Multivariate Nonlinear Regression with Semiparametric Latent Factor Models

Matthias Seeger
Computer Science Div.
UC Berkeley
Berkeley, CA 94720-1776
mseeger@eecs.berkeley.edu

Michael I. Jordan
Computer Science Div. and Dept. of Statistics
UC Berkeley
Berkeley, CA 94720-1776
jordan@eecs.berkeley.edu

Abstract

We propose a semiparametric model for regression problems involving multiple response variables. Conditional dependencies between the responses are represented through a linear mixture of Gaussian processes. We propose an efficient approximate inference scheme for this semiparametric model whose complexity is linear in the number of training data points, and show how the mixing matrix and kernel parameters can be learned by empirical Bayesian techniques. Our inference technique exploits conditional independencies between the latent variables and can be seen as a variant of belief propagation with nonparametric messages. We present experimental results on a meteorological task.

1 Introduction

We are interested in predicting multiple responses $y_c \in \mathbb{R},\ c = 1, \dots, C$ from covariates $\boldsymbol{x} \in \mathcal{X}$, and we would like to model the responses as conditionally dependent. In statistical terminology, we would like to "share statistical strength" between the y_c. Such sharing can be especially successful if the data for the responses is partially missing.

Models related to the one proposed here are known in geostatistics and spatial prediction as *co-kriging* techniques [2]. For example, suppose a spatial map of uranium concentration is sought after an accidental spill. Carbon concentration is easier to measure than uranium, so the space can be sampled more densely, and these two responses are known to be significantly (conditionally) correlated. In co-kriging we set up a joint spatial model for both responses with the aim of an improved prediction of at least one of them (uranium). Our model can be applied to co-kriging, but goes beyond many of the standard techniques in that conditional dependencies are represented directly using linearly mixed latent random fields, and all free parameters can be learned from the data using empirical Bayesian techniques. Problems with missing data arise frequently in statistics, for example as the result of sensor failures. If several responses are measured for each covariate, dependencies between the responses can be learned from the complete data part and can be used to reconstruct the complete sample. Other potential applications arise in computer vision, for example with the problem of estimating the pose of a human figure from images. In this case the response variables are the joint angles of the human body [1], and constraints on human body poses imply dependencies between these angles. In all these cases it is quite common for data sets to have missing response variables. Methods that share statistical strength among multiple responses can make full use of such data, in marked contrast to "independent" techniques which fit each response separately.

Our approach is related to a nonparametric conditional version of factor analysis where both the P latent factors (which are mixed using the factor loadings) and the C additive independent components are represented by Gaussian processes. This model is semiparametric, as it combines a nonparametric component ($P+C$ Gaussian processes) with a parametric one (the linear mixing). We refer to the model as extended *semiparametric latent factor model* (SLFM).

As in the case of simpler Gaussian process models, a significant part of the challenge of working with the SLFM is computational. In this paper we combine two very different techniques in order to meet this challenge. First, we exploit conditional independencies between latent variables through the belief propagation algorithm. Second, we employ the sparse Informative Vector Machine (IVM) framework [4] in order to represent the approximate beliefs and pass the messages in a way which scales only linearly in the number of training points. We are not aware of previous work combining techniques from parametric structured graphical models and nonparametric random fields on a level comparable to what we do here. Free parameters are adjusted by maximizing a variational lower bound on the marginal likelihood of the data. While most previous work for nonparametric random fields use more robust techniques like cross-validation for this purpose, the large number of $O(P\,C)$ parameters precludes such strategies here. It is straightforward to apply our method to models with non-Gaussian likelihoods, by approximating them using ADF projections as in the single process IVM, but this is not done here.

The extended SLFM presented here generalizes the work of [8] by allowing for additional additive independent components. This extension is significant in that the model of [8] can only represent data whose responses lie in a P-dimensional linear subspace, an assumption which is frequently violated for real world data (and $P < C$). Approximate inference for the extended SLFM presented here is significantly more challenging and requires genuinely new techniques for propagating information between latent variables (in the model of [8] the latent variables have a deterministic linear relationship).

The structure of the paper is as follows. We introduce the SLFM in Section 2. The approximate inference scheme, a version of belief propagation with nonparametric messages, is developed in Section 3. In Section 4, we describe our hyperparameter learning strategy. Experimental results on a meteorological task are presented in Section 5, and we close with a discussion in Section 6. We refer to [6] for many details which do not have space here.

2 Semiparametric Latent Factor Models

In order to model the relationship $\boldsymbol{x} \in \mathcal{X} \to \boldsymbol{y} \in \mathbb{R}^C$, we introduce latent variables $\boldsymbol{v} \in \mathbb{R}^C$ and assume independent Gaussian noise: $P(\boldsymbol{y}|\boldsymbol{v}) = N(\boldsymbol{v}, \mathrm{diag}(\sigma_c^2)_c)$. The prior $P(\boldsymbol{v}|\boldsymbol{x})$ will be modelled using Gaussian processes. The simplest possibility is to assume that the v_c are independent given $\boldsymbol{x}$, *i.e.* $P(\boldsymbol{v}|\boldsymbol{x}) = \prod_c P(v_c|\boldsymbol{x})$. In this case we can represent $P(v_c|\boldsymbol{x})$ as a Gaussian process (GP) with mean function 0 and covariance function $\tilde{K}^{(c)}$:

$$\mathrm{E}\left[v_c(\boldsymbol{x})v_{c'}(\boldsymbol{x}')\right] = \delta_{c,c'}\tilde{K}^{(c)}(\boldsymbol{x}, \boldsymbol{x}').$$

This model will be called the *baseline model* or the independent model in the sequel. Note that the inference and learning task simply decompose into C independent ones, so even if there are dependencies in the data the prediction under the baseline model cannot profit from them. This can be a problem especially in situations where part of the y_c data is missing. On the other end of the spectrum, $P(\boldsymbol{v}|\boldsymbol{x})$ can be dependent Gaussian processes with $C(C+1)/2$ cross-covariance functions. This setup will be called *naive*. While being most flexible, the computational and memory costs for inference are typically infeasible for the naive method. If n is the training set size, we have to deal with n variables at a time in the baseline model, but with $C\,n$ in the naive one.

We propose a model in which $\boldsymbol{v}|\boldsymbol{x}$ are dependent in a flexible adaptive way, yet inference and learning is much more tractable than for the naive model. The key is to restrict the dependencies in a way which can be exploited in inference. We represent $\boldsymbol{v}$ using latent variables $\boldsymbol{u} \in \mathbb{R}^P$, $\boldsymbol{v}^{(0)} \in \mathbb{R}^C$, where typically $P \ll C$, in that for a mixing matrix $\boldsymbol{\Phi} \in \mathbb{R}^{C,P}$ we have

$$\boldsymbol{v} = \boldsymbol{\Phi}\boldsymbol{u} + \boldsymbol{v}^{(0)}. \tag{1}$$

The components of $(\boldsymbol{u}^T\boldsymbol{v}^{(0)T})^T$ are independent *a priori*. The relationship to factor analysis is imminent from this equation. Just as in FA our aim is to represent dependencies which live in a space of lower dimension than the responses. However, our model is conditional in that $\boldsymbol{x}$ is not modeled, while FA represents a complete joint density model. Furthermore, the factors $\boldsymbol{u}$ and independent components $\boldsymbol{v}^{(0)}$ are flexible processes in our model and *not* i.i.d. between cases, even if we condition on $\boldsymbol{\Phi}$.

The components $v_c^{(0)}$ and u_p have GP priors with covariance functions $\tilde{K}^{(c)}$ and $K^{(p)}$ respectively. The baseline model is a special case ($P = 0$), but for $P > 0$ the v_c are dependent, and the dependencies themselves are represented by nonparametric latent random fields $\boldsymbol{u}$. We refer to this setup as *semiparametric latent factor model (SLFM)*, owing to the fact that the model combines nonparametric (the processes $\boldsymbol{u}$, $\boldsymbol{v}$) and parametric elements (the mixing matrix $\boldsymbol{\Phi}$). The model suggested in [8] is obtained by fixing $\boldsymbol{v}^{(0)} = \boldsymbol{0}$. Our model is crucially more flexible in that the responses are not restricted to lie in a P-dimensional subspace.

Note that by integrating out the $\boldsymbol{u}$ processes, we obtain induced cross-covariance functions for $\boldsymbol{x} \mapsto \boldsymbol{v}$:

$$\begin{aligned}\mathrm{E}[v_c(\boldsymbol{x})v_{c'}(\boldsymbol{x}')] &= \delta_{c,c'}\tilde{K}^{(c)}(\boldsymbol{x}, \boldsymbol{x}') \\ &\quad + \sum_p \phi_{c,p}\phi_{c',p}K^{(p)}(\boldsymbol{x}, \boldsymbol{x}'),\end{aligned} \tag{2}$$

and the naive method could be applied based on these. The main goal of this paper is to devise a representation and inference procedure which is significantly more efficient.

Suppose we observe some independently and identically distributed data $D = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\,|\,i = 1, \dots, n\}$. We are interested in approximating the predictive distributions $P(\boldsymbol{v}_*|\boldsymbol{x}_*, D)$, which requires the approximation of the posterior $P(\boldsymbol{v}|D)$. Note that $\boldsymbol{v} = (v_{i,c})_{i,c},\ v_{i,c} = v_c(\boldsymbol{x}_i)$, which consists of $C\,n$ dependent variables, so even though $P(\boldsymbol{v}|D)$ is just a Gaussian, it cannot be dealt with feasibly for realistic $n\,C$. We can accommodate missing values for $y_{i,c}$ effortlessly by simply dropping the corresponding likelihood terms. Note that if $y_{i,c}$ are given for some c, then all $\boldsymbol{v}_i = (v_{i,c'})_{c'}$ are constrained by this case.

We will make use of the following subindex notation: $\boldsymbol{x}_J = (x_j)_{j\in J}$, $\boldsymbol{X}_{I,J} = (x_{i,j})_{i\in I, j\in J}$. A dot denotes the complete range. Vectors such as $\boldsymbol{u}$, $\boldsymbol{v}$ have two indexes i (over cases) and c (over responses) or p (over latent processes) respectively, where i changes faster. We write $\boldsymbol{u}_J = (u_{i,p})_{i \in J,p} \in \mathbb{R}^{P\,|J|}$ and $\boldsymbol{u}_{J,p} = (u_{i,p})_{i \in J} \in \mathbb{R}^{|J|}$.

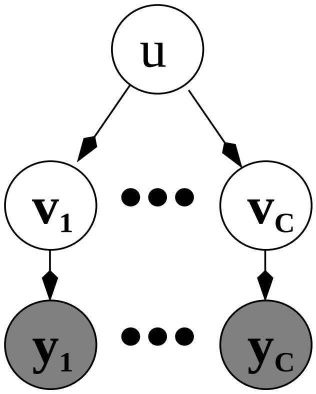
3 Gaussian Process Belief Propagation

The *informative vector machine (IVM)* was proposed in [4] in order to address large-scale binary classification and univariate regression problems with Gaussian process models. It can be applied to single process models, *i.e.* $C = 1$. In a nutshell, the IVM selects an *active* subset I of the training set of size d and represents an approximation to the posterior covariance matrix using $O(n\,d)$ memory. I is selected in order to greedily minimize an information-based criterion which can be computed efficiently given the representation. The insertion of a case into I is referred to as *inclusion*. Just as in Bayesian online techniques, cases are included (*i.e.* we condition on them) in a sequential manner. The complexity of inference is $O(n\,d^2)$. For details about the IVM and hyperparameter learning see [5]. In order to motivate our work here, we note that the IVM framework is a technique for representing the posterior belief for a single random field. In this paper we show how it can be used to represent messages and marginal beliefs in order to drive belief propagation in a graphical model representing a number of dependent fields.

If we applied the IVM technique to our model in the naive way, we would have $C\,n$ variables from which $C\,d$ are selected, leading to $O(C^3\,n\,d^2)$ time and $O(C^2\,n\,d)$ memory cost. This is similar to ignoring the particular structure of a graphical model, *i.e.* considering it as a single clique. In contrast to that, the baseline method scales linearly in C, because the C fields v_c are independent, *i.e.* the graph factorizes completely w.r.t. c. In order to improve upon the naive scaling, we need to exploit the conditional independence structure in $\boldsymbol{v}$ (Eq. 1).

Variables represented by a graphical model can be depicted in two dimensions of dependencies: between different variables in the domain (*model dimension*) and between data cases (*data dimension*). The latter is usually trivial in parametric graphical models: the data cases are i.i.d. given the parameters.[1] In contrast to that, in nonparametric models the data dimension has a rich structure: typically there is no finite number of parameters which render the data independent. On the other hand, the model dimension is trivial in most nonparametric random field techniques we known of, consisting either of a single variable only or assuming pairwise independence between all domain variables (the baseline method is an example).

In the SLFM we see that the v_c are independent given $\boldsymbol{u}$, so the graphical structure is that of a tree if the u_p are collected in one node. The graphical structure is shown on the right. The nodes in this tree extend through the data dimension as well, so that $\boldsymbol{u} \in \mathbb{R}^{n\,P}$ and $\boldsymbol{v}_c \in \mathbb{R}^n$. We can use the *belief propagation (BP)* algorithm [3] to maintain marginal posteriors of the $\boldsymbol{v}_c$ as more and more evidence (components of the $\boldsymbol{y}_c$ nodes) is included.



Note that running BP on this network is challenging due to the large number of components of each node. It is essential to combine the message passing scheme with the IVM framework in order to represent beliefs and messages and perform the local computations efficiently.

Just as in the single process IVM we maintain active sets I_c for every $\boldsymbol{y}_c$ marking the evidence already included. The marginals over the $\boldsymbol{v}_c$ are required to drive the forward selection of the I_c and to adapt hyperparameters (see Section 4). The evidence potentials have the form

$$\Psi_v(\boldsymbol{v}_c) = \exp\left(-\frac{1}{2}\boldsymbol{v}_{I_c,c}^T \boldsymbol{D}^{(c)} \boldsymbol{v}_{I_c,c} + \boldsymbol{v}_{I_c,c}^T \boldsymbol{b}^{(c)}\right)$$

where $\boldsymbol{D}^{(c)} \in \mathbb{R}^{d_c,d_c}$, $d_c = |I_c|$, is diagonal. In our case, $\boldsymbol{D}^{(c)} = \sigma_c^{-2}\boldsymbol{I}$, $\boldsymbol{b}^{(c)} = \sigma_c^{-2}\boldsymbol{y}_{I_c,c}$, but for non-Gaussian likelihoods one can fit Gaussian potentials with other site parameters. The edge potentials are $\Psi_{u\to v}(\boldsymbol{v}_c, \boldsymbol{u}) = P(\boldsymbol{v}_c|\boldsymbol{u}) = N((\boldsymbol{\phi}_c^T \otimes \boldsymbol{I})\boldsymbol{u}, \tilde{\boldsymbol{K}}^{(c)})$ where $\tilde{\boldsymbol{K}}^{(c)} \in \mathbb{R}^{n,n}$ is the kernel matrix for $\tilde{K}^{(c)}$ and $\boldsymbol{\phi}_c = \boldsymbol{\Phi}_{c,\cdot}^T$, furthermore $\Psi_u(\boldsymbol{u}) = P(\boldsymbol{u}) = N(\boldsymbol{0}, \boldsymbol{K})$ with $\boldsymbol{K} = \mathrm{diag}(\boldsymbol{K}^{(p)})_p$ (recall that the $\boldsymbol{u}$ components are independent). Now suppose new evidence is introduced in the sense that j is included into I_c with site parameters $b_{j,c}$, $d_{j,c}$. This will change the message

$$m_{\boldsymbol{v}_c \to \boldsymbol{u}}(\boldsymbol{u}) \propto \int \Psi_v(\boldsymbol{v}_c)\Psi_{u\to v}(\boldsymbol{v}_c, \boldsymbol{u})\, d\boldsymbol{v}_c$$

which in turns modifies the messages $\boldsymbol{u}$ sends to $\boldsymbol{v}_{c'}$, $c' \neq c$:

$$m_{\boldsymbol{u}\to \boldsymbol{v}_{c'}}(\boldsymbol{v}_{c'}) \propto \int \prod_{c'' \neq c'} m_{\boldsymbol{v}_{c''}\to \boldsymbol{u}}(\boldsymbol{u}) \\ \Psi_u(\boldsymbol{u})\Psi_{u\to v}(\boldsymbol{v}_{c'}, \boldsymbol{u})\, d\boldsymbol{u}.$$

The message $m_{\boldsymbol{u}\to \boldsymbol{v}_c}$ remains the same. Finally, all marginals have to be updated:

$$Q(\boldsymbol{v}_{c'}) \propto \Psi_v(\boldsymbol{v}_{c'}) m_{\boldsymbol{u}\to \boldsymbol{v}_{c'}}(\boldsymbol{v}_{c'}),$$

$Q(\boldsymbol{v}_c)$ because $\Psi_v(\boldsymbol{v}_c)$ changed, and $Q(\boldsymbol{v}_{c'})$ because $m_{\boldsymbol{u}\to \boldsymbol{v}_{c'}}$ changed, $c' \neq c$. In a nutshell, our conditional

[1] The graphical symbol for this conditional data independence is the *plate*.

inference approximation iterates between selecting a new pattern j to be included into I_c and the update of the marginals $Q(\boldsymbol{v}_c)$ after each inclusion. The latter are required to score the remaining patterns for the next inclusion. The key problem of applying BP to our nonparametric setup is that messages have to be represented by a number of parameters which *grows* as new evidence is incorporated. This situation is very different from BP on a parametric graphical model where messages have a fixed size. In order to limit the message growth, we need to apply a second approximation by assuming that all active sets I_c share a common prefix I of size d. Our method selects I first (*common inclusion phase*), followed by the choice of $I_c \setminus I$ for each c (*second inclusion phase*).

We will only sketch the representation and its update here, the details are involved and can be found in [6].[2] The message $m_{\boldsymbol{v}_c \to \boldsymbol{u}}$ requires a standard IVM representation $\mathcal{R}_1(c)$ of size d_c for the "prior" $\Psi_{u \to v}(\boldsymbol{v}_c, \boldsymbol{u})$ (as a function of $\boldsymbol{v}_c$) and the "likelihood" $\Psi_v(\boldsymbol{v}_c)$ with active set I_c, an inclusion into I_c triggers an update of $\mathcal{R}_1(c)$ as described in [4]. The message is a (Gaussian) function of $\boldsymbol{u}_{I_c}$, *i.e.* of $\boldsymbol{u}_I$ during the common inclusion phase. Our second approximation consists of limiting the growth of $m_{\boldsymbol{v}_c \to \boldsymbol{u}}$ during the second phase, in that the message must remain a function of $\boldsymbol{u}_I$ even if $I_c \setminus I \neq \emptyset$. This does not mean that $m_{\boldsymbol{v}_c \to \boldsymbol{u}}$ does not change during the second phase, but the changes are "squeezed" through the bottleneck $\boldsymbol{u}_I$. The message $m_{\boldsymbol{u} \to \boldsymbol{v}_c}$ is supported by another IVM representation $\mathcal{R}_2(c)$ which is determined by the product of the messages $m_{\boldsymbol{v}_{c'} \to \boldsymbol{u}}$, $c' \neq c$. It has the size of $\boldsymbol{u}_I$, *i.e.* $P\,d$.[3] Finally, the maintenance of the marginal $Q(\boldsymbol{v}_c)$ requires an IVM representation $\mathcal{R}_3(c)$ of size d_c which is similar to $\mathcal{R}_1(c)$, but the "prior" term is $\propto m_{\boldsymbol{u} \to \boldsymbol{v}_c}$ instead of $\propto \Psi_{u \to v}(\boldsymbol{v}_c, \boldsymbol{u})$.

An update during the second phase (say j into I_c) starts with changing $\mathcal{R}_1(c)$. The modified message $m_{\boldsymbol{v}_c \to \boldsymbol{u}}$ leads to updates of $\mathcal{R}_2(c')$ and $\mathcal{R}_3(c')$ for all $c' \neq c$ (in BP terms this means that the evidence at $\boldsymbol{v}_c$ has to be distributed to all other nodes $\boldsymbol{v}_{c'}$). Finally, the new evidence leads to a change of $\mathcal{R}_3(c)$ as well (the backward message $m_{\boldsymbol{u} \to \boldsymbol{v}_c}$ is not changed). The second phase is run until the sum of active set sizes $\sum_c d_c$ reaches a target value d_{tot}. We will describe the common inclusion phase shortly.

It is important to note that our representation of $Q(\boldsymbol{v})$ as a tree does not allow for efficient access to joint information spanning more than one $\boldsymbol{v}_c$ (the same restriction applies to parametric networks). This is not a problem during the second inclusion phase or for hyperparameter learning (see Section 4), but during the common inclusion phase we would like to score a pattern j using the joint marginal $Q(\boldsymbol{v}_j)$, $\boldsymbol{v}_j = (v_{j,c})_c$. Since the common active set size d is significantly smaller than most of the d_c (see comments below in this Section), we can actually afford to run the common inclusion phase in the naive way (see Section 2) based on the induced cross-covariance functions of Eq. 2, which of course allows easy computation of all joint marginals $Q(\boldsymbol{v}_j)$. At the end of this phase we simply compute the representation described above from scratch and start the second phase from there.

The overall running time complexity for conditional inference with full selection of all I_c is

$$O\left(n\,\left(P\,C\,d + \sum_c d_c\right)\,\sum_c d_c\right)$$

and the memory requirements are

$$O\left(n\,\left(P\,C\,d + \sum_c d_c\right)\right).$$

In large sample situations it makes sense to require $P\,C\,d$ to be of the same order as $d_{tot} = \sum_c d_c$, *i.e.* the common active set size d should be limited to $d_{total}/(P\,C)$ or d_{avg}/P. Under this assumption the second inclusion phase dominates the time requirements. In order to perform the common phase in the naive way, memory usage has to be limited (see [6] for details).

Note that the time requirements are at least $O(n\,(\sum_c d_c)^2)$ which is about C times faster than the naive method (the memory requirements are also C times smaller). In contrast, the baseline method requires $O(n\,\sum_c d_c^2)$ time which can be up to C times faster than our method (if the active sets are all of the same size), showing that modelling conditional dependencies comes at a significant additional price. However, if all the active sets I_c are fixed in advance, the complete representation can be computed in

$$O\left(n\,\left(\sum_c d_c^2 + P\,d\,\left(C\,P\,d + \sum_c d_c\right)\right)\right)$$

which is about $O(n\,\sum_c d_c^2)$ under the assumptions above.[4] We refer to this computation as *conditional inference in minor mode*, as opposed to major mode conditional inference which includes the re-selection of the I_c. During hyperparameter learning, conditional inference is used as a subroutine, and most of these runs can be done in minor mode (see Section 4).

In order to predict $\boldsymbol{v}_*$ and $\boldsymbol{y}_*$ on test datapoints $\boldsymbol{x}_*$, the dominant buffers (of size $O(n)$) in the representation are

[2] The basic ingredients are Cholesky factors and low rank updates thereof, similar to the single process IVM, but combined in more involved ways.

[3] Without our bottleneck approximation, this size could be as large as $P\sum_c d_c$.

[4] This difference is in marked contrast to the situation for the single process IVM where the time complexity is the same for computing the representation with or without selection of the active set.

not required. The test marginals are computed using a subset of the computations for inference in minor mode, namely $\mathcal{R}_2(c)$ and $\mathcal{R}_3(c)$ against the test rather than the training set. The time complexity is

$$O\left(m\left(\sum_c d_c^2 + P\,d\left(C\,P\,d + \sum_c d_c\right)\right)\right)$$

where m is the number of test points. By chunking the test set it is possible to control the space requirements, which means that the computational requirements of our method at prediction time are fairly small (they do not scale with the training set size n) and comparable to the baseline method.

3.1 Active Set Selection

For the single process IVM [4] information-based criteria are used to score remaining points in order to myopically select an optimal candidate for the next inclusion into the active set. These criteria (for a pattern j) are functions of the current marginal posterior approximation $Q(v_j)$. The IVM representation allows us to access *all* marginals at each inclusion while retaining practical feasibility in time and memory. Here, we attempt to adapt this strategy to our model, and we focus on the (instanteneous) *information gain* score being the negative relative entropy between the posteriors after and before the inclusion of j. Since the likelihood factorizes, it is easy to see that in this definition the complete posteriors Q can be replaced by the corresponding marginals at j, so that once the current marginal at j is known, the score for j can be computed easily.

During the second inclusion phase we need to score pairs (j, c) with $j \notin I_c$, using $\Delta_{j,c} = -\mathrm{D}[Q'(v_{j,c}) \,\|\, Q(v_{j,c})]$ (we aim to minimize this score). Here, Q' is the marginal after the inclusion of j into I_c which can be computed in $O(1)$ without updating the representation. Since our representation maintains all marginals $Q(v_{j,c})$ updated at all times, we can score all remaining patterns in $O(n\,C)$. In the common inclusion phase pattern j is scored w.r.t. potential inclusion into I, *i.e.* into all I_c at once, and our decision is based on the score $\Delta_j = -\mathrm{D}[Q'(\boldsymbol{v}_j) \,\|\, Q(\boldsymbol{v}_j)]$. Recall that we use a naive representation during this initial phase which maintains all joint marginals $Q(\boldsymbol{v}_j)$ updated. The computation of Δ_j is $O(C)$ due to the Gaussian likelihood.

For models with non-Gaussian likelihood, we can generalize the usage of ADF projections in the single process IVM to the C-dimensional case. In general, these projections require the evaluation of C-dimensional non-Gaussian integrals which is hard for larger C, but we suggest a proxy in [6] which needs one-dimensional integrals only, and the latter can be done using Gaussian quadrature.

4 Hyperparameter Learning

The set of hyperparameters in our model consists of the kernel parameters of $\tilde{K}^{(c)}$, $K^{(p)}$, $c = 1, \dots, C$, $p = 1, \dots, P$, $\boldsymbol{\Phi} \in \mathbb{R}^{C,P}$, and $(\sigma_c^2)_c$. While for single process models in which conditional inference is performed by IVM or support vector machines (SVM), the number of hyperparameters can be very small and simple robust techniques such as cross-validation can be applied, this is not an option for our setup where at least $C\,P + 1$ parameters have to be adapted to the training data. In this Section we show how an empirical Bayesian technique can be applied to perform this selection in a completely automatic way.

Let $\boldsymbol{\alpha}$ be the vector of all hyperparameters. *Marginal likelihood maximization* amounts to computing the marginal likelihood $P(\boldsymbol{y}|\boldsymbol{\alpha})$ of the data where all primary parameters of the model are integrated out, and to maximizing this score w.r.t. $\boldsymbol{\alpha}$. In our case, the primary parameters are the processes $\boldsymbol{v}$ and $\boldsymbol{u}$ (we can restrict ourselves to the process evaluations at the training points, because these separate the data from the rest of the process values), and the marginal likelihood is a Gaussian in $C\,n$ dimensions whose direct evaluation is practically infeasible (in the same sense as inference the naive way). We can use a standard variational “mean field” lower bound as follows:

$$\begin{aligned}\log P(\boldsymbol{y}|\boldsymbol{\alpha}) &\geq \mathrm{E}_Q\left[\log P(\boldsymbol{y}, \boldsymbol{v}|\boldsymbol{\alpha})\right] + \mathrm{H}[Q(\boldsymbol{v})] \\ &= \mathrm{E}_Q\left[\log P(\boldsymbol{y}|\boldsymbol{v}, \boldsymbol{\alpha})\right] - \mathrm{D}[Q(\boldsymbol{v}) \,\|\, P(\boldsymbol{v}|\boldsymbol{\alpha})]\end{aligned}$$

for any distribution $Q(\boldsymbol{v})$, a simple consequence of Jensen’s inequality and the concavity of $\log$. The bound is tight for the true posterior $Q(\boldsymbol{v}) = P(\boldsymbol{v}|\boldsymbol{y})$, but any other posterior approximation gives a valid lower bound. We will use the posterior approximation $Q(\boldsymbol{v})$ employed in our conditional inference approximation descrived above. In this case we have $\mathrm{D}[Q(\boldsymbol{v}) \,\|\, P(\boldsymbol{v})] = \mathrm{D}[Q(\boldsymbol{v}_I) \,\|\, P(\boldsymbol{v}_I)]$, where $\boldsymbol{v}_I = (v_{i,c})_{i \in I_c, c}$ (a slight abuse of notation). A second bounding step is necessary, because the relative entropy $\mathrm{D}[Q \,\|\, P]$ is not a function of the marginals $Q(\boldsymbol{v}_c)$ only, and only these can be extracted efficiently from our representation:

$$\mathrm{D}[Q(\boldsymbol{v}_I) \,\|\, P(\boldsymbol{v}_I)] \leq \sum_c \mathrm{D}[Q(\boldsymbol{v}_{I_c,c}) \,\|\, P(\boldsymbol{v}_{I_c,c})].$$

Thus, the learning criterion to be minimized is

$$\begin{aligned}\mathcal{G} = \sum_{c=1}^{C} \sum_{i=1}^{n} &\mathrm{E}_Q[-\log P(y_{i,c}|v_{i,c})] \\ &+ \mathrm{D}\left[Q(\boldsymbol{v}_{I_c,c}) \,\|\, P(\boldsymbol{v}_{I_c,c})\right].\end{aligned}$$

If the active sets I_c are fixed, it is possible to compute $\mathcal{G}$ and its gradient w.r.t. $\boldsymbol{\alpha}$ in a way which is as costly as conditional inference in minor mode (given the representation) and does not need additional memory. The idea is that the gradient parts are propagated along the network edges in the same way as the messages. The (involved) details can be found in [6].

We use a simple double-loop optimization strategy in order to descend on $\mathcal{G}$. In the inner loop, we fix the active sets I_c and perform gradient-based minimization of $\mathcal{G}$ using a Quasi-Newton method. Note that it is possible to compute the exact gradient even w.r.t. the noise variances σ_c^2 which appear in the site parameters $\boldsymbol{b}^{(c)}$, $\boldsymbol{D}^{(c)}$ of the posterior approximation Q. In the outer loop, the I_c are re-selected as described in Section 3. The scheme is run for a fixed number of outer loop iterations. It is important to note that all criterion/gradient computations during the inner loop require conditional inference in minor mode only, so that the more costly inference with full re-selection of the I_c has to be done only once for each outer loop iteration.

We close this Section by pointing out some differences to conventional variational Bayesian procedures. First, we do not keep the complete posterior approximation Q (the "variational distribution") fixed during the inner loop, but only the discrete part $\{I_c\}$ for which meaningful descent information such as gradients cannot be obtained easily. In nonparametric methods the posterior (or a sensible approximation) depends very strongly on the prior, so fixing the former while optimizing for the latter will usually lead to very small improvements only. Second, our method is not a variational Bayesian technique in that we do not reselect Q in order to descend on the upper bound $\mathcal{G}$. In fact, re-selection of Q by conditional inference in major mode could lead to an increase in $\mathcal{G}$, although we have not observed that in practice.

5 Experiments

In this Section we present experimental results on a meteorological task. We compare the extended SLFM method against the baseline method in which the single process IVM with hyperparameter learning [5] is applied to each problem $\boldsymbol{x} \mapsto v_c$ separately. Covariance functions are chosen from the *Matérn class* (see [7], Sect. 2.10)

$$K(r) = \frac{V}{2^{\nu-1}\Gamma(\nu)}(\alpha r)^\nu K_\nu(\alpha r), \quad \alpha = 2\nu^{1/2},$$
$$r = \left((\boldsymbol{x} - \boldsymbol{x}')^T \boldsymbol{W}(\boldsymbol{x} - \boldsymbol{x}')\right)^{1/2}$$

with parameters $\nu > 0$, $V > 0$, $\boldsymbol{W} = \operatorname{diag}(w_i)_i$, $w_i > 0$. Here, K_ν is the modified Bessel function of the second kind. ν controls the roughness of sample paths in that they are $\lfloor \nu \rfloor$ times mean-square differentiable. For $\nu = 1/2$ we have the "random walk" Ornstein-Uhlenbeck kernel $Ve^{-\alpha r}$, and the squared-exponential kernel Ve^{-r^2} (frequently used in machine learning) is obtained in the limit $\nu \to \infty$.[5] In all our experiments here, we used $\nu = 5/2$. Note that modeling the data to be described with the squared-exponential kernel and our baseline method results in a poor fit, while the SLFM is affected less severely by this oversmooth choice.

The data consists of $C = 7$ responses sampled on a grid of size 384×512, measured by the *Moderate Resolution Imaging Spectroradiometer (MODIS)*, one instrument in *NASA's Earth Observing System* (see `http://modis.gsfc.nasa.gov/`). The covariate $\boldsymbol{x} \in \mathbb{R}^2$ is the position on the grid, the responses are energy reflected/emitted in spectral bands at 13.9μm, 11μm, 6.7μm, 3.75μm, 1.375μm, 0.936μm, and 0.87μm. These measurements are used in systems for cloud detection, which can be challenging if the landmass is covered with ice and snow. The data was collected over Greenland in July 2002, the spatial resolution is 1km. All variables are normalized to zero mean, unit variance. For the experiments here, we extracted the central 43×58 window and used five different random splits into $n = 2000$ training, $m = 494$ test cases. Some prior analysis using the independent baseline reveals that reponses $5 - 7$ are more variable and harder to predict than $1 - 4$. The smoothest response is 2, the hardest is 5. Figure 1 shows four of the responses.[6] Note that 3 has some regular artefacts (straping), and that $5, 7$ show a more nonstationary behaviour. However, the responses $5 - 7$ seem to be more informative for cloud detection.

We designed the following experiment in order to analyze how baseline and SLFM cope with different degrees of missing data. To this end, for every split and a range of ρ values $\{0, 0.75, 0.9\}$ we removed a fraction ρ of the response values $y_{i,c}$, where (i, c) were drawn at random. Here, we made sure that at least 60 (of 2000) cases remain completely labeled which allows us to run the common inclusion phase for the SLFM (see Section 3) up to $d = 60$. We used $P = 4$ latent processes and $d_{tot} = 2700$ for the SLFM (final value of $\sum_c d_c$) and allowed active sets of size $d_{tot}/C = 386$ for each baseline run (for a single response). For $\rho = 0.9$, we used $d_{tot} = 1390$ (there are only 1400 given response values) and allowed the baseline to use all labeled cases. The hyperparameter optimization was run for 6 outer loops (active set re-selections) and 5 inner iterations (line searches) each. As for kernels, the baseline is allowed an independent covariance function for each response c. For the SLFM, the kernels $\tilde{K}^{(c)}$ share the w_i parameters, but have different V's, while the $K^{(p)}$ kernels

[5] The Matérn class is frequently used in geostatistical applications, arguments underlining its theoretical qualities are given in [7]. Apart from [9] it has not been used in the machine learning community where the squared-exponential kernel is generally recommended. Interestingly, the latter is considered unreasonable by spatial statisticians: it enforces a high degree of smoothness (sample paths are mean-square analytic) which can lead to overly small variance estimates and nonrobust behaviour during hyperparameter learning.

[6] We plot a 122×164 central frame. For each plot, the box marks the window we extracted for our task.

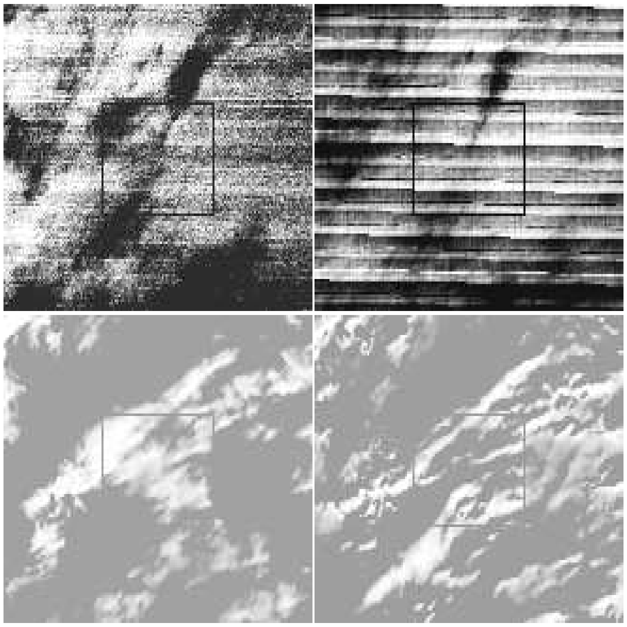


Figure 1: Four responses of the MODIS data. $c = 1$ upper left, $c = 3$ upper right, $c = 5$ lower right, $c = 7$ lower right.

are independent. All runs were started from the same hyperparameter values: $w_i = 1, V = 1$. The mixing matrix $\mathbf{\Phi}$ was initialized to random unit length rows. The results (we quote mean-square test error $\times 100$) can be found in Table 1.

	1	2	3	4	5	6	7
B0	2.912	.1751	2.219	.7646	13.09	4.883	9.181
S0	3.132	.2456	4.182	.8856	16.35	4.871	7.998
B75	3.421	.2627	3.070	1.021	16.88	7.763	12.70
S75	3.638	.2768	6.183	1.274	18.51	7.524	14.52
B90	4.755	.5896	6.482	2.555	43.18	15.45	32.10
S90	4.272	.5627	10.63	2.947	41.04	16.45	32.40

Table 1: Mean-square test error $\times 100$ for setups Bρ (baseline), Sρ (SLFM). ρ is the fraction of missing response variables.

These preliminary results do not suggest any strong conclusions. The SLFM is slightly less affected by the data removal on response 5 which is hardest to fit. On the other hand, it does less well on response 3 (the corruption by the regular straping pattern seems to be handled better by the independent model; see Figure 1). Follow-up experiments, possibly on different datasets, are required in order to obtain a better picture.

We also need to mention that our SLFM implementation is significantly slower than the baseline. A run of $S0$ takes about 1h on a single Pentium processor, while $B0$ needs less than 10min. To put these numbers into perspective, recall that a run includes full hyperparameter optimization over 52 hyperparameters in the SLFM case, performing 6 major and more than 60 minor mode conditional inference steps along the way. The single process IVM implementation does the same number of conditional inference steps (a single major mode step can be compared to a full optimization run of a support vector machine), but needs to optimize 4 parameters only.

6 Discussion

In this paper we have presented a semiparametric model for multi-response regression which represents multiple conditionally dependent Gaussian random fields. Inference and hyperparameter learning are practically efficient even for large datasets. We achieve this favourable scaling by exploiting conditional independencies between the latent variables and using the belief propagation algorithm for inference. We show how the problem of nonparametric message growth can be handled using the IVM technique in order to represent messages and beliefs. The techniques and representations we develop here may be applicable to graphical models with more complicated structure. While the present paper is limited to regression estimation in the presence of Gaussian noise, we have outlined how our work can be applied to models with non-Gaussian likelihood.

Our preliminary results on a single task do not suggest definite conclusions yet, especially in view of the fact that the method proposed here is significantly more costly in terms of running time than the simpler independent baseline. However, the improvement by a factor of C (in time and memory) over a naive implementation places our method halfway between the latter and the baseline in terms of efficiency.

The scheme as presented here grows the representation by starting with trying to identify coupling variables (the $\boldsymbol{u}_I$), and concentrate on the marginals $Q(\boldsymbol{v}_c)$ afterwards. A different promising approach would be to start from the solution of the independent baseline (which can be computed very efficiently using our single process IVM implementation) and try to introduce coupling variables starting from the factorized posterior approximation. In future work we will explore this direction and contrast it with the approach presented here.

Acknowledgements

We thank Tao Shi for pointing out and helping with the MODIS data, and Yee-Whye Teh for insight and important comments.

References

[1] A. Agarwal and B. Triggs. 3D human pose from silhouettes by relevance vector regression. In *Proceedings of the IEEE International Conference on Computer vision and Pattern Recognition*, 2004.

[2] N. Cressie. *Statistics for Spatial Data*. John Wiley & Sons, 2nd edition, 1993.

[3] S. Lauritzen. *Graphical Models*. Oxford Statistical Sciences. Clarendon Press, 1996.

[4] N. D. Lawrence, M. Seeger, and R. Herbrich. Fast sparse Gaussian process methods: The informative vector machine. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 609–616. MIT Press, 2003.

[5] M. Seeger. *Bayesian Gaussian Process Models: PAC-Bayesian Generalisation Error Bounds and Sparse Approximations*. PhD thesis, University of Edinburgh, July 2003. See `www.cs.berkeley.edu/˜mseeger`.

[6] M. Seeger, M. I. Jordan, and Y.-W. Teh. Semiparametric latent factor models. Technical report, University of California at Berkeley, 2004. See `www.cs.berkeley.edu/˜mseeger`.

[7] M. Stein. *Interpolation of Spatial Data: Some Theory for Kriging*. Springer, 1999.

[8] Y.-W. Teh, M. Seeger, and M. I. Jordan. Semiparametric latent factor models. In Z. Ghahramani and R. Cowell, editors, *Workshop on Artificial Intelligence and Statistics 10*, 2005.

[9] C. K. I. Williams and F. Vivarelli. Upper and lower bounds on the learning curve for Gaussian processes. *Machine Learning*, 40(1):77–102, 2000.