is to data visualization. Many of the models used in visualization are regarded as defining a projection from the D-dimensional data space onto a two-dimensional visualization space. We shall see that, by contrast, the GTM model is defined in terms of a mapping *from* the latent space *into* the data space. For the purposes of data visualization, the mapping is then inverted using Bayes' theorem, giving rise to a posterior distribution in latent space.

2 Latent Variables

The goal of a latent variable model is to find a representation for the distribution $p(\mathbf{t})$ of data in a D-dimensional space $\mathbf{t} = (t_1, \ldots, t_D)$ in terms of a number L of latent variables $\mathbf{x} = (x_1, \ldots, x_L)$. This is achieved by first considering a function $\mathbf{y}(\mathbf{x}; \mathbf{W})$ which maps points $\mathbf{x}$ in the latent space into corresponding points $\mathbf{y}(\mathbf{x}; \mathbf{W})$ in the data space. The mapping is governed by a matrix of parameters $\mathbf{W}$, and could consist, for example, of a feed-forward neural network in which case $\mathbf{W}$ would represent the weights and biases. We are interested in the situation in which the dimensionality L of the latent-variable space is less than the dimensionality D of the data space, since we wish to capture the fact that the data itself has an intrinsic dimensionality which is less than D. The transformation $\mathbf{y}(\mathbf{x}; \mathbf{W})$ then maps the latent-variable space into an L-dimensional non-Euclidean manifold $\mathcal{S}$ embedded within the data space[2]. This is illustrated schematically for the case of $L = 2$ and $D = 3$ in Figure 1.

If we define a probability distribution $p(\mathbf{x})$ on the latent-variable space, this will induce a corresponding distribution $p(\mathbf{y}|\mathbf{W})$ in the data space. We shall refer to $p(\mathbf{x})$ as the prior distribution of $\mathbf{x}$ for reasons which will become clear shortly. Since $L < D$, the distribution in $\mathbf{t}$-space would be

[1] Biological metaphor is sometimes invoked when motivating the SOM procedure. It should be stressed that our goal here is not neuro-biological modelling, but rather the development of effective algorithms for data analysis, for which biological realism need not be considered.

[2] We assume that the matrix of partial derivatives $\partial y_k / \partial x_i$ has full column rank.

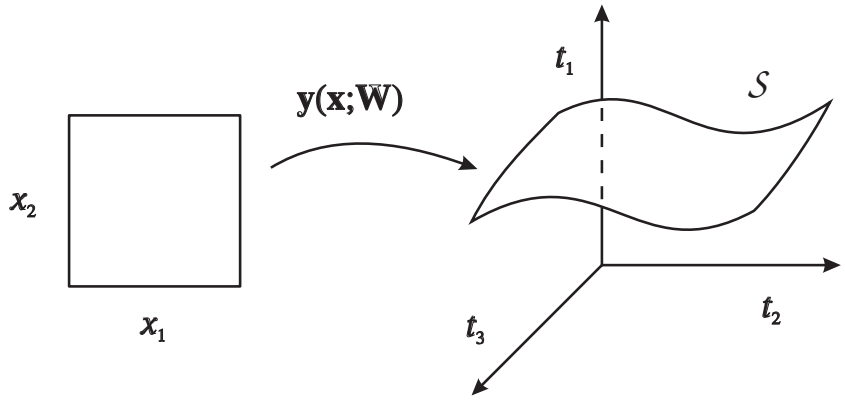

Figure 1: The non-linear function $\mathbf{y}(\mathbf{x};\mathbf{W})$ defines a manifold $\mathcal{S}$ embedded in data space given by the image of the latent-variable space under the mapping $\mathbf{x} \to \mathbf{y}$.

confined to the L-dimensional manifold and hence would be singular. Since in reality the data will only approximately live on a lower-dimensional manifold, it is appropriate to include a noise model for the $\mathbf{t}$ vector. We choose the distribution of $\mathbf{t}$, for given $\mathbf{x}$ and $\mathbf{W}$, to be a radially-symmetric Gaussian centred on $\mathbf{y}(\mathbf{x};\mathbf{W})$ having variance β^{-1} so that

$$p(\mathbf{t}|\mathbf{x},\mathbf{W},\beta) = \left(\frac{\beta}{2\pi}\right)^{D/2} \exp\left\{-\frac{\beta}{2}\|\mathbf{y}(\mathbf{x};\mathbf{W}) - \mathbf{t}\|^2\right\}. \tag{1}$$

Note that other models for $p(\mathbf{t}|\mathbf{x})$ might also be appropriate, such as a Bernoulli for binary variables (with a sigmoid transformation of $\mathbf{y}$) or a multinomial for mutually exclusive classes (with a 'softmax', or normalized exponential transformation of $\mathbf{y}$ (Bishop 1995)), or even combinations of these. The distribution in $\mathbf{t}$-space, for a given value of $\mathbf{W}$, is then obtained by integration over the $\mathbf{x}$-distribution

$$p(\mathbf{t}|\mathbf{W},\beta) = \int p(\mathbf{t}|\mathbf{x},\mathbf{W},\beta)p(\mathbf{x})\,d\mathbf{x}. \tag{2}$$

For a given a data set $\mathcal{D} = (\mathbf{t}_1,\ldots,\mathbf{t}_N)$ of N data points, we can determine the parameter matrix $\mathbf{W}$, and the inverse variance β, using maximum likelihood. In practice it is convenient to maximize the log likelihood, given by

$$\mathcal{L}(\mathbf{W},\beta) = \ln \prod_{n=1}^{N} p(\mathbf{t}_n|\mathbf{W},\beta). \tag{3}$$

Once we have specified the prior distribution $p(\mathbf{x})$ and the functional form of the mapping $\mathbf{y}(\mathbf{x};\mathbf{W})$, we can in principle determine $\mathbf{W}$ and β by maximizing $\mathcal{L}(\mathbf{W},\beta)$. However, the integral over $\mathbf{x}$ in (2) will, in general, be analytically intractable. If we choose $\mathbf{y}(\mathbf{x};\mathbf{W})$ to be a linear function of $\mathbf{W}$, and we choose $p(\mathbf{x})$ to be Gaussian, then the integral becomes a convolution of two Gaussians which is itself a Gaussian. For a noise distribution $p(\mathbf{t}|\mathbf{x})$ which is Gaussian with a diagonal covariance matrix, we obtain the standard factor analysis model. In the case of the radially symmetric Gaussian given by (1) the model is closely related to principal component analysis since the maximum likelihood solution for $\mathbf{W}$ has columns given by the scaled principal eigenvectors. Here we wish to extend this formalism to non-linear functions $\mathbf{y}(\mathbf{x};\mathbf{W})$, and in particular to develop a model which is similar in spirit to the SOM algorithm. We therefore consider a specific form for $p(\mathbf{x})$ given by a sum of delta functions centred on the nodes of a regular grid in latent space

$$p(\mathbf{x}) = \frac{1}{K}\sum_{i=1}^{K} \delta(\mathbf{x} - \mathbf{x}_i) \tag{4}$$

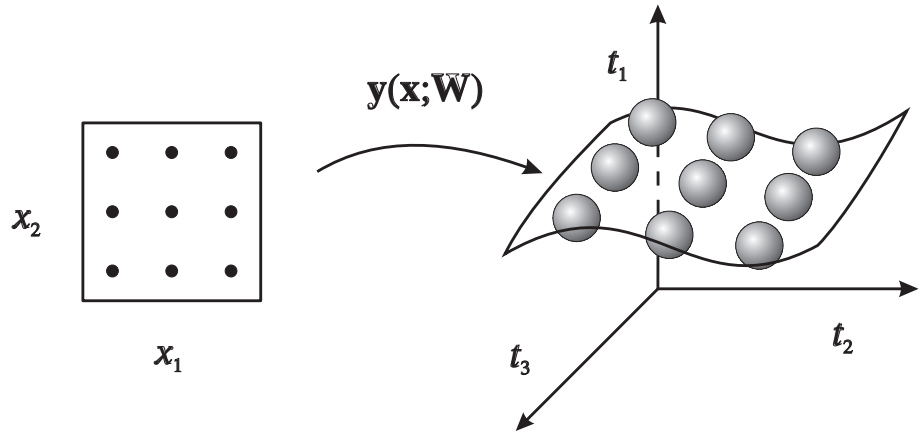

Figure 2: In order to formulate a latent variable model which is similar in spirit to the SOM, we consider a prior distribution $p(\mathbf{x})$ consisting of a superposition of delta functions, located at the nodes of a regular grid in latent space. Each node $\mathbf{x}_i$ is mapped to a corresponding point $\mathbf{y}(\mathbf{x}_i; \mathbf{W})$ in data space, and forms the centre of a corresponding Gaussian distribution.

in which case the integral in (2) can again be performed analytically. Each point $\mathbf{x}_i$ is then mapped to a corresponding point $\mathbf{y}(\mathbf{x}_i; \mathbf{W})$ in data space, which forms the centre of a Gaussian density function, as illustrated in Figure 2. From (2) and (4) we see that the distribution function in data space then takes the form

$$p(\mathbf{t}|\mathbf{W}, \beta) = \frac{1}{K} \sum_{i=1}^{K} p(\mathbf{t}|\mathbf{x}_i, \mathbf{W}, \beta) \tag{5}$$

and the log likelihood function becomes

$$\mathcal{L}(\mathbf{W}, \beta) = \sum_{n=1}^{N} \ln \left\{ \frac{1}{K} \sum_{i=1}^{K} p(\mathbf{t}_n|\mathbf{x}_i, \mathbf{W}, \beta) \right\}. \tag{6}$$

For the particular noise model $p(\mathbf{t}|\mathbf{x}, \mathbf{W}, \beta)$ given by (1), the distribution $p(\mathbf{t}|\mathbf{W}, \beta)$ corresponds to a *constrained* Gaussian mixture model (Hinton, Williams, and Revow 1992) since the centres of the Gaussians, given by $\mathbf{y}(\mathbf{x}_i; \mathbf{W})$, cannot move independently but are related through the function $\mathbf{y}(\mathbf{x}; \mathbf{W})$. Note that, provided the mapping function $\mathbf{y}(\mathbf{x}; \mathbf{W})$ is smooth and continuous, the projected points $\mathbf{y}(\mathbf{x}_i; \mathbf{W})$ will necessarily have a topographic ordering in the sense that any two points $\mathbf{x}_A$ and $\mathbf{x}_B$ which are close in latent space will map to points $\mathbf{y}(\mathbf{x}_A; \mathbf{W})$ and $\mathbf{y}(\mathbf{x}_B; \mathbf{W})$ which are close in data space.

2.1 The EM Algorithm

If we now choose a particular parametrized form for $\mathbf{y}(\mathbf{x}; \mathbf{W})$ which is a differentiable function of $\mathbf{W}$ (for example, a feed-forward network with sigmoidal hidden units) then we can use standard techniques for non-linear optimization, such as conjugate gradients or quasi-Newton methods, to find a weight matrix $\mathbf{W}^*$, and an inverse variance β^*, which maximize $L(\mathbf{W}, \beta)$.

However, our model consists of a mixture distribution which suggests that we might seek an EM (expectation-maximization) algorithm (Dempster, Laird, and Rubin 1977; Bishop 1995). By making a suitable choice of model $\mathbf{y}(\mathbf{x}; \mathbf{W})$ we will see that the M-step corresponds to the solution of a set of linear equations. In particular we shall choose $\mathbf{y}(\mathbf{x}; \mathbf{W})$ to be given by a generalized

linear regression model of the form

$$\mathbf{y}(\mathbf{x}; \mathbf{W}) = \mathbf{W}\boldsymbol{\phi}(\mathbf{x}) \tag{7}$$

where the elements of $\boldsymbol{\phi}(\mathbf{x})$ consist of M fixed basis functions $\phi_j(\mathbf{x})$, and $\mathbf{W}$ is a $D \times M$ matrix. Generalized linear regression models possess the same universal approximation capabilities as multi-layer adaptive networks, provided the basis functions $\phi_j(\mathbf{x})$ are chosen appropriately. The usual limitation of such models, however, is that the number of basis functions must typically grow exponentially with the dimensionality L of the input space (Bishop 1995). In the present context this is not a significant problem since the dimensionality is governed by the number of latent variable variables which will typically be small. In fact for data visualization applications we generally use $L = 2$.

The maximization of (6) can be regarded as a missing-data problem in which the identity i of the component which generated each data point $\mathbf{t}_n$ is unknown. We can formulate the EM algorithm for this model as follows. First, suppose that, at some point in the algorithm, the current weight matrix is given by $\mathbf{W}_{\text{old}}$ and the current inverse noise variance is given by β_{old}. In the E-step we use $\mathbf{W}_{\text{old}}$ and β_{old} to evaluate the posterior probabilities, or responsibilities, of each Gaussian component i for every data point $\mathbf{t}_n$ using Bayes' theorem in the form

$$R_{in}(\mathbf{W}_{\text{old}}, \beta_{\text{old}}) = p(\mathbf{x}_i|\mathbf{t}_n, \mathbf{W}_{\text{old}}, \beta_{\text{old}}) \tag{8}$$

$$= \frac{p(\mathbf{t}_n|\mathbf{x}_i, \mathbf{W}_{\text{old}}, \beta_{\text{old}})}{\sum_{i'=1}^{K} p(\mathbf{t}_n|\mathbf{x}_{i'}, \mathbf{W}_{\text{old}}, \beta_{\text{old}})}. \tag{9}$$

We now consider the expectation of the complete-data log likelihood in the form

$$\langle \mathcal{L}_{\text{comp}}(\mathbf{W}, \beta) \rangle = \sum_{n=1}^{N} \sum_{i=1}^{K} R_{in}(\mathbf{W}_{\text{old}}, \beta_{\text{old}}) \ln \{p(\mathbf{t}_n|\mathbf{x}_i, \mathbf{W}, \beta)\}. \tag{10}$$

Maximizing (10) with respect to $\mathbf{W}$, and using (1) and (7), we obtain

$$\sum_{n=1}^{N} \sum_{i=1}^{K} R_{in}(\mathbf{W}_{\text{old}}, \beta_{\text{old}}) \{\mathbf{W}_{\text{new}}\boldsymbol{\phi}(\mathbf{x}_i) - \mathbf{t}_n\} \boldsymbol{\phi}^{\text{T}}(\mathbf{x}_i) = 0. \tag{11}$$

This can conveniently be written in matrix notation in the form

$$\boldsymbol{\Phi}^{\text{T}}\mathbf{G}_{\text{old}}\boldsymbol{\Phi}\mathbf{W}_{\text{new}}^{\text{T}} = \boldsymbol{\Phi}^{\text{T}}\mathbf{R}_{\text{old}}\mathbf{T} \tag{12}$$

where $\boldsymbol{\Phi}$ is a $K \times M$ matrix with elements $\Phi_{ij} = \phi_j(\mathbf{x}_i)$, $\mathbf{T}$ is a $N \times D$ matrix with elements t_{nk}, $\mathbf{R}$ is a $K \times N$ matrix with elements R_{in}, and $\mathbf{G}$ is a $K \times K$ diagonal matrix with elements

$$G_{ii} = \sum_{n=1}^{N} R_{in}(\mathbf{W}, \beta). \tag{13}$$

We can now solve (12) for $\mathbf{W}_{\text{new}}$ using standard matrix inversion techniques, based on singular value decomposition to allow for possible ill-conditioning. Note that the matrix $\boldsymbol{\Phi}$ is constant throughout the algorithm, and so need only be evaluated once at the start.

Similarly, maximizing (10) with respect to β we obtain the following re-estimation formula

$$\frac{1}{\beta_{\text{new}}} = \frac{1}{ND} \sum_{n=1}^{N} \sum_{i=1}^{K} R_{in}(\mathbf{W}_{\text{old}}, \beta_{\text{old}}) \|\mathbf{W}_{\text{new}}\boldsymbol{\phi}(\mathbf{x}_i) - \mathbf{t}_n\|^2. \tag{14}$$

The EM algorithm alternates between the E-step, corresponding to the evaluation of the posterior probabilities in (9), and the M-step, given by the solution of (12) and (14). Jensen's inequality

can be used to show that, at each iteration of the algorithm, the objective function will increase unless it is already at a (local) maximum, as discussed for example in Bishop (1995). Typically the EM algorithm gives satisfactory convergence after a few tens of cycles, particularly since we are primarily interested in convergence of the distribution and this is often achieved much more rapidly than convergence of the parameters themselves.

If desired, a regularization term can be added to the objective function to control the mapping $\mathbf{y}(\mathbf{x}; \mathbf{W})$. This can be interpreted as a MAP (maximum a-posteriori) estimator corresponding to a choice of prior over the weights $\mathbf{W}$. In the case of a radially-symmetric Gaussian prior of the form

$$p(\mathbf{W}|\lambda) = \left(\frac{\lambda}{2\pi}\right)^{MD/2} \exp\left\{-\frac{\lambda}{2}\sum_{j=1}^{M}\sum_{k=1}^{D} w_{jk}^2\right\} \tag{15}$$

where λ is the regularization coefficient, this leads to a modification of the M-step (12) to give

$$(\mathbf{\Phi}^{\mathrm{T}}\mathbf{G}_{\mathrm{old}}\mathbf{\Phi} + \lambda\mathbf{I})\mathbf{W}_{\mathrm{new}}^{\mathrm{T}} = \mathbf{\Phi}^{\mathrm{T}}\mathbf{R}_{\mathrm{old}}\mathbf{T} \tag{16}$$

where $\mathbf{I}$ is the identity matrix.

2.2 Data Visualization

One application for GTM is in data visualization, in which Bayes' theorem is used to invert the transformation from latent space to data space. For the particular choice of prior distribution given by (4), the posterior distribution is again a sum of delta functions centred at the lattice points, with coefficients given by the responsibilities R_{in}. These coefficients can be used to provide a visualization of the posterior responsibility map for individual data points in the two-dimensional latent space. If it is desired to visualize a *set* of data points then a complete posterior distribution for each data point may provide too much information and it is often convenient to summarize the posterior by its mean, given for each data point $\mathbf{t}_n$ by

$$\langle \mathbf{x}|\mathbf{t}_n, \mathbf{W}^*, \beta^* \rangle = \int p(\mathbf{x}|\mathbf{t}_n, \mathbf{W}^*, \beta^*)\mathbf{x}\, d\mathbf{x} \tag{17}$$

$$= \sum_{i=1}^{K} R_{in}\mathbf{x}_i. \tag{18}$$

It should be borne in mind, however, that the posterior distribution can be multi-modal in which case the posterior mean can give a very misleading summary of the true distribution. An alternative approach is therefore to evaluate the mode of the distribution, given by

$$i^{\max} = \arg\max_{\{i\}} R_{in}. \tag{19}$$

In practice it is often convenient to plot both the mean and the mode for each data point, as significant differences between them can be indicative of a multi-modal distribution.

2.3 Choice of Model Parameters

The problem of density estimation from a finite data set is fundamentally ill-posed, since there exist infinitely many distributions which could have given rise to the observed data. An algorithm for density modelling therefore requires some form of 'prior knowledge' in addition to the data set. The assumption that the distribution can be described in terms of a reduced number of latent variables

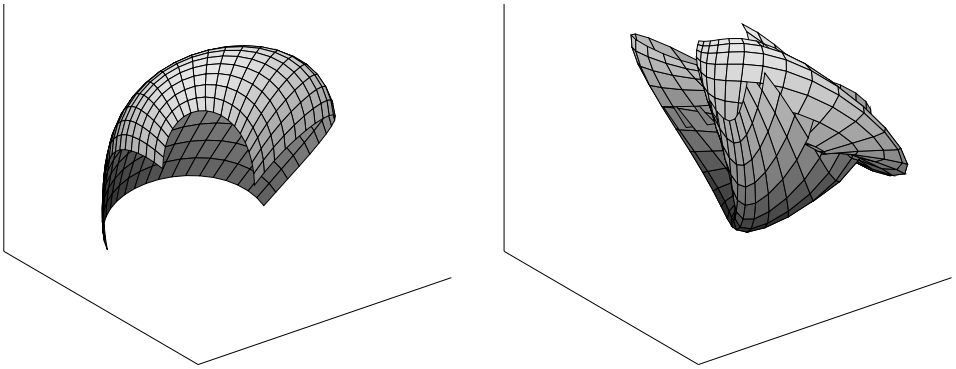

Figure 3: Examples of manifolds generated by sampling from the prior distribution over $\mathbf{W}$ given by (15), showing the effect of the choice of basis functions on the smoothness of the manifold. Here the basis functions are Gaussian with width $\sigma = 4s$ in the left-hand plot (where s is the spacing of the basis function centres), and $\sigma = 2s$ in the right-hand plot. Different values of λ simply affect the linear scaling of the embedded manifold.

is itself part of this prior. In the GTM algorithm, the prior distribution over mapping functions $\mathbf{y}(\mathbf{x}; \mathbf{W})$ is governed by the prior over weights $\mathbf{W}$, given for example by (15), as well as by the basis functions. We typically choose the basis functions $\phi_j(\mathbf{x})$ to be radially symmetric Gaussians whose centres are distributed on a uniform grid in $\mathbf{x}$-space, with a common width parameter σ, whose value, along with the number and spacing of the basis functions, determines the smoothness of the manifold. Examples of surfaces generated by sampling the prior are shown in Figure 3.

In addition to the basis functions $\phi_i(\mathbf{x})$, it is also necessary to select the latent-space sample points $\{\mathbf{x}_i\}$. Note that, if there are too few sample points in relation to the number of basis functions, then the Gaussian mixture centres in data space become relatively independent and the desired smoothness properties can be lost. Having a large number of sample points, however, causes no difficulty beyond increased computational cost. In particular, there is no 'over-fitting' if the number of sample points is increased since the number of degrees of freedom in the model is controlled by the mapping function $\mathbf{y}(\mathbf{x}; \mathbf{W})$. One way to view the role of the latent space samples $\{\mathbf{x}_i\}$ is as a Monte Carlo approximation to the integral over $\mathbf{x}$ in (2) (MacKay 1995; Bishop, Svensén, and Williams 1996a). The choice of the number K and location of the sample points $\mathbf{x}_i$ in latent space is not critical, and we typically choose Gaussian basis functions and set K so that, in the case of a two-dimensional latent space, $O(100)$ sample points lie within 2σ of the centre of each basis function.

Note that we have considered the basis function parameters (widths and locations) to be fixed, with a Gaussian prior on the weight matrix $\mathbf{W}$. In principle, priors over the basis function parameters could also be introduced, and these could again be treated by MAP estimation or by Bayesian integration.

We initialize the parameters $\mathbf{W}$ so that the GTM model initially approximates principal component analysis. To do this, we first evaluate the data covariance matrix and obtain the first and second principal eigenvectors, and then we determine $\mathbf{W}$ by minimizing the error function

$$E = \frac{1}{2} \sum_i \|\mathbf{W}\boldsymbol{\phi}(\mathbf{x}_i) - \mathbf{U}\mathbf{x}_i\| \tag{20}$$

where the columns of $\mathbf{U}$ are given by the eigenvectors. This represents the sum-of-squares error

between the projections of the latent points into data space by the GTM model and the corresponding projections obtained from PCA. The value of β^{-1} is initialized to be the larger of either the $L+1$ eigenvalue from PCA (representing the variance of the data away from the PCA plane) or the square of half of the grid spacing of the PCA-projected latent points in data space.

Finally, we note that in a numerical implementation care must be taken over the evaluation of the responsibilities since this involves computing the exponentials of the distances between the projected latent points and the data points, which may span a significant range of values.

2.4 Summary of the GTM Algorithm

Although the foregoing discussion has been somewhat detailed, the underlying GTM algorithm itself is straightforward and is summarized here for convenience.

GTM consists of a constrained mixture of Gaussians in which the model parameters are determined by maximum likelihood using the EM algorithm. It is defined by specifying a set of points $\{\mathbf{x}_i\}$ in latent space, together with a set of basis functions $\{\phi_j(\mathbf{x})\}$. The adaptive parameters $\mathbf{W}$ and β define a constrained mixture of Gaussians with centres $\mathbf{W}\boldsymbol{\phi}(\mathbf{x}_i)$ and a common covariance matrix given by $\beta^{-1}\mathbf{I}$. After initializing $\mathbf{W}$ and β, training involves alternating between the E-step in which the posterior probabilities are evaluated using (9), and the M-step in which $\mathbf{W}$ and β are re-estimated using (12) and (14) respectively. Evaluation of the log likelihood using (6) at the end of each cycle can be used to monitor convergence.

3 Experimental Results

We now present results from the application of this algorithm first to a toy problem involving data in two dimensions, and then to a more realistic problem involving 12-dimensional data arising from diagnostic measurements of oil flows along multi-phase pipelines. In both examples we choose the basis functions $\phi_j(\mathbf{x})$ to be radially symmetric Gaussians whose centres are distributed on a uniform grid in $\mathbf{x}$-space, with a common width parameter chosen equal to twice the separation of neighbouring basis function centres. Results from a toy problem for the case of a 2-dimensional data space and a 1-dimensional latent space are shown in Figure 4.

3.1 Oil Flow Data

Our second example arises from the problem of determining the fraction of oil in a multi-phase pipeline carrying a mixture of oil, water and gas (Bishop and James 1993). Each data point consists of 12 measurements taken from dual-energy gamma densitometers measuring the attenuation of gamma beams passing through the pipe. Synthetically generated data is used which models accurately the attenuation processes in the pipe, as well as the presence of noise (arising from photon statistics). The three phases in the pipe (oil, water and gas) can belong to one of three different geometrical configurations, corresponding to laminar, homogeneous, and annular flows, and the data set consists of 1000 points drawn with equal probability from the 3 configurations. We take the latent-variable space to be two-dimensional, since our goal is data visualization.

Figure 5 shows the oil data visualized in the latent-variable space in which, for each data point, we have plotted the posterior mean vector. Each point has then been labelled according to its

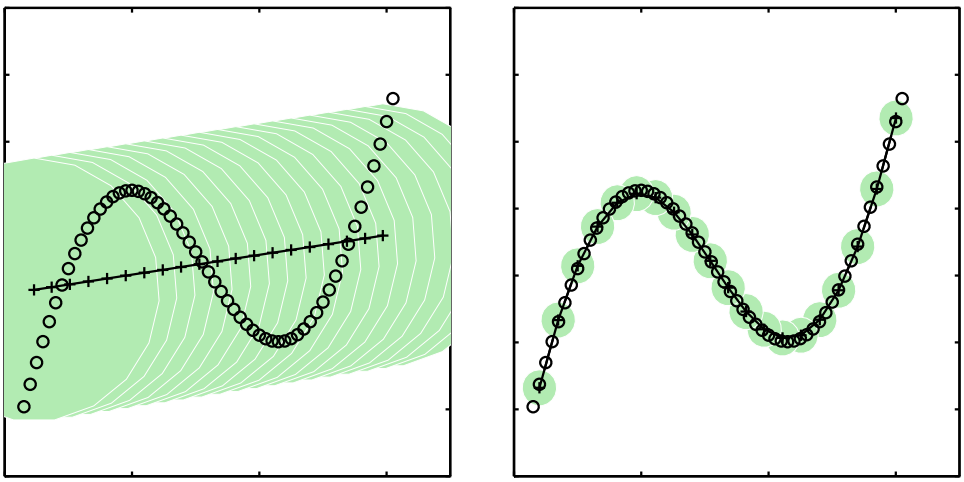

Figure 4: Results from a toy problem involving data ('○') generated from a 1-dimensional curve embedded in 2 dimensions, together with the projected latent points ('+') and their Gaussian noise distributions (filled circles). The initial configuration, determined by principal component analysis, is shown on the left, and the converged configuration, obtained after 15 iterations of EM, is shown on the right.

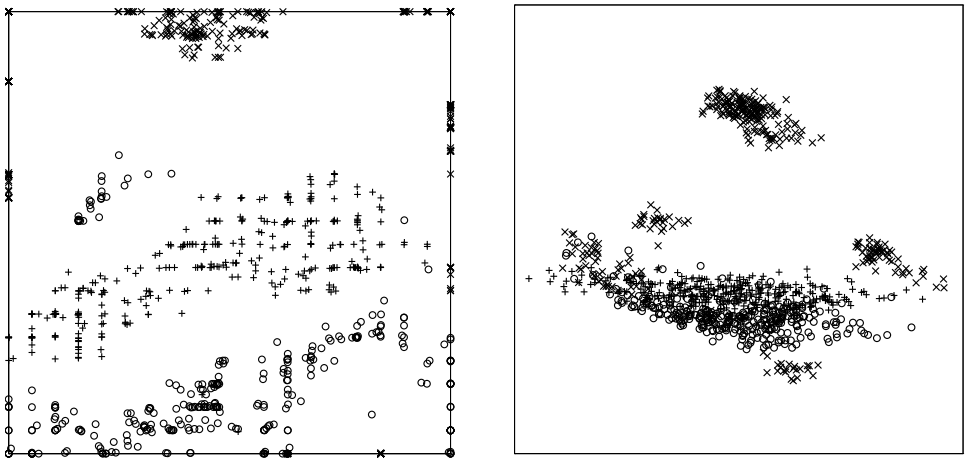

Figure 5: The left plot shows the posterior-mean projection of the oil flow data in the latent space of the GTM model, while the plot on the right shows the same data set visualized using principal component analysis. In both plots, crosses, circles and plus-signs represent stratified, annular and homogeneous multi-phase configurations respectively. Note how the non-linearity of GTM gives an improved separation of the clusters.

multi-phase configuration. For comparison, Figure 5 also shows the corresponding results obtained using principal component analysis.

4 Relation to the Self-Organizing Map

Since one motivation for GTM is to provide a principled alternative to the self-organizing map, it is useful to consider the precise relationship between GTM and SOM. We focus our attention on the batch versions of both algorithms as this helps to make the relationship particularly clear.

The batch version of the SOM algorithm (Kohonen 1995) can be described as follows. A set of K reference vectors $\mathbf{z}_i$ is defined in the data space, in which each vector is associated with a node on a regular lattice in a (typically) two-dimensional 'feature map' (analogous to the latent space of GTM). The algorithm begins by initializing the reference vectors (for example by setting them to random values, by setting them equal to a random subset of the data points, or by using principal component analysis). Each cycle of the algorithm then proceeds as follows. For every data vector $\mathbf{t}_n$ the corresponding 'winning node' $j(n)$ is identified, corresponding to the reference vector $\mathbf{z}_j$ having the smallest Euclidean distance $\|\mathbf{z}_j - \mathbf{t}_n\|^2$ to $\mathbf{t}_n$. The reference vectors are then updated by setting them equal to weighted averages of the data points given by

$$\mathbf{z}_i = \frac{\sum_n h_{ij(n)} \mathbf{t}_n}{\sum_n h_{ij(n)}}. \tag{21}$$

in which h_{ij} is a neighbourhood function associated with the ith node. This is generally chosen to be a uni-modal function of the feature map coordinates centred on the winning node, for example a Gaussian. The steps of identifying the winning nodes and updating the reference vectors are repeated iteratively. A key ingredient in the algorithm is that the width of the neighbourhood function h_{ij} starts with a relatively large value and is gradually reduced after each iteration.

4.1 Kernel versus Linear Regression

As pointed out by Mulier and Cherkassky (1995), the value of the neighbourhood function $h_{ij(n)}$ depends only on the identity of the winning node j and not on the value of the corresponding data vector $\mathbf{t}_n$. We can therefore perform partial sums over the groups $\mathcal{G}_j$ of data vectors assigned to each node j, and hence re-write (21) in the form

$$\mathbf{z}_i = \sum_j K_{ij} \mathbf{m}_j \tag{22}$$

in which $\mathbf{m}_j$ is the mean of the vectors in group $\mathcal{G}_j$ and is given by

$$\mathbf{m}_j = \frac{1}{N_j} \sum_{n \in \mathcal{G}_j} \mathbf{t}_n \tag{23}$$

where N_j is the number of data vectors in group $\mathcal{G}_j$. The result (22) is analogous to the Nadaraya-Watson kernel regression formula (Nadaraya 1964; Watson 1964) with the kernel functions given by

$$K_{ij} = \frac{h_{ij} N_j}{\sum_{j'} h_{ij'} N_{j'}}. \tag{24}$$

Thus the batch SOM algorithm replaces the reference vectors at each cycle with a convex combination of the node means $\mathbf{m}_j$, with coefficients determined by the neighbourhood function. Note that the kernel coefficients satisfy $\sum_j K_{ij} = 1$ for every i.

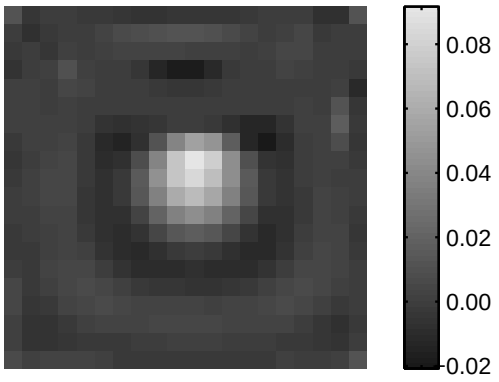

Figure 6: Example of the effective kernel F_{ij} plotted as a function of the node j for a given node i, for the oil flow data set after 3 iterations of EM. This kernel function is analogous to the (normalized) neighbourhood function in the SOM algorithm.

In the GTM algorithm, the centres $\mathbf{y}(\mathbf{x}_i; \mathbf{W})$ of the Gaussian components can be regarded as analogous to the reference vectors $\mathbf{z}_i$ of the SOM. We can evaluate $\mathbf{y}(\mathbf{x}_i; \mathbf{W})$ by solving the M-step equation (12) to find $\mathbf{W}$ and then using $\mathbf{y}(\mathbf{x}_i; \mathbf{W}) = \mathbf{W}\boldsymbol{\phi}(\mathbf{x}_i)$. If we define the weighted means of the data vectors by

$$\boldsymbol{\mu}_i = \frac{\sum_n R_{in}\mathbf{t}_n}{\sum_n R_{in}} \tag{25}$$

then we obtain

$$\mathbf{y}(\mathbf{x}_i; \mathbf{W}) = \sum_j F_{ij}\boldsymbol{\mu}_j \tag{26}$$

where we have introduced the *effective kernel* F_{ij} given by

$$F_{ij} = \boldsymbol{\phi}^{\mathrm{T}}(\mathbf{x}_i)\left(\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{G}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\phi}(\mathbf{x}_j)G_{jj}. \tag{27}$$

Note that the effective kernel satisfies $\sum_j F_{ij} = 1$. To see this, we first use (27) to show that $\sum_j F_{ij}\phi_l(\mathbf{x}_j) = \phi_l(\mathbf{x}_i)$. Then if one of the basis functions l corresponds to a bias, so that $\phi_l(\mathbf{x}) =$ const., the result follows.

The solution for $\mathbf{y}(\mathbf{x}_i; \mathbf{W})$ given by (26) and (27) can be interpreted as a weighted least-squares regression (Mardia, Kent, and Bibby 1979) in which the 'target' vectors are the $\boldsymbol{\mu}_i$, and the weighting coefficients are given by G_{jj}.

Figure 6 shows an example of the effective kernel for GTM corresponding to the oil flow problem discussed in Section 3.

From (22) and (26) we see that both GTM and SOM can be regarded as forms of kernel smoothers. However, there are two key differences. The first is that in SOM the vectors which are smoothed, defined by (23), correspond to hard assignments of data points to nodes, whereas the corresponding vectors in GTM, given by (25), involve soft assignments, weighted by the posterior probabilities. This is analogous to the distinction between K-means clustering (hard assignments) and fitting a standard Gaussian mixture model using EM (soft assignments).

The second key difference is that the kernel function in SOM is made to shrink during the course of the algorithm in an arbitrary, hand-crafted manner. In GTM the posterior probability distribution in latent space, for a given data point, forms a localised 'bubble' and the radius of this bubble shrinks *automatically* during training, as shown in Figure 7. This responsibility bubble governs the extent to which individual data points contribute towards the vectors $\boldsymbol{\mu}_i$ in (25) and hence towards the updating of the Gaussian centres $\mathbf{y}(\mathbf{x}_i; \mathbf{W})$ via (26).

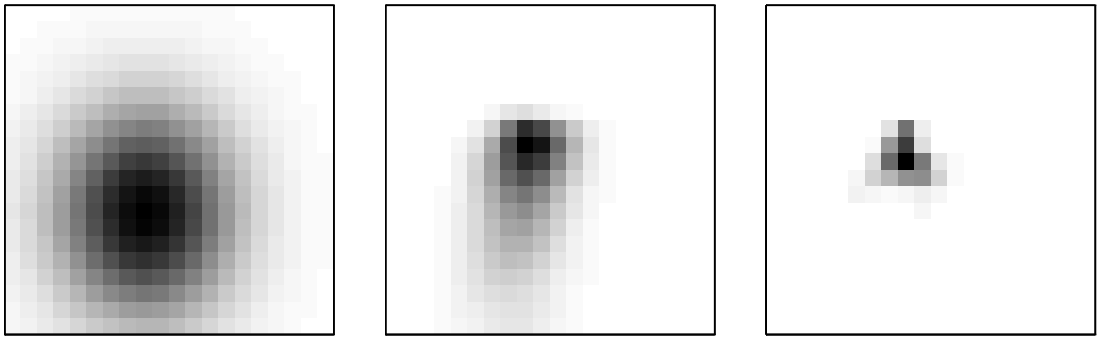

Figure 7: Examples of the posterior probabilities (responsibilities) R_{in} of the latent space points at an early stage (left), intermediate stage (centre) and late stage (right) during the convergence of the GTM algorithm. These have been evaluated for a single data point from the training set in the oil-flow problem discussed in Section 3, and are plotted using a non-linear scaling of the form $p(\mathbf{x}|\mathbf{t}_n)^{0.1}$ to highlight the variation over the latent space. Notice how the responsibility 'bubble', which governs the updating of the weight matrix, and hence the updating of the data-space vectors $\mathbf{y}(\mathbf{x}_i; \mathbf{W})$, shrinks *automatically* during the learning process.

4.2 Comparison of GTM with SOM

The most significant difference between the GTM and SOM algorithms is that GTM defines an explicit probability density given by the mixture distribution in (5). As a consequence there is a well-defined objective function given by the log likelihood (6), and convergence to a (local) maximum of the objective function is guaranteed by the use of the EM algorithm (Dempster, Laird, and Rubin 1977). This also provides a direct means to compare different choices of model parameters, and even to compare a GTM solution with another density model, by evaluating the likelihood of a test set under the generative distributions of the respective models. For the SOM algorithm, however, there is no probability density and no well-defined objective function which is being minimized by the training process. Indeed it has been proven (Erwin, Obermayer, and Schulten 1992) that such an objective function cannot exist for the SOM.

A further limitation of the SOM, highlighted in Kohonen (1995, page 234), is that the conditions under which so-called 'self-organization' of the SOM occurs have not been quantified, and so in practice it is necessary to confirm empirically that the trained model does indeed have the desired spatial ordering. In contrast, the neighbourhood-preserving nature of the GTM mapping is an automatic consequence of the choice of a continuous function $\mathbf{y}(\mathbf{x}; \mathbf{W})$.

Similarly, the smoothness properties of the SOM are determined indirectly by the choice of neighbourhood function and by the way in which it is changed during the course of the algorithm, and is therefore difficult to control. Thus, prior knowledge about the form of the map cannot easily be specified. The prior distribution for GTM, however, can be controlled directly, and properties such as smoothness are governed explicitly by basis function parameters, as illustrated in Figure 3.

Finally, we consider the relative computational costs of the GTM and SOM algorithms. For problems involving data in high-dimensional spaces the dominant computational cost of GTM arises from the evaluation of the Euclidean distances from every data point to every Gaussian centre $\mathbf{y}(\mathbf{x}_i; \mathbf{W})$. Since exactly the same calculations must be done for SOM (involving the distances of data points from the reference vectors $\boldsymbol{\mu}_i$) we expect one iteration of either algorithm to take approximately the same time. An empirical comparison of the computational cost of GTM and SOM was obtained by running each algorithm on the oil flow data until 'convergence' (defined as no discernible change in the appearance of the visualization map). The GTM algorithm took

1058 sec. (40 iterations) while the batch SOM took 1011 sec. (25 iterations) using a Gaussian neighbourhood function. With a simple 'top-hat' neighbourhood function, in which each reference vector is updated at each iteration using only data points associated with nearby reference vectors, the CPU time for the SOM algorithm is reduced to 305sec. (25 iterations). One potential advantage of GTM in practical applications arises from a reduction in the number of experimental training runs needed since both convergence and topographic ordering are guaranteed.

5 Relation to Other Algorithms

There are several algorithms in the published literature which have close links with GTM. Here we review briefly the most significant of these.

The elastic net algorithm of Durbin and Willshaw (1987) can be viewed as a Gaussian mixture density model, fitted by penalized maximum likelihood. The penalty term encourages the centres of Gaussians corresponding to neighbouring points along the (typically one-dimensional) chain to be close in data space. It differs from GTM in that it does not define a continuous data space manifold. Also, the training algorithm generally involves a hand-crafted annealing of the weight penalty coefficient.

There are also similarities between GTM and principal curves and principal surfaces (Hastie and Stuetzle 1989; LeBlanc and Tibshirani 1994) which again involve a two-stage algorithm consisting of projection followed by smoothing, although these are not generative models. It is interesting to note that Hastie and Stuetzle (1989) propose reducing the spatial width of the smoothing function during learning, in a manner analogous to the shrinking of the neighbourhood function in the SOM. A modified form of the principal curves algorithm (Tibshirani 1992) introduces a generative distribution based on a mixture of Gaussians, with a well-defined likelihood function, and is trained by the EM algorithm. However, the number of Gaussian components is equal to the number of data points, and smoothing is imposed by penalizing the likelihood function with the addition of a derivative-based regularization term.

The technique of parametrized self-organizing maps (PSOMs) involves first fitting a standard SOM model to a data set and then finding a manifold in data space which interpolates the reference vectors (Ritter 1993). Although this defines a continuous manifold, the interpolating surface does not form part of the training algorithm, and the basic problems in using SOM, discussed in Section 4.2, remain.

The SOM has also been used for vector quantization. In this context it has been shown how a re-formulation of the vector quantization problem (Luttrell 1990; Buhmann and Kühnel 1993; Luttrell 1994) can avoid many of the problems with the SOM procedure discussed earlier.

Finally, the 'density network' model of MacKay (1995) involves transforming a simple distribution in latent space to a complex distribution in data space by propagation through a non-linear network. A discrete distribution in latent space is again used, which is interpreted as an approximate Monte Carlo integration over the latent variables needed to define the data space distribution. GTM can be seen as a particular instance of this framework in which the sampling of latent space is regular rather than stochastic, a specific form of non-linearity is used, and the model parameters are adapted using EM.

6 Discussion

In this paper we have introduced a form of non-linear latent variable model which can be trained efficiently using the EM algorithm. Viewed as a topographic mapping algorithm, it has the key property that it defines a probability density model.

As an example of the significance of having a probability density, consider the important practical problem of dealing with missing values in the data set (in which some components of the data vectors $\mathbf{t}_n$ are unobserved). If the missing values are *missing at random* (Little and Rubin 1987) then the likelihood function is obtained by integrating out the unobserved values. For the GTM model the integrations can be performed analytically, leading to a simple modification of the EM algorithm.

A further consequence of having a probabilistic approach is that it is straightforward to consider a *mixture* of GTM models. In this case the overall density can be written as

$$p(\mathbf{t}) = \sum_r P(r)p(\mathbf{t}|r) \tag{28}$$

where $p(\mathbf{t}|r)$ represents the rth model, with its own set of independent parameters, and $P(r)$ are mixing coefficients satisfying $0 \leq P(r) \leq 1$ and $\sum_r P(r) = 1$. Again, it is straightforward to extend the EM algorithm to maximize the corresponding likelihood function.

The GTM algorithm can be extended in other ways, for instance by allowing independent mixing coefficients π_i (prior probabilities) for each of the Gaussian components, which again can be estimated by a straightforward extension of the EM algorithm. Instead of being independent parameters, the π_i can be determined as smooth functions of the latent variables using a normalized exponential applied to a generalized linear regression model, although in this case the M-step of the EM algorithm would involve non-linear optimization. Similarly, the inverse noise variance β can be generalized to a function of $\mathbf{x}$. An important property of GTM is the existence of a smooth manifold in data space, which allows the local 'magnification factor' between latent and data space to be evaluated as a function of the latent space coordinates using the techniques of differential geometry (Bishop, Svensén, and Williams 1996b). Finally, since there is a well-defined likelihood function, it is straightforward in principle to introduce priors over the model parameters (as discussed in Section 2.1) and to use Bayesian techniques in place of maximum likelihood.

Throughout this paper we have focussed on the batch version of the GTM algorithm in which all of the training data are used together to update the model parameters. In some applications it will be more convenient to consider sequential adaptation in which data points are presented one at a time. Since we are minimizing a differentiable cost function, given by (6), a sequential algorithm can be obtained by appealing to the Robbins-Monro procedure (Robbins and Monro 1951; Bishop 1995) to find a zero of the objective function gradient. Alternatively, a sequential form of the EM algorithm can be used (Titterington, Smith, and Makov 1985).

A web site for GTM is provided at:

`http://www.ncrg.aston.ac.uk/GTM/`

which includes postscript files of relevant papers, a software implementation in Matlab (a C implementation is under development), and example data sets used in the development of the GTM algorithm.

Acknowledgements

This work was supported by EPSRC grant GR/K51808: *Neural Networks for Visualization of High-Dimensional Data.* We would like to thank Geoffrey Hinton, Iain Strachan and Michael Tipping for useful discussions. Markus Svensén would like to thank the staff of the SANS group in Stockholm for their hospitality during part of this project.

References

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition.* Oxford University Press.

Bishop, C. M. and G. D. James (1993). Analysis of multiphase flows using dual-energy gamma densitometry and neural networks. *Nuclear Instruments and Methods in Physics Research A327*, 580–593.

Bishop, C. M., M. Svensén, and C. K. I. Williams (1996a). A fast EM algorithm for latent variable density models. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo (Eds.), *Advances in Neural Information Processing Systems*, Volume 8, pp. 465–471. MIT Press.

Bishop, C. M., M. Svensén, and C. K. I. Williams (1996b). Magnification factors for the GTM algorithm. To appear in Proceedings Fifth IEE International Conference on Artificial Neural Networks.

Buhmann, J. and K. Kühnel (1993). Vector quantization with complexity costs. *IEEE Transactions on Information Theory 39*(4), 1133–1145.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, B 39*(1), 1–38.

Durbin, R. and D. Willshaw (1987). An analogue approach to the travelling salesman problem. *Nature 326*, 689–691.

Erwin, E., K. Obermayer, and K. Schulten (1992). Self-organizing maps: ordering, convergence properties and energy functions. *Biological Cybernetics 67*, 47–55.

Hastie, T. and W. Stuetzle (1989). Principal curves. *Journal of the American Statistical Association 84*(406), 502–516.

Hinton, G. E., C. K. I. Williams, and M. D. Revow (1992). Adaptive elastic models for hand-printed character recognition. In J. E. Moody, S. J. Hanson, and R. P. Lippmann (Eds.), *Advances in Neural Information Processing Systems*, Volume 4, pp. 512–519. Morgan Kauffmann.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics 43*, 59–69.

Kohonen, T. (1995). *Self-Organizing Maps.* Berlin: Springer-Verlag.

LeBlanc, M. and R. Tibshirani (1994). Adaptive principal surfaces. *Journal of the American Statistical Association 89*(425), 53–64.

Little, R. J. A. and D. B. Rubin (1987). *Statistical Analysis with Missing Data.* New York: John Wiley.

Luttrell, S. P. (1990). Derivation of a class of training algorithms. *IEEE Transactions on Neural Networks 1*(2), 229–232.

Luttrell, S. P. (1994). A Bayesian analysis of self-organizing maps. *Neural Computation 6*(5), 767–794.

MacKay, D. J. C. (1995). Bayesian neural networks and density networks. *Nuclear Instruments and Methods in Physics Research, A 354*(1), 73–80.

Mardia, K., J. Kent, and M. Bibby (1979). *Multivariate analysis.* Academic Press.

Mulier, F. and V. Cherkassky (1995). Self-organization as an iterative kernel smoothing process. *Neural Computation 7*(6), 1165–1177.

Nadaraya, É. A. (1964). On estimating regression. *Theory of Probability and its Applications 9*(1), 141–142.

Ritter, H. (1993). Parametrized self-organizing maps. In *Proceedings ICANN'93 International Conference on Artificial Neural Networks*, Amsterdam, pp. 568–575. Springer-Verlag.

Robbins, H. and S. Monro (1951). A stochastic approximation method. *Annals of Mathematical Statistics 22*, 400–407.

Tibshirani, R. (1992). Principal curves revisited. *Statistics and Computing 2*, 183–190.

Titterington, D. M., A. F. M. Smith, and U. E. Makov (1985). *Statistical Analysis of Finite Mixture Distributions*. New York: John Wiley.

Watson, G. S. (1964). Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics. Series A 26*, 359–372.