Gaussian Process Bandits

Emile Mathieu
Department of Computer Science
Ecole des Ponts ParisTech
emile.mathieu@eleves.enpc.fr

Abstract

A lot of applications require to optimize black-box, noisy function that is expensive to evaluate. This task can be formulated as a multi-armed bandit problem, where the reward function is sampled from a Gaussian Process (GP) or has low RKHS norm. This setting was first studied by Srinivas et al. [9], where the authors proposed GP-UCB, a UCB-like algorithm. We propose a Thompson sampling algorithm adapted to this setting.

1 Introduction

1.1 Problem Statement

The problem we consider is the sequential optimization of an unknown reward function $f : D \to \mathbb{R}$. At each round t, we choose a point $\mathbf{x}_t \in D$ and get to observe the function value perturbed by noise: $y_t = f(\mathbf{x}_t) + \epsilon_t$. Our goal is to perform as well as $\mathbf{x}^\star = \arg\max_{\mathbf{x} \in D} f(\mathbf{x})$, which is maximizing the sum of rewards $\sum_t^T f(\mathbf{x}_t)$.

In bandit settings, the natural performance metric is the cumulative regret, which is the loss in reward due to not knowing f's maximum. Formally, the cumulative regret R_T after T rounds is the sum of instantaneous regret: $R_T = \sum_t^T r_t$, with $r_t = f(\mathbf{x}^\star) - f(\mathbf{x}_t)$.

1.2 Related work

The work of Srinivas et al. [9] generalizes stochastic linear optimization in a bandit setting, where the unknown function comes from a finite-dimension linear space. For the linear setting, Dani et al. [4] gave a near-complete characterization which explicitly depends on the dimensionality of the space.

However, in the GP setting, the challenge is to characterize complexity in a different manner since GPs are nonlinear random functions, and in some way, live in an infinite-dimensional linear space. Srinivas et al. [9] succeeded in bounding cumulative regret for GP-UCB in terms of the information gain $I(\mathbf{y}_A; \mathbf{f}_A) = \frac{1}{2} \log |I + \sigma^{-2} \mathbf{K}_A|$ (for a GP), with $A = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$ and $\mathbf{K}_A = [k(\mathbf{x}, \mathbf{x}')]_{\mathbf{x}, \mathbf{x}' \in A}$. They proved these bounds for reward functions which are either sampled from a GP, or have low RKHS norm. Then they also bounded the information gain for several popular class of kernels, and established sublinear regret bounds for GP optimization.

There is an extensive literature about GP optimization. Several heuristics for trading off exploration and exploitation have been proposed to tackle GP optimization: such as Expected Improvement (Mockus et al. [7]) and Most Probable Improvement (Mockus [6]).

2 Gaussian Processes

2.1 Overview

Brochu et al. [3] provides a comprehensive review of and motivation for Bayesian optimization using GPs. This review is a nice way to get familiar with GPs.

GP can be seen as an extension of the multivariate Gaussian distribution (as this distribution is a generalization of the univariate Gaussian) to an infinite-dimension distribution for which any finite combination of dimensions will be a Gaussian distribution. In this manner, a GP is a distribution over functions, completely specified by a mean function μ and a covariance function k: $\forall \mathbf{x} \in \mathcal{D}$

$$f(\mathbf{x}) \sim \mathcal{GP}\left(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right)$$

Common choices of covariance functions are:

- Finite dimensional linear: $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$
- Squared Exponential kernel: $k(\mathbf{x}, \mathbf{x}') = \exp\{-(2l^2)^{-1} \|\mathbf{x} - \mathbf{x}'\|^2\}$
- Matern kernel: $k(\mathbf{x}, \mathbf{x}') = (2^{1-\nu}/\Gamma(\nu)) r^\nu B_\nu(r)$, $r = (\sqrt{2\nu}/l)\,\|\mathbf{x} - \mathbf{x}'\|$

2.2 Gaussian Processes reward functions

Gaussian Processes (Rasmussen and Williams [8]) allow smoothness assumptions about the reward function f to be encoded through the choice of kernel in a flexible non parametric way.

Here, the reward function f can be assumed to come from a GP, and perturbed with gaussian noise: $y = f(\mathbf{x}) + \epsilon$, with $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and $f \sim \text{GP}(0_d, k(\cdot,\cdot))$. Hence $y = (f + \epsilon) \sim \text{GP}(0_d, k(\cdot,\cdot) + \sigma^2)$. The reward can also be assumed to have low RKHS norm.

Either way, algorithms (GP-UCB and GP-TS) uses a $\text{GP}(0_d, k(\cdot,\cdot))$ as a prior distribution over f. A major advantage of working with GPs is the existence of simple analytic formulae for the mean and the covariance of the posterior distribution.

Indeed, for a noisy sample $\mathbf{y}_t = [y_1, \dots, y_T]^T$ at points $A_T = \{\mathbf{x}_1, \dots, \mathbf{x}_T\}$, with $y_t = f(\mathbf{x}_t) + \epsilon_t$ (with i.i.d. $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$), the posterior over f is still a GP distribution. The mean and variance of this posterior GP distribution are:

- $\mu_t = k_{t-1}(\mathbf{x})^T \left(K_{t-1} + \sigma^2 I_d\right)^{-1} y_t$
- $k_t = k(\mathbf{x}, \mathbf{x}') - k_{t-1}(\mathbf{x})^T \left(K_{t-1} + \sigma^2 I_d\right)^{-1} k_{t-1}(\mathbf{x}')$
- $\sigma_t^2 = k_t(\mathbf{x}, \mathbf{x})$

with $k_t(\mathbf{x}) = [k(\mathbf{x}_1, \mathbf{x}), \dots, k(\mathbf{x}_T, \mathbf{x})]^T$ and $\mathbf{K}_T = [k(\mathbf{x}, \mathbf{x}')]_{\mathbf{x}, \mathbf{x}' \in A_T}$.

3 Bandit's Algorithms

3.1 GP-UCB

Srinivas et al. [9] introduces an upper-confidence based algorithm named Gaussian Process-Upper Confidence Bound (GP-UCB) which is detailed in Algorithm 1. At each time step t, it chooses

$$\mathbf{x}_t = \arg\max_{\mathbf{x} \in \mathcal{D}} \mu_{t-1}(\mathbf{x}) + \sqrt{\beta_t} \sigma_{t-1}(\mathbf{x})$$

This choice implicitly negotiates the exploration-exploitation trade-off. It can be interpreted as greedily selecting points $\mathbf{x}$ such that $f(\mathbf{x})$ should be a reasonable upper bound on $f(\mathbf{x}^\star)$. The exploration part $\sigma_{t-1}(\mathbf{x})$, can be derived from an experimental design approach, with the notion of information gain.

This selection rule is also motivated by the UCB algorithm for the classical multi-armed bandit problem (Auer [2], Kocsis and Szepesvári [5]).

Algorithm 1 GP-UCB

```
Require: $k$
$\mu \leftarrow 0_d$
for $t \leftarrow 1$ to $T$ do
    $\beta_t \leftarrow 2\log(|D|\, t^2\pi^2/6\delta)$
    Choose $a_t \leftarrow arg\max_i \mu_{t-1} + \sqrt{\beta_t}\sigma_{t-1}$
    Observe $y_t = f(\mathbf{x}_t) + \epsilon_t$
    $\mu_t = k_{t-1}(\mathbf{x})^T \left(K_{t-1} + \sigma^2 I_d\right)^{-1} y_t$
    $k_t = k(\mathbf{x}, \mathbf{x}') - k_{t-1}(\mathbf{x})^T \left(K_{t-1} + \sigma^2 I_d\right)^{-1} k_{t-1}(\mathbf{x}')$
    $\sigma_t^2 = k_t(\mathbf{x}, \mathbf{x})$
end for
```

3.2 GP-TS

Thompson Sampling (TS) is one of the earliest heuristics for multi-armed bandit problems. The first version of this Bayesian heuristic dates to Thompson [10]. It is a member of the family of *randomized probability matching* algorithms.

The main idea is to assume a prior on parameters θ of the reward function and to update these parameters by computing the posterior distribution $P(\theta|\mathcal{D}_{1:t})$ after observing a reward. At each time step t, the algorithm plays an arm according to its posterior probability of being the best arm.

A generalization of Thompson Sampling algorithm for the stochastic contextual multi-armed bandit problem with linear pay-off functions is given by Agrawal and Goyal [1].

We propose a generalization of Thompson Sampling algorithm for the stochastic multi-armed bandit problem with reward functions sampled from GP. In this setting, the elements of Thompson sampling are as follows:

- The set Θ of parameters θ of the distribution of f is the set of tuples $\{\mu(\cdot), k(\cdot,\cdot)\}$, with $\mu : \mathcal{D} \to \mathbb{R}$ and $k : \mathcal{D} \times \mathcal{D} \to \mathbb{R}$.
- The prior distribution $P(\theta)$ on those parameters is a gaussian process: $f \sim \text{GP}(\mu_0(\cdot), k_0(\cdot,\cdot))$. Without loss of generality[1], we assume $\mu_0 = 0_d$.
- We observe triplets $\mathcal{D} = \{\mathbf{x}; y\}_t$.
- At each step t, the reward y_t comes from $y_t = f(\mathbf{x}_t) + \epsilon$, with $\epsilon \sim \mathcal{N}(0, \sigma^2)$. Thus

  $$(f + \epsilon) \sim \text{GP}(\mu(\cdot), k(\cdot,\cdot) + \sigma^2)$$

  Consequently, the likelihood function is

  $$P(y|\theta, \mathbf{x}) = \frac{1}{\sqrt{2\pi(\sigma_t^2 + \sigma^2)}} \exp\left\{ -\frac{(\mathbf{x} - \mu_t)^2}{2(\sigma_t^2 + \sigma^2)} \right\}$$

- The posterior distribution $P(\theta|\mathcal{D}_{1:t}) \propto P(\mathcal{D}_{1:t}|\theta)P(\theta)$ yields the following update on (functions) parameters $\{\mu(\cdot), k(\cdot,\cdot)\}$:
  - $\mu_t = k_{t-1}(\mathbf{x})^T \left(K_{t-1} + \sigma^2 I_d\right)^{-1} y_t$
  - $k_t = k(\mathbf{x}, \mathbf{x}') - k_{t-1}(\mathbf{x})^T \left(K_{t-1} + \sigma^2 I_d\right)^{-1} k_{t-1}(\mathbf{x}')$.

The proposed GP-TS algorithm is summed up in pseudo-code in Algorithm 2.

3.3 Kernel learning

Once a parametrized kernel has been selected for the GP prior over f, one need to choose its parameters. For a squared exponential kernel $k(\mathbf{x}, \mathbf{x}') = \exp\{-(2l^2)^{-1}\|\mathbf{x} - \mathbf{x}'\|^2\}$, the lengthscale l is a parameter to be tuned.

[1] According to Rasmussen and Williams [8]

Algorithm 2 GP-TS

```
Require: $k$
$\mu \leftarrow 0_d$
for $t \leftarrow 1$ to $T$ do
    Sample $f_t \sim \mathbf{GP}(\mu_t, K_t)$
    Choose $a_t \leftarrow arg\max_a f_t(\mathbf{x}_{a,t})$
    Observe $y_t = f(\mathbf{x}_t) + \epsilon_t$
    $\mu_t = k_{t-1}(\mathbf{x})^T (K_{t-1} + \sigma^2 I_d)^{-1} y_t$
    $k_t = k(\mathbf{x}, \mathbf{x}') - k_{t-1}(\mathbf{x})^T (K_{t-1} + \sigma^2 I_d)^{-1} k_{t-1}(\mathbf{x}')$
    $\sigma_t^2 = k_t(\mathbf{x}, \mathbf{x})$
end for
```

A classical way to do so is to compute the log likelihood and to find parameters which maximizes it. These parameters are then called maximum likelihood estimators (MLE). For the squared exponential kernel, because $y_t =$ the log likelihood is the following:

$$\mathcal{N}(\mathbf{y}|0, \mathbf{K}) = (2\pi)^{-n/2} |\mathbf{K} + \sigma^2 I|^{-1/2} \exp\{-\mathbf{y}^T (\mathbf{K} + \sigma^2 I)^{-1} \mathbf{y}\}$$

with $\mathbf{y} = [y_1, \ldots, y_T]^T$ and $\mathbf{K}$ the positive definite kernel matrix $[k(\mathbf{x}, \mathbf{x}')]_{\mathbf{x}, \mathbf{x}' \in A_T}$. This MLE problem can then be formulated as the following otpimization problem:

$$l^\star \in \arg\min E(l)$$

$$\text{with} \quad E(l) = \frac{1}{2} \log |\mathbf{K} + \sigma^2 I| + \frac{\mathbf{y}^T (\mathbf{K} + \sigma^2 I)^{-1} \mathbf{y}}{2} \propto -\log \mathcal{N}(\mathbf{y}|0, \mathbf{K})$$

4 Experiments

4.1 Compared methods

We compare performances between GP-UCB, GP-TS and two naives methods which choose points of maximum mean or variance only, both on synthetic and real sensor network data.

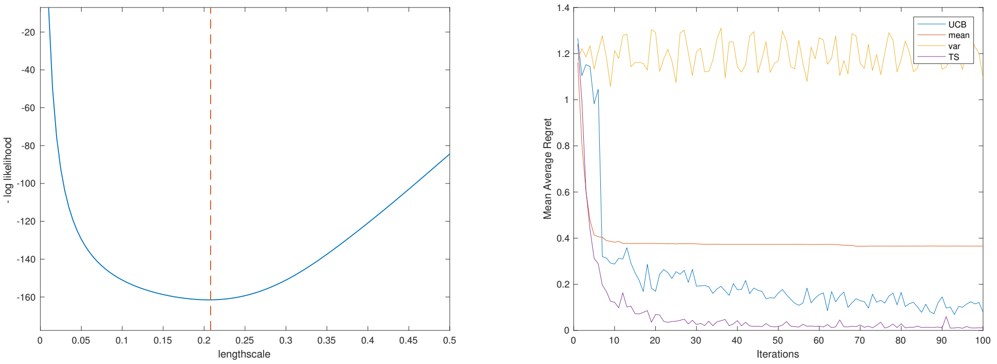
Everything needed to run the experiments is available online at `https://github.com/emilemathieu/Project_GP-bandits`. There are temperature and trafic data, but also the Matlab code needed to run the following experiments.

4.2 Synthetic data

For synthetic data, we use the same procedure and parameters as in [9]. Hence, we sample random functions from a GP with a squared exponential kernel with lengthscale parameter $l = 0.2$. The sampling noise variance σ^2 was set to 0.025. Our decision set $D = [0, 1]$ is uniformly discretized into 1000 points.

First, we learn the lengthscale of the squared exponential kernel with the method described in the Kernel Learning subsection. We discretize the lengthscale space, compute $E(l)$ for each lengthscale l and return the lengthscale l which minimizes $E(l)$). These values $E(l)$ are plotted in blue in Figure 1a, and the best lengthscale $l^\star$ is represented by a vertical red dashed line.

Then, we run each algorithm for $T = 100$ iterations with $\delta = 0.1$ and $l = l^\star$, averaging over 150 trials (samples from the kernel). We follow the choice of β_t as recommended by Theorem 1 in [9], which is $\beta_t = 2\log(|D|t^2\pi^2/6\delta)$. Obtained results are shown in Figure 1b.



(a) Kernel's lengthscale learning

(b) Performances on synthetic data

Figure 1: Comparison of performances: GP-UCB, TS-UCB and 2 naive heuristics on synthetic data (b). Log likelihood for different values of the squared exponential kernel's lengthscale (a).

4.3 Real data

4.3.1 Temperature data

Such as in Srinivas et al. [9], we use temperature data collected from 45 sensors deployed at Intel Research Berkeley [2] during a month at one hour intervals.

Indeed, we might want to find locations of highest temperature in a building by sequentially activating sensors in a spatial network and regressing on their measurements. In such a context, D consists of all sensor locations, $f(x)$ is the temperature at x, and sensor accuracy is quantified by the noise variance. Each activation draws battery power, so we want to sample from as few sensors as possible.

We take the first two-thirds of the dataset to compute the empirical covariance of the sensor mesures, and use it as the kernel matrix.The functions f for optimization are taken from the remaining third of the data set.

We take as parameters $T = 45$, $\sigma^2 = 5$, $\delta = 0.1$. Results are averaged over 187 runs. Mean averaged regrets are shown Figure 2a.

4.3.2 Trafic data

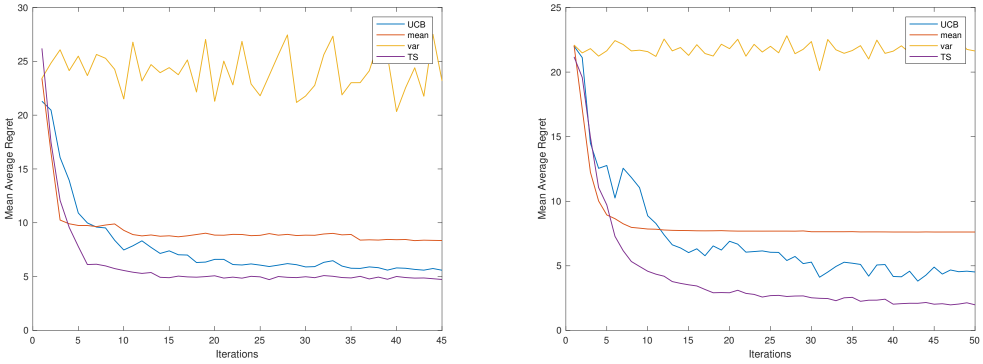
We also use data from trafic sensors deployed along the highway 5 South in San Diego [3]. The goal was to find the point of minimum speed in order to identify the most congested portion of the highway. We used trafic speed data from 48 sensors for 3 days (24/12/2016, 25/12/2016 and 03/01/2017) from 6 AM to 11 AM (local time), with 1 minute interval between each measure.

Again, the first two third of the dataset is used to compute the empirical covariance for the kernel matrix, and we test algorithms on the last third.

We take as parameters $T = 51$, $\sigma^2 = 4.78$, $\delta = 0.1$. Results are averaged over 360 runs. Mean averaged regrets are shown Figure 2b.

[2] This dataset and related information can be found at http://db.csail.mit.edu/labdata/labdata.html. Data preprocessing is needed so as to filter sensors with no missing data and to keep only one point by sensor each hour during a month.

[3] We wrote and scheduled a Python job which downloads every minutes data from the following webpage http://www.dot.ca.gov/dist11/d11tmc/sdmap/showmap.php?route=sb5



(a) Temperature data

(b) Trafic data

Figure 2: Comparison of performances: GP-UCB, TS-UCB and 2 naive heuristics on temperature data (a) and trafic data (b).

4.4 Analysis

On Figures 1b and 2, one can see that GP-TS slightly outperforms GP-UCB on both synthetic data and real data. Naive heuristics which choose points with the highest variance or the highest mean have clearly and hopefully worse performances.

5 Conclusion

We have shown that a Thompson sampling algorithm for the MAB problem where the payoff function is sampled from a GP or has low RKHS norm, can have better performances than GP-UCB.

References

[1] Shipra Agrawal and Navin Goyal. Thompson sampling for contextual bandits with linear payoffs. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, pages 127–135, 2013. URL `http://jmlr.org/proceedings/papers/v28/agrawal13.html`.

[2] Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *J. Mach. Learn. Res.*, 3:397–422, March 2003. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=944919.944941`.

[3] Eric Brochu, Vlad M. Cora, and Nando de Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. Technical Report TR-2009-23, Department of Computer Science, University of British Columbia, November 2009.

[4] Varsha Dani, Thomas P. Hayes, and Sham M. Kakade. Stochastic linear optimization under bandit feedback. In *In submission*, 2008.

[5] Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *Proceedings of the 17th European Conference on Machine Learning*, ECML'06, pages 282–293, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3-540-45375-X, 978-3-540-45375-8. doi: 10.1007/11871842_29. URL `http://dx.doi.org/10.1007/11871842_29`.

[6] Jonas Mockus. *Bayesian Approach to Global Optimization*. Kluwer Academic Publishers, 1989.

[7] Jonas Mockus, V. Tiesis, and Antanas Zilinskas. *Toward Global Optimization*, volume 2. 1978.

[8] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005. ISBN 026218253X.

[9] Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias W. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 1015–1022, 2010. URL `http://www.icml2010.org/papers/422.pdf`.

[10] W.R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3–4):285–294, 1933.