Multi-Information Source Optimization

Anonymous Author(s)
Affiliation
Address
email

Abstract

We consider Bayesian methods for multi-information source optimization (MISO), in which we seek to optimize an expensive-to-evaluate black-box objective function while also accessing cheaper but biased and noisy approximations ("information sources"). We present a novel algorithm that outperforms the state of the art for this problem by using a novel Gaussian process covariance kernel better suited to MISO than those used by previous approaches, and a novel acquisition function based on a one-step optimality analysis supported by efficient parallelization. We provide a guarantee on the asymptotic quality of the solution provided by this algorithm, and experimental evaluations demonstrate that this algorithm consistently finds designs of higher value at less cost than previous approaches.

1 Introduction

We consider Bayesian multi-information source optimization (MISO), in which we optimize an expensive-to-evaluate black-box objective function while optionally accessing cheaper biased noisy approximations, often referred to as "information sources (IS)". This arises when tuning hyperparameters of machine learning algorithms: one may evaluate hyperparameters on a smaller related dataset or subsets of the validation set [30, 16, 18]. This also arises in control in robotics: we can evaluate a parameterized robot control policy in simulation, in a laboratory, or in a field test [16]. Cheap approximations promise a route to tractability, but bias and noise complicate their use. While a number of existing methods are tolerant to noise in cheap approximations, few existing methods are designed to be tolerant to bias. Moreover, bias arises whenever a computational model incompletely models a real-world phenomenon, and is pervasive in applications.

We present a novel algorithm for this problem, `misoKG`, that is tolerant to both noise and bias and improves substantially over the state of the art. This improvement is enabled by two innovations:

- First, the algorithm uses a novel class of Gaussian process (GP) covariance kernels better suited to optimization with multiple IS than those used previously for MISO.
- Second, the algorithm uses a novel acquisition function that maximizes the incremental gain per unit cost. This acquisition function generalizes and parallelizes previously proposed knowledge-gradient methods for single-IS Bayesian optimization [7, 8, 25, 23, 33] to MISO.

We then prove that this algorithm provides an asymptotically near-optimal solution.

Related Work: To our knowledge, MISO was first considered by Swersky, Snoek, and Adams [30], under the the name multi-task Bayesian optimization. This name was used to suggest problems in which the auxiliary tasks could meaningfully be solved on their own, while we use the term MISO to indicate that the IS may be useful only in support of the primary task. Swersky et al. [30] showed that hyperparameter tuning in classification can be accelerated through evaluation on subsets of the validation data. They proposed a GP model to jointly model such "auxiliary tasks" and the primary task, building on previous work on GP regression for multiple tasks in [3, 10, 31]. They choose points to sample via cost-sensitive Entropy Search [12, 35], sampling in each iteration a point that maximally reduces uncertainty in the optimum's location, normalized by the query cost.

We demonstrate in experiments that our approach improves over the method of Swersky et al. [30], and we believe this improvement results from two factors: first, our statistical model is more flexible in its ability to model bias that varies across the domain; second, our acquisition function directly and maximally reduces simple regret in one step, unlike predictive entropy search which seeks to maximally reduce entropy in one step and only indirectly reduces regret.

Lam, Allaire, and Willcox [19] also considers MISO, under the name non-hierarchical multi-fidelity optimization. They propose a statistical model that maintains a separate GP for each IS, and fuse them via the method of Winkler [36]. They apply a modified expected improvement acquisition function on these surrogates to first decide what design x^* to evaluate and then select the IS to query; the latter is decided by a heuristic that aims to balance information gain and query cost. We demonstrate in experiments that our approach improves over the method of Lam et al. [19], and we believe this improvement results from two factors: first, their statistical approach assumes an independent prior on each IS, despite their being linked through modeling a common objective; and second their acquisition function selects the point to sample and the IS to query separately via a heuristic rather than jointly using an optimality analysis.

Beyond these two works, the most closely related work is in the related problem of multi-fidelity optimization. In this problem, IS may be noisy, but are assumed to be unbiased. While IS are truly unbiased in some applications (e.g., optimizing the output of a simulation or A/B test, when IS correspond to the number of independent samples performed), in many applications they are not. Thus, multi-fidelity methods can suffer limited applicability if we are only willing to use them when IS are truly unbiased, and degraded performance if we use them widely. Past work in this area includes [17, 5, 15, 6, 21, 16]. From this body of work, we compare against Kandasamy et al. [16], which presents an approach for minimizing both simple and cumulative regret. We demonstrate improved performance due to our ability to model bias.

Outside of both the MISO and multi-fidelity settings, Klein et al. [18] considered hyperparameter optimization of machine learning algorithms over a large dataset D. Supposing access to subsets of D of arbitrary sizes, they show how to exploit regularity of performance across dataset sizes to significantly speed up the optimization process for support vector machines and neural networks.

Our acquisition function is a generalization of the knowledge-gradient policy of Frazier, Powell, and Dayanik [8] to the MISO setting. This generalization requires extending the one-step optimality analysis used to derive the KG policy in the single-IS setting to account for the impact of sampling a cheap approximation on the marginal GP posterior on the primary task. From this literature, we leverage a computational method for computing the expectation of the maximum of a collection of linear functions of a normal random variable.

The class of GP covariance kernels we propose are a subset of the class of linear models of coregionalization kernels [10, 2], with a restricted form derived from a generative model particular to MISO. Focusing on a restricted class of kernels designed for our application supports accurate inference with less data, which is important when optimizing expensive-to-evaluate functions.

Our work also extends the knowledge-gradient acquisition function to the variable cost setting. Similar extensions of expected improvement to the variable cost setting can be found in Snoek, Larochelle, and Adams [28] (the EI per second criterion) or Gratiet and Cannamela [11].

We now formalize the problem we consider in Sect. 2, describe our statistical analysis in Sect. 3.1, specify our acquisition function and parallel computation method in Sects. 3.2 and 3.3, provide a theoretical guarantee in Sect. 3.4, present numerical experiments in Sect. 4, and conclude in Sect. 5.

2 Problem Formulation

We wish to find a best design $\mathrm{argmax}_{x \in \mathcal{D}}\, g(x)$ under a continuous objective function $g : \mathcal{D} \to \mathbb{R}$ on a compact set $\mathcal{D} \subset \mathbb{R}^d$ of feasible designs. We have access to M possibly biased and/or noisy IS indexed by $\ell \in [M]_0$. (Here, for any $a \in \mathbb{Z}^+$ we use $[a]$ as a shorthand for the set $\{1, 2, \ldots, a\}$, and further define $[a]_0 = \{0, 1, 2, \ldots, a\}$.) Observing IS ℓ at design x provides independent and normally distributed observations with mean value $f(\ell, x)$ and variance $\lambda_\ell(x)$. In [30], IS $\ell \in [M]_0$ are called "auxiliary tasks" and g the primary task. These sources are thought of as approximating g, with variable bias. We suppose that g can be observed directly without bias (but possibly with noise) and set $f(0, x) = g(x)$. The bias $g(x) - f(\ell, x)$ is also referred to as "model discrepancy" in the engineering and simulation literature [1, 4]. Each IS ℓ is also associated with a query cost function $c_\ell(x) : \mathcal{D} \to \mathbb{R}^+$. We assume that the cost function $c_\ell(x)$ and the variance function $\lambda_\ell(x)$ are both known and continuously differentiable. In practice, these functions may either be provided by domain experts or may be estimated along with other model parameters from data (see Sect. 4 and A.2, and [24]).

3 The misoKG Algorithm

We now present the misoKG algorithm and describe its two novel components: a MISO-focused statistical model in Sect. 3.1; and its acquisition function and parallel computation in Sect. 3.2. Sect. 3.3 summarizes the algorithm and Sect. 3.4 provides a theoretical performance guarantee.

3.1 Statistical Model

We now describe a generative model for f that results in a Gaussian process prior on f with a parameterized class of mean functions $\mu : [M] \times \mathcal{D} \mapsto \mathbb{R}$ and covariance kernels $\Sigma : ([M] \times \mathcal{D})^2 \mapsto \mathbb{R}$. This allows us to use standard tools for Gaussian process inference — first finding the MLE estimate of the parameters indexing this class, and then performing Gaussian process regression using the selected mean function and covariance kernel — while also providing better estimates for MISO than would a generic multi-output GP regression kernel that does not consider the MISO application.

We construct our generative model as follows. For each $\ell > 0$ suppose that a function $\delta_\ell : \mathcal{D} \mapsto \mathbb{R}$ for each $\ell > 0$ was drawn from a separate independent GP, $\delta_\ell \sim GP(\mu_\ell, \Sigma_\ell)$, and let δ_0 be identically 0. In our generative model δ_ℓ will be the bias $g(x) - f(\ell, x)$ for IS ℓ. We additionally set $\mu_\ell(x) = 0$ to encode a lack of a strong belief on the direction of the bias. (If one had a strong belief that an IS is consistently biased in one direction, one may instead set μ_ℓ to a constant estimated using maximum a posteriori estimation.) Next, within our generative model, we suppose that $g : \mathcal{D} \mapsto \mathbb{R}$ was drawn from its own independent GP, $g \sim GP(\mu_0, \Sigma_0)$, for some given μ_0 and Σ_0, and suppose $f(\ell, x) = f(0, x) + \delta_\ell(x)$ for each ℓ. We assume that μ_0 and Σ_ℓ $\ell \geq 0$ belong to one of the standard parameterized classes of mean functions and covariance kernels, e.g., constant μ_0 and Matérn Σ_ℓ.

With this construction, f is a GP: given any finite collection of points $\ell_i \in [M], x_i \in \mathcal{D}$ with $i = 1, \ldots, I$, $(f(\ell_i, x_i) : i = 1, \ldots, I)$ is a sum of independent multivariate normal random vectors, and thus is itself multivariate normal. Moreover, we compute the mean function and covariance kernel of f: for $\ell, m \in [M]_0$ and $x, x' \in \mathcal{D}$,

$$\mu(\ell, x) = \mathbb{E}\left[f(\ell, x)\right] = \mathbb{E}\left[g(x)\right] + \mathbb{E}\left[\delta_\ell(x)\right] = \mu_0(x)$$
$$\Sigma\left((\ell, x), (m, x')\right) = \mathrm{Cov}(g(x) + \delta_\ell(x), g(x') + \delta_m(x')) = \Sigma_0(x, x') + \mathbb{1}_{\ell,m} \cdot \Sigma_\ell(x, x'),$$

where $\mathbb{1}_{\ell,m}$ denotes Kronecker's delta, and where we have used independence of δ_ℓ, δ_m, and g.

This generative model draws model discrepancies δ_ℓ independently across IS. This is appropriate when IS are different in kind and share no relationship except that they model a common objective. In the supplement (Sect. A) we generalize this generative model to model correlation between model discrepancies, which is appropriate when IS can be partitioned into groups, such that IS within one group tend to agree more amongst themselves than they do with IS in other groups. Sect. A also discusses estimation of the hyperparameters in μ_0 and Σ_ℓ.

3.2 Acquisition Function

Our optimization algorithm proceeds in rounds, selecting a design $x \in \mathcal{D}$ and an information source $\ell \in [M]_0$ in each. The value of the information obtained by sampling IS ℓ at x is the expected gain in the quality of the best design that can be selected using the available information. That is, this value is the difference in the expected quality of the estimated optimum before and after the sample. We then normalize this expected gain by the cost $c_\ell(x)$ associated with the respective query, and sample the IS and design with the largest normalized gain. Without normalization we would always query the true objective, since no other IS provides more information about g than g itself.

We formalize this idea. Suppose that we have already sampled n points X_n and made the observations Y_n. Denote by $\mathbb{E}_n$ the expected value according to the posterior distribution given X_n, Y_n, and let $\mu^{(n)}(\ell, x) := \mathbb{E}_n[f(\ell, x)]$. The best *expected* objective value across the designs, as estimated by our statistical model, is $\max_{x' \in \mathcal{D}} \mu^{(n)}(0, x')$. Similarly, if we take an additional sample of IS $\ell^{(n+1)}$ at design $x^{(n+1)}$ and compute our new posterior mean, the new best expected objective value across the designs is $\max_{x' \in \mathcal{D}} \mu^{(n+1)}(0, x')$, whose distribution depends on what IS we sample, and where sample it. Thus, the expected value of sampling at (ℓ, x) normalized by the cost is

$$\mathrm{CKG}(\ell, x) = \mathbb{E}_n\left[\frac{\max_{x' \in \mathcal{D}} \mu^{(n+1)}(0, x') - \max_{x' \in \mathcal{D}} \mu^{(n)}(0, x')}{c_\ell(x)} \mid \ell^{(n+1)} = \ell, x^{(n+1)} = x\right], \tag{1}$$

which we refer to as the *cost-sensitive Knowledge Gradient factor* of the pair (ℓ, x). The *cost-sensitive Knowledge Gradient* policy (csKG) then samples at the pair (ℓ, x) that maximizes $\mathrm{CKG}(\ell, x)$, i.e., $(\ell^{(n+1)}, x^{(n+1)}) \in \mathrm{argmax}_{\ell \in [M]_0, x \in \mathcal{D}} \mathrm{CKG}(\ell, x)$, which is a nested optimization problem.

To make this nested optimization problem tractable, we first replace the search domain $\mathcal{D}$ in (1) by a discrete set $\mathcal{A} \subset \mathcal{D}$ of points, for example selected by a Latin Hypercube design. We may then compute $\mathrm{CKG}(\ell, x)$ exactly. Toward that end, note that

$$\begin{aligned}&\mathbb{E}_n\left[\max_{x' \in \mathcal{A}} \mu^{(n+1)}(0, x') \mid \ell^{(n+1)} = \ell, x^{(n+1)} = x\right] \\ &\qquad = \mathbb{E}_n\left[\max_{x' \in \mathcal{A}} \mu^{(n)}(0, x') + \bar{\sigma}^n_{x'}(\ell, x) \cdot Z \mid \ell^{(n+1)} = \ell, x^{(n+1)} = x\right],\end{aligned}$$

where $Z \sim \mathcal{N}(0, 1)$ and $\bar{\sigma}^n_{x'}(\ell, x) = \Sigma^n((0, x'), (\ell, x)) / \left[\lambda_\ell(x) + \Sigma^n((\ell, x), (\ell, x))\right]^{\frac{1}{2}}$. Here Σ^n is the posterior covariance matrix of f given X_n, Y_n.

We parallelize the computation of $\mathrm{CKG}(\ell, x)$ for fixed ℓ, x, enabling it to utilize multiple cores. Then $(\ell^{(n+1)}, x^{(n+1)})$ is obtained by computing $\mathrm{CKG}(\ell, x)$ for all $(\ell, x) \in [M]_0 \times \mathcal{A}$, a task that can be parallelized over multiple machines in a cluster. We begin by sorting the points in $\mathcal{A}$ in parallel by increasing value of $\bar{\sigma}^n_{x'}(\ell, x)$ (for fixed ℓ, x). Thereby we remove dominated points: a point x_j is *dominated* by x_k if $\bar{\sigma}^n_{x_j}(\ell, x) = \bar{\sigma}^n_{x_k}(\ell, x)$ and $\mu^{(n)}(0, x_j) \le \mu^{(n)}(0, x_k)$, since x_k has a higher expected value than x_j for any realization of Z. Let $S = \{x_1, \ldots, x_{|\mathcal{A}|}\}$ be the sorted sequence without dominated points.

Since $c_\ell(x)$ is a constant for fixed ℓ, x, we may express the conditional expectation in (1) as $\mathbb{E}_n\left[\frac{\max_i a_i + b_i Z - \max_i a_i}{c_\ell(x)}\right] = \frac{\mathbb{E}_n[\max_i a_i + b_i Z - \max_i a_i]}{c_\ell(x)}$, where $a_i = \mu^{(n)}(0, x_i)$ and $b_i = \bar{\sigma}^n_{x_i}(\ell, x)$ for $i \in [|\mathcal{A}|]$. We split S into consecutive sequences $S_1, S_2, \ldots, S_C$, where C is the number of cores used for $\mathrm{CKG}(\ell, x)$ and S_i, S_{i+1} overlap in one element: that is, for $S_j = \{x_{j_1}, \ldots, x_{j_k}\}$, $x_{(j-1)_k} = x_{j_1}$ and $x_{j_k} = x_{(j+1)_1}$ hold. Each $x_{j_i} \in S_j$ specifies a linear function $a_{j_i} + b_{j_i} Z$ (ordered by increasing slopes in S). We are interested in the realizations of Z for which $a_{j_i} + b_{j_i} Z \ge a_{i'} + b_{i'} Z$ for any i' and hence compute the intersections of these functions. The functions for x_{j_i} and $x_{j_{i+1}}$ intersect in $c_{j_i} = (a_{j_i} - a_{j_{i+1}}) / (b_{j_{i+1}} - b_{j_i})$. Observe if $c_{j_i} \le c_{j_{i-1}}$, then $a_{j_i} + b_{j_i} Z \le \max\{a_{j_{i-1}} + b_{j_{i-1}} Z, a_{j_{i+1}} + b_{j_{i+1}} Z\}$ for all Z: x_{j_i} is dominated and hence dropped from S_j. Points in S_j may be dominated by the rightmost point in S_{j-1}. Thus, we compare $c_{(j-1)_k}$ with $c_{j_1}, c_{j_2}, \ldots$ and drop all points of S_j whose c-value is smaller than $c_{(j-1)_k}$. If all points of S_j are dominated, we continue the scan with S_{j+1} and so on. Observe that we may stop scanning once there is a point that is not dominated, since the points in any sequence S_j have non-decreasing c-values. Subsequently, we check the other direction, i.e. whether x_{j_1} dominates elements of S_{j-1}, starting with the rightmost element of S_{j-1}. These checks for dominance are performed in parallel for neighboring sequences.

Eq. (14) in [8, p. 605] gives $\mathbb{E}_n[\max_i a_i + b_i Z - \max_i a_i] = \sum_{j=1}^{C} \sum_{h=1}^{k-1} (b_{j_{h+1}} - b_{j_h}) h(-|c_{j_h}|)$, where h is defined as $h(z) = z\Phi(z) + \phi(z)$ for the CDF and PDF of the normal distribution. The inner sums are computed simultaneously; the computation of the outer sum could be parallelized by recursively adding pairs of inner sums, although we do not do so to avoid communication overhead. We summarize the parallel algorithm below.

The Parallel Algorithm to compute $(\ell^{(n+1)}, x^{(n+1)})$:

1. Scatter the pairs $(\ell, x) \in [M]_0 \times \mathcal{A}$ among the machines. Each computes $\text{CKG}(\ell, x)$ for its pairs. Then determine $(\ell^{(n+1)}, x^{(n+1)}) \in \text{argmax}_{\ell \in [M]_0, x \in \mathcal{D}} \text{CKG}(\ell, x)$ in parallel.
2. To compute $\text{CKG}(\ell, x)$ in parallel:

a. Sort the points in $\mathcal{A}$ by ascending $\bar{\sigma}^n_{x'}(\ell, x)$ in parallel, thereby removing dominated points. Let $\{x_1, \ldots, x_{|\mathcal{A}|}\}$ be the sorted sequence.

b. Split $\{x_1, \ldots, x_{|\mathcal{A}|}\}$ into sequences $S_1, \ldots, S_C$, where C is the number of cores used to compute $\text{CKG}(\ell, x)$.

c. Each core computes $\sum_{x_i \in S_C} (b_{i+1} - b_i) h(-|c_i|)$ in parallel, then the partial sums are added to obtain $\mathbb{E}_n \left[\max_i a_i + b_i Z - \max_i a_i\right]$.

3.3 Summary of the `misoKG` Algorithm.

1. Using samples from all information sources, estimate hyperparameters of the Gaussian process prior as described in Sect. A.2.

   Then calculate the posterior on f based on the prior and samples.
2. Until the budget for samples is exhausted do:

   Determine the information source $\ell \in [M]_0$ and the design $x \in \mathcal{D}$ that maximize the cost-sensitive Knowledge Gradient proposed in Eq. (1) and observe IS $\ell(x)$.

   Update the posterior distribution with the new observation.
3. Return $\text{argmax}_{x' \in \mathcal{A}} \mu^{(n)}(0, x')$.

3.4 Provable Performance Guarantees.

The `csKG` chooses an IS and an x such that the expected gain normalized by the query cost is maximized. Thus, `csKG` is one-step Bayes optimal in this respect, by construction.

We establish an *additive bound* on the difference between `csKG`'s solution and the unknown optimum, as the number of queries $N \to \infty$. For this argument we suppose that $\mu(\ell, x)=0 \; \forall \ell, x$ and Σ is either the squared exponential kernel or a four times differentiable Matérn kernel. Then $f \sim GP(\mu, \Sigma)$ is twice continuously differentiable with probability one (e.g., see Ghosal and Roy [9]). Moreover, let $x^{\text{OPT}} \in \text{argmax}_{x' \in \mathcal{D}} f(0, x')$, and $d = \max_{x' \in \mathcal{D}} \min_{x'' \in \mathcal{A}} \text{dist}(x', x'')$.

Theorem 1. *For each $p \in [0, 1)$ there is constant K_p, such that the cost-sensitive Knowledge Gradient recommends an $x^*_N \in \mathcal{A}$, such that with probability p*

$$\lim_{N \to \infty} f(0, x^*_N) \geq f(0, x^{\text{OPT}}) - K_p \cdot d.$$

The proof was deferred to Sect. B due to space constraints. Note that we can compute K_p given Σ and p. Moreover, we can control the additive error $K_p \cdot d$ by increasing the density of $\mathcal{A}$, leveraging that the scalability of our parallel algorithm.

4 Numerical Experiments

We now compare `misoKG` to other state-of-the-art MISO algorithms. We implemented `misoKG`'s statistical model and acquisition function in `Python 2.7` and `C++` leveraging functionality from the `Metrics Optimization Engine` [22]. We used a multi-start gradient-based optimizer [25] to compute $(\ell^{(n+1)}, x^{(n+1)})$. This implementation is available at [URL blinded until acceptance].

We compare to `misoEI` of Lam et al. [19] and to `MTBO+`, an improved version of Multi-Task Bayesian Optimization proposed by Swersky et al. [30]. Following a recommendation of Snoek 2016, our implementation of `MTBO+` uses an improved formulation of the acquisition function given by Hernández-Lobato et al. [13], Snoek and et al. [27], but otherwise is identical to `MTBO`; in particular, it uses the statistical model of [30]. Sect. C provides detailed descriptions of these algorithms.

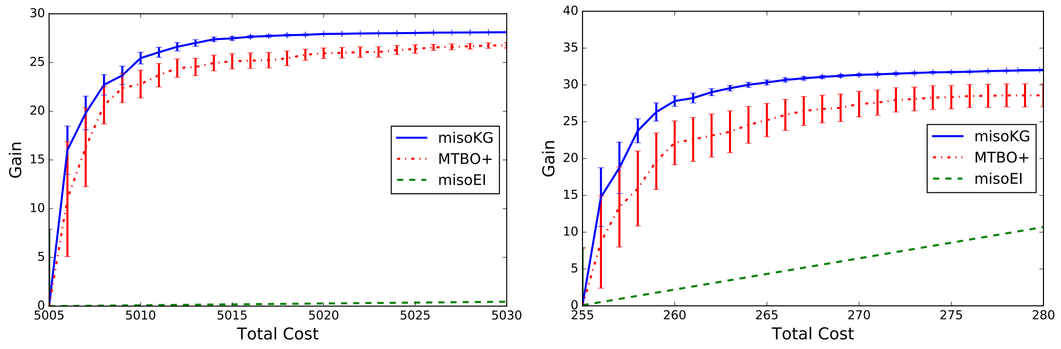

Figure 1: (l) The Rosenbrock benchmark with the parameter setting of [19]: `misoKG` offers an excellent gain-to-cost ratio and outperforms its competitors substantially. (r) The Rosenbrock benchmark with the alternative setup.

Experimental Setup. We conduct experiments on the following test problems: (1) the 2-dimensional Rosenbrock function modified to fit the MISO setting by Lam et al. [19]; (2) a MISO benchmark proposed by Swersky et al. [30] in which we optimize the 4 hyperparameters of a machine learning algorithm, using a small, related set of smaller images as cheap IS; (3) an assemble-to-order problem from Hong and Nelson [14] in which we optimize an 8-dimensional target stock vector to maximize the expected daily profit of a company as estimated by a simulator.

In MISO settings the amount of initial data that one can use to inform the methods about each information source is typically dictated by the application, in particular by resource constraints and the availability of the respective source. In our experiments all methods were given *identical initial datasets* for each information source in every replication; these sets were drawn randomly via Latin Hypercube designs. For the sake of simplicity, we provided the same number of points for each IS, deliberately set in advance to 2.5 points per dimension of the design space $\mathcal{D}$. Regarding the kernel and mean function, `MTBO+` uses the settings provided in [27]. The other algorithms used the squared exponential kernel and a constant mean function set to the average of a random sample.

We report the "gain" over the best initial solution, that is the true objective value of the respective design that a method would return at each iteration minus the best value in the initial data set. If the true objective value is not known for a given design, we report the value obtained from the information source of highest fidelity. This gain is plotted as a function of the *total cost*, that is the cumulative cost for invoking the information sources plus the fixed cost for the initial data; this metric naturally generalizes the number of function evaluations prevalent in Bayesian optimization. Note that the computational overhead of choosing the next information source and sample is omitted, as it is negligible compared to invoking an information source in real-world applications. Error bars are shown at the mean $\pm$ 2 standard errors averaged over at least 100 runs of each algorithm. For deterministic sources a jitter of 10^{-6} is added to avoid numerical issues during matrix inversion.

4.1 The Rosenbrock Benchmarks

We consider the design space $\mathcal{D} = [-2, 2]^2$, and $M = 2$ information sources. IS 0 is the Rosenbrock function $g(\boldsymbol{x}) = (1 - x_1)^2 + 100 \cdot (x_2 - x_1^2)^2$ plus optional Gaussian noise $u \cdot \varepsilon$. IS 1 returns $g(\boldsymbol{x}) + v \cdot \sin(10 \cdot x_1 + 5 \cdot x_2)$, where the additional oscillatory component serves as model discrepancy. We assume a cost of 1000 for each query to IS 0 and a cost of 1 for IS 1.

Since all methods converged to good solutions within few queries, we investigate the ratio of gain to cost: Fig. 1 (l) displays the gain of each method over the best initial solution as a function of the total cost inflicted by querying information sources. The new method `misoKG` offers a significantly better gain per unit cost and finds an almost optimal solution typically within $5 - 10$ samples. Interestingly, `misoKG` relies only on cheap samples, proving its ability to successfully handle the uncertainties. `MTBO+`, on the other hand, struggles initially but then eventually obtains a near-optimal solution, too. To this end, it makes usually one or two queries of the expensive truth source after about 40 steps. `misoEI` shows a odd behavior: it takes several queries, one of them to IS 0, before it improves over

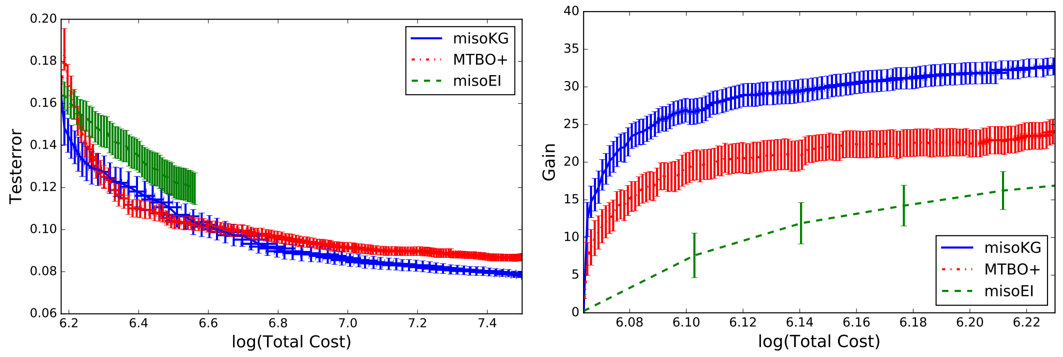

Figure 2: (l) The performance on the image classification benchmark of [30]. `misoKG` achieves better test errors after about 80 steps and converges to the global optimum. (r) `misoKG` outperforms the other algorithms on the assemble-to-order benchmark that has significant model discrepancy.

the best initial design for the first time. Then it jumps to a very good solution and subsequently samples only the cheap IS.

For the second setup, we set $u = 1$, $v = 2$, and suppose for IS 0 uniform noise of $\lambda_0(x) = 1$ and query cost $c_0(x) = 50$. Now the difference in costs is much smaller, while the variance is considerably bigger. The results are displayed in Fig. 1 (r): as for the first configuration, `misoKG` outperforms the other methods from the start. Interestingly, `misoEI`'s performance is drastically decreased compared to the first setup, since it only queries the expensive truth. Looking closer, we see that `misoKG` initially queries only the cheap information source IS 1 until it comes close to an optimal value after about five samples. It starts to query IS 0 occasionally later.

4.2 The Image Classification Benchmark

This classification problem was introduced by Swersky et al. [30] to demonstrate the performance of `MTBO`. The goal is to optimize four hyperparameters of the Logistic Regression algorithm [32] using a stochastic gradient method with mini-batches (the learning rate, the L2-regularization parameter, the batch size, and the number of epochs) in order to minimize the classification error on the MNIST dataset [20]. This dataset contains 70,000 images of handwritten digits: each image has 784 pixels. IS 1 uses the USPS dataset [34] of about 9000 images with 256 pixels each. The query costs are 4.5 for IS 1 and 43.69 for IS 0. A closer examination shows that IS 1 is subject to considerable bias with respect to IS 0, making it a challenge for MISO algorithms.

Fig.2 (l) summarizes the performances: initially, `misoKG` and `MTBO+` are on par. Both clearly outperform `misoEI` that hence we stopped after 50 iterations. `misoKG` and `MTBO+` continued for 150 steps (with a lower number of replications). `misoKG` usually achieves an optimal testerror of about 7.1% on the MNIST testset after about 80 queries, matching the classification performance of the best setting reported by Swersky et al. [30]. Moreover, `misoKG` achieves better solutions than `MTBO+` at the same costs. Note that the results in [30] show that `MTBO+` will also converge to the optimum eventually.

4.3 The Assemble-To-Order Benchmark

The assemble-to-order (ATO) benchmark is a reinforcement learning problem from a business application where the goal is to optimize an 8-dimensional target level vector over $[0, 20]^8$ (see Sect. D for details). We set up three information sources: IS 0 and 2 use the discrete event simulator of Xie et al. [37], whereas the cheapest source IS 1 invokes the implementation of Hong and Nelson. IS 0 models the truth.

The two simulators differ subtly in the model of the inventory system. However, the effect in estimated objective value is significant: on average the outputs of both simulators at the same target vector differ by about 5% of the score of the global optimum, which is about 120, whereas the largest observed bias out of 1000 random samples was 31.8. Thus, we are witnessing a significant model discrepancy.

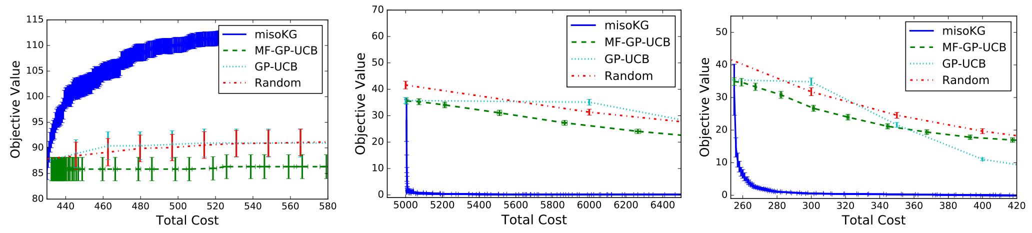

Figure 3: Evaluation of `misoKG`, `MF-GP-UCB`, `GP-UCB`, and `Random` on (l) the assemble-to-order benchmark, (m) the MISO Rosenbrock of [19], and (r) the alternative MISO Rosenbrock benchmark.

Fig. 2 (r) summarizes the performances. `misoKG` outperforms the other algorithms from the start: `misoKG` averages at a gain of 26.1, but inflicts only an average query cost of 54.6 to the information sources. This is only 6.3% of the query cost that `misoEI` requires to achieve a comparable score. Interestingly, `misoKG` and `MTBO+` utilize mostly the cheap biased IS, and therefore are able to obtain significantly better gain to cost ratios than `misoEI`. `misoKG`'s typically first calls IS 2 after about $60 - 80$ steps. In total, `misoKG` queries IS 2 about ten times within the first 150 steps; in some replications `misoKG` makes one late call to IS 0 when it has already converged. Our interpretation is that `misoKG` exploits the cheap, biased IS 1 to zoom in on the global optimum and switches to the unbiased but noisy IS 2 to identify the optimal solution exactly. This is the expected (and desired) behavior for `misoKG` when the uncertainty of $f(0, x^*)$ is not expected to be reduced sufficiently by queries to IS 1. `MTBO+` trades off the gain versus cost differently: it queries IS 0 once or twice after 100 steps and directs all other queries to IS 1, which might explain the observed lower performance. `misoEI` that employs a two-step heuristic for trading off predicted gain and query cost chose almost always the most expensive simulation to evaluate the selected design.

4.4 Comparison to `MF-GP-UCB`, `GP-UCB`, and `Random`

We compare `misoKG` to the multi-fidelity method `MF-GP-UCB` of Kandasamy et al. [16] and to the single-fidelity methods `GP-UCB` of Srinivas et al. [29] and `Random` that picks points randomly. `misoKG` and `MF-GP-UCB` received the same number of initial samples from each IS; `GP-UCB` and `Random` obtain the same number of samples from IS 0 as `misoKG`. Fig. 3 summarizes the performances. We see that `misoKG` clearly outperforms the other algorithms on all benchmarks, leveraging cheap IS with great success. `MF-GP-UCB` on the other hand is not able to benefit from cheap approximations; it performs slightly worse than the single-fidelity methods, which suggests that the bias misleads the algorithm. This demonstrates that the performance of multi-fidelity methods degrades in the presence of model discrepancy. `GP-UCB` and `Random` only query the expensive IS 0 and hence cannot be competitive.

5 Conclusion

We have presented a novel algorithm for MISO that uses a novel mean function and covariance matrix motivated by a MISO-specific generative model. We have also proposed a novel acquisition function that extends the knowledge gradient to the MISO setting and comes with a fast parallel method for computing it. Moreover, we have provided a theoretical guarantee on the solution quality delivered by this algorithm, and demonstrated through numerical experiments that it improves significantly over the state of the art.

References

[1] D. Allaire and K. Willcox. A mathematical and computational framework for multifidelity design and analysis with computer models. *International Journal for Uncertainty Quantification*, 4(1), 2014.

[2] M. A. Alvarez, L. Rosasco, and N. D. Lawrence. Kernels for vector-valued functions: A review. *Foundations and Trends in Machine Learning*, 4(3):195–266, 2012.

[3] E. V. Bonilla, K. M. Chai, and C. Williams. Multi-task gaussian process prediction. In *Advances in Neural Information Processing Systems*, pages 153–160, 2007.

[4] J. Brynjarsdottir and A. O'Hagan. Learning about physical parameters: the importance of model discrepancy. *Inverse Problems*, 30(11), 2014.

[5] M. S. Eldred, A. A. Giunta, and S. S. Collis. Second-order corrections for surrogate-based optimization with model hierarchies. In *Proceedings of the 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, pages 2013–2014, 2004.

[6] A. I. Forrester, A. Sóbester, and A. J. Keane. Multi-fidelity optimization via surrogate modelling. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 463(2088):3251–3269, 2007.

[7] P. I. Frazier, W. B. Powell, and S. Dayanik. A knowledge-gradient policy for sequential information collection. *SIAM Journal on Control and Optimization*, 47(5):2410–2439, 2008.

[8] P. I. Frazier, W. B. Powell, and S. Dayanik. The Knowledge Gradient Policy for Correlated Normal Beliefs. *INFORMS Journal on Computing*, 21(4):599–613, 2009.

[9] S. Ghosal and A. Roy. Posterior consistency of Gaussian process prior for nonparametric binary regression. *The Annals of Statistics*, 34(5):2413–2429, 2006.

[10] P. Goovaerts. *Geostatistics for Natural Resources Evaluation*. Oxford University, 1997.

[11] L. L. Gratiet and C. Cannamela. Cokriging-based sequential design strategies using fast cross-validation techniques for multi-fidelity computer codes. *Technometrics*, 57(3):418–427, 2015.

[12] P. Hennig and C. J. Schuler. Entropy search for information-efficient global optimization. *The Journal of Machine Learning Research*, 13(1):1809–1837, 2012.

[13] J. M. Hernández-Lobato, M. W. Hoffman, and Z. Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *Advances in Neural Information Processing Systems*, pages 918–926, 2014.

[14] L. J. Hong and B. L. Nelson. Discrete optimization via simulation using compass. *Operations Research*, 54(1):115–129, 2006.

[15] D. Huang, T. Allen, W. Notz, and R. Miller. Sequential kriging optimization using multiple-fidelity evaluations. *Structural and Multidisciplinary Optimization*, 32(5):369–382, 2006.

[16] K. Kandasamy, G. Dasarathy, J. B. Oliva, J. Schneider, and B. Poczos. Gaussian process bandit optimisation with multi-fidelity evaluations. In *Advances in Neural Information Processing Systems*, 2016. The code is available at `https://github.com/kirthevasank/mf-gp-ucb`. Last Accessed on 04/22/2017.

[17] M. C. Kennedy and A. O'Hagan. Predicting the output from a complex computer code when fast approximations are available. *Biometrika*, 87(1):1–13, 2000.

[18] A. Klein, S. Falkner, S. Bartels, P. Hennig, and F. Hutter. Fast bayesian optimization of machine learning hyperparameters on large datasets. *CoRR*, abs/1605.07079, 2016.

[19] R. Lam, D. Allaire, and K. Willcox. Multifidelity optimization using statistical surrogate modeling for non-hierarchical information sources. In *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2015.

[20] Y. LeCun, C. Cortes, and C. J. Burges. The MNIST database of handwritten digits, 2017. `http://yann.lecun.com/exdb/mnist/`. Last Accessed on 05/15/2017.

[21] L. Leifsson and S. Koziel. Multi-fidelity design optimization of transonic airfoils using physics-based surrogate modeling and shape-preserving response prediction. *Journal of Computational Science*, 1(2):98–106, 2010.

[22] MOE. Metrics optimization engine. `http://yelp.github.io/MOE/`, 2016. Last Accessed on 05/15/2017.

[23] H. Qu, I. O. Ryzhov, M. C. Fu, and Z. Ding. Sequential selection with unknown correlation structures. *Operations Research*, 63(4):931–948, 2015.

[24] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006. ISBN ISBN 0-262-18253-X.

[25] W. R. Scott, P. I. Frazier, and W. B. Powell. The correlated knowledge gradient for simulation optimization of continuous parameters using gaussian process regression. *SIAM Journal on Optimization*, 21(3):996–1026, 2011.

[26] J. Snoek. Personal communication, 2016.

[27] J. Snoek and et al. Spearmint. `http://github.com/HIPS/Spearmint`, 2017. Last Accessed on 05/15/2017.

[28] J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, pages 2951–2959, 2012.

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

[30] K. Swersky, J. Snoek, and R. P. Adams. Multi-task bayesian optimization. In *Advances in Neural Information Processing Systems*, pages 2004–2012, 2013.

[31] Y.-W. Teh, M. Seeger, and M. Jordan. Semiparametric latent factor models. In *Artificial Intelligence and Statistics 10*, 2005.

[32] Theano. Theano: Logistic regression, 2017. `http://deeplearning.net/tutorial/code/logistic_sgd.py`. Last Accessed on 05/16/2017.

[33] S. Toscano-Palmerin and P. I. Frazier. Stratified bayesian optimization. *ArXiv e-print 1602.02338*, 2016.

[34] USPS. USPS dataset, 2017. `http://mldata.org/repository/data/viewslug/usps/`. Last Accessed on 05/16/2017.

[35] J. Villemonteix, E. Vazquez, and E. Walter. An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, 44(4):509–534, 2009.

[36] R. L. Winkler. Combining probability distributions from dependent information sources. *Management Science*, 27(4):479–488, 1981.

[37] J. Xie, P. I. Frazier, and S. Chick. Assemble to order simulator. `http://simopt.org/wiki/index.php?title=Assemble_to_Order&oldid=447`, 2012. Last Accessed on 05/16/2017.

A The Model Revisited

A.1 Correlated Model Discrepancies

Next we demonstrate that our approach is flexible and can easily be extended to scenarios where some of the information sources have correlated model discrepancies. This arises for hyperparameter tuning if the auxiliary tasks are formed from data that was collected in batches and thus is correlated over time.

In engineering sciences we witness this if some sources share a common modeling approach, as for example, if one set of sources for an airfoil modeling problem correspond to different discretizations of a PDE that models wing flutter, while another set provides various discretizations of another PDE that modeling airflow. Two information sources that solve the same PDE will be more correlated than two that solve different PDEs.

Additionally, experiments conducted in the same location are exposed to the same environmental conditions or singular events, thus the outputs of these experiments might deviate from the truth by more than independent measurement errors. Another important factor is humans involved in the lab work, as typically workers have received the comparable training and may have made similar experiences during previous joint projects, which influences their actions and decisions.

For example, let $P = \{P_1, \ldots, P_Q\}$ denote a partition of $[M]_0$ and define the function $k : [M]_0 \to [Q]$ that gives for each IS its corresponding partition in P. Then we suppose an independent Gaussian process $\varepsilon(k(\ell), x) \sim GP(\mu_{k(\ell)}, \Sigma_{k(\ell)})$ for each partition. (Note that in principle we could take this approach further to arbitrary sets of $[M]_0$. However, this comes at the expense of a larger number of hyperparameters that need to be estimated.) Again our approach is to incorporate all Gaussian processes into a single one with prior distribution $f \sim GP(\mu, \Sigma)$:[1] therefore, for all $\ell \in [M]_0$ and $x \in \mathcal{D}$ we define $f(\ell, x) = f(0, x) + \varepsilon(k(\ell), x) + \delta_\ell(x)$, where $f(0, x) = g(x)$ is the objective function that we want to optimize. Due to linearity of expectation, we have

$$\begin{aligned}\mu(\ell, x) &= \mathbb{E}\left[f(0, x) + \varepsilon(k(\ell), x) + \delta_\ell(x)\right] \\ &= \mathbb{E}\left[f(0, x)\right] + \mathbb{E}\left[\varepsilon(k(\ell), x)\right] + \mathbb{E}\left[\delta_\ell(x)\right] \\ &= \mu_0(x),\end{aligned}$$

since $\mathbb{E}\left[\varepsilon(k(\ell), x)\right] = \mathbb{E}\left[\delta_\ell(x)\right] = 0$. Recall that the indicator variable $\mathbb{1}_{\ell,m}$ denotes Kronecker's delta. Let $\ell, m \in [M]_0$ and $x, x' \in \mathcal{D}$, then we define the following composite covariance function Σ:

$$\begin{aligned}&\Sigma\left((\ell, x), (m, x')\right) \\ &= \operatorname{Cov}\left(f(0, x) + \varepsilon(k(\ell), x) + \delta_\ell(x), f(0, x')\right. \\ &\quad \left. + \varepsilon(k(m), x') + \delta_m(x')\right) \\ &= \operatorname{Cov}(f(0, x), f(0, x')) + \operatorname{Cov}(\varepsilon(k(\ell), x), \varepsilon(k(m), x')) \\ &\quad + \operatorname{Cov}(\delta_\ell(x), \delta_m(x')) \\ &= \Sigma_0(x, x') + \mathbb{1}_{k(\ell),k(m)} \cdot \Sigma_{k(\ell)}(x, x') + \mathbb{1}_{\ell,m} \cdot \Sigma_\ell(x, x').\end{aligned}$$

A.2 Estimation of Hyperparameters

In this section we detail how to set the hyperparameters via *maximum a posteriori* (MAP) estimation and propose a specific prior that has proven its value in our application and thus is of interest in its own right.

In typical MISO scenarios little data is available, that is why we suggest MAP estimates that in our experience are more robust than maximum likelihood estimates (MLE) under these circumstances. However, we wish to point out that we observed essentially the same performances of the algorithms when conducting the Rosenbrock and Assemble-to-Order benchmarks with maximum likelihood estimates for the hyperparameters.

[1] For simplicity we reuse the notation from the first model to denote their pendants in this model.

In what follows we use the notation introduced in Sect. 3.1. One would suppose that the functions $\mu_0(\cdot)$ and $\Sigma_\ell(\cdot,\cdot)$ with $\ell \in [M]_0$ belong to some parameterized class: for example, one might set $\mu_0(\cdot)$ and each $\lambda_\ell(\cdot)$ to constants, and suppose that Σ_ℓ each belong to the class of Matérn covariance kernels (cp. Sect. 4 for the choices used in the experimental evaluation). The hyperparameters are fit from data using *maximum a posteriori* (MAP) estimation; note that this approach ensures that covariances between information sources and the objective function are inferred from data.

For a Matérn kernel we have to estimate $d+1$ hyperparameters for each information source (see next subsection): d length scales and the signal variance. We suppose a normal prior $\mathcal{N}\left(\mu_{\ell,i}, \sigma^2_{\ell,i}\right)$ for hyperparameter $\theta_{\ell,i}$ with $1 \le i \le d+1$ and $\ell \in [M]_0$. Let $D \in \mathcal{D}$ be a set of points, for example chosen via a Latin Hypercube design, and evaluate every information source at all points in D. We estimate the hyperparameters for $f(0,\cdot)$ and the δ_ℓ for $\ell \in [M]$, using the "observations" $\Delta_\ell = \{y(\ell,x) - y(0,x) \mid x \in D\}$ for the δ_ℓ. $y(\ell,x)$ is the observation including noise for design x at IS ℓ. The prior mean of the signal variance parameter of IS 0 is set to the variance of the observations at IS 0 minus their average observational noise. The mean for the signal variance of IS $_s$ with $\ell \in [M]$ is obtained analogously using the "observations" in Δ_s; here we subtract the mean noise variance of the observations at IS ℓ and the mean noise at IS 0, exploiting the assumption that observational noise is independent. Regarding the means of the priors for length scales, we found it useful to set each prior mean to the length of the interval that the corresponding parameter is optimized over. For each hyperparameters $\theta_{\ell,i}$, we set the variance of the prior to $\sigma^2_{\ell,i} = (\frac{\mu_{\ell,i}}{2})^2$, where $\mu_{\ell,i}$ is the mean of the prior.

A.3 How to Express Beliefs on Fidelities of Information Sources

In many applications one has beliefs about the relative accuracies of information sources. One approach to explicitly encode these is to introduce a new coefficient α_ℓ for each Σ_ℓ that typically would be fitted from data along with the other hyperparameters. But we may also set it at the discretion of a domain expert, which is particularly useful if none of the information sources is an unbiased estimator and we rely on regression to estimate the true objective. In case of the squared exponential kernel this coefficient is sometimes part of the formulation and referred to as "signal variance" (e.g., see [24, p. 19]). For the sake of completeness, we detail the effect for our model of uncorrelated information sources stated in Sect. 3.1. Recall that we suppose $f \sim GP(\mu, \Sigma)$ with a mean function μ and covariance kernel Σ, and observe that the introduction of the new coefficient α_ℓ does not affect $\mu(\ell,x)$. But it changes $\Sigma((\ell,x),(m,x'))$ to

$$\Sigma((\ell,x),(m,x')) = \Sigma_0(x,x') + \mathbb{1}_{\ell,m} \cdot \alpha_\ell \cdot \Sigma_\ell(x,x').$$

We observe that setting α_ℓ to a larger value results in a bigger uncertainty. The gist is that then samples from such an information source have less influence in the Gaussian process regression (e.g., see Eq. (A.6) on pp. 200 in [24]). It is instructive to consider the case that we observe a design x at a noiseless and deterministic information source: then its observed output coincides with $f(\ell,x)$ (with zero variance). Our estimate $f(0,x)$ for $g(x)$, however, is a Gaussian random variable whose variance depends (in particular) on the uncertainty of the above information source as encoded in α_ℓ, since $\lambda_\ell(x) = 0$ holds.

B Provable Performance Guarantees

First note that the `csKG` chooses an IS and an x such that the conditional expectation of the objective value of the recommendation is maximized per unit cost of the query. Let N denote the number of queries that the policy is allowed to make.

Proposition 1. *The cost-sensitive KG achieves an optimal information gain per unit cost (with respect to $\mathcal{A}$) for $N = 1$.*

The proposition follows immediately by definition of the cost-sensitive Knowledge Gradient factor $CKG(\cdot,\cdot)$ in Eq. (1) on p. 4.

Next we establish an *additive* bound on the loss of the solution obtained by the algorithm with respect to the unknown optimum, as the number of queries $N \to \infty$. In what follows suppose that $\mu(\ell,x)$=0 $\forall \ell, x$ and Σ is either the squared exponential kernel or a four times differentiable

Matérn kernel. Suppose that f is drawn from the prior, i.e. let $f \sim GP(\mu, \Sigma)$. Then the sample f from the distribution over function is itself twice continuously differentiable with probability one (e.g., see Ghosal and Roy [9, Theorem 5]). Moreover, let $x^{\mathrm{OPT}} \in \operatorname{argmax}_{x' \in \mathcal{D}} f(0, x')$ and $d = \max_{x' \in \mathcal{D}} \min_{x'' \in \mathcal{A}} \operatorname{dist}(x', x'')$, i.e. d is the maximum distance of any point in the continuous domain $\mathcal{D}$ to its closest point in the discrete set $\mathcal{A}$. Then we have the following bound on the objective value obtained by the cost-sensitive Knowledge Gradient.

Theorem 2 (Theorem 1 restated)**.** *For each $p \in [0, 1)$ there is constant K_p, such that the cost-sensitive Knowledge Gradient recommends an $x_N^* \in \mathcal{A}$, such that with probability p*

$$\lim_{N \to \infty} f(0, x_N^*) \geq f(0, x^{\mathrm{OPT}}) - K_p \cdot d.$$

First observe that if f is twice continuously differentiable over $\mathcal{D}$, then in particular the extrema of $\frac{\partial}{\partial x_i} f$ are bounded. Specifically, since the partial derivatives of $GP(\mu, \Sigma)$ are also GPs for our choice [24, Sect. 9.4], we can compute for every $p \in [0, 1)$ a constant K_p such that f is K_p-Lipschitz continuous on $\mathcal{D}$ with probability at least p. Then there is an $\bar{x} \in \mathcal{A}$ with

$$\operatorname{dist}(\bar{x}, x^{\mathrm{OPT}}) \leq d$$

and hence

$$f(0, x^{\mathrm{OPT}}) - f(0, \bar{x}) \leq K_p \cdot d. \tag{2}$$

We need one more caveat.

Corollary 1 (of Theorem 4 in [8])**.** *Let x_N^* be the recommendation of the cost-sensitive Knowledge Gradient after N queries to information sources. As $N \to \infty$, $\lim f(0, x_N^*) \geq f(0, \bar{x})$ a.s.*

This completes the proof of the theorem, since we already showed that Eq. (2) holds with probability p.

We provide an outline of the proof of Corollary 1; the details are deferred to the full version of the paper. Observe that the marginal distribution of the values at the points in $\{(\ell, x) \mid \ell \in [M]_0, x \in \mathcal{A}\}$ is multivariate normal as a consequence of our initial assumption $f \sim GP(\mu, \Sigma)$; recall that any set of points in $\mathcal{D}$ has a joint multivariate normal distribution under a Gaussian process prior. (This marginal distribution is known since μ and Σ are.) Frazier et al. [8] show in their Theorem 4 that, applied to a discrete set of points with joint multivariate normal distribution, the Knowledge Gradient recommends a point of maximum mean, as the number of samples N goes to infinity. The reason is that each point is sampled arbitrarily often or is learned perfectly. Now we observe that the only difference between the KG factor at (ℓ, x) used in their theorem and our utility $CKG(\ell, x)$ is that $CKG(\ell, x)$ is obtained by dividing the KG factor by the query cost $c_\ell(x)$ (see Eq. (1) on p. 4). This cost, however, is constant and hence in particular stays fixed as $N \to \infty$. Thus, the cost-sensitive Knowledge Gradient also finds a point of maximum value under $f(0, \cdot)$ in the discrete set $\{(\ell, x) \mid \ell \in [M]_0, x \in \mathcal{A}\}$ if $N \to \infty$.

C The MISO Benchmark Algorithms

The first benchmark method, `MTBO+`, is an improved version of Multi-Task Bayesian Optimization (`MTBO`) proposed by Swersky et al. [30]. It uses a cost-sensitive version of Entropy Search to select the next sample and information source: supposing a distribution over the location of the optima of the objective function, it maximizes in each iteration the reduction in differential entropy per query cost.

The algorithm requires a discretization $\mathcal{A}$ of $\mathcal{D}$. Let $P(x)$ be the probability that the optimum of g is at $x \in \mathcal{A}$ conditioned on our previous observations, the hyperparameters, and $\mathcal{A}$, and let $H[P(x)]$ be the differential entropy of the corresponding distribution. Moreover, denote by $H[P_\ell^y(x)]$ the expected entropy of the distribution if we had sampled x at IS ℓ and observed y. Then the cost-sensitive formulation of Entropy Search proposed in [30] is given by $\int \int (H[P(x)] - H[P_\ell^y(x)]) \, p(y \mid \vec{f}) p(\vec{f} \mid x, \ell) / c_\ell(x) \, \mathrm{d}y \mathrm{d}f$, where $p(\vec{f} \mid x, \ell)$ is the probability that the points in $\mathcal{A}$ take the values $\vec{f}$ conditioned on the hyperparameters and past observations, and $p(y \mid \vec{f})$ is the probability of observing y when querying x at IS ℓ.

`MTBO` combines this acquisition function with a "multi-task" Gaussian process model. Their kernel is given by the tensor product $K_t \otimes K_x$, where K_t (resp., K_x) denotes the covariance matrix of the tasks (resp., of the points), and hence is capable of exploiting correlations. Following a recommendation of Snoek 2016, our implementation `MTBO+` uses an improved formulation of the acquisition function given by Hernández-Lobato et al. [13], Snoek and et al. [27], but otherwise is identical to `MTBO`; in particular, it uses the statistical model of Swersky et al. [30].

The other algorithm, `misoEI` of [19], was developed to solve MISO problems that involve model discrepancy and therefore is a good competing method to compare with.

It maintains a separate Gaussian process for each information source: to combine this knowledge, the corresponding posterior distributions are fused for each design via Winkler's method (1981) into a single intermediate surrogate, which is a normally distributed random variable. Then Lam et al. adapt the Expected Improvement (`EI`) acquisition function to select the design which is to be sampled next: for the sake of simplicity, assume that observations are noiseless and that y^* is the objective value of a best sampled design. If Y_x denotes a Gaussian random variable with the posterior distribution of the objective value for design x, then $\mathbb{E}[\max\{Y_x - y^*, 0\}]$ is the expected improvement for x, and the `EI` acquisition function selects an x that maximizes this expectation. Based on this decision, the information source to invoke is chosen by a heuristic that aims at maximizing the `EI` per unit cost.

D Description of the Assemble-to-Order Benchmark

In the assemble-to-order (ATO) benchmark, a reinforcement learning problem from a business application, we are managing the inventory of a company that manufactures m products. Each product is made from a selection from n items, where we distinguish between key items and non-key items: a product can only be sold if all its key items are in stock. Non-key items are optional and increase the value. There is a target level for each item: the system automatically sends a replenishment order if the level drops below the target. The requested item is delivered after a random period. Since items in the inventory inflict holding cost, the goal is to find a target level vector that maximizes the expected profit per day (cp. Hong and Nelson [14] for details). The setting of Hong and Nelson supposes $m{=}5$ different products assembled from a subset of $n{=}8$ items, asking to optimize a 8-dimensional target vector $b \in [0, 20]^8$. We set up three information sources: IS 0 and IS 2 use the simulator of Xie et al. [37], whereas the cheapest source IS 1 invokes the implementation of Hong and Nelson. We assume that IS 0 models the truth. The IS differ in the number of replications: more replications increase the precision of the estimate but also the computational cost. IS 0 that models the truth has 500 replications, a noise variance of 0.056 and a cost of 17.1. IS 1 is the cheapest IS with 10 replications, a noise of 2.944, and cost 0.5. IS 2 has 100 replications, noise 0.332, and cost 3.9. The observational noise and query cost were estimated from data, supposing for the sake of simplicity that both functions are constant over the domain.

The two simulators differ subtly in the model of the inventory system. However, the effect in estimated objective value is significant: on average the outputs of both simulators at the same target vector differ by about 5% of the score of the global optimum, which is about 120, whereas the largest observed bias out of 1000 random samples was 31.8. Thus, we are witnessing a significant model discrepancy.