Leveraging Probabilistic Circuits for Nonparametric Multi-Output Regression

Zhongjie Yu[1] **Mingye Zhu**[2] **Martin Trapp**[3] **Arseny Skryagin**[1] **Kristian Kersting**[1,4]

[1]Department of Computer Science, TU Darmstadt, Darmstadt, Germany
[2]Department of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, China
[3]Department of Computer Science, Aalto University, Espoo, Finland
[4]Centre for Cognitive Science, TU Darmstadt, and Hessian Center for AI (hessian.AI)

Abstract

Inspired by recent advances in the field of expert-based approximations of Gaussian processes (GPs), we present an expert-based approach to large-scale multi-output regression using single-output GP experts. Employing a deeply structured mixture of single-output GPs encoded via a probabilistic circuit allows us to capture correlations between multiple output dimensions accurately. By recursively partitioning the covariate space and the output space, posterior inference in our model reduces to inference on single-output GP experts, which only need to be conditioned on a small subset of the observations. We show that inference can be performed exactly and efficiently in our model, that it can capture correlations between output dimensions and, hence, often outperforms approaches that do not incorporate inter-output correlations, as demonstrated on several data sets in terms of the negative log predictive density.

1 INTRODUCTION

Gaussian processes (GPs) are a popular class of stochastic processes that can be understood as priors over functions. Because of their expressiveness and interpretability—the generalisation properties of a GP are solely determined by choice of the kernel function; they have been heavily used for various machine learning tasks, *e.g.*, for regression or classification tasks [Rasmussen and Williams, 2006]. Moreover, GPs have been shown to be closely related to other machine learning models, *e.g.*, under certain assumptions they correspond to the infinite width limit of Bayesian neural networks [Neal, 1994]. However, exactly computing the posterior distribution of a GP, *i.e.*, conditioning a GP prior on D-dimensional observations, scales cubic in the number of observations (N), *i.e.*, $\mathcal{O}(N^3)$, and has quadratic memory cost, *i.e.*, $\mathcal{O}(N^2 + DN)$, thus, limiting their use to moderately sized data sets.

To enable posterior inference in GPs on large-scale problems, recent work (see *e.g.* Liu et al. [2020] for a detailed review) mainly resorts to global approximations to the posterior, *e.g.*, using inducing points, or local approximations that aim to distribute the computation of the posterior distribution onto local experts. Unfortunately, most of these approaches only focus on single-output regression, *i.e.*, the dependent variable is univariate, and in the case of local approximations, do not easily extend to multi-output regression tasks, see Bruinsma et al. [2020] for a detailed discussion on recent techniques on multi-output GPs.

Recently, Trapp et al. [2020] proposed a local expert-based approximation to GPs that leverages probabilistic circuits (*e.g.* Poon and Domingos [2011], Kisa et al. [2014], Peharz et al. [2020], Choi et al. [2020]), which are a class of deep tractable probabilistic models, allowing them to perform efficient and exact posterior inference in their model. In contrast to popular product-of-experts based approaches (*e.g.* Deisenroth and Ng [2015], Cohen et al. [2020]), their method does not approximate the posterior predictive distribution of a GP directly but instead is a model on its own, making it more suitable for further extensions than product-of-experts based approaches.

The contributions of this work are as follows:

1. We propose the multi-output mixture of Gaussian processes (MOMoGP), an extension of Trapp et al. [2020] for multi-output regression that scales in $\mathcal{O}(KM^3)$, where $M << N$ is the number of observations per expert, and $K \geq D$ is the number of local experts.
2. Moreover, we show that posterior inference in our model can be done exactly and reduces to posterior inference at the GP leaves of the networks.
3. Finally, we present a quantitative evaluation of our approach as well as an application to image upsampling, indicating that MOMoGP is a promising model for

Accepted for the 37th Conference on Uncertainty in Artificial Intelligence (UAI 2021).

multi-output regression.

The rest of the paper is organised as follows. We start by reviewing GP regression and probabilistic circuits in Section 3. In Section 4, we present MOMoGPs for multi-output regression and discuss posterior inference as well as hyperparameter optimisation. Finally, before concluding, we present a quantitative evaluation of the proposed method in Section 5.[1]

2 RELATED WORK

Time Series Regression. Time series regression is a central task in machine learning. We can categorise the existing approach into parametric models, *e.g.*, traditional machine learning approaches such as random forest [Breiman, 2001], Adaboost [Solomatine and Shrestha, 2004], XGboost [Chen and Guestrin, 2016], and multi-layer perceptrons [Murtagh, 1991]; and nonparametric models, with Gaussian process [Rasmussen and Williams, 2006], which are distribution over functions, taking the central role in probabilistic machine learning.

Multi-Output Regression. The methods mentioned above can always be applied to multi-output regression by assuming that all output dimensions are independent. However, simply ignoring the correlations among output dimensions will not lead to an accurate representation of the regression task at hand. One solution is to employ a neural network-based regression model, which can naturally model the output space jointly. However, accurately quantifying the uncertainties and interpreting the modelled dependencies is often challenging or only possible to a limited extend.

Existing methods for multi-output regression can be categorized into two categories: problem transformation methods and algorithm adaptation methods [Borchani et al., 2015]. Problem transformation methods are mainly based on transforming the multi-output regression problem into a single-target problem. Consequently, one aggregates the predictions from each single-target regression task to obtain the multi-output predictions. Single-Target Method, Multi-Target Regressor Stacking, and Regressor Chains [Spyromitros-Xioufis et al., 2012] are among problem transformation methods. Moreover, Zhang et al. [2012] presented a multi-output support vector regression approach based on problem transformation, which extends the original feature space and expresses the multi-output problem as an equivalent single-output problem. On the other hand, algorithm adaptation based methods [Kocev et al., 2009, Breiman and Friedman, 1997, Similä and Tikka, 2007] adapt a specific single-output method to handle multi-output data sets directly. These methods generally achieve better results as they consider the underlying relationships between the features and the corresponding targets and the relationships between the targets. Existing approaches include reduced-rank regression [Izenman, 1975, Abraham et al., 2013], multi-output support vector regression [Tuia et al., 2011, Xu et al., 2013], kernel methods [Baldassarre et al., 2012, Alvarez et al., 2011] and multi-target regression trees [Stojanova et al., 2012, Appice and Malerba, 2014, Levatić et al., 2014].

Williams et al. [2007] proposed a multi-task Gaussian process, where the model learns a shared covariance function on input-dependent features and a “free-form” covariance matrix over tasks. Further, Platanios and Chatzis [2012] presented a nonparametric Bayesian method for multivariate volatility modelling and proposed a mixture of multi-output heteroscedastic GPs to model the covariance matrices of multiple assets. However, this approach is computationally not tractable. More recently, Bruinsma et al. [2020] proposed a linear mixing model, which scales linearly in the number of output dimensions, and showed that their approach could be combined with variational approximations to the GP posterior.

Probabilistic Circuits for Time Series. Probabilistic circuits (PCs) have previously been used for time series modelling. Melibari et al. [2016] proposed dynamic sum-product networks for density estimation of time series, which was later extended to so-called recurrent sum-product networks [Kalra et al., 2018, Duan et al., 2020] by utilizing discriminative learning. Recently, Yu et al. [2021] proposed to model the distribution of time series in the spectral domain, using Whittle sum-product networks. While these approaches are able to model the joint distribution of the time series, they often do not allow for straightforward computation of the predictive distribution.

Trapp et al. [2020] proposed to define PCs for time series modelling in terms of their induced measure and to equip the PC with Gaussian measures induced by local GPs. The resulting model – called deep structured mixture of Gaussian processes (DSMGP) – is a deep mixture of naïve-local experts. While DSMGPs aim to approximate GPs by a deep mixture of GP experts, they are limited to single-output GP regression, hence Trapp et al. [2020] had to resort to a full factorisation for the multi-output regression.

Conversely, MOMoGPs offer a principled way of incorporating multiple output dimensions and model dependencies between outputs through the parameters of the PC. For univariate regression, MOMoGPs reduce to DSMGPs and can therefore be understood as a generalisation of DSMGPs.

3 NOTATION AND BACKGROUND

Notation. We use the following notations throughout the paper. $\mathfrak{D}$ is the data set, where $\mathcal{X}$ is the set of covariates, and $\mathcal{Y}$ is the set of target values. Bold font capitalised $\mathbf{X}$,

[1]Source code is available at: `https://github.com/ml-research/MOMoGP`

and $\mathbf{Y}$ are sets of random variables. $\mathbf{X}$ is the set of random variables in the covariate space (which is an uncountable set) and $\mathbf{Y}$ the set of output random variables (RVs). The input space has dimension D, the output space has dimension P, and the number of observations is N. Furthermore, $\mathbf{x}$ denotes the covariates of one observation and $\mathbf{y}$ the observed multivariate target/output. We use $\mathbf{y}_n$ to denote the n^{th} observed target/output value from the data set $\mathfrak{D}$, y_p for the p^{th} dimension in the output space, and $y_{n,p}$ denotes the p^{th} dimension of the n^{th} observed target/output value.

In a GP, as reviewed below, $k(\cdot,\cdot)$ denotes the covariance function and $\mathbf{K}$ the covariance (Gram) matrix. A single-output GP expert is parameterized by hyperparameters θ_{L}.

In a PC $\mathfrak{C}$, as reviewed in the next section, S represents a sum node, P_x represents a product node that partitions the covariate space, P_y represents a product node that partitions the output space, L is a leaf node, and N denotes a general node. Furthermore, K_{S} denotes the number of children of a sum node, and similarly, K_{P_x}, K_{P_y} denote the numbers of children of a product node. The maximum number of observations per leaf is denoted as M.

3.1 GAUSSIAN PROCESS REGRESSION

A Gaussian process (GP) is defined as an (uncountable) collection of random variables $\mathbf{X}$ indexed by an arbitrary covariate space $\mathbb{R}$, where any finite subset of the RVs is multivariate Gaussian distributed and overlapping finite subsets are marginally consistent [Rasmussen and Williams, 2006]. Moreover, a GP is fully specified by its mean function $m\colon \mathbb{R}^D \mapsto \mathbb{R}$ and its covariance function $k\colon \mathbb{R}^D \times \mathbb{R}^D \mapsto \mathbb{R}$. Throughout the paper, we assume a zero-mean function without loss of generality.

Let us assume a data set $\mathfrak{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^N$ consisting of N observations and denote $\mathcal{X} = \{\mathbf{x}_n\}_{n=1}^N$ and $\mathcal{Y} = \{y_n\}_{n=1}^N$. Then the covariance matrix $\mathbf{K}_{\mathcal{X},\mathcal{X}}$ is given as:

$$\mathbf{K}_{\mathcal{X},\mathcal{X}} = \begin{bmatrix} k(\mathbf{x}_1,\mathbf{x}_1) & k(\mathbf{x}_1,\mathbf{x}_2) & \cdots & k(\mathbf{x}_1,\mathbf{x}_N) \\ k(\mathbf{x}_2,\mathbf{x}_1) & k(\mathbf{x}_2,\mathbf{x}_2) & \cdots & k(\mathbf{x}_2,\mathbf{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_N,\mathbf{x}_1) & k(\mathbf{x}_N,\mathbf{x}_2) & \cdots & k(\mathbf{x}_N,\mathbf{x}_N) \end{bmatrix}. \tag{1}$$

In single-output GP regression tasks, and assuming a Gaussian likelihood model, the posterior mean and posterior covariance for a test datum $\mathbf{x}^*$ can be obtained in closed-form, *i.e.*,

$$m_{\mathfrak{D}}(\mathbf{x}^*) = \mathbf{K}_{\mathbf{x}^*,\mathcal{X}}\mathbf{C}^{-1}\mathcal{Y}\,, \tag{2}$$

and

$$V_{\mathfrak{D}}(\mathbf{x}^*) = \mathbf{K}_{\mathbf{x}^*,\mathbf{x}^*} - \mathbf{K}_{\mathbf{x}^*,\mathcal{X}}\mathbf{C}^{-1}\mathbf{K}_{\mathbf{x}^*,\mathcal{X}}^T\,, \tag{3}$$

where $\mathbf{C} = [\mathbf{K}_{\mathcal{X},\mathcal{X}} + \sigma^2\mathbf{I}]$, with σ^2 denoting the noise variance and $\mathbf{I}$ denoting the identity matrix. However, computing the posterior predictive distribution scales poorly with the number of observations, as the matrix inversion of $\mathbf{C}$ required in the computations has computational cost cubic in N, if solved using a Cholesky decomposition.

3.2 PROBABILISTIC CIRCUITS

Probabilistic circuits (PCs) are tractable probabilistic models, defined as rooted directed acyclic graphs (DAGs), in which leaf nodes represent univariate probability distributions and non-terminal nodes represent either a mixture (or states of an observed variable in case of a deterministic circuit) or an independence relation of their children.

More formally, a PC $\mathfrak{C}$ over a set of RVs $\mathbf{X}$ is a probabilistic model defined via a DAG, also called the computational graph, containing *input distributions* (*leaves*), *sums* S and *products* P; and a scope function $\mathbf{sc}(\cdot)$. We refer to Trapp et al. [2019] for further details.

For a given scope function, all leaves of the PC are density functions over some subset of RVs $\mathbf{U} \subset \mathbf{X}$. This subset is called the scope of the node, and for a node N is denoted as $\mathbf{sc}(\mathsf{N})$. The scope of inner nodes is defined as the union of the scope of its children. Inner nodes compute either a weighted sum of their children or a product of their children, *i.e.*, $\mathsf{S} = \sum_{\mathsf{N}\in\mathbf{ch}(\mathsf{S})} w_{\mathsf{S},\mathsf{N}}\mathsf{N}$ and $\mathsf{P} = \prod_{\mathsf{N}\in\mathbf{ch}(\mathsf{P})}\mathsf{N}$, where $\mathbf{ch}(\cdot)$ denotes the children of a node. The sum weights $w_{\mathsf{S},\mathsf{N}}$ are assumed to be non-negative and normalized, *i.e.*, $w_{\mathsf{S},\mathsf{N}} \geq 0, \sum_{\mathsf{N}} w_{\mathsf{S},\mathsf{N}} = 1$, without loss of generality [Peharz et al., 2015]. Further, we assume the PC to be smooth (complete) and decomposable [Darwiche, 2003]. Specifically, a PC is *smooth* if for each sum S it holds that $\mathbf{sc}(\mathsf{N}') = \mathbf{sc}(\mathsf{N}'')$, for all $\mathsf{N}', \mathsf{N}'' \in \mathbf{ch}(\mathsf{S})$. And the PC is called *decomposable* if it holds for each product P that $\mathbf{sc}(\mathsf{N}') \cap \mathbf{sc}(\mathsf{N}'') = \emptyset$, for all $\mathsf{N}' \neq \mathsf{N}'' \in \mathbf{ch}(\mathsf{P})$.

Note that PCs are typically defined only for a finite set of RVs, while Trapp et al. [2020] showed that it is possible to extend PCs to the stochastic process case by defining them based on their induced measure. We will, therefore, follow the approach in [Trapp et al., 2020] and recursively define our model as such.

4 MULTI-OUTPUT MIXTURE OF GAUSSIAN PROCESSES

Now we have everything at hand to introduce our MOMoGP model formally. We start off with the problem formulation, introduce our MOMoGP model, and subsequently show how to perform exact posterior inference as well as how to perform predictions using MOMoGPs. Finally, we discuss hyperparameter optimisation in MOMoGPs by maximising the marginal likelihood.

4.1 PROBLEM FORMULATION

Given a set of observations $\mathfrak{D} = \{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^N$ with covariates $\boldsymbol{x}_n \in \mathbb{R}^D$ and noisy target values $\boldsymbol{y}_n \in \mathbb{R}^P$, we aim to infer the latent functions:

$$f_p \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}) , \tag{4}$$

$$y_p \mid f_p \sim N(f_p(\boldsymbol{x}), \sigma^2 \mathbf{I}) , \tag{5}$$

while aiming to account for the correlations between the target values. One approach is to model the multi-output targets by adopting a multi-valued latent function. However, such an approach scales cubic in the number of output dimensions P, *i.e.*, $\mathcal{O}(N^3P^3)$ [Bruinsma et al., 2020]. Alternatively, we exploit a simple observation, that is, we can leverage a mixture of independent GP estimators to capture correlations between the output dimensions. This is akin to the Instantaneous Linear Mixing Model by Bruinsma et al. [2020] but explores recent work by Trapp et al. [2020] to perform efficient and exact posterior inference in a deep mixture over single-output GP experts to obtain correlated multi-output predictions. Note that the approach by Bruinsma et al. [2020] assumes that the outputs live on a low-dimensional linear subspace and exploits an orthogonal basis for efficient inference, while our approach assumes weak independence between output dimensions and exploits the multi-modality of the posterior to capture dependencies between output dimensions.

4.2 MULTI-OUTPUT MIXTURE OF GPS

The multi-output mixture of Gaussian processes (MOMoGP) can be recursively defined as follows:

1. A Gaussian measure induced by a GP [Rajput and Cambanis, 1972] is a MOMoGP,
2. a product of MOMoGPs with disjoint covariate space or disjoint output space is a MOMoGP, and
3. a convex combination of MOMoGPs over the same covariate and output space is a MOMoGP.

The recursive definition of a MOMoGP is illustrated in Fig. 1. Here, $w_{1,1}, \cdots, w_{1,k}, \cdots, w_{1,K_\mathsf{S}}$ are K_S many normalized weights of the sum node S. The product node P_x, which is a child of the root, splits the (input) covariate space $\mathbf{X}$, by assuming certain regions of the input space to be approximately independent. The second product node, P_y, partitions the output space $\mathbf{Y}$ into disjoint subsets. More specifically, its children are either MOMoGPs over a set of output variables, *e.g.*, $\mathbf{Y}_1^l$, or only a single variable, *e.g.*, Y_{l+1}. For a univariate output, *e.g.*, Y_{l+1} we construct a GP leaf L on the respective covariate subspace and output subspace. Otherwise, the process recurses by appending a new sum node, resulting in a deep hierarchical structure.

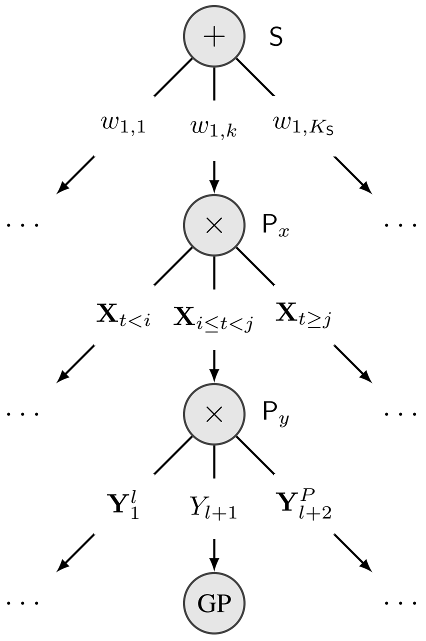


Figure 1: Illustration of the MOMoGP structure. $w_{1,k}$ represents for the normalized weight, $\mathbf{X}_{i\le t<j} \subset \mathbf{X}$ represents the subset of RVs X_t with index $i \le t < j$ of the covariate space, and $\mathbf{Y}_1^l$ denotes a subset of RVs of the output space, respectively. Note that we randomly permute the index set of the output space at each product node P_y.

4.3 MOMOGP CONSTRUCTION

The structure of a MOMoGP can either be manually defined or learned from data. To learn it from data, one can leverage Algorithm 1. In short, we alternate between introducing sum and product nodes and, finally, append GP experts as leaves once any of the termination criteria is fulfilled.

As illustrated in Fig. 2, to create a sum node S, we first construct K_S many children under the sum and then attach those children with uniform weights. In the next step, a product node P_x is constructed by partitioning the covariate space. For the k^{th} product node, the covariate space is partitioned by the K_{P_x}-quantiles of the dimension with the k^{th} largest sample variance. Afterwards, a product node P_y is created by randomly partitioning the output space. To split the output space one can also apply a conditional independence test on $\mathbf{y}$ instead of random splitting. Product nodes either enforce independence assumptions in the covariate space (resulting in weak discontinuities) or independence assumptions in the output space (assuming sets of the dependent variables are conditionally independent). The above sum and product nodes are constructed recursively until the number of observations in the subspace is smaller than a predefined threshold M. If the output space is still multidimensional, it will be directly factorized with a product node P_y. Finally, we construct leaf nodes by placing single-output GP experts at the respective covariate subspace parameterized

Algorithm 1: Construction of a MOMoGP

```
Input: $\mathbf{X}, \mathbf{Y}, \mathfrak{D}, K_{\mathsf{S}}, K_{\mathsf{P}x}, K_{\mathsf{P}y}, M$    Output: $\mathfrak{C}$
Function buildGP($\mathbf{X}, Y, \mathfrak{D}$)
    Equip $\mathsf{L}$ with a single output GP expert on the
      domain $\mathbf{X}$ and output space $Y$ ;
    Condition $\mathsf{L}$ on $\mathfrak{D}$ ;
    return $\mathsf{L}$
Function buildSumNode($\mathbf{X}, \mathbf{Y}, \mathfrak{D}$)
    $w \leftarrow \{\frac{1}{K_{\mathsf{S}}}\}_{k=1}^{K_{\mathsf{S}}}$;
    Let $d_1, \dots, d_D$ represent the dimensions of the
      covariate space in increasing order of their sample
      variance in $\mathfrak{D}$;
    for $k = 1, \dots, K_{\mathsf{S}}$ do
        $\mathsf{P}_x \leftarrow$ buildProdNodeX($\mathbf{X}, \mathbf{Y}, \mathfrak{D}, d_k$) ;
        $\mathsf{S} \leftarrow \mathsf{S} + w_k \, \mathsf{P}_x$;
    return $\mathsf{S}$
Function buildProdNodeX($\mathbf{X}, \mathbf{Y}, \mathfrak{D}, d$)
    $l \leftarrow$ lower bound of $\mathbf{X}$ for dimension $d$;
    $u \leftarrow$ upper bound of $\mathbf{X}$ for dimension $d$;
    $v \leftarrow u - l$ ;
    $s_1, \dots, s_{K_{\mathsf{P}x}-1} \leftarrow K_{\mathsf{P}x}$-quantiles of the interval
      $[l, u]$;
    $\tilde{l} \leftarrow l$ ;
    for $k = 1, \dots, K_{\mathsf{P}x} - 1$ do
        $\tilde{u} \leftarrow s_k$ ;
        $\tilde{\mathbf{X}} \leftarrow$ sub-domain of $\mathbf{X}$ such that upper and
          lower bounds for $d$ are equal to $\tilde{u}, \tilde{l}$,
          respectively. ;
        $\tilde{\mathfrak{D}} \leftarrow$ subset of $\mathfrak{D}$ such that the covariate of
          every $(\boldsymbol{x}_n, \boldsymbol{y}_n) \in \tilde{\mathfrak{D}}$ is defined on $\tilde{\mathbf{X}}$. ;
        $\mathsf{P}_y \leftarrow$ buildProdNodeY($\tilde{\mathbf{X}}, \mathbf{Y}, \tilde{\mathfrak{D}}$) ;
        $\mathsf{P} \leftarrow \mathsf{P} \times \mathsf{P}_y$ ;
        $\tilde{l} \leftarrow s_k$ ;
    return $\mathsf{P}$
Function buildProdNodeY($\mathbf{X}, \mathbf{Y}, \mathfrak{D}$)
    if Number of observations in $\mathfrak{D} > M$ then
        $\mathbf{Y}_1, \cdots, \mathbf{Y}_{K_{\mathsf{P}y}-1} \leftarrow$ random partitions of
          output space $\mathbf{Y}$;
        for $k = 1, \dots, K_{\mathsf{P}y} - 1$ do
            $\mathsf{S} \leftarrow$ buildSumNode($\mathbf{X}, \mathbf{Y}_k, \mathfrak{D}$) ;
            $\mathsf{P} \leftarrow \mathsf{P} \times \mathsf{S}$ ;
    else
        $Y_1, \cdots, Y_{K_{\mathsf{P}y}-1} \leftarrow$ each dimension of output
          space $\mathbf{Y}$;
        for $k = 1, \dots, K_{\mathsf{P}y} - 1$ do
            $\mathsf{L} \leftarrow$ buildGP($\mathbf{X}, Y_k, \mathfrak{D}$) ;
            $\mathsf{P} \leftarrow \mathsf{P} \times \mathsf{L}$ ;
    return $\mathsf{P}$
$\mathfrak{C} \leftarrow$ buildSumNode($\mathbf{X}, \mathbf{Y}, \mathfrak{D}$) ;
```

by hyperparameters θ_{L}.

4.4 EXACT POSTERIOR INFERENCE

Both PCs and GPs allow for exact posterior inference, likewise, we can formalize the exact posterior inference for MOMoGP as follows:

(Leaf node) If the MOMoGP is a leaf node L we can obtain the posterior distribution analytically, assuming a Gaussian likelihood.

(Sum node) If the MOMoGP is a sum node S, the likelihood terms distribute to the children, *i.e.*,

$$p_{\mathsf{S}}(f \mid \mathfrak{D}) \propto \prod_{(\mathbf{x}_n, \mathbf{y}_n) \in \mathfrak{D}} p(\mathbf{y}_n \mid f_n) \sum_{\mathsf{N} \in \mathrm{ch}(\mathsf{S})} w_{\mathsf{S},\mathsf{N}} \, p_{\mathsf{N}}(f_n \mid \mathbf{x}_n)$$
$$= \sum_{\mathsf{N} \in \mathrm{ch}(\mathsf{S})} w_{\mathsf{S},\mathsf{N}} \underbrace{\prod_{(\mathbf{x}_n, \mathbf{y}_n) \in \mathfrak{D}} p(\mathbf{y}_n \mid f_n) \, p_{\mathsf{N}}(f_n \mid \mathbf{x}_n)}_{= p_{\mathsf{N}}(f \mid \mathfrak{D})}, \quad (6)$$

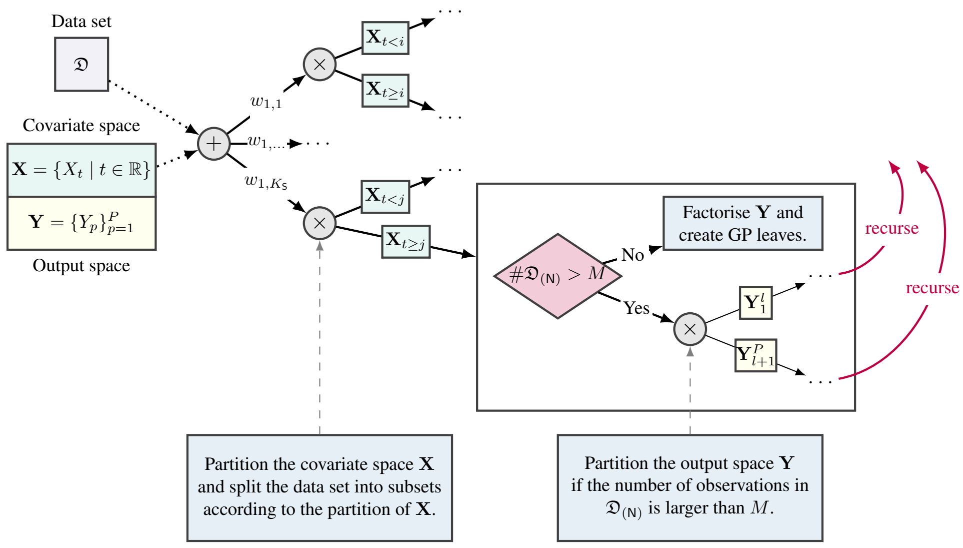
simplifying the computation of the unnormalised posterior distribution. Note that instead of being univariate as in Trapp et al. [2020], $\mathbf{y}_n$ is multidimensional in MOMoGP.

(Product node) If the MOMoGP is a product node P decomposing either the covariate space or the output space, we obtain:

$$p_{\mathsf{P}}(f \mid \mathfrak{D}) \propto \prod_{(\mathbf{x}_n, \mathbf{y}_n) \in \mathfrak{D}} p(\mathbf{y}_n \mid f_n) \prod_{\mathsf{N} \in \mathrm{ch}(\mathsf{P})} p_{\mathsf{N}}(f_n \mid \mathbf{x}_n)$$
$$= \prod_{\mathsf{N} \in \mathrm{ch}(\mathsf{P})} \left(\underbrace{\prod_{(\mathbf{x}_n, \mathbf{y}_n) \in \mathfrak{D}_{(\mathsf{N})}} p(\mathbf{y}_n \mid f_n) \, p_{\mathsf{N}}(f_n \mid \mathbf{x}_n)}_{= p_{\mathsf{N}}(f \mid \mathfrak{D}_{(\mathsf{N})})} \right), \quad (7)$$

for the computation of the unnormalised posterior, where $\mathfrak{D}_{(\mathsf{N})}$ denotes the set of observations in the subspace for which N is an expert of. In the case of the covariate space decomposition (P_x), $\mathfrak{D}_{(\mathsf{N})} = \{(\mathbf{x}_n, \mathbf{y}_n) \in \mathfrak{D} \,|\, \mathbf{x}_n \in \mathcal{X}_{\mathsf{N}}\}$, where $\mathcal{X}_{\mathsf{N}}$ is the subset of covariates falling into the subspace at node N. While for the output space decomposition (P_y), $\mathfrak{D}_{(\mathsf{N})}$ contains a subset of outputs for each observation, *i.e.*, $\mathfrak{D}_{(\mathsf{N})} = \Big\{(\mathbf{x}_n, \mathbf{y}_n) \in \mathfrak{D} \,|\, \mathbf{y}_n \in \mathcal{Y}_{\mathsf{N}}^{P_{\mathsf{N}}}, \mathcal{Y}_{\mathsf{N}}^{P_{\mathsf{N}}} \subseteq \mathcal{Y}_{\mathsf{N}}, P_{\mathsf{N}} < P\Big\}$. Therefore, $\mathfrak{D}_{(\mathsf{N})}$ will either contain fewer observations, *i.e.*, $\#\mathfrak{D}_{(\mathsf{N})} < \#\mathfrak{D}$, or fewer output dimensions, *i.e.*, for each n we have $\mathbf{y}_n \in \mathfrak{D}_{(\mathsf{N})}$ with $\mathbf{y}_n \in \mathbb{R}^{P_{\mathsf{N}}}$ and $P_{\mathsf{N}} < P$. In contrast to Trapp et al. [2020], product nodes in MOMoGPs partition either the covariate space or the output space, while DSMGPs [Trapp et al., 2020] only partition the covariate space and assume the output space to be univariate.

To obtain the normalised posterior distribution of a MOMoGPs we employ the re-normalisation algorithm by Peharz et al. [2015].



Data set
$\mathfrak{D}$
Covariate space
$\mathbf{X} = \{X_t \mid t \in \mathbb{R}\}$
$\mathbf{Y} = \{Y_p\}_{p=1}^P$
Output space
$w_{1,1}$
$w_{1,\dots}$
w_{1,K_S}
$\mathbf{X}_{t<i}$
$\mathbf{X}_{t\geq i}$
$\mathbf{X}_{t<j}$
$\mathbf{X}_{t\geq j}$
$\#\mathfrak{D}_{(\mathsf{N})} > M$
No
Yes
Factorise $\mathbf{Y}$ and create GP leaves.
$\mathbf{Y}_1^l$
$\mathbf{Y}_{l+1}^P$
recurse
recurse
Partition the covariate space $\mathbf{X}$ and split the data set into subsets according to the partition of $\mathbf{X}$.
Partition the output space $\mathbf{Y}$ if the number of observations in $\mathfrak{D}_{(\mathsf{N})}$ is larger than M.

Figure 2: Illustration of Algorithm 1 for learning MOMoGPs in a recursive fashion. Sum nodes have weighted children that are product nodes. Product nodes either enforce independence assumptions in the covariate space (resulting in discontinuities) or independence assumptions in the output space (assuming sets of the dependent variables are independent). Sum nodes replace the independence assumptions made by product nodes through conditional independence. Finally, leaf nodes are single-output GP experts at the respective covariate subspace. (Best viewed in color)

4.5 PREDICTIONS

The posterior predictive distribution of a MOMoGP for an unseen datum $\mathbf{x}^*$ is a mixture of multivariate Gaussian distributions. To obtain a single prediction for $\mathbf{x}^*$, we employ an approximation to the posterior predictive distribution of the MOMoGP using the first and the second moment. By this, we approximate the posterior predictive distribution with its closest multivariate Gaussian measured in Kullback–Leibler divergence [Rasmussen and Williams, 2006].

Let $\mathfrak{L}$ be the set of all leaves in a MOMoGP, and $\tau_i\colon \mathbb{R}^D \mapsto \mathfrak{L}$ a function which maps an unseen datum $\mathbf{x}^*$ to a leaf L for each induced tree. Then the posterior mean and variance given $\mathbf{x}^*$, are propagated bottom-up as follows. Let $m_{\tau_i(\mathbf{x}^*)}(\mathbf{x}^*)$ and $V_{\tau_i(\mathbf{x}^*)}(\mathbf{x}^*)$ denote the mean and variance of the posterior distribution from the GP at leaf $\tau_i(\mathbf{x}^*)$, respectively. Now, a product node P_y that partitions the output space performs concatenation of the mean and variance from its children:

$$m_{\mathsf{P}}(\mathbf{x}^*) = [m_{\mathsf{N}_1}(\mathbf{x}^*), \cdots, m_{\mathsf{N}_k}(\mathbf{x}^*)]\,, \qquad (8)$$

$$\text{and } V_{\mathsf{P}}(\mathbf{x}^*) = \operatorname{diag}(V_{\mathsf{N}_1}(\mathbf{x}^*), \cdots, V_{\mathsf{N}_k}(\mathbf{x}^*))\,. \qquad (9)$$

In contrast, a product node P_x that partitions the covariate space acts as a gate, *i.e.*, we select exactly one child of the product node. The selection of the child is dictated by the partition of covariate space employed by the product node and the location of $\mathbf{x}^*$. For a sum node S, it computes the first moment, *i.e.*,

$$m_{\mathsf{S}}(\mathbf{x}^*) = \sum_{\mathsf{N}\in\mathbf{ch}(\mathsf{S})} w_{\mathsf{S},\mathsf{N}}\, m_{\mathsf{N}}(\mathbf{x}^*)\,, \qquad (10)$$

and the second moment ,*i.e.*,

$$V_{\mathsf{S}}(\mathbf{x}^*) = \sum_{\mathsf{N}\in\mathbf{ch}(\mathsf{S})} w_{\mathsf{S},\mathsf{N}} V_{\mathsf{N}}(\mathbf{x}^*) + \sum_{\mathsf{N}\in\mathbf{ch}(\mathsf{S})} w_{\mathsf{S},\mathsf{N}}\, m_{\mathsf{N}}(\mathbf{x}^*)^T\, m_{\mathsf{N}}(\mathbf{x}^*) - m_{\mathsf{S}}(\mathbf{x}^*)^T\, m_{\mathsf{S}}(\mathbf{x}^*)\,, \qquad (11)$$

of the associated mixture distribution.

4.6 HYPERPARAMETER OPTIMISATION

To optimise the hyperparameters of a MOMoGP model, we can maximize its log marginal likelihood. When using a formulation in terms of a mixture over induced trees $\mathcal{T}$ and following the argument from Section 4.4, we can see that the marginal likelihood of a MOMoGP is obtained as follows:

$$\begin{aligned} &p(\mathcal{Y}\,|\,\mathcal{X},\theta) \\ &= \int \prod_{(\mathbf{x},\mathbf{y})\in\mathfrak{D}} \sum_{k=1}^{K} p(\mathcal{T}_k) \prod_{p=1}^{P} \prod_{\mathsf{L}\in\mathcal{T}_{k,V}} p(y_p\,|\,\mathbf{x},\theta_{\mathsf{L}}) \mathbb{1}_{\{Y_p,\mathsf{L}\}}\, \mathbf{df} \\ &= \int \sum_{k=1}^{K} p(\mathcal{T}_k) \prod_{p=1}^{P} \prod_{\mathsf{L}\in\mathcal{T}_{k,V}} \prod_{(\mathbf{x},y_p)\in\mathfrak{D}_{(\mathsf{L})}} p(y_p\,|\,\mathbf{x},\theta_{\mathsf{L}}) \mathbb{1}_{\{Y_p,\mathsf{L}\}}\, \mathbf{df} \\ &= \sum_{k=1}^{K} p(\mathcal{T}_k) \prod_{p=1}^{P} \prod_{\mathsf{L}\in\mathcal{T}_{k,V}} \underbrace{\int \prod_{(\mathbf{x},y_p)\in\mathfrak{D}_{(\mathsf{L})}} p(y_p\,|\,\mathbf{x},\theta_{\mathsf{L}}) \mathbb{1}_{\{Y_p,\mathsf{L}\}}\, \mathbf{df}}_{=p_{Y_p}(y_p\,|\,\mathcal{X}_{\mathsf{L}},\theta_{\mathsf{L}})}\,, \end{aligned} \qquad (12)$$

where $\mathfrak{D}_{(\mathsf{L})} = \{(\mathbf{x}, y_p) \in \mathfrak{D}\,|\,\mathbf{x} \in \mathcal{X}_{\mathsf{L}}\}$, with $\mathcal{X}_{\mathsf{L}}$ being the subset of covariates falling into the subspace at leaf L, and

Data set	N(train)	N(test)	D	P
Parkinsons	4,112	1,763	16	2
scm20d	7,173	1,793	61	16
WindTurbine	4,000	1,000	8	6
Energy	57,598	14,400	32	17
usFlight	500,000	200,000	8	2

Table 1: Statistics of the multi-output benchmark data sets used in our evaluation. A large variety of output dimensions P were chosen, ranging from 2 to 17.

$\mathbb{1}_{\{Y_p,\mathsf{L}\}} = \mathbb{1}_{\{Y_p \in \mathbf{sc}(\mathsf{L})\}}$.

We can now readily obtain the log marginal likelihood, which is given as: $\log p(\mathcal{Y} \mid \mathcal{X}, \theta) =$

$$\mathrm{L}\overset{K}{\underset{k=1}{\Sigma}}\mathrm{E}\left(\log p(\mathcal{T}_k) + \sum_{p=1}^{P}\sum_{\mathsf{L}\in\mathcal{T}_{k,V}} \log p_{Y_p}(y_p \mid \mathcal{X}_\mathsf{L}, \theta_\mathsf{L})\right), \tag{13}$$

where $\mathrm{L}\overset{K}{\underset{k=1}{\Sigma}}\mathrm{E}$ denotes the log-sum-exp operation and the marginal log likelihood of a GP expert is given as:

$$\log p_{Y_p}(y_p \mid \mathcal{X}, \theta) = -\frac{1}{2}(y_p^{\mathrm{T}}\mathbf{C}^{-1}y_p + \log|\mathbf{C}| + N\log 2\pi), \tag{14}$$

where $\mathbf{C} = [\mathbf{K}_{\mathcal{X},\mathcal{X}} + \sigma^2\mathbf{I}]$ and N is the number of observations the GP expert is conditioned on.

Note that we have assumed that each GP expert has its own hyperparameters θ_L. Doing so allows us to capture non-stationarities and heteroscedasticity, while potentially increasing the risk of overfitting [Zhang and Williamson, 2019].

If the underlying process is believed to be stationary, it is possible to tie the hyperparameters either by using one set of global hyperparameters for each output dimension or by adopting the approach described in Trapp et al. [2020] and use a similarity matrix to incorporate dependence between the otherwise independent local experts. Note that we consistently used independent hyperparameters for each GP expert in our experiments.

5 EXPERIMENTAL EVALUATION

In this section, we will examine the performance of MOMoGP on several benchmark data sets and compare it with other state-of-the-art approaches. First, we describe the data sets and then provide details about the experimental setup and evaluation measures we used. Finally, we discuss the experimental results obtained.

5.1 DATA SETS

We validate our model on several benchmark data sets for multi-output regression. The number of observations in the data sets varies from 4k to 500k, with output space dimensions from 2 to 17. The statistics of the data sets used in the evaluation are given in Table 1. The Parkinsons and usFlight data sets, as well as their training/test splits, are from Trapp et al. [2020]. Note that we applied Principal Component Analysis (PCA) [Wold et al., 1987] on the scm20d data set to reduce its input dimension from 61 to 30. Both MOMoGPs and DSMGPs recursively partition the covariate space using axis-aligned splits. Therefore, applying PCA to high-dimensional covariate spaces is essential for computational reasons for both approaches. For the Energy data set, we select its subset "Adelaide" for our experiments. The WindTurbine data set is simulated with the FAST simulator.[2]

5.2 EXPERIMENTAL PROTOCOL

To construct a MOMoGP for each experiment, we implemented Algorithm 1. More specifically, the root node of our hierarchy structure was a sum node, with K_S product nodes as children, initialized with uniform weights. Product nodes that split the input space had K_{P_x} children, and those that decompose the output space used a random split strategy to obtain K_{P_y} children. The structure construction terminated with GP leaves when the output space was completely decomposed and the number of observations in the subspace is smaller than a predefined threshold of M.

In the experiments, we set $K_S = 2$, $K_{\mathsf{P}_y} = 2$, and $M \in \{500, 1000, 5000\}$ based on the size of data set. Each GP leaf was equipped with a Matérn-$3/2$ covariance function with Automatic Relevance Detection (ARD), and a zero-mean function. The initialized lengthscale parameters of the GPs were randomly sampled. The learning rate and the number of training epochs were tuned to speed up the training, and at the same time, avoid overfitting. That is, when the training loss reached a plateau, the optimisation terminated. We used Adam to maximize the marginal likelihood of the GP leaves.

The hyperparameters θ_L of all GPs in our experiments were sampled from a Gamma distribution $\Gamma(2, 3)$. Note that we kept consistent settings for all the GP-related approaches, *e.g.*, covariance and mean functions. For MOSVGP, the number of inducing points was set $Q = 500$, and the number of latent functions corresponded to the number of output dimensions.

Additionally, we tested our hypothesis that a mixture of independent single-output GPs can model correlations between the outputs by employing a shallow mixture of exact single-output GPs denoted as sumGP. In fact, sumGPs are a sub-

[2] `https://www.nrel.gov/wind/nwtc/fast.html`

Data Set		LR	GP	MOGP	MOSVGP	DSMGP	sumGP	**MOMoGP**
Parkinsons	RMSE	0.974	0.783	0.793	0.864	**0.774**	0.784	0.775↓
	MAE	0.816	0.610	**0.603**	0.708	0.604	0.610	0.605↓
	NLPD	2.787	2.389	**1.766**	2.515	2.319	2.388	2.208↑
scm20d	RMSE	0.680	0.332	**0.307**	0.491	0.323	0.322	0.324↓
	MAE	0.503	0.195	**0.180**	0.352	0.188	0.187	0.187↑
	NLPD	16.568	0.543	6.882	8.682	1.341	8.097	**-0.201**↑
WindTurbine	RMSE	0.391	**0.133**	0.139	0.302	0.143	**0.133**	0.143
	MAE	0.311	0.074	0.080	0.236	**0.073**	0.074	**0.073**
	NLPD	1.435	-2.649	2.594	0.104	**-8.749**	-2.627	-7.467↓
Energy	RMSE	0.752	NA	NA	0.659	**0.547**	NA	0.556↓
	MAE	0.605	NA	NA	0.516	0.400	NA	**0.398**↑
	NLPD	14.775	NA	NA	14.169	11.745	NA	**8.610**↑
usFlight	RMSE	0.983	NA	NA	0.955	**0.927**	NA	0.934↓
	MAE	0.529	NA	NA	0.494	**0.492**	NA	0.505↓
	NLPD	2.331	NA	NA	2.251	2.178	NA	**2.091**↑

Table 2: Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Negative Log Predictive Density (NLPD) of state-of-the-art approaches and MOMoGP (our work) on benchmark data sets with 4K to 500K observations. Smaller values are better. The best result is indicated in **bold** and comparison to the DSMGP is indicated using arrows ↑ / ↓.

class of MOMoGPs which contain only sum nodes, product nodes which split the output space, and GP leaves.

For quantitative evaluation, we compared the Root Mean Squared Error (RMSE):

$$\mathrm{RMSE}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{P} \sum_{p=1}^{P} \sqrt{\frac{1}{N} \sum_{n=1}^{N} (y_{n,p} - \hat{y}_{n,p})^2}\,, \quad (15)$$

the Mean Absolute Error (MAE):

$$\mathrm{MAE}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{N \cdot P} \sum_{n=1}^{N} \sum_{p=1}^{P} |y_{n,p} - \hat{y}_{n,p}|\,, \quad (16)$$

and the Negative Log Predictive Density (NLPD):

$$\mathrm{NLPD}(\mathbf{y}_n) = -\log p(\mathbf{y}_n \mid \mathfrak{D}, \mathbf{x}_n, \theta)\,, \quad (17)$$

where $\hat{\mathbf{y}}_n$ is the ground truth of prediction $\mathbf{y}_n$ for datum $\mathbf{x}_n$.

5.3 EXPERIMENT RESULTS

Herein, we investigate the performance of seven different regression models, including linear regression (LR), exact GP (GP), deep structured mixtures of GPs (DSMGP) [Trapp et al., 2020], which all employ independence assumption for the output space, and Multitask GP regression (MOGP) [Williams et al., 2007], multi-output sparse Variational GP (MOSVGP) [Moreno-Muñoz et al., 2018], sumGP and MOMoGP, which models the output space jointly. Note that we use consistent settings for all the GP approaches.

Table 2 reports the RMSE, MAE and NLPD on each data set. Generally, the multi-output models provide smaller RMSE and MAE values compared with the single-output models. This means by modelling the joint of the output space, instead of assuming them to be independent, the models achieve a smaller approximation error. Moreover, MOMoGP captures predictive uncertainties better than expert-based approaches and DSMGP, resulting in lower NLPDs. Note that the NLPD gives rise to the output distribution, while the RMSE and the MAE only account for the mean value of the distribution. Thus, improvements in terms of the NLPD are strictly more important than in terms of the RMSE or the MAE. Additionally, MOMoGP provided the lowest NLPD values for large-scale data sets such as Energy and usFlight, and the corresponding RMSE and MAE values are also very competitive.

Overall, we can conclude that MOMoGP can achieve competitive regression results, and provides better predictive uncertainties at the same time.

5.4 EXTRA RESULTS ON IMAGE UPSAMPLING

To deepen the performance evaluation, we envisioned a MOMoGP application to the field of image upsampling. The horizontal and vertical locations of a pixel form the input space, while the RGB channels of the pixel are defined as the outputs. Therefore, we have $D = 2$ and $P = 3$ for the image upsampling task. For a given image, this task aims at enlarging the image via interpolation. The posterior mean from a MOMoGP is taken as the new pixel value, given the

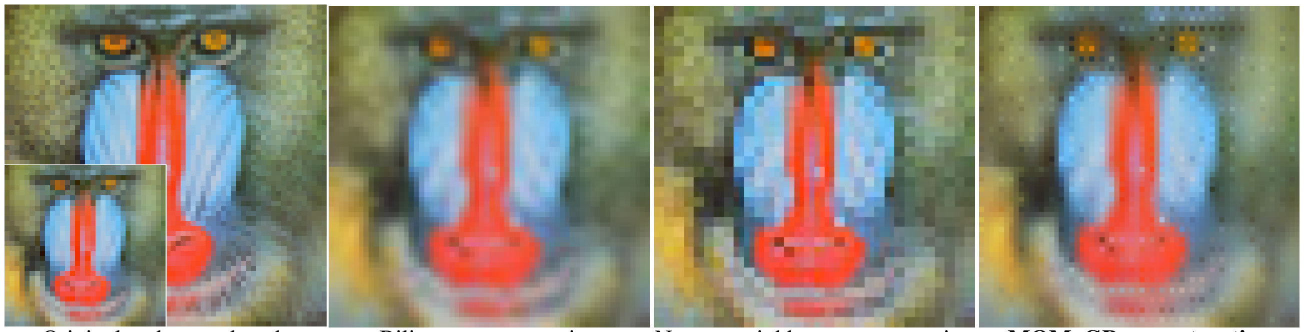

Original and ground truth
Bilinear reconstruction
Nearest neighbour reconstruction
MOMoGP reconstruction

Figure 3: Image upsampling using MOMoGP and other methods. MOMoGP obtains the best reconstruction with an RMSE of 1.289, while Bilinear has RMSE of 1.542 and Nearest neighbour of 1.855. (Best viewed in color)

location of the pixel to be interpolated. In this experiment, the original image has the size of 64×64 and was downsampled to 32×32. We then aim to reconstruct the original image from the downsampled version. For MOMoGP, we set $K_S = 2$, $K_{\mathsf{P}_x} = 2$, $K_{\mathsf{P}_y} = 2$ and $M = 256$. As visualized in Fig. 3, bilinear interpolation produces smooth and blurry artefacts. The nearest neighbour approach brings blocks in the image. MOMoGP as an interpolation approach achieves the best performance, exhibiting a more appropriate balance between colour flattening and salient edge.

6 CONCLUSION

We introduced the multi-output mixture of Gaussian processes (MOMoGP), which leverages a probabilistic circuit with single-output Gaussian process (GP) expert leaves to model correlations between the dependent variables. In comparison to [Trapp et al., 2020] and other expert-based approaches, our approach models the output space jointly, *i.e.*, our model is more general than the approach by [Trapp et al., 2020], by utilising a recursive decomposition of the output space. We have shown that this additional decomposition enables MOMoGPs to retain exact posterior inference while also allowing the model to capture dependencies between the outputs without introducing a cubic cost in the number of output dimensions. In particular, we have shown that we can efficiently approximate the predictive posterior distribution for an unseen datum with its closest multivariate Gaussian distribution through an extension of the approach proposed by Trapp et al. [2020]. Finally, we show that MOMoGPs provide competitive results for both RMSE and MAE and often outperform the model by Trapp et al. [2020] as well as multi-output sparse variational GPs [Moreno-Muñoz et al., 2018] in terms of the NLPD, indicating that MOMoGPs provide a better estimate of the target distribution.

Our work provides several avenues for future work, *e.g.*, exploiting the spectral representation of stationary GPs [Rasmussen and Williams, 2006], combining MOMoGPs with conditional sum-product networks for mixed multi-output regression [Shao et al., 2020] and applying MOMoGPs to multi-task Bayesian optimisation [Swersky et al., 2013].

Acknowledgements

The authors thank the anonymous reviewers for their valuable feedback. This work was supported by the Federal Ministry of Education and Research (BMBF; project “MADESI”, FKZ 01IS18043B, and Competence Center for AI and Labour; “kompAKI”, FKZ 02L19C150), the Hessian Ministry of Higher Education, Research, Science and the Arts (HMWK; projects “The Third Wave of AI” and “The Adaptive Mind”), the Hessian research priority programme LOEWE within the project “WhiteBox”, and the National Research Center for Applied Cybersecurity ATHENE, a joint effort of BMBF and HMWK. M.T. acknowledges funding from the Academy of Finland (grant number 324345).

References

Zubin Abraham, Pang-Ning Tan, Julie Winkler, Shiyuan Zhong, Malgorzata Liszewska, et al. Position preserving multi-output prediction. In *Proceedings of ECML/PKDD*, pages 320–335, 2013.

Mauricio A Alvarez, Lorenzo Rosasco, and Neil D Lawrence. Kernels for vector-valued functions: A review. *arXiv preprint arXiv:1106.6251*, 2011.

Annalisa Appice and Donato Malerba. Leveraging the power of local spatial autocorrelation in geophysical interpolative clustering. *Data Mining and Knowledge Discovery*, 28(5-6):1266–1313, 2014.

Luca Baldassarre, Lorenzo Rosasco, Annalisa Barla, and Alessandro Verri. Multi-output learning via spectral filtering. *Machine learning*, 87(3):259–301, 2012.

Hanen Borchani, Gherardo Varando, Concha Bielza, and Pedro Larrañaga. A survey on multi-output regression.

Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 5(5):216–233, 2015.

Leo Breiman. Random forests. *Machine learning*, 45(1): 5–32, 2001.

Leo Breiman and Jerome H Friedman. Predicting multivariate responses in multiple linear regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 59(1):3–54, 1997.

Wessel Bruinsma, Eric Perim, William Tebbutt, J. Scott Hosking, Arno Solin, and Richard E. Turner. Scalable exact inference in multi-output gaussian processes. In *Proceedings of ICML,*, volume 119, pages 1190–1201, 2020.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of KDD*, pages 785–794, 2016.

YooJung Choi, Antonio Vergari, and Guy Van den Broeck. Probabilistic circuits: A unifying framework for tractable probabilistic models. Technical report, University of California, Los Angeles, 2020.

Samuel Cohen, Rendani Mbuvha, Tshilidzi Marwala, and Marc Peter Deisenroth. Healing products of gaussian process experts. In *Proceedings of ICML*, pages 2068–2077, 2020.

Adnan Darwiche. A differential approach to inference in bayesian networks. *J. ACM*, 50(3):280–305, 2003.

Marc Peter Deisenroth and Jun Wei Ng. Distributed gaussian processes. In *Proceedings of ICML*, pages 1481–1490, 2015.

Haonan Duan, Abdullah Rashwan, Pascal Poupart, and Zhitang Chen. Discriminative training of feed-forward and recurrent sum-product networks by extended baum-welch. *International Journal of Approximate Reasoning*, 124:66–81, 2020.

Alan Julian Izenman. Reduced-rank regression for the multivariate linear model. *Journal of multivariate analysis*, 5 (2):248–264, 1975.

Agastya Kalra, Abdullah Rashwan, Wei-Shou Hsu, Pascal Poupart, Prashant Doshi, and Georgios Trimponias. Online structure learning for feed-forward and recurrent sum-product networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 31, 2018.

Doga Kisa, Guy Van den Broeck, Arthur Choi, and Adnan Darwiche. Probabilistic sentential decision diagrams. In *Proceedings of Principles of Knowledge Representation and Reasoning (KR)*, 2014.

Dragi Kocev, Sašo Džeroski, Matt D White, Graeme R Newell, and Peter Griffioen. Using single-and multi-target regression trees and ensembles to model a compound index of vegetation condition. *Ecological Modelling*, 220(8):1159–1168, 2009.

Jurica Levatić, Michelangelo Ceci, Dragi Kocev, and Sašo Džeroski. Semi-supervised learning for multi-target regression. In *Proceedings of the International workshop on new frontiers in mining complex patterns*, pages 3–18. Springer, 2014.

Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. When gaussian process meets big data: A review of scalable gps. *IEEE Trans. Neural Networks Learn. Syst.*, 31 (11):4405–4423, 2020.

Mazen Melibari, Pascal Poupart, Prashant Doshi, and George Trimponias. Dynamic sum product networks for tractable inference on sequence data. In *Proceedings of International Conference on PGM*, pages 345–355, 2016.

Pablo Moreno-Muñoz, Antonio Artés, and Mauricio Alvarez. Heterogeneous multi-output gaussian process prediction. In *Proceedings of NeurIPS*, pages 6711–6720, 2018.

Fionn Murtagh. Multilayer perceptrons for classification and regression. *Neurocomputing*, 2(5-6):183–197, 1991.

Radford Neal. *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, 1994.

Robert Peharz, Sebastian Tschiatschek, Franz Pernkopf, and Pedro M. Domingos. On theoretical properties of sum-product networks. In *Proceedings of AISTATS*, volume 38, 2015.

Robert Peharz, Steven Lang, Antonio Vergari, Karl Stelzner, Alejandro Molina, Martin Trapp, Guy Van den Broeck, Kristian Kersting, and Zoubin Ghahramani. Einsum networks: Fast and scalable learning of tractable probabilistic circuits. In *Proceedings of ICML*, pages 7563–7574, 2020.

Emmanouil A Platanios and Sotirios P Chatzis. Nonparametric mixtures of multi-output heteroscedastic gaussian processes for volatility modeling. In *Working Notes of the NeurIPS Workshop on Modern Nonparametric Methods in Machine Learning*. 2012.

Hoifung Poon and Pedro Domingos. Sum-product networks: A new deep architecture. In *Proceedings of UAI*, pages 337–346, 2011.

Balram S. Rajput and Stamatis Cambanis. Gaussian Processes and Gaussian Measures. *The Annals of Mathematical Statistics*, 43(6):1944 – 1952, 1972.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. MIT Press, 2006.

Xiaoting Shao, Alejandro Molina, Antonio Vergari, Karl Stelzner, Robert Peharz, Thomas Liebig, and Kristian Kersting. Conditional sum-product networks: Imposing structure on deep probabilistic architectures. In *Proceedings of PGM*, 2020.

Timo Similä and Jarkko Tikka. Input selection and shrinkage in multiresponse linear regression. *Computational Statistics & Data Analysis*, 52(1):406–422, 2007.

Dimitri P Solomatine and Durga L Shrestha. Adaboost. rt: a boosting algorithm for regression problems. In *Proceedings of IJCNN*, pages 1163–1168. IEEE, 2004.

Eleftherios Spyromitros-Xioufis, Grigorios Tsoumakas, William Groves, and Ioannis Vlahavas. Multi-label classification methods for multi-target regression. *arXiv preprint arXiv:1211.6581*, 2012.

Daniela Stojanova, Michelangelo Ceci, Annalisa Appice, and Sašo Džeroski. Network regression with predictive clustering trees. *Data Mining and Knowledge Discovery*, 25(2):378–413, 2012.

Eric V Strobl, Kun Zhang, and Shyam Visweswaran. Approximate kernel-based conditional independence tests for fast non-parametric causal discovery. *Journal of Causal Inference*, 7(1), 2019.

Kevin Swersky, Jasper Snoek, and Ryan Prescott Adams. Multi-task bayesian optimization. In *Annual Conference on Neural Information Processing Systems (NIPS)*, pages 2004–2012, 2013.

Martin Trapp, Robert Peharz, Hong Ge, Franz Pernkopf, and Zoubin Ghahramani. Bayesian learning of sum-product networks. In *Proceedings of NeurIPS*, pages 6344–6355, 2019.

Martin Trapp, Robert Peharz, Franz Pernkopf, and Carl Edward Rasmussen. Deep structured mixtures of gaussian processes. In *Proceedings of AISTATS*, pages 2251–2261, 2020.

Devis Tuia, Jochem Verrelst, Luis Alonso, Fernando Pérez-Cruz, and Gustavo Camps-Valls. Multioutput support vector regression for remote sensing biophysical parameter estimation. *IEEE Geoscience and Remote Sensing Letters*, 8(4):804–808, 2011.

Chris Williams, Edwin V Bonilla, and Kian M Chai. Multi-task gaussian process prediction. *Advances in neural information processing systems*, pages 153–160, 2007.

Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

Shuo Xu, Xin An, Xiaodong Qiao, Lijun Zhu, and Lin Li. Multi-output least-squares support vector regression machines. *Pattern Recognition Letters*, 34(9):1078–1084, 2013.

Zhongjie Yu, Fabrizio Ventola, and Kristian Kersting. Whittle networks: A deep likelihood model for time series. In *Proceedings of ICML*, 2021.

Michael Minyi Zhang and Sinead A. Williamson. Embarrassingly parallel inference for gaussian processes. *Journal of Machine Learning Research JMLR*, 20:169:1–169:26, 2019.

Wei Zhang, Xianhui Liu, Yi Ding, and Deming Shi. Multi-output ls-svr machine in extended feature space. In *Proceedings of the IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (CIMSA)*, pages 130–134, 2012.

Supplementary Material: Leveraging Probabilistic Circuits for Nonparametric Multi-Output Regression

Zhongjie Yu[1] **Mingye Zhu**[2] **Martin Trapp**[3] **Arseny Skryagin**[1] **Kristian Kersting**[1,4]

[1]Department of Computer Science, TU Darmstadt, Darmstadt, Germany
[2]Department of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, China
[3]Department of Computer Science, Aalto University, Espoo, Finland
[4]Centre for Cognitive Science, TU Darmstadt, and Hessian Center for AI (hessian.AI)

A MOMOGP STRUCTURE CONSTRUCTION

Table 3 compares the results obtained from MOMoGPs constructed either using conditional independence tests or using random splitting. We construct MOMoGPs using conditional independence tests for the splitting of the output space, we employed the randomized conditional correlation test Strobl et al. [2019]. In all experiments, we used a p-value of 0.5. We see that the use of a conditional independence test for the structure construction results in an overall improvement of the performance of MOMoGPs with respect to the RMSE and the MAE.

B MULTI-OUTPUT REGRESSION BENCHMARK RESULTS

We train and test GP, DSMGP, sumGP, and MOMoGP on all data sets (except for usFlight) five times with different random seeds. For usFlight, the above models are trained twice. Table 4 shows both average and standard deviation of the multiple runs. MOGP and MOSVGP have only been trained once for all data sets, thus, their results are not compared in Table 4.

Data Set		Random	CI
Parkinsons	RMSE	**0.775**	**0.775**
	MAE	**0.605**	**0.605**
	NLPD	**2.208**	**2.208**
scm20d	RMSE	0.820	**0.816**
	MAE	0.630	**0.627**
	NLPD	**11.416**	11.470
WindTurbine	RMSE	0.143	**0.137**
	MAE	0.073	**0.072**
	NLPD	**-7.467**	-7.478
Energy	RMSE	0.556	**0.516**
	MAE	0.398	**0.358**
	NLPD	**8.610**	9.402
usFlight	RMSE	0.934	**0.914**
	MAE	0.505	**0.485**
	NLPD	2.091	**2.075**

Table 3: Comparison of results obtained for MOMoGPs constructed using conditional independence tests (CI) or using random splitting.

Data Set		GP		DSMGP		sumGP		**MOMoGP**	
		mean	std	mean	std	mean	std	mean	std
Parkinsons	RMSE	0.783	4×10^{-4}	**0.774**	1×10^{-4}	0.784	3×10^{-4}	0.775$\downarrow$	5×10^{-4}
	MAE	0.610	3×10^{-4}	**0.604**	9×10^{-4}	0.610	4×10^{-4}	0.605$\downarrow$	9×10^{-4}
	NLPD	2.389	1.0×10^{-3}	2.319	3.0×10^{-3}	2.388	1.0×10^{-3}	**2.208**$\uparrow$	3.0×10^{-3}
scm20d	RMSE	0.332	1.80×10^{-2}	0.323	2.7×10^{-4}	**0.322**	1.6×10^{-4}	0.324$\downarrow$	1.1×10^{-3}
	MAE	0.195	1.59×10^{-2}	0.188	4.1×10^{-4}	**0.187**	3.2×10^{-4}	**0.187**$\uparrow$	1.2×10^{-3}
	NLPD	0.543	1.90×10^{-2}	1.341	5.15×10^{-2}	8.097	1.77×10^{-1}	**-0.201**$\uparrow$	1.90×10^{-1}
WindTurbine	RMSE	**0.133**	0	0.143	1×10^{-4}	**0.133**	2×10^{-4}	0.143	0
	MAE	0.074	3×10^{-4}	**0.073**	0	0.074	2×10^{-4}	**0.073**	0
	NLPD	-2.649	4.34×10^{-2}	**-8.749**	2.3×10^{-3}	-2.627	2.89×10^{-2}	-7.467$\downarrow$	3.0×10^{-3}
Energy	RMSE	NA	NA	**0.547**	6.0×10^{-3}	NA	NA	0.556$\downarrow$	6.0×10^{-3}
	MAE	NA	NA	0.400	2.0×10^{-3}	NA	NA	**0.398**$\uparrow$	3.0×10^{-3}
	NLPD	NA	NA	11.745	1.50×10^{-2}	NA	NA	**8.610**$\uparrow$	1.00×10^{-2}
usFlight	RMSE	NA	NA	**0.927**	2×10^{-4}	NA	NA	0.934$\downarrow$	3×10^{-4}
	MAE	NA	NA	**0.492**	3×10^{-4}	NA	NA	0.505$\downarrow$	3×10^{-4}
	NLPD	NA	NA	2.178	1×10^{-4}	NA	NA	**2.091**$\uparrow$	3.57×10^{-2}

Table 4: Mean and standard deviation of Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Negative Log Predictive Density (NLPD) of state-of-the-art approaches and MOMoGP (our work) on benchmark data sets. Smaller values are better. Best result is indicated in **bold** and comparison of MOMoGP to DSMGP is indicated using arrows $\uparrow$ / $\downarrow$.