A Nonmyopic Approach to Cost-Constrained Bayesian Optimization (Supplementary material)

Eric Hans Lee[1] **David Eriksson**[2] **Valerio Perrone**[3] **Matthias Seeger**[3]

[1]SigOpt: An Intel Company; eric.lee@intel.com
[2]Facebook; deriksson@fb.com
[3]Amazon; [vperrone, matthis]@amazon.de

A GAUSSIAN PROCESS REGRESSION

We place a GP prior on $f(\mathbf{x})$, denoted by $f \sim \mathcal{GP}(\mu, K)$, where $\mu : \Omega \to \mathbb{R}$ and $k : \Omega \times \Omega \to \mathbb{R}$ are the mean function and covariance kernel, respectively. The kernel $k(\mathbf{x}, \mathbf{x}')$ correlates neighboring points, and may contain *hyperparameters*, such as lengthscales that are learned to improve the quality of approximation [Rasmussen and Williams, 2006]. For a given $\mathcal{D}_t = \{(\mathbf{x}_i, y_i)\}_{i=1}^t$, we define:

$$\mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_t \end{pmatrix}, \ \mathbf{k}(\mathbf{x}) = \begin{pmatrix} k(\mathbf{x}, \mathbf{x}_1) \\ \vdots \\ k(\mathbf{x}, \mathbf{x}_t) \end{pmatrix}, \ \mathsf{K} = \begin{pmatrix} \mathbf{k}(\mathbf{x}_1)^\top \\ \vdots \\ \mathbf{k}(\mathbf{x}_k)^\top \end{pmatrix}.$$

We assume y_i is observed with Gaussian white noise: $y_i = f(\mathbf{x}_i) + \epsilon_i$, where $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$. Given a GP prior and data $\mathcal{D}_t$, the resulting posterior distribution for function values at a location $\mathbf{x}$ is the Normal distribution $\mathcal{N}(\mu_t(\mathbf{x}; \mathcal{D}_t), \sigma_t^2(\mathbf{x}; \mathcal{D}_t))$:

$$\mu_t(\mathbf{x}; \mathcal{D}_t) = \mu(\mathbf{x}) + \mathbf{k}(\mathbf{x})^\top (\mathsf{K} + \sigma^2 \mathsf{I}_t)^{-1} (\mathbf{y} - \mu(\mathbf{x})),$$
$$\sigma_t^2(\mathbf{x}; \mathcal{D}_t) = k(\mathbf{x}, \mathbf{x}) - \mathbf{k}(\mathbf{x})^\top (\mathsf{K} + \sigma^2 \mathsf{I}_t)^{-1} \mathbf{k}(\mathbf{x}),$$

where I_t is the $t \times t$ identity matrix. We use the Matérn 5/2 kernel in this paper:

$$k_{5/2}(\mathbf{x}, \mathbf{x}') = \alpha^2 \left(1 + \frac{\sqrt{5}}{\ell} + \frac{5}{3\ell^2}\right) \exp\left(-\frac{\sqrt{5}\|\mathbf{x} - \mathbf{x}'\|}{\ell}\right).$$

B THEORETICAL RESULTS

If a base policy $\tilde{\pi}$ is *sequentially consistent*, rollout in the MDP setting will perform better in expectation than the base policy itself The same holds true in the CMDP setting if $c(\mathbf{x})$ is also deterministic.

We first define a heuristic $\mathcal{H}$ as a method to generate decision rules in epochs $\{0, \ldots, h-1\}$, such that the resulting heuristic policy π generated from $\mathcal{H}$ on state s is defined as

$$\pi_{\mathcal{H}(s)} = \{\pi_0^{\pi_{\mathcal{H}(s)}} \ldots \pi_{h-1}^{\pi_{\mathcal{H}(s)}}\}$$

Having defined $\pi_{\mathcal{H}(s)}$, we define sequential consistency for stochastic MDPs below.

Definition 1 *[Goodson et al., 2017]: A heuristic $\mathcal{H}$ is sequentially consistent if, for every trajectory from any s and all subsequent s':*

$$\pi_{\mathcal{H}(s)} = \pi_{\mathcal{H}(s')},$$

or in other words, that the decision rules generated from the heuristic are the same:

$$\{\pi_0^{\pi_{\mathcal{H}(s)}} \ldots \pi_{h-1}^{\pi_{\mathcal{H}(s)}}\} = \{\pi_0^{\pi_{\mathcal{H}(s')}} \ldots \pi_{h-1}^{\pi_{\mathcal{H}(s')}}\}.$$

This frames sequential consistency in terms of decision rules instead of as sample trajectories Bertsekas [2017], which we did in the main text for notational clarity. However, the intuition of the theorem above is the same as its deterministic counterpart; a heuristic $\mathcal{H}$ is sequentially consistent if it produces the same subsequent state s' when started at any intermediate state of a path that it generates s.

Theorem 1 *[Bertsekas, 2005]: In the CMDP setting, a rollout policy $\pi_{roll} = \pi^{\pi_{\mathcal{H}(s)}}$ using sequentially consistent heuristic $\mathcal{H}$ does no worse than its base policy $\tilde{\pi}$ in expectation.*

$$V_h^{\pi_{roll}}(s_0) \geq V_h^{\tilde{\pi}}(s_0).$$

Thus, the value function of a rollout policy is always greater than the value function of the base policy.

To guarantee sequential consistency of our acquisition function, we need only consistently break ties if the acquisition function has multiple maxima.

C ADDITIONAL EXPERIMENT

Most of our HPO experiments were run on the OpenML w2a dataset. To sanity check our performance's robustness,

	EI	EIpu	R2	R4
mean	0.140	0.139	0.137	**0.131**
std	0.005	0.005	0.004	**0.001**

Table 1: Rollout performance for horizons 2 and 4 outperformed both EI and EIpu. The best method is bolded.

we run the same HPO problem for k-nearest-neighbors with the OpenML a1a dataset. Rollout performance remained superior, and we record the mean and standard error in Table 1.

References

D Bertsekas. Rollout algorithms for constrained dynamic programming. *Lab. for Information and Decision Systems Report*, 2646, 2005.

Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*, volume I. Athena scientific Belmont, MA, 4th edition, 2017.

Justin C Goodson, Barrett W Thomas, and Jeffrey W Ohlmann. A rollout algorithm framework for heuristic solutions to finite-horizon stochastic dynamic programs. *European Journal of Operational Research*, 258(1):216–229, 2017.

Carl Rasmussen and Chris Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.