Gaussian Process Belief Propagation

Matthias Seeger
Max Planck Institute for Biological Cybernetics
P.O. Box: 21 69
72012 Tübingen, Germany
seeger@tuebingen.mpg.de

July 6, 2006

Abstract

The framework of graphical models is a cornerstone of applied Statistics, allowing for an intuitive graphical specification of the main features of a model, and providing a basis for general Bayesian inference computations though belief propagation (BP). In the latter, messages are passed between marginal beliefs of groups of variables. In parametric models, where all variables are of fixed finite dimension, these beliefs and messages can be represented easily in tables or parameters of exponential families, and BP techniques are widely used in this case. In this paper, we are interested in nonparametric models, where belief representations do not have a finite dimension, but grow with the dataset size. In the presence of several dependent domain variables, each of which is represented as a nonparametric random field, we aim for a synthesis of BP and nonparametric approximate inference techniques. We highlight the difficulties in exercising this venture and suggest possible techniques for remedies. We demonstrate our program using the example of semiparametric latent factor models [15], which can be used to model conditional dependencies between multiple responses.

1 Introduction

Graphical models provide an intuitive way of specifying assumed conditional independence relationships between several domain variables. Moreover, they come with unified *belief propagation* (BP) algorithms[1] to compute Bayesian inference, conditioned on observed data. If the graphical model is parametric, in that the local conditional distributions are such that the marginal posterior beliefs of all unobserved variables have a representation of fixed size, independent of the number of data cases, BP involves passing messages between neighbouring variables in the graph, where the message representations are of fixed size, and the scaling is typically linear in the number n of data cases. The distinctive feature of parametric graphical models is the existence of a mediator variable of fixed *finite* size, such that conditioning on this variable renders the observed cases $i = 1, \ldots, n$ (together with associated latent variables) independent for different i. Because the mediator separates

[1] BP is a variant of the dynamic programming principle, making use of tree structure in the model. If the model graph is not a tree, variables have to be grouped in cliques until the corresponding hypergraph becomes a tree.

training from test cases, it is clear that all information from the training cases required for prediction is represented in the posterior belief for the mediator only. Mediator variables are usually called *parameters*, but our naming here is more specific.

For nonparametric models, such as Gaussian random field models, there is no mediator variable of finite size. We can propose such a variable, but it will be an infinite random object, such as a Gaussian process (GP), and for any practical inference or prediction, we need to integrate it out, which renders all variables across all cases i mutually dependent in a way which has to be represented explicitly (say, by storing the covariance matrix). We may call this problem "curse of dependency", it leads to the fact that a direct generalization of BP to random field models is not tractable in practice.

In this paper, we are interested in generalizing BP to Gaussian random field graphical models in a way which is efficient in practice. A general idea is to apply what we may call *bottleneck* approximations. For example, a factorization assumption means that a joint distribution is approximated by a product of marginal ones. A low rank (or sparsity) assumption means that we introduce artificial mediator variables in a data-dependent way. The concept of a mediator variable illustrates what we mean with the term "bottleneck". Dependence arises through information flowing between variables. A mediator variable creates a narrow bottleneck for this flow, separating many variables by instantiation (conditioning). A different kind of bottleneck approximation is used in Section 4 in the context of the $m_{\boldsymbol{v}^{(c)} \to \boldsymbol{u}}$ messages. In the IVM framework [6, 10] these mediators are a subset of the variables we want to represent beliefs over, and this "active" subset is selected depending on the observed data. The IVM framework has been proposed and applied to single domain variable Gaussian process models, but in this paper we show that the representations and inference computations developed there can be used to obtain an efficient nonparametric variant of BP as well.

Our study is preliminary, in that we try to develop ideas, point out the major problems, and suggest remedies which come as bottleneck approximations. Bayesian inference in graphical models, whether parametric or nonparametric, can be intractable for a great number of reasons. Since in this paper, we are interested in the "curse of dependency" problem only, we focus on the example of semiparametric latent factor models [15] for multi-output regression. While this is a useful nontrivial model, all variables are Gaussian, and domain variables are related linearly, with the model graph being a tree, so that many of the usual difficulties with Bayesian inference do not occur. However, we will hint at these additional problems as we go.

The structure of the paper is as follows. In Section 2, we point out the main differences between parametric and nonparametric graphical models, using the concepts of data and model dimension. In Section 3, SLFMs are introduced as our working example. GP belief propagation for conditional inference in SLFMs is developed in Section 4. Finally, we comment on how to learn hyperparameters in Section 5. Conclusions are given in Section 6.

The notation in this paper is as follows. Vectors $\boldsymbol{a} = (a_i)_i$ and matrices $\boldsymbol{A} = (a_{i,j})_{i,j}$ are bold face. We use subindex notation, in that $\boldsymbol{a}_I = (a_i)_{i \in I}$, "$\cdot$" being the full index. For matrices, $\boldsymbol{A}_I$ is short for $\boldsymbol{A}_{I,I}$. $\boldsymbol{I}$ denotes the identity matrix. Note that $\boldsymbol{a}_I = \boldsymbol{I}_{I,\cdot}\boldsymbol{a}$. Some vectors $\boldsymbol{y}$ will have two indices: $y_{i,c}$, the index i over cases (data dimension), and c over variable components (part of model dimension), and the standard ordering (if nothing else said) is $\boldsymbol{y} = (y_{1,1}, y_{2,1}, \ldots, y_{n,1}, y_{1,2}, \ldots)^T$. We write $\boldsymbol{y}_i = (y_{i,c})_c$, $\boldsymbol{y}^{(c)} = (y_{i,c})_i$. The Kronecker

product is $\boldsymbol{A} \otimes \boldsymbol{B} = (a_{i,j}\boldsymbol{B})_{i,j}$. Our subscript notation for such "double index" vectors is applied to the case index i only: $\boldsymbol{y}_I = (y_{i,c})_{i \in I, c} \in \mathbb{R}^{C|I|}$. $N(\cdot|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the Gaussian distribution, we also use an unnormalized form of the density:

$$N^U(\boldsymbol{z}|\boldsymbol{b}, \boldsymbol{B}) = \exp\left(\boldsymbol{b}^T\boldsymbol{z} - \frac{1}{2}\boldsymbol{z}^T\boldsymbol{B}\boldsymbol{z}\right).$$

2 Data and Model Dimension

Statistical models may be divided into *parametric* and *nonparametric* ones. If $P(\boldsymbol{z}|\boldsymbol{\theta})$ is a model for the observed variable $\boldsymbol{z}$, indexed by parameters $\boldsymbol{\theta}$, and we observe some data $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$, a parametric model is characterized by the existence of *sufficient statistics* $\boldsymbol{\phi}(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n) \in \mathbb{R}^s$ of fixed size s, in that the likelihood of the data can be written solely in terms of the statistics: $P(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n|\boldsymbol{\theta}) = f(\boldsymbol{\phi}(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n), \boldsymbol{\theta})$. For example, if we model $\boldsymbol{z} \sim N(\cdot|\boldsymbol{\theta}, \sigma^2\boldsymbol{I})$ with known variance σ^2, where $N(\cdot|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, then the sample mean $n^{-1}\sum_{i=1}^n \boldsymbol{z}_i$ is a sufficient statistics, so the Gaussian model is parametric.

The existence of finite sufficient statistics has important implications in practice. Estimators of $\boldsymbol{\theta}$ or posterior beliefs for $\boldsymbol{\theta}$ can typically be represented in $O(s)$, independent of the dataset size n. If there are several domain variables, we may devise a *parametric graphical model* to describe their relationships. Here, conditional independence relationships are captured in a graphical manner, in that the variables are nodes in a graph, and roughly speaking, conditional independence is represented in terms of separation in this graph. Finally, conditional distribution between neighbouring nodes are represented by parametric local models. In order to do inference on the global model, namely to compute marginal posterior beliefs of unobserved variables given data, *belief propagation* (BP) techiques can be employed, which essentially pass messages (local conditional distributions) between nodes and update the node marginals accordingly until convergence. For an introduction to parametric graphical models, see [3, 5, 8, 4]. Note that one important factor to make this work is that messages and node marginals *can* actually be represented finitely.

The situation is different for nonparametric models, where dependencies between the cases in a dataset are represented directly, and are not mediated through some finite number of parameters $\boldsymbol{\theta}$. We can still employ a model of the form $P(\boldsymbol{z}|\boldsymbol{\theta})$, but $\boldsymbol{\theta}$ does not have a finite representation anymore. It is often easier to work with nonparametric models after the "parameters" $\boldsymbol{\theta}$ have been integrated out[2]. To this end, a prior distribution $P(\boldsymbol{\theta})$ is chosen, and the joint distribution for variables $\boldsymbol{z}_i$ of interest is obtained as $P(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n) = \int P(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n|\boldsymbol{\theta})P(\boldsymbol{\theta})\, d\boldsymbol{\theta}$. To give an example, we may associate each case $\boldsymbol{z}_i$ with a real variable u_i, then assume that the u_i are *a priori* jointly distributed as a Gaussian with a covariance matrix depending on parts (say $\boldsymbol{x}_i$) of the observations $\boldsymbol{z}_i$. It is useful to regard the relationship $\boldsymbol{x}_i \to u_i$ as a *random field* (or random function). Certain properties, such as the u_i changing smoothly on average w.r.t. $\boldsymbol{x}_i$, can be encoded directly into this setup, by assuming a correlation coefficient between u_i, u_j which grows with shrinking distance between $\boldsymbol{x}_i, \boldsymbol{x}_j$. In this example, a convenient $\boldsymbol{\theta}$ would be a *Gaussian process* (GP), and

[2] In fact, any practical application of nonparametric models has to work on such an integrated out representation, because $\boldsymbol{\theta}$ cannot be represented on a machine.

the prior $P(\boldsymbol{\theta})$ would be a GP distribution. Note that we need at least countably infinitely many variables to describe a GP.

Now, suppose we have several domain variables, whose dependencies we would like to capture with a graphical model. However, we would also like to represent the individual variables by nonparametric random fields. In this paper, we investigate the feasibility of applying the parametric BP technique to such a nonparametric graphical model. In this case, marginal beliefs and messages are random fields themselves. In simple cases, these can be represented as joint distributions over cases of interest, namely after the parameters (GPs in our case) have been integrated out. However, the representations for node marginals and messages grows superlinearly in the number n of cases, as does the cost for message propagations, and a straightforward BP extension along these lines would hardly be of more than academic interest. We propose to use the recently developed IVM technique for sparse approximations for single domain variable GP models [6], in order to represent marginal beliefs and to propagate messages. The central idea is that the BP message passing operations can be expressed in terms of common primitives of Bayesian inference, such as combining information from multiple sources by multiplication of beliefs, or marginalization of variables, and the IVM framework provides efficient approximate solutions for these primitives.

A central problem when trying to deal with nonparametric structured graphical models, is the "curse of dependency" effect. In a parametric model, the cases $\boldsymbol{z}_i$ are independent given the parameters (mediator) $\boldsymbol{\theta}$, but in a nonparametric model, our only option is to integrate out $\boldsymbol{\theta}$, introducing dependencies between the $\boldsymbol{z}_i$ which cannot be represented by a finite mediator variable. This problem becomes worse with several domain variables.

We can think of a *model dimension* (along different domain variables) and a *data dimension* (along cases $\boldsymbol{z}_i$ we are interested in). For a parametric model, the mediator separates variables along the data dimension, although they still may have some complicated dependence structure along the model dimension. Figure 1 illustrates the situation. We see that if the mediator $\boldsymbol{\theta}$ is conditioned upon, paths between the different replicas of the model along the data dimension are blocked, which means that these blocks are conditionally independent. BP may be run on each block independently, and the relevant messages are represented in a way which does not depend on n.

In a nonparametric model, the infinite mediator must be integrated out, which leads to all variables along model and data dimension becoming dependent in a way which has to be represented explicitly. This fact is illustrated in Figure 2. The bidirectional edges are not part of the usual directed graphical models semantics[3], they simply state that the replicas of x, y, z along the data dimension are all fully dependent, in that they constitute a nonparametric random field. There is no useful conditional factorization in this model, and belief representations and their manipulations are formidably expensive. Bottleneck approximations through artificial mediators are required in order to obtain practically efficient inference.

[3]The correct way of drawing these models would be to contract the data dimension and use single nodes for x, y, z, representing the whole random fields.

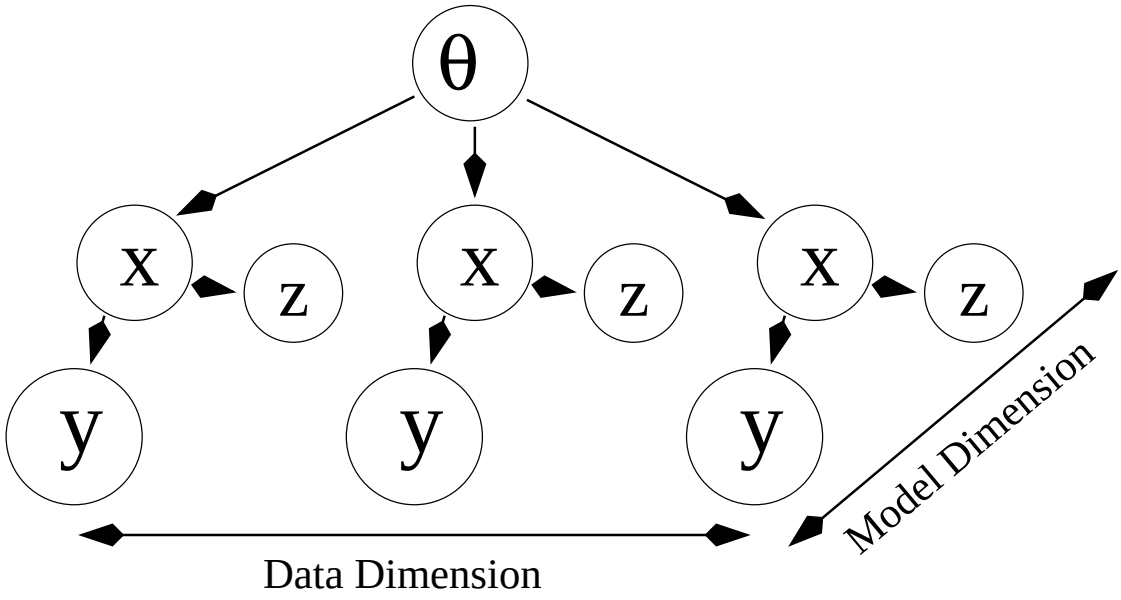

Figure 1: Model and data dimension for a parametric graphical model.

3 Semiparametric Latent Factor Models

In this Section, we introduce the model for which we will demonstrate our GP belief propagation ideas. We are interested in predicting multiple responses $y_c \in \mathbb{R}$ $c = 1, \ldots, C$ 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; in machine learning parlance this is often referred to as "transfer of learning." Such sharing can be especially powerful if the data for the responses is partially unobserved.

Models related to the one proposed here are used in geostatistics and spatial prediction under the name of *co-kringing* [1], where a typical problem can be described as follows. After an accidental uranium spill, a spatial map of uranium concentration is sought. We can take soil samples at selected locations and interpolate from these measurements using GP prediction. However, carbon concentration is easier to measure than uranium, and the two responses are often significantly correlated. In co-kriging, we set up a joint spatial model for several responses with the aim of improving our prediction of one of them. The model to be introduced here can be used for co-kriging, in which the nature of dependence between the responses is conditional, in that it depends on the covariates $\boldsymbol{x}$ (spatial location in our example).

Writing $\boldsymbol{y} = (y_c)_c$ and introducing a latent variable $\boldsymbol{v} \in \mathbb{R}^C$, our model has a factorizing likelihood $P(\boldsymbol{y}|\boldsymbol{v}) = \prod_c P(y_c|v_c)$, $P(y_c|v_c) = N(y_c|v_c, \sigma_c^2)$, *i.e.* the signal $\boldsymbol{v}$ is obscured by Gaussian noise, independent for each c. We intend to model the prior $P(\boldsymbol{v}|\boldsymbol{x})$ 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 $\tilde{K}^{(c)}$:

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

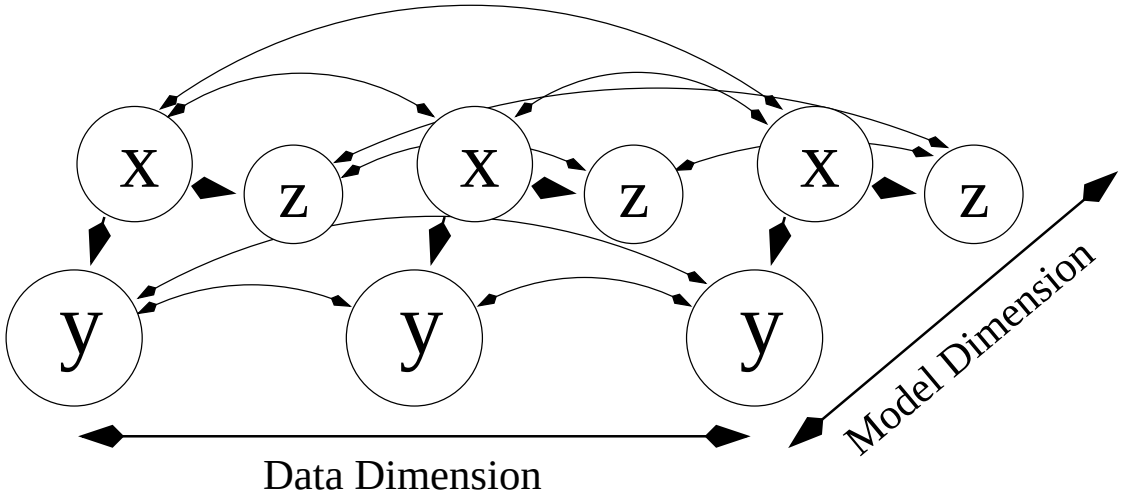

Figure 2: Model and data dimension for a nonparametric graphical model.

Details on GPs for Machine Learning may be found in [11]. The factorizing model will be called the *baseline model* in the sequel. The components v_c are independent *a posteriori* under the baseline model, so statistical strength is not shared among the different components.

On the other end of the spectrum, we can model $P(\boldsymbol{v}|\boldsymbol{x})$ as a set of dependent Gaussian processes with $C(C+1)/2$ cross-covariance functions. Tasks such as inference, hyperparameter learning and prediction can be performed in much the same way as in a single process model, by simply extending the covariate $\boldsymbol{x}$ to $(\boldsymbol{x}, c)$. This model will be called the *naive model*. Due to the "curse of dependency", approximate inference in the naive model scales superlinearly in $C\,n$, which is acceptable only for rather small datasets and number of outputs C.

The *semiparametric latent factor model* (SLFM) [15, 13] lies in between, in that $\boldsymbol{v}|\boldsymbol{x}$ are dependent in a flexible way, yet inference and learning is more tractable than for the naive model. The key is to restrict the dependencies in a way which can be exploited in inference. We introduce a second latent variable $\boldsymbol{u} \in \mathbb{R}^P$. Here and in the following it is understood that for typical applications of our model we will have $P \ll C$. For a mixing matrix $\boldsymbol{\Phi} \in \mathbb{R}^{C,P}$ we set

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

where $\boldsymbol{u}$ and $\boldsymbol{v}^{(0)}$ are independent. The components $v_c^{(0)}$ have independent GP priors with mean 0 and covariance function $\tilde{K}^{(c)}$, and the components u_p have independent zero-mean GP priors with kernel $K^{(p)}$. Our model is a conditional nonparametric version of *factor analysis*. P independent factors u_p are mixed through $\boldsymbol{\Phi}$, and further independent factors $v_c^{(0)}$ are added to the result. The factors have different roles. The $v_c^{(0)}$ represent parts in the signal $\boldsymbol{v}$ which behave independently, while the u_p parameterize conditional dependencies (or correlations in this Gaussian case). The baseline model is a special case ($P = 0$), but for $P > 0$ the components v_c will be dependent *a posteriori*. The model combines nonparametric (processes u_p, $v_c^{(0)}$) and parametric elements (the mixing matrix $\boldsymbol{\Phi}$). Note that the defition here extends on the model of [15], where they had $\boldsymbol{v}^{(0)} \equiv \boldsymbol{0}$.

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

$$\mathrm{E}[v_c(\boldsymbol{x})v_{c'}(\boldsymbol{x}')] = \delta_{c,c'}\tilde{K}^{(c)}(\boldsymbol{x},\boldsymbol{x}') + \sum_p \phi_{c,p}\phi_{c',p}K^{(p)}(\boldsymbol{x},\boldsymbol{x}').$$

We could therefore perform inference and prediction the naive way. However, the relationship between the domain variables $\boldsymbol{u}$ and $\boldsymbol{v}$ is structured, and in this paper we are interested in exploiting this structure in order to obtain a more efficient method for representing the posterior $Q(\boldsymbol{v}) = P(\boldsymbol{v}|D)$ for data D, and to do predictions on unseen points. In the sequel, the posterior over $\boldsymbol{v}$ is denoted[4] by $Q(\boldsymbol{v})$.

4 Gaussian Process Belief Propagation

In this Section, we derive GP belief propagation (BP) for the SLFM introduced in Section 3. As noted above, a naive approach would treat all $O(n\,C)$ variables as dependent and represent their covariance explicitly, which is not feasible in interesting practical situations. To repeat our motivation, we first make use of the tree structure of SLFM (see below) by applying BP for marginal inference, leading to a representation which is factorized along the model dimension. Dependencies along the data dimension are not structured, and additional bottleneck approximations, such as introduction of artificial mediator variables, have to be applied in order to represent and update the posterior marginals $P(v_{i,c}|D)$ efficiently. The details of these representations are formidable even in the case of SLFMs, and for simplicity we will skip over many of them. Our aim here is to point out typical problems that arise as "curse of dependency", and to suggest general remedies. All details for the SLFM case can be found in [13].

We noted in Section 2 that the dependencies along the data dimension are unstructured (in the sense of structure through a sparse graphical model) and have to be represented explicitly. This is a problem, because we would like to use BP techniques to exploit conditional independencies along the model dimension. This involves passing messages between node marginal beliefs, and either has to be represented in a way which scales superlinearly in n. We propose to use low rank bottleneck approximations in order to represent and work with these entities. The *informative vector machine (IVM)* [6, 10] is a general framework for finding such low rank bottlenecks in a data-dependent way, and for performing inference based on them. We will not go into details here, but merely state what will be required later on. For a single domain variable GP model ($C = 1$) with n cases $\boldsymbol{x}_i, y_i$ and latent variables $v_i \in \mathbb{R}$, the prior at the datapoints is $P(\boldsymbol{v}) = N(\boldsymbol{0}, \boldsymbol{K})$, where $\boldsymbol{K} = (K(\boldsymbol{x}_i, \boldsymbol{x}_j))_{i,j}$ is the covariance matrix over the input datapoints $\boldsymbol{x}_i$. The bottleneck variables are $\boldsymbol{v}_I = (v_i)_{i\in I}$, a part of the variables $\boldsymbol{v}$ whose posterior belief is to be represented. Here, $I \subset \{1,\ldots,n\}$ is the *active set* of size $d \ll n$. In the case of SLFMs, the likelihood $P(y_i|v_i)$ is Gaussian, but if it is not, the *expectation propagation (EP)* technique [7] can be used to replace them by a Gaussian function $N^U(v_i|b_i,\pi_i)$, $\pi_i \geq 0$, so the approximate posterior $Q(\boldsymbol{v})$ is Gaussian. If $P(y_i|v_i) = N(y_i|v_i,\sigma^2)$, then $\pi_i = \sigma^{-2}$, $b_i = \sigma^{-2}y_i$. The approximate IVM representation

[4] This convention comes from *approximate* inference, where the true intractable posterior $P(\boldsymbol{v}|D)$ is approximated by a feasible $Q(\boldsymbol{v})$. We use $Q(\boldsymbol{v})$ in the same sense, because due to several bottleneck approximations, our final posterior marginals are approximate as well.

is obtained by constraining $b_i = \pi_i = 0$ for $i \notin I$. If $Q(\boldsymbol{u}) = N(\boldsymbol{h}, \boldsymbol{A})$, we see that

$$\begin{aligned} \boldsymbol{A} &= \left(\boldsymbol{K}^{-1} + \boldsymbol{I}_{\cdot,I}\boldsymbol{\Pi}\boldsymbol{I}_{I,\cdot}\right)^{-1} = \boldsymbol{K} - \boldsymbol{M}\boldsymbol{M}^T, \quad \boldsymbol{h} = \boldsymbol{M}\boldsymbol{\beta}, \\ \boldsymbol{M} &= \boldsymbol{K}_{\cdot,I}\boldsymbol{\Pi}^{1/2}\boldsymbol{L}^{-T}, \quad \boldsymbol{L}\boldsymbol{L}^T = \boldsymbol{B} = \boldsymbol{I} + \boldsymbol{\Pi}^{1/2}\boldsymbol{K}_I\boldsymbol{\Pi}^{1/2}, \quad \boldsymbol{\beta} = \boldsymbol{L}^{-1}\boldsymbol{\Pi}^{-1/2}\boldsymbol{b}, \end{aligned} \tag{1}$$

where $\boldsymbol{\Pi} = \mathrm{diag}(\pi_i)_{i\in I}$ and $\boldsymbol{b} = (b_i)_{i\in I}$ are called *site parameters*. This representation can be derived using the Sherman-Morrison-Woodbury formula [9]. $\boldsymbol{L}$ is lower triangular, and $\boldsymbol{M}$ is called *stub matrix*. The IVM method convolves updates of this *IVM representation* (*i.e.* inclusions of a new i into I) with greedy selections of the best point to include next. To this end, the marginal posterior moments $\boldsymbol{h}, \boldsymbol{a} = \mathrm{diag}\,\boldsymbol{A}$ are kept up-to-date at any time, and the forward selection score is based on those. Patterns are scored highly if their inclusion into I leads to a large information gain, or reduction of posterior entropy. Details about the IVM framework are given in [10].

More generally, an IVM representation is determined by a prior $P(\boldsymbol{u}) = N(\boldsymbol{h}^{(0)}, \boldsymbol{A}^{(0)})$, an active set I (determining the bottleneck variables) of size d, and $2d$ site parameters $\boldsymbol{b}$, $\boldsymbol{\Pi}$, and consists of the variables $\boldsymbol{M}$, $\boldsymbol{\beta}$, $\boldsymbol{h}$, and $\boldsymbol{a}$. These are defined as in Eq. 1, with $\boldsymbol{K}$ being replaced by $\boldsymbol{A}^{(0)}$ and the modifications

$$\boldsymbol{\beta} = \boldsymbol{L}^{-1}\left(\boldsymbol{\Pi}^{-1/2}\boldsymbol{b} - \boldsymbol{\Pi}^{1/2}\boldsymbol{h}_I^{(0)}\right), \quad \boldsymbol{h} = \boldsymbol{h}^{(0)} + \boldsymbol{M}\boldsymbol{\beta}.$$

This general tool for representing a single Gaussian random field belief efficiently through a data-dependent bottleneck will be applied below in several contexts.

Recall the SLFM from Section 3. $\boldsymbol{v} \in \mathbb{R}^{nC}$ are the latent variables directly associated with the responses $\boldsymbol{y}$, and $\boldsymbol{u} \in \mathbb{R}^{nP}$ are the latent variables representing conditional dependencies. What is the graphical structure in the model dimension? If we group $\boldsymbol{v}$ into $\boldsymbol{v}^{(c)} = (v_{i,c})_i$, we see that the $\boldsymbol{v}^{(c)}$ are conditionally independent given $\boldsymbol{u} \in \mathbb{R}^{nP}$. In other words, the graphical model along the model dimension is tree-structured, as shown in Figure 3.

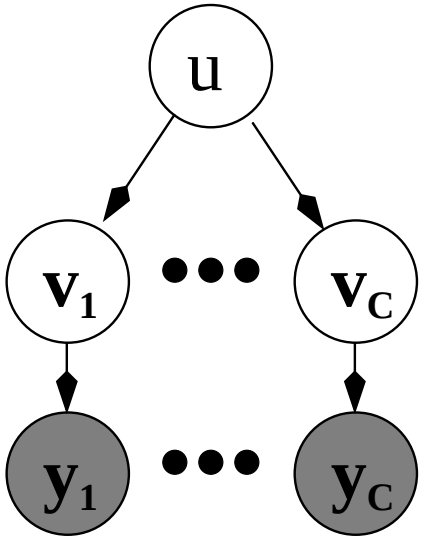

Figure 3: SLFM as a tree-structured graphical model

The belief propagation (BP) algorithm performs inference in tree-structured graphical models[5] by designating an arbitrary root, passing messages outwards from this root to the leafs, then collecting messages back to the root. This sweep has to be done at any time new evidence (observations) become available (and are conditioned on). In our case, all observed data is known at once, but recall that we would like to use IVM bottlenecks for efficiency. The iterative procedure of selecting variables to become mediators one at a time, then including them into the representation, is very similar to observations becoming available in a sequential manner. We therefore employ the following "sequential" scheme. In each iteration, we select new variables to become active. As we will see, this can be interpreted as new evidence in the graphical model, and the posterior representation is updated by a BP sweep.

The local conditional distributions in a graphical model are also called potentials[6], for our purposes they are positive functions sitting on nodes joined by an edge, or on single nodes. Since $\boldsymbol{v}_i = \boldsymbol{\Phi}\boldsymbol{u}_i + \boldsymbol{v}_i^{(0)}$, the edge potentials are

$$\Psi_{u\to v}(\boldsymbol{v}^{(c)}, \boldsymbol{u}) = P(\boldsymbol{v}^{(c)}|\boldsymbol{u}) = N\left((\boldsymbol{\phi}_c^T \otimes \boldsymbol{I})\boldsymbol{u}, \tilde{\boldsymbol{K}}^{(c)}\right)$$

where $\boldsymbol{\phi}_c = \boldsymbol{\Phi}_{c,\cdot}^T$ is the c-th row of $\boldsymbol{\Phi}$. The single node potentials are $\Psi_u(\boldsymbol{u}) = N(\boldsymbol{0}, \boldsymbol{K})$ with $\boldsymbol{K} = \mathrm{diag}(\boldsymbol{K}^{(p)})_p$ and

$$\Psi_v(\boldsymbol{v}^{(c)}) = N^U\left(\boldsymbol{I}_{\cdot,I_c}\boldsymbol{b}^{(c)}, \boldsymbol{I}_{\cdot,I_c}\boldsymbol{\Pi}^{(c)}\boldsymbol{I}_{I_c,\cdot}\right),$$

where I_c, $\boldsymbol{b}^{(c)}$, $\boldsymbol{\Pi}^{(c)}$ are active set and site parameters for an IVM representation. Since we will use IVM forward selection in order to determine good active sets I_c for each marginal belief $Q(\boldsymbol{v}^{(c)})$, we see that the SLFM representation to be developed has to be able to keep the marginal posterior beliefs $Q(v_{i,c}) = N(h_{i,c}, a_{i,c})$ up-to-date at all times. Furthermore, by the form of Ψ_v, including a new entry into I_c can be interpreted as introducing new evidence into the model, as has been noted above. Note that this setup does not allow us to access *joint* information about Q spanning different $\boldsymbol{v}^{(c)}$ blocks (in general, BP delivers *marginal* posterior distributions only).

In the belief propagation method, nodes pass messages to their neighbours. A message can be seen as belief of a node in what the neighbouring node should be, based on information the node receives from its other neighbours *excluding* the receiver of the message[7]. Now suppose that new evidence is introduced into our SLFM clustered graphical model, in the sense that j is included into I_c with site parameters $b_{j,c}$, $\pi_{j,c}$. This will change the message $\boldsymbol{v}^{(c)}$ sends to $\boldsymbol{u}$ which is

$$m_{\boldsymbol{v}^{(c)}\to\boldsymbol{u}}(\boldsymbol{u}) \propto \int \Psi_v(\boldsymbol{v}^{(c)})\Psi_{u\to v}(\boldsymbol{v}^{(c)}, \boldsymbol{u})\, d\boldsymbol{v}^{(c)}, \tag{2}$$

[5]If the graphical model is not tree-structured, BP is often used anyway, known as "loopy BP" in this case. There is no guarantee of convergence in general, and even if BP converges, the result is usually just an approximation to the true posterior marginals. Still, loopy BP often works well in practice. Loopy GP BP is not in the scope of this paper.

[6]We are dealing with directed graphical models here. In undirected models (Markov random fields), potentials can be arbitrary positive functions.

[7]Given this idea, the exact definition of messages is straightforward to derive, and consists of standard Bayesian computations involving sums (marginalization) and products. However, the fact that this intuitive procedure results in correct marginals after a single sweep, is less obvious.

which in turn modifies the messages $\boldsymbol{u}$ sends to $\boldsymbol{v}^{(c')}$, $c' \neq c$:

$$m_{\boldsymbol{u}\to\boldsymbol{v}^{(c')}}(\boldsymbol{v}^{(c')}) \propto \int \prod_{c''\neq c'} m_{\boldsymbol{v}^{(c'')}\to\boldsymbol{u}}(\boldsymbol{u})\Psi_u(\boldsymbol{u})\Psi_{u\to v}(\boldsymbol{v}^{(c')},\boldsymbol{u})\,d\boldsymbol{u}. \tag{3}$$

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 the remainder of this Section, we show how this program can be executed using a sequence of IVM representations, such that after each inclusion into one of the I_c, the marginals $Q(v_{i,c})$ can be updated efficiently. The development is very technical, and some details are omitted here and given in [13]. Our aim here is merely to highlight specific difficulties in the nonparametric BP extension, and these do not depend on specific details, but will rather universally appear for other models as well.

Recall our notation and the standard ordering of variables depending on double indexes (i,c) or (i,p) from Section 1. The kernel matrices $\boldsymbol{K}$ and $\tilde{\boldsymbol{K}}^{(c)}$ are block-diagonal in the standard ordering, because the processes u_p, $v_c^{(0)}$ are all independent *a priori*. However, for $\boldsymbol{u}_I$ it turns out to be simpler to use the opposite ordering $\boldsymbol{u}_I = (u_{i_1,1}, u_{i_1,2}, \dots)^T$, where $I = \{i_1, i_2, \dots\}$. Let $\boldsymbol{\Psi}$ be the permutation matrix[8] such that $\boldsymbol{\Psi}\boldsymbol{u}_I$ is in standard ordering. We use the superscript $\hat{}$ to denote vectors and matrices in the $\boldsymbol{u}_I$ ordering, for example $\hat{\boldsymbol{K}}_I = \boldsymbol{\Psi}^T\boldsymbol{K}_I\boldsymbol{\Psi}$. Note that $\boldsymbol{\Psi}^T(\boldsymbol{\phi}_c \otimes \boldsymbol{I}) = (\boldsymbol{I} \otimes \boldsymbol{\phi}_c)$, so that $(\boldsymbol{\phi}_c \otimes \boldsymbol{I})$ in the standard ordering becomes $(\boldsymbol{I} \otimes \boldsymbol{\phi}_c)$ in the $\boldsymbol{u}_I$ ordering. We begin with the message $m_{\boldsymbol{v}^{(c)}\to\boldsymbol{u}}(\boldsymbol{u})$. We require an IVM representation $\mathcal{R}_1(c)$ (Eq. 1) based on the prior $N(\boldsymbol{v}^{(c)}|\boldsymbol{0}, \tilde{\boldsymbol{K}}^{(c)})$ and I_c, $\boldsymbol{b}^{(c)}$, $\boldsymbol{\Pi}^{(c)}$. Let $\boldsymbol{\mu} = (\boldsymbol{\phi}_c^T \otimes \boldsymbol{I})\boldsymbol{u}$. Some algebra gives

$$m_{\boldsymbol{v}^{(c)}\to\boldsymbol{u}} \propto \exp\left(\boldsymbol{\beta}^{(1,c)T}\boldsymbol{\gamma} - \frac{1}{2}\boldsymbol{\gamma}^T\boldsymbol{\gamma}\right), \quad \boldsymbol{\gamma} = \boldsymbol{L}^{(1,c)-1}\boldsymbol{\Pi}^{(c)1/2}\boldsymbol{\mu}_{I_c} = \boldsymbol{L}^{(1,c)-1}\boldsymbol{\Pi}^{(c)1/2}(\boldsymbol{\phi}_c^T \otimes \boldsymbol{I})\boldsymbol{u}_{I_c}.$$

We do not require the stub matrix $\boldsymbol{M}^{(1,c)}$ in $\mathcal{R}_1(c)$, because this representation is not used to maintain marginal beliefs (but see $\mathcal{R}_3(c)$ below), however we maintain $\boldsymbol{E}^{(c)} = \boldsymbol{\Pi}^{(c)1/2}\boldsymbol{L}^{(1,c)-T}$ with $\mathcal{R}_1(c)$. If $\boldsymbol{P}^{(c)} = (\boldsymbol{\phi}_c \otimes \boldsymbol{I})\boldsymbol{E}^{(c)}$, we have that $\boldsymbol{\gamma} = \boldsymbol{P}^{(c)T}\boldsymbol{u}_{I_c}$, so that $m_{\boldsymbol{v}^{(c)}\to\boldsymbol{u}}$ depends on $\boldsymbol{u}_{I_c}$ only.

At this point, we encounter a "curse of dependency" problem. Even if $m_{\boldsymbol{v}^{(c)}\to\boldsymbol{u}}$ depends on $\boldsymbol{u}_{I_c}$ only, these messages have to combined for all $c' \neq c$ in order to form the reverse message $m_{\boldsymbol{u}\to\boldsymbol{v}^{(c)}}$. If the I_c are to be selected independently of size d_c (say), their union can be of size $\sum_c d_c$, this size governing the representation for the reverse messages. The problem is that we have bottlenecks I_c associated with the nodes $\boldsymbol{v}^{(c)}$, but by combining these we obtain an implied bottleneck for $\boldsymbol{u}$ of size $\sum_c d_c$, which is too large. This problem is especially severe in the SLFM case, because $\boldsymbol{u}$ has *all* other nodes $\boldsymbol{v}^{(c)}$ as neighbours, but similar problems will occur in other examples as well. We deal with it by imposing an explicit bottleneck I on $\boldsymbol{u}$ as well. Such "internal bottlenecks" on variables not directly related to observations have to be chosen depending specifics of the model. In the SLFM case the following seems

[8] In Matlab, $\boldsymbol{\Psi}$ is implemented by reshaping the vector into a matrix, transposing it, and reshaping back into a vector.

sensible. We restrict all I_c to have the common prefix I of size d. Inclusions are therefore done in two phases. In the *common inclusion phase*, patterns are included into I, therefore into all I_c at the same time. In the subsequent *separate inclusion phase*, the I_c are extended independently. However, the messages $m_{\boldsymbol{v}^{(c)}\to\boldsymbol{u}}$ are always restricted to depend on $\boldsymbol{u}_I$ only. Namely,

$$m_{\boldsymbol{v}^{(c)}\to\boldsymbol{u}}(\boldsymbol{u}_I) = N^U\left(\hat{\boldsymbol{P}}^{(c)}\boldsymbol{\beta}^{(1,c)}, \hat{\boldsymbol{P}}^{(c)}\hat{\boldsymbol{P}}^{(c)T}\right), \quad \hat{\boldsymbol{P}}^{(c)} = (\boldsymbol{I}\otimes\boldsymbol{\phi}_c)\boldsymbol{E}^{(c)}_{1\ldots d,\cdot},$$

which simply drops the dependence on $\boldsymbol{u}_{I_c\setminus I}$.[9] Since the bottlenecks are the same for each $m_{\boldsymbol{v}^{(c)}\to\boldsymbol{u}}$, the implied bottleneck for $\boldsymbol{u}$ is I as well (of size $P\,d$). Again, while the generic IVM bottleneck technique of selecting mediator variables is suitable if the variables in question are closely linked to observations, we have just described a second class of bottleneck approximations for "inner messages", namely to limit the dependence of incoming messages to a common set I.

The representation $\mathcal{R}_2(c)$ is needed to form the message $m_{\boldsymbol{u}\to\boldsymbol{v}^{(c)}}$, it basically represents the distribution

$$R_c(\boldsymbol{u}) \propto P(\boldsymbol{u})\prod_{c'\neq c} m_{\boldsymbol{v}_{c'}\to\boldsymbol{u}}(\boldsymbol{u}_I).$$

Because all $m_{\boldsymbol{v}_{c'}\to\boldsymbol{u}}$ are functions of $\boldsymbol{u}_I$ we have $R_c(\boldsymbol{u}) = R_c(\boldsymbol{u}_I)P(\boldsymbol{u}\setminus\boldsymbol{u}_I|\boldsymbol{u}_I)$, thus $\mathcal{R}_2(c)$ needs to be of size $P\,d$ only, and its size grows only during the common inclusion phase. In order to motivate the form of $\mathcal{R}_2(c)$, we need to look ahead to determine the requirements for maintaining the $Q(\boldsymbol{v}^{(c)})$ marginals. Here, $Q(\boldsymbol{v}^{(c)})$ is obtained by combining the evidence potential $\Psi_v(\boldsymbol{v}^{(c)})$ with the reverse message $m_{\boldsymbol{u}\to\boldsymbol{v}^{(c)}}$ in an IVM representation $\mathcal{R}_3(c)$ of size d_c, where now the message $m_{\boldsymbol{u}\to\boldsymbol{v}^{(c)}}$ plays the role of the prior distribution. The representations $\mathcal{R}_1(c)$, $\mathcal{R}_3(c)$ differ in their prior distributions only. Denote the message $m_{\boldsymbol{u}\to\boldsymbol{v}^{(c)}}$ by $N(\boldsymbol{v}^{(c)}|\boldsymbol{\mu}^{(c)}, \boldsymbol{\Sigma}^{(c)})$. A glance at Eq. 3 reveals that we have

$$\boldsymbol{\Sigma}^{(c)} = \tilde{\boldsymbol{K}}^{(c)} + (\boldsymbol{\phi}_c^T\otimes\boldsymbol{I})\mathrm{Var}_{R_c}[\boldsymbol{u}](\boldsymbol{\phi}_c\otimes\boldsymbol{I}). \tag{4}$$

Next, let $d_{\setminus c} = \sum_{c'\neq c} d_c$ and

$$\hat{\boldsymbol{P}}^{(\setminus c)} = \left(\hat{\boldsymbol{P}}^{(1)}\ \ldots\ \hat{\boldsymbol{P}}^{(c-1)}\ \hat{\boldsymbol{P}}^{(c+1)}\ \ldots\ \hat{\boldsymbol{P}}^{(C)}\right) \in \mathbb{R}^{Pd,d_{\setminus c}},$$
$$\hat{\boldsymbol{\beta}}^{(\setminus c)} = \left(\boldsymbol{\beta}^{(1,1)T}\ \ldots\ \boldsymbol{\beta}^{(1,c-1)T}\,\boldsymbol{\beta}^{(1,c+1)T}\ \ldots\ \boldsymbol{\beta}^{(1,C)T}\right)^T \in \mathbb{R}^{d_{\setminus c}}.$$

The order of the columns of $\hat{\boldsymbol{P}}^{(\setminus c)}$ is not important as long as $\hat{\boldsymbol{\beta}}^{(\setminus c)}$ follows the same ordering. These variables represent the combination $\prod_{c'\neq c} m_{\boldsymbol{v}^{(c')}\to\boldsymbol{u}}$ in the definition of $m_{\boldsymbol{u}\to\boldsymbol{v}^{(c)}}$ (see Eq. 3). Some tedious algebra [13] reveals that the following IVM-like representation $\mathcal{R}_2(c)$ is required in order to maintain $\boldsymbol{\mu}^{(c)}$, $\boldsymbol{\Sigma}^{(c)}$, the central parameters of R_c:

$$\begin{aligned}\mathcal{R}_2(c):\quad \boldsymbol{L}^{(2,c)}\boldsymbol{L}^{(2,c)T} &= \boldsymbol{B}^{(2,c)} = \hat{\boldsymbol{K}}_I + \hat{\boldsymbol{K}}_I\hat{\boldsymbol{P}}^{(\setminus c)}\hat{\boldsymbol{P}}^{(\setminus c)T}\hat{\boldsymbol{K}}_I,\\ \boldsymbol{\beta}^{(2,c)} &= \boldsymbol{L}^{(2,c)-1}\hat{\boldsymbol{K}}_I\hat{\boldsymbol{P}}^{(\setminus c)}\hat{\boldsymbol{\beta}}^{(\setminus c)},\\ \boldsymbol{M}^{(2,c)} &= (\boldsymbol{\phi}_c^T\otimes\boldsymbol{I})\boldsymbol{K}_{\cdot,I}\boldsymbol{\Psi}\boldsymbol{L}^{(2,c)-T} \in \mathbb{R}^{n,Pd}\end{aligned} \tag{5}$$

[9] Note that $\hat{\boldsymbol{P}}^{(c)} = \boldsymbol{\Psi}^T\boldsymbol{P}^{(c)}_{1\ldots d,\cdot}$, because $\boldsymbol{u}_I$ is not in standard ordering.

This representation is of size $O(n\,P\,d)$. It is not an IVM representation in the strict sense, but is updated in a very similar manner. We now have $\boldsymbol{\mu}^{(c)} = \boldsymbol{M}^{(2,c)}\boldsymbol{\beta}^{(2,c)}$ and

$$\boldsymbol{\Sigma}^{(c)} = \tilde{\boldsymbol{K}}^{(c)} + (\boldsymbol{\phi}_c^T \otimes \boldsymbol{I})\boldsymbol{K}(\boldsymbol{\phi}_c \otimes \boldsymbol{I}) - (\boldsymbol{\phi}_c^T \otimes \boldsymbol{I})\boldsymbol{M}^{(4)}\boldsymbol{M}^{(4)T}(\boldsymbol{\phi}_c \otimes \boldsymbol{I}) + \boldsymbol{M}^{(2,c)}\boldsymbol{M}^{(2,c)T},$$

where

$$\mathcal{R}_4: \quad \boldsymbol{L}^{(4)}\boldsymbol{L}^{(4)T} = \boldsymbol{K}_I, \quad \boldsymbol{M}^{(4)} = \boldsymbol{K}_{\cdot,I}\boldsymbol{L}^{(4)-T} \tag{6}$$

is another simple representation, which is block-diagonal and therefore of size $O(n\,P\,d)$ only.

Finally, $\mathcal{R}_3(c)$ is a standard IVM representation based on the prior $N(\boldsymbol{\mu}^{(c)}, \boldsymbol{\Sigma}^{(c)})$ and with the same I_c and site parameters as $\mathcal{R}_1(c)$ (see Eq. 1). As opposed to $\mathcal{R}_1(c)$, we need to maintain the stub matrix $\boldsymbol{M}^{(3,c)} \in \mathbb{R}^{n,d_c}$ here, because we want to keep the marginal moments $\boldsymbol{h}^{(c)}$, $\boldsymbol{a}^{(c)}$ of $Q(\boldsymbol{v}^{(c)})$ up-to-date at all times.

The size of the combined representation is $O(n\,(\sum_c d_c + d\,C\,P))$. This should be compared to $O(n\,\sum_c d_c)$ for the baseline method and to $O(n\,C\,\sum_c d_c)$ for the naive method. It is shown in [13] how to update the representation after an inclusion in the common and the separate inclusion phase (the former is more expensive). The details are tedious, but the idea is to apply a sequence of IVM updates of the corresponding IVM representations, with some intervening algebra.

The scaling behaviour is enlightening[10], in that another problem is revealed. The overall running time complexity (for both phases) is

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

In large sample situations it makes sense to require $P\,C\,d$ to be of the same order of magnitude as $\sum_c d_c$. In that case, the memory requirements of our method are the same as for the baseline up to a constant factor. However, it seems that modelling conditional dependencies between classes comes at a significant additional price of at least $O(n\,(\sum_c d_c)^2)$ as compared to $O(n\,\sum_c d_c^2)$ for the independent baseline. On the other hand, our method is faster than the naive implementation[11] by a factor of C. Interestingly, if the active sets I_c and site parameters $\boldsymbol{b}^{(c)}$, $\boldsymbol{\Pi}^{(c)}$ are known, then 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 significantly faster and actually fairly close to what the independent baseline requires. Therefore, in marked contrast to the situation for IVM applied to single process models, conditional inference *with* active set selection comes at a significantly higher cost than without.

The problem is identified easily, and points to another difference between parametric and nonparametric BP. While $\mathcal{R}_2(c)$ is of limited size $P\,d$, for each of the $\sum_c d_c$ inclusions, $C-1$ of the representations have to be updated by rank 1 (in what amounts to an IVM update).

[10] We are aware that the reader has to take these scaling figures for granted, if the details in [13] are not consulted. However, again our purpose is to describe a problem together with an intuitive explanation, whose appreciation would not be helped, or would even be obscured by challenging details.

[11] The naive implementation is not feasible due to memory requirements in the first place.

In other words, the matrices $\hat{\boldsymbol{P}}^{(\setminus c)}\hat{\boldsymbol{P}}^{(\setminus c)T}$ are of size $P\,d$, but are in fact updated $d_{\setminus c} > P\,d$ times by rank 1. Each such update has to be worked through the whole representation in order to make sure that the marginals $\boldsymbol{h}^{(c)}$, $\boldsymbol{a}^{(c)}$ are up-to-date all the time, and delaying these updates does not help either. This is in marked contrast to the situation of parametric BP. In the latter case, we would essentially maintain a *single* marginal belief for $\boldsymbol{u}$ in some representation $\mathcal{R}_2$, independent of c, combining all messages $m_{\boldsymbol{v}^{(c)}\to\boldsymbol{u}}$. We would then obtain the reverse messages $m_{\boldsymbol{u}\to\boldsymbol{v}^{(c)}}$ by dividing this belief by the message $m_{\boldsymbol{v}^{(c)}\to\boldsymbol{u}}$, thus deleting its influence. Since parametric models are typically used with potentials from an exponential family, such a division operation is cheap, and the cost is *independent* of how much evidence has already been included, because evidence is accumulated in the sufficient statistics of the potentials. However, such a division cannot be done efficiently in the Gaussian process case. We need to maintain *different* representations $\mathcal{R}_2(c)$ in order to form each of the reverse messages $m_{\boldsymbol{u}\to\boldsymbol{v}^{(c)}}$, and $C-1$ of them have to be updated *separately* after each inclusion. At this time, we do not have a satisfying remedy for this problem, but it is clearly an important point for future research.

4.1 Prediction

In order to predict on test data, the dominant buffers scaling as $O(n)$ are not required. We need to compute the marginals of Q on the test point which is done just as above for the training points: compute $\boldsymbol{M}^{(2,c)}$, $\boldsymbol{\Sigma}^{(c)}_{\cdot,I_c}$, $\boldsymbol{M}^{(4)}$, and $\boldsymbol{M}^{(3,c)}$ w.r.t. the test points. The cost is the same as computing the representation for the training set from scratch (with fixed active sets and site parameters), but with n replaced by the number of test points m:

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

Again, this is fairly close to the requirements of the baseline method. Predictive distributions can be computed from the marginals using Gaussian quadrature in general (for non-Gaussian likelihoods in models different from SLFMs).

4.2 Selection of Points. Computing Site Parameters

Recall that if the likelihoods $P(y_{i,c}|u_{i,c})$ are not Gaussian[12], we use EP (or assumed density filtering, ADF) projections in order to replace them by Gaussian factors $N^U(u_{i,c}|b_{i,c},\pi_{i,c})$. This is done in much the same way as for single process models [6, 10] during the separate inclusion phase. In the common inclusion phase, a pattern i is included jointly into all I_c. The correct way of doing this would be to determine the joint marginal $Q(\boldsymbol{v}_i),\ \boldsymbol{v}_i \in \mathbb{R}^C$ and doing the ADF projection by computing mean and covariance of $\propto Q(\boldsymbol{v}_i)\prod_c P(y_{i,c}|v_{i,c})$. This has to be done not only for patterns to be included, but also in order to score an inclusion candidate, thus many times between each inclusion, requiring the joint marginals even in case of a Gaussian likelihood. We do not know a way of maintaining these joint marginals more efficiently than using the naive method.[13] The problem is akin to joint rather than

[12] For SLFM, the likelihoods are Gaussian, and the site parameters are given by $b_{i,c} = \sigma_c^{-2}y_{i,c},\ \pi_{i,c} = \sigma_c^{-2}$.

[13] We suggest to use a joint common inclusion phase in [13], which essentially amounts to running the naive method during the common inclusion phase. This requires additional approximations and is not covered here.

marginal inference in parametric graphical models, which also comes at substantial extra costs. Another problem is that for non-Gaussian likelihoods, we need to do C-dimensional quadrature in order to implement the ADF projection, which quickly becomes infeasible with growing C. We settle for the approximation of doing ADF projections separately for each $v_{i,c}$, resulting in site parameters $b_{i,c}$, $\pi_{i,c}$. The updated marginal is $Q'(v_{i,c}) \propto Q(v_{i,c}) N^U(v_{i,c}|b_{i,c}, \pi_{i,c})$, its moments match the ones of $\propto Q(v_{i,c}) P(y_{i,c}|v_{i,c})$.

Next, we need an efficient way of selecting active variables (i, c) for inclusion into I_c. Information-theoretic scores (from active learning) which measure the improvement from $Q(v_{j,c})$ to $Q'(v_{j,c})$ (after inclusion), have been suggested in [6, 10]. For example, we may choose the information gain score $\Delta_{i,c} = -\mathrm{D}[Q'(v_{j,c}) \,\|\, Q(v_{j,c})]$. During the separate inclusion phase, we can score a set of candidates (i', c') and pick the overall winner (i, c), which means that i is included into I_c. During the common inclusion phase, the values $\Delta_{j,c},\ c = 1, \ldots, C$ need to be combined[14] into a score for j, suggestions include

$$\Delta_j^{avg} = C^{-1} \sum_c \Delta_{j,c}, \quad \Delta_j^{min} = \min_c \Delta_{j,c}.$$

5 Parameter Learning

A nonparametric model comes with parameters as well, although they are less directly related to observations than in parametric models. We refer to them as *hyperparameters* $\boldsymbol{\alpha}$. In the case of the SLFM, $\boldsymbol{\alpha}$ includes $\boldsymbol{\Phi}$ and the parameters of the $\tilde{K}^{(c)}$ and $K^{(p)}$. Note that we cannot avoid the hyperparameter learning issue[15] in this case, because $\boldsymbol{\Phi}$ has to be fitted to the data in any case. The derivations in this Section are preliminary and rather serve as suggestions for future work, in that a number of approximations are proposed without experimental validation. However, once more these suggestions are derived from the single process model case, where they have proven useful.

An empirical Bayesian method for estimating $\boldsymbol{\alpha}$ is to maximize the marginal likelihood

$$P(\boldsymbol{y}|\boldsymbol{\alpha}) = \int P(\boldsymbol{y}|\boldsymbol{v}) P(\boldsymbol{v}|\boldsymbol{\alpha})\, d\boldsymbol{v}. \tag{7}$$

This computation is of course intractable, but inference approximation techniques typically come with some approximation for $\log P(\boldsymbol{y}|\boldsymbol{\alpha})$. It is shown in [13] that

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

is an upper bound on $-\log P(\boldsymbol{y}|\boldsymbol{\alpha})$. We can minimize $\mathcal{G}$ in order to choose $\boldsymbol{\alpha}$. If we neglect the dependence of I_c, $\boldsymbol{b}^{(c)}$, $\boldsymbol{\Pi}^{(c)}$ on $\boldsymbol{\alpha}$, we can even compute the gradient $\nabla_{\boldsymbol{\alpha}} \mathcal{G}$ efficiently. As suggested in [10], we can use a double loop optimization scheme. In the outer loop, I_c, $\boldsymbol{b}^{(c)}$, $\boldsymbol{\Pi}^{(c)}$ are computed by conditional inference, as discussed in Section 4. In the inner

[14] Again, we cannot compute exact the information gain for including j into all I_c, since this requires knowledge of the joint $Q(\boldsymbol{v}_j),\ \boldsymbol{v}_j \in \mathbb{R}^C$.

[15] Many kernel methods proposed so far "avoid" the learning issue by selecting hyperparameters by semi-manual techniques such as cross-validation. This is not possible with $\boldsymbol{\Phi}$ in general, because these are too many parameters.

loop, $\mathcal{G}$ is minimized for fixed I_c, $\boldsymbol{b}^{(c)}$, $\boldsymbol{\Pi}^{(c)}$. These inner optimizations should be run for few steps only.

Alternatively, EP comes with its own marginal likelihood approximation (see [12]) which can be adopted to the case of SLFMs. First, our representation only allows access to the marginals $Q(\boldsymbol{v}^{(c)})$, so we should look for an approximation $P(\boldsymbol{y}) \approx e^{-\phi}$ with $\phi = \sum_c \phi_c$. ϕ_c is taken to be the EP marginal likelihood approximation for the "prior" $P(\boldsymbol{v}^{(c)}) = N(\boldsymbol{\mu}^{(c)}, \boldsymbol{\Sigma}^{(c)})$ and the posterior $Q(\boldsymbol{v}^{(c)})$. Following Seeger [12], we obtain after some algebra involving $\mathcal{R}_3(c)$:

$$\phi_c = -\sum_{i=1}^{n} \log Z_{i,c} + \sum_{i \in I_c} \log \tilde{Z}_{i,c} + \frac{1}{2}\left(\log\left|\boldsymbol{B}^{(3,c)}\right| + \boldsymbol{b}^{(c)T}\boldsymbol{\Pi}^{(c)-1}\boldsymbol{b}^{(c)} - \boldsymbol{\beta}^{(3,c)T}\boldsymbol{\beta}^{(3,c)}\right).$$

Here, $\log Z_i = \log \mathrm{E}_{\setminus i}[P(y_{i,c}|v_{i,c})]$ and $\log \tilde{Z}_i = \log \mathrm{E}_{\setminus i}[N^U(v_{i,c}|b_{i,c}, \pi_{i,c})]$, where $Q^{\setminus i}(v_{i,c}) = Q(v_{i,c})$ for $i \notin I_c$, and $Q^{\setminus i}(v_{i,c}) \propto Q(v_{i,c})/N^U(v_{i,c}|b_{i,c}, \pi_{i,c})$ for $i \in I_c$. The criterion simplifies for the case of Gaussian noise, $P(y_{i,c}|v_{i,c}) = N(y_{i,c}|v_{i,c}, \sigma_c^2)$, since $\tilde{Z}_i = Z_i$ for $i \in I_c$ in this case:

$$\phi_c = -\sum_{i \notin I_c} \log Z_{i,c} + \frac{1}{2}\left(\log\left|\boldsymbol{B}^{(3,c)}\right| + \boldsymbol{b}^{(c)T}\boldsymbol{\Pi}^{(c)-1}\boldsymbol{b}^{(c)} - \boldsymbol{\beta}^{(3,c)T}\boldsymbol{\beta}^{(3,c)}\right).$$

The exact gradient of ϕ cannot be computed in general, because the active sets I_c and site parameters depend on $\boldsymbol{\alpha}$. However, if we assume these to be fixed, the gradient $\nabla_{\boldsymbol{\alpha}}\phi$ can be computed efficiently along the same lines as for the variational bound. We may use the same double-loop scheme in order to optimize ϕ.

6 Conclusions

A general marriage of parametric graphical models and nonparametric random field models, each of which are well-understood and frequently used in Machine Learning and Statistics, should prove very rewarding in that models of complicated structure can be dealt with using a more flexible nonparametric setup. However, it is not clear how this connection can be achieved in a sufficiently efficient manner in order to be of interest in practice. In this paper, we showed where the principal problems lie and provided some ideas for tackling them. We used the example of semiparametric latent factor models, for which the main difficulties of the "curse of dependency" arise and can be demonstrated clearly, and we suggested a number of different bottleneck approximations in order to deal with these problems. Our main proposal is to use the generic IVM framework for sparse approximations, which so far has been applied to single process models only. While the techniques we employ here already result in a significant reduction in computational complexity for the SLFM in comparison to a naive approach, additional approximation ideas will have to be developed in order to render this marriage useful to practitioners. Our work presented here is preliminary at this stage, an experimental validation of our ideas is subject to future work.

We are not aware of prior work combining Bayesian GP models with structured graphical models in an efficient manner. Friedman and Nachman [2] suggest Gaussian process directed graphical models, but they do not deal with reducing computational complexity in a principled manner. In fact, they propose either to use the naive approach or to assume that the posterior processes at each node are completely independent (this was called the

baseline method here). Nonparametric belief propagation has been used in a very different context by Sudderth *et.al.*[14]. They represent a belief state by a parametric mixture whose components are updated using ADF projections, but also resampled in a way related to particle filtering. Their method is somewhat more general than ours here, since they are able to represent beliefs with multiple modes[16], but they do not deal with the "curse of dependency" arising through unstructured dependencies of random fields along the data dimension. It would be possible to combine their approach with ours for a structured GP model with multi-modal posterior, but the complexity of such a framework would probably be formidable.

Acknowledgments

The ideas presented here originate in joint work done with Yee-Whye Teh and Michael I. Jordan. This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.

References

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

[2] N. Friedman and I. Nachman. Gaussian process networks. In C. Boutilier and M. Goldszmidt, editors, *Uncertainty in Artificial Intelligence 16*, pages 211–219. Morgan Kaufmann, 2000.

[3] Finn V. Jensen. *An Introduction to Bayesian Networks*. UCL Press, 1st edition, 1996.

[4] M. I. Jordan, editor. *Learning in Graphical Models*. Kluwer, 1997.

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

[6] 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.

[7] Thomas Minka. Expectation propagation for approximate Bayesian inference. In J. Breese and D. Koller, editors, *Uncertainty in Artificial Intelligence 17*. Morgan Kaufmann, 2001.

[8] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1990.

[9] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 2nd edition, 1992.

[10] M. Seeger. *Bayesian Gaussian Process Models: PAC-Bayesian Generalisation Error Bounds and Sparse Approximations*. PhD thesis, University of Edinburgh, July 2003. See `www.kyb.tuebingen.mpg.de/bs/people/seeger`.

[16] The SLFM is a jointly Gaussian model, and the posterior $Q(\boldsymbol{v})$ (for fixed hyperparameters) is a Gaussian, so the problem of multiple modes does not arise.

[11] M. Seeger. Gaussian processes for machine learning. *International Journal of Neural Systems*, 14(2):69–106, 2004.

[12] M. Seeger. Expectation propagation for exponential families. Technical report, University of California at Berkeley, 2005. See `www.kyb.tuebingen.mpg.de/bs/people/seeger`.

[13] M. Seeger, M. I. Jordan, and Y.-W. Teh. Semiparametric latent factor models. Technical report, University of California at Berkeley, 2004. See `www.kyb.tuebingen.mpg.de/bs/people/seeger`.

[14] E. Sudderth, M. Mandel, W. Freeman, and A. Willsky. Distributed occlusion reasoning for tracking with nonparametric belief propagation. In L. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*. MIT Press, 2005.

[15] 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.