Sparse Gaussian Processes for Bayesian Optimization

Mitchell McIntire
Stanford University
Stanford, CA 94305
mcint286@stanford.edu

Daniel Ratner
SLAC National Accelerator Laboratory
Menlo Park, CA 94025
dratner@slac.stanford.edu

Stefano Ermon
Stanford University
Stanford, CA 94305
ermon@cs.stanford.edu

Abstract

Bayesian optimization schemes often rely on Gaussian processes (GP). GP models are very flexible, but are known to scale poorly with the number of training points. While several efficient sparse GP models are known, they have limitations when applied in optimization settings.

We propose a novel Bayesian optimization framework that uses sparse online Gaussian processes. We introduce a new updating scheme for the online GP that accounts for our preference during optimization for regions with better performance. We apply this method to optimize the performance of a free-electron laser, and demonstrate empirically that the weighted updating scheme leads to substantial improvements to performance in optimization.

1 Introduction

Bayesian nonparametric models have seen growing popularity due to their flexibility and modeling power. The core strength of nonparametrics lies in their ability to scale in complexity with the data, making them useful in cases where parametric model selection is challenging. These models have therefore been used successfully in a variety of applications (Kulis & Jordan (2012); Tank et al. (2015); Miller et al. (2015); Johnson & Willsky (2013)).

Gaussian processes (GPs) have emerged as an elegant nonparametric approach to regression. GPs provide a full probabilistic model of the data, and allow us to compute not only the model's prediction at input points but also to quantify the uncertainty in the predictions. While powerful and elegant, the application of GP regression is limited by the poor scaling of GPs (Rasmussen & Williams (2005)). This limitation has motivated the introduction of numerous efficient approaches for approximating the exact GP solution, e.g. Gal et al. (2014); Hensman et al. (2013); Ranganathan et al. (2011).

A common approach to this approximation is to use sparse GPs, which rely on lower-dimensional representations defined by a smaller set of "inducing points" to represent the full GP. Various types of sparse GPs have been introduced, e.g. Snelson & Ghahramani (2006); Lawrence et al. (2003); Titsias (2009); Csató & Opper (2002); Seeger et al. (2003). These varieties tend to differ most in how they perform the selection and management of inducing points; usually a greedy method of some form is used to select points from the data set that minimize an entropy or information loss criterion. A notable exception is the method of Snelson & Ghahramani (2006), who treat inducing point selection as a continuous optimization problem.

Our focus here is on optimization when it is extremely costly to evaluate the objective function. Bayesian optimization is a natural choice in this setting (Jones et al. (1998)). In Bayesian optimization, a probabilistic model of the objective function is used to select sampling points by maximizing an acquisition function based on e.g. the expected improvement in the target variable. Gaussian processes are naturally applicable to Bayesian optimization due to their full probabilistic formulation, which can effectively model the observations of the optimization process; see e.g. Osborne et al. (2009); Snoek et al. (2012) for recent applications of Bayesian optimization using GPs. Other approaches to Bayesian optimization include deep neural networks, as in Snoek et al. (2015).

To date, applications of GPs to Bayesian optimization have typically used full Gaussian process regression. In these settings, it is either assumed that computation time is relatively less important (as compared to e.g. function evaluations), or that convergence will occur quickly enough that the size of the full GP is not an issue. These assumptions might not hold in large parameter spaces, however, particularly in settings with noisy observations that can significantly slow the rate of convergence.

As a result, we consider the application of sparse GPs to Bayesian optimization, as in Nickson et al. (2014). Since sparse GPs have bounded size, the time taken to update during optimization will not increase regardless of how long

the procedure takes to converge. Using sparse GPs for Bayesian optimization presents a different set of challenges than in a typical regression problem, however. In particular, existing sparse GP approaches seek to model the full GP as accurately as possible given the limited size of their representation. This goal is obviously desirable for regression, but has key shortcomings in optimization, namely that the limited resources of the sparse GP may be allocated to closely model regions of parameter space that perform poorly and are therefore less important for optimization.

We propose weighted-update online Gaussian processes (WOGP) as an alternative to typical sparse GP set selection that is better suited to optimization; rather than tailoring the sparse GP for predictive accuracy, WOGPs use an online update scheme that weights the feature space of the GP according to which regions are promising from the optimization perspective. During Bayesian optimization over a large parameter space, this ensures that the sparse model does not waste resources by attempting to accurately model regions that are clearly irrelevant to the optimization problem.

Our work is motivated by an application of Bayesian optimization to improve the performance of the Linac Coherent Light Source (LCLS) free-electron laser (FEL) at the SLAC National Accelerator Laboratory (Emma et al. (2010)). The operational costs of this machine are daunting, and current tuning procedures consume hundreds of hours of machine (and machine operator) time annually that could be better spent conducting the various scientific experiments that rely on LCLS. In this setting, we demonstrate empirically that WOGPs significantly outperform competing techniques.

2 Background

In this section we describe Gaussian process regression and the online sparse GP algorithm introduced in Csató (2002). This algorithm uses online updates and a sparse representation to reduce the GP training complexity. This online scheme is particularly useful for large scale and online regression tasks, since it reduces the time taken to update the GP in each iteration with efficient individual updates.

2.1 Review of Gaussian Process Regression

Formally, a Gaussian process is a collection of random variables $\mathcal{X}$ such that any finite subset $(X_1, \cdots, X_n) \subset \mathcal{X}$ have a joint Gaussian distribution. For example, if we have a GP over the interval $[0, 1]$, then the joint distribution of any finite set of points in $[0, 1]$ is multivariate normal; the mean and covariance of this distribution will be discussed shortly. This GP can be thought of as a distribution over functions $f : [0, 1] \to \mathbb{R}$, as every assignment of values to this interval (or any domain on which a GP is defined) has some probability associated with it via this joint distribution.

A Gaussian process prior is fully defined by its covariance function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and its prior mean $\mu_0 : \mathcal{X} \to \mathbb{R}$. To simplify the discussion, we will assume that the prior mean function is zero, though this need not be the case. The covariance function is required only to be a valid covariance function in that the Gram matrix $K_{i,j} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ of values of any finite subset of $\mathcal{X}$ must be positive semidefinite.

In Gaussian process regression, a GP prior is conditioned on training data to obtain the posterior distribution over the function space. Following the notation of Rasmussen & Williams (2005), given training samples X with corresponding observations $\boldsymbol{f}$ and test inputs X_*, distribution of training observations and test outputs $\boldsymbol{f}_*$ is multivariate Gaussian; conditioning the latter on the former gives us

$$\begin{aligned}\boldsymbol{f}_*|X, X_*, \boldsymbol{f} \sim \mathcal{N}(&K(X_*, X)K(X, X)^{-1}\boldsymbol{f}, \\ &K(X_*, X_*) - K(X_*, X)K(X, X)^{-1}K(X, X_*)) \,. \quad (1)\end{aligned}$$

The resulting posterior at the test locations is a multivariate Gaussian distribution whose mean and covariance are then used in regression. Incorporating the assumption of i.i.d. Gaussian noise into this model is straightforward, involving a simple change to the covariance function according to the standard deviation σ of the assumed noise. This yields the posterior distribution conditioned on noisy observations $\boldsymbol{y}$:

$$\begin{aligned}\boldsymbol{f}_*|X, X_*, \boldsymbol{y} \sim \mathcal{N}(&K(X_*, X)[K(X, X) + \sigma^2 I]^{-1}\boldsymbol{y}, \\ &K(X_*, X_*) - K(X_*, X)[K(X, X) + \sigma^2 I]^{-1}K(X, X_*)) \,. \quad (2)\end{aligned}$$

See Rasmussen & Williams (2005) for a full treatment of GP regression and Gaussian processes in general.

For a predictive distribution conditioned on n training inputs, the time complexity of Gaussian process regression is $\mathcal{O}(n^3)$. This is prohibitively expensive for applications with more than a few thousand inputs (or fewer in settings where runtime is an important consideration). Several approaches have been introduced for approximating full GP regression with more efficient algorithms. The most common category for these approximations is the sparse GP, several varieties of which are listed in Section 1. See Quionero-Candela & Rasmussen (2005) for a thorough treatment of the variety and theory of sparse approximations to full Gaussian process regression.

The common thread among these methods is the attempt to represent the full Gaussian process using a set of $m < n$ inducing inputs, typically leading to a complexity of $\mathcal{O}(m^2 n)$ to train the sparse GP. These inducing inputs are often chosen as a subset of the input data, leading to a difficult combinatorial problem that is typically solved using some form

Algorithm 1 Online GP Update

1: **Input:** data point $\boldsymbol{x}$, output y
2: **Persistent:** Inducing variables $X_{\mathcal{I}}$, GP model parameters
3: Assess novelty γ of point $\boldsymbol{x}$.
4: **if** $\gamma < \epsilon_{tol}$ **then**
5: Perform sparse update without expanding the model.
6: **else**
7: Perform full update, adding $\boldsymbol{x}$ to $X_{\mathcal{I}}$ and extending GP model parameters.
8: **end if**
9: **if** Model size exceeds m **then**
10: Score inducing inputs $X_{\mathcal{I}}$ on impact of removal.
11: Remove the lowest-scoring element of $X_{\mathcal{I}}$; update the GP model to minimize the impact of removal.
12: **end if**

of greedy minimization of information loss. Snelson & Ghahramani (2006) provide one alternative, in which inducing variable selection is treated as a continuous optimization problem. Our approach is most closely related to the algorithm introduced in Csató (2002), which iteratively trains the approximating GP by processing each input individually. This method selects inducing points by continually comparing new data points to the existing set of inducing variables in the model and keeping whichever subset yields the best approximation. This method is described in more detail below.

2.2 Online Sparse GPs

The online sparse GP algorithm of Csató & Opper (2002); Csató (2002) handles the sparse selection problem by observing input data one point at a time. In each iteration, the new data point is added to the sparse model (assuming that the sample passes a geometric novelty threshold), which may increase its size to $m+1$ inducing variables. A reduction step is then performed, which removes one of these inducing variables to restore the sparse GP to size m. Pseudocode for the online update is given in Algorithm 1.

Following Csató (2002), we represent a GP by its covariance function K and its posterior parameterization after t iterations in terms of the $(m \times 1)$ dimensional vector $\boldsymbol{\alpha}_t$ of inducing point coefficients and the $(m \times m)$ matrix C_t which specifies the posterior covariance. We denote by $X_{\mathcal{I}}$ the set of inducing variables, giving us the posterior predictive distribution at query points X^* as

$$\mathcal{N}(K(X^*, X_{\mathcal{I}})\boldsymbol{\alpha}_t, \\ K(X^*, X^*) + K(X^*, X_{\mathcal{I}})C_t K(X_{\mathcal{I}}, X^*)) \,. \quad (3)$$

The covariance function K corresponds to a (possibly infinite-dimensional) feature space $\mathcal{F}$. Specifically, if d is the dimension of the data, there exists a function $\phi : \mathbb{R}^d \to \mathcal{F}$ such that $K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{F}}$ is the inner product of $x_i, x_j \in \mathbb{R}^d$ in $\mathcal{F}$. It is shown in Csató (2002) that the Gaussian process can be viewed as a Gaussian distribution in $\mathcal{F}$. Let Φ be the feature space representation of $X_{\mathcal{I}}$, so $K_{\mathcal{I}} \equiv K(X_{\mathcal{I}}, X_{\mathcal{I}}) = \Phi^\top \Phi$. Then in the feature space $\mathcal{F}$ we have

$$\mathrm{GP}_K(\boldsymbol{\alpha}, C) \sim \mathcal{N}(\Phi\boldsymbol{\alpha}, I_{\mathcal{F}} + \Phi C \Phi^\top) \,, \quad (4)$$

where $I_{\mathcal{F}}$ is the identity matrix in $\mathcal{F}$ and we use the notation $\mathrm{GP}_K(\boldsymbol{\alpha}, C)$ to denote the GP with the corresponding covariance function and parameters. Henceforth we will omit the K in this notation, as it will be clear from context.

This allows for a straightforward computation of the Kullback-Leibler (KL) divergence between two GPs that have the same kernel function. The KL divergence between distributions P and Q is defined as

$$D_{KL}(P \| Q) = \int_{\mathcal{F}} P(\boldsymbol{x}) \log \frac{P(\boldsymbol{x})}{Q(\boldsymbol{x})} d\boldsymbol{x} \,. \quad (5)$$

Note that we need never concern ourselves with the cases $P(\boldsymbol{x}) = 0$ or $Q(\boldsymbol{x}) = 0$ since we deal exclusively with normal distributions in this paper.

Suppose that in iteration $t+1$ a new inducing variable is added to the model, increasing its size to $m+1$. In the approach of Csató (2002), the optimal reduced parameters $\hat{\boldsymbol{\alpha}}$ and $\hat{C}$ are computed by minimizing the KL divergence between $GP \sim \mathrm{GP}(\alpha, C)$, the over-sized GP that we are reducing, and the approximation $\widehat{GP} \sim \mathrm{GP}(\hat{\boldsymbol{\alpha}}, \hat{C})$, subject to the constraint that $\hat{\boldsymbol{\alpha}}$ and $\hat{C}$ have entries of zero corresponding to some inducing variable (i.e. there exists an index i such that the ith element of $\hat{\boldsymbol{\alpha}}$ and the ith row and column of $\hat{C}$ are zero). We assume without loss of generality that it is the last inducing variable that is removed.

Csató (2002) minimizes $D_{KL}(\widehat{GP} \| GP)$ with respect to the parameters $\hat{\boldsymbol{\alpha}}$ and $\hat{C}$, resulting in the update equations (3.19) and (3.22). With $Q \equiv K_{\mathcal{I}}^{-1}$, these equations are

$$\hat{\boldsymbol{\alpha}} = \boldsymbol{\alpha}^{(r)} - \frac{\alpha^*}{c^* + q^*}(\boldsymbol{C}^* + \boldsymbol{Q}^*) \\ \hat{C} = C^{(r)} + \frac{1}{q^*}\boldsymbol{Q}^*\boldsymbol{Q}^{*\top} - \frac{1}{c^* + q^*}(\boldsymbol{C}^* + \boldsymbol{Q}^*)(\boldsymbol{C}^* + \boldsymbol{Q}^*)^\top \,, \quad (6)$$

where $\boldsymbol{\alpha}^{(r)}$ denotes the first m entries of $\boldsymbol{\alpha}$, $C^{(r)}$ is the $(m \times m)$ matrix obtained by omitting the last row and column of C, α^*, c^*, and q^* are the last elements of $\boldsymbol{\alpha}$, $\mathrm{diag}(C)$, and $\mathrm{diag}(Q)$ respectively, and the $(m \times 1)$ vectors $\boldsymbol{C}^*$ and $\boldsymbol{Q}^*$ are the last columns of C and Q respectively, excluding the last entry. Applying the block matrix inversion formula, we can also compute the reduced inverse

Algorithm 2 Bayesian optimization

while Not converged **do**
  Compute $\boldsymbol{x}_{t+1} = \arg\max_{\boldsymbol{x}}(EI(\boldsymbol{x}))$.
  Query objective function at $\boldsymbol{x}_{t+1}$ to get y_{t+1}.

  Augment the data: $\mathcal{D}_{t+1} = \{\mathcal{D}_t, (\boldsymbol{x}_{t+1}, y_{t+1})\}$
  Update the model: $\mathcal{M}_{t+1} = \mathcal{M}_t \leftarrow (\boldsymbol{x}_{t+1}, y_{t+1})$
  $t = t + 1$
end while

Gram matrix $\hat{Q}$:

$$K_{\mathcal{I}} = Q^{-1} \Rightarrow K_{\mathcal{I}}^{(r)} = (Q^{(r)} - \frac{1}{q^*}\boldsymbol{Q}^*\boldsymbol{Q}^{*\top})^{-1}$$
$$\Rightarrow \hat{Q} = Q^{(r)} - \frac{1}{q^*}\boldsymbol{Q}^*\boldsymbol{Q}^{*\top} . \quad (7)$$

Using the update equations (6), scores are computed for each of the $m+1$ inducing variables based on the minimum KL divergence that can be achieved when omitting them. The worst-scoring point is then removed, with updates (6) performed to the appropriate coordinate.

2.3 Bayesian Optimization

Bayesian optimization is a probabilistic approach to optimization that is generally used when queries to the function being optimized are expensive. This method lessens the number of evaluations needed, shifting the burden instead to computation over probabilities by utilizing information from all of the function evaluations to choose the next sampling location. We deal here with Bayesian optimization using Gaussian processes as probabilistic models.

Let $\mathcal{D}_t = \{(\boldsymbol{x}_i, y_i)_{i=1}^t\}$ be the observed data of the first t iterations of optimization, where y_i is the observation of the target variable at the location $\boldsymbol{x}_i$ in parameter space. Then we denote by $\mathcal{M}_t$ the model trained on $\mathcal{D}_t$ (where the ordering of $\mathcal{D}_t$ may matter, e.g. if the model is an online sparse GP). Let $\mathcal{M}_t(\boldsymbol{x}) = (\mu_t(\boldsymbol{x}), \sigma_t(\boldsymbol{x}))$, so that μ and σ give the posterior mean function and variance of the model.

The central idea behind Bayesian optimization is to explore according to an acquisition function which incorporates the current set of observations. In this paper we use the expected improvement as our acquisition function. If $\boldsymbol{x}^*$ is the observed location that maximizes μ_t, the improvement at a point $\boldsymbol{x}$ is defined[1] as

$$I_t(\boldsymbol{x}) = \max(0, \mu_t(\boldsymbol{x}) - \mu_t(\boldsymbol{x}^*)) . \quad (8)$$

As seen in Jones et al. (1998) the expected improvement at a point $\boldsymbol{x}$ can be computed as

$$EI(\boldsymbol{x}) \equiv E[I_t(\boldsymbol{x})] =$$
$$\begin{cases} (\mu_t(\boldsymbol{x}) - \mu_t(\boldsymbol{x}^*))\Phi(Z) + \sigma_t(\boldsymbol{x})\phi(Z) & \sigma_t(\boldsymbol{x}) > 0 \\ 0 & \sigma_t(\boldsymbol{x}) = 0 \end{cases},$$
$$Z = \frac{\mu_t(\boldsymbol{x}) - \mu_t(\boldsymbol{x}^*)}{\sigma_t(\boldsymbol{x})} . \quad (9)$$

Here Φ and ϕ respectively represent the CDF and PDF of the standard normal distribution. Recently, Bull (2011) showed that optimization using the EI criterion gives provably efficient convergence in many settings.

With EI defined and the method of updating the GP model specified, Bayesian optimization is straightforward; pseudocode for this procedure is given in Algorithm 2. Note that to maximize expected improvement we use numerical optimization, since EI cannot be maximized analytically but is extremely cheap to evaluate as compared with the objective function. See Brochu et al. (2010) for a more thorough introduction to Bayesian optimization.

3 Online Sparse Gaussian Processes for Bayesian Optimization

In this section we apply online sparse GPs to Bayesian optimization. This is complicated by the limited size of the sparse GP, which can reduce exploitation by preventing the information gained in an iteration from being fully incorporated, as well as hinder exploration of promising areas by dedicating resources to model regions of poor performance. We therefore introduce the weighted-update online GP (WOGP), our modified online sparse GP scheme, and the resulting Bayesian optimization algorithm. Our approach to online sparse GPs is similar to that of Csató (2002); Csató & Opper (2002), but utilizes a weighted measure of divergence between the Gaussian processes' predictive distributions. This allows us to better allocate the limited modeling capacity of the sparse GP to further the goal of optimization (rather than predictive accuracy).

For example, imagine that we are attempting to maximize performance over a large parameter space. The online sparse GPs studied previously may devote multiple inducing points to modeling a complex region of low performance to minimize the divergence in this area. These points may serve our goal of maximization better by improving the model's resolution in promising regions of parameter space while maintaining only a vague notion of poor performance in other regions.

3.1 The Weighted KL Divergence

We now describe a weighted divergence measure between distributions and compute this divergence for two GPs.

[1] We use $\mu_t(\boldsymbol{x}^*)$ rather than the best observation itself to account for the assumed noise in the observations.

Definition 1. *For probability distributions P and Q and real-valued weighting function f, we define the weighted KL divergence as*

$$D_{KL}^{f}(P\|Q) = \int_{\mathcal{F}} P(\boldsymbol{x}) \log \left(\frac{P(\boldsymbol{x})}{Q(\boldsymbol{x})}\right)^{f(\boldsymbol{x})} d\boldsymbol{x} = \int_{\mathcal{F}} f(\boldsymbol{x}) P(\boldsymbol{x}) \log \frac{P(\boldsymbol{x})}{Q(\boldsymbol{x})} d\boldsymbol{x} \,. \tag{10}$$

We note that f should be non-negative to prevent rewarding differences between the distributions; the goal of weighting is to regard divergence in low-weighted areas not as good, but as acceptable if accuracy in highly weighted regions can be obtained in its place.

Proposition 1. *For real-valued weighting function f, and constant $c \in \mathbb{R}$, the following hold:*

$$D_{KL}^{cf} = cD_{KL}^{f},\ D_{KL}^{f+c} = D_{KL}^{f} + cD_{KL} \,. \tag{11}$$

Proof. Easily computed from (5) and (10). □

We can also see from Proposition 1 that scaling f by a constant factor does not affect relative divergence.

We can write the prediction of GP at a point $\boldsymbol{x}$ in feature space as $\boldsymbol{x}^\top \Phi \boldsymbol{\alpha}$, since $\boldsymbol{x}^\top \Phi$ is the inner product in the feature space corresponding to the evaluation of the kernel function K (see Equation 3). Then we define f^* in terms of the proportional improvement expected at $\boldsymbol{x}$:

$$f^*(\boldsymbol{x}) = 1 + \frac{\boldsymbol{x}^\top \Phi \boldsymbol{\alpha} - y^*}{|y^*|} = \frac{\boldsymbol{x}^\top \Phi \boldsymbol{\alpha}}{|y^*|} \,, \tag{12}$$

where y^* is the best value observed thus far during optimization.

This weighting function f^* will cause promising regions of feature space to be weighted more heavily in the divergence computation. Of course, we immediately see that f^* is negative wherever the GP's posterior mean is negative. In our motivating application this is not much of a concern, as we deal with a non-negative target (laser pulse energy). In other settings and with other weighting functions, adjustments may be necessary to prevent f being negative.

These adjustments may simply take the form of shifting the observations to be positive; if the minimum observation is y_{min} and the minimum value attained by the GPs prior mean function is p_{min}, then we can define $y_0 = -\min(y_{min}, p_{min})$. Incrementing the prior mean and observations of the GP by y_0 simply shifts its posterior mean function above zero without changing the shape of the distribution. This can be seen from the linearity of the GP formulation, for example in Equation 2.

Alternatively, f can be shifted directly to prevent it being negative; from Proposition 1 we have $D_{KL}^{f+c} = D_{KL}^{f} + cD_{KL}$, so this has the effect of averaging the weighted and unweighted divergences in order to ensure that differences between the distributions P and Q where $f < 0$ are not rewarded (since the rewards given by D_{KL}^{f} in these regions will be offset by the cD_{KL} term).

Surprisingly, we can compute a closed form equation for $D_{KL}^{f^*}(GP\|\widehat{GP})$ in terms of m- and $(m+1)$-dimensional GP parameters $\hat{\boldsymbol{\alpha}}$, $\hat{C}$, $\boldsymbol{\alpha}$, and C, despite its formulation in the possible infinite dimensional feature space $\mathcal{F}$. The derivation of this equation can be found the full version of this paper, available online (McIntire et al. (2016b)).

Proposition 2. *Let $GP = \mathrm{GP}(\boldsymbol{\alpha}, C)$, $\widehat{GP} = \mathrm{GP}(\hat{\boldsymbol{\alpha}}, \hat{C})$ be GPs which share the same inducing inputs and covariance function K. Let $K_{\mathcal{I}} = K(X_{\mathcal{I}}, X_{\mathcal{I}}) \equiv Q^{-1}$ and define*

$$\Gamma = I + \frac{(I + K_{\mathcal{I}} C)^\top}{\boldsymbol{\alpha}^\top K_{\mathcal{I}} \boldsymbol{\alpha}},\ \hat{V} = (\hat{C} + Q)^{-1}, \quad w = \mathrm{Tr}[(C+Q)\hat{V} - I] - \log|(C+Q)\hat{V}| \,. \tag{13}$$

Then the weighted KL divergence (10) between GP and $\widehat{GP}$ using weighting function f^* (12) is given by

$$\begin{aligned} D_{KL}^{f^*}(GP\|\widehat{GP}) &\propto 2\boldsymbol{\alpha}^\top(\Gamma^\top - I)\hat{V}(\boldsymbol{\alpha} - \hat{\boldsymbol{\alpha}}) + \\ &\quad (\boldsymbol{\alpha} - \hat{\boldsymbol{\alpha}})^\top \hat{V}(\boldsymbol{\alpha} - \hat{\boldsymbol{\alpha}}) + w \\ &= (2\Gamma\boldsymbol{\alpha} - (\boldsymbol{\alpha} + \hat{\boldsymbol{\alpha}}))^\top \hat{V}(\boldsymbol{\alpha} - \hat{\boldsymbol{\alpha}}) + w \,. \end{aligned} \tag{14}$$

We can obtain some intuition for this formula by separately considering the cases $\hat{\boldsymbol{\alpha}} = \boldsymbol{\alpha}$ and $\hat{C} = C$. In the former case, the first term of (14) vanishes, while if $\hat{C} = C$ the second term vanishes; we can therefore infer roughly that the first term encodes the loss due to reducing α, while w measures loss from reducing C to $\hat{C}$. For a noise-free, full (non-sparse) GP, we have $C = -K_{\mathcal{I}}^{-1} \Rightarrow \Gamma = I$. In this case, (14) reduces to the unweighted KL divergence:

$$D_{KL}(GP\|\widehat{GP}) = (\boldsymbol{\alpha} - \hat{\boldsymbol{\alpha}})^\top \hat{V}(\boldsymbol{\alpha} - \hat{\boldsymbol{\alpha}}) + w \,. \tag{15}$$

Note that the full GP is used to weight the divergence rather than the reduced approximating GP. The reduced $\widehat{GP}$ that minimizes (14) is therefore a moment projection (or M-projection) of GP onto the space of reduced size-m GPs. As shown by Koller & Friedman (2009), this type of projection punishes $\widehat{GP}$ (viewed as a normal distribution) for failing to assign probability mass to regions which are assigned non-negligible probability by GP. The reverse direction $D_{KL}^{f}(\widehat{GP}\|GP)$ corresponds to the I-projection, which instead punishes $\widehat{GP}$ for assigning probability to regions that GP considers low-probability. Interpreting this in terms of the function space defined by the GPs, we use the M-projection, which ensures that all functions plausible under GP are assigned some probability by $\widehat{GP}$, a highly desirable property as contrasted with the I-projection.

3.2 WOGPs: Weighted Sparse GP Reduction

We now address the problem of reducing the full GP to $\widehat{GP}$, which uses only m of the inducing variables. The goal of this reduction is of course to minimize the impact of removing an inducing variable; in our case, we attempt to minimize $D_{KL}^{f^*}(GP\|\widehat{GP})$.

Note that the weighting function f^* uses the full GP prediction rather than the reduced GP. This is desirable for two reasons. First, we expect the full GP to predict more accurately than the reduced GP because it is fully utilizing the information from the size-m model of the previous iteration and the new point, while the reduced GP must approximate this information. Second, using the reduced GP's predictions to weight the feature space confounds the optimization problem, since the weighting of the feature space is then malleable; for example, if f is the prediction of $\widehat{GP}$, it may be optimal to simply let $\hat{\boldsymbol{\alpha}} = 0 \Rightarrow f = 0$, but this is not helpful in approximating the full GP.

Fix $GP = \mathrm{GP}(\boldsymbol{\alpha}, C)$ and $K_{\mathcal{I}}$. Let $d^*(\hat{\boldsymbol{\alpha}}, \hat{C}) = cD_{KL}^{f^*}(GP\|\widehat{GP})$ be the weighted KL divergence (times scaling constant c) between this GP and its approximation $\widehat{GP} = \mathrm{GP}(\hat{\boldsymbol{\alpha}}, \hat{C})$. Recall our definition of $\hat{Q}$, the update to the inverse Gram matrix, from Equation 7 (and note that $\hat{Q}$ is a function only of Q).

Formally we address the following problem:

$$\begin{aligned} \underset{\hat{\boldsymbol{\alpha}}, \hat{C}}{\text{minimize}} \quad & d^*(\hat{\boldsymbol{\alpha}},\ \hat{C}) \\ \text{subject to} \quad & \hat{\boldsymbol{\alpha}}^\top \boldsymbol{e}_{m+1} = 0 \\ & \hat{C}\boldsymbol{e}_{m+1} = \mathbf{0}_{m+1} \\ & \hat{C} = \hat{C}^\top \\ & (\hat{C} + \hat{Q}) \succeq 0 \\ & |\hat{C} + Q| > 0. \end{aligned} \tag{16}$$

Here $\boldsymbol{e}_{m+1}$ is the final standard basis vector and $\mathbf{0}_{m+1}$ is the zero vector in $\mathbb{R}^{m+1}$.

Proposition 3. *For fixed $\hat{C}$, the optimization problem (16) is convex with respect to $\hat{\boldsymbol{\alpha}}$.*

Proof. Differentiating (14) with respect to the nonzero entries of $\hat{\boldsymbol{\alpha}}$ yields

$$\begin{aligned} \frac{\partial\, d^*(\hat{\boldsymbol{\alpha}}, \hat{C})}{\partial \hat{\boldsymbol{\alpha}}} &= -2[I_m \mathbf{0}_m]\hat{V}(\Gamma - I)\boldsymbol{\alpha} \\ &\quad - 2[I_m \mathbf{0}_m]\hat{V}(\boldsymbol{\alpha} - [I_m \mathbf{0}_m]^\top \hat{\boldsymbol{\alpha}}) \\ &= -2[I_m \mathbf{0}_m]\hat{V}(\Gamma\boldsymbol{\alpha} - [I_m \mathbf{0}_m]^\top \hat{\boldsymbol{\alpha}})\,, \end{aligned} \tag{17}$$

where $\hat{\boldsymbol{\alpha}}$ is now assumed to be m-dimensional, I_m represents the m-dimensional identity matrix, $\mathbf{0}_m$ is a column vector of m zeros, and $[I_m \mathbf{0}_m]$ denotes the corresponding $(m \times (m+1))$-dimensional matrix.

Now let $\boldsymbol{v}_i$ denote the ith column of $\hat{V}$, excluding the last entry, and observe that

$$\frac{\partial^2\, d^*(\hat{\boldsymbol{\alpha}}, \hat{C})}{\partial \hat{\alpha}_i \hat{\alpha}_j} = 2\frac{\partial\, (\boldsymbol{v}_i^\top \hat{\boldsymbol{\alpha}})}{\partial \hat{\alpha}_j} = 2\hat{V}_{i,j}\,. \tag{18}$$

Thus we have that the Hessian matrix of $d^*(\hat{\boldsymbol{\alpha}}, \hat{C})$ with respect to $\hat{\boldsymbol{\alpha}}$ is just twice the leading $(m \times m)$ submatrix of $\hat{V}$, which we denote $\hat{V}^{(r)}$.

Note that we can compute $\hat{V}^{(r)}$ using block matrix inversion:

$$\hat{V}^{(r)} = (\hat{C} + Q^{(r)} - \frac{1}{q^*}Q^* Q^{*\top})^{-1} = (\hat{C} + \hat{Q})^{-1}\,. \tag{19}$$

Our result follows from the constraints $(\hat{C} + \hat{Q}) \succeq 0$ and $|\hat{C} + Q| > 0$, with the additional observation that the domain of (16) in $\hat{\boldsymbol{\alpha}}$ is a convex set. □

Having established convexity, we now use Equation 17 to compute an update rule for $\hat{\boldsymbol{\alpha}}$ that minimizes the resulting weighted KL divergence. Solving (17) for zero, we have

$$\begin{aligned} &[I_m \mathbf{0}_m]\hat{V}(\Gamma\boldsymbol{\alpha} - [I_m \mathbf{0}_m]^\top \hat{\boldsymbol{\alpha}}) = \mathbf{0}_m \Rightarrow \\ &\qquad \hat{V}(\Gamma\boldsymbol{\alpha} - [I_m \mathbf{0}_m]^\top \hat{\boldsymbol{\alpha}}) = [\mathbf{0}_m^\top\ u]^\top \Rightarrow \\ &\Gamma\boldsymbol{\alpha} - [I_m \mathbf{0}_m]^\top \hat{\boldsymbol{\alpha}} = (\hat{C} + Q)[\mathbf{0}_m^\top\ u]^\top = u[\boldsymbol{Q}^{*\top} q^*]^\top\,, \end{aligned} \tag{20}$$

where in the last step we recall that the last column of $\hat{C}$ is zero. We let $\Gamma^{(r)}$ denote the first m rows of Γ and $\boldsymbol{\Gamma}^*$ the last row of Γ; observe then that

$$\boldsymbol{\Gamma}^*\boldsymbol{\alpha} = uq^* \Rightarrow u = \frac{\boldsymbol{\Gamma}^*\boldsymbol{\alpha}}{q^*}\,, \tag{21}$$

which leads us to the following solution.

Proposition 4. *The update rule for $\hat{\boldsymbol{\alpha}}$ which minimizes $D_{KL}^{f^*}(GP\|\widehat{GP})$ is given by*

$$\hat{\boldsymbol{\alpha}} = \Gamma^{(r)}\boldsymbol{\alpha} - \frac{\boldsymbol{\Gamma}^*\boldsymbol{\alpha}}{q^*}\boldsymbol{Q}^*\,. \tag{22}$$

Fixing $\hat{\boldsymbol{\alpha}}$ to be optimal in the above sense, we would then like to solve the optimization problem (16) with respect to $\hat{C}$. Unfortunately, this problem is not easily solved for local minima. Differentiating Equation 14 with respect to $\hat{C}$, we have

$$\begin{aligned} &\frac{\partial d^*(\hat{\boldsymbol{\alpha}}, \hat{C})}{\partial \hat{C}} = \frac{\partial w}{\partial \hat{C}} + \\ &- [I_m \mathbf{0}_m]\hat{V}(2\Gamma\boldsymbol{\alpha} - (\boldsymbol{\alpha} + \hat{\boldsymbol{\alpha}}))(\boldsymbol{\alpha} - \hat{\boldsymbol{\alpha}})^\top \hat{V}[I_m \mathbf{0}_m]^\top\,. \end{aligned} \tag{23}$$

Evaluating the derivative of w in the same way, we arrive at

$$\begin{aligned} \frac{\partial w}{\partial \hat{C}} &= [I_m \mathbf{0}_m]\hat{V}[I_m \mathbf{0}_m]^\top \\ &\quad - [I_m \mathbf{0}_m]\hat{V}(C + Q)\hat{V}[I_m \mathbf{0}_m]^\top \end{aligned} \tag{24}$$

However, we are not aware of a way to solve Equation 23 to minimize $D_{KL}^{f^*}(GP\|\widehat{GP})$ analytically with respect to $\hat{C}$.

Furthermore, we find that the objective $d^*(\hat{\boldsymbol{\alpha}}, \hat{C})$ is not convex with respect to $\hat{C}$: since $\log|X|$ is known to be concave, the second term of w

$$-\log|(C+Q)\hat{V}| = \log|\hat{C}+Q| - \log|C+Q| \quad (25)$$

is concave. This prevents us from using convex optimization to solve for $\hat{C}$. However, we have the following:

Proposition 5. *For fixed $\hat{\boldsymbol{\alpha}}$, the objective $d^*(\hat{\boldsymbol{\alpha}}, \hat{C})$ can be written as $d^*(\hat{\boldsymbol{\alpha}}, \hat{C}) = g_1(\hat{C}) - g_2(\hat{C})$, where g_1, g_2 are real-valued convex functions on the intersection of $\mathbb{R}^{(m+1)\times(m+1)}$ with the constraints on $\hat{C}$ in (16).*

Proof. Due to Theorem 1 of Yuille & Rangarajan (2003), we can show this by demonstrating that the Hessian of $d^*(\hat{\boldsymbol{\alpha}}, \hat{C})$ for fixed $\hat{\boldsymbol{\alpha}}$ is bounded. Since $\mathrm{Tr}[X^{-1}]$ is convex, we already have the desired result for w and will therefore compute only the Hessian of

$$d_0^*(\hat{C}) \equiv (2\Gamma\boldsymbol{\alpha} - (\boldsymbol{\alpha}+\hat{\boldsymbol{\alpha}}))^\top \hat{V}(\boldsymbol{\alpha}-\hat{\boldsymbol{\alpha}}) \,. \quad (26)$$

To do this, we first compute

$$\frac{\partial d_0^*}{\partial \hat{C}} = -[I_m \mathbf{0}_m]\hat{V}(2\Gamma\boldsymbol{\alpha} - (\boldsymbol{\alpha}+\hat{\boldsymbol{\alpha}}))(\boldsymbol{\alpha}-\hat{\boldsymbol{\alpha}})^\top \hat{V}[I_m \mathbf{0}_m]^\top \,, \quad (27)$$

where we use the matrices $[I_m \mathbf{0}_m]$ to confine the expression to the $(m \times m)$ derivative with respect to the nonzero entries of $\hat{C}$. Let $\boldsymbol{u}_1 = 2\Gamma\boldsymbol{\alpha} - (\boldsymbol{\alpha}+\hat{\boldsymbol{\alpha}})$ and $\boldsymbol{u}_2 = (\boldsymbol{\alpha}-\hat{\boldsymbol{\alpha}})$ for notational convenience. Confining this expression to a particular entry, we arrive at

$$\frac{\partial d_0^*}{\partial \hat{C}_{i,j}} = -\boldsymbol{v}_i^\top \boldsymbol{u}_1 \boldsymbol{u}_2^\top \boldsymbol{v}_j \,, \quad (28)$$

letting $\boldsymbol{v}_i$ denote the ith column of $\hat{V}$ (and recalling that $\hat{V}$ is symmetric).

Let $H_{k,l}^{i,j}$ represent an entry of the Hessian matrix of d_0^*, and compute

$$\begin{aligned} H_{k,l}^{i,j} = \frac{\partial^2 d_0^*}{\partial \hat{C}_{i,j}\partial \hat{C}_{k,l}} &= -\frac{\partial}{\partial \hat{C}_{k,l}} \boldsymbol{v}_i^\top \boldsymbol{u}_1 \boldsymbol{u}_2^\top \boldsymbol{v}_j \\ &= \boldsymbol{u}_1^\top (\hat{V}_{i,k}\boldsymbol{v}_l \boldsymbol{v}_j^\top + \hat{V}_{j,k}\boldsymbol{v}_i \boldsymbol{v}_l^\top)\boldsymbol{u}_2 \,. \end{aligned} \quad (29)$$

Evidently $H_{k,l}^{i,j}$ is bounded for all i, j, k, l, and thus we have the desired result. □

The above result allows us to employ methods of concave-convex minimization, e.g. the CCCP procedure of Yuille & Rangarajan (2003). Specifically, to minimize $g_1 - g_2$ we employ an iterative method of updating $\hat{C}$ according to the rule

$$\frac{\partial g_1}{\partial \hat{C}}(\boldsymbol{x}^{t+1}) = \frac{\partial g_2}{\partial \hat{C}}(\boldsymbol{x}^t) \,. \quad (30)$$

It is not straightforward to explicitly decompose d^* into convex terms g_1 and g_2 due to the difficulty of solving Equation 23, so we instead use the iterative method for optimization given in Yuille & Rangarajan (2003) for such cases, which finds $\boldsymbol{x}^{t+1}$ in each step by minimizing a function of $\boldsymbol{x}^{t+1}$ and $\boldsymbol{x}^t$.

In Section 4, we demonstrate that CCCP optimization of $\hat{C}$ can significantly reduce the weighted KL divergence of the online GP update. Since we are minimizing with respect to the matrix $\hat{C}$, the size of the CCCP optimization problem is $\mathcal{O}(m^2)$. For larger sparse models, this approach may therefore not be feasible if computation time is a primary concern. As runtime is extremely costly for our application, we also propose as a heuristic the update rule for $\hat{C}$ of Csató (2002), given in Equation 6, which minimizes the unweighted divergence between the distributions as well as the w-term of the weighted divergence. The analytic form for this update can provide a significant speedup over CCCP if m is large; we justify this heuristic by comparing with CCCP updating for $\hat{C}$ in Section 4.

4 Experiments

We perform two types of experiments to demonstrate the efficacy of WOGPs in Bayesian optimization. First, we provide easily visualized examples of the comparative performance of WOGPs and standard online sparse GPs on synthetic optimization problems. Second, we use data from the LCLS free-electron laser to test the optimization algorithms in a real-world setting with noisy observations.

4.1 Example Problem

First, we consider the simple problem of optimizing over a function given by $y = 20 + x - (x-1)^2(x+1)^2$, shown in Figure 1a, which achieves a maximum slightly greater than 21 at $x \approx 1.1$ and has a local maximum at $x \approx -.84$. Each type of model is allowed at most five inducing variables. In each trial, we choose five training points in $[-3, 3]$ uniformly at random and train each model on them. The Bayesian optimization procedure described in Section 2.3 is then performed for 80 iterations. Figure 1b shows the results of this experiment averaged across 80 trials. On average, both models tend to quickly find a value near the global optimum; however, while the WOGP tends to converge near this optimum, the unweighted model does not. Instead, when the unweighted model explores other areas, it essentially loses focus on the optimal area by devoting some of its limited resources to modeling the new, lower-scoring region. This can be seen in the decline on average of the unweighted model's score to roughly $y = 19$, in which it converges within the near-plateau between $x = -1$ and $x = 0$.

In the bottom of Figure 1, examples are shown of the con-

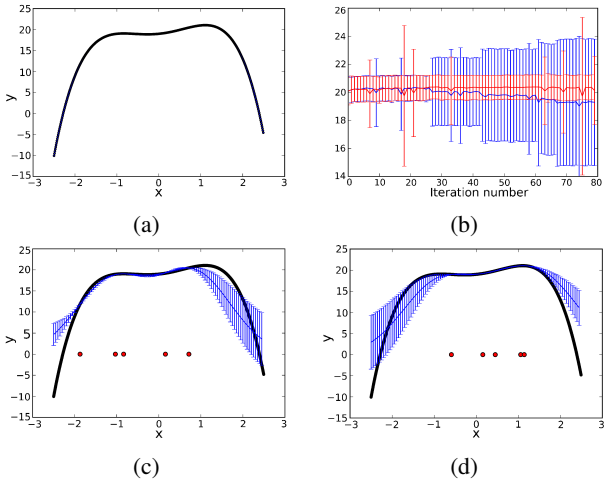


Figure 1: Simple 1-D optimization problem. (a) The objective function. (b) Average y-value explored in each iteration, for unweighted (blue) and weighted (red) models. Standard deviations are shown as error bars. (c,d) Sample unweighted and weighted (respectively) final GP models. The objective is shown in black, GP mean function with uncertainty in blue, and inducing points as red dots.

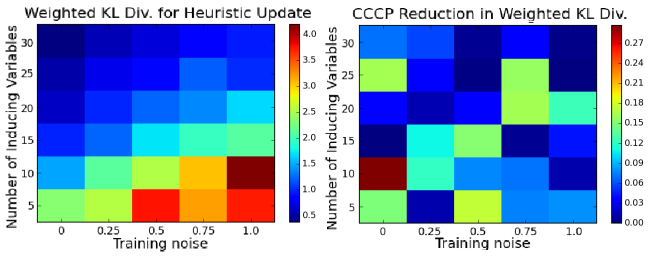


Figure 2: Results demonstrating the effectiveness of CCCP for optimization over $\hat{C}$, shown for a particular event. Each pixel shows the median value over 20 trials at the given configuration. (a) The weighted KL divergence d^* obtained using the heuristic (6) update rule for $\hat{C}$. (b) The proportional reduction in d^* from CCCP optimization over $\hat{C}$.

verged online GP models in this problem. In (1c), the unweighted model is plotted in blue with its predictive variance, while the underlying function $f(x)$ is plotted in black. The red dots underneath show the locations of the inducing variables for each model. The models were trained on the same points, and both initially explore in the region around $x = -1$. However, the unweighted model allocates one of its inducing variables to $x \approx -2$ in order to capture the curvature of $f(x)$ in the negative direction. The weighted model instead stabilizes its inducing variables around $x = -1$ and $x = 0$ during its exploration in $x < 0$ and then begins to explore in $x > 0$. This rightward exploration quickly converges around $x = 1$.

4.2 FEL Performance Optimization

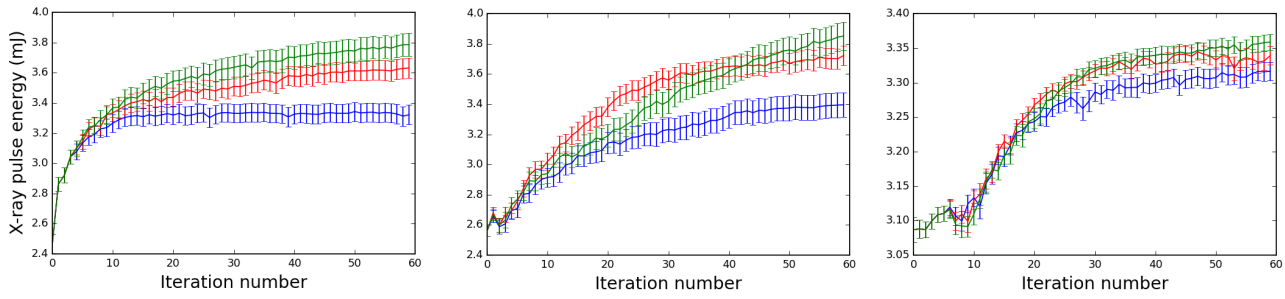
Free-electron lasers operate by accelerating electrons to nearly the speed of light, and then passing this electron beam through a series of magnetic dipoles to separate the electrons into coherent microbunches (Huang & Kim (2007)). Through this coherence, an FEL can generate x-ray pulses 10 billion fold brighter than any other x-ray source. Here, we focus on the tuning of quadrupole magnets, which are placed upstream of the FEL to manipulate the shape of the electron beam.

Currently at LCLS, quadrupole magnets are tuned by hand to optimize the beam pulse energy. The existing tuning procedure is repeated frequently due to machine configuration changes and drift over time. This extensive tuning time is problematic due to the operational cost of the beam and the heavy over-subscription of LCLS users; the FEL is used by scientists in a variety of disciplines for field-leading research, and the demand for machine time outstrips its availability by a factor of 5. Reducing the time spent tuning the machine would directly increase its availability for scientific use.

Our experiments on the FEL data thus far have used isolated optimization 'events': we define such an event as a consecutive period of time using a fixed accelerator configuration, such that the measured x-ray pulse energy is a function only of the controlled variables, i.e. quadrupole magnet settings. Under this assumption, we then train two online GP models with the same number of inducing variables, one using a WOGP, and one using a standard online sparse GP, on a noisy subset of the event data. A much larger sparse GP (with e.g. 500 inducing variables) is trained on the event data without noise. This large 'truth' model is then used in the Bayesian optimization procedure, with its predictions used as feedback for the online GP optimizers. We introduce noise for the online GP models and not the truth model to simulate the use case of online tuning, which must be done each time the beam is used due to the tendency of the machine settings to drift over time.

We first use this data to test the WOGP update rules for $\hat{C}$. Using CCCP to minimize $D_{KL}^{f^*}(GP \| \widehat{GP})$ with respect to $\hat{C}$ can be used to minimize the weighted KL divergence of the approximating GP. Alternatively, we propose as a heuristic the update given by Csató (2002), shown in (6), which is optimal for the KL divergence (5) between $\widehat{GP}$ and GP, and optimal for the w term of $D_{KL}^{f^*}(GP \| \widehat{GP})$.

Figure 2 shows results of a comparison between these update rules for $\hat{C}$. For this testing, a single representative event was chosen, and 20 optimization trials were run for a short time with various levels of training noise and numbers of inducing variables. For each trial, the value of $D_{KL}^{f^*}$ is then computed for an additional size reduction step for both



(a) (b) (c)

Figure 3: Results of optimization on FEL data, with events at different beam configurations and electron energies of 11.45, 13.2, and 14.5 GeV respectively. The plots show the average x-ray pulse energy in mJ of the region explored in each iteration by the weighted (red) and unweighted (blue) sparse optimizers, as well as by a full GP, shown in green. Error bars indicate the normalized standard deviation of the values in each iteration.

the heuristic and CCCP-computed values of $\hat{C}$. In Figure 2a, the median value of $D_{KL}^{f^*}$ obtained using the heuristic update for $\hat{C}$ across these trials is shown for each configuration. Note that as the modeling problem becomes more challenging (as noise increases and the number of inducing variables decrease), the typical divergence of the update increases. We do not show a similar plot for the CCCP updating because it is visually very similar.

In Figure 2b, the percentage decrease in $D_{KL}^{f^*}$ achieved by using CCCP updating is shown. We see that using CCCP to optimize the value of $\hat{C}$ typically leads to a 5-15% decrease in the weighted KL divergence of the update. This demonstrates that the heuristic update for $\hat{C}$ is justified in cases where the runtime of CCCP is prohibitive. These results also indicate that the CCCP updating scheme detailed here can provide nontrivial improvements to $D_{KL}^{f^*}$ over the heuristic update. Over the course of optimization the accumulated benefit from the CCCP updating may lead to substantial improvements in performance.

Next, we compare the performance of WOGPs and standard online sparse GPs in optimization over the FEL data events described above. Results from three such events, averaged over 200 trials (with different, randomly sampled initial training data), are shown in Figure 3. The results of optimization are compared in terms of final y-value and regret (which is an additive constant away from the negative sum of observations), which accounts for speed of improvement as well. We can see that in general WOGPs tend to yield better performance than the unweighted sparse GP: in the first two events (3a) and (3b), the difference in final y-values is statistically significant ($p < .001$, $p < .002$ respectively in the two-sided t-test), as is the difference in regret ($p < .05$, $p < .005$). In the final event (3c), we can see that the WOGP performs similarly to the full GP, though its improvement over the unweighted sparse GP is not statistically significant for this event.

5 Conclusions

Bayesian optimization is known to be effective for optimization in settings where the objective function is expensive to evaluate. A complex parameter space and noisy observations can slow the convergence of Bayesian optimization, however, and using a full Gaussian process model leads to poor scaling in these cases. In this paper, we introduce sparse online GPs for Bayesian optimization.

Our main contribution is a novel weighted updating scheme for sparse online GPs, which enables a trade-off during optimization between overall predictive accuracy and a specific focus on better-performing regions of parameter space. This addresses the core problem with using sparse GPs in Bayesian optimization: the limited size of the GP representation, which prevents the information from new data from being fully incorporated into the model. As a result, traditional sparse GPs perform poorly since the sparse set selection does not necessarily attempt to preserve resolution in promising areas of parameter space and may even 'blur' local optima to preserve accuracy elsewhere.

Our new weighted-update online GP, WOGP, outperforms the standard online sparse GP in optimization by preferentially updating the model to incorporate information that is more valuable to the optimization procedure. We are able to analytically evaluate the weighted KL divergence between Gaussian processes for a simple weighting function, and we demonstrate empirically that updating the sparse GP to minimize this weighted divergence significantly improves performance during Bayesian optimization. Live tests of this approach are currently underway at LCLS (McIntire et al. (2016a)).

Acknowledgements

Work is supported by Department of Energy Contract No. DE-AC02-76SF00515. SE is partially supported by NSF grant 1522054 through subcontract 72954-10597.

References

Brochu, Eric, Cora, Vlad M, and de Freitas, Nando. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. eprint arXiv:1012.2599, arXiv.org, December 2010.

Bull, Adam D. Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12:2879–2904, November 2011.

Csató, Lehel. *Gaussian Processes - Iterative Sparse Approximations*. PhD thesis, Aston University, 2002.

Csató, Lehel and Opper, Manfred. Sparse on-line Gaussian processes. *Neural Computation*, 14(3):641–668, 2002.

Emma, P. et al. First lasing and operation of an ngstrom-wavelength free-electron laser. *Nature Photonics*, 4:641 – 647, 2010.

Gal, Yarin, van der Wilk, Mark, and Rasmussen, Carl. Distributed variational inference in sparse Gaussian process regression and latent variable models. In *Advances in Neural Information Processing Systems 27*, pp. 3257–3265. Curran Associates, Inc., 2014.

Hensman, James, Fusi, Nicoló, and Lawrence, Neil D. Gaussian processes for big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, 2013.

Huang, Z. and Kim, K. J. Review of x-ray free-electron laser theory. *Phys. Rev. ST Accel. Beams*, 10, 2007.

Johnson, Matthew J. and Willsky, Alan S. Bayesian nonparametric hidden semi-markov models. *J. of Machine Learning Research*, 14(1):673–701, February 2013.

Jones, Donald R., Schonlau, Matthias, and Welch, William J. Efficient global optimization of expensive black-box functions. *J. of Global Optimization*, 13(4): 455–492, December 1998.

Koller, Daphne and Friedman, Nir. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.

Kulis, Brian and Jordan, Michael I. Revisiting k-means: New algorithms via Bayesian nonparametrics. In *Proceedings of the 29th Int'l Conference on ML*, 2012.

Lawrence, Neil, Seeger, Matthias, and Herbrich, Ralf. Fast sparse Gaussian process methods: The informative vector machine. In *Advances in Neural Information Processing Systems 15*, pp. 625–632. MIT Press, 2003.

McIntire, Mitchell, Cope, Tyler, Ermon, Stefano, and Ratner, Daniel. Bayesian optimization of FEL performance at LCLS. In *Proceedings of the 7th International Particle Accelerator Conference*, 2016a.

McIntire, Mitchell, Ratner, Daniel, and Ermon, Stefano. Sparse Gaussian processes for Bayesian optimization: Technical report. Technical report, 2016b.

Miller, Andrew et al. A Gaussian process model of quasar spectral energy distributions. In *Advances in Neural Information Processing Systems 28*, 2015.

Nickson, Thomas, Osborne, Michael A., Reece, Steven, and Roberts, Stephen. Automated machine learning using stochastic algorithm tuning. In *NIPS Workshop on Bayesian Optimization*, 2014.

Osborne, Michael A., Garnett, Roman, and Roberts, Stephen J. Gaussian processes for global optimization. In *In Learning and Intelligent Optimization*, 2009.

Quionero-Candela, Joaquin and Rasmussen, Carl Edward. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6: 1939–1959, 2005.

Ranganathan, A., Yang, M. H., and Ho, J. Online sparse Gaussian process regression and its applications. *IEEE Transactions on Image Processing*, 20(2), Feb 2011.

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

Seeger, Matthias, Williams, Christopher K. I., and Lawrence, Neil D. Fast forward selection to speed up sparse gaussian process regression. In *Workshop on AI and Statistics 9*, 2003.

Snelson, Edward and Ghahramani, Zoubin. Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, pp. 1257–1264. MIT press, 2006.

Snoek, Jasper, Larochelle, Hugo, and Adams, Ryan P. Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems 25*, pp. 2960–2968, 2012.

Snoek, Jasper et al. Scalable bayesian optimization using deep neural networks. In *Proceedings of the 32nd Int'l Conference on Machine Learning (ICML-15)*, 2015.

Tank, A., Foti, N., and Fox, E.B. Streaming variational inference for Bayesian nonparametric mixture models. In *Proc. Int'l Conference on AI and Statistics*, May 2015.

Titsias, Michalis K. Variational learning of inducing variables in sparse Gaussian processes. In *In Artificial Intelligence and Statistics 12*, pp. 567–574, 2009.

Yuille, A. L. and Rangarajan, Anand. The concave-convex procedure. *Neural Computation*, 15(4):915–936, 2003.