Portfolio Allocation for Bayesian Optimization

Matthew Hoffman, Eric Brochu, Nando de Freitas
Department of Computer Science
University of British Columbia
Vancouver, Canada
{hoffmanm,ebrochu,nando}@cs.ubc.ca

Abstract

Bayesian optimization with Gaussian processes has become an increasingly popular tool in the machine learning community. It is efficient and can be used when very little is known about the objective function, making it popular in expensive black-box optimization scenarios. It uses Bayesian methods to sample the objective efficiently using an *acquisition function* which incorporates the posterior estimate of the objective. However, there are several different parameterized acquisition functions in the literature, and it is often unclear which one to use. Instead of using a single acquisition function, we adopt a portfolio of acquisition functions governed by an online multi-armed bandit strategy. We propose several portfolio strategies, the best of which we call GP-Hedge, and show that this method outperforms the best individual acquisition function. We also provide a theoretical bound on the algorithm's performance.

1 INTRODUCTION

Bayesian optimization is a powerful strategy for finding the extrema of objective functions that are expensive to evaluate. It is applicable in situations where one does not have a closed-form expression for the objective function, but where one can obtain noisy evaluations of this function at sampled values. It is particularly useful when these evaluations are costly, when one does not have access to derivatives, or when the problem at hand is non-convex. Bayesian optimization has two key ingredients. First, it uses the entire sample history to compute a posterior distribution over the unknown objective function. Second, it uses an *acquisition function* to automatically trade off between exploration and exploitation when selecting the points at which to sample next. As such, Bayesian optimization techniques are some of the most efficient approaches in terms of the number of function evaluations required [30, 23, 25, 3, 5]. The term Bayesian optimization was coined in the seventies [30], but a version of the method has been known as Efficient Global Optimization (EGO) in the experimental design literature since the nineties [37]. In recent years, the machine learning community has increasingly used Bayesian optimization to optimize expensive objective functions. Examples can be found in robot gait design [26], online path planning [28, 29], intelligent user interfaces for animation [6, 4], algorithm configuration [20], efficient MCMC [34], sensor placement [38, 33], and reinforcement learning [5]. Consistency of the method was shown in [27] for 1D processes and in [39] for general Gaussian processes with an acquisition function known as expected improvement. Rates of convergence for an acquisition function, known as upper confidence bound, were provided last year in [38]. A more recent report [9] discusses more general convergence rates. We refer readers interested in a more in depth review of Bayesian optimization to [5].

Our main argument is that the choice of acquisition function is not trivial. Several different acquisition functions have been proposed in the literature, none of which work well for all classes of functions. Building on recent developments in the field of online learning and multi-armed bandits [10], this paper proposes a solution to this problem. The solution is based on a hierarchical hedging approach for managing an adaptive portfolio of acquisition functions.

The paper will show that the proposed strategy of combining acquisition functions results in large improvements over the single acquisitions strategies proposed in statistics and optimization (expected improvement) and more recently in machine learning (upper confidence bounds). This will be shown with synthetic experiments (so that we can assess the effect of dimensionality), a suite of optimization problems bor-

rowed from the global optimization literature (some of which are repeatedly cited as being very hard) and a hard, nonlinear, 9D continuous Markov decision process with a reward that has many modes and relatively large plateaus in between. The nature of the reward function in the control problem will cause gradient methods to do much worse than the Bayesian optimization strategies. Finally, the paper will also present a theoretical analysis of the proposed techniques.

We review Bayesian optimization and popular acquisition functions in Section 2. In Section 3, we propose the use of various hedging strategies for Bayesian optimization [2, 11]. In Section 4, we present experimental results using standard test functions from the literature of global optimization. The experiments show that the proposed hedging approaches outperform any of the individual acquisition functions. We also provide detailed comparisons among the hedging strategies. Finally, in Section 5 we present a bound on the cumulative regret which helps provide some intuition as to algorithm's performance.

2 BAYESIAN OPTIMIZATION

We are concerned with the task of optimization on a d-dimensional space: $\max_{\mathbf{x}\in A\subseteq\mathbb{R}^d} f(\mathbf{x})$.

We define $\mathbf{x}_t$ as the tth sample and $y_t = f(\mathbf{x}_t)+\epsilon_t$, with $\epsilon_t \overset{iid}{\sim} \mathcal{N}(0,\sigma^2)$, as a noisy observation of the objective function at $\mathbf{x}_t$. Other observation models are possible [5, 12, 14, 36], but we will focus on real, Gaussian observations for ease of presentation.

The Bayesian optimization procedure is shown in Algorithm 1. As mentioned earlier, it has two components: the posterior distribution over the objective and the acquisition function. Let us focus on the posterior distribution first and come back to the acquisition function in Section 2.2. As we accumulate observations[1] $\mathcal{D}_{1:t} = \{\mathbf{x}_{1:t}, y_{1:t}\}$, a prior distribution $P(f)$ is combined with the likelihood function $P(\mathcal{D}_{1:t}|f)$ to produce the posterior distribution: $P(f|\mathcal{D}_{1:t}) \propto P(\mathcal{D}_{1:t}|f)P(f)$. The posterior captures the updated beliefs about the unknown objective function. One may also interpret this step of Bayesian optimization as estimating the objective function with a *surrogate function* (also called a *response surface*). We will place a Gaussian process (GP) prior on f. Other nonparametric priors over functions, such as random forests, have been considered [5], but the GP strategy is the most popular alternative.

[1]Here we use subscripts to denote sequences of data, i.e. $y_{1:t} = \{y_1, \ldots, y_t\}$.

Algorithm 1 Bayesian Optimization

1: **for** $t = 1, 2, \ldots$ **do**
2: Find $\mathbf{x}_t$ by optimizing the acquisition function over the GP: $\mathbf{x}_t = \text{argmax}_{\mathbf{x}}\, u(\mathbf{x}|\mathcal{D}_{1:t-1})$.
3: Sample the objective function: $y_t = f(\mathbf{x}_t) + \epsilon_t$.
4: Augment the data $\mathcal{D}_{1:t} = \{\mathcal{D}_{1:t-1}, (\mathbf{x}_t, y_t)\}$.
5: **end for**

2.1 GAUSSIAN PROCESSES

The objective function is distributed according to a GP prior: $f(\mathbf{x}) \sim \text{GP}(m(\mathbf{x}), k(\mathbf{x}_i, \mathbf{x}_j))$. For convenience, and without loss of generality, we assume that the prior mean is the zero function (but see [29, 35, 4] for examples of nonzero means). This leaves us the more interesting question of defining the covariance function. A very popular choice is the squared exponential kernel with a vector of automatic relevance determination (ARD) hyperparameters $\boldsymbol{\theta}$ [35]:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\big(-\tfrac{1}{2}(\mathbf{x}_i - \mathbf{x}_j)^T \,\text{diag}(\boldsymbol{\theta})^{-2}(\mathbf{x}_i - \mathbf{x}_j)\big),$$

where $\text{diag}(\boldsymbol{\theta})$ is a diagonal matrix with entries $\boldsymbol{\theta}$ along the diagonal and zeros elsewhere. The choice of hyperparameters will be discussed in the experimental section, but we note that it is not trivial in this domain because of the paucity of data. For an in depth analysis of this issue we refer the reader to e.g. [4, 33].

We can sample the GP at t points by choosing the indices $\{\mathbf{x}_{1:t}\}$ and sampling the values of the function at these indices to produce the data $\mathcal{D}_{1:t}$. The function values are distributed according to a multivariate Gaussian distribution $\mathcal{N}(0, \mathbf{K})$, with covariance entries $k(\mathbf{x}_i, \mathbf{x}_j)$. Assume that we already have the observations, say from previous iterations, and that we want to use Bayesian optimization to decide what point $\mathbf{x}_{t+1}$ should be considered next. Let us denote the value of the function at this arbitrary point as f_{t+1}. Then, by the properties of GPs, $f_{1:t}$ and f_{t+1} are jointly Gaussian:

$$\begin{bmatrix} f_{1:t} \\ f_{t+1} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} & \mathbf{k} \\ \mathbf{k}^T & k(\mathbf{x}_{t+1}, \mathbf{x}_{t+1}) \end{bmatrix}\right),$$

where $\mathbf{k} = [k(\mathbf{x}_{t+1}, \mathbf{x}_1), k(\mathbf{x}_{t+1}, \mathbf{x}_2), \ldots, k(\mathbf{x}_{t+1}, \mathbf{x}_t)]$. Using the Sherman-Morrison-Woodbury formula, see [35] for a comprehensive treatment, one can easily arrive at an expression for the predictive distribution:

$$P(y_{t+1}|\mathcal{D}_{1:t}, \mathbf{x}_{t+1}) = \mathcal{N}(\mu_t(\mathbf{x}_{t+1}), \sigma_t^2(\mathbf{x}_{t+1}) + \sigma^2),$$

where

$$\mu_t(\mathbf{x}_{t+1}) = \mathbf{k}^T[\mathbf{K} + \sigma^2\mathbf{I}]^{-1}\mathbf{y}_{1:t},$$
$$\sigma_t^2(\mathbf{x}_{t+1}) = k(\mathbf{x}_{t+1}, \mathbf{x}_{t+1}) - \mathbf{k}^T[\mathbf{K} + \sigma^2\mathbf{I}]^{-1}\mathbf{k}.$$

In this sequential decision making setting, the number of query points is relatively small and, consequently, the GP predictions are easy to compute.

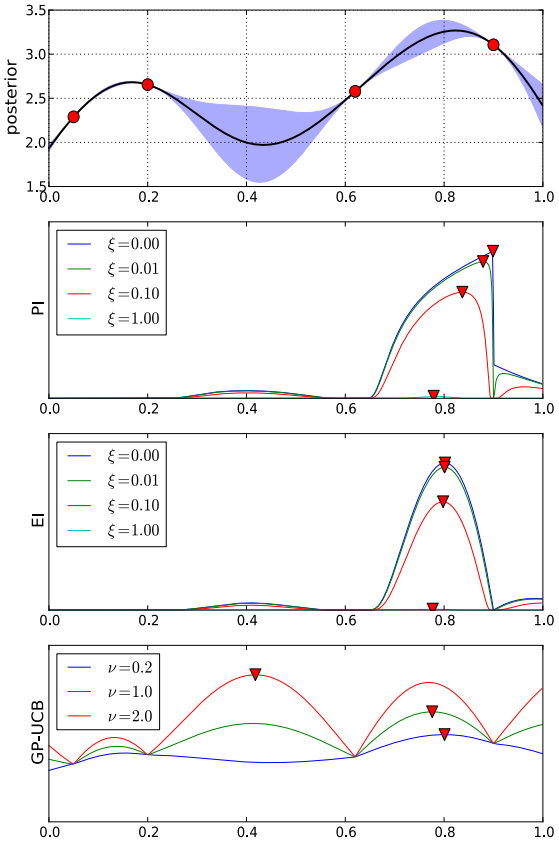

Figure 1: *Acquisition functions with different values of the exploration parameters ν and ξ. The GP posterior is shown at the top. The other images show the acquisition functions for that GP. From the top: Probability of improvement, expected improvement and upper confidence bound. The maximum of each acquisition function, where the GP is to be sampled next, is shown with a triangle marker. Note the increased preference for exploration exhibited by GP-UCB.*

2.2 ACQUISITION FUNCTIONS

The role of the acquisition function is to guide the search for the optimum. Typically, acquisition functions are defined such that high values correspond to *potentially* high values of the objective function, whether because the prediction is high, the uncertainty is great, or both. The acquisition function is maximized to select the next point at which to evaluate the objective function. That is, we wish to sample the objective function at $\operatorname{argmax}_{\mathbf{x}} u(\mathbf{x}|\mathcal{D})$. This auxiliary maximization problem, where the objective is known and easy to evaluate, can be easily carried out with standard numerical techniques such as multi-start, sequential quadratic programming or DIRECT [22, 16, 32]. The acquisition function is sometimes called the *infill* or simply the "utility" function. In the following sections, we will look at the three most popular choices. Figure 1 shows how these give rise to distinct sampling behaviour.

Probability of improvement (PI): The early work of Kushner [24] suggested maximizing the *probability of improvement* over the incumbent $\mu^+ = \max_t \mu(\mathbf{x}_t)$. The drawback, intuitively, is that this formulation is biased toward exploitation only. To remedy this, practitioners often add a trade-off parameter $\xi \geq 0$, so that

$$\mathrm{PI}(\mathbf{x}) = P(f(\mathbf{x}) \geq \mu^+ + \xi) = \Phi\left(\frac{\mu(\mathbf{x}) - \mu^+ - \xi}{\sigma(\mathbf{x})}\right),$$

where $\Phi(\cdot)$ is the standard Normal cumulative distribution function (CDF). The exact choice of ξ is left to the user. Kushner recommends using a (unspecified) schedule for ξ, which should start high in order to drive exploration and decrease towards zero as the algorithm progresses. Lizotte, however, found that using such a schedule did not offer improvement over a constant value of ξ on a suite of test functions [25].

Expected improvement (EI): More recent work has tended to take into account not only the probability of improvement, but the magnitude of the improvement a point can potentially yield. Močkus et al. [30] proposed maximizing the *expected improvement* with respect to the best function value yet seen, given by the incumbent $\mathbf{x}^+ = \operatorname{argmax}_{\mathbf{x}_t} f(\mathbf{x}_t)$. For our Gaussian process posterior, one can easily evaluate this expectation, see [21], yielding:

$$\mathrm{EI}(\mathbf{x}) = \begin{cases} d\Phi(d/\sigma(\mathbf{x})) + \sigma(\mathbf{x})\phi(d/\sigma(\mathbf{x})) & \text{if } \sigma(\mathbf{x}) > 0 \\ 0 & \text{if } \sigma(\mathbf{x}) = 0 \end{cases}$$

where $d = \mu(\mathbf{x}) - \mu^+ - \xi$ and where $\phi(\cdot)$ and $\Phi(\cdot)$ denote the PDF and CDF of the standard Normal distribution respectively. Here ξ is an optional trade-off parameter analogous to the one defined above.

Upper confidence bound (UCB & GP-UCB): Cox and John [13] introduce an algorithm they call "Sequential Design for Optimization", or *SDO*. Given a random function model, SDO selects points for evaluation based on a confidence bound consisting of the mean and weighted variance: $\mu(\mathbf{x}) + \kappa\sigma(\mathbf{x})$. As with the other acquisition models, however, the parameter κ is left to the user. A principled approach to selecting this parameter is proposed by Srinivas et al. [38]. In this work, the authors define the instantaneous regret of the selection algorithm as $r(\mathbf{x}) = f(\mathbf{x}^\star) - f(\mathbf{x})$ and attempt to select a sequence of weights κ_t so as to minimize the cumulative regret $R_T = r(\mathbf{x}_1) + \cdots + r(\mathbf{x}_T)$. Using the *upper confidence bound* selection criterion with $\kappa_t = \sqrt{\nu\beta_t}$ and the hyperparameter $\nu > 0$ Srinivas et al. define

$$\text{GP-UCB}(\mathbf{x}) = \mu(\mathbf{x}) + \sqrt{\nu\beta_t}\sigma(\mathbf{x}).$$

It can be shown that this method has cumulative regret bounded by $\mathcal{O}(\sqrt{T\beta_T\gamma_T})$ with high probability.

Here β_T is a carefully selected learning rate and γ_T is a bound on the information gained by the algorithm at selected points after T steps. Both of these terms depend upon the particular form of kernel-function used, but for most kernels their product can be shown to be sublinear in T. We refer the interested reader to the original paper [38] for exact bounds.

The sublinear bound on cumulative regret implies that the method is *no-regret*, i.e. that $\lim_{T\to\infty} R_T/T = 0$. This in turn provides a bound on the convergence rate for the optimization process, since the regret at the maximum $f(\mathbf{x}^*) - \max_t f(\mathbf{x}_t)$ is upper bounded by the average regret $R_T/T = f(\mathbf{x}^*) - \frac{1}{T}\sum_{t=1}^{T} f(\mathbf{x}_t)$. As we will note later, however, this bound can be quite loose in practice.

Algorithm 2 GP-Hedge

1: Select parameter $\eta \in \mathbb{R}^+$.
2: Set $g_0^i = 0$ for $i = 1, \ldots, N$.
3: **for** $t = 1, 2, \ldots$ **do**
4: Nominate points from each acquisition function: $\mathbf{x}_t^i = \text{argmax}_{\mathbf{x}}\, u_i(\mathbf{x}|\mathcal{D}_{1:t-1})$.
5: Select nominee $\mathbf{x}_t = \mathbf{x}_t^j$ with probability $p_t(j) = \exp(\eta g_{t-1}^j)/\sum_{\ell=1}^{k}\exp(\eta g_{t-1}^\ell)$.
6: Sample the objective function $y_t = f(\mathbf{x}_t) + \epsilon_t$.
7: Augment the data $\mathcal{D}_{1:t} = \{\mathcal{D}_{1:t-1}, (\mathbf{x}_t, y_t)\}$.
8: Receive rewards $r_t^i = \mu_t(\mathbf{x}_t^i)$ from the updated GP.
9: Update gains $g_t^i = g_{t-1}^i + r_t^i$.
10: **end for**

3 PORTFOLIO STRATEGIES

There is no choice of acquisition function that can be guaranteed to perform best on an arbitrary, unknown objective. In fact, it may be the case that no single acquisition function will perform the best over an entire optimization — a mixed strategy in which the acquisition function samples from a pool (or portfolio) at each iteration might work better than any single acquisition. This can be treated as a hierarchical multi-armed bandit problem, in which each of the N arms is an acquisition function, each of which is itself an infinite-armed bandit problem. In this section we propose solving the selection problem using three strategies from the literature, the application of which we believe to be novel.

Hedge is an algorithm which at each time step t selects an action i with probability $p_t(i)$ based on the cumulative rewards (gain) for that action (see Auer et al. [2]). After selecting an action the algorithm receives reward r_t^i for each action and updates the gain vector. In the Bayesian optimization setting, we can define N bandits each corresponding to a single acquisition function. Choosing action i corresponds to sampling from the point nominated by function u_i, i.e. $\mathbf{x}_t^i = \text{argmax}_{\mathbf{x}}\, u_i(\mathbf{x}|\mathcal{D}_{1:t-1})$ for $i = 1, \ldots, N$. Finally, while in the conventional Bayesian optimization setting the objective function is sampled only once per iteration, Hedge is a full information strategy and requires a reward for every action at every time step. We can achieve this by defining the reward at $\mathbf{x}_t^i$ as the expected value of the GP model at $\mathbf{x}_t^i$. That is, $r_t^i = \mu_t(\mathbf{x}_t^i)$. We refer to this method as GP-Hedge. Provided that the objective function is smooth, this reward definition is reasonable.

Auer et al. also propose the *Exp3* algorithm, a variant of Hedge that applies to the partial information setting. In this setting it is no longer assumed that rewards are observed for all actions. Instead at each iteration a reward is only associated with the particular action selected. The algorithm uses Hedge as a subroutine where rewards observed by Hedge at each iteration are $r_t^i/\hat{p}_t(i)$ for the action selected and zero for all actions. Here $\hat{p}_t(i)$ is the probability that Hedge would have selected action i. The Exp3 algorithm, meanwhile, selects actions from a distribution that is a mixture between $\hat{p}_t(i)$ and the uniform distribution. Intuitively this ensures that the algorithm does not miss good actions because the initial rewards were low (i.e. it continues exploring).

Finally, another possible strategy is the *NormalHedge* algorithm [11]. This method, however, is built to take advantage of situations where the number of bandit arms (acquisition functions) is large, and may not be a good match to problems where N is relatively small.

The GP-Hedge procedure is shown in Algorithm 2. In practice any of these hedging strategies could be used, however in our experiments we find that Hedge tends to outperform the others. Note that it is necessary to optimize N acquisition functions at each time step rather than 1. While this might seem expensive, this is unlikely to be a major problem in practice for small N, as (i) Bayesian optimization is typically employed when sampling the objective is so expensive as to dominate other costs; (ii) it has been shown that fast approximate optimization of u is usually sufficient [6, 25, 20]; and (iii) it is straightforward to run the optimizations in parallel on a modern multicore processor.

We will also note that the setting of our problem is somewhere "in between" the full and partial information settings. Consider, for example, the situation that all points sampled by our algorithm are "too distant" in the sense that the kernels evaluated at these points exert negligible influence on each other. In this case, we can see that only information obtained by the sampled point is available, and as a result GP-Hedge will be over-confident in its predictions when using the full-

information strategy. However, this behaviour is not observed in practical situations because of smoothness properties, as well as our particular selection of acquisition functions. In the case of adversarial acquisition functions one might instead choose to use the Exp3 variant.

4 EXPERIMENTS

To validate the use of GP-Hedge, we tested the optimization performance on a set of test functions with known maxima $f(\mathbf{x}^\star)$. To see how effective each method is at finding the global maximum, we use the "gap" metric [19], defined as

$$G_t = \Big[f(\mathbf{x}^+) - f(\mathbf{x}_1)\Big] \Big/ \Big[f(\mathbf{x}^\star) - f(\mathbf{x}_1)\Big],$$

where again $\mathbf{x}^+$ is the incumbent or best function sample found up to time t. The gap G_t will therefore be a number between 0 (indicating no improvement over the initial sample) and 1 (if the incumbent is the maximum). Note, while this performance metric is evaluated on the true function values, this information is not available to the optimization methods.

4.1 STANDARD TEST FUNCTIONS

We first tested performance using functions common to the literature on Bayesian optimization: the Branin, Hartman 3, and Hartman 6 functions. All of these are continuous, bounded, and multimodal, with 2, 3, and 6 dimensions respectively. We omit the formulae of the functions for space reasons, but they can be found in [25]. These functions have been proposed by [15] as benchmarks for comparing global search methods and are widely used for this purpose, see e.g. [22].

For each experiment, we optimized 25 times and computed the mean and variance of the gap metric over time. In these experiments we used hyperparameters $\boldsymbol{\theta}$ chosen offline so as to maximize the log marginal likelihood of a (sufficiently large) set of sample points; see [35]. We compared the standard acquisition functions using parameters suggested by previous authors, i.e. $\xi = 0.01$ for EI and PI, $\delta = 0.1$ and $\nu = 0.2$ for GP-UCB [25, 38]. For the GP-Hedge trials, we tested performance under using both 3 acquisition functions and 9 acquisition functions. For the 3-function variant we use the standard acquisition functions with default hyperparameters. The 9-function variant uses these same three as well as 6 additional acquisition functions consisting of: both PI and EI with $\xi = 0.1$ and $\xi = 1.0$, GP-UCB with $\nu = 0.1$ and $\nu = 1.0$. While we omit trials of these additional acquisition functions for space reasons, these values are not expected to perform as well as the defaults and our experiments confirmed this hypothesis. However, we are curious to see if adding known suboptimal acquisition functions will help or hinder GP-Hedge in practice.

Results for the gap measure G_t are shown in Figure 2. While the improvement GP-Hedge offers over the best single acquisition function varies, there is almost no combination of function and time step in which the 9-function GP-Hedge variant is not the best-performing method. The results suggest that the extra acquisition functions assist GP-Hedge in exploring the space in the early stages of the optimization process. Figure 2 also displays, for a single example run, how the the arm probabilities $p_t(i)$ used by GP-Hedge evolve over time. We have observed that the distribution becomes more stable when the acquisition functions come to a general consensus about the best region to sample. As the optimization progresses, exploitation becomes more rewarding than exploration, resulting in more probability being assigned to methods that tend to exploit. However, note that if the initial portfolio had consisted only of these more exploitative acquisition functions, the likelihood of becoming trapped at suboptimal points would have been higher.

In Figure 3 we compare against the other Hedging strategies introduced in Section 3 under both the gap measure and mean average regret. We also introduce a baseline strategy which utilizes a portfolio uniformly distributed over the same acquisition functions. The results show that mixing across multiple acquisition functions provides significant performance benefits under the gap measure, and as the problems' difficulty/dimensionality increases we see that GP-Hedge outperforms other mixed strategies. The uniform strategy performs well on the easier test functions, as the individual acquisition functions are reasonable. However, for the hardest problem (Hartman 6) we see that the performance of the naive uniform strategy degrades. NormalHedge performs particularly poorly on this problem. We observed that this algorithm very quickly collapses to an exclusively exploitative portfolio which becomes very conservative in its departures from the incumbent. We again note that this strategy is intended for large values of N, which may explain this behaviour.

In the case of the regret measure we see that the hedging strategies perform comparable to GP-UCB, a method designed to optimize this measure. We further note that although the average regret can prove quite useful in assessing the convergence behavior of Bayesian optimization methods, the bounds provided by this regret can be loose in practice. Further, in the setting of Bayesian optimization we are typically concerned not with the cumulative regret during optimization, but instead with the regret incurred by the incumbent after optimization is complete. Similar no-

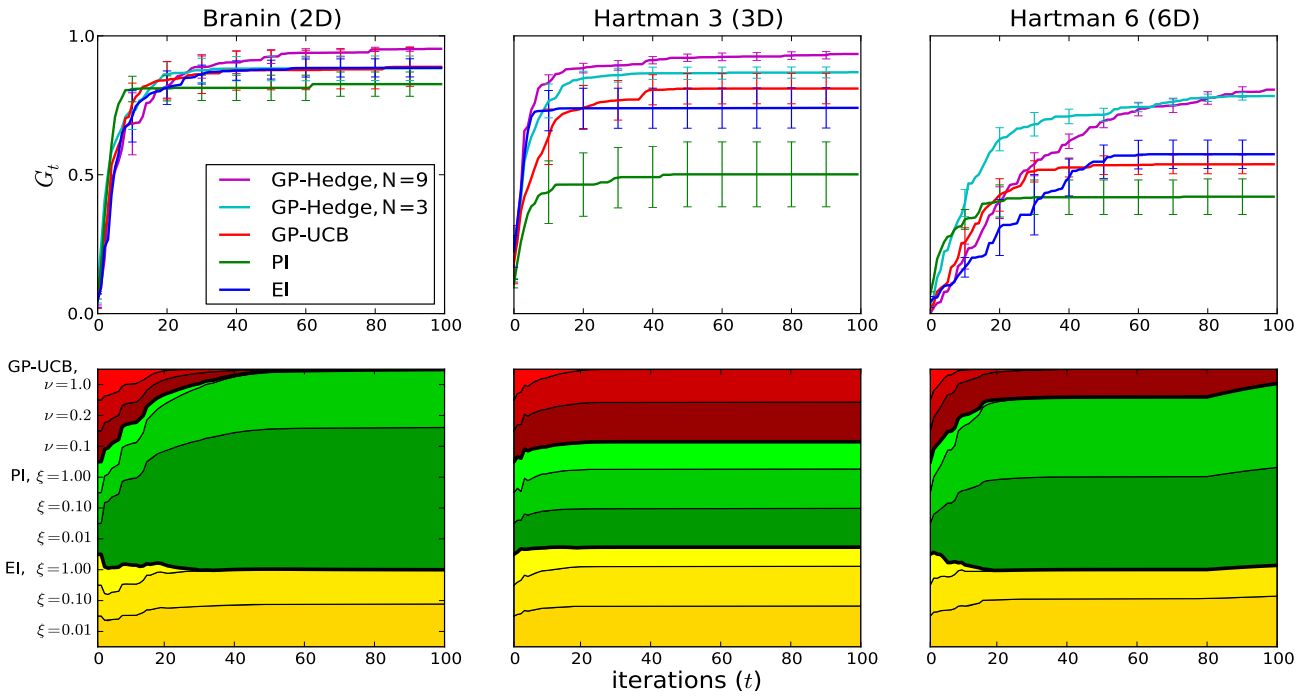

Figure 2: *(Best viewed in colour.) Comparison of different acquisition approaches on three commonly used literature functions. The top plots show the mean and variance of the gap metric averaged over 25 trials. We note that the top two performing algorithms use a portfolio strategy. With $N = 3$ acquisition functions, GP-Hedge beats the best-performing acquisition function in almost all cases. With $N = 9$, we add additional instances of the three acquisition functions, but with different parameters. Despite the fact that these additional functions individually perform worse than the ones with default parameters, adding them to GP-Hedge improves performance in the long run. The bottom plots show an example evolution of GP-Hedge's portfolio with $N = 9$ for each objective function. The height of each band corresponds to the probability $p_t(i)$ at each iteration.*

tions of "simple regret" have been studied in [1, 8].

Based on the performance in these experiments, we use Hedge as the underlying algorithm for GP-Hedge in the remainder of the experiments.

4.2 SAMPLED TEST FUNCTIONS

As there is no generally-agreed-upon set of test functions for Bayesian optimization in higher dimensions, we seek to sample synthetic functions from a known GP prior similar to [25]. For further details on how these functions are sampled see Appendix A. As can be seen in Figure 4, GP-Hedge with $N = 9$ is again the best-performing method, which becomes even more clear as the dimensionality increases. Interestingly, the *worst*-performing function changes as dimensionality increases. In the 40D experiments, GP-UCB, which generally performed well in other experiments, does quite poorly. Examining the behaviour, it appears that by trying to minimize regret instead of maximizing improvement, GP-UCB favours regions of high variance. However, since a 40D space remains extremely sparsely populated even with hundreds of samples, the vast majority of the space still has high variance, and thus high acquisition value.

4.3 CONTROL OF A PARTICLE SIMULATION

We also applied these methods to optimize the behavior of a simulated physical system in which the trajectories of falling particles are controlled via a set of repelling forces. This is a difficult, nonlinear control task whose resulting objective function exhibits fairly isolated regions of high value surrounded by severe plateaus. Briefly, the four-dimensional state-space in this problem consists of a particle's 2D position and velocity $(p, \dot{p})$ with two-dimensional actions consisting of forces which act on the particle. Particles are also affected by gravity and a frictional force resisting movement. The goal is to direct the path of the particle through regions of the state space with high reward $r(p)$ so as to maximize the *total reward* accumulated over many time-steps. In our experiments we use a finite, but large, time-horizon H. In order to control this system we employ a set of "repellers" each of which is located at some position $c_i = (a_i, b_i)$ and has strength w_i (see the top plot of Figure 5). The force on a particle at position p is a weighted sum of the individual forces from all repellers, each of which is inversely proportional to the distance $p - c_i$. For

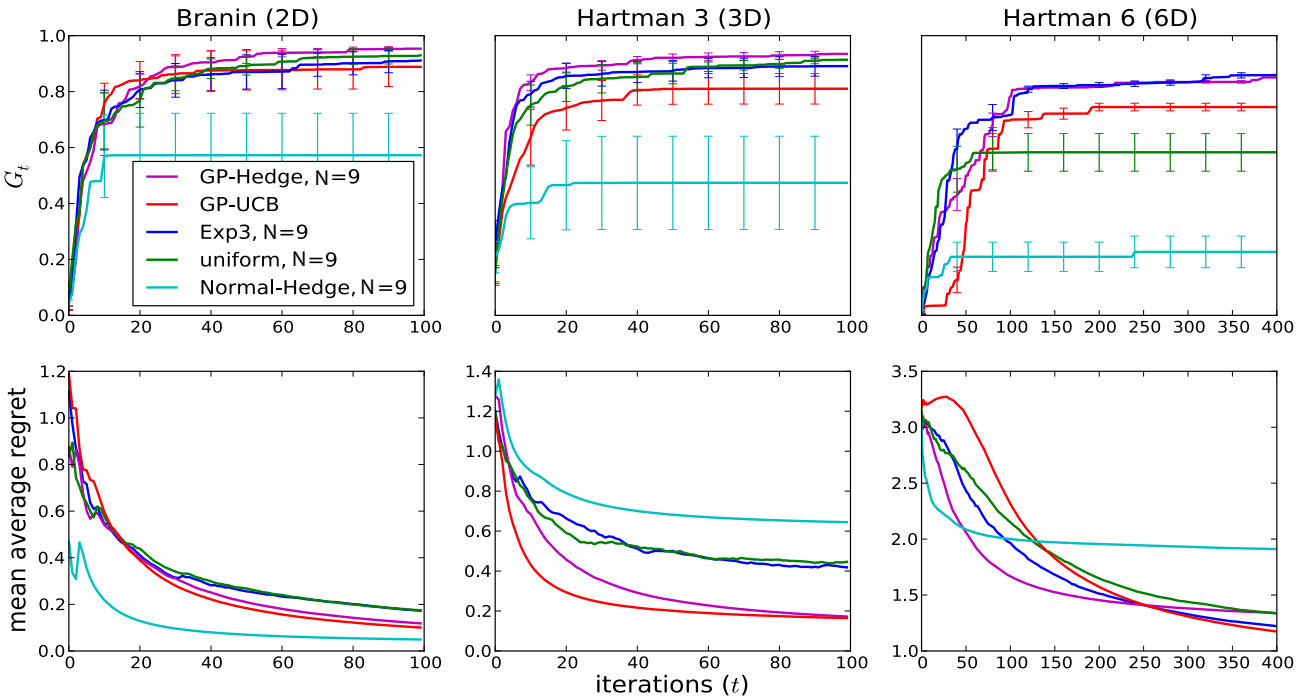

Figure 3: *(Best viewed in colour.) Comparison of different hedging strategies on three commonly used literature functions. The top plots show the mean and variance of the gap metric averaged over 25 trials. Note that both Hedge and Exp3 outperform the best single acquisition function, GP-UCB. The bottom plots show the mean average regret for each method (lower is better). Average regret is shown in order to compare with previous work [38], however as noted in the text the gap measure provides a more direct comparison of optimization performance. We see that mixed strategies (i.e. GP-Hedge) perform comparably to GP-UCB under the regret measure and outperform this individual strategy under the gap measure. As the problems get harder, and with higher dimensionality, GP-Hedge significantly outperforms other acquisition strategies.*

further details we refer the reader to [18].

This problem can be formulated in the setting of Bayesian optimization by defining the vector of repeller parameters $\mathbf{x} = (w_1, a_1, b_1, \dots)$. In the experiments shown in Figure 5 we will utilize three repellers, resulting in a 9D optimization task. We can then define our objective as the total H-step expected reward $f(\mathbf{x}) = \mathbb{E}\left[\sum_{n=0}^{H} r(p_n)|\mathbf{x}\right]$. Finally, since the model defines a probability distribution $P_{\mathbf{x}}(p_{0:H})$ over particle trajectories we can obtain a noisy estimate of this objective function by sampling a single trajectory and evaluating the sum over its immediate rewards.

Results for this optimization task are shown in Figure 5. As with the previous synthetic examples GP-Hedge outperforms each of its constituent methods. We further note the particularly poor performance of PI on this example, which in part we hypothesize is a result of plateaus in the resulting objective function. In particular PI has trouble exploring after it has "locked on" to a particular mode, a fact that seems exacerbated when there are large regions with very little change in objective. The figure also shows that gradient based methods, even when using smart tricks such as PEGASUS [31], perform badly in comparison as the reward is severely multi-modal with large plateaus in between.

5 CONVERGENCE BEHAVIOR

Properly assessing the convergence behaviour of hedging algorithms of this type is very problematic. The main difficulty lies with the fact that decisions made at iteration t affect the state of the problem and the resulting rewards at all future iterations. As a result we cannot relate the regret of our algorithm directly to the regret of the *best* underlying acquisition strategy: had we actually used the best underlying strategy we would have selected completely different points [10, section 7.11].

Regret bounds for the underlying GP-UCB algorithm have been shown [38]. Starting with Auer et al. we also have regret bounds for the hedging strategies used to select between acquisition functions [2] (improved bounds can also be found in [10]). However, because of the points stated in the previous paragraph, and expounded in more detail in the appendix, we cannot simply combine both regret bounds.

With these caveats in mind we will consider a slightly

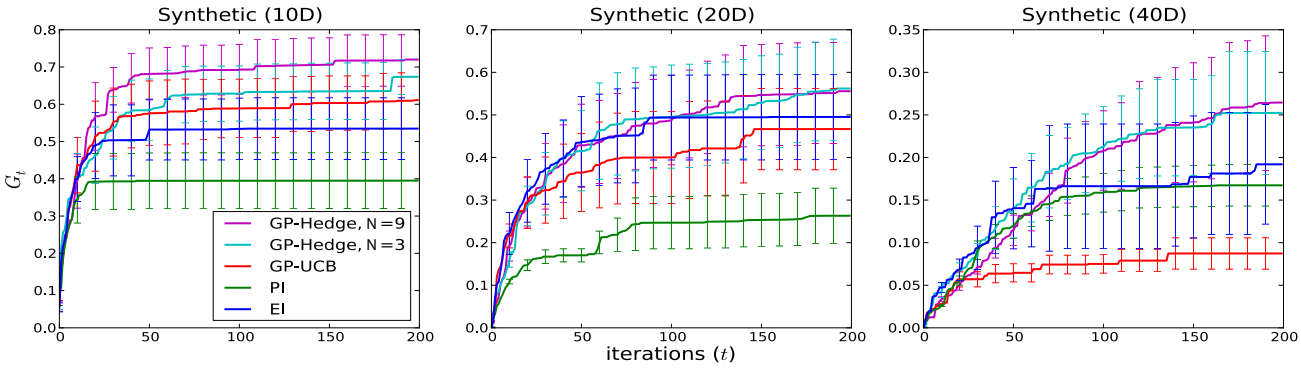

Figure 4: *(Best viewed in colour.) We compare the performance of the acquisition approaches on synthetic functions sampled from a GP prior with randomly initialized hyperparameters. Shown are the mean and variance of the gap metric over 25 sampled functions. Here, the variance is a relative measure of how well the various algorithms perform while the functions themselves are varied. While the variance is high (which is to be expected over diverse functions), we can see that GP-Hedge is at least comparable to the best acquisition functions and ultimately superior for both $N = 3$ and $N = 9$. We also note that for the 10D and 20D experiments GP-UCB performs quite well but suffers in the 40D experiment. This helps to confirm our hypothesis that a mixed strategy is particularly useful in situations where we do not possess strong prior information with regards to the choice of acquisition function.*

different algorithmic framework. In particular we will consider rewards at iteration t given by the mean $\mu_{t-1}(\mathbf{x}_t)$, where this assumption is made merely to simplify the following proof. We will also assume that GP-UCB is included as one of the possible acquisition functions due to its associated convergence results (see Section 2.2). In this scenario we can obtain the following bound on our cumulative regret.

Theorem 1. *Assume GP-Hedge is used with a collection of acquisition strategies, one of which is GP-UCB with parameters β_t. If we also have a bound γ_T on the information gained at points selected by the algorithm after T iterations, then with probability at least $1 - \delta$ the cumulative regret is bounded by*

$$R_T \leq \sqrt{TC_1\beta_T\gamma_T} + \Big[\sum_{t=1}^{T} \beta_t \sigma_{t-1}(\mathbf{x}_t^{\mathrm{UCB}})\Big] + \mathcal{O}(\sqrt{T}),$$

where $\mathbf{x}_t^{\mathrm{UCB}}$ is the tth point proposed by GP-UCB.

We give a full proof of this theorem in the extended version of this paper [7]. We will note that this theorem on its own does not guarantee the convergence of the algorithm, i.e. that $\lim_{T\to\infty} R_T/T = 0$. We can see, however, that our regret is bounded by two sublinear terms and an additional term which depends on the information gained at points proposed, but not necessarily selected. In some sense this additional term depends on the proximity of points proposed by GP-UCB to points previously selected, the expected distance of which should decrease as the number of iterations increases.

We should point out, also, that the regret incurred by the hedging procedure is with respect to the *best underlying strategy*, which need not necessarily be GP-UCB. We then relate this *strategy regret* to the regret incurred by GP-UCB on the actual points proposed due to the known regret bounds for GP-UCB. An interesting extension to these ideas would be to incorporate bounds on the other underlying strategies, such as recent bounds for EI [9].

6 CONCLUSIONS

Hedging strategies are a powerful tool in the design of acquisition functions for Bayesian optimization. In this paper we have shown that strategies that adaptively modify a portfolio of acquisition functions often perform substantially better — and almost never worse — than the best-performing individual acquisition function. This behavior was observed consistently across a broad set of experiments including high-dimensional GPs, standard test problems recommended in the bounded global optimization literature, and a hard continuous, 9D, nonlinear Markov decision process. These improvements will allow for advances in many practical domains of interest where we have already demonstrated the benefits of simple Bayesian optimization techniques [29, 6, 5], including robotics, online planning, hierarchical reinforcement learning, experimental design and interactive user interfaces.

Our experiments have also shown that full-information strategies are able to outperform partial-information strategies in many situations. However, partial-information strategies can be beneficial in instances of high N or in situations where the acquisition functions provide very conflicting advice. Evaluating these tradeoffs is an interesting area of future research.

In this work we give a regret bound for our hedging

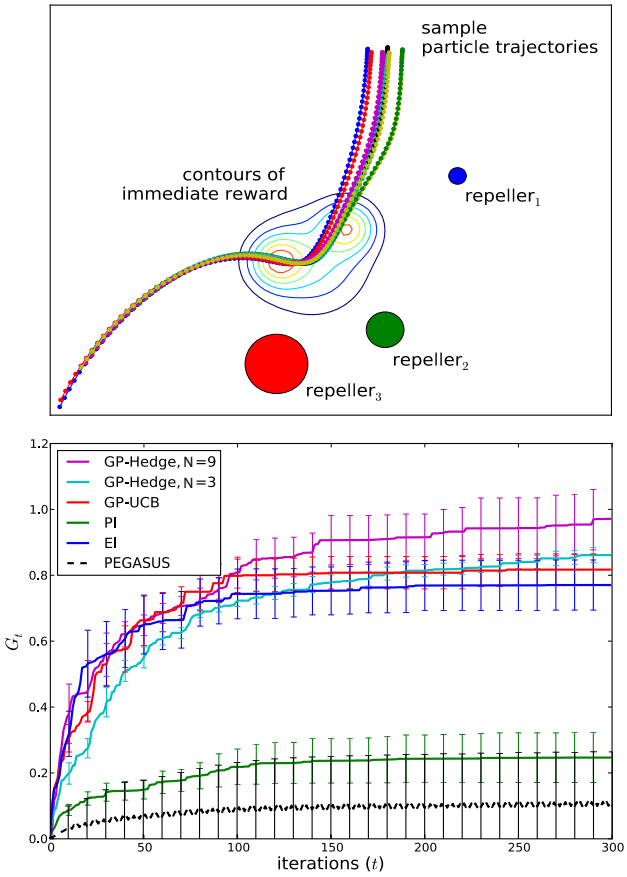

Figure 5: *(Best viewed in colour.) Results of experiments on the repeller control problem. The top plot displays 10 sample trajectories over 100 time-steps for a particular repeller configuration (not necessarily optimal). The bottom plot shows the progress of each of the described Bayesian optimization methods on a similar model, averaged over 25 runs. For comparison, it also shows the progress of a gradient method with PEGASUS.*

strategy by relating its performance to existing bounds for GP-UCB. Although our bound does not guarantee convergence it does provide some intuition as to the success of hedging methods in practice. Another interesting line of future research involves finding similar bounds for the gap measure.

Acknowledgements

We would like to thank Csaba Szepesvári, Rémi Munos, and Yoav Freund for providing very helpful comments and criticism on the theoretical and practical aspects of this work. We thank MITACS for financial support.

References

[1] J. Audibert, S. Bubeck, and R. Munos. Best arm identification in multi-armed bandits. In *Proceedings of the Conference on Learning Theory*, 2010.

[2] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. Gambling in a rigged casino: the adversarial multi-armed bandit problem. Technical Report NC2-TR-1998-025, NeuroCOLT2 Technical Report Series, 1998.

[3] P. Boyle. *Gaussian Processes for Regression and Optimisation*. PhD thesis, Victoria University of Wellington, Wellington, New Zealand, 2007.

[4] E. Brochu, T. Brochu, and N. de Freitas. A Bayesian interactive optimization approach to procedural animation design. In *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, 2010.

[5] E. Brochu, V. M. Cora, and N. de Freitas. A tutorial on Bayesian optimization of expensive cost functions with application to active user modeling and hierarchical reinforcement learning. eprint arXiv:1012.2599, arXiv, 2010.

[6] E. Brochu, N. de Freitas, and A. Ghosh. Active preference learning with discrete choice data. In *Advances in Neural Information Processing Systems*, 2007.

[7] E. Brochu, M. Hoffman, and N. de Freitas. Technical Report arXiv:1009.5419, arXiv.

[8] S. Bubeck, R. Munos, and G. Stoltz. Pure exploration in multi-armed bandits problems. In *Algorithmic Learning Theory*, pages 23–37. Springer, 2009.

[9] A. D. Bull. Convergence rates of efficient global optimization algorithms. Technical Report arXiv:1101.3501v2, 2011.

[10] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, New York, 2006.

[11] K. Chaudhuri, Y. Freund, and D. Hsu. A parameter-free hedging algorithm. In *Advances in Neural Information Processing Systems*, 2009.

[12] W. Chu and Z. Ghahramani. Preference learning with Gaussian processes. In *Proc. 22nd International Conf. on Machine Learning*, 2005.

[13] D. D. Cox and S. John. SDO: A statistical method for global optimization. In M. N. Alexandrov and M. Y. Hussaini, editors, *Multidisciplinary Design Optimization: State of the Art*, pages 315–329. SIAM, 1997.

[14] P. J. Diggle, J. A. Tawn, and R. A. Moyeed. Model-based geostatistics. *Journal of the Royal Statistical Society. Series C*, 47(3):299–350, 1998.

[15] L. Dixon and G. Szegö. The global optimization problem: an introduction. *Towards Global Optimization*, 2, 1978.

[16] J. M. Gablonsky. *Modification of the DIRECT Algorithm*. PhD thesis, Department of Mathematics, North Carolina State University, Raleigh, 2001.

[17] S. Grunewalder, J. Audibert, M. Opper, and J. Shawe-Taylor. Regret bounds for Gaussian process bandit problems. In *Proceedings of the Conference on Artificial Intelligence and Statistics*, 2010.

[18] M. Hoffman, H. Kück, N. de Freitas, and A. Doucet. New inference strategies for solving Markov decision processes using reversible jump MCMC. In *Uncertainty in Artificial Intelligence*, 2009.

[19] D. Huang, T. T. Allen, W. I. Notz, and N. Zheng. Global optimization of stochastic black-box systems via sequential Kriging meta-models. *J. Global Optimization*, 3(34):441–466, March 2006.

[20] F. Hutter. *Automating the Configuration of Algorithms for Solving Hard Computational Problems*. PhD thesis, University of British Columbia, Vancouver, Canada, August 2009.

[21] D. R. Jones. A taxonomy of global optimization methods based on response surfaces. *J. Global Optimization*, 21:345–383, 2001.

[22] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *J. Optimization Theory and Apps*, 79(1):157–181, 1993.

[23] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *J. Global Optimization*, 13(4):455–492, 1998.

[24] H. J. Kushner. A new method of locating the maximum of an arbitrary multipeak curve in the presence of noise. *J. Basic Engineering*, 86:97–106, 1964.

[25] D. Lizotte. *Practical Bayesian Optimization*. PhD thesis, University of Alberta, Edmonton, Alberta, Canada, 2008.

[26] D. Lizotte, T. Wang, M. Bowling, and D. Schuurmans. Automatic gait optimization with Gaussian process regression. In *Proc. Intl. Joint Conf. on Artificial Intelligence (IJCAI)*, 2007.

[27] M. Locatelli. Bayesian algorithms for one-dimensional global optimization. *J. Global Optimization*, 1997.

[28] R. Martinez–Cantin, N. de Freitas, E. Brochu, J. Castellanos, and A. Doucet. A Bayesian exploration-exploitation approach for optimal online sensing and planning with a visually guided mobile robot. *Autonomous Robots*, 27(2):93–103, 2009.

[29] R. Martinez–Cantin, N. de Freitas, A. Doucet, and J. A. Castellanos. Active policy learning for robot planning and exploration under uncertainty. *Robotics: Science and Systems (RSS)*, 2007.

[30] J. Močkus, V. Tiesis, and A. Žilinskas. *Toward Global Optimization*, volume 2, chapter The Application of Bayesian Methods for Seeking the Extremum, pages 117–128. Elsevier, 1978.

[31] A. Y. Ng and M. I. Jordan. PEGASUS: A policy search method for large MDPs and POMDPs. In *Uncertainty in Artificial Intelligence (UAI2000)*, 2000.

[32] J. Nocedal and S. Wright. *Numerical optimization*. Springer Verlag, 1999.

[33] M. Osborne. *Bayesian Gaussian Processes for Sequential Prediction, Optimization and Quadrature*. PhD thesis, University of Oxford, 2010.

[34] C. E. Rasmussen. Gaussian processes to speed up hybrid Monte Carlo for expensive Bayesian integrals. In *Bayesian Statistics 7*, pages 651–659. Oxford University Press, 2003.

[35] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, Massachusetts, 2006.

[36] H. Rue, S. Martino, and N. Chopin. Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal Of The Royal Statistical Society Series B*, 71(2):319–392, 2009.

[37] M. Schonlau, W. J. Welch, and D. R. Jones. Global versus local search in constrained optimization of computer models. *Lecture Notes-Monograph Series*, 34:11–25, 1998.

[38] N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proc. Intl. Conf. on Machine Learning (ICML)*, 2010.

[39] E. Vasquez and J. Bect. Convergence properties of the expected improvement algorithm. *ArXiv*, (0712.3744v4), Dec 2007.

A SYNTHETIC TEST FUNCTIONS

As there is no generally-agreed-upon set of test functions for Bayesian optimization in higher dimensions, we seek to sample synthetic functions from a known GP prior, similar to the strategy of Lizotte [25]. A GP prior is infinite-dimensional, so on a practical level for performing experiments we simulate this by sampling points and using the posterior mean as the synthetic objective test function.

For each trial, we use an ARD kernel with $\boldsymbol{\theta}$ drawn uniformly from $[0, 2]^d$. We then sample $100d$ d-dimensional points, compute $\mathbf{K}$ and then draw $\mathbf{y} \sim \mathcal{N}(0, \mathbf{K})$. The posterior mean of the resulting predictive posterior distribution $\mu(\mathbf{x})$ (Section 2.1) is used as the test function. However it is possible that for particular values of $\boldsymbol{\theta}$ and $\mathbf{K}$, large parts of the space will be so far from the samples that they will form plateaus along the prior mean. To reduce this, we evaluate the test function at 500 random locations. If more than 25 of these are 0, we recompute $\mathbf{K}$ using $200d$ points. This process is repeated, adding $100d$ points each time until the test function passes the plateau test (this is rarely necessary in practice).

Using the response surface $\mu(\mathbf{x})$ as the objective function, we can then approximate the maximum using conventional global optimization techniques to get $f(\mathbf{x}^\star)$, which permits us to use the gap metric for performance.

Note that these sample points are only used to construct the objective, and are not known to the optimization methods.

B PROOF OF THEOREM 1

We will consider a portfolio-based strategy using rewards $r_t = \mu_{t-1}(\mathbf{x}_t)$ and selecting between acquisition functions using the Hedge algorithm. In order to discuss this we will need to write the gain over T steps, in hindsight, that would have been obtained had we used strategy i,

$$g_T^i = \sum_{t=1}^{T} r_t^i = \sum_{t=1}^{T} \mu_{t-1}(\mathbf{x}_t^i).$$

We must emphasize however that this gain is conditioned on the actual decisions made by Hedge, namely that points $\{\mathbf{x}_1, \dots, \mathbf{x}_{t-1}\}$ were selected by Hedge. If we define the maximum strategy $g_T^{\max} = \max_i g_T^i$ we can then bound the regret of Hedge with respect to this gain.

Lemma 1. *With probability at least $1 - \delta_1$ and for a suitable choice of Hedge parameters, $\eta = \sqrt{8 \ln k / T}$, the regret is bounded by*

$$g_T^{\max} - g_T^{\text{Hedge}} \leq \mathcal{O}(\sqrt{T}).$$

This result is given without proof as it follows directly from [10, Section 4.2] for rewards in the range $[0, 1]$. At the cost of slightly worsening the bound in terms of its multiplicative/additive constants, the following generalizations can also be noted:

- For rewards instead in the arbitrary range[2] $[a, b]$ the same bound can be shown by referring to [10, Section 2.6].
- The choice of η in the above Lemma requires knowledge of the time horizon T. By referring to [10, Section 2.3] we can remove this restriction using a time-varying term $\eta_t = \sqrt{8 \ln k / t}$.
- By referring to [10, Section 6.8] we can also extend this bound to the partial-information strategy Exp3.

Finally, we should also note that this regret bound trivially holds for any strategy i, since $g_T^{\max}$ is the maximum. It is also important to note that this lemma holds for any choice of r_t^i, *with rewards depending on the actual actions taken by Hedge*. The particular choice of rewards we use for this proof have been selected in order to achieve the following derivations.

For the next two lemmas we will refer the reader to [38, Lemma 5.1 and 5.3] for proof. We point out, however, that these two lemmas only depend on the underlying Gaussian process and as a result can be used separately from the GP-UCB framework.

Lemma 2. *Assume $\delta_2 \in (0, 1)$, a finite sample space $|A| < \infty$, and $\beta_t = 2 \log(|A| \pi_t / \delta_2)$ where $\sum_t \pi_t^{-1} = 1$ and $\pi_t > 0$. Then with probability at least $1 - \delta_2$ the absolute deviation of the mean is bounded by*

$$|f(\mathbf{x}) - \mu_{t-1}(\mathbf{x})| \leq \sqrt{\beta_t} \sigma_{t-1}(\mathbf{x}) \quad \forall x \in A, \forall t \geq 1.$$

In order to simplify this discussion we have assumed that the sample space A is finite, however this can also be extended to compact spaces [38, Lemma 5.7].

Lemma 3. *The information gain for points selected by the algorithm can be written as*

$$I(y_{1:T}; f_{1:T}) = \frac{1}{2} \sum_{t=1}^{T} \log(1 + \sigma^{-2} \sigma_{t-1}^2(\mathbf{x}_t)).$$

The following lemma follows the proof of [38, Lemma 5.4], however it can be applied outside the GP-UCB framework. Due to the slightly different conditions we recreate this proof here.

Lemma 4. *Given points x_t selected by the algorithm the following bound holds for the sum of variances:*

$$\sum_{t=1}^{T} \beta_t \sigma_t^2(\mathbf{x}_t) \leq C_1 \beta_T \gamma_T,$$

where $C_1 = 2 / \log(1 + \sigma^{-2})$.

Proof. Because β_t is nondecreasing we can write the following inequality

$$\begin{aligned} \beta_t \sigma_{t-1}^2(\mathbf{x}_t) &\leq \beta_T \sigma^2(\sigma^{-2} \sigma_{t-1}^2(\mathbf{x}_t)) \\ &\leq \beta_T \sigma^2 \frac{\sigma^{-2}}{\log(1 + \sigma^{-2})} \log(1 + \sigma^{-2} \sigma_{t-1}^2(\mathbf{x}_t)). \end{aligned}$$

The second inequality holds because the posterior variance is bounded by the prior variance, $\sigma_{t-1}^2(\mathbf{x}) \leq k(\mathbf{x}, \mathbf{x}) \leq 1$, which allows us to write

$$\sigma^{-2} \sigma_{t-1}^2(\mathbf{x}_t) \leq \sigma^{-2} \frac{\log(1 + \sigma^{-2} \sigma_{t-1}^2(\mathbf{x}_t))}{\log(1 + \sigma^{-2})}.$$

By summing over both sides of the original bound and applying the result of Lemma 3 we can see that

$$\begin{aligned} \sum_{t=1}^{T} \beta_t \sigma_{t-1}^2(\mathbf{x}_t) &\leq \beta_T \frac{1}{2} C_1 \sum_{t=1}^{T} \log(1 + \sigma^{-2} \sigma_{t-1}^2(\mathbf{x}_t)) \\ &= \beta_T C_1 I(y_{1:T}; f_{1:T}). \end{aligned}$$

The result follows by bounding the information gain by $I(y_{1:T}; f_{1:T}) \leq \gamma_T$, which can be done for many common kernels, including the squared exponential [38, Theorem 5]. □

[2] To obtain rewards bounded within some range $[a, b]$ we can assume that the additive noise ϵ_t is truncated above some large absolute value, which guarantees bounded means.

Finally, the next lemma follows directly from [38, Lemma 5.2]. We will note that this lemma depends only on the definition of the GP-UCB acquisition function, and as a result does not require that points at any previous iteration were actually selected via GP-UCB.

Lemma 5. *If the bound from Lemma 2 holds, then for a point $\mathbf{x}_t^{\text{UCB}}$ proposed by GP-UCB with parameters β_t the following bound holds:*

$$f(\mathbf{x}^*) - \mu_{t-1}(\mathbf{x}_t^{\text{UCB}}) \leq \sqrt{\beta_t}\sigma_{t-1}(\mathbf{x}_t^{\text{UCB}}).$$

We can now combine these results to construct the proof of Theorem 1.

Proof of Theorem 1. With probability at least $1 - \delta_1$ the result of Lemma 1 holds. If we assume that GP-UCB is included as one of the acquisition functions we can write

$$-g_T^{\text{Hedge}} \leq \mathcal{O}(\sqrt{T}) - g_T^{\text{UCB}}$$

and by adding $\sum_{t=1}^{T} f(\mathbf{x}^*)$ to both sides this inequality can be rewritten as

$$\sum_{t=1}^{T} f(\mathbf{x}^*) - \mu_{t-1}(\mathbf{x}_t) \leq \mathcal{O}(\sqrt{T}) + \sum_{t=1}^{T} f(\mathbf{x}^*) - \mu_{t-1}(\mathbf{x}_t^{\text{UCB}}).$$

With probability at least $1 - \delta_2$ the bound from Lemma 2 can be applied to the left-hand-side and the result of Lemma 5 can be applied to the right. This allows us to rewrite this inequality as

$$\begin{aligned}\sum_{t=1}^{T} f(\mathbf{x}^*) - f(\mathbf{x}_t) - \sqrt{\beta_t}\sigma_{t-1}(\mathbf{x}_t) \\ \leq \mathcal{O}(\sqrt{T}) + \sum_{t=1}^{T} \sqrt{\beta_t}\sigma_{t-1}(\mathbf{x}_t^{\text{UCB}})\end{aligned}$$

which means that the regret is bounded by

$$\begin{aligned}R_T &= \sum_{t=1}^{T} f(\mathbf{x}^*) - f(\mathbf{x}_t) \\ &\leq \mathcal{O}(\sqrt{T}) + \sum_{t=1}^{T} \sqrt{\beta_t}\sigma_{t-1}(\mathbf{x}_t^{\text{UCB}}) + \sum_{t=1}^{T} \sqrt{\beta_t}\sigma_{t-1}(\mathbf{x}_t) \\ &\leq \mathcal{O}(\sqrt{T}) + \sum_{t=1}^{T} \sqrt{\beta_t}\sigma_{t-1}(\mathbf{x}_t^{\text{UCB}}) + \sqrt{C_1 T \beta_T \gamma_T}.\end{aligned}$$

This final inequality follows directly from Lemma 4 by application of the Cauchy-Schwarz inequality. We should note that we cannot use Lemma 4 to further simplify the terms involving the sum over $\sigma_{t-1}(\mathbf{x}_t^{\text{UCB}})$. This is because the lemma only holds for points that are sampled by the algorithm, which may not include those proposed by GP-UCB.

Finally, this result depends upon Lemmas 1 and 5 holding. By a simple union bound argument we can see that these both hold with probability at least $1 - \delta_1 - \delta_2$, and by setting $\delta_1 = \delta_2 = \delta/2$ we recover our result. □