Institut für Numerische Simulation
Rheinische Friedrich-Wilhelms-Universität Bonn

Wegelerstraße 6 • 53115 Bonn • Germany
phone +49 228 73-3427 • fax +49 228 73-7527
www.ins.uni-bonn.de

M. Griebel, A. Hullmann

Dimensionality reduction of high-dimensional data with a non-linear principal component aligned generative topographic mapping

INS Preprint No. 1311

July 2013

DIMENSIONALITY REDUCTION OF HIGH-DIMENSIONAL DATA WITH A NON-LINEAR PRINCIPAL COMPONENT ALIGNED GENERATIVE TOPOGRAPHIC MAPPING

M. GRIEBEL* AND A. HULLMANN*

Abstract. Most high-dimensional real-life data exhibit some dependencies such that data points do not populate the whole data space but lie approximately on a lower-dimensional manifold. A major problem in many data mining applications is the detection of such a manifold and the expression of the given data in terms of a moderate number of latent variables. We present a method which is derived from the generative topographic mapping (GTM) and can be seen as a non-linear generalization of the Principal Component Analysis (PCA). It can detect certain non-linearities in the data but does not suffer from the curse of dimension with respect to the latent space dimension as the original GTM and thus allows for higher embedding dimensions. We provide experiments that show that our approach leads to an improved data reconstruction compared to the purely linear PCA and that it can furthermore be used for classification.

Key words. dimensionality reduction, generative topographic mapping, principal component analysis, density estimation, additive model, classification

AMS subject classifications. 62G07, 62H25, 62H30

1. Introduction. Many high-dimensional data exhibit some correlation structure, which means that they do not populate the whole data space and thus possibly have a lower intrinsic dimension. Then, a suitable low-dimensional projection of the data allows for a more compact description, a better visualization and a more efficient processing. Figure 1.1 shows the Swiss roll as an example, a typical synthetic dataset [18], and its two-dimensional embedding.

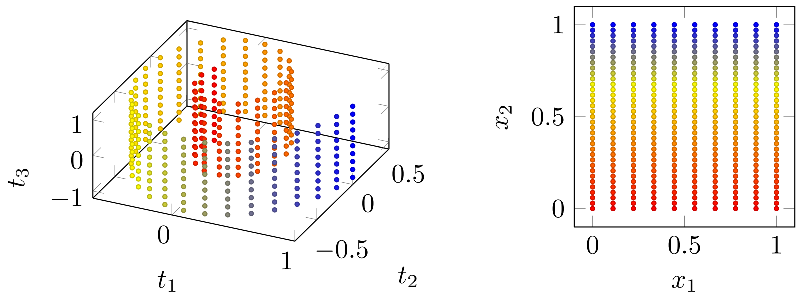

FIG. 1.1. *The Swiss roll (left) and the two-dimensional embedding (right). The point colors are for visualization purposes only*

One common approach to dimensionality reduction is to express the high-dimensional data in terms of latent variables. Here, the Principal Component Analysis (PCA) [13], which is based on the diagonalization of the data covariance matrix, is the best-known method. However, it is by construction a linear method and as such it is not capable of modeling non-linear lower-dimensional dependencies. Figure 1.2 shows that the topological structure of the Swiss roll dataset is not preserved under the embedding into two dimensions, which means that points originally far apart on the

*Institute for Numerical Simulation, University of Bonn, Wegelerstr. 6, D-53115 Bonn, Germany ({`griebel, hullmann`}`@ins.uni-bonn.de`)

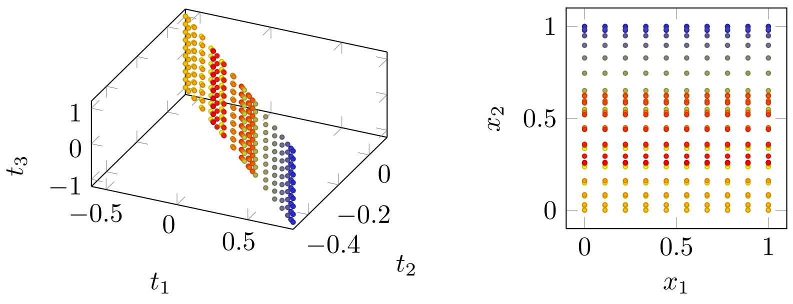

FIG. 1.2. *The reconstructed Swiss roll with a two-dimensional PCA model (left) and the two-dimensional embedding (right)*

manifold are close-by in the two-dimensional projection. Moreover, the reconstructed structure does not at all look similar to the original Swiss roll.

That is why various non-linear methods have been developed. Some common approaches are multidimensional scaling (MDS), curvilinear component analysis (CCA), curvilinear distance analysis (CDA), Laplacian eigenmaps (LE), locally linear embedding (LLE), Kohonen's self-organizing map (SOM) and generative topographic mapping (GTM), cf. [18]. Of course, there are also many methods closely related to the PCA with additional capabilities. One approach that can deal with non-linearities is kernel PCA (KPCA) [21], although this method does not ultimately lead to a low-dimensional embedding of the data. If the assumption of Gaussianity does not hold, the Independent component analysis (ICA) [6] is useful. The probabilistic PCA [24] and the sensible PCA [20] provide a latent variable model view on the PCA that allows the use of the Expectation Maximization algorithm [8, 19] and supports various extensions, e.g., mixture models [23]. Robustness towards outliers is achieved by using an L_1-norm maximization [17].

We base our work on the generative topographic mapping (GTM) [4], which explicitly constructs a non-linear mapping from the L-dimensional latent space $[0,1]^L$ into the D-dimensional data space $\mathbb{R}^D$. This method has many advantages. For example, it allows to embed new data points that were not available when fitting the model. Furthermore, it is possible to generate new data points or synthetic data similar to the given data. Unfortunately, the discretization of the mapping by a full grid suffers from an exponential dependence of the number of degrees of freedom on the latent space dimension L. This effect is often referred to as the "curse of dimension" [2]. This limits the use of the GTM to dimensions $L \leq 3$. Consequently, it is not suited for a drastic initial dimensionality reduction (also called *hard dimensionality reduction* [18]) of high-dimensional data to a moderate amount of latent space dimensions, e.g. $L \approx 6$. While there are modifications of the original GTM with a semi-linear approach [3] or sparse grids [12] which allow to treat somewhat higher latent space dimensions, their practical use is still limited.

In this paper, we present a low-cost but yet partially non-linear approach that relates to the Principal Component Analysis and the GTM. Our Principal Component Generative Topographic Mapping (PCGTM) uses a generative model with a mapping $\mathbf{y} : [0,1]^L \to \mathbb{R}^D$ with the structure

$$\mathbf{y}(\mathbf{x}) = \sum_{d=1}^{D} g^{(d)}\left(x^{(m(d))}\right)\mathbf{v}^{(d)} \;, \tag{1.1}$$

where $m : \{1, \ldots, D\} \to \{1, \ldots, L\}$ assigns the orthonormal vectors $\mathbf{v}^{(d)} \in \mathbb{R}^D, d = 1, \ldots, D$, typically the principal components of the data, to the L latent space dimensions. The not yet specified functions[1] $g^{(d)} : [0,1] \to \mathbb{R}$ depend only on the latent variables $x^{(m(d))} \in [0,1]$ for $d = 1, \ldots, D$, respectively, with $\mathbf{x} = (x^{(1)}, \ldots, x^{(L)}) \in [0,1]^L$. Our model (1.1) offers two advantages over the PCA: We do not have to assume multivariate Gaussian distributions in the latent and in the data space, but can account for non-parametric distributions by our degrees of freedom $g^{(d)}, d = 1, \ldots, D$. Moreover, the PCA approximation has a 1:1 relationship of the L latent variables with the first[2] L principal components, where all $\mathbf{v}^{(d)}$ with $d = L+1, \ldots, D$ are discarded. This is optimal under the model assumption of Gaussian distributions, but, for non-Gaussian distributions, linear decorrelation does not imply independence and the principal components $\mathbf{v}^{(d)}$ with $d = L+1, \ldots, D$ can still depend non-linearly on the first L ones. We can model these dependencies to some extent by letting the same latent variable $x^{(l)}$ for $l = 1, \ldots, L$ control all principal components $\mathbf{v}^{(d)}$ with $m(d) = l$. In consequence, we assign *all* principal components to the latent variables and do not need to truncate after L dimensions anymore. Of course, our model (1.1) is still restrictive compared to a model where the principal components may depend on more than one latent variable, but this limitation allows us to fit the model to given data without a significant increase in computational costs compared to standard PCA and furthermore prevents overfitting.

This paper is organized as follows: In Section 2, we describe our non-linear generative model, state a target functional for a desirable mapping between latent space and data space and describe the necessary discretizations. In Section 3, we give the computational steps necessary for the minimization of the target functional and discuss the computational complexity involved. In Section 4, we consider various synthetic and publicly available real-world datasets and show that the PCGTM mostly surpasses the PCA with the same latent space dimension. Finally, in Section 5, we give some concluding remarks.

2. The Principal Component GTM.

In the following subsection, we introduce the generative model of the new Principal Component GTM. Then, in Subsection 2.2, we describe the necessary discretization steps for its numerical treatment. Note that the following presentation is a slight generalization of the original GTM [4] that allows us to express several generative models in this framework.

2.1. Construction.

Our aim is to represent a D-dimensional probability density $p(\mathbf{t}) \geq 0, \mathbf{t} \in \mathbb{R}^D$, in the data space by a density that is intrinsically low-dimensional. To this end, our generative model relies on a general vector-valued function

$$\mathbf{y} : [0,1]^L \to \mathbb{R}^D \tag{2.1}$$

with $L \ll D$ that connects the L-dimensional latent space $[0,1]^L$ and the data space $\mathbb{R}^D$. The generated density is

$$q_{\mathbf{y},\beta}(\mathbf{t}) = \left(\frac{\beta}{2\pi}\right)^{D/2} \int_{[0,1]^L} \exp\left(-\frac{\beta}{2}\|\mathbf{y}(\mathbf{x}) - \mathbf{t}\|^2\right) \mathrm{d}\mathbf{x}\ , \tag{2.2}$$

[1] In the context of additive models and non-parametric regression of one-dimensional response variables [10, 5], the $g^{(d)}$ are called *smooth functions*.

[2] We assume that the principal components $\mathbf{v}^{(d)}, d = 1, \ldots, D$, are sorted by their eigenvalues in descending order.

which is the image of a L-dimensional uniform distribution under the mapping $\mathbf{y}$ with additional D-dimensional Gaussian noise with variance β^{-1}, see Fig. 2.1 for an illustration. It is easy to see that $\int_{\mathbb{R}^D} q_{\mathbf{y},\beta}(\mathbf{t})\mathrm{d}\mathbf{t} = 1$, i.e., $q_{\mathbf{y},\beta}$ is indeed a probability density in the D-dimensional data space.

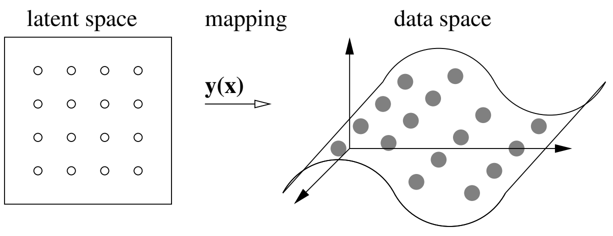

FIG. 2.1. *The L-dimensional data space is mapped by* $\mathbf{y}$ *into the D-dimensional data space. There, the model assumes multivariate Gaussian noise with variance* β^{-1}

The aim is now to choose a mapping $\mathbf{y}$ and an inverse variance $\beta \in \mathbb{R}_+$ such that the dissimilarity between $q_{\mathbf{y},\beta}$ and p is minimized. To be precise, we want to minimize the cross-entropy [16]

$$\begin{aligned}\mathcal{H}(p, q_{\mathbf{y},\beta}) &:= -\int_{\mathbb{R}^D} p(\mathbf{t}) \log q_{\mathbf{y},\beta}(\mathbf{t})\mathrm{d}\mathbf{t} \\ &= -\int_{\mathbb{R}^D} p(\mathbf{t}) \log \int_{[0,1]^L} \exp\left(-\frac{\beta}{2}\|\mathbf{y}(\mathbf{x}) - \mathbf{t}\|^2\right) \mathrm{d}\mathbf{x}\mathrm{d}\mathbf{t} - \frac{D}{2}\log\frac{\beta}{2\pi} \end{aligned} \tag{2.3}$$

between the model density $q_{\mathbf{y},\beta}$ and the given data density $p(\mathbf{t})$. However, in most practical settings, no continuous data space density $p(\mathbf{t})$ is available, and an empirical density based on N data points or samples $(\mathbf{t}_n)_{n=1}^N$ is given instead. Therefore, we replace the continuous density $p(\mathbf{t})$ by a sum of Dirac delta functions $p_N^{\mathrm{emp}}(\mathbf{t}) = \frac{1}{N}\sum_{n=1}^N \delta_{\mathbf{t}_n}(\mathbf{t})$. Then, the d$\mathbf{t}$-integral in (2.3) is replaced by a sum, and we now have to minimize

$$\mathcal{H}_N(\mathbf{y}, \beta) := -\frac{1}{N}\sum_{n=1}^N \log \int_{[0,1]^L} \exp\left(-\frac{\beta}{2}\|\mathbf{y}(\mathbf{x}) - \mathbf{t}_n\|^2\right) \mathrm{d}\mathbf{x} - \frac{D}{2}\log\frac{\beta}{2\pi} \tag{2.4}$$

with respect to $\mathbf{y}$ and β.

Of course, the problem of minimizing (2.4) is not well-posed and we have to require as a prior that $\mathbf{y} \in V$, where the function set $V \subset \{f : [0,1]^L \to \mathbb{R}^D\}$ imposes a certain structure or smoothness on the mapping $\mathbf{y}$. For example, we could assume that the components of $\mathbf{y}$ exhibit some Sobolev smoothness and that the sum of their $H^s([0,1]^L)$-norms is bounded, i.e.

$$\mathbf{y} \in V_{H^s} := \left\{ f = (f^{(1)}, \dots, f^{(D)})^T : [0,1]^L \to \mathbb{R}^D \quad \text{with} \quad \sum_{d=1}^D \|f^{(d)}\|^2_{H^s([0,1]^L)} < R \right\}. \tag{2.5}$$

This would amount to adding a regularization term weighted by a regularization parameter λ to the functional (2.4). For an in-depth discussion of the relation between regularization terms and associated function spaces, see [21]. Then, when discretizing $\mathbf{y} \in V_{H^s}$ by a typical tensor product type method, we suffer from the curse of dimensionality in terms of L. Similarly, the original GTM assumes a superposition of Gaussians for the components of $\mathbf{y}$, which means

$$\mathbf{y} \in V_{\mathrm{GTM}} := \Bigg\{ f = (f^{(1)}, \ldots, f^{(D)})^T : [0,1]^L \to \mathbb{R}^D \quad \text{with} \\ f^{(d)}(\mathbf{x}) = \sum_i \alpha_i^{(d)} \exp\left(-\frac{\|\mathbf{z}_i - \mathbf{x}\|^2}{2\sigma^2} \right) \quad \text{for} \quad d = 1, \ldots, D \Bigg\}$$

with coefficients $(\alpha_i^{(d)})_i$ and points $(\mathbf{z}_i)_i$ that are centered on a uniform grid structure. This grid structure again introduces the curse of dimension. The standard PCA with L principal components $(\mathbf{v}^{(l)})_{l=1}^L$ emerges by the choice

$$\mathbf{y} \in V_{\mathrm{PCA}} := \left\{ f(\mathbf{x}) = \sum_{l=1}^{L} \Phi^{-1}(x^{(l)}) \mathbf{v}^{(l)} \quad \text{with} \quad \mathbf{v}^{(l)} \in \mathbb{R}^D, l = 1, \ldots, L \right\}, \tag{2.6}$$

where $\mathbf{x} = (x^{(1)}, \ldots, x^{(L)})$ and $\Phi^{-1} : (0,1) \to \mathbb{R}$ denotes the inverse cumulative distribution function of the normal distribution. Indeed, observe that for (2.6) the model (2.2) is the density of a D-variate Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{V}\mathbf{V}^T + \beta^{-1}\mathbf{I})$, where $\mathbf{I} \in \mathbb{R}^{D \times D}$ is the identity matrix and $\mathbf{V} = [\mathbf{v}^{(l)}]_{l=1}^L \in \mathbb{R}^{D \times L}$. Since $\dim V_{\mathrm{PCA}} = L \cdot D$, there is no curse of dimension anymore, but the cost of this simplification is a very rigid model.

In our Principal Component GTM, we assume that the number L of latent space dimensions is fixed and that

$$\mathbf{y}(\mathbf{x}) \in V_{\mathrm{PCGTM}} := \left\{ f : [0,1]^L \to \mathbb{R}^D \quad \text{with} \quad f(\mathbf{x}) = \sum_{d=1}^{D} g^{(d)}\big(x^{(m(d))}\big) \mathbf{v}^{(d)} \right\}, \tag{2.7}$$

where $\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(D)}$ are orthonormal vectors, typically the principal components of the data covariance matrix, the functions $g^{(d)} : [0,1] \to \mathbb{R}, d = 1, \ldots, D$, are our degrees of freedom and

$$m : \{1, \ldots, D\} \to \{1, \ldots, L\}$$

is a surjective function that assigns the D principal components of the data to the L latent space dimensions.

We now describe how we build $\mathbf{v}^{(d)}, d = 1, \ldots, D$, and the mapping m from the given data samples $(\mathbf{t}_n)_{n=1}^N$. To this end, we assume that the samples are centered, i.e.

$$\frac{1}{N} \sum_{n=1}^{N} \mathbf{t}_n = \mathbf{0} ,$$

otherwise we have to demean them in the first place. Then, the sample covariance matrix is given by

$$\mathbf{C} := \frac{1}{N-1} \sum_{n=1}^{N} \mathbf{t}_n \cdot \mathbf{t}_n^T = \frac{1}{N-1} \mathbf{T}\mathbf{T}^T \in \mathbb{R}^{D \times D} ,$$

where $\mathbf{T} \in \mathbb{R}^{D \times N}$ is a matrix that contains the data points $(\mathbf{t}_n)_{n=1}^N$ as columns. Now, we decompose the positive semi-definite matrix $\mathbf{C}$ into

$$\mathbf{C} = \mathbf{V}\mathbf{D}\mathbf{V}^T$$

with the orthogonal matrix $\mathbf{V} \in \mathbb{R}^{D \times D}$ and the diagonal matrix

$$\mathbf{D} = \operatorname{diag}\left(\lambda^{(1)}, \dots, \lambda^{(D)}\right) \in \mathbb{R}^{D \times D} \ .$$

The (non-negative) eigenvalues $(\lambda^{(d)})_{d=1}^D$ of $\mathbf{C}$ are assumed to be arranged in descending order. Then, we choose the $\mathbf{v}^{(d)}, d = 1, \dots, D$, simply as the columns of $\mathbf{V}$.

We now want to define a meaningful mapping $m : \{1, \dots, D\} \to \{1, \dots, L\}$ that does a sensible matching of principal components to latent variables. To this end, we look at the data in terms of projections onto the principal components, i.e., of the sets $\mathbf{S}^{(d)} := \left(\langle \mathbf{v}^{(d)}, \mathbf{t}_n \rangle\right)_{n=1}^N$ for all $d = 1, \dots, D$, where $\langle \cdot, \cdot \rangle$ denotes the standard Euclidean scalar product. Just like the PCA we form a 1:1 relationship for the first L principal components and the latent variables by setting

$$m(d) = d \quad \text{for} \quad d = 1, \dots, L \ .$$

Now we want to match the remaining $\mathbf{S}^{(d)}, d = L + 1, \dots, D$, with the latent space variables. Note that by assigning $\mathbf{S}^{(d)}$ to the latent variable $x^{(m(d))}$, we are able to model the dependencies of $\mathbf{S}^{(d)}$ and $\mathbf{S}^{(m(d))}$ because they are then controlled by the same latent variable. However, we cannot use linear correlations to determine which $\mathbf{S}^{(d)}$ and $\mathbf{S}^{(m(d))}$ belong together since they are all linearly decorrelated. This can be seen from the fact that the $\mathbf{S}^{(d)}, d = 1, \dots, D$, are the rows of $\mathbf{V}^T\mathbf{T} \in \mathbb{R}^{D \times N}$ and their sample covariance matrix

$$\frac{1}{N-1}\mathbf{V}^T\mathbf{T}\mathbf{T}^T\mathbf{V} = \mathbf{V}^T\mathbf{C}\mathbf{V} = \mathbf{D}$$

is diagonal. However, we can still use non-linear measures of correlation like Spearman's [22] rank correlation coefficient ρ_{ld} or Kendall's [15] tau correlation coefficient τ_{ld} to measure the pairwise dependencies between the $\mathbf{S}^{(d)}, d = 1, \dots, D$. Then, we map every $\mathbf{S}^{(d)}$ for $d = L + 1, \dots, D$ to the latent variable $x^{(m(d))}$ for which the corresponding $\mathbf{S}^{(m(d))}$ has the highest measure of non-linear correlation to $\mathbf{S}^{(d)}$. Formally, this reads as

$$m(d) = \begin{cases} d & \text{for} \quad 1 \le d \le L \ , \\ \arg\max_{1 \le l \le L} |\rho_{ld}| \quad \text{or} \quad \arg\max_{1 \le l \le L} |\tau_{ld}| & \text{else} \ . \end{cases} \tag{2.8}$$

Thus, by forming groups of the $\mathbf{S}^{(d)}, d = 1, \dots, D$, that share the same latent space variable, we allow some of the non-linear features in the data to be captured by our model. Of course, this model is heuristic, but we consider it the natural extension of the PCA, and in our experiments it turned out to produce more accurate results than other choices.

2.2. Discretization. We have already replaced the data space density by the empirical distribution of the samples. Now, we need two further types of discretization. One is the use of a quadrature rule for the $[0, 1]^L$-integral in (2.4), the other one is related to the approximation of the functions $g^{(d)} : [0, 1] \to \mathbb{R}, d = 1, \dots, D$, in (2.7).

We start with the functions $g^{(d)} : [0,1] \to \mathbb{R}, d = 1, \ldots, D$, and choose a linear spline basis on $[0,1]$. On level J, we have $n_J = 2^J + 1$ hat functions

$$\phi_{J,i}(x) = \max(1 - 2^J \left|x - z_{J,i}\right|, 0) ,$$

which are centered at equidistant points

$$z_{J,i} = 2^{-J}(i-1)$$

for $i = 1, \ldots, n_J$. Then, for $d = 1, \ldots, D$ and given the coefficient vectors

$$\bar{\alpha}_J^{(d)} = \left(\alpha_{J,1}^{(d)}, \ldots, \alpha_{J,n_J}^{(d)}\right)^T \in \mathbb{R}^{n_J} ,$$

we set

$$g_J^{(d)}(x) := \sum_{i=1}^{n_J} \alpha_{J,i}^{(d)} \phi_{J,i}(x) \tag{2.9}$$

and

$$V_{\mathrm{PCGTM},J} := \left\{ f : [0,1]^L \to \mathbb{R}^D \quad \text{with} \quad f(\mathbf{x}) = \sum_{d=1}^{D} g_J^{(d)}\left(x^{(m(d))}\right)\mathbf{v}^{(d)} \right\} \subset V_{\mathrm{PCGTM}} . \tag{2.10}$$

We have not yet specified how we treat the $[0,1]^L$-integral in (2.2) or (2.4) without suffering from the curse of dimension. To this end, we can choose any tensorized quadrature rule, and due to the special structure of our $\mathbf{y} \in V_{\mathrm{PCGTM}}$ or $\mathbf{y}_J \in V_{\mathrm{PCGTM},J}$, we only have to deal with the complexity of the one-dimensional quadrature rule as we see in the following Section 3. Thus, at the moment, we just assume to have a one-dimensional quadrature rule on level K with points $(x_{K,i})_{i=1}^{m_K}$ and positive weights $(\omega_{K,i})_{i=1}^{m_K}$, i.e.,

$$Q_K f = \sum_{i \in \chi_K} \omega_{K,i} f(x_{K,i}) \approx \int_0^1 f(x)\mathrm{d}x \tag{2.11}$$

for $\chi_K := \{1, \ldots, m_K\}$. The L-dimensional case is then covered by

$$Q_K^L f = \sum_{\mathbf{i} \in \chi_K^L} \omega_{K,\mathbf{i}} f(\mathbf{x}_{K,\mathbf{i}}) \approx \int_{[0,1]^L} f(\mathbf{x})\mathrm{d}\mathbf{x} \tag{2.12}$$

with

$$\omega_{K,\mathbf{i}} = \prod_{l=1}^{L} \omega_{K,i^{(l)}} \quad \text{and} \quad \mathbf{x}_{K,\mathbf{i}} = \left(x_{K,i^{(1)}}, \ldots, x_{K,i^{(L)}}\right) \tag{2.13}$$

for $\mathbf{i} = \left(i^{(1)}, \ldots, i^{(L)}\right) \in \chi_K^L := \times_{l=1}^{L} \chi_K$.

Instead of our target functional (2.4), we now switch to its discretized version for numerical treatment. The objective then reads: Minimize

$$\mathcal{H}_{N,K}(\mathbf{y}_J, \beta) := -\frac{1}{N} \sum_{n=1}^{N} \log \sum_{\mathbf{i} \in \chi_K^L} \omega_{K,\mathbf{i}} \exp\left(-\frac{\beta}{2}\left\|\mathbf{y}_J(\mathbf{x}_{K,\mathbf{i}}) - \mathbf{t}_n\right\|^2\right) - \frac{D}{2} \log \frac{\beta}{2\pi} \tag{2.14}$$

with respect to $\mathbf{y}_J \in V_{\mathrm{PCGTM},J}$ and β.

In most numerical problems, the asymptotics of the discretization error is relevant. However, as our data consists only of a finite amount of samples and is noisy in most cases anyway, we content ourselves with discretization and quadrature levels that are sufficient. This means that the discretization level J needs to be high enough so that our model captures the features of the data, but not too high in order to avoid overfitting.[3] The level K of the quadrature rule (2.12) should be high enough such that there is a sufficient number of quadrature points in the support of every basis function $\phi_{J,i}, i = 1, \ldots, n_J$. In our experiments we observed that $K = J + 3$ was enough for a simple midpoint rule.

Remark. We could have started directly with a grid based latent space distribution instead of starting with a uniform one and choosing a quadrature rule. However, in that case we cannot describe the PCA in our setting, see (2.6), since the multivariate Gaussian distribution has a continuous density.

In the following Section 3, we show how the minimization of (2.14) can be done efficiently.

3. Minimization routine.

We need to minimize the GTM functional $\mathcal{H}_{N,K}$ efficiently even though it is non-linear and non-convex in $\mathbf{y}_J$ and β. Typically, the EM-algorithm [8] is used, which can also be regarded as the minimization of a free energy functional, cf. [19]. First, we start with a simple definition.

Definition 3.1 (Posterior probabilities). *First, we define the posterior probabilities* $R_{K,\mathbf{y}_J,\beta} : \{\mathbf{t}_n\}_{n=1}^N \times \{\mathbf{x}_{K,\mathbf{i}}\}_{\mathbf{i}\in\chi_K^L} \to \mathbb{R}_{\geq 0}$ *by*

$$R_{K,\mathbf{y}_J,\beta}(\mathbf{t}_n, \mathbf{x}_{K,\mathbf{i}}) := \frac{\exp\left(-\frac{\beta}{2}\|\mathbf{y}_J(\mathbf{x}_{K,\mathbf{i}}) - \mathbf{t}_n\|^2\right)}{\sum_{\mathbf{j}\in\chi_K^L} \omega_{K,\mathbf{j}} \exp\left(-\frac{\beta}{2}\|\mathbf{y}_J(\mathbf{x}_{K,\mathbf{j}}) - \mathbf{t}_n\|^2\right)} \tag{3.1}$$

for all $n = 1, \ldots, N$. Then, the following Lemma shows that, using (3.1), we can rearrange (2.14) to the free energy form.

Lemma 3.2. *It holds that*

$$\mathcal{H}_{N,K}(\mathbf{y}_J, \beta) = \frac{1}{N}\sum_{n=1}^N \sum_{\mathbf{i}\in\chi_K^L} \omega_{K,\mathbf{i}} R_{K,\mathbf{y}_J,\beta}(\mathbf{t}_n, \mathbf{x}_{K,\mathbf{i}}) \log R_{K,\mathbf{y}_J,\beta}(\mathbf{t}_n, \mathbf{x}_{K,\mathbf{i}}) \tag{3.2}$$

$$+ \frac{\beta}{2N}\sum_{n=1}^N \sum_{\mathbf{i}\in\chi_K^L} \omega_{K,\mathbf{i}} R_{K,\mathbf{y}_J,\beta}(\mathbf{t}_n, \mathbf{x}_{K,\mathbf{i}}) \|\mathbf{y}_J(\mathbf{x}_{K,\mathbf{i}}) - \mathbf{t}_n\|^2 \tag{3.3}$$

$$- \frac{D}{2}\log\frac{\beta}{2\pi}\,.$$

[3]This is in fact regularization by discretization and poses an alternative to employing a regularization term.

Proof. The proof consists of a straightforward calculation. We first plug the definition of (3.1) into the term $\log R_{K,\mathbf{y}_J,\beta}$ of the right-hand side of (3.2) and obtain

$$\begin{aligned}
&\frac{1}{N}\sum_{n=1}^{N}\sum_{\mathbf{i}\in\chi_K^L}\omega_{K,\mathbf{i}}R_{K,\mathbf{y}_J,\beta}(\mathbf{t}_n,\mathbf{x}_{K,\mathbf{i}})\log R_{K,\mathbf{y}_J,\beta}(\mathbf{t}_n,\mathbf{x}_{K,\mathbf{i}})\\
=&-\frac{\beta}{2N}\sum_{n=1}^{N}\sum_{\mathbf{i}\in\chi_K^L}\omega_{K,\mathbf{i}}R_{K,\mathbf{y}_J,\beta}(\mathbf{t}_n,\mathbf{x}_{K,\mathbf{i}})\|\mathbf{y}_J(\mathbf{x}_{K,\mathbf{i}})-\mathbf{t}_n\|^2\\
&-\frac{1}{N}\sum_{n=1}^{N}\underbrace{\sum_{\mathbf{i}\in\chi_K^L}\omega_{K,\mathbf{i}}R_{K,\mathbf{y}_J,\beta}(\mathbf{t}_n,\mathbf{x}_{K,\mathbf{i}})}_{=1}\log\sum_{\mathbf{j}\in\chi_K^L}\omega_{K,\mathbf{j}}\exp\Big(-\frac{\beta}{2}\|\mathbf{y}_J(\mathbf{x}_{K,\mathbf{j}})-\mathbf{t}_n\|^2\Big)\,.
\end{aligned}$$

The last line corresponds to $\mathcal{H}_{N,K}(\mathbf{y}_J,\beta)+\frac{D}{2}\log\frac{\beta}{2\pi}$, see (2.14). The Lemma is then proven after a simple rearrangement. $\square$

Obviously, (2.14) is a non-linear functional and hard to minimize and the same holds for the free energy form (3.2) and (3.3). Fortunately, we can introduce a third parameter ψ_K that results in the closely related functional

$$\begin{aligned}
\mathcal{G}_{N,K}(\psi_K,\mathbf{y}_J,\beta) :=&\frac{1}{N}\sum_{n=1}^{N}\sum_{\mathbf{i}\in\chi_K^L}\omega_{K,\mathbf{i}}\psi_K(\mathbf{t}_n,\mathbf{x}_{K,\mathbf{i}})\log\psi_K(\mathbf{t}_n,\mathbf{x}_{K,\mathbf{i}}) &\quad (3.4)\\
&+\frac{\beta}{2N}\sum_{n=1}^{N}\sum_{\mathbf{i}\in\chi_K^L}\omega_{K,\mathbf{i}}\psi_K(\mathbf{t}_n,\mathbf{x}_{K,\mathbf{i}})\|\mathbf{y}_J(\mathbf{x}_{K,\mathbf{i}})-\mathbf{t}_n\|^2 &\quad (3.5)\\
&-\frac{D}{2}\log\frac{\beta}{2\pi}\,,
\end{aligned}$$

where $\psi_K:\{\mathbf{t}_n\}_{n=1}^N\times\{\mathbf{x}_{K,\mathbf{i}}\}_{\mathbf{i}\in\chi_K^L}\to\mathbb{R}_{\geq 0}$ is any function with

$$\sum_{\mathbf{i}\in\chi_K^L}\omega_{K,\mathbf{i}}\psi_K(\mathbf{t}_n,\mathbf{x}_{K,\mathbf{i}})=1 \tag{3.6}$$

for all $n=1,\ldots,N$. Obviously,

$$\mathcal{G}_{N,K}(R_{K,\mathbf{y}_J,\beta},\mathbf{y}_J,\beta)=\mathcal{H}_{N,K}(\mathbf{y}_J,\beta) \tag{3.7}$$

holds. We now minimize $\mathcal{G}_{N,K}$ by successively optimizing with respect to its single parameters ψ_K, $\mathbf{y}_J$ and β. We see in the following subsections that this is superior to the direct minimization of (2.14), as the resulting subproblems are uniquely and efficiently solvable even though the minimization problem for $\mathcal{H}_{N,K}$ is not. Moreover, it will be shown in Lemma 3.3 that the posterior probabilities $R_{K,\mathbf{y},\beta}$ minimize $\mathcal{G}_{N,K}$ with respect to ψ_K, i.e., we have

$$\arg\min\nolimits_{\psi_K}\mathcal{G}_{N,K}(\psi_K,\mathbf{y}_J,\beta)=R_{K,\mathbf{y}_J,\beta}\,. \tag{3.8}$$

This is analogous to statistical physics where the Boltzmann distribution minimizes the free energy. In combination with (3.7), this step can be understood as a projection back to the permissible search space since

$$\mathcal{G}_{N,K}(\arg\min\nolimits_{\psi_K}\mathcal{G}_{N,K}(\psi_K,\mathbf{y}_J,\beta),\mathbf{y}_J,\beta)=\mathcal{G}_{N,K}(R_{K,\mathbf{y}_J,\beta},\mathbf{y}_J,\beta)=\mathcal{H}_{N,K}(\mathbf{y}_J,\beta)\,.$$

Altogether, the minimization steps with respect to the three parameters of $\mathcal{G}_{N,K}$ have to be carried out in an outer iteration until convergence into a local minimum is achieved. In the next subsections, we give an in-depth description of the necessary computational steps and show that the computational complexity does not depend in an exponential way on the latent space dimension L. In order not to confuse the reader with another index carrying the iteration count, we just add $^{\text{old}}$- and $^{\text{new}}$-superscripts where necessary.

3.1. Minimization with respect to the first parameter ψ_K. Minimizing with respect to ψ_K is equivalent to the E-Step of the Expectation Maximization algorithm. The following Lemma shows that there exists a closed form solution.

Lemma 3.3. *It holds that*

$$\arg\min_{\psi_K} \mathcal{G}_{N,K}(\psi_K, \mathbf{y}_J, \beta) = R_{K,\mathbf{y}_J,\beta} \ .$$

Proof. Considering the structure of $\mathcal{G}_{N,K}$, see (3.4) and (3.5), it can be seen that the problem of finding the optimal ψ_K decouples into N independent problems of finding the optimal functions

$$\psi_K(\mathbf{t}_n, \cdot) : \{\mathbf{x}_{K,\mathbf{i}}\}_{\mathbf{i}\in\chi_K^L} \to \mathbb{R}_{\geq 0}$$

for $n = 1, \dots, N$. Assuming that we are at a critical point, the derivative of $\mathcal{G}_{N,K}$ needs to be orthogonal to the constraint surface (3.6), i.e., it needs to be collinear with $(\omega_{K,\mathbf{i}})_{\mathbf{i}\in\chi_K^L}$. This means that for all $n = 1, \dots, N$ a constant c_n exists such that

$$\begin{aligned} c_n \omega_{K,\mathbf{i}} &\overset{!}{=} \frac{\partial}{\partial \psi_K(\mathbf{t}_n, \mathbf{x}_{K,\mathbf{i}})} \mathcal{G}_{N,K}(\psi_K, \mathbf{y}_J, \beta) \\ &= \frac{1}{N}\omega_{K,\mathbf{i}} + \frac{1}{N}\omega_{K,\mathbf{i}} \log \psi_K(\mathbf{t}_n, \mathbf{x}_{K,\mathbf{i}}) + \frac{\beta}{2N}\omega_{K,\mathbf{i}} \|\mathbf{y}_J(\mathbf{x}_{K,\mathbf{i}}) - \mathbf{t}_n\|^2 \end{aligned}$$

for all $\mathbf{i} \in \chi_K^L$ and thus

$$\psi_K(\mathbf{t}_n, \mathbf{x}_{K,\mathbf{i}}) = \exp(Nc_n - 1) \exp\left(-\frac{\beta}{2}\|\mathbf{y}_J(\mathbf{x}_{K,\mathbf{i}}) - \mathbf{t}_n\|^2\right) .$$

The constraint (3.6) implies that $\exp(Nc_n - 1)$ is equal to the denominator of (3.1).$\square$

Now that we know how the E-step looks like, we would like to find an efficient way to compute it. The following Lemma shows that this can be done by solving only one-dimensional subproblems.

Lemma 3.4. *For $n = 1, \dots, N$, it holds that*

$$R_{K,\mathbf{y}_J,\beta}(\mathbf{t}_n, \mathbf{x}_{K,\mathbf{i}}) = \prod_{l=1}^{L} R^{(l)}_{K,\mathbf{y}_J,\beta}(\mathbf{t}_n, x_{K,i^{(l)}}) \tag{3.9}$$

for $\mathbf{i} = (i^{(1)}, \dots, i^{(L)}) \in \chi_K^L$ *with*

$$\begin{aligned} &R^{(l)}_{K,\mathbf{y}_J,\beta}(\mathbf{t}_n, x_{K,i}) \\ &\quad = \frac{\prod_{d\in m^{-1}(l)} \exp\left(-\frac{\beta}{2} g_J^{(d)}(x_{K,i})^2 + \beta\langle \mathbf{v}^{(d)}, \mathbf{t}_n\rangle g_J^{(d)}(x_{K,i})\right)}{\sum_{j\in\chi_K} \omega_{K,j} \prod_{d\in m^{-1}(l)} \exp\left(-\frac{\beta}{2} g_J^{(d)}(x_{K,j})^2 + \beta\left\langle \mathbf{v}^{(d)}, \mathbf{t}_n\right\rangle g_J^{(d)}(x_{K,j})\right)} \end{aligned}$$

for $i = 1, \dots, m_K$, $l = 1, \dots, L$ *and*

$$m^{-1}(l) := \{d \in \{1, \dots, D\} : m(d) = l\} .$$

Proof. First, we look at the numerator of $R_{K,\mathbf{y}_J,\beta}$, see (3.1), and rearrange it to

$$\begin{aligned}
&\exp\Big(-\frac{\beta}{2}\|\mathbf{y}_J(\mathbf{x}_{K,\mathbf{i}}) - \mathbf{t}_n\|^2\Big) \\
&= \exp\Big(-\frac{\beta}{2}\Big\|\sum_{d=1}^{D} g_J^{(d)}\big(x_{K,i^{(m(d))}}\big)\mathbf{v}^{(d)} - \mathbf{t}_n\Big\|^2\Big) \\
&= \exp\Big(-\frac{\beta}{2}\langle \mathbf{t}_n, \mathbf{t}_n\rangle\Big) \cdot \exp\Big(\beta\sum_{d=1}^{D} g_J^{(d)}\big(x_{K,i^{(m(d))}}\big)\big\langle \mathbf{v}^{(d)}, \mathbf{t}_n\big\rangle\Big) \\
&\quad \cdot \exp\Big(-\frac{\beta}{2}\sum_{d=1}^{D}\sum_{d'=1}^{D} g_J^{(d)}\big(x_{K,i^{(m(d))}}\big) g_J^{(d')}\big(x_{K,i^{(m(d'))}}\big)\big\langle \mathbf{v}^{(d)}, \mathbf{v}^{(d')}\big\rangle\Big) . \qquad (3.10)
\end{aligned}$$

The orthonormality property of the $\big(\mathbf{v}^{(d)}\big)_{d=1}^{D}$ removes any terms with $d \neq d'$ in (3.10). The term $\exp\big(-\frac{\beta}{2}\langle \mathbf{t}_n, \mathbf{t}_n\rangle\big)$ appears in the nominator and denominator of (3.1), so it cancels out. Thus, we obtain

$$\begin{aligned}
&R_{K,\mathbf{y}_J,\beta}(\mathbf{t}_n, \mathbf{x}_{K,\mathbf{i}}) \\
&= \frac{\exp\Big(-\frac{\beta}{2}\sum_{d=1}^{D} g_J^{(d)}\big(x_{K,i^{(m(d))}}\big)^2 + \beta\sum_{d=1}^{D} g_J^{(d)}\big(x_{K,i^{(m(d))}}\big)\big\langle \mathbf{v}^{(d)}, \mathbf{t}_n\big\rangle\Big)}{\sum_{\mathbf{j}\in\chi_K^L}\omega_{K,\mathbf{j}}\exp\Big(-\frac{\beta}{2}\sum_{d=1}^{D} g_J^{(d)}\big(x_{K,j^{(m(d))}}\big)^2 + \beta\sum_{d=1}^{D} g_J^{(d)}\big(x_{K,j^{(m(d))}}\big)\big\langle \mathbf{v}^{(d)}, \mathbf{t}_n\big\rangle\Big)} \\
&= \frac{\prod_{d=1}^{D}\exp\Big(-\frac{\beta}{2} g_J^{(d)}\big(x_{K,i^{(m(d))}}\big)^2 + \beta g_J^{(d)}\big(x_{K,i^{(m(d))}}\big)\big\langle \mathbf{v}^{(d)}, \mathbf{t}_n\big\rangle\Big)}{\sum_{\mathbf{j}\in\chi_K^L}\omega_{K,\mathbf{j}}\prod_{d=1}^{D}\exp\Big(-\frac{\beta}{2} g_J^{(d)}\big(x_{K,j^{(m(d))}}\big)^2 + \beta g_J^{(d)}\big(x_{K,j^{(m(d))}}\big)\big\langle \mathbf{v}^{(d)}, \mathbf{t}_n\big\rangle\Big)} \\
&= \prod_{l=1}^{L}\frac{\prod_{d\in m^{-1}(l)}\exp\Big(-\frac{\beta}{2} g_J^{(d)}\big(x_{K,i^{(m(d))}}\big)^2 + \beta g_J^{(d)}\big(x_{K,i^{(m(d))}}\big)\big\langle \mathbf{v}^{(d)}, \mathbf{t}_n\big\rangle\Big)}{\sum_{j\in\chi_K}\omega_{K,j}\prod_{d\in m^{-1}(l)}\exp\Big(-\frac{\beta}{2} g_J^{(d)}(x_{K,j})^2 + \beta g_J^{(d)}(x_{K,j})\big\langle \mathbf{v}^{(d)}, \mathbf{t}_n\big\rangle\Big)} .
\end{aligned}$$

In the last line, we separated the D factors of $\prod_{d=1}^{D}$ ... into L groups $(m^{-1}(l))_{l=1}^{L}$ and exploited the tensor product structure (2.13) of the quadrature rule. ∎

Lemma 3.4 states that the minimum in the first parameter of $\mathcal{G}_{N,K}$ can be computed and stored by precomputing only the one-dimensional functions $R^{(l)}_{K,\mathbf{y}_J,\beta}, l = 1, \dots, L$, from (3.9) for every $x_{K,i}, i \in \chi_K$, and every $\mathbf{t}_n, n = 1, \dots, N$. In total, the amount of storage is $\mathcal{O}(L \cdot N \cdot m_K)$ and the cost of precomputation is $\mathcal{O}(D \cdot N \cdot m_K)$.

Note that analogously to

$$\sum_{\mathbf{i}\in\chi_K^L}\omega_{K,\mathbf{i}} R_{K,\mathbf{y}_J,\beta}(\mathbf{t}_n, \mathbf{x}_{K,\mathbf{i}}) = 1 \qquad (3.11)$$

it also holds that

$$\sum_{i\in\chi_K}\omega_{K,i} R^{(l)}_{K,\mathbf{y}_J,\beta}(\mathbf{t}_n, x_{K,i}) = 1 \qquad (3.12)$$

for all $l = 1, \dots, L$ and $n = 1, \dots, N$.

3.2. Minimization with respect to the second parameter $\mathbf{y}_J$. To minimize $\mathcal{G}_{N,K}$ with respect to $\mathbf{y}_J$, we need to solve the quadratic regression type problem

$$\begin{aligned}\mathbf{y}_J^{\text{new}} &= \arg \min_{\mathbf{y}_J \in V_{\text{PCGTM},J}} \mathcal{G}_{N,K}(R_{K,\mathbf{y}_J^{\text{old}},\beta}, \mathbf{y}_J, \beta) \\ &= \arg \min_{\mathbf{y}_J \in V_{\text{PCGTM},J}} \frac{1}{N} \sum_{n=1}^{N} \sum_{\mathbf{i} \in \chi_K^L} \omega_{K,\mathbf{i}} R_{K,\mathbf{y}_J^{\text{old}},\beta}(\mathbf{t}_n, \mathbf{x}_{K,\mathbf{i}}) \|\mathbf{y}_J(\mathbf{x}_{K,\mathbf{i}}) - \mathbf{t}_n\|^2 \, . \end{aligned} \tag{3.13}$$

In order to compute (3.13) for given β and $R_{K,\mathbf{y}_J^{\text{old}},\beta}$, we have to use the representations (2.9) and (2.10) for $\mathbf{y}_J \in V_{\text{PCGTM},J}$. Then, the problem reads

$$\arg \min_{\left(\bar{\alpha}_J^{(d)}\right)_{d=1}^{D}} \frac{1}{N} \sum_{n=1}^{N} \sum_{\mathbf{i} \in \chi_K^L} \omega_{K,\mathbf{i}} R_{K,\mathbf{y}_J^{\text{old}},\beta}(\mathbf{t}_n, \mathbf{x}_{K,\mathbf{i}}) \Big\| \sum_{d=1}^{D} \sum_{j=1}^{n_J} \alpha_{J,j}^{(d)} \phi_j\big(x_{K,i^{(m(d))}}\big) \mathbf{v}^{(d)} - \mathbf{t}_n \Big\|^2 , \tag{3.14}$$

which can be tackled efficiently as shown by the following Lemma.

Lemma 3.5. *The problem* (3.14) *results in D decoupled systems of linear equations*

$$\mathbf{A}^{(d)} \bar{\alpha}_J^{(d)} = \mathbf{b}^{(d)} \tag{3.15}$$

for $d = 1, \dots, D$ with $\bar{\alpha}_J^{(d)} = \big(\alpha_{J,1}^{(d)}, \dots, \alpha_{J,n_J}^{(d)}\big)^T \in \mathbb{R}^{n_J}$, $\mathbf{A}^{(d)} \in \mathbb{R}^{n_J \times n_J}$ with

$$(\mathbf{A}^{(d)})_{jk} = \sum_{i \in \chi_K} \omega_{K,i} \Big(\frac{1}{N} \sum_{n=1}^{N} R_{K,\mathbf{y}_J^{old},\beta}^{(m(d))}(\mathbf{t}_n, x_{K,i}) \Big) \phi_j(x_{K,i}) \phi_k(x_{K,i})$$

for $j, k = 1, \dots, n_J$, and $\mathbf{b}_d \in \mathbb{R}^{n_J}$ with

$$(\mathbf{b}^{(d)})_k = \sum_{i \in \chi_K} \omega_{K,i} \Big(\frac{1}{N} \sum_{n=1}^{N} R_{K,\mathbf{y}_J^{old},\beta}^{(m(d))}(\mathbf{t}_n, x_{K,i}) \big\langle \mathbf{v}^{(d)}, \mathbf{t}_n \big\rangle \Big) \phi_k(x_{K,i})$$

for $k = 1, \dots, n_J$.

Proof. A critical point of (3.14) satisfies

$$\begin{aligned} &\frac{\partial}{\partial \alpha_{J,k}^{(d)}} \frac{1}{N} \sum_{n=1}^{N} \sum_{\mathbf{i} \in \chi_K^L} \omega_{K,\mathbf{i}} R_{K,\mathbf{y}_J^{\text{old}},\beta}(\mathbf{t}_n, \mathbf{x}_{K,\mathbf{i}}) \\ &\qquad \cdot \Big\| \sum_{d'=1}^{D} \sum_{j=1}^{n_J} \alpha_{J,j}^{(d')} \phi_j\big(x_{K,i^{(m(d'))}}\big) \mathbf{v}^{(d')} - \mathbf{t}_n \Big\|^2 \overset{!}{=} 0 \end{aligned}$$

$$\Leftrightarrow \frac{\partial}{\partial \alpha_{J,k}^{(d)}} \frac{1}{N} \sum_{n=1}^{N} \sum_{\mathbf{i} \in \chi_K^L} \omega_{K,\mathbf{i}} R_{K,\mathbf{y}_J^{\text{old}},\beta}(\mathbf{t}_n, \mathbf{x}_{K,\mathbf{i}}) \cdot \Big(\sum_{d'=1}^{D} \Big(\sum_{j=1}^{n_J} \alpha_{J,j}^{(d')} \phi_j\big(x_{K,i^{(m(d'))}}\big) \Big)^2 \tag{3.16}$$

$$- 2 \sum_{d'=1}^{D} \sum_{j=1}^{n_J} \alpha_{J,j}^{(d')} \phi_j\big(x_{K,i^{(m(d'))}}\big) \big\langle \mathbf{v}^{(d')}, \mathbf{t}_n \big\rangle + \|\mathbf{t}_n\|^2 \Big) = 0 \tag{3.17}$$

for $d = 1, \dots, D$ and $k = 1, \dots, n_J$, where we have again used the orthonormality property of the $\big(\mathbf{v}^{(d)}\big)_{d=1}^{D}$ in (3.16) and (3.17). Then, carrying out the differentiation yields

$$\frac{1}{N}\sum_{n=1}^{N}\sum_{\mathbf{i}\in\chi_K^L}\omega_{K,\mathbf{i}}R_{\mathbf{y}_J^{\text{old}},\beta}(\mathbf{t}_n,\mathbf{x}_{K,\mathbf{i}})\Big(2\sum_{j=1}^{n_J}\alpha_{J,j}^{(d)}\phi_j\big(x_{K,i^{(m(d))}}\big)\phi_k\big(x_{K,i^{(m(d))}}\big)$$

$$-2\phi_k\big(x_{K,i^{(m(d))}}\big)\big\langle\mathbf{v}^{(d)},\mathbf{t}_n\big\rangle\Big)=0$$

$$\Leftrightarrow\sum_{j=1}^{n_J}\Bigg(\frac{1}{N}\sum_{n=1}^{N}\sum_{\mathbf{i}\in\chi_K^L}\omega_{K,\mathbf{i}}R_{K,\mathbf{y}_J^{\text{old}},\beta}\big(\mathbf{t}_n,\mathbf{x}_{K,\mathbf{i}}\big)\cdot\phi_j\big(x_{K,i^{(m(d))}}\big)\phi_k\big(x_{K,i^{(m(d))}}\big)\Bigg)\alpha_{J,j}^{(d)}$$

$$=\frac{1}{N}\sum_{n=1}^{N}\sum_{\mathbf{i}\in\chi_K^L}\omega_{K,\mathbf{i}}R_{K,\mathbf{y}_J^{\text{old}},\beta}(\mathbf{t}_n,\mathbf{x}_{K,\mathbf{i}})\phi_k\big(x_{K,i^{(m(d))}}\big)\big\langle\mathbf{v}^{(d)},\mathbf{t}_n\big\rangle\,.$$

As there appear no $\alpha_{J,j}^{(d')}$ with $d'\neq d$, we can obviously determine all $\bar{\alpha}_J^{(d)}, d=1,\ldots,D$, by solving the separate systems (3.15) of linear equations with n_J unknowns, where

$$(\mathbf{A}^{(d)})_{kj}=\frac{1}{N}\sum_{n=1}^{N}\sum_{\mathbf{i}\in\chi_K^L}\omega_{K,\mathbf{i}}R_{K,\mathbf{y}_J^{\text{old}},\beta}(\mathbf{t}_n,\mathbf{x}_{K,\mathbf{i}})\phi_j\big(x_{K,i^{(m(d))}}\big)\phi_k\big(x_{K,i^{(m(d))}}\big)$$

$$(3.18)\qquad=\frac{1}{N}\sum_{n=1}^{N}\sum_{\mathbf{i}\in\chi_K^L}\Big(\prod_{l=1}^{L}\omega_{K,i^{(l)}}R_{K,\mathbf{y}_J^{\text{old}},\beta}^{(l)}\big(\mathbf{t}_n,x_{K,i^{(l)}}\big)\Big)\phi_j\big(x_{K,i^{(m(d))}}\big)\phi_k\big(x_{K,i^{(m(d))}}\big)$$

$$(3.19)\qquad=\frac{1}{N}\sum_{n=1}^{N}\sum_{i\in\chi_K}\omega_{K,i}R_{K,\mathbf{y}_J^{\text{old}},\beta}^{(m(d))}(\mathbf{t}_n,x_{K,i})\phi_j(x_{K,i})\phi_k(x_{K,i})$$

for $k,j=1,\ldots,n_J$ and

$$(\mathbf{b}^{(d)})_k=\frac{1}{N}\sum_{n=1}^{N}\sum_{\mathbf{i}\in\chi_K^L}\omega_{K,\mathbf{i}}R_{K,\mathbf{y}_J^{\text{old}},\beta}(\mathbf{t}_n,\mathbf{x}_{K,\mathbf{i}})\phi_k\big(x_{K,i^{(m(d))}}\big)\big\langle\mathbf{v}^{(d)},\mathbf{t}_n\big\rangle$$

$$(3.20)\qquad=\frac{1}{N}\sum_{n=1}^{N}\sum_{\mathbf{i}\in\chi_K^L}\Big(\prod_{l=1}^{L}\omega_{K,i^{(l)}}R_{K,\mathbf{y}_J^{\text{old}},\beta}^{(l)}\big(\mathbf{t}_n,x_{K,i^{(l)}}\big)\Big)\phi_k\big(x_{K,i^{(m(d))}}\big)\big\langle\mathbf{v}^{(d)},\mathbf{t}_n\big\rangle$$

$$(3.21)\qquad=\frac{1}{N}\sum_{n=1}^{N}\sum_{i\in\chi_K}\omega_{K,i}R_{K,\mathbf{y}_J^{\text{old}},\beta}^{(m(d))}\big(\mathbf{t}_n,x_{K,i}\big)\phi_k(x_{K,i})\big\langle\mathbf{v}^{(d)},\mathbf{t}_n\big\rangle$$

for $k=1,\ldots,n_J$. In (3.18) and (3.20), we have used the tensor product structure of the quadrature rule (2.13) and the separability of posterior probabilities (3.9). Then, in (3.19) and (3.21), we have used (3.12) for all dimensions $l\neq m(d)$. □

The set up of the systems (3.15) can easily be done with computational costs of no more than $\mathcal{O}(D\cdot N\cdot m_k)$ floating point operations. As the matrices $\mathbf{A}^{(d)}, d=1,\ldots,D$, are tridiagonal for linear splines, the solution of the systems (3.15) is possible with a complexity of $\mathcal{O}(D\cdot n_J)$ floating point operations.

3.3. Minimization with respect to the β-parameter. The minimization with respect to β is simple. We are given $\mathbf{y}_J$ and $R_{\mathbf{y}_J,\beta^{\text{old}}}$, and we want to determine an optimal β^{new}, i.e.,

$$\frac{\partial}{\partial\beta}\mathcal{G}_{N,K}(R_{K,\mathbf{y}_J,\beta^{\text{old}}},\mathbf{y}_J,\beta^{\text{new}}) \overset{!}{=} 0$$

$$\Leftrightarrow(\beta^{\text{new}})^{-1} = \frac{1}{DN}\sum_{n=1}^{N}\sum_{\mathbf{i}\in\chi_K^L}\omega_{K,\mathbf{i}}R_{K,\mathbf{y}_J,\beta^{\text{old}}}(\mathbf{t}_n,\mathbf{x}_{K,\mathbf{i}})\|\mathbf{y}_J(\mathbf{x}_{K,\mathbf{i}})-\mathbf{t}_n\|^2\ . \tag{3.22}$$

The computation of the right-hand side of (3.22) can be done efficiently using the separated representation of the posterior probabilities (3.9) as the following Lemma shows.

LEMMA 3.6. *It holds that*

$$\begin{aligned}(\beta^{new})^{-1} =&\frac{1}{DN}\sum_{n=1}^{N}\sum_{d=1}^{D}\sum_{i\in\chi_K}\omega_{K,i}R^{(m(d))}_{K,\mathbf{y}_J,\beta^{old}}(\mathbf{t}_n,x_{K,i})\\ &\cdot\Big(g_J^{(d)}(x_{K,i})^2-2g_J^{(d)}(x_{K,i})\big\langle\mathbf{v}^{(d)},\mathbf{t}_n\big\rangle\Big)+\frac{1}{DN}\sum_{n=1}^{N}\langle\mathbf{t}_n,\mathbf{t}_n\rangle\ .\end{aligned}\tag{3.23}$$

Proof. By plugging (2.13) and (3.9) into (3.22), we obtain

$$\begin{aligned}(\beta^{\text{new}})^{-1} =& \frac{1}{DN}\sum_{n=1}^{N}\sum_{\mathbf{i}\in\chi_K^L}\left(\prod_{l=1}^{L}\omega_{K,i^{(l)}}R^{(l)}_{K,\mathbf{y}_J,\beta^{\text{old}}}\big(\mathbf{t}_n,x_{K,i^{(l)}}\big)\right)\\ &\cdot\left\langle\sum_{d=1}^{D}g_J^{(d)}\big(x_{K,i^{(m(d))}}\big)\mathbf{v}^{(d)}-\mathbf{t}_n,\sum_{d=1}^{D}g_J^{(d)}\big(x_{K,i^{(m(d))}}\big)\mathbf{v}^{(d)}-\mathbf{t}_n\right\rangle\ .\end{aligned}$$

We use the orthogonality property of the $\big(\mathbf{v}^{(d)}\big)_{d=1}^{D}$ and (3.11) to arrive at

$$\begin{aligned}(\beta^{\text{new}})^{-1} =&\frac{1}{DN}\sum_{n=1}^{N}\sum_{\mathbf{i}\in\chi_K^L}\left(\prod_{l=1}^{L}\omega_{K,i^{(l)}}R^{(l)}_{K,\mathbf{y}_J,\beta^{\text{old}}}\big(\mathbf{t}_n,x_{i^{(l)}}\big)\right)\\ &\cdot\left(\sum_{d=1}^{D}g_J^{(d)}(x_{i^{(m(d))}})^2-2\sum_{d=1}^{D}g_J^{(d)}(x_{i^{(m(d))}})\left\langle\mathbf{v}^{(d)},\mathbf{t}_n\right\rangle\right)+\frac{1}{DN}\sum_{n=1}^{N}\langle\mathbf{t}_n,\mathbf{t}_n\rangle\end{aligned}\tag{3.24}$$

A reordering of the summations $\sum_{d=1}^{D}\dots$ and $\sum_{\mathbf{i}\in\chi_K^L}\dots$, and the application of (3.12) for any $l\neq m(d)$ to (3.24) results in

$$\begin{aligned}(\beta^{\text{new}})^{-1} =& \frac{1}{DN}\sum_{n=1}^{N}\sum_{d=1}^{D}\sum_{i\in\chi_K}\omega_{K,i}R^{(m(d))}_{K,\mathbf{y}_J,\beta^{\text{old}}}(\mathbf{t}_n,x_{K,i})\\ &\cdot\left(g_J^{(d)}(x_{K,i})^2-2g_J^{(d)}(x_{K,i})\left\langle\mathbf{v}^{(d)},\mathbf{t}_n\right\rangle\right)+\frac{1}{DN}\sum_{n=1}^{N}\langle\mathbf{t}_n,\mathbf{t}_n\rangle\ .\quad\square\end{aligned}\tag{3.25}$$

Note that in this step all dimensions interact, so our model does not decompose into independent problems. The optimization in β^{new} results in computational costs of $\mathcal{O}(D\cdot N\cdot m_K)$ floating point operations.

3.4. Complexity discussion. We now discuss the total computational cost of the PCGTM method. The initialization step requires the set up of the sample covariance matrix, which has computational costs of $\mathcal{O}(N \cdot D^2)$ floating point operations. We assume that the cost of computing non-linear correlation coefficients for determining the mapping $m : \{1, \dots, D\} \to \{1, \dots, L\}$ is essentially the same, e.g., up to a log-term in N. The computation of the principal component vectors $\left(\mathbf{v}^{(d)}\right)_{d=1}^{D}$ needs $\mathcal{O}(D^3)$ floating point operations.

Then, the minimization of $\mathcal{G}_{N,K}$ has to be carried out. In the Subsections 3.1, 3.2 and 3.3 we have described how the minimization steps of $\mathcal{G}_{N,K}$ can be implemented where none of them costs more than $\mathcal{O}(D \cdot N \cdot m_K)$ operations. Of course, the minimization steps have to be repeated multiple times, so for #it iterations we arrive at total costs of

$$\mathcal{O}(N \cdot D^2 + D^3 + \#it \cdot D \cdot N \cdot m_K) \, .$$

In our experiments, #it showed to be never larger than 50. It is noteworthy that our method is only polynomial in the data space dimension and essentially linear in the number of data points. Moreover, the cost of the L-dimensional quadrature rule (2.12) appears only with the cost $\mathcal{O}(m_K)$ of the one-dimensional rule (2.11), and, thus, there is no curse of dimension involved.

4. Numerical experiments. In this section, we demonstrate the abilities of the PCGTM in terms of data reconstruction and classification. We first use three-dimensional synthetic datasets and then consider high-dimensional real-world data.

The average reconstruction error of the PCGTM is given by

$$\frac{1}{N} \sum_{n=1}^{N} \left\| \mathbf{y}_J\left(E_{K,\mathbf{y}_J,\beta} \mathbf{t}_n\right) - \mathbf{t}_n \right\| \, , \tag{4.1}$$

where $E_{K,\mathbf{y}_J,\beta} : \mathbb{R}^D \to [0,1]^L$ is an embedding with

$$E_{K,\mathbf{y}_J,\beta} : \mathbf{t} \mapsto \mathbf{x}_{K,\mathbf{i}} \quad \text{with} \quad \mathbf{i} = \arg\max_{\mathbf{j} \in \chi_K^L} R_{K,\mathbf{y}_J,\beta}(\mathbf{t}, \mathbf{x}_{K,\mathbf{j}}) \, .$$

In order to rule out that our model suffers from overfitting, we split our data in training and test data and evaluate the reconstruction errors separately. We compare (4.1) to the average reconstruction error of the PCA. To this end, we do not use the probabilistic PCA [24], but a simple diagonalization of the sample covariance matrix $\mathbf{C}$. Recall from Section 2 that

$$\mathbf{C} = \mathbf{V}\mathbf{D}\mathbf{V}^T \, .$$

We choose the rectangular matrix $\mathbf{I}_L \in \mathbb{R}^{D \times L}$ with $(\mathbf{I})_{dl} = \delta_{dl}$ for $d = 1, \dots, D$ and $l = 1, \dots, L$. Then, the projection of $\mathbf{t} \in \mathbb{R}^D$ onto the first L eigenvectors is

$$\mathbf{I}_L^T \mathbf{V}^T \mathbf{t} \in \mathbb{R}^L \, ,$$

and the mean reconstruction error of the PCA is given by

$$\frac{1}{N} \sum_{n=1}^{N} \left\| \mathbf{V}\mathbf{I}_L\mathbf{I}_L^T\mathbf{V}^T\mathbf{t}_n - \mathbf{t}_n \right\| \, .$$

For quadrature, we always choose $K = J + 3$ and a simple midpoint rule. This ensures a fixed amount of quadrature points in the support of every basis function. We initialize the PCGTM mapping $\mathbf{y}_J$ by linear functions $g_J^{(d)}, d = 1, \ldots, D$, such that the initial PCGTM model, i.e., before carrying out any minimization of $\mathcal{G}_{N,K}$, resembles approximately the PCA model. We denote the initial value of β by β_0 and choose here some value which has the same magnitude as the number that minimizes $\mathcal{G}_{N,K}$.

4.1. Synthetic datasets. In this subsection, we deal with one- and two-dimensional structures in the three-dimensional space which are frequently used as test cases in the literature, cf. [18].

4.1.1. Helix. We randomly generate 5000 points on a helix-structure, distort them with isotropic Gaussian noise and split them in $N_{\text{train}} = 3316$ training data points and $N_{\text{test}} = 1684$ test data points. We fix the PCGTM parameters $J = 5$ and $\beta_0 = 5$. The matching of principal components to latent space dimensions is done according to (2.8) with Spearman's rank correlation coefficient. This results in $m(d) = 1$ for $d = 1, 2, 3$ in the case $L = 1$ and $m(1) = 1, m(2) = 2, m(3) = 1$ in the case $L = 2$. Of course, $L = 1$ is the correct choice and the overestimation of the embedding dimension with $L = 2$ leads to an unnecessarily complex model.

In Fig. 4.1, we show the structure of the PCGTM models after 50 iterations on the training data for $L = 1$ and $L = 2$. A comparison of the reconstruction error of the PCGTM and the PCA for the training data and the test data is given in Fig. 4.2. We observe that the PCGTM model has a significantly lower reconstruction error than the PCA with the same latent space dimension. Of course, the PCA error is reduced by adding a second latent space dimension, whereas the PCGTM model reconstruction error stays roughly the same due to the fact the helix structure has already been learned for $L = 1$. The stability of the errors on training and test data indicates that there is no overfitting.

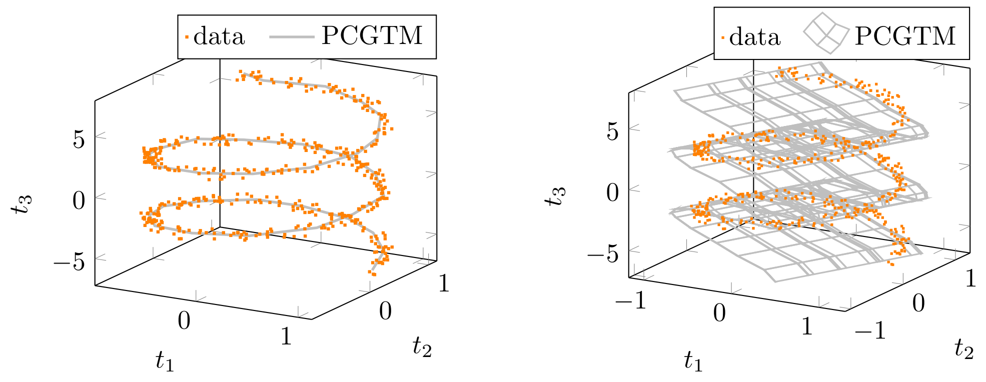

FIG. 4.1. *The helix structure (number of points reduced for visualization only) and the image of* $[0,1]^L$ *under the PCGTM mapping* $\mathbf{y}_J$ *for* $L = 1$ *(left) and* $L = 2$ *(right)*

4.1.2. Swiss roll. We repeat the above experiment with the two-dimensional Swiss roll structure in the three-dimensional space. The parameters are $J = 4$ and $\beta_0 = 1$. Figure 4.3 shows the PCGTM models for $L = 1$ and $L = 2$ after 50 iterations. We observe that $L = 1$ leads to a rather poor model, whereas the PCGTM model with $L = 2$ is able to almost correctly learn the two-dimensional structure of the Swiss roll. Accordingly, the reconstruction error is considerably reduced by using a

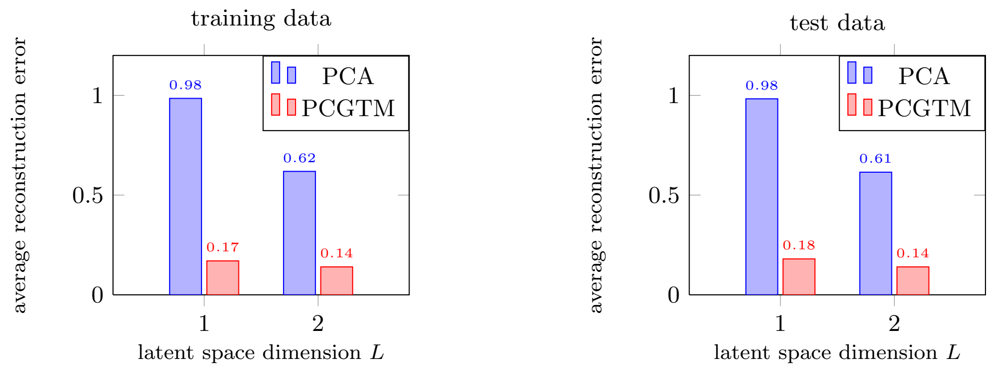

FIG. 4.2. *The average reconstruction error for the helix training and test dataset with the one- and two-dimensional PCA and PCGTM models*

PCGTM model with $L = 2$, see Fig. 4.4, whereas the reconstruction error of the PCA is still much larger. This reflects the fact that the non-linear structure of the Swiss roll cannot be recovered well using a purely linear approach.

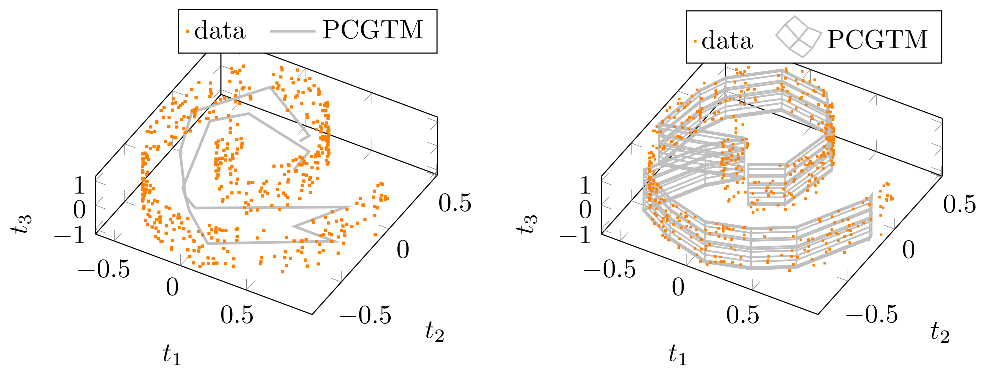

FIG. 4.3. *The Swiss roll (number of data points reduced for visualization only) and the image of* $[0, 1]^L$ *under the PCGTM mapping* $\mathbf{y}_J$ *for* $L = 1$ *(left) and* $L = 2$ *(right)*

4.2. Reconstruction error on real datasets. In this subsection, we consider real-world datasets. For $L = 1, 2, 3$ we complement our results with the reconstruction errors of the original GTM [4] with a linear spline tensor product basis. When possible, the parameters employed for the GTM are the same as for the PCGTM.

4.2.1. Wine quality dataset. We first analyze the wine quality dataset [7] available at the UCI Machine Learning Repository [1]. In particular, we look at the white wines with 4898 instances and 11 physicochemical inputs. There is one sensory output variable, the wine quality, which we simply append to the inputs and regard this data as 12-dimensional. We split the data randomly in $N_{\text{train}} = 3243$ and $N_{\text{test}} = 1655$ test data points and choose the PCGTM parameters $J = 8$ and $\beta_0 = 0.05$. The mappings m are determined for all $L = 1, \ldots, 6$ according to (2.8).

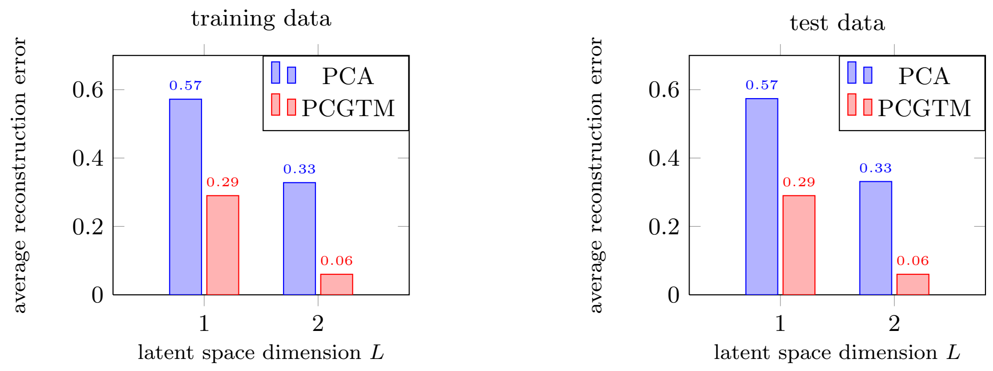

FIG. 4.4. *The average reconstruction error for the Swiss roll training and test dataset with the one- and two-dimensional PCA and PCGTM models*

For example, for $L = 6$ this leads to

$$m(d) = \begin{cases} 1 \quad \text{for } d = 1 & 4 \quad \text{for } d = 4 & 4 \quad \text{for } d = 7 & 6 \quad \text{for } d = 10 \\ 2 \quad \text{for } d = 2 & 5 \quad \text{for } d = 5 & 3 \quad \text{for } d = 8 & 4 \quad \text{for } d = 11 \\ 3 \quad \text{for } d = 3 & 6 \quad \text{for } d = 6 & 2 \quad \text{for } d = 9 & 3 \quad \text{for } d = 12\,. \end{cases}$$

We use 15 iterations for the minimization of $\mathcal{G}_{N,K}$ to determine a good $\mathbf{y}_J$ and β.

As in the previous experiments, we measure the reconstruction error of the data after the embedding and subsequent application of $\mathbf{y}_J$ to the data. The results for the training and test data can be seen in Fig. 4.5.

We see that the reconstruction error of the original GTM and the PCGTM is roughly the same for $L = 1$, which is what we would expect, as both methods offer essentially the same flexibility. Due to memory constraints, we have to reduce the discretization level of the original GTM for $L = 2$ and $L = 3$ to $J = 6$ and $J = 4$, respectively, which leads to deteriorating classification results compared to the PCGTM even though the number of degrees of freedom of the original GTM is larger by more than a factor of 10.

Again, we observe that the PCGTM offers a significant reduction of the reconstruction error compared to the PCA. Note however that, for $L = 6$, the PCA error is smaller. This is due to the fact that it goes analytically to zero for $L \to D$, whereas the PCGTM model is affected by discretization errors and possibly a local minimum. We do not consider this to be a problem since here the latent space dimension is so high, i.e., $L = D/2$, that already the reconstruction error of a purely linear method is negligible. In the next experiment, we see that the number of latent space dimensions for which the PCGTM outperforms the PCA grows with the dimensionality of the data.

4.2.2. Libras hand movement dataset. We now analyze the Libras hand movement dataset [9] which is also available at the UCI Machine Learning Repository [1]. It consists of 15 classes with 24 instances, i.e., 360 data points in total. Every data point has 91 attributes and describes one hand movement in LIBRAS, which is the official Brazilian sign language. We do not perform a classification but, similar to the wine data, compare the reconstruction errors. In the experimental setting, we split the data in $N_{\text{train}} = 251$ training data and $N_{\text{test}} = 109$ test data points. As PCGTM parameters we choose $J = 6$ and $\beta_0 = 35$ for the latent space dimensions

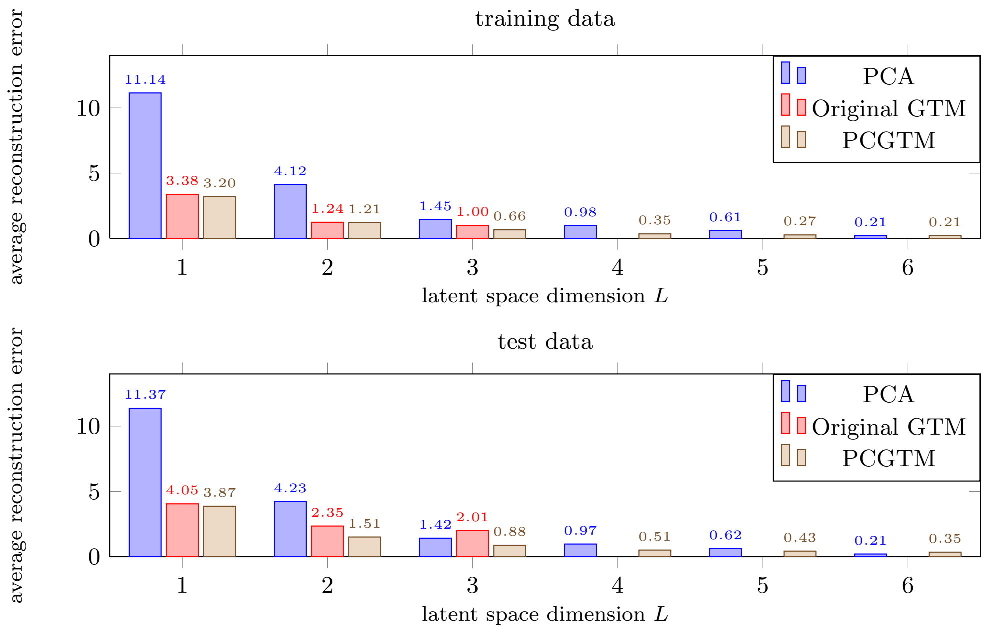

FIG. 4.5. *The average reconstruction error for the wine quality training and test dataset with the original GTM for $L = 1, 2, 3$ and the PCA and PCGTM models for $L = 1, \dots, 6$*

$L = 1, \dots, 10$. After 5 iterations, we measure the reconstruction error.

Figure 4.6 reveals a moderate amount of overfitting for the PCGTM, since the test data reconstruction errors are slightly larger than the corresponding errors on the training data. Furthermore, we observe that the PCGTM model has a lower error than the PCA at least up to latent dimension $L = 10$. For complexity reasons, the discretization level of the original GTM needs to be reduced to $J = 4$ for $L = 3$. For $L = 2$ and $L = 3$ we employ a H^1-seminorm regularization with $\lambda = 0.2$ and $\lambda = 1.0$, respectively, to reduce the amount of overfitting. Still, the larger number of degrees of freedoms leads to lower reconstruction errors on the training data than for the PCGTM, but on the test data the results are slightly worse.

In conclusion this experiment shows that the restrictive PCGTM still captures relevant features of the data and is also superior to the PCA in terms of the reconstruction error in cases that cannot be treated with the original GTM anymore.

4.3. Classification: Connectionist bench. In this subsection we finally use the PCGTM for classification. We can easily turn our unobserved learning method into an observed one by reconstructing missing values. We achieve this by appending a class variable $c_n \in \{-1, 1\}$ to the data points by

$$\mathbf{t}'_n := \left(t_n^{(1)}, \dots, t_n^{(D)}, c_n\right)^T \quad \text{for} \quad n = 1, \dots, N\,. \tag{4.2}$$

We first use the PCGTM to fit the mapping $\mathbf{y}_J$ and the inverse variance β to these points. Then, we can classify previously unknown data points with help of the density $q_{\mathbf{y}_J,\beta}$, see (2.2), by

$$c(\mathbf{t}) := \begin{cases} 1 & \text{if } q_{\mathbf{y},\beta}((t^{(1)}, \dots, t^{(D)}, 1)^T) \geq q_{\mathbf{y},\beta}((t^{(1)}, \dots, t^{(D)}, -1)^T)\,, \\ -1 & \text{else}\,. \end{cases}$$

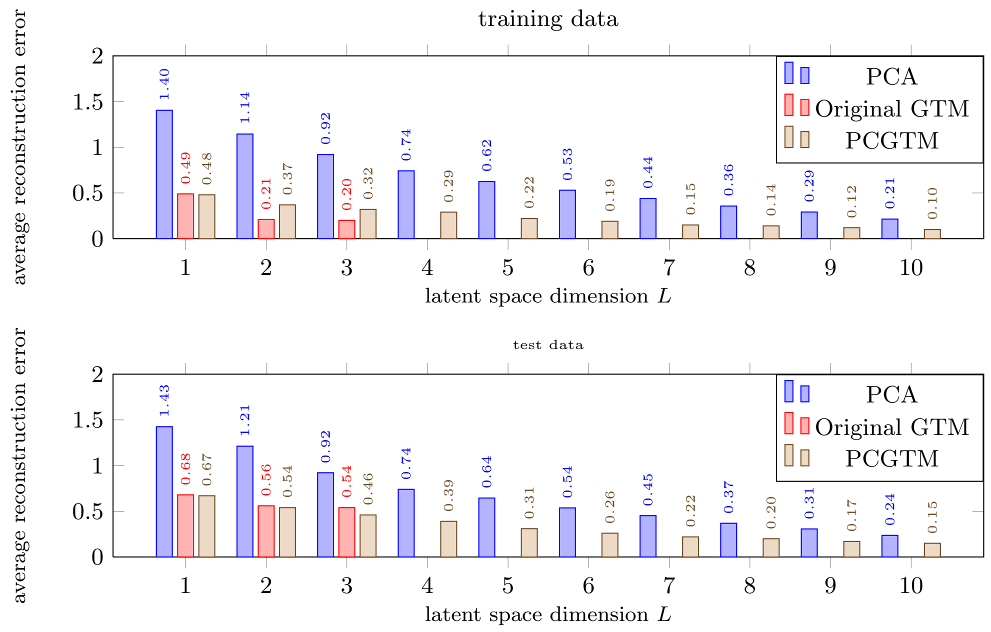

FIG. 4.6. *The average reconstruction error of the Libras training and test dataset with the original GTM for $L = 1, 2, 3$ and the PCA and PCGTM models for $L = 1, \ldots, 10$*

We apply this technique to the 'Connectionist Bench (Sonar, Mines vs. Rocks)', a real-world dataset from the UCI Machine Learning Repository [1]. It consists of 208 measurements with 60 dimensions and two class labels $\{-1, 1\}$. The classes are roughly equally distributed, i.e., 97 data points have class 1 and 111 data points have class -1.

In [11], this data was randomly partitioned in 13 blocks with 16 data points each, which were then used to create test cases with one block serving as test data and the remaining 12 blocks as training data. In the original paper, the best neuronal networks achieved an average classification rate of 84.7%.

In order to reduce the influence of the way we split the data in training and test data, we independently generate 50 random test cases with $N_{\text{train}} = 192$ and $N_{\text{test}} = 16$. Then, we set $\beta_0 = 5$ and $J = 5$ and perform a classical cross-validation to determine the optimal amount of iteration steps $S = 1, 2, 3$ and the correct number of latent space dimensions $L = 1, \ldots, 10$ for our PCGTM model. We have to exclude 4 of the 30 cross-validation runs for which overfitting results in indefinite systems in (3.15). The remaining 26 runs achieve on average a 77.6% correctness rate on the training data and a 71.2% correctness rate on the test data with a linear correlation coefficient of $\rho = 0.94$. This demonstrates that a good result on the training data is an indicator for a good result on the test data. Indeed, the two highest results on the training data were 88.9% ($\sigma = 9.2$) and 87.4% ($\sigma = 8.5$) for $L = 1, S = 3$ and $L = 2, S = 3$, respectively, and their results on the test data were 78.4% and 82.3%, respectively. These results are roughly comparable to the performance of the best neuronal networks in the original paper [11].

5. Conclusion.

We presented the PCGTM which can be used for dimensionality reduction and classification of high-dimensional data. The model is fairly comprehensible and easy to compute, i.e., it can be implemented with costs that are polynomial in the data space dimension D and linear in N. However, it is still able to capture some non-linearities and leads to substantially smaller reconstruction errors than the PCA. In summary, we advocate the PCGTM instead of the PCA as a preprocessing step for latent space dimensions $L > 3$, where grid based methods like the original GTM suffer from the curse of dimensionality. For further experiments and results, cf. [14].

REFERENCES

[1] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[2] R. Bellman. *Adaptive Control Processes: A Guided Tour.* Princeton University Press, 1961.

[3] C. Bishop, M. Svensén, and C. Williams. Developments of the generative topographic mapping. *Neurocomputing*, 21:203–224, 1998.

[4] C. Bishop, M. Svensen, and C. Williams. GTM: The generative topographic mapping. *Neural Computation*, 10(1):215–234, 1998.

[5] A. Buja, T. Hastie, and R. Tibshirani. Linear smoothers and additive models. *The Annals of Statistics*, 17:453–510, 1989.

[6] P. Comon. Independent component analysis, a new concept? *Signal Process.*, 36(3):287–314, 1994.

[7] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547 – 553, 2009.

[8] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 1977.

[9] D. Dias, R. Madeo, T. Rocha, H. Bíscaro, and S. Peres. Hand movement recognition for Brazilian sign language: A study using distance-based neural networks. In *IJCNN*, pages 697–704. IEEE, 2009.

[10] J. Friedman and W. Stuetzle. Projection pursuit regression. *Journal of the American Statistical Association*, 76:817–823, 1981.

[11] R. Gorman and T. Sejnowski. Analysis of hidden units in a layered network trained to classify sonar targets. *Neural Networks*, 1:75, 1988.

[12] M. Griebel and A. Hullmann. A sparse grid based generative topographic mapping for the dimensionality reduction of high-dimensional data. submitted to the Proceedings of the 5th International Conference on High Performance Scientific Computing - Modeling, Simulation and Optimization of Complex Processes, March 2012, Hanoi. Also available as INS Preprint No. 1206.

[13] H. Hotelling. Analysis of a complex of statistical variables into principal components. In *Journal of Educational Psychology, 24:417441 and 498520*, 1933.

[14] A. Hullmann. Schnelle Varianten des Generative Topographic Mapping. Diploma thesis, Institute for Numerical Simulation, University of Bonn, 2009.

[15] M. Kendall and J. Gibbons. *Rank correlation methods.* A Charles Griffin Book. E. Arnold, 1990.

[16] S. Kullback. *Information Theory and Statistics.* Wiley, New York, 1959.

[17] N. Kwak. Principal component analysis based on L1-norm maximization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(9):1672–1680, 2008.

[18] J. Lee and M. Verleysen. *Nonlinear Dimensionality Reduction.* Springer, 2007.

[19] R. Neal and G. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.

[20] S. Roweis. EM algorithms for PCA and SPCA. In *Advances in Neural Information Processing Systems*, pages 626–632. MIT Press, 1998.

[21] B. Schölkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, Cambridge, MA, USA, 2001.

[22] C. Spearman. The proof and measurement of association between two things. *American Journal of Psychology*, 15:88–103, 1904.

[23] M. Tipping and C. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482, 1999.

[24] M. Tipping and C. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, Series B*, 61:611–622, 1999.