Supplementary material: Efficient Strategies in Rollout for Bayesian Optimization

1 KERNELS

The kernel functions we use in this paper are the squared exponential (SE) kernel, Matérn 5/2 kernel, and Matérn 3/2 kernel, respectively:

$$K_{\text{SE}}(r) = \alpha^2 \exp\left(-\frac{r^2}{2\ell^2}\right),$$

$$K_{5/2}(r) = \alpha^2 \left(1 + \frac{\sqrt{5}}{\ell} + \frac{5}{3\ell^2}\right) \exp\left(-\frac{\sqrt{5}r}{\ell}\right),$$

$$K_{3/2}(r) = \alpha^2 \left(1 + \frac{\sqrt{3}}{\ell}\right) \exp\left(-\frac{\sqrt{3}r}{\ell}\right),$$

where $r = \| \mathbf{x} - \mathbf{x}' \|_2$.

2 ACQUISITION FUNCTIONS

PI, EI, and UCB-κ have the closed forms:

$$\Lambda_{PI}(\mathbf{x}) = \Phi(\frac{y(\mathbf{x}) - y^*}{\sigma(\mathbf{x})}).$$

$$\Lambda_{EI}(\mathbf{x}) = (y(\mathbf{x}) - y^*)\Phi(\frac{y(\mathbf{x}) - y^*}{\sigma(\mathbf{x})}) + \sigma(\mathbf{x})\phi(\frac{y(\mathbf{x}) - y^*}{\sigma(\mathbf{x})}).$$

$$\Lambda_{UCB\kappa}(\mathbf{x}) = \mu(\mathbf{x}) + \kappa\sigma(\mathbf{x}).$$

KG does not have a closed form. It is defined as the expected value of the posterior minimum:

$$\Lambda_{KG}(\mathbf{x}) = \mathbb{E}_y[\mu^*(y|\,\mathbf{x})].$$

Where $\mu^*(y|\,\mathbf{x})$ is the value of the the posterior mean having sampled y at $\mathbf{x}$. The distribution of y is the posterior distribution of the GP.

3 CONTROL VARIATES

The general idea behind control variates is to find a covariate $g(y)$ with known mean and negative correlation with $f(y)$. The quantity $c(y) = f(y) + \beta g(y)$, known as a regression control variate (RCV), is estimated, and then debiased afterwards. If Var$[f(y)] = \sigma_f$ and Var$[g(y)] = \sigma_g$, then:

$$\text{Var}[c(y)] = \sigma_f^2 + \beta^2\sigma_g^2 - 2\beta\text{Cov}[f(y), g(y)].$$

The optimal value minimizing the variance of $c(y)$ is thus:

$$\beta = \text{Cov}[f(y), g(y)]/\sigma_h^2.$$

In practice, both Cov$[f(y), g(y)]$ and σ_h^2 must be either estimated from samples of f and g or computed a-priori.

In the case of $k > 1$ control variates, we consider a vector of control variates $\mathbf{g}(y) = [g_1(y), g_2, \ldots, g_k(y)]^T$. Our estimator will have the form $c(y) = f(y) - \beta^T\mathbf{g}(y)$, where β is an length k vector of constants. The optimal β minimizing the variance of $c(y)$ is:

$$\beta = \Sigma_{\mathbf{g}}^{-1} * \sigma_{\mathbf{g},f},$$

where $\Sigma_{\mathbf{g}}$ is the covariance matrix of $\mathbf{g}(y)$ and $\sigma_{\mathbf{g},f}$ is the vector of covariances between each variate and $f(y)$.

4 NAS BENCHMARK

The NAS benchmark is a tabular benchmark containing all possible hyperparameter configurations evaluated for a two-layer multi-layer perceptron on different datasets. The search space we consider is:

- Batch size in $\{8, 16, 32, 64\}$.
- Epochs in $\{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$.
- Layer 1 width in $\{16, 32, 64, 128, 256, 512\}$.

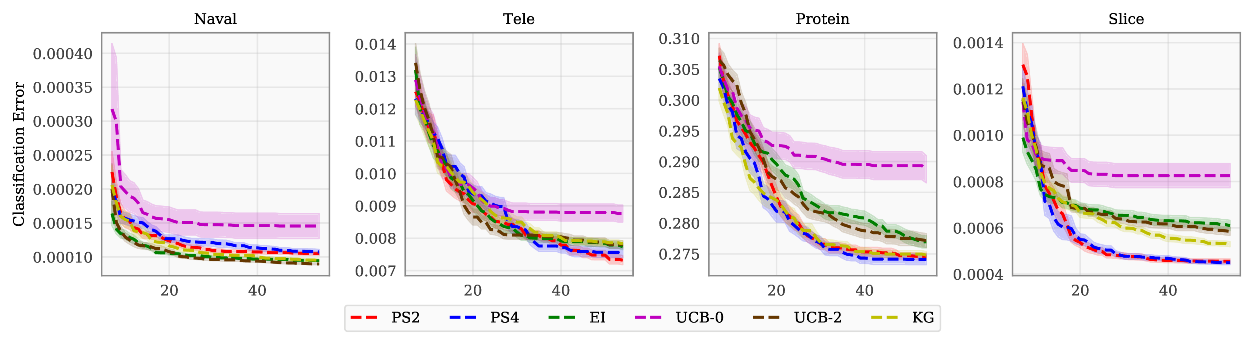

Figure 1: The classification error achieved by PS2 and PS4 is largely on par with, if not better than, the performance of EI, KG, and UCB variants. The only exception is the *Tele* dataset.

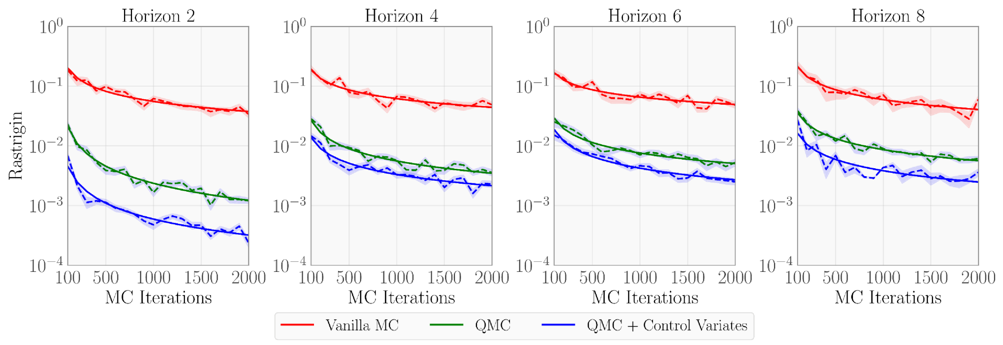

Figure 2: The estimation errors of MC (red), QMC (green), and QMC combined with control variates (blue).

- Layer 2 width in $\{16, 32, 64, 128, 256, 512\}$.

The resulting search space is four-dimensional. We optimize over the unit hypercube $[0, 1]^4$ and scale and round evaluation points to the corresponding NAS search space entry. Note that the NAS benchmark contains other hyperparameters as well, which we set to the default. These include the activation functions (default: tanh), the dropout (default: 0), the learning rate (default: 0.005), and the learning rate schedule (default: cosine).

The datasets in the NAS benchmark are all classification tasks taken from the UC Irvine repository for machine learning datasets. We run our method on all four in the NAS benchmark: *Naval*, *Tele*, *Protein*, and *Splice*. The achievable classification error for each dataset is different, so we compare methods by regret, which is defined as:

$$\frac{y_{init} - y_{best}}{y_{init}},$$

where y_{init} is the starting value during optimization and y_{best} is the best observed value during iteration so far. For each dataset, we run BO using EI, KG, UCB0, UCB2, and our policy search methods for horizons 2 and 4, labeled PS2 and PS4 respectively. We replicate BO runs 50 times.

In our main paper, we plotted the average regret among the four datasets, and PS2 and PS4 beat the competing methods. In this supplement, we plot the the individual classification errors for further clarity in Figure 1. We find the performance of both PS2 and PS4 performance are largely on par with, if not better than, the performance of EI, KG, and UCB variants.

5 ABLATION STUDY

Recall that we combine QMC and control variates to achieve high levels of variance reduction in the resulting Monte carlo estimator.

In Figure 2, we empirically measure the individual impact of QMC and control variates. We roll out EI for horizons 2, 4, 6, and 8, and calculate the variance of estimators for MC sample sizes in $[100, 200, 300, \ldots, 2000]$, using 50 trials each. We compare the Vanilla MC estimator, a QMC estimator, and a QMC estimator that also uses control variates. The underlying function is the Rastrigin function. As we mentioned before, the effectiveness of our control variates, which consist of myopic acquisition functions, are less effective as h increases. As a whole, QMC contributes to a greater drop in variance.