Journal of Machine Learning Research 22 (2021) 1-49 Submitted 2/21;Revised 9/21; Published 9/21

GIBBON: General-purpose Information-Based Bayesian OptimisatioN

Henry B. Moss * HENRY.MOSS@SECONDMIND.AI
Secondmind.ai
Cambridge, UK

David S. Leslie D.LESLIE@LANCASTER.AC.UK
Lancaster University
Lancaster, UK

Javier González GONZALEZ.JAVIER@MICROSOFT.COM
Microsoft Research
Cambridge, UK

Paul Rayson P.RAYSON@LANCASTER.AC.UK
Lancaster University
Lancaster, UK

Editor: Marc Peter Deisenroth

Abstract

This paper describes a general-purpose extension of max-value entropy search, a popular approach for Bayesian Optimisation (BO). A novel approximation is proposed for the information gain — an information-theoretic quantity central to solving a range of BO problems, including noisy, multi-fidelity and batch optimisations across both continuous and highly-structured discrete spaces. Previously, these problems have been tackled separately within information-theoretic BO, each requiring a different sophisticated approximation scheme, except for batch BO, for which no computationally-lightweight information-theoretic approach has previously been proposed. GIBBON (General-purpose Information-Based Bayesian OptimisatioN) provides a single principled framework suitable for all the above, out-performing existing approaches whilst incurring substantially lower computational overheads. In addition, GIBBON does not require the problem's search space to be Euclidean and so is the first high-performance yet computationally light-weight acquisition function that supports batch BO over general highly structured input spaces like molecular search and gene design. Moreover, our principled derivation of GIBBON yields a natural interpretation of a popular batch BO heuristic based on determinantal point processes. Finally, we analyse GIBBON across a suite of synthetic benchmark tasks, a molecular search loop, and as part of a challenging batch multi-fidelity framework for problems with controllable experimental noise.

Keywords: Bayesian optimisation, entropy search, experimental design, multi-fidelity, batch

∗. Work completed while at the STOR-i Centre for Doctoral Training, Lancaster University, UK.

1. Introduction

A popular solution for the optimisation of high-cost black-box functions is Bayesian optimisation (Mockus et al., 1978, BO). By sequentially deciding where to make each evaluation as the optimisation progresses, BO can direct resources into evaluating promising areas of the search space to provide efficient optimisation. BO frameworks consist of two key components - a surrogate model and an acquisition function. By fitting a probabilistic surrogate model, typically a Gaussian process (Rasmussen, 2004, GP), to the previously collected objective function evaluations, we are able to quantify our current belief about which areas of the search space maximize our objective function. An acquisition function then uses this belief to predict the utility of making a particular evaluation, producing large values at 'reasonable' locations. BO automatically evaluates the location that maximises this acquisition function and repeats until a sufficiently high-performing solution is found. A popular application of BO is hyper-parameter tuning, with successful applications in computer vision (Bergstra et al., 2013), text-to-speech (Moss et al., 2020a) and reinforcement learning (Chen et al., 2018). Of particular note are the recent extensions of BO beyond Euclidean search spaces, for example when optimising synthetic genes (González et al., 2014; Tanaka and Iwata, 2018; Moss et al., 2020b) or performing molecular search (Gómez-Bombarelli et al., 2018; Griffiths and Hernández-Lobato, 2020; Vakili et al., 2020).

Various heuristic strategies have been developed to form BO acquisition functions, including Expected Improvement (Jones et al., 1998, EI), Knowledge Gradient (Frazier et al., 2008, KG) and Upper-Confidence Bound (Srinivas et al., 2010, UCB). More recently, a particularly intuitive and empirically effective class of acquisition functions has arisen based on information theory. Information-theoretic BO seeks to reduce uncertainty in the location of high-performing areas of the search space, as measured in terms of differential entropy. Such entropy-reduction arguments have motivated the three primary information-theoretic acquisition functions of Entropy Search (Hennig and Schuler, 2012, ES), Predictive Entropy Search (Hernández-Lobato et al., 2014, PES) and Max-value Entropy Search (Wang and Jegelka, 2017, MES), differing in their chosen measure of global uncertainty and employed approximation methods. Of particular popularity are acquisition functions based on MES, which reduce uncertainty in the maximum value attained by the objective function, a one-dimensional quantity. In contrast, both ES and PES seek to reduce uncertainty in the location of the maximum, a quantity which, as well as being well-defined only for Euclidean search spaces, requires prohibitively expensive approximation schemes. Due to the large number of acquisition function evaluations required to identify the next query point for each BO step, computational complexity is an important practical consideration when designing acquisition functions, particularly for applications with structured search spaces containing combinatorial elements.

Although the advent of MES acquisition functions has enabled the application of information-theoretic BO beyond problems with low-dimensional Euclidean search spaces, MES can not yet be regarded as a general-purpose acquisition function for two reasons.

1. Firstly, the existing extensions of MES supporting common BO extensions like Multi-fidelity BO (Moss et al., 2020d) and batch BO (Takeno et al., 2020) require additional approximations beyond those of vanilla MES, typically through the numerical integration of low-dimensional integrals. Multi-fidelity BO (also known as multi-task BO) leverages cheap approximations of the objective function to speed up optimisation, for example through exploiting coarse resolution simulations when calibrating large climate models (Prieß et al., 2011), whereas batch BO allows multiple objective function evaluations to be queried in parallel, a scenario arising

often in science applications, for example when training a collection of robots to cook (Junge et al., 2020). Therefore, although still cheaper than their ES- and PES-based counterparts, extensions of MES for multi-fidelity and batch BO do not inherit the simplicity and low-cost of vanilla MES.

2. Secondly, missing from the current extensions of MES is a computationally efficient method for general batch BO. Asynchronous batch BO supports scenarios where each of B workers are allocated individually to evaluate different areas of the search space, returning queries and being re-allocated one by one. In contrast, synchronous batch BO considers scenarios where where B workers are to be allocated in parallel, as is the case for many real-world settings including those relying on wet-lab evaluations, physical experiments, or any framework where workers do not have sufficient autonomy to be controlled separately. Takeno et al. (2020) propose an low-cost MES formulation suitable for asynchronous batch BO, however, their proposed extension of this method to synchronous batch BO (a distinction discussed in depth by Kandasamy et al. (2018a)) require prohibitively expensive approximations. Consequently, synchronous batch applications of MES have so far relied on generic batch heuristics suitable for any BO acquisition function, including greedy allocation through local penalisation (González et al., 2016a; Alvi et al., 2019) or using probabilistic repulsion models like determinantal point processes (Kathuria et al., 2016; Dodge et al., 2017), both of which support only Euclidean search spaces.

In this work we provide a single generalisation of MES suitable for BO problems arising from any combination of noisy, batch, single-fidelity, and multi-fidelity optimisation tasks. Crucially, unlike existing extensions of MES, our general-purpose acquisition function retains the computational cost of vanilla MES, with no requirement for numerical integration schemes. Therefore, we provide the first high-performing yet computationally light-weight framework for synchronous batch BO suitable for search spaces consisting of discrete structures.

Our primary contributions are as follows:

1. We propose an approximation for a general-propose extension of MES named General-purpose Information-Based Bayesian Optimisation (GIBBON). This approximation enables application of MES to a wide variety of problems, including those with combinations of synchronous batch BO, multi-fidelity BO and non-Euclidean highly-structured input spaces.

2. Analysis of GIBBON leads to a novel connection between information-theoretic search, determinantal point processes (Kulesza et al., 2012, DPP) and local penalisation (González et al., 2016a), providing currently missing theoretical justification for key attributes of these two popular heuristics previously chosen arbitrarily by users.

3. We analyse the computational complexity of GIBBON in the wider context of information-theoretic acquisition functions, providing a comprehensive evaluation of the computational overheads of information-theoretic BO.

4. We demonstrate the performance of GIBBON across a suite of popular benchmark optimisation tasks, including the first application of information-theoretic acquisition functions to high-cost string optimisation tasks and a sophisticated batch multi-fidelity framework for BO under controllable observation noise.

The remainder of the paper is structured as follows. Section 2 reviews prior work on MES and introduces the extended acquisition function that will be the focus of this work. In section 3, we propose the GIBBON acquisition function, before examining GIBBON in the context of existing heuristics for batch BO (Section 4). In Section 5 we consider the computational complexity of GIBBON in the wider context of information-theoretic BO. Finally, Section 6 provides a thorough empirical evaluation.

2. Max-value Entropy Search for Black-Box Function Optimisation

We now introduce max-value entropy search (MES) for BO, providing an information-theoretic motivation for the general-purpose framework that is the focus of this manuscript. We then introduce existing work that has applied more restrictive formulations of MES to deal with specific BO tasks, before briefly summarising additional popular acquisition functions that are not based on MES.

BO routines seek the global maximum

$$\mathbf{x}^* = \underset{\mathbf{x} \subset \mathcal{X}}{\operatorname{argmax}}\, g(\mathbf{x})$$

of a 'smooth' but expensive to evaluate black-box function $g : \mathcal{X} \to \mathbb{R}$. By sequentially choosing where and how to make each evaluation, BO directs resources into promising areas to efficiently explore the search space $\mathcal{X} \subset \mathbb{R}^d$ and provide fast optimisation. In its simplest formulation (henceforth referred to as standard BO), BO controls the locations $\mathbf{x} \in \mathcal{X}$ at which to collect (potentially noisy) queries of the objective function. A more general framework is that of multi-fidelity BO (Swersky et al., 2013) (also known as multi-task BO), where the 'quality' of each function query can also be controlled, for example by choosing the level of noise or bias across a (possibly continuous) space of fidelities $\mathbf{s} \in \mathcal{F}$. If these lower-fidelity estimates of g are cheaper to evaluate, then BO has access to cheap but approximate information sources that can be used to efficiently maximise g. In practical terms, each step of multi-fidelity BO needs to choose a location-fidelity pair $\mathbf{z} = (\mathbf{x}, \mathbf{s}) \in \mathcal{Z} = \mathcal{X} \times \mathcal{F}$ upon which to make the next evaluation. A further extension arises as batch BO, where we wish to exploit parallel resources by choosing a set of $B \geq 1$ locations $\{\mathbf{z}_1, .., \mathbf{z}_B\} \in \mathcal{Z}^B$ to be evaluated in parallel.

BO's decisions are governed by two primary components - a surrogate model and an acquisition function. The surrogate model makes probabilistic predictions of the objective function at not-yet-evaluated locations using the already collected location-evaluation tuples $D_n = \{(\mathbf{z}_i, y_{\mathbf{z}_i})\}_{i=1,..,n}$. The most most popular choice of model is a Gaussian process (Rasmussen, 2004, GP). GPs provide non-parametric regression over all functions of a smoothness controlled by a kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Crucially, our GP conditioned on D_n produces a tractable Gaussian predictive distribution that quantifies our current belief about the objective function across the whole search space. GP models can also be defined for multi-fidelity optimisation tasks (Kennedy and O'Hagan, 2000; Le Gratiet and Garnier, 2014; Klein et al., 2017a; Perdikaris et al., 2017; Cutajar et al., 2019) and when modelling highly-structured input spaces likes strings (Beck and Cohn, 2017), trees (Beck et al., 2015) and molecules (Moss and Griffiths, 2020).

Given such a probabilistic model over the search space, all that remains to perform an iteration of BO is an acquisition function measuring the utility of making evaluations. The Max-value Entropy Search (MES) of Wang and Jegelka (2017), with similar formulations considered by Hoffman and Ghahramani (2015) and Ru et al. (2018), seeks to query the objective function at locations that reduce

our current uncertainty in the maximum value of our objective function $g^* = \text{argmax}_{\mathbf{x}\in\mathcal{X}}\, g(\mathbf{x})$. In information theory (see Cover and Thomas, 2012, for a comprehensive introduction), uncertainty in the unknown g^* is measured by its differential entropy $H(g^*|D_n) = -\mathbb{E}_{g^*}\left[\log p(g^*)\right]$, where p is the predictive probability distribution function for g^* (as induced by the surrogate model). In particular, the utility of making an evaluation is measured as the reduction in the uncertainty of g^* it provides, a quantity known as the mutual information (MI).

Although initially proposed for just standard BO problems, an MES-based search strategy can be readily formulated for the general batch multi-fidelity framework (described above) by measuring the utility of evaluating a batch of fidelity evaluations as their joint mutual information with the maximum value. We henceforth refer to this general formulation, formally expressed in Definition 1, as General-purpose MES (GMES).

Definition 1 (The GMES acquisition function) *The GMES acquisition function is defined as*

$$\begin{aligned}\alpha_n^{GMES}(\{\boldsymbol{z}_i\}_{i=1}^B) \coloneqq & MI(g^*; \{y_{\boldsymbol{z}_i}\}_{i=1}^B | D_n) \\ = & H(g^*|D_n) - \mathbb{E}_{\{y_{\boldsymbol{z}_i}\}_{i=1}^B}\left[H(g^*|D_n \cup \{y_{\boldsymbol{z}_i}\}_{i=1}^B)\right], \end{aligned} \tag{1}$$

where $\{\boldsymbol{z}_i\}_{i=1}^B$ denotes the location-fidelity pairs of the batch elements and $y_{\boldsymbol{z}}$ denote the yet-unobserved results of querying location-fidelity pair $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{s}) \in \mathcal{X} \times \mathcal{F}$.

Note that standard BO, batch BO and multi-fidelity BO are trivial special cases of this general-purpose framework obtained by either or both of fixing the fidelity space $\mathcal{F}$ to a singleton containing just the true objective function or setting $B = 1$.

To provide resource-efficient optimisation, we must balance how much we expect to learn about g^* with the computational cost of the evaluations. Therefore, following the arguments of Swersky et al. (2013), each BO step chooses to evaluate the set of B locations that maximises the cost-weighted mutual information, i.e

$$\{\mathbf{z}_{|D_n|+1}, .., \mathbf{z}_{|D_n|+B}\} = \underset{\{\mathbf{z}_i\}_{i=1}^B \in \mathcal{Z}^B}{\text{argmax}} \frac{\alpha_n^{\text{GMES}}(\{\mathbf{z}_i\}_{i=1}^B)}{c(\{\mathbf{z}_i\}_{i=1}^B)},$$

where $c : \mathcal{Z}^B \to \mathbb{R}^+$ measures the costs of evaluating the batch. This cost function could be known *a priori* or estimated from observed costs (Snoek et al., 2012). The optimisation of acquisition functions is known as the *inner-loop* maximisation and, when considering continuous search spaces, is typically performed with a gradient-based optimiser. For discrete search spaces it is common to use local optimisation routines like DIRECT (Jones et al., 1993) or genetic algorithms (Moss et al., 2020b). For search spaces with discrete and continuous dimensions, hybrid optimisers can be used (Ru et al., 2020).

Unfortunately, calculating GMES in its full generality is challenging and providing a practically viable approximation strategy is the major contribution of this work. The primary difficulty in its computation arises from the lack of closed-form expression for the distribution of g^*, as required for all differential entropy calculations. We now end this section by discussing the three scenarios where specific sub-cases of GMES have already been used to provide highly effective BO — a noiseless variant of standard BO, multi-fidelity BO, and a special case of batch BO.

2.1 Max-value Entropy Search for noiseless standard BO

Firstly, we consider the original MES formulation of Wang and Jegelka (2017), where they perform standard BO with noiseless observations. This acquisition function is formally expressed as

$$\alpha_n^{\text{MES}}(\mathbf{x}) \coloneqq \text{MI}(y_{\mathbf{x}}; g^*|D_n) = H(y_{\mathbf{x}}|D_n) - \mathbb{E}_{g^*}\left[H(y_{\mathbf{x}}|g^*, D_n)|D_n\right]. \tag{2}$$

Here, the symmetric property of mutual information has been used to swap $y_{\mathbf{x}}$ and g^* in its definition, yielding an equivalent (albeit less intuitive) expansion. Crucially, the first term of the expansion of (2) is now simply the entropy of a multivariate Gaussian distribution with a convenient closed-form. Moreover, Wang and Jegelka (2017) note that under the assumption of exact objective function evaluations (where $y_{\mathbf{x}} = g(\mathbf{x})$), the distribution of $y_{\mathbf{x}}$ conditional on its maximum possible value (i.e knowing that $y_{\mathbf{x}} \leq g^*$) is simply that of a truncated Gaussian, also with a closed-form differential entropy. All that remains to calculate MES is to approximate an expectation over g^*. Wang and Jegelka (2017) build a Monte-Carlo estimate of the expectation with a set of samples $\mathcal{M}$ from g^*, providing a closed-form approximation of MES as

$$\alpha_n^{MES}(\mathbf{x}) \approx \frac{1}{|\mathcal{M}|}\sum_{m\in\mathcal{M}}\left[\frac{\gamma_{\mathbf{x}}(m)\phi\left(\gamma_{\mathbf{x}}(m)\right)}{2\Phi\left(\gamma_{\mathbf{x}}(m)\right)} - \log\Phi\left(\gamma_{\mathbf{x}}(m)\right)\right], \tag{3}$$

where Φ and ϕ are the standard normal cumulative distribution and probability density functions (as arising from the expression for the differential entropy of a truncated Gaussian) and $\gamma_{\mathbf{x}}(m) = \frac{m-\mu_n(\mathbf{x})}{\sigma_n(\mathbf{x})}$. Here, $\mu_n(\mathbf{x})$ and $\sigma_n^2(\mathbf{x})$ are the predictive mean and standard deviation for the objective function value g at location $\mathbf{x}$ as easily extracted from our surrogate model. The set of sample max-values $\mathcal{M}$ is built by modelling the empirical cumulative distribution function of g^* with a Gumbel distribution (see Wang and Jegelka (2017) for details) which can be sampled to yield M cheap but approximate sampled max-values. This Gumbel approximation provides a fast sampling strategy and has been successful across a wide range of BO applications (Wang and Jegelka, 2017; Moss et al., 2020d,c)

For the limited set of BO problems supported by this original MES acquisition function, MES has had great empirical success, typically outperforming other information-theoretic BO methods with an order of magnitude smaller computational overhead. However, once MES arguments are extended to support the more sophisticated BO frameworks (or even just to support noisy function evaluations), we will see that the second term of (3) is no longer (the expectation of) the differential entropy of a truncated Gaussian and additional approximations have to be made.

2.2 Max-value Entropy Search for multi-fidelity BO

MES-based search strategies have also been previously used for multi-fidelity BO through the MUlti-task Max-value Bayesian Optimisation (MUMBO) acquisition function of Moss et al. (2020d) (proposed in parallel by Takeno et al. (2020)) and, just like original MES, MUMBO has been shown to perform highly efficient BO. However, unlike when collecting exact observations of g, fidelity evaluations $y_{\mathbf{z}}|g^*$ no longer follow a truncated Gaussian distribution and instead follow an extended skew Gaussian distribution (as shown by Moss et al. (2020d) and re-derived in Section 3) which has no closed-form expression for its differential entropy (Azzalini, 1985). Therefore, the MUMBO acquisition function does not inherit all the computational savings of standard MES, requiring numerical integration. Note that by considering a single fidelity system, where low-fidelity

evaluations are just noisy observations of the true objective function, a multi-fidelity formulation of MES also serves as an extended standard (single-fidelity) MES suitable for when evaluations are contaminated with observation noise.

2.3 Max-value Entropy Search for Batch BO

Motivated by the empirical success of MES-based acquisition functions, it is natural to wonder if they can be used for batch BO. However, of the two popular batch scenarios of asynchronous and synchronous batches commonly considered in the BO literature, only asynchronous batch BO is currently supported by a MES-based acquisition function (Takeno et al., 2020). The primary practical distinction is that, while synchronous batch acquisition functions must be able to measure the total reduction in entropy provided by the joint evaluation of B locations, asynchronous batch BO has only to measure the relative reduction in entropy provided by making a single evaluation whilst taking into account the $B-1$ pending evaluations. Through clever algebraic manipulations, Takeno et al. (2020) require only single-dimensional numerical integrations when calculating the relative entropy reduction required for asynchronous batch BO. Unfortunately, as demonstrated in Section 3, complex interactions between each of the B fidelity evaluations $y_{\mathbf{z}_i}$ once conditioned on g^* (as present in the second term of (1)) prevents the approximation strategies employed by Takeno et al. (2020) (or those of Wang and Jegelka (2017) or Moss et al. (2020d)) being extended to the synchronous batch setting. In particular, a naive extension of Takeno et al. (2020)'s approach requires the prohibitively expensive numerical approximations of B-dimensional multivariate Gaussian cumulative density functions. In this work, we propose a novel approximation strategy for (1) completely free from numerical integrations, thus providing the first computationally light-weight information-theoretic acquisition function for synchronous batch BO.

2.4 Alternatives to Max-value Entropy Search

As discussed in Section 1, MES is not the only information-theoretic BO acquisition function and is a descendent of ES and PES. However, the original ES and PES, as-well as their extensions for batch BO (Shah and Ghahramani, 2015; Hernández-Lobato et al., 2017) and multi-fidelity BO (Swersky et al., 2013; Zhang et al., 2017), seek to reduce the differential entropy of the d-dimensional maximiser $\mathbf{x}^*$ (rather than the single dimensional g^* targeted by MES). The calculation of this entropy is challenging, requiring sophisticated and expensive approximation strategies (see Section 5). As well as being substantially more expensive than MES, the reliance of ES and PES on coarse approximations means they provide less effective optimisation (Wang and Jegelka, 2017; Moss et al., 2020d; Takeno et al., 2020). Moreover, the approximation strategy employed by PES restricts its use to only Euclidean search spaces

Of course, attempts have been made to adapt other standard acquisition functions to multi-fidelity and batch BO, with examples including EI (Picheny et al., 2010; Chevalier and Ginsbourger, 2013; Marmin et al., 2015), UCB (Contal et al., 2013; Kandasamy et al., 2016, 2017) and KG (Wu and Frazier, 2016, 2017). However, extensions of EI and UCB, although computationally cheap and often enjoying strong theoretical guarantees, are typically lacking in performance and even though KG-based methods can provide highly effective optimisation, their large computational cost restricts them to problems with function query costs large enough to overshadow very significant overheads (as demonstrated in Section 6). For batch BO, additional heuristic strategies have been developed that are compatible with any acquisition function, with the most popular and empirically successful

being the Local Penalisation of González et al. (2016a) and DPP-based approach of Kathuria et al. (2016) (see Section 4 for a thorough discussion). Alternative but less performant heuristics include approaches based on Stein methods (Gong et al., 2019) and Thompson sampling (Kandasamy et al., 2018a).

3. A Novel Approximation of General-purpose Max-value Entropy Search

In this section, we present the key theoretical contribution of this work: a novel approximation of the GMES acquisition function proposed in Section 2. In particular, we formulate GMES in terms of a the Information Gain (IG) — a measure of entropy reduction often used when pruning decision tree classifiers (Raileanu and Stoffel, 2004) and when selecting features for statistical models of textual data (Yang and Pedersen, 1997). The remainder of the section then details a novel approximation strategy for the information gain based on simple well-known information-theoretic inequalities, before demonstrating explicitly how this IG approximation can be used to approximate the GMES acquisition function.

3.1 GMES as a Function of Information Gain

Recall our proposed GMES acquisition function (1), defined as the mutual information between a set of B fidelity evaluations and the objective function's maximum value g^*. As in the derivation of the original MES acquisition function (2), the symmetric property of mutual information can be used to yield the expansion

$$\alpha_n^{\text{GMES}}(\{\mathbf{z}_i\}_{i=1}^B) \coloneqq H(\{y_{\mathbf{z}_i}\}_{i=1}^B|D_n) - \mathbb{E}_{g^*}\left[H(\{y_{\mathbf{z}_i}\}_{i=1}^B|D_n, g^*)|D_n\right]. \tag{4}$$

For ease of notation, we now define $A_i = y_{\mathbf{z}_i}$ and $C_i = g(\mathbf{x}_i)$ for each of the B candidate location-fidelity tuples $\mathbf{z}_i$, as well as the multivariate random variables $\mathbf{A} = (A_1, .., A_B)$ and $\mathbf{C} = (C_1, .., C_B)$. The information gain is then defined as the reduction in the entropy of $\mathbf{A}$ provided by knowing the maximal value of $C^* = \max \mathbf{C}$, i.e.

$$IG_n(\mathbf{A}, m|D_n) \coloneqq H(\mathbf{A}|D_n) - H(\mathbf{A}|C^* < m, D_n), \tag{5}$$

Comparing (4) and (5), it follows that the GMES acquisition function can be expressed in terms of IG as

$$\alpha_n^{\text{GMES}}(\{\mathbf{z}_i\}_{i=1}^B) = \mathbb{E}_{m\sim g^*}\left[IG_n(\mathbf{A}, m|D_n)\right].$$

We can now see that efficiently calculating (5) in general scenarios will allow principled max-value entropy search across a wide range of BO settings. This goal is therefore the focus of the remainder of this section.

3.2 Required Predictive Quantities

Before presenting our proposed approximation for IG, it is convenient to discuss the distributional forms induced by our surrogate GP model. All random variables are now assumed to be conditioned on the arbitrary information set D_n, which, alongside references to n, is henceforth dropped from our notation.

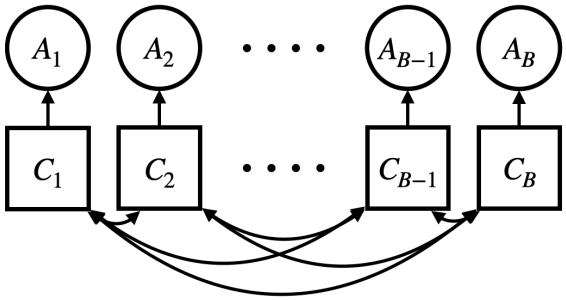


Figure 1: The considered dependency structure between the two set of random variables $\{A_1, .., A_B\}$ and $\{C_1, .., C_B\}$. Arrows denote the direction of dependence and latent variables are drawn in squares.

Courtesy of our GP surrogate model, we have that

$$\mathbf{A} \sim N(\boldsymbol{\mu}^A, \Sigma^A), \quad \mathbf{C} \sim N(\boldsymbol{\mu}^C, \Sigma^C) \quad \text{and} \quad \text{Corr}(A_i, C_i) = \rho_i,$$

for predictive means $\boldsymbol{\mu}^C, \boldsymbol{\mu}^A \in \mathbb{R}^B$, predictive covariances $\Sigma^C, \Sigma^A \in \mathbb{R}^{B \times B}$ and a vector of pairwise predictive correlations $\boldsymbol{\rho} \in \mathbb{R}^B$ (Rasmussen, 2004; see Appendix A for details on how these predictive quantities are easily extracted from a GP).

In addition to these well-known distributional forms, we can exploit the specific conditional structure of our GP surrogate model (which we describe below and summarise in Figure 1) to derive the conditional distribution of the random variable $\mathbf{A}$ given that $C^* < m$. In particular, our planned BO applications ensure that each A_j is conditionally independent of $\{C_i\}_{i \neq j}$ given C_j. This condition holds trivially for single-fidelity BO, where the difference between each A_i and C_i is just independent Gaussian noise. For multi-fidelity BO, this condition corresponds exactly to the *multi-fidelity Markov property* that is a key assumption underlying multi-fidelity GP modelling (Kennedy and O'Hagan, 2000; Le Gratiet and Garnier, 2014; Perdikaris et al., 2017). This is not a restrictive assumption, with O'Hagan (1998) showing that the multivariate Markov property holds for any GP surrogate model with a kernel that can be factorised into a product of kernels, one defined across the fidelity and one across the search space.

Under these dependence assumptions, Theorem 2 provides the distribution of $\mathbf{A}|C^* < m$ in closed-form, yielding a probability density function that, to the authors' knowledge, has not been previously considered in the statistics literature. Theorem 2 provides our first intuition for why the efficient calculation of the differential entropy $H(\mathbf{A}|C^* < m)$ is challenging, i.e. the presence of the multivariate Gaussian cumulative density in its probability density function.

Theorem 2 (Distribution of A given $C^* < m$) *Consider two B-dimensional multivariate Gaussian random variables $\boldsymbol{A}$ and $\boldsymbol{C}$ where $\boldsymbol{C} \sim N(\boldsymbol{\mu}^C, \Sigma^C)$ and each individual component of $\boldsymbol{A}$ is distributed as $A_j \sim N(\mu_j^A, \Sigma_{j,j}^A)$. Suppose further that each each pair $\{A_j, C_j\}$ are jointly Gaussian with correlation ρ_j, and that each A_j is conditionally independent of $\{C_i\}_{i \neq j}$ given C_j. Define $C^* = \max \boldsymbol{C}$. Then the conditional density of $\boldsymbol{A}$ given that $C^* < m$ is given by*

$$\frac{1}{\mathbb{P}(C^* < m)} \phi_{\boldsymbol{X}_1}(\boldsymbol{a}) \Phi_{\boldsymbol{X}_2}(\boldsymbol{m}),$$

where $\boldsymbol{m} = (m, .., m) \in \mathbb{R}^B$ *and* $\phi_{\boldsymbol{X}_1}$ *and* $\Phi_{\boldsymbol{X}_2}$ *are the probability density and cumulative density functions for the multivariate Gaussian random variables*

$$\boldsymbol{X}_1 \sim \boldsymbol{N}\left(\boldsymbol{\mu}^A, S + D\Sigma^C D\right) \quad \textit{and} \quad \boldsymbol{X}_2 \sim \boldsymbol{N}\left(\boldsymbol{\mu}^C + \Sigma^{-1} D S^{-1}(\boldsymbol{a} - \boldsymbol{\mu}^A), \Sigma^{-1}\right),$$

where $\Sigma^A = D\Sigma^C D + S$ *for* D *and* S*, diagonal matrices with elements* $D_{j,j} = \rho_j \sqrt{\frac{\Sigma^A_{j,j}}{\Sigma^C_{j,j}}}$ *and* $S_{j,j} = (1 - \rho_j^2)\Sigma^A_{j,j}$*, and* $\Sigma = \left(\left(\Sigma^C\right)^{-1} + DS^{-1}D\right)$.

Proof See Appendix B ■

Note that in the uni-variate case (i.e $B = 1$ and $C^* = C_1$), Theorem 2 collapses to the settings already considered when calculating MES and MUMBO in Section 2. Firstly, under the strong restriction that $C_1 = A_1$ (arising from BO without observation noise), $A_1|C^* < m$ follows the well-known truncated Gaussian distribution, which can be seen directly from Theorem 2 by setting $\rho_j = 1$, $\mu^C_{j,j} = \mu^A_j$ and $\Sigma^C_{j,j} = \Sigma^A_j$. This truncated Gaussian has a simple analytical expression for its differential entropy which is exploited by standard MES. Secondly, if C_j and A_j are not perfectly correlated, we see that the density of Theorem 2 reduces to that of an Extended Skew Gaussian (ESG) distribution (Azzalini, 1985) as required for the MUMBO acquisition function (see Appendix A of Moss et al. (2020d)). Although the differential entropy of an ESG has no closed-form expression (Arellano-Valle et al., 2013), we will later exploit the fact that its variance has an analytical form

$$\mathrm{Var}(A_j|C_j < m) = \Sigma^A_j \left(1 - \rho_j^2 \frac{\phi(\gamma_j(m))}{\Phi(\gamma_j(m))}\left[\gamma_j(m) + \frac{\phi(\gamma_j(m))}{\Phi(\gamma_j(m))}\right]\right), \tag{6}$$

where $\gamma_j(m) = (m - \mu^C_j)/\sqrt{\Sigma^C_{j,j}}$. We stress that, due to the complex interactions between each $A_j|C^* < m$, the joint distribution of $\mathbf{A}|C^* < m$ is not the multivariate ESG discussed by Azzalini and Valle (1996)).

3.3 Approximating Information Gain

We now present a lower bound $\mathrm{IG}^{\mathrm{APPROX}}$ for IG as Theorem 3. This bound is to be used as an approximation $IG \approx IG^{\mathrm{Approx}}$. We stress that replacing the maximisation of an intractable quantity with the maximisation of a lower bound is a well established strategy in the ML literature, for example in variational inference (Blei et al., 2017).

Theorem 3 (A lower bound for information gain) *Under the assumptions of Theorem 2, it holds that* $IG(\boldsymbol{A}, m) \geq IG^{Approx}(\boldsymbol{A}, m)$*, where*

$$IG^{Approx}(\boldsymbol{A}, m) \coloneqq \frac{1}{2}\log|R^A| - \frac{1}{2}\sum_{i=1}^{B} \log\left(1 - \rho_i^2 \frac{\phi(\gamma_i(m))}{\Phi(\gamma_i(m))}\left[\gamma_i(m) + \frac{\phi(\gamma_i(m))}{\Phi(\gamma_i(m))}\right]\right), \tag{7}$$

where $R^A \in \mathbb{R}^{B\times B}$ *is the predictive correlation matrix of* $\boldsymbol{A}$ *with entries* $R^A_{i,j} = \Sigma^A_{i,j}/\sqrt{\Sigma^A_{i,i}\Sigma^A_{j,j}}$.

Proof

Recall the definition of information gain $IG\left(\mathbf{A},m\right) \coloneqq H(\mathbf{A}) - H(\mathbf{A}|C^* < m)$. The first term of IG is simply the differential entropy of a multivariate Gaussian distribution and so can be written in closed-form as $H(\mathbf{A}) = \frac{1}{2}\log\left[(2\pi e)^B\left|\Sigma_A\right|\right]$, where $\left|\Sigma_A\right|$ is the determinant of the $B \times B$ co-variance matrix of $\mathbf{A}$. Unfortunately calculating the second term of IG is significantly more complicated, with a closed form expression only in the limited cases discussed above.

We now build an analytical upper bound for $H\left(\mathbf{A}|C^* < m\right)$ by exploiting three common information-theoretic inequalities. As derived in Cover and Thomas (2012), we know that,

$$H(\mathbf{A}) \leq \sum_{i=1}^{B} H(A_i), \quad H(A_i|C^* < m) \leq H(A_i|C_i < m), \quad \text{and} \quad H(A_i) \leq \frac{1}{2}\log 2\pi \mathrm{e}\mathrm{Var}(A_i),$$

where the second inequality is due to $\{C^* < m\}$ being a stronger condition than (i.e. implying that) $\{C_i < m\}$.

Applying the first two of these inequalities in sequence to $\mathbf{A}|C^* < m$ yields the upper-bound

$$H(\mathbf{A}|C^* < m) \leq \sum_{i=1}^{B} H(A_i|C_i < m).$$

Then, as we know that $A_j|C_j < m$ is an ESG (with a closed form expression for its variance), we can apply the third information-theoretic inequality to yield the analytical upper bound

$$\begin{aligned} H(\mathbf{A}|C^* < m) \leq & \frac{1}{2}\sum_{i=1}^{B}\log(2\pi e\mathrm{Var}(A_i|C_i < m)) \\ = & \frac{1}{2}\sum_{i=1}^{B}\log 2\pi e\Sigma_{i,i}^{A}\left(1-\rho_j^2\frac{\phi(\gamma_i(m))}{\Phi(\gamma_i(m))}\left[\gamma_i(m)+\frac{\phi(\gamma_i(m))}{\Phi(\gamma_i(m))}\right]\right). \end{aligned}$$

Substituting this upper bound into (5), we have a lower bound for the information gain

$$\begin{aligned} IG^{\text{Approx}}\left(\mathbf{A},m\right) \coloneqq & \frac{1}{2}\log\left|\Sigma^A\right| - \frac{1}{2}\sum_{i=1}^{B}\log\Sigma_{i,i}^{A}\left(1-\rho_j^2\frac{\phi(\gamma_i(m))}{\Phi(\gamma_i(m))}\left[\gamma_i(m)+\frac{\phi(\gamma_i(m))}{\Phi(\gamma_i(m))}\right]\right) \\ = & \frac{1}{2}\log\left|\Sigma^A\right| + \frac{1}{2}\log\prod_{i=1}^{b}\left(\Sigma_{i,i}^{A}\right)^{-1} - \\ & \frac{1}{2}\sum_{i=1}^{b}\log\left(1-\rho_j^2\frac{\phi(\gamma_i(m))}{\Phi(\gamma_i(m))}\left[\gamma_i(m)+\frac{\phi(\gamma_i(m))}{\Phi(\gamma_i(m))}\right]\right), \end{aligned}$$

which after defining the predictive correlation matrix R^A (with entries $R_{i,j}^A = \Sigma_{i,j}^A/\sqrt{\Sigma_{i,i}^A\Sigma_{j,j}^A}$) and noting that

$$\begin{aligned} \frac{1}{2}\log\left|\Sigma^A\right| + \frac{1}{2}\log\prod_{i=1}^{b}(\Sigma_{i,i}^{A})^{-1} &= \frac{1}{2}\log\left|\begin{pmatrix}\frac{1}{\sqrt{\Sigma_{1,1}^A}} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{\sqrt{\Sigma_{b,b}^A}}\end{pmatrix}\Sigma^A\begin{pmatrix}\frac{1}{\sqrt{\Sigma_{1,1}^A}} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{\sqrt{\Sigma_{b,b}^A}}\end{pmatrix}\right| \\ &= \frac{1}{2}\log\left|R^A\right|, \end{aligned}$$

provides the claimed expression. ■

3.4 GIBBON: General-purpose Information-Based Bayesian OptimisatioN

We end this section with explicitly demonstrating how IG_{Approx} can be used to approximate the GMES acquisition function. Recall that GMES can be expressed in terms of IG as

$$\alpha_n^{\text{GMES}}(\{\mathbf{z}_i\}_{i=1}^B) = \mathbb{E}_{m\sim g^*}\left[IG_n\left(\mathbf{A}, m|D_n\right)\right].$$

We have already provided an approximation for IG and so all that remains to approximate GMES is to deal with its outer expectation over g^*. Following the arguments of Wang and Jegelka (2017), we build a Monte-Carlo approximation of this expectation using a Gumbel-based sampler. Therefore, given a set of sampled max-values $\mathcal{M} = \{m_1, .., m_M\}$ of $g^*|D_n$ and access to the predictive distributions

$$\{y_{\mathbf{z}_i}\}_{i=1}^B|D_n \sim N(\boldsymbol{\mu}^y, \Sigma^y), \quad \{g(\mathbf{x}_i)\}_{i=1}^B|D_n \sim N(\boldsymbol{\mu}^g, \Sigma^g) \quad \text{and} \quad \text{Corr}(y_{\mathbf{z}_i}, g(\mathbf{x}_i)|D_n) = \rho_i,$$

we can approximate GMES with

$$\alpha_n^{\text{GIBBON}}(\{\mathbf{z}\}_{i=1}^B) = \frac{1}{|\mathcal{M}|}\sum_{m\in\mathcal{M}} IG^{APPROX}(\{y_{\mathbf{z}_1}, .., y_{\mathbf{z}_b}\}, m).$$

This construction is henceforth referred to as the General Information-Based Bayesian OptimisatioN (GIBBON) acquisition function and is defined as the closed-form expression in Definition 4 and demonstrated within a BO loop as Algorithm 1.

Definition 4 (The GIBBON acquisition function.) *The GIBBON acquisition function is defined as*

$$\alpha_n^{GIBBON}(\{\boldsymbol{z}\}_{i=1}^B) = \frac{1}{2}\log\left|R\right| - \frac{1}{2|\mathcal{M}|}\sum_{m\in\mathcal{M}}\sum_{i=1}^{B}\log\left(1-\rho_i^2\frac{\phi(\gamma_i(m))}{\Phi(\gamma_i(m))}\left[\gamma_i(m)+\frac{\phi(\gamma_i(m))}{\Phi(\gamma_i(m))}\right]\right),$$

where R is the correlation matrix with elements $R_{i,j} = \Sigma^y_{i,j}/\sqrt{\Sigma^y_{i,i}\Sigma^y_{j,j}}$ and $\gamma_i(m) = \frac{m-\mu_i^g}{\sqrt{\Sigma^g_{i,i}}}$.

At first glance, GIBBON's analytical form looks complex. However, as GIBBON contains only simple algebraic operations, it can be easily calculated in just a few lines of code, unlike existing ES-based and PES-based acquisition functions and all existing extensions of MES (as discussed in depth in Section 5). An important practical consideration for GIBBON is that, for continuous search spaces, it has accessible gradients that can easily be derived from its analytical expression, allowing efficient inner-loop optimisation.

We end this section with a visual analysis of the accuracy of the GIBBON approximation. We consider a standard BO task with exact objective function evaluations (i.e not multi-fidelity or batch optimisation) as, in this setting, the MES acquisition function provides an exact calculation of the entropy reductions. In Figure 2 we see that the approximation provided by GIBBON is very close to the ground truth provided by MES, with GIBBON and MES sharing modes and differing only in areas of the space that would never be selected by BO, i.e those locations with very low utility.

Algorithm 1: GIBBON for general-purpose BO tasks.

Input: Resource budget R, Batch size B, Gumbel sample size N

Initialise $n \leftarrow 0$ and spent resource counter $r \leftarrow 0$

Propose initial design I

while $r \leq R$ **do**

  Begin new iteration $n \leftarrow n + 1$

  Fit GP model to collected evaluations D_n

  Simulate N samples from $g^*|D_n$

  Compute α_n^{GIBBON} as given by Definition 4

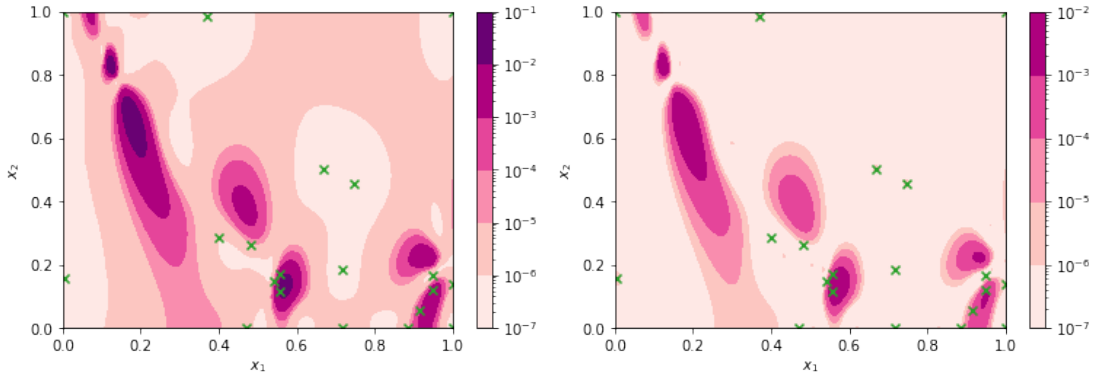
  Find B locations $\{\mathbf{z}_i\}_{i=1}^B$ maximising $\frac{\alpha_n^{\text{GIBBON}}(\{\mathbf{z}_i\}_{i=1}^B)}{c(\{\mathbf{z}_i\}_{i=1}^B)}$

  Evaluate new locations and collect evaluations $D_{n+1} \leftarrow D_n \bigcup \{(\mathbf{z}_i, y_{\mathbf{z}_i})\}_{i=1}^B$

  Update spent budget $r \leftarrow r + c(\{\mathbf{z}_i\}_{i=1}^B)$

end

Output: Believed maximiser $\text{argmax}_{\mathbf{x} \in D_n} g(\mathbf{x})$



(a) MES acquisition function surface (ground truth).

(b) GIBBON acquisition function surface.

Figure 2: Comparison of the MES and GIBBON acquisition functions for a two-dimensional BO task where MES can calculate entropy reductions exactly. The crosses denote the locations already queried by the BO routine. GIBBON provides a very close approximation of MES that reliably captures all its modes.

4. Relationship Between GIBBON and Heuristics for Batch Bayesian Optimisation

We now provide insights into the batch capabilities of our GIBBON acquisition function by drawing equivalences between GIBBON and two popular heuristics for batch BO — determinantal point processes (Section 4.1) and local penalisation (Section 4.2).

Recall that performing an iteration of BO requires the identification of optimal candidate points across the search space, i.e the maximisation of our acquisition function. For GIBBON, this *inner-loop* maximisation task corresponds to allocating a batch of B locations as

$$\{\mathbf{z}_{|D_n|+1}, .., \mathbf{z}_{|D_n|+B}\} = \underset{\mathbf{z}\in\mathcal{Z}}{\operatorname{argmax}}\, \alpha_n^{\text{GIBBON}}(\{\mathbf{z}_i\}_{i=1}^B).$$

Before introducing the two batch BO heuristics, it is convenient to provide an alternative expression for the GIBBON acquisition function. From Definition 4, we see that the GIBBON acquisition function for a candidate batch of B location-fidelity tuples can be decomposed into a sum of B GIBBON acquisition function evaluated separately for each tuple with an additional determinant term as

$$\alpha_n^{\text{GIBBON}}(\{\mathbf{z}\}_{i=1}^B) = \frac{1}{2}\log\left|R\right| + \sum_{i=1}^{B} \alpha_n^{\text{GIBBON}}(\mathbf{z}_i), \tag{8}$$

where R is the predictive correlation matrix of the batch. Note that the first term of this decomposition encourages diversity within the batch (achieving high values for points with low predictive correlation) whereas the second term ensures that evaluations are targeted in areas of the search space providing large amounts of information about g^*.

4.1 Relationship with Determinantal Point Processes

We can now interpret GIBBON in the context of a popular heuristic approach for batch design based on probabilistic models of repulsion known as Determinantal Point Processes (DPPs) (Kulesza et al., 2012). This comparison provides previously missing theoretical justification for choices of key DPP attributes which previously had to be chosen arbitrarily by practitioners.

DPPs provide a probability distribution over sets of points, such that sets of high-quality points (as measured by a quality function $q : \mathcal{X} \to \mathbb{R}$) with a diverse spread (as measured by a similarity kernel $s : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$) occur with high probability. More precisely, a particular set of points $\{\mathbf{x}_i\}_{i=1}^B$ occurs with probability.

$$\mathbb{P}(\{\mathbf{x}_j\}_{j=1}^B) \propto |L(\{\mathbf{x}_j\}_{j=1}^B)|, \tag{9}$$

where $L(\{\mathbf{x}_j\}_{j=1}^B)$ is a $b \times b$ matrix with elements $L_{i,j} = q(\mathbf{x}_i)q(\mathbf{x}_j)s(\mathbf{x}_i, \mathbf{x}_j)$.

Generating diverse but high-quality collections of points is exactly what we seek when allocating batches in BO problems. Unfortunately, a lack of understanding of how to choose appropriate quality functions and similarity kernels *a-priori* have previously limited the performance of DPP methods in BO, with existing applications requiring users to plug in arbitrary choices. The primary complication is that the relative scales of q and s trade-off the quality and diversity of batches, and so, for high-performance BO, these measures must be carefully chosen to complement (rather than dominate) each other. Consequently, the most common approach for using DPPs for BO is as part of *pure exploration* strategies, where the quality function is ignored ($q(\mathbf{x}) = 1$) and a DPP with a

radial basis function kernel as its similarity measure is sampled to allocate a whole batch (Dodge et al., 2017), or to allocate the $B-1$ elements remaining after choosing an initial point through a standard sequential BO routine (Kathuria et al., 2016). Related approaches have also been used for high-dimensional BO (Wang et al., 2017), where DPPs are used to sample a subset of the available search space dimensions. Note that these existing applications of DPPs to batch BO are limited in scope, supporting only single-fidelity problems over Euclidean search spaces, i.e those over which a standard similarity kernel can easily be defined.

We now explicitly show that our GIBBON acquisition function is equivalent to a DPP with specific choices of quality functions and similarity kernels. First define the exponential of our GIBBON acquisition function (with $B=1$) as a quality function $q^{\mathrm{G}}(\mathbf{z}) = \exp\left(\alpha^{\mathrm{GIBBON}}(\mathbf{z})\right)$ and the predictive correlation (as specified by our GP surrogate model) as a similarity kernel $s^{\mathrm{G}}(\mathbf{z}_i, \mathbf{z}_j) = R_{i,j}$. Then, after defining $L^{\mathrm{G}}(\{\mathbf{z}_j\}_{j=1}^{B})$ as the matrix with elements $L^{\mathrm{G}}_{i,j} = q^{\mathrm{G}}(\mathbf{z}_i) q^{\mathrm{G}}(\mathbf{z}_j) s^{\mathrm{G}}(\mathbf{z}_i, \mathbf{z}_j)$, simple algebraic manipulations allow the batch GIBBON acquisition function (8) to be expressed as

$$\alpha_n^{\mathrm{GIBBON}}(\{\mathbf{z}_j\}_{j=1}^{B}) = \frac{1}{2} log|L^{\mathrm{G}}|,$$

i.e the maximisation of our acquisition function corresponds to allocating the batch with maximal $|L^G|$, known as the *maximum a posteriori* (MAP) problem of DPPs. This is known to be NP-hard (Ko et al., 1995). However, the submodularity of DPPs ensures reasonable performance of greedy approximate solutions (as demonstrated by Gillenwater et al., 2012), explaining the observed effectiveness of a greedy batch-filling strategy when optimising our GIBBON acquisition function (see Section 6).

Recasting GIBBON as a DPP provides the first theoretical motivation for using DPPs for batch BO, with the particular choices of quality and similarity function arising from our information-theoretical derivation leading to significant improvements over existing DPP heuristics (Section 6). Moreover, we have greatly increased the generality of DPP-based BO, providing a formulation that supports multi-fidelity and structured search spaces, or any other framework using a surrogate model where posterior correlation is easily accessible.

4.2 Relationship with Local Penalisation

Another class of popular heuristics for batch BO are those based on local penalisation (LP) (González et al., 2016a; Alvi et al., 2019). Rather than explicitly balancing the diversity and quality of batches as two additive contributions, LP methods apply a multiplicative scaling to down-weight an acquisition function around locations already present in the batch, thus ensuring the selection of a diverse set of points. We now show that GIBBON can be interpreted as a penalisation strategy and consequently, we can make an explicit link between DPP- and LP-based BO routines. By recasting GIBBON as a local penalisation, we are able to derive a novel theoretically-justified penalisation function that outperforms existing LP methods.

For any choice of acquisition function $\alpha_n : \mathcal{X} \to \mathbb{R}$ taking positive values, an LP strategy greedily chooses the i^{th} element of the $n+1^{th}$ batch as

$$\mathbf{x}_{n+1,i} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmax}}\, \alpha_n(\mathbf{x}) \prod_{j=1}^{i-1} \psi(\mathbf{x}; \mathbf{x}_{n+1,j}),$$

where $\psi(\mathbf{x}, \mathbf{x}') : \mathcal{X} \times \mathcal{X} \to [0,1]$ is a *penalisation function*. By requiring that $\psi(\mathbf{x}, \mathbf{x}')$ is a non-increasing function of $||\mathbf{x} - \mathbf{x}'||$, we ensure that penalisation is largest when considering $\mathbf{x}$ close to

elements already present in the batch. The most popular penalisation function is the soft penaliser of González et al. (2016a)

$$\psi_{soft}(\mathbf{x}, \mathbf{x}') = \frac{1}{2}\text{erfc}(-z) \quad \text{for} \quad z = \frac{1}{\sqrt{\sigma_n^2(\mathbf{x}')}}\left(L||\mathbf{x} - \mathbf{x}'|| - g^* + \mu_n(\mathbf{x}')\right),$$

where erfc is the complementary error function and g^* is the current believed optimum. An important practical consideration of LP routines is that their performance is sensitive to predicting a Lipschitz constant L (i.e $|g(\mathbf{x}) - g(\mathbf{x}')| \leq L||\mathbf{x} - \mathbf{x}'|| \quad \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}$), for which point-estimates must be carefully extracted from previous function evaluations. Note that this Lipschitz constant can only be defined for Euclidean search spaces.

We now show that allocating batches by performing a greedy maximisation of GIBBON can be interpreted as an LP routine for specific choices of acquisition and penalisation functions. Define a re-scaled GIBBON acquisition function $\alpha_n^{scaled}(\mathbf{x}) = \left(e^{\alpha_n^{gibbon}(\mathbf{x})}\right)^2$ and a penaliser $\psi_{corr}(\mathbf{x}; \{\mathbf{x}_j\}_{j=1}^{i-1}) = \left|R(\{\mathbf{x}_j\}_{j=1}^{i-1} \cup \{\mathbf{x}\})\right|$ as the determinant of the batch's predictive correlation. After routine algebraic manipulations, we can see that allocating the i^{th} element of the $n+1^{th}$ batch according to a greedy maximisation of our GIBBON acquisition function is equivalently expressed as

$$\begin{aligned}\mathbf{x}_{n+1,i} &= \underset{\mathbf{x}\in\mathcal{X}}{\text{argmax}}\, \alpha_n^{\text{GIBBON}}\left(\{\mathbf{x}\} \cup \{\mathbf{x}_{n+1,j}\}_{j=1}^{i-1}\right) \\ &= \underset{\mathbf{x}\in\mathcal{X}}{\text{argmax}}\, \alpha_n^{scaled}(\mathbf{x})\psi_{corr}(\mathbf{x}; \{\mathbf{x}_{n+1,j}\}_{j=1}^{i-1}),\end{aligned}$$

i.e. the predictive correlation term in GIBBON can be interpreted as a form of local penalisation. However, unlike ψ_{soft} and the hard penaliser of Alvi et al. (2019), ψ_{corr} does not require the estimation of L, instead just using the easily accessible predictive correlation of our GP. In fact the superior performance of our proposed approach over existing LP methods suggests that complicated penalisation functions are not needed at all.

5. The Computational Complexity of Information-theoretic Bayesian Optimisation

In this final section before our experimental results, we analyse the computational overhead incurred by GIBBON and compare with all other existing information-theoretic acquisition functions, many of which are included in our experimental results of Section 6. We discuss the complexity of the information-theoretic acquisition functions mentioned in Sections 1 and 2: Entropy Search (Hennig and Schuler, 2012, ES), Predictive Entropy Search (Hernández-Lobato et al., 2014, PES) and its extensions PPES (Hernández-Lobato et al., 2017) and MF-PES (Zhang et al., 2017), Max-value Entropy Search (Wang and Jegelka, 2017, MES) and its extensions MUMBO (Moss et al., 2020d) and MF-MES (Takeno et al., 2020), as well as the Fast Information-Theoretic BO of Ru et al. (2018, FITBO). Although MFMES was originally designed for asynchronous batch BO, Takeno et al. (2020) do discuss (in their Appendix D.4) an alteration that allows the support for synchronous batch BO problems but with large computational cost. It is this variant of MFMES that we consider in this section and for our experimental results (Section 6).

The computational complexity of BO routines is hard to measure exactly as we do not know *a-priori* how many evaluations are required to maximise the highly multi-modal acquisition function in each inner loop. However, there are two main contributors to the computational cost of information-theoretic BO that can be analysed: a one-off initialisation calculation required to 'prepare' the

Method	Noise?	Multi-Fidelity ?	Batch?	Non-Euclidean ?	Initialisation costs	Acquisition query costs
ES	✓	✓	×	✓	$n^2e^{2d} + e^{3d}$	n^2e^d
PES	✓	×	×	×	$n^2e^{2d} + (n+d)^3e^d$	$n^2 + (n+d)e^d$
PPES	✓	×	✓	×	$n^2e^{2d} + (n+d)^3e^d$	$B^2n^2 + (B^3 + n + d)e^d$
MF-PES	✓	✓	×	×	$n^2e^{2d} + (n+d)^3e^d$	$n^2 + (n+d)e^d$
FITBO	×	×	×	×	1	n^2
MES	×	×	×	✓	n^2e^d	n^2
MUMBO	✓	✓	×	✓	n^2e^d	n^2
MF-MES	✓	✓	✓	✓	n^2e^d	$B^2n^2 + B^3 + B^2$
GIBBON	✓	✓	✓	✓	n^2e^d	$B^2n^2 + B^3$

Table 1: Computational complexity of existing entropy-based acquisition functions. d denotes the dimensions of the search space, n is the number of observations already collection, and B denotes batch size. Complexity results are correct to highest order terms only and ignore constant factors. We also summarise the types of BO problems supported by these acquisition functions (columns 1-4). For example, although standard MES's calculations strategy assumes exact, single-fidelity and purely sequential evaluations, MES does support non-Euclidean search space.

acquisition functions for each separate BO step, and the costs of each acquisition function query required for the inner-loop maximisation. These two complexity contributions are presented in Table 1, alongside a summary of the type of extended BO problems supported by each acquisition function, i.e whether they permit noisy, multi-fidelity, batch observations or non-Euclidean search spaces. We now derive the stated complexity results for initialisation and acquisition function query costs.

5.1 Acquisition Function Initialisation Costs

All BO routines incur a computational cost at the start of each individual BO step through the fitting of the surrogate model. The primary contribution to the cost of fitting a GP surrogate model on n data points is an $n \times n$ matrix inversion, i.e an $O(n^3)$ computation. Extracting a single predictive mean or co-variance from this GP then costs $O(n)$ and $O(n^2)$, respectively. As the overhead of fitting the GP is incurred across all BO routines, we leave out its contribution from our complexity analysis. We instead focus purely on the initialisation overheads specific to each information-theoretic acquisition function incurred when collecting sets of samples required for their approximation strategies. This set is reused for all acquisition function evaluations during a single inner-loop maximisation but re-sampled for each BO step.

All the samples required for information-theoretic acquisition functions can be separated into two distinct classes — those approximating single-dimensional quantities and those approximating quantities with the same dimensions as the search space. To paint a clear picture of computational cost, we consider BO problems with a search space of fixed dimension d and focus primarily on how the costs scale with respect to d, the batch size B, and the number of previously queried points n. Although all sample sizes are user-controllable, the efficiency of the resulting acquisition function depends sensitively on appropriately large sample sizes (as demonstrated for PES and MES by Wang and Jegelka (2017)). Therefore, sample sizes used when approximating d-dimensional quantities

must grow exponentially as $O(e^d)$ in order to preserve approximation accuracy. In contrast, the sample sizes required for effective approximations of single dimensional quantities can be chosen independently of d and so are denoted as $O(1)$ in our complexity analysis.

As discussed in Section 2, MES-based acquisition functions (including GIBBON), uses a Gumbel sampler to access samples of the maximum value g^*. This sampler evaluates our GP surrogate model's posterior (at $O(n^2)$ cost) across $O(e^d)$ points to form a discretisation of the d-dimensional search space. Each of the required $O(1)$ samples of g^* (a single dimensional quantity) can then be extracted with $O(1)$ cost, yielding an overall complexity of $O(n^2e^d)$. As shown in Table 1, GIBBON's initialisation costs are substantially lower than those of the acquisition functions based on PES and ES. Only FITBO has a lower initialisation cost, however it has not seen widespread use as it supports only noiseless standard BO tasks and employs a complicated construction requiring linear approximations of non-central χ^2 process (operations not supported by GP libraries). For the ES and PES-based acquisition functions, which require samples from the d-dimensional objective function maximiser $\mathbf{x}^*$, initialisation costs are substantial.

In ES, each sample of $\mathbf{x}^*$ is the maximum of a sample function drawn from the GP across an $O(e^d)$ discretisation of the search space. Simulating these function draws requires a one-off $O(e^{3d})$ computation for the Cholesky factor of the predictive co-variance matrix evaluated across the discretisation, as accessed with an $O(n^2)$ cost for each of its $O(e^{2d})$ elements. Consequently, the initialisation of ES incurs a sizeable $O(n^2e^{2d} + e^{3d})$ complexity scaling. PES also requires samples of $\mathbf{x}^*$ but instead maximises the sample draws from a finite feature approximation of the GP surrogate model (Rahimi and Recht, 2008), requiring just an $O(n^2)$ cost for each of the required $O(e^d)$ samples. However, unlike ES, PES incurs the additional cost of pre-computing an $n + d$-dimensional matrix inversion for each sample. Therefore, PES has a total initialisation cost of $O(n^2e^d + (n + d)^3e^d))$. Note that the finite feature approximation employed by PES and its variants is only rigorously defined for GPs with stationary kernels and Euclidean search spaces.

5.2 Acquisition Function Query Costs

We now discuss the computational complexity of each individual acquisition function query. As highlighted in Table 1, not only does the GIBBON acquisition function match the lowest query costs attained by any information-theoretic acquisition functions, but it is suitable for standard, stochastic, multi-fidelity and batch optimisation.

To calculate GIBBON and the other MES-based acquisition functions, we require the joint predictive distribution across B proposed batch locations. Accessing these B^2 predictive co-variance terms from a GP surrogate model and then taking its determinant cost $O(B^2n^2)$ and $O(B^3)$, respectively. Finally, GIBBON calculates an analytical expression for each of the $O(1)$ samples from g^* and across each of the batch elements, yielding an overall complexity of $O(B^2n^2 + B^3)$. MF-MES has a similar construction to GIBBON, but requires the additional calculation of a B-dimensional integral, for which a naive numerical approximation would require an $O(e^B)$ cost. Following Takeno et al. (2020), this integral can also be evaluated using a sophisticated recursive strategy for calculating multi-variate Gaussian cumulative density functions with $O(B^2)$ cost, however, we found this routine to incur a large constant overhead that dominated our acquisition function calculations. Similarly, we stress that although all MES-based acquisition functions have $O(n^2)$ cost (in the non-batch setting), FITBO, MUMBO and MF-MES all require additional numerical integrations (over GIBBON) that incur a significant constant cost factor that does not show in our highest order complexity analy-

sis. Consequently, the experiments of Section 6 show that GIBBON is substantially cheaper than MUMBO and MF-MES in practice.

The ES and PES-based acquisition functions incur a substantially larger query cost than GIBBON. Their primary computational bottleneck is the requirement of separate calculations for each of their $O(e^d)$ samples of $\mathbf{x}^*$. In ES, each evaluation requires an n^2 prediction from the GP for each location across a small $O(1)$-sized collection of points for each sampled $\mathbf{x}^*$. In contrast, PES requires only a single prediction from the GP but additional $O(n+d)$ manipulations for each of its $O(e^d)$ pre-computed kernel matrices. For batch BO, PPES requires B^2 GP predictions and a B^3 calculation to access the determinant of the batch's posterior co-variance, as well as an additional B^3 determinant calculations for each pre-computed kernel matrix.

6. Experiments

We now finish this manuscript with a comprehensive empirical evaluation of our GIBBON acquisition function. In particular, we consider batch (Section 6.1) and multi-fidelity (Section 6.3) synthetic benchmarks, as-well as well as a molecular design loop over a non-Euclidean and highly-structured search space (Section 6.5). Finally, we examine the performance of GIBBON when inserted into a challenging real-world BO framework that requires both batch and multi-task decision making. Implementations of GIBBON are available in three popular Python libraries for BO: Emukit (Paleyes et al., 2019), BoTorch (Balandat et al., 2020) and Trieste (Berkeley et al., 2021) .

For clarity, all of our experiments follow a similar format. We run each of the considered BO methods across 50 random seeds, plotting mean performance and a single standard error. For batch algorithms, we count the evaluation of a batch as a single BO iteration. Suboptimality of the current believed optimum $\hat{\mathbf{x}}$ is measured by the regret $g(\mathbf{x}^*) - g(\hat{\mathbf{x}})$, where $\mathbf{x}^*$ is the true maximiser. For some experiments we also measure the time taken to choose the next query points (referred to as the optimisation overhead). This computational cost of performing BO includes fitting the GP surrogate model as well as initialising and maximising the acquisition function. All experiments reporting optimisation overheads were performed on a quad core Intel Xeon 2.30GHz processor.

Across all our experiments, we see the same general behaviour: GIBBON at least matches, and often exceeds, the performance of existing high-performance acquisition functions whilst incurring an order of magnitudes lower computational overhead. Moreover, the breadth of our experiments showcases that GIBBON is truly a general-purpose acquisition function, forming a computationally light-weight acquisition function suitable for standard BO extensions, batch high-cost string design problems and sophisticated synchronous batch multi-task BO frameworks.

Overall, the purpose of our experiments is to demonstrate how GIBBON performs relative to other BO acquisition functions, with a primary focus on existing MES-based approaches. For completeness, we also compare against a range of additional methods, chosen to reflect their popularity, code availability and suitability for the particular experiment. To this end, we compare GIBBON with all the acquisition functions supported by BoTorch and Emukit, as-well as our own implementations of the batch heuristics discussed in Section 4. We will introduce these competitors alongside the relevant empirical results. Unfortunately, the PES-based methods discussed in Section 5 do not have implementations in BoTorch or Emukit. Moreover, we could not find any other comparable maintained software implementations, likely due to demonstrably worse performance of PES than MES (as shown by Wang and Jegelka, 2017) and PES's difficult-to-implement subroutines (Section 5).

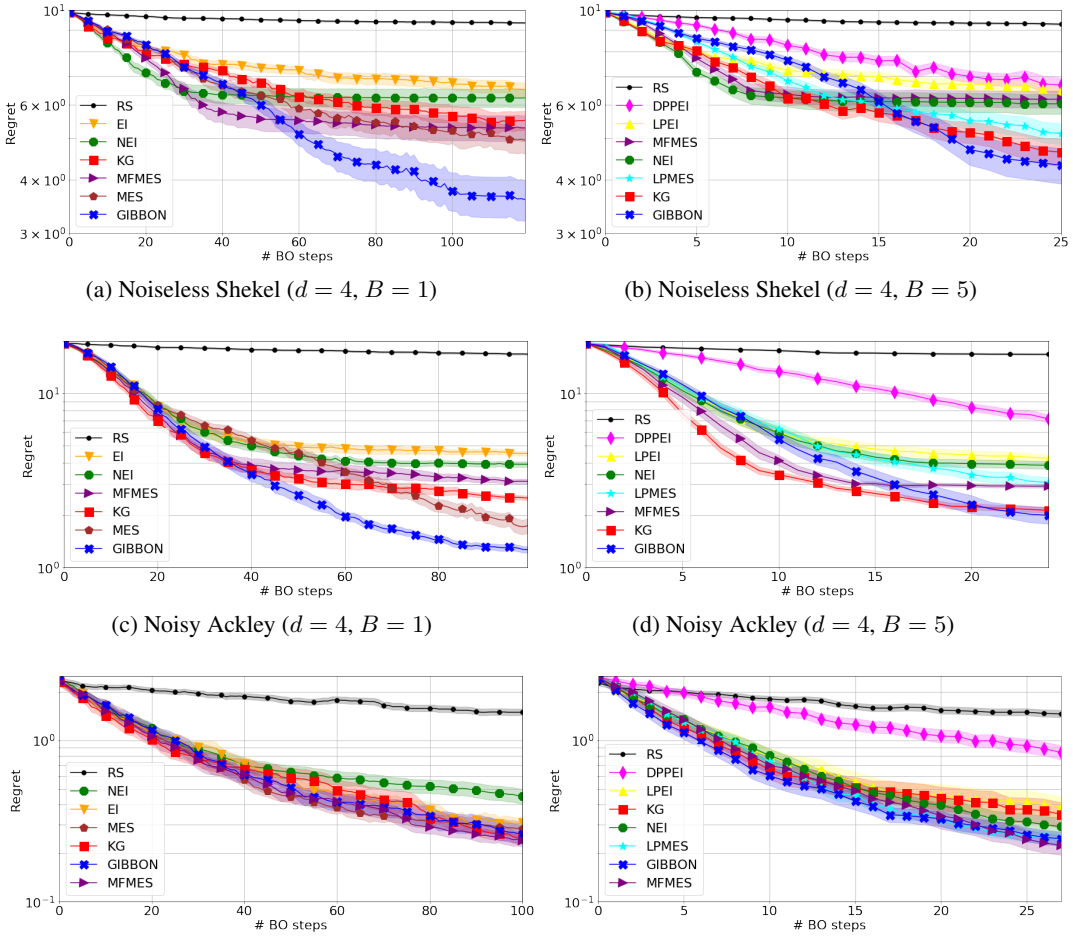
6.1 Standard and Batch Optimisation

For our first set of experiments, we consider a set of synthetic functions provided with the BoTorch package. In particular, we recreate two of the experiments of Balandat et al. (2020) by maximising the Hartmann ($d = 6$) and Ackley functions ($d = 4$), each with observations perturbed by centred Gaussian noise with a variance of 0.25. In addition, we also consider the Shekel function ($d = 4$) under exact observations. For details of these synthetic functions, we refer readers to Appendix C.1. Following the setup of Balandat et al. (2020), we initialise all routines by evaluating $2d + 2$ random locations, refit our GP's kernel parameters after each BO step, and choose the current believed optimum $\mathbf{x}^*$ by maximising the posterior mean of the GP surrogate model. For each experiment, we separately consider purely sequential BO ($B = 1$) and batch BO ($B = 5$), recording the evaluation of the whole batch as a single optimisation step.

For all our experiments, we report the performance of GIBBON and Expected Improvement (EI), as well as standard MES (applied to noisy problems by assuming exact observations). In addition, we also ran the acquisition functions already supported in BoTorch, i.e Knowledge Gradient (KG), Noisy Expected Improvement (NEI) (Picheny et al., 2010), and MFMES (the multi-fidelity MES extension of Takeno et al. (2020), used here to support noisy observations). We stress that MFMES was designed to provide computationally light-weight asynchronous batch BO and we will see that its adaptation to synchronous problems (as implemented by BoTorch and discussed earlier in Sections 2 and 5) incurs a substantial computational overhead. For our batch problems, we also implemented BoTorch versions of Local Penalisation (LPEI) and the DPP heuristic (DPPEI) of Kathuria et al. (2016), both using EI as their base acquisition function (as considered by González et al. (2016a) and Kathuria et al. (2016)). In addition, we also provide local penalisation with an MES base acquisition function (LPMES), a combination not tested by González et al. (2016a) but found to be particularly effective in our experimentation. All MES-based acquisition functions (including GIBBON) use 5 max-values sampled from a Gumbel distribution fit to surrogate model predictions at $10,000 * d$ random locations and are re-sampled for each BO step. All other implementation parameters follow the BoTorch defaults.

For acquisition function maximisation we use BoTorch's gradient-based maximiser. However, as this inner-loop maximisation can be challenging since it corresponds to a highly multi-modal maximisation across a $B \times d$-dimensional space. Therefore most batch BO routines build batches greedily by breaking batch design into B separate d-dimensional maximisations. Consequently, for all approaches (including our GIBBON acquisition function) except KG , batches are constructed in this greedy manner with a maximisation budget of $10 * d$ random restarts for each element of the batch. Although KG is able to jointly allocate batches, its large computational cost restricted us to 20 restarts (the amount recommended by the BoTorch authors).

Across the three synthetic experiments (Figure 3) we see that GIBBON provides efficient high-precision optimisation, yielding small regret in competitively few iterations for both sequential and batch BO. Note that in the noiseless and purely sequential case (Figure 3a), although MFMES can be shown to collapse exactly to the acquisition function of MES, the performance of these two methods differ as BOTorch's implementation of MFMES still relies on numerical approximations (albeit with a a very small observation noise term). Moreover, MFMES's reliance on rough numerical approximations means that its acquisition function struggles to provide high-precision optimisation once the acquisition function values are sufficiently small (i.e towards the end of the optimisation). Consequently, although sometimes achieving fast initial optimisation, MFMES fails to achieve as



(a) Noiseless Shekel ($d = 4$, $B = 1$)

(b) Noiseless Shekel ($d = 4$, $B = 5$)

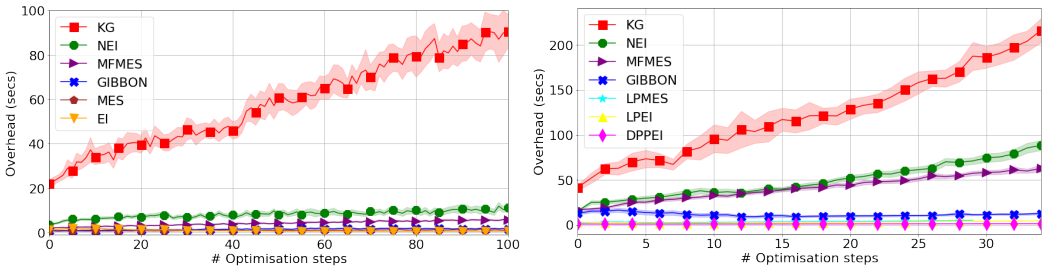
(c) Noisy Ackley ($d = 4$, $B = 1$)

(d) Noisy Ackley ($d = 4$, $B = 5$)

(e) Noisy Hartmann ($d = 6$, $B = 1$)

(f) Noisy Hartmann ($d = 6$, $B = 5$)

Figure 3: Optimisation of synthetic benchmark functions. GIBBON provides efficient and high-precision optimisation, matching or exceeding the performance of existing approaches.



(a) B=1 (b) B=5

Figure 4: The computational overheads incurred while optimising the Hartmann function. GIBBON's costs remains low throughout the optimisation, whereas the other high-performing batch acquisition functions costs increase dramatically as the optimisation progresses.

small final regret as GIBBON. Surprisingly, GIBBON is able to outperform even standard MES in the noiseless optimisation task of Figure 3a, the scenario for which standard MES is exact. As GIBBON approximates MES, we expected it to perform strictly worse for this example. We delve deeper into this phenomenon in Appendix E.

Of particular note is the order of magnitude smaller overhead incurred by GIBBON over the other high-performing acquisition functions (NEI, KG and MFMES) as summarised in Table 2a (for $B = 1$) and Table 2b (for $B = 5$). In particular, batch KG incurs at least a 10 times larger overhead than GIBBON. Moreover, Figure 4 shows that, while the computational overhead of batch KG, MFMES and NEI increase substantially as the optimisation progresses, GIBBON's overhead settles to fixed cost. We hypothesise that the initial (small) rise in the computational overhead of GIBBON is caused due to early acquisition functions having wider modes that require more local optimisation steps, a property also likely shared by other acquisition functions but disguised by their growing acquisition function cost. Although MFMES and GIBBON share the same order complexity with respect to the number of BO steps (see Table 1), we see that the large cost of numerical integration renders MFMES significantly more expensive than GIBBON in practice. Moreover, the BoTorch implementation of synchronous batch MFMES employs multiple model fits within each batch allocation to ensure approximation accuracy and so its cost scales poorly with the number of optimisation steps.

Figure 5 confirms our earlier claim that GIBBON is indeed a high-performance yet computationally light-weight acquisition function, showing that GIBBON performs better than all competing acquisition functions while incurring a computational overhead only slightly worse than the simple but low-performance approaches.

6.2 Ablation Study

Before assessing GIBBON across a wider range of BO tasks, we now perform a brief ablation study into GIBBON's user-controllable parameters and how they affect performance on the noisy Hartmann function introduced above. In particular, we focus on batch size (B) and sensitivity to the quality of max-value samples used to calculate GIBBON.

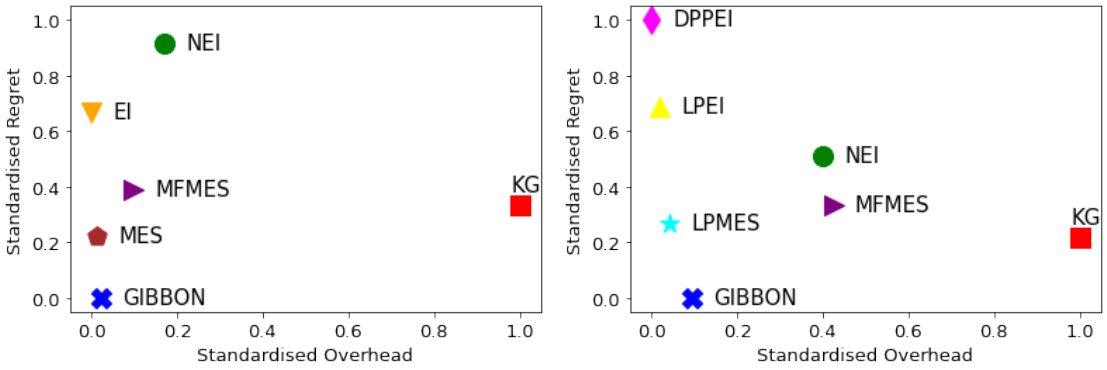
	Computational Overhead (seconds 1 d.p.)		
	Shekel (d=4)	Ackley (d=4)	Hartmann (d=6)
EI	0.2 (±0.0)	0.2 (±0.1)	0.8 (±0.1)
MES	0.5 (±0.1)	**0.5 (±0.0)**	1.0 (±0.1)
NEI	3.5 (±0.3)	3.0 (±0.2)	8.9 (±0.7)
MFMES	**3.0 (±0.4)**	0.7 (±0.1)	4.5 (±0.2)
KG	13.0 (±0.8)	22 (±1.0)	**66.6 (±4.6)**
GIBBON	**0.6 (±0.1)**	**0.8 (±0.1)**	**1.5 (±0.1)**

(a) Computational overheads for sequential BO ($B = 1$).

	Computational Overhead (seconds 1 d.p.)		
	Shekel (d=4)	Ackley (d=4)	Hartmann (d=6)
DPPEI	0.8 (±0.1)	0.8 (±0.0)	1.2 (±0.0)
LPEI	1.4 (±0.2)	2.3 (±0.1)	2.9 (±0.1)
LPMES	2.9 (±0.1)	3.3 (± 0.1)	3.5 (± 0.1)
NEI	21.3 (±1.8)	23.4 (±0.6)	43.0 (±2.6)
MFMES	24.4 (±2.3)	26.7 (±0.6)	**38.6 (±1.9)**
KG	**58.1 (±4.4)**	**53.0 (±3.1)**	103.4 (±6.2)
GIBBON	**5.0 (±0.5)**	**5.8 (±0.7)**	**13.3 (±1.3)**

(b) Computational overheads for batch BO ($B = 5$)

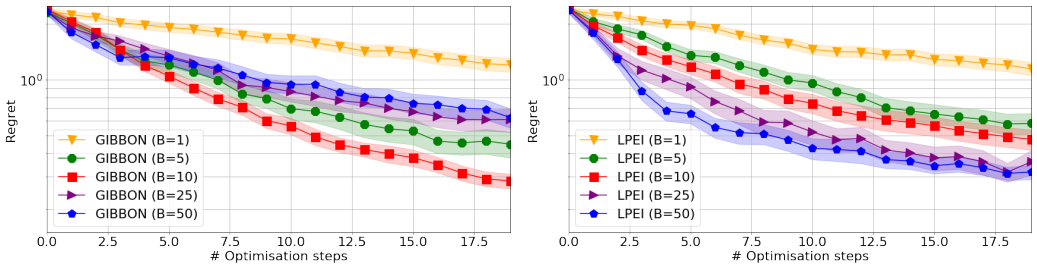
Table 2: Computational overheads for the synthetic benchmarks of Figure 3 averaged over the whole optimisation run. The two algorithms achieving lowest regret for each task are highlighted, demonstrating that GIBBON at least matches the overhead of other high-performing sequential acquisition functions and incurs a significantly lower overhead than other batch high-performing acquisition functions.



(a) B=1

(b) B=5

Figure 5: Comparison of the final regret achieved by each BO method with their computational overheads. Scores are standardised to sit within $[0, 1]$ and averaged across the three synthetic benchmark tasks. Lower scores on the x and y axis represent a smaller computational overheads and more effective optimisation, respectively.



(a) GIBBON (b) LPEI

Figure 6: Optimisation of the noisy Hartmann function over 20 iterations across a range of batch sizes. GIBBON is able to provide effective batch optimisation for small to moderate batch size ($B < 25$), however, fails to effectively control large batches. Although LPEI is able to leverage larger parallel resources than GIBBON, it still fails to match the performance of GIBBON (B=10) even when controlling much larger batches (B=50).

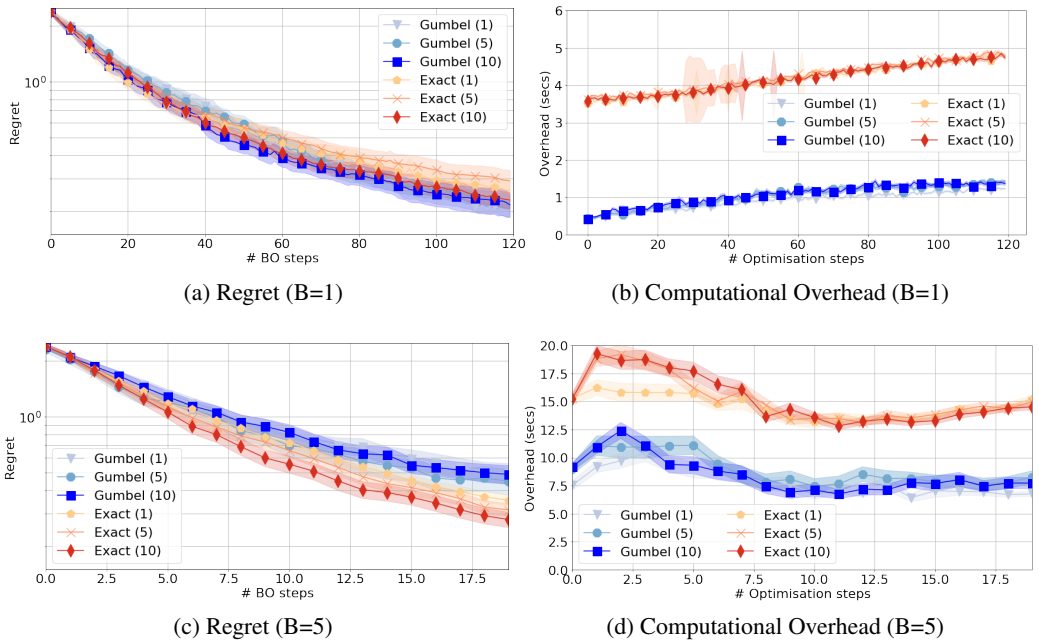
6.2.1 GIBBON FOR LARGE BATCH OPTIMISATION

GIBBON is a promising candidate for optimising under a large degree of parallelism as its batches can be constructed greedily without requiring B posterior updates. Unfortunately, GIBBON fails to realise this promise in practice. Figure 6 shows that GIBBON fails to effectively leverage large parallel resources and even displays a significant drop in performance once considering batches of size 25. In contrast, LPEI is able to continually improve regret by considering larger and larger batches. We stress that LPEI, even when controlling batches of 50 elements (i.e. $1,000$ total evaluations), still achieves lower regret than GIBBON with batches of size 10 (i.e 200 evaluations).

As demonstrated in Appendix D, GIBBON can be easily modified to support optimisation under large batches by a simple down-weighting of its repulsion term. Therefore, we posit that poor performance of GIBBON in this large batch setting is due to a degradation of the approximation accuracy in our analytical lower bound as we increase batch size. To see this, consider GIBBON's diversity-quality decomposition first introduced in Section 4 (i.e. Equation (8)). Considering large batches ensures that at least some candidate elements must be close together and so have high correlation. Consequently, GIBBON is dominated by its repulsion term (the determinant of the batch's predictive correlation matrix) and the maximisation of GIBBON leads to repeated query points around the edge of the search space, resulting in a substantial degradation in the stability of our GP surrogate model and poor exploration in more important areas of the space. Therefore, in this large batch setting, GIBBON effectively collapses to an almost pure exploration DPP-based method similar to the poorly performing DPPEI examined in our synthetic experiments.

6.2.2 GIBBON WITH THOMPSON-SAMPLED MAXIMUM VALUES

Our proposed calculation strategy for GIBBON requires access to M samples from the objective function's currently unknown maximum value g^*. We now investigate the sensitivity of GIBBON with respect to the quality of these random samples. For all the other experiments in this work we used the low-cost but approximate Gumbel sampler, as proposed by Wang and Jegelka (2017). By approximating the empirical CDF of g^* with an analytical Gumbel distribution, Gumbel sampling is able to return M approximate max-value samples over a grid of N candidate locations with



(a) Regret (B=1)

(b) Computational Overhead (B=1)

(c) Regret (B=5)

(d) Computational Overhead (B=5)

Figure 7: Regret performance (left) and computational overhead (right) of GIBBON when optimising the noisy Hartmann function using batches of size 1 (top row) or 5 (bottom row) across different max-value sampling strategies. Although exact sampling provides a small boost in GIBBON's performance for the batch experiment, Gumbel sampling seems adequate for sequential optimisation. Moreover, the resulting computational overhead of GIBBON with exact sampling is five and three times the cost of GIBBON with Gumbel sampling, when controlling batches of size $B = 1$ and $B = 5$, respectively.

cost $O(M + N)$. Of course, we can access exact samples of g^* by maximising sample functions drawn from our GP (i.e a Thompson-sampling style approach). However, extracting M such exact samples incurs an $O(MN + N^3)$ cost and so, if used as part of GIBBON's calculation strategy, would add significantly to GIBBON's optimisation overhead. However, as using exact max-value samples removes the only source of approximation in GIBBON aside from our information-theoretic lower bound, this alternative Thompson sampling strategy may lead to improved optimisation — a hypothesis we now investigate.

In Figure 7, we present the performance of GIBBON when using 1, 5 and 10 approximate (Gumbel) or exact (Thompson) sampled maximum values. Due to the significant cost of the exact Thompson sampler, we can sample over only $1,000 * d$ random candidate locations, as opposed to the $10,000 * d$ used for our Gumbel sampler. We see that, in exchange for a large increase in computational overhead, the exact sampler can sometimes lead to a small increase in performance over our standard Gumbel-based batch GIBBON implementation. We stress that changing sampler had no effect on the performance of purely sequential ($B = 1$) BO. This small and inconsistent performance improvement is not enough to justify the additional overhead of exact sampling and so, in order to remain loyal to our motivation of GIBBON as a computationally light acquisition

	Overhead for Multi-fidelity Optimisation (Seconds 1 d.p.)			
	Curin (d=4)	Hartmann (d=3)	Hartmann (d=6)	Borehole (d=8)
ES	16.6 ($\pm$0.7)	59.7 ($\pm$4.2)	229.8 ($\pm$15.3)	-
MUMBO	13.7 ($\pm$0.6)	18.6 ($\pm$1.0)	79.9 ($\pm$6.2)	51.5 ($\pm$7.5)
GIBBON	**4.0 ($\pm$0.2)**	**9.9 ($\pm$0.7)**	**50.2 ($\pm$4.0)**	**46.1 ($\pm$7.5)**

Table 3: Computational overheads of the multi-fidelity synthetic benchmarks of Figure 8. GIBBON enjoys the lowest overheads for all the tasks (as highlighted in bold), often less than half those of MUMBO.

function, we continue using the Gumbel sampler for all our remaining experiments. Investigating alternative sampling strategies to use within GIBBON is an important area of future work.

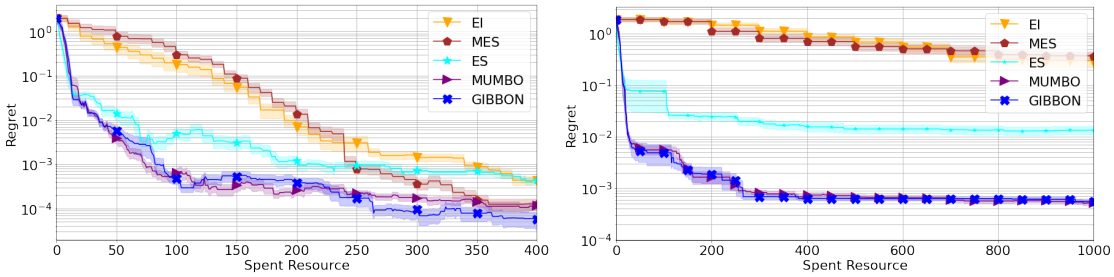
6.3 Multi-fidelity Optimisation

We now turn to multi-fidelity optimisation, where the current state-of-the-art acquisition functions are the effectively equivalent MUMBO (Moss et al., 2020d) and MFMES (Takeno et al., 2020) acquisition functions. Moss et al. (2020d) demonstrates comprehensively that MUMBO outperforms a wide range of existing multi-fidelity acquisition functions, including the entropy search-based approach of Swersky et al. (2013), the upper-confidence bound variants of Kandasamy et al. (2016) and Kandasamy et al. (2017), as well as extensions of EI (Huang et al., 2006) and KG (Wu and Frazier, 2016). Therefore, to test GIBBON's multi-fidelity optimisation capabilities, it is sufficient to compare with MUMBO. To this end, we provide an implementation of GIBBON for the Emukit Python library and recreate exactly the synthetic experiments from Figure 2 of Moss et al. (2020d). These experiments consider popular synthetic multi-fidelity benchmarks with discrete fidelity spaces consisting of between 2 and 4 fidelity levels (each with differing query costs) and search space dimensions ranging from 2 to 8 dimensions (see Appendix C.2 for the analytical forms of these synthetic benchmarks). In these experiments, we use the linear multi-fidelity GP model of Kennedy and O'Hagan (2000) as our surrogate model, initialise the GP with a random sample of $2 * d$ points queried across all fidelity levels, and fit the GP's kernel parameters to maximise model marginal likelihood after each BO step.

Figure 8 shows that GIBBON provides at least as effective optimisation as MUMBO and Table 3 shows that GIBBON has a significantly lighter computational overhead. To provide context for the high performance and low overhead of GIBBON we also present the performance of EI and MES when restricted to just querying the true objective function (i.e no access to low-fidelity observations) and the performance of the ES acquisition function, used to perform multi-fidelity optimisation by Swersky et al. (2013). Although the difference in overhead between MUMBO and GIBBON decreases as we consider higher-dimensional search spaces (primarily due to the growing cost of the Gumbel sampler used by both approaches), the difference in achieved regret increases in GIBBON's favour.

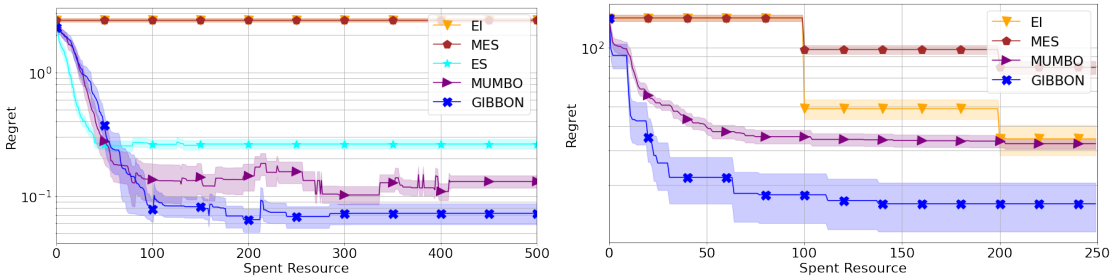
6.4 Batch Molecular Search

BO has recently been applied to high-cost string design problems by Moss et al. (2020b), who consider, among other problems, the task of optimising over molecules. Such tasks are well-suited for BO, due to the high cost of evaluating candidate molecules via wet-lab experiments. Moss



(a) Maximisation of the 2D Currin function (2 fidelity levels with evaluation costs 10 and 1).

(b) Minimisation of 3D Hartmann function (3 fidelity levels with evaluations costs 100, 10 and 1).



(c) Minimisation of 6D Hartmann function (4 fidelity levels).

(d) Maximisation of the 8D Borehole function (2 fidelity levels with evaluation costs 10 and 1).

Figure 8: GIBBON provides high-precision multi-fidelity optimisation with low computational overheads across a range of synthetic multi-fidelity benchmarks. Due to the high-cost of ES, we were not able to run it on the higher-dimensional Borehole task. As is standard in multi-fidelity optimisation, the x-axis for these results measures the resources spent on function evaluations (rather than raw BO steps).

et al. (2020b) propose a BO framework that fits a GP surrogate model to a popular string-based representation of molecules known as SMILES strings (Anderson et al., 1987) through a string kernel GP (Beck et al., 2015). Standard EI arguments are then applied, yielding a highly effective strategy for searching large candidate set of molecules. One practical limitation of this framework, however, is the large computational cost of string kernels, as incurred for each prediction from the surrogate model GP. Consequently, the framework of Moss et al. (2020b) is limited to acquisition functions that require a small number sof surrogate model predictions. Aside from GIBBON, our other considered high-performing batch acquisition functions (MFMES, NEI and KG) require many kernel evaluations for each acquisition function query and the low-cost approaches of DPPEI and LPEI are limited to only Euclidean search spaces. In contrast, GIBBON requires only B surrogate model predictions to measure the utility of a candidate batch and makes no assumptions on the properties of the search space. Therefore, GIBBON can be used to extend the framework of Moss et al. (2020b) to batch designs, a property particularly attractive for molecular search applications where it is common practice to synthesis collections of candidate molecules in parallel.

We now recreate the Zinc example (also considered by Kusner et al. (2017) and Griffiths and Hernández-Lobato (2020)), where we seek to explore a large collection of 250,000 molecules. The task is then to quickly find molecules that score highly according to a chemically-inspired metric, i.e. forming a proxy molecular design loop with this metric forming our objective function. As string kernel GPs, which are used to model our moelcules' SMILE strings, have a very large evaluation cost, we cannot evaluate our acquisition function across all the candidate molecules. Therefore, we randomly sample $1,000$ molecules for each BO step from which we (greedily) choose to evaluate the B molecules that maximise our GIBBON acquisition function.

We fit our Gumbel sampler on this same sample, re-sampling both the max-values required for GIBBON and the considered $1,000$ molecules at the start of each BO step. We evaluate 20 randomly chosen molecules to initialise our GP and then allow BO to choose 100 further molecules, either one by one or as 20 batches of 5 molecules or 10 batches of 10 molecules. Figure 9 shows that even in the purely sequential case, GIBBON provides a modest boost in performance over EI (the acquisition function previously used by Moss et al. (2020b)). More importantly, Figure 9 also shows that GIBBON is able to provide effective batch optimisation over batches of size 5 and 10, therefore providing an extension of Moss et al. (2020b)'s framework where parallel synthesising resources can be used to speed up the molecular search.

6.5 Bayesian Optimisation by Sampling Hierarchically

For our final set of examples, we demonstrate the efficacy of GIBBON as part of a real-world optimisation framework. In particular, we turn to a challenging batch multi-fidelity BO problem inspired by the Knowledge Gradient for Common Random Numbers (KG-CRN) framework of Pearce et al. (2019). We now provide a brief very overview of the KG-CRN framework and we refer the reader to Pearce et al. (2019) for further details. Our implementation is built upon the Emukit Python package (Paleyes et al., 2019) and was first reported in a workshop paper (Moss et al., 2020c).

KG-CRN considers BO under highly stochastic evaluations, a scenario where it is commonplace to disregard the original objective function entirely and instead optimise the average of a collection of K specific function realisations, e.g. K-fold cross validation (CV) (Kohavi, 1995) or sample average approximations (Kleywegt et al., 2002). However, as demonstrated for hyper-parameter tuning (Moss et al., 2018), model selection (Moss et al., 2019) and simulation optimisation (Kim

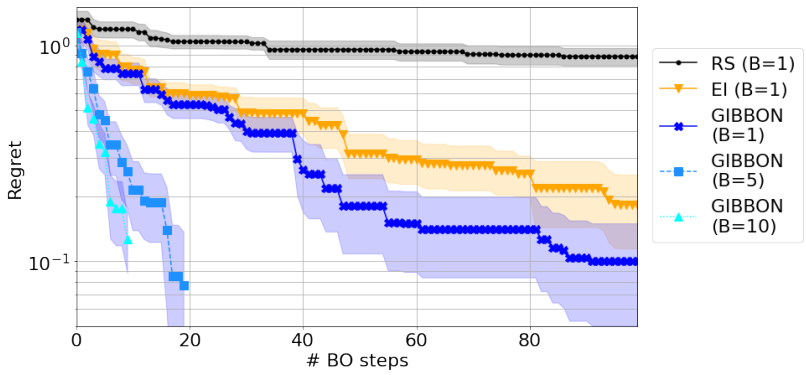
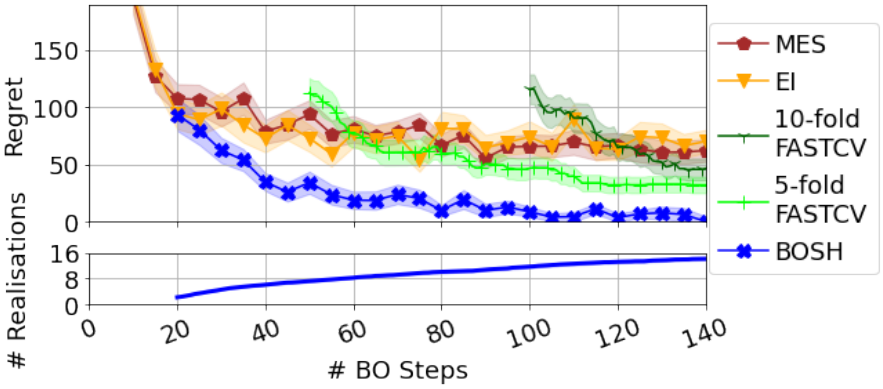


Figure 9: Exploring the Zinc database of molecules with GIBBON. In the purely sequential case, GIBBON finds higher-scoring molecules than EI. The batched GIBBON approaches reach roughly the same final regret after the same total number of 100 synthesised molecules even when GIBBON must choose these evaluations in batches of size 1, 5 or 10. Consequently, GIBBON is able to effectively leverage parallel synthesis resources, reaching the best solutions in fewer BO steps than non-batch alternatives. For context, we also report the performance of Random Search (RS).

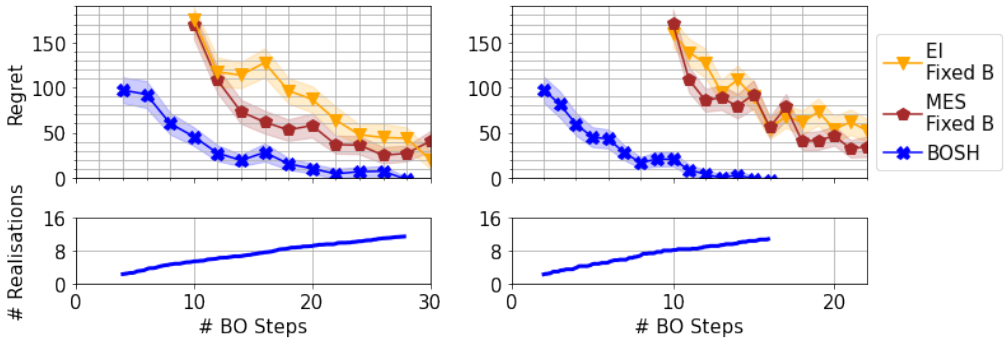
et al., 2015), optimisation efficiency depends subtly on the choice of K. If K is set too low we cannot optimise to high precision, however, setting K too large wastes computation on unnecessarily expensive evaluations. To avoid having to choose K *a-priori*, KG-CRN instead maintains a pool of randomly sampled realisations (e.g. train-test splits or initial environmental conditions) that grows as the optimisation progresses. This construction yields a multi-task BO framework where each individual realisation of the objective function is modelled separately as a perturbation of the true objective function through a Hierarchical Gaussian Process (HGP) (Hensman et al., 2013) (see Appendix F.1 for details). Consequently, KG-CRN not only chooses where to evaluate the objective function but also chooses which test problem in which to make the evaluations — either choosing a member of the previously considered pool of realisations or by generating an entirely new realisation (to be absorbed into the candidate pool for subsequent optimisation steps).

Unfortunately, KG-CRN's acquisition function, a variant of the knowledge gradient of Frazier et al. (2008), incurs a computational overhead that grows exponentially with the dimensions of the search space and does not support batch optimisation. By replacing this unwieldy acquisition function with GIBBON, we provide our own version of this framework, which we name Bayesian Optimisation Sampled Hierarchically (BOSH). Courtesy of GIBBON, BOSH enjoys small computational overheads and naturally supports batch decision making. We now demonstrate that the GIBBON-based BOSH framework can can provide more efficient and higher-precision optimisation than standard BO across reinforcement learning and hyper-parameter tuning tasks. Full experimental details are provided in Appendix F.

We report performance across a range of parallel computing resources ($B = 1, 5, 10$), comparing BOSH allocating batches of B points with standard BO routines using the EI and MES acquisition functions to optimise the average of B fixed realisations of the objective function. For these experiments, we measure regret as sub-optimality with respect to the best found solution across all methods and replicates. Unfortunately, Pearce et al. (2019) have yet to provide code for their KG-CRN approach, so we have been unable to provide direct comparisons. However, for the $B = 1$



(a) B=1



(b) B=5

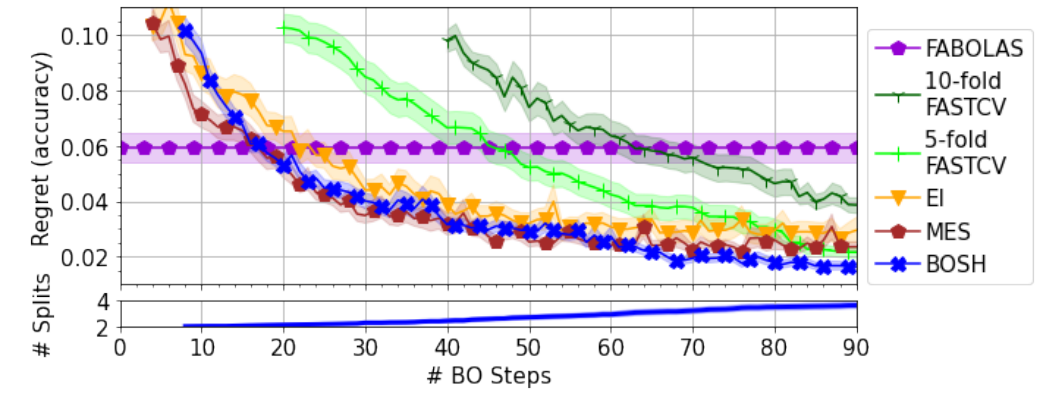
(c) B=10

Figure 10: Optimising 7 parameters of a Lunar Lander controller. We present the regret achieved by each algorithm (top panel) alongside a running count of the number of realisations considered by BOSH (bottom panel). Courtesy of our GIBBON acquisition function, BOSH is able to adaptive consider up to 15 random conditions to quickly find the optimal controller configuration.

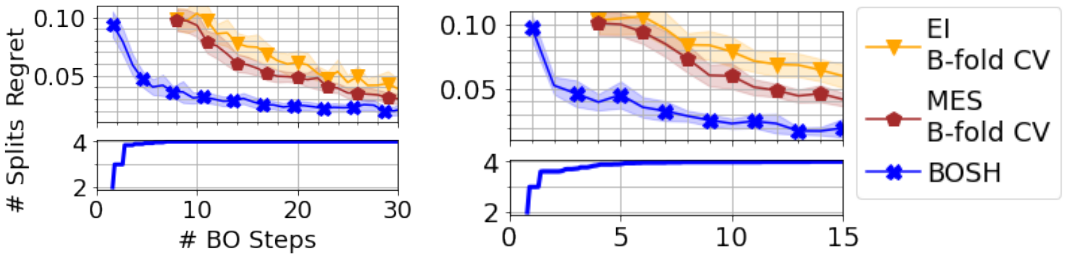
case, we are able to consider FASTCV (Swersky et al., 2013), an EI-based framework that speeds up optimisation by allowing the evaluation of the individual splits making up K-fold CV (for a specific choice of K). Unlike our previous examples, we now include the evaluations spent on random initialisation in our plots as the required size of this initialisation is different for each framework (see Appendix F.2). BOSH, for example, is initialised with evaluations at $d + 5$ random locations for each of two initial seeds.

6.5.1 REINFORCEMENT LEARNING

For our first experiment, we consider a challenging seven-dimensional stochastic optimisation test-case. We wish to fine-tune a controller for a well-studied reinforcement learning problem, where we must guide a lunar lander across a randomly initialised space to its landing zone with minimal thruster usage (as provided in the OpenAI Gym). Our controller is parameterised by seven unknown constants and a particular configuration can be tested by running a single (or B) randomly generated scenarios. We seek to minimise the extra fuel required to land the lander over OpenAI's hard-coded controller (as measured according to a 'true' performance measured over a set of 100 fixed initial



(a) B=1



(b) B=5

(c) B=10

Figure 11: Tuning SVM hyper-parameters for IMDB movie review classification with BOSH. BOSH achieves higher-precision optimisation than all other techniques. When batch computing resources are available, the batch capabilities of GIBBON allow BOSH to substantially improve optimisation efficiency over standard BO based on B-fold CV.

conditions). In this task, each objective function realisation corresponds to an initial environmental condition. As there is substantial variation across different initial conditions, optimising the controller over a small and fixed collection of initialisation fails to provide good 'true' performance according to the initial 100 condition test set (Figure 10). Note that FASTCV's need to initialise and then update the large between-realisation correlation matrix severely hampers its optimisation efficiency, as seen by the late start of the corresponding curves in Figure 10.

6.5.2 Hyper-parameter Tuning

We now test the performance of BOSH on a simple ML hyper-parameter tuning task: using a support vector machine (SVM) to classify the sentiment in IMDB movie reviews (Maas et al., 2011). Here, we seek hyper-parameter values that provide the highest model performance. True model performance is calculated on a large held-out test set. We stress that these high-cost estimates are only performed retrospectively, after stopping the optimisation, and during the actual tuning our individual performance estimates are generated using a pool of randomly generated train-test splits for BOSH or single train-test splits and K-fold CV as fixed evaluation strategies for standard BO. We also consider the multi-fidelity hyper-parameter tuning framework of FABOLAS (Klein et al., 2017a) (following the code provided in Klein et al. (2017b)). As FABOLAS is able to query models using only small

proportions of the available data, it is able to find reasonably well performing hyper-parameter configurations in a fraction of the computation used by standard BO and BOSH. However, even if allowed a significantly longer run-time, FABOLAS fails to improve upon this chosen configuration (which we plot as a horizontal line). Figure 11a shows that BOSH adaptively considers a pool of up to four train-test splits as the optimisation progresses, providing higher-precision tuning than standard BO based on single train-test splits and substantially faster tuning than standard BO under 5-fold and 10-fold cross-validation (Figures 11b and 11c).

7. Discussion

We have presented GIBBON, a general-purpose acquisition function that extends max-value entropy search to provide computationally light-weight yet high performing optimisation for a wide range of BO problems. The efficiency of GIBBON relies on a novel information-theoretical approximation. Moreover, the derivation of this approximation allowed the exploration of the first explicit connection between information-theoretic search, determinantal point process and local penalisation, tying together large sections of the BO literature previously developed and analysed independently.

Not only does GIBBON provide competitive optimisation for common BO extensions like batch and multi-fidelity optimisation, but it forms high-performance batch acquisition function suitable for applying BO across highly-structured search spaces, as we demonstrated within a molecular design loop. BO for structured optimisation tasks is a fast growing frontier of the BO literature, with recent work tackling BO for strings (Moss et al., 2020b; Swersky et al., 2020), combinatorial spaces (Deshwal et al., 2020) and spaces of neural network architectures (Kandasamy et al., 2018b). Therefore, we believe that GIBBON (and our flexible software implementation) will have substantial utility for the machine learning community.

7.1 Limitations and Future Work

GIBBON, in its current form, has the two primary practical limitations investigated in our ablation study of Section 6.2. Firstly, GIBBON performs poorly for large batch sizes. Improving the large batch capabilities, perhaps though artificially manipulating GIBBON's diversity-quality trade-off, is an important avenue of future work, particularly as the low-cost construction of GIBBON is especially well-suited to large batch scenarios which are currently dominated by simple sampling-based approaches like Thompson sampling (Vakili et al., 2020). Secondly, the performance of GIBBON is sensitive to the quality of the max-value samples used within its calculation strategy. Although using a Gumbel sampler to calculate GIBBON provides a truly light-weight acquisition function, we have shown that the performance of the acquisition function can be improved by considering exact max-value samples. In future work, we will investigate alternative sampling strategies that are more accurate than Gumbel samplers but cheaper than exact Thompson sampling. A promising approach is to follow Hernández-Lobato et al. (2016) or Takeno et al. (2020) and employ approximate Thompson sampling methods though kernel decompositions.

Although shown to be empirically successful, GIBBON has no theoretical guarantees, primarily due to the lack of analysis around our information-theoretical lower bound. Analysing the tightness of this bound could help disentangle which aspects of GIBBON's behaviour are caused by approximation error and which are due to limitations of information-theoretic search strategies in general. In particular, a stronger understanding of approximation quality should explain why GIBBON's performance degrades when building large batches or show exactly when our lower bound is a

better approximation of the mutual information than the sampling-based approximations of existing MES extensions. A bound on the approximation error of this bound would also pave the way for convergence guarantees for GIBBON through extensions of the regret bounds of Wang and Jegelka (2017). To the author's knowledge, no such bound exists for noisy, batch or multi-fidelity MES-based acquisition functions.

As a final comment, we would like to point out that, although we have already shown GIBBON to have wide applicability, GIBBON can be readily applied to an even wider collection of BO problems. For example, GIBBON can be combined with MESMO (Belakaria et al., 2019), an extension of MES for multi-objective optimisation, to provide computationally light-weight acquisition function for batch multi-objective BO. Similarly, GIBBON can also provide a computationally light-weight approach for batch constrained optimisation by extending the MES-based approach of Belakaria et al. (2020). Finally, GIBBON can be used to improve the performance and reduce the computational cost of any framework relying on batch BO heuristics, for example in non-myopic BO (González et al., 2016b; Jiang et al., 2020).

Appendix A. Extracting The Required Predictive Quantities from a Gaussian Process Surrogate Model

We now demonstrate how the distributional quantities required to calculate GIBBON can easily be extracted from a GP surrogate model. For observations D_n, let $\mathbf{y}_n$ be the already observed evaluations y , and define the kernel matrix $\mathbf{K}_n = [k(\mathbf{z}_i, \mathbf{z}_j)]_{\mathbf{z}_i, \mathbf{z}_j \in D_n}$ and kernel vectors $\mathbf{k}_n(\mathbf{z}) = [k(\mathbf{z}_i, \mathbf{z})]_{\mathbf{z}_i \in D_n}$ for any valid kernel defined over the combined search space $\mathcal{Z} = \mathcal{X} \times \mathcal{F}$. Finally, denote the location in the fidelity space corresponding to the true objective function as $\mathbf{s}_0$ (i.e $f_{\mathbf{s}_0}(\mathbf{x}) = g(\mathbf{x})$). Here, as is standard in multi-fidelity optimisation, we have assumed the ability to query (at least noisily) the true objective function. Then, following Rasmussen (2004) our GP surrogate model provides the following:

$$
\begin{aligned}
\mu_i^C =& \mathbf{k}_n((\mathbf{x}_i, \mathbf{s}_0))^T (\mathbf{K}_n + \text{diag}(\boldsymbol{\sigma}_n))^{-1} \mathbf{y}_n \\
\mu_i^A =& \mathbf{k}_n(\mathbf{z}_i)^T (\mathbf{K}_n + \text{diag}(\boldsymbol{\sigma}_n))^{-1} \mathbf{y}_n \\
\Sigma_{i,j}^C =& k((\mathbf{x}_i, \mathbf{s}_0), (\mathbf{x}_j, \mathbf{s}_0)) - \mathbf{k}_n((\mathbf{x}_i, \mathbf{s}_0))^T (\mathbf{K}_n + \text{diag}(\boldsymbol{\sigma}_n))^{-1} \mathbf{k}_n((\mathbf{x}_j, \mathbf{s}_0)) \\
\Sigma_{i,j}^A =& k(\mathbf{z}_i, \mathbf{z}_j) - \mathbf{k}_n(\mathbf{z}_i)^T (\mathbf{K}_n + \text{diag}(\boldsymbol{\sigma}_n))^{-1} \mathbf{k}_n(\mathbf{z}_j) \\
\rho_i =& \frac{k(\mathbf{z}_i, (\mathbf{x}_i, \mathbf{s}_0)) - \mathbf{k}_n(\mathbf{z}_i)^T (\mathbf{K}_n + \text{diag}(\boldsymbol{\sigma}_n))^{-1} \mathbf{k}_n((\mathbf{x}_i, \mathbf{s}_0)))}{\sqrt{\Sigma_{i,i}^g \Sigma_{i,i}^y}},
\end{aligned}
$$

where $\text{diag}(\boldsymbol{\sigma_n})$ is the $|D_n| \times |D_n|$ diagonal matrix of observation noises in the evaluations D_n.

Appendix B. Proof of Theorem 2

Theorem 2 (Distribution of A given $C^* < m$) *Consider two B-dimensional multivariate Gaussian random variables $\boldsymbol{A}$ and $\boldsymbol{C}$ where $\boldsymbol{C} \sim N(\boldsymbol{\mu}^C, \Sigma^C)$ and each individual component of $\boldsymbol{A}$ is distributed as $A_j \sim N(\mu_j^A, \Sigma_{j,j}^A)$. Suppose further that each each pair $\{A_j, C_j\}$ are jointly Gaussian with correlation ρ_j, and that each A_j is conditionally independent of $\{C_i\}_{i \neq j}$ given C_j. Define $C^* = \max \boldsymbol{C}$. Then the conditional density of $\boldsymbol{A}$ given that $C^* < m$ is given by*

$$
\frac{1}{\mathbb{P}(C^* < m)} \phi_{\boldsymbol{X}_1}(\boldsymbol{a}) \Phi_{\boldsymbol{X}_2}(\boldsymbol{m}),
$$

where $\boldsymbol{m} = (m, .., m) \in \mathbb{R}^B$ and $\phi_{\boldsymbol{X}_1}$ and $\Phi_{\boldsymbol{X}_2}$ are the probability density and cumulative density functions for the multivariate Gaussian random variables

$$
\boldsymbol{X}_1 \sim \boldsymbol{N}\left(\boldsymbol{\mu}^A, S + D\Sigma^C D\right) \quad \textit{and} \quad \boldsymbol{X}_2 \sim \boldsymbol{N}\left(\boldsymbol{\mu}^C + \Sigma^{-1} D S^{-1}(\boldsymbol{a} - \boldsymbol{\mu}^A), \Sigma^{-1}\right),
$$

where $\Sigma^A = D\Sigma^C D + S$ for D and S, diagonal matrices with elements $D_{j,j} = \rho_j \sqrt{\frac{\Sigma_{j,j}^A}{\Sigma_{j,j}^C}}$ and $S_{j,j} = (1 - \rho_j^2)\Sigma_{j,j}^A$, and $\Sigma = \left(\left(\Sigma^C\right)^{-1} + DS^{-1}D\right)$.

Proof

As detailed in the main body of this report, we have that

$$
\mathbf{C} \sim N(\boldsymbol{\mu}^C, \Sigma^C) \quad \text{and} \quad A_j \sim N_1\left(\mu_j^A, \Sigma_j^A\right),
$$

for some known mean vectors $\boldsymbol{\mu}^C, \boldsymbol{\mu}^A \in \mathbb{R}^B$, a variance vector $\boldsymbol{\Sigma}^A \in \mathbb{R}^B$ and a co-variance matrix $\Sigma^C \in \mathbb{R}^{B\times B}$, as well as a vector $\boldsymbol{\rho} \in \mathbb{R}^B$ of the correlation between each pair $\{A_j, C_j\}$. In this section we use $f_{\mathbf{X}}$ to denote the probability density function for the random variable $\mathbf{X}$ and $f_{\mathbf{X},\mathbf{Y}}$ to denote the joint probability density function for the random variables $\mathbf{X}$ and $\mathbf{Y}$.

Now, consider the probability distribution function of random variable of interest:

$$\begin{aligned} f_{\mathbf{A}|C^*\leq m}(\mathbf{a}) =& \frac{1}{\mathbb{P}(C^* \leq m)} \int^{\mathbf{m}} f_{\mathbf{A},\mathbf{C}}(\mathbf{a},\mathbf{b})\, d\mathbf{b} \\ =& \frac{1}{\mathbb{P}(C^* \leq m)} \int^{\mathbf{m}} f_{\mathbf{A}|\mathbf{C}=\mathbf{b}}(\mathbf{a}) f_{\mathbf{C}}(\mathbf{b})\, d\mathbf{b} \\ =& \frac{1}{\mathbb{P}(C^* \leq m)} \int^{\mathbf{m}} \prod_{i=1}^{B} \left[f_{A_i|C_i=b_i}(a_i)\right] f_{\mathbf{C}}(\mathbf{b})\, d\mathbf{b}, \end{aligned} \tag{10}$$

where $\mathbf{b} \in \mathbb{R}^B$ and $\mathbf{m} = (m, .., m) \in \mathbb{R}^B$. The factorisation of $f_{\mathbf{A}|\mathbf{C}=\mathbf{b}}$ is due to the conditional independence of $A_j|C_j$ from $\{C_i\}_{i\neq j}$.

A well-known result for the conditional distribution from a bi-variate Gaussian gives us that for each $i \in \{1, .., B\}$

$$A_i = a_i | C_i = b_i \sim N_1\left(\mu_i^A + \rho_i\sqrt{\frac{\Sigma_i^A}{\Sigma_{i,i}^C}}(b_i - \mu_i^C), (1-\rho_i^2)\Sigma_i^A\right),$$

i.e. we have that

$$\mathbf{A}|\mathbf{C} = \mathbf{b} \sim N\left(\boldsymbol{\mu}^A + D(\mathbf{b} - \boldsymbol{\mu}^C), S\right), \tag{11}$$

for diagonal matrices $D, S \in \mathbb{R}^B$ with elements $D_{i,i} = \rho_i\sqrt{\frac{\Sigma_i^A}{\Sigma_{i,i}^C}}$ and $S_{i,i} = (1-\rho_i^2)\Sigma_i^A$.

Using (11), the integrand of (10) can now be regarded as the product of two b-dimensional Gaussian densities

$$\begin{aligned} \left[\prod_{i=1}^{b} f_{A_i|C_i=b_i}(a_i)\right] f_{\mathbf{C}}(\mathbf{b}) &= N\left(\mathbf{a}; \boldsymbol{\mu}^A + D(\mathbf{b} - \boldsymbol{\mu}^C), S\right) * N(\mathbf{b}; \boldsymbol{\mu}^C, \Sigma^C) \\ &= |D| N\left(\mathbf{b}; \boldsymbol{\mu}^C + D^{-1}(\mathbf{a} - \boldsymbol{\mu}^A), D^{-1}SD^{-1}\right) * N(\mathbf{b}; \boldsymbol{\mu}^C, \Sigma^C), \end{aligned}$$

which, using the following standard formula for the product of Gaussians densities

$$\begin{aligned} N(\mathbf{x}; \mathbf{m}_1, \Sigma_1) * N(\mathbf{x}; \mathbf{m}_2, \Sigma_2) =& N(\mathbf{m}_1; \mathbf{m}_2, \Sigma_1 + \Sigma_2) \\ & * N(\mathbf{x}; \left(\Sigma_1^{-1} + \Sigma_2^{-1}\right)^{-1}\left(\sigma_1^{-1}\mathbf{m}_1 + \Sigma_2^{-1}\mathbf{m}_2\right), \left(\Sigma_1^{-1} + \Sigma_2^{-1}\right)^{-1}), \end{aligned}$$

can be re-expressed as

$$\begin{aligned}\left[\prod_{i=1}^{b} f_{A_i|C_i=b_i}(a_i)\right] f_{\mathbf{C}}(\mathbf{b}) &= |D|N\left(\boldsymbol{\mu}^C; \boldsymbol{\mu}^C + D^{-1}(\mathbf{a}-\boldsymbol{\mu}^A), D^{-1}SD^{-1} + \Sigma^C\right) \\ &\quad * N\left(\mathbf{b}; \boldsymbol{\mu}^C + \Sigma^{-1}DS^{-1}(\mathbf{a}-\boldsymbol{\mu}^A), \Sigma^{-1}\right) \\ &= N\left(\mathbf{a}; \boldsymbol{\mu}^A, S + D\Sigma^C D\right) \\ &\quad * N\left(\mathbf{b}; \boldsymbol{\mu}^C + \Sigma^{-1}DS^{-1}(\mathbf{a}-\boldsymbol{\mu}^A), \Sigma^{-1}\right)\end{aligned}$$

where $\Sigma = \left(\left(\Sigma^C\right)^{-1} + DS^{-1}D\right)$.

Therefore, we have rewritten the integrand of (10) as a product of two Gaussian densities, where only one depend on $\mathbf{b}$. Consequently, the first Gaussian term can be taken outside the integral, yielding the claimed expression

$$f_{\mathbf{A}|C^*<m}(\mathbf{a}) = \frac{1}{\mathbb{P}(C^* < m)}\phi_{\mathbf{X}_1}(\mathbf{a})\Phi_{\mathbf{X}_2}(\mathbf{m}), \tag{12}$$

where $\phi_{\mathbf{X}_1}$ and $\Phi_{\mathbf{X}_2}$ are the probability density and cumulative density functions for the multivariate Gaussian variables

$$\mathbf{X}_1 \sim \mathbf{N}_b\left(\boldsymbol{\mu}^A, S + D\Sigma^C D\right) \quad \text{and} \quad \mathbf{X}_2 \sim \mathbf{N}_b\left(\boldsymbol{\mu}^C + \Sigma^{-1}DS^{-1}(\mathbf{a}-\boldsymbol{\mu}^A), \Sigma^{-1}\right).$$

■

Appendix C. Experimental Details for Synthetic Benchmarks.

We now provide detailed information about each of our synthetic benchmarks.

C.1 Standard BO benchmarks

Shekel function. A four-dimensional function with ten local and one global minima defined on $\mathcal{X} \in [0, 10]^4$:

$$f(\mathbf{x}) = -\sum_{i=1}^{10}\left(\sum_{j=1}^{4}(x_j - A_{j,i})^2 + \beta_i\right)^{-1},$$

where

$$\beta = \begin{pmatrix}1\\2\\2\\4\\4\\6\\3\\7\\5\\5\end{pmatrix} \quad \text{and} \quad A = \begin{pmatrix}4&1&8&6&3&2&5&8&6&7\\4&1&8&6&7&9&3&1&2&3.6\\4&1&8&6&3&2&5&8&6&7\\4&1&8&6&7&9&3&1&2&3.6\end{pmatrix}.$$

Ackley function. A four-dimensional function with many local minima and a nearly flat outer region surrounding a single global minima defined on $\mathcal{X} \in [-32.768, 32.768]^4$:

$$f(\mathbf{x}) = -20 \exp\left(-0.2 * \sqrt{\frac{1}{4}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{4}\sum_{i=1}^{4}\cos(2\pi x_i)\right) + 20 + \exp(1).$$

Hartmann 6 function. A six-dimensional function with six local minima and a single global minima defined on $\mathcal{X} \in [0, 1]^6$:

$$f(\mathbf{x}) = -\sum_{i=1}^{4} \alpha_i \exp\left(-\sum_{j=1}^{6} A_{i,j}(x_j - P_{i,j})^2\right),$$

where

$$A = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix}, \qquad \alpha = \begin{pmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{pmatrix},$$

$$P = 10^{-4} \begin{pmatrix} 1312 & 1696 & 5569 & 124 & 8283 & 5886 \\ 2329 & 4135 & 8307 & 3736 & 1004 & 9991 \\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650 \\ 4047 & 8828 & 8732 & 5743 & 1091 & 381 \end{pmatrix}.$$

C.2 Multi-fidelity benchmarks

Currin exponential function (discrete fidelity space). A two-dimensional function defined on $\mathcal{X} = [0, 1]^2$ with two fidelities queried with costs 10 and 1:

$$\begin{aligned} f(x_1, x_2, 0) =& \left(1 - \exp(-\frac{1}{2x_2})\right) \frac{2300x_1^3 + 1900x_1^2 + 2092x_1 + 60}{100x_1^3 + 500x_1^2 + 4x_1 + 20} \\ f(x_1, x_2, 1) =& \frac{1}{4} f(x_1 + 0.05, x_2 + 0.05, 0) \\ &+ \frac{1}{4} f(x_1 + 0.05, max(0, x_2 - 0.05), 0) \\ &+ \frac{1}{4} f(x_1 - 0.05, x_2 + 0.05, 0) \\ &+ \frac{1}{4} f(x_1 - 0.05, max(0, x_2 - 0.05), 0). \end{aligned}$$

Hartmann 3 function. A three-dimensional function with 4 local extrema defined on $\mathcal{X} = [0, 1]^3$ with three fidelities ($m = 0, 1, 2$) queried at costs $100, 10$ and 1:

$$f(x_1, x_2, x_3, m) = -\sum_{i=1}^{4} \alpha_{i,m+1} \exp\left(-\sum_{j=1}^{3} A_{i,j}(x_j - P_{i,j})^2\right),$$

where

$$A = \begin{pmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{pmatrix}, \quad \alpha = \begin{pmatrix} 1 & 1.01 & 1.02 \\ 1.2 & 1.19 & 1.18 \\ 3 & 2.9 & 2.8 \\ 3.2 & 3.3 & 3.4 \end{pmatrix}, \quad P = \begin{pmatrix} 3689 & 1170 & 2673 \\ 4699 & 4387 & 7470 \\ 1091 & 8732 & 5547 \\ 381 & 5743 & 8828 \end{pmatrix}.$$

Hartmann 6 function. A six-dimensional function defined on $\mathcal{X} = [0, 1]^6$ with four fidelities ($m = 0, 1, 2, 3$) queried at costs $1000, 100, 10$ and 1:

$$f(x_1, x_2, x_3, x_4, x_5, x_6, m) = -\sum_{i=1}^{4} \alpha_{i,m+1} \exp\left(-\sum_{j=1}^{6} A_{i,j}(x_j - P_{i,j})^2\right),$$

where

$$A = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix}, \qquad \alpha = \begin{pmatrix} 1 & 1.01 & 1.02 & 1.03 \\ 1.2 & 1.19 & 1.18 & 1.17 \\ 3 & 2.9 & 2.8 & 2.7 \\ 3.2 & 3.3 & 3.4 & 3.5 \end{pmatrix},$$

$$P = 10^{-4} \begin{pmatrix} 1312 & 1696 & 5569 & 124 & 8283 & 5886 \\ 2329 & 4135 & 8307 & 3736 & 1004 & 9991 \\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650 \\ 4047 & 8828 & 8732 & 5743 & 1091 & 381 \end{pmatrix}.$$

Borehole function. An eight-dimensional function defined on

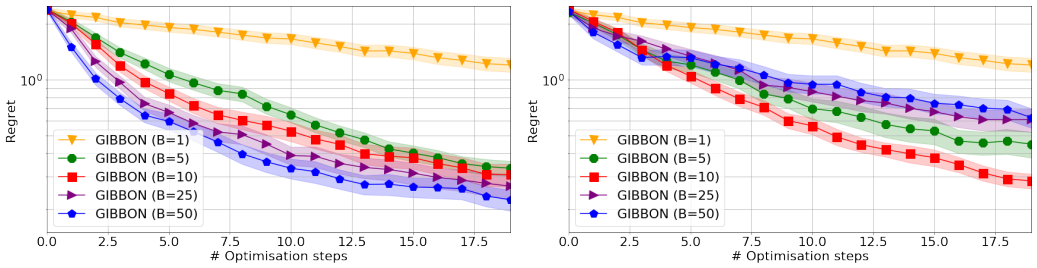
$$\begin{aligned} \mathcal{X} = [&0.05, 0.15; 100, 50,000; 63070, 115600; 990, \\ &1110; 63.1, 116; 700, 820; 1120, 1680; 9855, 12055] \end{aligned}$$

with two fidelities queried with costs 10 and 1:

$$\begin{aligned} f(\mathbf{x}, 0) =& \frac{2\pi x_3(x_4 - x_6)}{\log(x_2/x_1)\left(1 + \frac{2x_7x_3}{log(x_2/x_1)x_1^2x_8} + \frac{x_3}{x_5}\right)}, \\ f(\mathbf{x}, 1) =& \frac{5x_3(x_4 - x_6)}{\log(x_2/x_1)\left(1.5 + \frac{2x_7x_3}{log(x_2/x_1)x_1^2x_8} + \frac{x_3}{x_5}\right)}. \end{aligned}$$

Appendix D. A modified GIBBON for BO with large batches

In Section 6.2, we demonstrated that GIBBON fails to effectively control large batches ($B >> 10$) and hypothesised that this was due to a dominance of GIBBON's diversity term over its quality term (8) in these large batch regimes. To support this hypothesis and to propose a variant of GIBBON suitable for large batches, we now investigate a simple modification to GIBBON. In particular, we down-weight GIBBON's diversity term by a factor of B^2, with this scaling chosen to reflect the B^2



(a) Standard GIBBON (b) Modified

Figure 12: Optimisation of the noisy Hartmann function over 20 iterations across a range of batch sizes. Modified GIBBON (left) is able to effectively allocate even the largest batches ($B = 50$), achieving faster convergence for each increase in batch size. In contrast, standard GIBBON (right) fails to control batches of size $B > 10$.

elements present in the the predictive co-variance of a candidate batch of size B. Therefore, we have the modified GIBBON acquisition function

$$\alpha_n(\{\mathbf{z}\}_{i=1}^B) = \frac{1}{2B^2} \log \left|R\right| + \sum_{i=1}^{B} \alpha_n^{\text{GIBBON}}(\mathbf{z}_i)$$

with performance demonstrated in Figure 12, which repeats the experiment of Section 6.2. We see that this simple re scaling is all that is required to allow GIBBON to effectively control large parallel resources.

Appendix E. Comparing GIBBON with MES

In our synthetic experiments of Section 6, we were surprised to see that GIBBON was able to outperform MES even in the noiseless standard BO case for which MES provides an exact calculation of entropy reductions. As GIBBON approximates MES, we actually expected GIBBON to perform strictly worse than MES in this particular setting. However, we stress that although GIBBON is designed to approximate MES, they are still distinct acquisition functions with differing analytical expressions (see Definition 4 and Equation 2). Consequently, MES and GIBBON induce (potentially slightly) different exploration-exploitation trade-offs, with the behaviour of GIBBON being particularly well-suited to the Shekel and Ackley, but not Hartmann functions (see Figure 3).

In the specific case (not used in practice) where where we base our MES and GIBBON calculations on a single max-value sample, we can show that GIBBON and MES always choose the same query points (see Section E.1). However this equivalence does not hold for practical implementations of GIBBON and MES (where we typically use 5 or 10 samples of g^*)

E.1 Equivalence of the degenerate forms of MES and GIBBON

To gain further intuition about the relationship between MES and GIBBON, we analyse the so-called degenerate forms their acquisition functions. In the degenerate setting, the acquisition functions are

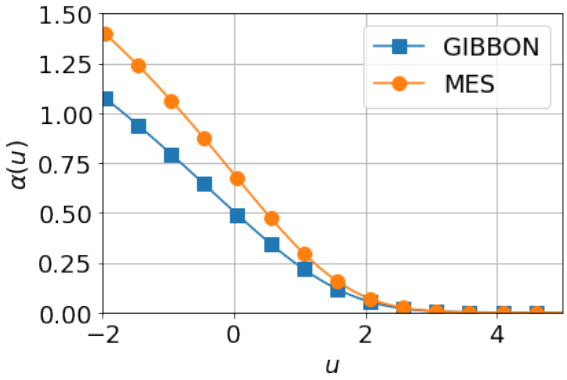


Figure 13: Degenerate GIBBON and MES as functions of u (the standardised difference between the GP posterior at a candidate point and the current estimated maximum value). The two acquisition functions are monotonically decreasing, taking the same maximiser across a given range of u values.

built using only a single max-value sample. By defining the function $u(\mathbf{x}) = \frac{m^* - \mu_n^g(\mathbf{x})}{\sqrt{\Sigma^g(\mathbf{x})}}$, degenerate GIBBON and MES can be expressed as

$$\alpha_n^{\text{GIBBON}}(\mathbf{x}) = -\log\left(1 - \frac{\phi(u(\mathbf{x}))}{\Phi(u(\mathbf{x}))}\left(u(\mathbf{x}) + \frac{\phi(u(\mathbf{x}))}{\Phi(u(\mathbf{x}))}\right)\right)$$
$$\alpha_n^{\text{MES}}(\mathbf{x}) = \frac{u(\mathbf{x})\phi(u(\mathbf{x}))}{2\Phi(u(\mathbf{x}))} - \log\Phi(u(\mathbf{x})).$$

Although taking very different analytical forms, these two acquisition functions are strictly decreasing in u (as shown in Figure 13), with GIBBON a strict lower bound on MES. So, in this degenerate and noiseless setting, GIBBON and MES would choose exactly the same points under given exact inner-loop maximisation.

Note that in this degenerate setting, Wang and Jegelka (2017) provide a bound on the simple regret of degenerate MES. As degenerate GIBBON and degenerate MES choose the same query points, the regret bound of degenerate MES is also inherited by degenerate GIBBON. Although this result does not hold for practical implementations of GIBBON based on multiple samples of g^*, or when we perform batch or multi-fidelity BO, the existence of this theoretical guarantee provides reassuring evidence for the validity of our approach.

Appendix F. Experimental Details for BOSH

We now provide additional details about our implementation of BOSH and the exact set-ups of our experiments.

F.1 Hierarchical Gaussian Process

A natural framework for modelling function realisations as perturbations of a true objective function is a Hierarchical Gaussian Process (HGP) (Hensman et al., 2013), where the true objective function is modelled as a GP with an 'upper' kernel k_g, and the deviations to all the individual realisations f_s modelled by another GP with a 'lower' kernel k_f. As is common in BO, we use Matérn 5/2 kernels

(Matérn, 1960). The HGP structure is equivalently understood as each f_s being a conditionally independent GPs with shared mean function g, i.e.

$$y_i = f_{s_i}(\mathbf{x}_i) + \epsilon_i \quad \text{for} \quad f_s \sim \mathcal{GP}(g, k_f) \quad \text{where} \quad g \sim \mathcal{GP}(0, k_g),$$

for $\epsilon_i \overset{\text{i.i.d}}{\sim} \mathcal{N}(0, \sigma^2)$. This induces a prior covariance structure of

$$Cov(f_s(\mathbf{x}), f_{s'}(\mathbf{x}'))) = k_g(\mathbf{x}, \mathbf{x}') + \mathbb{I}_{s=s'} k_f(\mathbf{x}, \mathbf{x}') \quad \text{and} \quad Cov(f_s(\mathbf{x}), g(\mathbf{x}')) = k_g(\mathbf{x}, \mathbf{x}'),$$

where $\mathbb{I}$ is an indicator function.

F.1.1 Predictive Distribution of an HGP

Crucially, given observations $D_n = \{(\mathbf{x}_i, s_i, y_i)\}_{i=1}^n$, the HGP provides a bi-variate Gaussian joint distribution for $(y_s(\mathbf{x}), g(\mathbf{x})) \mid D_n$, the quantities required to evaluate GIBBON. We will now provide closed form expressions for this joint predictive distributions of $g(\mathbf{x})$ and $y_s(\mathbf{x})$ given a set of collected evaluations $D_n = \{(\mathbf{x}_i, s_i, y_i)\}_{i=1}^n$ (location-realisation-evaluations tuples), where $y_i = f_{s_i}(\mathbf{x}_i) + \epsilon$ under Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$.

Defining a compound kernel $\tilde{k}$ (defined over $\mathcal{X} \times S$) as $\tilde{k}((\mathbf{x}, s), (\mathbf{x}', s')) = k_g(\mathbf{x}, \mathbf{x}') + \mathbb{I}_{s=s'} k_f(\mathbf{x}, \mathbf{x}')$ and following Rasmussen (2004) and Hensman et al. (2013), our joint posterior distribution can be written as

$$\begin{pmatrix} g(\mathbf{x}) \\ y_s(\mathbf{x}) \end{pmatrix} \,\middle|\, D_n \sim N\left[\begin{pmatrix} \mu_n^g(\mathbf{x}) \\ \mu_n(\mathbf{x}, s) \end{pmatrix}, \begin{pmatrix} \sigma_n^{g2}(\mathbf{x}) & \Sigma_n(\mathbf{x}, s) \\ \Sigma_n(\mathbf{x}, s) & \sigma_n^2(\mathbf{x}, s) + \sigma^2 \end{pmatrix} \right],$$

where

$$\begin{aligned}
\mu_n(\mathbf{x}, s) =& \tilde{\mathbf{k}}_n((\mathbf{x}, s))^T \left(\tilde{\mathbf{K}}_n + \sigma^2 I_n\right)^{-1} \mathbf{y}_n \\
\mu_n^g(\mathbf{x}) =& \mathbf{k}_n^g((\mathbf{x}, s))^T \left(\tilde{\mathbf{K}}_n + \sigma^2 I_n\right)^{-1} \mathbf{y}_n \\
\sigma_n^2(\mathbf{x}, s) =& \tilde{k}\left((\mathbf{x}, s), (\mathbf{x}, s)\right) - \tilde{\mathbf{k}}_n((\mathbf{x}, s))^T \left(\tilde{\mathbf{K}}_n + \sigma^2 I_n\right)^{-1} \tilde{\mathbf{k}}_n((\mathbf{x}, s)) \\
\sigma_n^{g2}(\mathbf{x}) =& k^g\left(\mathbf{x}, \mathbf{x}\right) - \mathbf{k}_n^g(\mathbf{x})^T \left(\tilde{\mathbf{K}}_n + \sigma^2 I_n\right)^{-1} \mathbf{k}_n^g(\mathbf{x}) \\
\Sigma_n(\mathbf{x}, s) =& k^g\left((\mathbf{x}, s), (\mathbf{x}, s)\right) - \tilde{\mathbf{k}}_n((\mathbf{x}, s))^T \left(\tilde{\mathbf{K}}_n + \sigma^2 I_n\right)^{-1} \mathbf{k}_n^g(\mathbf{x}),
\end{aligned}$$

for $\tilde{\mathbf{K}}_n = \left[\tilde{k}((\mathbf{x}_i, s_i), (\mathbf{x}_j, s_j))\right]_{i,j=1,..,n}$, $\tilde{\mathbf{k}}_n((\mathbf{x}, s)) = \left[\tilde{k}((\mathbf{x}_i, s_i), (\mathbf{x}, s))\right]_{i=1,..,n}$, $\mathbf{k}_n^g(\mathbf{x}) = \left[k_g(\mathbf{x}_i, \mathbf{x})\right]_{i=1,..n}$ and $\mathbf{y} = [y_i]_{i=1,..,n}$.

Note that predicting from our HGP requires the inversion of the $n \times n$ matrix $\tilde{\mathbf{K}}_n + \sigma^2 I_n$ and so has comparable cost to predictions from standard GPs.

F.1.2 BOSH's Kernel Structure

Our implementation of BOSH uses the following structure for the upper and lower kernels of the HGP:

$$\begin{aligned}
k_g(\mathbf{x}, \mathbf{x}') &= k_{\alpha_g, \beta}(\mathbf{x}, \mathbf{x}') \\
k_f(\mathbf{x}, \mathbf{x}') &= k_{\alpha_f, \beta}(\mathbf{x}, \mathbf{x}') + \sigma_f^2,
\end{aligned}$$

where $k_{\alpha,\beta}$ denotes the Matérn 5/2 (Matérn, 1960) kernel with variance $\alpha \in \mathbb{R}$ term and length scales $\beta \in \mathbb{R}^d$ hyper-parameters, i.e

$$k_{\alpha,\beta}(\mathbf{x}, \mathbf{x}') = \alpha(1 + \sqrt{5}d_\beta(\mathbf{x}, \mathbf{x}') + \frac{5}{3}d_\beta(\mathbf{x}, \mathbf{x}')^2)e^{-\sqrt{5}d_\beta(\mathbf{x},\mathbf{x}')},$$

for a weighted distance measure $d_\beta(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^T diag(\beta)(\mathbf{x} - \mathbf{x}')$.

As the length-scales are shared between the lower and upper kernels, the total number of kernel parameters for BOSH (including the scale of observation noise σ^2in our Gaussian likelihood) is $d + 4$, only two more than a standard GP with a Matérn 5/2 kernel.

F.2 Initialisation Costs

Before beginning any BO routine, we must collect an initialisation of points to fit the surrogate model. To allow stable maximisation of the marginal likelihood, it is common to initialise with at least as many evaluations as unknown kernel parameters (to guarantee identifiability). For standard BO, this corresponds to $d + 3$ evaluations of the chosen evaluation strategy (i.e requiring $B * (d + 3)$ individual function evaluations). For BOSH, rather than using separate lower and upper kernels for our HGP, we found that tying length-scales between each kernel greatly improved the stability of the HGP. Therefore, our HGP has $d + 4$ kernel parameters. we allowed BOSH $d + 5$ evaluations for each of the seeds in an initial seed pool with two elements. In contrast, reliable initialisation of FASTCV's $B \times B$ correlation matrix (of which its performance was very sensitive) required at least $d + 3$ evaluations for each of its B considered seeds. We found that using fewer initial points severely limited the initial performance of all these methods. Therefore, as well as providing improved efficiency and precision once optimisation begins, BOSH's ability to model only as many individual seeds as required allows significantly lower initialisation costs.

F.3 Reinforcement Learning: Lunar Lander

The Lunar Lander problem is a well-known reinforcement learning task, where we must control three engines (left, main and right) to successfully land a rocket. The learning environment and a hard-coded PID controller is provided in the OpenAI gym [1]. We seek to optimize the 7 thresholds present in the description of the controller to provide the largest average reward over 100 random initial conditions. Our RL environment is exactly as provided by OpenAI, with the small modification of randomly initializing the initial lander location (as-well as random initial velocities and terrain) to make a more challenging stochastic optimization problem. We lose 0.3 points per second of fuel use and 100 if we crash. We gain 10 points each time a leg makes contact with the ground, 100 points for any successful landing, and 200 points for a successful landing in the specified landing zone· Each individual run of the environment allows the testing of a controller on a specific random seed.

F.4 Hyper-parameter Tuning: IMDB SVM

We tested the performance of BOSH on a real ML problem: tuning a sentiment classification model on the collection of $25,000$ positive and $25,000$ negative IMDB movie reviews used by Maas et al. (2011), seeking the hyper-parameter values that provide the model with the highest accuracy. We tune the flexibility of the decision boundary (C) and the RBF kernel coefficient (gamma) for an SVM

1. *https://gym.openai.com/*

(Cortes and Vapnik, 1995), a standard model for binary text classification. As is common in the natural language processing literature, we train our classifier on a bag-of-words representation of the data (Jurafsky and Martin, 2014), using tf-idf weightings (Salton and Buckley, 1988). In order to measure the true performance of tuned hyper-parameters, we must use the available data in an unconventional way. By restricting our model fitting and tuning to a randomly sub-sampled $1,000$ review subset to act as our training set for all our experiments, we provide a large held-out collection of $49,000$ movie reviews, upon which we can calculate the 'true' performance of the hyper-parameter configurations chosen by our tuning algorithms. We then randomly draw our train-test splits from this fixed training set, with test sets of 10%. As already argued, the model scores based on a particular evaluation strategy do not necessarily correspond to the true performance and so, although we acknowledge that this contrived use of the data is not standard, this set-up is necessary to measure the improved efficiency and reliability provided by BOSH.

References

Ahsan S Alvi, Binxin Ru, Jan Calliess, Stephen J Roberts, and Michael A Osborne. Asynchronous batch Bayesian optimisation with improved local penalisation. *International Conference on Machine Learning*, 2019.

Eric Anderson, Gilman D Veith, and David Weininger. *SMILES, a line notation and computerized interpreter for chemical structures*. US Environmental Protection Agency, 1987.

Reinaldo B Arellano-Valle, Javier E Contrera-Reyes, and Marc G Genton. Shannon entropy and mutual information for multivariate skew-elliptical distributions. *Scandinavian Journal of Statistics*, 2013.

Adelchi Azzalini. A class of distributions which includes the normal ones. *Scandinavian Journal of Statistics*, 1985.

Adelchi Azzalini and A Dalla Valle. The multivariate skew-normal distribution. *Biometrika*, 1996.

Maximilian Balandat, Brian Karrer, Daniel R. Jiang, Samuel Daulton, Benjamin Letham, Andrew Gordon Wilson, and Eytan Bakshy. BoTorch: a framework for efficient Monte-Carlo Bayesian optimization. In *Advances in Neural Information Processing Systems*, 2020.

Daniel Beck and Trevor Cohn. Learning kernels over strings using Gaussian processes. In *International Joint Conference on Natural Language Processing*, 2017.

Daniel Beck, Trevor Cohn, Christian Hardmeier, and Lucia Specia. Learning structural kernels for natural language processing. *Transactions of the Association for Computational Linguistics*, 2015.

Syrine Belakaria, Aryan Deshwal, and Janardhan Rao Doppa. Max-value entropy search for multi-objective Bayesian optimization. In *Advances in Neural Information Processing Systems*, 2019.

Syrine Belakaria, Aryan Deshwal, and Janardhan Rao Doppa. Max-value entropy search for multi-objective Byesian optimization with constraints. *arXiv preprint arXiv:2009.01721*, 2020.

James Bergstra, Daniel Yamins, and David Daniel Cox. Making a science of model search: hyper-parameter optimization in hundreds of dimensions for vision architectures. *Journal of Machine Learning Research*, 2013.

Joel Berkeley, Henry B. Moss, Artem Artemev, Sergio Pascual-Diaz, Uri Granta, Hrvoje Stojic, Ivo Couckuyt, Jixiang Quing, Loka Satrio, and Victor Picheny. Trieste, 2021. URL `https://github.com/secondmind-labs/trieste`.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: a review for statisticians. *Journal of the American Statistical Association*, 2017.

Yutian Chen, Aja Huang, Ziyu Wang, Ioannis Antonoglou, Julian Schrittwieser, David Silver, and Nando de Freitas. Bayesian optimization in alphago. *arXiv preprint arXiv:1812.06855*, 2018.

Clément Chevalier and David Ginsbourger. Fast computation of the multi-points expected improvement with applications in batch selection. In *International Conference on Learning and Intelligent Optimization*, 2013.

Emile Contal, David Buffoni, Alexandre Robicquet, and Nicolas Vayatis. Parallel Gaussian process optimization with upper confidence bound and pure exploration. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2013.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 1995.

Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

Kurt Cutajar, Mark Pullin, Andreas Damianou, Neil Lawrence, and Javier González. Deep Gaussian processes for multi-fidelity modeling. *arXiv preprint arXiv:1903.07320*, 2019.

Aryan Deshwal, Syrine Belakaria, and Janardhan Rao Doppa. Mercer features for efficient combinatorial Bayesian optimization. *arXiv preprint arXiv:2012.07762*, 2020.

Jesse Dodge, Kevin Jamieson, and Noah A Smith. Open loop hyperparameter optimization and determinantal point processes. *arXiv preprint arXiv:1706.01566*, 2017.

Peter I Frazier, Warren B Powell, and Savas Dayanik. A knowledge-gradient policy for sequential information collection. *SIAM Journal on Control and Optimization*, 2008.

Jennifer Gillenwater, Alex Kulesza, and Ben Taskar. Near-optimal map inference for determinantal point processes. In *Advances in Neural Information Processing Systems*, 2012.

Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Central Science*, 2018.

Chengyue Gong, Jian Peng, and Qiang Liu. Quantile stein variational gradient descent for batch Bayesian optimization. In *International Conference on Machine Learning*, 2019.

Javier González, Joseph Longworth, David C James, and Neil D Lawrence. Bayesian optimization for synthetic gene design. *Advances in Neural Information Processing Systems: Workshop in Bayesian Optimization*, 2014.

Javier González, Zhenwen Dai, Philipp Hennig, and Neil Lawrence. Batch Bayesian optimization via local penalization. In *Artificial intelligence and statistics*, 2016a.

Javier González, Michael Osborne, and Neil Lawrence. Glasses: Relieving the myopia of Bayesian optimisation. In *Artificial Intelligence and Statistics*, 2016b.

Ryan-Rhys Griffiths and José Miguel Hernández-Lobato. Constrained Bayesian optimization for automatic chemical design. *Chemical Sciences*, 2020.

Philipp Hennig and Christian J Schuler. Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, 2012.

James Hensman, Neil D Lawrence, and Magnus Rattray. Hierarchical Bayesian modelling of gene expression time series across irregularly sampled replicates and clusters. *BMC Bioinformatics*, 2013.

Daniel Hernández-Lobato, Jose Hernandez-Lobato, Amar Shah, and Ryan Adams. Predictive entropy search for multi-objective bayesian optimization. In *International Conference on Machine Learning*, pages 1492–1501, 2016.

José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *Neural Information Processing Systems*, 2014.

José Miguel Hernández-Lobato, James Requeima, Edward O Pyzer-Knapp, and Alán Aspuru-Guzik. Parallel and distributed Thompson sampling for large-scale accelerated exploration of chemical space. In *International Conference on Machine Learning*, 2017.

Matthew W Hoffman and Zoubin Ghahramani. Output-space predictive entropy search for flexible global optimization. In *Advances in Neural Information Processing Systems: Workshop on Bayesian Optimization*, 2015.

Deng Huang, Theodore T Allen, William I Notz, and Ning Zeng. Global optimization of stochastic black-box systems via sequential kriging meta-models. *Journal of Global Optimization*, 2006.

Shali Jiang, Henry Chai, Javier Gonzalez, and Roman Garnett. Binoculars for efficient, nonmyopic sequential experimental design. In *International Conference on Machine Learning*, 2020.

Donald R Jones, Cary D Perttunen, and Bruce E Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of optimization Theory and Applications*, 1993.

Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 1998.

Kai Junge, Josie Hughes, Thomas George Thuruthel, and Fumiya Iida. Improving robotic cooking using batch Bayesian optimization. *IEEE Robotics and Automation Letters*, 2020.

Dan Jurafsky and James H Martin. *Speech and Language Processing*. 2014.

Kirthevasan Kandasamy, Gautam Dasarathy, Junier B Oliva, Jeff Schneider, and Barnabás Póczos. Gaussian process bandit optimisation with multi-fidelity evaluations. In *Advances in Neural Information Processing Systems*, 2016.

Kirthevasan Kandasamy, Gautam Dasarathy, Jeff Schneider, and Barnabás Póczos. Multi-fidelity Bayesian optimisation with continuous approximations. *International Conference on Machine Learning*, 2017.

Kirthevasan Kandasamy, Akshay Krishnamurthy, Jeff Schneider, and Barnabás Póczos. Parallelised Bayesian optimisation via Thompson sampling. In *International Conference on Artificial Intelligence and Statistics*, 2018a.

Kirthevasan Kandasamy, Willie Neiswanger, Jeff Schneider, Barnabas Poczos, and Eric P Xing. Neural architecture search with Bayesian optimisation and optimal transport. In *Advances in Neural Information Processing Systems*, 2018b.

Tarun Kathuria, Amit Deshpande, and Pushmeet Kohli. Batched Gaussian process bandit optimization via determinantal point processes. In *Advances in Neural Information Processing Systems*, 2016.

Marc C Kennedy and Anthony O'Hagan. Predicting the output from a complex computer code when fast approximations are available. *Biometrika*, 2000.

Sujin Kim, Raghu Pasupathy, and Shane G Henderson. A guide to sample average approximation. In *Handbook of simulation optimization*. 2015.

Aaron Klein, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter. Fast Bayesian optimization of machine learning hyperparameters on large datasets. *International Conference on Artificial Intelligence and Statistics*, 2017a.

Aaron Klein, Stefan Falkner, Numair Mansur, and Frank Hutter. RoBo: A flexible and robust Bayesian optimization framework in Python. In *Neural Information Processing Systems: Workshop on Bayesian Optimization*, 2017b.

Anton J Kleywegt, Alexander Shapiro, and Tito Homem-de Mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 2002.

Chun-Wa Ko, Jon Lee, and Maurice Queyranne. An exact algorithm for maximum entropy sampling. *Operations Research*, 1995.

Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International Joint Conference on Artificial Intelligence*, 1995.

Alex Kulesza, Ben Taskar, et al. Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 2012.

Matt J Kusner, Brooks Paige, and José Miguel Hernández-Lobato. Grammar variational autoencoder. *arXiv preprint arXiv:1703.01925*, 2017.

Loic Le Gratiet and Josselin Garnier. Recursive co-kriging model for design of computer experiments with multiple levels of fidelity. *International Journal for Uncertainty Quantification*, 2014.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Association for Computational Linguistics*, 2011.

Sébastien Marmin, Clément Chevalier, and David Ginsbourger. Differentiating the multipoint expected improvement for optimal batch design. In *International Workshop on Machine Learning, Optimization and Big Data*, 2015.

Bertil Matérn. Spatial variation, volume 36 of. *Lecture Notes in Statistics*, 1960.

Jonas Mockus, Vytautas Tiesis, and Antanas Zilinskas. The application of Bayesian methods for seeking the extremum. *Towards global optimization*, 1978.

Henry Moss, David Leslie, and Paul Rayson. Using J-K-fold cross validation to reduce variance when tuning nlp models. In *International Conference on Computational Linguistics*, 2018.

Henry B Moss and Ryan-Rhys Griffiths. Gaussian process molecule property prediction with flowmo. *Advances in Neural Information Processing Systems: Workshop on Machine Learning for Molecules*, 2020.

Henry B Moss, Andrew Moore, David S Leslie, and Paul Rayson. Fiesta: Fast identification of state-of-the-art models using adaptive bandit algorithms. *Association for Computational Linguistics*, 2019.

Henry B Moss, Vatsal Aggarwal, Nishant Prateek, Javier González, and Roberto Barra-Chicote. Boffin tts: few-shot speaker adaptation by Bayesian optimization. In *International Conference on Acoustics, Speech and Signal Processing*, 2020a.

Henry B. Moss, Daniel Beck, Javier Gonzalez, David L. Leslie, and Paul. Rayson. Boss: Bayesian optimisation over string spaces. In *Advances in neural information processing systems*, 2020b.

Henry B Moss, David S Leslie, and Paul Rayson. Bosh: Bayesian optimization by sampling hierarchically. *International Conference on Machine Learning: Workshop on Active Learning and Experimental Design*, 2020c.

Henry B. Moss, David S. Leslie, and Paul Rayson. MUMBO: Multi-task max-value Bayesian optimization. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2020d.

Anthony O'Hagan. A Markov property for covariance structures. *Statistics Research Report*, 1998.

Andrei Paleyes, Mark Pullin, Maren Mahsereci, Neil Lawrence, and Javier González. Emulation of physical processes with emukit. In *Advances in Neural Infromation Processing Systems: Workshop on Machine Learning and the Physical Sciences*, 2019.

Michael Pearce, Matthias Poloczek, and Juergen Branke. Bayesian optimization allowing for common random numbers. *Winter Simulation Conference*, 2019.

Paris Perdikaris, Maziar Raissi, Andreas Damianou, Neil D Lawrence, and George Em Karniadakis. Nonlinear information fusion algorithms for data-efficient multi-fidelity modelling. *Royal Society A: Mathematical, Physical and Engineering Sciences*, 2017.

Victor Picheny, David Ginsbourger, and Yann Richet. Noisy expected improvement and on-line computation time allocation for the optimization of simulators with tunable fidelity. *International Conference on Engineering Optimisation*, 2010.

Malte Prieß, Slawomir Koziel, and Thomas Slawig. Surrogate-based optimization of climate model parameters using response correction. *Journal of Computational Science*, 2011.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, 2008.

Laura Elena Raileanu and Kilian Stoffel. Theoretical comparison between the gini index and information gain criteria. *Annals of Mathematics and Artificial Intelligence*, 2004.

Carl Edward Rasmussen. Gaussian processes in machine learning. In *Advanced Lectures on Machine Learning*. 2004.

Binxin Ru, Michael A Osborne, Mark McLeod, and Diego Granziol. Fast information-theoretic Bayesian optimisation. In *International Conference on Machine Learning*, pages 4384–4392, 2018.

Binxin Ru, Ahsan S Alvi, Vu Nguyen, Michael A Osborne, and Stephen J Roberts. Bayesian optimisation over multiple continuous and categorical inputs. *International Conference on Machine Learning*, 2020.

Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing an Management*, 1988.

Amar Shah and Zoubin Ghahramani. Parallel predictive entropy search for batch global optimization of expensive objective functions. In *Advances in Neural Information Processing Systems*, pages 3330–3338, 2015.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical Bayesian optimization of machine learning algorithms. In *Neural Information Processing Systems*, 2012.

Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. *International Conference on Machine Learning*, 2010.

Kevin Swersky, Jasper Snoek, and Ryan P Adams. Multi-task Bayesian optimization. In *Neural Information Processing Systems*, 2013.

Kevin Swersky, Yulia Rubanova, David Dohan, and Kevin Murphy. Amortized Bayesian optimization over discrete spaces. In *Conference on Uncertainty in Artificial Intelligence*, 2020.

Shion Takeno, Hitoshi Fukuoka, Yuhki Tsukada, Toshiyuki Koyama, Motoki Shiga, Ichiro Takeuchi, and Masayuki Karasuyama. Multi-fidelity Bayesian optimization with max-value entropy search. *International Conference on Machine Learning*, 2020.

Ryokei Tanaka and Hiroyoshi Iwata. Bayesian optimization for genomic selection: a method for discovering the best genotype among a large number of candidates. *Theoretical and Applied Genetics*, 2018.

Sattar Vakili, Victor Picheny, and Artem Artemev. Scalable Thompson sampling using sparse Gaussian process models. *arXiv preprint arXiv:2006.05356*, 2020.

Zi Wang and Stefanie Jegelka. Max-value entropy search for efficient Bayesian optimization. In *International Conference in Machine Learning*, 2017.

Zi Wang, Chengtao Li, Stefanie Jegelka, and Pushmeet Kohli. Batched high-dimensional Bayesian optimization via structural kernel learning. In *International Conference on Machine Learning*, 2017.

Jian Wu and Peter Frazier. The parallel knowledge gradient method for batch Bayesian optimization. In *Advances in Neural Information Processing Systems*, 2016.

Jian Wu and Peter I Frazier. Continuous-fidelity Bayesian optimization with knowledge gradient. *Advances in Neural Information Processing Systems: Workshop in Bayesian Optimization*, 2017.

Yiming Yang and Jan O Pedersen. A comparative study on feature selection in text categorization. In *International Conference on Machine Learning*, 1997.

Yehong Zhang, Trong Nghia Hoang, Bryan Kian Hsiang Low, and Mohan Kankanhalli. Information-based multi-fidelity Bayesian optimization. In *Advances in Neural Information Processing Systems: Workshop on Bayesian Optimization*, 2017.