How Bayesian Should Bayesian Optimisation Be?

George De Ath
g.de.ath@exeter.ac.uk
Department of Computer Science
University of Exeter
Exeter, United Kingdom

Richard M. Everson
r.m.everson@exeter.ac.uk
Department of Computer Science
University of Exeter
Exeter, United Kingdom

Jonathan E. Fieldsend
j.e.fieldsend@exeter.ac.uk
Department of Computer Science
University of Exeter
Exeter, United Kingdom

ABSTRACT

Bayesian optimisation (BO) uses probabilistic surrogate models – usually Gaussian processes (GPs) – for the optimisation of expensive black-box functions. At each BO iteration, the GP hyperparameters are fit to previously-evaluated data by maximising the marginal likelihood. However, this fails to account for uncertainty in the hyperparameters themselves, leading to overconfident model predictions. This uncertainty can be accounted for by taking the Bayesian approach of marginalising out the model hyperparameters. We investigate whether a fully-Bayesian treatment of the Gaussian process hyperparameters in BO (FBBO) leads to improved optimisation performance. Since an analytic approach is intractable, we compare FBBO using three approximate inference schemes to the maximum likelihood approach, using the Expected Improvement (EI) and Upper Confidence Bound (UCB) acquisition functions paired with ARD and isotropic Matérn kernels, across 15 well-known benchmark problems for 4 observational noise settings. FBBO using EI with an ARD kernel leads to the best performance in the noise-free setting, with much less difference between combinations of BO components when the noise is increased. FBBO leads to over-exploration with UCB, but is not detrimental with EI. Therefore, we recommend that FBBO using EI with an ARD kernel as the default choice for BO.

CCS CONCEPTS

• **Theory of computation → Gaussian processes**; **Mathematical optimization**; • **Mathematics of computing → Bayesian computation**.

KEYWORDS

Bayesian optimisation, Surrogate modelling, Gaussian process, Approximate inference

ACM Reference Format:
George De Ath, Richard M. Everson, and Jonathan E. Fieldsend. 2021. How Bayesian Should Bayesian Optimisation Be?. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3449726.3463164

1 INTRODUCTION

Bayesian optimisation (BO) is a popular sequential approach for optimising costly or time-consuming black-box functions that have no derivative information or closed form [5, 53]. It is a surrogate model-based approach that employs a probabilistic model built with previous evaluations. BO comprises of two main steps, which are repeated until budget exhaustion or convergence. Firstly, a probabilistic surrogate model is constructed, which is typically a Gaussian process (GP) because of their strength in function approximation and uncertainty quantification [15, 45, 52]. Secondly, an acquisition function is optimised to select the next location to expensively evaluate. Acquisition functions combine the surrogate model's predictions and associated uncertainty to strike a balance between exploiting areas of design space with good predicted values and exploring locations with high uncertainty in their predicted values.

During the first step of a BO iteration, the surrogate model must be learned from the data. The predominant strategy in BO is to find the model hyperparameters that maximise the marginal likelihood, also known as the model evidence. This point-based estimate is known as the *maximum likelihood* (ML) estimate, or, if we also take into account some prior belief about the model's hyperparameters, the *maximum a posteriori* (MAP) estimate. These are normally found via gradient-based optimisation [1, 17, 33]. However, the marginal likelihood landscape may contain multiple optima of similar quality, as well as flat, ridge-like structures on which gradient-based optimisers may get stuck [60]; a common partial remedy is to take the best from multiple optimisations from randomly chosen initial parameters. However, in addition the ML or MAP estimate does not take into account any uncertainty that exists about the true hyperparameters, leading to naturally overconfident predictions.

Viewing this from a Bayesian perspective tells us that we need to marginalise out the hyperparameters of the model; that is, every possible hyperparameter choice should be weighted by how well its corresponding model explains the data. Then, we can use the prediction of these weighted models to take into account the uncertainty in the hyperparameters. In all but the simplest of cases, this requires calculation of an intractable integral, so in practise approximations to the integral are made using methods such as variational inference [25] and Markov Chain Monte Carlo (MCMC) [11, 19, 35]. Several works have performed a fully-Bayesian treatment of the hyperparameters in BO, and some advocate for it to become the prevailing strategy [43, 53]. Yet most works that apply a fully-Bayesian approach, e.g. [2, 21, 59] only use it because it is the *correct* thing to do, without any more justification. Therefore, in this work, we investigate whether a fully-Bayesian treatment of the surrogate model's hyperparameters leads to improved performance in BO. Specifically, we investigate the performance of BO using two acquisition functions (Expected Improvement (EI) and Upper Confidence Bound (UCB)), for two different Gaussian process kernel types (isotropic and ARD), and under four different

GECCO '21 Companion, July 10–14, 2021, Lille, France

levels of observation noise. These comparisons are made for the traditional MAP approach and three approximate inference schemes for a fully-Bayesian treatment.

Our main contributions can be summarised as follows:

(1) We provide the first empirical study of the effect on BO of a fully-Bayesian treatment of the hyperparameters.
(2) We evaluate different combinations of acquisition function, GP kernel type, and inference method on fifteen well-known test functions over a range of dimensions (2 to 10) and for a range of noise levels.
(3) We show empirically that using the EI with an ARD kernel and fully-Bayesian inference using MCMC leads to superior BO performance in the noise-free setting.
(4) We show that a fully-Bayesian treatment of the hyperparameters always leads to (even more) over-exploration with UCB. However, for EI a fully Bayesian treatment only increases exploration on higher-dimensional functions with increased observational noise level.

We begin in Section 2 by reviewing BO and how to perform fully-Bayesian BO. In Section 3 we review GPs, paying particular attention to the hyperparameter learning, and follow this up in Section 4 by reviewing the approximate inference schemes used in this work. An extensive experimental evaluation is carried out in Section 5, along with a discussion of the results. We finish with concluding remarks in Section 6.

2 BAYESIAN OPTIMISATION

Bayesian optimisation (BO), also known as Efficient Global Optimisation, is a surrogate-assisted global search strategy that sequentially samples the problem domain at locations likely to contain the global optimum. It takes into account both the predictions of a probabilistic surrogate model, typically a Gaussian process (GP), and its corresponding uncertainty [24]. It was first proposed by Kushner [29], and improved and popularised by both Močkus et al. [37] and Jones et al. [24]. See [5, 15, 52] for comprehensive reviews of BO. Without loss of generality, we can define the problem of finding a global minimum of an unknown, potentially noise-corrupted objective function $f : \mathbb{R}^d \mapsto \mathbb{R}$ as

$$\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}), \tag{1}$$

defined on a compact domain $\mathcal{X} \subset \mathbb{R}^d$. In BO it is assumed that f is black-box, i.e. it has no (known) closed form and no derivative information is available. However, we are able to access the results of its evaluations $f(\mathbf{x})$ at any location $\mathbf{x} \in \mathcal{X}$. BO is particularly effective in cases where the evaluation budget is limited due to function evaluations being expensive in terms of time and/or money. In this case we wish to optimise f in either as few function evaluations as possible, or as well as possible for a given budget T.

Algorithm 1 outlines the standard Bayesian optimisation procedure. It starts (line 1) by generating S initial sample locations $\mathbf{X} = \{\mathbf{x}_s\}_{s=1}^S$ with a space-filling algorithm, such as Latin hypercube sampling [34]. These are expensively evaluated with the function: $\mathbf{y} = \{y_s \triangleq f(\mathbf{x}_s)\}_{s=1}^S$. Then, at each BO iteration, a GP model is usually trained [53] by maximising the log marginal likelihood $\log p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta})$ with respect to the model hyperparameters $\boldsymbol{\theta}$, which are usually parameters of the GP kernel, such as a length scale, and the noise variance assumed to be corrupting the observed value of $f(\mathbf{x})$; see Section 3.3. The marginal likelihood may also be multiplied by a prior probability of the parameters, $p(\boldsymbol{\theta})$, expressing *a priori* beliefs about the parameters. Maximisation of $\log[p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta})p(\boldsymbol{\theta})]$ obtains the *maximum a posteriori* (MAP) estimate of the model hyperparameters (line 4). The choice of where to evaluate next in BO is determined by an acquisition function $\alpha(\mathbf{x} \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\theta})$, also known as an infill criterion, which balances the exploitation of good regions of the design space found thus far with the exploration regions where the predictive uncertainty is high. Acquisition functions are discussed further in the next section. The location $\mathbf{x}'$ to be expensively evaluated next is selected by maximising the acquisition function (line 5), via heuristic search, often using an evolutionary algorithm or gradient-based methods, which is possible because the acquisition function is cheap to evaluate. The selected $\mathbf{x}'$ is then expensively evaluated with f, the training data is augmented, and the process is repeated until budget exhaustion.

Algorithm 1 Standard Bayesian optimisation.

Inputs:
S : Number of initial samples
T : Budget on the number of expensive evaluations

Steps:
1: $\mathbf{X} \leftarrow \text{SpaceFillingSampling}(\mathcal{X}, S)$ ▷ Initial samples
2: $\mathbf{y} \leftarrow \{y_s \triangleq f(\mathbf{x}_s)\}_{s=1}^S$ ▷ Expensively evaluate all initial samples
3: **for** $t = S + 1 \rightarrow T$ **do**
4: $\quad\boldsymbol{\theta} \leftarrow \text{argmax}_{\boldsymbol{\theta}} \log[p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta})p(\boldsymbol{\theta})]$ ▷ MAP estimate
5: $\quad\mathbf{x}' \leftarrow \text{argmax}_{\mathbf{x}} \alpha(\mathbf{x} \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\theta})$ ▷ Maximise infill criterion
6: $\quad f' \leftarrow f(\mathbf{x}')$ ▷ Expensively evaluate $\mathbf{x}'$
7: $\quad\mathbf{X} \leftarrow \mathbf{X} \cup \{\mathbf{x}'\}$ ▷ Augment training data
8: $\quad\mathbf{y} \leftarrow \mathbf{y} \cup \{f'\}$
9: **return** $\mathcal{D}$

2.1 Acquisition Functions

Acquisition functions $\alpha(\mathbf{x} \mid \boldsymbol{\theta}) : \mathbb{R}^d \mapsto \mathbb{R}$ are measures of quality that enable us to decide which location $\mathbf{x} \in \mathcal{X}$ is the most promising, and thus where we should expend our next expensive evaluation. They are based on the predictive distribution $p(f \mid \mathbf{x}, \boldsymbol{\theta})$ of the surrogate model, where the dependence on the observed expensive evaluations $(\mathbf{X}, \mathbf{y})$ are summarised in $\boldsymbol{\theta}$. Acquisition functions usually depend on both the posterior mean prediction $\mu(\mathbf{x}) = \mathbb{E}_{p(f \mid \mathbf{x}, \boldsymbol{\theta})}[f(\mathbf{x})]$ and its associated uncertainty captured by the variance $v(\mathbf{x}) = \mathbb{V}[p(f \mid \mathbf{x}, \boldsymbol{\theta})]$. Two of the most popular acquisition functions are the Expected Improvement (EI) [24] and Upper Confidence Bound (UCB) [56]. EI measures expected positive improvement over the best function value observed so far $f^\star$:

$$\alpha_{\text{EI}}(\mathbf{x} \mid \boldsymbol{\theta}) = \mathbb{E}_{p(f \mid \mathbf{x}, \boldsymbol{\theta})}[\max(f^\star - f(\mathbf{x}), 0)], \tag{2}$$

which can be expressed analytically as [24]:

$$\alpha_{\text{EI}}(\mathbf{x} \mid \boldsymbol{\theta}) = \sqrt{v(\mathbf{x})}(s\Phi(s) + \phi(s)), \tag{3}$$

where $s = (f^\star - \mu(\mathbf{x}))/\sqrt{v(\mathbf{x})}$ is the predicted improvement normalised by its corresponding uncertainty, and $\phi(\cdot)$ and $\Phi(\cdot)$ are the Gaussian probability density and cumulative density functions respectively. EI is known to often be too exploitative, resulting in optimisation runs that can converge prematurely to a local minima

[3]. Various works have tried to curtail this behaviour by increasing the amount of exploration. Berk et al. [3], for example, try to do this by averaging over realisations drawn from surrogate model's posterior distribution instead of just using the mean prediction. Chen et al. [7], like other authors [13, 55], equate the two terms in EI to exploitation and exploration and up-weight the amount of the latter accordingly. However, as shown by De Ath et al. [9], only certain weight combinations allow for this version of EI to be monotonic in both $\mu(\cdot)$ and $v(\cdot)$; otherwise it can prefer inferior solutions, i.e. with a worse predicted value.

UCB is an optimistic strategy that is the weighted sum of the surrogate model's posterior mean prediction and its associated uncertainty:

$$\alpha_{\text{UCB}}(\mathbf{x} \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\theta}) = -\left(\mu(\mathbf{x}) - \sqrt{\beta_t v(\mathbf{x})}\right), \tag{4}$$

where $\beta_t \geq 0$ is a weight that usually depends on the number of function evaluations performed thus far and that explicitly controls the exploration-exploitation trade-off. Setting β_t is non-trivial: too small a value and UCB will get stuck in local optima; too large a value and UCB will become too exploratory, taking too many evaluations to converge. The convergence proofs for UCB of Srinivas et al. [56] rely on a particular schedule for β_t in which it increases proportional to the logarithm of t, although this scheme has been shown to be over-exploratory for many practical problems [9].

Other acquisition functions have also been proposed such as ϵ-greedy methods [9], Probability of Improvement [29], Knowledge Gradient [51], and various information-theoretic approaches [20, 21, 48, 59]. However, in this work we focus on EI and UCB due to their popularity.

2.2 Fully-Bayesian Bayesian Optimisation

Interestingly, even though fully-Bayesian approaches for Gaussian process (GP) modelling have been proposed in the literature for several decades, e.g. [18, 42], the vast majority of BO works follow Algorithm 1, i.e. they perform a MAP estimate of the GP hyperparameters at each iteration. However, there are some exceptions to this. Osborne [43] developed a Bayesian approach for global optimisation along with a novel acquisition function, and showed that their fully-Bayesian approach outperformed the standard MAP approach. The hugely influential tutorial of Snoek et al. [53] advocates a fully-Bayesian treatment of the GP hyperparameters and shows that it is sometimes superior to MAP estimation. Other works [2, 21, 59] have also performed a fully-Bayesian approach and have used it to illustrate the effectiveness of their proposed acquisition functions, rather than specifically recommending it.

In order to carry out a fully-Bayesian treatment of the hyperparameters in BO, we need to marginalise out the hyperparameters of the surrogate model, i.e. the acquisition function is averaged weighted by the posterior probability of the hyperparameters [53]:

$$\alpha(\mathbf{x} \mid \mathbf{y}, \mathbf{X}) = \int \alpha(\mathbf{x} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \mathbf{y}, \mathbf{X})\, d\boldsymbol{\theta}, \tag{5}$$

where $p(\boldsymbol{\theta} \mid \mathbf{y}, \mathbf{X})$ is the surrogate model's posterior hyperparameter distribution. The integral appearing in (5) is generally intractable, but it can be approximated via Monte Carlo integration:

$$\alpha(\mathbf{x} \mid \mathbf{y}, \mathbf{X}) \simeq \frac{1}{M} \sum_{m=1}^{M} \alpha(\mathbf{x} \mid \boldsymbol{\theta}^{(m)}), \tag{6}$$

where $\{\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(M)}\}$ are samples drawn from $p(\boldsymbol{\theta} \mid \mathbf{y}, \mathbf{X})$. These samples can be drawn via an approximate inference method such as Hamiltonian Monte Carlo or variational inference, both of which are discussed further in Section 4. We note that the MAP estimate of the hyperparameters can be regarded as approximating $p(\boldsymbol{\theta} \mid \mathbf{y}, \mathbf{X})$ by a delta function that places all the posterior probability mass at $\boldsymbol{\theta}_{MAP} = \text{argmax}_{\boldsymbol{\theta}}\, p(\boldsymbol{\theta} \mid \mathbf{y}, \mathbf{X})$. In using the estimated integrated acquisition function (6), the uncertainty in the surrogate model hyperparameters is explicitly taken into account and may therefore be expected to lead to acquisition of better locations.

3 GAUSSIAN PROCESS SURROGATES

A Gaussian process (GP) defines a prior distribution over functions, such that any finite number of function values are distributed as a multivariate Gaussian [45]. In GP regression we aim to learn a mapping from a collection of inputs $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ to their corresponding outputs $\mathbf{y} = \{y_1, \ldots, y_n\}$, where the outputs are often noisy realisations of the underlying function $f(\mathbf{x})$ we wish to model. Assuming that the observations are corrupted with additive Gaussian noise, $y = f + \epsilon$ where $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$, the observation model is defined as $p(y \mid f) = \mathcal{N}(y \mid f, \sigma_\epsilon^2)$. In GP regression we place a multivariate Gaussian prior over the latent variables $\mathbf{f} = \{f_1, \ldots, f_n\}$:

$$p(\mathbf{f} \mid \mathbf{X}, \boldsymbol{\theta}) \sim \mathcal{N}(\mathbf{0}, \mathbf{K}), \tag{7}$$

with a covariance $\mathbf{K} \in \mathbb{R}^{n \times n}$ with $K_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j \mid \boldsymbol{\theta})$ and associated hyperparameters $\boldsymbol{\theta}$. Here, $\kappa(\cdot, \cdot \mid \boldsymbol{\theta})$ is a positive semidefinite covariance function modelling the covariance between any pair of locations. For notational simplicity, and without loss of generality, we take the mean function of the GP to be zero; [10] discusses BO performance with different choices of mean function.

3.1 Covariance Functions

The covariance function $\kappa(\cdot, \cdot \mid \boldsymbol{\theta})$, also known as a kernel (function), encodes prior beliefs about the characteristics of the modelled function, such as its smoothness. Kernels are typically stationary, meaning that they are a function of the distance between the two inputs, i.e. $r = |\mathbf{x} - \mathbf{x}'|$. One of the most frequently used kernels is the squared exponential (SE) kernel:

$$\kappa_{SE}(\mathbf{x}, \mathbf{x}' \mid \boldsymbol{\theta}) = \sigma_o^2 \exp\left(-\frac{r^2}{\ell^2}\right), \tag{8}$$

with parameters ℓ and signal variance σ_o^2 defining its characteristic length-scale and output-scale respectively. Using a SE kernel for each input dimension with a separate length-scale ℓ_i results in the SE automatic relevance determination (ARD) kernel:

$$\kappa_{SE}(\mathbf{x}, \mathbf{x}' \mid \boldsymbol{\theta}) = \sigma_o^2 \exp\left(-\sum_{i=1}^{d} \frac{r_i^2}{\ell_i^2}\right), \tag{9}$$

where $r_i = |x_i - x_i'|$. Allowing separate length scales for each dimension allows irrelevant dimensions to be suppressed by placing priors over them which favour large ℓ_i [32, 39].

It has been argued [53, 57] that the SE kernel has too strong smoothness assumptions for the realistic modelling of physical processes. Popular alternatives include the Matérn family of covariance functions [57]. Here we use the Matérn kernel with $\nu = 5/2$, as recommended by Snoek et al. [53]:

$$\kappa_{Matern}(\mathbf{x}, \mathbf{x}' \mid \boldsymbol{\theta}) = \sigma_o^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu} r\right)^{\nu} K_\nu \left(\sqrt{2\nu} r\right), \tag{10}$$

where K_ν is a modified Bessel function and $r^2 = \sum_{i=1}^{d} (x_i - x_i')^2 / \ell_i^2$ is the squared distance between $\mathbf{x}$ and $\mathbf{x}'$ scaled by the length-scales ℓ_i which again allows ARD suppression of irrelevant dimensions. Further information on GP kernels can be found in [12, 45].

3.2 Making Prediction with GPs

Given some noisy observations $\mathbf{y}$ at locations $\mathbf{X}$ and a covariance function $\kappa(\cdot, \cdot \mid \boldsymbol{\theta})$, predictions about the underlying function f can be made using the GP model. Assuming a Gaussian noise model, the joint distribution of the observed training values $(\mathbf{X}, \mathbf{y})$ function values at a test location $(\mathbf{x}', f')$ is

$$\begin{bmatrix} \mathbf{y} \\ f' \end{bmatrix} \Bigg| \mathbf{X}, \boldsymbol{\theta}, \sigma_\epsilon \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \boldsymbol{\kappa}(\mathbf{X}, \mathbf{X} \mid \boldsymbol{\theta}) + \sigma_\epsilon^2 \mathbf{I} & \boldsymbol{\kappa}(\mathbf{X}, \mathbf{x}' \mid \boldsymbol{\theta}) \\ \boldsymbol{\kappa}(\mathbf{X}, \mathbf{x}' \mid \boldsymbol{\theta})^\top & \kappa(\mathbf{x}', \mathbf{x}' \mid \boldsymbol{\theta}) \end{bmatrix}\right), \tag{11}$$

where the elements of the n-dimensional vector $\boldsymbol{\kappa}(\mathbf{X}, \mathbf{x}' \mid \boldsymbol{\theta})$ are $[\boldsymbol{\kappa}(\mathbf{X}, \mathbf{x}' \mid \boldsymbol{\theta})]_i = \kappa(\mathbf{x}_i, \mathbf{x}' \mid \boldsymbol{\theta})$. Hereafter, the observation noise σ_ϵ is incorporated into $\boldsymbol{\theta}$ so that $\boldsymbol{\theta}$ represents both the kernel hyperparameters and the likelihood noise. We also drop the explicit dependence of the kernel on $\boldsymbol{\theta}$ for ease of exposition.

Conditioning the joint distribution (11) on the observations $\mathbf{y}$ yields the predictive distribution of $f' \mid \mathbf{x}', \mathbf{y}, \mathbf{X}, \boldsymbol{\theta}$ as a Gaussian distribution:

$$p(f' \mid \mathbf{x}', \mathbf{y}, \mathbf{X}, \boldsymbol{\theta}) \sim \mathcal{N}(\mu(\mathbf{x}'), v(\mathbf{x}')), \tag{12}$$

with mean and variance

$$\mu(\mathbf{x}) = \boldsymbol{\kappa}(\mathbf{X}, \mathbf{x})^\top (\boldsymbol{\kappa}(\mathbf{X}, \mathbf{X}) + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y} \tag{13}$$

$$v(\mathbf{x}) = \kappa(\mathbf{x}, \mathbf{x}) - \boldsymbol{\kappa}(\mathbf{X}, \mathbf{x})^\top (\boldsymbol{\kappa}(\mathbf{X}, \mathbf{X}) + \sigma_\epsilon^2 \mathbf{I})^{-1} \boldsymbol{\kappa}(\mathbf{X}, \mathbf{x}). \tag{14}$$

However, before the GP can be used to make predictions the kernel hyperparameters and the observational noise must be inferred.

3.3 Learning Hyperparameters

One of the most useful properties of GPs is that we are able to calculate the marginal likelihood $p(\mathbf{y}, \mathbf{X} \mid \boldsymbol{\theta}) \propto p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta})$, otherwise known as the model evidence [31], of data $(\mathbf{X}, \mathbf{y})$ for a particular model defined by a set of hyperparameters. The marginal likelihood may be found by direct integration of the product of the likelihood function of the latent variables $\mathbf{f}$ and the GP prior. For numerical reasons, the log marginal likelihood is normally used instead:

$$\begin{aligned} \log p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta}) = &-\frac{1}{2} \mathbf{y}^\top (\boldsymbol{\kappa}(\mathbf{X}, \mathbf{X}) + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y} \\ &- \frac{1}{2} \log \left| (\boldsymbol{\kappa}(\mathbf{X}, \mathbf{X}) + \sigma_\epsilon^2 \mathbf{I})^{-1} \right| - \frac{n}{2} \log 2\pi. \end{aligned} \tag{15}$$

The first term in (15) is the data-fit term, i.e. how well the model predicts the observed targets $\mathbf{y}$, while the second corresponds to a complexity penality that depends only on the magnitude of the covariance function, and the third is a normalisation constant.

The predominant strategy in BO for inferring the hyperparameters is to find the value of $\boldsymbol{\theta}$ that maximises the log marginal likelihood (15). In doing so, we find the *maximum likelihood* (ML) parameters $\boldsymbol{\theta}_{ML} = \operatorname{argmax}_{\boldsymbol{\theta}} \log p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta})$. Predictions can then be made by using $\boldsymbol{\theta}_{ML}$ in (13) and (14).

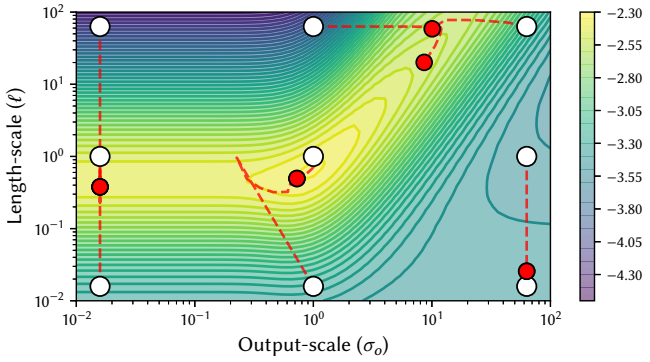

Figure 1: Multi-start gradient-based optimisation of the log marginal likelihood (15). Starting locations are shown in white, with optimisation paths shown with red dashed lines and ending locations shown as red circles. Note how only a few runs successfully find the likelihood peak at $(1, 1)$.

However, we often have some prior beliefs about the hyperparameters. For example, if our observations were standardised to have zero mean and unit variance, then it would be extremely unlikely that the likelihood noise would be larger than one because then the model would predict the majority of the observations as noise. These beliefs maybe encoded in a prior distribution $p(\boldsymbol{\theta})$, and can be taken into account in the likelihood evaluation by multiplying the marginal likelihood by the prior distribution:

$$p(\boldsymbol{\theta} \mid \mathbf{y}, \mathbf{X}) \propto p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta}) p(\boldsymbol{\theta}). \tag{16}$$

In practice, the logarithm of $p(\boldsymbol{\theta})$ is added to (15) to arrive at the log posterior distribution, which is then maximised to find the *maximum a posteriori* (MAP) parameters $\boldsymbol{\theta}_{MAP}$. Note here that if we *a priori* believe that all hyperparameter configurations are equally likely, then $\boldsymbol{\theta}_{MAP} \equiv \boldsymbol{\theta}_{ML}$.

The optimal hyperparameters, $\boldsymbol{\theta}_{MAP}$ or $\boldsymbol{\theta}_{ML}$, are normally found by gradient-based optimisation [1, 17, 33] with multiple restarts, using e.g. L-BFGS-B [6]. These restarts are needed because the landscape may contain multiple local maxima [45]. However, the gradient-based optimisation can be sensitive to the starting locations because hyperparameters that are weakly identified can give rise to flat, ridge-like structures [60].

Figure 1 shows nine gradient-based optimisation runs on the log marginal likelihood (15) as a function of the length-scale ℓ and output-scale σ_o. Training data was generated by drawing a realisation from a GP with an Matérn 5/2 kernel having hyperparameters $(\ell, \sigma_o, \sigma_\epsilon) = (1, 1, 0.1)$. The figure shows the paths taken and final positions (red circles) by gradient based optimisations starting at the locations shown as white circles. Note that σ_ϵ was fixed at $\sigma_\epsilon = 0.1$. Only two of the optimisation runs ended up at a location close to the maximum, with the other runs failing to get close. The three runs that started on the left-hand side of the figure illustrate the flat, ridge-like structures that can occur – in all three cases the optimiser got stuck in a suboptimal region of almost zero gradient.

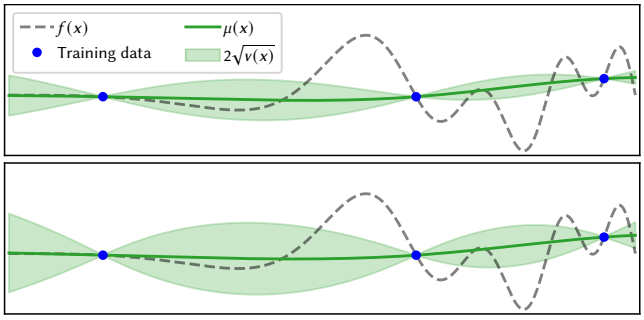

Figure 2: Posterior predictive distributions for two GPs, one fitted with the MAP hyperparameter estimate (upper), and one with a fully-Bayesian treatment of the hyperparameters via Monte Carlo estimation (lower). Note the increased amount of uncertainty in the lower GP as the uncertainty in the hyperparameters has been taken into account.

As illustrated, point-based estimates of the hyperparameters are subject to high variability, because they may become stuck in local minima or on vast plateaux, and do not take into account the uncertainty in the hyperparameters themselves. Multiple restarts are often sufficient to avoid local minima and plateaux, but accounting for hyperparameter uncertainty requires a fully-Bayesian formulation. Given a prior $p(\boldsymbol{\theta})$ expressing beliefs about the hyperparameters, the posterior distribution is given by:

$$p(\boldsymbol{\theta} \mid \mathbf{y}, \mathbf{X}) = \frac{p(\mathbf{y}, \mathbf{X} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\mathbf{y}, \mathbf{X})}. \tag{17}$$

As noted above, with the posterior distribution on hand, the acquisition function weighted by the posterior probability of the hyperparameters can be optimised to find the next location to be expensively evaluated (cf. (5)). However, it is also often of interest to find the posterior predictive distribution of the function at $\mathbf{x}$:

$$p(f \mid \mathbf{x}, \mathbf{y}, \mathbf{X}) = \int p(f \mid \mathbf{x}, \mathbf{y}, \mathbf{X}, \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \mathbf{y}, \mathbf{X}) d\boldsymbol{\theta}. \tag{18}$$

The left-hand term of the integrand is a Gaussian (12) and the right-hand term is the hyperparameter posterior. As illustrated in Figure 2, performing a fully-Bayesian treatment of the hyperparameters in GP regression leads to increased uncertainty in the model predictions. This is because we also account for the uncertainty in the hyperparameters, rather than assuming that the MAP estimate is correct. Similarly, a fully-Bayesian treatment of Bayesian optimisation accounts for the uncertainty in the hyperparameters when optimising the acquisition function.

However, as is frequently the case with Bayesian inference, the integral necessary to evaluate $p(\mathbf{y}, \mathbf{X})$ is intractable, and we therefore turn to approximate methods.

4 APPROXIMATE INFERENCE

Practical fully-Bayesian optimisation requires averaging with respect to the GP hyperparameter distribution. In particular, full account of the model uncertainty is taken by using the averaged acquisition function (5). Since this cannot be done analytically, the averages can be approximated either by simulating draws directly from the posterior or to approximate the posterior and draw samples from the approximation; e.g. (6). The former is primarily carried out by Markov Chain Monte Carlo (MCMC) [19, 35] methods, and the latter by many density approximation methods such as Laplace's method, Expectation Propagation [36], and variational inference [25, 58]. Compared to MCMC, density approximation methods tend to be much faster and easier to scale to larger amounts of data [4], but lack the guarantees of producing asymptotically exact samples from the target density [47]. Here, we consider Hamiltonian Monte Carlo and variational inference as representative methods from the two main approximate inference styles.

Monte Carlo estimation in the context of GP modelling has been carried out by a number of authors [14, 30, 38, 40, 44]. We particularly note Lalchand and Rasmussen [30], who compared the quality of GP models using Hamiltonian Monte Carlo and variational inference with ML estimates.

4.1 Hamiltonian Monte Carlo

Hamiltonian or Hybrid Monte Carlo [11] (HMC), is an MCMC method which constructs a Markov chain exploring the target probability density, ensuring that locations are visited proportional to their probability. HMC is a promising method for approximate inference in cases where gradient information is available [44]. It is preferred to the more traditional Metropolis-Hastings (MH) [19] algorithm because it is able to rapidly explore regions of high probability by introducing auxiliary momentum variables that are associated with the parameters of the target density. New proposals as to where to next move are generated by simulating Hamiltonian dynamics for a predefined number of steps L, with the dynamics themselves simulated using a leap-frog symplectic integrator. At each iteration of the algorithm, the gradients of the log marginal likelihood of the target density are required for each of the L steps. In the context of approximate inference for GP regression, this results in the need to invert the kernel matrix $\boldsymbol{\kappa}(\mathbf{X}, \mathbf{X} \mid \boldsymbol{\theta})$ L times, hence making HMC a costly procedure to carry out. See [40, 41] for tutorials on HMC and its application to GPs.

Here, we use a self-tuning variant of HMC known as the No-U-Turn Sampler (NUTS) [22], in which both the path length L and integration step size are automatically tuned. This avoids the additional overhead in manually tuning both parameters each time inference is performed with different data and is thus of particular benefit in BO where the quantity of data grows at each iteration.

4.2 Variational Inference

Variational Inference (VI), tries to find a most similar approximate density to a given probability density function [4]. More formally, we first assume a parametrisable family $\boldsymbol{Q}$ of approximate densities that captures features of the target density $p(\boldsymbol{\theta})$. Then, we try to find $q^*(\boldsymbol{\theta}) \in \boldsymbol{Q}$ that minimises the Kullback-Leibler (KL) divergence to the target density, $p(\boldsymbol{\theta} \mid \mathcal{D})$ for data $\mathcal{D}$:

$$q^*(\boldsymbol{\theta}) = \underset{q(\boldsymbol{\theta}) \in \boldsymbol{Q}}{\operatorname{argmin}} \operatorname{KL}(q(\boldsymbol{\theta}) \,||\, p(\boldsymbol{\theta} \mid \mathcal{D})). \tag{19}$$

Finally, we can approximate the posterior with $q^*(\boldsymbol{\theta})$ and draw arbitrarily many samples from it to perform approximate inference.

There any many choices for the family Q of approximation, although one of the key ideas behind VI is to choose Q to be expressive enough to model the target density well, but simple enough to allow for efficient optimisation [4]. A few examples include mean-field (MFVI) and full-rank (FRVI) Gaussian approximations, as well as the more recent, expressive, and computationally expensive normalising flows [46].

In this work, we focus on MFVI and FRVI because these have been empirically shown to be suitable for approximating the GP hyperparameter posterior distribution [30], while still being relatively cheap computationally. The mean-field approximation defines $q(\boldsymbol{\theta})$ as a product of independent densities. Here we choose $q(\boldsymbol{\theta})$ to be a product of normal densities, one for each hyperparameter:

$$q(\boldsymbol{\theta}; \boldsymbol{\xi}_{mf}) = \mathcal{N}\big(\boldsymbol{\theta} \mid \mathbf{m}, \operatorname{diag}(\boldsymbol{\sigma}^2)\big) = \prod_{j=1}^{J} \mathcal{N}(\theta_j \mid m_j, \sigma_j^2), \tag{20}$$

where $\boldsymbol{\xi}_{mf} = (m_1, \ldots, m_J, \omega_1, \ldots, \omega_J) \in \mathbb{R}^{2J}$ is a vector of unconstrained variational parameters, $\log(\sigma_j^2) = \omega_j$ and $J = |\boldsymbol{\theta}|$ is the number of parameters in the target distribution. Of course, because this is a diagonal approximation to the true posterior, the mean-field approximation will not capture correlation between parameters. In contrast to this, the full-rank Gaussian approximation allows for cross-covariance terms to be modelled explicitly. To ensure that the learned covariance $\boldsymbol{\Sigma}$ is always positive semidefinite, the covariance matrix is written in terms of its Cholesky factorisation, $\boldsymbol{\Sigma} = \mathbf{L}\mathbf{L}^\top$. This results in an approximating distribution defined as

$$q(\boldsymbol{\theta}; \boldsymbol{\xi}_{fr}) = \mathcal{N}(\boldsymbol{\theta} \mid \mathbf{m}, \mathbf{L}\mathbf{L}^\top), \tag{21}$$

where $\boldsymbol{\xi}_{fr} = (\mathbf{m}, \mathbf{L}) \in \mathbb{R}^{J+J(J+1)/2}$ is the vector of unconstrained variational parameters.

VI seeks to minimise the KL divergence between the target density $p(\boldsymbol{\theta} \mid \mathcal{D})$, and the approximating distribution $q(\boldsymbol{\theta})$. It can be shown (e.g. [4]) that the KL divergence is minimised by maximising the variational free energy or evidence lower bound (ELBO),

$$\text{ELBO}(q) = \mathbb{E}_q[\log p(\boldsymbol{\theta}, \mathcal{D})] - \mathbb{E}_q[\log q(\boldsymbol{\theta})]. \tag{22}$$

Both terms in ELBO are readily computed, although the model-specific computations required to find the gradient with respect to the parameters $\boldsymbol{\xi}$ for gradient-based optimisation may be cumbersome. More recently, however, the ubiquity of automatic derivative calculations have led to the development of scalable VI algorithms, namely automatic differentiation VI (ADVI) [28]. ADVI first transforms the inference problem into one with unconstrained real-values by, for example, taking the logarithm of parameters that need to be kept strictly positive. It then recasts the gradient of the ELBO as an expectation of q instead, allowing for the use of Monte Carlo methods to perform gradient approximation where needed. Lastly, it reparameterises other gradients in terms of a standard Gaussian, again allowing for the expectation of a gradient to be calculated (easier) instead of the gradient of an expectation (hard to impossible); this is known as the reparameterisation trick [26].

5 EXPERIMENTAL EVALUATION

We now compare the performance of using a fully-Bayesian treatment of the hyperparameters in BO (denoted FBBO), with the standard approach of using the MAP solution (denoted MAP). The EI, and UCB acquisition functions (Section 2.1) are compared for both MAP and FBBO using direct MCMC, MFVI and FRVI across the 15 well-known benchmark functions listed in Table 1[1]. We investigate several scenarios to compare FBBO with MAP. Specifically, we consider the noise-free case, where the function of interest is assumed to not be significantly corrupted by noise; for this we fix $\sigma_\epsilon = 10^{-4}$. We also look at the case where the function is corrupted by additive Gaussian noise for three different levels of noise. Note that, because the functions in Table 1 are noise-free, we add stochastically generated noise to their evaluations to simulate a noisy setting. We modify the functions to have Gaussian additive noise with a standard deviation that is proportional to the range $|f|$ of possible function values. Concretely, we estimate $|f|$ by evaluating 10^6 Latin hypercube samples (LHS) across the function domain, finding the maximum f_{max} of these samples, and calculating $|f| = f_{max} - f_{min}$, where f_{min} is the known minimum of the function. Therefore, for a given σ_n, each function evaluation is a draw from a Gaussian distribution: $y = \mathcal{N}(f(\mathbf{x}), (\sigma_n|f|)^2) = f(\mathbf{x}) + \mathcal{N}(0, (\sigma_n|f|)^2)$. We investigate three noise levels, $\sigma_n \in \{0.05, 0.1, 0.2\}$. We also compare FBBO and MAP using ARD and isotropic kernels, i.e. using one length-scale for each dimension or using one length-scale for all dimensions of the problem. One might expect, given the reduction in the number of hyperparameters in the isotropic case, that the hyperparameter posterior distribution would be less complex and therefore less likely to benefit from a fully-Bayesian treatment. Overall, this results in 8 sets of experiments across the test functions: the noise-free setting and three different amounts of noise, repeated for the ARD and isotropic kernels.

Table 1: Benchmark functions and dimensionality (d).

Name	d	Name	d
Branin	2	Ackley	5, 10
Eggholder	2	Michalewicz	5, 10
GoldsteinPrice	2	StyblinskiTang	5, 7, 10
SixHumpCamel	2	Hartmann6	6
Hartmann3	3	Rosenbrock	7, 10

A zero-mean GP with a Matérn 5/2 kernel (10) was used for all experiments. At each BO iteration, input variables were rescaled to reside in $[0, 1]^d$, and observations were rescaled to have zero-mean and unit variance prior to GP inference. Relatively uninformative priors were used, based on BoTorch recommendations [1], for the three types of hyperparameters: $\ell \sim \text{Ga}(3, 6)$, $\sigma_o \sim \text{Ga}(2, 0.15)$, and $\sigma_\epsilon \sim \text{Ga}(1.1, 0.05)$, where $\text{Ga}(a, b)$ is a Gamma distribution with concentration and rate parameters a and b respectively. Models were initially trained on $S = 2d$ observations generated by maximin LHS and then optimised for a further $200 - S$ function evaluations. The trade-off β_t between exploitation and exploration in UCB was set using Theorem 1 in [56], which implies that β_t increases logarithmically with t. Each optimisation run was repeated 51 times from a different set of LHS samples, and the sets of initial locations were used for all methods to enable statistical comparison. An odd number (51) of repeats were chosen to allow for the calculation of the median fitness value without the need for rounding. For MAP

[1] Function formulae can be found at: http://www.sfu.ca/~ssurjano/optimization.html.

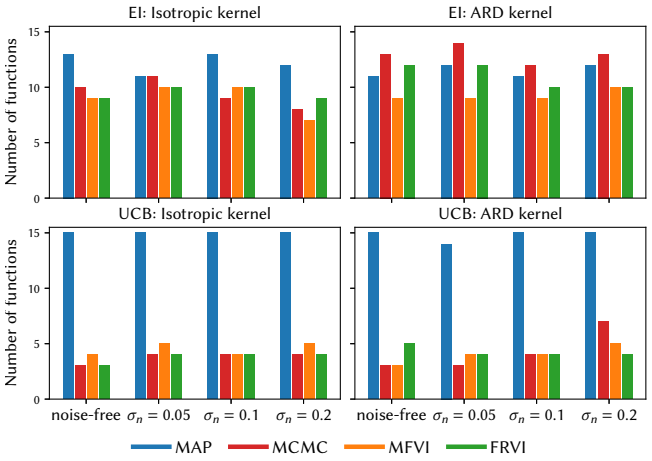

Figure 3: EI and UCB optimisation summary. Bar heights correspond to the number of times that an inference method is the best or statistically equivalent to the best method across the 15 functions, for the 4 noise scenarios and 2 kernel types.

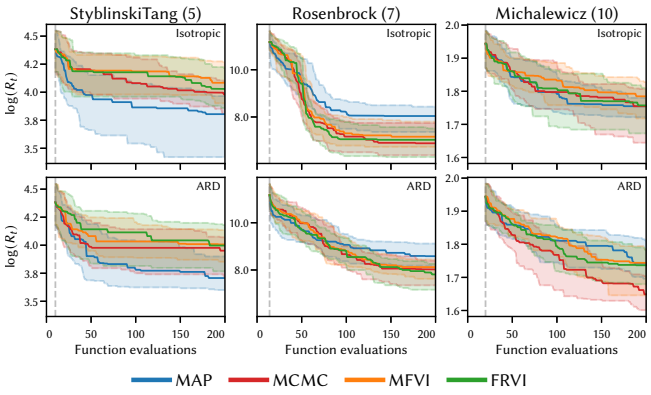

Figure 4: Illustrative convergence plots for 3 benchmark problems for EI using an isotropic (upper) and ARD kernel (lower). Each plot shows the median log simple regret, with shading representing the interquartile range over 51 runs.

estimation, the GP hyperparameters were estimated by maximising $\log[p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta})p(\boldsymbol{\theta})]$ using a multi-start strategy with L-BFGS-B [6] and 10 restarts. In all the approximate fully-Bayesian methods the acquisition functions were averaged over $M = 256$ posterior samples (6). HMC/MCMC sampling inference (Section 4.1) was carried out using PyMC3 [50], with 4 chains, discarding the first 2048 samples as burn-in and taking every 50th sample. We note that this is significantly more than previous works, e.g. [21, 59] only performed inference every 10 BO iterations, using MCMC to draw $M = 200$ samples, discarding only the first 50 of them, and taking every 3rd sample. In the non-MCMC BO iterations, the authors reused the latest set of samples. When carrying out variational inference (Section 4.2), optimisation of the ELBO (22) was undertaken for a maximum of 40000 steps or until convergence. Finally, GP models were built using GPyTorch [16] and the resulting acquisition functions were optimised using BoTorch [1]. Full experimental results are available in the supplementary material, and all code, initial locations and optimisation runs are available online[2].

Here, we report performance in terms of the logarithm of the simple regret R_t, which is the difference between the true minimum value f_{min} and the best seen function evaluation up to iteration t: $\log(R_t) = \log(\min\{f_1, \ldots, f_t\} - f_{min})$.

5.1 Results and Discussion

Each combination of acquisition function (EI and UCB) and kernel (isotropic and ARD) were evaluated on the test problems shown in Table 1, for the four different noise levels ($\sigma_n \in \{0, 0.05, 0.1, 0.2\}$).

Figure 3 shows, for the EI and UCB acquisition functions and noise levels, the number of times BO with each inference method was the best-performing according to a one-sided, paired Wilcoxon signed-rank test [27] with Holm-bonferroni [23] correction ($p \geq 0.05$). As can be seen from the plot, for EI with the isotropic kernel, MAP outperforms the other inference methods. Given that there are only three hyperparameters $\boldsymbol{\theta} = (\ell, \sigma_\epsilon, \sigma_o)$ regardless of the problem dimensionality d, this is not wholly surprising. It is likely that the hyperparameter posterior distribution (18) quickly becomes sharply unimodal, particularly given the lack of freedom in the parameters due the single length-scale. This matches the observations of Rasmussen and Williams [45], who note that as the amount of data increases, as it does in BO, that one often finds a local optimum that is orders of magnitude more probable than any other, and, indeed, that it becomes more sharply peaked.

Conversely, for EI using an ARD kernel MCMC outperforms MAP. With ARD there are $2 + d$ hyperparameters, and thus much more freedom to allow for different, but equally likely, explanations for the data. This corresponds to a much more diffuse, and potentially more multimodal, posterior density. We note that this does not conflict with Rasmussen and Williams's observation because the curse of dimensionality means that much more data would be required for one mode to become dominant and sharply-peaked. Figure 4 shows some illustrative convergence plots using EI with an isotropic (upper) and ARD kernel (lower).

Figure 3 also shows that for EI, BO with MCMC inference (top row, red bars) was often statistically significantly better than both MFVI and FRVI (orange and green bars respectively) on the majority of test problems, kernels, and noise levels. In fact, when MCMC was equal to or better than MAP, the ordering of the inference methods was always MCMC $\geq$ MAP $>$ FRVI $>$ MFVI. This suggests that the posterior hyperparameter distribution is not well approximated by either VI method. Figure 1 lends weight to this interpretation because the posterior distribution was boomerang-shaped, which cannot be well approximated by either the mean field approximation, in which each parameter is independent, or the full-rank approximation, which is constrained to be Gaussian. Note that the greater flexibility of FRVI allows it to perform better than MFVI. Due to the superiority of MCMC over the variational methods, we compare MAP with MCMC for the remainder of this work.

Figure 5 summarises the performance of MAP vs. MCMC for each combination of acquisition function and kernel (columns),

[2]https://github.com/georgedeath/how-bayesian-should-BO-be

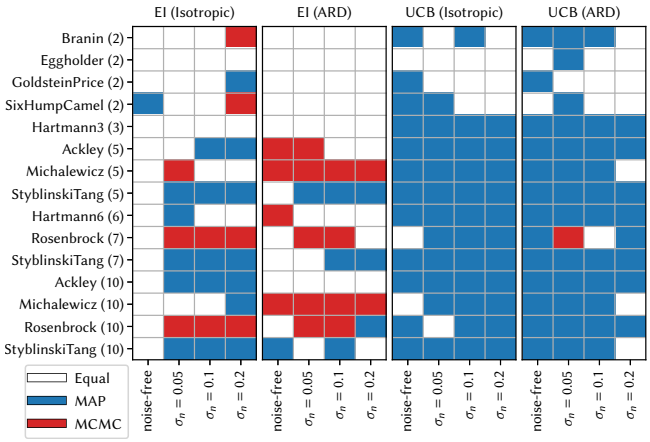

Figure 5: MAP vs. MCMC summary. The colour of each cell indicates whether both inference methods were statistically indistinguishable from one another (white), MAP outperformed MCMC (blue) or MCMC outperformed MAP (red).

and test problem (rows). As can be seen from the figure, when using UCB, MCMC-based inference is almost always worse. We suspect that the increased uncertainty of the full posterior (MCMC) leads to increased exploration with an already exploratory acquisition function, which ultimately hampers convergence. Conversely, using EI with an isotropic kernel, neither MAP nor MCMC is consistently superior. MCMC is almost always better with ARD when using EI, probably due to the better representation of the posterior hyperparameter distribution, which allows better prediction and exploration of the function being optimised.

In order to investigate whether UCB and EI are really more exploratory when using MCMC compared with MAP, we calculated the Euclidean distance between consecutively evaluated locations in each optimisation run, and found that this is indeed the case – for plots of the distances see the supplementary material. With UCB, the distances between consecutive locations are the smallest for MAP and increase substantially for the other inference methods. Distances between consecutively evaluated locations for EI, however, did not follow a similar trend: In the noise-free setting, there was practically no difference in the behaviour of EI with MAP or MCMC. Conversely, as the noise level increases, the approximate inference-based methods become more exploratory than the MAP estimates, particularly on the higher-dimensional problems. This indicates that the MAP estimates are sufficient for the lower-dimensional problems where the hyperparameter posterior distribution quickly becomes sharply peaked. EI is known to be too exploitative, and, as discussed in Section 2.1, several works have tried *ad hoc* schemes to increase the amount of exploration it performs. Therefore, we argue that if the EI acquisition function is being used, then taking into account the hyperparameter uncertainty via a fully-Bayesian treatment of them is essential, particularly in higher dimensions. Doing so in a principled, Bayesian way leads to the incorporation of an increased amount of exploration when needed, and does so without recourse to *ad hoc* rules.

Finally, we compare the different combinations of acquisition function, inference method, and kernel type to give some general advice to practitioners as to which of these BO components should be used. As shown in the supplementary material, EI with a fully-Bayesian treatment of the hyperparameters and an ARD kernel leads to the best performance in the noise-free setting, with EI vastly outperforming UCB for all combinations of components. However, as the noise level increases, the picture becomes less clear. UCB improves its performance in comparison to EI and the methods are roughly equal for the different noise levels (supplementary material, Figure 3). EI's deteriorating performance at higher noise levels may indicate that the surrogate model is poorer at representing the noisy function, leading to insufficient exploration. This is in contrast to the more exploratory UCB, which over-explores in the noise-free case [3]. It would be interesting to investigate BO strategies for optimisation in the presence of noise. It is clear that a fully-Bayesian treatment of the hyperparameters should be avoided with the UCB acquisition function because the increased level of exploration led to poor performance for both kernel types across all levels of noise.

6 CONCLUSION

We investigated how a fully-Bayesian treatment of the Gaussian process hyperparameters affects optimisation performance. With noise-free evaluations, EI with a MCMC inference and an ARD kernel was the best combination of evaluated BO components. However, as the observational noise level increased, there was less to differentiate between the components. In the case of EI, MCMC was found to be more effective with an ARD kernel than an isotropic one. We attribute this to additional flexibility to model the complex and possibly multi-modal hyperparameter posterior that is afforded by a kernel that treats different dimensions with different length scales. MCMC is generally superior to variational methods (MFVI and FRVI) because the marginal posterior is often sufficiently complex that it is poorly modelled with the (necessarily unimodal) mean field or full-rank Gaussian approximations.

We do not recommend the fully-Bayesian approach with UCB because the additional hyperparameter uncertainty leads to even greater exploration with the already over-exploratory UCB acquisition function. However, this is not the case for EI and we, therefore, recommend the fully-Bayesian treatment of the hyperparameters in BO using MCMC because it allows for a principled way to increase exploration without any *ad hoc* enhancements to EI.

Other important future directions also focus on the modelling ability of the surrogate. Particular aspects are the non-stationarity induced as the optimiser converges [e.g. 54] and improving the modelling of functions with degenerate features, such as discontinuities, using deep GPs [8, 49].

ACKNOWLEDGMENTS

This work was supported by Innovate UK [grant numbers 104400 and 105874]. The authors would like to acknowledge the use of the University of Exeter High-Performance Computing (HPC) facility.

REFERENCES

[1] Maximilian Balandat, Brian Karrer, Daniel R. Jiang, Samuel Daulton, Benjamin Letham, Andrew Gordon Wilson, and Eytan Bakshy. 2020. BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization. In *Advances in Neural*

Information Processing Systems. Curran Associates, Inc., New York, 21524–21538.
[2] Romain Benassi, Julien Bect, and Emmanuel Vazquez. 2011. Robust Gaussian Process-Based Global Optimization Using a Fully Bayesian Expected Improvement Criterion. In *Learning and Intelligent Optimization (Lecture Notes in Computer Science)*. Springer, Berlin, Heidelberg, 176–190.
[3] Julian Berk, Vu Nguyen, Sunil Gupta, Santu Rana, and Svetha Venkatesh. 2019. Exploration Enhanced Expected Improvement for Bayesian Optimization. In *Machine Learning and Knowledge Discovery in Databases (Lecture Notes in Computer Science)*. Springer International Publishing, Cham, Switzerland, 621–637.
[4] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. 2017. Variational Inference: A Review for Statisticians. *J. Amer. Statist. Assoc.* 112, 518 (2017), 859–877.
[5] Eric Brochu, Vlad M Cora, and Nando De Freitas. 2010. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. arXiv:1012.2599
[6] Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. 1995. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing* 16, 5 (1995), 1190–1208.
[7] Zhehui Chen, Simon Mak, and C. F. Jeff Wu. 2021. A hierarchical expected improvement method for Bayesian optimization. arXiv:1911.07285
[8] Andreas Damianou and Neil D. Lawrence. 2013. Deep Gaussian Processes. In *Artificial Intelligence and Statistics*. PMLR, 207–215.
[9] George De Ath, Richard M. Everson, Alma A. M. Rahat, and Jonathan E. Fieldsend. 2021. Greed is Good: Exploration and Exploitation Trade-offs in Bayesian Optimisation. *ACM Transactions on Evolutionary Learning and Optimization* 1, 1 (2021).
[10] George De Ath, Jonathan E. Fieldsend, and Richard M. Everson. 2020. What Do You Mean? The Role of the Mean Function in Bayesian Optimisation. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*. Association for Computing Machinery, New York, 1623–1631.
[11] Simon Duane, A. D. Kennedy, Brian J. Pendleton, and Duncan Roweth. 1987. Hybrid Monte Carlo. *Physics Letters B* 195, 2 (1987), 216–222.
[12] David Duvenaud. 2014. *Automatic model construction with Gaussian processes*. Ph.D. Dissertation. University of Cambridge.
[13] Zhiwei Feng, Qingbin Zhang, Qingfu Zhang, Qiangang Tang, Tao Yang, and Yang Ma. 2015. A multiobjective optimization based framework to balance the global exploration and local exploitation in expensive optimization. *Journal of Global Optimization* 61, 4 (2015), 677–694.
[14] Maurizio Filippone, Mingjun Zhong, and Mark Girolami. 2013. A comparative evaluation of stochastic-based inference methods for Gaussian process models. *Machine Learning* 93, 1 (2013), 93–114.
[15] Peter I Frazier. 2018. A tutorial on Bayesian optimization. arXiv:1807.02811
[16] Jacob Gardner, Geoff Pleiss, Kilian Q Weinberger, David Bindel, and Andrew G Wilson. 2018. GPyTorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., New York, 7576–7586.
[17] GPy. since 2012. GPy: A Gaussian process framework in python. http://github.com/SheffieldML/GPy.
[18] Mark S. Handcock and Michael L. Stein. 1993. A Bayesian Analysis of Kriging. *Technometrics* 35, 4 (1993), 403–410.
[19] Wilfred K. Hastings. 1970. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* 57, 1 (1970), 97–109.
[20] Philipp Hennig and Christian J. Schuler. 2012. Entropy search for information-efficient global optimization. *The Journal of Machine Learning Research* 13, 1 (2012), 1809–1837.
[21] José Miguel Henrández-Lobato, Matthew W. Hoffman, and Zoubin Ghahramani. 2014. Predictive entropy search for efficient global optimization of black-box functions. In *Advances in Neural Information Processing Systems*. MIT Press, Cambridge, Massachusetts, 918–926.
[22] Matthew D. Hoffman and Andrew Gelman. 2014. The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research* 15, 47 (2014), 1593–1623.
[23] Sture Holm. 1979. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics* 6, 2 (1979), 65–70.
[24] Donald R. Jones, Matthias Schonlau, and William J. Welch. 1998. Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization* 13, 4 (1998), 455–492.
[25] Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. 1999. An Introduction to Variational Methods for Graphical Models. *Machine Learning* 37, 2 (1999), 183–233.
[26] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. arXiv:1312.6114
[27] Joshua D. Knowles, Lothar Thiele, and Eckart Zitzler. 2006. *A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers*. Technical Report TIK214. Computer Engineering and Networks Laboratory, ETH Zurich, Zurich, Switzerland.
[28] Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M. Blei. 2017. Automatic Differentiation Variational Inference. *Journal of Machine Learning Research* 18, 14 (2017), 1–45.
[29] Harold J. Kushner. 1964. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal Basic Engineering* 86, 1 (1964), 97–106.
[30] Vidhi Lalchand and Carl Edward Rasmussen. 2020. Approximate Inference for Fully Bayesian Gaussian Process Regression. In *Symposium on Advances in Approximate Bayesian Inference*. PMLR, 1–12.
[31] David J. C. MacKay. 1992. *Bayesian methods for adaptive models*. Ph.D. Dissertation. California Institute of Technology.
[32] David J. C. MacKay. 1992. The evidence framework applied to classification networks. *Neural Computation* 4, 5 (1992), 720–736.
[33] David J. C. MacKay. 1999. Comparison of approximate methods for handling hyperparameters. *Neural Computation* 11, 5 (1999), 1035–1068.
[34] Michael D. McKay, Richard J. Beckman, and William J. Conover. 1979. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics* 21, 2 (1979), 239–245.
[35] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. 1953. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics* 21, 6 (1953), 1087–1092.
[36] Thomas Minka. 2001. *A family of algorithms for approximate Bayesian inference*. Ph.D. Dissertation. Massachusetts Institute of Technology.
[37] Jonas Močkus, Vytautas Tiešis, and Antanas Žilinskas. 1978. The application of Bayesian methods for seeking the extremum. *Towards Global Optimization* 2, 1 (1978), 117–129.
[38] Iain Murray and Ryan P Adams. 2010. Slice sampling covariance hyperparameters of latent Gaussian models. In *Advances in neural information processing systems*, Vol. 23. Curran Associates, Inc., New York, 1723–1731.
[39] Radford M. Neal. 1996. *Bayesian Learning for Neural Networks*. Lecture Notes in Statistics, Vol. 118. Springer New York, New York.
[40] Radford M. Neal. 1997. *Monte Carlo Implementation of Gaussian Process Models for Bayesian Regression and Classification*. Technical Report 9702. Department of Statistics, University of Toronto.
[41] Radford M. Neal. 2011. MCMC Using Hamiltonian Dynamics. In *Handbook of Markov Chain Monte Carlo*, Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng (Eds.). Chapman and Hall/CRC, New York, 113–162.
[42] Anthony O'Hagan. 1978. Curve Fitting and Optimal Design for Prediction. *Journal of the Royal Statistical Society: Series B (Methodological)* 40, 1 (1978), 1–24.
[43] Michael A. Osborne. 2010. *Bayesian Gaussian processes for sequential prediction, optimisation and quadrature*. Ph.D. Dissertation. Oxford University.
[44] Carl Edward Rasmussen. 1996. *Evaluation of Gaussian Processes and other Methods for Non-Linear Regression*. Ph.D. Dissertation. Department of Computer Science, University of Toronto.
[45] Carl Edward Rasmussen and Christopher K. I. Williams. 2006. *Gaussian processes for machine learning*. MIT Press, Cambridge, Massachusetts.
[46] Danilo Rezende and Shakir Mohamed. 2015. Variational Inference with Normalizing Flows. In *International Conference on Machine Learning*. PMLR, 1530–1538.
[47] Christian P. Robert and George Casella. 2005. *Monte Carlo Statistical Methods (Springer Texts in Statistics)*. Springer-Verlag, New York.
[48] Binxin Ru, Michael A. Osborne, Mark Mcleod, and Diego Granziol. 2018. Fast Information-theoretic Bayesian Optimisation. In *International Conference on Machine Learning*. PMLR, 4384–4392.
[49] Hugh Salimbeni, Vincent Dutordoir, James Hensman, and Marc Deisenroth. 2019. Deep Gaussian Processes with Importance-Weighted Variational Inference. In *International Conference on Machine Learning*. PMLR, 5589–5598.
[50] John Salvatier, Thomas V. Wiecki, and Christopher Fonnesbeck. 2016. Probabilistic programming in Python using PyMC3. *PeerJ Computer Science* 2 (2016), e55.
[51] Warren Scott, Peter Frazier, and Warren Powell. 2011. The Correlated Knowledge Gradient for Simulation Optimization of Continuous Parameters using Gaussian Process Regression. *SIAM Journal on Optimization* 21, 3 (2011), 996–1026.
[52] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. 2016. Taking the human out of the loop: A review of Bayesian optimization. *Proc. IEEE* 104, 1 (2016), 148–175.
[53] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical Bayesian Optimization of Machine Learning Algorithms. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., New York, 2951–2959.
[54] Jasper Snoek, Kevin Swersky, Rich Zemel, and Ryan Adams. 2014. Input Warping for Bayesian Optimization of Non-Stationary Functions. In *International Conference on Machine Learning*. PMLR, 1674–1682.
[55] András Sóbester, Stephen J. Leary, and Andy J. Keane. 2005. On the Design of Optimization Strategies Based on Global Response Surface Approximation Models. *Journal of Global Optimization* 33 (2005), 31–59.
[56] Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. 2010. Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. In *International Conference on Machine Learning*. Omnipress, Madison, 1015–1022.
[57] Michael L. Stein. 1999. *Interpolation of Spatial Data*. Springer New York, New York.

[58] Martin J. Wainwright and Michael I. Jordan. 2008. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends in Machine Learning* 1, 1–2 (2008), 1–305.
[59] Zi Wang and Stefanie Jegelka. 2017. Max-value Entropy Search for Efficient Bayesian Optimization. In *International Conference on Machine Learning*. PMLR, 3627–3635.
[60] Jeremy J. Warnes and Brian D. Ripley. 1987. Problems with likelihood estimation of covariance functions of spatial Gaussian processes. *Biometrika* 74, 3 (1987), 640–642.