Optimize to generalize in Gaussian processes: An alternative objective based on the Rényi divergence

Xubo Yue and Raed Al Kontar

Industrial & Operations Engineering, University of Michigan, Ann Arbor, MI, USA

ABSTRACT
We introduce an alternative closed-form objective function $\alpha$-ELBO for improved parameter estimation in the Gaussian process ($GP$) based on the Rényi $\alpha$-divergence. We use a decreasing temperature parameter $\alpha$ to iteratively deform the objective function during optimization. Ultimately, our objective function converges to the exact log-marginal likelihood function of $GP$. At early optimization stages, $\alpha$-ELBO can be viewed as a regularizer that smooths some unwanted critical points. At late stages, $\alpha$-ELBO recovers the exact log-marginal likelihood function that guides the optimizer to solutions that best explain the observed data. Theoretically, we derive an upper bound of the Rényi divergence under the proposed objective and derive convergence rates for a class of smooth and non-smooth kernels. Case studies on a wide range of real-life engineering applications demonstrate that our proposed objective is a practical alternative that offers improved prediction performance over several state-of-the-art inference techniques.

1. Introduction

The Gaussian process ($GP$, also known as kriging) is a collection of random variables, any finite number of which has a joint Gaussian distribution (Sacks et al., 1989; Currin et al., 1991). It is widely used to reconstruct functions based on their scattered observations. In the literature, $GP$s were originally used to tackle regression problems in meteorology (Thompson, 1956; Daley, 1993), geostatistics (Matheron, 1973; Journel and Huijbregts, 1978) and spatial statistics (Ripley, 1981).

Over the past two decades, $GP$ theory and its application have seen great success in various statistics areas. These include experimental design (Gramacy and Apley, 2015; Joseph et al., 2019; Krishna et al., 2020), Bayesian optimization (Snoek et al., 2012; Rana et al., 2017), computer experiments and calibration (Kennedy and O’Hagan, 2001; Plumlee, 2019; Gramacy, 2020; Plumlee et al., 2020; Sung et al., 2020; Zhang et al., 2021), reliability (Jones and Johnson, 2009; Alshraideh and Khatatbeh, 2014; Wei et al., 2018), reinforcement learning and bandits (Srinivas et al., 2009) and recently deep learning (Damianou and Lawrence, 2013; Bui et al., 2016; Matthews et al., 2018). Indeed, this success is due to the many desirable properties $GP$s possess, such as their uncertainty quantification capability and highly flexible model priors where prior knowledge can often be readily accommodated in the mean and covariance function. This progress was also observed on a theoretical level. Matthews et al. (2018) proved that a fully connected, feed-forward network would converge to a $GP$ as the network width goes to infinity. This exciting work has brought many insightful connections between $GP$s and deep neural networks (Jacot et al., 2018; Yang, 2019). Chen et al. (2020) showed that mini-batch stochastic gradient descent can be applied in correlated settings, specifically within a $GP$, a result that allowed scaling $GP$s far beyond what is currently possible. Wang, Tuo, and Wu (2019) derived uniform error bounds for $GP$s trained using a Matérn kernel. These bounds were then used to find generalization bounds for Bayesian optimization and sequential experimental design (Martinez-Cantin, 2014; Yue and Kontar, 2020a; Tuo and Wang, 2020).

One critical challenge in $GP$s is parameter estimation, which will subsequently affect the $GP$s predictive and uncertainty quantification capability. Under most commonly used kernel functions, the $GP$ objective function is nonlinear and non-convex. As a result, $GP$s are vulnerable to obtaining parameter estimates with bad generalization power.

To highlight this challenge, we first provide an illustrative example. Consider a $GP$ with a squared exponential kernel whose length and variance scale parameters are 0.1 and 1.5, respectively. Also, assume that the $GP$ has additive noise variance 0.1. Now we generate 50 input points uniformly on $[0, 5]$ from this $GP$. The log-marginal likelihood using these points with respect to the variance scale and additive noise variance is plotted in Figure 1. One can directly observe that even for this simple example, the log-marginal likelihood has two solutions. A good solution (A) with parameters close to the generating model and another solution (B) with a variance scale estimate close to zero and a very large additive noise variance estimate (around 1.65 $\geq$ 0.1). Therefore,
solution B mainly interprets the data as pure noise. Needless to say, with a large initialization of the additive noise variance, one can directly get stuck at B early on in the optimization process. Indeed, this phenomenon is not surprising for GP practitioners, as often GP solutions may predict the data to be random noise around the pre-specified mean function.

Our work aims to take a step forward towards addressing such aforementioned challenges. **Our overarching goal is to help guide GP parameter estimation to better solutions that have improved generalization power to new data.**

Inspired by simulated annealing in optimization (Rose et al., 1990; Dowsland and Thompson, 2012), we provide an approach that can avoid getting stuck in bad solutions (such as B) early on in the optimization procedure. To do so, we construct an alternative objective that is a lower bound to the original GP log-marginal likelihood. Our new objective features a temperature parameter that controls its shape. During the optimization course, we adjust the temperature parameter to gradually deform our objective function. Ultimately, it converges to the exact log-marginal likelihood function of the GP. At early optimization stages, our objective can be viewed as a regularizer that smoothens some unwanted critical points. At late stages, our objective recovers the exact log-marginal likelihood function that guides the optimizer to solutions that best explain the observed data.

We test our model’s performance on a wide variety of simulated and real-life engineering applications. The results highlight the improved generalization power of our approach compared to the current state-of-the-art algorithms. We note that codes are publicly available on this GitHub https://github.com/UMDataScienceLab/Renyi-GP.

### 2. Literature overview

Though it is by no means an exhaustive list, recent advances in parameter estimation for Gaussian processes can be roughly split into six main trends (for further details, see the recent survey in Liu et al. (2018)). **First,** sampling methods such as Markov chain Monte Carlo (MCMC) (Gramacy and Lian, 2012; Frigola et al., 2013; Hensman et al., 2015) and Hamiltonian Monte Carlo (Havasi et al., 2018) have been extensively studied. However, a sampling approximation is usually computationally intensive. Notably, a recent comparison study (Lalchand and Rasmussen, 2019) shows that Variational Inference (VI) can achieve a remarkable performance compared with sampling approaches, whereas the former has better theoretical properties and can be fitted into many existing efficient optimization frameworks. **Second,** Expectation Propagation (EP) (Deisenroth and Mohamed, 2012) is an iterative local message passing method designed for approximate Bayesian inference. Based on this approach, Bui et al. (2017) propose the Power EP (PEP) framework to learn GPs and demonstrate that PEP encapsulates a rich family of approximated GPs such as Fully Independent Training Conditional (FITC) and Deterministic Training Conditional (DTC) (Bui et al., 2017). Though accurate and promising, the EP family, in general, is not guaranteed to converge (Bishop, 2006). **Third,** the VI is an approach to estimate probability densities through efficient optimization algorithms (Hoffman et al., 2013; Hoang et al., 2015; Blei et al., 2017). It approximates intractable posterior distributions using a tractable distributional family $Q$. This approximation, in turn, yields a lower bound that is optimized to learn model parameters. VI has caught the most attention compared with the other approximate inference algorithms, due to its ease of use and elegant theoretical properties. **Fourth,** there has been a recent push on utilizing GPU acceleration and distributed computing to optimize the log-marginal likelihood in GPs. Such approaches leverage Blackbox Matrix-Matrix multiplication, distributed Cholesky factorization, and kernel partitioning (Gardner et al., 2018; Wang, Pleiss, Gardner, Tyree, Weinberger, and Wilson, 2019). **Fifth,** some literature consider low rank or sparse approximation techniques (Gramacy and Haaland, 2016) and covariance tapering (Furrer et al., 2006; Kaufman et al., 2008). In addition to those trends, some notable work also approximates the Gaussian process using Vecchia’s approximation method (Guinness, 2018) and stochastic partial differential equation approximation methods (Lindgren et al., 2011). **Last,** Simulated Annealing (SA) (Henderson et al., 2003) is a temperature-based optimization method that achieves a similar goal to our algorithm. SA is a sampling-based method that seeks to optimize a fixed objective function without changing any landscape structure. Empirically,

![Plots of Log-marginal Likelihood Function](image)

*Figure 1.* Surface of the log-marginal likelihood of $GP$ with respect to the additive noise variance and variance scale parameters.
research has shown that SA needs much more iterations than gradient-based methods to converge (Zomaya and Kazman, 2010) and comes with a heavy computation burden. Indeed, SA has seen very scarce implementation in the GP literature.

3. Notation and brief review

We start by introducing some notations and briefly review the Gaussian process. Assume we have collected $N$ training data points $Y = [y_i]_{i=1}^N$ with corresponding $D$-dimensional inputs $X = [x_i]_{i=1}^N$, where $y_i \in \mathbb{R}$ and $x_i \in \mathbb{R}^D$. We decompose the output as $y_i = f(x_i) + \varepsilon_i$, where $f(\cdot)$ is a GP and $\varepsilon_i(\cdot)$ denotes independent and identically distributed (i.i.d.) additive noise with zero mean and $\sigma^2_r$ additive noise variance. The GP places a prior over functions such that

$$ p(f|X) = \phi(0, K_{f,f}, f) $$

where $\phi(A, B; f)$ represents the normal density function of $f$ with mean $A$ and variance $B$, $f = [f_1, \ldots, f_N]$ is a vector of latent function values $f_i = f(x_i)$ and $K_{f,f} := K(X, X)$ is a covariance matrix whose entries are determined by a kernel function $k(x, x'; \theta)$ parameterized through $\theta$. For example, the squared exponential (SE) kernel admits the form

$$ k(x, x'; \theta) = v^2 \exp \left( \frac{||x - x'||^2}{2\ell^2} \right), $$

where $v$ is a variance scale parameter, $\ell$ is a length parameter and $\theta = (v, \ell)$. Here, we note that for notational simplicity, we assume zero mean GP, and hereon we neglect conditioning on the input $X$ for the sake of compactness.

Often the end goal of a GP is to predict output $f^*$ given new input $x^*$. To do so, the predictive distribution is attained via

$$ p(f^*|Y) = \int p(f^*|f)p(f|Y)df $$

and is given as

$$ f^*|Y \sim N(K_{f^*,f}K_{f,f}^{-1}Y, K_{f^*,f} - K_{f^*,f}K_{f,f}^{-1}K_{f,f^*}). $$

Here $p(f^*|f)$ is the conditional prior derived from the GP prior

$$ f, f^* \sim N\left(0, \begin{pmatrix} K_{f,f} & K_{f,f^*} \\ K_{f^*,f} & K_{f^*,f^*} \end{pmatrix} \right) $$

and $p(f|Y)$ is the posterior of $f$.

Given the predictive distribution, it is clear that good parameter estimation of $(\sigma^2_r, \theta)$ is imperative to GPs. Perhaps the most popular approach for parameter estimation is by directly maximizing the well-known log-marginal likelihood function

$$ p(Y) = \int p(Y|f)p(f)df. $$

Given that $Y|f \sim N(f, \sigma^2_r I)$ the log-marginal likelihood can be written as

$$ L_{\text{marginal}} := \log p(Y) = \log \phi(0, \sigma^2_r I + K_{f,f}; Y). $$

4. An alternative GP objective

4.1. Overview

Instead of directly maximizing the marginal likelihood, we offer an alternative objective that iteratively converges to it. We start by introducing this alternative objective and then discuss its advantages and motivation in Section 5.

We follow the general philosophy of VI, which turns inference into an optimization problem, where an optimal density ($q^*$), relative to some distance measure, is chosen from a distributional family ($Q$) to approximate a target distribution - here the posterior of a GP. To do so, we augment our probability space by $M (\ll N)$ continuous latent variables $U = [u(z_i)]_{i=1}^M$ observed at inputs $Z = [z_1, \ldots, z_M]^T$. We assume that the $U$ are drawn from the same GP prior $p(f)$. $Z$ may be a subset of the input ($X$) or some free parameters, often referred to as pseudo-inputs or inducing points (Snelson and Ghahramani, 2006), over which the optimization is performed.

Notice that from the augmented joint model $p(Y, f, U)$ we still reach the same log-marginal likelihood in (2) through marginalization

$$ p(Y) = \int p(Y, f, U)dfdU = \int p(Y|f)p(f|U)p(U)dfdU $$

where $p(f|U) = \phi(K_{f,u}K_{u,u}^{-1}U, K_{f,f} - Q; f)$ and $Q = K_{f,u}K_{u,u}^{-1}K_{u,f}$. This hints at the fact that one may let $p(U)$ be a density that adds some level of flexibility to the model.

4.2. The $\alpha$-ELBO

Exploiting this added flexibility, we now take a variational route to approximate the joint posterior $p(f, U|Y)$ over the latent variables. We use the Rényi $\alpha$-divergence as a distance measure. This, in turn, will lead to our proposed objective.

The Rényi $\alpha$-divergence, first proposed in Rényi (1961), is a distance measure between two probability density functions ($q$ and $p$) of a continuous random variable:

$$ D_{\alpha}[q||p] = \frac{1}{\alpha - 1} \log \int q(w)^\alpha p(w)^{-1}dw, \alpha \in [0, 1). $$

This divergence contains a rich family of distance measures such as KL-divergence, Bhattacharyya coefficient, and $\chi^2$-divergence. Also, $D_{\alpha}[q||p]$ is continuous and non-decreasing on $\alpha \in [0, 1]$.

In the context of GPs, our goal is to find an optimal posterior density $q^*$ over the latent variables $f, U$ belonging to some distributional family $Q$, by minimizing the Rényi $\alpha$-divergence between the variational density $q(f, U)$ and the target posterior $p(f, U|Y)$

$$ q^*(f, U) := \operatorname{argmin}_{q(f, U) \in Q} D_{\alpha}[q(f, U)||p(f, U|Y)]. $$

Through some algebraic manipulations (Appendix A.1), one can find that

$$ D_{\alpha}[q(f, U)||p(f, U|Y)] = \log p(Y) - L_{z}(q; Y), $$

such that

$$ L_{z}(q; Y) := \frac{1}{1 - \alpha} \log E_{q(f, U)} \left[ \left( \frac{p(f, U|Y)}{q(f, U)} \right)^{-1} \right]. $$

Following (4) and since $D_{\alpha}[q||p] \geq 0$, we have that $L_{z}(q; Y) \leq L_{\text{marginal}}$ is a lower bound on the log-marginal
likelihood and maximizing \( \mathcal{L}_a(q; Y) \) will equivalently minimize \( D_2[q||p] \).

To maximize \( \mathcal{L}_s(q; Y) \), one first needs to find the optimal density. To this end, we exploit a mean-field assumption \( q(f, U) = p(f|U)q(U) \). This, in turn, poses \( q(U) \) as the variational density to be optimized.

Under this mean-field assumption (See Appendix A.2),

\[
\mathcal{L}_s(q; Y) = \frac{1}{1 - \alpha} \log \int p_s(Y|U)q(U)^{\alpha}p(U)^{1-\alpha} dU \\
= \frac{1}{1 - \alpha} \log \mathbb{E}_{q(U)} \{ p_s(Y|U)q(U)^{\alpha-1}p(U)^{1-\alpha} \},
\]

where \( p_s(Y|U) = \int p(f|U)p(Y|f)^{1-\alpha} df \). We slightly abuse notation as \( p_s(Y|U) \) is not a probability density anymore. Our next goal is to find an upper bound of \( \mathcal{L}_s(q; Y) \).

Now, let \( \mathcal{L}_s(q^*; Y) = \max_U p_s(Y|U) \). Fortunately, \( \mathcal{L}_s(q^*; Y) \) can be found in a closed form (See Appendices A.3 and A.4):

\[
\mathcal{L}_s(q^*; Y) = \log \int p_s(Y|U)^{1/(1-\alpha)}p(U)dU \\
= \log \{ \phi(0, \sigma_e^2 I) + (1 - \alpha)K_{f,f} + \alpha Q; Y) \} \\
+ \log \left| I + \frac{1-\alpha}{\sigma_e^2} (K_{f,f} - Q) \right|^{\frac{1}{1-\alpha}}.
\]

Here, we refer to our bound as the Rényi GP or the \( \alpha \)-ELBO as it is an evidence lower bound (ELBO) on the log-marginal likelihood.

Through scrutinizing (6) we directly observe that the new lower bound unifies components from the exact covariance (\( K_{f,f} \)) and an approximate covariance (\( Q = K_{f,U}K_{U,U}^{-1}K_{U,f} \)), also known as the Nyström approximation. Interestingly, when \( \alpha = 0 \), the log-marginal likelihood is recovered, \( \mathcal{L}_0 = \mathcal{L}_{\text{marginal}} = \log p(Y) \). While, for \( \alpha \to 1 \), we recover the traditional VI bound obtained from minimizing the KL divergence distance measure. This indeed is a direct consequence of the fact that \( \lim_{\alpha \to 1} D_2[q||p] = KL[q||p] \) (Titsias and Lawrence, 2010; Tran et al., 2015; Liu et al., 2018; Yue and Kontar, 2020b).

5. Why use \( \mathcal{L}_s \)

\( \mathcal{L}_s \) literature has shown that inference via \( \mathcal{L}_{\text{marginal}} \) can be advantageous on some datasets and \( \mathcal{L}_{\text{VI}} \) on others (Wang, Pleiss, Gardner, Tyree, Weinberger, and Wilson, 2019; Chen et al., 2020). In this section, we shed light on the advantages of our alternative bound \( \mathcal{L}_s \). As will be clarified shortly, \( \mathcal{L}_s \) inherits the regularization benefits of VI at early stages to restrict the complexity of estimated posterior density and then gradually switches to the exact \( \mathcal{L}_s \) to get a good model fit.

5.1. A regularization perspective

We first define the lower bounds below to understand why \( \mathcal{L}_s(q^*; Y) \) is a promising objective. Here \( \mathcal{L}_{\text{VI}} = \lim_{\alpha \to 1} D_2[q||p] \) while \( \mathcal{L}_{\text{jensen}} \) is obtained from a direct Jensen’s inequality on \( \mathcal{L}_s(q; Y) \) (see Appendix B):

\[
\mathcal{L}_s(q; Y) = \frac{1}{1 - \alpha} \log \mathbb{E}_{q(U)} \left[ p_s(Y|U)q(U)^{\alpha-1}p(U)^{1-\alpha} \right] \\
\mathcal{L}_{\text{jensen}} = \frac{1}{1 - \alpha} \int q(U) \log p_s(Y|U)dU - KL[q(U)||p(U)]
\]

where \( \mathcal{L}_{\text{marginal}} \geq \mathcal{L}_s(q; Y) \geq \mathcal{L}_{\text{jensen}} \geq \mathcal{L}_{\text{VI}} \) holds true for any \( \alpha \in [0, 1) \). We focus on \( \mathcal{L}_s(q; Y) \) as \( \mathcal{L}_s(q^*; Y) \) is a by-product from optimizing \( \mathcal{L}_s(q; Y) \). Note that \( KL[q(U)||p(U)] = KL[q(U,f)||p(U,f)] \) since \( q(f, U) = p(f|U)q(U) \). Also, \( p(Y|f) = p(Y|f, U) \).

One can directly observe that the bounds mirror the trade-off between the likelihood and prior (Titsias, 2009). For instance, in \( \mathcal{L}_{\text{jensen}} \), the first term denotes the model fit and encourages the density of the latent variables to place probability mass on configurations that best explain the observed data; this often induces an objective with many local critical points each with a specific interpretation of the data. Whereas the KL term is a regularizer that encourages latent variables to stay close to the prior class. Intrinsically, this regularization restricts the complexity of the estimated posterior density, and hence, offers a trade-off between the fit and complexity of the latent variable estimators. Here \( \alpha \) plays the role of controlling this enforced regularization and is data-dependent. This, in turn, allows data to speak for themselves. Note that in a \( \mathcal{L}_s \), the prior is imposed via the kernel. At the early stages of the optimization, prior regularization encourages kernel hyper-parameters that satisfy both the prior class while, in the meantime, suit the observed data. Ultimately, we are optimizing the log-marginal likelihood to get a good model fit.

Indeed, our method takes advantage of the smoothness imposed by VI. This smoothness was also highlighted from a theoretical perspective. In Stein (2014), it is shown that a low-rank approximation works well only when the underlying \( \mathcal{L}_s \) is very smooth and has a relatively large nugget term. For a highly smooth \( \mathcal{L}_s \) with a very small nugget term (which is usually the case for computer model experiments) or a highly non-smooth \( \mathcal{L}_s \) (e.g., with a Matérn kernel) with or without a large nugget term, the low-rank approximation will be far from optimal. Burt et al. (2019) further theoretically show that a \( \mathcal{L}_s \) with a non-smooth kernel requires more inducing points than a \( \mathcal{L}_s \) with a smooth kernel to achieve a smaller approximation error. In Section 7, we also theoretically validate this statement for our proposed objective function.

In essence, the variational approach acts as a smoother on the objective. We exploit this in our early optimization stages to avoid getting stuck in local optimal solutions early on.

5.2. A fractional posterior perspective

Another insight on the advantages of \( \mathcal{L}_s \) is through fractional posteriors (often referred to as tempered or inexact
Posteriors) where a likelihood is raised to some fractional power. Indeed, fractional posteriors have gained renewed interest in recent years within Bayesian statistics due to their empirical success in improving generalization and their robustness to model mis-specifications (Grünwald, 2012; Miller and Dunson, 2018; Bhattacharya et al., 2019). For instance, Miller and Dunson (2018) showed that raising the likelihood to a well-chosen power induces robustness to a mismatch between the model used and the true data-generating process.

In our context, our posterior over the latent variables is given as

$$q^*(f, U) = p(f|U)q^*(U) \propto p(f|U)p(Y|f)^{1-\alpha} df.$$  

Notice that the data likelihood $p(Y|f)$ is raised to the power of $1-\alpha$. As $\alpha \to 1$, the likelihood $p(Y|f)$ will be flattened and its impact will be reduced. Therefore, as $\alpha \to 1$, the induced regularization will prefer $q$ to be more spread out across configurations of the hidden variables and not only concentrated around ones that best explain the observed data. This decreased dependence on the likelihood also leads to a smoother surface where bad local critical points (often anomalies in the data) are smoothened out. Therefore, when the $\alpha$ value is large, the optimizer will not get stuck in those bad local optima and move to a region that contains desirable optimal solutions. As we iteratively decrease $\alpha$, the surface of $L_\alpha(q^*; Y)$ will be closer to the log-marginal likelihood function and the optimizer will finally converge to a solution that optimizes the exact log-marginal likelihood.

**5.3. An empirical demonstration**

Finally, we demonstrate the advantage of $L_\alpha$ using an illustrative example. We generate 50 data points from a GP with a SE kernel with length parameter 0.01, variance scale parameter 1.5, and additive noise variance parameter 0.01. $\mathbf{x}_{i\sim 50}$ and 10 inducing points are uniformly generated from $[0, 5]$. In Figure 2, we plot $L_\alpha$ with respect to the length parameter (x-axis) and the variance scale parameter (y-axis). Let us start by looking at Figure 2(d) where $\alpha = 0$. This represents the exact GP. It can be seen that there are two maxima: (0.01, 1.5) and (11, 0.3). The former is the global maximal point we aim to recover. Now, if one initializes the gradient method at (12, 0.8), near the local solution (11, 0.3), the optimizer will converge to this local optimal solution, resulting in suboptimal predictions. In contrast, if we start from (11, 0.3), but now use $L_{0.99}$ instead of the exact log-marginal likelihood (see Figure 2(a)), then the optimizer will directly move to (1, 0.3) which is near the region that contains the maximizer (0.01, 1.5). This is because the regularization imposed by $L_{0.99}$ leads to a
smoother and benign surface close to the bad local solution of \((11,0.3)\). Now, starting from \((1.0,0.3)\), as we decrease \(\alpha\), we get closer to \((0.01,1.5)\) and eventually converge to this global optimum \((0.01,1.5)\) as shown in Figures 2(b-d).

We note that a higher-dimensional illustrative example is provided in Appendix F.

### 6. Computation and prediction

We optimize (6) with respect to \(\theta = (\sigma, \theta)\). To optimize (6), one needs to select inducing points \(\mathcal{Z}\) properly. There are three approaches to selecting those inducing points: (i) we can select a subset of data from \(X\) and treat them as inducing points; (ii) we can randomly or uniformly sample inducing points from the input domain; (iii) we can add \(\mathcal{Z}\) as hyper-parameters within our model. In this case, we need to optimize over \((\sigma, \theta, \mathcal{Z})\). All are common approaches in the literature (Alvarez and Lawrence, 2008; Titsias, 2009; Hensman et al., 2013).

During the optimization process, we gradually decrease \(\alpha\) from a large value (e.g., 0.99) to zero. This means we start with the variational objective function and end with the original exact log-marginal likelihood function. Here we emphasize that \(\alpha\) is not a tuning parameter. Instead, it is a temperature that decreases over iterations.

The recent work of Chen et al. (2020) theoretically shows that mini-batch Stochastic Gradient Descent (SGD) can be used for optimizing \(\mathcal{G}P\)s. This result in turn allows scaling \(\mathcal{G}P\)s to very large data size regimes. For instance, in their work, a \(\mathcal{G}P\) with a million data points can be trained within half an hour on a standard laptop. In the experimental section, we use SGD to estimate all parameters. Here we detail the SGD procedure. Denote by \(\xi\) the set of indices corresponding to a subset of training data and \(X_i, Y_i\) the respective subset of inputs and outputs indexed by \(\xi\). Note that \(X_i\) and inducing points \(\mathcal{Z}\) are different. The former represents a subset of the input data \(X\) that will be used to run the SGD algorithm, while the latter is used for \(\mathcal{V}\) and does not need to be a subset of \(X\). Given the subset of training data, we can obtain the stochastic gradient of \(L_n(q'; Y)\) as

\[
\nabla_s(\theta; \xi) := \frac{1}{2} Y_i^T \Xi_1^{-1} \frac{d \Xi}{d \theta} \Xi_1^{-1} Y_i - \frac{1}{2} \text{Tr} \left( \Xi_1^{-1} \frac{d \Xi}{d \theta} \right)
\]

where

\[
\Xi_1 := \sigma^2 \xi + (1 - \alpha) K_{f_f, f_f} + \alpha K_{f_f, v} K_{U, v} K_{U, v},
\]

\[
A_1 := I_1 + \frac{1 - \alpha}{\sigma^2} \left( K_{f_f, f_f} - K_{f_f, v} K_{U, v} K_{U, v} \right),
\]

and

\[
K_{f_f, v} := K(X_i, X_i), K_{f_f, v} := K(X_i, U).
\]

To use SGD, at each iteration \(t\), a subset of training data is taken to update model parameters as

\[
\theta^{(t+1)} = \theta^{(t)} - \eta^{(t)} \left( -\nabla_s(\theta^{(t)}; \xi^{(t)}) \right)
\]

where \(\eta^{(t)}\) is the learning rate at iteration \(t\). We summarize this procedure in Algorithm 1.

#### Algorithm 1: Optimization Algorithm

**Data:** Number of iterations \(T\), SGD learning rate schedule \(\{\eta^{(t)}\}_{t=1}^T\), initial model parameter \(\theta^{(0)}\), \(\alpha^{(0)} = 0.99\).

**for** \(t = 0 : (T - 1)\) **do**

\[
\alpha^{(t+1)} = \alpha^{(t)} - \frac{\eta^{(t)}}{C_0} \quad \text{(other decreasing schedules may be also used)};
\]

Randomly sample a subset of data from \((X, Y)\) and denote indices as \(\xi^{(t)}\);

\[
\theta^{(t+1)} = \theta^{(t)} - \eta^{(t)} \left( -\nabla_s(\theta^{(t)}; \xi^{(t)}) \right);
\]

**end**

Return \(\theta^{(T)}\).

Upon obtaining the estimated \(\mathcal{G}P\) model parameters, our next step is to conduct prediction. Here the predictive equations are the same as the exact \(\mathcal{G}P\) predictions defined in (1). Note that in (1) we describe \(p(f^*|Y)\). To find the density of the predicted output (i.e., \(p(Y^*|Y)\)) one needs to add the additive noise to the predictive variance.

One drawback of SGD (Chen et al., 2020) is that in the prediction phase, using SGD implies only using a batch of the data to perform predictions. This is sub-optimal, specifically since the prediction is a one-shot problem, unlike the iterative procedure of learning parameters. To overcome this difficulty, we employ the recently proposed algorithm - Blackbox Matrix-Matrix Multiplication (BBMM) (Gardner et al., 2018; Wang, Pleiss, Gardner, Tyree, Weinberger, and Wilson, 2019). The BBMM is an efficient approach to conducting matrix products. This algorithm offers a fast way to calculate the predictive distribution using conjugate gradients (CG), pivoted Cholesky decomposition, and parallel computing. We here note that, in addition to using the BBMM algorithm in the prediction, we also employed it in the parameter estimation stage to further speed up the SGD algorithm. We defer all detailed information to Appendix C.

#### 7. Theoretical properties

In this section, we briefly give an overview of our theoretical results and some important insights. Detailed results can be found in Appendices D and E.

Below we provide our main theorem on the convergence rate of the proposed \(\alpha\)-ELBO in terms of the number of samples.

**Theorem 1.** Suppose the input dimension is \(D = 1\) and \(k(x, x; \theta) \leq v_0, \forall x \in \mathbb{R}\). Assume \(N\) training inputs are drawn i.i.d. according to an input density \(p(x)\). Under some mild conditions (See Appendix D.3), for \(\epsilon > 0\), with probability at least \(1 - \delta\),

\[
D_\alpha[p|p] \leq \frac{\alpha}{2\delta(1 - \alpha)} \log \left[ 1 + \frac{1 - \alpha}{\sigma^2} \left( (M + 1)C + 2Nv_0\epsilon \right) \frac{N}{N} \right]
\]

\[
+ \alpha \frac{(M + 1)C + 2Nv_0\epsilon}{2\delta\alpha^2} \frac{|Y|^2}{\alpha^2},
\]

where

\[
M := T \left( \sum_{t=1}^{T} \frac{\alpha^{(t-1)}}{\eta^{(t)}} \right) \left( \sum_{t=1}^{T} \frac{\eta^{(t)}}{\alpha^{(t)}} \right),
\]

\[
C := \max_{t, \xi} \frac{1}{2} Y_i^T \Xi_1^{-1} \frac{d \Xi}{d \theta} \Xi_1^{-1} Y_i - \frac{1}{2} \text{Tr} \left( \Xi_1^{-1} \frac{d \Xi}{d \theta} \right),
\]

\[
\sigma^2 := \frac{1}{N} \sum_{t=1}^{T} \frac{\alpha^{(t-1)}}{\eta^{(t)}} \left( \frac{1}{2} Y_i^T \Xi_1^{-1} \frac{d \Xi}{d \theta} \Xi_1^{-1} Y_i - \frac{1}{2} \text{Tr} \left( \Xi_1^{-1} \frac{d \Xi}{d \theta} \right) \right),
\]

\[
v_0 := \frac{1}{N} \sum_{t=1}^{T} \frac{\alpha^{(t-1)}}{\eta^{(t)}} \left( \frac{1}{2} Y_i^T \Xi_1^{-1} \frac{d \Xi}{d \theta} \Xi_1^{-1} Y_i - \frac{1}{2} \text{Tr} \left( \Xi_1^{-1} \frac{d \Xi}{d \theta} \right) \right).
where \( C = N \sum_{m=M+1}^{\infty} \lambda_m \) and \( \lambda_m \) are the eigenvalues of the integral operator associated with the kernel \( k(\cdot, \cdot; \theta) \) and its input density \( p(x) \).

To see the value of Theorem 1, we will further simplify the derived upper bound by considering different kernels.

We will provide a convergence result with the SE kernel. For the SE kernel, if \( p(x) = \phi(0, \tau; x) \), then the \( m \)th eigenvalue of \( k(\cdot, \cdot; \theta) = \lambda_m = \sqrt{2a/AB}^{m-1} \), where \( a = 1/(4\tau^2) \), \( b = 1/(2\tau^2) \), \( c = \sqrt{a^2 + 2ab} \), \( A = a + b + c \) and \( B = b/A \). Recall that \( \ell \) is the length parameter, and \( v \) is the variance scale parameter. We then can obtain \( \sum_{m=M+1}^{\infty} \lambda_m = \frac{\sqrt{\pi} \pi}{(1-B)^{3/2}} B^M \) (Zhu et al., 1997).

**Corollary 1.** Suppose \( ||Y||^2 \leq RN \), where \( R \) is a constant. Fix \( \gamma > 0 \) and take \( \epsilon = \frac{6\log^2 M}{N^2 \gamma^2} \). Using an SE kernel we have that with high probability \( 1 - \delta \),

\[
D^s[q||p] \leq 2\epsilon \left( 1 + \frac{6\log^2 M}{N^2 \gamma^2} \right) \frac{1}{\delta^2 (1 - \epsilon)} \log \left[ 1 + (1 - \epsilon) \left( \frac{4\delta}{N^2 + 2} \right)^N \right],
\]

when inference is performed with

\[
M = \frac{(3 + \gamma) \log N + \log \eta}{\log (B^{-1})}, \quad \eta = \frac{\sqrt{2a}}{a \sqrt{A} \delta (1 - B)}.
\]

This corollary is proved in Appendix E.3. It implies that the number of inducing points should be of order \( O(\log N) \) (i.e., sparse). In a high-dimensional input space, following a similar proof, we can show that this order becomes \( O(\log^2 N) \). Additionally, we can see that the upper bound goes to zero as \( N \to \infty \). This implies \( D^s[q||p] \) can be arbitrarily small, and the approximation of the true posterior becomes more accurate as the sample size increases. For the Matérn kernels, we show that the number of required inducing points is \( O(N^t) \), where \( t \) is a factor that controls the smoothness of the kernel. It can be seen that we require more inducing points when we are using non-smooth kernels such as Matérn kernels. Details can be found in Appendices D.4 and E.4.

### 8. Experiments

We benchmark our model with the recent state-of-the-art methods: (i) the exact inference procedure for GPs (EGP) (Wang, Pleiss, Gardner, Tyree, Weinberger, and Wilson, 2019; Chen et al., 2020). This method directly optimizes the exact likelihood function \( \mathcal{L}_{\text{marginal}} \). We use SGD to estimate parameters and use BBMM to obtain predictions; (ii) the stochastic variational GP (SGP) (Hensman et al., 2013; Hoffman et al., 2013). This method performs stochastic VI to the exact GP and optimizes the derived variational lower bound; (iii) the Power Expectation Propagation (PEP) (Bui et al., 2017) with optimal tuning \( \alpha_{\text{PEP}} \) values. To ensure a fair comparison, parameters for all benchmark models are estimated by SGD. Here please note that \( \alpha_{\text{PEP}} \) is a tuning parameter.

#### 8.1. A toy example

We first investigate the performance of our method on well-known simulated functions with 1000 data points in various dimensions. Data is from the Virtual Library of Simulation Experiments (http://www.sfu.ca/~surjano/index.html). The testing functions are Gramacy & Lee function (GL, \( D = 1 \)), Branin-Hoo function and Griewank-D function (GD, \( D \geq 2 \)). For each dataset, we randomly split 60% data as training sets and 40% as testing sets. We set the number of inducing points to 50. Throughout the experiments, we use the Matérn kernel. For each function, we run our model 30 times with different initial parameters. For each repetition, during the optimization process, we gradually decrease \( \alpha \) from 0.99 to 0. The performance of each model is measured by Root Mean Square Error (RMSE).

We report results from Gramacy & Lee function, Branin-Hoo function, and Griewank-D function in Table 1. The results clearly indicate that our model has the smallest RMSE among all benchmark models. Here, we note that EGP outperforms the SGP in some cases whereas the opposite happens sometimes. This observation aligns with the conclusion made by Wang, Pleiss, Gardner, Tyree, Weinberger, and Wilson (2019): SGP is not necessarily better than EGP and vice versa. Overall, our Rényi objective can deliver improvements compared with other benchmark models. This credits to the additional temperature parameter \( \alpha \). As we have illustrated previously (Section 5), the variational objective \( (\alpha > 0) \) acts like a smoother and more likely pushes optimizer to the global optimal solution of the exact log-marginal likelihood function \( (\alpha = 0) \).

| Table 1. Simulation results. |
|--------------------------------|
| RMSE | GL | BH | GD (D = 4) |
|------|----|----|----------|
| Rényi | 0.001 (±0.000) | 0.009 (±0.002) | 0.020 (±0.003) |
| EGP  | 0.003 (±0.000) | 0.017 (±0.003) | 0.027 (±0.002) |
| SGP  | 0.002 (±0.000) | 0.014 (±0.002) | 0.033 (±0.002) |
| PEP  | 0.002 (±0.000) | 0.018 (±0.001) | 0.029 (±0.003) |

| Table 2. RMSE of all models on different datasets. The RMSE is calculated over 30 replications with different initial points. Standard deviations of RMSEs are also reported. The log-marginal likelihood values are reported in Table 3. |
|---------------------------------|
| Dataset | N  | EGP | SGP | PEP (optimal \( \alpha_{\text{PEP}} \)) | Rényi |
|--------|----|-----|-----|---------------------------------|------|
| Bike   | 17,389 | 0.221 ± 0.003 | 0.305 ± 0.001 | 0.288 ± 0.009 | 0.203 ± 0.001 |
| C-MAPSS | 33,727 | 0.633 ± 0.051 | 0.597 ± 0.055 | 0.642 ± 0.083 | 0.545 ± 0.032 |
| Protein | 29,267 | 0.536 ± 0.012 | 0.577 ± 0.008 | 0.539 ± 0.012 | 0.500 ± 0.008 |
| Traffic | 48,204 | 0.125 ± 0.001 | 0.121 ± 0.003 | 0.124 ± 0.001 | 0.119 ± 0.000 |
| Battery | 104,046 | 0.194 ± 0.003 | 0.305 ± 0.005 | 0.242 ± 0.014 | 0.155 ± 0.001 |
| House Electric | 1,311,539 | 0.053 ± 0.000 | 0.088 ± 0.005 | 0.049 ± 0.007 | 0.041 ± 0.001 |

\( N \) is the number of data points, \( EGP \) is the exact inference procedure for GPs, \( SGP \) is the stochastic variational GP (SGP) (Hensman et al., 2013; Hoffman et al., 2013). This method performs stochastic VI to the exact GP and optimizes the derived variational lower bound; \( PEP \) is the Power Expectation Propagation (PEP) (Bui et al., 2017) with optimal tuning \( \alpha_{\text{PEP}} \) values. To ensure a fair comparison, parameters for all benchmark models are estimated by SGD. Here please note that \( \alpha_{\text{PEP}} \) is a tuning parameter.
8.2. Real engineering data

We benchmark the Rényi $\mathcal{GP}$ on a range of datasets that include the (1) Bike, (2) Traffic (3) Protein and (4) House electric datasets from the UCI data repository (Asuncion and Newman, 2007) (https://archive.ics.uci.edu/ml/datasets.php). (5) Battery data from the General Motors Onstar System and (6) C-MAPSS aircraft turbofan engines dataset provided by the National Aeronautics and Space Administration (NASA) (https://ti.arc.nasa.gov/tech/dash/groups/pcoe/).

The size of these datasets ranges from 17,389 to 1,311,539 data points. For each dataset, we randomly split 60% data as training sets and 40% as testing sets and replicate the experiment 30 times. All data are standardized to have a mean of zero and a variance of one.

Inference: For $\alpha$-ELBO, SGP, and PEP, we use SGD to optimize all hyperparameters. The inducing points are uniformly generated within the input domain without using pseudo-random numbers. For EGP, we use batch size 64 with a learning rate 0.01. The number of epochs is set to be 100. Prediction: For mBCG algorithm, we use a diagonalscaling-preconditioning-matrix to stabilize the algorithm and boost convergence speed (Takapoui and Javadi, 2016). In mBCG, the maximum number of iterations is set to be 10N.

8.3. Results and discussion

Experimental results are reported in Table 2. The performance of each model is measured by RMSE. The RMSE is calculated over 30 experiments with different initial points. We also report the log-likelihood in Table 3. Based on Table 2, we can obtain some important insights. First, the results indicate that Rényi $\mathcal{GP}$ achieves the smallest RMSE among all benchmarks on all datasets ranging from small data regimes ($N \approx 17,000$) to large data regimes ($N \approx 1,300,000$). As initial points for all replications are different, results in Table 2 further show that our proposed approach is robust to parameter initialization. This robustness is also a direct consequence of smoothing at early stages, which allows optimizers to avoid getting stuck early on at neighboring solutions. Second, EGP seems to outperform SGP on some data sets while the opposite happens on others; both while being inferior to PEP and $\alpha$-ELBO. This confirms that neither marginal nor sparse/variational inference is always superior over the other as the extent of regularization needed is data dependent. We also observe that PEP sometimes does not converge as seen in Table 2 where standard deviations for RMSE of PEP are sometimes large. This is not surprising as PEP is a heuristic that currently lacks theoretical backing. Furthermore, the advantages of our model become increasingly significant when the sample size increases. This reveals that controlling the smoothness of the ELBO is necessary and promising when we have big and high dimensional data. Thirdly, we report the running time of all models in Table 4. As shown in the table, the training time for $\alpha$-ELBO is comparable to PEP on the House Electric dataset with 1,300,000 data points (note that all methods are trained with 100 epochs). They are slower than EGP since EGP does not need to learn the distribution of inducing points. When we uniformly distribute inducing points and do not optimize them, the running times are similar to EGP. Lastly, as shown in Table 3, our approach can retain higher likelihood values. These results further demonstrate that our algorithm is able to recover better solutions.

We here should note one shortcoming of our algorithm. We observed that our proposed algorithm, using the decaying scheduler $x^{t+1} = x^{t} - \frac{a}{T}$, failed to work well when the temperature parameter $\alpha$ was decreased too fast (i.e., with few numbers of iterations). This is intuitively understandable as the optimizer does not have enough iterations to explore the optimization landscape and will be more likely to get trapped in a bad local optimal solution. This issue is in fact very common in many temperature-based optimizers such as simulated annealing (Henderson et al., 2003).

8.4. Different choice of $M$

In our model formulation, we augment our probability space by $M$ continuous latent variables observed at inputs $Z = [z_1, ..., z_M]^T$. One natural question is how many inducing points we should have. In Section 7, we provide some solutions based on our derived theory. For example, we show that $M = O(\log^2 N)$ for smooth kernels. In this section, we conduct a sensitivity analysis with different $M$ on the large-scale House Electric dataset from the UCI repository. By employing SGD, the $\alpha$-ELBO can be trained efficiently using one RTX-2080 GPU. Tables 2 (using $M = 50$) and 5 (using $M = 128, 256, 512, 1152$) report the RMSE of all models. The results show that across all $M$s, the performance of our model is superior to exact $\mathcal{GP}$ (see Table 2 - EGP). Also, there are marginal differences in the performance across the

| Dataset            | EGP       | SGP       | PEP       | Rényi     |
|--------------------|-----------|-----------|-----------|-----------|
| Bike               | 0.41 ± 0.02 | 0.15 ± 0.03 | 0.10 ± 0.01 | 0.45 ± 0.01 |
| C-MAPSS            | −1.00 ± 0.01 | −1.45 ± 0.01 | −1.55 ± 0.02 | −0.01 ± 0.01 |
| Protein            | 2.15 ± 0.01 | 1.42 ± 0.01 | 1.97 ± 0.05 | 2.99 ± 0.04 |
| Traffic            | −0.42 ± 0.01 | −0.47 ± 0.02 | −1.07 ± 0.01 | −0.20 ± 0.04 |
| Battery            | 2.23 ± 0.06 | 2.10 ± 0.02 | 2.17 ± 0.02 | 2.55 ± 0.01 |

| Dataset            | N         | EGP       | SGP       | PEP       | Rényi     |
|--------------------|-----------|-----------|-----------|-----------|-----------|
| House Electric     | 1,311,539 | 66.2 ± 0.7 | 1269.3 ± 9.3 | 123.1 ± 8.2 | 129.6 ± 8.5 |
| House Electric     | 1,311,539 | 66.2 ± 0.7 | 64.2 ± 1.3  | 67.3 ± 1.1  | 69.6 ± 0.8  |

| Dataset            | N         | EGP       | SGP       | PEP       | Rényi     |
|--------------------|-----------|-----------|-----------|-----------|-----------|
| House Electric     | 1,311,539 | 66.2 ± 0.7 | 1269.3 ± 9.3 | 123.1 ± 8.2 | 129.6 ± 8.5 |
| House Electric     | 1,311,539 | 66.2 ± 0.7 | 64.2 ± 1.3  | 67.3 ± 1.1  | 69.6 ± 0.8  |
In Algorithm 1, we decrease the temperature parameter linearly. Specifically, at each iteration \( t \), we set \( \alpha^{t+1} = \alpha^t - \frac{0.99}{T} \). We here note that it is possible to consider other types of temperature schedulers.

One option is to define \( \alpha \) in terms of the gradient norm. To do so, we define

\[
\alpha^{(t)} = \frac{\|g^{(t)}\|}{\|g^{(1)}\|} \alpha^{(1)},
\]

where \( g^{(t)} \) is the stochastic gradient at iteration \( t \). In essence, this schedule specifies \( \alpha \) based on the ratio between the gradient norm at iteration \( t \) and the initial gradient norm. As the algorithm converges to the optimal solution, \( \|g^{(t)}\| \) will be close to zero, and \( \alpha^{(t)} \) will go to zero accordingly. The overall trend of \( \alpha^{(t)} \) with respect to \( t \) is decreasing, although it is not necessarily monotonic due to stochasticity. We have conducted additional experiments using this scheduler. The results are presented in Table 6. It can be seen that the norm-based scheduler offers a similar result compared to our original scheduler.

We also added experiments using a two-staged scheduler that admits a large \( \alpha \) at the early stage and switches into a small \( \alpha \) afterward. Specifically, we set \( \alpha = 0.9 \) in the first half of the iterations and set \( \alpha \) to 0.05 during the rest of the iterations. The results are shown in Table 7. This scheduler offers slightly inferior performance. We believe the reason is that the temperature parameter \( \alpha \) is not decreasing gradually. Instead, it suddenly transitions from a large value to a small value. This issue is similar to the observation we had when using a small \( T \) for our decaying schedule.

### Table 6. RMSE of all models on different datasets. The RMSE is calculated over 30 replications with different initial points.

| Dataset       | N   | EGP       | SGP       | PEP (optimal \(\alpha_{EP}\)) | R\(\alpha\) (norm scheduler) |
|---------------|-----|-----------|-----------|-------------------------------|-------------------------------|
| Bike          | 17,389 | 0.221 ± 0.003 | 0.305 ± 0.001 | 0.288 ± 0.009 | 0.215 ± 0.002 |
| C-MAPSS       | 33,727 | 0.633 ± 0.051 | 0.597 ± 0.055 | 0.642 ± 0.083 | 0.507 ± 0.018 |
| Protein       | 29,267 | 0.536 ± 0.012 | 0.577 ± 0.008 | 0.539 ± 0.012 | 0.520 ± 0.011 |
| Traffic       | 48,204 | 0.125 ± 0.001 | 0.121 ± 0.003 | 0.124 ± 0.001 | 0.120 ± 0.002 |
| Battery       | 104,046 | 0.194 ± 0.003 | 0.305 ± 0.005 | 0.242 ± 0.014 | 0.149 ± 0.001 |
| House Electric| 1,311,539 | 0.053 ± 0.000 | 0.088 ± 0.005 | 0.049 ± 0.007 | 0.044 ± 0.002 |

Note: Bold values signify the best performance.

### Table 7. RMSE of all models on different datasets. The RMSE is calculated over 30 replications with different initial points.

| Dataset               | \(R\)\(\alpha\) (Two-staged scheduler) |
|-----------------------|-----------------------------------|
| Bike                  | 0.225 ± 0.001                     |
| C-MAPSS               | 0.588 ± 0.014                     |
| Protein               | 0.529 ± 0.013                     |
| Traffic               | 0.130 ± 0.003                     |
| Battery               | 0.147 ± 0.001                     |
| House Electric        | 0.046 ± 0.001                     |

\( M \) values, with \( M = 50 \) being the best. This observation may be explained by the fact that a smaller \( M \) will further restrict the complexity, allowing the model to better navigate the early stages of the complex and large-scale \((N \approx 10^6)\) House Electric dataset.

### 8.5. Different \( \alpha \) scheduler

In Algorithm 1, we decrease the temperature parameter linearly. Specifically, at each iteration \( t \), we set \( \alpha^{t+1} = \alpha^t - \frac{0.99}{T} \). We here note that it is possible to consider other types of temperature schedulers.

One option is to define \( \alpha \) in terms of the gradient norm. To do so, we define

\[
\alpha^{(t)} = \frac{\|g^{(t)}\|}{\|g^{(1)}\|} \alpha^{(1)},
\]

where \( g^{(t)} \) is the stochastic gradient at iteration \( t \). In essence, this schedule specifies \( \alpha \) based on the ratio between the gradient norm at iteration \( t \) and the initial gradient norm. As the algorithm converges to the optimal solution, \( \|g^{(t)}\| \) will be close to zero, and \( \alpha^{(t)} \) will go to zero accordingly. The overall trend of \( \alpha^{(t)} \) with respect to \( t \) is decreasing, although it is not necessarily monotonic due to stochasticity. We have conducted additional experiments using this scheduler. The results are presented in Table 6. It can be seen that the norm-based scheduler offers a similar result compared to our original scheduler.

We also added experiments using a two-staged scheduler that admits a large \( \alpha \) at the early stage and switches into a small \( \alpha \) afterward. Specifically, we set \( \alpha = 0.9 \) in the first half of the iterations and set \( \alpha \) to 0.05 during the rest of the iterations. The results are shown in Table 7. This scheduler offers slightly inferior performance. We believe the reason is that the temperature parameter \( \alpha \) is not decreasing gradually. Instead, it suddenly transitions from a large value to a small value. This issue is similar to the observation we had when using a small \( T \) for our decaying schedule.

### 9. Conclusion

We introduce an alternative objective for obtaining parameter estimates in GPs, based on the R\(\alpha\)nyi-z divergence. This bound offers a structured balance between model fit and prior regularization and, therefore, is capable of controlling the enforced regularization on the objective function. Through many case studies on engineering datasets and applications, we demonstrate that our proposed objective is a practical alternative that offers improved prediction performance over several state-of-the-art inference techniques.

One possible future direction is to study the link between good parameter estimation and good generalization power. Recent literature on generalization has shown that flat solutions are often very desirable for better generalization. Recent papers have also investigated this from a theoretical perspective (Kawaguchi et al., 2017; Foret et al., 2020). Indeed, we believe that the smoothing effect at early stages imposed by the \( \alpha \) divergence may help the algorithm avoid sharp solutions (by smoothing them out). This is an interesting observation that we are actively investigating.

### Funding

The authors acknowledge the generous support from the NSF CAREER AWARD 2144147.

### Notes on contributors

**Xubo Yue** is a PhD candidate in the Department of Industrial & Operations Engineering at the University of Michigan. His research focuses on federated and distributed data analytics. Currently, he is developing federated data analytics methods that rethink how both prescriptive and predictive analytics are achieved within IoT-enabled systems, specifically manufacturing and renewable energy. He has received several best paper awards from the Institute for Operations Research and the Management Sciences (INFORMS), the Institute of Industrial and Systems Engineers (IIE), and other renowned organizations.

**Raed Al Kontar** is an assistant professor in the Industrial & Operations Engineering Department at the University of Michigan and an affiliate with the Michigan Institute for Data Science. Raed’s research focuses on collaborative, distributed, and decentralized data science. Raed obtained an undergraduate degree in civil & environmental engineering and mathematics from the American University of Beirut in 2014 and a master’s degree in statistics in 2017 and a PhD degree in Industrial & System Engineering in 2018, both from the University of Wisconsin-Madison.

### ORCID

Raed Al Kontar https://orcid.org/0000-0002-4546-324X
Matheron, G. (1973) The intrinsic random functions and their applications. Advances in Applied Probability, 5(3), 439–468.

Matthews, A.G.d.G., Rowland, M., Hron, J., Turner, R.E. and Ghahramani, Z. (2018) Gaussian process behaviour in deep neural networks. arXiv preprint arXiv:1804.11271.

Miller, J.W. and Dunson, D.B. (2018) Robust Bayesian inference via coarsening. Journal of the American Statistical Association, 81(3), 519–545.

Plumlee, M. (2019) Computer model calibration with confidence and consistency. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 81(3), 519–545.

Plumlee, M., Erickson, C., Ankenman, B. and Lawrence, E. (2020) Composite grid designs for adaptive computer experiments with fast inference. Biometrika, 108(3), 749–755.

Rana, S., Li, C., Gupta, S., Nguyen, V. and Venkatesh, S. (2017) High dimensional Bayesian optimization with elastic Gaussian process, in Proceedings of the 34th International Conference on Machine Learning, Volume 70. Association for Computing, New York, NY, pp. 2883–2891.

Rényi, A. (1961) On measures of entropy and information, in Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics, The Regents of the University of California, Berkeley, CA.

Ripley, B.D. (1981) Spatial Statistics, Volume 575. John Wiley & Sons, Hoboken, NJ.

Rose, K., Gurewitz, E. and Fox, G. (1990) A deterministic annealing approach to clustering. Pattern Recognition Letters, 11(9), 589–594.

Sacks, J., Welch, W.J., Mitchell, T.J. and Wynn, H.P. (1989) Design and analysis of computer experiments. Statistical Science, 4, 409–423.

Snelson, E. and Ghahramani, Z. (2006) Sparse Gaussian processes using pseudo-inputs, in Advances in Neural Information Processing Systems, pp. 1257–1264.

Snook, J., Larochelle, H. and Adams, R.P. (2012) Practical Bayesian optimization of machine learning algorithms, in Advances in Neural Information Processing Systems, Curran Associates Inc., Red Hook, NY, pp. 2951–2959.

Srinivas, N., Krause, A., Kakade, S.M. and Seeger, M. (2009) Gaussian process optimization in the bandit setting: No regret and experimental design. arXiv preprint arXiv:0912.3995.

Stein, M.L. (2014) Limitations on low rank approximations for covariance matrices of spatial data. Spatial Statistics, 8, 1–19.

Sung, C.-L., Hung, Y., Rittase, W., Zhu, C. and Jeff Wu, C. (2020) A generalized Gaussian process model for computer experiments with binary time series. Journal of the American Statistical Association, 115(530), 945–956.

Takapoui, R. and Javadi, H. (2016) Preconditioning via diagonal scaling. arXiv preprint arXiv:1610.03871.

Thompson, P.D. (1956) Optimum smoothing of two-dimensional fields. Tellus, 8(3), 384–393.

Titsias, M. (2009) Variational learning of inducing variables in sparse Gaussian processes, in Artificial Intelligence and Statistics, Artificial Intelligence and Statistics, Clearwater Beach, FL, pp. 567–574.

Titsias, M. and Lawrence, N.D. (2010) Bayesian Gaussian process latent variable model, in Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Artificial Intelligence and Statistics, Sardinia, Italy, pp. 844–851.

Tran, D., Ranganath, R. and Blei, D.M. (2015) The variational Gaussian process. arXiv preprint arXiv:1511.06499.

Tuo, R. and Wang, W. (2020) Kriging prediction with isotropic matern correlations: Robustness and experimental designs. Journal of Machine Learning Research, 21(187), 1–38.

Wang, K.A., Pleiss, G., Gardner, J.R., Tyree, S., Weinberger, K.Q. and Wilson, A.G. (2019) Exact Gaussian processes on a million data points. In 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

Yang, G. (2019) Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation. arXiv preprint arXiv:1902.04760.

Yue, X. and Kontar, R.A. (2020a) Why non-myopic Bayesian optimization is promising and how far should we look-ahead? A study via roll-out, in International Conference on Artificial Intelligence and Statistics, Artificial Intelligence and Statistics, Sicily, Italy, pp. 2808–2818.

Zhang, Q., Chien, P., Liu, Q., Xu, L. and Hong, Y. (2021) Mixed-input Gaussian process emulators for computer experiments with a large number of categorical levels. Journal of Quality Technology, 53(4), 410–420.

Zhu, H., Williams, C.K., Rohwer, R. and Morciniec, M. (1997) Gaussian regression and optimal finite dimensional linear models, Aston University, Birmingham, UK.