Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design

Rudenko Varvara

April, 2021

Summary

Abstract

- Many applications require optimizing an unknown, noisy function that is expensive to evaluate. We formalize this task as a multi-armed bandit problem, where the payoff function is either sampled from a Gaussian process (GP).
- We resolve the important open problem of deriving regret bounds for this setting, which imply novel convergence rates for GP optimization. We analyze GP-UCB, an intuitive upper-confidence based algorithm, and bound its cumulative regret in terms of maximal information gain, establishing a novel connection between GP optimization and experimental design.

Problem Statement and Background

- UCB in bandit setting:

$$A_t = argmax \left[Q_t(a) + \beta_t \sqrt{\frac{lnt}{N_t(a)}} \right], \tag{1}$$

  where Q_t - evaluation of the action value after selecting t rounds.
- D - unity of all sensor locations, f(x) is the temperature at x, and sensor accuracy is quantified by the noise variance.
- In our case, there are too many hands - we complicate the algorithm
- Gaussian process - a stochastic process, such that every finite collection of those random variables has a multivariate normal distribution.

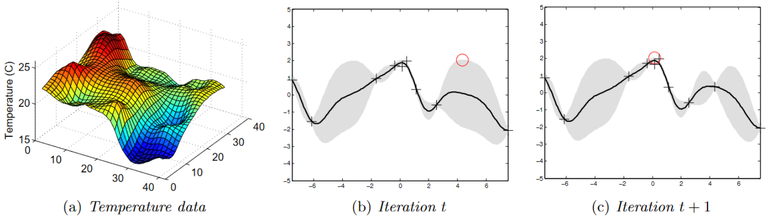


Figure: (a) Example of temperature data collected by a network of 46 sensors at Intel Research Berkeley.

- Gaussian noise, the posterior over f is a GP distribution again, with mean $\mu_T(x)$, covariance $k_T(x, x')$ and variance ${\sigma_T}^2(x)$:

$$\mu_T(x) = k_T(x)^T (K_T + \sigma^2 I)^{-1} y_T \tag{2}$$

$$k_T(x, x') = k(x, x') - k_T(x)^T (K_T + \sigma^2 I)^{-1} k_T(x') \tag{3}$$

$${\sigma_T}^2(x) = k_T(x, x) \tag{4}$$

$$x_t = argmax_{x \in D} \ \sigma_{t-1}(x), \tag{5}$$

- Another idea is to pick points as:

$$x_t = argmax_{x \in D} \ \mu_{t-1}(x), \tag{6}$$

- A combined strategy is to choose:

$$x_t = (argmax_{x \in D} \ \mu_{t-1}(x) + {\beta_t}^{\frac{1}{2}} \ \sigma_{t-1}(x)) \tag{7}$$

Algorithm 1 The GP-UCB algorithm.

Input: Input space D; GP Prior $\mu_0 = 0,\ \sigma_0,\ k$

for $t = 1, 2, \ldots$ **do**

  Choose $\boldsymbol{x}_t = \operatorname*{argmax}_{\boldsymbol{x} \in D} \mu_{t-1}(\boldsymbol{x}) + \sqrt{\beta_t}\sigma_{t-1}(\boldsymbol{x})$

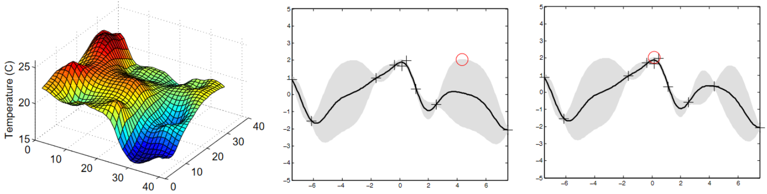
  Sample $y_t = f(\boldsymbol{x}_t) + \epsilon_t$

  Perform Bayesian update to obtain μ_t and σ_t

end for

where β_t are appropriate constants.

- Bayesian update is a method of statistical inference in which Bayes' theorem is used to update the probability for a hypothesis as more evidence or information becomes available.



(a) *Temperature data* (b) *Iteration t* (c) *Iteration $t + 1$*

Figure: (b,c) Two iterations of the GP-UCB algorithm. It samples points that are either uncertain (b) or have high posterior mean (c).

Experiments

- We compare GP-UCB with heuristics such as the Expected Improvement (EI) and Most Probable Improvement (MPI), and with naive methods which choose points of maximum mean or variance only, both on synthetic and real sensor network data.
- Expected improvement is defined as

$$EI(x) = Emax(f(x) - f(x^+), 0), \tag{8}$$

where $f(x^+)$ is the value of the best sample so far and x^+ is the location of that sample.

- Most Probable Improvement:

$$PI(x) = P(f(x) \geq f(x^+)) = F\left(\frac{\mu(x) - f(x^+)}{\sigma(x)}\right) \tag{9}$$

where $F(.)$ is the normal cumulative distribution function.

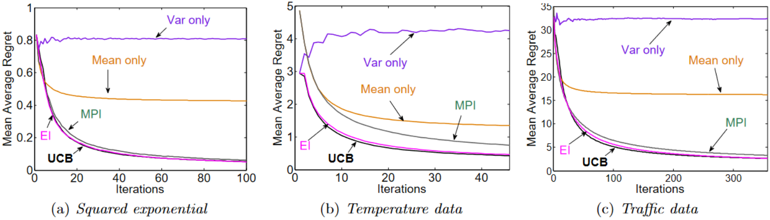


Figure: Comparison of performance: GP-UCB and various heuristics on synthetic (a), and sensor network data (b, c).

Conclusions

- We analyze GP-UCB, an intuitive algorithm for GP optimization, when the function is either sampled from a known GP, or has low RKHS norm.
- We bound the cumulative regret for GP-UCB interms of the information gain due to sampling, establishing a novel connection between experimental design and GP optimization.
- By bounding the information gain for popular classes of kernels, we establish sublinear regret bounds for GP optimization for the first time.
- We evaluate GP-UCB on sensor network data, demonstrating that it compares favorably to existing algorithms for GP optimization.

Thank you!