Semi-Supervised Learning of Hierarchical Latent Trait Models for Data Visualisation

Ian T. Nabney, Yi Sun, Peter Tiňo, and Ata Kabán

Ian T. Nabney is with the Neural Computing Research Group, Aston University, Birmingham, B4 7ET, United Kingdom.
E-mail: i.t.nabney@aston.ac.uk
Yi Sun is with the School of Computer Science, University of Hertfordshire, Hatfield, Herts AL10 9AB, United Kingdom.
E-mail: Y.2.Sun@herts.ac.uk
Peter Tiňo and Ata Kabán are with the School of Computer Science, University of Birmingham, Birmingham B15 2TT, United Kingdom.
E-mail: P.Tino, A.Kaban@cs.bham.ac.uk.

Abstract

Recently, we have developed the hierarchical Generative Topographic Mapping (HGTM), an interactive method for visualisation of large high-dimensional real-valued data sets. In this paper, we propose a more general visualisation system by extending HGTM in 3 ways, which allow the user to visualise a wider range of datasets and better support the model development process. (i) We integrate HGTM with noise models from the exponential family of distributions. The basic building block is the Latent Trait Model (LTM). This enables us to visualise data of inherently discrete nature, e.g. collections of documents in a hierarchical manner. (ii) We give the user a choice of initialising the child plots of the current plot in either *interactive*, or *automatic* mode. In the interactive mode the user selects "regions of interest", whereas in the automatic mode an unsupervised minimum message length (MML)-inspired construction of a mixture of LTMs is employed. The unsupervised construction is particularly useful when high-level plots are covered with dense clusters of highly overlapping data projections, making it difficult to use the interactive mode. Such a situation often arises when visualising large data sets. (iii) We derive general formulas for magnification factors in latent trait models. Magnification factors are a useful tool to improve our understanding of the visualisation plots, since they can highlight the boundaries between data clusters.

We illustrate our approach on a toy example and evaluate it on three more complex real data sets.

Index Terms

Hierarchical model, Latent trait model, Magnification factors, Data visualisation, Document mining.

I. Introduction

Topographic visualisation of multi-dimensional data has been an important method of data analysis and data mining for several years [4], [18]. Visualisation is an effective way for domain experts to detect clusters, outliers and other important structural features in data. In addition, it can be used to guide the data mining process itself by giving feedback on the results of analysis [23]. In this paper we use latent variable models to visualise data, so that a single plot may contain several data clusters; our aim is to provide sufficiently informative plots that the clusters can be *seen* to be distinct rather than confining each model to a single cluster (as would be appropriate for cluster analysis).

In a complex domain, however, a single two-dimensional projection of high-dimensional data may not be sufficient to capture all of the interesting aspects of the data. Therefore, hierarchical extensions of visualisation methods [7], [22] have been developed. These allow the user to 'drill down' into the data; each plot covers a smaller region and it is therefore easier to discern the structure of data. Also plots may be at an angle and so reveal more information. For example, clusters may be split apart instead of lying on top of each other.

Recently, we have developed a general and principled approach to the interactive construction of non-linear visualisation hierarchies [27], the basic building block of which is the Generative Topographic Mapping (GTM) [4]. GTM is a probabilistic reformulation of the self-organizing map (SOM) [17] in the form of a non-linear latent variable model with a spherical Gaussian noise model.

The extension of the GTM algorithm to discrete variables was described in [5] and a generalisation of this to the Latent Trait Model (LTM), a latent variable model class whose noise models are selected from the exponential family of distributions, was developed in [14]. In this paper we extend the hierarchical GTM (HGTM) visualisation system to incorporate LTMs. This enables us to visualise data of an inherently discrete nature, e.g. collections of documents.

A hierarchical visualisation plot is built in a recursive way; after viewing the plots at a given level, the user may add further plots at the next level down in order to provide more insight. These child plots can be trained using the EM algorithm [10], but their parameters must be initialized in some way. Existing hierarchical models do this by allowing the user to select the position of each child plot in an *interactive* mode; see [27]. In this paper, we show how to provide the user with an *automatic* initialization mode which works within the same principled probabilistic framework as is used for the overall hierarchy. The automatic mode allows the user to determine both the number and the position of child LTMs in an *unsupervised* manner. This is particularly valuable when dealing with large quantities of data that make visualisation plots at higher levels complex and difficult to deal with in an interactive manner.

An intuitively simple but flawed approach would be to use a data partitioning technique (e.g. [25]) for segmenting the data set, followed by constructing visualisation plots in the individual compartments. Clearly, in this case there would be no direct connection between the criterion for choosing the quantization regions and that of making the local low-dimensional projections. By employing LTM, however, such a connection can be established in a principled manner. This is achieved by exploiting the probabilistic nature of the model, which enables us to use a principled minimum message length (MML)-based learning of mixture models with an embedded model selection criterion this approach has been used for Gaussian mixture models [11][1]. Hence, given a parent LTM, the number and position of its children is based on the modelling properties of the children themselves – without any ad-hoc criteria which would be exterior to the model.

Previous experience has indicated that magnification factors may provide valuable additional information to the user's understanding of the visualisation plots, since they can highlight the boundaries between data clusters. In [6], formulas for magnification factors were only derived for the GTM. In this paper, we derive formulas for magnification factors in full generality for latent trait models.

In the next section we briefly review the latent trait model. In Section III, a hierarchical latent trait model is developed. Section IV presents the model selection criterion based on minimum message length that we apply to mixtures of LTMs. Section V presents and discusses experimental results and compares them with existing methods. We derive a general formula for magnification factors in LTMs in Section VI. Finally, Section VII summarizes the key contributions of the paper.

II. The Latent Trait Model (LTM)

Latent trait models [14] are generative models which are powerful and principled tools for data analysis and visualisation. As a generalisation of the Generative Topographic Mapping (GTM) [4], the latent trait model family [14] offers a framework which includes the definition of appropriate probability models for discrete observations.

Consider an L-dimensional *latent space* $\mathcal{H}$, which, for visualisation purposes is typically a bounded 2-D Euclidean domain, e.g. the square $[-1, 1] \times [-1, 1]$. The aim is to represent multi-dimensional data vectors $\{\mathbf{t}_n\}_{n=1,\ldots,N}$ using the latent space so that "important" structural characteristics are revealed. A non-linear function maps the latent space to the data space $\mathcal{D} = \Re^D$. The latent plane (assuming a two-dimensional latent space) becomes a (non-linear) 2-D manifold in the high dimensional data space.

For tractability, the latent space is discretized by introducing a regular array (or grid) of K latent points $\boldsymbol{x}_k \in \mathcal{H}, k = 1, \ldots, K$ (which are analogous to the nodes of the SOM [18]). A

[1]This framework uses Jeffrey's priors, which implies that the estimation of the model parameters is equivalent to a maximum likelihood (ML) formulation. The MML criterion penalises overly-complex models but does *not* regularise the model parameters themselves.

uniform prior distribution is imposed over the latent points $\boldsymbol{x}_k$, leading to the following expression for the unconditional data density of an observed data point $\mathbf{t} \in \Re^D$

$$p(\mathbf{t}) = \sum_{k=1}^{K} p(\mathbf{t}|\boldsymbol{x}_k)p(\boldsymbol{x}_k) = K^{-1}\sum_{k=1}^{K} p(\mathbf{t}|\boldsymbol{x}_k). \tag{1}$$

The conditional data distribution, $p(\mathbf{t}|\boldsymbol{x}_k)$, (conditioned on the kth latent space point $\boldsymbol{x}_k \in \mathcal{H}$ is modelled as a member of the exponential family in a parameterised functional form [2]

$$p_B(\mathbf{t}|\boldsymbol{x}_k, \boldsymbol{\Theta}) = \exp\left\{\boldsymbol{f}_{\boldsymbol{\Theta}}(\boldsymbol{x}_k)\mathbf{t} - B(\boldsymbol{f}_{\boldsymbol{\Theta}}(\boldsymbol{x}_k))\right\} p_0(\mathbf{t}). \tag{2}$$

Here $\boldsymbol{\Theta}$ is the parameter vector of the model, $B(\boldsymbol{f}_{\boldsymbol{\Theta}}(\boldsymbol{x}_k)) = \ln\int \exp(\boldsymbol{f}_{\boldsymbol{\Theta}}(\boldsymbol{x}_k)\mathbf{t})p_0(\mathbf{t})\, d\mathbf{t}$ denotes the cumulant generating function of $p(\mathbf{t}|\boldsymbol{x}_k)$, and $p_0(\mathbf{t})$ is a factor independent of $\boldsymbol{\Theta}$. Recall that the exponential family includes the Gaussian and Student t-distributions and also discrete random variables such as the Bernoulli and multinomial distributions.

The function $\boldsymbol{f}(\cdot)$ represents a smooth mapping from latent to data space; in order to make training fast, $\boldsymbol{f}$ has the form of a General Linear Regression model, and is defined by $\boldsymbol{f}_{\boldsymbol{\Theta}}(\boldsymbol{x}_k) = \boldsymbol{\Theta}\boldsymbol{\phi}(\boldsymbol{x}_k)$, where $\boldsymbol{\Theta} \in \Re^{D\times M}$ is a parameter matrix and $\boldsymbol{\phi}(\cdot) = (\phi_1(\cdot), ..., \phi_M(\cdot))^T, \phi_m(\cdot) : \mathcal{H} \rightarrow \Re$, is a fixed set of M non-parametric nonlinear basis functions. These could be any smooth functions; typically Gaussian radial basis functions are employed. A linear basis function $\phi_0(\boldsymbol{x}) = 1, \forall \boldsymbol{x}$, may be included to account for the bias term (which is set to the data mean). The notation $\boldsymbol{\phi}_k = \boldsymbol{\phi}(\boldsymbol{x_k})$ will be used as a shorthand.

A latent trait model with fixed parameters $\boldsymbol{\Theta}$ defines a density in the data space,

$$\boldsymbol{z} : \mathcal{H} \rightarrow \Re^D, \quad \boldsymbol{z}(\boldsymbol{x}_k) = \boldsymbol{b}(\boldsymbol{\Theta}\boldsymbol{\Phi}(\boldsymbol{x}_k)) = \boldsymbol{b}(\boldsymbol{\Theta}\boldsymbol{f}(\boldsymbol{x}_k)). \tag{3}$$

This probabilistic interpretation is fundamental to our approach to constructing a hierarchy of models. We refer to the manifold $\boldsymbol{f}(\mathcal{H})$ as the *projection manifold* of the LTM.

LTMs are trained to maximize the likelihood of the training set $\zeta = \{\mathbf{t}_1, ..., \mathbf{t}_N\}$ using an EM algorithm [10], the M-step of which consists of solving the equation

$$\mathbf{T}\mathbf{R}^T\boldsymbol{\Phi}^T = \mathbf{b}(\boldsymbol{\Theta}\boldsymbol{\Phi})\boldsymbol{G}\boldsymbol{\Phi}^T \tag{4}$$

for $\boldsymbol{\Theta}$, where the function $\mathbf{b}(\cdot)$ denotes the gradient of the cumulant function $B(\cdot)$[2], $\boldsymbol{\Phi}$ is an $M \times K$ matrix with $\boldsymbol{\phi}_k$ in its k-th column, $\mathbf{T}$ is the data matrix including N data vectors $\{\mathbf{t}_n\}$ as

[2]It is the inverse link function [21] of the noise distribution.

columns, $\mathbf{R} = (R_{kn})_{k=1,\ldots,K,n=1,\ldots,N}$ and $\boldsymbol{G}$ is a diagonal matrix with elements $g_{kk} = \sum_{n=1}^{N} R_{kn}$, where R_{kn}, computed via Bayes' theorem in the E-step,

$$R_{kn} = p(\boldsymbol{x}_k|\mathbf{t}_n) = \frac{p(\mathbf{t}_n|\boldsymbol{x}_k, \boldsymbol{\Theta})p(\boldsymbol{x}_k)}{\sum_{k'=1}^{K} p(\mathbf{t}_n|\boldsymbol{x}_{k'}, \boldsymbol{\Theta})p(\boldsymbol{x}_{k'})}, \tag{5}$$

is the "responsibility" of the latent point $\boldsymbol{x}_k$ for generating $\mathbf{t}_n$. The E-step and M-step are iterated until the change in likelihood falls below a user-defined threshold.

Once trained, the LTM can be used for visualisation. To do this, the map $\boldsymbol{f}$ has to be 'inverted' so that there is a point latent space corresponding to each data point. The latent space representation of a point $\mathbf{t}_n$ is taken to be the mean of the posterior distribution $p(\boldsymbol{x}_k|\mathbf{t}_n)$ over the latent space. This can be computed using (5) and averaging R_{kn} over k (because the prior density for each x_k is equal).

III. General Framework for Hierarchical Latent Trait Models

When dealing with large and complex data sets, a single global visualisation plot is often not sufficient to get a good understanding of the relationships in the data. In order to represent complex intrinsic information when visualizing large data sets, hierarchical visualisation systems have been proposed and developed in the literature, [7], [27]. In [7], a locally linear hierarchical visualisation system was defined. We have recently extended this system to hierarchies of non-linear GTM projection manifolds in [27]. This paper showed that in many cases the use of a non-linear latent space model significantly reduced the number of visualisation plots required to get good inter-cluster separation and represent the data structure.

In this section we provide a general formulation of hierarchical latent trait mixture models. The benefit of this to the user is that a wider range of conditional density models $p_B(\mathbf{t}|\boldsymbol{x}_k, \boldsymbol{\Theta})$ can be used. For example, binary data can be visualised using a Bernoulli distribution [14]. If the data contains outliers, a Student t-distribution may be more appropriate than the Gaussian used in HGTM. Preliminary results of organising LTMs into a hierarchy have been encouraging [15], and motivated the work described in this paper.

The hierarchical LTM arranges a set of LTMs and their corresponding plots in a tree structure $\mathcal{T}$. The *Root* is at level 1, children of level-ℓ models are at level $\ell + 1$.

Each model $\mathcal{M}$ in the hierarchy, except for *Root*, has an associated parent-conditional mixture coefficient, or prior, $\pi(\mathcal{M}|Parent(\mathcal{M}))$. The priors are non-negative and satisfy the consistency condition: $\sum_{\mathcal{M} \in Children(\mathcal{N})} \pi(\mathcal{M}|\mathcal{N}) = 1$. Unconditional priors for the models are recursively calculated as follows: $\pi(Root) = 1$, and for all other models

$$\pi(\mathcal{M}) = \prod_{i=2}^{Level(\mathcal{M})} \pi(Path(\mathcal{M})_i | Path(\mathcal{M})_{i-1}), \tag{6}$$

where $Path(\mathcal{M}) = (Root, \ldots, \mathcal{M})$ is the $\mathcal{P}$-tuple of nodes defining the path of length $\mathcal{P}$ in $\mathcal{T}$ from *Root* to $\mathcal{M}$.

The leaves($\mathcal{T}$) of the tree are defined to be the set of nodes of $\mathcal{T}$ without children. The distribution defined by the hierarchical model is a mixture of distributions defined by the leaves of $\mathcal{T}$

$$P(\mathbf{t}|\mathcal{T}) = \sum_{\mathcal{M} \in Leaves(\mathcal{T})} \pi(\mathcal{M}) P(\mathbf{t}|\mathcal{M}). \tag{7}$$

Non-leaf models have two roles in the hierarchy.

1) Every model is a leaf model at some point during the construction of the hierarchy.
2) Non-leaf models are useful for determining the relationship between sub-plots in the hierarchy.

A. Training

The hierarchical LTM is trained using EM to maximize its likelihood with respect to the data sample $\zeta = \{\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_N\}$. Training of a hierarchy of LTMs proceeds in a recursive fashion. First, the *Root* LTM is trained and used to visualize the data. Then the user identifies interesting regions on the visualisation plot that they would like to model in a greater detail.

Having trained models $\mathcal{N}$ at level ℓ, the expectation of the complete data likelihood of level-$(\ell+1)$ is

$$\begin{aligned} < \mathcal{L}_{comp}^{\ell+1} > = \sum_{n=1}^{N} \sum_{\mathcal{N} \in Nodes(l)} & P(\mathcal{N}|\mathbf{t}_n) \\ & \sum_{\mathcal{M} \in Children(\mathcal{N})} P(\mathcal{M}|\mathcal{N}, \mathbf{t}_n) \\ & \sum_{k=1}^{K_{\mathcal{M}}} R_{kn}^{\mathcal{M}} \ln\{\pi(\mathcal{N})\pi(\mathcal{M}|\mathcal{N}) P(\mathbf{t}_n, \boldsymbol{x}_k^{\mathcal{M}})\} \end{aligned} \tag{8}$$

1) E-step: In the E-step, we estimate the posterior distribution of all hidden variables, using the "old" values of LTM parameters. Given a data point $\mathbf{t}_n$, we compute the model responsibilities corresponding to the competition among models belonging to the same parent as

$$P(\mathcal{M}|Parent(\mathcal{M}), \mathbf{t}_n) = \frac{\pi(\mathcal{M}|Parent(\mathcal{M}))P(\mathbf{t}_n|\mathcal{M})}{\sum_{\mathcal{M}' \in [\mathcal{M}]} \pi(\mathcal{M}'|Parent(\mathcal{M}))P(\mathbf{t}_n|\mathcal{M}')}, \tag{9}$$

where

$$[\mathcal{M}] = Children(Parent(\mathcal{M})). \tag{10}$$

Imposing $P(Root|\mathbf{t}_n) = 1$, the unconditional (on parent) model responsibilities are recursively determined by

$$P(\mathcal{M}|\mathbf{t}_n) = P(\mathcal{M}|Parent(\mathcal{M}), \mathbf{t}_n)P(Parent(\mathcal{M})|\mathbf{t}_n). \tag{11}$$

Responsibilities of the latent space centres $\boldsymbol{x}_k^{\mathcal{M}}$, $k = 1, 2, ..., K_{\mathcal{M}}$, corresponding to the competition among the latent space centres in each model $\mathcal{M}$, are calculated using (5).

2) M-step: In the M-step, we estimate the parameters using the posterior over hidden variables computed in the E-step.

Parent-conditional mixture coefficients are determined using

$$\pi(\mathcal{M}|Parent(\mathcal{M})) = \frac{\sum_{n=1}^{N} P(\mathcal{M}|\mathbf{t}_n)}{\sum_{n=1}^{N} P(Parent(\mathcal{M})|\mathbf{t}_n)}. \tag{12}$$

Parameters $\mathbf{\Theta}^{(\mathcal{M})}$ of the LTM $\mathcal{M}$ are calculated by solving

$$\mathbf{T}\mathbf{R}^{(\mathcal{M})T}\mathbf{\Phi}^T = \mathbf{b}(\mathbf{\Theta}^{(\mathcal{M})}\mathbf{\Phi})\boldsymbol{G}^{(\mathcal{M})}\mathbf{\Phi}^T, \tag{13}$$

where $\mathbf{R}^{(\mathcal{M})} = (R_{kn}^{\mathcal{M}})_{k=1,...,K,n=1,...,N}$. $R_{kn}^{\mathcal{M}}$ are scaled (by (11)) responsibilities (5), $R_{kn}^{\mathcal{M}} = P(\mathcal{M}|\mathbf{t}_n)R_{kn}$; $\boldsymbol{G}^{(\mathcal{M})}$ is a diagonal matrix with elements $g_{kk}^{\mathcal{M}} = \sum_{n=1}^{N} R_{kn}^{\mathcal{M}}$.

When solving (13), if the link function $\mathbf{b}(\cdot)$ is the identity, one gets the closed form M-step of HGTM[3] [27], but in general a non-linear optimization algorithm is required. In the simplest

[3]Even though we treat GTM as a special case of LTM with spherical Gaussian noise model, (2) does not account for the "width" parameter. We decided to use the simplified formulation (2), because it is sufficient for all other interesting noise models, such as Bernoulli, Poisson, multinomial etc. In the case of spherical Gaussian noise model, solving (13) sets the means of the Gaussians and the width parameter needs to be updated as in [27].

case, we may employ a gradient-based inner loop M-step[4]:

$$\Delta\Theta^{(\mathcal{M})} \propto \left\{ \mathbf{T}\mathbf{R}^{(\mathcal{M})T} - \mathbf{b}(\Theta^{(\mathcal{M})}\Phi)\boldsymbol{G}^{(\mathcal{M})} \right\} \Phi^T. \tag{14}$$

Training times are dependent on the dataset and the number of levels in the hierarchy. For the examples presented in this paper (of up to 8000 data points), training times for the complete hierarchy were in the order of 2–4 hours for a 1GHz Linux PC running MATLAB. For data of a fixed complexity, the algorithm scales linearly in the number of examples and the dimensionality of the data space. Note that visualisation of a large dataset using a trained model is relatively quick (less than a minute). Because our model can generalise, it is always possible to train it on a smaller subset of the data and thus to tackle very large datasets in practice.

B. Model initialization

When initializing sub-models there are two things to determine: the number of sub-models and the initial parameters of the sub-models. We view the problem of initializing sub-model parameters primarily as one of locating which region of data space each sub-model should be responsible for. To do this, regions of interest are defined by the user in the latent (visualisation) space. The points $\mathbf{c}_i$ selected in the latent space $\mathcal{H}$ correspond to the "centres" of these regions.

These "centres" of the "regions of interest" are mapped back to the data space and Voronoi compartments [1] defined by the mapped points $\boldsymbol{z}(\mathbf{c}_i) \in \mathcal{D}$, where $\boldsymbol{z}$ is the map (3) of the corresponding LTM, are calculated in the data space. In the case of a Gaussian noise model, the child LTMs are initialized by local PCA in the corresponding Voronoi compartments [27]. When using other noise models such as Bernoulli or multinomial distributions, the PCA-initialised LTMs are in addition individually trained (section II) in their corresponding Voronoi compartments for 1 EM iteration. The EM iteration "settles" the component LTMs to their corresponding modelling regions. Empirically, this initialisation strategy works very well. We perform this additional initialization step when the PCA initialisation alone does not "match" the noise distribution well, e.g. when the noise distribution is non-symmetric or the data space is discrete.

After the initialisation of each child model, the full hierarchical training described in section III-A is used.

[4]In this partial M-step we could alternatively use iterative reweighted least squares [35].

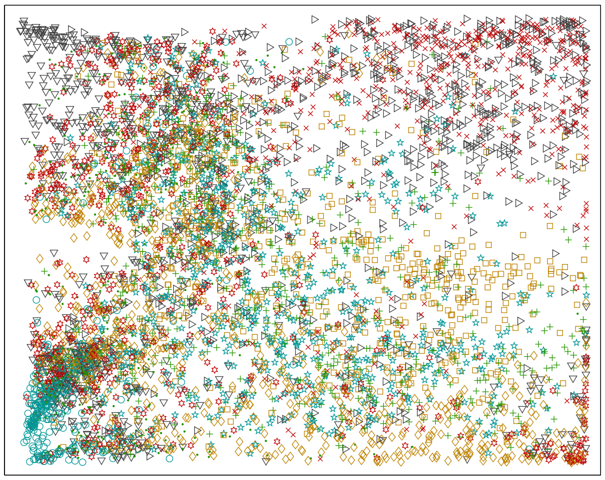


Fig. 1. An example of strongly overlapping clusters: visualisation of a collection of documents with a single Latent Trait Model. The documents are classified according to the topic they cover. Each of 10 topic classes is assigned a unique marker.

C. Plotting the projections

We adopt the strategy used in [7], [27] and take advantage of the probabilistic nature of our model by plotting projections of all the data points on every plot, but modifying the intensity in proportion to the responsibility $P(\mathcal{M}|\ \mathbf{t}_n)$ (11) which each plot (sub-model $\mathcal{M}$) has for the data point $\mathbf{t}_n$. Points that are not well captured by a particular plot will appear with low intensity.

IV. Unsupervised learning of mixtures of LTMs

In previous sections, we have developed a general framework for a visualisation hierarchy. The user selects the "regions of interest" to select initial locations of child models and extend the visualisation hierarchy. This method is powerful when the clusters are separated clearly in the 2-D latent space. On the other hand, when facing a cluttered plot like that in Figure 1, where thousands of data points are shown (with densely clustered and overlapping projections), the user may be unable to determine where sub-models should be placed. In order to resolve this problem, we have developed an alternative initialisation algorithm which decides both the number of sub-models and their location automatically. As far as we are aware, there is no other algorithm for automatic initialisation of sub-plots in a hierarchical visualisation model. In this section we will focus solely on the algorithm for mixture models.

A. MML formulation for unsupervised learning of mixture models

Given a set $\zeta = \{\mathbf{t}_1, \mathbf{t}_2, ..., \mathbf{t}_N\}$ of data points, minimum message length (MML) strategies select, among the models inferred from ζ, the one which minimizes length of the message transmitting ζ [28]. Given that the data is modeled by a parametric probabilistic model $P(\zeta|\boldsymbol{\theta})$, the message consists of two parts – one specifying the model parameters, the other specifying the data given the model: $\text{Length}(\boldsymbol{\theta}, \zeta) = \text{Length}(\boldsymbol{\theta}) + \text{Length}(\zeta|\boldsymbol{\theta})$.

The MML principle was first applied to unsupervised learning of mixture models in [29] and was extended to hierarchical models in [8]. A computer program, Snob, that uses these principles for both parameter estimation and model selection was described in [30]; this provides a flat clustering model. The hierarchical model used in these papers differs from ours in three main ways: firstly, only the leaf nodes define a probability density, while our hierarchy defines a density at all levels; secondly, the earlier model has relatively simple distribution models for clustering, while we allow more complex component models (LTMs) which support visualisation; thirdly, a heuristic algorithm is used to train the hierarchy, while we use EM.

Recently, Figueiredo and Jain [11] have developed an MML framework for unsupervised learning of mixture models; with the choice of a Jeffrey's prior, the algorithm selects the "appropriate" number of components while the parameters of each model are estimated by ML. (A similar approach for other density models was formulated in [30] and [32]). The novelty of their proposed approach is that parameter estimation and model selection are integrated in a single EM algorithm, rather than using a model selection criterion on a set of pre-estimated candidate models.

The particular form of MML criterion adopted in [11] is of the form $\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\text{argmin}}\ \boldsymbol{\mathcal{L}}(\boldsymbol{\theta}, \zeta)$, where

$$\begin{aligned} \boldsymbol{\mathcal{L}}(\boldsymbol{\theta}, \zeta) = -\log P(\boldsymbol{\theta}) - \log P(\zeta|\boldsymbol{\theta}) + \\ \frac{1}{2}\log|\mathbf{I}(\boldsymbol{\theta})| + \frac{c}{2}\left(1 + \log\frac{1}{12}\right), \end{aligned} \tag{15}$$

where $\mathbf{I}(\boldsymbol{\theta})$ is the expected Fisher information matrix, $|\mathbf{I}(\boldsymbol{\theta})|$ is its determinant, and c is the number of free parameters, i.e. the dimension of $\boldsymbol{\theta}$. This approach was first proposed in [33].

By imposing a non-informative Jeffreys' prior [3] on both the vector of mixing coefficients $\{\pi(\mathcal{M})\}$ and the parameters $\boldsymbol{\Theta}^{(\mathcal{M})}$ of individual mixture components [11], the equation (15) becomes

$$\begin{aligned}\boldsymbol{\mathcal{L}}(\boldsymbol{\theta},\zeta) = \frac{Q}{2}\sum_{\pi(\mathcal{M})>0}\log\left(\frac{N\cdot\pi(\mathcal{M})}{12}\right)+ \\ \frac{A}{2}\log\frac{N}{12}+\frac{A(Q+1)}{2}-\log P(\zeta|\boldsymbol{\theta}),\end{aligned} \tag{16}$$

where A is the number of mixture components with positive prior $\pi(\mathcal{M}) > 0$ and Q is the number of free parameters of each individual mixture component. The use of a non-informative prior is mathematically convenient, since it cancels out the Fisher information matrix term $\mathbf{I}(\boldsymbol{\theta})$, which is complex to analyse and very expensive to compute. However, such a prior is formally equivalent to a Bayesian prior which favours parameter values around the values where the model is most sensitive [31], which is less than ideal for $p(\boldsymbol{\Theta})$. We can justify the choice by the very good empirical results that have been achieved [11] and by the fact that we will use this criterion only for child model initialization and not for child model training. The Jeffrey's prior over mixing coefficients favours extreme estimates (0 or 1) more strongly than other priors (such as minimum entropy and negative Dirichlet) but this stronger component pruning is beneficial for this application.

Minimizing (16) with respect to $\pi(\mathcal{M})$ under the constraint that the priors $\pi(\mathcal{M})$ sum to 1, the following re-estimation formulas are obtained [11]:

$$\hat{\pi}(\mathcal{M}) = \frac{\max\left\{0,\ -\frac{Q}{2}+\sum_{n=1}^{N}P(\mathcal{M}|\mathbf{t}_n)\right\}}{\sum_{\mathcal{M}'}\max\left\{0,\ -\frac{Q}{2}+\sum_{n=1}^{N}P(\mathcal{M}'|\mathbf{t}_n)\right\}}, \tag{17}$$

where component responsibilities $P(\mathcal{M}|\mathbf{t}_n)$ are determined by

$$P(\mathcal{M}|\mathbf{t}_n) = \frac{\pi(\mathcal{M})P(\mathbf{t}_n|\mathcal{M})}{\sum_{\mathcal{M}'}\pi(\mathcal{M}')P(\mathbf{t}_n|\mathcal{M}')}, \tag{18}$$

$$\pi(\mathcal{M}) = \frac{\sum_{n=1}^{N}P(\mathbf{t}_n|\mathcal{M})}{\sum_{\mathcal{M}'=1}^{A}\sum_{n=1}^{N}P(\mathbf{t}_n|\mathcal{M}')} \tag{19}$$

Free parameters of the individual LTMs are fitted to the data ζ using the EM algorithm outlined in section III applied to mixtures of LTMs[5]. This approach is not fully within the MML formalism, since there is no regularisation of the LTM model parameters themselves: instead, the mixing coefficients are regularised by (17). Note that LTMs corresponding to zero $\hat{\pi}(\mathcal{M})$ become irrelevant and so (17) effectively performs component annihilation [11].

[5]A mixture of LTMs can be considered a two-level hierarchical LTM. Mixture components are children of the *root*.

B. The algorithm

Given the training data $\zeta = \{\mathbf{t}_1, \mathbf{t}_2, ..., \mathbf{t}_N\}$, we use the MML approach to find the "appropriate" number of mixture component LTMs that "explain" ζ in a probabilistic manner. LTMs that are good probabilistic generating models of the data capture the data distribution well and hence yield "good" visualisation plots[6]. To start the training process, we choose the maximum number of components A_{max} we are willing to consider at the next level. This can be set to a large value and the MML training procedure will select the optimal number of components no greater than A_{max}. If more components are needed, then the child model can be further refined at lower levels of the hierarchy. Then, the algorithm initialises the component LTMs by randomly selecting A_{max} points from ζ and applying the method described in section III-B. The selected A_{max} points act as centres of regions of interest in the data space. In other words, they play the role of vectors $\boldsymbol{z}(\mathbf{c}_i)$ from section III-B.

As in [11], we adopt the component-wise EM (CEM) algorithm [9], i.e. rather than simultaneously updating all the LTMs, we first update the parameters $\mathbf{\Theta}^{(1)}$ of the first LTM (13), while parameters of the remaining LTMs are fixed, then we recompute the component responsibilities $\{P(\mathcal{M}|\mathbf{t}_n)\}$ (18) and mixture coefficients $\{\hat{\pi}(\mathcal{M})\}$ (17) for all components in the mixture. After this, we move to the second component, update $\mathbf{\Theta}^{(2)}$ in the same way, and recompute $\{P(\mathcal{M}|\mathbf{t}_n)\}$, $\{\hat{\pi}(\mathcal{M})\}$, etc., looping through all mixture components. If one of the component LTMs dies ($\hat{\pi}(\mathcal{M}) = 0$), redistribution of its probability mass to the remaining components increases their chance of survival. After convergence of CEM, we still have to check whether a shorter message length can be achieved by using a smaller number of mixture LTMs (down to $A = 1$).[7] This is achieved by iteratively killing off the weakest LTM (with the smallest $\hat{\pi}(\mathcal{M})$) and re-running CEM until convergence. Finally, the winning mixture of LTMs is the one that leads to the shortest message length $\boldsymbol{\mathcal{L}}(\boldsymbol{\theta}, \zeta)$ (16).

This training algorithm provides a very flexible approach to building a visualisation model. The user can specify a different maximal number of child plots at each decision point, and there

[6]This is a non-trivial issue, since while we can measure the quality of probabilistic models e.g. via likelihood, there is no universal quality measure for visualisation plots. But intuitively, good probabilistic properties of a LTM mean that the projection manifold follows closely the data distribution and so the visualisation plot is a "good" representation of the data distribution.

[7]If we knew that the number of mixture components was no less than some number A_{min}, we would stop at $A = A_{min}$ [11].

is no upper limit to the total number of plots in the hierarchy. Different parts of the hierarchy can be trained to different depths (i.e. there is no need for the tree to be balanced if that does not provide information). In addition, because each level of the hierarchy forms a probabilistic model of the data, it is possible for the user to 'retract' decisions; if a set of child plots does not provide additional insight, that group can be removed, returning the tree to its previous optimal state.

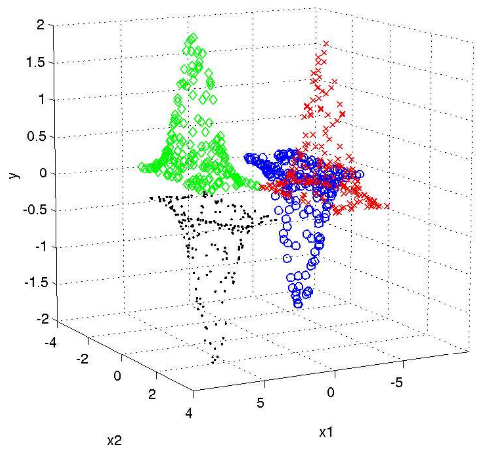
To demonstrate this algorithm, we did an experiment on a toy data set of 800 points $\mathbf{t} = (t_1, t_2, t_3)^T$ lying on four two-dimensional manifolds ("humps") (see Figure 2 (a)). We associated the points in the four "humps" with four different classes, $\mathcal{C}_i$, $i = 1, 2, 3, 4$, having four different labels. After training[8] ($A_{max} = 10$), a 6-component mixture was constructed. Projection manifolds of the 6 LTMs are shown in Figure 2 (b). Note that 6 child plots provide understandable subgroups of the data; and that the 6 projection manifolds closely approximate the four "humps"of the original generating manifold. The corresponding hierarchy of visualisation plots can be seen in Figure 3.

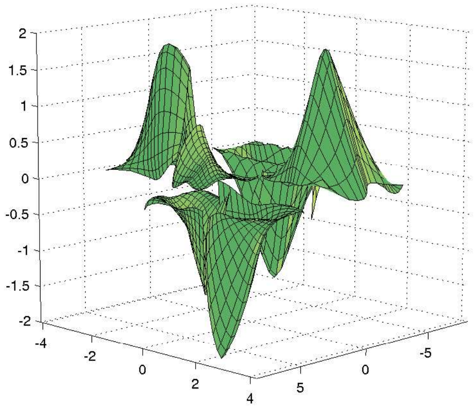
We stress that there is no contradiction between the number of components 6 in the final mixture of LTMs and the dataset composed of four "humps". There is no driving force in the MML formalism to achieve this and this is not the point of our study. The important thing is that the MML method finds a good number of subplots so that the overall probability of the data set is high (good projections) and the mixture model is not too complex (unnecessarily high number of subplots). At the same time, the MML method *automatically* finds appropriate *positions* of the projection manifolds in the data space. Another advantage of using the MML criterion with the EM algorithm is that training is less sensitive to model initialization [11]. The criterion reduces the number of local optima in the error function (for example, removing the pathological cases where the variance of a component collapses to zero) and so the fact that EM (like all deterministic algorithms) only finds a local optimum is less of a problem.

V. Semi-Supervised Learning of Visualisation Hierarchies

The procedure for unsupervised learning of mixture models discussed in section IV becomes more complex for nodes (subplots) in hierarchical models at levels > 2. In this case, we should

[8]we used LTMs with Gaussian noise model





(a) (b)

Fig. 2. (a) A two dimensional manifolds in data space; (b) Projection manifolds in data space of the second-level LTMs trained on the toy data.

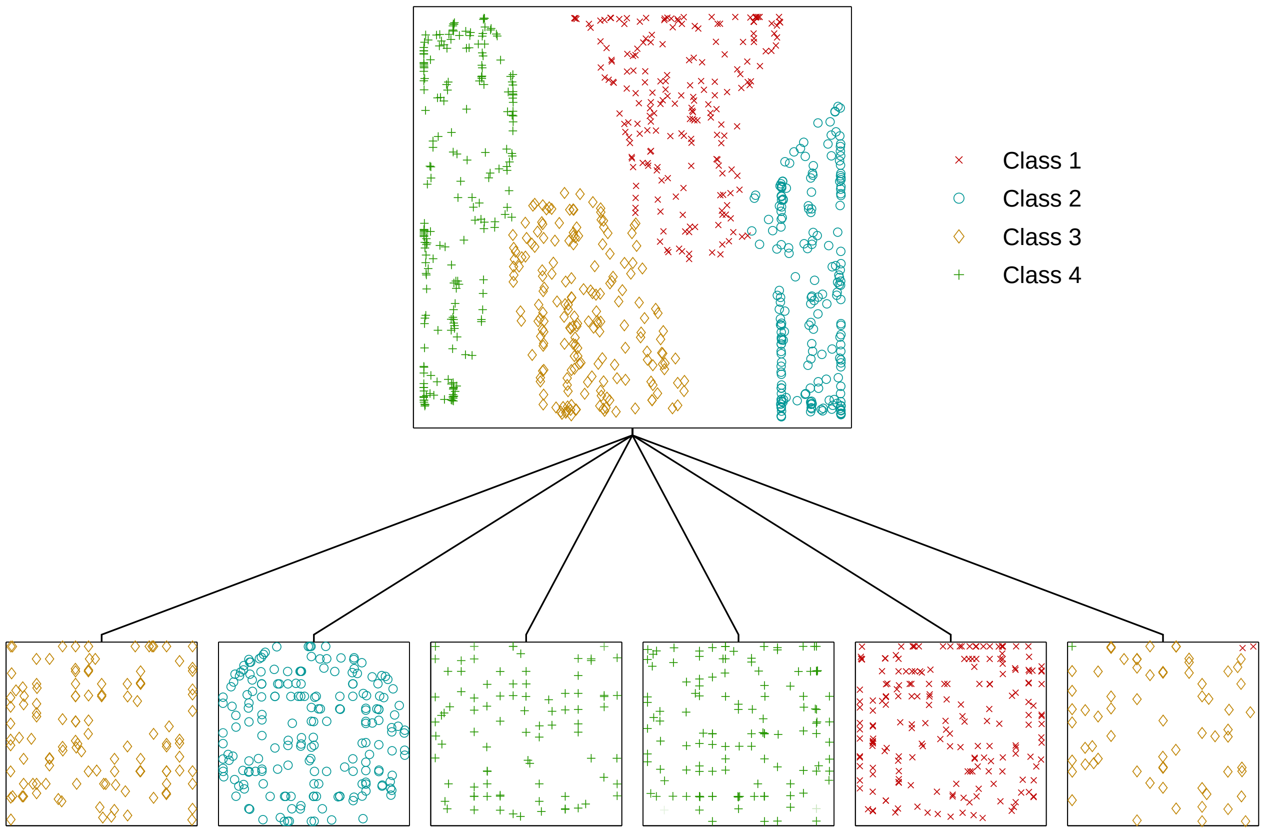


Fig. 3. Visualisation of the toy data constructed with unsupervised MML.

consider model responsibilities of parent nodes for the data points and these are recursively propagated as we incrementally build the hierarchy. So equations (9) and (11) are used in hierarchical models instead of (18) used in the simple mixture case. Also (6) is applied in place of (19).

The proposed system for constructing hierarchies of non-linear visualisation plots is similar to the one described in [27]. The important difference is that now, given a parent plot, its children are not always constructed in the interactive way by letting the user identify "regions of interest" for the sub-plots. In densely populated higher-level plots with many overlapping projections, this may not be possible. Instead, we let the user decide whether they want the children to be constructed in an interactive or unsupervised way.

In the unsupervised case, we use the MML technique to decide an "appropriate" number and approximate position of children LTMs. We collect data points from ζ for which the parent LTM has responsibility higher than a threshold Δ (in our experiments Δ was set to[9] 0.9). We then run MML-based learning of *mixtures* of LTMs (section IV-B) on this reduced data set. The resulting local mixture is viewed as an *initialization* for the full EM algorithm for training *hierarchies* of LTMs described in section III-A. This way, an "appropriate" number of LTMs is determined along with their initial locations.

It should be noted, that by using Jeffrey's prior, the approach suggested in [11] implies an improper Dirichlet prior (over the mixing coefficients) with negative parameters. As pointed out in [31], the use of the non-informative Jeffrey's prior in general raises problems from the Bayesian point of view. For instance, improper priors may lead to inadmissible estimates [26]. However, such priors have been extensively used mainly due to mathematical convenience: we do not have to compute the Fisher information matrix (typically a computationally expensive step). Nevertheless, as we will demonstrate in the next subsection, we have experimentally found that for mixtures of LTMs the use of Jeffrey's prior leads to sufficiently good initial estimates to be fed to the hierarchical EM described in section III-A. Moreover, the issue is less critical since the MML-based model selection is used solely to initialize the child models at higher levels of the hierarchy, while typically the user himself will refine the plots at lower levels of the hierarchy

[9]Other values for Δ, e.g. $\Delta = 0.8$, could have been used. However, the final local mixture of LTMs in the hierarchy is not very sensitive to the exact value of Δ, since this is just an initialization step, before running full EM for hierarchical LTM.

as in [27]. The Jeffrey's prior over the mixing coefficients favours strong component pruning, which is beneficial for our purposes.

A. Experiments

In this section we illustrate the semi-supervised hierarchical LTM visualisation algorithm on three "real-world" data collections.

Although the algorithm is derived in a general setting in which individual LTMs $\mathcal{M}$ in the hierarchy can have different sets of latent points $\boldsymbol{x}_k^{\mathcal{M}}$, $k = 1, 2, ..., K_{\mathcal{M}}$, and basis functions ϕ_j, $j = 1, 2, ..., M_{\mathcal{M}}$, in the experiments reported here, we used a common configuration for all models in the hierarchy. In particular, the latent space $\mathcal{H}$ was taken to be the two-dimensional interval $\mathcal{H} = [-1, 1] \times [-1, 1]$, the latent points $\boldsymbol{x}_k^{\mathcal{M}} \in \mathcal{H}$ were positioned on a regular 15×15 square grid and there were 16 radial basis functions ϕ_j centered on a regular 4×4 square grid. As usual in the GTM literature, the basis functions were spherical Gaussians of the same width[10] $\sigma = 1.0$. We account for a bias term by using an additional constant basis function $\phi_0(\boldsymbol{x}) = 1$, for all $\boldsymbol{x} \in \mathcal{H}$. If the noise model in LTM is Gaussian, we always consider only spherical Gaussians, as in the original formulation of GTM [4]. Complete training equations for hierarchical GTM can be found in [27].

Note that in the interactive mode, the "centres" of the regions of interest are shown as circles labeled by numbers. These numbers determine the order of the corresponding child LTM subplots from left to right.

1) Image segmentation data: As the first example we visualize image segmentation data obtained by randomly sampling patches of 3x3 pixels from a database of outdoor images. The patches are characterized by 18 continuous attributes and are classified into 4 classes: *cement + path*, *brickface + window*, *grass + foliage* and *sky* (see [27]). The final visualisation plot of hierarchical LTM with Gaussian noise model can be seen in Figure 4. The *Root* plot contains clusters of overlapping projections. Six plots at the second level were constructed using the unsupervised MML technique ($A_{max} = 10$). Note that the second-level LTMs already separate the four classes fairly well and are interpretable enough to be analysed further in the interactive

[10]The width of the basis functions is related to the "flexibility" of the generalized linear regression, $\boldsymbol{f}_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \boldsymbol{\Theta}\ \ (\boldsymbol{x})$, from the latent space to the data space. For a discussion on appropriate values for σ see [4], [27].

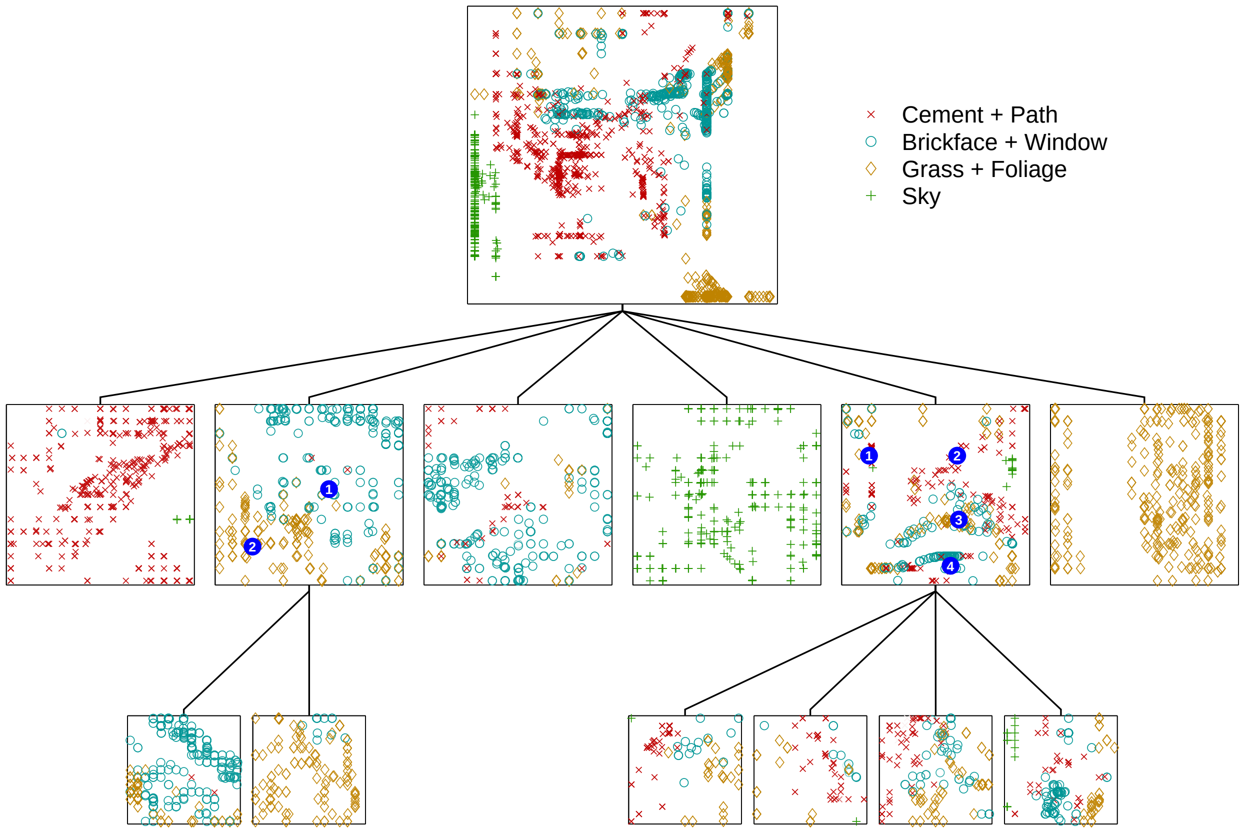


Fig. 4. Hierarchical visualisation of the image segmentation data constructed in a semi-interactive way. Numbered circles represent user-selected locations for sub-models.

mode. For example, we selected two and four “centres” for regions of interest (shown as circles) in the second and fifth level-two plots, respectively.

2) Text data set: Since our system is based on the LTM, it can deal with discrete data sets. As an illustration, we tested our system on a text-collection of 8000 documents formed by 10 topic classes from a newsgroup[11] text corpus. The documents were binary encoded over a dictionary of $D = 100$ words. The initial pre-processing, word-stemming and removal of “stop-words” was done using the Bow toolkit[12]. To match the binary encoding, a Bernoulli noise model was employed, as a Gaussian would be inappropriate. Hence a hierarchy of LTMs was used instead of HGTM.

The visualisation plot generated in a semi-interactive way is shown in Figure 5. The *Root*

[11]http://www.cs.cmu.edu/˜textlearning

[12]http://www-2.cs.cmu.edu/˜mccalum/bow

is extremely densely populated with highly overlapping data projections. After using the unsupervised MML technique ($A_{max} = 10$), a 4-component mixture of LTMs was obtained on the second level. Sub-clusters in these four level-two plots are decipherable. The user can then choose more detailed regions of interest by using the interactive mode. This is illustrated in the Figure, but for complete class separation, more plots would be required.

As in [27], this system also includes the child-modulated ancestor plot technique, which can visualise the regions captured by a particular child LTM $\mathcal{M}$. This is done by modifying all the ancestor plots up to the *Root*, so that instead of the ancestor responsibilities, the responsibilities of the model $\mathcal{M}$, $P(\mathcal{M}|\mathbf{t}_n)$, are used in every plot on the path from $\mathcal{M}$ to *Root*. This improves the understanding of the relationships among sub-plots in the visualisation hierarchy. In Figure 6, we highlight the visualisation plots which include the data points from the topic "sci.space", captured by the first model on the 4th-level.

3) Protein localization site data set: In the last experiment we visualise a data set of 1484 proteins encoded as real-valued vectors[13]. The 6-dimensional data points[14] are classified into 10 classes (localization sites). The class names are shown in the legend of Figure 7. Here we demonstrate the application of the unsupervised MML technique at a lower level in the hierarchy.

We trained a four-level hierarchy of LTMs (Gaussian noise model) on the protein data and the resulting projections are displayed in Figure 7. Again, the *Root* plot looks cluttered. Two plots at the second level were constructed using the unsupervised MML technique ($A_{max} = 10$). The first level-two plot is legible enough for the user to select the 'centres' in the interactive mode (as shown in the figure). We used the MML algorithm as an initialisation technique for constructing child plots of the second level-two plot ($A_{max} = 5$). Two resulting child plots included readable clusters.

Note that visualisation plots for this dataset do not provide a good separation, even at lower levels of the hierarchy. It follows that the features used to describe the data are not very discriminative with respect to the 10 binding site classes and the classes are highly overlapping. Our system enables the user to detect such situations by understanding the underlying data

[13]The data set can be downloaded from the UCI Machine Learning page: ftp://ftp.ics.uci.edu/pub/machine-learning-databases/yeast/

[14]The original data is 8-dimensional. Two of the dimensions are effectively constant and were removed.

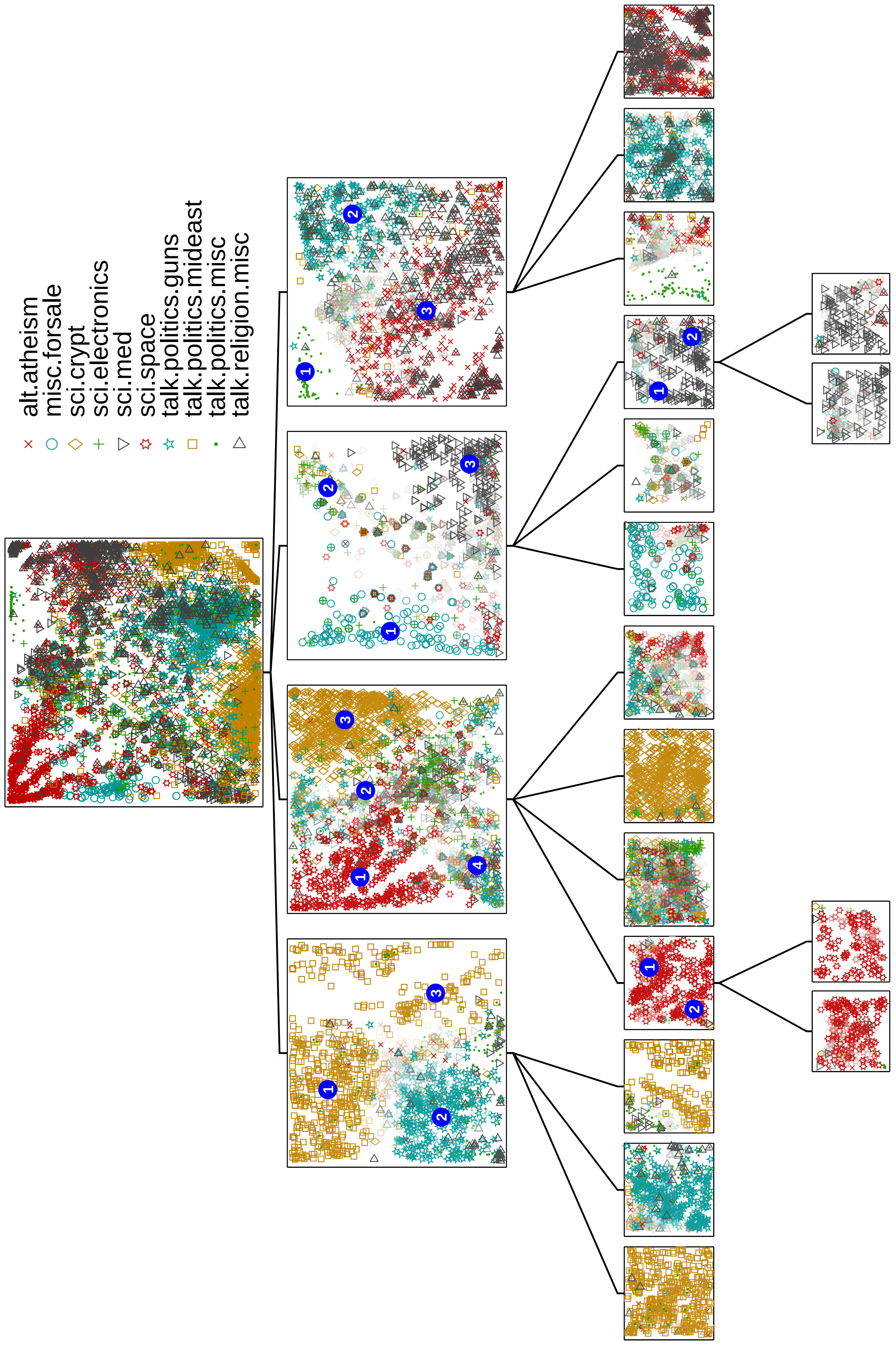


Fig. 5. Hierarchical visualisation of the document data constructed in a semi-interactive way.

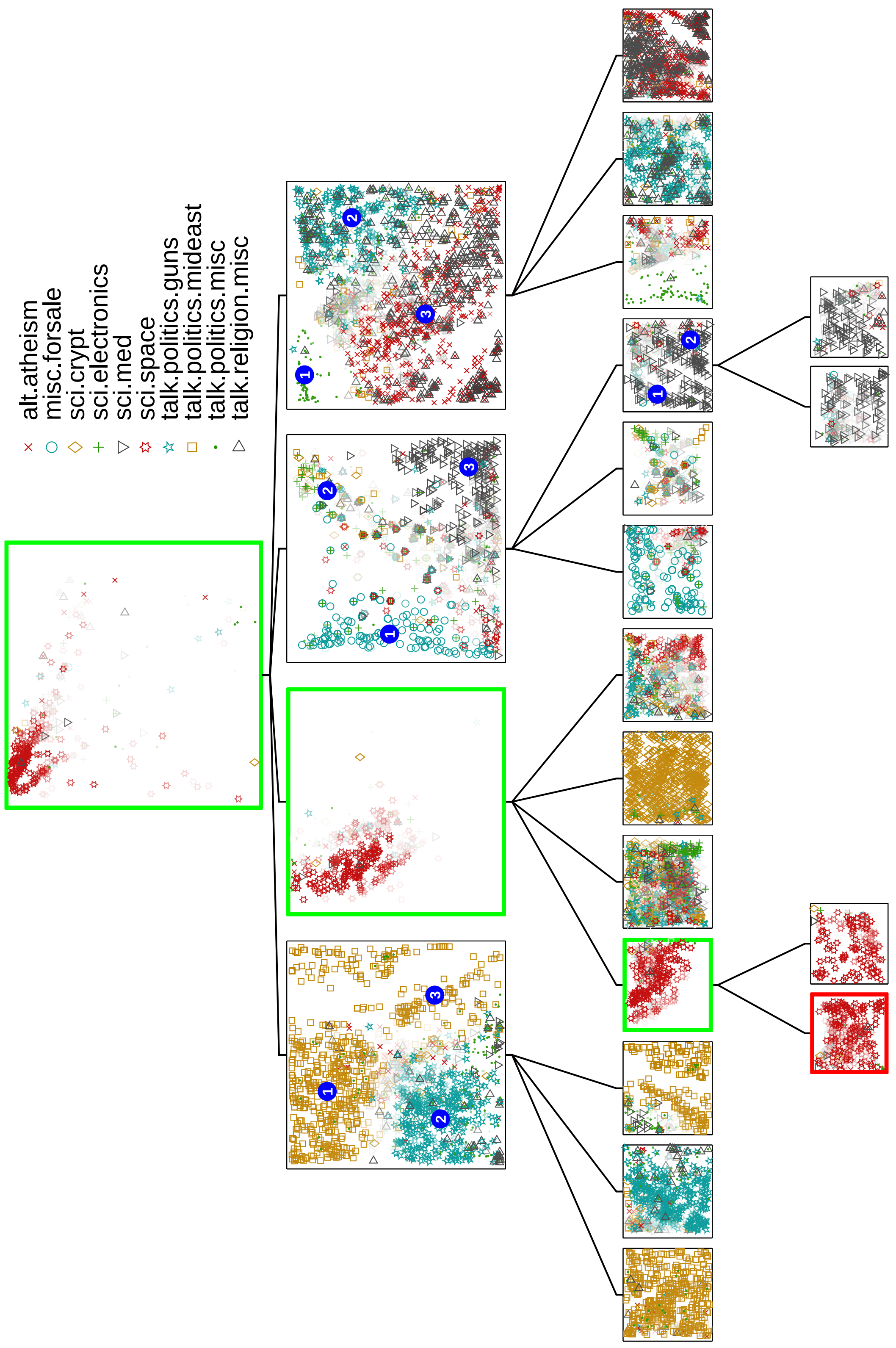


Fig. 6. Hierarchical visualisation of the document data constructed in a semi-interactive way. The set of points captured by the first LTM at level 4 of the hierarchy is highlighted in the visualisation plots of all its ancestors.

distribution. Our findings are confirmed by the poor classification results (around 55% accuracy) obtained on this data set using various classification techniques [12].

B. *Comparison*

Although the primary focus of this paper is on automating the development of hierarchical models, it is also useful to compare our results with another hierarchical visualisation technique. Interest in visualisation of large multivariate datasets has been growing recently; as well as the generative approach taken by [7], [27] and this paper, more heuristic methods have also been developed [36], [19], [16]. We have selected the first of these, Interactive Hierarchical Displays (IHDs), as a benchmark for two main reasons: it is recent work that unifies several features of earlier techniques and it is the most closely related to our own in several respects. It also has the advantage that an implementation is publically available[15].

Like HGTM, IHDs are designed to tackle the clutter problem faced by traditional multivariate visualisation techniques when analysing large datasets. The key strategy is to put fewer items on the screen. This is achieved by first constructing a hierarchical cluster tree. The tree can then be visualised at different levels of detail; the user specifies the point at which the tree is cut. Rather than showing all the datapoints, each cluster is displayed. The cluster is summarised by its mean with a band around it giving the minimum and maximum values in each variable of the cluster. This can be depicted using any multivariate visualisation technique that shows all the variables: [36] uses parallel coordinates [13], [34], star glyphs [24], scatterplot matrices and dimensional stacking [20]. A band is assigned the colour of the cluster it represents. The strategy for this, called *proximity-based colouring* maps colours by cluster proximity based on the structure of the hierarchical tree. The strategy has the following properties:

- sibling clusters have similar colours;
- a parent cluster has a colour within the range of its children's colours.

To create this map, it is necessary to impose a linear ordering on all the clusters.

Figure 8 shows the result produced from hierarchical parallel coordinates when applied to the image segmentation dataset. In current level, 5 clusters are captured. The mean points of individual clusters are mapped to a polyline across all the dimensions with a bind indicating the

[15]`http://davis.wpi.edu/∼xmdv`

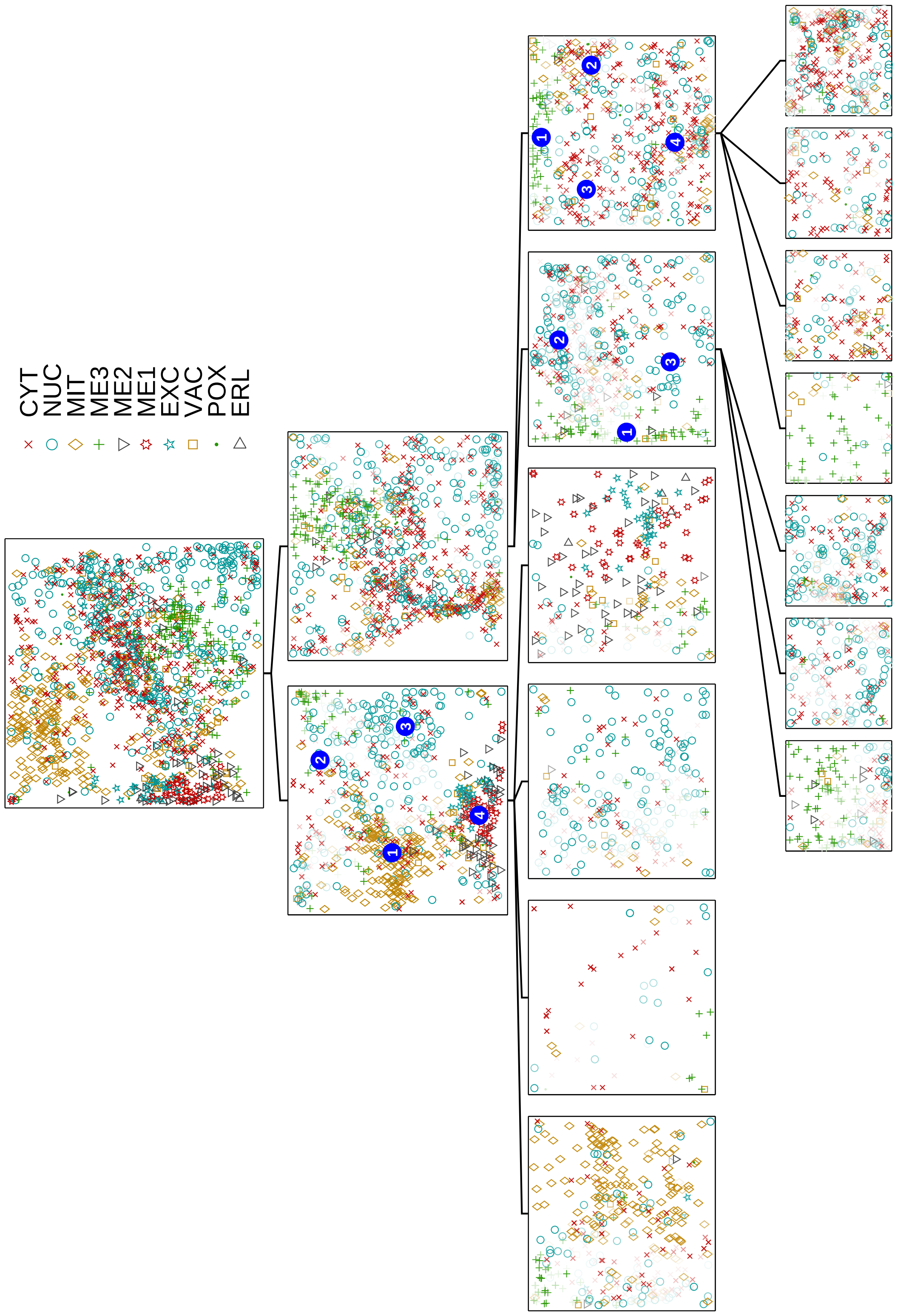


Fig. 7. Hierarchical visualisation of the protein data set constructed in a semi-interactive way.

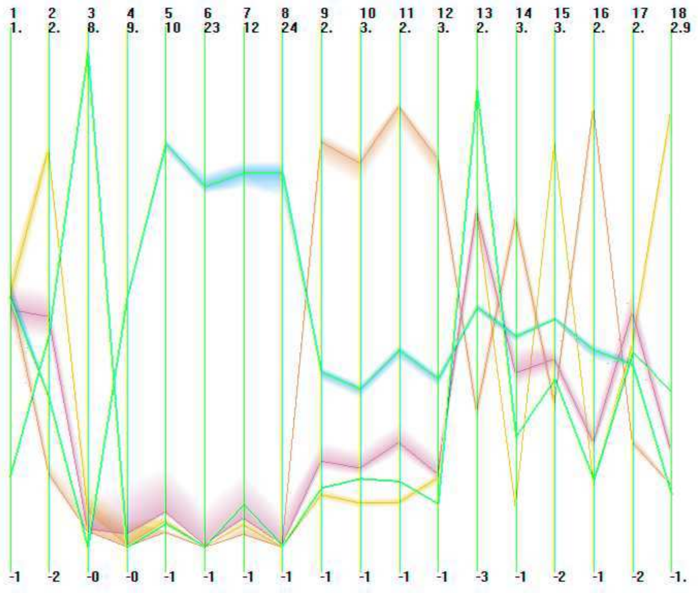


Fig. 8. Hierarchical parallel coordinates visualisation plot using the image dataset.

range of each cluster. This graph shows that it is difficult to see the shape of the clusters when compared with the results from HGTM (see Figure 4). Figure 9 displays a hierarchical glyphs, where the mean values are used to generate the basic shape. Although it suggests the shape of each cluster, it is not clear which data points belong to it. In figure 10, a hierarchical scatterplot matrix is presented. Again, the points shown in the figure are the mean points of individual clusters. It is not clear which clusters are significant.

C. Discussion

IHDs are a useful means of visualising hierarchical clusters. In [36] controlled experiments showed that most users could find more structure in datasets using IHDs rather than a single 'flat' plot. They are a generic approach in that they can be used with any hierarchical tree clustering algorithm and any multivariate visualisation that uses all the original variables.

In contrast, our aim is more general: we want to represent the whole dataset in two dimensions without loss of information. Only when this is not possible do we split the plot up. Consequently, the hierarchical trees that HGTM generates are usually much shallower and simpler than those produced by other hierarchical clustering methods. This allows the user to see the *whole* dataset

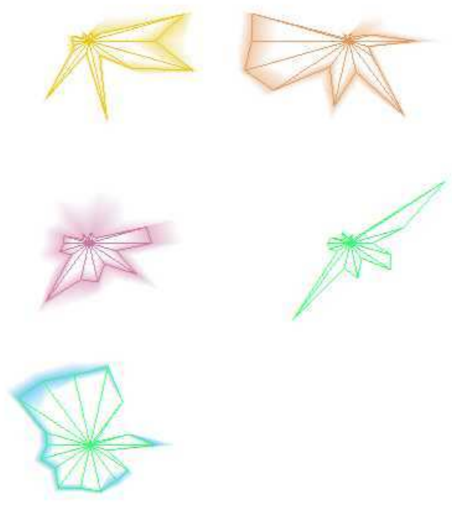


Fig. 9. Hierarchical star glyphs visualisation plot using the image dataset.

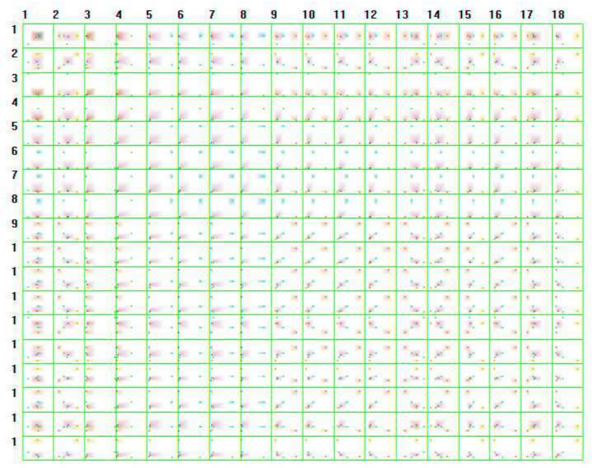


Fig. 10. Hierarchical scatterplot matrix visualisaion plot using the image dataset.

on a few plots which means they are less likely to get lost in multiple plots, but can still see the global picture

Standard hierarchical clustering algorithms tend to perform poorly when there is a lot of noise in the data or when, as is often the case, the data is not split into well-separated clusters. In addition, they are usually based on heuristic distance measures. HGTM trees are a powerful method of clustering data and do not suffer from these disadvantages. In particular, they provide a *probabilistic density model* for the data, which brings many benefits (including principled automation of structure selection using Bayesian methods, as demonstrated in this paper). The fact that all the data is shown is also helpful; it allows the users to drill down into different regions and find out more about the *data* as well as the *clusters*. Users have expressed some concern that the coordinate system in the plots does not correspond with any meaningful variables. However, this drawback has been overcome by allowing them to specify regions where they can view the data locally using parallel coordinates.

Another important benefit of HGTM is that it projects the data onto a lower dimensional space, which makes the plots much easier to interpret. HGTM has been applied to drug discovery data with more than 30 variables; at this size, multivariate visualisation techniques like glyphs are very hard for users to understand. This two-dimensional representation also captures the relationships between clusters (which is very important to develop real understanding of the data); IHDs use a one-dimensional representation of inter-cluster relationships (the proximity-based colouring) that is necessarily less rich in expressive power.

IHDs are a display technique, so the only time consuming aspect is the hierarchical clustering algorithm. HGTM has an efficient EM algorithm: it takes 2–3 minutes to train a model for a dataset of 1000 examples and c. 15 variables on a ‘standard’ 1GHz PC. The automated initialisation algorithm takes somewhat longer since there is a need to train models of several different structures at each level: the image segmentation and protein datasets required in the order of 10 minutes to train. The text dataset, with 8000 examples and 100 variables, takes rather longer: in the order of 2 hours. It is worth noting that these times are based on a program written using the MATLAB mathematical toolkit; an implementation in a 3GL such as C would normally be around twice as fast. Once the model is trained the user can interact with the plot with no time delays.

VI. Local Magnification Factors of the Latent Trait Manifolds

In this section we first briefly review the notion of local magnification factors for the original GTM [6]. We then re-derive the formula for computing magnification factors for the more general LTM.

The term “magnification factor” [6] refers to the degree of stretching or compression of the latent space when embedded into the data space. Previous experience has indicated that magnification factors are a useful tool for interpreting 2-D non-linear visualisation plots. For example, projections of well-separated dense clusters of data points will occupy compressed regions on the visualisation plot (small magnification factors), separated by a band of highly stretched area (high magnification factors).

Let us consider the Cartesian coordinate system defined on the latent space and the mapping of this space to a curvilinear coordinate system defined on the manifold embedded in the data space. The local magnification factor corresponding to a point $\boldsymbol{x}_0$ in the latent space can be defined as the ratio between the area of an infinitesimal rectangle in the latent Cartesian space and the area generated by mapping it through (3) on the projection manifold. This ratio is equal to $\sqrt{|\boldsymbol{S}(\boldsymbol{x}_0)|}$, where $|\boldsymbol{S}(\boldsymbol{x}_0)|$ is the determinant of the metric tensor $\boldsymbol{S} = \boldsymbol{\Gamma}^T\boldsymbol{\Gamma}$, where $\boldsymbol{\Gamma}$ denotes the Jacobian of the mapping (3) at $\boldsymbol{x}_0$. For GTM, since $\boldsymbol{b}(.)$ is identity function, $\boldsymbol{\Gamma}$ evaluates as $\boldsymbol{\Theta V}$, where $\boldsymbol{V}$ is the $M \times L$ matrix $\left(\frac{\partial \phi_m(\boldsymbol{x})}{\partial x_l}_{|\boldsymbol{x}=\boldsymbol{x}_0}\right)_{m=1,\dots,M,l=1,\dots,L}$.

For the Latent Trait Models, we have

$$\boldsymbol{\Gamma} = \frac{\partial \boldsymbol{z}(\boldsymbol{x}_0)}{\partial \boldsymbol{x}} = \frac{\partial \boldsymbol{b}(\boldsymbol{\Theta}\boldsymbol{\phi}(\boldsymbol{x}_0))}{\partial \boldsymbol{x}} = \boldsymbol{F\Theta V}, \tag{20}$$

where the $D \times D$ matrix $\boldsymbol{F} = \left(\frac{b_{d'}(\boldsymbol{y})}{\partial y_d}_{|\boldsymbol{y}=\boldsymbol{\Theta}\boldsymbol{\phi}(\boldsymbol{x}_0)}\right)_{d'=1,\dots,D,d=1,\dots,D}$ is the Fisher information matrix of the noise distribution. Indeed, if the noise model is Gaussian, $\boldsymbol{F}$ turns out to be the identity matrix. With the choice of RBF nonlinearities for $\phi(\cdot)$, the (l,m)-th element of the matrix $\boldsymbol{V}$ is $v_{l,m} = -\phi_m(\boldsymbol{x}_0)(x_l - c_{m,l})\sigma^{-2}$, where $c_{m,l}$ denotes the l-th coordinate of the radial basis center corresponding to the m-th basis function and σ is the width of the RBF functions.

In summary, the magnification factor associated with a latent space point $\boldsymbol{x}_0$ is

$$\sqrt{|\boldsymbol{V}^T\boldsymbol{\Theta}^T\boldsymbol{F}^T\boldsymbol{F\Theta V}|}. \tag{21}$$

It is easy to see, that this formula differs from the one derived in [6] for the original GTM in the presence of the matrix $\boldsymbol{F}^T\boldsymbol{F}$, which reduces to identity in the case of Gaussian noise. So the formula for computing magnification factors for GTM derived in [6] is recovered in the special case of (21), when the noise is Gaussian.

Note also that in all independent noise models this matrix will be diagonal; therefore the increase in computational complexity will not be significant. However, this is not the case for the multinomial trait model (as can be seen in appendix I-C).

As an example we show in Figure 11 the magnification factor plots (log scaled) for the projection hierarchy of the text data set in Figure 5. In general, dark bands in the plots indicate well-separated clusters of points in the data space. For example, there is a dark band slightly left of the center of the eleventh level-three model. The band divides different topics in the data space. From the corresponding model in Figure 5, we see that the left region mostly involves topic “talk.politics.misc”, and the right region contains a mixture of topics.

As an example of detailed analysis of magnification factors, we focus on the fourth level-three LTM model in Figure 11. The corresponding projection plot in Figure 5 contains mostly documents from a single topic, “sci.space”. An enlarged (locally scaled) view of the magnification factor plot is presented in Figure 12. There is a dark band around the diagonal line of the plot. Hence, we infer that documents on either side of the band correspond to different clusters and that a change of *sub-topic* occurs. The list of 5 most probable dictionary words for each latent space centre of the corresponding LTM is shown in Figure 13. With reference to Figure 12, two clusters can be found on each side of the separating band. Key words for each latent space centre inside the region bounded by the solid border are completely the same and have the same ordering. They appear to refer to documents relating to space shuttle launches, while key words inside the region with the dashed border seem to be associated with articles concerning space orbits.

VII. Conclusion

In this paper we have presented a general system for hierarchical visualisation of large data sets which may be of either continuous or discrete type. We also derived formulas for magnification factors in latent trait models. The proposed system gives the user a choice of initializing the child plots of the current plot in either *interactive*, or *automatic* mode. This latter feature is particularly useful when the user has no idea how to choose the area of interest due to highly overlapping dense data projections. We have evaluated this system on three real world datasets and compared

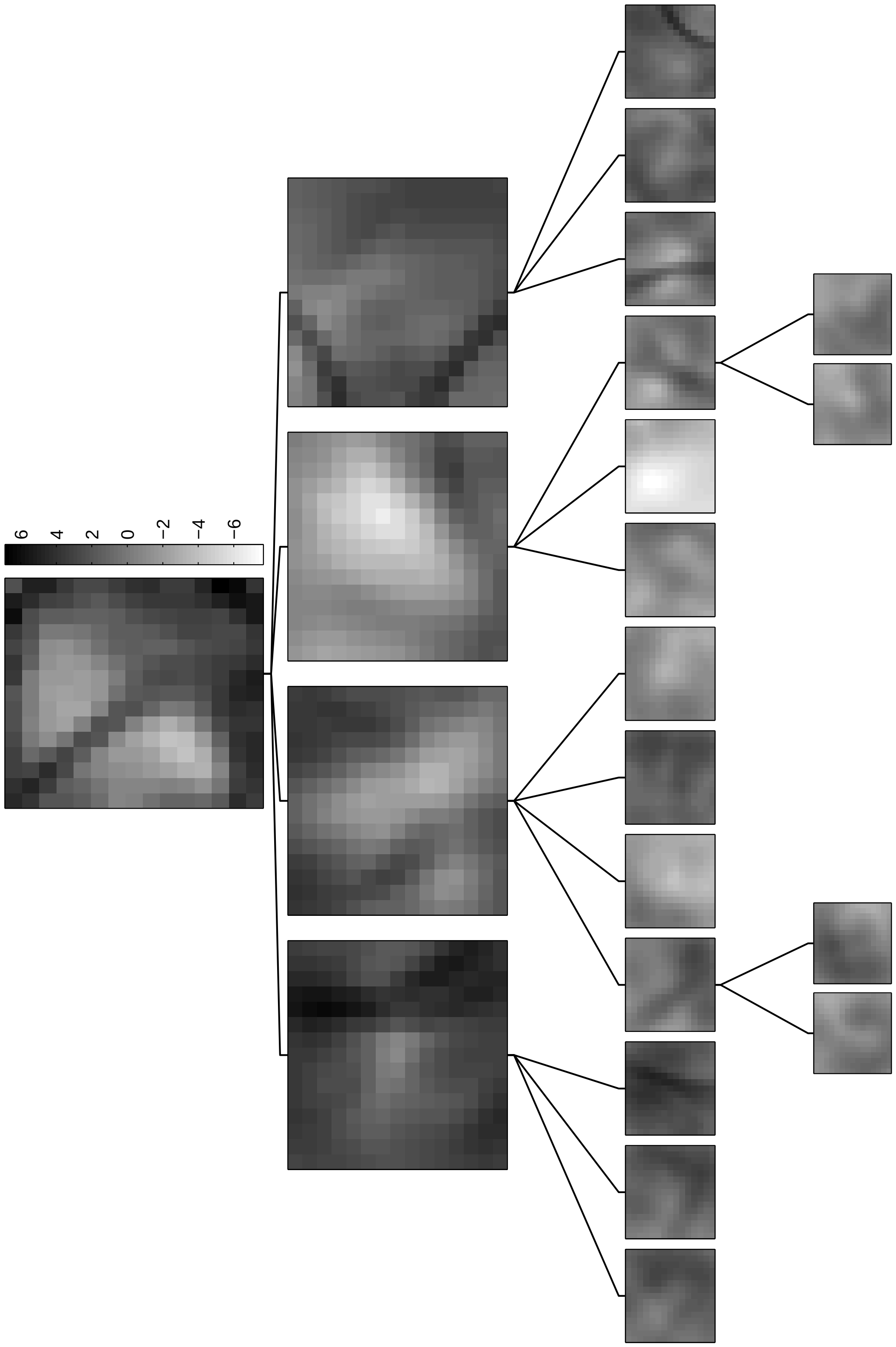


Fig. 11. Plots of magnification factors (log2 scaled) in the hierarchy of LTMs fitted on the document data.

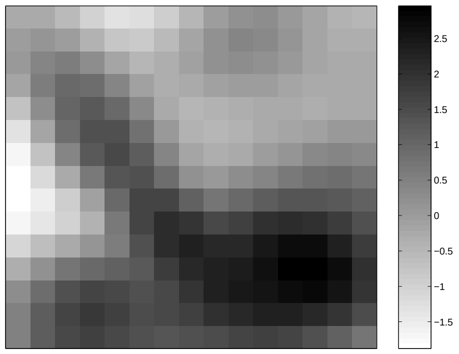


Fig. 12. A rescaled visualisation plot of magnification factors for the 4th LTM at level 3 in the hierarchy shown in figure 11.

the results with an existing method for hierarchical visualisation. In many problems, particularly where there are not clearly defined and separated clusters of data, hierarchical LTMs offer significant benefits.

The system can be used in many different fields, such as document data mining, tele-communications, bio-informatics, market-basket analysis or information retrieval. We are currently developing this system further to provide more user feedback during the data exploration process and to combine visualisation with localised modelling (for example, to predict properties of chemical compounds).

APPENDIX I

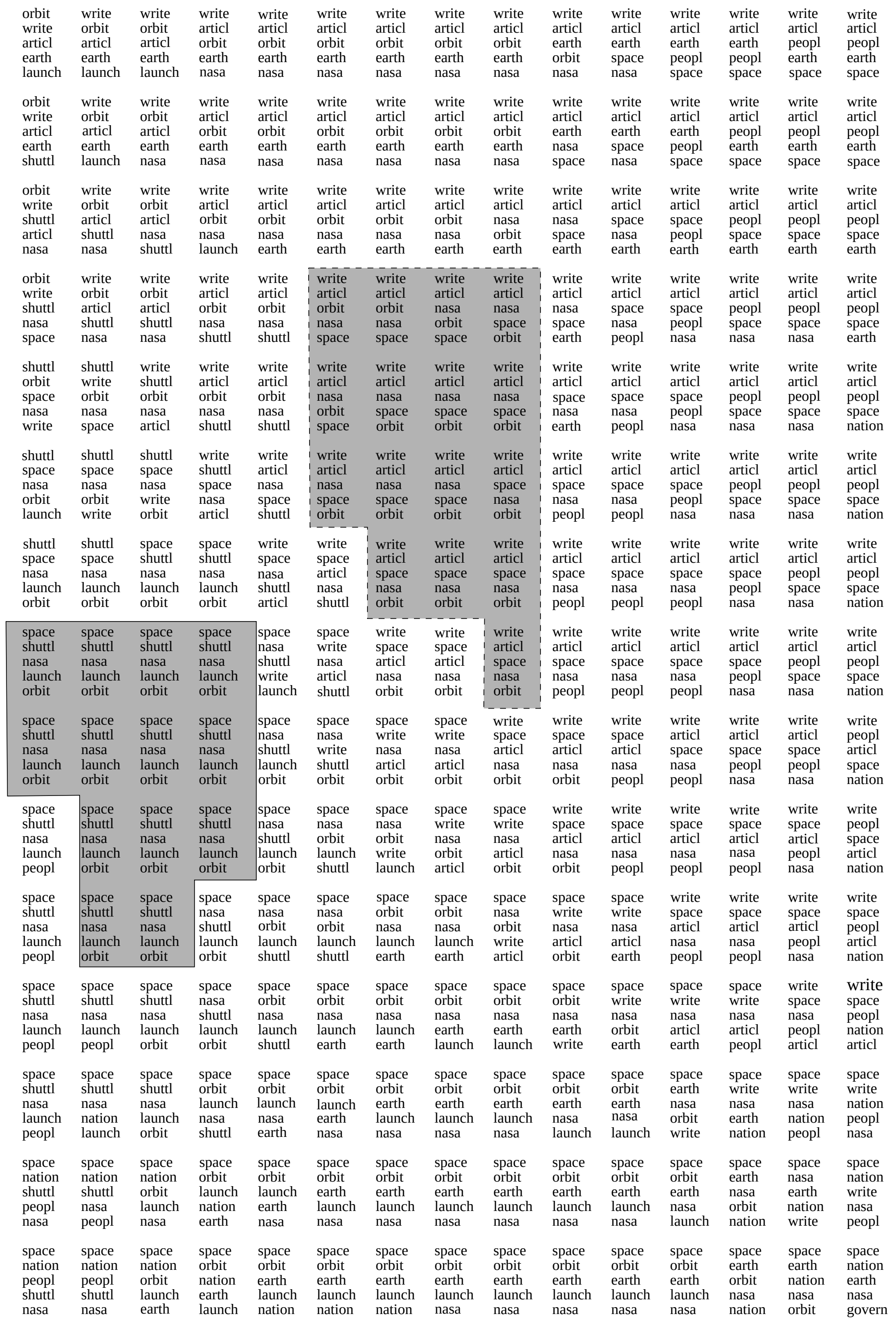
QUANTITIES REQUIRED FOR COMPUTING MAGNIFICATION FACTORS IN THE REPORTED EXPERIMENTAL SETTINGS

The exact form of the matrices $\boldsymbol{F}$ is dependent on the specific noise-model being employed. These quantities require the computation of the first derivatives of the inverse link function $b()$. In this appendix we will provide the expressions for those members of the exponential model family which have been employed in the reported experimental settings.

A. Independent Gaussian noise model

The Gaussian model is the only member of the exponential family of distributions which is characterised by a quadratic cumulant function

$$B_t(\boldsymbol{y}) = \frac{1}{2} y_t^2. \tag{22}$$



orbit write articl earth launch	write orbit articl earth launch	write orbit articl earth launch	write articl orbit earth nasa	write articl orbit earth nasa	write articl orbit earth nasa	write articl orbit earth nasa	write articl orbit earth nasa	write articl orbit earth nasa	write articl earth orbit nasa	write articl earth space nasa	write articl earth peopl space	write articl earth peopl space	write articl peopl earth space	write articl peopl earth space
orbit write articl earth shuttl	write orbit articl earth launch	write orbit articl earth nasa	write articl orbit earth nasa	write articl orbit earth nasa	write articl orbit earth nasa	write articl orbit earth nasa	write articl orbit earth nasa	write articl orbit earth nasa	write articl earth nasa space	write articl earth space nasa	write articl earth peopl space	write articl peopl earth space	write articl peopl earth space	write articl peopl earth space
orbit write shuttl articl nasa	write orbit articl shuttl nasa	write orbit articl nasa shuttl	write articl orbit nasa launch	write articl orbit nasa earth	write articl orbit nasa earth	write articl orbit nasa earth	write articl orbit nasa earth	write articl nasa orbit earth	write articl nasa space earth	write articl space nasa earth	write articl space peopl earth	write articl peopl space earth	write articl peopl space earth	write articl peopl space earth
orbit write shuttl nasa space	write orbit articl shuttl nasa	write orbit articl shuttl nasa	write articl orbit nasa shuttl	write articl orbit nasa shuttl	write articl orbit nasa space	write articl orbit nasa space	write articl nasa orbit space	write articl nasa space orbit	write articl nasa space earth	write articl space nasa peopl	write articl space peopl nasa	write articl peopl space nasa	write articl peopl space nasa	write articl peopl space earth
shuttl orbit space nasa write	shuttl write orbit nasa space	write shuttl orbit nasa articl	write articl orbit nasa shuttl	write articl orbit nasa shuttl	write articl nasa orbit space	write articl nasa space orbit	write articl nasa space orbit	write articl nasa space orbit	write articl space nasa earth	write articl space nasa peopl	write articl space peopl nasa	write articl peopl space nasa	write articl peopl space nasa	write articl peopl space nation
shuttl space nasa orbit launch	shuttl space nasa orbit write	shuttl space nasa write orbit	write shuttl space nasa articl	write articl nasa space shuttl	write articl nasa space orbit	write articl nasa space orbit	write articl nasa space orbit	write articl space nasa orbit	write articl space nasa peopl	write articl space nasa peopl	write articl space peopl nasa	write articl peopl space nasa	write articl peopl space nasa	write articl peopl space nation
shuttl space nasa launch orbit	shuttl space nasa launch orbit	space shuttl nasa launch orbit	space shuttl nasa launch orbit	write space nasa shuttl articl	write space articl nasa shuttl	write articl space nasa orbit	write articl space nasa orbit	write articl space nasa orbit	write articl space nasa peopl	write articl space nasa peopl	write articl space nasa peopl	write articl space peopl nasa	write articl peopl space nasa	write articl peopl space nation
space shuttl nasa launch orbit	space shuttl nasa launch orbit	space shuttl nasa launch orbit	space shuttl nasa launch orbit	space nasa shuttl write launch	space write nasa articl shuttl	write space articl nasa orbit	write space articl nasa orbit	write articl space nasa orbit	write articl space nasa peopl	write articl space nasa peopl	write articl space nasa peopl	write articl space peopl nasa	write articl peopl space nasa	write articl peopl space nation
space shuttl nasa launch orbit	space shuttl nasa launch orbit	space shuttl nasa launch orbit	space shuttl nasa launch orbit	space nasa shuttl launch orbit	space nasa write shuttl orbit	space write nasa articl orbit	space write nasa articl orbit	write space articl nasa orbit	write space articl nasa orbit	write space articl nasa peopl	write articl space nasa peopl	write articl space peopl nasa	write articl space peopl nasa	write peopl articl space nation
space shuttl nasa launch peopl	space shuttl nasa launch orbit	space shuttl nasa launch orbit	space shuttl nasa launch orbit	space nasa shuttl launch orbit	space nasa orbit launch shuttl	space nasa orbit write launch	space write nasa orbit articl	space write nasa articl orbit	write space articl nasa orbit	write space articl nasa peopl	write space articl nasa peopl	write space articl nasa peopl	write space articl peopl nasa	write peopl space articl nation
space shuttl nasa launch peopl	space shuttl nasa launch orbit	space shuttl nasa launch orbit	space nasa shuttl launch orbit	space nasa orbit launch shuttl	space nasa orbit launch shuttl	space orbit nasa launch earth	space orbit nasa launch earth	space nasa orbit write articl	space write nasa articl orbit	space write nasa articl earth	write space articl nasa peopl	write space articl nasa peopl	write space articl peopl nasa	write space peopl articl nation
space shuttl nasa launch peopl	space shuttl nasa launch peopl	space shuttl nasa launch orbit	space nasa shuttl launch orbit	space orbit nasa launch shuttl	space orbit nasa launch earth	space orbit nasa launch earth	space orbit nasa earth launch	space orbit nasa earth launch	space orbit nasa earth write	space write nasa orbit earth	space write nasa articl earth	space write nasa articl peopl	write space nasa peopl articl	write space peopl nation articl
space shuttl nasa launch peopl	space shuttl nasa nation launch	space shuttl nasa launch orbit	space orbit launch nasa shuttl	space orbit launch nasa earth	space orbit launch earth nasa	space orbit earth launch nasa	space orbit earth launch nasa	space orbit earth launch nasa	space orbit earth nasa launch	space orbit earth nasa launch	space earth nasa orbit write	space write nasa earth nation	space write nasa nation peopl	space write nation peopl nasa
space nation shuttl peopl nasa	space nation shuttl nasa peopl	space nation orbit launch nasa	space orbit launch nation earth	space orbit launch earth nasa	space orbit earth launch nasa	space orbit earth launch nasa	space orbit earth launch nasa	space orbit earth launch nasa	space orbit earth launch nasa	space orbit earth launch nasa	space orbit earth nasa launch	space earth nasa orbit nation	space nasa earth nation write	space nation write nasa peopl
space nation peopl shuttl nasa	space nation peopl shuttl nasa	space nation orbit launch earth	space orbit nation earth launch	space orbit earth launch nation	space orbit earth launch nation	space orbit earth launch nation	space orbit earth launch nasa	space orbit earth launch nasa	space orbit earth launch nasa	space orbit earth launch nasa	space orbit earth launch nasa	space earth orbit nasa nation	space earth nation nasa orbit	space nation earth nasa govern

Fig. 13. The most probable words formed in each of the 15 by 15 latent grid points by the Bernoulli latent trait model obtained in the experiments on text documents data.

Therefore, it has a linear inverse-link function and higher derivatives vanish.

$$b_{t'}(\boldsymbol{y}) = y_{t'}, \tag{23}$$

$$\frac{\partial b_{t'}(\boldsymbol{y})}{\partial y_t} = 0. \tag{24}$$

B. Independent Bernoulli noise model

In the case of the Bernoulli model, the cumulant function has the following form:

$$B_t(\boldsymbol{y}) = \log(1 + exp(y_t)). \tag{25}$$

The required derivatives are then computed as follows:

$$b_{t'}(\boldsymbol{y}) = \frac{\exp(y_{t'})}{1 + \exp(y_{t'})}, \tag{26}$$

$$\frac{\partial b_{t'}(\boldsymbol{y})}{\partial y_t} = \begin{cases} 0 & t \neq t' \\ b_t(\boldsymbol{y})(1 - b_t(\boldsymbol{y})) & t = t'. \end{cases} \tag{27}$$

It can be seen that for independent noise models, the Fisher information matrix $\boldsymbol{F}$ is diagonal.

C. Multinomial noise model

The multinomial distribution is identified by the following cumulant function:

$$B(\boldsymbol{y}) = \log\left(\sum_{t=1:T} exp(y_t)\right). \tag{28}$$

Accordingly, the derivatives are given by

$$b_{t'}(\boldsymbol{y}) = \frac{\exp(y_{t'})}{\sum_{t''=1}^{T} \exp(y_{t''})}, \tag{29}$$

$$\frac{\partial b_{t'}(\boldsymbol{y})}{\partial y_t} = \begin{cases} -b_{t'}(\boldsymbol{y}) b_t(\boldsymbol{y}) & t \neq t' \\ b_{t'}(\boldsymbol{y}) - b_{t'}(\boldsymbol{y}) b_t(\boldsymbol{y}) & t = t'. \end{cases} \tag{30}$$

ACKNOWLEDGMENT

This research has been funded by BBSRC grant 92/BIO12093 and Pfizer Central Research. Part of the work has been done while Ata Kabán was supported by re:source, The Council for Museums, Archives and Libraries, Grant Number RE/092, at the University of Paisley. The experiments were carried out with the NETLAB neural network toolbox, available from `http://www.ncrg.aston.ac.uk/netlab`. The hierarchical graph plots were based on the PhiVis software `http://www.ncrg.aston.ac.uk/PhiVis`, and we are grateful to Michael Tipping for allowing us to use his software. Yi Sun would like to thank Mário A. T. Figueiredo for providing his software. Finally, we are most grateful to the anonymous referees for their valuable and insightful comments on the manuscript.

REFERENCES

[1] F. Aurenhammer, “Voronoi diagrams—survey of a fundamental geometric data structure”, *ACM Computing Surveys*, vol.3, pp.345–405, 1991.

[2] O. Barndorff-Nielsen, *Information and Exponential Families in Statistical Theory*, Wiley, Chichester, 1978.

[3] J. Bernardo and A. Smith, *Bayesian Theory*, Chichester, UK: J. Wiley & Sons, 1994.

[4] C. M. Bishop and M. Svensén and C. K. I. Williams, “GTM: The Generative Topographic Mapping”, *Neural Computation*, vol.10, no.1, pp.215–235, 1998.

[5] C. M. Bishop and M. Svensén and C. K. I. Williams, “Developments of the Generate Topographic Mapping”, *Neurocomputing*, vol.21, pp.203–224, 1998.

[6] C. M. Bishop and M. Svensén and C. K. I. Williams, “Magnification factors for the GTM algorithm”, *Proceedings IEE Fifth International Conference on Artificial Neural Networks*, London, pp.64–69, 1997.

[7] C. M. Bishop and M. E. Tipping, “A hierarchical latent variable model for data visualization”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.20, no.3, pp.281–293, 1998.

[8] D. M. Boulton and C. S. Wallace, “An information measure for hierarchic classification”, *Computer Journal*, vol.16, no.3, 1973, pp.254–261.

[9] G. Celeux and S. Chrétien, F. Forbes and A. Mkhadri, “A component-wise EM algorithm for mixtures”, *J. Comput. Graphical Statistics*, vol.10, pp.699–712, 2001.

[10] A. P. Dempster and N. M. Laird and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm”, *J. Roy. Stat. Soc.* B, vol.39, pp.1–38, 1977.

[11] M. Figueiredo and A. K. Jain, “Unsupervised learning of finite mixture models”, *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol.24, pp.381–396, 2002.

[12] P. Horton and K. Nakai, “A probabilistic classification system for predicting the cellular localization sites of proteins”, *Intelligent System in Molecular Biology*, Vol.4, pp.109–115, 1996.

[13] A. Inselberg and B. Dimsdale. “Parallel coordinates: A tool for visualizing mulitdimensional geometry”. *Proc. of Visualization ’90*, pp. 361–78, 1990.

[14] A. Kabán and M. Girolami, “A combined latent class and trait model for the analysis and visualization of discrete data”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.23, no.8, pp.859–872, 2001.

[15] A. Kabán, P. Tiňo and M. Girolami, “A General Framework for a Principled Hierarchical Visualisation of Multivariate Data”, IDEAL’02, pp.17–23, Lecture Notes in Computer Science, Springer Verlag, 2002.

[16] D. A. Keim, H. P. Kriegel and M. Ankerst. “Recursive pattern: a technique for visualizing very large amounts of data”, *Proc. of Visualization ’95*, pp. 279–86, 1995.

[17] T. Kohonen, “The self-organizing map”, *Proceedings of the IEEE*, vol.78, no.9, pp.1464–1479, 1990.

[18] T. Kohonen, *Self-Organizing Maps*, Berlin: Springer-Verlag, 1999.

[19] Y. Koren and D. Harel,“A two-way visualization method for clustered data”, *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, Washington, D.C., ACM Press, pp.589–594, 2003.

[20] J. LeBlanc, M. O. Ward and N. Wittels. “Exploring n-dimensional databases”, *Proc. of Visualization ’90*, pp. 230–7, 1990.

[21] P. McCullagh and L. Nelder, *Generalized Linear Models*, Chapman and Hall, 1985.

[22] R. Miikkulainen, “Script recognition with hierarchical feature maps”, *Connection Science*, vol.2, pp.83–101, 1990.

[23] E. Pampalk, W. Goebl and G. Widmer, “Visualizing Changes in the Structure of Data for Exploratory Feature Selection”, in P. Domingos, C. Faloutsos, T. Senator, H. Kargupta and L. Getoor, *KDD 2003*, pp. 157–166, 2003.

[24] W. Ribarsky, E. Ayers, J. Eble and S. Mukherjea. “Glyphmaker: Creating customized visualization of complex data. *IEEE Computer*, vol. 27 (7), pp. 57–64, 1994.

[25] S. J. Roberts and C. Holmes and D. Denison, “Minimum-Entropy Data Partitioning Using Reversible Jump Markov Chain Monte Carlo”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.23, pp.909–914, 2001.

[26] C. Stein, “Approximation of improper prior measures by proper probability measures”, Bernoulli, Bayes, Laplace Festschrift. (J.Neyman and L. LeCam, eds.). Berlin: Springer, 1965, pp. 217–240.

[27] P. Tiňo and I. T. Nabney, “Hierarchical GTM: Constructing localized nonlinear projection manifolds in a principled way”, *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol.24, pp.639–656, 2002.

[28] C. S. Wallace and D. L. Dowe, “Minimum Message Length and Kolmogorov Complexity”, *The Computer Journal*, vol.42, pp.270–283, 1999.

[29] C. S. Wallace and D. M. Boulton, “An information measure for classification”, *The Computer Journal*, vol.11, no.2, 1968, pp.185–194.

[30] C. S. Wallace and D. L. Dowe, “Intrinsic classification by MML — the Snob program”, *Proc. Seventh Australian Joint Conf. on Artificial Intelligence*, ed. C. Zhang et al., World Scientific publisher, pp.37–44, 1994.

[31] C. S. Wallace and D. L. Dowe, “Refinements of MDL and MML Coding”, *Computer Journal*, vol.42, no.4, pp.330–337, 1999.

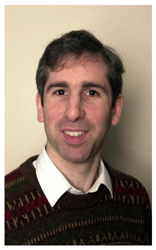
[32] C. S. Wallace and D. L. Dowe, “MML clustering of multi-state, Poisson, von Mises circular and Gaussian distributions”, *Statistics and Computing*, vol. 10, pp.73–83, 2000.

[33] C. S. Wallace and P. R. Freeman, “Estimation and Inference by Compact Coding”, *Journal of the Royal Statistical Society*, series B, vol. 49, pp.240–265, 1987.

[34] E. J. Wegman, “Hyperdimensional data analysis using parallel coordinates.” *Journal of the American Statistical Association*, vol. 411 (85), p. 664, 1990.

[35] R. Wolke and H. Schwetlick, “Iterative reweighted least squares: algorithms, convergence analysis, and numerical comparisons”, *SIAM Journal on Scientific and Statistical Computing*, vol.9, no.5, pp.907–921, 1999.

[36] J. Yang, M. O. Ward, and E. A. Rundensteiner, “Interactive Hierarchical Displays: A General Framework for Visualization and Exploration of Large Multivariate Data Sets”, *Computers and Graphics Journal*, vol.27, pp.265-283, 2002.



Ian T. Nabney is a Professor at Aston University, UK. He studied maths at Oxford, and defected to Cambridge to do his PhD, which was in algebra (infinite group theory, to be precise). He then spent five years with Logica (a UK software house) at their R&D lab, where he developed a wide range of neural network applications. In 1995 he was appointed as a Lecturer in the Department of Computer Science and Applied Maths at Aston University. In 2001 he was promoted to Senior Lecturer as part of what (several reorganisations later) is now the Information Engineering Subject Group. In 2004 he was promoted to a Chair in Computer Science. He is also on the committee of the Natural Computing Applications Forum (NCAF) as Treasurer and also organise the scientific program some of their meetings. In his spare time he plays the piano. His research spans both the theory and applications of neural networks and other pattern recognition techniques. Much of his work is inspired, directly or indirectly, by industrial problems in bioinformatics, biosignal processing, condition monitoring, remote sensing and financial forecasting. He has put his experience of software engineering to good use through developing the Netlab toolbox for neural networks and related techniques. Between October 2000 and June 2002 he was the Director of the Cardionetics Institute of Bioinformatics, which researched methods for extracting clinically valuable information from electrocardiogram (ECG) data.

Yi Sun studied at thermal energy engineering and received her BSc in University of Science & Technology Beijing (USTB), China. In 1999, she studied at Aston University and obtained her PhD in 2003. She works currently as a research fellow in the neural network group at University of Hertforshire, UK. Her current research interests are probabilistic modeling, data visualization, pattern classification and clustering, and neural computation models of cognitive and psychological processes.

Peter Tiňo studied at Slovak University of Technology and obtained PhD (1997) from Slovak Academy of Sciences. He worked at NEC Research Institute in Princeton NJ, USA, Austrian Research Institute for AI in Vienna, Austria and Aston University in Birmingham, UK. He is currently a lecturer at the School of Computer Science, University of Birmingham, UK. His scientific interests are probabilistic modeling and visualization of structured data, dynamical systems and fractal analysis.

Ata Kabán received her BA with honours in Musical Composition, her MA and PhD in Musicology from the Music Academy ’GH. Dima’ of Cluj-Napoca, Romania. Her thesis treated algebraic elements in XX-th century’s musical creation, defended against Professor Gh. Firca from Bucharest. She received her BSc with honours in Computer Science from ’Babes-Bolyai’ University of Cluj-Napoca, Romania. She is a research assistant at the University of Paisley, working on the development of probabilistic models for text based Information Retrieval and from June to December 2000 was a visiting researcher at the Laboratory of Computer and Information Science, Helsinki University of Technology. She is currently a lecturer at the School of Computer Science, University of Birmingham, UK. Her research interests are probabilistic modelling,statistical information processing, machine learning, scientific data mining,information retrieval, activity profiling.