RECURRENT MEMORY FOR ONLINE INTERDOMAIN GAUSSIAN PROCESSES

Wenlong Chen[1,*], **Naoki Kiyohara**[1,2,*], **Harrison Bo Hua Zhu**[1,3,*], **Yingzhen Li**[1],
[1]Imperial College London [2]Canon Inc. [3]University of Copenhagen
wenlong.chen21@imperial.ac.uk n.kiyohara23@imperial.ac.uk
harrison.zhu@sund.ku.dk yingzhen.li@imperial.ac.uk

ABSTRACT

We propose a novel online Gaussian process (GP) model that is capable of capturing long-term memory in sequential data in an online regression setting. Our model, Online HiPPO Sparse Variational Gaussian Process Regression (OHSGPR), leverages the HiPPO (High-order Polynomial Projection Operators) framework, which is popularized in the RNN domain due to its long-range memory modeling capabilities. We interpret the HiPPO time-varying orthogonal projections as inducing variables with time-dependent orthogonal polynomial basis functions, which allows the SGPR inducing points to memorize the process history. We show that the HiPPO framework fits naturally into the interdomain GP framework and demonstrate that the kernel matrices can also be updated online in a recurrence form based on the ODE evolution of HiPPO. We evaluate our method on time series regression tasks, showing that it outperforms the existing online GP method in terms of predictive performance and computational efficiency.

1 INTRODUCTION

Gaussian processes (GPs) are a popular choice for modeling time series due to their functional expressiveness and uncertainty quantification abilities (Roberts et al., 2013; Fortuin et al., 2020). However, GPs are computationally expensive and memory intensive, with cubic and quadratic complexities, respectively. In online regression settings, such as weather modeling, the number of time steps can be very large, quickly making GPs infeasible. Although variational approximations, such as utilizing sparse inducing points (SGPR (Titsias, 2009); SVGP (Hensman et al., 2013; 2015a)) and Markovian GPs (Wilkinson et al., 2021), have been proposed to address the computational complexity, it would still be prohibitive to re-fit the GP model from scratch every time new data arrives.

Bui et al. (2017) proposes an online sparse GP (OSGPR) learning method that sequentially updates the GP posterior distribution only based on the newly arrived data. However, as indicated in their paper, their models may not maintain the memory of the previous data, as the inducing points will inevitably shift as new data arrive. This is a major drawback, as their models may not model long-term memory unless using a growing number of inducing points.

In deep learning, as an alternative to Transformers (Vaswani, 2017), significant works on state space models (SSMs) have been proposed to model long-term memory in sequential data. Originally proposed to instill long-term memory in recurrent neural networks, the HiPPO (High-order Polynomial Projection Operators) framework (Gu et al., 2020) provides mathematical foundations for compressing continuous-time signals into memory states through orthogonal polynomial projections. HiPPO is the basis for the state-of-the-art SSMs, such as the structured state space sequential (S4) model (Gu et al., 2022) and Mamba (Gu & Dao, 2023; Dao & Gu, 2024). HiPPO is computationally efficient and exhibits strong performance in long-range memory tasks.

Inspired by HiPPO, we propose Online HiPPO SGPR (OHSGPR), by applying the HiPPO framework to SGPR in order to leverage the long-range memory modeling capabilities. Our method interprets the HiPPO time-varying orthogonal projections as inducing variables of an interdomain SGPR (Lázaro-Gredilla & Figueiras-Vidal, 2009; Leibfried et al., 2020; Van der Wilk et al., 2020), where

*Equal contribution.

the basis functions are time-dependent orthogonal polynomials. We show that we are able to significantly resolve the memory-loss issue in OSGPR, thereby opening up the possibility of applying GPs to long-term time series regression tasks. In more detail, we make the following contributions:

- We demonstrate that HiPPO integrates into the interdomain GPs by interpreting the HiPPO time-varying orthogonal projections as inducing variables with time-dependent orthogonal polynomial basis functions. This allows the inducing variables to memorize the process history, capturing long-term memory in the data.
- We show that the kernel matrices can leverage the efficient ODE evolution of the HiPPO framework, bringing an extra layer of computational efficiency to OHSGPR.
- We demonstrate OHSGPR on time series regression tasks, showing that it outperforms vanilla OSGPR in terms of predictive performance and computational efficiency.

2 Background

In this section, we provide a brief overview of GPs, inducing point methods, and online learning with GPs. In addition, we review the HiPPO method, which is the basis of our proposed method.

2.1 Gaussian processes

Let $\mathcal{X}$ be the input space. For time series data, $\mathcal{X} = [0, \infty)$, the set of non-negative real numbers. A Gaussian process (GP) $f \sim \mathcal{GP}(0, k)$ is defined with a covariance function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. It has the property that for any finite set of input points $\mathbf{X} = [x_1, \ldots, x_n]^\intercal$, the random vector $\mathbf{f} \equiv f(\mathbf{X}) = [f(x_1), \ldots, f(x_n)]^\intercal \sim \mathcal{N}(0, \mathbf{K}_{\mathbf{ff}})$, where $\mathbf{K}_{\mathbf{ff}}$ is the kernel matrix with entries $[k(\mathbf{X}, \mathbf{X})]_{ij} \equiv [\mathbf{K}_{\mathbf{ff}}]_{ij} = k(x_i, x_j)$. For notational convenience and different sets of input points $\mathbf{X}_1$ and $\mathbf{X}_2$, we denote the kernel matrix as $\mathbf{K}_{\mathbf{f}_1\mathbf{f}_2}$ or $k(\mathbf{X}_1, \mathbf{X}_2)$. The computational and memory complexities of obtaining the GP posterior on $\mathbf{X}_1$ scale cubically and quadratically respectively, according to $n_1 = |\mathbf{X}_1|$. Given responses $\mathbf{y}$ and inputs $\mathbf{X}$, a probabilistic model can be defined as $y_i \sim p(y_i \mid f(x_i))$ with a GP prior $f \sim \mathcal{GP}(0, k)$, where $p(y_i \mid f(x_i))$ is the likelihood distribution. However, for non-conjugate likelihoods, the posterior distribution is intractable, and approximate inference methods are required, such as, but not limited to, variational inference (Titsias, 2009; Hensman et al., 2013; 2015a) and Markov chain Monte Carlo (MCMC) (Hensman et al., 2015b).

2.2 Variational inference and interdomain Gaussian processes

To address the intractability and cubic complexity of GPs, Sparse Variational Gaussian Processes (SGPR (Titsias, 2009); SVGP (Hensman et al., 2013; 2015b)) cast the problem as an optimization problem. By introducing M inducing points $\mathbf{Z} \in \mathcal{X}^M$ that correspond to M inducing variables $\mathbf{u} = [f(\mathbf{z}_1), \ldots, f(\mathbf{z}_M)]^\intercal$, the variational distribution $q(\mathbf{f}, \mathbf{u})$ is defined as $q(\mathbf{f}, \mathbf{u}) := p(\mathbf{f} \mid \mathbf{u}) q_\theta(\mathbf{u})$, where $q_\theta(\mathbf{u})$ is the variational distribution of the inducing variables with parameters θ. Then, the evidence lower bound (ELBO) is defined as

$$\log p(\mathbf{y}) \geq \sum_{i=1}^{n} \mathbb{E}_{q(f_i)}[\log p(y_i \mid f_i)] - \mathrm{KL}\left[q_\theta(\mathbf{u}) \| p(\mathbf{u})\right] =: \mathcal{L}_\theta, \tag{1}$$

where $q(f_i) = \int p(f_i \mid \mathbf{u}) q_\theta(\mathbf{u}) \mathrm{d}\mathbf{u}$ is the posterior distribution of $f_i \equiv f(x_i)$. Typical choices for the variational distribution are $q_\theta(\mathbf{u}) = \mathcal{N}(\mathbf{u}; \mathbf{m}_\mathbf{u}, \mathbf{S}_\mathbf{u})$, where $\mathbf{m}_\mathbf{u}$ and $\mathbf{S}_\mathbf{u}$ are the free-form mean and covariance of the inducing variables, and yields the posterior distribution:

$$q(f_i) = \mathcal{N}(f_i; \mathbf{K}_{f_i\mathbf{u}}\mathbf{K}_{\mathbf{uu}}^{-1}\mathbf{m}_\mathbf{u}, \mathbf{K}_{f_if_i} - \mathbf{K}_{f_i\mathbf{u}}\mathbf{K}_{\mathbf{uu}}^{-1}[\mathbf{K}_{\mathbf{uu}} - \mathbf{S}_\mathbf{u}]\mathbf{K}_{\mathbf{uu}}^{-1}\mathbf{K}_{\mathbf{u}f_i}). \tag{2}$$

When the likelihood is conjugate Gaussian, the ELBO can be optimized in closed form and $\mathbf{m}_\mathbf{u}$ and $\mathbf{S}_\mathbf{u}$ can be obtained in closed form (SGPR; Titsias (2009)). In addition to setting the inducing variables as the function values, interdomain GPs (Lázaro-Gredilla & Figueiras-Vidal, 2009) propose to generalize the inducing variables to $u_m := \int f(x)\phi_m(x)\mathrm{d}x$, where $\phi_m(x)$ are basis functions, to allow for further flexibility. This yields $[\mathbf{K}_{f\mathbf{u}}]_m = \int k(x, x')\phi_m(x')\mathrm{d}x'$ and $[\mathbf{K}_{\mathbf{uu}}]_{nm} = \iint k(x, x')\phi_n(x)\phi_m(x')\mathrm{d}x\mathrm{d}x'$.

We see that the interdomain SGPR bypasses the selection of the inducing points $\mathbf{Z} \in \mathbb{R}^M$, and reformulates it with the selection of the basis functions ϕ_i. The basis functions dictate the structure

of the kernel matrices, which in turn modulate the function space of the GP approximation. In contrast, SGPR relies on the inducing points $\mathbf{Z}$, which can shift locations according to the training data. Some examples of basis functions include Fourier basis functions (Hensman et al., 2018) and the Dirac delta function $\delta_{\mathbf{z}_m}$, the latter recovering the standard SGPR inducing variables.

2.3 Online Gaussian processes

In this paper, we focus on online learning with GPs, where data arrives sequentially in batches $(\mathbf{X}_{t_1}, \mathbf{y}_{t_1}), (\mathbf{X}_{t_2}, \mathbf{y}_{t_2}), \ldots$ etc. In the time series regression setting, the data arrives in intervals of $(0, t_1), (t_1, t_2), \ldots$ etc. The online GP learning problem is to sequentially update the GP posterior distribution as data arrives. Suppose that we have already obtained $p_{t_1}(y|f)p_{t_1}(f|\mathbf{u}_{t_1})q_{t_1}(\mathbf{u}_{t_1})$ of the likelihood and variational approximation (with inducing points $\mathbf{Z}_{t_1}$), from the first data batch $(\mathbf{X}_{t_1}, \mathbf{y}_{t_1})$. Online SGPR (OSGPR; Bui et al. (2017)) utilizes the online learning ELBO

$$\begin{aligned}\sum_{i=1}^{n_{t_2}} \mathop{\mathbb{E}}_{q_{t_2}(f_i)} \left[\log p_{t_2}(y_i \mid f_i)\right] + \mathrm{KL}\left(\tilde{q}_{t_2}(\mathbf{u}_{t_1}) \,\|\, p_1(\mathbf{u}_{t_1})\right) \\ - \mathrm{KL}\left(\tilde{q}_{t_2}(\mathbf{u}_{t_2}) \,\|\, q_{t_1}(\mathbf{u}_{t_1})\right) - \mathrm{KL}\left(q_{t_2}(\mathbf{u}_{t_2}) \,\|\, p_{t_2}(\mathbf{u}_{t_2})\right),\end{aligned} \tag{3}$$

where $y_i \in \mathbf{y}_{t_2}$ for $i = 1, \ldots, n_{t_2}$ and $\tilde{q}_{t_2}(\mathbf{u}_{t_1}) := \int p_{t_2}(\mathbf{u}_{t_1}|\mathbf{u}_{t_2})q_{t_2}(\mathbf{u}_{t_2})\mathrm{d}\mathbf{u}_{t_2}$. Note that subscripts $()_{t_1}$ are utilizing hyperparameters from the first batch, and $()_{t_2}$ are hyperparameters from the second batch that are to be optimized with the ELBO. Theoretically, Bui et al. (2017) show that the OSGPR method achieves the same posterior distribution as the batch SGPR method when the inducing points are fixed. Unfortunately, for online time series regression, OSGPR may not capture the long-term memory in the time series data since as new data arrives, it is not guaranteed that the inducing points after optimization can sufficiently cover all the previous tasks' data domains.

2.4 HiPPO: recurrent memory with optimal polynomial projections

The HiPPO framework (Gu et al., 2020) provides mathematical foundations for compressing continuous-time signals into finite-dimensional memory states through optimal polynomial projections. Given a time series $y(t)$, HiPPO maintains a memory state $c(t) \in \mathbb{R}^M$ that optimally approximates the historical signal $\{y(s)\}_{s \le t}$. The framework consists of a time-dependent measure $\omega^{(t)}(s)$ over $(-\infty, t]$ that defines input importance, along with normalized polynomial basis functions $\{g_n^{(t)}(s)\}_{n=0}^{M-1}$ that are orthonormal under $\omega^{(t)}(s)$, satisfying $\int_{-\infty}^{t} g_m^{(t)}(s) g_n^{(t)}(s) \omega^{(t)}(s) \mathrm{d}s = \delta_{mn}$. The historical signal is encoded through projection coefficients given by

$$c_n(t) = \int_{-\infty}^{t} y(s) g_n^{(t)}(s) \omega^{(t)}(s) \mathrm{d}s. \tag{4}$$

This yields the approximation $y(s) \approx \sum_{n=0}^{M-1} c_n(t) g_n(s)$ for $s \in (-\infty, t]$, minimizing the L^2-error $\int_{-\infty}^{t} \|y(s) - \sum_n c_n(t) g_n(s)\|^2 \omega^{(t)}(s) \mathrm{d}s$.

Differentiating $\mathbf{c}(t) := [c_0(t), \ldots, c_{M-1}(t)]^\intercal$ induces a linear ordinary differential equation $\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{c}(t) = \mathbf{A}(t)\mathbf{c}(t) + \mathbf{B}(t)y(t)$ with matrices $\mathbf{A}(t), \mathbf{B}(t)$ encoding measure-basis dynamics. Discretization yields the recurrence $\mathbf{c}_t = \mathbf{A}_t \mathbf{c}_{t-1} + \mathbf{B}_t y_t$ enabling online updates. The structured state space sequential (S4) model (Gu et al., 2022) extends HiPPO with trainable parameters and convolutional kernels, while Mamba (Gu & Dao, 2023; Dao & Gu, 2024) introduces hardware-aware selective state mechanisms, both leveraging HiPPO for efficient long-range memory modeling.

HiPPO supports various measure-basis configurations (Gu et al., 2020; 2023). A canonical instantiation, HiPPO-LegS, uses a uniform measure $\omega^{(t)}(s) = \frac{1}{t}\mathbf{1}_{[0,t]}(s)$ with scaled Legendre polynomials adapted to $[0, t]$, $g_n^{(t)}(s) = (2m+1)^{1/2} P_m\left(\frac{2s}{t} - 1\right)$. This uniform measure encourages HiPPO-LegS to keep the whole past in memory.

3 Interdomain inducing point Gaussian processes with HiPPO

We bridge the HiPPO framework with interdomain Gaussian processes by interpreting HiPPO's state vector defined by time-varying orthogonal projections as interdomain inducing points. This enables adaptive compression of the history of a GP while preserving long-term memory.

3.1 HiPPO AS INTERDOMAIN INDUCING VARIABLES

Recall that in an interdomain setting in Section 2.2, inducing variables are defined through an integral transform against a set of basis functions. Let $f \sim \mathcal{GP}(0, k)$, and consider time-dependent basis functions $\phi_m^{(t)}(x) = g_m^{(t)}(x)\omega^{(t)}(x)$, where $g_m^{(t)}$ are the orthogonal functions of HiPPO and $\omega^{(t)}$ is the associated measure. We define the corresponding interdomain inducing variables as $u_m^{(t)} = \int f(x)\phi_m^{(t)}(x)\mathrm{d}x$. These inducing variables adapt in time, capturing long-range historical information in a compact form via HiPPO's principled polynomial projections.

3.2 ADAPTING THE KERNEL MATRICES OVER TIME

When new observations arrive at later times in a streaming scenario, we must adapt both the prior cross-covariance $\mathbf{K_{fu}}$ and the prior covariance of the inducing variables $\mathbf{K_{uu}}$. In particular, the basis functions in our HiPPO construction evolve with time, so the corresponding kernel quantities also require updates. Below, we describe how to compute and update these matrices at a new time t_2 given their values at time t_1. For clarity, we first discuss $\mathbf{K_{fu}}$, then $\mathbf{K_{uu}}$.

3.2.1 PRIOR CROSS-COVARIANCE $\mathbf{K_{fu}^{(t)}}$

Recall that for a single input x_n, the prior cross-covariance with the m-th inducing variable is $\left[\mathbf{K_{fu}^{(t)}}\right]_{nm} = \int k\left(x_n, x\right)\phi_m^{(t)}(x)\mathrm{d}x$. We can compute the temporal evolution of $\mathbf{K_{fu}^{(t)}}$ in a manner consistent with the HiPPO approach, leveraging the same parameters $\mathbf{A(t)}$ and $\mathbf{B(t)}$. Specifically,

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[\mathbf{K_{fu}^{(t)}}\right]_{n,:} = \mathbf{A(t)}\left[\mathbf{K_{fu}^{(t)}}\right]_{n,:} + \mathbf{B(t)}k\left(x_n, t\right), \tag{5}$$

where $\left[\mathbf{K_{fu}^{(t)}}\right]_{n,:}$ is the n-th row of $\mathbf{K_{fu}^{(t)}}$. The matrices $\mathbf{A(t)}$ and $\mathbf{B(t)}$ depend on the specific choice of the HiPPO measure and basis functions. In our experiments, we employ the HiPPO-LegS variant, whose explicit matrix forms are provided in Appendix A. One then discretizes in t (e.g. using an Euler method or a bilinear transform) to obtain a recurrence update rule.

3.2.2 PRIOR COVARIANCE OF THE INDUCING VARIABLES $\mathbf{K_{uu}^{(t)}}$

The lm-th element of the prior covariance matrix for the inducing variables is given by $\left[\mathbf{K_{uu}^{(t)}}\right]_{\ell m} = \iint k\left(x, x'\right)\phi_\ell^{(t)}(x)\phi_m^{(t)}(x')\mathrm{d}x\mathrm{d}x'$. Since $k(x, x')$ depends on both x and x', a recurrence update rule based on the original HiPPO formulation, which is designed for single integral, can not be obtained directly for $\mathbf{K_{uu}^{(t)}}$. Fortunately, for stationary kernels, Bochner Theorem (Rudin, 1994) can be applied to factorize the double integrals into two separate single integrals, which gives rise to Random Fourier Features (RFF) approximation (Rahimi & Recht, 2007): for a stationary kernel $k(x, x') = k(|x - x'|)$, RFF approximates it with $k(x, x') \approx \frac{1}{N}\sum_{n=1}^{N}\left[\cos\left(w_n x\right)\cos\left(w_n x'\right) + \sin\left(w_n x\right)\sin\left(w_n x'\right)\right]$, where $w_n \sim p(w)$ is the spectral density of the kernel. Substituting this into the double integral factorizes the dependency on x and x', reducing $[\mathbf{K_{uu}^{(t)}}]_{\ell m}$ to addition of products of one-dimensional integrals. Each integral, with the form of either $\int \cos(w_d x)\phi_\ell^{(t)}(x)\mathrm{d}x$ or $\int \sin(w_d x)\phi_\ell^{(t)}(x)\mathrm{d}x$, again corresponds to a HiPPO-ODE in time. By sampling sufficiently many random features, updating them recurrently to time t, and averaging, we obtain an accurate approximation of $\mathbf{K_{uu}^{(t)}}$. The details of the ODE for recurrent updates of the RFF samples appear in Appendix B.1.

Alternatively, one may differentiate $\mathbf{K_{uu}^{(t)}}$ directly with respect to t. This yields a matrix ODE of the form different from the original HiPPO formulation. Empirically, we note that a naive implementation of this approach is numerically unstable. Therefore, we conduct our experiments based on RFF approximation for now. For details, see Appendix B.2.

3.2.3 SEQUENTIAL VARIATIONAL UPDATES

Having obtained $\mathbf{K}_{\mathbf{fu}}^{(t_2)}, \mathbf{K}_{\mathbf{uu}}^{(t_2)}$ at a new time $t_2 > t_1$, we perform variational updates following the online GP framework described in Section 2.3. This ensures the posterior at time t_2 remains consistent with both the new data and the previous posterior at time t_1, based on $\mathbf{K}_{\mathbf{fu}}^{(t_1)}, \mathbf{K}_{\mathbf{uu}}^{(t_1)}$. Overall, this procedure endows interdomain HiPPO-based GPs with the ability to capture long-term memory online. By viewing the induced kernel transforms as ODEs in time, we efficiently preserve the memory of past observations while adapting our variational posterior in an online fashion.

4 EXPERIMENTS

We evaluate OHSGPR on online time series regression tasks. For all experiments, we construct inducing variables based on HiPPO-LegS (Gu et al., 2020). We consider the following experimental setup:

Datasets. We consider two benchmarks:

- **Solar Irradiance** (Lean, 2004). Following Gal & Turner (2015): we scale the original outputs with its standard deviation and removed 5 segments of length 20 to create the test set.
- **Audio signal prediction dataset** (Bui & Turner, 2014). It was produced from the TIMIT database (Garifolo et al., 1993) and we shifted the signal down to the baseband and select a segment of length 18,000 to construct interleaved training and testing sets consisting of 9,000 time stamps. Moreover, we linearly scale the input time stamps to the range [0, 10].

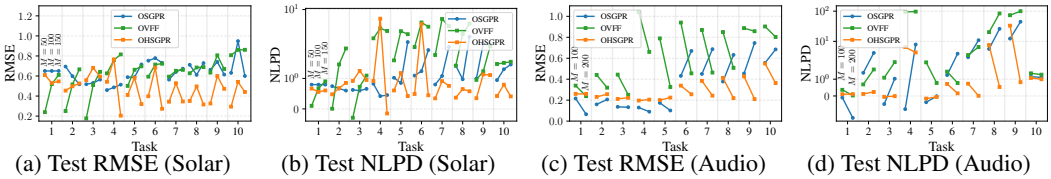
We construct online learning tasks by splitting each dataset into 10 sequential partitions with an equal number of training instances, and the model is updated incrementally as new task/split becomes available without re-visiting the data from past tasks/splits.

Baseline. We compare OHSGPR with OSGPR (based on standard inducing points) (Bui et al., 2017), and OVFF (OSGPR based on variational Fourier feature (VFF), an interdomain inducing point approach from Hensman et al. (2018)) across different numbers of inducing variables M. For all experiments, we use RBF kernel with a tunable length-scale for OSGPR and OHSGPR, and Matérn-$\frac{5}{2}$ kernel with a tunable length-scale for OVFF, since VFF is tailored specifically to Matérn kernels. It is also worth noting that, in contrast with OHSGPR where after each intermediate task, the kernel matrices only need to be evolved up to the maximum times tamp observed so far, OVFF requires computing kernel matrices as integrals over a predefined interval that covers the whole range of the time stamps from all tasks (including the unobserved ones), which is impractical in real-world online learning tasks. Here, for OVFF, we set the two edges of this interval to be the minimum and maximum time stamp among the data points from all the 10 tasks, respectively. All the models are trained using ADAM (Kingma & Ba, 2015) with a learning rate of 0.01 and 5000 (by default) iterations per task. We set the default RFF sample size during training and testing to be 500 and 5000, respectively. The total number of discretization steps (for all 10 tasks) is 160 by default.

Evaluations & metrics. After observing task i, we report Root Mean Squared Error (RMSE) and Negative Log Predictive Density (NLPD) over all the past test points from task 1 to task $i-1$. Additionally, we report wall-clock accumulated running time for learning all the tasks for OHSGPR and the baseline methods to demonstrate the computational efficiency of OHSGPR, compared with OSGPR, in the case where the kernel hyperparameters and the likelihood variance are fixed.

4.1 RESULTS BASED ON TRAINABLE KERNEL HYPERPARAMETERS AND LIKELIHOOD VARIANCE

Figure 1 shows test RMSE and NLPD (over the past data) of OHSGPR and the baseline methods during online learning over the 10 tasks constructed from the Solar Irradiance and Audio dataset. Overall, OHSGPR outperforms OSGPR and OVFF, and whilst OHSGPR does not outperform OSGPR after seeing the first few tasks, OHSGPR demonstrates superior performance as more and more tasks are revealed. In particular, OHSGPR consistently achieves better overall performance than OSGPR and OVFF after seeing the final task, suggesting OHSGPR is able to maintain long-term memory, while OSGPR and OVFF may suffer from catastrophic forgetting. We find that as OSGPR



(a) Test RMSE (Solar) (b) Test NLPD (Solar) (c) Test RMSE (Audio) (d) Test NLPD (Audio)

Figure 1: Comparison of OSGPR, OVFF and OHSGPR (ours) on the Solar Irradiance and Audio signal prediction datasets across 10 tasks. Log-scale is used in the y-axes of the plots for NLPD

seeing more and more tasks, it may exhibit a noticeable degradation in performance over the initial few data regions. In particular, the inducing points tend to move to the regions where the later tasks live after training, and the prediction of OSGPR in the initial regions without sufficient inducing points becomes more erratic. In contrast, OHSGPR maintains consistent performance across both early and recent time periods, suggesting effective preservation of long-term memory through its HiPPO-based memory mechanism. We illustrate this catastrophic forgetting pathology of OSGPR on Solar Irradiance in Figure 2 in Appendix C.

4.2 Wall-clock accumulated running time of OHSGPR with fixed kernel

Table 1 shows the accumulated wall-clock time for OSGPR (varying iterations per task), OVFF and OHSGPR (varying discretization steps and RFF samples) with fixed kernel hyperparameters and likelihood variance (from a full GP trained on the first two tasks). Unlike OSGPR, which must iteratively optimize inducing points, OHSGPR and OVFF, based on interdomain inducing points bypass this cumbersome optimization. In particular, OHSGPR recurrently evolves $\mathbf{K_{fu}}$ and $\mathbf{K_{uu}}$ for each new task with no training required. As a result, OHSGPR and OVFF run significantly faster, adapting to all tasks within a couple of seconds for Solar Irradiance and Audio data. Appendix D compares fixed-kernel models to trainable-kernel ones and finds that they are as competitive, if not superior, in performance. In addition, OHSGPR still outperforms OSGPR in preserving long-term memory in this comparison. Interestingly, while OVFF underperforms with trainable kernel (see Section 4.1), it is competitive with OHSGPR when kernel hyperparameters and likelihood variance are fixed, which suggests hyperparameter tuning for VFF with the online learning ELBO may be challenging.

Table 1: Wall-clock accumulated runtime for learning all the 10 tasks on a single NVIDIA RTX3090 GPU in seconds), of OSGPR, OVFF and OHSGPR models with fixed kernel on the Solar dataset. In addition to different number M of inducing points, we also compare OHSGPRs with 160, 320, and 480 total number of discretization steps and 500, and 5000 RFF samples.

Method	**Solar Irradiance**			**Audio Data**	
	M			M	
	50	100	150	100	200
OSGPR (1000 iterations)	134	134	140	144	199
OSGPR (5000 iterations)	672	675	698	720	997
OVFF	0.288	0.313	0.349	0.295	0.356
OHSGPR (160 disc, 500 RFF)	0.262	0.289	0.333	0.282	0.402
OHSGPR (320 disc, 500 RFF)	0.289	0.334	0.401	0.312	0.485
OHSGPR (480 disc, 500 RFF))	0.301	0.346	0.410	0.353	0.576
OHSGPR (160 disc, 5000 RFF	0.310	0.447	0.650	0.739	1.271
OHSGPR (320 disc, 5000 RFF)	0.388	0.629	0.938	0.902	1.822
OHSGPR (480 disc, 5000 RFF)	0.450	0.787	1.211	1.044	2.369

5 Conclusion

We introduce OHSGPR, a novel online Gaussian process model that leverages the HiPPO framework for robust long-range memory in online regression. By interpreting HiPPO's time-varying orthogonal projections as interdomain GP basis functions, we establish a link between SSMs and GPs. This connection allows OHSGPR to harness HiPPO's efficient ODE-based recurrent updates while preserving GP-based uncertainty quantification. Empirical results on time series regression tasks show that OHSGPR significantly outperforms existing online GP methods, especially in scenarios requiring long-term memory. Moreover, with fixed kernel hyperparameters, its recurrence-based kernel updates yield far lower computational overhead than OSGPR's sequential inducing point optimization. This efficient streaming capability and preservation of historical information make OHSGPR well-suited for real-world applications demanding both speed and accuracy. Future work includes exploring alternative basis and measure selections and integrating OHSGPR with deep learning (e.g., enabling online learning in GPVAE models (Fortuin et al., 2020; Jazbec et al., 2021)).

REFERENCES

Thang D. Bui and Richard E. Turner. Tree-structured gaussian process approximations. In *Advabooktitleces in Neural Information Processing Systems*, 2014.

Thang D. Bui, Cuong V. Nguyen, and Richard E. Turner. Streaming Sparse Gaussian Process Approximations. In *Advances in Neural Information Processing Systems*, 2017.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *International Conference on Machine Learning (ICML)*, 2024.

Vincent Fortuin, Dmitry Baranchuk, Gunnar Rätsch, and Stephan Mandt. Gp-vae: Deep probabilistic time series imputation. In *International Conference on Artificial Intelligence and Statistics*, pp. 1651–1661. PMLR, 2020.

Yarin Gal and Richard E. Turner. Improving the gaussian process sparse spectrum approximation by representing uncertainty in frequency inputs. In *International Conference on Machine Learning (ICML)*, 2015.

J. Garifolo, L. Lamel, W. Fisher, J. Fiscus, D. Pallett, N. Dahlgren, and V. Zue. Timit acoustic-phonetic continuous speech corpus ldc93s1. In *Philadelphia: Linguistic Data Consortium*, 1993.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. HiPPO: Recurrent Memory with Optimal Polynomial Projections. In *Advances in Neural Information Processing Systems*, 2020.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *The International Conference on Learning Representations (ICLR)*, 2022.

Albert Gu, Isys Johnson, Aman Timalsina, Atri Rudra, and Christopher Re. How to train your HIPPO: State space models with generalized orthogonal basis projections. In *International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=klK17OQ3KB`.

James Hensman, Nicolò Fusi, and Neil D. Lawrence. Gaussian Processes for Big Data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, UAI'13, pp. 282–290, Arlington, Virginia, USA, 2013. AUAI Press.

James Hensman, Alexander Matthews, and Zoubin Ghahramani. Scalable variational Gaussian process classification. In *Artificial Intelligence and Statistics*, pp. 351–360. PMLR, 2015a.

James Hensman, Alexander G Matthews, Maurizio Filippone, and Zoubin Ghahramani. MCMC for variationally sparse Gaussian processes. *Advances in Neural Information Processing Systems*, 28, 2015b.

James Hensman, Nicolas Durrande, and Arno Solin. Variational fourier features for gaussian processes. *Journal of Machine Learning Research*, 18(151):1–52, 2018.

Metod Jazbec, Matt Ashman, Vincent Fortuin, Michael Pearce, Stephan Mandt, and Gunnar Rätsch. Scalable gaussian process variational autoencoders. In *International Conference on Artificial Intelligence and Statistics*, pp. 3511–3519. PMLR, 2021.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Judith Lean. Solar irradiance reconstruction. In *Data contribution series # 2004-035, IGBP PAGES/World Data Center for Paleoclimatology NOAA/NGDC Paleoclimatology Program, Boulder, CO, USA*, 2004.

Felix Leibfried, Vincent Dutordoir, ST John, and Nicolas Durrande. A tutorial on sparse Gaussian processes and variational inference. *arXiv preprint arXiv:2012.13962*, 2020.

Miguel Lázaro-Gredilla and Anibal Figueiras-Vidal. Inter-domain Gaussian Processes for Sparse Inference using Inducing Features. In *Advances in Neural Information Processing Systems*, 2009.

Ali Rahimi and Benjamin Recht. Random Features for Large-Scale Kernel Machines. In *Advances in Neural Information Processing Systems*, 2007.

Stephen Roberts, Michael Osborne, Mark Ebden, Steven Reece, Neale Gibson, and Suzanne Aigrain. Gaussian processes for time-series modelling. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 371(1984):20110550*, 2013.

W. Rudin. Fourier analysis on groups. *Wiley Classics Library. Wiley-Interscience New York, reprint edition*, 1994.

Michalis Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial intelligence and statistics*, pp. 567–574. PMLR, 2009.

Mark Van der Wilk, Vincent Dutordoir, ST John, Artem Artemev, Vincent Adam, and James Hensman. A framework for interdomain and multioutput Gaussian processes. *arXiv preprint arXiv:2003.01115*, 2020.

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

William Wilkinson, Arno Solin, and Vincent Adam. Sparse algorithms for Markovian Gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, pp. 1747–1755. PMLR, 2021.

A HiPPO-LegS matrices

Here we provide the explicit form of matrices used in our implementation of HiPPO-LegS (Gu et al., 2020). For a given time t, the measure $\omega^{(t)}(x) = \frac{1}{t}\mathbf{1}_{[0,t]}(x)$ and basis functions $\phi_m^{(t)}(x) = g_m^{(t)}(x)\omega^{(t)}(x) = \frac{\sqrt{2m+1}}{t}P_m\left(\frac{2x}{t}-1\right)\mathbf{1}_{[0,t]}(x)$ are used, where $P_m(\cdot)$ is the m-th Legendre polynomial and $\mathbf{1}_{[0,t]}(x)$ is the indicator function on the interval $[0,t]$. These basis functions are orthonormal, i.e.,

$$\int_0^t \frac{g_m^{(t)}(x)g_n^{(t)}(x)}{t}\mathrm{d}x = \delta_{mn} \tag{6}$$

Following Gu et al. (2020), the HiPPO-LegS framework maintains a coefficient vector $\mathbf{c}(t) \in \mathbb{R}^{M\times 1}$ that evolves according to the ODE:

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{c}(t) = \mathbf{A}(t)\mathbf{c}(t) + \mathbf{B}(t)f(t) \tag{7}$$

where $f(t)$ is the input signal at time t. The matrices $\mathbf{A}(t) \in \mathbb{R}^{M\times M}$ and $\mathbf{B}(t) \in \mathbb{R}^{M\times 1}$ are given by:

$$[\mathbf{A}(\mathbf{t})]_{nk} = \begin{cases} -\frac{1}{t}\sqrt{(2n+1)(2k+1)} & \text{if } n > k \\ -\frac{n+1}{t} & \text{if } n = k \\ 0 & \text{if } n < k \end{cases} \tag{8}$$

and

$$[\mathbf{B}(\mathbf{t})]_n = \frac{\sqrt{2n+1}}{t} \tag{9}$$

These matrices govern the evolution of the basis function coefficients over time, where the factor $1/t$ reflects the time-dependent scaling of the basis functions to the adaptive interval $[0,t]$. When discretized, this ODE yields the recurrence update used in our implementation.

B Computing prior covariance of the inducing variables $\mathbf{K}_{\mathbf{uu}}^{(t)}$

We provide the detailed derivation for the following two approaches when the inducing functions are defined via HiPPO-LegS. Recall that

$$[\mathbf{K}_{\mathbf{uu}}^{(t)}]_{\ell m} = \iint k(x,x')\phi_\ell^{(t)}(x)\phi_m^{(t)}(x')\mathrm{d}x\mathrm{d}x', \tag{10}$$

where $\phi_\ell^{(t)}(x) = g_\ell^{(t)}(x)\,\omega^{(t)}(x)$ are the time-varying basis functions under the HiPPO-LegS framework.

B.1 RFF approximation

For stationary kernels (e.g. an RBF kernel), Bochner's theorem permits a representation of the kernel as an expectation with respect to its spectral measure:

$$k(x,x') = \mathbb{E}_{p(w)}\Big[\cos(wx)\,\cos(wx') + \sin(wx)\,\sin(wx')\Big], \tag{11}$$

where $p(w)$ is the spectral density of the kernel.

Substituting this into equation 10, for a given Monte Carlo sample of w we define

$$Z_{w,\ell}^{(t)} = \int \cos(wx)\phi_\ell^{(t)}(x)\,\mathrm{d}x, \quad Z_{w,\ell}^{\prime(t)} = \int \sin(wx)\phi_\ell^{(t)}(x)\,\mathrm{d}x, \tag{12}$$

and

$$\mathbf{Z}_w^{(t)} = \left[Z_{w,1}^{(t)},\cdots,Z_{w,M}^{(t)}\right]^\intercal, \quad \mathbf{Z}_w^{\prime(t)} = \left[Z_{w,1}^{\prime(t)},\cdots,Z_{w,M}^{\prime(t)}\right]^\intercal. \tag{13}$$

Collecting N Monte Carlo samples, we form the feature matrix

$$\mathbf{Z}^{(t)} = \begin{bmatrix} \mathbf{Z}_{w_1}^{(t)} & \mathbf{Z}_{w_2}^{(t)} & \cdots & \mathbf{Z}_{w_N}^{(t)} & \mathbf{Z}_{w_1}^{\prime(t)} & \mathbf{Z}_{w_2}^{\prime(t)} & \cdots & \mathbf{Z}_{w_N}^{\prime(t)} \end{bmatrix}, \tag{14}$$

and the RFF approximation of the covariance is

$$\mathbf{K}_{\mathbf{uu}}^{(t)} \approx \frac{1}{N} \mathbf{Z}^{(t)} \left(\mathbf{Z}^{(t)}\right)^{\intercal}. \tag{15}$$

Moreover, as before, the evolution of $\mathbf{Z}_w^{(t)}$ and $\mathbf{Z}_w^{\prime(t)}$ is governed by the ODE

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{Z}_w^{(t)} = \mathbf{A}(t)\,\mathbf{Z}_w^{(t)} + \mathbf{B}(t)\,h(t), \quad \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{Z}_w^{\prime(t)} = \mathbf{A}(t)\,\mathbf{Z}_w^{\prime(t)} + \mathbf{B}(t)\,h'(t), \tag{16}$$

with $h(t) = \cos(wt)$ and $h'(t) = \sin(wt)$ and these ODEs can be solved in parallel for different Monte Carlo samples, thereby fully exploiting the computational advantages of GPU architectures.

B.2 Direct ODE evolution

Differentiating $[\mathbf{K}_{\mathbf{uu}}^{(t)}]_{\ell m}$ with respect to t gives

$$\frac{d}{dt}\,[\mathbf{K}_{\mathbf{uu}}^{(t)}]_{\ell m} = \iint k(x,x')\frac{\mathrm{d}}{\mathrm{d}t}\left[\phi_\ell^{(t)}(x)\phi_m^{(t)}(x')\right]\mathrm{d}x\mathrm{d}x'. \tag{17}$$

Applying the product rule:

$$\frac{\mathrm{d}}{\mathrm{d}t}[\mathbf{K}_{\mathbf{uu}}^{(t)}]_{\ell m} = \iint k(x,x')\frac{\mathrm{d}}{\mathrm{d}t}\phi_\ell^{(t)}(x)\phi_m^{(t)}(x')\mathrm{d}x\mathrm{d}x' + \iint k(x,x')\phi_\ell^{(t)}(x)\frac{\mathrm{d}}{\mathrm{d}t}\phi_m^{(t)}(x')\mathrm{d}x\mathrm{d}x'. \tag{18}$$

In HiPPO-LegS, each $\phi_\ell^{(t)}(x)$ obeys an ODE governed by scaled Legendre polynomials on $[0,t]$ and a Dirac boundary term at $x = t$. Concretely,

$$\frac{\mathrm{d}}{\mathrm{d}t}\phi_\ell^{(t)}(x) = -\frac{\sqrt{2\ell+1}}{t}\left[\frac{\ell+1}{\sqrt{2\ell+1}}\phi_\ell^{(t)}(x) + \sqrt{2\ell-1}\phi_{\ell-1}^{(t)}(x)\right] + \frac{1}{t}\,\delta_t(x), \tag{19}$$

where $\delta_t(x)$ is the Dirac delta at $x = t$.

Substituting this expression into the integrals yields the boundary terms $\int k(t,x')\,\phi_m^{(t)}(x')\,\mathrm{d}x'$, along with lower-order terms involving $[\mathbf{K}_{\mathbf{uu}}^{(t)}]_{\ell-1,m}$, etc. Summarizing in matrix form leads to

$$\frac{\mathrm{d}}{\mathrm{d}t}\,\mathbf{K}_{\mathbf{uu}}^{(t)} = -\frac{1}{t}\left[\mathbf{A}\,\mathbf{K}_{\mathbf{uu}}^{(t)} + \mathbf{K}_{\mathbf{uu}}^{(t)}\mathbf{A}^{\intercal}\right] + \frac{1}{t}\left[\tilde{\mathbf{B}}(t) + \tilde{\mathbf{B}}(t)^{\intercal}\right], \tag{20}$$

where $\mathbf{K}_{\mathbf{uu}}^{(t)} \in \mathbb{R}^{M\times M}$ has entries $[\mathbf{K}_{\mathbf{uu}}^{(t)}]_{\ell m}$, $\mathbf{A} \in \mathbb{R}^{M\times M}$ is the same lower-triangular matrix from the HiPPO-LegS framework defined in equation equation 8, and $\tilde{\mathbf{B}}(t) \in \mathbb{R}^{M\times M}$ is built from the boundary contributions as

$$\tilde{\mathbf{B}}(t) = \mathbf{c}(t)\mathbf{1}_M, \tag{21}$$

where $\mathbf{c}(t) \in \mathbb{R}^{M\times 1}$ is the coefficient vector and $\mathbf{1}_M \in \mathbb{R}^{1\times M}$ is a row vector of ones of size M with

$$c_\ell(t) = \int k\,(t,x)\;\phi_\ell^{(t)}(x)\,\mathrm{d}x. \tag{22}$$

After discretizing in t (e.g. an Euler scheme), one repeatedly updates $\mathbf{K}_{\mathbf{uu}}^{(t)}$ and the boundary vector $\mathbf{c}(t)$ over time.

B.2.1 Efficient computation of $\tilde{\mathbf{B}}(t)$

Computing $\tilde{\mathbf{B}}(t)$ directly at each time step requires evaluating M integrals, which can be computationally intensive, especially when t changes incrementally and we need to update the matrix $\tilde{\mathbf{B}}(t)$ repeatedly.

To overcome this inefficiency, we propose an approach that leverages the HiPPO framework to compute $\tilde{\mathbf{B}}(t)$ recursively as s evolves. This method utilizes the properties of stationary kernels and the structure of the Legendre polynomials to enable efficient updates.

Leveraging Stationary Kernels Assuming that the kernel $k(x,t)$ is stationary, it depends only on the difference $d = |x - t|$, so $k(x,t) = k(d)$. In our context, since we integrate over $x \in [t_{\text{start}}, t]$ with $x \leq t$, we have $d = t - x \geq 0$. Therefore, we can express $k(x,t)$ as a function of d over the interval $[0, t - t_{\text{start}}]$:

$$k(x,t) = k(t-x) = k(d), \quad \text{with} \quad d \in [0, t - t_{\text{start}}]. \tag{23}$$

Our goal is to approximate $k(d)$ over the interval $[0, t - t_{\text{start}}]$ using the orthonormal Legendre basis functions scaled to this interval. Specifically, we can represent $k(d)$ as

$$k(d) \approx \sum_{m=0}^{M-1} \tilde{c}_m(t)\, g_m^{(t)}(d), \tag{24}$$

where $g_m^{(t)}(d)$ are the Legendre polynomials rescaled to the interval $[0, t - t_{\text{start}}]$.

Recursive Computation via HiPPO-LegS To efficiently compute the coefficients $\tilde{c}_m(t)$, we utilize the HiPPO-LegS framework, which provides a method for recursively updating the coefficients of a function projected onto an orthogonal basis as the interval expands. In our case, as t increases, the interval $[t_{\text{start}}, t]$ over which $k(d)$ is defined also expands, and we can update $\tilde{c}_m(t)$ recursively.

Discretizing time with step size Δt and indexing $t_k = t_{\text{start}} + k\Delta t$, the update rule using the Euler method is:

$$\tilde{\mathbf{c}}_{k+1} = \left(\mathbf{I} - \frac{1}{k}\mathbf{A}\right)\tilde{\mathbf{c}}_k + \frac{1}{k}\mathbf{B}\, k(t_k), \tag{25}$$

where $\tilde{\mathbf{c}}_k = [\tilde{c}_0(t_k), \tilde{c}_1(t_k), \ldots, \tilde{c}_{M-1}(t_k)]^\intercal$, and $\mathbf{A} \in \mathbb{R}^{M \times M}$ and $\mathbf{B} \in \mathbb{R}^M$ are again matrices defined by the HiPPO-LegS operator (Gu et al., 2020).

Accounting for Variable Transformation and Parity The change of variables from x to $d = t - x$ introduces a reflection in the function domain. Since the Legendre polynomials have definite parity, specifically,

$$P_m(-x) = (-1)^m P_m(x), \tag{26}$$

we need to adjust the coefficients accordingly when considering the reflected function.

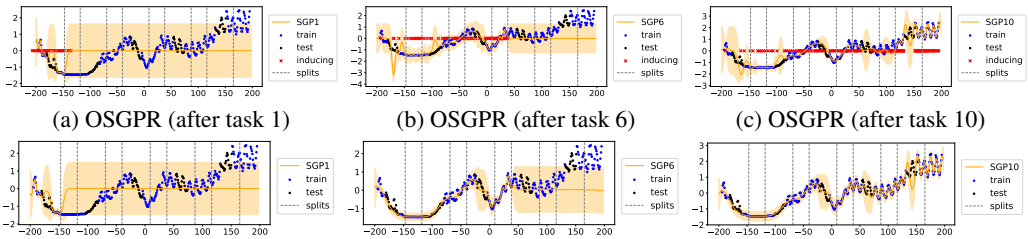
As a result of this reflection, when projecting $k(d)$ onto the Legendre basis, the coefficients $\tilde{c}_m(t)$ computed via the HiPPO-LegS updates will correspond to a reflected version of the function. To account for this, we apply a parity correction to the coefficients. Specifically, the corrected coefficients $c_m(t)$ are related to $\tilde{c}_m(t)$ by a sign change determined by the degree m:

$$c_m(t) = (-1)^m \tilde{c}_m(t). \tag{27}$$

This parity correction ensures that the computed coefficients properly represent the function over the interval $[t_{\text{start}}, t]$ without the effect of the reflection.

By computing $\mathbf{c}(t)$ recursively as t evolves, we can efficiently update $\tilde{\mathbf{B}}(t) = \mathbf{c}(t)[1, \ldots, 1]$ at each time step without the need to evaluate the integrals directly. This approach significantly reduces the computational burden associated with updating $\tilde{\mathbf{B}}(t)$ and allows for efficient computation of $\mathbf{K}_{\mathbf{uu}}^{(t)}$ via the ODE.

C ILLUSTRATION OF THE CATASTROPHIC FORGETTING PATHOLOGY OF OSGPR



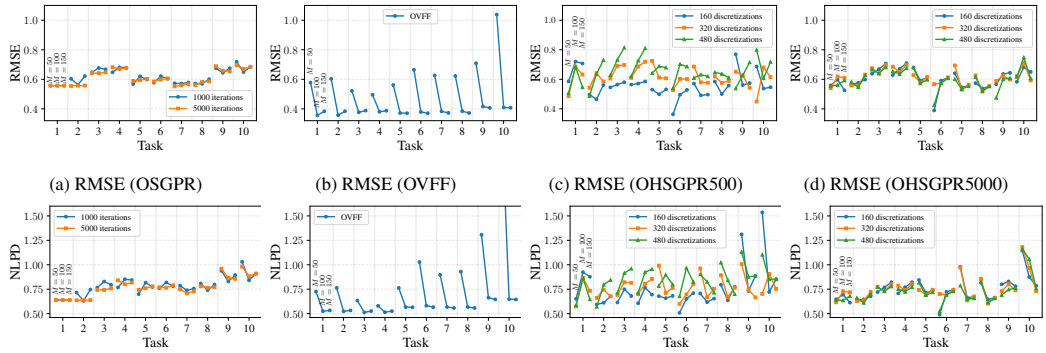
(a) OSGPR (after task 1) (b) OSGPR (after task 6) (c) OSGPR (after task 10)

(d) OHSGPR (after task 1) (e) OHSGPR (after task 6) (f) OHSGPR (after task 10)

Figure 2: Predictive mean ± 2 standard deviation of OSGPR and OHSGPR after tasks 1, 6, and 10 of the Solar dataset. $M = 150$ inducing variables are used.

D PERFORMANCE OF OHSGPR AND BASELINES WITH FIXED KERNEL

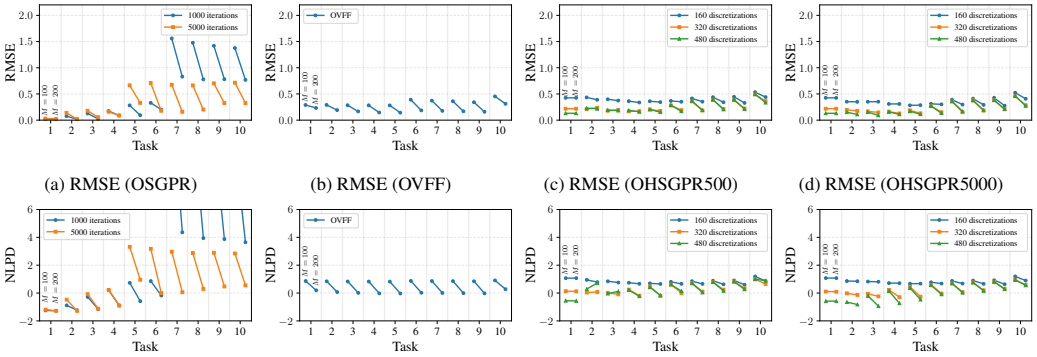
In Figure 3 and 4, we present the performance, measured by RMSE and NLPD, of OSGPR, OVFF and OHSGPR models with fixed kernel shown in Section 4.2 for Solar Irradiance and Audio dataset, respectively. OHSGPR again outperforms OSGPR in terms of long-term memory preservation and achieves better RMSE and NLPD at the later stage where most of the tasks have been revealed. Interestingly, the overall performance of these fixed-kernel models is competitive or even better with the models with trainable kernel in Section 4. The performance of OVFF with fixed kernel is significantly better than OVFF with trainable kernel, suggesting it may be hard to find good hyperparamters with the online ELBO objective for OVFF. Furthermore, compared to OHSGPR with a trainable kernel, OHSGPR with a fixed kernel achieves better performance on the audio dataset when we increase the total number of discretizations from 160 to 320 or 480. The best RMSE and NLPD for OHSGPR after the final task for audio data in this work (0.2738 and 0.5510, respectively) are achieved by a fixed-kernel OHSGPR with 480 total discretization steps (Figure 4d and 4h), suggesting in practice, for some applications, if the kernel hyperparameters are already in a good range, further optimization over it might show diminishing returns and in this case the effect of fine-grained discretization might play a more critical role in the performance.



(a) RMSE (OSGPR) (b) RMSE (OVFF) (c) RMSE (OHSGPR500) (d) RMSE (OHSGPR5000)

(e) NLPD (OSGPR) (f) NLPD (OVFF) (g) NLPD (OHSGPR500) (h) NLPD (OHSGPR5000)

Figure 3: Test RMSE and NLPD results for OSGPR, OVFF and OHSGPR (with RFF sample size of 500 or 5000) on the Solar Irradiance dataset (10 splits). Results are shown for three values of M ($M = 50, 100, 150$), and different numbers of total discretization steps for OHSGPR. The kernel hyperparameters are fixed to those obtained from a full GP trained on splits 1-2.



(a) RMSE (OSGPR) (b) RMSE (OVFF) (c) RMSE (OHSGPR500) (d) RMSE (OHSGPR5000)

(e) NLPD (OSGPR) (f) NLPD (OVFF) (g) NLPD (OHSGPR500) (h) NLPD (OHSGPR5000)

Figure 4: Test RMSE and NLPD results for OSGPR, OVFF and OHSGPR (with RFF sample size of 500 or 5000) on the Audio signal prediction dataset (10 splits). Results are shown for two values of M ($M = 100, 200$), and different numbers of total discretization steps for OHSGPR. The kernel hyperparameters are fixed to those obtained from a full GP trained on splits 1-2.