Stochastic Modeling of Multiple Streamflow Time Series in Colombian Based on Gaussian Processes

Author: Julián David Pastrana-Cortés

Director: Álvaro Angel Orozco-Gutiérrez

Co-director: David Augusto Cardenas-Peña

Automatic Research Group

September 5, 2024

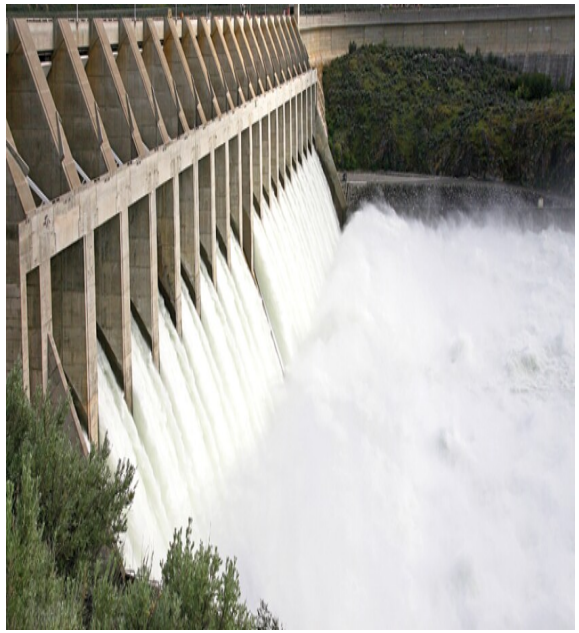
Introduction

Motivation

Understanding the implications of time series associated with hydrological variables, such as flow rates or reservoir levels, is essential for hydroelectric generation and the planning of other generation systems in Colombia

(a) Irrigation

(b) Flood control



(c) Hydropower generation

Challenges: non-linearities, high stochasticity, and complex water resource patterns.

The Importance of Hydrological Forecasting

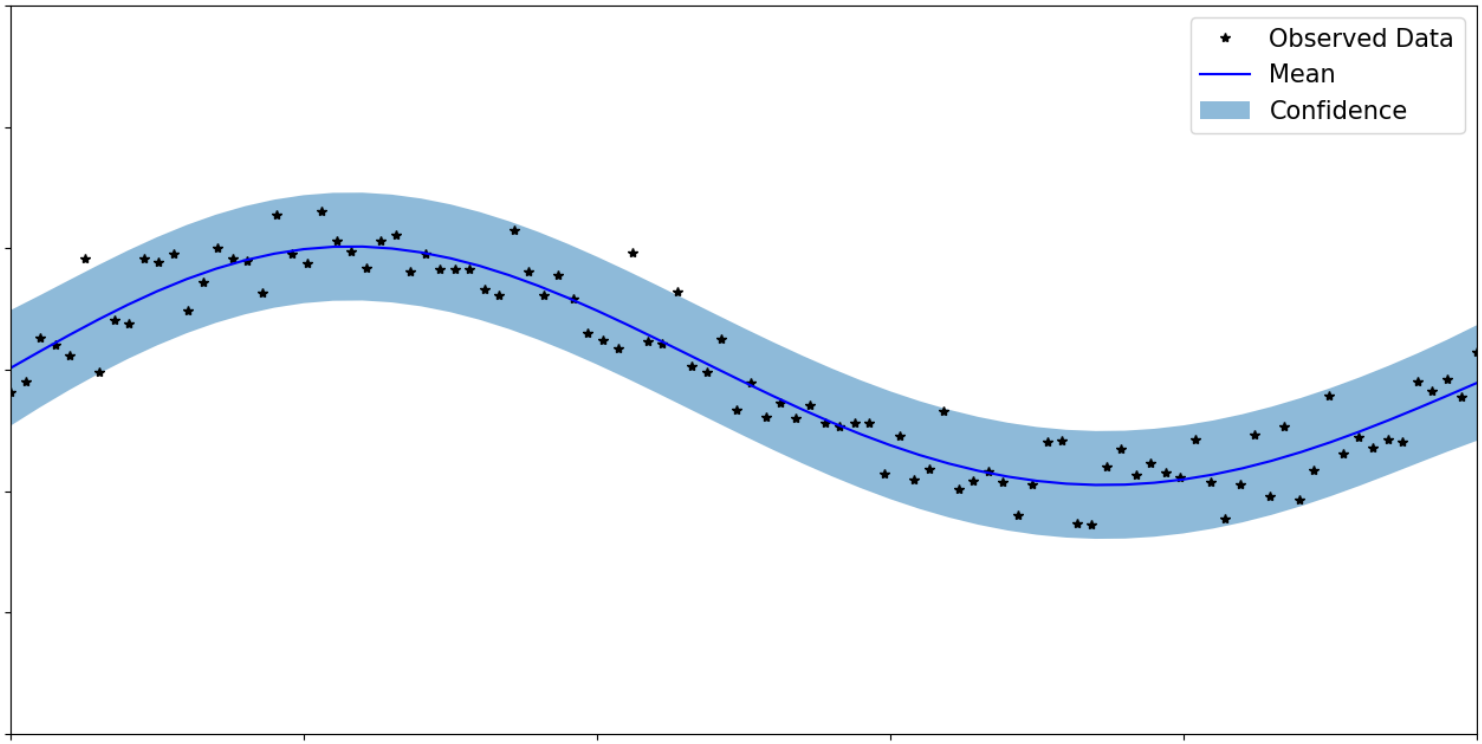
Understanding hydrological processes has become increasingly critical in the field of natural resource management, anticipation capacity of extreme hydrological events such as droughts and heavy rainfall.

(a) Drought Condition

(b) Full Dam

The Proposed Model

A Gaussian Process (GP) is a Bayesian non-parametric model that provides not just point predictions but a full probability distribution, capturing uncertainty and enabling confidence intervals in time series forecasting.



Objectives

General Objective

Develop a stochastic forecasting model for making multiple simultaneous predictions of hydrological time series, taking advantage of cross-correlations among the tasks to improve the performance keeping the scalability in its implementation for the short-term horizon.

Specific Objectives

- Develop a model that allows the forecasting of hydrological time series, properly quantifying the uncertainty associated with each value within the prediction horizons.
- Design a multi-task forecasting methodology that captures and models cross-correlations between hydrological time series, to improve forecast accuracy within forecast horizons.
- Develop a multi-task prediction methodology that handles data constraints across reservoirs while maintaining high forecasting performance as measured by probabilistic metrics.

The Dataset

Problem Setting - Part 1

Consider a time-series vector of hydrological resources observed across all D outputs at the n-th time instant, denoted as $\mathbf{v}_n \in \mathbb{R}^D$. Our model employs the entire sequence of resource vectors from time n back to $n-T+1$ as input to predict the resource vector at the future time step $n+H$. Here, T represents the model order, and H denotes the prediction horizon.
Consequently, we define the input vector $\mathbf{x}_n$ as follows:

$$\mathbf{x}_n = \begin{bmatrix} \mathbf{v}_n^\top \\ \mathbf{v}_{n-1}^\top \\ \vdots \\ \mathbf{v}_{n-T+1}^\top \end{bmatrix} \in \mathbb{R}^{DT}$$

Problem Setting - Part 2

The target output vector $\mathbf{y}_n$ is defined as follows:

$$\mathbf{y}_n = \mathbf{v}_{n+H}.$$

This formulation enables the model to leverage historical hydrological data for accurate future predictions. Accordingly, we build a dataset $\mathcal{D} = \{\boldsymbol{x}_n, \boldsymbol{y}_n\}_{n=1}^{N} = \{\mathbf{X}, \mathbf{y}\}$ comprising N input–output pairs, where $\boldsymbol{x}_n \in \mathcal{X} \subset \mathbb{R}^{DT}$ represents the dimensional input space. The target function is a vector-valued function where $\boldsymbol{y}_n \in \mathbb{R}^{D}$ comprises the observations of all outputs (tasks) at the same input $\boldsymbol{x}_n$.

The hydrological forecasting task for validating the proposed model regressors considers time-series streamflow contributions data from 23 Colombian reservoirs.

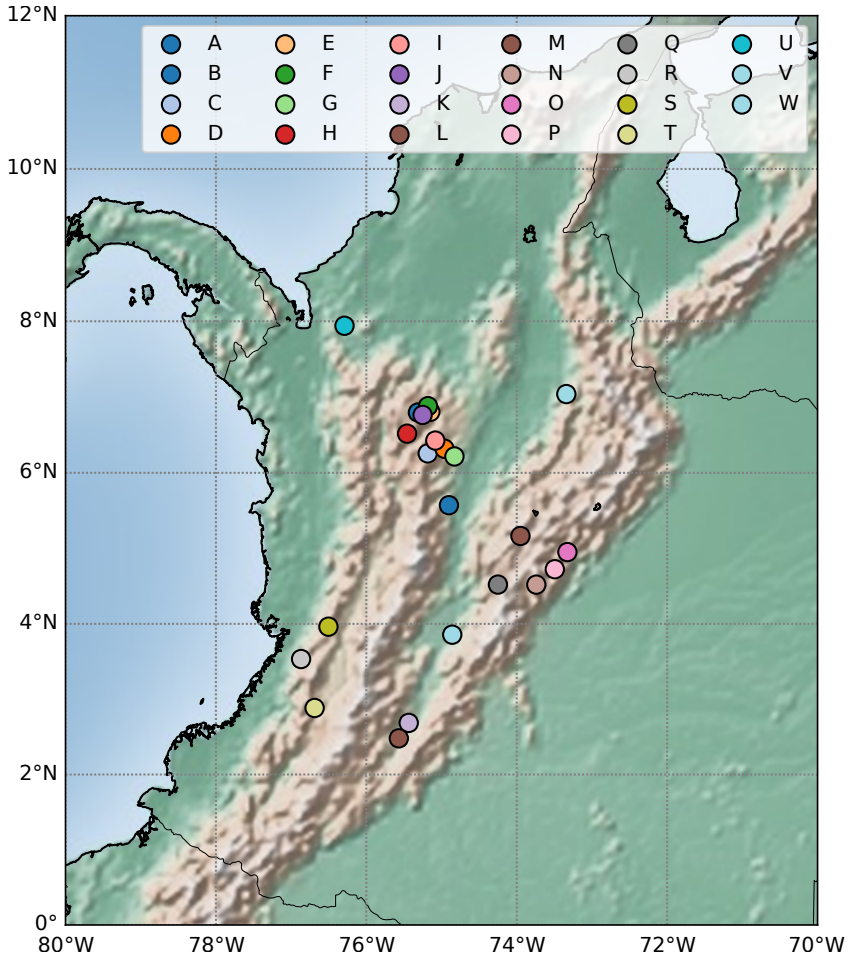


Figure: Reservoir locations in Colombia.

The dataset was selected due to the strong dependence of time series on weather patterns, which are closely tied to hydropower dispatch operations.

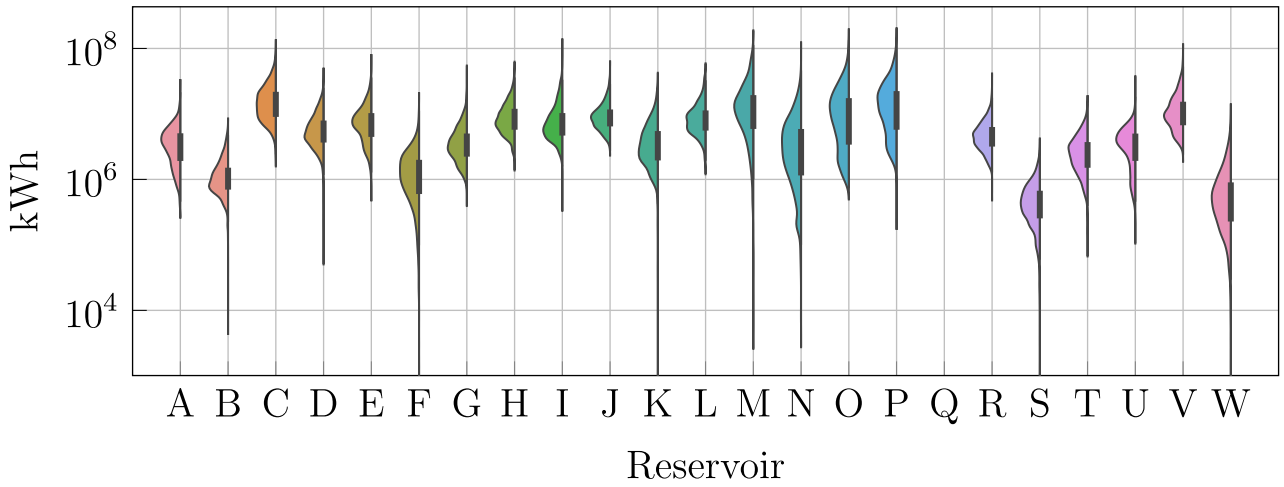


Figure: Time-series Violin plot depicting the streamflow contribution for each reservoir within the dataset.

These streamflow contributions were recorded daily from 1 January 2010 to 28 February 2022. While these contributions represent volumetric values, they are reported in kilowatt-hours (kWh) by the hydroelectric power plants.

Gaussian Process Regression: Bayesian Non-Parametric Model

In the GP framework, the dataset $\mathcal{D}$ is used to learn a random mapping function $\boldsymbol{f}(\cdot)$, capturing the relationship between $\boldsymbol{x}_n$ and $\boldsymbol{y}_n$. We define a GP distribution over $\boldsymbol{f}(\cdot)$ as:

$$\boldsymbol{f}(\boldsymbol{x}) \sim \mathcal{GP}\left(\mathbf{0}, \boldsymbol{k}(\boldsymbol{x}, \boldsymbol{x}' \mid \boldsymbol{\theta})\right) \tag{1}$$

where $\boldsymbol{f} : \mathcal{X} \to \mathbb{R}^D$ maps the input space, and $\boldsymbol{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^{D \times D}$ is the cross-covariance matrix function, parameterized by $\boldsymbol{\theta}$. Adding i.i.d Gaussian noise $\boldsymbol{\epsilon}$ with covariance $\Sigma_\epsilon = \text{diag}\{\sigma_{Nd}^2\}_{d=1}^D$, the model prediction becomes $\boldsymbol{y}_n = \boldsymbol{f}(\boldsymbol{x}_n) + \boldsymbol{\epsilon}$.

Joint Distribution

Let $\mathbf{X}_* = \{\boldsymbol{x}_{n*}\}_{n=1}^{N_*}$ be a set of test points $\boldsymbol{x}_{n*} \in \mathcal{X}$ with test output vector $\mathbf{f}_* = [\boldsymbol{f}(\boldsymbol{x}_1)^\top, \boldsymbol{f}(\boldsymbol{x}_2)^\top, \cdots, \boldsymbol{f}(\boldsymbol{x}_{N_*})^\top]^\top \in \mathbb{R}^{N_* D}$. The joint Gaussian distribution over $\mathbf{y}$ and $\mathbf{f}_*$ is given by:

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K}_y & \mathbf{K}_* \\ \mathbf{K}_*^\top & \mathbf{K}_{**} \end{bmatrix}\right) \tag{2}$$

Here, $\mathbf{K}_y = \mathbf{K} + \Sigma_\epsilon$ with $\mathbf{K} \in \mathbb{R}^{ND \times ND}$, formed by evaluating the covariance function at all pairs in $\mathbf{X}$. The matrices $\mathbf{K}_{**} \in \mathbb{R}^{N_* D \times N_* D}$ and $\mathbf{K}_* \in \mathbb{R}^{ND \times N_* D}$ are formed by evaluating the covariance function across test inputs in $\mathbf{X}_*$ and test-train inputs, respectively.

Posterior Distribution

This notation allows for deriving the conditional distribution named posterior as follows:

$$\mathbf{f}_*|\mathbf{X}_*, \mathcal{D} \sim \mathcal{N}(\bar{\mathbf{f}_*}, \text{cov}(\mathbf{f}_*)) \tag{3}$$

with

$$\bar{\mathbf{f}_*} = \mathbf{K}_*^\top \mathbf{K}_y^{-1} \mathbf{y} \tag{4}$$

$$\text{cov}(\mathbf{f}_*) = \mathbf{K}_{**} - \mathbf{K}_*^\top \mathbf{K}_y^{-1} \mathbf{K}_* \tag{5}$$

The Marginal Log-likelihood

Furthermore, the prediction performance archived by the conditional distribution depends on the selected parameter set $\boldsymbol{\theta}$ and observation noise matrix Σ_ϵ. Both parameters are calculated by maximizing the marginal log-likelihood from Equation (2) where $p(\mathbf{y}) = \mathcal{N}(\mathbf{y} \mid \mathbf{0}, \mathbf{K}_y)$ as follows:

$$\begin{aligned} \{\boldsymbol{\theta}_{\text{opt}}, \Sigma_{\epsilon\text{opt}}\} &= \arg\max_{\boldsymbol{\theta}, \Sigma_\epsilon} \quad \ln p(\mathbf{y}) \\ &= \arg\min_{\boldsymbol{\theta}, \Sigma_\epsilon} \quad \frac{1}{2}\mathbf{y}^\top \mathbf{K}_y^{-1}\mathbf{y} + \frac{1}{2}\ln \mid \mathbf{K}_y \mid + \frac{ND}{2}\ln 2\pi \end{aligned} \tag{6}$$

The main challenge is their computational complexity of $\mathcal{O}(N^3D^3)$ and storage demand of $\mathcal{O}(N^2D^2)$ due to inverting the matrix $\mathbf{K}_y$.

Variational Inference

A common approach to reduce GP complexity, consists of the introduction of a reduced set of $M \ll N$ inducing point locations, denoted by Z, containing vectors $\mathbf{z} \in \mathcal{X}$, and an inducing variable vector $\mathbf{u} = [\boldsymbol{f}(\boldsymbol{z}_1)^\top, \boldsymbol{f}(\boldsymbol{z}_2)^\top, \cdots, \boldsymbol{f}(\boldsymbol{z}_M)^\top]^\top \in \mathbb{R}^{MD}$. This approach leads to the following extended joint distribution:

$$\begin{bmatrix} \mathbf{u} \\ \mathbf{f} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0} , \begin{bmatrix} \mathbf{K}_{uu} & \mathbf{K}_{uf} \\ \mathbf{K}_{uf}^\top & \mathbf{K} \end{bmatrix} \right) \tag{7}$$

Here, $\mathbf{K}_{uu} \in \mathbb{R}^{MD \times MD}$ and $\mathbf{K}_{uf} \in \mathbb{R}^{MD \times ND}$ represent the block covariance matrices evaluated at the inducing point pairs and between the inducing points and training inputs, respectively. We call this model the Sparse Variational Gaussian Process (SVGP).

The Variational Distribution

From the above, the conditional distribution is given by:

$$p(\mathbf{f} \mid \mathbf{u}) = \mathcal{N}\left(\mathbf{f} \mid \mathbf{K}_{uf}^{\top}\mathbf{K}_{uu}^{-1}\mathbf{u}, \mathbf{K} - \mathbf{K}_{uf}^{\top}\mathbf{K}_{uu}^{-1}\mathbf{K}_{uf}\right), \tag{8}$$

and the prior for the inducing variables is:

$$p(\mathbf{u}) = \mathcal{N}\left(\mathbf{u} \mid \mathbf{0}, \mathbf{K}_{uu}\right). \tag{9}$$

The joint posterior distribution over the latent variable $\mathbf{f}$ and inducing variable $\mathbf{u}$ is:

$$p(\mathbf{f}, \mathbf{u} \mid \mathbf{y}) = p(\mathbf{f} \mid \mathbf{u})p(\mathbf{u} \mid \mathbf{y}). \tag{10}$$

However, computing $p(\mathbf{u} \mid \mathbf{y})$ is typically intractable, so we use variational inference to approximate the posterior with a variational distribution $q(\mathbf{u}) = \mathcal{N}(\mathbf{u} \mid \boldsymbol{\mu}, \boldsymbol{S})$, where $\boldsymbol{\mu} \in \mathbb{R}^{MD}$ and $\boldsymbol{S} \in \mathbb{R}^{MD \times MD}$. The approximate joint posterior then becomes:

$$p(\mathbf{f}, \mathbf{u} \mid \mathbf{y}) \approx q(\mathbf{f}, \mathbf{u}) = p(\mathbf{f} \mid \mathbf{u})q(\mathbf{u}). \tag{11}$$

ELBO

Accordingly, the approximation of the posterior distribution consists of estimating the variational parameters $\boldsymbol{\mu}$ and $\boldsymbol{S}$, performed by maximizing an evidence lower bound (ELBO). Such ELBO is obtained from the log marginal likelihood:

$$\ln p(\mathbf{y}) \geq \int \int q(\mathbf{f}, \mathbf{u}) \ln \frac{p(\mathbf{y} \mid \mathbf{f}) p(\mathbf{f} \mid \mathbf{u}) p(\mathbf{u})}{q(\mathbf{f}, \mathbf{u})} d\mathbf{f} d\mathbf{u} = \mathcal{L} \tag{12}$$

resulting from the Jensen's inequality. We can rewrite the previous expression to derive the following:

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{f})}\{\ln p(\mathbf{y} \mid \mathbf{f})\} - \mathsf{KL}\{q(\mathbf{u}) \parallel p(\mathbf{u})\} \tag{13}$$

Here, KL represents the Kullback–Leibler divergence between the Gaussian-shaped distributions.

ELBO

With the i.i.d assumption, the likelihood function $p(\mathbf{y} \mid \mathbf{f})$ can be factorized over observations and outputs:

$$p(\mathbf{y} \mid \mathbf{f}) = \prod_{d=1}^{D} \prod_{n=1}^{N} p(y_{dn} \mid f_d(\mathbf{x}_n)), \tag{14}$$

where $f_d(\mathbf{x}_n)$ represents the latent function value for the d-th output at the n-th input, and y_{dn} is the corresponding label. The lower bound can then be expressed as:

$$\mathcal{L} = \sum_{d=1}^{D} \sum_{n=1}^{N} \mathbb{E}_{q(f_d(\mathbf{x}_n))}\{\ln p(y_{dn} \mid f_d(\mathbf{x}_n))\} - \sum_{d=1}^{D} \mathrm{KL}\{q(\mathbf{u}_d) \parallel p(\mathbf{u}_d)\}. \tag{15}$$

The summation over D and N facilitates training through a mini-batch fashion.

Posterior and Predictive Distribution

We define the variational distribution for $\mathbf{f}$ as:

$$q(\mathbf{f}) := \int p(\mathbf{f} \mid \mathbf{u}) q(\mathbf{u}) d\mathbf{u}$$
$$= \mathcal{N}\left(\mathbf{f} \mid \mathbf{K}_{uf}^{\top} \mathbf{K}_{uu}^{-1} \boldsymbol{\mu}, \mathbf{K} + \mathbf{K}_{uf}^{\top} \mathbf{K}_{uu}^{-1} (\mathbf{S} - \mathbf{K}_{uu}) \mathbf{K}_{uu}^{-1} \mathbf{K}_{uf}\right). \tag{16}$$

This approach reduces complexity to $\mathcal{O}(NM^2D^3)$, as $\mathbf{K}_{uu}$ is smaller than $\mathbf{K}_y$. For predictions at new points $\mathbf{x}_*$, we compute:

$$p(\mathbf{f}_* \mid \mathbf{y}) \approx q(\mathbf{f}_*) = \int p(\mathbf{f}_* \mid \mathbf{u}) q(\mathbf{u}) d\mathbf{u}, \tag{17}$$

and adding Gaussian noise Σ_ϵ to the latent posterior $\mathbf{f}_* \mid \mathbf{y}$.

Model Setup

The proposed approach factorizes the GP scalar covariance function into two kernels: $k_{\mathcal{X}}$, which models input correlations via relative distances, and k_D, which models task pair-wise correlations as follows:

$$k\left((\boldsymbol{x}, d), (\boldsymbol{x}', d') \mid \boldsymbol{\theta}\right) = k_{\mathcal{X}}\left(\boldsymbol{x}, \boldsymbol{x}' \mid \boldsymbol{\Theta}_d\right) k_D\left(d, d' \mid \sigma_d\right), \tag{18}$$

$$k_{\mathcal{X}}\left(\boldsymbol{x}, \boldsymbol{x}' \mid \boldsymbol{\Theta}_d\right) = \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}')^\top \boldsymbol{\Theta}_d^{-2}(\boldsymbol{x} - \boldsymbol{x}')\right), \tag{19}$$

$$k_D\left(d, d' \mid \sigma_d^2\right) = \sigma_d^2\, \delta_{d,d'}, \tag{20}$$

where $\delta_{d,d'}$ is the Kronecker delta, σ_d^2 is the output scale for task d, and $\boldsymbol{\Theta}_d$ is a diagonal matrix of lengthscale factors. The input kernel $k_{\mathcal{X}}$ uses a squared-exponential function, ensuring smooth data mapping. The task kernel k_D avoids modeling task correlations directly, reducing complexity to $\mathcal{O}(NM^2D)$. Despite not modeling task correlations explicitly, the autoregressive inputs and shared inducing points still enable exploration of task dependencies.

Performance Metrics

To evaluate our model's performance, we use the following three metrics:

- Mean Squared Error (MSE)
- Mean Standardized Log Loss (MSLL)
- Continuous Ranked Probability Score (CRPS)

Mean Squared Error

The MSE, measures the average squared difference between the observed outcome at the n-th test time step for the d-th output, denoted as y_{dn}, and the expected value of the predicted outcome, denoted as $\mathbb{E}\{y_{dn*}\}$, over N_* test observations.

$$\text{MSE} = \frac{1}{DN_*} \sum_{d=1}^{D} \sum_{n=1}^{N_*} (y_{dn} - \mathbb{E}\{y_{dn*}\})^2 \tag{21}$$

Note that the MSE metric does not account for the prediction uncertainty.

Mean Standardized Log Loss (MSLL)

MSLL measures the quality of probabilistic predictions by evaluating the log probability of observed values under the predicted distribution. Assuming normally distributed errors, MSLL is given by:

$$\text{MSLL} = \frac{1}{2DN_*} \sum_{d=1}^{D} \sum_{n=1}^{N_*} \left(\frac{(y_{dn} - \mathbb{E}\{y_{dn*}\})^2}{\text{var}\{y_{dn*}\}} - \frac{(y_{dn} - \mu_d)^2}{\sigma_d^2} + \ln \left(\frac{\text{var}\{y_{dn*}\}}{\sigma_d^2} \right) \right) \tag{22}$$

where $\text{var}\{y_{dn*}\}$, μ_d, and σ_d^2 are the predicted variance, mean, and variance of the training data for the d-th output. A positive MSLL indicates worse performance than a Gaussian baseline, zero indicates parity, and negative indicates improvement.

Continuous Ranked Probability Score

The CRPS evaluates the quality of probabilistic predictions by measuring the area of the squared difference between the predicted cumulative distribution function (CDF) $\Phi(y_{dn})$ and the empirical degenerate CDF of the observed values. For a Gaussian predictive distribution, the CRPS becomes the following:

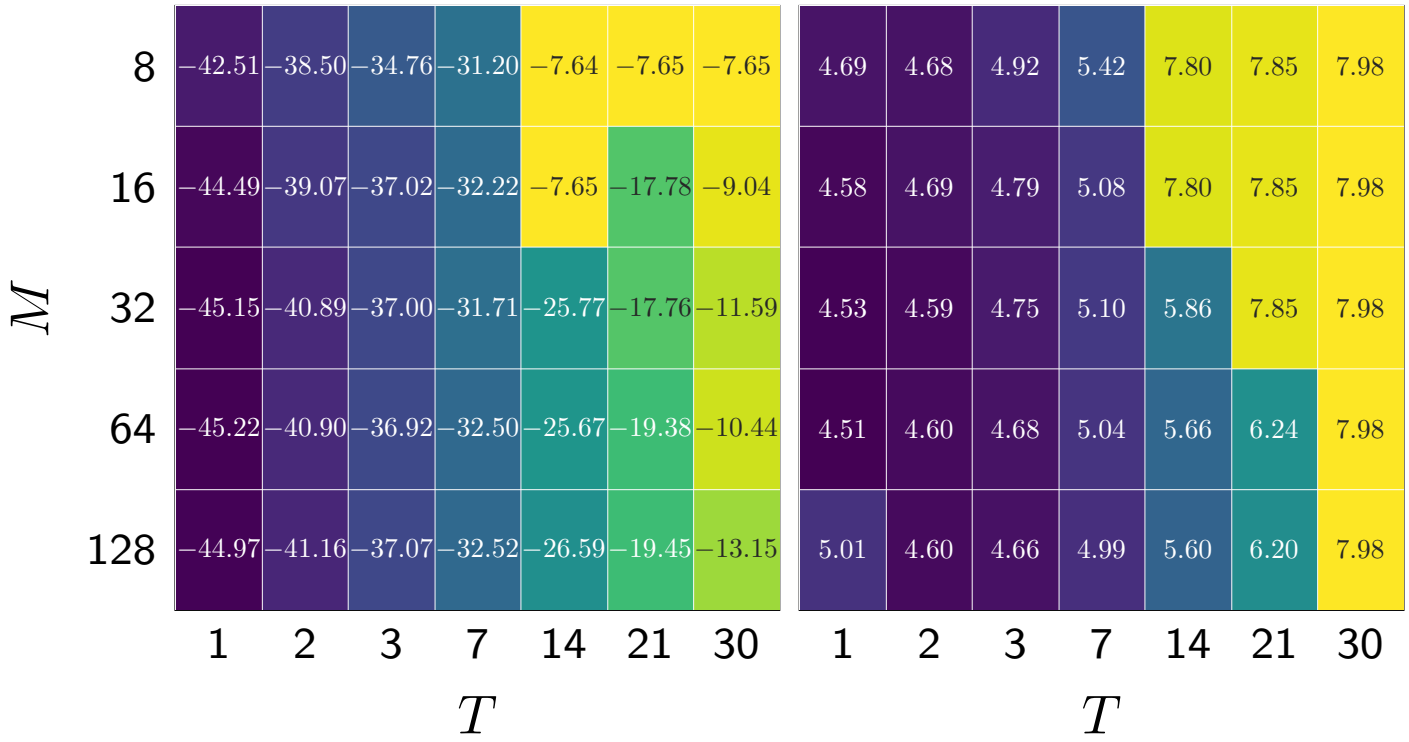
$$\text{CRPS} = \frac{1}{DN_*} \sum_{d=1}^{D} \sum_{n=1}^{N_*} \sqrt{\text{var}\{y_{dn*}\}} \left(\beta_{dn} \left(2\Phi \left(\beta_{dn} \right) - 1 \right) + 2\phi \left(\beta_{dn} \right) - \frac{1}{\sqrt{\pi}} \right) \tag{23}$$

with $\beta_{dn} = \frac{y_{dn} - \mathbb{E}\{y_{dn*}\}}{\sqrt{\text{var}\{y_{dn*}\}}}$ being the zero-mean unit variance standardization of the random variable y_{dn}, and a standard Gaussian for predictive distribution $\phi(\beta_{dn}) = \mathcal{N}(\beta_{dn} \mid 0, 1)$.

Hyperparameter Tuning

The task involves predicting the $D = 23$ reservoir contributions at day $n + H$ using the $L = 23$ contributions at day n. The forecasting horizon H includes short-term (up to a week ahead, $H \in \{1, 2, 3, 4, 5, 6, 7\}$) and medium-term (up to one month ahead, $H \in \{9, 14, 21, 30\}$) predictions. The SVGP model requires tuning the number of inducing points M and order size T.

Results evidence that the median MSLL and MSE generally decrease when increasing the number of inducing points, up to $M = 64$, after which the error increases, and increases for large values of T. This suggests underfitting with too few inducing points and overfitting with too many.

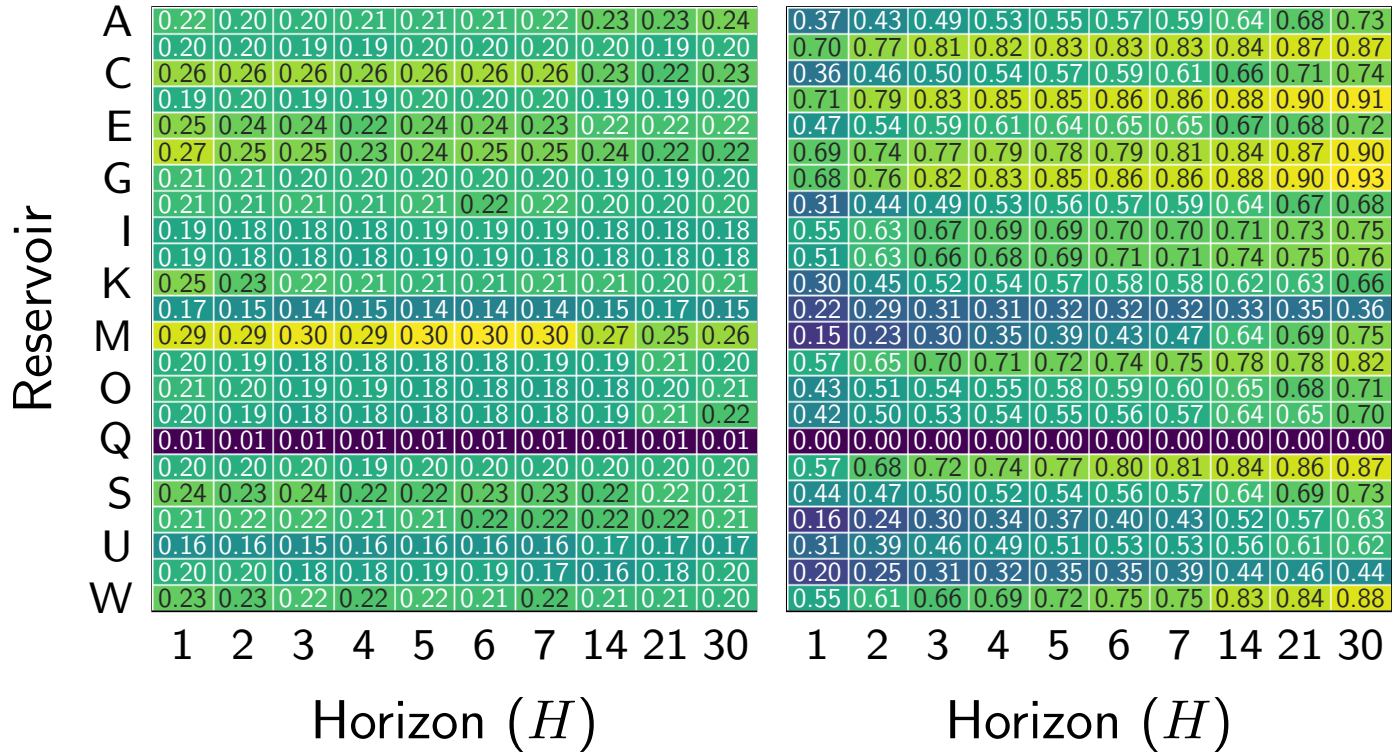


(a) MSLL ($\times 10^{-2}$) (b) MSE ($\times 10^{-1}$)

The optimal settings are $M = 64$ and $T = 1$.

Reservoir-Wise Output Scales and Noise Variance

Using the tuned number of inducing points M and model order T, We present the trained reservoir-wise output scales σ_d^2 and the estimated noise variance Σ_ϵ for different horizons (H).



(a) Output scales (b) Noise variances

Output Scales

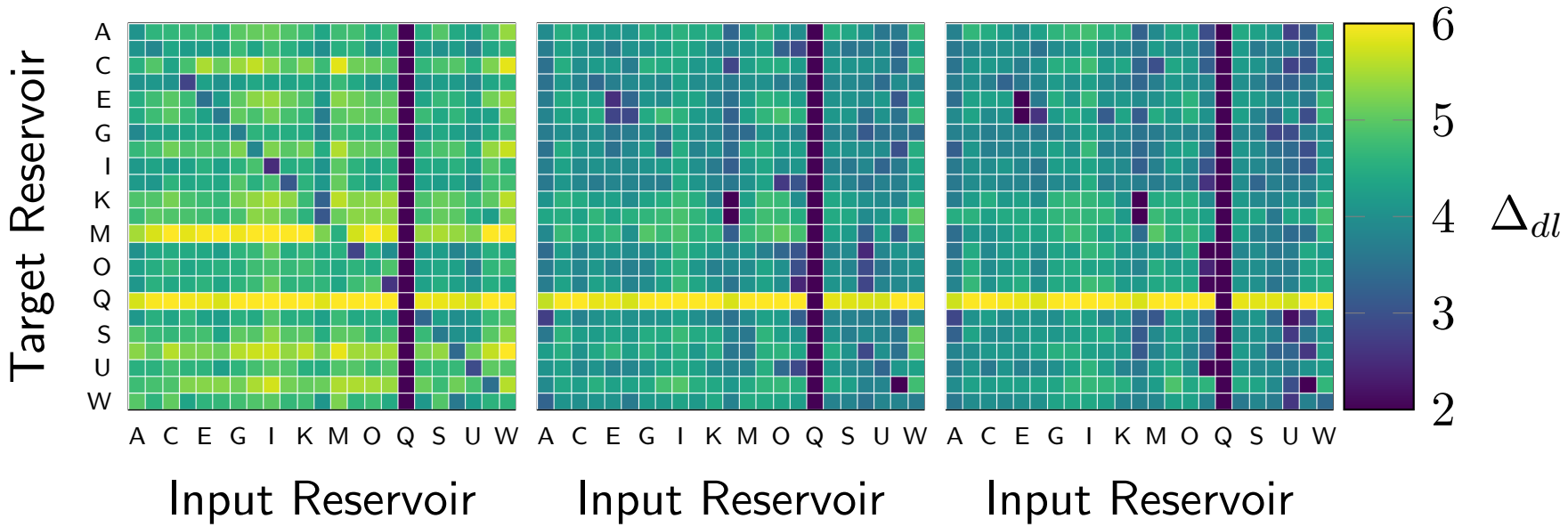
Overall, the longer the horizon, the smaller the output scale for all reservoirs. The above suggests a reduction in correlation for long horizons, making the SVGP approximate the prior distribution with a time-uncorrelated function. Nonetheless, for reservoirs P and V, an output scale increment at long horizons follows the initial reduction, implying the existence of relevant periodicity components within such time series. Lastly, reservoir Q consistently renders low-scale values along horizons to enforce the expected zero streamflow.

Noise Variance

Regarding the noise variance, the heightened unpredictability in the output data at long horizons yields an increment in Σ_ϵ, which also supports the positive definiteness of the covariance matrix $\mathbf{K}_y$. For such horizons, the model uncertainty mainly depends on the noise variance rather than the output scale.
Therefore, the SVGP model exploits the data correlations at short horizons, yielding a low uncertainty while understanding the lack of information for long-horizon forecasting as a zero-mean isotropic Gaussian posterior.

Lengthscale Analysis

The lengthscale analysis relies on its capacity to deactivate nonessential features on a kernel machine according to the Automatic Relevance Determination (ARD). Hence, a small lengthscale value implies a high relevance for predicting the future streamflow of a reservoir from the past of another.



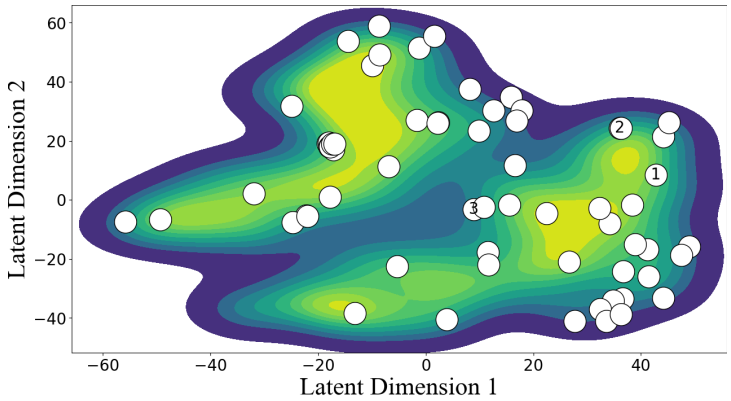
(a) $H = 1$. (b) $H = 14$. (c) $H = 30$.

Figure: Trained lengthscales from input feature (columns) to each output task (rows) for three prediction horizons.

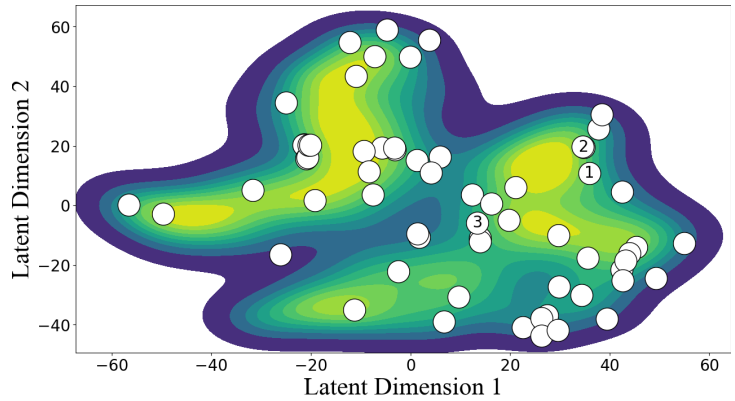
The shortest horizon distributes the smallest lengthscale values over the main diagonal contrary to the high off-diagonal values. This result proves the relevance of short-term memory components within each streamflow.
At a longer horizon of fourteen days, the lengthscales over the main diagonal become higher, losing relevance, and several off-diagonal become lower, gaining relevance, than at a one-day horizon. Such a change implies that a reservoir provides less information to predict itself at longer horizons, leading the model to look for knowledge in other reservoirs.
The longest horizon illustrates even less relevant lengthscales, indicating that the SVGP restrains the number of relevant features. Thus, the lengthscale variations across different prediction horizons reveal the dynamic interplay between reservoirs and highlight the Gaussian Process's ability to select relevant features adaptively.

t-distributed Stochastic Neighbor Embedding (t-SNE)

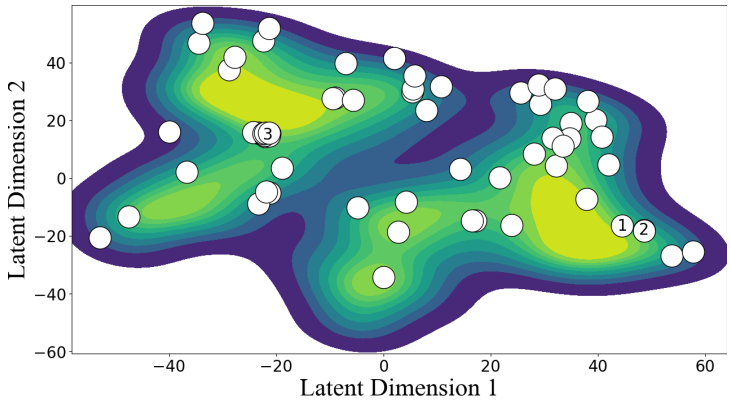
For visual interpretation of SVGP inner workings, the following figure maps in 2D the reservoir data along with the optimized inducing points using the t-distributed Stochastic Neighbor Embedding (t-SNE) technique for each target reservoir. The filled contours outline the density distribution of training data, whereas the color scatter locates the optimized inducing points. We strategically labeled three inducing points for analyzing the SVGP behavior. Since the forecasting tasks share the inducing points, not the kernel parameters, each target reservoir holds a different t-SNE mapping.



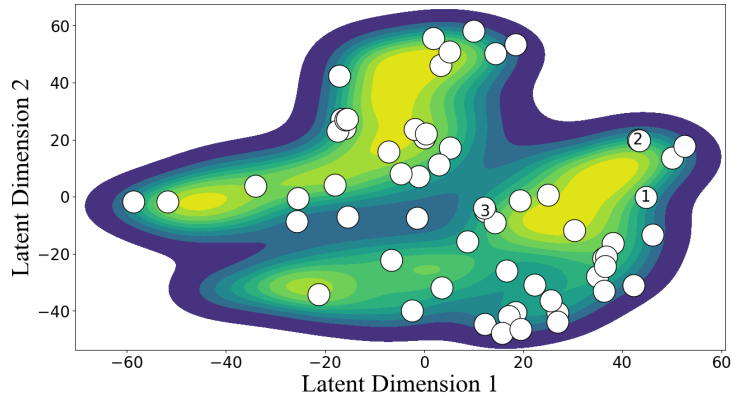
(a) Reservoir E.



(b) Reservoir I.



(c) Reservoir L.



(d) Reservoir U.

Figure: t-SNE-based two-dimensional mapping of the SVGP latent space and inducing points' locations for four target reservoirs. Three inducing points are numbered to analyze the model behavior.

The 2D plots exhibit multi-modal distributions for all tasks, indicating clustered information and varying dynamics within a reservoir. Hence, the SVGP trades off the location of the inducing points between the within-task centroids and the among-task relationships. Therefore, the shared inducing points allow for the capturing of task-wise, group-wise, and global information about the streamflow dynamics.

Model Comparison

For performance analysis, the proposed SVGP-based forecasting is contrasted against the straightforward Linear AutoRegressive model (LAR) and the nonlinear Long-Short-Term Memory network (LSTM).
Particularly for LAR, we trained a single output model for each task. Further, the frequentist perspective treats the model weights as random estimators following a Gaussian distribution. Such a treatment turns the LAR into a probabilistic model, allowing it to generate confidence intervals for its predictions.
In the case of LSTM, we set up a single multi-output model to forecast all reservoirs simultaneously. To tune the number of hidden units of the LSTM, we run a grid search, yielding an optimum of one unit. As a deterministic model, LSTM lacks a predictive distribution and cannot be assessed with the probabilistic performance metrics MSLL and CRPS.

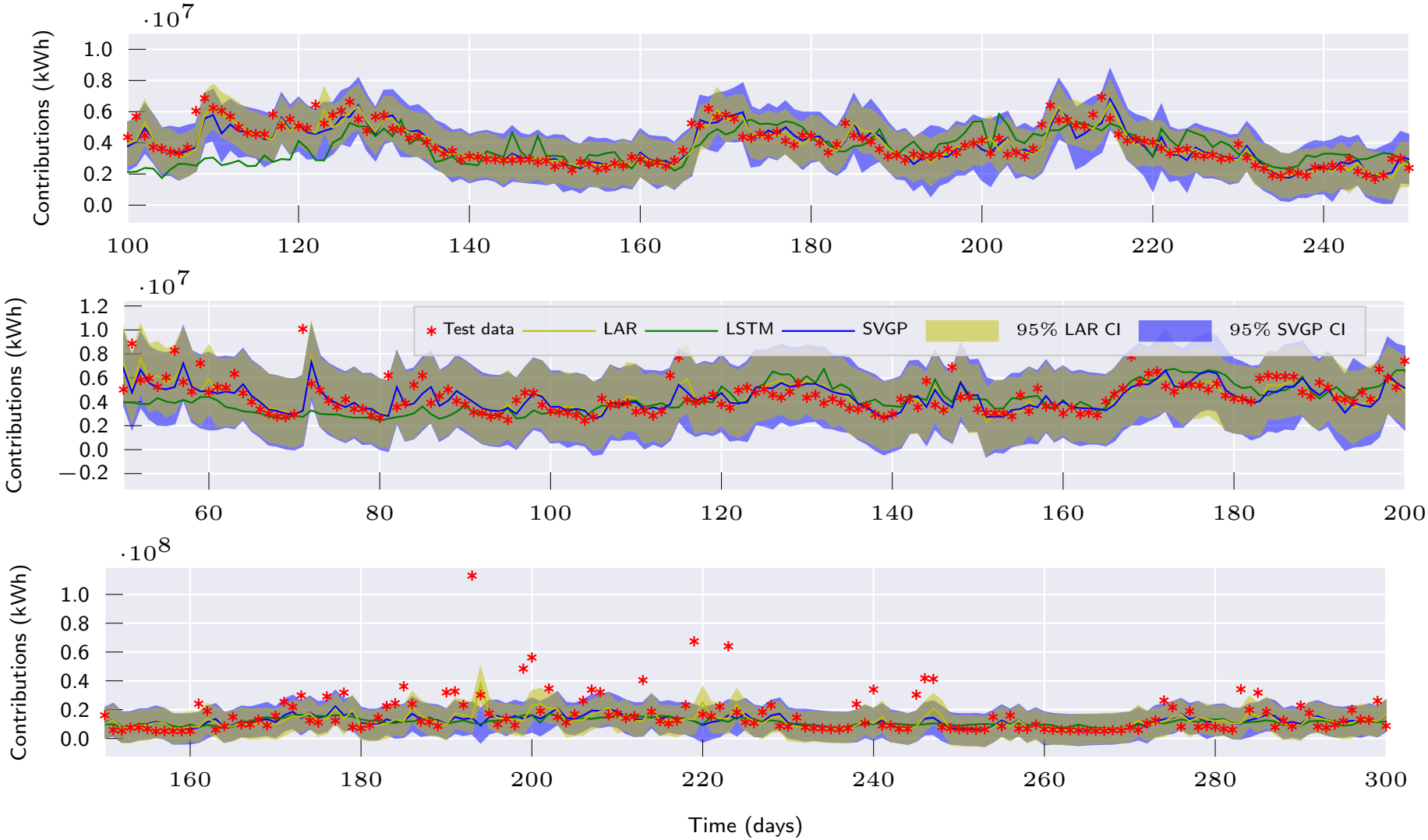
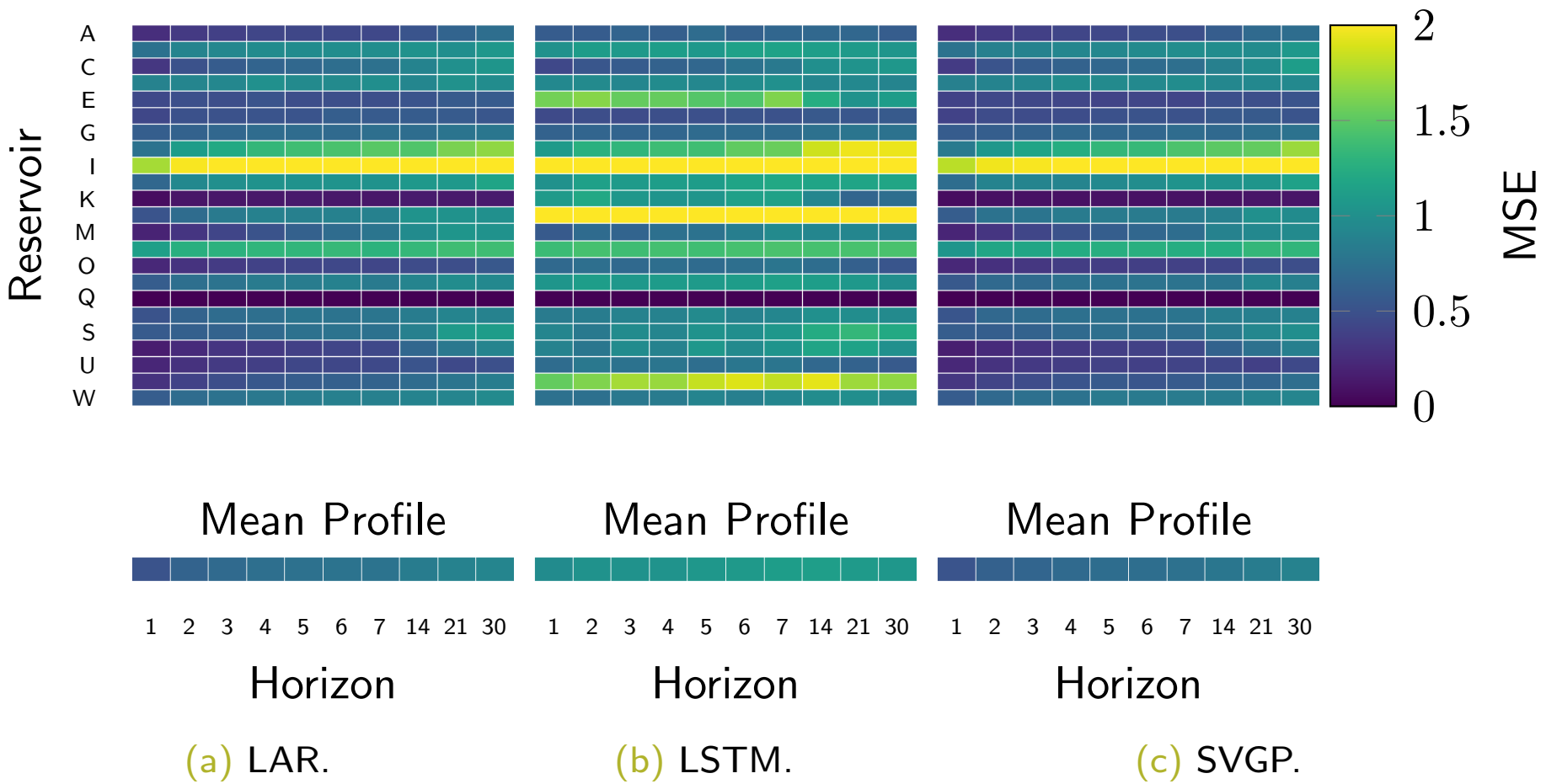


Figure: Test data for reservoirs T (low complexity), A (medium complexity), and I (high complexity), top to bottom in a one-day ahead model's prediction. Yellow and blue shaded areas represent the 95% centered confidence interval for the LAR and SVGP predictions, respectively.

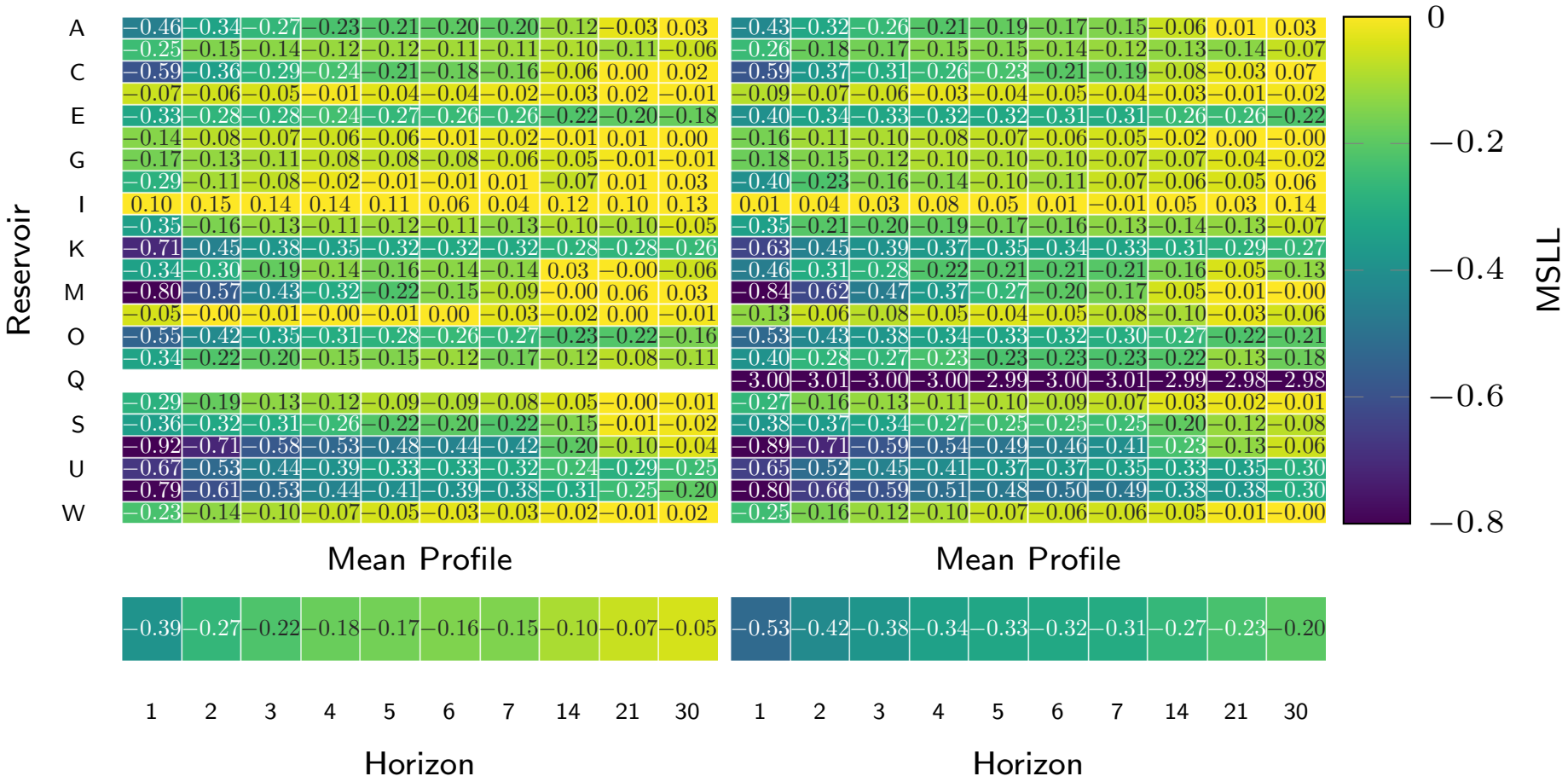
MSE Scores



(a) LAR. (b) LSTM. (c) SVGP.

Figure: Achieved MSE for LAR, LSTM, and SVGP forecasting models at each horizon and reservoir.

MSLL Score



(a) LAR. (b) SVGP.

Figure: Achieved MSLL for LAR and SVGP forecasting models at each horizon and reservoir.

Performance t-test

Table: Performance metrics for LAR, LSTM, and SVGP on the considered horizons H. Bold and asterisk denote a p-value $p < 1\%$ in a one-tailed paired t-test for LAR vs. SVGP and LSTM vs. SVGP.

	MSE			MSLL		CRPS	
H	**LAR**	**LSTM**	**SVGP**	**LAR**	**SVGP**	**LAR**	**SVGP**
1	0.51	0.96	0.52 *	−0.39	−0.53	0.34	0.32
2	0.63	1.01	**0.61** *	−0.27	**−0.42**	0.39	**0.36**
3	0.68	1.02	**0.65** *	−0.22	**−0.38**	0.41	**0.38**
4	0.72	1.03	**0.69** *	−0.18	**−0.34**	0.42	**0.39**
5	0.74	1.06	**0.71** *	−0.17	**−0.33**	0.43	**0.40**
6	0.76	1.07	**0.72** *	−0.16	**−0.32**	0.44	**0.40**
7	0.76	1.11	**0.74** *	−0.15	**−0.31**	0.44	**0.41**
14	0.83	1.12	**0.79** *	−0.10	**−0.27**	0.46	**0.43**
21	0.88	1.08	**0.83** *	−0.07	**−0.23**	0.48	**0.45**
30	0.91	1.06	**0.88** *	−0.05	**−0.20**	0.49	**0.46**
Grand Average	0.74	1.05	**0.71** *	−0.18	**−0.33**	0.43	**0.40**

To Conclude

- The proposed methodology diminishes the computational complexity from cubic to linear regarding the number of samples, becoming more scalable to forecasting tasks on large datasets.
- The optimal number of inducing points performs as an inherent regularization by encoding the most information from the data while avoiding overfitting.
- The t-SNE-based distribution plots reveal that the proposed model strategically places the shared inducing points to capture task-wise, group-specific, and global dynamics, trading off between capturing the reservoir-wise unique characteristics and the between-reservoir dependencies, improving the overall performance in streamflow forecasting.

To Conclude

- The trained lengthscales prove that the GP model successfully adjusts its focus to prediction horizon changes between short-term memory components and long-term relationships. This adaptability makes the Gaussian Process model robust and versatile for multiple-output forecasting tasks with varying time features.
- The performance analysis evidences the advantage of the proposed SVGP model over the baseline LAR and LSTM models for streamflow prediction in three main aspects. Firstly, adaptability to changing dynamics within and between reservoirs. Secondly, the Bayesian scheme returns a posterior distribution and provides informative confidence intervals. Thirdly, the SVGP better copes with the enhanced forecasting complexity for long horizons than baseline approaches, as proved by the slower error growth.

Multi-Output Gaussian Processes: Modeling Inter-Output Dependencies

Multi Output Gaussian Processes

We start to extending the notation developed so far to learning multiple outputs with a single model, we consider

$$\boldsymbol{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_d(\mathbf{x}), \cdots, f_D(\mathbf{x})]^\top$$

as a multi-output Gaussian process, comprising D single-output Gaussian processes. Now our target is a vector-valued function.

Independent Gaussian Process (IGP)

Combining all independent GP-based models evaluated in the N_* test point set X_*, assuming that all outputs are fully observed for each input, the vector function space inference corresponds to the following generative model:

$$\begin{bmatrix} \mathbf{f}_{1*} \\ \mathbf{f}_{2*} \\ \vdots \\ \mathbf{f}_{D*} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0} \ , \begin{bmatrix} K_{1**} & 0 & \cdots & 0 \\ 0 & K_{2**} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & K_{D**} \end{bmatrix} \right)$$

Here, $K_{d**} \in \mathbb{R}^{N_* \times N_*}$ correspond to the covariance matrix of the d-th output. Due to the independence assumption, the grand covariance matrix of the multi-output model $\mathbf{K}_{**} \in \mathbb{R}^{DN_* \times DN_*}$ is diagonal by block.

We call this model Independent Gaussian process (IGP).

Modeling Output Interactions

We introduce a framework based on a set of independent Gaussian processes $\{u_q(\mathbf{x})\}_{q=1}^{Q}$, with their own covariance function $k_q(\mathbf{x}, \mathbf{x}')$ and each latent process $f_d(\mathbf{x})$ is represented as an instantaneous, time-invariant linear combination of the independent processes:

$$f_d(\mathbf{x}) = \sum_{q=1}^{Q} a_{d,q} u_q(\mathbf{x}) \qquad u_q(\mathbf{x}) \sim \mathcal{GP}(0, k_q(\mathbf{x}, \mathbf{x}'))$$

Graphical Representation

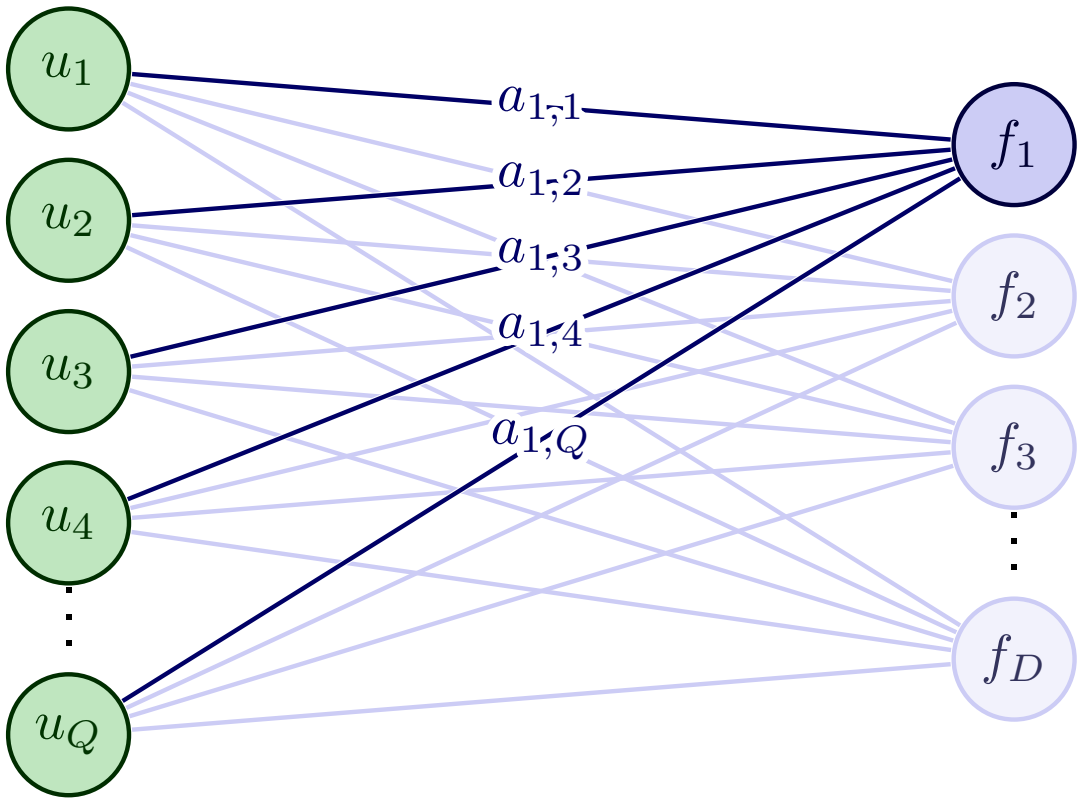


Figure: Graphical representation of the Linear Model of Coregionalization GP

Cross-covariance function

The cross-covariance function of the latent Gaussian process is expressed as:

$$
\begin{aligned}
k_{d,d'}(\mathbf{x}, \mathbf{x}') &= \text{cov}\{f_d(\mathbf{x}), f_{d'}(\mathbf{x}')\} \\
&= \sum_{q=1}^{Q} \sum_{q'=1}^{Q} a_{d,q} a_{d',q'} \text{cov}\{u_q(\mathbf{x}), u_{q'}(\mathbf{x}')\} \\
&= \sum_{q=1}^{Q} a_{d,q} a_{d',q} k_q(\mathbf{x}, \mathbf{x}') \\
&= \sum_{q=1}^{Q} b^q_{d,d'} k_q(\mathbf{x}, \mathbf{x}')
\end{aligned}
\tag{24}
$$

Here, $b^q_{d,d'} = a_{d,q} a_{d',q}$ captures the interactions among the outputs induced by the q-th independent process, while $k_q(\mathbf{x}, \mathbf{x}')$ characterizes the interaction among input spaces viewed from the perspective of the q-th independent process.

The Matrix Kernel Function

Instead of a scalar kernel function, we now have a matrix kernel function $\mathbf{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^{D \times D}$ with elements $k_{d,d'}(\mathbf{x}, \mathbf{x}')$:

$$\mathbf{k}(\mathbf{x}, \mathbf{x}') = \sum_{q=1}^{Q} \mathbf{B}_q k_q(\mathbf{x}, \mathbf{x}') \tag{25}$$

Here, each $\mathbf{B}_q \in \mathbb{R}^{D \times D}$ is referred to as the coregionalization matrix, with elements $(\mathbf{B}_q)_{d,d'} = b^q_{d,d'}$. Evaluating this at all test points X_* allows us to recover the covariance matrix as follows:

$$\mathbf{K}_{**} = \sum_{q=1}^{Q} \mathbf{B}_q \otimes K_{q**} \tag{26}$$

where $\otimes$ denotes the Kronecker product between matrices.

Linear Model of Coregionalization GP (LMCGP)

This model can effectively capture the cross-covariances of the output given by $k_{d,d'}(\mathbf{x}, \mathbf{x}')$, allowing to fill zeros in covariance matrix of IGP as follows:

$$\begin{bmatrix} \mathbf{f}_{1*} \\ \mathbf{f}_{2*} \\ \vdots \\ \mathbf{f}_{D*} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0} \ , \sum_{q=1}^{Q} \begin{bmatrix} b_{1,1}^q & b_{1,2}^q & \cdots & b_{1,D}^q \\ b_{2,1}^q & b_{2,2}^q & \cdots & b_{2,D}^q \\ \vdots & \vdots & \ddots & \vdots \\ b_{D,1}^q & b_{D,2}^q & \cdots & b_{D,D}^q \end{bmatrix} \otimes K_{q**} \right)$$

We call this model Linear Model of Coregionalization Gaussian Process (LMCGP).

Variational Inference and ELBO

We can extent our variational inference to include the independent set. Furthermore, instead of utilizing the latent processes f_d, we utilize the inducing variables derived from the independent processes u_q providing the following ELBO:

$$\mathcal{L} = \sum_{d=1}^{D} \sum_{n=1}^{N} \mathbb{E}_{q(f_{dn})}\{\log p(y_{dn} \mid f_{dn})\} - \sum_{q=1}^{Q} \mathsf{KL}\{q(\mathbf{u}_q) \parallel p(\mathbf{u}_q)\} \tag{27}$$

Latent posterior and Predictive distribution

The posterior over test points X_*, denoted as $p(\mathbf{f}_* \mid \mathbf{y})$, is approximated as follows:

$$p(\mathbf{f}_* \mid \mathbf{y}) \approx q(\mathbf{f}_*) = \int p(\mathbf{f}_* \mid \mathbf{u}) q(\mathbf{u}) d\mathbf{u} \tag{28}$$

We add Gaussian noise σ^2_{Nd} to the corresponding task into the above random vector to obtain the predictive distribution.

Adam + Natural Gradient Optimization

Optimization Parameters for LMCGP:

- **Kernel:** lengthscales, output scales
- **Likelihood:** data noise $\{\sigma^2_{Nd}\}_{d=1}^{D}$
- **Inducing Points:** Z
- **Variational Parameters:** $\{\mu_{\mathbf{q}}, S_q\}_{q=1}^{Q}$

Challenges: Strong dependency between variational parameters and others makes the model sensitive. ELBO loss function is non-convex, leading to poor local minima with traditional optimizers.

Solution: Combine Natural Gradient (NG) with Adam. NG optimizes variational parameters, while Adam optimizes other parameters.

Benefits: This hybrid method, Adam + NG, improves optimization performance, allowing better convergence.

Model Setup

The proposed methodology constructs the LMCGP covariance function using the widely applied squared exponential kernel:

$$k_q\left(\mathbf{x}, \mathbf{x}' \mid \Theta_q\right) = \exp\left(-\frac{1}{2}(\mathbf{x}-\mathbf{x}')^\top \Theta_q^{-2}(\mathbf{x}-\mathbf{x}')\right) \tag{29}$$

Upon examining, one might question the absence of an output scale parameter. However, a detailed look at shows that the elements in the matrix $\mathbf{B}_q$ perform the function of rescaling the exponential term in the k_q kernel.

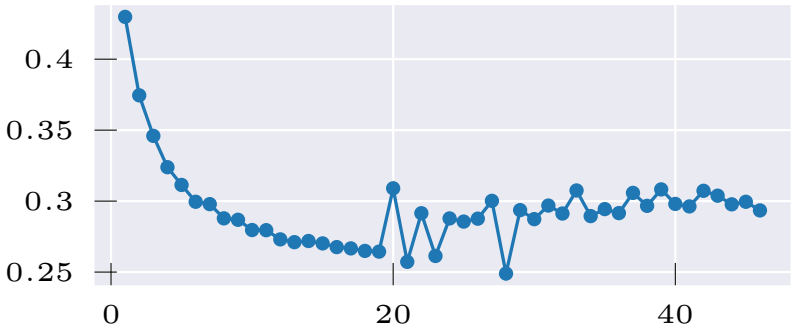
The trainable covariance parameters are $\{\mathbf{B}_q, \Theta_q\}_{q=1}^{Q}$.

Negative Log Predictive Density

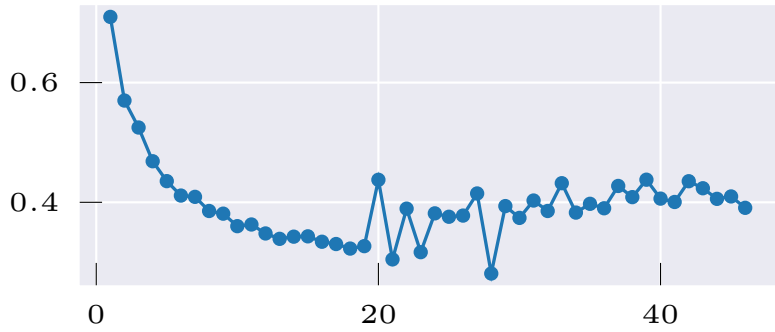
To evaluate the models' performance, we use the implemented metrics MSE, MSLL, and CRPS. Additionally, we propose a new metric called Negative Log Predictive Density (NLPD), which applies to any type of predictive distribution and is defined as:

$$\begin{aligned} \text{NLPD} &= -\frac{1}{N_*} \log p(\mathbf{y}_* \mid \mathbf{y}) \\ &= -\frac{1}{N_*} \sum_{n=1}^{N_*} \log p(\mathbf{y}_{n*} \mid \mathbf{y}) \end{aligned} \tag{30}$$

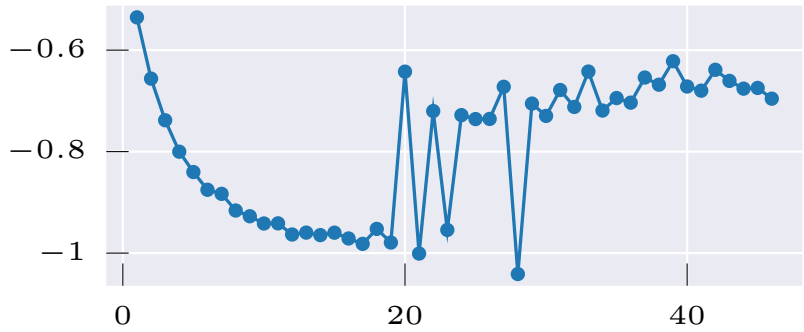
Tuning Q



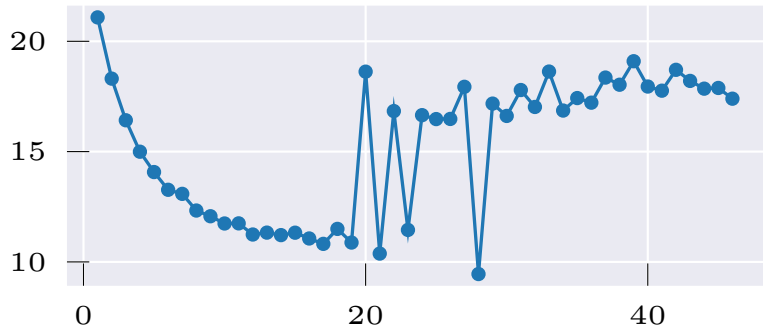
(a) CRPS



(b) MSE



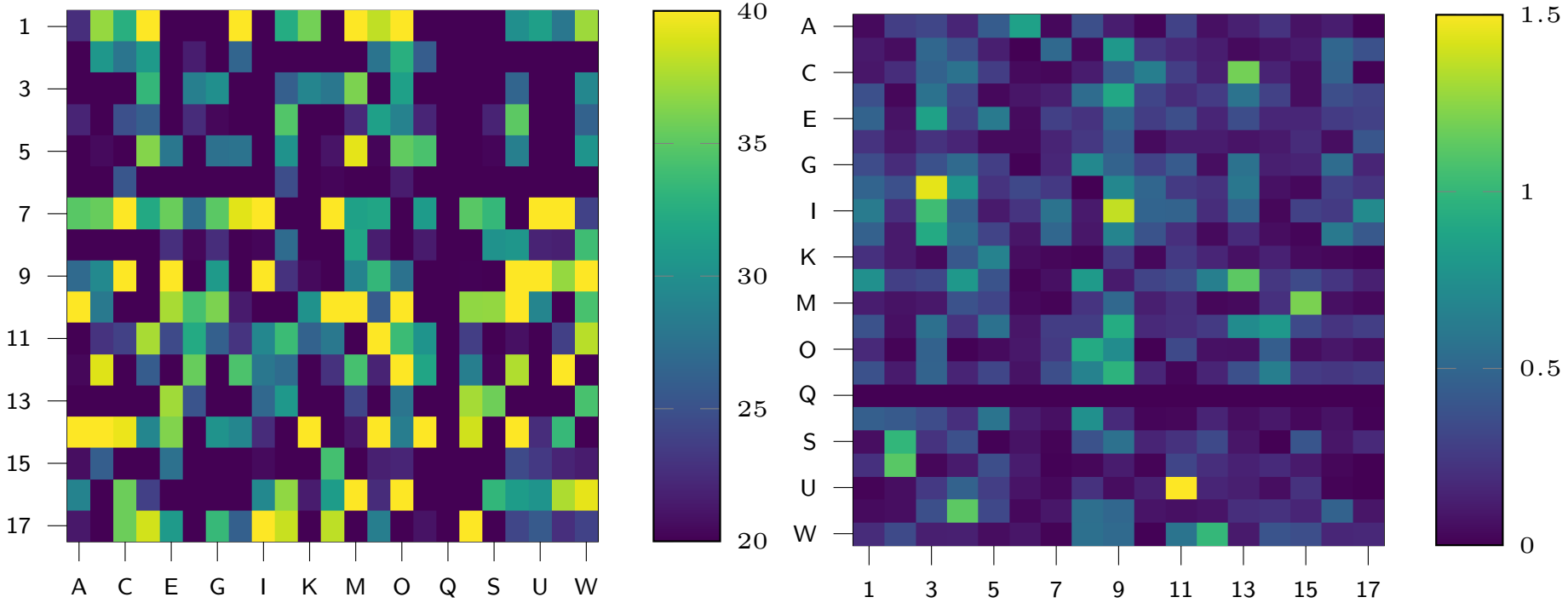
(c) MSLL



(d) NLPD

Figure: Performance metrics for LMCGP models as a function of the number of independent GPs. We finally select $Q = 17$ as the proper parameter

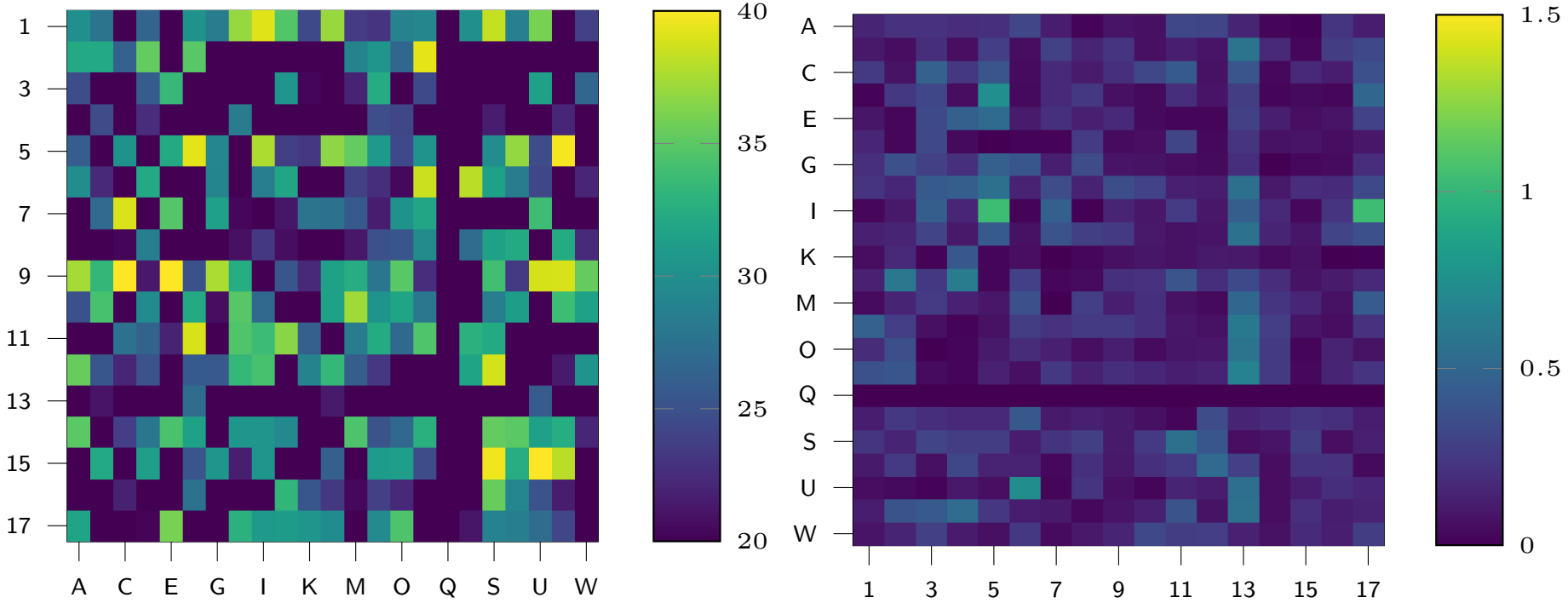
Lengthscale and $a_{d,q}$ Values (H=1)



(a) (b)

Figure: Lengthscale values of Input Features for Each Independent GP Model (left) and coefficients $a_{d,q}$ (right) for horizon $H = 1$.

Lengthscale and $a_{d,q}$ Values (H=30)

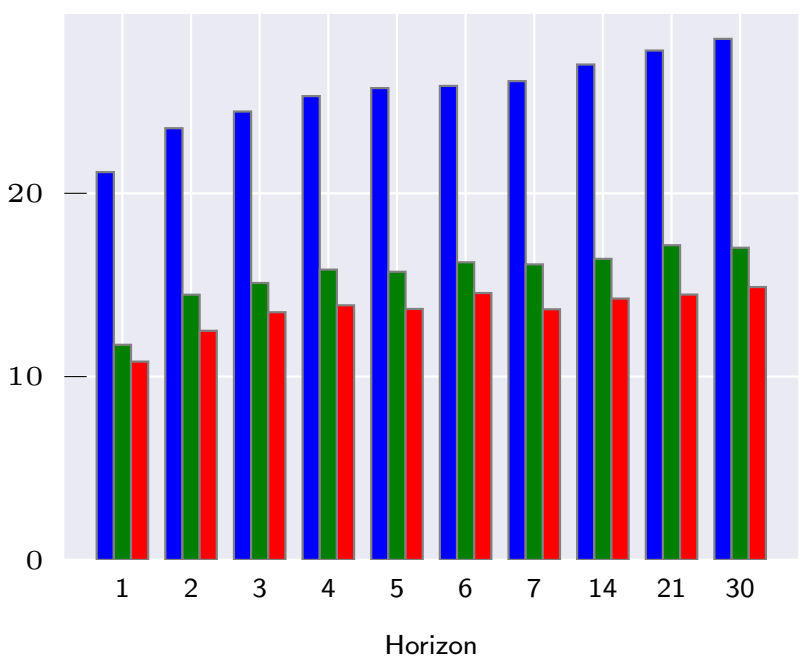


(a) (b)

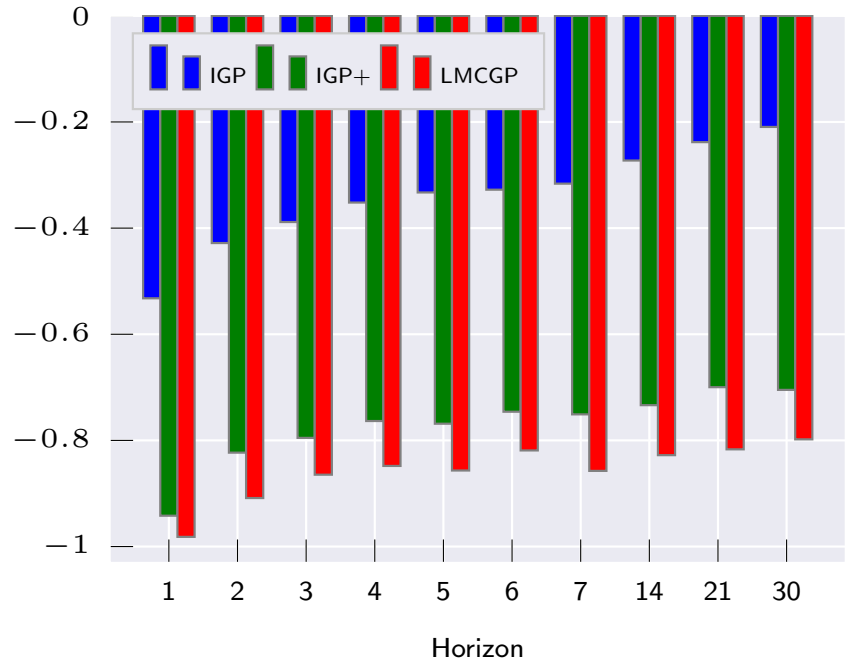
Figure: Lengthscale values of Input Features for Each Independent GP Model (left) and coefficients $a_{d,q}$ (right) for horizon $H = 30$.

LMCGP vs IGP+ vs IGP

The performance analysis compares LMCGP against two multi-output GP models: the IGP implemented, trained using only Adam optimizer, and another IGP trained into Adam + NG framework, called here IGP+.



(a) NLPD



(b) MSLL

Figure: Bar plots comparing the performance of LMCGP, IGP+, and IGP models for different H values.