

Correlated Product of Experts for Sparse Gaussian Process Regression

Manuel Schürch^{1,2*}, Dario Azzimonti^{1*}, Alessio Benavoli^{3,1}
and Marco Zaffalon¹

¹ Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA), Lugano, Switzerland.

² Università della Svizzera italiana (USI), Lugano, Switzerland.

³ University of Limerick (UL), Limerick, Ireland.

*Corresponding author(s). E-mail(s): manuel.schuerch@idsia.ch;
dario.azzimonti@idsia.ch;

Contributing authors: alessio.benavoli@tcd.ie;
marco.zaffalon@idsia.ch;

Abstract

Gaussian processes (GPs) are an important tool in machine learning and statistics. However, off-the-shelf GP inference procedures are limited to datasets with several thousand data points because of their cubic computational complexity. For this reason, many sparse GPs techniques have been developed over the past years. In this paper, we focus on GP regression tasks and propose a new approach based on aggregating predictions from several local and correlated experts. Thereby, the degree of correlation between the experts can vary between independent up to fully correlated experts. The individual predictions of the experts are aggregated taking into account their correlation resulting in consistent uncertainty estimates. Our method recovers independent Product of Experts, sparse GP and full GP in the limiting cases. The presented framework can deal with a general kernel function and multiple variables, and has a time and space complexity which is linear in the number of experts and data samples, which makes our approach highly scalable. We demonstrate superior performance, in a time vs. accuracy sense, of our proposed method against state-of-the-art GP approximations for synthetic as well as several real-world datasets with deterministic and stochastic optimization.

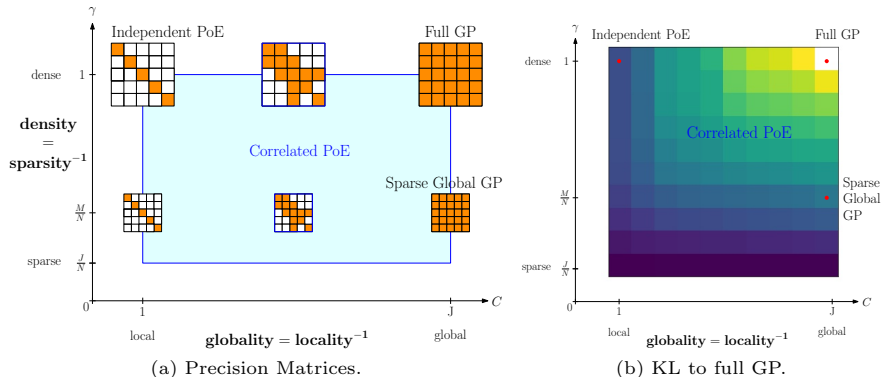
Keywords: Gaussian processes, probabilistic regression, expert fusion.

1 Introduction

Gaussian processes (GPs) are a class of powerful probabilistic method used in many statistical models due to their modelling flexibility, robustness to overfitting and availability of well-calibrated predictive uncertainty estimates with many applications in machine learning and statistics. However, off-the-shelf GP inference procedures are limited to datasets with a few thousand data points N , because of their computational complexity $\mathcal{O}(N^3)$ and memory complexity $\mathcal{O}(N^2)$ due to the inversion of a $N \times N$ kernel matrix [1]. For this reason, many GP approximation techniques have been developed over the past years. There are at least two different approaches to circumvent the computational limitation of full GP. On the one hand, there are *sparse and global* methods [1–4] based on $M_g \ll N$ so-called (global) inducing points, which cover sparsely the input space and optimally summarizing the dependencies of the training points. This results in a low-rank approximation of the kernel matrix of size $M_g \times M_g$, which is less expensive to invert. These methods consistently approximate full GP, for instance the authors in [5] have shown that it converges to full GP as $M_g \rightarrow N$. However, all these methods are still cubic in the number of global inducing points M_g and for many applications - in particular in higher dimensions - the amount of inducing points has to be rather large to capture the pattern of the function properly. A lot of work has been done to optimize the locations of the inducing inputs e.g. [5–7], which allows to have less inducing points but more optimization parameters. This optimization procedures were further improved by stochastic optimization e.g. [8–11], which allows to update the parameters in mini-batches and thus speed up the inference. Optimization of these (variational) parameters helps to scale GP approximations, however, the large number of optimization parameters makes these methods hard to train and they are still limited to M_g global inducing points.

On the other hand, there are *independent and local* models based on averaging predictions from J independent local experts/models resulting in a block-diagonal approximation of the kernel matrix. The final probabilistic aggregation is then based on a product of the individual predictive densities, thus they are called *Product of Experts (PoEs)*, see [12–17]. PoE methods provide fast and rather accurate predictions, because they have fewer hyperparameters than inducing point methods and are locally exact. However, the predictive aggregation of complete independent experts leads to unreliable uncertainty estimates and less accurate predictions in regions between experts. Further, also a rigorous connection to full GP is missing. Beside the mentioned local and global methods, there are also numerical approaches, for instance by exploiting parallelism in specialized hardware [18]. For a more thorough overview of GP approximations we refer to [1, 19].

Our approach aims to overcome these limitations by introducing a framework based on J correlated experts, so that it approximates full GP in two orthogonal directions: sparsity and locality. Thereby, our model is a generalization of the independent PoEs and sparse global GPs by introducing local correlations between experts. These experts correspond to local and sparse GP



models represented by a set of *local inducing points*, which are points on the GP summarizing locally the dependencies of the training data. The degree of correlation C between the experts can vary between independent up to fully correlated experts in a consistent way, so that our model recovers independent PoEs, sparse global GP and full GP in the limiting cases. Our method exploits the conditional independence between the experts resulting in a sparse and low-rank prior as well as posterior precision (inverse of covariance) matrix, which can be used to efficiently obtain local and correlated predictions from each expert. These correlated predictions are aggregated by the covariance intersection method [20], which is useful for combining consistently several estimates with unknown correlations. The resulting predictive distribution is a smooth weighted average of the predictive distributions of the individual experts. Our algorithm works with a general kernel function and performs well in higher dimensional input spaces. The number of hyperparameters to optimize of our method is the same as for full GP, which are just a few parameters (depending on the kernel). These parameters can be similarly estimated via the log marginal likelihood, which is analytically and efficiently computable for our model. In our inference, also log normal priors can be incorporated leading to maximum-a-posteriori estimates for the hyperparameters.

Compared to the number of *global* inducing point M_g , which is usual much smaller than the number of data points N , our approach allows a much higher of total *local* inducing points in the order of N which helps to cover the space and therefore model more complicated functions. Compared to the independent PoEs, the performance can already significantly improve by modelling just a few of the pairwise correlations between the experts. Our method shares also some similarities with other sparse precision matrix GP approximations. The works [21, 22] exploit a band precision matrix together with univariate kernels, whereas [23] propose a precision structure according to a tree. The authors [24, 25] use a more general precision matrix structure, however they need to know the prediction points in advance and are only well suited for low dimensional data (i.e. 1D and 2D), which is usually not useful in the context

of machine learning, where the dimension is higher and predictions are needed after training.

In Section 2, we briefly review full GP for regression and *sparse and global* as well as *independent and local* approaches for GP approximation. In Section 3, we propose our method *Correlated Product of Experts* (CPoEs), where we introduce the graphical model (3.1) of our method and explain the local and sparse character of the prior approximation (3.2). Further, we discuss how to make inference (3.3) and prediction (3.4) in our model. In Section 3.5, we show that the quality of our approximation consistently improves in terms of Kullback-Leibler-(KL)-divergence (B11) w.r.t. full GP for increasing degree of correlation. Further, we present deterministic and stochastic hyperparameter optimization techniques (3.6). In Section 4 we compare against state-of-the-art GP approximation methods in a time versus accuracy sense, for synthetic as well as several real-world datasets. Moreover, comparison to non-GP regression methods are provided. We demonstrate superior performance of our proposed method for different (non-trivial) kernels in multiple dimensions. Section 5 concludes the work and presents future research directions.

2 GP Regression

Suppose we are given a training set $\mathcal{D} = \{y_i, X_i\}_{i=1}^N$ of N pairs of inputs $X_i \in \mathbb{R}^D$ and noisy scalar outputs y_i generated by adding independent Gaussian noise to a latent function f , that is $y_i = f(X_i) + \varepsilon_i$, where $\varepsilon_i \sim \mathcal{N}(0, \sigma_n^2)$. We denote $\mathbf{y} = [y_1, \dots, y_N]^T$ the vector of observations and with $\mathbf{X} = [X_1^T, \dots, X_N^T]^T \in \mathbb{R}^{N \times D}$. We model f with a *Gaussian Process*, i.e. $f \sim \text{GP}(m, k_\theta)$ with mean $m(X)$ and a covariance function (or kernel) $k_\theta(X, X')$ for any $X, X' \in \mathbb{R}^D$, where θ is a set of hyperparameters. For the sake of simplicity, we assume $m(X) \equiv 0$ and a *squared exponential* (SE) kernel with individual lengthscales for each dimension if not otherwise stated, however, the mean function can be arbitrary and the covariance any positive definite kernel function (see, e.g., [1], Chap. 4). For any input matrix $\mathbf{A} = [A_1; \dots; A_M] \in \mathbb{R}^{M \times D}$ consisting of rows $A_i \in \mathbb{R}^D$, we define the GP output value $\mathbf{a} = f(\mathbf{A}) = [f(A_1), \dots, f(A_M)]^T = [a_1, \dots, a_M]^T \in \mathbb{R}^M$, so that the joint distribution $p(\mathbf{a}) = p(a_1, \dots, a_M)$ is Gaussian $\mathcal{N}(\mathbf{a}|\mathbf{0}, \mathbf{K}_{\mathbf{AA}})$ with a kernel matrix $\mathbf{K}_{\mathbf{AA}} \in \mathbb{R}^{M \times M}$, where the entries $[\mathbf{K}_{\mathbf{AA}}]_{ij} = \mathbf{K}_{A_i A_j}$ correspond to the kernel evaluations $k_\theta(A_i, A_j) \in \mathbb{R}$. In particular, the joint distribution $p(\mathbf{f}, f_*)$ of the training values $\mathbf{f} = f(\mathbf{X}) = [f(X_1), \dots, f(X_N)]^T$ and a test function value $f_* = f(X_*)$ at test point $X_* \in \mathbb{R}^D$ is Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{K}_{[\mathbf{X}; X_*] [\mathbf{X}; X_*]})$, where $[\mathbf{X}; X_*]$ is the resulting matrix when stacking the matrices above each other. For GP regression, the Gaussian likelihood $p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma_n^2 \mathbb{I})$ can be combined with the joint prior $p(\mathbf{f}, f_*)$, so that the predictive posterior distribution can be analytically derived [1]. Alternatively, the posterior distribution over the latent variables given the data can

be explicitly formulated as

$$p(\mathbf{f}|\mathbf{y}) \propto p(\mathbf{f}, \mathbf{y}) = \prod_{j=1}^J p(\mathbf{y}_j|\mathbf{f}_j) p(\mathbf{f}_j|\mathbf{f}_{1:j-1}), \quad (1)$$

where the data is split into J mini-batches of size B , i.e. $\mathcal{D} = \{\mathbf{y}_j, \mathbf{X}_j\}_{j=1}^J$ with inputs $\mathbf{X}_j \in \mathbb{R}^{B \times D}$, outputs $\mathbf{y}_j \in \mathbb{R}^B$ and the corresponding latent function values $\mathbf{f}_j = f(\mathbf{X}_j) \in \mathbb{R}^B$. In (1) we used the notation $\mathbf{f}_{k:j}$ indicating $[\mathbf{f}_k, \dots, \mathbf{f}_j]$ and the conditionals $p(\mathbf{f}_j|\mathbf{f}_{1:j-1})$ can be derived from the joint Gaussian, where we define $p(\mathbf{f}_1|\mathbf{f}_{1:0}) = p(\mathbf{f}_1)$. Given the posterior $p(\mathbf{f}|\mathbf{y})$, the predictive posterior distribution from above is equivalently obtained as $p(f_*|\mathbf{y}) = \int p(f_*|\mathbf{f}) p(\mathbf{f}|\mathbf{y}) d\mathbf{f}$ via Gaussian integration (B7). The corresponding graphical model is depicted in Fig. 1(a)i) and 1(b)i), respectively.

The GP depends via the kernel matrix on the hyperparameters $\boldsymbol{\theta}$, which are typically estimated by maximizing the log marginal likelihood $\log p(\mathbf{y}|\boldsymbol{\theta}) = \log \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}_{\mathbf{X}\mathbf{X}} + \sigma_n^2\mathbb{I})$. Although GP inference is an elegant probabilistic approach for regression, the computations for inference and parameter optimization require the inversion of the matrix $\mathbf{K}_{\mathbf{X}\mathbf{X}} + \sigma_n^2\mathbb{I} \in \mathbb{R}^{N \times N}$, which scales as $\mathcal{O}(N^3)$ in time and $\mathcal{O}(N^2)$ for memory which is infeasible for large N .

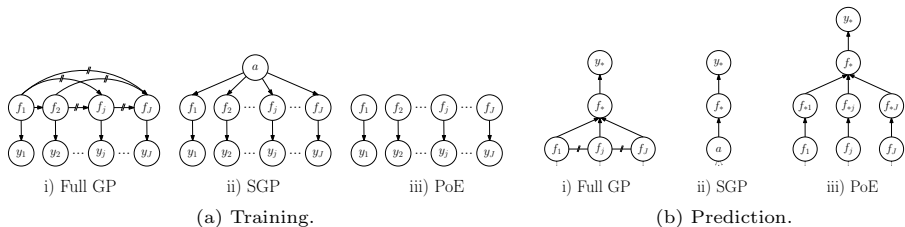


Fig. 1: Graphical models of different GP approaches.

2.1 Global Sparse GPs

Sparse GP regression approximations based on *global inducing points* reduce the computational complexity by introducing $M_g \ll N$ inducing points $\mathbf{a} \in \mathbb{R}^{M_g}$ that optimally summarize the dependency of the whole training data globally, as illustrated in the graphical model in Fig. 1(a)ii) and is denoted in the following as SGP(M_g). Thereby the inducing *inputs* $\mathbf{A} \in \mathbb{R}^{M_g \times D}$ are in the D -dimensional input data space and the inducing *outputs* $\mathbf{a} = f(\mathbf{A}) \in \mathbb{R}^{M_g}$ are the corresponding GP-function values. Similarly to full GP in Eq. (1), the posterior over the inducing points $p(\mathbf{a}|\mathbf{y}) \propto \int p(\mathbf{a}, \mathbf{f}, \mathbf{y}) d\mathbf{f}$ can be derived

from the joint distribution

$$p(\mathbf{a}, \mathbf{f}, \mathbf{y}) = \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{a}) p(\mathbf{a}), \quad (2)$$

where the usual Gaussian likelihood $p(\mathbf{y}_j | \mathbf{f}_j) = \mathcal{N}(\mathbf{f}_j, \sigma_n^2 \mathbb{I})$ and the Gaussian conditional $p(\mathbf{f}_j | \mathbf{a})$ are used. Based on the joint distribution in (2), the posterior $p(\mathbf{a} | \mathbf{y})$ can be derived from which prediction can be performed using the predictive conditional $p(f_* | \mathbf{a})$ as more precisely explained in Appendix E.1 and illustrated in Figure 1(b)ii). Batch inference in these sparse global models can be done in $\mathcal{O}(M_g^2 N)$ time and $\mathcal{O}(M_g N)$ space ([3]).

In order to find optimal inducing inputs \mathbf{A} and hyperparameters $\boldsymbol{\theta}$, a sparse variation of the log marginal likelihood similar to full GP can be used [5–7]. For larger datasets, stochastic optimization has been applied e.g. [8–11] to obtain faster and more data efficient optimization procedures. For recent reviews on the subject we refer to [1, 3, 19].

2.2 Local Independent GPs

Local approaches constitute an alternative to global sparse inducing point methods, which exploit multiple local GPs combined with averaging techniques to perform predictions. In this work we focus on *Product of Expert (PoE)* [14], where individual predictions from J experts based on the local data \mathbf{y}_j are aggregated to the final predictive distribution

$$p(f_* | \mathbf{y}) = \prod_{j=1}^J g_j(p(f_{*j} | \mathbf{y}_j)), \quad (3)$$

where g_j is a function introduced in order to increase or decrease the importance of the experts and depends on the particular PoE method [12, 14, 16, 17, 19]. Note, in particular, the *generalized PoE (GPoE)* [12], where the weights are set to the difference in entropy of the local prior and posterior. The individual predictions $p(f_{*j} | \mathbf{y}_j)$ are based on a local GP, for which the implicit joint posterior can be formulated as

$$p(\mathbf{f} | \mathbf{y}) \propto p(\mathbf{f}, \mathbf{y}) = \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j), \quad (4)$$

where the corresponding graphical model is depicted in Figure 1iii) and more details are provided in Appendix E.2. Other important contributions in this field are distributed local GPs [13], parallel hierarchical PoEs [26], and local experts with consistent aggregations [15, 27]. A different category of averaging techniques are for instance *mixture of experts* [28, 28, 29]), which basically replace the product in (3) by a sum. A particularly interesting approach is *deep*

structured mixtures of GPs [29], which exploits a sum-product network of local and independent GPs. Moreover, simple baseline methods for local methods are the *minimal variance* (*minVar*) and the *nearest expert* (*NE*) aggregation, where only the prediction from the expert with minimal variance and nearest expert is used, respectively. Although both methods show often surprisingly good performance, they suffer from the important disadvantage that there are serious discontinuities at the boundaries between the experts (see for instance Fig. 2) and thus often not useful in practice. This is also the main limitation of all local methods based only on the prediction of one single expert (e.g. deep structured mixture GPs [29]), which was the main reason for introducing smooth PoEs with combined experts. We refer to [19] for a recent overview.

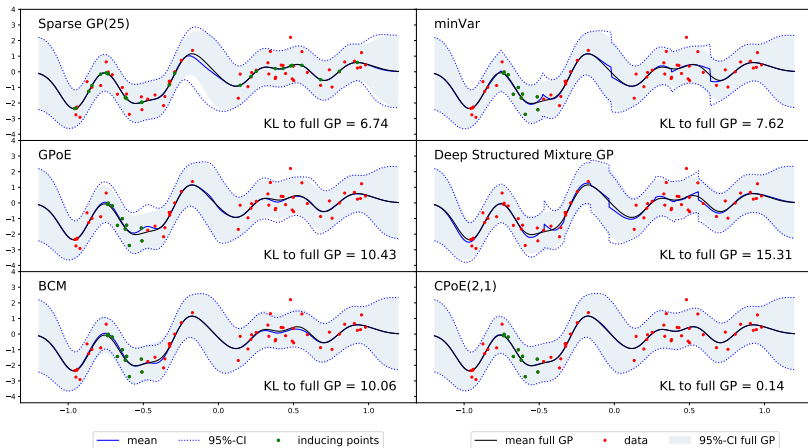


Fig. 2: Different GP approximations (with comparable time complexity) indicated with predictive mean (solid blue) and 95%-credible interval (dotted blue) compared to full GP (black and shaded blue area). The number in the right bottom corner indicates the KL-divergence (B11) to full GP. In the last plot, our method *Correlated Product of Expert* (*CPoE*) is presented for a degree of correlation $C = 2$ and sparsity $\gamma = 1$. We provide a second example in Figure A6 and a discussion about the relation of our method to *deep structured mixture GPs* [29] is given in Section A.5.

3 Correlated Product of Experts

In this section, we present our GP regression method *Correlated Product of Expert* $CPoE(C, \gamma)$, which is a generalization of the independent PoEs and sparse global GPs. The first generalization is the introduction of correlations between the experts, which can be adjusted by the parameter $1 \leq C \leq J$ and allows to interpolate between local and global models. Secondly, similar to the sparse global approximation, our method allows to sparsify the inducing points by sparsity parameter $0 < \gamma \leq 1$. We refer to Table 1 in the Appendix for an overview of the used notation.

3.1 Graphical Model

Assuming $N = BJ$ data samples which are divided into J ordered partitions (or experts) of size B , i.e. $\mathcal{D} = \{\mathbf{y}_j, \mathbf{X}_j\}_{j=1}^J$ with inputs $\mathbf{X}_j \in \mathbb{R}^{B \times D}$ and outputs $\mathbf{y}_j \in \mathbb{R}^B$. We denote $\mathbf{f}_j = f(\mathbf{X}_j) \in \mathbb{R}^B$ the corresponding latent function values on the GP f . We abbreviate $\mathbf{y} = \mathbf{y}_{1:J} \in \mathbb{R}^N$, $\mathbf{X} = \mathbf{X}_{1:J} \in \mathbb{R}^{N \times D}$ and $\mathbf{f} = \mathbf{f}_{1:J} \in \mathbb{R}^N$.

Definition 1 (Local Inducing Points) We refer to local inducing points $\{\mathbf{a}_j, \mathbf{A}_j\}_{j=1}^J$ with inducing inputs $\mathbf{A}_j \in \mathbb{R}^{L \times D}$ and the corresponding inducing outputs $\mathbf{a}_j = f(\mathbf{A}_j) \in \mathbb{R}^L$ of size $L = \lceil \gamma B \rceil$ with $0 < \gamma \leq 1$.

These L local inducing points $(\mathbf{a}_j, \mathbf{A}_j)$ of expert j serve as local summary points for the data $(\mathbf{y}_j, \mathbf{X}_j)$, where the sparsity level can be adjusted by γ . If $\gamma = 1$, the inducing inputs \mathbf{A}_j correspond exactly to \mathbf{X}_j and correspondingly $\mathbf{a}_j = \mathbf{f}_j$. We abbreviate $\mathbf{a} = \mathbf{a}_{1:J} \in \mathbb{R}^M$, where $M = LJ$, for all local inducing outputs with the corresponding local inducing inputs $\mathbf{A} = \mathbf{A}_{1:J} \in \mathbb{R}^{M \times D}$. Next, we model connections between the experts by a set of neighbour experts according to the given ordering.

Definition 2 (Predecessor and Correlation Index Sets) Let $\phi_i(j) \in \{1, \dots, j-1\}$ the index of the i th predecessor of the j th expert. For a given correlation parameter $1 \leq C \leq J$, we introduce the *predecessor set* $\boldsymbol{\pi}_C(j) = \bigcup_{i=1}^j \phi_i(j)$ satisfying

$$\boldsymbol{\pi}_C(j) \subset \{1, \dots, j-1\} \quad \text{and} \quad \boldsymbol{\pi}_{C+1}(j) = \boldsymbol{\pi}_C(j) \cup \phi_{C+1}(j),$$

such that the size of the set $I_j = |\boldsymbol{\pi}_C(j)| = \min(j-1, C-1)$. Further, we define the region of correlation with the *correlation indices* as $\boldsymbol{\psi}_C(j) = \boldsymbol{\pi}_C(j) \cup j$ if $j > C$ and $\boldsymbol{\psi}_C(j) = \boldsymbol{\psi}_C(C) = \{1, \dots, C\}$ otherwise, so that $|\boldsymbol{\psi}_C(j)| = C$ for all j .

The purpose of these predecessor and correlation indices is to model the local correlations among the experts of degree C . If for all j the indices $\boldsymbol{\pi}_C(j)$ are the $C-1$ previous indices, we say that the predecessors are *consecutive* and *non-consecutive* otherwise. If C is clear from the context, $\boldsymbol{\pi}_C(j)$ and $\boldsymbol{\psi}_C(j)$ are abbreviated by $\boldsymbol{\pi}(j)$ and $\boldsymbol{\psi}(j)$, respectively. Details about the specific choices of the ordering, partition, inducing points and predecessor indices are given in Section 3.6.1.

Definition 3 (Graph) We define a directed graph $\mathcal{G}(V, E)$ with nodes $V = \mathbf{a} \cup \mathbf{f} \cup \mathbf{y}$ and directed edges

$$E = \{ \{(\mathbf{a}_{\boldsymbol{\pi}_C^i(j)}, \mathbf{a}_j)\}_{i=1}^{I_j} \cup \{(\mathbf{a}_{\boldsymbol{\psi}_C^i(j)}, \mathbf{f}_j)\}_{i=1}^C \cup \{(\mathbf{f}_j, \mathbf{y}_j)\}_{j=1}^J \},$$

where $\boldsymbol{\pi}_C^i(j)$ and $\boldsymbol{\psi}_C^i(j)$ denote the i th element in the corresponding set.

The directed graph \mathcal{G} is depicted in Fig. 4a(ii), where the local inducing points of the j th expert are connected with the inducing points of the I_j experts in $\pi_C(j)$. Further, the function values \mathbf{f}_j are connected in the region of correlation $\psi_C(j)$ to the local inducing points. The graph $\mathcal{G} = (V, E)$ can be equipped with a probabilistic interpretation, in particular, each node $\mathbf{v} \in V$ and each incoming edge $(\mathbf{v}_i, \mathbf{v}) \in E$ for all predecessors $i = 1, \dots, I$ can be interpreted as a conditional probability density $p(\mathbf{v}|\mathbf{v}_1, \dots, \mathbf{v}_I)$.

Proposition 1 (Graphical Model; Proof 1) *We define a graphical model corresponding to the graph $\mathcal{G}(V, E)$ with the conditional probability distributions*

$$p(\mathbf{y}_j|\mathbf{f}_j) = \mathcal{N}(\mathbf{y}_j|\mathbf{f}_j, \sigma_n^2 \mathbb{I}), \quad (5)$$

$$p(\mathbf{f}_j|\mathbf{a}_{\psi(j)}) = \mathcal{N}(\mathbf{f}_j|\mathbf{H}_j \mathbf{a}_{\psi(j)}, \bar{\mathbf{V}}_j) \quad (6)$$

$$p(\mathbf{a}_j|\mathbf{a}_{\pi(j)}) = \mathcal{N}(\mathbf{a}_j|\mathbf{F}_j \mathbf{a}_{\pi(j)}, \mathbf{Q}_j), \quad (7)$$

where (5) is the usual Gaussian likelihood for GP regression with noise variance σ_n^2 , (6) the projection conditional and (7) the prior transition. Thereby, the matrices are defined as $\mathbf{H}_j = \mathbf{K}_{\mathbf{X}_j \mathbf{A}_{\psi(j)}} \mathbf{K}_{\mathbf{A}_{\psi(j)} \mathbf{A}_{\psi(j)}}^{-1} \in \mathbb{R}^{B \times LC}$, $\bar{\mathbf{V}}_j = \text{Diag}[\mathbf{K}_{\mathbf{X}_j \mathbf{X}_j} - \mathbf{K}_{\mathbf{X}_j \mathbf{A}_{\psi(j)}} \mathbf{K}_{\mathbf{A}_{\psi(j)} \mathbf{A}_{\psi(j)}}^{-1} \mathbf{K}_{\mathbf{A}_{\psi(j)} \mathbf{X}_j}] \in \mathbb{R}^{B \times B}$, $\mathbf{F}_j = \mathbf{K}_{\mathbf{A}_j \mathbf{A}_{\pi(j)}} \mathbf{K}_{\mathbf{A}_{\pi(j)} \mathbf{A}_{\pi(j)}}^{-1} \in \mathbb{R}^{L \times LI_j}$, and $\mathbf{Q}_j = \mathbf{K}_{\mathbf{A}_j \mathbf{A}_j} - \mathbf{K}_{\mathbf{A}_j \mathbf{A}_{\pi(j)}} \mathbf{K}_{\mathbf{A}_{\pi(j)} \mathbf{A}_{\pi(j)}}^{-1} \mathbf{K}_{\mathbf{A}_{\pi(j)} \mathbf{A}_j} \in \mathbb{R}^{L \times L}$ with $\mathbf{F}_1 = \mathbf{0}$ and $\mathbf{Q}_1 = \mathbf{K}_{\mathbf{A}_1 \mathbf{A}_1}$.

The two conditional distributions (6) and (7) can be derived from the true joint prior distribution $p(\mathbf{a}, \mathbf{f}, \mathbf{y})$ as shown in Proof 1. Alternatively, a generalization of this model can be obtained when using a modified projection distribution $p(\mathbf{f}_j|\mathbf{a}_{\psi(j)})$, so that for $C \rightarrow J$ and $\gamma < 1$ our model recovers a range of well known global sparse GP methods as described in Section A.1 and Prop. 5. In any case, these local conditional distributions lead to the following joint distribution.

Definition 4 (Joint Distribution) For the graphical model corresponding to graph \mathcal{G} , the joint distribution over all variables $\mathbf{f}, \mathbf{a}, \mathbf{y}$ can be written as

$$q_{c,\gamma}(\mathbf{f}, \mathbf{a}, \mathbf{y}) = \prod_{j=1}^J p(\mathbf{y}_j|\mathbf{f}_j) p(\mathbf{f}_j|\mathbf{a}_{\psi(j)}) p(\mathbf{a}_j|\mathbf{a}_{\pi(j)}).$$

In the case $\gamma = 1$ and thus $\mathbf{a} = \mathbf{f}$, the joint distribution simplifies (Proof 2) to

$$q_{c,1}(\mathbf{f}, \mathbf{y}) = \prod_{j=1}^J p(\mathbf{y}_j|\mathbf{f}_j) p(\mathbf{f}_j|\mathbf{f}_{\pi(j)}).$$

We use $q = q_{c,\gamma}$ instead of p in order to indicate that it is an approximate distribution. The joint distributions in Def. 4 and the corresponding graphical model in Fig. 4a allow interesting comparisons to other GP models in Fig. 1

and the corresponding formulas (1), (2), (4). Whereas the conditioning set for full GP are all the previous latent values $\mathbf{f}_{1:j-1}$, for sparse GPs some global inducing points \mathbf{a} and for local independent experts the empty set, we propose to condition on the $C - 1$ predecessors $\mathbf{f}_{\pi(j)}$ (or a sparsified version in the general case). From this point of view, we can notice that our probabilistic model is equal to full GP, sparse GP and PoEs under certain circumstances, which are more precisely formulated in Prop. 5.

3.2 Sparse and Local Prior Approximation

The conditional independence assumptions between the experts induced by the predecessor structure π_C lead to an approximate prior $q_{c,\gamma}(\mathbf{a})$ and approximate projection $q_{c,\gamma}(\mathbf{f}|\mathbf{a})$ yielding a sparse and local joint prior $q_{c,\gamma}(\mathbf{a}, \mathbf{f}, \mathbf{y})$.

Proposition 2 (Joint Prior Approximation, Proof 4) *The prior over all local inducing points \mathbf{a} in our CPoE model is*

$$q_{c,\gamma}(\mathbf{a}) = \prod_{j=1}^J p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) = \mathcal{N}(\mathbf{a} | \mathbf{0}, \mathbf{S}_C^{-1}),$$

with prior precision $\mathbf{S}_C = \mathbf{S} = \mathbf{F}^T \mathbf{Q}^{-1} \mathbf{F} \in \mathbb{R}^{M \times M}$, where $\mathbf{Q} = \text{Diag}[\mathbf{Q}_1, \dots, \mathbf{Q}_J] \in \mathbb{R}^{M \times M}$ and $\mathbf{F} \in \mathbb{R}^{M \times M}$ is given as the sparse lower triangular matrix in Fig. 5. Moreover, the projection is

$$q_{c,\gamma}(\mathbf{f}|\mathbf{a}) = \prod_{j=1}^J p(\mathbf{f}_j | \mathbf{a}_{\psi(j)}) = \mathcal{N}(\mathbf{f} | \mathbf{H}\mathbf{a}, \bar{\mathbf{V}}),$$

where $\mathbf{H} \in \mathbb{R}^{N \times M}$ defined in Figure 5 and $\bar{\mathbf{V}} = \text{Diag}[\bar{\mathbf{V}}_1, \dots, \bar{\mathbf{V}}_J] \in \mathbb{R}^{N \times N}$. Together with the exact likelihood $p(\mathbf{y}|\mathbf{f}) = \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) = \mathcal{N}(\mathbf{y} | \mathbf{f}, \sigma_n^2 \mathbb{I})$ determines the joint approximate prior

$$q_{c,\gamma}(\mathbf{a}, \mathbf{f}, \mathbf{y}) = p(\mathbf{y}|\mathbf{f}) q_{c,\gamma}(\mathbf{f}|\mathbf{a}) q_{c,\gamma}(\mathbf{a}).$$

Note that the joint prior $q_{c,\gamma}(\mathbf{a}, \mathbf{f}, \mathbf{y})$ is Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{W})$ with dense covariance \mathbf{W} and sparse precision $\mathbf{Z} = \mathbf{W}^{-1}$ as shown in Fig. C7 in the Appendix. If the predecessor set is consecutive, the matrix \mathbf{F} is a lower band (block)matrix with bandwidth C and in the non-consecutive case each row has exactly C non-zero blocks. The sparsity pattern of \mathbf{F} is inherited to the prior precision $\mathbf{S} = \mathbf{F}^T \mathbf{Q}^{-1} \mathbf{F}$, which is also a sparse matrix (see Fig. 3). For the consecutive case, \mathbf{S} is a block-band matrix with bandwidth $2C - 1$. Note that, the inverse \mathbf{S}^{-1} is dense. The likelihood matrix \mathbf{H} is exact in the corner up to indices C which ensures that our model recovers sparse global GP in the limiting case $C = J$. The quality of the approximation of our CPoE(C, γ) model is discussed in Section 3.5, where we show that $q_{c,\gamma}(\mathbf{a}, \mathbf{f}, \mathbf{y})$ converges to the true prior $p(\mathbf{a}, \mathbf{f}, \mathbf{y})$ for $C \rightarrow J$.

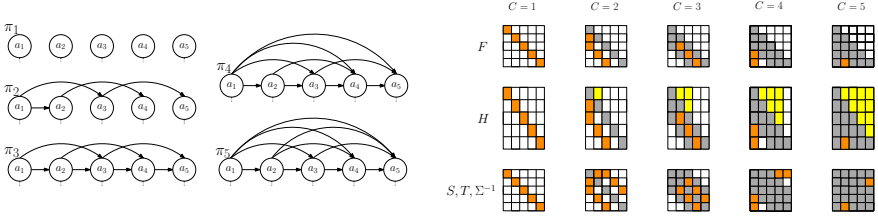


Fig. 3: Correlation structure π_C between the $J = 5$ experts for different degrees of correlation $1 \leq C \leq J$. Left: Graphical model among the local inducing points \mathbf{a}_j . Right: Structure of sparse transition matrix \mathbf{F} , projection matrix \mathbf{H} , prior precision \mathbf{S} , likelihood precision \mathbf{T} and posterior precision Σ^{-1} . Note that π_C does not have to be consecutive, e.g. $2 \notin \pi_2(3)$.

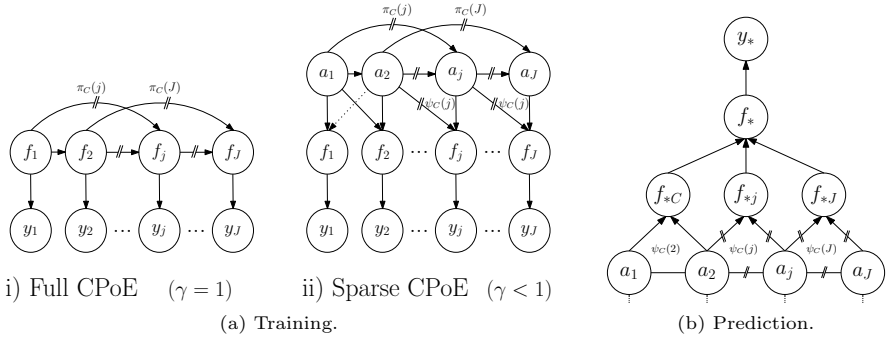


Fig. 4: Graphical model for training and prediction of CPoE(C, γ).

$$\mathbf{F} = \begin{bmatrix} I & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ -F_2^1 & I & 0 & 0 & 0 & 0 & 0 & \ddots & \vdots \\ -F_3^1 & -F_3^2 & I & 0 & 0 & 0 & 0 & \ddots & 0 & 0 \\ 0 & -F_4^1 & -F_4^2 & I & 0 & \ddots & 0 & 0 & 0 & 0 \\ 0 & -F_5^1 & 0 & -F_5^2 & \ddots & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \ddots & \vdots & I & 0 & 0 & 0 & 0 \\ \vdots & \ddots & 0 & 0 & \vdots & \ddots & 0 & I & 0 & 0 \\ 0 & \dots & 0 & -F_j^1 & \dots & -F_j^2 & \dots & I & 0 & 0 \\ \vdots & \ddots & 0 & 0 & \vdots & \ddots & 0 & 0 & I & 0 \\ 0 & \dots & 0 & -F_j^1 & \dots & -F_j^{C-2} & 0 & -F_j^{C-1} & I & 0 \end{bmatrix}$$

$$\mathbf{H} = \begin{bmatrix} H_1^1 & H_1^2 & H_1^3 & 0 & 0 & 0 & 0 & \dots & 0 \\ H_2^1 & H_2^2 & H_2^3 & 0 & 0 & 0 & 0 & \ddots & \vdots \\ H_3^1 & H_3^2 & H_3^3 & 0 & 0 & 0 & 0 & \ddots & 0 & 0 \\ 0 & H_4^1 & H_4^2 & H_4^3 & 0 & \ddots & 0 & 0 & 0 & 0 \\ 0 & H_5^1 & 0 & H_5^2 & \ddots & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & H_{j-1}^C & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & H_j^1 & \dots & H_j^C & 0 & 0 \\ \vdots & \ddots & 0 & 0 & \vdots & \ddots & 0 & H_{j+1}^C & 0 & 0 \\ 0 & \dots & 0 & H_j^1 & \dots & H_j^{C-2} & 0 & H_j^{C-1} & H_j^C & 0 \end{bmatrix}$$

Fig. 5: Sparse transition $\mathbf{F} \in \mathbb{R}^{M \times M}$ and projection $\mathbf{H} \in \mathbb{R}^{N \times M}$ matrices, where $\mathbf{F}_j^i \in \mathbb{R}^{L \times L}$ and $\mathbf{H}_j^i \in \mathbb{R}^{B \times L}$ are the i th part of $\mathbf{F}_j \in \mathbb{R}^{L \times L(C-1)}$ and $\mathbf{H}_j \in \mathbb{R}^{B \times LC}$, respectively, corresponding to the i th entries in $\pi^i(j)$ and $\psi^i(j)$.

3.3 Inference

For our model it is possible to infer analytically the posterior $q_{c,\gamma}(\mathbf{a}|\mathbf{y})$ and the marginal likelihood $q_{c,\gamma}(\mathbf{y})$ used later for prediction and for hyperparameter estimation, respectively.

Proposition 3 (Posterior Approximation; Proof 12) *From the joint distribution, the latent function values \mathbf{f} can be integrated out yielding*

$$q_{c,\gamma}(\mathbf{a}, \mathbf{y}) = \int q_{c,\gamma}(\mathbf{f}, \mathbf{a}, \mathbf{y}) d\mathbf{f} = q_{c,\gamma}(\mathbf{y}|\mathbf{a})q_{c,\gamma}(\mathbf{a}) = \mathcal{N}(\mathbf{y}|\mathbf{H}\mathbf{a}, \mathbf{V})\mathcal{N}(\mathbf{a}|\mathbf{0}, \mathbf{S}^{-1})$$

with $\mathbf{V} = \bar{\mathbf{V}} + \sigma_n^2\mathbb{I} \in \mathbb{R}^{N \times N}$. The posterior can be analytically computed by

$$q_{c,\gamma}(\mathbf{a}|\mathbf{y}) = \frac{q_{c,\gamma}(\mathbf{a}, \mathbf{y})}{q_{c,\gamma}(\mathbf{y})} \propto q_{c,\gamma}(\mathbf{a}, \mathbf{y}) = \mathcal{N}(\mathbf{a}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}^{-1}(\mathbf{a}|\boldsymbol{\eta}, \boldsymbol{\Lambda}),$$

with $\boldsymbol{\Sigma}^{-1} = \boldsymbol{\Lambda} = \mathbf{T} + \mathbf{S} \in \mathbb{R}^{M \times M}$, $\boldsymbol{\mu} = \boldsymbol{\Sigma}\boldsymbol{\eta} \in \mathbb{R}^M$, $\boldsymbol{\eta} = \mathbf{H}^T\mathbf{V}^{-1}\mathbf{y} \in \mathbb{R}^M$ and $\mathbf{T} = \mathbf{H}^T\mathbf{V}^{-1}\mathbf{H} \in \mathbb{R}^{M \times M}$.

The posterior precision matrix $\boldsymbol{\Sigma}^{-1} = \mathbf{T} + \mathbf{S}$ inherits the sparsity pattern of the prior, since the addition of the projection precision $\mathbf{T} = \mathbf{H}^T\mathbf{V}^{-1}\mathbf{H}$ has the same sparsity structure, as depicted in Figs. 3 and 6. On the other hand, the posterior covariance $\boldsymbol{\Sigma}$ is dense, therefore it will be never explicitly fully computed. Instead, the sparse linear system of equations $\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} = \boldsymbol{\eta}$ can be efficiently solved for $\boldsymbol{\mu} = \boldsymbol{\Sigma}\boldsymbol{\eta}$. Further, in our CPoE model, the marginal likelihood $q_{c,\gamma}(\mathbf{y}|\boldsymbol{\theta})$ can be analytically computed by $\int q_{c,\gamma}(\mathbf{y}, \mathbf{a}) d\mathbf{a} = \mathcal{N}(\mathbf{0}, \mathbf{P})$ (see Proof 9) with the (dense) matrix $\mathbf{P} = \mathbf{H}\mathbf{S}^{-1}\mathbf{H}^T + \mathbf{V} \in \mathbb{R}^{N \times N}$, which is used in Section 3.6.2 for hyperparameter optimization. The posterior approximation $q_{c,\gamma}(\mathbf{a}|\mathbf{y})$ as well as the approximate marginal likelihood $q_{c,\gamma}(\mathbf{y})$ converge to the true distributions $p(\mathbf{a}|\mathbf{y})$ and $p(\mathbf{y})$, respectively, for $C \rightarrow J$. In particular, they correspond exactly to the posterior and marginal likelihood of full GP and sparse global GP with $\lfloor \gamma N \rfloor$ inducing points for $C = J, \gamma = 1$ and $C = J, \gamma < 1$, respectively.

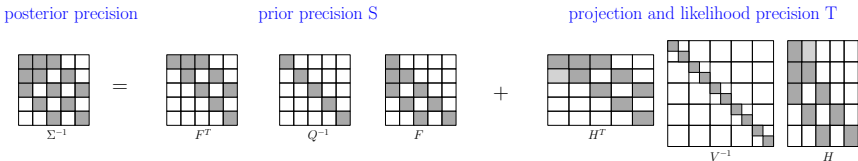


Fig. 6: Sparse posterior precision approximation.

3.4 Prediction

The final predictive posterior distribution is obtained by an adaptation of the PoE aggregation in (3). The main idea is to consistently aggregate

weighted local predictions form the experts, such that the correlations between them are taken into account resulting in a smooth and continuous predictive distribution.

Proposition 4 (Prediction Aggregation; Proof 17) *Similarly to the PoE aggregation (3), we define the final predictive posterior distribution for a query point $\mathbf{x}_* \in \mathbb{R}^D$ as*

$$q_{c,\gamma}(f_*|\mathbf{y}) = \prod_{j=C}^J q_{c,\gamma}(f_{*j}|\mathbf{y})^{\beta_{*j}}, \quad (8)$$

involving the local predictions $q_{c,\gamma}(f_{*j}|\mathbf{y}) = \mathcal{N}(m_{*j}, v_{*j})$ and weights $\beta_{*j} \in \mathbb{R}$ defined in Prop. 8 and Def. 5, respectively. Moreover, the distribution $q_{c,\gamma}(f_*|\mathbf{y}) = \mathcal{N}(m_*, v_*)$ with $m_* = v_* \sum_{j=C}^J \beta_{*j} \frac{m_{*j}}{v_{*j}}$ and $\frac{1}{v_*} = \sum_{j=C}^J \frac{\beta_{*j}}{v_{*j}}$ is analytically available. The final noisy prediction is $p(y_*|\mathbf{y}) = \mathcal{N}(m_*, v_* + \sigma_n^2)$.

The graphical model corresponding to this prediction procedure is depicted in Fig. 4b and A3 in the Appendix. Further, the local predictions $q_{c,\gamma}(f_{*j}|\mathbf{y})$ in Equation (8) are based on the region $\psi(j)$, where the correlations are modelled and can be computed as $q_{c,\gamma}(f_{*j}|\mathbf{y}) = \int p(f_{*j}|\mathbf{a}_{\psi(j)}) q_{c,\gamma}(\mathbf{a}_{\psi(j)}|\mathbf{y}) d\mathbf{a}_{\psi(j)}$ involving the local posteriors $q_{c,\gamma}(\mathbf{a}_{\psi(j)}|\mathbf{y}) = \mathcal{N}(\boldsymbol{\mu}_{\psi(j)}, \boldsymbol{\Sigma}_{\psi(j)})$ and the predictive conditional $p(f_{*j}|\mathbf{a}_{\psi(j)})$, as thoroughly shown in Proposition 8 in the Appendix. Thereby, the local posteriors with mean $\boldsymbol{\mu}_{\psi(j)}$ and covariance entries $\boldsymbol{\Sigma}_{\psi(j)}$ could be obtained from the corresponding entries $\psi(j)$ of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. However, computing explicitly some entries in the dense covariance $\boldsymbol{\Sigma}$ based on the sparse precision $\boldsymbol{\Sigma}^{-1}$ is not straightforward since in the inverse the blocks are no longer independent. However, we can exploit the particular sparsity and block-structure of our precision matrix and obtain an efficient implementation of this part, which is key to achieve a competitive performance of our algorithm. More details are given in the Appendix in Section A.2.

Definition 5 (Aggregation Weights) The input depending weights $\beta_{*j} = \beta_j(X_*)$ at query point X_* model the impact of expert j . In particular, the unnormalized weights

$$\bar{\beta}_{*j} = H[p(f_*)] - H[p(f_{*j}|\mathbf{y})] = \frac{1}{2} \log \left(\frac{v_{*0}}{v_{*j}} \right),$$

are set to the difference in entropy H (B10) before and after seeing the data similarly proposed by [12]. Thereby, the predictive prior is $p(f_*) = \mathcal{N}(0, v_{*0})$ with $v_{*0} = \mathbf{k}_{X_* X_*}$ and the predictive posterior defined in Prop. 8. The normalized weights are then obtained by $\beta_{*j} = b^{-1} \bar{\beta}_{*j}^Z$ where $b = \sum_{j=C}^J \bar{\beta}_{*j}^Z$ and $Z = \log(N)C$.

These weights bring the flexibility of increasing or reducing the importance of the experts based on the predictive uncertainty. However, independent of the particular weights, our aggregation of the predictions is consistent since it is based on the *covariance intersection* method [20], which is useful for combining several estimates of random variables with known mean and variance but unknown correlation between them.

3.5 Properties

Proposition 5 (Equality; Proof 3) *Our model correlated Product of Experts $CPoE(C, \gamma)$ is equal to full GP for $C = J$ and $\gamma = 1$. For $\gamma < 1$, our model correspond to sparse global GP with $M_g = \lfloor \gamma N \rfloor$ inducing points. Further, with $C = 1$ and $\gamma = 1$, our model is equivalent to independent PoEs. That is, we have*

$$CPoE(J, 1) = GP; \quad CPoE(J, \gamma) = SGP(\lfloor \gamma N \rfloor); \quad CPoE(1, 1) = GPoE^*,$$

where SGP refers to the FITC model [7] and $GPoE^*$ correspond to $GPoE$ [12] with slightly different weights ($Z = 1$) in the prediction.

In Section A.1 in the Appendix we present a generalization of our model, so that $CPoE(J, \gamma)$ correspond to a range of other well known versions of sparse global GP by changing the projection distribution and adding a correction term in the log marginal likelihood similarly discussed in [11] for the global case. For instance, we can extend our model analogously to the variational version of [5].

For correlations between the limiting cases $C = 1$ and $C = J$, we investigate the difference in KL of the true GP model with $CPoE(C, \gamma)$ and $CPoE(C_2, \gamma)$ for $1 \leq C \leq C_2 \leq J$. For that reason, we define the difference in KL between the true distribution of \mathbf{x} and two different approximate distributions, i.e.

$$\mathbb{D}_{(C, C_2)}[\mathbf{x}] = KL[p(\mathbf{x}) \parallel q_{c, \gamma}(\mathbf{x})] - KL[p(\mathbf{x}) \parallel q_{c_2, \gamma}(\mathbf{x})].$$

Similarly, the difference in KL for a conditional distribution is defined in Eq. (B15). Using these definitions, we show that the approximation quality of the prior $q_{c, \gamma}(\mathbf{a})$ and projection approximation $q_{c, \gamma}(\mathbf{f}|\mathbf{a})$ monotonically improves for $C \rightarrow J$, so that the KL between the true joint distribution $p(\mathbf{a}, \mathbf{f}, \mathbf{y})$ and our approximate joint distribution $q_{c, \gamma}(\mathbf{a}, \mathbf{f}, \mathbf{y})$ is decreasing for $C \rightarrow J$.

Proposition 6 (Decreasing KL; Proof 6) *For any predecessor structure π_C and any $0 < \gamma \leq 1$ and $1 \leq C \leq C_2 \leq J$, the difference in KL of the marginal prior, projection and data likelihood are non negative, i.e.*

$$\mathbb{D}_{(C, C_2)}[\mathbf{a}] \geq 0, \quad \mathbb{D}_{(C, C_2)}[\mathbf{f}|\mathbf{a}] \geq 0, \quad \mathbb{D}_{(C, C_2)}[\mathbf{y}|\mathbf{f}] = 0,$$

so that the joint difference in KL is also non-negative

$$\mathbb{D}_{(C, C_2)}[\mathbf{a}, \mathbf{f}, \mathbf{y}] = \mathbb{D}_{(C, C_2)}[\mathbf{a}] + \mathbb{D}_{(C, C_2)}[\mathbf{f}|\mathbf{a}] + \mathbb{D}_{(C, C_2)}[\mathbf{y}|\mathbf{f}] \geq 0.$$

Moreover, we can quantify the approximation quality, in particular $\mathbb{D}_{(C, C_2)}[\mathbf{a}] = \frac{1}{2} \log \frac{|\mathbf{Q}_C|}{|\mathbf{Q}_{C_2}|}$ and $\mathbb{D}_{(C, C_2)}[\mathbf{f}|\mathbf{a}] = \frac{1}{2} \log \frac{|\mathbf{V}_C|}{|\mathbf{V}_{C_2}|}$.

The last statement demonstrates that our CPoE model is a sound GP prior precision approximation, which converges monotonically to the true prior for $C \rightarrow J$. The decreasing KL of the joint prior is depicted in Fig. 7 together with the decreasing KL of the posterior, marginal likelihood and predictive posterior. More details and proofs are given in Appendix C.

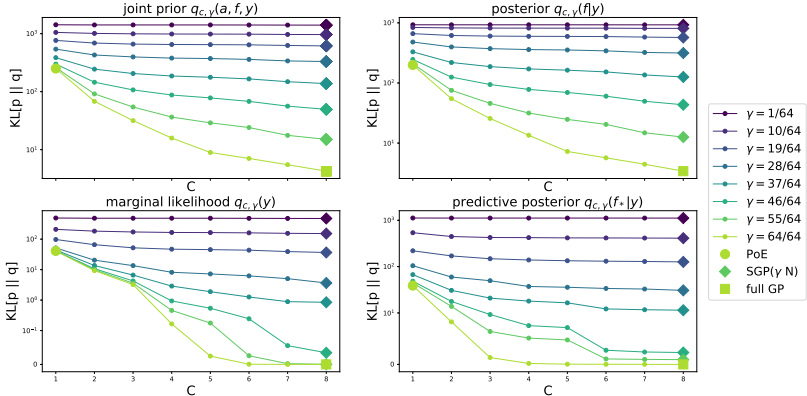


Fig. 7: Decreasing $\text{KL}[p||q]$ between true distribution p of full GP and approximate distribution $q = q_{c,\gamma}$ of CPoE for increasing values of C and γ for the joint prior, posterior, marginal likelihood and predictive posterior for synthetic GP data ($N = 1024$, $D = 2$, SE kernel).

3.6 Computational Details

3.6.1 Graph

The graphical model in Section 3.1 is generically defined and several choices are left for completely specifying the graph $\mathcal{G}(V, E)$ for a particular dataset: the partition method, the ordering of the partition, the selection of the predecessors and the local inducing points. We tried to make these choices as simple and straightforward as possible with focus on computational efficiency, however, there might be more sophisticated heuristics. Concretely, we use KD-trees [30] for partitioning the data \mathcal{D} into J regions and the ordering starts with a random partition which is then greedily extended by the closest partition in euclidean distance (represented by the mean of the inducing points). The $L \leq B$ inducing inputs $\mathbf{A}_j \in \mathbb{R}^{L \times D}$ of the j th partition (or expert) can be in principle arbitrary, however, in this work they are chosen as a random subset of the data inputs $\mathbf{X}_j \in \mathbb{R}^{B \times D}$ of the j th expert for the sake of simplicity. For the predecessors (block-)indices π_C , the $C - 1$ closest partitions among the previous (according to the ordering) predecessors in euclidean distance are greedily selected. These concepts are illustrated for a toy example in Fig. 8.

3.6.2 Hyperparameter Estimation

In Section 3, we introduced CPoE for fixed hyperparameters θ where implicitly all distributions are conditioned on θ , however, we omitted the dependencies on θ in the most cases for the sake of brevity. Similar to full GP, sparse GP or PoEs, the *log marginal likelihood (LML)* can be used as an objective function for optimizing the few hyperparameters θ . The log of the marginal likelihood of our model formulated in Section 3.3 is $\mathcal{L}(\theta) = \log q(\mathbf{y}|\theta) = \log \mathcal{N}(\mathbf{0}, \mathbf{P})$ with $\mathbf{P} = \mathbf{H}\mathbf{S}^{-1}\mathbf{H}^T + \mathbf{V}$ which can be efficiently computed as detailed in

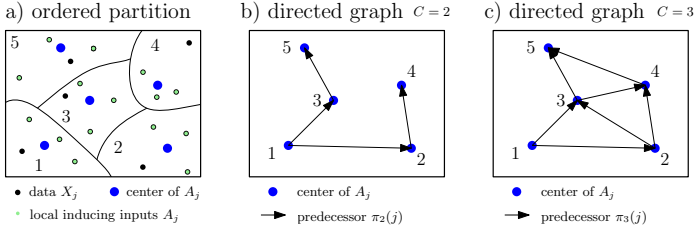


Fig. 8: Toy example for partition, local inducing points, predecessors and directed graph illustrated for $D = 2$ with $J = 5$ experts/partitions each with $B = 4$ samples, $\gamma = 0.75$ and thus $L = 3$ local inducing points. In a) the ordered partition with the data (black), local inducing points (green) and their mean (blue) are depicted. In b) and c) the directed graph for $C = 2$ and $C = 3$ are shown with corresponding predecessors $\pi_2(1) = \{\}$, $\pi_2(2) = \{1\}$, $\pi_2(3) = \{1\}$, $\pi_2(4) = \{2\}$, $\pi_2(5) = \{3\}$ and $\pi_3(1) = \{\}$, $\pi_3(2) = \{1\}$, $\pi_3(3) = \{1, 2\}$, $\pi_3(4) = \{2, 3\}$, $\pi_3(5) = \{3, 4\}$, respectively. In the previous example, π_3 is consecutive and π_2 is non-consecutive.

Section A.3 and can be used for *deterministic optimization* with full batch \mathbf{y} for moderate sample size N . However, in order to scale this parameter optimization part to larger number of samples N in a competitive time, *stochastic optimization* techniques exploiting subsets of data have to be developed similarly done for the global sparse GP model (SVI [9]; REC [11]; IF [10]). We adapt the hybrid approach IF of [10] where we can also exploit an independent factorization of the log marginal likelihood which decomposes into a sum of J terms, so that it can be used for stochastic optimization. This constitutes a very fast and accurate alternative for our method as shown in the Appendix A.3 and will also be exploited in Section 4 for large data sets. Alternatively to the log marginal likelihood (LML) maximization as presented above, the *maximum a posteriori* (MAP) estimator for $\boldsymbol{\theta}$ can be used. This means, that some suitable prior on the hyperparameters are introduced, as explained in Section A.3.3 and an example is presented in Section 4.4.

3.6.3 Complexity

The time complexity for computing the posterior and the marginal likelihood in our algorithm is dominated by J operations which are cubic in LC (inversion, matrix-matrix multiplication, determinants). This leads to $\mathcal{O}(NB^2\alpha^3)$ and $\mathcal{O}(NB\alpha^2)$ for time and space complexity, respectively, where we define the approximation quality parameter $\alpha = C\gamma$. Similarly, for N_t testing points the time and space complexities are $\mathcal{O}(NB\alpha^2N_t)$ and $\mathcal{O}(N\alpha N_t)$ (an approach to remove the dependency of N is outlined in A.4). In Table 1, the asymptotic complexities of our model together with other GP algorithms are indicated. It is interesting that for $\alpha = 1$, our algorithm has the same asymptotic complexity for training as sparse global GP with $M_g = B$ global inducing points but we can have $M = LJ = \gamma BJ = \gamma N$ total local inducing points! Thus, our approach allows much more total local inducing points M in the order of N

| | full GP | sparse GP | PoE | CPoE |
|--------------------|-------------------------|------------------------------|-------------------------|------------------------------|
| time | $\mathcal{O}(N^3)$ | $\mathcal{O}(NM_g^2)$ | $\mathcal{O}(NB^2)$ | $\mathcal{O}(NB^2\alpha^3)$ |
| space | $\mathcal{O}(N^2)$ | $\mathcal{O}(NM_g)$ | $\mathcal{O}(NB)$ | $\mathcal{O}(NB\alpha^2)$ |
| time _t | $\mathcal{O}(N^2N_t)$ | $\mathcal{O}(M_g^2N_t)$ | $\mathcal{O}(NBN_t)$ | $\mathcal{O}(NBN_t\alpha^2)$ |
| space _t | $\mathcal{O}(NN_t)$ | $\mathcal{O}(M_gN_t)$ | $\mathcal{O}(NN_t)$ | $\mathcal{O}(NN_t\alpha)$ |
| #pars | $ \boldsymbol{\theta} $ | $MD + \boldsymbol{\theta} $ | $ \boldsymbol{\theta} $ | $ \boldsymbol{\theta} $ |

Table 1: Complexity for training, pointwise predictions for N_t points and number of optimization parameters for different GP algorithms.

(e.g. $M = 0.5N$ with $C = 2$) whereas for sparse global GP usually $M_g \ll N$. This has the consequence that the local inducing points can cover the input space much better and therefore represent much more complicated functions. As a consequence, there is also no need to optimize the local inducing points resulting in much fewer parameters to optimize. Consider the following example with $N = 10'000$ in $D = 10$ dimensions. Suppose a sparse global GP model with $M_g = 500$ global inducing points. A CPoE model with the same asymptotic complexity has a batch size $B = M_g = 500$ and $\alpha = 1$. Therefore, we have $J = \frac{N}{B} = 20$ experts and we choose $C = 2$ and $\gamma = \frac{1}{2}$ such that we obtain $L = \gamma B = 250$ local inducing points per experts and $M = \gamma N = 5'000$ total local inducing points! Further, the number of hyperparameters to optimize with a SE kernel is for global sparse GP $M_g D + |\boldsymbol{\theta}| = 5012$, whereas for CPoE there are only $|\boldsymbol{\theta}| = 12$. For an extended version of this section consider [A.4](#) in the Appendix.

4 Comparison

In this section, we compare the performance with competitor methods for GP approximations using *synthetic* and several *real world* datasets as summarized in [Table 3a](#). Moreover, we provide a comparison to non-GP regression methods as well as an application about probabilistic time series prediction both exploiting non-trivial kernels. More details about the experiments and implementations are provided in [Sections A.6, A.7](#) and [F](#) in the Appendix.

4.1 Synthetic Data

First, we examine the accuracy vs. time performance of different GP algorithms for fixed hyperparameters in a simulation study with *synthetic GP data*. We generated $N = 8192$ data samples in $D = 2$ with 5 repetitions from the sum of two SE kernels with a shorter and longer lengthscale such that both global and local patterns are present in the data (compare [Fig. A5](#)). In [Fig. 9](#) the mean results are shown for the KL and RMSE to full GP, the 95%-coverage and the log marginal likelihood against time in seconds. The results for sparse GP with increasing number of global inducing points M are shown in blue,

the results for minVar, GPoE and BCM for increasing number of experts J are depicted in red, cyan and magenta, respectively. For CPoE, the results for increasing correlations C are shown in green. We observe superior performance of our method compared to competitors in terms of accuracy compared to full GP vs. time. Moreover, one can observe that the confidence information of our model are reliable already for small approximation orders since it is based on the consistent covariance intersection method. A precise description of the experiment is provided in Section A.7.1 in the Appendix.

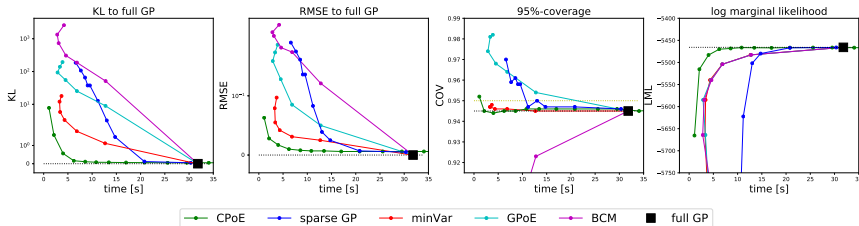


Fig. 9: Average accuracy vs. time performance of different GP algorithms.

4.2 Real World Data

Second, we benchmark our method with 10 real world datasets as summarized in Table 3a and more details are given in Section A.7.2 in the Appendix (e.g. how to access and pre-process the data). For the 5 smaller datasets in the first block we use deterministic parameter optimization for which the average results over 10 training/testing splits are depicted in Table 2. In particular, the KL to full GP (left) and time (right) for different GP methods are shown. Similarly, the average accuracy and times for the 4 larger datasets in the second block where stochastic parameter optimization is exploited can be found in Table A4 in the Appendix.

In general, the local methods perform better than the global sparse method. Further, the performance of our correlated PoEs is superior to the one of independent PoEs for all datasets. In particular, the KL to full GP can be continuously improved for increasing degree of correlation, i.e. larger C values. The time for CPoE(1) is comparable with the independent PoEs and for increasing C , our approximation has a moderate increase in time with a significant decrease in KL. For more details about the experiments consider Section A.7.2 in the Appendix and more results including standard deviations are provided in Appendix F.

| | KL | | | | | time | | | | |
|----------------|-------------|------------|-------------|------------|-------------|----------|------|-------|---------|-------|
| | concrete | mg | space | abalone | kin | concrete | mg | space | abalone | kin |
| fullGP | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 7.3 | 25.5 | 114.8 | 237.9 | 161.5 |
| SGP(100) | 352.9 | 9.9 | 108.1 | 15.6 | 603.7 | 36.4 | 14.4 | 46.6 | 58.9 | 42.2 |
| minVar | 122.2 | 19.4 | 63.6 | 25.1 | 211.0 | 1.5 | 2.0 | 7.2 | 6.4 | 9.3 |
| GPoE | 174.4 | 54.2 | 98.0 | 50.3 | 342.3 | 1.4 | 1.9 | 7.2 | 6.3 | 9.4 |
| GRBCM | 224.6 | 69.1 | 105.6 | 36.4 | 129.8 | 1.7 | 2.3 | 6.5 | 7.6 | 11.9 |
| CPoE(1) | 111.1 | 12.2 | 63.0 | 16.8 | 152.4 | 1.5 | 2.1 | 7.8 | 6.4 | 9.2 |
| CPoE(2) | 89.6 | 8.4 | 36.5 | 8.1 | 79.9 | 2.1 | 2.8 | 10.6 | 7.5 | 12.9 |
| CPoE(3) | 82.2 | 7.8 | 36.3 | 6.2 | 46.9 | 2.5 | 3.1 | 12.9 | 9.3 | 19.8 |
| CPoE(4) | 79.5 | 7.6 | 36.0 | 4.7 | 32.8 | 2.8 | 3.3 | 14.9 | 10.4 | 27.8 |

Table 2: Average KL to full GP (left) and time (right) for different GP methods and 5 datasets with 10 repetitions. More results are provided in Appendix F.

| | N | D | N_{test} | J | | KL | KL IN | KL OUT | time |
|------------|----------|-----|------------|-----|----------------|-------------|-------------|-------------|------|
| | concrete | 927 | 8 | 103 | | 4 | full GP | 0.0 | 0.0 |
| mg | 1247 | 6 | 138 | 8 | SGP(100) | 120.9 | 110.5 | 146.7 | 56.3 |
| space | 2797 | 6 | 310 | 8 | SGP(200) | 114.9 | 65.6 | 238.3 | 75.2 |
| abalone | 3760 | 8 | 417 | 16 | minVar | 503.0 | 406.5 | 744.5 | 20.7 |
| kin | 5192 | 8 | 3000 | 16 | GPoE | 328.0 | 336.0 | 307.9 | 20.4 |
| | | | | | GRBCM | 393.4 | 382.1 | 421.8 | 28.2 |
| kin2 | 7373 | 8 | 819 | 16 | CPoE(1) | 289.5 | 255.1 | 375.5 | 20.5 |
| cadata | 19640 | 8 | 1000 | 64 | CPoE(2) | 113.1 | 108.5 | 124.3 | 36.8 |
| sarcos | 43484 | 21 | 1000 | 128 | CPoE(3) | 86.4 | 61.9 | 147.6 | 39.7 |
| casp | 44730 | 9 | 1000 | 128 | CPoE(4) | 58.3 | 59.4 | 55.5 | 52.9 |
| elecdemand | 2184 | 3 | 15288 | 13 | | | | | |

(a) Description of datasets.

(b) KL to full GP and time of different methods.

Table 3: Summary of used datasets and results for the *elecdemand* time series.

4.3 Comparison to non-GP methods

Third, we compare our probabilistic regression method CPoE to other popular non-GP regression methods, in particular, dense neural networks (MLPs), eXtreme Gradient Boosting (XGboost) and linear regression¹. We use three different architectures for the neural networks, that is, MLP(100,100), MLP(500,500), MLP(100,100,100), where the numbers in the parentheses correspond to the number of hidden nodes per hidden layer. Moreover, we used ADAM optimizer with learning rate 0.01. For XGboost(*max_depth*, *n_estimators*, *learning_rate*), we use XGboost(3, 100, 0.1). All these hyperparameters are chosen in primary experiments so that those methods obtain advantageous test performance. For our CPoE method, we use a SE kernel as in the previous sections, and in addition, we run the algorithm with a more flexible kernel, namely

$$k_{\theta}(\mathbf{x}_1, \mathbf{x}_2) = k_{SM_1}(\mathbf{x}_1, \mathbf{x}_2) + k_{SM_2}(\mathbf{x}_1, \mathbf{x}_2) + k_{MLP}(\mathbf{x}_1, \mathbf{x}_2) + k_{LIN}(\mathbf{x}_1, \mathbf{x}_2), \quad (9)$$

where k_{SM_i} is a spectral-mixture kernel [31], k_{MLP} an (infinite) wide 1-hidden-layer neural network kernel [32] and k_{LIN} a linear kernel. We run full GP for

¹We use the algorithms in <https://scikit-learn.org>.

| | concrete | mg | space | abalone | kin | cadata | sarcos | casp |
|---------------------|--------------|--------------|--------------|--------------|-------------|-------------|--------------|--------------|
| fullGP-SE | 0.311 | 0.511 | 0.471 | 0.635 | 0.267 | | | |
| fullGP-FLEX | 0.254 | 0.509 | 0.455 | 0.638 | 0.28 | | | |
| CPoE(1)-SE | 0.333 | 0.508 | 0.506 | 0.637 | 0.31 | 0.476 | 0.099 | 0.597 |
| CPoE(2)-SE | 0.326 | 0.512 | 0.49 | 0.634 | 0.292 | 0.47 | 0.1 | 0.59 |
| CPoE(3)-SE | 0.323 | 0.513 | 0.489 | 0.634 | 0.28 | 0.47 | 0.099 | 0.59 |
| CPoE(1)-FLEX | 0.266 | 0.511 | 0.631 | 0.687 | 0.334 | 0.456 | 0.094 | 0.525 |
| CPoE(2)-FLEX | 0.259 | 0.515 | 0.446 | 0.669 | 0.315 | 0.423 | 0.094 | 0.522 |
| CPoE(3)-FLEX | 0.255 | 0.516 | 0.444 | 0.659 | 0.303 | 0.42 | 0.092 | 0.522 |
| MLP(100-100) | 0.289 | 0.525 | 0.482 | 0.652 | 0.287 | 0.456 | 0.117 | 0.591 |
| MLP(500-500) | 0.292 | 0.522 | 0.475 | 0.761 | 0.284 | 0.485 | 0.097 | 0.577 |
| MLP(100-100-100) | 0.285 | 0.531 | 0.476 | 0.762 | 0.299 | 0.485 | 0.106 | 0.585 |
| XGboost | 0.323 | 0.545 | 0.543 | 0.65 | 0.667 | 0.474 | 0.251 | 0.767 |
| LinReg | 0.626 | 0.633 | 0.645 | 0.66 | 0.765 | 0.605 | 0.27 | 0.854 |

Table 4: Average *RMSE* for our CPoE methods compared to non-GP regression methods. The methods ending with SE were run with a squared-exponential and a flexible kernel (9), respectively. Best method (beside GP full) is indicated in bold.

smaller datasets as comparison. The average *RMSE*, *ABSE* and *time* results are provided in Tables 4, F22 and F23, respectively. For instance in Table 4, we can observe that the GP approximation methods using either a SE kernel or a more flexible kernel achieve competitive performance.

Finally, we would like to emphasize that our probabilistic CPoE model provides a predictive *distribution*, that is, it models the predictive uncertainty and can thus provide reliable credible-intervals. Computing also the predictive variances is a harder task than only computing the predictive means, as the most other regression algorithms do. Therefore, the slightly higher computational times (Table F23) for similar accuracy (Tables 4 and F22) are very reasonable in our opinion. More detailed results are given in Tables F15-F21 and on github.²

4.4 Time series application

In this section, our method is applied on time series data with covariates using a non-stationary kernel together with priors on the hyperparameters as discussed in Section A.3.3 by using MAP estimation. A recent work [33] demonstrates that GPs constitute a competitive method for modelling time series using a sum of kernels including priors on the hyperparameters, which are previously learnt from a large set of different time series. We adapt their idea by using the same priors and a slightly modified kernel. In particular, for two data points $\mathbf{x}_1 = [t_1, x_{1,2}, \dots, x_{1,D}]$ and $\mathbf{x}_2 = [t_2, x_{2,2}, \dots, x_{2,D}]$, we model the kernel as the sum of 4 components

$$k_{\theta}(\mathbf{x}_1, \mathbf{x}_2) = k_{P_1}(t_1, t_2) + k_{P_2}(t_1, t_2) + k_{SM}(t_1, t_2) + k_{SE}(\mathbf{x}_1, \mathbf{x}_2),$$

²https://github.com/manschuer/CPoE/experiments/comparison_non-GP.ipynb

where k_{P_1} and k_{P_2} are standard periodic kernels with period p_1 and p_2 , respectively, k_{SM} a spectral-mixture kernel and k_{SE} a squared-exponential kernel. Note that, the former 3 kernels only depend on the first variable corresponding to time, whereas the SE-kernel depends on all variables, thus models the influence of the additional variables. With our CPoE model it is straightforward to handle time series with covariates, as opposed to other time series methods [33–36]. We demonstrate the MAP estimation for θ on the *elecdemand* time series ([37], Table 3a), which contains the electricity demand as response y together with the time as the first variable x_1 , the the corresponding temperature as x_2 and the variable whether it is a working day as x_3 which is depicted in the plots in Fig. 10 on the left, where we shifted the first and third variable in the second plot for the sake of clarity. Similarly as in the previous section, we run full GP, SGP, PoEs and CPoE and optimized the hyperparameter deterministically using the MAP as objective function taking into account the priors. The results are provided in Table 3b and in Fig. 10 on the right, which again show very competitive performance also for a general kernel with priors on the hyperparameters. More details about the experiment is given in Section A.7.3 in the Appendix.

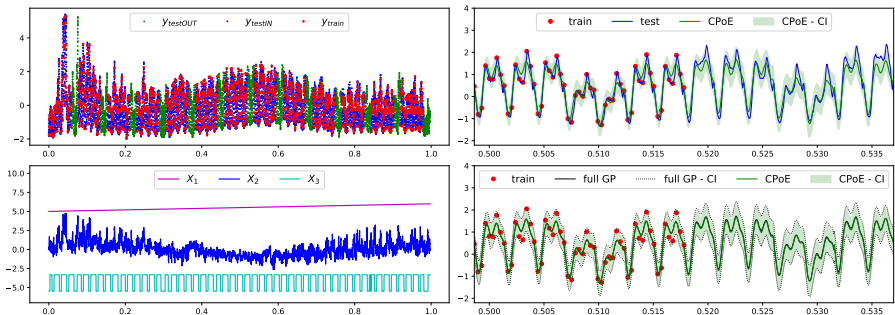


Fig. 10: Time series data with covariates and prior on hyperparameters.

5 Conclusion

In this paper, we introduce a novel GP approximation algorithm CPoE, where the degree of approximation can be adjusted by a locality and a sparsity parameter, so that the proposed method recovers independent PoEs, sparse global GP and full GP. Thereby, our method consistently approximates full GP, in particular, we proved that increasing the correlations between the experts decreases monotonically the KL of the joint prior of full GP to them of our model. The presented algorithm has only a few hyperparameters, which allows an efficient deterministic and stochastic optimization. Further, our presented algorithm works with a general kernel, with several variables and also priors on the hyperparameters can be included. Moreover, the time and space complexity is linear in the number of experts and number of data samples, which

makes it highly scalable. This is demonstrated with efficient implementations, so that a dataset with several ten thousands of samples can be processed in around a minute on a standard laptop. In several experiments with synthetic and real world data, superior performance in accuracy vs. time sense compared to state-of-the-art methods, is demonstrated, which makes our algorithm a competitive GP regression approximation method.

Our approach could be enhanced in several directions. The first improvement would be more practical. While the current implementation of our algorithm works very competitively for moderate large datasets (on a standard laptop), further work has been done to scale it up to very large datasets. The current limitations are particularly factorizing the sparse block Cholesky matrices. We are convinced, that the theoretical properties of our algorithm - in particular the linearity in the number of experts and data samples - enables large scale implementations when exploiting more low level linear algebra tools. Another interesting direction would be to investigate the connection of our sparse precision matrix to state space systems, such that sequential learning algorithm could be exploited, which is briefly outlined in D. Further, it would be interesting to apply variational methods to our model, so that a connection to full GP in a posterior sense might be established, where some ideas are outlined in A.1.

Declarations

- *Funding*: This work is supported by the Swiss National Research Programme 75 "Big Data" (NRP 75) with grant number 167199.
- *Conflicts of interest/Competing interests*: The authors do not have any conflict of interests or competing interests.
- *Ethics approval*: This paper does not require special ethics approval.
- *Consent to participate*: Not Applicable.
- *Consent for publication*: Not Applicable.
- *Availability of data and material*: The data used in this paper is available on public data repositories as indicated in the supplementary material.
- *Code availability*: We provide the code of the proposed algorithms in the paper on <https://github.com/manschuer/CPoE> including descriptions how to use it and the performed experiments.
- *Authors' contributions*: Manuel Schürch conceived the presented idea, developed the theory, carried out the experiments and had the lead in writing the manuscript. Dario Azzimonti and Alessio Benavoli assisted by developing the idea and theory, designed the experiments, wrote part of the manuscript and reviewed drafts of the paper. Marco Zaffalon supervised the project, gave inputs for writing the manuscript and reviewed drafts of the paper.

References

- [1] Rasmussen, C.E., Williams, C.K.: Gaussian Processes for Machine Learning vol. 1. MIT press, Cambridge (2006)

- [2] Csató, L., Opper, M.: Sparse online gaussian processes. *Neural computation* **14**(3), 641–668 (2002)
- [3] Quiñonero-Candela, J., Rasmussen, C.E.: A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research* **6**(Dec), 1939–1959 (2005)
- [4] Seeger, M., Williams, C., Lawrence, N.: Fast forward selection to speed up sparse gaussian process regression. In: *Artificial Intelligence and Statistics* 9 (2003)
- [5] Titsias, M.: Variational learning of inducing variables in sparse gaussian processes. In: *Artificial Intelligence and Statistics*, pp. 567–574 (2009)
- [6] Bui, T.D., Yan, J., Turner, R.E.: A unifying framework for sparse gaussian process approximation using power expectation propagation. *Journal of Machine Learning Research* **18**, 1–72 (2017)
- [7] Snelson, E., Ghahramani, Z.: Sparse gaussian processes using pseudo-inputs. In: *Advances in Neural Information Processing Systems*, pp. 1257–1264 (2006)
- [8] Bui, T.D., Nguyen, C., Turner, R.E.: Streaming sparse gaussian process approximations. In: *Advances in Neural Information Processing Systems*, pp. 3301–3309 (2017)
- [9] Hensman, J., Fusi, N., Lawrence, N.D.: Gaussian processes for big data. In: *Conference for Uncertainty in Artificial Intelligence* (2013)
- [10] Kania, L., Schürch, M., Azzimonti, D., Benavoli, A.: Sparse information filter for fast gaussian process regression. *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases* (2021)
- [11] Schürch, M., Azzimonti, D., Benavoli, A., Zaffalon, M.: Recursive estimation for sparse gaussian process regression. *Automatica* **120**, 109127 (2020)
- [12] Fleet, Y.C.D.J.: Generalized product of experts for automatic and principled fusion of gaussian process predictions. *arXiv preprint arXiv:1410.7827* (2014)
- [13] Deisenroth, M.P., Ng, J.W.: Distributed gaussian processes. *arXiv preprint arXiv:1502.02843* (2015)
- [14] Hinton, G.E.: Training products of experts by minimizing contrastive divergence. *Neural computation* **14**(8), 1771–1800 (2002)

- [15] Rullièrè, D., Durrande, N., Bachoc, F., Chevalier, C.: Nested kriging predictions for datasets with a large number of observations. *Statistics and Computing* **28**(4), 849–867 (2018)
- [16] Tresp, V.: A bayesian committee machine. *Neural computation* **12**(11), 2719–2741 (2000)
- [17] Liu, H., Cai, J., Wang, Y., Ong, Y.S.: Generalized robust bayesian committee machine for large-scale gaussian process regression. In: *International Conference on Machine Learning*, pp. 3131–3140 (2018). PMLR
- [18] Wang, K., Pleiss, G., Gardner, J., Tyree, S., Weinberger, K.Q., Wilson, A.G.: Exact gaussian processes on a million data points. *Advances in Neural Information Processing Systems* **32**, 14648–14659 (2019)
- [19] Liu, H., Ong, Y.-S., Shen, X., Cai, J.: When gaussian process meets big data: A review of scalable gps. *IEEE transactions on neural networks and learning systems* **31**(11), 4405–4423 (2020)
- [20] Julier, S.J., Uhlmann, J.K.: A non-divergent estimation algorithm in the presence of unknown correlations. In: *Proceedings of the 1997 American Control Conference (Cat. No. 97CH36041)*, vol. 4, pp. 2369–2373 (1997). IEEE
- [21] Durrande, N., Adam, V., Bordeaux, L., Eleftheriadis, S., Hensman, J.: Banded matrix operators for gaussian markov models in the automatic differentiation era. In: *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 2780–2789 (2019). PMLR
- [22] Grigorievskiy, A., Lawrence, N., Särkkä, S.: Parallelizable sparse inverse formulation gaussian processes (spingp). In: *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6 (2017). IEEE
- [23] Bui, T.D., Turner, R.E.: Tree-structured gaussian process approximations. *Advances in Neural Information Processing Systems* **27**, 2213–2221 (2014)
- [24] Datta, A., Banerjee, S., Finley, A.O., Gelfand, A.E.: Hierarchical nearest-neighbor gaussian process models for large geostatistical datasets. *Journal of the American Statistical Association* **111**(514), 800–812 (2016)
- [25] Katzfuss, M., Guinness, J.: A general framework for vecchia approximations of gaussian processes. *Statistical Science* **36**(1), 124–141 (2021)
- [26] Buschjäger, S., Liebig, T., Morik, K.: Gaussian model trees for traffic imputation. In: *ICPRAM*, pp. 243–254 (2019)

- [27] Nakai-Kasai, A., Tanaka, T.: Nested aggregation of experts using inducing points for approximated gaussian process regression. *Machine Learning*, 1–24 (2021)
- [28] Masoudnia, S., Ebrahimpour, R.: Mixture of experts: a literature survey. *Artificial Intelligence Review* **42**(2), 275–293 (2014)
- [29] Trapp, M., Peharz, R., Pernkopf, F., Rasmussen, C.E.: Deep structured mixtures of gaussian processes. In: *International Conference on Artificial Intelligence and Statistics*, pp. 2251–2261 (2020). PMLR
- [30] Maneewongvatana, S., Mount, D.M.: On the efficiency of nearest neighbor searching with data clustered in lower dimensions. In: *International Conference on Computational Science*, pp. 842–851 (2001). Springer
- [31] Wilson, A., Adams, R.: Gaussian process kernels for pattern discovery and extrapolation. In: *International Conference on Machine Learning*, pp. 1067–1075 (2013). PMLR
- [32] Neal, R.M.: Bayesian learning for neural networks. PhD thesis, CAN (1995). AAINN02676
- [33] Corani, G., Benavoli, A., Zaffalon, M.: Time series forecasting with gaussian processes needs priors. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 103–117 (2021). Springer
- [34] Benavoli, A., Corani, G.: State space approximation of gaussian processes for time series forecasting
- [35] Sarkka, S., Solin, A., Hartikainen, J.: Spatiotemporal learning via infinite-dimensional bayesian filtering and smoothing: A look at gaussian process regression through kalman filtering. *IEEE Signal Processing Magazine* **30**(4), 51–61 (2013)
- [36] Hyndman, R.J., Athanasopoulos, G.: *Forecasting: Principles and Practice*. OTexts, Melbourne (2018)
- [37] Hyndman, R.: Fpp2: Data for "Forecasting: Principles and Practice" (2nd Edition). (2020). R package version 2.4. <https://CRAN.R-project.org/package=fpp2>
- [38] Takahashi, K.: Formation of sparse bus impedance matrix and its application to short circuit study. In: *Proc. PICA Conference*, June, 1973 (1973)
- [39] Chen, Y., Davis, T.A., Hager, W.W., Rajamanickam, S.: Algorithm 887:

Cholmod, supernodal sparse cholesky factorization and update/down-date. *ACM Transactions on Mathematical Software (TOMS)* **35**(3), 1–14 (2008)

- [40] GPy: GPy: A Gaussian process framework in python. <http://github.com/SheffieldML/GPy> (since 2012)
- [41] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
- [42] Ajjanagadde, G., Makur, A., Klusowski, J., Xu, S., et al.: *Lecture notes on information theory* (2017)
- [43] Murphy, K.P.: *Machine Learning: a Probabilistic Perspective*. MIT press, Cambridge (2012)
- [44] Katzfuss, M., Guinness, J., Gong, W., Zilber, D.: Vecchia approximations of gaussian-process predictions. *Journal of Agricultural, Biological and Environmental Statistics* **25**(3), 383–414 (2020)

Appendix A Extensions and Details

A.1 Generalized CPoE

Alternatively to the graphical model defined in Def. 4 (and more precisely in Prop. 1 with Proof 1) which recovers sparse global GP model FITC [7] in the limiting case $C \rightarrow J$ (as shown in Prop. 5), we present in this section a generalization of our CPoE model such that it recovers other sparse global GP models such as VFE [5] or PEP [6]. As shown by the authors in [11] for the global case, these model differ in the training only by the choice of the projection matrix $\bar{\mathbf{V}}_j$ in Def. 4 and in the hyperparameter optimization by a modification of the log marginal likelihood $\mathcal{L}(\boldsymbol{\theta}) = \log q(\mathbf{y}|\boldsymbol{\theta})$ in Section 3.6.2. These two changes can also be made for our local sparse CPoE model. In particular, using $\bar{\mathbf{V}}_j$ and λ_j according to the values in Table A2 in the projection conditional

$$p(\mathbf{f}_j|\mathbf{a}_{\psi(j)}) = \mathcal{N}(\mathbf{f}_j|\mathbf{H}_j\mathbf{a}_{\psi(j)}, \bar{\mathbf{V}}_j)$$

and in a lower bound to the log marginal likelihood

$$\tilde{\mathcal{L}}(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}) - \sum_{j=1}^J \lambda_j(\boldsymbol{\theta})$$

and $\tilde{l}_j(\boldsymbol{\theta}) = l_j(\boldsymbol{\theta}) - \lambda_j(\boldsymbol{\theta})$ in the deterministic and stochastic case, respectively, generalizes the CPoE method such that for $C \rightarrow J$ we recover the mentioned method global methods in Table A2. Thereby, we used

$$\mathbf{D}_j = \mathbf{K}_{\mathbf{X}_j \mathbf{X}_j} - \mathbf{K}_{\mathbf{X}_j \mathbf{A}_{\psi(j)}} \mathbf{K}_{\mathbf{A}_{\psi(j)} \mathbf{A}_{\psi(j)}}^{-1} \mathbf{K}_{\mathbf{A}_{\psi(j)} \mathbf{X}_j}$$

which is the difference of the true and local approximated covariance.

The setting in VFE [5] is particularly interesting, since it constitutes in the global case a direct posterior approximation derived via a variational maximization of the lower bound of the log marginal likelihood. Moving a bit away from the true marginal likelihood of full GP has the effect that overfitting (w.r.t. full GP) can not happen when optimizing the hyperparameters with the lower bound. This is particularly important when all inducing inputs are optimized as it is usually recommended in sparse global methods which is not the case for our model since it allows to have a number of inducing points in the order of the number of data samples. In the adapted 'local VFE' CPoE model when using $\bar{\mathbf{V}}_j = 0$ and minimize also $\lambda_j = \text{tr}\{\mathbf{D}_j\}$ has the effect that the model is locally variationally optimal, however, it would be interesting to directly derive a lower bound analogously to [5] so that the posterior of our CPoE model is rigorously connected to full GP. Since this is not a straight-forward extension, we postpone this task to future work. Below, we present the connection to full GP for this adapted model in the joint prior sense analogously to Prop. 6 for the local FITC model.

| variable | domain | explanation |
|-------------------------------------|----------------------------------|--|
| N | \mathbb{N}^+ | number of data samples |
| D | \mathbb{N}^+ | number of variables/dimension of data |
| J | $\{1, \dots, N\}$ | number of experts/partitions |
| B | $\{1, \dots, N\}$ | size of expert/partition |
| L | $\{1, \dots, B\}$ | number of local inducing points |
| M | $\{1, \dots, N\}$ | number of total local inducing points |
| C | $\{1, \dots, J\}$ | degree of correlation |
| γ | $(0, 1]$ | sparsity parameter |
| α | $(0, J]$ | approximation quality parameter |
| y_i | \mathbb{R} | individual data output |
| \mathbf{y}_j | \mathbb{R}^B | data output of expert j |
| $\mathbf{y}_{k:j}$ | $\mathbb{R}^{B(j-k+1)}$ | data output of experts k up to j |
| \mathbf{y}_* | \mathbb{R} | pointwise noisy prediction output |
| \mathbf{y} | \mathbb{R}^N | all data output |
| \mathbf{x}_i | \mathbb{R}^D | individual data input |
| \mathbf{X}_j | $\mathbb{R}^{B \times D}$ | data input of expert j |
| $\mathbf{X}_{k:j}$ | $\mathbb{R}^{B(j-k+1) \times D}$ | data input of experts k up to j |
| \mathbf{X} | $\mathbb{R}^{N \times D}$ | all data input |
| \mathbf{x}_* | \mathbb{R}^D | query input for prediction |
| \mathbf{f}_j | \mathbb{R}^B | latent function outputs of expert j |
| $\mathbf{f}_{k:j}$ | $\mathbb{R}^{B(j-k+1)}$ | latent function outputs of experts k up to j |
| \mathbf{f} | \mathbb{R}^N | all latent function outputs |
| \mathbf{f}_* | \mathbb{R} | pointwise (latent) prediction output |
| $f(\mathbf{X}_j)$ | \mathbb{R}^B | GP evaluation for input matrix |
| \mathbf{a}_j | \mathbb{R}^L | local inducing outputs of expert j |
| $\mathbf{a}_{k:j}$ | $\mathbb{R}^{L(j-k+1)}$ | local inducing outputs of experts k up to j |
| \mathbf{a} | \mathbb{R}^M | all local inducing outputs |
| \mathbf{A}_j | $\mathbb{R}^{L \times D}$ | local inducing inputs of expert j |
| $\mathbf{A}_{k:j}$ | $\mathbb{R}^{L(j-k+1) \times D}$ | local inducing inputs of experts k up to j |
| \mathbf{A} | $\mathbb{R}^{M \times D}$ | all local inducing inputs |
| I_j | $\{1, \dots, C-1\}$ | number of predecessors of expert j |
| $\phi_i(j)$ | $\{1, \dots, j-1\}$ | i th predecessor of expert j |
| $\pi(j)$ | $\{1, \dots, j-1\}^{I_j}$ | predecessor index set |
| $\pi^+(j)$ | $\{1, \dots, j\}^{I_j+1}$ | predecessor index set including j |
| $\psi(j)$ | $\{1, \dots, \max(j, C)\}^C$ | correlation index set |
| σ_n^2 | \mathbb{R}^+ | observation noise variance |
| θ | $\mathbb{R}^{ \theta }$ | kernel hyperparameters including σ_n^2 |
| $k_\theta(\mathbf{x}, \mathbf{x}')$ | \mathbb{R} | kernel evaluation for 2 query points |
| \mathbf{K}_{AB} | $\mathbb{R}^{M_a \times M_b}$ | kernel matrix of two query matrices |
| $p(\mathbf{z})$ | \mathbb{R} | evaluation of (true) probability density |
| $q(\mathbf{z})$ | \mathbb{R} | evaluation of approximated probability density |
| \mathbf{S} | $\mathbb{R}^{M \times M}$ | prior precision matrix |
| \mathbf{T} | $\mathbb{R}^{M \times M}$ | projection precision matrix |
| Σ^{-1} | $\mathbb{R}^{M \times M}$ | posterior precision matrix |
| Σ | $\mathbb{R}^{M \times M}$ | posterior covariance matrix |
| μ | \mathbb{R}^M | posterior mean vector |
| $\mu_{\psi(j)}$ | \mathbb{R}^{CL} | local posterior mean |
| $\Sigma_{\psi(j)}$ | $\mathbb{R}^{CL \times CL}$ | local posterior covariance |
| \mathbf{F} | $\mathbb{R}^{M \times M}$ | prior transition matrix |
| \mathbf{Q} | $\mathbb{R}^{M \times M}$ | prior noise matrix |
| \mathbf{H} | $\mathbb{R}^{N \times M}$ | projection matrix |
| \mathbf{V} | $\mathbb{R}^{N \times N}$ | projection noise matrix |
| \mathbf{P} | $\mathbb{R}^{N \times N}$ | projection noise matrix including observation noise |
| J_2 | \mathbb{N}^+ | marginal likelihood covariance matrix |
| $\bar{\beta}_{*j}$ | \mathbb{R}^+ | number of prediction experts |
| β_{*j} | \mathbb{R}^+ | unnormalized predictive weight of expert j at \mathbf{x}_* |
| m_{*j} | \mathbb{R} | normalized predictive weight of expert j at \mathbf{x}_* |
| v_{*j} | \mathbb{R}^+ | predictive mean of expert j at \mathbf{x}_* |
| $\mathbb{D}_{[C, C_2]}$ | \mathbb{R}^+ | predictive variance of expert j at \mathbf{x}_* |
| | | difference in KL between two approximate models |

Table 1: Overview of notation.

| | $\bar{\mathbf{V}}_j$ | λ_j |
|------------------|------------------------------------|--|
| DTC | 0 | 0 |
| FITC | $\text{Diag}[\mathbf{D}_j]$ | 0 |
| PITC | \mathbf{D}_j | 0 |
| VFE | 0 | $\frac{1}{2\sigma_n^2} \text{tr}[\mathbf{D}_j]$ |
| PEP | $\alpha \text{Diag}[\mathbf{D}_j]$ | $\frac{1-\alpha}{2\alpha} \sum_i \log \left(1 + \frac{\alpha}{\sigma_n^2} \mathbf{D}_j^{(i)} \right)$ |
| PEP _B | $\alpha \mathbf{D}_j$ | $\frac{1-\alpha}{2\alpha} \sum_i \log \ \mathbb{I} + \frac{\alpha}{\sigma_n^2} \mathbf{D}_j\ $ |

Table A2: Generalizations of CPoE model.

Proposition 7 (Local VFE) *Using a deterministic projection $q(\mathbf{f}_j | \mathbf{a}_{\psi(j)}) = \mathcal{N}(\mathbf{f}_j | \mathbf{H}_j \mathbf{a}_{\psi(j)}, \bar{\mathbf{V}}_j)$ in the graphical model in Def. 4 and Prop. 1, that is, setting the covariance $\bar{\mathbf{V}}_j = 0$ in the projection step recovers global VFE for $C \rightarrow J$. Moreover, the difference in KL to full GP of the joint prior is also decreasing. In particular, the difference in KL of the prior of the local VFE model for $1 \leq C \leq C_2 \leq J$ is*

$$\mathbb{D}_{(C, C_2)}[\mathbf{a}] = \frac{1}{2} \log \frac{|\mathbf{Q}_C|}{|\mathbf{Q}_{C_2}|} \geq 0.$$

Further, the difference in KL of the projection is

$$\mathbb{D}_{(C, C_2)}[\mathbf{y} | \mathbf{a}] = \frac{1}{2\sigma_n^2} \text{tr}\{\bar{\mathbf{V}}_C - \bar{\mathbf{V}}_{C_2}\} \geq 0.$$

The overall prior approximation quality is

$$\mathbb{D}_{(C, C_2)}[\mathbf{a}, \mathbf{y}] = \frac{1}{2} \log \frac{|\mathbf{Q}_C|}{|\mathbf{Q}_{C_2}|} + \frac{1}{2\sigma_n^2} \text{tr}\{\bar{\mathbf{V}}_C - \bar{\mathbf{V}}_{C_2}\}$$

where

$$\text{tr}\{\bar{\mathbf{V}}_C\} = \sum_{i=1}^N K_{\mathbf{X}_i \mathbf{X}_i} - K_{\mathbf{X}_i \mathbf{A}_{\psi(j_i)}} K_{\mathbf{A}_{\psi(j_i)} \mathbf{A}_{\psi(j_i)}}^{-1} K_{\mathbf{A}_{\psi(j_i)} \mathbf{X}_i}.$$

Compared to the FITC model is the difference in the trace instead of the fraction of the log-determinants.

A.2 Solving Linear System & Partial Inversion

For solving the sparse linear system $\Sigma^{-1} \boldsymbol{\mu} = \boldsymbol{\eta}$ in Prop. 3, sparse Cholesky decomposition is exploited, that is, $\mathbf{M} \Sigma^{-1} \mathbf{M}^T = \mathbf{L} \mathbf{L}^T =: \mathbf{Y}$ is computed so that $\boldsymbol{\nu}$ and $\bar{\boldsymbol{\mu}}$ can be efficiently obtained via solving $\mathbf{L} \boldsymbol{\nu} = \boldsymbol{\eta}$ and $\mathbf{L}^T \bar{\boldsymbol{\mu}} = \boldsymbol{\nu}$, respectively, where \mathbf{M} is a so-called fill-reduction permutation matrix such that the Cholesky matrix \mathbf{L} is as sparse as possible and thus $\boldsymbol{\mu} = \mathbf{M}^{-1} \bar{\boldsymbol{\mu}}$. Note that \mathbf{M} is computed only via the structure on the block level which is only J dimensional instead of JL . Additionally to the mean $\boldsymbol{\mu}$, also some entries $\Sigma_{\psi(j)}$ in the covariance matrix Σ has to be explicitly computed, which are needed for computing local predictions used in Section 3.4.

Proposition 8 (Local Predictions, Proof 18) *The local prediction $q_{c,\gamma}(f_{*j}|\mathbf{y}) = \mathcal{N}(m_{*j}, v_{*j})$ of the j th expert are based on the region $\psi(j)$, where the correlations are modelled and can be computed as*

$$q_{c,\gamma}(f_{*j}|\mathbf{y}) = \int p(f_{*j}|\mathbf{a}_{\psi(j)}) q_{c,\gamma}(\mathbf{a}_{\psi(j)}|\mathbf{y}) d\mathbf{a}_{\psi(j)} = \mathcal{N}(\mathbf{h}_* \boldsymbol{\mu}_{\psi(j)}, \mathbf{h}_*^T \boldsymbol{\Sigma}_{\psi(j)} \mathbf{h}_* + v_*)$$

*involving the local posteriors $q_{c,\gamma}(\mathbf{a}_{\psi(j)}|\mathbf{y}) = \mathcal{N}(\boldsymbol{\mu}_{\psi(j)}, \boldsymbol{\Sigma}_{\psi(j)})$ and the predictive conditional $p(f_{*j}|\mathbf{a}_{\psi(j)}) = \mathcal{N}(\mathbf{h}_* \mathbf{a}_{\psi(j)}, v_*)$ (which is exactly defined in Proof 18).*

For computing the local posteriors $q_{c,\gamma}(\mathbf{a}_{\psi(j)}|\mathbf{y}) = \mathcal{N}(\boldsymbol{\mu}_{\psi(j)}, \boldsymbol{\Sigma}_{\psi(j)})$, we need to compute the entries $\boldsymbol{\Sigma}_{\psi(j)}$ in the covariance matrix $\boldsymbol{\Sigma}$, which correspond to the non-zeros in the precision matrix $\boldsymbol{\Sigma}^{-1}$. Computing efficiently these entries is not straightforward since in the inverse the blocks are no longer independent. However, we can exploit the particular sparsity and block-structure of our precision matrix and obtain an efficient implementation of this part which is key to achieve a competitive performance of our algorithm.

Computing some entries in $\mathbf{Z} = \mathbf{Y}^{-1}$ is also known as *partial inversion*. We adapted the approach in [38] where the recursive equations with J blocks for computing the full inverse \mathbf{Z} are provided

$$\mathbf{Z}_{B_j} = -\mathbf{Z}_{C_j} \mathbf{L}_{B_j} \mathbf{L}_{A_j}^{-1} \quad \text{and} \quad \mathbf{Z}_{A_j} = \mathbf{L}_{A_j}^{-T} \mathbf{L}_{A_j}^{-1} - \mathbf{Z}_{B_j}^T \mathbf{L}_{B_j} \mathbf{L}_{A_j}^{-1}$$

where the recursion starts from $j = J$ with $\mathbf{Z}_{A_j} = \mathbf{L}_{A_j}^{-T} \mathbf{L}_{A_j}^{-1}$.

The diagram shows two matrices, Z and L , with their block structures. Z is a square matrix with a light blue background. It is partitioned into four blocks: Z_{A_j} (top-left), $Z_{B_j}^T$ (top-right), Z_{B_j} (bottom-left), and Z_{C_j} (bottom-right). L is a square matrix with a light blue background, representing a lower triangular matrix. It has blocks L_{A_j} and L_{B_j} on the diagonal, and a shaded region below the diagonal indicating the lower triangular structure.

Instead of computing the full inverse using this recursion, we exploited the block-sparsity structure of our posterior precision matrix in order to gain significant speed-up. We only computed the entries in the inverse \mathbf{Z} which are symbolically non-zero in \mathbf{L} . In Algorithm 1 in the Appendix we provide efficient pseudo-code using sparse-block-matrices in the block-sparse-row format.

Alternatively for computing the Cholesky factor of $\boldsymbol{\Sigma}^{-1} = \mathbf{S} + \mathbf{H}\mathbf{V}^{-1}\mathbf{H}$, we could directly exploit that the prior precision $\mathbf{S} = \mathbf{F}^T \mathbf{Q}^{-1} \mathbf{F}$ is already decomposed into an upper/lower-triangular form since \mathbf{F} is lower triangular. However, when updating the Cholesky factor with $\mathbf{H}\mathbf{V}^{-1}\mathbf{H}$ needs quadratic time in the number of nonzeros for each expert.

Algorithm 1 Partial Sparse Block Inversion

Require: Cholesky matrix \mathbf{L} of size $JB \times JB$ in block-sparse-row (bsr) format with $J \times J$ total blocks, block-size B and N non-zero blocks. Data array d of size $N \times B \times B$, the column-block-indices r of size N , row-block-pointer p of length $J + 1$, and lookup table M of dimension $J \times J$.

Ensure: The lower part of the symmetric partial inversion is computed in bsr-format with the same row-block-indices r and row-block-pointer p and data array z of size $N \times B \times B$.

```

for  $i \in \{J, \dots, 1\}$  do
   $L_A^{-1} \leftarrow d[M[i, i], :, :]^{-1}$ 
   $z[M[i, i]] \leftarrow (L_A^{-1})^T \cdot L_A^{-1}$ 
  for  $j \in \{r[p[i + 1]], r[p[i + 1] - 1], \dots, r[p[i]]\}$  do
     $Q \leftarrow 0$ 
    for  $l \in \{r[p[i]], r[p[i] + 1], \dots, r[p[i + 1]]\}$  do
       $R \leftarrow z[M[j, l]]$ 
      if  $l > j$  then
         $R \leftarrow R^T$ 
      end if
       $Q \leftarrow Q + R \cdot d[M[l, i]] \cdot L_A^{-1}$ 
    end for
     $z[M[i, j]] \leftarrow z[M[i, j]] - Q$ 
  end for
end for

```

A.3 Hyperparameter Estimation

In Section 3, we introduced CPoE for fixed hyperparameters θ where implicitly all distributions are conditioned on θ , however, we omitted the dependencies on θ in the most cases for the sake of brevity. Similar to full GP, sparse GP or PoEs, the *log marginal likelihood (LML)* can be used as an objective function for optimizing the few hyperparameters θ .

A.3.1 Deterministic Optimization

The log of the marginal likelihood of our model formulated in Section 3.3 can be written as

$$\mathcal{L}(\theta) = \log q(\mathbf{y}|\theta) = \log \mathcal{N}(\mathbf{0}, \mathbf{P}) = -\frac{1}{2} (\mathbf{y}^T \mathbf{P}^{-1} \mathbf{y} + \log |\mathbf{P}| + N \log 2\pi)$$

with $\mathbf{P} = \mathbf{H}\mathbf{S}^{-1}\mathbf{H}^T + \mathbf{V}$. Since \mathbf{P} is dense, we can apply the inversion (B2) and determinant lemma (B3) to \mathbf{P} and exploit $|\mathbf{F}| = 1$ yielding

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}) &= -\frac{1}{2} \left(\mathbf{y}^T \mathbf{V}^{-1} \mathbf{y} - \boldsymbol{\mu}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} + \log \frac{|\boldsymbol{\Sigma}^{-1}| |\mathbf{V}|}{|\mathbf{S}|} + N \log 2\pi \right) \\ &= -\frac{1}{2} \left(\mathbf{y}^T \mathbf{V}^{-1} \mathbf{y} - \boldsymbol{\mu}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} + \log |\boldsymbol{\Sigma}^{-1}| |\mathbf{V}| |\mathbf{Q}| + N \log 2\pi \right) \end{aligned} \quad (\text{A1})$$

so that all involved quantities $\boldsymbol{\Sigma}^{-1}$, \mathbf{V} and \mathbf{Q} are sparse. For efficient parameter minimization, the derivative of the log marginal likelihood with respect to each parameter in $\boldsymbol{\theta}$ is needed for which the derivations are provided in Appendix C.3. Thereby also some parts of the covariance matrix $\boldsymbol{\Sigma}$ are needed which is explained in Section A.2. Alternatively to the marginal likelihood, we can maximize a lower bound of it which is a generalization of our model so that we recover a range of well known sparse global GP models for $C \rightarrow J$ as discussed in Section A.1. [6, 11].

For moderate sample size N , *deterministic optimization* with full batch \mathbf{y} can be performed. That means, the log marginal likelihood for the whole data is computed for which the sparse system of equations with the sparse posterior precision as well as the partial inversion of the posterior covariance has to be solved. In particular, the functions for computing $\mathcal{L}(\boldsymbol{\theta})$ and $\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$ for each $\boldsymbol{\theta}$ and full data \mathbf{y} are repetitively called by a numerical minimizer. Fig. A1 illustrates the performance of this deterministic batch hyperparameter optimization where the convergence for the log marginal likelihood, average KL divergence, 95%-coverage (both quantities exactly defined in Appendix F) for different number of experts J compared to full GP are depicted. The $N = 2048$ data samples are generated with a $D = 2$ -dimensional SE-kernel and the test KL and coverage mean values are reported for $N_{test} = 1000$ samples with 5 repetitions. We used $\gamma = 1$ and $C \in \{1, \dots, 7\}$. We observe that the log marginal likelihood and KL are getting better for increasing C , and the deterministic parameter estimates converge to the ones of full GP for increasing function calls. It is interesting to observe that also for smaller C values, the coverage of our methods are consistent. In particular, they are slightly too big, meaning our confidence information are conservative. This is due to the aggregation based on the covariance intersection method with normalized weights, which guarantees consistent second order information.

A.3.2 Stochastic Optimization

The presented method in the previous section works fine for small datasets, however, in order to scale this parameter optimization part to larger number of samples N in a competitive time, *stochastic optimization* techniques has to be exploited similarly done for the global sparse GP model (SVI [9]; REC [11]; IF [10]). In the approximation method REC [11], the recursive derivatives are exactly propagated which would also be possible for our model, however, it

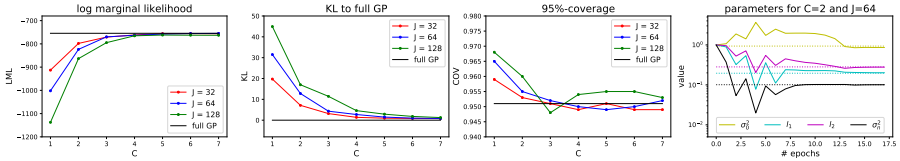


Fig. A1: Convergence of deterministic batch hyperparameter optimization for increasing C and trace of the parameters (solid) compared to the optimal values of full GP (dotted).

turned out that in practice the differences in accuracy are very small when using instead the hybrid approach IF of [10]. Thereby, the independent factorization of the log marginal likelihood is used for the computations of the optimization part, whereas the exact posterior is used for inference and prediction. Adapted to our setting, the independent factorized log marginal likelihood $\log q(\mathbf{y}|\boldsymbol{\theta})$ can be approximated by

$$\begin{aligned}
 \log q(\mathbf{y}|\boldsymbol{\theta}) &\approx \log \prod_{j=1}^J \int q(\mathbf{y}_j|\mathbf{a}_j) q(\mathbf{a}_j) d\mathbf{a}_j \\
 &= \log \prod_{j=1}^J \int \mathcal{N}(\mathbf{H}_j \mathbf{a}_j, \mathbf{V}_j) \mathcal{N}(\mathbf{0}, \mathbf{S}_j^{-1}) d\mathbf{a}_j \\
 &= \sum_{j=1}^J \log \mathcal{N}(\mathbf{0}, \mathbf{P}_j) =: \tilde{\mathcal{L}}(\boldsymbol{\theta})
 \end{aligned}$$

where $\mathbf{P}_j = \mathbf{H}_j \mathbf{S}_j^{-1} \mathbf{H}_j^T + \mathbf{V}_j$ with $\mathbf{S}_j = \mathbf{K}_{\mathbf{A}_j \mathbf{A}_j}^{-1}$. The difference compared to the deterministic case in (A1) and to [10] for the global sparse model is the independent prior $q(\mathbf{a}_j)$ instead of $q(\mathbf{a})$ and $p(\mathbf{a})$, respectively. In the approximate case, we can write

$$\tilde{\mathcal{L}}(\boldsymbol{\theta}) = -\frac{1}{2} N \log(2\pi) + \sum_{j=1}^J l_j(\boldsymbol{\theta})$$

with $l_j(\boldsymbol{\theta}) = -\frac{1}{2} (\mathbf{y}_j^T \mathbf{P}_j^{-1} \mathbf{y}_j + \log |\mathbf{P}_j|)$ which has the advantage that it decomposes into the J terms l_j in the sum, so that it can be used for stochastic optimization. This constitutes a very fast and accurate alternative for our method as shown in Figure A2 and is exploited in Section 4 for large data sets.

A.3.3 Prior on Hyperparameters

Alternatively to the log marginal likelihood (LML) maximization as presented above, the *maximum a posteriori* (MAP) estimator for $\boldsymbol{\theta}$ can be used. It is the log of the posterior distribution $p(\boldsymbol{\theta}|\mathbf{y}) \propto p(\mathbf{y}|\boldsymbol{\theta}) p(\boldsymbol{\theta})$ where $p(\boldsymbol{\theta})$ is a suitable prior on the hyperparameters yielding $\log p(\boldsymbol{\theta}|\mathbf{y}) = \log p(\mathbf{y}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta})$.

In the following, we assume $p(\boldsymbol{\theta}) = \prod_j p(\boldsymbol{\theta}_j)$ and a log-normal prior for each hyperparameter $p(\boldsymbol{\theta}_j) = \log \mathcal{N}(\boldsymbol{\theta}_j | \nu_j, \lambda_j^2)$ for means ν_j and variances λ_j^2 . For the deterministic case, the MAP estimator can be straightforwardly computed by just adding the log prior on $\boldsymbol{\theta}$ to the batch log marginal likelihood, i.e. $\log p(\boldsymbol{\theta} | \mathbf{y}) = \mathcal{L}(\boldsymbol{\theta}) + \log p(\boldsymbol{\theta})$. Similarly for the stochastic case, the stochastic MAP can be decomposed as $\log p(\boldsymbol{\theta} | \mathbf{y}) \approx \sum_{j=1}^J (l_j(\boldsymbol{\theta}) + \frac{1}{J} \log p(\boldsymbol{\theta}))$, where $l_j(\boldsymbol{\theta})$ is the j th term in the stochastic marginal likelihood, so that it can be used again for stochastic mini-batch optimization. An example using priors for the hyperparameters is presented in Section 4.4.

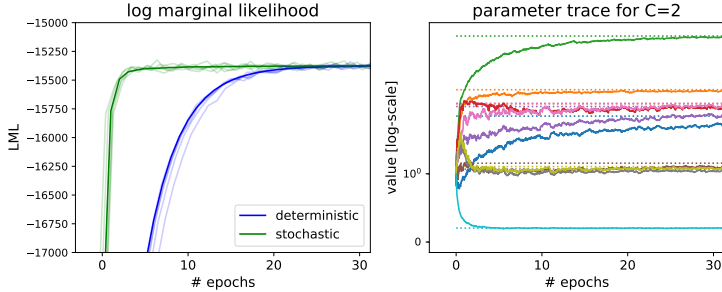


Fig. A2: Convergence of stochastic vs. deterministic hyperparameter optimization of our model CPoE. This experiment compares the convergence of stochastic vs. deterministic hyperparameter optimization for the log marginal likelihood and the trace of the 10 parameters $\boldsymbol{\theta}_j$ for the dataset *cadata* with $N_{tot} = 20640$ and $D = 8$. We used 5 different splits with $N = 0.9N_{tot}$ training data and the rest for testing. The values for our algorithm are $C = 2$, $J = 64$, $\gamma = 1$ and learning rate $\delta = 0.01$. In the right plot, the dotted horizontal lines and the solid traces correspond to the final deterministic value and the current stochastic values, respectively. We note that the stochastic LML and trace of hyperparameters converge faster to a very similar value as in the deterministic case.

A.4 Complexity

The time complexity for computing the posterior and the marginal likelihood in our algorithm is dominated by J operations which are cubic in LC (inversion, matrix-matrix multiplication, determinants). This leads to $\mathcal{O}(J(LC)^3) = \mathcal{O}(J(BC\gamma)^3) = \mathcal{O}(NB^2\alpha^3)$ where we define the approximation quality parameter $\alpha = C\gamma$. Similarly for the needed space $\mathcal{O}(J(LC)^2) = \mathcal{O}(J(BC\gamma)^2) = \mathcal{O}(NB\alpha^2)$. For N_t testing points, the time for (pointwise) predictions is dominated by J inversions of matrices with dimension LC and matrix multiplications with dimensions $LC \times LC \times N_t$ leading to $\mathcal{O}(J(LC)^3 + J(LC)^2 N_t) = \mathcal{O}(J(B\gamma C)^3 + J(B\gamma C)^2 N_t) = \mathcal{O}(NB^2\alpha^3 + NB\alpha^2 N_t)$ where the operations independent of the test points can be precomputed in the inference part leading to $\mathcal{O}(NB\alpha^2 N_t)$ for testing. Similarly for the space. A further reduction in complexity would be achieved if the product over all experts

in Prop. 4 is approximated only with the $W < J$ nearest experts, leading to $\mathcal{O}((LC)^2 N_t W) = \mathcal{O}((B\gamma C)^2 N_t W) = \mathcal{O}(N \frac{W}{J} B \alpha^2 N_t)$ time complexity for testing. This might be interesting if we want to make fast predictions for many points N_t . For reasonable values of W , for instance $W = 1$, $W = C$ or $W = Z = \log(N)C$ (used in prediction aggregation), preliminary experiments show very comparable performance. Note that the consistency properties for covariance intersection method are preserved as long as the weights are normalized over the used W experts. Table 1 compares the asymptotic complexities with other GP algorithms.

It is interesting that for $\alpha = 1$, our algorithm has the same asymptotic complexity for training as sparse global GP with $M_g = B$ global inducing points but we can have $M_l = LJ = \gamma BJ = \gamma N$ total local inducing points! Thus, our approach allows much more total local inducing points M in the order of N (e.g. $M = 0.5N$ with $C = 2$) whereas for sparse global GP usually $M_g \ll N$. This has the consequence that the local inducing points can cover the input space much better and therefore represent much more complicated functions. As a consequence, there is also no need to optimize the local inducing points resulting in much fewer parameters to optimize. Consider the following example with $N = 10'000$ in $D = 10$ dimensions. Suppose a sparse global GP model with $M_g = 500$ global inducing points. A CPoE model with the same asymptotic complexity has a batch size $B = M_g = 500$ and $\alpha = 1$. Therefore, we have $J = \frac{N}{B} = 20$ experts and we choose $C = 2$ and $\gamma = \frac{1}{2}$ such that we obtain $L = \gamma B = 250$ local inducing points per experts and $M = \gamma N = 5'000$ total inducing points! Further, the number of hyperparameters to optimize for a SE kernel is for global sparse GP $M_g D + |\theta| = 5012$, whereas for CPoE there are only $|\theta| = 12$.

For our method, the time and space complexity is linear in the number of samples N and the number of experts J which makes our approach highly scalable. The approximation quality parameter $\alpha = C\gamma$ appears cubic/quadratic in the time/space complexity. The optimal approximation quality (and thus equivalent to full GP) is achieved for $\alpha = J$ which implies $C = J$ and $\gamma = 1$. However, it is clear that this is not feasible for big datasets and thus some moderate values of C and γ have to be selected to trade off time and accuracy which is illustrated in the Appendix in Table A3 and Fig. A4.

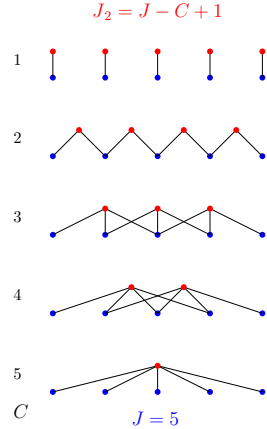


Fig. A3: Prediction Aggregation in CPoE(C, γ) model with J base experts and $J_2 = J - C + 1$ predictive experts.

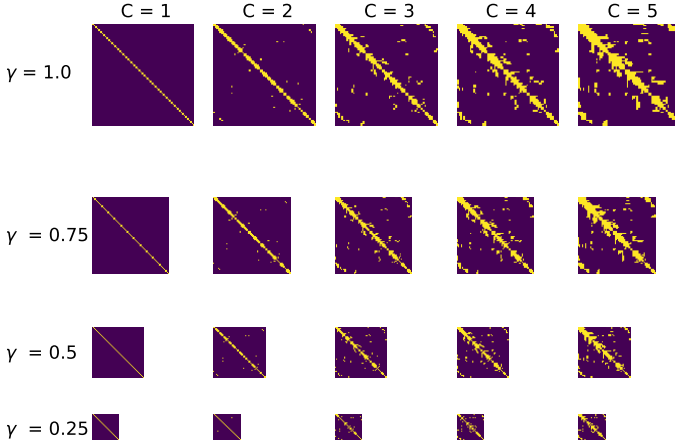


Fig. A4: Influence of the approximation order C and sparseness parameter γ to the non-zeros and size of the posterior precision matrix Σ^{-1} , for an example with synthetic GP data with $D = 2$, $N = 8192$, $J = 64$ and $B = 128$ and a SE-kernel. Compare also Table A3.

| KL | C=1 | C=2 | C=3 | C=4 | C=5 |
|----------------|------|-----|-----|------|------|
| $\gamma = 1/4$ | 12.3 | 5.0 | 1.3 | 0.9 | 0.7 |
| $\gamma = 1/2$ | 12.2 | 4.9 | 1.0 | 0.8 | 0.6 |
| $\gamma = 3/4$ | 12.1 | 4.9 | 0.9 | 0.7 | 0.5 |
| $\gamma = 1$ | 12.1 | 4.8 | 0.9 | 0.6 | 0.4 |
| time | C=1 | C=2 | C=3 | C=4 | C=5 |
| $\gamma = 1/4$ | 0.2 | 0.4 | 0.9 | 1.2 | 1.4 |
| $\gamma = 1/2$ | 0.4 | 0.7 | 1.9 | 2.7 | 3.8 |
| $\gamma = 3/4$ | 0.9 | 2.4 | 4.1 | 5.7 | 9.1 |
| $\gamma = 1$ | 1.5 | 3.0 | 6.4 | 12.4 | 15.7 |

Table A3: KLs to full GP (above) and times (below) of our method CPoE for varying C and γ for experiment in Section 3.6.3. Compare also Fig. A4 .

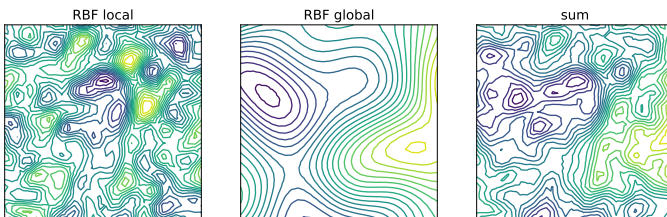


Fig. A5: Generated data with a sum of two SE-kernels with local and global lengthscales for experiment in Section 4.

A.5 Relation to Deep Structured Mixtures of GPs

There are several approaches exploiting local experts, as discussed in Section 2.2. In particular, the recent approach *Deep Structured Mixtures of Gaussian Processes* (DSMGP) [29] shares some similarities with our approach. However, our approach has a main advantages against DSMGP, namely, the continuous predictive distribution, as illustrated in Figure A6. In the following, we briefly summaries their approach, discuss the similarities and differences to our approach, and compare them in two illustrative experiments.

The GP approximation approach DSMGP exploits a sum-product network of local GPs, which corresponds to a mixture of independent base experts as in the ordinary PoEs. This approach allows analytic posterior inference, however, in order to compute efficient predictions, the posterior of the DSMGP is projected to the closest GP, i.e. the GP with minimal KL divergence from the DSMGP. This has the effect that the aggregated predictive distribution might discontinuous, as illustrated in Figure A6. This is a common disadvantage of local aggregation methods based on predictions form a single region. DSMGP uses independent base experts similarly as the local PoE methods discussed in Section 2.2. However, the predictions are averaged by a mixture over different partitions of the input space. On the other hand, our approach directly models correlations between the base experts, which recovers the exact dependencies between the different partitions, leading to smooth predictions.

We compared both methods on two illustrative examples. In particular, we used the data from the main example on the GitHub repository³ of their approach, which is illustrated on the left in Figure A6. The $N = 100$ data samples are generated from a sine and the number of splits per product nodes is $K = 4$, the number of children per sum node is $V = 3$, and the minimum number of experts is $M = 20$ corresponding to $J = 5$ base experts. We optimized the hyperparameters with full GP and fixed them for all approximation methods. The results for this kind of data are consistent with the findings in [29], where DSMGP can improve over the other local independent methods (min-Var, GPoE, BCM). However, the global sparse method and in particular our method CPoE with $C = 2$ significantly outperforms DSMGP, as demonstrated with the KL to full GP. Similarly, we run their algorithm on the example of Figure 2, where the generated data has a smaller lengthscale and bigger observation noise. The parameters for DSMGP were set to $K = 4$, $V = 3$, $M = 10$, which correspond to $J = 6$ independent experts. This more complex setting illustrates the main limitation of DSMGP with the discontinuous predictions, as depicted in Figure A6 on the right. Moreover, this is also reflected in the accuracy in term of KL to full GP. We run some preliminary comparisons of the experiment depicted in Figure 9, with similar performance as obtained in the both examples shown in Figure A6. This comparison can be found on our github repository.⁴ However, it is hard to compare the running times of both algorithms based on *Python* and *Julia*, respectively. Therefore, it would require

³<https://github.com/trappmartin/DeepStructuredMixtures>

⁴https://github.com/manschuer/CPoE/experiments/comparisonDSM_py.ipynb

to reimplement their algorithm in Python for proper empirical comparisons in a time vs. accuracy sense.

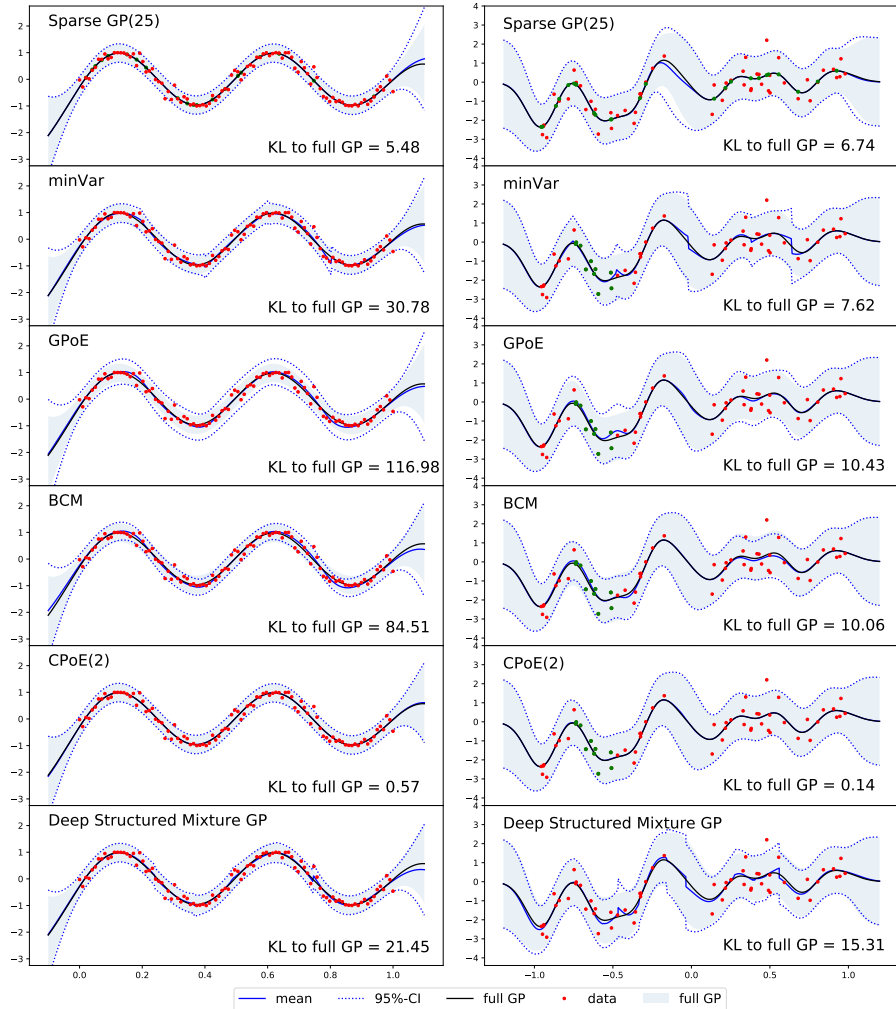


Fig. A6: The data and the example on the left corresponds exactly to the example provided on the GitHub repository of the method DSMGPs, whereas the example on the right corresponds to the Example in Figure 2.

A.6 Implementation Details

All experiments were run on a standard Laptop (IntelCore i7, 8 CPU 1.9GHz). Our code is implemented in Python and is available on GitHub⁵. It contains

⁵<https://github.com/manschuer/CPoE>

several examples^{6 7} how to use it together with experiments from this paper. For solving the sparse linear system of equations, we used Cholmod [39] in the Python package scikit-sparse which relies on sparse Cholesky decomposition. It would be advantageous to use/implement a sparse block Cholesky decomposition and solver which exploits directly our structure. This was indeed needed for computing some entries in the posterior covariance, since with available implementation of *partial matrix inversion* we could not exploit the block sparsity and thus did not obtain competitive performance as discussed in Section A.2. An efficient implementation of this part is presented in Algorithm 1.

In our current implementation the size of each partition has to be equal; which is in theory not necessary, but it allows more efficient implementations since then the block character can be easily exploited in the computation of the sparse posterior precision. Using the KD-tree construction with $J = 2^K$, the sizes of the partitions differ at most by 1. Thus, if the partitions are not equal, the number of local inducing points are set to $L = \min(B_j)_{j=1}^J$. Our implementation exploits the kernel and likelihood functions of GPY [40]. For the optimization of the hyperparameters we used the L-BFGS-B algorithm in the Python package scipy in the deterministic full batch case. For stochastic optimization we used the stochastic optimizer ADAM [41] (implemented from scratch) with appropriate learning rates which are learnt in preliminary experiments.

For the competitor methods we used the implementation in GPY [40] for full and sparse global GP (the approach of [5]). For PoE, GPoE, BCM, RBCM and GRBCM we implemented the corresponding aggregation algorithms based on the GPY implementations for the independent experts in Python for the sake of comparisons. For the stochastic version of SGP, the hybrid information filter approach in [10] and their implementations are used. We also run the approaches REC [11] and SVI [9], however the former approach shows superior accuracy vs. time performance in preliminary experiments.

For the sparse global GP model there is the choice of optimized or fixed inducing points. For the same number of inducing points the accuracy is obviously better with optimized inducing points, however taking into account the time for optimizing them, we found in the experiments with batch optimization (i.e. also smaller datasets) that the fixed random subset approach was superior. Therefore we report here the results for fixed (random subset of data) inducing points in the deterministic case and optimized in the stochastic case. The reason for that is that the sparse global approximation with unknown inducing inputs has $MD + |\theta|$ (variational) parameters to optimize in the batch version. In the stochastic version REC & IF there are as well $MD + |\theta|$ parameters, whereas SVI has even $M + 0.5M^2 + MD + |\theta|$ number of parameters since the posterior mean and covariance has to be optimized.

⁶https://github.com/manschuer/CPoE/experiments/example_1D.ipynb

⁷https://github.com/manschuer/CPoE/experiments/example_2D.ipynb

On the other hand, full GP has only a few kernel hyperparameters $|\boldsymbol{\theta}|$ to optimize. Similarly, our method CPoE (and also independent PoEs) inherit this property because there is no necessity to optimize the local inducing points since the total amount of them can be in the order of N . This is also true for the stochastic version of our algorithm. Assume for instance $D = 8$ and $M = 100$, the number of parameters with a SE kernel for full GP and CPoE are only $|\boldsymbol{\theta}| = 10$ parameters to optimize, whereas for batch SGP, REC & IF 810 and even 5910 for SVI. For fixed inducing points, SGP and IF also only have $|\boldsymbol{\theta}| = 10$ hyperparameters which allows to have more inducing points but speed-up the optimization a lot and makes the accuracy vs. time comparison more competitive.

We used the KD-partition for our method as discussed in 3.1 while in the PoE-literature [12–16], often K-Means is used for partitioning. However, for large J and N this is quite inefficient and often the partition sizes for each expert differs significantly which introduces an imbalance among the experts in the prediction aggregation as well as in the stochastic optimization. Therefore we also used the KD-tree partition for these algorithms for the sake of comparisons.

For assessing the quality of the different algorithms in the next sections, we report the two quantities *the Kullback-Leibler-(KL)-divergence* to full GP and the *Continuous Ranked Probability Score (CRPS)* both depending on the pointwise predictive distributions $p(f_*|\mathbf{y})$. The reported values correspond always to an average of N_{test} prediction points which are not contained in the training data.

A.7 Experiments

In this section, we provide more details about the experiments in Section 4.

A.7.1 Synthetic Data

In this simulation study with synthetic GP data, we examine the accuracy vs. time performance of different GP algorithms for fixed hyperparameters. We generated $N = 8192$ data samples in $D = 2$ with 5 repetitions from the sum of two SE kernels with a shorter and longer lengthscale ($l_s = 0.125, v_s = 0.2$ and $l_l = 0.5, v_l = 1.1$; see Fig. A5) such that both global and local patterns are present in the data. In Fig. 9 the mean results are shown for the KL and RMSE to full GP, the 95%-coverage and the log marginal likelihood against time in seconds.

For the sparse GP, we use different number of fixed global inducing points $M = \{50, \dots, 1000\}$ for which the results are shown in blue.⁸ From the PoE-family, the results for minVar, GPoE and BCM are depicted for different number of

⁸We also run sparse GP with optimized inducing points, however the performance compared to time was worse.

| | CRPS | | | | time | | | |
|----------------|--------------|--------------|--------------|--------------|-------|--------|--------|--------|
| | kin2 | cadata | sarcos | casp | kin2 | cadata | sarcos | casp |
| SGP(500) | 0.183 | 0.253 | 0.069 | 0.329 | 112.1 | 346.9 | 730.1 | 632.9 |
| SGP(1000) | 0.166 | 0.252 | 0.063 | 0.325 | 244.1 | 727.6 | 1718.5 | 1362.5 |
| minVar | 0.173 | 0.257 | 0.052 | 0.294 | 14.4 | 28.2 | 71.3 | 45.8 |
| GPoE | 0.193 | 0.289 | 0.086 | 0.302 | 14.4 | 28.3 | 71.4 | 45.6 |
| GRBCM | 0.164 | 0.262 | 0.060 | 0.310 | 16.5 | 33.5 | 84.6 | 59.4 |
| CPoE(1) | 0.163 | 0.259 | 0.052 | 0.289 | 13.8 | 24.5 | 45.4 | 45.1 |
| CPoE(2) | 0.155 | 0.251 | 0.051 | 0.287 | 18.9 | 33.4 | 67.3 | 70.3 |
| CPoE(3) | 0.151 | 0.249 | 0.051 | 0.282 | 31.7 | 52.0 | 134.3 | 123.8 |

Table A4: Average CRPS (left) and time (right) for different GP methods and 4 datasets with 5 repetitions. More details and results are provided in Sections A.7.2 and F in the Appendix.

experts $J = \{1, 2, 4, \dots, 128\}$ in red, cyan and magenta, respectively. For our correlated PoEs, the results for the correlations $C = \{1, \dots, 12\}$ are shown in green for $J = 32$ and $\gamma = 0.5$.

In the first two plots, the superior performance of our method compared to competitors in accuracy to full GP vs. time can be observed. Our method constitutes a fast and accurate method for a range of different approximation qualities. Moreover, in the third plot, one can observe that the confidence informations are reliable already for small approximation orders since it is based on the consistent covariance intersection method. This experiment is available in a notebook⁹ on github.

A.7.2 Real World Data

In this section, we provide more details about the experiments with real world data as summarized in Section 4. The used datasets in this paper, as summarized in Table 3a, are from public data repositories and can be downloaded as shown in a jupyter notebook¹⁰ on github. In particular, it shows where and how to download the raw datasets, which variables are used as input and the target variables, where we followed the remarks on the repositories or other authors previously worked with these datasets. Finally, the preprocessing of the data is shown, where we standardized all variables to mean zero and standard deviation of one (for *elecdemand* see details below). We use $N = \min(0.9N_{tot}, 1000)$ data sample for training, the rest for testing; except for *kin* and *elecdemand* we run experiments with $N_{test} = 3000$ and $N_{test} = 15288$ such that we could also run full GP a standard Laptop. For each dataset we fixed the number J of experts (given in Table 3a) such that the partitions/mini-batches have a reasonable size (≈ 500).

For the deterministic SGP we used $M = 100$ and for the stochastic SGP $M \in \{500, 1000\}$ inducing points (more results are provided in Appendix F).

⁹<https://github.com/manschuer/CPoE/experiments/syntheticData.ipynb>

¹⁰https://github.com/manschuer/CPoE/experiments/download_data.ipynb

For our method CPoE we run the algorithm for $C \in \{1, 2, 3, 4\}$ for the small and $C \in \{1, 2, 3\}$ for the large datasets with always $\gamma = 1$. For the stochastic versions we used learning rates $\delta = 0.03$ for the dataset *kin2* and $\delta = 0.01$ for the remaining for all methods. The maximum number of epochs is set to 15 together with a relative stopping criteria of $1e^{-2}$. We use a SE-kernel with a different lengthscale per dimension and initialized all hyperparameters to 1, and the global inducing point to a random subset of the data. These experiments are provided at github. ^{11 12}

A.7.3 Application

This section contains additional details to the application described in Section 4.4, where our method is applied to the *elecDemand* time series [37], which contains the half-hourly measured electricity demand together with the corresponding temperature and the variable whether it is a working day for 1 year. In particular, the preprocessed dataset contains the standardized electricity demand (mean=0, sd=1) as the response variable y , the normalized time as the first variable $X_1 \in [0, 1]$, the standardized temperature and indicators as X_2 and X_3 , respectively. The data is depicted in the first two plots in Fig. 10, where we shifted the first and third variable in the second plot for the sake of clarity. We removed the last day resulting in 364 days = 52 weeks = 13 "months" consisting of 4 weeks. In each of the 13 "months", we used the first 3 weeks for training and the last week for testing the out-of-sample accuracy. In order that it is possible to run full GP as comparison, we only used every 6th sample (corresponding to a measurement every 3h) of the training weeks for the actual training and the remaining for testing the in-sample accuracy. This gives $N = 2184$, $N_{IN} = 10920$ and $N_{OUT} = 4368$ samples as depicted in the first plot in Fig. 10.

With our CPoE model it is straightforward to handle time series with covariates, as opposed to other time series methods [33–36]. The kernel k_{θ} depends on several hyperparameters θ , for which we use the parametrization in [33]. We assume a log-normal prior on θ as described in Section A.3.3 in which the corresponding means and variances are taken from Table 1 in [33]. Similarly as in the previous section, we run full GP, SGP and PoEs and CPoE and optimized the hyperparameter deterministically using the MAP as objective function taking into account the priors. For SGP we used $M \in \{100, 200\}$ fixed inducing points, for PoEs and CPoE we used $J = 13$ partitions which are obtained by splitting the first variable into J blocks. For CPoE we used $C \in \{1, 2, 3, 4\}$ and $\gamma = 1$. The results are provided in Table 3b which again shows very competitive performance also for a general kernel with priors on the hyperparameters.

¹¹ <https://github.com/manschuer/CPoE/experiments/realData1.ipynb>

¹² <https://github.com/manschuer/CPoE/experiments/realData2.ipynb>

Appendix B Useful properties

B.0.1 Inversion Lemma

Given invertible matrices $\mathbf{A} \in \mathbb{R}^{B \times B}$, $\mathbf{C} \in \mathbb{R}^{M \times M}$ and matrices $\mathbf{U} \in \mathbb{R}^{B \times M}$, $\mathbf{V} \in \mathbb{R}^{M \times B}$, it holds

$$(\mathbf{A} + \mathbf{UCV})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}(\mathbf{C}^{-1} + \mathbf{VA}^{-1}\mathbf{U})^{-1}\mathbf{VA}^{-1}. \quad (\text{B2})$$

B.0.2 Determinant Lemma

Given invertible matrices $\mathbf{A} \in \mathbb{R}^{B \times B}$, $\mathbf{C} \in \mathbb{R}^{M \times M}$ and matrices $\mathbf{U} \in \mathbb{R}^{B \times M}$, $\mathbf{V} \in \mathbb{R}^{M \times B}$, it holds

$$|\mathbf{A} + \mathbf{UCV}| = |\mathbf{C}^{-1} + \mathbf{VA}^{-1}\mathbf{U}||\mathbf{C}||\mathbf{A}|. \quad (\text{B3})$$

B.0.3 Block Inversion

Given an invertible, symmetric block matrix

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{D} \end{bmatrix},$$

the inverse can be computed as

$$\mathbf{M}^{-1} = \begin{bmatrix} \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{B}\mathbf{Z}^{-1}\mathbf{B}^T\mathbf{A}^{-1} & -\mathbf{A}^{-1}\mathbf{B}\mathbf{Z}^{-1} \\ -\mathbf{Z}^{-1}\mathbf{B}^T\mathbf{A}^{-1} & \mathbf{Z}^{-1} \end{bmatrix} \quad (\text{B4})$$

with $\mathbf{Z} = \mathbf{D} - \mathbf{B}^T\mathbf{A}^{-1}\mathbf{B}$.

B.0.4 Block Determinant

Given an invertible, symmetric block matrix

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{D} \end{bmatrix},$$

the determinant can be computed as

$$|\mathbf{M}| = |\mathbf{A}| |\mathbf{D} - \mathbf{B}^T\mathbf{A}^{-1}\mathbf{B}| = |\mathbf{D}| |\mathbf{A} - \mathbf{BD}^{-1}\mathbf{B}^T|. \quad (\text{B5})$$

B.0.5 Conditional Gaussians

From the joint Gaussian $[\mathbf{a}, \mathbf{b}]^T \sim \mathcal{N}(\mathbf{0}, \mathbf{K}_{[\mathbf{A}, \mathbf{B}][\mathbf{A}, \mathbf{B}]})$, the conditional can be computed as follows

$$\mathbf{a}|\mathbf{b} \sim \mathcal{N}(\mathbf{K}_{\mathbf{AB}}\mathbf{K}_{\mathbf{BB}}^{-1}\mathbf{b}, \mathbf{K}_{\mathbf{AA}} - \mathbf{K}_{\mathbf{AB}}\mathbf{K}_{\mathbf{BB}}^{-1}\mathbf{K}_{\mathbf{BA}}) = \mathcal{N}(\mathbf{H}_{\mathbf{AB}}\mathbf{b}, \mathbf{V}_{\mathbf{AA}}^{\mathbf{B}}) \quad (\text{B6})$$

B.0.6 Marginalization/Integration

Given the densities $p(\mathbf{a}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $p(\mathbf{b}|\mathbf{a}) = \mathcal{N}(\mathbf{F}\mathbf{a} + \mathbf{v}, \mathbf{Q})$, then

$$p(\mathbf{b}) = \int p(\mathbf{a}, \mathbf{b}) d\mathbf{a} = \int p(\mathbf{b}|\mathbf{a})p(\mathbf{a}) d\mathbf{a} = \mathcal{N}(\mathbf{F}\boldsymbol{\mu} + \mathbf{v}, \mathbf{F}\boldsymbol{\Sigma}\mathbf{F}^T + \mathbf{Q}) \quad (\text{B7})$$

B.0.7 Gaussian & Bayes

Given the densities $p(\mathbf{a}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $p(\mathbf{b}|\mathbf{a}) = \mathcal{N}(\mathbf{F}\mathbf{a} + \mathbf{v}, \mathbf{Q})$, applying Bayes' formula yields

$$p(\mathbf{a}|\mathbf{b}) = \mathcal{N}(\mathbf{P}(\mathbf{F}^T\mathbf{Q}^{-1}(\mathbf{b} - \mathbf{v}) + \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}), \mathbf{P}), \quad (\text{B8})$$

with $\mathbf{P} = (\boldsymbol{\Sigma}^{-1} + \mathbf{F}^T\mathbf{Q}^{-1}\mathbf{F})^{-1}$.

B.0.8 Product of Gaussians

Assume J Gaussians $p_j(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$ and $a_j \in \mathbb{R}$. Then the product can be written as

$$\prod_{j=1}^J p_j(\mathbf{x})^{a_j} = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \quad (\text{B9})$$

with

$$\boldsymbol{\Sigma} = \left(\sum_{j=1}^J a_j \boldsymbol{\Sigma}_j^{-1} \right)^{-1} \quad \text{and} \quad \boldsymbol{\mu} = \boldsymbol{\Sigma} \left(\sum_{j=1}^J a_j \boldsymbol{\Sigma}_j^{-1} \boldsymbol{\mu}_j \right)$$

as long as $\boldsymbol{\Sigma}$ positive-semi-definite (if $a_j \equiv 1$ then always the case).

B.0.9 Entropy of Gaussian

The *Entropy* H of $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with $|\mathbf{x}| = B$ is defined as

$$H[\mathbf{x}] = H[p(\mathbf{x})] = \frac{1}{2} (\log |\boldsymbol{\Sigma}| + B(1 + \log 2\pi)), \quad (\text{B10})$$

where we use *log* as the natural logarithm and thus the entropy is measured in nats (natural units).

B.0.10 Kullback-Leibler-Divergence (KL)

The KL between $p_0(\mathbf{x})$ and $p_1(\mathbf{x})$ is defined as

$$KL[p_0(\mathbf{x}) | p_1(\mathbf{x})] = \int p_0(\mathbf{x}) \log \frac{p_0(\mathbf{x})}{p_1(\mathbf{x})} d\mathbf{x}. \quad (\text{B11})$$

B.0.11 KL between 2 Gaussians

The *Kullback-Leibler-Divergence* (KL) between $p_0(\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ and $p_1(\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ with $|\mathbf{x}| = B$ can be computed by

$$\begin{aligned} KL[p_0(\mathbf{x}) | p_1(\mathbf{x})] &= \frac{1}{2} \left(\text{tr}(\boldsymbol{\Sigma}_1^{-1} \boldsymbol{\Sigma}_0) - B \right. \\ &\left. + (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T \boldsymbol{\Sigma}_1^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) + \log \frac{|\boldsymbol{\Sigma}_1|}{|\boldsymbol{\Sigma}_0|} \right). \end{aligned} \quad (\text{B12})$$

B.0.12 Difference in KL of Gaussian with Zero Mean

The difference in KL between $p_1(\mathbf{x}) = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_1)$ and $p_2(\mathbf{x}) = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_2)$ with same base distribution $p_0(\mathbf{x}) = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_0)$ can be computed by

$$KL[p_0(\mathbf{x}) | p_1(\mathbf{x})] - KL[p_0(\mathbf{x}) | p_2(\mathbf{x})] = \frac{1}{2} \left(\text{tr}((\boldsymbol{\Sigma}_1^{-1} - \boldsymbol{\Sigma}_2^{-1}) \boldsymbol{\Sigma}_0) + \log \frac{|\boldsymbol{\Sigma}_1|}{|\boldsymbol{\Sigma}_2|} \right). \quad (\text{B13})$$

B.0.13 General Difference in KL

Let $1 \leq C < J$ and $0 < \gamma \leq 1$ be fixed. For any $C_2 \in \{C, \dots, J\}$ we define the difference in KL, denoted as $\mathbb{D}_{(C, C_2)}[\mathbf{x}]$, between the true distribution of \mathbf{x} and two different approximate distributions, i.e.

$$\mathbb{D}_{(C, C_2)}[\mathbf{x}] = KL[p(\mathbf{x}) || q_{c, \gamma}(\mathbf{x})] - KL[p(\mathbf{x}) || q_{c_2, \gamma}(\mathbf{x})] = \mathbb{E}_{p(\mathbf{x})} \left[\log \frac{q_{c_2, \gamma}(\mathbf{x})}{q_{c, \gamma}(\mathbf{x})} \right] \quad (\text{B14})$$

using the definition of KL (B11). Similarly, we define the the difference in KL, denotes as $\mathbb{D}_{(C, C_2)}[\mathbf{x}|\mathbf{y}]$, of a conditional distribution $\mathbf{x}|\mathbf{y}$ to be

$$\begin{aligned} &KL[p(\mathbf{x}|\mathbf{y}) || q_{c, \gamma}(\mathbf{x}|\mathbf{y})] - KL[p(\mathbf{x}|\mathbf{y}) || q_{c_2, \gamma}(\mathbf{x}|\mathbf{y})] \\ &= \mathbb{E}_{p(\mathbf{y})} \left[\mathbb{E}_{p(\mathbf{x}|\mathbf{y})} \left[\log \frac{q_{c_2, \gamma}(\mathbf{x}|\mathbf{y})}{q_{c, \gamma}(\mathbf{x}|\mathbf{y})} \right] \right]. \end{aligned} \quad (\text{B15})$$

which follows from the the definition of KL (B11).

Appendix C Proofs and Additional Results

C.1 Additional Results

Proposition 9 (Marginal Likelihood; Proof 13) *The marginal likelihood is*

$$q_{C, \gamma}(\mathbf{y}|\boldsymbol{\theta}) = q_{C, \gamma}(\mathbf{y}) = \int q_{C, \gamma}(\mathbf{y}, \mathbf{a}) \, d\mathbf{a} = \mathcal{N}(\mathbf{0}, \mathbf{P})$$

with $\mathbf{P} = \mathbf{H}\mathbf{S}^{-1}\mathbf{H}^T + \mathbf{V} \in \mathbb{R}^{N \times N}$ where all dependencies on $\boldsymbol{\theta}$ of the matrices are omitted.

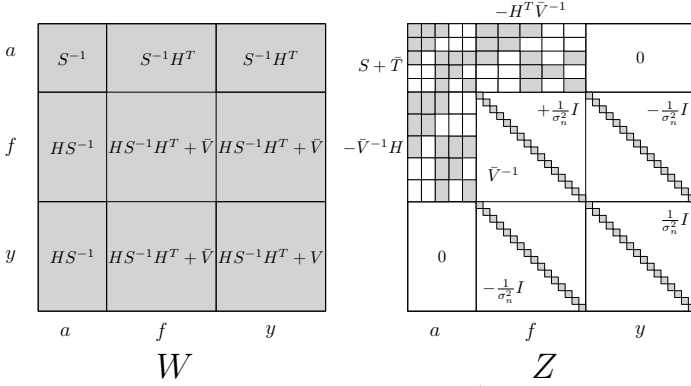


Fig. C7: Covariance \mathbf{W} and precision $\mathbf{Z} = \mathbf{W}^{-1}$ of joint prior approximation $q_{c,\gamma}(\mathbf{a}, \mathbf{f}, \mathbf{y} = \mathcal{N}(\mathbf{0}, \mathbf{W}))$ of CPoE model. Compare Proof 19 and Figures C8 and C9 for comparison.

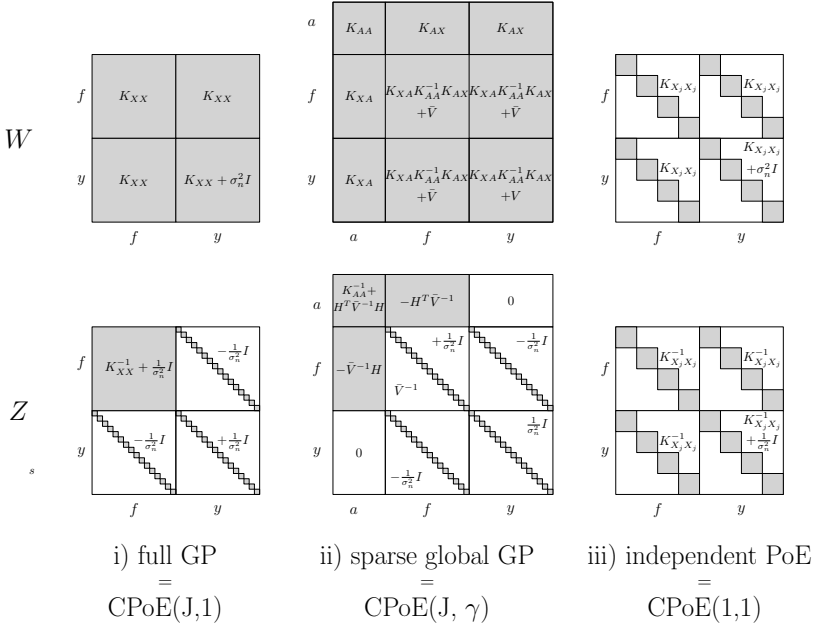


Fig. C8: Covariance \mathbf{W} and precision $\mathbf{Z} = \mathbf{W}^{-1}$ of joint prior of different GP models. Compare Figure C7 for the corresponding covariance and precision matrices for CPoE model. Note that we used $\mathbf{H} = \mathbf{K}_{\mathbf{X}\mathbf{A}}\mathbf{K}_{\mathbf{A}\mathbf{A}}^{-1}$ and $\bar{\mathbf{V}}$ the same as in the local CPoE.

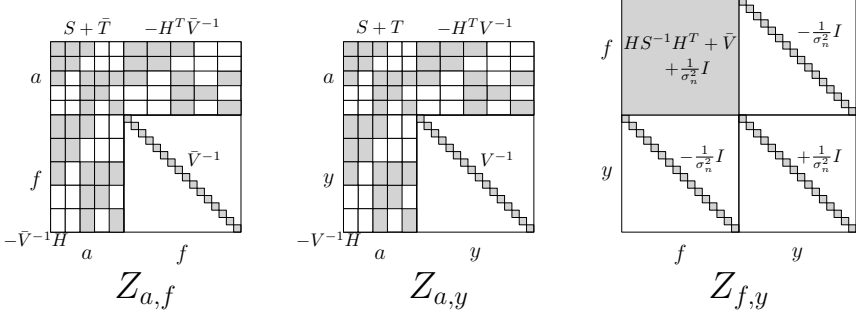


Fig. C9: Marginalized precision corresponding to $q_{c,\gamma}(\mathbf{a}, \mathbf{f}) = \mathcal{N}(\mathbf{0}, \mathbf{Z}_{\mathbf{a},\mathbf{f}}^{-1})$, $q_{c,\gamma}(\mathbf{a}, \mathbf{y}) = \mathcal{N}(\mathbf{0}, \mathbf{Z}_{\mathbf{a},\mathbf{y}}^{-1})$ and $q_{c,\gamma}(\mathbf{f}, \mathbf{a}) = \mathcal{N}(\mathbf{0}, \mathbf{Z}_{\mathbf{f},\mathbf{y}}^{-1})$, respectively. Thereby we used the notation $\mathbf{V} = \bar{\mathbf{V}} + \sigma_n^2 \mathbb{I}$, $\bar{\mathbf{T}} = \mathbf{H}^T \bar{\mathbf{V}}^{-1} \mathbf{H}$ and $\mathbf{T} = \mathbf{H}^T \mathbf{V}^{-1} \mathbf{H}$. Note that the corresponding dense covariance matrices are directly obtained from \mathbf{W} in Fig. C7 by selecting the corresponding entries.

Proposition 10 (Prior Approximation II; Proof 15) *Alternatively to Proposition 2, the prior approximation $q(\mathbf{a}) = \mathcal{N}(\mathbf{a}|\mathbf{0}, \mathbf{S}^{-1})$ can be equivalently written as*

$$q(\mathbf{a}) = \prod_{j=1}^J p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) = \prod_{j=1}^J \mathcal{N}(\mathbf{a}_{\pi^+(j)} | \mathbf{0}, \mathbf{S}_{(j)}^{-1})$$

with $\mathbf{S}_{(j)} = \tilde{\mathbf{F}}_j^T \mathbf{Q}_j^{-1} \tilde{\mathbf{F}}_j$, $\tilde{\mathbf{F}}_j = [-\mathbf{F}_j \ \mathbb{I}]$ and $\pi^+(j) = \pi(j) \cup j$. Further, the prior precision matrix can also be written as

$$\mathbf{S} = \sum_{j=1}^J \bar{\mathbf{S}}_{(j)}$$

where $\bar{\mathbf{S}}_{(j)} \in \mathbb{R}^{M \times M}$ is the augmented matrix consisting of $\mathbf{S}_{(j)} \in \mathbb{R}^{LC \times LC}$ at the entries $[\pi^+(j), \pi^+(j)]$ and 0 otherwise.

Proposition 11 (Prior Approximation III; Proof 16) *Alternatively to Prop. 2 and Prop. 10 the prior approximation $q(\mathbf{a})$ can be equivalently written as*

$$q(\mathbf{a}) = \prod_{j=1}^J \frac{p(\mathbf{a}_j, \mathbf{a}_{\pi(j)})}{p(\mathbf{a}_{\pi(j)})} = \prod_{j=C}^J \frac{p(\mathbf{a}_{\pi^+(j)})}{p(\mathbf{a}_{\pi(j)})}$$

which is a Gaussian $\mathcal{N}(\mathbf{a}|\mathbf{0}, \mathbf{S}^{-1})$ with prior precision

$$\mathbf{S} = \sum_{j=1}^J \bar{\mathbf{K}}_{\mathbf{A}_{\pi^+(j)} \mathbf{A}_{\pi^+(j)}}^{-1} - \bar{\mathbf{K}}_{\mathbf{A}_{\pi(j)} \mathbf{A}_{\pi(j)}}^{-1}$$

where $\bar{\mathbf{K}}_{\mathbf{A}_\phi \mathbf{A}_\phi}^{-1} \in \mathbb{R}^{M \times M}$ is the augmented matrix consisting of $\mathbf{K}_{\mathbf{A}_\phi \mathbf{A}_\phi}^{-1} \in \mathbb{R}^{T \times T}$ at the entries $[\phi, \phi]$ and 0 otherwise with $T = |\phi|$.

Proposition 12 (Exact Diagonal of Prior; Proof 14) *The precision matrix \mathbf{S}_C of the prior approximation $q_C(\mathbf{a})$ is exact on the diagonal, that is,*

$$\text{tr}(\mathbf{S}_C \mathbf{K}_{\mathbf{AA}}) = JL$$

where JL is the dimension of the matrices.

Proposition 13 (Band-Diagonal Approximation) *In the **consecutive** case, i.e. $\psi(j) = \{j - C + 1, \dots, j\}$, the block-entries*

$$\mathbf{S}_{[\psi(j), \psi(j)]}^{-1} = \mathbf{K}_{\mathbf{AA}[\psi(j), \psi(j)]},$$

are equal which means that the block-band-diagonals $-C + 1, \dots, 0, \dots, C - 1$ of the both matrices are the same. For the case $C = J$ it holds $\mathbf{S}^{-1} = \mathbf{K}_{\mathbf{AA}}$.

Proposition 14 (Decreasing Prior Entropy; Proof 10) *For any predecessor structure π_C as in Def. 2, the entropy H of the approximate prior $q_C(\mathbf{a})$ is decreasing for $C \rightarrow J$, in particular*

$$H[q_1(\mathbf{a})] \geq \dots \geq H[q_j(\mathbf{a})] \geq \dots \geq H[q_J(\mathbf{a})]$$

where it holds $H[q_J(\mathbf{a})] = H[p(\mathbf{a})]$ and

$$H[q_j(\mathbf{a})] = \frac{1}{2} \log |\mathbf{Q}_C| + \frac{M}{2} (1 + \log 2\pi).$$

Similar results can be obtained for the joint prior $q_C(\mathbf{a}, \mathbf{f}, \mathbf{y})$.

From the last proposition we know that increasing the degree of correlation C add always more information to the prior. In particular, the prior of complete independent PoEs (i.e. $C = 1$) encodes the least of information since all correlations between the experts are missing, whereas the prior of full GP incorporates the most information since all correlations are modeled.

Proposition 15 (Prior Quality II) *The prior approximation quality improvement $\mathbb{D}_{(C,T)}[\mathbf{a}]$ in Prop. 6 can be equivalently written as*

$$\begin{aligned} \mathbb{D}_{(C,T)} &= \frac{1}{2} \log \frac{|\mathbf{S}_{C+T}|}{|\mathbf{S}_C|} = \frac{1}{2} \sum_{j=1}^J \log \frac{|\mathbf{Q}_j | \pi_j | |\mathbf{Q}_{\phi_j | \pi_j}|}{|\mathbf{Q}_{j \cup \phi_j | \pi_j}|} \\ &= \frac{1}{2} \sum_{j=1}^J \log \frac{|\mathbf{K}_{\mathbf{A}_j \cup \pi_j} \mathbf{A}_{j \cup \pi_j}| |\mathbf{K}_{\mathbf{A}_{\phi_j \cup \pi_j}} \mathbf{A}_{\phi_j \cup \pi_j}|}{|\mathbf{K}_{\mathbf{A}_j \cup \phi_j \cup \pi_j} \mathbf{A}_{j \cup \phi_j \cup \pi_j}| |\mathbf{K}_{\mathbf{A}_{\pi_j}} \mathbf{A}_{\pi_j}|} \end{aligned}$$

where $\pi_j = \pi_C(j)$, $\phi_j = \cup_{i=C+1}^{C+T} \phi_i(j)$ and $\mathbf{Q}_{\varphi_1 | \varphi_2} = \mathbf{K}_{\mathbf{A}_{\varphi_1} \mathbf{A}_{\varphi_1}} - \mathbf{K}_{\mathbf{A}_{\varphi_1} \mathbf{A}_{\varphi_2}} \mathbf{K}_{\mathbf{A}_{\varphi_2} \mathbf{A}_{\varphi_2}}^{-1} \mathbf{K}_{\mathbf{A}_{\varphi_2} \mathbf{A}_{\varphi_1}}$.

C.2 Proofs

Proof 1 (Proof of Prop. 1; Joint Distribution) . The matrices in the conditional distributions (6) and (7) in Prop. 1 can be obtained via Gaussian conditioning (B6) from the assumed joint densities

$$p(\mathbf{f}_j^i, \mathbf{a}_{\psi(j)}) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{[\mathbf{X}_j^i; \mathbf{A}_{\psi(j)}][\mathbf{X}_j^i; \mathbf{A}_{\psi(j)}]});$$

$$p(\mathbf{a}_j, \mathbf{a}_{\pi(j)}) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{[A_j, A_{\pi(j)}][A_j, A_{\pi(j)}]}),$$

resulting in

$$\begin{aligned} \mathbf{H}_j &= \mathbf{K}_{X_j A_{\psi(j)}} \mathbf{K}_{A_{\psi(j)} A_{\psi(j)}}^{-1}; \\ \bar{\mathbf{V}}_j &= \text{Diag}[\mathbf{K}_{X_j X_j} - \mathbf{K}_{X_j A_{\psi(j)}} \mathbf{K}_{A_{\psi(j)} A_{\psi(j)}}^{-1} \mathbf{K}_{A_{\psi(j)} X_j}]; \\ \mathbf{F}_j &= \mathbf{K}_{A_j A_{\pi(j)}} \mathbf{K}_{A_{\pi(j)} A_{\pi(j)}}^{-1}; \\ \mathbf{Q}_j &= \mathbf{K}_{A_j A_j} - \mathbf{K}_{A_j A_{\pi(j)}} \mathbf{K}_{A_{\pi(j)} A_{\pi(j)}}^{-1} \mathbf{K}_{A_{\pi(j)} A_j} \end{aligned}$$

with $\mathbf{F}_1 = \mathbf{0}$ and $\mathbf{Q}_1 = \mathbf{K}_{A_1 A_1}$.

Proof 2 (Proof used in Def. 4; Joint Distribution II) In the case $\gamma = 1$, thus $\mathbf{a}_j = \mathbf{f}_j$ and $\mathbf{a} = \mathbf{f}$, the joint distribution can be written as $q(\mathbf{f}, \mathbf{a}, \mathbf{y}) = q(\mathbf{f}, \mathbf{f}, \mathbf{y}) = q(\mathbf{f}, \mathbf{y})$ is

$$\begin{aligned} q(\mathbf{f}, \mathbf{y}) &= \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{a}_{\psi(j)}) p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) \\ &= \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{f}_{\psi(j)}) p(\mathbf{f}_j | \mathbf{f}_{\pi(j)}) \\ &= \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) \frac{p(\mathbf{f}_j \mathbf{f}_{\psi(j)})}{p(\mathbf{f}_{\psi(j)})} p(\mathbf{f}_j | \mathbf{f}_{\pi(j)}) \\ &= \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{f}_{\pi(j)}) \end{aligned}$$

since

$$\frac{p(\mathbf{f}_j, \mathbf{f}_{\psi(j)})}{p(\mathbf{f}_{\psi(j)})} = \frac{p(\mathbf{f}_j, \mathbf{f}_j, \mathbf{f}_{\psi(j) \setminus j})}{p(\mathbf{f}_j, \mathbf{f}_{\psi(j) \setminus j})} = 1.$$

Proof 3 (Proof of Prop. 5; Equality to Full GP) **Full GP:** For $\gamma = 1$, the joint distribution of our model is formulated in Def. 4 and Proof 2. For $C = J$, we have

$$q_J(\mathbf{f}, \mathbf{y}) = \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{f}_{\pi_J(j)}),$$

where the predecessor set $\pi_J(j)$ correspond to $\{1, \dots, j-1\}$ and thus the conditional variables $\mathbf{f}_{\pi(j)} = \mathbf{f}_{1:j-1}$. The posterior $q_J(\mathbf{f} | \mathbf{y})$ is proportional to the joint distribution $q_J(\mathbf{f}, \mathbf{y})$ (see Proof 12), thus we have

$$q_J(\mathbf{f} | \mathbf{y}) \propto \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{f}_{1:j-1})$$

which is equal to the posterior distribution of full GP (1). Also the hyperparameter optimization is the same since the marginal likelihood $q_J(\mathbf{y})$ can be derived from the joint $q_C(\mathbf{f}, \mathbf{y})$ (see Proof 13). Further, in the prediction step, for $C = J$

where $\mathbf{F}_j^i \in \mathbb{R}^{L \times L}$ is the i th part of $\mathbf{F}_j \in \mathbb{R}^{L \times L(C-1)}$ which correspond to the contribution of the i th predecessor $\boldsymbol{\pi}^i(j)$.

By using the property in (B9), the original product $q(\mathbf{a})$ is then

$$\begin{aligned} \prod_{j=1}^J \mathcal{N}\left(\mathbf{a}|\mathbf{0}, \left(\bar{\mathbf{F}}_j^T \bar{\mathbf{Q}}_j^{-1} \bar{\mathbf{F}}_j\right)^{-1}\right) &= \mathcal{N}\left(\mathbf{a}|\mathbf{0}, \left(\sum_{j=1}^J \bar{\mathbf{F}}_j^T \bar{\mathbf{Q}}_j^{-1} \bar{\mathbf{F}}_j\right)^{-1}\right) \\ &= \mathcal{N}\left(\mathbf{a}|\mathbf{0}, \left(\mathbf{F}^T \mathbf{Q}^{-1} \mathbf{F}\right)^{-1}\right) = \mathcal{N}\left(\mathbf{a}|\mathbf{0}, \mathbf{S}^{-1}\right) \end{aligned}$$

with $\mathbf{Q}^{-1} = \text{Diag}[\mathbf{Q}_1^{-1}, \dots, \mathbf{Q}_J^{-1}]$ and \mathbf{F} correspond then to the matrix depicted in Fig. 5. Note that \mathbf{S} is positive definite since \mathbf{Q}^{-1} positive definite because each \mathbf{Q}_j^{-1} is positive definite which concludes the proof.

Proof 5 ((Sub)proof of Prop. 2 (Projection Approximation)) The projection $q(\mathbf{f}|\mathbf{a}) = q_C(\mathbf{f}|\mathbf{a})$ is

$$q_C(\mathbf{f}|\mathbf{a}) = \prod_{j=1}^J p\left(\mathbf{f}_j|\mathbf{a}_{\psi(j)}\right) = \prod_{j=1}^J \mathcal{N}\left(\mathbf{f}_j|\mathbf{H}_j \mathbf{a}_{\psi(j)}, \bar{\mathbf{V}}_j\right),$$

where $\mathbf{H}_j \in \mathbb{R}^{B \times LC}$ and $\bar{\mathbf{V}}_j \in \mathbb{R}^{B \times B}$. The log of this density in $\mathbf{f}_j \in \mathbb{R}^B$ is proportional to

$$\propto -\frac{1}{2}(\mathbf{f}_j - \mathbf{H}_j \mathbf{a}_{\psi(j)})^T \bar{\mathbf{V}}_j^{-1} (\mathbf{f}_j - \mathbf{H}_j \mathbf{a}_{\psi(j)})$$

which can be equivalently written as

$$-\frac{1}{2}(\mathbb{I}_j \mathbf{f} - \bar{\mathbf{H}}_j \mathbf{a})^T \bar{\mathbf{V}}_j^{-1} (\mathbb{I}_j \mathbf{f} - \bar{\mathbf{H}}_j \mathbf{a})$$

with $\bar{\mathbf{V}}_j \in \mathbb{R}^{M \times M}$ with $\bar{\mathbf{V}}_j$ at $[\psi(j), \psi(j)]$ and $\bar{\mathbf{H}}_j \in \mathbb{R}^{BJ \times M}$ the following matrix

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & \mathbf{H}_j^1 & 0 & \mathbf{H}_j^i & \dots & \mathbf{H}_j^{C-1} & \mathbf{H}_j^C & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

where j th row not empty with $\mathbf{H}_j^i \in \mathbb{R}^{B \times L}$ the i th entry in \mathbf{H}_j which correspond to $\boldsymbol{\psi}^i(j)$. Further, $\mathbb{I}_j \in \mathbb{R}^{BJ \times BJ}$ a zero matrix with \mathbb{I} in $\mathbb{R}^{B \times B}$ at $[j, j]$. For the original product of the projections

$$q_C(\mathbf{f}|\mathbf{a}) = \prod_{j=1}^J \mathcal{N}\left(\mathbf{0}|\mathbb{I}_j^T \mathbf{f} - \bar{\mathbf{H}}_j \mathbf{a}, \bar{\mathbf{V}}_j\right),$$

using the product rule of Gaussians in (B9), we obtain

$$\mathcal{N}\left(\mathbf{0}|\bar{\mathbf{V}} \sum_j \bar{\mathbf{V}}_j^{-1} (\mathbb{I}_j \mathbf{f} - \bar{\mathbf{H}}_j \mathbf{a}), \left(\sum_j \bar{\mathbf{V}}_j^{-1}\right)^{-1}\right)$$

$$\begin{aligned}
&= \mathcal{N} \left(\mathbf{0} | \overline{\mathbf{V}} \overline{\mathbf{V}}^{-1} \sum_j^J (\mathbb{I}_j \mathbf{f} - \overline{\mathbf{H}}_j \mathbf{a}), \overline{\mathbf{V}} \right) = \mathcal{N} \left(\mathbf{0} | \sum_j^J (\mathbb{I}_j) \mathbf{f} - \sum_j^J (\overline{\mathbf{H}}_j) \mathbf{a}, \overline{\mathbf{V}} \right) \\
&= \mathcal{N} (\mathbf{0} | \mathbb{I} \mathbf{f} - \mathbf{H} \mathbf{a}, \overline{\mathbf{V}}) = \mathcal{N} (\mathbf{f} | \mathbf{H} \mathbf{a}, \overline{\mathbf{V}}).
\end{aligned}$$

Since $\overline{\mathbf{V}}$ positive definite this concludes the statement.

Proof 6 (Proof of Prop. 6; Decreasing Prior KL) We first show the decomposition

$$\mathbb{D}_{(C,T)}[\mathbf{f}, \mathbf{a}, \mathbf{y}] = \mathbb{D}_{(C,T)}[\mathbf{a}] + \mathbb{D}_{(C,T)}[\mathbf{f} | \mathbf{a}] + \mathbb{D}_{(C,T)}[\mathbf{y} | \mathbf{f}].$$

Starting with the definition in Def. B14 we get

$$\begin{aligned}
&\mathbb{D}_{(C,T)}[\mathbf{f}, \mathbf{a}, \mathbf{y}] \\
&= \mathbb{E}_{p(\mathbf{f}, \mathbf{a}, \mathbf{y})} \left[\log \frac{q_{C+T}(\mathbf{f}, \mathbf{a}, \mathbf{y})}{q_C(\mathbf{f}, \mathbf{a}, \mathbf{y})} \right] \\
&= \mathbb{E}_{p(\mathbf{y} | \mathbf{f}) p(\mathbf{f} | \mathbf{a}) p(\mathbf{a})} \left[\log \frac{q_{C+T}(\mathbf{y} | \mathbf{f}) q_{C+T}(\mathbf{f} | \mathbf{a}) q_{C+T}(\mathbf{a})}{q_C(\mathbf{y} | \mathbf{f}) q_C(\mathbf{f} | \mathbf{a}) q_C(\mathbf{a})} \right] \\
&= \int p(\mathbf{y} | \mathbf{f}) p(\mathbf{f} | \mathbf{a}) p(\mathbf{a}) \log \frac{q_{C+T}(\mathbf{y} | \mathbf{f}) q_{C+T}(\mathbf{f} | \mathbf{a}) q_{C+T}(\mathbf{a})}{q_C(\mathbf{y} | \mathbf{f}) q_C(\mathbf{f} | \mathbf{a}) q_C(\mathbf{a})} d\mathbf{a} d\mathbf{f} d\mathbf{y} \\
&= \int p(\mathbf{a}) \left(\int p(\mathbf{f} | \mathbf{a}) \left[\int p(\mathbf{y} | \mathbf{f}) \log \frac{q_{C+T}(\mathbf{y} | \mathbf{f})}{q_C(\mathbf{y} | \mathbf{f})} d\mathbf{y} \dots \right. \right. \\
&\quad \left. \left. + \log \frac{q_{C+T}(\mathbf{f} | \mathbf{a})}{q_C(\mathbf{f} | \mathbf{a})} \right] d\mathbf{f} + \log \frac{q_{C+T}(\mathbf{a})}{q_C(\mathbf{a})} \right) d\mathbf{a} \\
&= \int p(\mathbf{a}) \log \frac{q_{C+T}(\mathbf{a})}{q_C(\mathbf{a})} d\mathbf{a} + \int p(\mathbf{a}) \int p(\mathbf{f} | \mathbf{a}) \log \frac{q_{C+T}(\mathbf{f} | \mathbf{a})}{q_C(\mathbf{f} | \mathbf{a})} d\mathbf{f} \\
&\quad + \int p(\mathbf{f}) \int p(\mathbf{y} | \mathbf{f}) \log \frac{q_{C+T}(\mathbf{y} | \mathbf{f})}{q_C(\mathbf{y} | \mathbf{f})} d\mathbf{y} d\mathbf{f} \\
&= \mathbb{E}_{p(\mathbf{a})} \left[\log \frac{q_{C+T}(\mathbf{a})}{q_C(\mathbf{a})} \right] + \mathbb{E}_{p(\mathbf{a})} \left[\mathbb{E}_{p(\mathbf{f} | \mathbf{a})} \left[\log \frac{q_{C+T}(\mathbf{f} | \mathbf{a})}{q_C(\mathbf{f} | \mathbf{a})} \right] \right] \\
&\quad + \mathbb{E}_{p(\mathbf{f})} \left[\mathbb{E}_{p(\mathbf{y} | \mathbf{f})} \left[\log \frac{q_{C+T}(\mathbf{y} | \mathbf{f})}{q_C(\mathbf{y} | \mathbf{f})} \right] \right] \\
&= \mathbb{D}_{(C,T)}[\mathbf{a}] + \mathbb{D}_{(C,T)}[\mathbf{f} | \mathbf{a}] + \mathbb{D}_{(C,T)}[\mathbf{y} | \mathbf{f}],
\end{aligned}$$

where we used the definitions in Def. B14. We also immediately see that

$$\mathbb{D}_{(C,T)}[\mathbf{y} | \mathbf{f}] = 0$$

since $q_C(\mathbf{y} | \mathbf{f}) = q_{C+T}(\mathbf{y} | \mathbf{f}) = p(\mathbf{y} | \mathbf{f})$ is exact. The proofs for $\mathbb{D}_{(C,T)}[\mathbf{a}] \geq 0$ and $\mathbb{D}_{(C,T)}[\mathbf{f} | \mathbf{a}] \geq 0$ are given in Proof 7, 8 and 8, respectively.

Proof 7 (Proof of Subproof I of Proof 6) We prove

$$\mathbb{D}_{(C,T)}[\mathbf{a}] = \mathbb{E}_{p(\mathbf{a})} \left[\log \frac{q_{C+T}(\mathbf{a})}{q_C(\mathbf{a})} \right] \geq 0.$$

We abbreviate $q_1(\mathbf{a}) = q_C(\mathbf{a})$ and $q_2(\mathbf{a}) = q_{C+T}(\mathbf{a})$. The difference $\mathbb{D}_{(C,T)}[\mathbf{a}]$ is

$$\int p(\mathbf{a}) \log \frac{q_2(\mathbf{a})}{q_1(\mathbf{a})} d\mathbf{a} = \int p(\mathbf{a}) \log \frac{\prod_{j=1}^J p(\mathbf{a}_j | \mathbf{a}_{\pi_2(j)})}{\prod_{j=1}^J p(\mathbf{a}_j | \mathbf{a}_{\pi_1(j)})} d\mathbf{a}$$

$$= \int p(\mathbf{a}) \sum_{j=1}^J \log \frac{p(\mathbf{a}_j | \mathbf{a}_{\pi_2(j)})}{p(\mathbf{a}_j | \mathbf{a}_{\pi_1(j)})} d\mathbf{a} = \sum_{j=1}^J \int p(\mathbf{a}) \log \frac{p(\mathbf{a}_j | \mathbf{a}_{\pi_2(j)})}{p(\mathbf{a}_j | \mathbf{a}_{\pi_1(j)})} d\mathbf{a}$$

We recall property (iii) in Def. 2, thus we have $\pi_2(j) = \pi_1(j) \cup \phi(j)$ where $\phi(j)$ is the additional predecessor of expert j in the model 2 compared to model 1. In the following, we abbreviate $\pi_1(j) = \pi(j)$ yielding

$$\begin{aligned} & \sum_{j=1}^J \int p(\mathbf{a}) \log \frac{p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}, \mathbf{a}_{\phi(j)})}{p(\mathbf{a}_j | \mathbf{a}_{\pi(j)})} d\mathbf{a} \\ &= \sum_{j=1}^J \int p(\mathbf{a}) \log \frac{p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}, \mathbf{a}_{\phi(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)})}{p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)})} d\mathbf{a} \\ &= \sum_{j=1}^J \int p(\mathbf{a}) \log \frac{p(\mathbf{a}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)})}{p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)})} d\mathbf{a} \\ &= \sum_{j=1}^J \int p(\tilde{\mathbf{a}}_j) \log \frac{p(\mathbf{a}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)})}{p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)})} d\tilde{\mathbf{a}}_j \\ &= \sum_{j=1}^J I(\mathbf{a}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)}) \geq 0 \end{aligned}$$

where $\tilde{\mathbf{a}}_j = \mathbf{a}_j \cup \mathbf{a}_{\phi(j)} \cup \mathbf{a}_{\pi(j)}$ and $I(\mathbf{a}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\pi(j)})$ the conditional mutual information which is always positive [42, p. 30] and therefore concludes the first part of the proof.

Proof 8 (Subproof II of Proof 6) We prove

$$\mathbb{D}_{(C,T)}[\mathbf{f} | \mathbf{a}] = \mathbb{E}_{p(\mathbf{a})} \left[\mathbb{E}_{p(\mathbf{f} | \mathbf{a})} \left[\log \frac{q_{C+T}(\mathbf{f} | \mathbf{a})}{q_C(\mathbf{f} | \mathbf{a})} \right] \right] \geq 0.$$

We abbreviate $q_1(\mathbf{f} | \mathbf{a}) = q_C(\mathbf{f} | \mathbf{a})$ and $q_2(\mathbf{f} | \mathbf{a}) = q_{C+T}(\mathbf{f} | \mathbf{a})$. The difference $\mathbb{D}_{(C,T)}[\mathbf{f} | \mathbf{a}]$ is

$$\begin{aligned} & \int p(\mathbf{a}) \int p(\mathbf{f} | \mathbf{a}) \log \frac{q_2(\mathbf{f} | \mathbf{a})}{q_1(\mathbf{f} | \mathbf{a})} d\mathbf{f} d\mathbf{a} = \int p(\mathbf{a}, \mathbf{f}) \log \frac{\prod_{j=1}^J p(\mathbf{f}_j | \mathbf{a}_{\psi_2(j)})}{\prod_{j=1}^J p(\mathbf{f}_j | \mathbf{a}_{\psi_1(j)})} d\mathbf{f} d\mathbf{a} \\ &= \int p(\mathbf{a}, \mathbf{f}) \sum_{j=1}^J \log \frac{p(\mathbf{f}_j | \mathbf{a}_{\psi_2(j)})}{p(\mathbf{f}_j | \mathbf{a}_{\psi_1(j)})} d\mathbf{f} d\mathbf{a} = \sum_{j=1}^J \int p(\mathbf{a}, \mathbf{f}) \log \frac{p(\mathbf{f}_j | \mathbf{a}_{\psi_2(j)})}{p(\mathbf{f}_j | \mathbf{a}_{\psi_1(j)})} d\mathbf{f} d\mathbf{a} \end{aligned}$$

We recall the definition of $\psi_C(j)$ in Def. 2, where we have $\psi_C(j) = \pi_C(j) \cup \{j, \dots, C\}$ if $j < C$ and $\psi_C(j) = \pi_C(j) \cup j$ otherwise. Further, we have $\pi_2(j) = \pi_1(j) \cup \phi(j)$ where $\phi(j)$ is the additional predecessor of expert j in the model 2 compared to model 1. Therefore, we have $\psi_2(j) = \psi_1(j) \cup \phi(j)$ for all j .

[Proof: If $j < C$, we have $\pi_1(j) = \pi_2(j)$ since $\phi(j)$ empty. Therefore, we have $\psi_1(j) = \pi_1(j) \cup \{j, \dots, C\} = \pi_2(j) \cup \{j, \dots, C\} = \psi_2(j)$ for all $j = 1, \dots, C-1$. If $j \geq C$, we have $\psi_1(j) = \pi_1(j) \cup j$ and $\psi_2(j) = \pi_2(j) \cup j = \pi_1(j) \cup \phi(j) \cup j = \psi_1(j) \cup \phi(j)$ for all $j = C, \dots, J$.]

We abbreviate $\boldsymbol{\psi}_1(j) = \boldsymbol{\psi}(j)$ and substitute $\boldsymbol{\psi}_2(j) = \boldsymbol{\psi}(j) \cup \phi(j)$ yielding

$$\begin{aligned}
& \sum_{j=1}^J \int p(\mathbf{a}, \mathbf{f}) \log \frac{p(\mathbf{f}_j | \mathbf{a}_{\boldsymbol{\psi}(j)}, \mathbf{a}_{\phi(j)})}{p(\mathbf{f}_j | \mathbf{a}_{\boldsymbol{\psi}(j)})} d\mathbf{f} d\mathbf{a} \\
&= \sum_{j=1}^J \int p(\mathbf{a}, \mathbf{f}) \log \frac{p(\mathbf{f}_j | \mathbf{a}_{\boldsymbol{\psi}(j)}, \mathbf{a}_{\phi(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\boldsymbol{\psi}(j)})}{p(\mathbf{f}_j | \mathbf{a}_{\boldsymbol{\psi}(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\boldsymbol{\psi}(j)})} d\mathbf{a} \\
&= \sum_{j=1}^J \int p(\mathbf{a}, \mathbf{f}) \log \frac{p(\mathbf{f}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\boldsymbol{\psi}(j)})}{p(\mathbf{f}_j | \mathbf{a}_{\boldsymbol{\psi}(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\boldsymbol{\psi}(j)})} d\mathbf{a} \\
&= \sum_{j=1}^J \int p(\tilde{\mathbf{a}}_j, \mathbf{f}_j) \log \frac{p(\mathbf{f}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\boldsymbol{\psi}(j)})}{p(\mathbf{f}_j | \mathbf{a}_{\boldsymbol{\psi}(j)}) p(\mathbf{a}_{\phi(j)} | \mathbf{a}_{\boldsymbol{\psi}(j)})} d\tilde{\mathbf{a}}_j \\
&= \sum_{j=1}^J I(\mathbf{f}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\boldsymbol{\psi}(j)}) \geq 0
\end{aligned}$$

where $\tilde{\mathbf{a}}_j = \mathbf{a}_{\phi(j)} \cup \mathbf{a}_{\boldsymbol{\psi}(j)}$ and $I(\mathbf{f}_j, \mathbf{a}_{\phi(j)} | \mathbf{a}_{\boldsymbol{\psi}(j)})$ the conditional mutual information which is always positive [42, p. 30] and therefore concludes the first part of the proof.

Moreover, the difference in the joint prior is

$$\begin{aligned}
\mathbb{D}_{(C,T)}[\mathbf{f}, \mathbf{a}, \mathbf{y}] &= \mathbb{D}_{(C,T)}[\mathbf{f}, \mathbf{a}] = \mathbb{D}_{(C,T)}[\mathbf{a}] + \mathbb{D}_{(C,T)}[\mathbf{f} | \mathbf{a}] \\
&= \frac{1}{2} \log \frac{|\bar{\mathbf{V}}_C| |\mathbf{S}_C^{-1}|}{|\bar{\mathbf{V}}_{C+T}| |\mathbf{S}_{C+T}^{-1}|} = \frac{1}{2} \log \frac{|\bar{\mathbf{V}}_C| |\mathbf{Q}_{C+T}|}{|\bar{\mathbf{V}}_{C+T}| |\mathbf{Q}_C|} \geq 0.
\end{aligned}$$

Proof 9 (Subproof III of Proof 6; Prior KL) For the second part, we use (B13) where the difference in KL of 2 Gaussians with zero mean and same base distribution is formulated. In our case we have

$$\frac{1}{2} \left(\text{tr}((\mathbf{S}_C - \mathbf{S}_{C+T}) \mathbf{K}_{AA}) + \log \frac{|\mathbf{S}_{C+T}|}{|\mathbf{S}_C|} \right)$$

where the trace is 0 by Prop. 12 and thus $\mathbb{D}_{(C,T)} = \frac{1}{2} \log \frac{|\mathbf{S}_{C+T}|}{|\mathbf{S}_C|}$. Since $\mathbf{S}_C = \mathbf{F}^T \mathbf{Q}^{-1} \mathbf{F}$ and $|\mathbf{F}| = 1$, we have $\mathbb{D}_{(C,T)} = \frac{1}{2} \log \frac{|\mathbf{Q}_C|}{|\mathbf{Q}_{C+T}|}$ which concludes the proof.

Proof 10 (Proof of Prop. 14; Decreasing Prior Entropy) For the third part of the statement, the entropy H of $q_C(\mathbf{a})$ is

$$H[q_C(\mathbf{a})] = \frac{1}{2} (-\log |\mathbf{S}_C| + JL(1 + \log 2\pi)) = \frac{1}{2} (\log |\mathbf{Q}_C| + JL(1 + \log 2\pi))$$

where we used Eq. (B10) and $|\mathbf{F}| = 1$. The second part follows from Prop. 5. Using Proof 9 which states

$$\mathbb{D}_{(C,T)} = \frac{1}{2} \log \frac{|\mathbf{Q}_C|}{|\mathbf{Q}_{C+T}|} \geq 0,$$

it follows

$$\log |\mathbf{Q}_C| \geq \log |\mathbf{Q}_{C+T}|$$

for any $T \in \{1, \dots, C-1\}$ and therefore

$$H[q_{C+T}(\mathbf{a})] \leq H[q_C(\mathbf{a})]$$

which concludes the proof.

Proof 11 ((Sub)Proof of Prop. 3; Marginalized Joint Distribution) From the joint distribution in Def. 4 over all variables, the latent function values \mathbf{f} can be integrated out resulting in

$$q(\mathbf{a}, \mathbf{y}) = \int q(\mathbf{f}, \mathbf{a}, \mathbf{y}) d\mathbf{f} = \int p(\mathbf{y}|\mathbf{f}) q(\mathbf{f}|\mathbf{a}) d\mathbf{f} q(\mathbf{a}),$$

where the integral can be computed via (B7) yielding

$$q(\mathbf{a}, \mathbf{y}) = \int p(\mathbf{y}|\mathbf{f}) q(\mathbf{f}|\mathbf{a}) d\mathbf{f} = \int \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma_n^2 \mathbb{I}) \mathcal{N}(\mathbf{f}|\mathbf{H}\mathbf{f}, \bar{\mathbf{V}}) d\mathbf{f} = \mathcal{N}(\mathbf{y}|\mathbf{H}\mathbf{a}, \mathbf{V})$$

with $\mathbf{V} = \bar{\mathbf{V}} + \sigma_n^2 \mathbb{I}$ and thus

$$q(\mathbf{a}, \mathbf{y}) = q(\mathbf{y}|\mathbf{a}) q(\mathbf{a}) = \mathcal{N}(\mathbf{y}|\mathbf{H}\mathbf{a}, \mathbf{V}) \mathcal{N}(\mathbf{a}|\mathbf{0}, \mathbf{S}^{-1})$$

which concludes the proof.

Proof 12 (Proof of Prop.3; Posterior Approximation) The posterior approximation is

$$q(\mathbf{a}|\mathbf{y}) = \frac{q(\mathbf{a}, \mathbf{y})}{q(\mathbf{y})} \propto q(\mathbf{a}, \mathbf{y}) = q(\mathbf{y}|\mathbf{a}) q_C(\mathbf{a})$$

where the first equality comes from the definition of conditional probabilities, the proportionality because the marginal likelihood $q(\mathbf{y})$ is independent of \mathbf{a} and the last equality exploits Proof 11. Since

$$q(\mathbf{y}|\mathbf{a}) q_C(\mathbf{a}) = \mathcal{N}(\mathbf{y}|\mathbf{H}\mathbf{a}, \mathbf{V}) \mathcal{N}(\mathbf{a}|\mathbf{0}, \mathbf{S}^{-1}),$$

the desired posterior distribution can be analytically computed via (B8) yielding

$$q(\mathbf{a}|\mathbf{y}) = \mathcal{N}(\mathbf{a}|\boldsymbol{\mu}, \boldsymbol{\Sigma}),$$

with $\boldsymbol{\Sigma} = (\mathbf{H}^T \mathbf{V}^{-1} \mathbf{H} + \mathbf{S})^{-1}$, $\boldsymbol{\mu} = \boldsymbol{\Sigma} \boldsymbol{\eta}$ and $\boldsymbol{\eta} = \mathbf{H}^T \mathbf{V}^{-1} \mathbf{y}$.

Proof 13 (Proof of Prop. 9; Marginal Likelihood) The marginal likelihood $q(\mathbf{y})$ is obtained by integrating (B7) over the joint distribution $q(\mathbf{y}, \mathbf{a})$ in Prop. 2 leading to

$$q(\mathbf{y}) = \int q(\mathbf{y}, \mathbf{a}) d\mathbf{a} = \int q(\mathbf{y}|\mathbf{a}) q(\mathbf{a}) d\mathbf{a} = \int \mathcal{N}(\mathbf{H}\mathbf{a}, \mathbf{V}) \mathcal{N}(\mathbf{0}, \mathbf{S}^{-1}) d\mathbf{a} = \mathcal{N}(\mathbf{0}, \mathbf{P})$$

where $\mathbf{P} = \mathbf{H}\mathbf{S}^{-1}\mathbf{H}^T + \mathbf{V}$.

Proof 14 (Proof of Prop. 12; Exact Diagonal of Prior) Using Prop. 11, the trace can be written as

$$\begin{aligned} \text{tr}(\mathbf{S}\mathbf{K}_{\mathbf{AA}}) &= \text{tr} \left(\left(\sum_{j=1}^J \overline{\mathbf{K}}_{\mathbf{A}_{\pi^+(j)} \mathbf{A}_{\pi^+(j)}}^{-1} - \overline{\mathbf{K}}_{\mathbf{A}_{\pi(j)} \mathbf{A}_{\pi(j)}}^{-1} \right) \mathbf{K}_{\mathbf{AA}} \right) \\ &= \sum_{j=1}^J \text{tr} \left(\overline{\mathbf{K}}_{\mathbf{A}_{\pi^+(j)} \mathbf{A}_{\pi^+(j)}}^{-1} \mathbf{K}_{\mathbf{AA}} \right) - \text{tr} \left(\overline{\mathbf{K}}_{\mathbf{A}_{\pi(j)} \mathbf{A}_{\pi(j)}}^{-1} \mathbf{K}_{\mathbf{AA}} \right). \end{aligned}$$

By construction of the matrices $\overline{\mathbf{K}}_{\mathbf{A}_\phi, \mathbf{A}_\phi}^{-1}$, they contain the matrix $\mathbf{K}_{\mathbf{A}_\phi, \mathbf{A}_\phi}^{-1}$ at the entries $[\phi, \phi]$. Therefore, the resulting product when multiplying with $\mathbf{K}_{\mathbf{AA}}$ is a matrix with identity \mathbb{I}_T at the position $[\phi, \phi]$ with $T = |\phi|$ and 0 at the diagonal where not ϕ . The quantity above is then

$$\sum_{j=1}^J L|\pi^+(j)| - L|\pi(j)| = \sum_{j=1}^J L(\min(j, C) - \min(j-1, C-1)) = JL.$$

Proof 15 (Proof of Prop. 10; Prior Approximation II) The prior approximation is

$$q(\mathbf{a}) = \prod_{j=1}^J p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) = \prod_{j=1}^J \mathcal{N}(\mathbf{a}_j | \mathbf{F}_j \mathbf{a}_{\pi(j)}, \mathbf{Q}_j)$$

for which the quadratic term inside the exponential of the individual Gaussian can be written as

$$\begin{aligned} & -\frac{1}{2}(\mathbf{a}_j - \mathbf{F}_j \mathbf{a}_{\pi(j)})^T \mathbf{Q}_j^{-1} (\mathbf{a}_j - \mathbf{F}_j \mathbf{a}_{\pi(j)}) \\ & -\frac{1}{2} \begin{bmatrix} \mathbf{a}_{\pi(j)}^T & \mathbf{a}_j^T \end{bmatrix} \begin{bmatrix} -\mathbf{F}_j^T \\ \mathbb{I} \end{bmatrix} \mathbf{Q}_j^{-1} \begin{bmatrix} -\mathbf{F}_j \\ \mathbb{I} \end{bmatrix} \begin{bmatrix} \mathbf{a}_{\pi(j)} \\ \mathbf{a}_j \end{bmatrix} \end{aligned}$$

which correspond to a Gaussian

$$\mathcal{N}(\mathbf{a}_{\pi^+(j)} | \mathbf{0}, \mathbf{S}_{(j)}^{-1})$$

with $\mathbf{S}_{(j)} = \tilde{\mathbf{F}}_j^T \mathbf{Q}_j^{-1} \tilde{\mathbf{F}}_j \in \mathbb{R}^{LC \times LC}$ and $\tilde{\mathbf{F}}_j = [-\mathbf{F}_j \ \mathbb{I}] \in \mathbb{R}^{L \times LC}$ which proves the first part. We can augment this Gaussian for $\mathbf{a}_{\pi^+(j)} \in \mathbb{R}^{LC}$ to

$$-\frac{1}{2} \mathbf{a}^T \tilde{\mathbf{S}}_{(j)}^{-1} \mathbf{a} \propto \mathcal{N}(\mathbf{a} | \mathbf{0}, \tilde{\mathbf{S}}_{(j)}^{-1})$$

over $\mathbf{a} \in \mathbb{R}^M$ where $\tilde{\mathbf{S}}_{(j)} \in \mathbb{R}^{M \times M}$ is the augmented matrix consisting of $\mathbf{S}_{(j)}$ at the entries $[\pi^+(j), \pi^+(j)]$ and 0 otherwise. Using (B9), the original product $q(\mathbf{a})$ is then

$$\prod_{j=1}^J \mathcal{N}(\mathbf{a} | \mathbf{0}, \tilde{\mathbf{S}}_{(j)}^{-1}) = \mathcal{N} \left(\mathbf{a} | \mathbf{0}, \left(\sum_{j=1}^J \tilde{\mathbf{S}}_{(j)} \right)^{-1} \right)$$

and thus $\mathbf{S} = \sum_{j=1}^J \tilde{\mathbf{S}}_{(j)}$ positive definite which concludes the proof.

Proof 16 (Proof of Prop. 11; Prior Approximation III) The prior $q(\mathbf{a})$ can be written as

$$q(\mathbf{a}) = \prod_{j=1}^J p(\mathbf{a}_j | \mathbf{a}_{\pi(j)}) = \prod_{j=1}^J \frac{p(\mathbf{a}_j, \mathbf{a}_{\pi(j)})}{p(\mathbf{a}_{\pi(j)})} = \prod_{j=1}^J \frac{p(\mathbf{a}_{\pi^+(j)})}{p(\mathbf{a}_{\pi(j)})}$$

$$= \prod_{j=1}^J \frac{\mathcal{N}(\mathbf{a}_{\pi+(j)} | \mathbf{0}, \mathbf{K}_{\mathbf{A}_{\pi+(j)} \mathbf{A}_{\pi+(j)}})}{\mathcal{N}(\mathbf{a}_{\pi(j)} | \mathbf{0}, \mathbf{K}_{\mathbf{A}_{\pi(j)} \mathbf{A}_{\pi(j)}})}$$

Similarly to the Proof 15, we can augment the CL -dimensional and the $(C-1)L$ -dimensional Gaussian in the nominator and denominator, respectively, to M -dimensional Gaussians with covariance $\overline{\mathbf{K}}_{\mathbf{A}_\phi \mathbf{A}_\phi}^{-1}$ consisting of $\mathbf{K}_{\mathbf{A}_\phi \mathbf{A}_\phi}^{-1}$ at the entries $[\phi, \phi]$ and 0 otherwise. This gives with (B9)

$$\begin{aligned} & \prod_{j=1}^J \frac{\mathcal{N}(\mathbf{a} | \mathbf{0}, \overline{\mathbf{K}}_{\mathbf{A}_{\pi+(j)} \mathbf{A}_{\pi+(j)}})}{\mathcal{N}(\mathbf{a} | \mathbf{0}, \overline{\mathbf{K}}_{\mathbf{A}_{\pi(j)} \mathbf{A}_{\pi(j)}})} \\ &= \mathcal{N} \left(\mathbf{a} | \mathbf{0}, \left(\sum_{j=1}^J \overline{\mathbf{K}}_{\mathbf{A}_{\pi+(j)} \mathbf{A}_{\pi+(j)}}^{-1} - \overline{\mathbf{K}}_{\mathbf{A}_{\pi(j)} \mathbf{A}_{\pi(j)}}^{-1} \right)^{-1} \right), \end{aligned}$$

which concludes the proof with $\mathbf{S} = \sum_{j=1}^J \overline{\mathbf{K}}_{\mathbf{A}_{\pi+(j)} \mathbf{A}_{\pi+(j)}}^{-1} - \overline{\mathbf{K}}_{\mathbf{A}_{\pi(j)} \mathbf{A}_{\pi(j)}}^{-1}$ which is positive definite.

Proof 17 (Proof of Prop. 4; Prediction Aggregation) The predictive posterior distribution is defined as

$$p(f_* | \mathbf{y}) = \prod_{j=C}^J p(f_{*j} | \mathbf{y})^{\beta_{*j}}.$$

Since the local predictions $p(f_{*j} | \mathbf{y}) = \mathcal{N}(m_{*j}, v_{*j})$ are all univariate Gaussians, we obtain via the product rule of Gaussians in (B9) directly

$$m_* = v_{*j} \sum_{j=C}^J \beta_{*j} \frac{m_{*j}}{v_{*j}} \quad \text{and} \quad \frac{1}{v_*} = \sum_{j=C}^J \frac{\beta_{*j}}{v_{*j}}.$$

Using the usual likelihood $p(y_* | f_*) = \mathcal{N}(f_*, \sigma_n^2)$ yields with (B7) the final noisy prediction $p(y_* | \mathbf{y}) = \int p(y_* | f_*) p(f_* | \mathbf{y}) df_* = \mathcal{N}(m_*, v_* + \sigma_n^2)$. Note that, in Definition 5 of the weights, we introduced a scaling factor Z . This Z in the exponent of the normalization of the weights has a sharpening effect, so that the informative experts have even more weight compared to the non-informative experts for more data N and more correlations C . This is a heuristic but showed quite robust performance in experiments. Moreover, the consistency properties are more relevant than the particular weights.

Proof 18 (Proof of Prop. 8; Local Predictions) The predictive conditional $p(f_{*j} | \mathbf{a}_{\psi(j)})$ can be again derived via (B6) from the assumed joint

$$p(f_{*j}, \mathbf{a}_{\psi(j)}) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{[\mathbf{x}_*, \mathbf{A}_{\psi(j)}][\mathbf{x}_*, \mathbf{A}_{\psi(j)}]})$$

leading to $\mathcal{N}(\mathbf{h}_* \mathbf{a}_{\psi(j)}, v_*)$ with

$$\mathbf{h}_* = \mathbf{K}_{\mathbf{x}_* \mathbf{A}_{\psi(j)}} \mathbf{K}_{\mathbf{A}_{\psi(j)} \mathbf{A}_{\psi(j)}}^{-1}$$

and

$$v_* = \mathbf{K} \mathbf{x}_* \mathbf{x}_* - \mathbf{K} \mathbf{x}_* \mathbf{A}_{\psi(j)} \mathbf{K}_{\mathbf{A}_{\psi(j)} \mathbf{A}_{\psi(j)}}^{-1} \mathbf{K} \mathbf{A}_{\psi(j)} \mathbf{x}_*.$$

Moreover, the local posteriors $q(\mathbf{a}_{\psi(j)} | \mathbf{y}) = \mathcal{N}(\boldsymbol{\mu}_{\psi(j)}, \boldsymbol{\Sigma}_{\psi(j)})$ are obtained from the corresponding entries $\psi(j)$ of the mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ (via partial inversion A.2) in Prop. 3. Finally, the local predictions $p(f_{*j} | \mathbf{y})$ in Prop. (8) can then be computed with Gaussian integration (B7) yielding

$$q(f_{*j} | \mathbf{y}) = \int p(f_{*j} | \mathbf{a}_{\psi(j)}) p(\mathbf{a}_{\psi(j)} | \mathbf{y}) d\mathbf{a}_{\psi(j)}$$

which correspond to the desired quantities

$$\mathcal{N}(m_{*j}, v_{*j}) = \mathcal{N}(\mathbf{h}_* \boldsymbol{\mu}_{\psi(j)}, \mathbf{h}_*^T \boldsymbol{\Sigma}_{\psi(j)} \mathbf{h}_* + v_*).$$

Proof 19 (Proof for Figure C7; Joint Prior Covariance) For the joint prior

$$q_C(\mathbf{a}, \mathbf{f}, \mathbf{y}) = \mathcal{N}([\mathbf{a}; \mathbf{f}; \mathbf{y}] | \mathbf{0}, \mathbf{W}_\gamma)$$

with covariance

$$\mathbf{W}_\gamma = \begin{bmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{af} & \boldsymbol{\Sigma}_{ay} \\ \boldsymbol{\Sigma}_{fa} & \boldsymbol{\Sigma}_{ff} & \boldsymbol{\Sigma}_{fy} \\ \boldsymbol{\Sigma}_{ya} & \boldsymbol{\Sigma}_{yf} & \boldsymbol{\Sigma}_{yy} \end{bmatrix}$$

corresponding to Fig. C7, we show that we recover the marginal and conditional distributions $q_C(\mathbf{a})$, $q_C(\mathbf{f} | \mathbf{a})$ and $p(\mathbf{y} | \mathbf{f})$. For $q_C(\mathbf{a})$, the marginalization correspond to selecting the corresponding mean and covariance, i.e. $\mathcal{N}(\mathbf{a} | \mathbf{0}, \boldsymbol{\Sigma}_{aa}) = \mathcal{N}(\mathbf{a} | \mathbf{0}, \mathbf{S}^{-1})$. For $q_C(\mathbf{f} | \mathbf{a})$, we use Eq. (B6) yielding

$$\begin{aligned} & \mathcal{N}(\mathbf{f} | \boldsymbol{\Sigma}_{fa} \boldsymbol{\Sigma}_{aa}^{-1} \mathbf{a}, \boldsymbol{\Sigma}_{ff} - \boldsymbol{\Sigma}_{fa} \boldsymbol{\Sigma}_{aa}^{-1} \boldsymbol{\Sigma}_{af}) \\ &= \mathcal{N}(\mathbf{f} | \mathbf{H} \mathbf{a}, (\mathbf{H} \mathbf{S}^{-1} \mathbf{H}^T + \bar{\mathbf{V}}) - \mathbf{H} (\mathbf{S}^{-1} \mathbf{H}^T)) \\ &= \mathcal{N}(\mathbf{f} | \mathbf{H} \mathbf{a}, \bar{\mathbf{V}}) \end{aligned}$$

since $\boldsymbol{\Sigma}_{fa} \boldsymbol{\Sigma}_{aa}^{-1} = (\mathbf{H} \mathbf{S}^{-1}) \mathbf{S} = \mathbf{H}$. Similarly for $p(\mathbf{y} | \mathbf{f})$, with Eq. (B6) we get

$$\begin{aligned} & \mathcal{N}(\mathbf{y} | \boldsymbol{\Sigma}_{yf} \boldsymbol{\Sigma}_{ff}^{-1} \mathbf{f}, \boldsymbol{\Sigma}_{yy} - \boldsymbol{\Sigma}_{yf} \boldsymbol{\Sigma}_{ff}^{-1} \boldsymbol{\Sigma}_{fy}) \\ &= \mathcal{N}(\mathbf{y} | \mathbb{I} \mathbf{f}, (\mathbf{H} \mathbf{S}^{-1} \mathbf{H}^T + \mathbf{V}) - \mathbb{I} (\mathbf{H} \mathbf{S}^{-1} \mathbf{H}^T + \bar{\mathbf{V}})) \\ &= \mathcal{N}(\mathbf{y} | \mathbf{f}, \sigma_n^2 \mathbb{I}) \end{aligned}$$

since $\boldsymbol{\Sigma}_{yf} \boldsymbol{\Sigma}_{ff}^{-1} = \mathbb{I}$.

C.3 Derivative of LML

The log marginal likelihood in Section 3.6.2 in Eq. (A1) is proportional to

$$-\frac{1}{2} \mathbf{y}^T \mathbf{V}^{-1} \mathbf{y} + \frac{1}{2} \boldsymbol{\mu}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} - \frac{1}{2} \log |\boldsymbol{\Sigma}^{-1}| - \frac{1}{2} \log |\mathbf{V}| - \frac{1}{2} \log |\mathbf{Q}|.$$

In the following, we provide the partial derivative with respect to θ for each additive term.

$$\frac{\partial}{\partial \theta} \left[-\frac{1}{2} \mathbf{y}^T \mathbf{V}^{-1} \mathbf{y} \right] = \frac{1}{2} \mathbf{y}^T \mathbf{V}^{-1} \frac{\partial \mathbf{V}}{\partial \theta} \mathbf{V}^{-1} \mathbf{y}$$

$$\frac{\partial}{\partial \theta} \left[\frac{1}{2} \boldsymbol{\mu}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} \right] = \frac{\partial \boldsymbol{\eta}^T}{\partial \theta} \boldsymbol{\mu} - \frac{1}{2} \boldsymbol{\mu}^T \frac{\partial \boldsymbol{\Sigma}^{-1}}{\partial \theta} \boldsymbol{\mu}$$

$$\frac{\partial}{\partial \theta} \left[-\frac{1}{2} \log |\boldsymbol{\Sigma}^{-1}| \right] = -\frac{1}{2} \text{tr} \left\{ \boldsymbol{\Sigma} \frac{\partial \boldsymbol{\Sigma}^{-1}}{\partial \theta} \right\}$$

In the last expression the whole posterior covariance is needed, however, it turns out that only the entries which are non-zero in the precision are needed. The right term in the last expression equals $\text{sum} \left\{ \boldsymbol{\Sigma} \odot \frac{\partial \boldsymbol{\Sigma}^{-1}}{\partial \theta} \right\}$, where \odot denotes the pointwise multiplication. Therefore it is enough to only compute $\text{sum} \left\{ \bar{\boldsymbol{\Sigma}} \odot \frac{\partial \boldsymbol{\Sigma}^{-1}}{\partial \theta} \right\}$, where $\bar{\boldsymbol{\Sigma}}$ is the partial inversion (for more details A.2) which is sparse as well and already computed for the local predictions in Prop. 8.

$$\frac{\partial}{\partial \theta} \left[-\frac{1}{2} \log |\mathbf{V}| \right] = -\frac{1}{2} \text{sum} \left\{ \mathbf{V}^{-1} \odot \frac{\partial \mathbf{V}}{\partial \theta} \right\}$$

$$\frac{\partial}{\partial \theta} \left[-\frac{1}{2} \log |\mathbf{Q}| \right] = -\frac{1}{2} \text{sum} \left\{ \mathbf{Q}^{-1} \odot \frac{\partial \mathbf{Q}}{\partial \theta} \right\}$$

The derivatives $\frac{\partial \boldsymbol{\Sigma}^{-1}}{\partial \theta}$, $\frac{\partial \mathbf{V}}{\partial \theta}$ and $\frac{\partial \mathbf{Q}}{\partial \theta}$ can be computed via chain rule of derivatives.

Appendix D Sequential Algorithm

The probabilistic equations in Section 3 can be equivalently formulated as

$$\mathbf{a}_j = \mathbf{F}_j \mathbf{a}_{\pi(j)} + \boldsymbol{\gamma}_j;$$

$$\mathbf{f}_j = \mathbf{H}_j \mathbf{a}_{\psi(j)} + \boldsymbol{\nu}_j;$$

$$\mathbf{y}_j = \mathbf{f}_j + \boldsymbol{\varepsilon}_j,$$

with $\boldsymbol{\gamma}_j \sim \mathcal{N}(0, \mathbf{Q}_j)$, $\boldsymbol{\nu}_j \sim \mathcal{N}(0, \bar{\mathbf{V}}_j)$ and $\boldsymbol{\varepsilon}_j \sim \mathcal{N}(0, \sigma_n^2 \mathbb{I})$. Instead to the inference procedure described in Prop. 3, the posterior could be alternatively computed with sequential algorithms. Assuming $C = 2$ and $\boldsymbol{\pi}(j) = \{j-1\}$, the Kalman Filter and Smoother (e.g. [43]) provide an equivalent solution to

the posterior distribution in Prop. 3. For $C > 2$ and general neighbourhood set, the Gaussian (loopy) belief propagation algorithm or Gaussian expectation propagation (e.g. [43]) might constitute an interesting approach for sequential/online and distributed learning procedures. Together with the competitive results in the application to time series with covariates makes this idea very promising for future work.

Appendix E More Details about GPR

In this section we provide more details for Section 2.

Suppose we are given a training set $\mathcal{D} = \{y_i, \mathbf{x}_i\}_{i=1}^N$ of N pairs of inputs $\mathbf{x}_i \in \mathbb{R}^D$ and noisy scalar outputs y_i generated by adding independent Gaussian noise to a latent function $f(\mathbf{x})$, that is $y_i = f(\mathbf{x}_i) + \varepsilon_i$, where $\varepsilon_i \sim \mathcal{N}(0, \sigma_n^2)$. We denote $\mathbf{y} = [y_1, \dots, y_N]^T$ the vector of observations and with $\mathbf{X} = [\mathbf{x}_1^T, \dots, \mathbf{x}_N^T]^T \in \mathbb{R}^{N \times D}$.

We can model f with a *Gaussian Process* (GP), which defines a prior over functions and can be converted into a posterior over functions once we have observed some data (consider e.g. [1]). To describe a GP, we only need to specify a mean $m(\mathbf{x})$ and a covariance function $k_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{x}')$ where $\boldsymbol{\theta}$ is a set of a few hyperparameters. Thereby, $k_{\boldsymbol{\theta}}$ is a positive definite kernel function (see [1]), for instance the *squared exponential* (SE) kernel with individual lengthscales for each dimension, that is $k_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{x}') = \sigma_0^2 \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^T \mathbf{L}^{-1}(\mathbf{x} - \mathbf{x}')\right)$ with $\mathbf{L} = \text{Diag}[l_1^2, \dots, l_D^2]$ and $\{\sigma_0, l_1, \dots, l_D\} \in \boldsymbol{\theta}$. For the sake of simplicity, we assume $m(\mathbf{x}) \equiv 0$, however it could be any function. Given the training values $\mathbf{f} = f(\mathbf{X}) = [f(\mathbf{x}_1), \dots, f(\mathbf{x}_N)]^T$ and a test latent function value $f_* = f(\mathbf{x}_*)$ at a test point $\mathbf{x}_* \in \mathbb{R}^D$, then the joint distribution $p(\mathbf{f}, f_*)$ is Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{K}_{[\mathbf{X}; \mathbf{x}_*][\mathbf{X}; \mathbf{x}_*]})$. Thereby, we use the notation $[\mathbf{A}_1; \mathbf{A}_2]$ for the resulting matrix after stacking $\mathbf{A}_1 \in \mathbb{R}^{N_1 \times D}$ and $\mathbf{A}_2 \in \mathbb{R}^{N_2 \times D}$ above each other and $\mathbf{K} \in \mathbb{R}^{M_1 \times M_2}$ denotes the kernel covariance matrix with entries $[\mathbf{K}_{\mathbf{AB}}]_{ij}$ corresponding to the kernel evaluation $k_{\boldsymbol{\theta}}(\mathbf{a}_i, \mathbf{b}_j)$ with the corresponding rows $\mathbf{a}_i, \mathbf{b}_j$ for any $\mathbf{A} \in \mathbb{R}^{M_1 \times D}$ and $\mathbf{B} \in \mathbb{R}^{M_2 \times D}$.

Typically, in GP regression, the likelihood is Gaussian, that is, $p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma_n^2 \mathbb{I})$, and with Bayes theorem (B8) we obtain analytically the predictive posterior distribution $p(f_*|\mathbf{y}) = \mathcal{N}(f_*|\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$ with $\boldsymbol{\mu}_* = \mathbf{K}_{\mathbf{x}_* \mathbf{X}} (\mathbf{K}_{\mathbf{X} \mathbf{X}} + \sigma_n^2 \mathbb{I})^{-1} \mathbf{y}$ and $\boldsymbol{\Sigma}_* = \mathbf{K}_{\mathbf{x}_* \mathbf{x}_*} - \mathbf{K}_{\mathbf{x}_* \mathbf{X}} (\mathbf{K}_{\mathbf{X} \mathbf{X}} + \sigma_n^2 \mathbb{I})^{-1} \mathbf{K}_{\mathbf{X} \mathbf{x}_*}$. Alternatively to the standard derivation shown above, the posterior distribution over the latent variables \mathbf{f} given the data \mathbf{y} can be explicitly formulated as

$$p(\mathbf{f}|\mathbf{y}) \propto p(\mathbf{f}, \mathbf{y}) = p(\mathbf{y}|\mathbf{f}) p(\mathbf{f}) = \prod_{j=1}^J p(\mathbf{y}_j|\mathbf{f}_j) p(\mathbf{f}_j|\mathbf{f}_{1:j-1}), \quad (\text{E16})$$

where the data is split into J mini-batches of size B , i.e. $\mathcal{D} = \{\mathbf{y}_j, \mathbf{X}_j\}_{j=1}^J$ with inputs $\mathbf{X}_j \in \mathbb{R}^{B \times D}$, outputs $\mathbf{y}_j \in \mathbb{R}^B$ and the corresponding latent function values $\mathbf{f}_j = f(\mathbf{X}_j) \in \mathbb{R}^B$. In (1) we used the notation $\mathbf{f}_{k:j}$ indicating $[\mathbf{f}_k, \dots, \mathbf{f}_j]$ and the conditionals $p(\mathbf{f}_j | \mathbf{f}_{1:j-1})$ can be derived from the joint Gaussian $p(\mathbf{f}_j, \mathbf{f}_{1:j-1}) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{[\mathbf{x}_j; \mathbf{x}_{1:j-1}][\mathbf{x}_j; \mathbf{x}_{1:j-1}]})$ via Gaussian conditioning (B6). The corresponding graphical model of (1) is depicted in Figure 1(a)i). Given the posterior over $\mathbf{f} | \mathbf{y}$, the predictive posterior distribution from above is equivalently obtained as $p(f_* | \mathbf{y}) = \int p(f_* | \mathbf{f}) p(\mathbf{f} | \mathbf{y}) d\mathbf{f}$ via Gaussian integration (B7) where $p(f_* | \mathbf{f})$ is derivable from the joint via (B6). The graphical model of the prediction procedure is depicted in Figure 1(b)i). We present this alternative two stage procedure to highlight later connections to our model with full GP.

E.1 Global Sparse GPs

Sparse GP regression approximations based on *global inducing points* reduce the computational complexity by introducing $M \ll N$ inducing points $\mathbf{a} \in \mathbb{R}^M$ that optimally summarize the dependency of the whole training data globally, compare the graphical model in Figure 1b). Thereby the inducing *inputs* $\mathbf{A} \in \mathbb{R}^{M \times D}$ are in the D -dimensional input data space and the inducing *outputs* $\mathbf{a} = f(\mathbf{A}) \in \mathbb{R}^M$ are the corresponding GP-function values. In the following, this model is denoted by SGP(M). Similarly to full GP in Eq. (1), the posterior over the inducing points $p(\mathbf{a} | \mathbf{y}) \propto \int p(\mathbf{a}, \mathbf{f}, \mathbf{y}) d\mathbf{f}$ can be derived from the joint distribution

$$p(\mathbf{a}, \mathbf{f}, \mathbf{y}) = p(\mathbf{y} | \mathbf{f}) p(\mathbf{f} | \mathbf{a}) p(\mathbf{a}) = \prod_{j=1}^J p(\mathbf{y}_j | \mathbf{f}_j) p(\mathbf{f}_j | \mathbf{a}) p(\mathbf{a}_j | \mathbf{a}_{1:j-1}), \quad (\text{E17})$$

where the usual Gaussian likelihood $p(\mathbf{y}_j | \mathbf{f}_j) = \mathcal{N}(\mathbf{f}_j, \sigma_n^2 \mathbb{I})$ is used and $p(\mathbf{f}_j | \mathbf{a})$ can be derived from the joint Gaussian $p(\mathbf{f}_j, \mathbf{a}) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{[\mathbf{x}_j; \mathbf{A}][\mathbf{x}_j; \mathbf{A}]})$ with (B6). Using the posterior computed via (2) together with the predictive conditional $p(f_* | \mathbf{a})$ derived by (B6) from the assumed joint $p(f_*, \mathbf{a}) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{[\mathbf{x}_*, \mathbf{A}][\mathbf{x}_*, \mathbf{A}]})$ and integrating $\int p(f_* | \mathbf{a}) p(\mathbf{f}_j, \mathbf{a}) d\mathbf{a}$ via (B7) provides an approximation to the predictive posterior of full GP. Batch inference in these sparse global models can be done in $\mathcal{O}(M^2 N)$ time and $\mathcal{O}(MN)$ space (e.g. [3]).

In order to find optimal inducing inputs \mathbf{A} and hyperparameters $\boldsymbol{\theta}$, a sparse variation of the log marginal likelihood similar can be used e.g. [5–7]. In particular, the authors in [5] proposed to maximize a variational lower bound to the true GP marginal likelihood which has the effect that the sparse GP predictive distribution converges to the full GP predictive distribution as the number of inducing points increases. For larger datasets, stochastic optimization has been applied e.g. [8–11] to obtain faster and more data efficient optimization procedures. For recent reviews on the subject consider e.g. [1, 3, 19].

E.2 Local Independent GPs

An alternative to the global sparse inducing point methods constitute local approaches, which exploit multiple local GPs combined with averaging techniques to boost predictions. Beside other averaging techniques (e.g. mixture of experts [28, 28, 29]), the *Product of Expert (PoE)* scheme was proposed by [14], where individual predictions $p(f_{*j}|\mathbf{y}_j)$ from J experts based on the local data \mathbf{y}_j are aggregated to the final predictive distribution

$$p(f_*|\mathbf{y}) = \prod_{j=1}^J g_j(p(f_{*j}|\mathbf{y}_j)), \quad (\text{E18})$$

where g_j is a function depending on the particular PoE method discussed below and is in the original work of [14] just the identity. Note that we present here the version of PoEs where the noiseless predictions f_{*j} are aggregated instead of noisy aggregation with y_{*j} as described in some work of PoEs. The individual predictions $p(f_{*j}|\mathbf{y}_j)$ are local GP fits $\int p(f_{*j}|\mathbf{f}_j) p(\mathbf{f}_j|\mathbf{y}_j) d\mathbf{f}_j$ involving the predictive conditionals $p(f_{*j}|\mathbf{f}_j)$ derived by (B6) from the assumed joint $p(f_{*j}, \mathbf{f}_j) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{[\mathbf{x}_*, \mathbf{x}_j][\mathbf{x}_*, \mathbf{x}_j]})$ and the local posteriors $p(\mathbf{f}_j|\mathbf{y}_j) \propto p(\mathbf{y}_j|\mathbf{f}_j) p(\mathbf{f}_j)$, where the individual prior $p(\mathbf{f}_j) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{\mathbf{x}_j \mathbf{x}_j})$. Together with the usual Gaussian likelihood $p(\mathbf{y}_j|\mathbf{f}_j) = \mathcal{N}(\mathbf{f}_j, \sigma_n^2 \mathbb{I})$, the final noisy predictive distribution $p(y_*|\mathbf{y})$ can be obtained via $\int p(y_*|f_*) p(f_*|\mathbf{y}) df_*$. Similarly to Eqs. (1) and (2), the implicit posterior in all PoE method is

$$p(\mathbf{f}|\mathbf{y}) \propto p(\mathbf{f}, \mathbf{y}) = p(\mathbf{y}|\mathbf{f}) \prod_{j=1}^J p(\mathbf{f}_j) = \prod_{j=1}^J p(\mathbf{y}_j|\mathbf{f}_j) p(\mathbf{f}_j), \quad (\text{E19})$$

where the corresponding graphical model is depicted in Figure 1(iii).

The function g_j in (E18) takes as argument the predictive distribution $p_{*j} := p(f_{*j}|\mathbf{y}_j)$ which depends implicitly also on \mathbf{x}_* . In the original work [14] the authors used the identity $g_j(p_{*j}) = p_{*j}$ which produce underconfident prediction variances [19]. In order to mitigate this issue, the aggregation weights $g_j(p_{*j}) = p_{*j}^{1/J}$ were proposed [12] but still resulting in too large predictive uncertainty estimates [19]. The reason is that the experts are all equally weighted, however, the predictions at a particular point \mathbf{x}_* are not equally reliable, therefore in the generalized PoE (GPoE) [12] some varying weights $\beta_j(x_*)$ were introduced to quantify the contribution of the expert j at \mathbf{x}_* . Thus, $g_j(p_{*j}) = p_{*j}^{\beta_j(x_*)}$ with weights set to the difference in entropy between the expert's prior and posterior, that is, $\bar{\beta}_{*j} = \frac{1}{2} \log\left(\frac{v_{*0}}{v_{*j}}\right)$. This has the effect of increase or decreasing the importance of the experts based on the corresponding prediction uncertainty v_{*0} and v_{*j} . However, these general weights can produce overconfident uncertainty estimates, therefore the authors in [12] proposed also an version with normalized weights such that $\sum_j^J \bar{\beta}_j(x_*) = 1$.

In the following, PoE and GPoE refer to the version with normalized weights. Other important contributions in this field are BCM [16] and its robustified version RBCM [12], GRBCM [17], distributed local GPs [13], hierarchical PoEs [26], and local experts with consistent aggregations [15, 27]. We refer to [19] for a recent overview.

Simple baseline methods are the *minimal variance (minVar)* and the *nearest expert (NE)* aggregation, where only the prediction from the expert with minimal variance or nearest expert is used, respectively. Although both these methods show often surprisingly good performance, they suffer from a huge disadvantage, namely that there are serious discontinuities at the boundaries between the experts (see for instance Fig. 2) and thus often not useful in practice. This is also the main limitation of all local methods based only on the prediction of one expert (e.g. [23–25, 44]) and it was one of the reasons for introducing smooth PoEs with combined experts. Since in basically all cases minVar is better than NE (which is also consistent with the findings in [15]), we only compare our method to minVar and not NE for the sake of simplicity.

Appendix F Tables

Here we provide more results for the experiments in Section 4 and the datasets in Table 3a. In the following, we report different average quantities for several test points \mathbf{x}_*, y_* corresponding to the predictive distributions $p(y_*|\mathbf{y}) = \mathcal{N}(m_*, v_*)$. The considered quantities are *Kullback-Leibler-(KL)-divergence (KL)* to full GP, *Continuous Ranked Probability Score (CRPS)* and *95%-coverage (COV)*, *root mean squared error (RMSE)*, *absolut error (ABSE)*, *negative log probability (NLP)*, *root mean squared error to full GP (ERR)* and *log marginal likelihood (LML)*.

We use the KL to compare the closeness of predictive distributions of different GP approximation models to the one of full GP $\mathcal{N}(m, v)$. Since both are univariate Gaussians, the $KL(\mathcal{N}(m, v) \parallel \mathcal{N}(m_*, v_*))$ can be computed as $\frac{1}{2} \left(\log \frac{v_*}{v} + \frac{v}{v_*} + \frac{(m-m_*)^2}{v_*} - 1 \right)$.

The CRPS can be used to assess the respective accuracy of two probabilistic forecasting models. In particular, it is a measure between the forecast CDF F_* of $\mathcal{N}(m_*, v_*)$ and the empirical CDF of the observation \mathbf{y}_* and is defined as $CRPS(F_*, y_*) = \int (F(z) - 1_{z \geq y_*})^2 dz$.

The *95%-confidence interval* can be computed as $c_{1,2} = m_* \pm 1.96\sqrt{v_*}$. The 95%-coverage is then defined as $COV = 1_{c_1 \leq y_* \leq c_2}$.

The negative log probability is $-p(y_*|\mathbf{y}) = \frac{1}{2} \log(2\pi v_*) + \frac{(y_* - m_*)^2}{2v_*}$.

For all quantities except LML (large values are better) and COV (should be close to 0.95), small values mean better predictions.

| | time | LML | KL | ERR | CRPS | RMSE | ABSE | NLP | COV |
|----------|------------|---------------|--------------------|----------------------|---------------------|---------------------|---------------|--------------------|-------------|
| fullGP | 7.3 ± 0.6 | -314.2 ± 5.1 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.162 ± 0.004 | 0.311 ± 0.011 | 0.218 ± 0.005 | 0.47 ± 0.12 | 0.92 ± 0.01 |
| SGP(25) | 6.4 ± 0.6 | -595.4 ± 10.7 | 440.3 ± 19.6 | 0.314 ± 0.008 | 0.234 ± 0.005 | 0.422 ± 0.01 | 0.324 ± 0.005 | 1.11 ± 0.04 | 0.96 ± 0.01 |
| SGP(50) | 14.5 ± 2.6 | -539.6 ± 10.2 | 405.0 ± 31.3 | 0.291 ± 0.012 | 0.222 ± 0.004 | 0.402 ± 0.008 | 0.308 ± 0.005 | 1.01 ± 0.03 | 0.95 ± 0.01 |
| SGP(100) | 36.4 ± 2.9 | -494.6 ± 7.8 | 352.9 ± 29.5 | 0.264 ± 0.011 | 0.211 ± 0.004 | 0.384 ± 0.007 | 0.292 ± 0.006 | 0.92 ± 0.03 | 0.95 ± 0.01 |
| minVar | 1.5 ± 0.1 | -389.8 ± 2.9 | 122.2 ± 13.1 | 0.156 ± 0.012 | 0.175 ± 0.004 | 0.335 ± 0.011 | 0.236 ± 0.005 | 0.61 ± 0.09 | 0.92 ± 0.01 |
| GpOE | 1.4 ± 0.1 | -389.8 ± 2.9 | 174.4 ± 9.4 | 0.166 ± 0.01 | 0.186 ± 0.004 | 0.342 ± 0.01 | 0.255 ± 0.007 | 0.68 ± 0.05 | 0.96 ± 0.01 |
| BCM | 1.4 ± 0.1 | -389.8 ± 2.9 | 338.1 ± 32.7 | 0.185 ± 0.012 | 0.195 ± 0.005 | 0.354 ± 0.01 | 0.265 ± 0.007 | 1.16 ± 0.12 | 0.82 ± 0.01 |
| RBCM | 1.4 ± 0.1 | -389.8 ± 2.9 | 427.9 ± 35.0 | 0.166 ± 0.013 | 0.187 ± 0.005 | 0.342 ± 0.011 | 0.249 ± 0.006 | 1.43 ± 0.21 | 0.79 ± 0.01 |
| GRBCM | 1.7 ± 0.1 | -465.0 ± 3.1 | 224.6 ± 30.3 | 0.202 ± 0.011 | 0.19 ± 0.004 | 0.352 ± 0.01 | 0.262 ± 0.006 | 0.71 ± 0.05 | 0.92 ± 0.01 |
| CpOE(1) | 1.5 ± 0.0 | -397.0 ± 2.8 | 111.1 ± 12.5 | 0.146 ± 0.011 | 0.175 ± 0.004 | 0.333 ± 0.011 | 0.237 ± 0.006 | 0.59 ± 0.09 | 0.93 ± 0.01 |
| CpOE(2) | 2.1 ± 0.1 | -345.1 ± 5.6 | 89.6 ± 14.3 | 0.124 ± 0.013 | 0.172 ± 0.004 | 0.326 ± 0.011 | 0.232 ± 0.006 | 0.6 ± 0.1 | 0.91 ± 0.01 |
| CpOE(3) | 2.5 ± 0.1 | -337.0 ± 5.5 | 82.2 ± 14.3 | 0.116 ± 0.013 | 0.17 ± 0.004 | 0.323 ± 0.01 | 0.231 ± 0.005 | 0.59 ± 0.1 | 0.91 ± 0.01 |
| CpOE(4) | 2.8 ± 0.1 | -339.4 ± 5.0 | 79.5 ± 13.9 | 0.111 ± 0.012 | 0.171 ± 0.004 | 0.324 ± 0.011 | 0.232 ± 0.005 | 0.6 ± 0.1 | 0.91 ± 0.01 |

Table F5: Results for dataset *concrete*.

| | time | LML | KL | ERR | CRPS | RMSE | ABSE | NLP | COV |
|----------|------------|---------------|--------------------|----------------------|----------------------|----------------------|---------------------|--------------------|------------|
| fullGP | 25.5 ± 1.1 | -994.2 ± 1.1 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.283 ± 0.002 | 0.511 ± 0.004 | 0.39 ± 0.005 | 1.49 ± 0.02 | 0.94 ± 0.0 |
| SGP(25) | 7.5 ± 0.7 | -1082.8 ± 0.9 | 93.49 ± 3.86 | 0.232 ± 0.005 | 0.316 ± 0.003 | 0.561 ± 0.004 | 0.445 ± 0.005 | 1.68 ± 0.02 | 0.94 ± 0.0 |
| SGP(50) | 9.7 ± 1.4 | -1042.7 ± 5.2 | 41.4 ± 5.59 | 0.146 ± 0.012 | 0.299 ± 0.003 | 0.537 ± 0.003 | 0.416 ± 0.006 | 1.59 ± 0.01 | 0.94 ± 0.0 |
| SGP(100) | 14.4 ± 0.8 | -1009.6 ± 1.2 | 9.86 ± 1.73 | 0.069 ± 0.006 | 0.285 ± 0.002 | 0.514 ± 0.004 | 0.395 ± 0.005 | 1.51 ± 0.02 | 0.94 ± 0.0 |
| minVar | 2.0 ± 0.2 | -1025.8 ± 1.1 | 19.39 ± 1.78 | 0.101 ± 0.005 | 0.282 ± 0.002 | 0.508 ± 0.005 | 0.39 ± 0.003 | 1.48 ± 0.02 | 0.93 ± 0.0 |
| GpOE | 1.9 ± 0.1 | -1025.8 ± 1.1 | 54.22 ± 1.64 | 0.162 ± 0.003 | 0.301 ± 0.002 | 0.535 ± 0.004 | 0.424 ± 0.006 | 1.6 ± 0.01 | 0.96 ± 0.0 |
| BCM | 1.9 ± 0.1 | -1025.8 ± 1.1 | 257.61 ± 1.81 | 0.209 ± 0.005 | 0.313 ± 0.003 | 0.555 ± 0.006 | 0.422 ± 0.004 | 2.02 ± 0.04 | 0.82 ± 0.0 |
| RBCM | 1.9 ± 0.1 | -1025.8 ± 1.1 | 38.35 ± 1.56 | 0.132 ± 0.003 | 0.295 ± 0.003 | 0.528 ± 0.005 | 0.408 ± 0.005 | 1.56 ± 0.02 | 0.92 ± 0.0 |
| GRBCM | 2.3 ± 0.2 | -1048.9 ± 1.7 | 69.12 ± 6.48 | 0.196 ± 0.01 | 0.307 ± 0.004 | 0.551 ± 0.007 | 0.431 ± 0.006 | 1.64 ± 0.02 | 0.94 ± 0.0 |
| CpOE(1) | 2.1 ± 0.1 | -1025.8 ± 1.1 | 12.18 ± 0.92 | 0.079 ± 0.003 | 0.284 ± 0.002 | 0.51 ± 0.004 | 0.393 ± 0.003 | 1.49 ± 0.02 | 0.94 ± 0.0 |
| CpOE(2) | 2.8 ± 0.1 | -1010.1 ± 1.5 | 8.44 ± 0.66 | 0.066 ± 0.003 | 0.285 ± 0.002 | 0.512 ± 0.004 | 0.394 ± 0.004 | 1.5 ± 0.02 | 0.93 ± 0.0 |
| CpOE(3) | 3.1 ± 0.1 | -1007.0 ± 1.5 | 7.83 ± 0.58 | 0.064 ± 0.002 | 0.285 ± 0.002 | 0.513 ± 0.004 | 0.394 ± 0.004 | 1.5 ± 0.02 | 0.93 ± 0.0 |
| CpOE(4) | 3.3 ± 0.1 | -1004.8 ± 1.5 | 7.59 ± 0.63 | 0.062 ± 0.003 | 0.285 ± 0.002 | 0.513 ± 0.004 | 0.393 ± 0.004 | 1.5 ± 0.02 | 0.93 ± 0.0 |

Table F6: Results for dataset *mg*.

| | time | LML | KL | ERR | CRPS | RMSE | ABSE | NLP | COV |
|----------|-------------|---------------|--------------------|--------------------|----------------------|----------------------|----------------------|--------------------|-------------|
| fullGP | 114.8 ± 4.3 | -2113.6 ± 5.6 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.255 ± 0.005 | 0.471 ± 0.01 | 0.348 ± 0.007 | 1.3 ± 0.04 | 0.95 ± 0.0 |
| SGP(50) | 34.8 ± 4.8 | -2319.6 ± 7.4 | 137.62 ± 7.41 | 0.259 ± 0.009 | 0.288 ± 0.005 | 0.531 ± 0.012 | 0.395 ± 0.007 | 1.57 ± 0.04 | 0.95 ± 0.0 |
| SGP(100) | 46.6 ± 6.1 | -2242.4 ± 7.5 | 108.14 ± 6.24 | 0.229 ± 0.008 | 0.279 ± 0.005 | 0.514 ± 0.012 | 0.382 ± 0.007 | 1.5 ± 0.04 | 0.95 ± 0.0 |
| SGP(150) | 56.6 ± 6.8 | -2205.9 ± 6.6 | 90.94 ± 6.01 | 0.21 ± 0.009 | 0.275 ± 0.005 | 0.508 ± 0.012 | 0.376 ± 0.007 | 1.47 ± 0.04 | 0.94 ± 0.0 |
| minVar | 7.2 ± 0.2 | -2312.6 ± 6.8 | 63.58 ± 2.93 | 0.19 ± 0.01 | 0.272 ± 0.006 | 0.508 ± 0.016 | 0.374 ± 0.008 | 1.41 ± 0.04 | 0.95 ± 0.0 |
| GpOE | 7.2 ± 0.2 | -2312.6 ± 6.8 | 98.01 ± 3.06 | 0.2 ± 0.013 | 0.279 ± 0.006 | 0.515 ± 0.02 | 0.378 ± 0.008 | 1.49 ± 0.03 | 0.97 ± 0.0 |
| BCM | 7.2 ± 0.2 | -2312.6 ± 6.8 | 222.78 ± 4.12 | 0.2 ± 0.008 | 0.28 ± 0.007 | 0.511 ± 0.016 | 0.38 ± 0.008 | 1.75 ± 0.1 | 0.87 ± 0.01 |
| RBCM | 7.2 ± 0.2 | -2312.6 ± 6.8 | 635.61 ± 21.61 | 0.194 ± 0.011 | 0.285 ± 0.007 | 0.513 ± 0.018 | 0.378 ± 0.008 | 2.54 ± 0.18 | 0.77 ± 0.01 |
| GRBCM | 6.5 ± 0.2 | -2397.3 ± 6.2 | 105.64 ± 5.13 | 0.24 ± 0.008 | 0.284 ± 0.005 | 0.525 ± 0.012 | 0.391 ± 0.007 | 1.5 ± 0.04 | 0.95 ± 0.01 |
| CpOE(1) | 7.8 ± 0.2 | -2316.1 ± 6.8 | 62.99 ± 2.94 | 0.186 ± 0.011 | 0.272 ± 0.006 | 0.507 ± 0.018 | 0.372 ± 0.008 | 1.41 ± 0.04 | 0.96 ± 0.0 |
| CpOE(2) | 10.6 ± 0.2 | -2164.9 ± 6.7 | 36.45 ± 3.02 | 0.142 ± 0.011 | 0.264 ± 0.005 | 0.491 ± 0.015 | 0.361 ± 0.008 | 1.36 ± 0.04 | 0.95 ± 0.0 |
| CpOE(3) | 12.9 ± 0.2 | -2165.9 ± 6.7 | 36.27 ± 2.99 | 0.141 ± 0.01 | 0.263 ± 0.005 | 0.49 ± 0.014 | 0.361 ± 0.008 | 1.36 ± 0.04 | 0.95 ± 0.0 |
| CpOE(4) | 14.9 ± 0.2 | -2166.2 ± 6.7 | 36.03 ± 3.0 | 0.14 ± 0.01 | 0.263 ± 0.005 | 0.489 ± 0.014 | 0.361 ± 0.008 | 1.36 ± 0.04 | 0.95 ± 0.0 |

Table F7: Results for dataset *space*.

| | time | LML | KL | ERR | CRPS | RMSE | ABSE | NLP | COV |
|----------|--------------|---------------|------------------|-------------------|---------------------|----------------------|----------------------|--------------------|-------------|
| fullGP | 237.9 ± 12.2 | -3722.3 ± 7.4 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.34 ± 0.005 | 0.635 ± 0.012 | 0.459 ± 0.006 | 1.92 ± 0.04 | 0.94 ± 0.0 |
| SGP(20) | 21.9 ± 2.5 | -3785.3 ± 5.8 | 27.8 ± 4.1 | 0.15 ± 0.01 | 0.343 ± 0.004 | 0.635 ± 0.011 | 0.463 ± 0.005 | 1.93 ± 0.03 | 0.95 ± 0.0 |
| SGP(50) | 26.4 ± 3.6 | -3758.7 ± 7.6 | 22.4 ± 3.9 | 0.14 ± 0.01 | 0.342 ± 0.004 | 0.633 ± 0.011 | 0.461 ± 0.006 | 1.93 ± 0.03 | 0.94 ± 0.0 |
| SGP(100) | 58.9 ± 7.0 | -3746.9 ± 7.4 | 15.6 ± 3.5 | 0.11 ± 0.01 | 0.34 ± 0.005 | 0.631 ± 0.012 | 0.457 ± 0.006 | 1.92 ± 0.04 | 0.94 ± 0.0 |
| minVar | 6.4 ± 0.4 | -3847.3 ± 7.2 | 25.1 ± 1.5 | 0.15 ± 0.0 | 0.346 ± 0.005 | 0.647 ± 0.013 | 0.466 ± 0.006 | 1.94 ± 0.04 | 0.94 ± 0.0 |
| GpOE | 6.3 ± 0.4 | -3847.3 ± 7.2 | 50.3 ± 1.0 | 0.19 ± 0.0 | 0.353 ± 0.004 | 0.652 ± 0.011 | 0.478 ± 0.006 | 1.99 ± 0.02 | 0.96 ± 0.0 |
| BCM | 6.3 ± 0.3 | -3847.3 ± 7.2 | 1838.2 ± 46.8 | 0.16 ± 0.0 | 0.373 ± 0.006 | 0.642 ± 0.011 | 0.473 ± 0.006 | 5.33 ± 0.24 | 0.67 ± 0.01 |
| RBCM | 6.3 ± 0.3 | -3847.3 ± 7.2 | 1147.4 ± 64.8 | 0.12 ± 0.0 | 0.362 ± 0.006 | 0.638 ± 0.012 | 0.466 ± 0.006 | 4.01 ± 0.21 | 0.73 ± 0.01 |
| GRBCM | 7.6 ± 0.4 | -3864.0 ± 7.6 | 36.4 ± 1.9 | 0.18 ± 0.0 | 0.353 ± 0.004 | 0.661 ± 0.011 | 0.477 ± 0.005 | 1.98 ± 0.03 | 0.94 ± 0.0 |
| CpOE(1) | 6.4 ± 0.4 | -3848.6 ± 7.3 | 16.8 ± 0.6 | 0.12 ± 0.0 | 0.342 ± 0.004 | 0.638 ± 0.012 | 0.463 ± 0.005 | 1.92 ± 0.03 | 0.95 ± 0.0 |
| CpOE(2) | 7.5 ± 0.3 | -3737.3 ± 7.0 | 8.1 ± 0.5 | 0.08 ± 0.0 | 0.341 ± 0.005 | 0.636 ± 0.012 | 0.463 ± 0.006 | 1.92 ± 0.04 | 0.94 ± 0.0 |
| CpOE(3) | 9.3 ± 0.5 | -3736.5 ± 7.2 | 6.2 ± 0.6 | 0.07 ± 0.0 | 0.341 ± 0.005 | 0.636 ± 0.012 | 0.461 ± 0.006 | 1.92 ± 0.04 | 0.94 ± 0.0 |
| CpOE(4) | 10.4 ± 0.3 | -3733.7 ± 7.0 | 4.7 ± 0.5 | 0.06 ± 0.0 | 0.34 ± 0.005 | 0.635 ± 0.012 | 0.46 ± 0.006 | 1.91 ± 0.04 | 0.94 ± 0.0 |

Table F8: Results for dataset *abalone*.

| | time | LML | KL | ERR | CRPS | RMSE | ABSE | NLP | COV |
|----------|-------------|----------------|-------------------|-------------------|----------------------|----------------------|----------------------|--------------------|------------|
| fullGP | 161.5 ± 3.6 | -1232.1 ± 7.4 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.148 ± 0.001 | 0.267 ± 0.001 | 0.207 ± 0.001 | 0.17 ± 0.01 | 0.94 ± 0.0 |
| SGP(100) | 42.2 ± 6.5 | -4033.6 ± 27.1 | 603.7 ± 9.4 | 0.4 ± 0.01 | 0.265 ± 0.003 | 0.476 ± 0.005 | 0.369 ± 0.004 | 1.35 ± 0.02 | 0.96 ± 0.0 |
| SGP(200) | 49.8 ± 3.3 | -3141.3 ± 17.8 | 408.4 ± 3.7 | 0.29 ± 0.0 | 0.218 ± 0.001 | 0.392 ± 0.001 | 0.303 ± 0.001 | 0.96 ± 0.0 | 0.96 ± 0.0 |
| SGP(300) | 54.8 ± 2.2 | -2732.8 ± 13.5 | 323.1 ± 5.0 | 0.25 ± 0.0 | 0.201 ± 0.001 | 0.363 ± 0.001 | 0.281 ± 0.001 | 0.8 ± 0.01 | 0.96 ± 0.0 |
| minVar | 9.3 ± 0.2 | -2820.5 ± 9.0 | 211.0 ± 2.3 | 0.2 ± 0.0 | 0.183 ± 0.001 | 0.333 ± 0.001 | 0.256 ± 0.001 | 0.59 ± 0.01 | 0.94 ± 0.0 |
| GPoE | 9.4 ± 0.1 | -2820.5 ± 9.0 | 342.3 ± 2.6 | 0.23 ± 0.0 | 0.202 ± 0.001 | 0.354 ± 0.002 | 0.278 ± 0.002 | 0.84 ± 0.01 | 0.99 ± 0.0 |
| BCM | 9.4 ± 0.1 | -2820.5 ± 9.0 | 1629.2 ± 24.7 | 0.25 ± 0.0 | 0.218 ± 0.002 | 0.367 ± 0.002 | 0.278 ± 0.002 | 3.45 ± 0.07 | 0.64 ± 0.0 |
| RBCM | 9.4 ± 0.2 | -2820.5 ± 9.0 | 939.3 ± 17.4 | 0.2 ± 0.0 | 0.193 ± 0.001 | 0.331 ± 0.002 | 0.253 ± 0.001 | 2.06 ± 0.05 | 0.71 ± 0.0 |
| GRBCM | 11.9 ± 0.2 | -2981.3 ± 9.6 | 129.8 ± 3.0 | 0.14 ± 0.0 | 0.168 ± 0.001 | 0.303 ± 0.001 | 0.235 ± 0.001 | 0.43 ± 0.01 | 0.94 ± 0.0 |
| CPoE(1) | 9.2 ± 0.1 | -2822.7 ± 8.9 | 152.4 ± 1.7 | 0.15 ± 0.0 | 0.17 ± 0.001 | 0.307 ± 0.001 | 0.237 ± 0.001 | 0.46 ± 0.0 | 0.97 ± 0.0 |
| CPoE(2) | 12.9 ± 0.1 | -1811.2 ± 11.1 | 79.9 ± 1.3 | 0.11 ± 0.0 | 0.161 ± 0.001 | 0.29 ± 0.001 | 0.225 ± 0.001 | 0.33 ± 0.01 | 0.95 ± 0.0 |
| CPoE(3) | 19.8 ± 0.3 | -1466.0 ± 9.9 | 46.9 ± 1.0 | 0.09 ± 0.0 | 0.155 ± 0.001 | 0.279 ± 0.001 | 0.217 ± 0.001 | 0.26 ± 0.01 | 0.95 ± 0.0 |
| CPoE(4) | 27.8 ± 0.2 | -1363.8 ± 9.2 | 32.8 ± 1.0 | 0.07 ± 0.0 | 0.153 ± 0.001 | 0.276 ± 0.001 | 0.215 ± 0.001 | 0.24 ± 0.01 | 0.94 ± 0.0 |

Table F9: Results for dataset *kin*.

| | time | LML | CRPS | RMSE | ABSE | NLP | COV |
|-----------|-------------|----------------|----------------------|---------------------|----------------------|--------------------|------------|
| SGP(250) | 77.7 ± 0.4 | -4163.9 ± 23.7 | 0.207 ± 0.002 | 0.366 ± 0.004 | 0.282 ± 0.002 | 0.93 ± 0.01 | 0.98 ± 0.0 |
| SGP(500) | 112.1 ± 1.2 | -3242.2 ± 12.6 | 0.183 ± 0.001 | 0.324 ± 0.002 | 0.252 ± 0.001 | 0.67 ± 0.01 | 0.98 ± 0.0 |
| SGP(1000) | 244.1 ± 2.9 | -2534.7 ± 9.0 | 0.166 ± 0.001 | 0.294 ± 0.002 | 0.23 ± 0.001 | 0.46 ± 0.01 | 0.98 ± 0.0 |
| minVar | 14.4 ± 0.5 | -3388.8 ± 7.9 | 0.173 ± 0.002 | 0.314 ± 0.004 | 0.242 ± 0.002 | 0.48 ± 0.02 | 0.94 ± 0.0 |
| GPoE | 14.4 ± 0.5 | -3388.8 ± 7.9 | 0.193 ± 0.001 | 0.34 ± 0.003 | 0.267 ± 0.002 | 0.76 ± 0.01 | 0.99 ± 0.0 |
| BCM | 14.4 ± 0.5 | -3388.8 ± 7.9 | 0.21 ± 0.001 | 0.35 ± 0.003 | 0.266 ± 0.002 | 3.6 ± 0.1 | 0.63 ± 0.0 |
| RBCM | 14.4 ± 0.5 | -3388.8 ± 7.9 | 0.188 ± 0.001 | 0.318 ± 0.003 | 0.244 ± 0.002 | 2.39 ± 0.09 | 0.69 ± 0.0 |
| GRBCM | 16.5 ± 0.4 | -3388.8 ± 7.9 | 0.164 ± 0.001 | 0.294 ± 0.003 | 0.229 ± 0.002 | 0.37 ± 0.02 | 0.94 ± 0.0 |
| CPoE(1) | 13.8 ± 0.2 | -3393.9 ± 8.0 | 0.163 ± 0.001 | 0.292 ± 0.003 | 0.226 ± 0.002 | 0.38 ± 0.01 | 0.97 ± 0.0 |
| CPoE(2) | 18.9 ± 0.3 | -2076.6 ± 12.9 | 0.155 ± 0.001 | 0.278 ± 0.002 | 0.217 ± 0.001 | 0.27 ± 0.01 | 0.95 ± 0.0 |
| CPoE(3) | 31.7 ± 0.6 | -1655.2 ± 8.7 | 0.151 ± 0.001 | 0.27 ± 0.002 | 0.211 ± 0.001 | 0.21 ± 0.01 | 0.95 ± 0.0 |

Table F10: Results for dataset *kin2* for the stochastic versions.

| | time | LML | CRPS | RMSE | ABSE | NLP | COV |
|-----------|-------------|----------------|---------------------|---------------------|----------------------|--------------------|------------|
| SGP(250) | 70.9 ± 3.7 | -3905.6 ± 23.3 | 0.207 ± 0.002 | 0.373 ± 0.004 | 0.287 ± 0.002 | 0.85 ± 0.02 | 0.96 ± 0.0 |
| SGP(500) | 86.1 ± 1.8 | -2968.6 ± 11.7 | 0.181 ± 0.001 | 0.325 ± 0.003 | 0.252 ± 0.001 | 0.57 ± 0.01 | 0.96 ± 0.0 |
| SGP(1000) | 143.6 ± 3.6 | -2277.2 ± 8.6 | 0.162 ± 0.001 | 0.292 ± 0.002 | 0.225 ± 0.001 | 0.36 ± 0.01 | 0.96 ± 0.0 |
| minVar | 13.8 ± 0.2 | -3384.5 ± 7.8 | 0.173 ± 0.002 | 0.314 ± 0.004 | 0.241 ± 0.002 | 0.48 ± 0.02 | 0.94 ± 0.0 |
| GPoE | 13.8 ± 0.2 | -3384.5 ± 7.8 | 0.193 ± 0.001 | 0.34 ± 0.002 | 0.267 ± 0.002 | 0.75 ± 0.01 | 0.99 ± 0.0 |
| BCM | 13.8 ± 0.2 | -3384.5 ± 7.8 | 0.209 ± 0.001 | 0.35 ± 0.003 | 0.266 ± 0.002 | 3.63 ± 0.07 | 0.63 ± 0.0 |
| RBCM | 13.8 ± 0.2 | -3384.5 ± 7.8 | 0.187 ± 0.001 | 0.317 ± 0.003 | 0.243 ± 0.001 | 2.38 ± 0.06 | 0.69 ± 0.0 |
| GRBCM | 18.8 ± 0.4 | -3608.7 ± 8.4 | 0.164 ± 0.001 | 0.294 ± 0.002 | 0.229 ± 0.002 | 0.38 ± 0.02 | 0.94 ± 0.0 |
| CPoE(1) | 16.2 ± 0.8 | -3389.8 ± 8.0 | 0.162 ± 0.001 | 0.292 ± 0.003 | 0.225 ± 0.002 | 0.37 ± 0.01 | 0.97 ± 0.0 |
| CPoE(2) | 21.5 ± 0.7 | -2071.4 ± 13.0 | 0.155 ± 0.001 | 0.278 ± 0.002 | 0.217 ± 0.001 | 0.26 ± 0.01 | 0.95 ± 0.0 |
| CPoE(3) | 34.3 ± 0.9 | -1650.7 ± 8.3 | 0.15 ± 0.001 | 0.27 ± 0.002 | 0.211 ± 0.001 | 0.21 ± 0.01 | 0.94 ± 0.0 |

Table F11: Results for dataset *kin2* for the deterministic batch version.

| | time | LML | CRPS | RMSE | ABSE | NLP | COV |
|-----------|-------------|-----------------|----------------------|----------------------|----------------------|--------------------|-------------|
| SGP(250) | 248.6 ± 0.6 | -15182.0 ± 35.7 | 0.254 ± 0.003 | 0.48 ± 0.009 | 0.335 ± 0.004 | 1.42 ± 0.03 | 0.95 ± 0.0 |
| SGP(500) | 346.9 ± 3.4 | -15074.6 ± 37.2 | 0.253 ± 0.003 | 0.478 ± 0.009 | 0.333 ± 0.004 | 1.41 ± 0.03 | 0.95 ± 0.0 |
| SGP(1000) | 727.6 ± 3.5 | -14961.2 ± 31.4 | 0.252 ± 0.003 | 0.476 ± 0.009 | 0.332 ± 0.004 | 1.4 ± 0.03 | 0.95 ± 0.0 |
| minVar | 28.2 ± 1.0 | -15387.4 ± 17.5 | 0.257 ± 0.003 | 0.491 ± 0.009 | 0.337 ± 0.005 | 1.42 ± 0.04 | 0.94 ± 0.0 |
| GPoE | 28.3 ± 1.0 | -15387.4 ± 17.5 | 0.289 ± 0.003 | 0.534 ± 0.009 | 0.371 ± 0.004 | 1.64 ± 0.02 | 0.96 ± 0.0 |
| BCM | 28.5 ± 0.9 | -15387.4 ± 17.5 | 0.321 ± 0.004 | 0.536 ± 0.01 | 0.373 ± 0.004 | 20.72 ± 1.0 | 0.45 ± 0.0 |
| RBCM | 28.5 ± 0.9 | -15387.4 ± 17.5 | 0.303 ± 0.005 | 0.515 ± 0.01 | 0.358 ± 0.004 | 15.98 ± 0.9 | 0.51 ± 0.01 |
| GRBCM | 33.5 ± 1.2 | -15387.4 ± 17.5 | 0.262 ± 0.003 | 0.499 ± 0.009 | 0.346 ± 0.004 | 1.44 ± 0.03 | 0.94 ± 0.0 |
| CPoE(1) | 24.5 ± 0.1 | -15404.2 ± 17.8 | 0.259 ± 0.004 | 0.492 ± 0.01 | 0.335 ± 0.005 | 1.43 ± 0.04 | 0.95 ± 0.0 |
| CPoE(2) | 33.4 ± 0.2 | -13645.5 ± 19.8 | 0.251 ± 0.003 | 0.479 ± 0.009 | 0.328 ± 0.004 | 1.36 ± 0.04 | 0.94 ± 0.0 |
| CPoE(3) | 52.0 ± 0.5 | -13483.2 ± 15.6 | 0.249 ± 0.004 | 0.476 ± 0.01 | 0.324 ± 0.004 | 1.34 ± 0.04 | 0.94 ± 0.0 |

Table F12: Results for dataset *cadata*.

| | time | LML | CRPS | RMSE | ABSE | NLP | COV |
|-----------|--------------|------------------|------------------------|------------------------|----------------------|---------------------|-------------|
| SGP(250) | 473.4 ± 1.0 | 9370.3 ± 60.7 | 0.0746 ± 0.0005 | 0.1407 ± 0.0008 | 0.097 ± 0.001 | -0.39 ± 0.01 | 0.95 ± 0.0 |
| SGP(500) | 730.1 ± 1.1 | 12112.0 ± 68.2 | 0.0695 ± 0.0003 | 0.1304 ± 0.001 | 0.09 ± 0.001 | -0.49 ± 0.01 | 0.95 ± 0.0 |
| SGP(1000) | 1718.5 ± 1.8 | 16034.0 ± 91.6 | 0.0628 ± 0.0003 | 0.1172 ± 0.0009 | 0.081 ± 0.0 | -0.64 ± 0.01 | 0.96 ± 0.0 |
| minVar | 71.3 ± 23.1 | 27128.2 ± 20.2 | 0.0516 ± 0.0008 | 0.1024 ± 0.0034 | 0.067 ± 0.001 | -1.88 ± 0.04 | 0.93 ± 0.0 |
| GPoE | 71.4 ± 23.2 | 27128.2 ± 20.2 | 0.0862 ± 0.0004 | 0.1322 ± 0.0013 | 0.096 ± 0.001 | -0.57 ± 0.01 | 1.0 ± 0.0 |
| BCM | 71.5 ± 23.2 | 27128.2 ± 20.2 | 0.095 ± 0.001 | 0.1544 ± 0.001 | 0.115 ± 0.001 | 7.86 ± 0.3 | 0.48 ± 0.01 |
| RBCM | 71.6 ± 23.2 | 27128.2 ± 20.2 | 0.0726 ± 0.0009 | 0.1196 ± 0.0013 | 0.086 ± 0.001 | 11.45 ± 0.47 | 0.5 ± 0.01 |
| GRBCM | 84.6 ± 23.0 | 27128.2 ± 20.2 | 0.06 ± 0.0007 | 0.1102 ± 0.001 | 0.079 ± 0.001 | -0.52 ± 0.08 | 0.79 ± 0.01 |
| CPoE(1) | 45.4 ± 0.2 | -41213.2 ± 883.2 | 0.0516 ± 0.0005 | 0.0998 ± 0.0019 | 0.067 ± 0.001 | -1.86 ± 0.02 | 0.96 ± 0.0 |
| CPoE(2) | 67.3 ± 0.4 | -37867.5 ± 911.8 | 0.0509 ± 0.0006 | 0.0977 ± 0.0015 | 0.067 ± 0.001 | -1.8 ± 0.02 | 0.93 ± 0.0 |
| CPoE(3) | 134.3 ± 1.2 | -37204.6 ± 949.1 | 0.0507 ± 0.0005 | 0.0975 ± 0.0011 | 0.067 ± 0.001 | -1.78 ± 0.02 | 0.92 ± 0.0 |

Table F13: Results for dataset *sarcos*.

| | time | LML | CRPS | RMSE | ABSE | NLP | COV |
|-----------|--------------|-----------------|----------------------|----------------------|----------------------|--------------------|-------------|
| SGP(250) | 443.2 ± 2.1 | -53395.2 ± 80.2 | 0.334 ± 0.004 | 0.59 ± 0.008 | 0.475 ± 0.007 | 1.77 ± 0.02 | 0.96 ± 0.0 |
| SGP(500) | 632.9 ± 2.7 | -52988.7 ± 58.9 | 0.329 ± 0.005 | 0.582 ± 0.008 | 0.467 ± 0.007 | 1.75 ± 0.02 | 0.96 ± 0.0 |
| SGP(1000) | 1362.5 ± 4.8 | -52592.1 ± 46.9 | 0.325 ± 0.005 | 0.575 ± 0.008 | 0.459 ± 0.007 | 1.74 ± 0.02 | 0.96 ± 0.0 |
| minVar | 45.8 ± 1.0 | -39976.0 ± 22.6 | 0.294 ± 0.003 | 0.607 ± 0.006 | 0.387 ± 0.003 | 1.4 ± 0.03 | 0.93 ± 0.0 |
| GPoE | 45.6 ± 0.8 | -39976.0 ± 22.6 | 0.302 ± 0.003 | 0.6 ± 0.006 | 0.409 ± 0.005 | 1.43 ± 0.02 | 0.97 ± 0.0 |
| BCM | 45.7 ± 0.9 | -39976.0 ± 22.6 | 0.316 ± 0.005 | 0.615 ± 0.009 | 0.416 ± 0.007 | 2.47 ± 0.1 | 0.82 ± 0.01 |
| RBCM | 45.7 ± 0.9 | -39976.0 ± 22.6 | 0.312 ± 0.004 | 0.647 ± 0.008 | 0.425 ± 0.006 | 1.61 ± 0.05 | 0.91 ± 0.01 |
| GRBCM | 59.4 ± 1.1 | -39976.0 ± 22.6 | 0.31 ± 0.004 | 0.642 ± 0.008 | 0.421 ± 0.005 | 1.5 ± 0.04 | 0.92 ± 0.01 |
| CPoE(1) | 45.1 ± 0.3 | -40075.2 ± 22.1 | 0.289 ± 0.003 | 0.596 ± 0.006 | 0.38 ± 0.004 | 1.35 ± 0.03 | 0.94 ± 0.0 |
| CPoE(2) | 70.3 ± 0.6 | -39571.2 ± 65.5 | 0.287 ± 0.004 | 0.589 ± 0.007 | 0.38 ± 0.005 | 1.36 ± 0.03 | 0.93 ± 0.0 |
| CPoE(3) | 123.8 ± 1.4 | -39439.5 ± 98.8 | 0.282 ± 0.004 | 0.575 ± 0.008 | 0.372 ± 0.006 | 1.37 ± 0.04 | 0.92 ± 0.01 |

Table F14: Results for dataset *casp*.

| | time | RMSE | ABSE | CRPS | COV |
|------------------|--------------|---------------------|----------------------|----------------------|-------------|
| fullGP-SE | 12.52 ± 0.98 | 0.311 ± 0.011 | 0.218 ± 0.005 | 0.162 ± 0.004 | 0.92 ± 0.01 |
| fullGP-FLEX | 28.23 ± 7.99 | 0.254 ± 0.01 | 0.169 ± 0.005 | 0.128 ± 0.004 | 0.94 ± 0.01 |
| CPoE(1)-SE | 1.97 ± 0.06 | 0.333 ± 0.011 | 0.236 ± 0.006 | 0.175 ± 0.004 | 0.92 ± 0.01 |
| CPoE(2)-SE | 2.39 ± 0.06 | 0.326 ± 0.011 | 0.231 ± 0.006 | 0.171 ± 0.004 | 0.91 ± 0.01 |
| CPoE(3)-SE | 2.77 ± 0.11 | 0.323 ± 0.01 | 0.23 ± 0.005 | 0.17 ± 0.004 | 0.91 ± 0.01 |
| CPoE(4)-SE | 2.89 ± 0.06 | 0.324 ± 0.011 | 0.231 ± 0.005 | 0.171 ± 0.004 | 0.91 ± 0.01 |
| CPoE(1)-FLEX | 16.37 ± 1.11 | 0.266 ± 0.009 | 0.175 ± 0.005 | 0.133 ± 0.004 | 0.93 ± 0.01 |
| CPoE(2)-FLEX | 17.06 ± 1.14 | 0.259 ± 0.01 | 0.172 ± 0.005 | 0.13 ± 0.004 | 0.94 ± 0.01 |
| CPoE(3)-FLEX | 17.35 ± 1.09 | 0.255 ± 0.01 | 0.171 ± 0.005 | 0.129 ± 0.004 | 0.94 ± 0.01 |
| CPoE(4)-FLEX | 17.7 ± 1.1 | 0.255 ± 0.01 | 0.171 ± 0.005 | 0.129 ± 0.004 | 0.95 ± 0.01 |
| MLP(100-100) | 4.52 ± 0.39 | 0.289 ± 0.011 | 0.204 ± 0.005 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| MLP(500-500) | 10.73 ± 0.62 | 0.292 ± 0.008 | 0.208 ± 0.004 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| MLP(100-100-100) | 4.14 ± 0.18 | 0.285 ± 0.011 | 0.2 ± 0.006 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| XGboost | 32.16 ± 1.87 | 0.323 ± 0.008 | 0.235 ± 0.007 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| LinReg | 0.01 ± 0.0 | 0.626 ± 0.01 | 0.492 ± 0.009 | 0.0 ± 0.0 | 0.0 ± 0.0 |

Table F15: Comparison with non-GP methods for dataset *concrete*. Best method (besides full GP) is indicated in bold.

| | time | RMSE | ABSE | CRPS | COV |
|---------------------|----------------|----------------------|----------------------|----------------------|-------------|
| fullGP-SE | 38.89 ± 1.93 | 0.511 ± 0.004 | 0.39 ± 0.005 | 0.283 ± 0.002 | 0.94 ± 0.0 |
| fullGP-FLEX | 161.63 ± 12.31 | 0.509 ± 0.004 | 0.388 ± 0.004 | 0.282 ± 0.002 | 0.93 ± 0.0 |
| CPoE(1)-SE | 2.37 ± 0.14 | 0.508 ± 0.005 | 0.39 ± 0.003 | 0.282 ± 0.002 | 0.94 ± 0.01 |
| CPoE(2)-SE | 2.81 ± 0.11 | 0.512 ± 0.004 | 0.393 ± 0.004 | 0.285 ± 0.002 | 0.93 ± 0.0 |
| CPoE(3)-SE | 3.13 ± 0.09 | 0.513 ± 0.004 | 0.394 ± 0.004 | 0.285 ± 0.002 | 0.93 ± 0.0 |
| CPoE(4)-SE | 3.38 ± 0.08 | 0.513 ± 0.004 | 0.393 ± 0.004 | 0.284 ± 0.002 | 0.93 ± 0.0 |
| CPoE(1)-FLEX | 24.06 ± 2.31 | 0.511 ± 0.005 | 0.388 ± 0.004 | 0.281 ± 0.003 | 0.93 ± 0.01 |
| CPoE(2)-FLEX | 24.68 ± 2.31 | 0.515 ± 0.005 | 0.391 ± 0.005 | 0.283 ± 0.003 | 0.93 ± 0.01 |
| CPoE(3)-FLEX | 25.37 ± 2.31 | 0.516 ± 0.005 | 0.392 ± 0.005 | 0.284 ± 0.004 | 0.93 ± 0.01 |
| CPoE(4)-FLEX | 26.14 ± 2.35 | 0.516 ± 0.005 | 0.391 ± 0.005 | 0.284 ± 0.004 | 0.93 ± 0.01 |
| MLP(100-100) | 3.83 ± 0.35 | 0.525 ± 0.004 | 0.398 ± 0.006 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| MLP(500-500) | 11.97 ± 0.8 | 0.522 ± 0.004 | 0.397 ± 0.006 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| MLP(100-100-100) | 4.84 ± 0.36 | 0.531 ± 0.006 | 0.403 ± 0.007 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| XGboost | 0.65 ± 0.01 | 0.545 ± 0.008 | 0.405 ± 0.008 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| LinReg | 0.01 ± 0.0 | 0.633 ± 0.006 | 0.509 ± 0.006 | 0.0 ± 0.0 | 0.0 ± 0.0 |

Table F16: Comparison with non-GP methods for dataset *mg*. Best method (besides full GP) is indicated in bold.

| | time | RMSE | ABSE | CRPS | COV |
|---------------------|-----------------|----------------------|----------------------|----------------------|-------------|
| fullGP-SE | 137.76 ± 5.07 | 0.471 ± 0.01 | 0.348 ± 0.007 | 0.255 ± 0.005 | 0.95 ± 0.0 |
| fullGP-FLEX | 700.27 ± 255.47 | 0.455 ± 0.014 | 0.328 ± 0.007 | 0.238 ± 0.004 | 0.94 ± 0.0 |
| CPoE(1)-SE | 11.64 ± 0.59 | 0.506 ± 0.017 | 0.371 ± 0.008 | 0.27 ± 0.005 | 0.95 ± 0.0 |
| CPoE(2)-SE | 14.86 ± 0.53 | 0.49 ± 0.015 | 0.36 ± 0.008 | 0.263 ± 0.005 | 0.95 ± 0.0 |
| CPoE(3)-SE | 18.78 ± 0.56 | 0.489 ± 0.014 | 0.361 ± 0.008 | 0.263 ± 0.005 | 0.95 ± 0.0 |
| CPoE(4)-SE | 22.57 ± 0.56 | 0.489 ± 0.014 | 0.361 ± 0.008 | 0.263 ± 0.005 | 0.95 ± 0.0 |
| CPoE(1)-FLEX | 72.08 ± 4.54 | 0.631 ± 0.121 | 0.347 ± 0.012 | 0.252 ± 0.011 | 0.95 ± 0.01 |
| CPoE(2)-FLEX | 76.73 ± 4.55 | 0.446 ± 0.019 | 0.324 ± 0.005 | 0.236 ± 0.004 | 0.95 ± 0.0 |
| CPoE(3)-FLEX | 82.48 ± 4.6 | 0.444 ± 0.02 | 0.321 ± 0.005 | 0.234 ± 0.004 | 0.95 ± 0.0 |
| CPoE(4)-FLEX | 86.95 ± 4.57 | 0.449 ± 0.023 | 0.324 ± 0.006 | 0.236 ± 0.005 | 0.95 ± 0.01 |
| MLP(100-100) | 7.22 ± 0.68 | 0.482 ± 0.011 | 0.355 ± 0.008 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| MLP(500-500) | 33.31 ± 2.41 | 0.475 ± 0.011 | 0.353 ± 0.008 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| MLP(100-100-100) | 9.87 ± 1.13 | 0.476 ± 0.01 | 0.351 ± 0.006 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| XGboost | 0.72 ± 0.01 | 0.543 ± 0.026 | 0.386 ± 0.007 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| LinReg | 0.01 ± 0.0 | 0.645 ± 0.014 | 0.485 ± 0.008 | 0.0 ± 0.0 | 0.0 ± 0.0 |

Table F17: Comparison with non-GP methods for dataset *space*. Best method (besides full GP) is indicated in bold.

| | time | RMSE | ABSE | CRPS | COV |
|---------------------|------------------|----------------------|----------------------|---------------------|------------|
| fullGP-SE | 327.61 ± 23.85 | 0.635 ± 0.012 | 0.459 ± 0.006 | 0.34 ± 0.005 | 0.94 ± 0.0 |
| fullGP-FLEX | 1008.11 ± 165.51 | 0.638 ± 0.013 | 0.461 ± 0.007 | 0.34 ± 0.005 | 0.94 ± 0.0 |
| CPoE(1)-SE | 8.92 ± 0.5 | 0.637 ± 0.012 | 0.459 ± 0.006 | 0.34 ± 0.005 | 0.94 ± 0.0 |
| CPoE(2)-SE | 10.44 ± 0.41 | 0.634 ± 0.012 | 0.458 ± 0.006 | 0.34 ± 0.005 | 0.94 ± 0.0 |
| CPoE(3)-SE | 12.62 ± 0.39 | 0.634 ± 0.012 | 0.458 ± 0.006 | 0.34 ± 0.005 | 0.94 ± 0.0 |
| CPoE(4)-SE | 16.35 ± 0.97 | 0.635 ± 0.012 | 0.459 ± 0.006 | 0.34 ± 0.005 | 0.94 ± 0.0 |
| CPoE(1)-FLEX | 130.92 ± 10.31 | 0.687 ± 0.019 | 0.487 ± 0.01 | 0.365 ± 0.009 | 0.96 ± 0.0 |
| CPoE(2)-FLEX | 133.52 ± 10.34 | 0.669 ± 0.016 | 0.48 ± 0.01 | 0.355 ± 0.007 | 0.95 ± 0.0 |
| CPoE(3)-FLEX | 137.64 ± 10.23 | 0.659 ± 0.014 | 0.474 ± 0.009 | 0.35 ± 0.006 | 0.95 ± 0.0 |
| CPoE(4)-FLEX | 141.31 ± 10.28 | 0.655 ± 0.013 | 0.471 ± 0.008 | 0.348 ± 0.006 | 0.94 ± 0.0 |
| MLP(100-100) | 19.19 ± 3.86 | 0.652 ± 0.014 | 0.469 ± 0.008 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| MLP(500-500) | 166.5 ± 17.02 | 0.761 ± 0.012 | 0.53 ± 0.008 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| MLP(100-100-100) | 33.85 ± 2.41 | 0.762 ± 0.017 | 0.527 ± 0.009 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| XGboost | 0.74 ± 0.04 | 0.65 ± 0.011 | 0.466 ± 0.005 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| LinReg | 0.02 ± 0.0 | 0.66 ± 0.009 | 0.488 ± 0.005 | 0.0 ± 0.0 | 0.0 ± 0.0 |

Table F18: Comparison with non-GP methods for dataset *abalone*. Best method (besides full GP) is indicated in bold.

| | time | RMSE | ABSE | CRPS | COV |
|---------------------|----------------|----------------------|----------------------|----------------------|------------|
| fullGP-SE | 178.61 ± 3.76 | 0.267 ± 0.001 | 0.207 ± 0.001 | 0.148 ± 0.001 | 0.94 ± 0.0 |
| fullGP-FLEX | 714.07 ± 18.51 | 0.28 ± 0.001 | 0.215 ± 0.001 | 0.154 ± 0.001 | 0.94 ± 0.0 |
| CPoE(1)-SE | 15.92 ± 0.77 | 0.31 ± 0.001 | 0.239 ± 0.001 | 0.172 ± 0.001 | 0.96 ± 0.0 |
| CPoE(2)-SE | 20.65 ± 0.57 | 0.292 ± 0.001 | 0.226 ± 0.001 | 0.162 ± 0.001 | 0.95 ± 0.0 |
| CPoE(3)-SE | 30.6 ± 1.03 | 0.28 ± 0.001 | 0.218 ± 0.001 | 0.155 ± 0.001 | 0.94 ± 0.0 |
| CPoE(4)-SE | 45.01 ± 1.33 | 0.276 ± 0.001 | 0.215 ± 0.001 | 0.153 ± 0.001 | 0.94 ± 0.0 |
| CPoE(1)-FLEX | 64.24 ± 2.05 | 0.334 ± 0.001 | 0.256 ± 0.001 | 0.185 ± 0.001 | 0.97 ± 0.0 |
| CPoE(2)-FLEX | 69.43 ± 2.02 | 0.315 ± 0.001 | 0.241 ± 0.001 | 0.174 ± 0.001 | 0.96 ± 0.0 |
| CPoE(3)-FLEX | 78.69 ± 2.1 | 0.303 ± 0.002 | 0.232 ± 0.001 | 0.167 ± 0.001 | 0.95 ± 0.0 |
| CPoE(4)-FLEX | 90.09 ± 2.11 | 0.297 ± 0.002 | 0.228 ± 0.001 | 0.164 ± 0.001 | 0.95 ± 0.0 |
| MLP(100-100) | 21.77 ± 1.0 | 0.287 ± 0.001 | 0.223 ± 0.001 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| MLP(500-500) | 100.1 ± 5.9 | 0.284 ± 0.001 | 0.221 ± 0.001 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| MLP(100-100-100) | 33.92 ± 2.19 | 0.299 ± 0.002 | 0.233 ± 0.001 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| XGboost | 0.9 ± 0.05 | 0.667 ± 0.003 | 0.528 ± 0.002 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| LinReg | 0.04 ± 0.0 | 0.765 ± 0.002 | 0.613 ± 0.002 | 0.0 ± 0.0 | 0.0 ± 0.0 |

Table F19: Comparison with non-GP methods for dataset *kin*. Best method (besides full GP) is indicated in bold.

| | time | RMSE | ABSE | CRPS | COV |
|---------------------|----------------|----------------------|----------------------|--------------------|------------|
| CPoE(1)-SE | 73.17 ± 3.12 | 0.099 ± 0.002 | 0.066 ± 0.001 | 0.051 ± 0.001 | 0.95 ± 0.0 |
| CPoE(2)-SE | 107.82 ± 6.0 | 0.1 ± 0.002 | 0.068 ± 0.001 | 0.051 ± 0.001 | 0.93 ± 0.0 |
| CPoE(3)-SE | 197.54 ± 7.42 | 0.099 ± 0.001 | 0.069 ± 0.001 | 0.052 ± 0.0 | 0.92 ± 0.0 |
| CPoE(1)-FLEX | 205.74 ± 5.65 | 0.094 ± 0.002 | 0.062 ± 0.001 | 0.048 ± 0.0 | 0.95 ± 0.0 |
| CPoE(2)-FLEX | 244.57 ± 7.19 | 0.094 ± 0.002 | 0.061 ± 0.001 | 0.048 ± 0.0 | 0.94 ± 0.0 |
| CPoE(3)-FLEX | 346.93 ± 11.4 | 0.092 ± 0.002 | 0.061 ± 0.001 | 0.047 ± 0.0 | 0.94 ± 0.0 |
| MLP(100-100) | 64.83 ± 4.7 | 0.117 ± 0.001 | 0.084 ± 0.001 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| MLP(500-500) | 407.75 ± 17.83 | 0.097 ± 0.001 | 0.069 ± 0.001 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| MLP(100-100-100) | 104.19 ± 8.83 | 0.106 ± 0.001 | 0.076 ± 0.001 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| XGboost | 4.72 ± 0.45 | 0.251 ± 0.002 | 0.183 ± 0.002 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| LinReg | 0.28 ± 0.04 | 0.27 ± 0.003 | 0.193 ± 0.002 | 0.0 ± 0.0 | 0.0 ± 0.0 |

Table F20: Comparison with non-GP methods for dataset *sarcos*. Best method (besides full GP) is indicated in bold.

| | time | RMSE | ABSE | CRPS | COV |
|---------------------|------------------|----------------------|----------------------|----------------------|-------------|
| CPoE(1)-SE | 68.54 ± 1.67 | 0.597 ± 0.006 | 0.381 ± 0.004 | 0.289 ± 0.003 | 0.94 ± 0.0 |
| CPoE(2)-SE | 102.61 ± 1.54 | 0.59 ± 0.007 | 0.38 ± 0.005 | 0.288 ± 0.004 | 0.93 ± 0.01 |
| CPoE(3)-SE | 175.91 ± 3.32 | 0.59 ± 0.009 | 0.382 ± 0.006 | 0.289 ± 0.004 | 0.92 ± 0.01 |
| CPoE(1)-FLEX | 215.81 ± 4.6 | 0.525 ± 0.006 | 0.339 ± 0.003 | 0.261 ± 0.002 | 0.95 ± 0.0 |
| CPoE(2)-FLEX | 895.7 ± 563.48 | 0.522 ± 0.005 | 0.336 ± 0.003 | 0.259 ± 0.002 | 0.94 ± 0.0 |
| CPoE(3)-FLEX | 353.4 ± 4.53 | 0.522 ± 0.005 | 0.336 ± 0.003 | 0.259 ± 0.002 | 0.93 ± 0.0 |
| MLP(100-100) | 397.14 ± 34.0 | 0.591 ± 0.006 | 0.426 ± 0.006 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| MLP(500-500) | 4710.19 ± 810.95 | 0.577 ± 0.015 | 0.399 ± 0.009 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| MLP(100-100-100) | 639.71 ± 44.16 | 0.585 ± 0.006 | 0.406 ± 0.005 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| XGboost | 2.55 ± 0.04 | 0.767 ± 0.007 | 0.615 ± 0.007 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| LinReg | 0.14 ± 0.0 | 0.854 ± 0.009 | 0.714 ± 0.008 | 0.0 ± 0.0 | 0.0 ± 0.0 |

Table F21: Comparison with non-GP methods for dataset *casp*. Best method (besides full GP) is indicated in bold.

| | concrete | mg | space | abalone | kin | cadata | sarcos | casp |
|---------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| fullGP-SE | 0.218 | 0.39 | 0.348 | 0.459 | 0.207 | | | |
| fullGP-FLEX | 0.169 | 0.388 | 0.328 | 0.461 | 0.215 | | | |
| CPoE(1)-SE | 0.236 | 0.39 | 0.371 | 0.459 | 0.239 | 0.323 | 0.066 | 0.381 |
| CPoE(2)-SE | 0.231 | 0.393 | 0.36 | 0.458 | 0.226 | 0.318 | 0.068 | 0.38 |
| CPoE(3)-SE | 0.23 | 0.394 | 0.361 | 0.458 | 0.218 | 0.317 | 0.069 | 0.382 |
| CPoE(1)-FLEX | 0.175 | 0.388 | 0.347 | 0.487 | 0.256 | 0.288 | 0.062 | 0.339 |
| CPoE(2)-FLEX | 0.172 | 0.391 | 0.324 | 0.48 | 0.241 | 0.277 | 0.061 | 0.336 |
| CPoE(3)-FLEX | 0.171 | 0.392 | 0.321 | 0.474 | 0.232 | 0.273 | 0.061 | 0.336 |
| MLP(100-100) | 0.204 | 0.398 | 0.355 | 0.469 | 0.223 | 0.297 | 0.084 | 0.426 |
| MLP(500-500) | 0.208 | 0.397 | 0.353 | 0.53 | 0.221 | 0.312 | 0.069 | 0.399 |
| MLP(100-100-100) | 0.2 | 0.403 | 0.351 | 0.527 | 0.233 | 0.316 | 0.076 | 0.406 |
| XGboost | 0.235 | 0.405 | 0.386 | 0.466 | 0.528 | 0.329 | 0.183 | 0.615 |
| LinReg | 0.492 | 0.509 | 0.485 | 0.488 | 0.613 | 0.439 | 0.193 | 0.714 |

Table F22: Comparison for non-GP methods regarding *ABSE*. Best method (besides full GP) is indicated in bold.

| | concrete | mg | space | abalone | kin | cadata | sarcos | casp |
|---------------------|----------|-------|-------|---------|-------|--------|--------|--------|
| fullGP-SE | 12.5 | 38.9 | 137.8 | 327.6 | 178.6 | | | |
| fullGP-FLEX | 28.2 | 161.6 | 700.3 | 1008.1 | 714.1 | | | |
| CPoE(1)-SE | 2.0 | 2.4 | 11.6 | 8.9 | 15.9 | 84.1 | 73.2 | 68.5 |
| CPoE(2)-SE | 2.4 | 2.8 | 14.9 | 10.4 | 20.6 | 118.2 | 107.8 | 102.6 |
| CPoE(3)-SE | 2.8 | 3.1 | 18.8 | 12.6 | 30.6 | 246.0 | 197.5 | 175.9 |
| CPoE(1)-FLEX | 16.4 | 24.1 | 72.1 | 130.9 | 64.2 | 159.9 | 205.7 | 215.8 |
| CPoE(2)-FLEX | 17.1 | 24.7 | 76.7 | 133.5 | 69.4 | 195.0 | 244.6 | 895.7 |
| CPoE(3)-FLEX | 17.4 | 25.4 | 82.5 | 137.6 | 78.7 | 274.6 | 346.9 | 353.4 |
| MLP(100-100) | 4.5 | 3.8 | 7.2 | 19.2 | 21.8 | 82.9 | 64.8 | 397.1 |
| MLP(500-500) | 10.7 | 12.0 | 33.3 | 166.5 | 100.1 | 925.6 | 407.7 | 4710.2 |
| MLP(100-100-100) | 4.1 | 4.8 | 9.9 | 33.8 | 33.9 | 195.3 | 104.2 | 639.7 |
| XGboost | 32.2 | 0.6 | 0.7 | 0.7 | 0.9 | 1.2 | 4.7 | 2.6 |
| LinReg | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.1 | 0.3 | 0.1 |

Table F23: Comparison for non-GP methods regarding *time* in seconds. Note that, GPs provide predictive uncertainty estimates, which is a much harder task. Therefore, these computational times for competitive accuracy are very reasonable in our opinion.