The reparameterization trick for acquisition functions

James T. Wilson[1,2]
j.wilson17@imperial.ac.uk

Riccardo Moriconi[1]
r.moriconi16@imperial.ac.uk

Frank Hutter[2]
fh@cs.uni-freiburg.de

Marc Peter Deisenroth[1]
m.deisenroth@imperial.ac.uk

[1]Imperial College of London

[2]University of Freiburg

Abstract

Bayesian optimization is a sample-efficient approach to solving global optimization problems. Along with a surrogate model, this approach relies on theoretically motivated value heuristics (acquisition functions) to guide the search process. Maximizing acquisition functions yields the best performance; unfortunately, this ideal is difficult to achieve since optimizing acquisition functions *per se* is frequently non-trivial. This statement is especially true in the parallel setting, where acquisition functions are routinely non-convex, high-dimensional, and intractable. Here, we demonstrate how many popular acquisition functions can be formulated as Gaussian integrals amenable to the reparameterization trick [14, 17] and, ensuingly, gradient-based optimization. Further, we use this reparameterized representation to derive an efficient Monte Carlo estimator for the upper confidence bound acquisition function [19] in the context of parallel selection.

1 Introduction

In Bayesian optimization (BO), acquisition functions H, with few exceptions, amount to integrals defined in terms of a belief p over the unknown outcomes $\mathbf{y} = \{y_1, \dots, y_q\}$ revealed when evaluating a black-box function f at corresponding input locations $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_q\}$. This formulation naturally occurs as part of a Bayesian approach whereby we would like to assess how valuable different queries $\mathbf{X}$ are to the optimization process by accounting for all conceivable realizations of $\mathbf{y} = f(\mathbf{X})$. Denoting by h the function used to convey the value-added for observing a given realization, this paradigm gives rise to acquisition functions defined as

$$H(\mathbf{X}; \boldsymbol{\phi}) = \int_{\mathcal{A}} h(\mathbf{y}; \boldsymbol{\phi}) p(\mathbf{y}|\mathbf{X}, \mathcal{D}) d\mathbf{y}\,, \tag{1}$$

where integration region $\mathcal{A} \subseteq \mathcal{Y}$ represents the set of all possible outcomes $\mathbf{y}$, $\boldsymbol{\phi}$ any additional parameters associated with integrand h, and $\mathcal{D}$ the available prior information.[1] Without loss of generality, we express acquisition functions as q-dimensional integrals, where q denotes the total number of queries with unknown outcomes after each decision. For pool-size $q = 1$, we recover strictly sequential decision-making rules; whereas, for $q > 1$, we obtain strategies for parallel selection.[2] As an exception to this rule, *non-myopic* acquisition functions, which assign value by further considering how different realizations of $(\mathbf{X}, \mathbf{y})$ impact our broader understanding of black-box f, generally correspond to higher-dimensional integrals. Specifically, non-myopic instances of the above formulation typically recurse, with the integrand h amounting to an additional integral of the form (1). While in a minority of cases closed-form solutions exist, these integrals are generally intractable and therefore difficult to optimize.

[1]Henceforth, we omit explicit reference to prior information $\mathcal{D}$.

[2]To avoid confusion when discussing SGD, we reserve the term *batch-size* for description of minibatches.

Examples of reparameterizable acquisition functions			
Acquisition function	Parameters	Integrand h	Reparameterization
Expected Improvement (EI)	$\boldsymbol{\mu}, \boldsymbol{\Sigma}; \alpha$	$\max(0, \max(\mathbf{y}) - \alpha)$	$\max(0, \max(\boldsymbol{\mu} + \mathbf{L}\mathbf{z}) - \alpha)$
Probability of Improvement (PI)	$\boldsymbol{\mu}, \boldsymbol{\Sigma}; \alpha, \tau$	$\mathbb{1}^{-}(\max(\mathbf{y}) - \alpha)$	$\sigma\left(\frac{\max(\boldsymbol{\mu}+\mathbf{L}\mathbf{z})-\alpha}{\tau}\right)$
Upper Confidence Bound (UCB)	$\boldsymbol{\mu}, \boldsymbol{\Sigma}; \beta$	$\max(\boldsymbol{\mu} + \lvert\tilde{\mathbf{y}} - \boldsymbol{\mu}\rvert)$	$\max\left(\boldsymbol{\mu} + \sqrt{\beta\pi/2}\lvert\mathbf{L}\mathbf{z}\rvert\right)$
Simple Regret (SR)	$\boldsymbol{\mu}, \boldsymbol{\Sigma}$	$\max(\mathbf{y})$	$\max(\boldsymbol{\mu} + \mathbf{L}\mathbf{z})$
Entropy Search (ES)	$\boldsymbol{\mu}, \boldsymbol{\Sigma}; \tau$	$\mathbb{1}^{+}(\mathbf{y} - \max(\mathbf{y}))$	$\mathrm{softmax}\left(\frac{\boldsymbol{\mu}+\mathbf{L}\mathbf{z}}{\tau}\right)$

Table 1: Above, we use the following notation: Cholesky factor $\mathbf{L}\mathbf{L}^\top \triangleq \boldsymbol{\Sigma}$; $\mathbb{1}^{+/-}$ denotes the right-/left-continuous Heaviside step function; σ the sigmoid nonlinearity; α the improvement threshold; τ the temperature parameter described in Section 2; and, random variables $\tilde{\mathbf{y}} \sim \mathcal{N}(\boldsymbol{\mu}, {}^{\beta\pi}/_{2}\boldsymbol{\Sigma})$. For Entropy Search, a non-myopic acquisition function, only the innermost integrand (used to approximate p_{max}) and its corresponding reparameterization are shown.

For this reason, a variety of methods have been proposed for evaluating intractable acquisition functions. These approaches have ranged from expectation propagation-based approximations of Gaussian probabilities [5, 9, 10] to bespoke approximation strategies [4, 6] to sample-based Monte Carlo techniques [9, 16, 18].

The special case of parallel Expected Improvement (q-EI) has received considerable attention [3, 7, 18, 20]; however, excepting [20], proposed methods do not scale gracefully in pool-size q. Still within the context of q-EI and independent of our work, [20] derive results analogous to our own, but refer to the reparameterization trick (discussed below) as *infinitesimal perturbation analysis* [8].

In this work, we focus on the most common estimation technique: Monte Carlo integration. Despite their generality and myriad other desirable properties, Monte Carlo approaches have consistently been regarded as non-differentiable and, therefore, inefficient in practice given the need to optimize (1). However, it seems to have been overlooked that sample-based approaches can indeed be used to estimate gradients, well-known examples of which include stochastic backpropagation and the reparameterization trick [14, 17]. In the following, we exploit this insight to demonstrate gradient-based optimization of acquisition functions estimated via Monte Carlo integration.

The *reparameterization trick* is a way of rewriting functions of random variables that makes their differentiability w.r.t. the parameters of an underlying distribution transparent. The trick applies a deterministic mapping $\rho : \mathcal{Z} \to \mathcal{Y}$ from random variables $\mathbf{z} \in \mathcal{Z}$ with a parameter-free base distribution to random variables $\mathbf{y} \in \mathcal{Y}$ with the target distribution. This change of variables helps clarify that if h is a differentiable function of $\mathbf{y} = \rho(\mathbf{z}; \boldsymbol{\theta})$ then, by the chain rule of derivatives $\frac{dh}{d\boldsymbol{\theta}} = \frac{dh}{d\rho}\frac{d\rho}{d\boldsymbol{\theta}}$, i.e., we can use gradient information to optimize the target distribution's parameters $\boldsymbol{\theta}$. We now explore the importance of this fact for BO and, in particular, for parallel selection.

2 Reparameterizing acquisition functions

As is arguably the natural way of expressing uncertainty over interrelated values, beliefs $p(\mathbf{y}|\mathbf{X})$ over the q outcomes for pool $\mathbf{X}$ are typically defined in terms of a multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. In the context of the reparameterization trick, the corresponding deterministic mapping for Gaussian random variables $\mathbf{y}$ is $\rho(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \triangleq \boldsymbol{\mu} + \mathbf{L}\mathbf{z}$, where $\mathbf{L}$ denotes the Cholesky factor of $\boldsymbol{\Sigma}$, s.t. $\mathbf{L}\mathbf{L}^\top = \boldsymbol{\Sigma}$ and $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Rewriting (1) as a Gaussian integral and reparameterizing, we have

$$H(\mathbf{X}; \boldsymbol{\phi}) = \int_{\mathbf{a}}^{\mathbf{b}} h(\mathbf{y}; \boldsymbol{\phi})\mathcal{N}(\mathbf{y}; \boldsymbol{\mu}, \boldsymbol{\Sigma})d\mathbf{y} = \int_{\mathbf{a}'}^{\mathbf{b}'} h(\boldsymbol{\mu} + \mathbf{L}\mathbf{z}; \boldsymbol{\phi}')\mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I})d\mathbf{z}\,, \tag{2}$$

where each of the q terms c_i' in both $\mathbf{a}'$ and $\mathbf{b}'$ is transformed as $c_i' = (c_i - \mu_i - \sum_{j<i} L_{ij}z_j)/L_{ii}$ and where values in $\boldsymbol{\phi}'$ have similarly been mapped to $\mathcal{Z}$. By taking the gradient of $H(\mathbf{X}; \boldsymbol{\phi})$ w.r.t. model-based posterior $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{M}(\mathbf{X})$ and further differentiating through the model to inputs $\mathbf{X}$, we can perform gradient ascent on acquisition values.[3]

[3]Parameters associated with model $\mathcal{M}$ are not differentiated through and are therefore omitted for clarity.

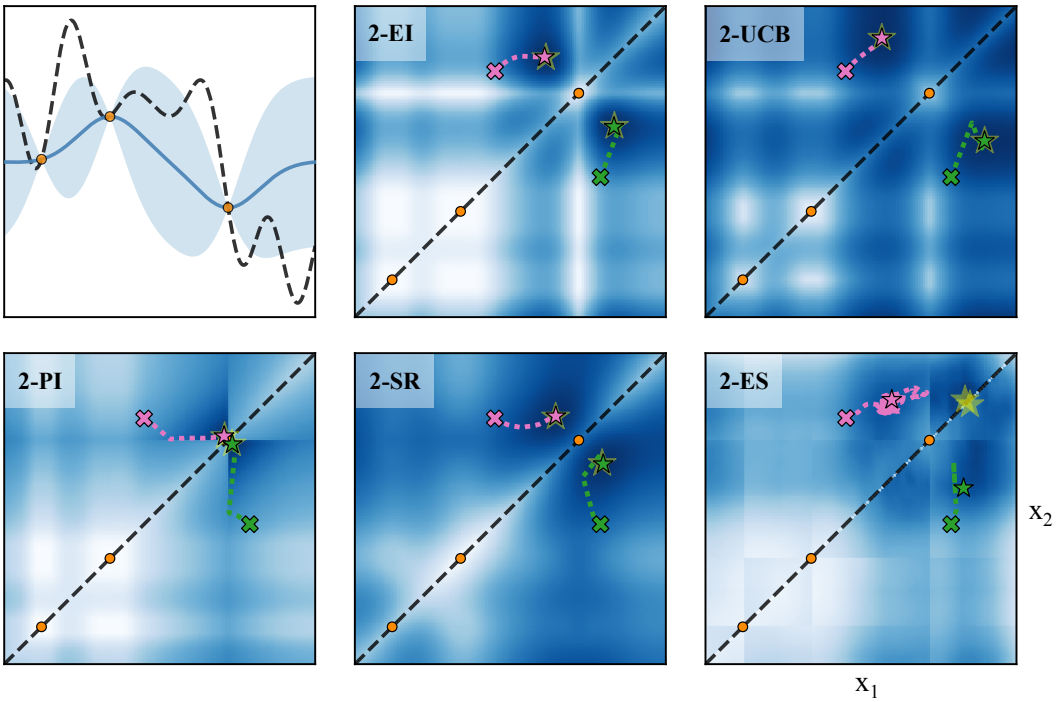

Figure 1: Top left: GP-based posterior over 1-dimensional black-box f given three initial observations (orange dots). Remaining: Response surfaces of various acquisition functions for pool-size $q = 2$. From '×' to '☆', paths explored by gradient descent (green) and stochastic gradient descent (pink) when optimizing the various acquisition functions. Dashed horizontal lines denote axes of symmetry and large '☆' (yellow) indicate the global maximum of each acquisition function.

When Monte Carlo integrating (2), an unbiased estimate to the acquisition gradients is then

$$\frac{d\,H(\mathbf{X};\boldsymbol{\phi})}{d\mathbf{X}} \approx \frac{1}{n}\sum_{k=1}^{n} \frac{d\,h(\mathbf{y}_k;\boldsymbol{\phi})}{d\mathbf{y}_k}\frac{d\mathbf{y}_k}{d\,\mathcal{M}(\mathbf{X})}\frac{d\,\mathcal{M}(\mathbf{X})}{d\mathbf{X}}, \tag{3}$$

where, by minor abuse of notation, we have substituted in $\mathbf{y}_k = \rho(\mathbf{z}_k; \mathcal{M}(\mathbf{X}))$. The availability of gradient information is especially important for $q > 1$, both because parallel acquisition functions are generally intractable and because the dimensionality of the acquisition space scales linearly in q.

Examples of well-known acquisition functions amenable to this treatment are presented in Table 1. Figure 1 provides a visual example of the corresponding (stochastic) gradient ascent process, for each of the five acquisition functions shown in the table. Before going further, several points of interest in Table 1 warrant attention:

1. **Parallelizing UCB**: To the best of our knowledge, the integral representation of UCB is novel and leads to the first truly parallel formulation of UCB (q-UCB). Relevantly, using the reparameterization trick greatly simplifies the associated derivation. As with other acquisition functions discussed here, q-UCB can be efficiently estimated via Monte Carlo and optimized using gradients. For the complete derivation and related formulae, please refer to Appendix A.

2. **Relaxing Heaviside step functions**: Both Probability of Improvement (PI) and Entropy Search (ES) contain Heaviside step functions, whose derivatives are Dirac delta functions. Since these gradients are zero a.e., we instead propose the use of a softmax function with temperature parameter τ. This combination has the appealing property that the resulting approximation becomes exact as $\tau \to 0$, a property recently exploited in [11, 15]. To the extent that this soft approximation introduces an additional source of error, we argue that this downside is largely outweighed by the availability of informative gradients, which enable us to greatly reduce optimization error [2].

3. **Differentiating though the max()**: Many acquisition functions, such as EI, use the max operator. While not technically differentiable, this operator is known to be subdifferentiable and affords well-behaved (sub)gradients.

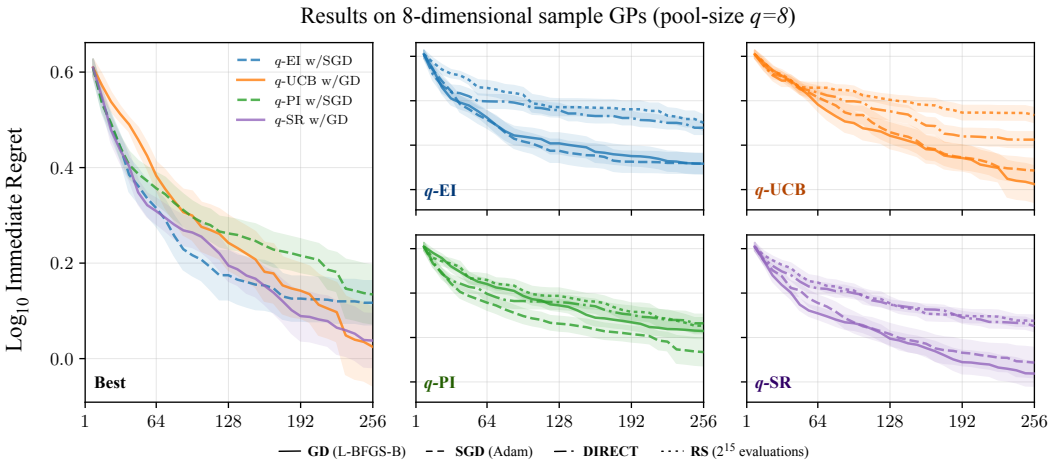

Figure 2: Left: For equivalent runtimes, best average case performance of each acquisition function given 256 evaluations of 8-dimensional samples from a GP prior with known hyperparameters when choosing pool-size $q = 8$ queries in parallel. Remaining: Performance of individual acquisition functions for different optimizers thereof.

3 Experiments

As baselines, we compared gradient-based approaches to optimizing acquisition functions with Random Search [1] and Dividing Rectangles [12] based ones. For stochastic gradient descent (SGD), we experimented with several off-the-shelf optimizers; of these, Adam [13] produced the best results and is reported here. Similarly, we tested various batch-sizes m_b and report results for $m_b = 64$. For gradient descent (GD), we used a standard implementation of L-BFGS-B [21]. In both cases, gradient-based optimizers were run using 32 starting points sampled from the acquisition function. Finally, for both q-PI and q-ES, we set the temperature $\tau = 0.01$; and, for q-UCB, we set the confidence parameter $\beta = \sqrt{3}$.

Prior to running our experiments, we configured each acquisition function optimizer such that its runtime approximately matched that of the others. Further details regarding our experiments, including individual runtimes, are provided in Appendix B.

To help reduce the number of potentially confounding variables, we experimented on 8-dimensional tasks drawn from a Gaussian process prior with known hyperparameters. For each combination of acquisition function and optimizer, trials began with q randomly chosen observations and iterated by choosing q queries at a time.[4] Each pair was run on a total of 16 sampled tasks, with results shown in Figure 2. Across acquisition functions, gradient-based strategies markedly outperformed gradient-free alternatives. Further, stochastic and deterministic gradient methods delivered comparable performance.

4 Conclusion

We show how many popular acquisition functions can be written as Gaussian integrals amenable to the reparameterization trick. By reparameterizing these integrals, we clarify the differentiability of their Monte Carlo estimates and, in turn, provide a generalized method for using gradients to optimize acquisition values. Our results clearly demonstrate the superiority of gradient-based approaches for optimizing acquisition functions, even in modest dimensional cases. Further, we show how, by looking at the associated integrals through the lens of the reparameterization trick, the difficult process of deriving theoretically sound acquisition functions may be greatly simplified.

[4] Methods discussed here extend to the parallel asynchronous setting; but, we did not explore this option.

Acknowledgments

The support of the EPSRC Centre for Doctoral Training in High Performance Embedded and Distributed Systems (reference EP/L016796/1) is gratefully acknowledged. This work has partly been supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant no. 716721.

References

[1] J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 2012.

[2] O. Bousquet and L. Bottou. The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems 22*, 2008.

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

[4] E. Contal, D. Buffoni, A. Robicquet, and N. Vayatis. Parallel Gaussian process optimization with upper confidence bound and pure exploration. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2013.

[5] J. P. Cunningham, P. Hennig, and S. Lacoste-Julien. Gaussian probabilities and expectation propagation. *arXiv preprint arXiv:1111.6832*, 2011.

[6] T. Desautels, A. Krause, and J. W. Burdick. Parallelizing exploration-exploitation tradeoffs in Gaussian process bandit optimization. *Journal of Machine Learning Research*, 2014.

[7] D. Ginsbourger, R. Le Riche, and L. Carraro. *Kriging is well-suited to parallelize optimization*, chapter 6. Springer, 2010.

[8] P. Glasserman. *Monte Carlo methods in financial engineering*. Springer, 2013.

[9] P. Hennig and C. Schuler. Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, 2012.

[10] J. Hernández-Lobato, M. Hoffman, and Z. Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *Advances in Neural Information Processing Systems 27*, 2014.

[11] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with Gumbel-Softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[12] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, 1993.

[13] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations*, 2014.

[15] C. J. Maddison, A. Mnih, and Y. W. Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

[16] M. A. Osborne, R. Garnett, and S. J. Roberts. Gaussian processes for global optimization. In *Proceedings of the 3rd International Conference on Learning and Intelligent Optimization*, 2009.

[17] D. J. Rezende, M. Shakir, and D. Wierstra. Stochastic backpropagation and variational inference in deep latent Gaussian models. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.

[18] J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems 25*, 2012.

[19] N. Srinivas, A. Krause, S. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the 27th International Conference on Machine Learning*, 2010.

[20] J. Wang, S. C. Clark, E. Liu, and P. I. Frazier. Parallel Bayesian global optimization of expensive functions. *arXiv preprint arXiv:1602.05149*, 2016.

[21] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software*, 1997.

A Parallel Upper Confidence Bound (q-UCB)

Working backward through (2), we derive an exact expression for parallel UCB. In doing so, we begin with the definition

$$\int_0^\infty \sqrt{2\pi} y \mathcal{N}(y; 0, \sigma^2) dy = \frac{1}{2} \int_{-\infty}^\infty |\sqrt{2\pi}\sigma z| \mathcal{N}(z; 0, 1) dz = \sigma, \tag{4}$$

where $|\cdot|$ denotes the (element-wise) absolute value operator.[5] Using this fact and given $z \sim \mathcal{N}(0, 1)$, let $\tilde{\sigma}^2 \triangleq (\beta\pi/2)\sigma^2$ such that $\mathbb{E}[|\tilde{\sigma} z|] = \beta^{1/2}\sigma$. Under this notation, marginal UCB can be expressed as

$$\begin{aligned} \text{1-UCB}(\mathbf{x}; \beta) &= \mu + \beta^{1/2}\sigma & (5) \\ &= \int_{-\infty}^\infty \mu + |\tilde{\sigma} z| \mathcal{N}(z; 0, 1) dz & (6) \\ &= \int_{-\infty}^\infty \mu + |y - \mu| \mathcal{N}(y; \mu, \tilde{\sigma}^2) dy & (7) \end{aligned}$$

where (μ, σ^2) parameterize a Gaussian posterior over $y = f(\mathbf{x})$. This integral form of 1-UCB is advantageous precisely because it naturally lends itself to the generalized expression

$$\begin{aligned} q\text{-UCB}(\mathbf{X}; \beta) &= \int_{-\infty}^\infty \max(\boldsymbol{\mu} + |\mathbf{y} - \boldsymbol{\mu}|) \mathcal{N}(\mathbf{y}; \boldsymbol{\mu}, \tilde{\boldsymbol{\Sigma}}) d\mathbf{y} & (8) \\ &= \int_{-\infty}^\infty \max(\boldsymbol{\mu} + |\tilde{\mathbf{L}}\mathbf{z}|) \mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I}) d\mathbf{z} & (9) \\ &\approx \frac{1}{n} \sum_{k=1}^n \max(\boldsymbol{\mu} + |\tilde{\mathbf{L}}\mathbf{z}_k|) \quad \text{for} \quad \mathbf{z}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), & (10) \end{aligned}$$

where $\tilde{\mathbf{L}}\tilde{\mathbf{L}}^\top = \tilde{\boldsymbol{\Sigma}} \triangleq (\beta\pi/2)\boldsymbol{\Sigma}$. This representation has the requisite property that, for any size $q' \leq q$ subset of $\mathbf{X}$, the value obtained when marginalizing out the remaining $q - q'$ terms is its q'-UCB value.

Previous methods for parallelizing UCB have approached the problem by imitating a purely sequential strategy [4, 6]. Because a fully Bayesian approach to sequential selection generally involves an exponential number of posteriors, these works incorporate various well-chosen heuristics for the purpose of efficiently approximate parallel UCB.[6] By directly addressing the associated q-dimensional integral however, Equation (10) avoids the need for such approximations and, instead, unbiasedly estimates the true value.

Finally, the special case of marginal UCB (6) can be further simplified as

$$\text{1-UCB}(\mathbf{x}; \beta) = \mu + 2 \int_0^\infty \tilde{\sigma} z \mathcal{N}(z; 0, 1) dz = \int_\mu^\infty y \mathcal{N}(y; \mu, 2\pi\beta\sigma^2) dy, \tag{11}$$

revealing an intuitive form, namely, the expectation of a Gaussian random variable (with rescaled covariance) above its mean.

[5] This definition comes directly from the standard integral identity $\int_0^\infty x e^{-ax^2} dx = 1/2a$.

[6] Due to the stochastic nature of the mean updates, the number of posteriors grows exponentially in q.

B Experiment details

Runtimes of acquisition function optimizers				
Optimizer	q-EI	q-UCB	q-PI	q-SR
Random Search (RS)	23.9 ± 2.3	17.8 ± 1.6	20.1 ± 1.9	20.4 ± 1.9
Dividing Rectangles (DIRECT)	19.8 ± 1.5	21.5 ± 1.9	21.0 ± 1.7	20.2 ± 1.5
GD (L-BFGS-B)	19.9 ± 9.0	18.2 ± 1.4	17.6 ± 7.8	13.7 ± 1.2
SGD (Adam)	17.6 ± 9.2	13.6 ± 5.8	15.6 ± 6.0	15.4 ± 5.9

Table 2: Average runtime in seconds for each combination of acquisition function and optimizer when choosing the next pool of inputs. Reported numbers denote the mean and standard deviation of recorded wall-clock times.

To provide fair comparison between acquisition function optimizers, efforts were made to approximately match their respective runtimes. First, Random Search was run using a set of 2^{15} uniform random pools $\mathbf{X}$, at each step during BO. Subsequently, RS's average runtime, measured over a handful of preliminary trials, was used as a target value when configuring the remaining optimizers. Table 2 provides individual runtimes for each combination of acquisition function and optimizer.

For stochastic gradient descent, we tested the following optimizers: SGD with momentum, RMSProp, and Adam. Trials were run using batch-sizes $m_b \in \{32, 64, 128, 256\}$, each time tuning the number of SGD steps for equivalent runtimes. Of the tested configurations, 1024 steps using $m_b = 64$ delivered the best performance.