Near-linear Time Gaussian Process Optimization with Adaptive Batching and Resparsification

Daniele Calandriello*,†,¶, Luigi Carratino*,‡, Alessandro Lazaric§, Michal Valko¶, Lorenzo Rosasco†,‡,‖
*Equal contribution, †LCSL - Istituto Italiano di Tecnologia, ‡Università degli Studi di Genova,
§Facebook AI Research Paris, ¶DeepMind Paris, ‖Massachusetts Institute of Technology

In a nutshell

Gaussian process optimization (GP-Opt) is emerging as a valuable tool for stochastic black-box optimization (e.g. hyperparameter tuning), slowly replacing grad student descent. However provably convergent GP-Opt algorithms (e.g. GP-UCB) suffer from:
- **computational bottleneck**: $\mathcal{O}(T^3)$ time and $\mathcal{O}(T^2)$ space
- **experimental bottleneck**: need feedback at each iteration

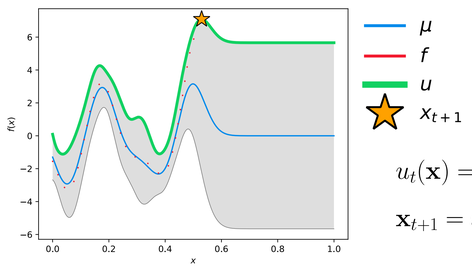
We introduce the first general GP optimization algorithm (BBKB) that is provably scalable, with near-linear runtime $\mathcal{O}(Td_{\text{eff}}^2)$, no regret, and also maintains valid posterior variance estimates at all steps.

Gaussian process optimization and GP-UCB

Candidates $\mathcal{A} = \{\mathbf{x}_i\}_{i=1}^A$ with $\mathbf{x}_i \in \mathbb{R}^D$ (or RKHS $\mathcal{H}$)

For $t \in [1, \dots, T]$: (1) select $\mathbf{x}_{t+1} = \arg\max_{\mathbf{x}_i} u_t(\mathbf{x}_i)$
(2) Receive noisy feedback $y_{t+1} = f(\mathbf{x}_{t+1}) + \eta_{t+1}$
(3) Improve u_{t+1} for next time

Goal: minimize regret $R_T = \sum_{t=1}^T f(\mathbf{x}_*) - f(\mathbf{x}_t)$ vs. $\mathbf{x}_* = \arg\max_{\mathbf{x}_i} f(\mathbf{x}_i)$



$$u_t(\mathbf{x}) = \mu_t(\mathbf{x}) + \beta_t \sigma_t(\mathbf{x})$$

$$\mathbf{x}_{t+1} = \arg\max_{\mathbf{x} \in \mathcal{A}} u_t(\mathbf{x})$$

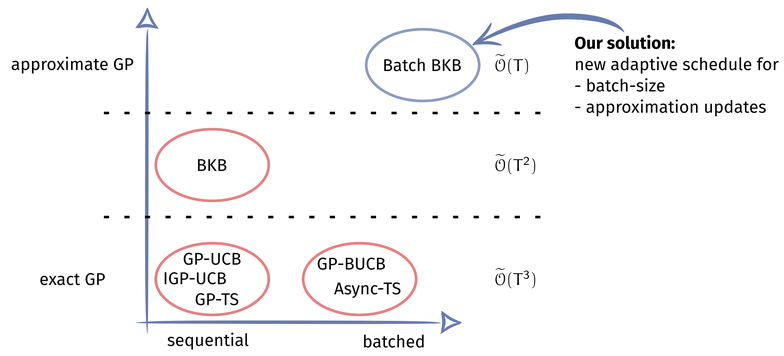
$$\mu_t(\mathbf{x}) = \mathbf{x}^\intercal(\mathbf{X}_t^\intercal \mathbf{X}_t + \lambda \mathbf{I})^{-1} \mathbf{X}_t^\intercal \mathbf{y}_t, \qquad \sigma_t^2(\mathbf{x}) = \mathbf{x}^\intercal(\mathbf{X}_t^\intercal \mathbf{X}_t + \lambda \mathbf{I})^{-1} \mathbf{x}$$

🤔 How to improve bottlenecks and still achieve low regret? 🤔

sequential vs batch

exact GP vs approximate GP

💡 Sparse GP with dictionary $\mathcal{D} = \{\mathbf{x}_j\}_{j=1}^m$ of m inducing points



Batch Budgeted Kernelized Bandits (BBKB)

BKB

for $t = \{1, \dots, T-1\}$ **do**
  for $i = \{1, \dots, A\}$ **do**
    $\widetilde{u}_t(\mathbf{x}_i) = \widetilde{\mu}_t(\mathbf{x}_i, \mathcal{D}_t) + \widetilde{\beta}_t \widetilde{\sigma}_t^2(\mathbf{x}_i, \mathcal{D}_t)$;
  end
  Select $\mathbf{x}_{t+1} \leftarrow \arg\max_{\mathbf{x}_i \in \mathcal{A}} \widetilde{u}_t(\mathbf{x}_i)$;
  Set $\widetilde{\mathbf{p}}_{t+1} \propto [\widetilde{\sigma}_t^2(\mathbf{x}_1, \mathcal{D}_t), \dots, \widetilde{\sigma}_t^2(\mathbf{x}_{t+1}, \mathcal{D}_t)]$;
  Sample $\mathcal{D}_{t+1} \sim \widetilde{\mathbf{p}}_{t+1}$;
end

BBKB At step t start batch and for $t' > t$ continue selecting

$$\widetilde{u}_{t'}(\mathbf{x}_i) = \widetilde{\mu}_t(\mathbf{x}_i, \mathcal{D}_t) + \widetilde{\alpha}_t \widetilde{\sigma}_{t'}^2(\mathbf{x}_i, \mathcal{D}_t)$$

while $\sum_{s=t}^{t'} \widetilde{\sigma}_s^2(\mathbf{x}_s, \mathcal{D}_t) \le 1$.

Main result: BBKB is scalable and no regret

Theorem: Let $\widetilde{\alpha}_t \approx \sqrt{\sum_{s=1}^t \log(1 + 2\widetilde{\sigma}_{s-1}^2(\widetilde{\mathbf{x}}_s, \mathcal{D}_{s-1}))} + \sqrt{\lambda} F$. Assume $f \in \mathcal{H}$ arbitrary and $\|f\| \le F$. Then w.h.p., $\forall\, t \in [T]$ and all $\mathbf{x} \in \mathcal{A}$,

$$\sigma_t^2(\mathbf{x})/2 \le \widetilde{\sigma}_t^2(\mathbf{x}) \le 2\sigma_t^2(\mathbf{x}) \qquad \text{and} \qquad |\mathcal{D}_t| \le \mathcal{O}(d_{\text{eff}} \log(t/\delta)),$$

and BBKB incurs at most regret $R_T^{\text{BBKB}} \le 4R_T^{\text{GP-UCB}}$
and runs in $\mathcal{O}(TAd_{\text{eff}}^2)$ time with average batch size $P \ge \Omega(T/d_{\text{eff}})$.

🙂 adapts to GP's effective dimension/rank: $d_{\text{eff}} \triangleq \sum_{i=1}^T \sigma_T^2(\widetilde{\mathbf{x}}_i)$
🙂 $\widetilde{\beta}_t$ computable in $\widetilde{\mathcal{O}}(Ad_{\text{eff}}^2)$ time replacing worst-case bounds
🙂 No assumptions on GP structure (e.g., not only stationary GPs)
😞 No free lunch: worst-case falls back to GP-UCB

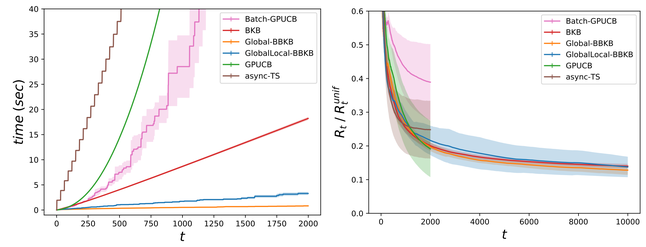
🤔 DTC is not a GP (not consistent), but now a justified heuristic
🤔 Easy extension to infinite $\mathcal{A}$, but how to optimize posterior?

🤔 Big batches and rare resparsification with no loss of accuracy?

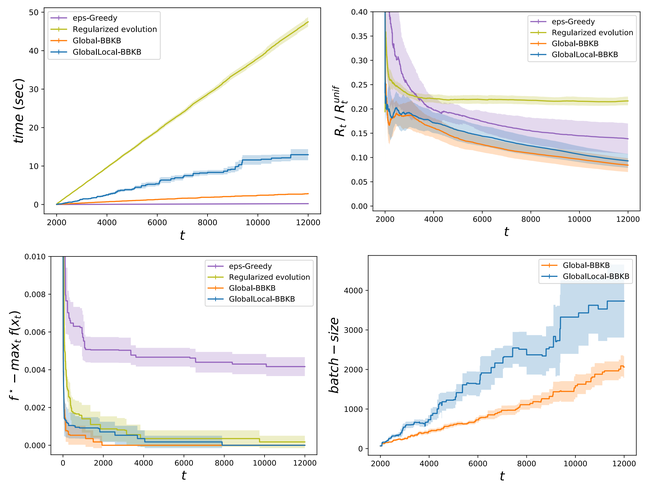
"Not too big" Lemma: ↳ valid UCBs vs **"Not too small" Lemma:** ↳ batch-size $= \Omega(t)$

Experiments

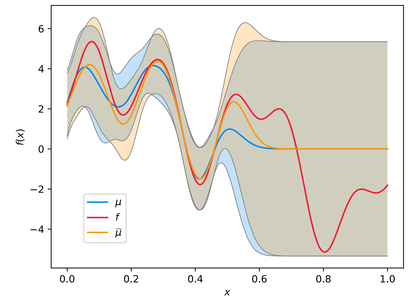
Cadata dataset ($A = 20640$, $d = 8$, $T = 10000$)



NAS-bench-101 dataset ($A = 12416$, $d = 19$, $T = 12000$)



Not simply an approximate GP-UCB



Sparse approximation **regularizes!** New bias-variance trade-off!

[1] Srinivas et al. Gaussian process optimization in the bandit setting: No regret and experimental design. ICML'10 [2] Mutný et al. Efficient high-dimensional Bayesian optimization with additivity and quadrature Fourier features. NeurIPS'18
[3] Kuzborskij et al. Efficient linear bandits through matrix sketching. AISTATS'19 [4] Abbasi-Yadkori et al. Improved algorithms for linear stochastic bandits. NeurIPS'11 [5] Calandriello et al. Gaussian Process Optimization with Adaptive Sketching: Scalable and No Regret. COLT'19