Exact Uncertainty Propagation via Gaussian Process Neurons

Qiuxian Meng[1] **Yongyou Zhang**[1]

[1]Xiamen University

Abstract

Exact marginalization in deep Gaussian processes (DGPs) is analytically intractable, typically requiring computationally costly sampling-based approximations. We introduce the *Gaussian process neuron*, a computational unit built on a novel *Wasserstein exponential kernel*. By defining a composable map between factorized Gaussian measures, this unit enables analytical uncertainty propagation; concurrently, optimizing deterministic inducing variables via maximum a posteriori estimation ensures strict probabilistic rigor. By eliminating sampling variance and the depth-scaling computational bottlenecks of Monte Carlo methods, our approach enables exact probabilistic variants of diverse architectures, including DGPs, multilayer perceptrons, and Kolmogorov-Arnold Networks. Empirically, these networks demonstrate superior uncertainty calibration relative to state-of-the-art approximate DGP frameworks, alongside high-dimensional scalability and robust compositional expressiveness.

1 INTRODUCTION

Gaussian processes (GPs) [Rasmussen and Williams, 2005] provide a rigorous probabilistic framework for approximating functions with well-calibrated uncertainty. However, the shallow architectural priors limit their capacity to represent highly complex or compositional features. Deep Gaussian processes (DGPs) [Damianou and Lawrence, 2013] address this limitation by hierarchically composing multiple sparse Gaussian processes (SGPs) [Hensman et al., 2013]. Nevertheless, the composition of latent stochastic functions in DGPs renders exact marginalization analytically intractable.

To circumvent this intractability, the standard approach relies on approximate inference, such as Markov chain Monte Carlo (MCMC) [Havasi et al., 2018] or variational inference (VI) combined with Monte Carlo (MC) sampling [Salimbeni and Deisenroth, 2017, Yu et al., 2019, Xu et al., 2024]. However, these sampling-based techniques introduce substantial computational cost scaling multiplicatively with the number of MC samples. Furthermore, as model depth increases, achieving low-variance gradient estimates requires a correspondingly large sample size.

To avoid the burdens of sampling, recent research has explored parametric adaptations for deterministic optimization. Interpretable DGPs [Lu et al., 2020], for example, approximate the DGP posterior by calculating exact moments, but this is only viable in the low-variance regime and lacks the scalable sparse approximations utilized by other DGP models. Another prominent deterministic approach is the deep sigma point process (DSPP) [Jankowiak et al., 2020b], which explicitly optimizes a finite Gaussian mixture derived via a learnable quadrature across the layers. While these methods successfully circumvent MC sampling, they heavily rely on structural approximations or restrictive assumptions that fundamentally limit their topological generality and expressive capacity.

In this work, we propose a paradigm shift from approximate stochastic sampling to exact, deterministic uncertainty propagation. We introduce the *Gaussian process neuron*, a probabilistic computational unit that enables exact uncertainty propagation across factorized Gaussian measures. By incorporating a novel *Wasserstein exponential kernel* (WEK), this unit analytically maps input Gaussian moments to output moments, thereby defining a composable map between factorized Gaussians. This fully parametric formulation completely eliminates sampling variance and avoids scaling issues incurred by MC sampling, remaining computationally efficient regardless of model depth.

The analytical tractability of this unit allows it to practically serve as a modular building block for diverse topological structures. While constructing probabilistic variants of architectures like multilayer perceptrons (MLPs) and

Proceedings of the 42nd Conference on Uncertainty in Artificial Intelligence (UAI 2026), PMLR 337:4497–4513.

Kolmogorov-Arnold Networks (KANs) [Liu et al., 2025] is theoretically possible with standard DGPs, sampling-based inference renders them computationally prohibitive. In contrast, the proposed unit makes exact probabilistic variants of these classical topologies computationally feasible. Our empirical evaluations confirm that these exact formulations achieve superior uncertainty calibration compared to existing approximate GP models on regression tasks, scale efficiently to high-dimensional image classification, and accurately represent symbolic functions. Codes are available at github.com/Meng-QX/Gaussian-Process-Neuron.

2 BACKGROUND

Gaussian Process. A Gaussian process (GP) [Rasmussen and Williams, 2005] represents a distribution denoted by $\mathcal{GP}$ over a real-valued function $f : \mathcal{X} \to \mathbb{R}$. A GP generalizes multivariate Gaussian distributions to functions such that the function values at any finite subset of $\mathcal{X}$ are jointly Gaussian distributed. Formally, a GP is specified by a mean function $m : \mathcal{X} \to \mathbb{R}$ and a positive definite covariance function (a.k.a. kernel) $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$,

$$f(\cdot) \sim \mathcal{GP}\big(m(\cdot), k(\cdot, \cdot')\big), \tag{1}$$

where $m(\cdot) = \mathbb{E}[f(\cdot)]$ and $k(\cdot, \cdot') = \mathrm{Cov}(f(\cdot), f(\cdot'))$. The function values $\mathbf{u}$ evaluated at inputs $\mathbf{Z} = \{\boldsymbol{z}_m\}_{m=1}^M \subset \mathcal{X}$ follow an M-variate Gaussian distribution

$$p(\mathbf{u}|\mathbf{Z}) = \mathcal{N}(\mathbf{u}|\mathbf{m_u}, \mathbf{K_{uu}}), \tag{2}$$

where $\mathbf{m_u}[m] = m(\boldsymbol{z}_m)$ and $\mathbf{K_{uu}}[m, m'] = k(\boldsymbol{z}_m, \boldsymbol{z}_{m'})$.

Sparse Gaussian Process. The uncountably infinite set of random variables represented by a GP can be partitioned into a finite subset $\mathbf{u} = f(\mathbf{Z})$ given by Equation 2 and its complement evaluated at $\mathcal{X} \setminus \mathbf{Z}$,

$$\begin{bmatrix} f(\cdot) \\ \mathbf{u} \end{bmatrix} \sim \mathcal{GP}\left(\begin{bmatrix} m(\cdot) \\ \mathbf{m_u} \end{bmatrix}, \begin{bmatrix} k(\cdot, \cdot') & k(\cdot, \mathbf{Z}) \\ k(\mathbf{Z}, \cdot') & \mathbf{K_{uu}} \end{bmatrix}\right), \tag{3}$$

where $k(\cdot, \mathbf{Z})[m] = k(\cdot, \boldsymbol{z}_m)$ and $k(\mathbf{Z}, \cdot')[m] = k(\boldsymbol{z}_m, \cdot')$. Then a sparse Gaussian process (SGP) [Snelson and Ghahramani, 2005, Titsias, 2009, Hensman et al., 2013] can be obtained by conditioning $f(\cdot)$ on the inducing variables $\mathbf{u}$ evaluated at learnable inducing points $\mathbf{Z}$,

$$\begin{aligned} f(\cdot)|\mathbf{u}, \mathbf{Z} &\sim \mathcal{GP}\big(\tilde{m}(\cdot), \tilde{k}(\cdot, \cdot')\big), \\ \tilde{m}(\cdot) &= m(\cdot) - k(\cdot, \mathbf{Z})\mathbf{K_{uu}}^{-1}(\mathbf{m_u} - \mathbf{u}), \\ \tilde{k}(\cdot, \cdot') &= k(\cdot, \cdot') - k(\cdot, \mathbf{Z})\mathbf{K_{uu}}^{-1}k(\mathbf{Z}, \cdot'). \end{aligned} \tag{4}$$

Intuitively, by restricting the original GP (prior) to only those functions consistent with the pseudo observations $\{\mathbf{u}, \mathbf{Z}\}$, the resulting SGP (posterior) encapsulates more information than the prior.

Deep Gaussian Process. A deep Gaussian process (DGP) [Damianou and Lawrence, 2013] is a hierarchical composition of SGPs in which the outputs of a layer become the inputs of the subsequent layer. An L-layered DGP represents a distribution over a real-valued function $g(\cdot) : \mathbb{R}^{D_0} \to \mathbb{R}^{D_L}$,

$$g(\cdot) = \big[(f^L|\mathbf{u}^L, \mathbf{Z}^{L-1}) \circ \cdots \circ (f^1|\mathbf{u}^1, \mathbf{Z}^0)\big](\cdot), \tag{5}$$

where $f^l(\cdot)|\mathbf{u}^l, \mathbf{Z}^{l-1}$ is the l-th multi-output SGP, a collection of D_l independent SGPs sharing the same inputs from $\mathbb{R}^{D_{l-1}}$. The inherent stochasticity of GPs renders exact inference for DGPs analytically intractable. This difficulty can be viewed either as the challenge of propagating a random variable through the non-linear SGP layers, or equivalently, as the impossibility of analytically integrating out the intermediate latent variables to obtain the marginal likelihood. Consequently, the intractability necessitates approximate inference techniques, such as variational inference (VI) and Monte Carlo (MC) methods [Salimbeni and Deisenroth, 2017, Havasi et al., 2018].

Wasserstein Metric. The Wasserstein metric, which was introduced by L. V. Kantorovich [Kantorovich, 1960], describes the minimal cost to transform one probability measure to another. Formally, the p-Wasserstein distance between two probability measures α and β on a metric space (M, d) is

$$W_p(\alpha, \beta) = \left(\inf_{\gamma \in \Gamma(\alpha,\beta)} \int_{M \times M} d^p(x, y)\, \mathrm{d}\gamma(x, y)\right)^{\frac{1}{p}}, \tag{6}$$

where $\Gamma(\alpha, \beta)$ is the set of joint measures on $M \times M$ with marginals α and β. When equipped with the Euclidean metric $d_E(\boldsymbol{x}_1, \boldsymbol{x}_2) = \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2$, the squared 2-Wasserstein (W_2^2) distance between the Gaussian measures induced by random variables $\mathbf{x}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $\mathbf{x}_2 \sim \mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$ admits an analytical expression [Dowson and Landau, 1982, Olkin and Pukelsheim, 1982],

$$\begin{aligned} W^2_{2,d_E}(\mathbf{x}_1, \mathbf{x}_2) &= \|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_2^2 \\ &\quad + \mathrm{Tr}\big(\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2 - 2(\boldsymbol{\Sigma}_1^{\frac{1}{2}}\boldsymbol{\Sigma}_2\boldsymbol{\Sigma}_1^{\frac{1}{2}})^{\frac{1}{2}}\big). \end{aligned} \tag{7}$$

Note that the Wasserstein metric given by Equation 6 can alternatively be defined over random variables as in Equation 7, and the two notations will be used interchangeably hereinafter.

3 GAUSSIAN PROCESS NEURONS

In this section, we present the formulation of the *Gaussian process neuron*, an SGP-based probabilistic computational unit that allows for exact uncertainty propagation.

Firstly, we define a novel positive definite Wasserstein exponential kernel on the space of factorized Gaussian measures (joint Gaussian measures that factorize as a product of their

marginals). Subsequently, an SGP neuron is constructed by incorporating the proposed kernel. It takes as inputs the mean and variance specifying a Gaussian measure and computes the principled output mean and variance, thereby defining a composable map between factorized Gaussians. Finally, we regularize the neuron via maximum a posteriori (MAP) estimation to prevent overfitting while preserving its principled uncertainty propagation, and demonstrate that its computational complexity matches a standard SGP up to a marginal overhead.

3.1 WASSERSTEIN EXPONENTIAL KERNEL

For factorized Gaussians $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \mathrm{diag}(\boldsymbol{\sigma}^2))$, the W^2_{2,d_E} distance given by Equation 7 simplifies to

$$W^2_{2,d_E}(\mathbf{x}_1, \mathbf{x}_2) = \|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_2^2 + \|\boldsymbol{\sigma}_1 - \boldsymbol{\sigma}_2\|_2^2. \quad (8)$$

A similar simplification exists for the W_2^2 distance based on the weighted Euclidean metric $d_W(\boldsymbol{x}_1, \boldsymbol{x}_2) = \|\boldsymbol{c} \odot (\boldsymbol{x}_1 - \boldsymbol{x}_2)\|_2$ with positive weights $\boldsymbol{c}$, as it is equivalent to the W^2_{2,d_E} distance between linearly transformed Gaussians,

$$W^2_{2,d_W}(\mathbf{x}_1, \mathbf{x}_2) = W^2_{2,d_E}(\boldsymbol{c} \odot \mathbf{x}_1, \boldsymbol{c} \odot \mathbf{x}_2). \quad (9)$$

The W^2_{2,d_W} distance under the factorization assumption is identical to the squared Euclidean distance between the concatenations of the weighted mean and standard deviation vectors. Consequently, as an application of Schoenberg's theorem [Schoenberg, 1938], any valid radial kernel defined on a Euclidean space naturally induces a valid, positive definite kernel on the space G^s of s-variate factorized Gaussians by simply substituting the Euclidean distance with the W_{2,d_W} distance.

Wasserstein Exponential Kernel. Applying this substitution to the canonical squared exponential (SE) kernel $k_{se}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp(-\|\boldsymbol{c} \odot (\boldsymbol{x}_1 - \boldsymbol{x}_2)\|_2^2)$, the *Wasserstein exponential kernel* (WE kernel) $k_{we} : G^s \times G^s \to \mathbb{R}$ with positive weights $\boldsymbol{c}$ can be defined as

$$k_{we}(\mathbf{x}_1, \mathbf{x}_2) = \exp\big(-W^2_{2,d_E}(\boldsymbol{c} \odot \mathbf{x}_1, \boldsymbol{c} \odot \mathbf{x}_2)\big). \quad (10)$$

A WE kernel can be scaled by a positive parameter τ^2 to model covariances greater than 1, which is omitted here for simplicity.

Since the WE kernel is an extension of the SE kernel to the space of factorized Gaussians, it inherits the parameter simplicity, infinite smoothness, and separability across dimensions [Schölkopf and Smola, 2002]. Moreover, the computation on the means and standard deviations is also separable, that is, a WE kernel factorizes as the product of one SE kernel on the mean space and another on the standard deviation space,

$$k_{we}(\mathbf{x}_1, \mathbf{x}_2) = k_{se}(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2) k_{se}(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2), \quad (11)$$

both of which have the same weights $\boldsymbol{c}$ as the WE kernel.

3.2 SPARSE GAUSSIAN PROCESS NEURONS

WEK-SGP. A *Wasserstein exponential kernel sparse Gaussian process* (WEK-SGP) is a conditional GP over the space of real-valued functions $f : G^s \to \mathbb{R}$. It is formulated by conditioning a prior GP with the WE kernel on a set of inducing points $\mathbf{Z} = \{\boldsymbol{z}_m\}_{m=1}^M \subset \mathbb{R}^s$ and inducing variables $\mathbf{u} \in \mathbb{R}^M$, as defined in Equation 4.

Since modeling output correlations is not of primary concern, we can interpret the WEK-SGP as a parametric function over individual inputs. Specifically, for an input $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_\mathbf{x}, \mathrm{diag}(\boldsymbol{\sigma}^2_\mathbf{x}))$, the output is a univariate Gaussian

$$\begin{aligned} &f(\mathbf{x})|\mathbf{u}, \mathbf{Z} \sim \mathcal{N}(\mu_f, \sigma_f^2), \\ &\mu_f = m(\mathbf{x}) - k_{we}(\mathbf{x}, \mathbf{Z})\mathbf{K}_{\mathbf{uu}}^{-1}(\mathbf{m}_\mathbf{u} - \mathbf{u}), \\ &\sigma_f^2 = k_{we}(\mathbf{x}, \mathbf{x}) - k_{we}(\mathbf{x}, \mathbf{Z})\mathbf{K}_{\mathbf{uu}}^{-1} k_{we}(\mathbf{Z}, \mathbf{x}), \end{aligned} \quad (12)$$

where $\mathbf{K}_{\mathbf{uu}}[m, m'] = k_{se}(\boldsymbol{z}_m, \boldsymbol{z}_{m'})$, as the inducing points are deterministic. By effectively defining a map from G^s to G^1, the WEK-SGP can serve as the computational unit (neuron) of an intricate map represented by a neural network that is obtained by the vectorization and composition of neurons. Importantly, as the map defined by a WEK-SGP is analytically tractable, it enables the exact propagation of uncertainty through neurons, which is represented by the variances of Gaussians.

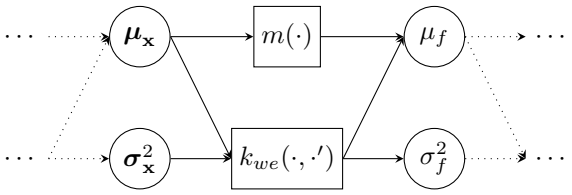

Figure 1: The structure of a WEK-SGP neuron.

Since the output variance σ_f^2 in Equation 12 entirely depends on the WE kernel, it is reasonable to define the mean function $m(\cdot)$ on the mean space as $m(\mathbf{x}) = m(\boldsymbol{\mu}_\mathbf{x})$ for simplicity. The uncertainty is thereby propagated exclusively by the WE kernel, making it easier to quantify. The structure of the resulting WEK-SGP neuron is shown in Figure 1.

3.3 PRINCIPLED UNCERTAINTY PROPAGATION

Following Equation 11, the cross-covariance between the input $\mathbf{x}$ and inducing points $\mathbf{Z}$ (degenerate Gaussians with zero variances) factorizes as $k_{we}(\mathbf{x}, \mathbf{Z}) = \lambda_\mathbf{x} k_{se}(\boldsymbol{\mu}_\mathbf{x}, \mathbf{Z})$, where $\lambda_\mathbf{x} = k_{se}(\boldsymbol{\sigma}_\mathbf{x}, \mathbf{0}) = \exp(-\|\boldsymbol{c} \odot \boldsymbol{\sigma}_\mathbf{x}\|_2^2)$. Substituting this representation into Equation 12 yields

$$\begin{aligned} &\mu_f = m(\boldsymbol{\mu}_\mathbf{x}) - \lambda_\mathbf{x} k_{se}(\boldsymbol{\mu}_\mathbf{x}, \mathbf{Z})\mathbf{K}_{\mathbf{uu}}^{-1}(\mathbf{m}_\mathbf{u} - \mathbf{u}), \\ &\sigma_f^2 = 1 - \lambda_\mathbf{x}^2 k_{se}(\boldsymbol{\mu}_\mathbf{x}, \mathbf{Z})\mathbf{K}_{\mathbf{uu}}^{-1} k_{se}(\mathbf{Z}, \boldsymbol{\mu}_\mathbf{x}), \end{aligned} \quad (13)$$

where we exploit the fact that $k_{we}(\mathbf{x}, \mathbf{x}) \equiv 1$ for any $\mathbf{x}$.

An SGP given by Equation 4 can be interpreted as a posterior resulting from updating the GP prior to correspond with the pseudo observations $\{\mathbf{Z}, \mathbf{u}\}$, where the updates are represented by the additive terms. For a WEK-SGP rewritten in Equation 13, the input variance $\boldsymbol{\sigma}_{\mathbf{x}}^2$ only appears in the term $\lambda_{\mathbf{x}} \in (0, 1]$, which acts as a coefficient for the update terms. To elucidate the effect of input uncertainty on the outputs, we express the output mean and variance of a neuron as convex combinations of their prior values and posterior values, using $\lambda_{\mathbf{x}}$ and $\lambda_{\mathbf{x}}^2$ as the respective weighting coefficients,

$$\begin{aligned} \mu_f &= (1 - \lambda_{\mathbf{x}}) m(\boldsymbol{\mu}_{\mathbf{x}}) + \lambda_{\mathbf{x}} \tilde{m}(\boldsymbol{\mu}_{\mathbf{x}}), \\ \sigma_f^2 &= (1 - \lambda_{\mathbf{x}}^2) + \lambda_{\mathbf{x}}^2 \tilde{k}(\boldsymbol{\mu}_{\mathbf{x}}), \end{aligned} \tag{14}$$

where

$$\begin{aligned} \tilde{m}(\boldsymbol{\mu}_{\mathbf{x}}) &= m(\boldsymbol{\mu}_{\mathbf{x}}) - k_{se}(\boldsymbol{\mu}_{\mathbf{x}}, \mathbf{Z}) \mathbf{K}_{\mathbf{uu}}^{-1} (\mathbf{m}_{\mathbf{u}} - \mathbf{u}), \\ \tilde{k}(\boldsymbol{\mu}_{\mathbf{x}}) &= 1 - k_{se}(\boldsymbol{\mu}_{\mathbf{x}}, \mathbf{Z}) \mathbf{K}_{\mathbf{uu}}^{-1} k_{se}(\mathbf{Z}, \boldsymbol{\mu}_{\mathbf{x}}). \end{aligned} \tag{15}$$

This formulation explicitly demonstrates how the model interpolates between the prior and the posterior based on input uncertainty. The interpolation coefficient $\lambda_{\mathbf{x}}$ is strictly decreasing with respect to each dimension of variance $\sigma_{x,i}^2$. Consequently, as the input variance increases, the coefficient $\lambda_{\mathbf{x}}$ decreases from 1 toward 0, causing the output mean μ_f and variance σ_f^2 to monotonically revert from the posteriors $\tilde{m}(\boldsymbol{\mu}_{\mathbf{x}})$ and $\tilde{k}(\boldsymbol{\mu}_{\mathbf{x}})$ to the priors $m(\boldsymbol{\mu}_{\mathbf{x}})$ and 1. Intuitively, as the input becomes more uncertain, the output loses confidence in the specific posterior and relies more on the uninformed prior.

As a sanity check, for deterministic inputs with zero variances $\boldsymbol{\sigma}_{\mathbf{x}}^2 = \mathbf{0}$, the coefficient becomes $\lambda_{\mathbf{x}} = 1$ and the WEK-SGP reduces to a standard SGP with the SE kernel. Conversely, for inputs where some $\sigma_{x,i}^2 \to \infty$, the coefficient $\lambda_{\mathbf{x}} \to 0$ and the output approaches the prior, indicating that no information can be extracted from extremely noisy inputs.

Notably, the posterior variance $\tilde{k}(\boldsymbol{\mu}_{\mathbf{x}})$ is strictly bounded above by the prior variance 1 by the properties of conditional Gaussians. Therefore, an increment in the input variances leads to an increment in the output variance, aligning with the intuition that a more uncertain input should produce a more uncertain output.

3.4 POSTERIOR REGULARIZATION VIA MAP

In standard DGPs, the intractability of intermediate latent variables typically requires approximate inference. Variational inference (VI), for instance, introduces a stochastic posterior distribution $q(\mathbf{u})$ over the inducing variables [Salimbeni and Deisenroth, 2017, Yu et al., 2019, Xu et al., 2024]. Instead, we maintain the inducing variables $\mathbf{u}$ as deterministic point estimates optimized via maximum a posteriori (MAP) estimation, a strategy firstly proposed for parametric GPs [Jankowiak et al., 2020a]. For our WEK-SGP architecture, this deterministic approach is necessary for two reasons.

Firstly, because the WEK-SGP neuron propagates Gaussian distributions, the final output of our model remains a tractable Gaussian. This enables the direct evaluation and optimization of the exact marginal likelihood, bypassing approximate lower bounds entirely. Secondly, introducing a distribution over $\mathbf{u}$ would inject detrimental stochasticity into the neuron. As derived in Equation 13, the posterior variance $\tilde{k}(\boldsymbol{\mu}_{\mathbf{x}})$ is bounded strictly above by 1 only when conditioned on deterministic inducing variables. Marginalizing over a stochastic $q(\mathbf{u})$ would inflate this predictive variance, potentially violating the bound $\sigma_f^2 < 1$ and destroying the monotonicity of the variance interpolation.

Therefore, maintaining $\mathbf{u}$ as a deterministic point estimate is essential. To prevent overfitting, we regularize this estimate using the GP prior $p(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{m}_{\mathbf{u}}, \mathbf{K}_{\mathbf{uu}})$. Let $\ln p(\boldsymbol{y}|\mathbf{X})$ denote the log-marginal likelihood (LML) of the observations $\{\mathbf{X}, \boldsymbol{y}\}$ (detailed in Section 4). The MAP objective can be formulated as

$$\mathcal{L} = \ln p(\boldsymbol{y}|\mathbf{X}) + \ln p(\mathbf{u}). \tag{16}$$

In this objective, the log-prior $\ln p(\mathbf{u})$ serves as a structural regularizer. Crucially, as proved in Appendix A.1, the intensity of this regularization on $\mathbf{u}$ is equivalent to the penalty imposed on the mean of the variational posterior $q(\mathbf{u})$ in mean-field VI [Hensman et al., 2013, Salimbeni and Deisenroth, 2017]. Thus, our MAP formulation preserves the essential regularization properties of VI without injecting additional stochasticity that would compromise our principled uncertainty propagation.

3.5 COMPUTATIONAL COMPLEXITY

The computational complexity of a WEK-SGP matches that of a standard SGP up to a marginal overhead for the computation on the input-variance, avoiding the sample-dependent scaling inherent to Monte Carlo (MC) approximations in standard DGPs. For a batch of B inputs in $\mathbb{R}^D$ and M inducing points, the optimization cost is dominated by forming and inverting the covariance $\mathbf{K}_{\mathbf{uu}}$ ($\mathcal{O}(M^2(D + M))$), and computing the input-dependent terms $k_{se}(\mathbf{Z}, \boldsymbol{\mu}_{\mathbf{x}})$ and $\mathbf{K}_{\mathbf{uu}}^{-1} k_{se}(\mathbf{Z}, \boldsymbol{\mu}_{\mathbf{x}})$ ($\mathcal{O}(BM(D + M))$). Conventional DGPs and extensions thereof approximate intractable expectations using S MC samples per input, which scales the input-dependent costs to $\mathcal{O}(SBM(D+M))$. Moreover, as model depth increases, achieving low-variance gradient estimates requires a correspondingly large S, aggravating the computational burden. By parametrically propagating distributions, our WEK-SGP neuron circumvents the sampling method, enabling deterministic optimization and evaluation that remains efficient regardless of model depth. We empirically validate this computational advantage over doubly stochas-

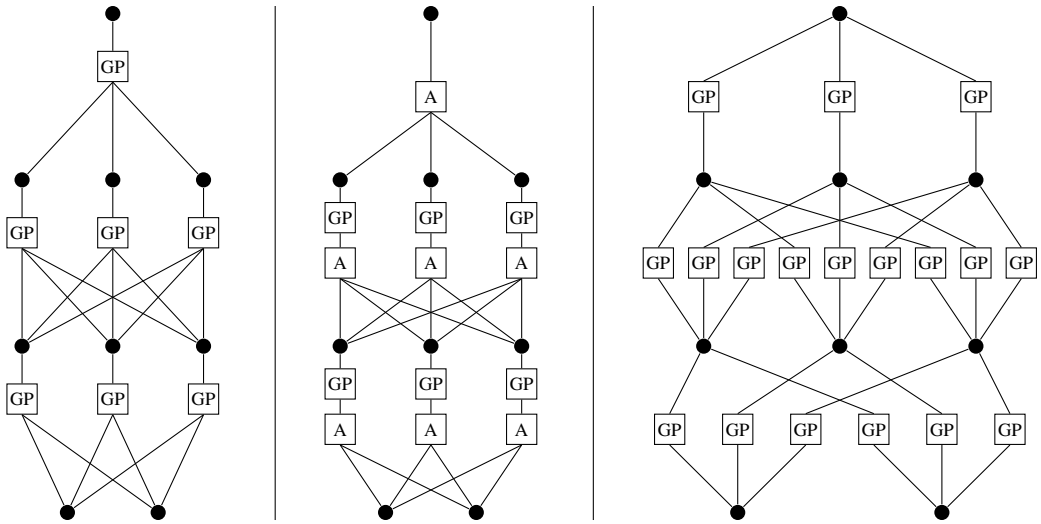

Figure 2: The structures of WEK-DGP (left), WEK-GPLAN (middle), and WEK-GPKAN (right). Black nodes represent the variables propagated from bottom to top (deterministic inputs, intermediate Gaussian moments, and final predictive distributions). Transformations applied along the edges are denoted by square blocks, where "GP" signifies a WEK-SGP neuron and "A" represents an affine transformation.

tic (DS)-DGP [Salimbeni and Deisenroth, 2017], profiling the runtime across varying network depths, widths, numbers of inducing points, and MC sample sizes in Appendix C.

4 PROBABILISTIC NETWORKS WITH GAUSSIAN PROCESS NEURONS

In this section, we detail the construction and inference formulation of probabilistic networks built from WEK-SGP neurons. Firstly, we outline the integration of WEK-SGP neurons into various network topologies. Subsequently, we formulate the predictive distribution as a finite Gaussian mixture to capture non-Gaussian marginals. Finally, we define the training objective via MAP estimation, yielding an exact and variance-free optimization framework.

4.1 NETWORK ARCHITECTURES

To formalize the networks architectures, we characterize the dense representation of a layer by how it computes a scalar output node from a D-dimensional input ($G^D \to G^1$). Deep networks are subsequently constructed by stacking layers, where each layer concatenates multiple independent node computations. We highlight three distinct paradigms, as illustrated in Figure 2.

WEK-DGP. The foundational architecture follows the standard DGP [Damianou and Lawrence, 2013]. We define the *WEK-DGP*, where the i-th output node is computed by a single multivariate WEK-SGP neuron $f_i : G^D \to G^1$ that processes the entire D-dimensional input,

$$y_i = f_i(\mathbf{x}). \tag{17}$$

While highly expressive, this paradigm requires maintaining inducing points in high-dimensional spaces, which can be computationally demanding.

To alleviate the dimensionality challenge, we can decouple the dimension mixing from the probabilistic non-linear mapping. Crucially, the family of Gaussians is closed under affine transformations, that is, if $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, then $\mathbf{A}\mathbf{x} + \mathbf{b} \sim \mathcal{N}(\mathbf{A}\boldsymbol{\mu} + \mathbf{b}, \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^\mathsf{T})$ for any matrix $\mathbf{A}$ and vector $\mathbf{b}$. This property guarantees that a univariate WEK-SGP can seamlessly substitute a standard point-wise activation function following any linear operation.

WEK-GPLAN. Following the MLP structure, we define the *WEK-GP learnable activation network* (WEK-GPLAN). The i-th output node projects the input to a scalar via an affine transformation, followed by a univariate WEK-SGP neuron $f_i : G^1 \to G^1$,

$$y_i = f_i\big(\langle \boldsymbol{a}_i, \mathbf{x} \rangle + b_i\big). \tag{18}$$

The baseline mean function of f_i is typically set to standard non-linear activations, such as ReLU and Tanh.

WEK-GPKAN. Alternatively, the Kolmogorov-Arnold Network (KAN) [Liu et al., 2025] decouples the mapping by placing learnable non-linearities on the edges. Because the sum of independent univariate Gaussians remains Gaussian, this paradigm also fits the compositional GP model. We define the *WEK-GPKAN*, where the i-th output node applies

an independent univariate WEK-SGP neuron $f_{i,j} : G^1 \to G^1$ to each input dimension and sums the results,

$$y_i = \sum_{j=1}^{D} f_{i,j}(x_j). \tag{19}$$

While these three paradigms are formulated as dense networks without loss of generality, the underlying WEK-SGP neurons can be adapted immediately to other topological structures. Provided the architectural dimension-mixing operations remain linear, these dense formulations naturally extend to spatial and graph-based DGPs, such as Convolutional DGPs [Blomqvist et al., 2020, Dutordoir et al., 2020] and Graph Convolutional DGPs [Opolka and Lió, 2022]. Furthermore, by substituting standard activation functions with univariate WEK-SGP neurons, we can construct exact probabilistic variants of classical deterministic topologies, including CNNs [Lecun et al., 1998], GCNs [Kipf and Welling, 2017], vanilla RNNs [Elman, 1990], and recent KAN extensions [Bodner et al., 2025, Kiamari et al., 2024].

Beyond supervised learning, our WEK-SGP neuron provides a rigorous foundation for broader probabilistic frameworks. By circumventing sampling bottlenecks while maintaining strict tractability, our modular unit is highly suitable for integration into probabilistic circuits (PCs) [Choi et al., 2020] and generative GP architectures—specifically GP-LVMs [Lawrence, 2005] and GP-VAEs [Casale et al., 2018]—which we defer to future work.

4.2 PREDICTIVE DISTRIBUTION

The predictive marginal of a standard DGP is intrinsically non-Gaussian and multi-modal [Lu et al., 2020]. While conventional VI approximates it via MC sampling, standard mean-field VI [Salimbeni and Deisenroth, 2017] fundamentally fails to model multi-modal marginals [Salimbeni et al., 2019]. We formally demonstrate in Appendix A.2 that this limitation stems from the expected log-likelihood (ELL) in the VI objective, which inherently suppresses multi-modality.

Various strategies address this limitation, including introducing latent variables [Salimbeni et al., 2019], employing MCMC-based inference [Havasi et al., 2018], or explicitly optimizing a finite Gaussian mixture [Jankowiak et al., 2020b]. Building upon the explicit mixture strategy, we extend our architecture to output a Gaussian mixture, thereby circumventing the pitfalls of VI. The final layer employs parallel heads to generate K components with means $\mu_k(\boldsymbol{x})$ and variances $\sigma_k^2(\boldsymbol{x})$, weighted by parameters π_k constrained such that $\sum_{k=1}^{K} \pi_k = 1$. Incorporating a Gaussian likelihood with observation noise variance σ_{obs}^2, the predictive distribution is

$$p(y|\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}\big(y|\mu_k(\boldsymbol{x}), \sigma_k^2(\boldsymbol{x}) + \sigma_{obs}^2\big). \tag{20}$$

While optimizing the LML resolves the multi-modality limitations of standard VI, unconstrained mixture models are susceptible to capacity misallocation [Makansi et al., 2019, Jankowiak et al., 2020b]. When the mapping is single-valued but highly variable, the objective can improperly favor distributional complexity over functional complexity. For instance, rather than learning a complex, high-frequency mapping, the model may converge to a simplified mean function compensated by a static, multi-modal predictive distribution (exemplified in Section 6.1). To mitigate this risk, we restrict the predictive distribution to a single component ($K = 1$) by default for standard regression tasks.

4.3 OPTIMIZATION OBJECTIVE

The proposed networks are trained via MAP estimation, as formulated in Section 3.4. Given the observations $\{\boldsymbol{x}_n, y_n\}_{n=1}^{N}$, we maximize the sum of the log-marginal likelihood of the predictive mixture and the log-priors over the model parameters. The objective function is defined as

$$\mathcal{L} = \sum_{n=1}^{N} \ln p(y_n|\boldsymbol{x}_n) + \ln p(\mathbf{U}) + \ln p(\boldsymbol{\Theta}), \tag{21}$$

where $p(y_n|\boldsymbol{x}_n)$ is the marginal given by Equation 20, and $\mathbf{U}$ denotes all inducing variables in the model. The term $\boldsymbol{\Theta}$ encapsulates all remaining learnable parameters, including the kernel hyperparameters and the affine weights.

5 RELATED WORK

The composition of latent functions in DGPs [Damianou and Lawrence, 2013] renders exact marginalization intractable. The seminal work of DS-VI [Salimbeni and Deisenroth, 2017] effectively approximates the DGP posterior by combining mean-field VI with MC sampling. Subsequent work introduces techniques such as importance weighting [Salimbeni et al., 2019], adversarially generated implicit posterior [Yu et al., 2019] and denoising diffusion SDEs [Xu et al., 2024] within the VI framework to enhance posterior modeling. Alternative non-VI inference strategies include expectation propagation [Bui et al., 2016] and Markov chain Monte Carlo methods [Havasi et al., 2018].

Distinct from the work dedicated to posterior approximation for the probabilistic DGP, some research explores the parametric formulations to enable deterministic optimization. Deep sigma point process [Jankowiak et al., 2020b] explicitly optimizes a finite Gaussian mixture derived by a learnable quadrature across the layers. Interpretable DGPs [Lu et al., 2020] approximates DGPs as shallow GPs by calculating the exact moments in the low variance regime. Similarly, a recent work [Chen, 2024] considers the first moments of univariate SGP outputs to achieve analytical uncertainty propagation in the Kolmogorov-Arnold Network architecture [Liu et al., 2025].

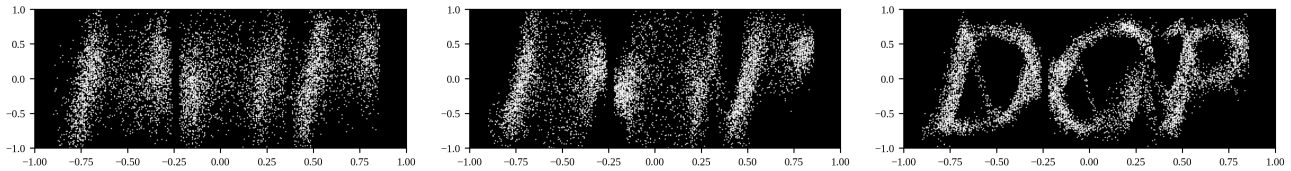

Figure 3: Predictive distributions on a 1D '*DGP*' synthetic dataset. Random samples shown in white are drawn from the marginals of: DS-DGP (left), which exhibits mode collapse; WEK-DGP with $K = 1$ (middle), which averages across disjoint clusters; and WEK-DGP with $K = 4$ (right), which captures the multi-modal structure.

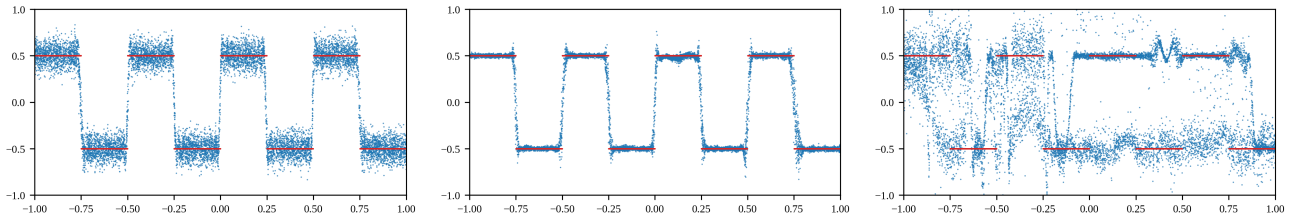

Figure 4: Predictive distributions on a 1D periodic step function denoted by the red lines. Random samples shown in blue are drawn from the marginals of: DS-DGP (left), which overestimates variance; WEK-DGP with $K = 1$ (middle), which tightly fits the mapping; and WEK-DGP with $K = 4$ (right), which misallocates capacity via spurious bi-modality or placing density between plateaus.

While the aforementioned methods primarily focus on feedforward architectures, further research adapts DGPs to diverse structural topologies. Deep convolutional GP [Blomqvist et al., 2020, Dutordoir et al., 2020] integrates convolutional operations within the DGP framework to process spatial data, while deep graph convolutional GP [Opolka and Lió, 2022] extends these formulations to graph-structured domains. Exploring other architectural compositions, thin and deep GP [de Souza et al., 2023] utilizes GPs to successively parameterize linear transformations applied to the inputs.

In the broader landscape of deep probabilistic modeling, Bayesian neural networks [Jospin et al., 2022] place prior distributions over network weights to capture epistemic uncertainty. Conversely, deep kernel learning [Wilson et al., 2016b,a] applies a GP over a deterministic network feature extractor to construct a highly expressive model, despite its known susceptibility to overfitting [Ober et al., 2021].

6 EMPIRICAL EVALUATION

In this section, we empirically evaluate the proposed probabilistic networks. We first analyze its inference dynamics on synthetic tasks, illustrating both its capability to model non-Gaussian marginals and the risks of capacity misallocation. We then assess its predictive performance on standard regression benchmarks, highlighting the stability and efficiency of the exact optimization objective. Finally, we validate its topological versatility across proposed network architectures. Experimental settings are detailed in Appendix B.

6.1 SYNTHETIC TASKS

To evaluate predictive distributions on non-Gaussian data, we adopt a 1D synthetic dataset that maps the horizontal coordinates of the letters '*DGP*' to their vertical coordinates. As shown in Figure 3, the mean-field approximation of DS-DGP exhibits mode collapse, while the unimodal WEK-DGP averages across the disjoint data clusters. In contrast, WEK-DGP with $K = 4$ components successfully captures the discontinuous, multi-modal marginals, accurately recovering the structural branches of the glyphs.

To illustrate the potential for capacity misallocation, we evaluate the models on a 1D periodic step function. As shown in Figure 4, WEK-DGP ($K = 4$) exhibits two distinct behaviors: in some regions, it places predictive mass in the empty space between plateaus, while in others, it models spurious bi-modal distributions across both ± 0.5. While both DS-DGP and WEK-DGP ($K = 1$) recover the step structure, DS-DGP yields visibly larger predictive variance, a known issue of its variational objective [Jankowiak et al., 2020a,b]. This empirically justifies restricting WEK-SGP to $K = 1$ for standard regression.

6.2 BENCHMARK REGRESSION DATASETS

We evaluate the WEK-DGP on 12 UCI regression datasets [Dua and Graff, 2017], with the sample size ranging from $N \sim 10^4$ to $N \sim 10^6$ and input dimension $3 \leq D \leq 380$ (detailed in Appendix B.4). We compare against four strong GP baselines, including SGP with a recent variational bound

Table 1: Average ranking of the methods across 12 regression datasets. Results are aggregated across 5 random training/test/validation splits and reported as mean rank ± standard deviation.

	SVGP-new	SV-DKL	DS-DGP	DSPP	WEK-DGP
NLL	3.90±1.36	3.50±0.79	3.78±0.92	2.23±1.41	**1.58±0.72**
RMSE	3.52±1.48	2.72±1.33	3.18±1.05	2.99±1.49	**2.58±1.52**
CRPS	3.82±1.49	3.20±1.05	3.78±0.83	2.26±1.44	**1.94±1.02**

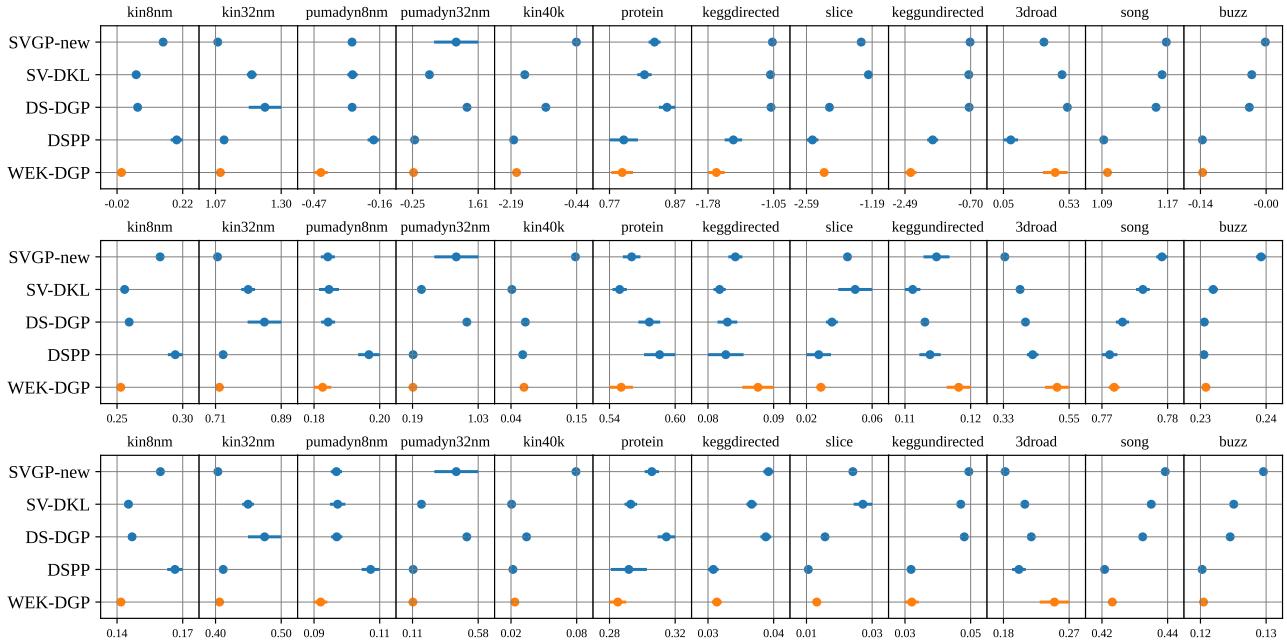

Figure 5: We depict NLL (top), RMSE (middle), and CRPS (bottom) for 12 regression datasets. Results are averaged over 5 random training/test/validation splits. Uncertainty bars denote standard errors.

(SVGP-new) [Titsias, 2025], stochastic variational DKL (SV-DKL) [Wilson et al., 2016a], DS-DGP [Salimbeni and Deisenroth, 2017], and deep sigma point process (DSPP) [Jankowiak et al., 2020b].

Table 1 summarizes the average model rankings across all 12 datasets for negative log-likelihood (NLL), root mean square error (RMSE), and continuous ranked probability score (CRPS) [Gneiting and Raftery, 2007]. These metrics capture distinct aspects of model performance: RMSE evaluates deterministic point prediction accuracy, NLL assesses uncertainty calibration, and CRPS measures overall uncertainty quantification. WEK-DGP consistently outperforms all baselines, achieving the best average rank across all three metrics. Notably, while SV-DKL exhibits competitive point-predictive accuracy by closely matching WEK-DGP in average RMSE rank, its intrinsic defects severely degrade uncertainty calibration [Ober et al., 2021]. Conversely, WEK-DGP maintains accurate point prediction while yielding superior uncertainty quantification. The dataset-specific performance is detailed in Figure 5 and Appendix B.4.

To further evaluate optimization dynamics and scalability, we analyze convergence on another large-scale dataset, houseelectric ($N \approx 2 \times 10^6$, $D = 11$). Figure 6 tracks the validation NLL across varying network depths ($L \in \{2, 3, 4\}$). Throughout the optimization process, the NLL trajectory of WEK-DGP remains below that of the sampling-based DS-DGP. This demonstrates that the exact LML objective provides a highly efficient learning signal across all iterations, ultimately stabilizing at a lower final optimum with better-calibrated predictive uncertainty.

6.3 TOPOLOGICAL VERSATILITY

To evaluate the high-dimensional scalability of WEK-GPLAN, we apply it to flattened image classification tasks on MNIST ($D = 784$) and CIFAR-10 ($D = 3072$). Using identical network structures of depth $L = 4$, we compare WEK-GPLAN with ReLU and SiLU mean functions against deterministic learnable activation networks (LANs) utilizing their respective parametric counterparts, PReLU [He et al., 2015] and Swish [Ramachandran et al., 2018]. The

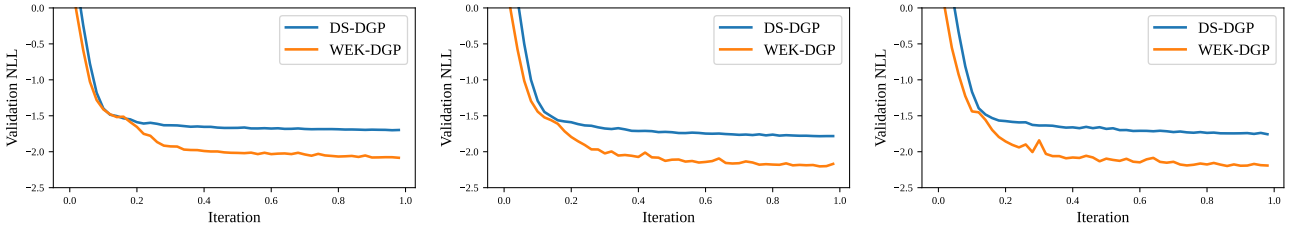

Figure 6: Validation NLL convergence on the houseelectric dataset for model depths $L = 2$ (left), $L = 3$ (middle), and $L = 4$ (right) in 5000 iterations. The optimization trajectory of WEK-DGP remains below that of DS-DGP across all iterations, stabilizing at a lower final NLL plateau.

Table 2: Classification error rates (%) on MNIST and CIFAR-10. Results are reported as mean ± standard deviation across 5 random training/validation splits.

	PReLU	GP-ReLU	Swish	GP-SiLU
MNIST	2.68±0.07	**1.79±0.09**	2.08±0.06	**1.67±0.04**
CIFAR-10	50.88±0.24	**46.08±0.48**	46.19±0.46	**45.17±0.37**

Table 3: Average ranking of RMSE on 10 symbolic function datasets across varying network depths. Results are aggregated across 5 random training/test/validation splits and reported as mean rank ± standard deviation.

	$L = 2$	$L = 3$	$L = 4$
KAN	5.22±1.48	4.03±1.05	3.30±1.22
WEK-GPKAN	**4.02±1.49**	**2.42±1.34**	**2.02±1.33**

LANs allocate independent activations per channel, whereas WEK-GPLAN applies a single univariate WEK-SGP neuron across all dimensions. Both strategies yield comparable total parameter counts.

As reported in Table 2, WEK-GPLAN consistently yields lower error rates than the LAN baselines. Notably, while the deterministic Swish network significantly outperforms PReLU, this gap narrows considerably in our probabilistic framework. This indicates that the expressiveness of the WEK-GPLAN derives primarily from local adjustments of WEK-SGP to the baseline function to create complex segmentations in the feature space rather than depending solely on the global geometry of the chosen activation.

Finally, to further verify the compositional capabilities of our method, we evaluate the WEK-GPKAN on 10 2-dimensional symbolic functions, mirroring the symbolic regression benchmarks of Liu et al. [2025]. We compare the standard deterministic KAN against our probabilistic WEK-GPKAN across varying network depths ($L \in \{2, 3, 4\}$).

As detailed in Table 3 and Appendix B.5, WEK-GPKAN consistently achieves strictly lower average RMSE ranks than the spline-based KAN at every equivalent depth. Structurally, the mean of an SGP operates as a weighted sum of kernel basis functions centered at the inducing points, making it analogous to the B-splines. Given this structural equivalence, we conjecture that the exceptional expressiveness of our WEK-SGP neuron arises from its probabilistic nature, which provides a more robust landscape for making local geometric adjustments to the function space.

7 CONCLUSIONS

We introduce the GP neuron, a probabilistic computational unit that utilizes a novel Wasserstein exponential kernel to bypass the computational bottlenecks of Monte Carlo approximations. By defining a composable map between factorized Gaussian measures, this framework introduces a mechanism for exact analytical uncertainty propagation. Unlike the intractable or approximate marginalization inherent to standard DGPs, this exact formulation completely eliminates sampling variance and drastically reduces multiplicative computational costs. We demonstrate that this computational efficiency makes probabilistic variants of various deep network architectures practically viable. Empirically, these networks achieve highly competitive uncertainty calibration compared to state-of-the-art GP baselines on regression tasks, scale efficiently to high-dimensional image classification, and demonstrate remarkable accuracy on symbolic functions. Ultimately, our approach provides a rigorous and adaptable foundation for deep probabilistic modeling.

References

Kenneth Blomqvist, Samuel Kaski, and Markus Heinonen. Deep convolutional gaussian processes. In *Machine Learning and Knowledge Discovery in Databases*, pages 582–597. Springer International Publishing, 2020.

Alexander Dylan Bodner, Antonio Santiago Tepsich, Jack Natan Spolski, and Santiago Pourteau. Convolutional kolmogorov-arnold networks, 2025.

Thang Bui, Daniel Hernandez-Lobato, Jose Hernandez-Lobato, Yingzhen Li, and Richard Turner. Deep gaussian processes for regression using approximate expectation propagation. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1472–1481, 2016.

Francesco Paolo Casale, Adrian Dalca, Luca Saglietti, Jennifer Listgarten, and Nicolo Fusi. Gaussian process prior variational autoencoders. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Andrew Siyuan Chen. Gaussian process kolmogorov-arnold networks, 2024.

Y Choi, Antonio Vergari, and Guy Van den Broeck. Probabilistic circuits: A unifying framework for tractable probabilistic models. *UCLA*, 6, 2020.

Andreas Damianou and Neil D. Lawrence. Deep gaussian processes. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, volume 31 of *Proceedings of Machine Learning Research*, pages 207–215, 2013.

Daniel Augusto de Souza, Alexander V Nikitin, S. T. John, Magnus Ross, Mauricio A Álvarez, Marc Peter Deisenroth, João Paulo Pordeus Gomes, Diego Mesquita, and César Lincoln Mattos. Thin and deep gaussian processes. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

D. C. Dowson and B. V. Landau. The fréchet distance between multivariate normal distributions. *Journal of Multivariate Analysis*, 12(3):450–455, 1982.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

Vincent Dutordoir, Mark van der Wilk, Artem Artemev, and James Hensman. Bayesian image classification with deep convolutional gaussian processes. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 1529–1539, 2020.

Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

Marton Havasi, José Miguel Hernández-Lobato, and Juan José Murillo-Fuentes. Inference in deep gaussian processes using stochastic gradient hamiltonian monte carlo. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015.

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, pages 282–290, 2013.

Martin Jankowiak, Geoff Pleiss, and Jacob Gardner. Parametric gaussian process regressors. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4702–4712, 2020a.

Martin Jankowiak, Geoff Pleiss, and Jacob Gardner. Deep sigma point processes. In *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI)*, volume 124 of *Proceedings of Machine Learning Research*, pages 789–798, 2020b.

Laurent Valentin Jospin, Hamid Laga, Farid Boussaid, Wray Buntine, and Mohammed Bennamoun. Hands-on bayesian neural networks—a tutorial for deep learning users. *IEEE Computational Intelligence Magazine*, 17(2): 29–48, 2022.

L. V. Kantorovich. Mathematical methods of organizing and planning production. *Management Science*, 6(4): 366–422, 1960.

Mehrdad Kiamari, Mohammad Kiamari, and Bhaskar Krishnamachari. Gkan: Graph kolmogorov-arnold networks, 2024.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Neil Lawrence. Probabilistic non-linear principal component analysis with gaussian process latent variable models. *Journal of Machine Learning Research*, 6(60):1783–1816, 2005.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljacic, Thomas Y. Hou, and Max Tegmark. Kan: Kolmogorov-arnold networks. In *The Thirteenth International Conference on Learning Representations*, 2025.

Chi-Ken Lu, Scott Cheng-Hsin Yang, Xiaoran Hao, and Patrick Shafto. Interpretable deep gaussian processes with moments. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 613–623, 2020.

Osama Makansi, Eddy Ilg, Ozgun Cicek, and Thomas Brox. Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Sebastian W. Ober, Carl E. Rasmussen, and Mark van der Wilk. The promises and pitfalls of deep kernel learning. In *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pages 1206–1216, 2021.

I. Olkin and F. Pukelsheim. The distance between two random vectors with given dispersion matrices. *Linear Algebra and its Applications*, 48:257–263, 1982.

Felix Opolka and Pietro Lió. Bayesian link prediction with deep graph convolutional gaussian processes. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 4835–4852, 2022.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions, 2018.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2005.

Hugh Salimbeni and Marc Deisenroth. Doubly stochastic variational inference for deep gaussian processes. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Hugh Salimbeni, Vincent Dutordoir, James Hensman, and Marc Deisenroth. Deep gaussian processes with importance-weighted variational inference. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5589–5598, 2019.

I. J. Schoenberg. Metric spaces and completely monotone functions. *Annals of Mathematics*, 39(4):811–841, 1938.

B. Schölkopf and A.J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press, 2002.

Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, volume 18, 2005.

Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pages 567–574, 2009.

Michalis Titsias. New bounds for sparse variational gaussian processes. In *Forty-second International Conference on Machine Learning*, 2025.

Andrew G Wilson, Zhiting Hu, Russ R Salakhutdinov, and Eric P Xing. Stochastic variational deep kernel learning. In *Advances in Neural Information Processing Systems*, volume 29, 2016a.

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P. Xing. Deep kernel learning. In Arthur Gretton and Christian C. Robert, editors, *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 370–378, 2016b.

Jian Xu, Delu Zeng, and John Paisley. Sparse inducing points in deep Gaussian processes: Enhancing modeling with denoising diffusion variational inference. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 55490–55500, 2024.

Haibin Yu, Yizhou Chen, Bryan Kian Hsiang Low, Patrick Jaillet, and Zhongxiang Dai. Implicit posterior variational inference for deep gaussian processes. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Exact Uncertainty Propagation via Gaussian Process Neurons (Supplementary Material)

Qiuxian Meng[1] **Yongyou Zhang**[1]

[1]Xiamen University

A FURTHER DETAILS ABOUT THE PROPOSED METHOD

A.1 EQUIVALENCE OF VI AND MAP REGULARIZATION

In mean-field VI for GPs [Hensman et al., 2013, Salimbeni and Deisenroth, 2017], the method introduces a variational posterior $q(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ on the inducing variables to approximate the prior $p(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{m_u}, \mathbf{K_{uu}})$. The optimization objective, known as the evidence lower bound (ELBO), regularizes this distribution via the Kullback-Leibler (KL) divergence

$$D_{\mathrm{KL}}\big(q(\mathbf{u})\|p(\mathbf{u})\big) = \frac{1}{2}\left\{\mathrm{Tr}(\mathbf{K_{uu}^{-1}}\boldsymbol{\Sigma}) + (\boldsymbol{\mu}-\mathbf{m_u})^{\mathsf{T}}\mathbf{K_{uu}^{-1}}(\boldsymbol{\mu}-\mathbf{m_u}) - M - \ln|\mathbf{K_{uu}}| + \ln|\boldsymbol{\Sigma}|\right\}.$$

During optimization, the regularization specifically applied to the variational mean $\boldsymbol{\mu}$ is governed entirely by the quadratic term, as all other terms are independent of $\boldsymbol{\mu}$. Thus, the negative gradient with respect to the mean is

$$-\nabla_{\boldsymbol{\mu}} D_{\mathrm{KL}}\big(q(\mathbf{u})\|p(\mathbf{u})\big) = -\mathbf{K_{uu}^{-1}}(\boldsymbol{\mu}-\mathbf{m_u}).$$

Our deterministic method directly optimizes the point estimate $\mathbf{u}$ via MAP estimation. The regularizer in Equation 16 is the log-prior

$$\ln p(\mathbf{u}) = -\frac{1}{2}\left\{(\mathbf{u}-\mathbf{m_u})^{\mathsf{T}}\mathbf{K_{uu}^{-1}}(\mathbf{u}-\mathbf{m_u}) - M\ln(2\pi) - \ln|\mathbf{K_{uu}}|\right\}.$$

Similarly, isolating the terms dependent on the inducing variables $\mathbf{u}$ leaves only the identical quadratic form. The gradient of this regularization term with respect to the deterministic estimate $\mathbf{u}$ is

$$\nabla_{\mathbf{u}} \ln p(\mathbf{u}) = -\mathbf{K_{uu}^{-1}}(\mathbf{u}-\mathbf{m_u}).$$

Comparing the two gradients reveals that they are identical. Therefore, the structural penalty—the intensity with which the estimate is pulled toward the prior mean $\mathbf{m_u}$ based on the covariance $\mathbf{K_{uu}}$—is perfectly preserved when transitioning from the variational mean $\boldsymbol{\mu}$ in VI to the deterministic point estimate $\mathbf{u}$ in MAP.

A.2 LML VERSUS ELL FOR MIXTURE DISTRIBUTIONS

We provide a comparison between the expected log-likelihood (ELL) objective utilized in standard VI and the log-marginal likelihood (LML) of a finite mixture, specifically examining how each formulation influences the modeling of multi-modal distributions.

Assume the predictive marginal is approximated by a K-component mixture, where the latent function f follows a variational distribution $q(f) = \sum_{k=1}^{K} \pi_k q_k(f)$, with each component $q_k(f) = \mathcal{N}(f|\mu_k, \sigma_k^2)$ and the mixture weights π_k summing to 1. For an observation y with the standard Gaussian likelihood $p(y|f) = \mathcal{N}(y|f, \sigma_{obs}^2)$, standard VI maximizes the ELL

$$\begin{aligned}\mathbb{E}_{q(f)}\big[\ln p(y|f)\big] &= \sum\nolimits_{k=1}^{K} \pi_k \mathbb{E}_{q_k(f)}\big[\ln p(y|f)\big] \\ &= -\frac{1}{2\sigma_{obs}^2}\sum\nolimits_{k=1}^{K}\Big(\pi_k\big((y-\mu_k)^2+\sigma_k^2\big)\Big) - \frac{1}{2}\ln(2\pi) - \frac{1}{2}\ln(\sigma_{obs}^2).\end{aligned}$$

This linear summation in the first data-fitting term demonstrates that the ELL imposes a simultaneous penalty across all mixture components. If a component j specializes on a distant mode relative to the current observation y, its expected squared error grows large, driving its ELL toward $-\infty$. Because the total objective is a weighted linear sum, a single mismatched component exerts a lethal penalty that drags the entire ELL toward ∞. To avoid this strong regularization, the optimization encourages all $q_k(f)$ to cover the global mean of the data, which inherently restricts the capacity to represent true multi-modality.

Conversely, explicitly optimizing the exact LML of the finite mixture yields

$$\begin{aligned}\ln \mathbb{E}_{q(f)}\left[p(y|f)\right] &= \ln\left(\sum\nolimits_{k=1}^{K} \pi_k \mathbb{E}_{q_k(f)}\left[p(y|f)\right]\right) \\ &= \ln\left(\sum\nolimits_{k=1}^{K} \frac{\pi_k}{\sqrt{2\pi(\sigma_k^2+\sigma_{obs}^2)}} \exp\left(-\frac{(y-\mu_k)^2}{2(\sigma_k^2+\sigma_{obs}^2)}\right)\right).\end{aligned}$$

Due to this log-sum-exp structure, the LML fundamentally alters how failed components are penalized. The squared error of any distant component is passed through an exponential function, strictly bounding its contribution from below at 0. Consequently, the LML requires only a single component to adequately explain y. if one component successfully captures the relevant mode, it provides a strictly positive mass to the inner summation. Crucially, the mismatched components merely asymptote to 0 and safely drop out of the sum, entirely avoiding the lethal $-\infty$ penalty. By placing the logarithm outside the expectation, the LML inherently facilitates component specialization, enabling the robust modeling of distinct modes without the zero-forcing penalties induced by the VI expectation.

B FURTHER EXPERIMENTAL DETAILS AND RESULTS

B.1 MODEL INITIALIZATION

We include affine mean functions in WEK-DGP and WEK-GPKAN, with the weights initialized by Kaiming normal distribution [He et al., 2015]. The same initialization applies to the linear weights in the affine layers of WEK-GPLAN. The kernel length-scales are initialized to $\boldsymbol{c} = \mathbf{1}$. The kernel output-scale is initialized to $\tau^2 = 1$ in WEK-GPLAN and WEK-GPKAN, whereas in WEK-DGP we initialize it to $\tau^2 = D_{out}^{-1}$ to avoid an extremely small weighting coefficient $\lambda_{\mathbf{x}} \to 0$ in the subsequent layer. The inducing points $\mathbf{Z}$ are initialized by k-means clustering on the dataset, utilizing the k-means++ initialization scheme. The inducing variables are initialized to the prior means as $\mathbf{u} = m(\mathbf{Z})$. The observation noise variance is initialized to $\sigma_{obs}^2 = 1$.

The baseline models follow the above initialization strategy. We use the squared exponential kernel with automatic relevance determination (ARD-SE kernel) for the models that contain GPs.

B.2 OPTIMIZATION DETAILS

For UCI regression, image classification and symbolic function datasets, results are averaged over five independent runs with random data splits. For the regression datasets, the data are partitioned into training, validation, and test sets with a ratio of $0.6/0.2/0.2$. For the image classification task, we randomly select 10000 samples from the standard training set for validation.

All models are optimized using the Adam optimizer with the learning rate decayed exponentially from 10^{-2} to 10^{-3}. We use a minibatch size $B = 1024$ in all experiments except for the image classification tasks, for which we set $B = 256$. For the medium size regression datasets ($N < 10^5$), we train for 5000 epochs. For the remaining large scale regression datasets, we train for 500 epochs. For the image classification datasets, we train for 100 epochs.

B.3 SYNTHETIC TASKS

For both synthetic datasets, we use 2-layered DS-DGPs and WEK-DGPs with shape $[1, 4, 1]$. We set $M = 16$ for DS-DGPs and single-component WEK-DGPs ($K = 1$), and $M = 8$ for multi-component WEK-DGPs ($K = 4$), ensuring comparable total parameter counts across models. We perform 1000 optimization iterations on these two datasets without monitoring, as these experiments are intended solely to evaluate each model's ability to learn the underlying data modes.

B.4 BENCHMARK REGRESSION DATASETS

Table 4 summarizes the total number of observations (N) and input feature dimensions (D) for the 12 standard benchmark datasets utilized in our regression experiments.

Table 4: Summary of the benchmark regression datasets, detailing the number of samples (N) and input dimensions (D).

	kin8nm	kin32nm	pumadyn8nm	pumadyn32nm	kin40k	protein
N	8192	8192	8192	8192	40000	45730
D	8	32	8	32	8	9
	keggdirected	slice	keggundirected	3droad	song	buzz
N	48827	53500	63608	434874	515345	583250
D	20	380	27	3	90	77

To ensure a fair evaluation across the regression benchmarks, we configure all methods to have comparable total parameter counts. For the deep architectures (DS-DGP, DSPP and WEK-DGP), we employ 2-layered models with a hidden layer width of $D_1 = \min(D, 16)$. Each of these deep models utilizes $M = 64$ inducing points. To match this capacity in the shallow SVGP-new, we allocate a proportionally larger budget of $M = 1024$ inducing points. For SV-DKL, the deterministic feature extractor is structured as a single-hidden-layer perceptron with a width of $\min(D, 16) \times 32$, followed by an output SGP utilizing $M = 256$ inducing points. Crucially, under these controlled capacity constraints, WEK-DGP strictly maintains the fewest learnable parameters among all evaluated methods. We configure DS-DGP with $S = 16$ Monte Carlo samples. To avoid the capacity misallocation phenomena detailed in Section 4.2, we restrict DSPP to a single quadrature point ($S = 1$), and our proposed models to a single Gaussian output component ($K = 1$) for all subsequent experiments.

Table 5 provides the detailed NLL, RMSE and CRPS results across the 12 regression datasets.

B.5 TOPOLOGICAL VERSATILITY

To evaluate the scalability and architectural adaptability of the proposed models, we detail the structural configurations used for the image classification and symbolic regression tasks.

For the image classification benchmarks, the feedforward architecture for MNIST is configured with layer widths of $[784, 256, 64, 16, 10]$. For the more complex CIFAR-10 dataset, we expand the network capacity to $[3072, 1024, 256, 64, 10]$. Across both tasks, the univariate GP neurons within the WEK-GPLAN models are consistently parameterized by $M = 64$ inducing points.

For the symbolic regression tasks mirroring Liu et al. [2025], the networks are constructed to process 2-dimensional inputs and predict a scalar output. For each target function, the dataset consists of 10000 uniformly sampled data points. Both standard KAN and WEK-GPKAN maintain a uniform width of 5 across all hidden layers. The spline-based edges in the standard KAN employ a grid size of $G = 128$, whereas the probabilistic edges in the WEK-GPKAN utilize $M = 64$ inducing points. Under this parameterization, WEK-GPKAN requires strictly fewer learnable parameters than standard KAN at every equivalent depth L.

Table 6 provides the detailed RMSE results across the 10 special functions, corroborating the aggregated rank improvements discussed in Section 6.3.

While WEK-GPKAN demonstrates superior expressiveness across the majority of the evaluated functions, Table 6 reveals a distinct failure mode on the highly oscillatory real spherical harmonic ($l = 5, m = 3$), where our model suffers an order-of-magnitude degradation and extreme variance compared to the standard KAN. This failure stems from the inherent optimization dynamics of sparse GPs in high-frequency regimes. Because the inducing point locations and kernel lengthscales are jointly optimized, dense oscillations create a severely non-convex objective landscape. If the initial inducing points fail to align with the function's local extrema, the gradient-based optimization frequently collapses into poor local optima—often learning an overly broad lengthscale that oversmooths the target, explaining the massive variance across random initializations. Consequently, for practitioners, this indicates that WEK-GPKAN may struggle with extreme high-frequency signals unless the architecture is augmented with specialized, frequency-aware initialization strategies.

C EMPIRICAL RUNTIME PROFILING

To empirically validate the computational efficiency discussed in the main text, we profile the training runtime of the proposed WEK-DGP against the sampling-based DS-DGP [Salimbeni and Deisenroth, 2017]. The evaluations measure the average computational time per training iteration required to process a fixed batch size $B = 256$. All profiling experiments are executed on a single NVIDIA RTX 3090 GPU to ensure hardware consistency.

As illustrated in Figure 7, the deterministic nature of WEK-DGP completely circumvents the sample-dependent scaling factor inherent to DS-DGP. Across all combinations of network widths (D) and numbers of inducing points (M), the DS-DGP runtimes scale multiplicatively with the number of MC samples (S). As the network depth increases along the x-axis of each subplot, the computational overhead of DS-DGP compounds significantly, represented by the steep slopes. Crucially, we emphasize that this computational penalty is severely exacerbated in wider models and configurations with more inducing points. This highlights a fundamental limitation of standard DGPs: attempting to scale to wider, more expressive architectures using MC sampling incurs a prohibitive computational cost. In contrast, WEK-DGP maintains a consistently flat, highly efficient trajectory across all depths, widths, and numbers of inducing points, confirming that parametrically propagating distributions avoids the severe computational bottlenecks associated with sequential MC sampling.

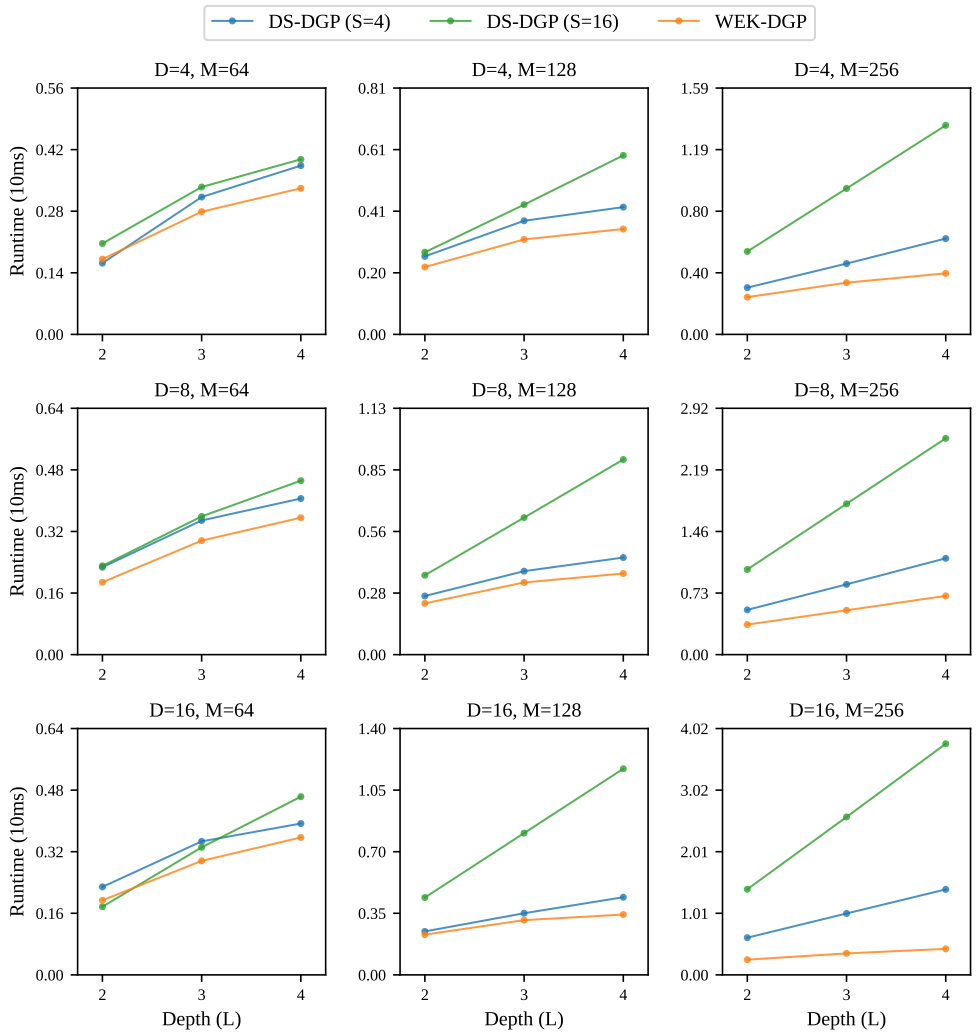

Figure 7: Training runtime per iteration across varying architectural configurations. The 3×3 grid displays varying network widths $D \in \{4, 8, 16\}$ (rows) and numbers of inducing points $M \in \{64, 128, 256\}$ (columns). Within each subplot, the x-axis represents the network depths $L \in \{2, 3, 4\}$. Lines correspond to DS-DGP with $S = 4$ MC samples (blue), DS-DGP with $S = 16$ MC samples (green), and WEK-DGP (orange).

Table 5: NLL, RMSE and CRPS on 12 regression datasets. Results are reported as mean ± standard deviation across 5 random training/test/validation splits.

		SVGP-new	SV-DKL	DS-DGP	DSPP	WEK-DGP
NLL	kin8nm	0.145±0.006	0.046±0.015	0.052±0.013	0.194±0.022	**-0.007±0.016**
	kin32nm	**1.079±0.008**	1.198±0.017	1.244±0.057	1.101±0.011	1.088±0.007
	pumadyn8nm	-0.289±0.017	-0.286±0.024	-0.288±0.017	-0.188±0.028	**-0.434±0.032**
	pumadyn32nm	0.989±0.626	0.231±0.069	1.297±0.019	-0.189±0.012	**-0.221±0.028**
	kin40k	-0.447±0.007	-1.826±0.026	-1.263±0.028	**-2.122±0.072**	-2.046±0.037
	protein	0.841±0.009	0.825±0.011	0.860±0.013	0.792±0.023	**0.789±0.017**
	keggdirected	-1.062±0.014	-1.083±0.011	-1.077±0.013	-1.500±0.098	**-1.691±0.093**
	slice	-1.427±0.005	-1.274±0.079	-2.103±0.048	**-2.464±0.127**	-2.216±0.084
	keggundirected	-0.718±0.015	-0.753±0.010	-0.743±0.005	-1.734±0.149	**-2.330±0.156**
	3droad	0.348±0.013	0.481±0.016	0.520±0.013	**0.104±0.054**	0.431±0.091
	song	1.170±0.002	1.165±0.002	1.158±0.002	**1.093±0.002**	1.098±0.002
	buzz	-0.003±0.002	-0.032±0.003	-0.037±0.002	**-0.135±0.004**	**-0.135±0.003**
RMSE	kin8nm	0.282±0.002	0.252±0.003	0.256±0.003	0.295±0.006	**0.249±0.003**
	kin32nm	**0.713±0.005**	0.796±0.019	0.840±0.046	0.728±0.006	0.718±0.005
	pumadyn8nm	0.181±0.003	0.182±0.004	0.181±0.003	0.200±0.005	**0.179±0.004**
	pumadyn32nm	0.748±0.281	0.302±0.022	0.884±0.017	0.196±0.003	**0.193±0.005**
	kin40k	0.147±0.002	**0.042±0.001**	0.064±0.002	0.060±0.004	0.062±0.003
	protein	0.562±0.007	**0.553±0.005**	0.576±0.008	0.584±0.012	0.554±0.009
	keggdirected	0.085±0.001	**0.083±0.001**	0.084±0.001	0.084±0.002	0.087±0.002
	slice	0.043±0.001	0.049±0.012	0.033±0.004	**0.024±0.009**	0.025±0.003
	keggundirected	0.118±0.002	**0.114±0.001**	0.116±0.000	0.117±0.002	0.121±0.002
	3droad	**0.339±0.004**	0.389±0.007	0.406±0.005	0.429±0.019	0.509±0.039
	song	0.780±0.001	0.775±0.002	0.771±0.002	**0.768±0.002**	0.769±0.001
	buzz	0.241±0.001	0.234±0.001	**0.233±0.001**	**0.233±0.000**	0.234±0.000
CRPS	kin8nm	0.157±0.001	0.141±0.002	0.142±0.002	0.164±0.004	**0.137±0.002**
	kin32nm	**0.401±0.004**	0.448±0.009	0.474±0.026	0.409±0.004	0.404±0.003
	pumadyn8nm	0.100±0.002	0.100±0.003	0.100±0.002	0.111±0.003	**0.095±0.002**
	pumadyn32nm	0.422±0.159	0.170±0.012	0.498±0.010	0.111±0.002	**0.109±0.003**
	kin40k	0.081±0.001	**0.021±0.001**	0.035±0.001	0.022±0.002	0.024±0.001
	protein	0.305±0.003	0.295±0.003	0.311±0.004	0.294±0.009	**0.289±0.004**
	keggdirected	0.037±0.001	0.035±0.001	0.037±0.001	**0.030±0.001**	0.031±0.001
	slice	0.027±0.000	0.031±0.004	0.016±0.001	**0.009±0.001**	0.012±0.002
	keggundirected	0.051±0.001	0.049±0.001	0.049±0.000	**0.035±0.001**	**0.035±0.002**
	3droad	**0.181±0.002**	0.209±0.003	0.219±0.002	0.201±0.010	0.252±0.021
	song	0.436±0.001	0.432±0.001	0.430±0.001	**0.418±0.001**	0.420±0.001
	buzz	0.129±0.000	0.125±0.000	0.125±0.000	**0.121±0.000**	**0.121±0.000**

Table 6: RMSE ($\times 10^{-2}$) on 10 symbolic functions across varying network depths (L). Results are reported as mean ± standard deviation across 5 random training/test/validation splits.

	$L = 2$		$L = 3$		$L = 4$	
	KAN	ours	KAN	ours	KAN	ours
Incomplete elliptic int. (1st kind)	5.63±1.47	**2.57±0.55**	1.60±0.39	**0.97±0.32**	1.24±0.10	**0.90±0.42**
Bessel function (1st kind)	**0.60±0.36**	0.86±0.17	0.17±0.13	**0.09±0.04**	0.18±0.05	**0.16±0.16**
Assoc. Laguerre poly. ($k = 0$)	18.98±4.24	**0.79±0.26**	3.74±1.01	**0.20±0.05**	1.54±0.55	**0.68±0.46**
Even Mathieu func. ($m = 3$)	7.23±1.78	**0.14±0.01**	2.12±0.94	**0.07±0.01**	2.33±2.46	**0.09±0.05**
Chebyshev poly. (1st kind)	5.94±4.15	**1.47±0.62**	2.01±0.83	**0.46±0.09**	0.95±0.37	**0.37±0.18**
Chebyshev poly. (2nd kind)	**23.32±1.72**	23.56±0.71	12.32±6.05	**5.99±0.46**	6.98±3.37	**5.05±2.25**
Assoc. Legendre func. ($m = 0$)	5.41±2.87	**4.61±0.37**	4.11±2.41	**3.18±0.54**	3.21±0.39	**2.14±0.44**
Assoc. Legendre func. ($m = 1$)	19.18±5.29	**13.58±0.93**	11.21±2.60	**5.14±2.44**	4.15±1.64	**3.23±1.05**
Real spherical harm. ($l = 3, m = 2$)	2.19±1.59	**0.04±0.01**	2.32±0.32	**0.05±0.02**	1.16±0.40	**0.03±0.01**
Real spherical harm. ($l = 5, m = 3$)	**4.61±0.52**	5.61±7.07	**3.89±0.20**	17.27±8.71	**2.20±0.53**	16.01±10.49