Master research Internship

Bibliographic report

Hierarchical Bandits for "Black Box" Optimization and Monte-Carlo Tree Search

Domain: Optimization - Machine Learning

Author:
Xuedong Shang

Supervisor:
Emilie Kaufmann
Michal Valko
SequeL - Inria Lille

Abstract Heuristics like Monte-Carlo Tree Search (MCTS), which trades off well *exploration* and *exploitation*, are widely used for sequential global optimization problems, and has led to some great success especially in game AI designing. In many cases, the exploration phase follows the famous *optimism in the face of uncertainty* principle, which is encountered in the so-called *multi-armed bandits* problem. However, recent studies on these models shows that they are not optimal for the optimization purpose, and that methods based on *best arm identification* are preferred. During this internship, some approaches based on these new statistic tools will be investigated, in the purpose of finding some new procedures which are more efficient to tackle with sequential optimization problems both in a practical and theoretical way.

Keywords —- reinforcement learning, global optimization, multi-armed bandits, simple regret, cumulative regret, hyperparameter optimization.

Contents

Introduction

Sequential global optimization consists in optimizing some complicated function by using a sequence of (noisy) observations of it. Optimization could be for example maximization of a *reward* or minimization of a cost. It is a crucial problem in many different domains such as biology and chemistry Floudas and Pardalos 2000, engineering Wang and Shan 2007, bioinformatics Moles et al. 2003, finance Ziemba and Vickson 2006, etc. In these cases, we often do not make extra hypothesis on the regularity of the function we want to optimize (so-called "black box"). Thus it can be very costly to evaluate the function and a good strategy for choosing the next observation is needed so that we can find the optimum as quickly as possible. Recently, this kind of black box optimization is motivated in particular by applications in automatic parameter optimization of machine learning algorithms Hoffman et al. 2014, Li et al. 2016.

In the past few years, such optimization algorithms have been widely inspired by literature of the so-called multi-armed bandits models. These algorithms are based on a hierarchical exploration (in a tree form) of the domain of the function, with the help of the optimism principle for choosing which part of the tree to explore. This work brought some breakthroughs especially in the field of AI designing, e.g. for the game of Go Silver et al. 2016.

A simple way to describe the multi-armed bandits scenario is to consider K arms labeled by integers from 1 to K. Each arm $k \in \{1, \ldots, K\}$ is characterized by an unknown distribution ν_k. At each step t, an arm k_t is selected and some reward $x_t \sim \nu_{k_t}$ is returned. The optimism principle is used when we are interested in maximizing the sum of rewards. Meanwhile, another objective could have been preferred, which is to decide as quickly as possible which arm has the highest reward on average. Recent works showed that optimal algorithms for this kind of best arm identification problems are totally different from those for reward maximization problems.

In this internship, we will focus on the sequential global optimization problem using the hierarchical exploration with the help of best arm identification techniques instead of the classic optimistic approaches. Most existing algorithms for the Monte-Carlo Tree Search problem are variants of the Upper-Confidence Tree (UCT) algorithm in Kocsis and Szepesvári 2006. Our goal is to use a different approach, in which the exploration phase will be based on a process of best arm identification, like LUCB Kalyanakrishnan et al. 2012. Several methods could be considered for this purpose, in particular methods of hierarchical optimization (hierarchical optimistic optimization Grill et al. 2015, Bubeck et al. 2011) and methods using Gaussian processes (Bayesian optimization Brochu et al. 2010).

For the rest of this bibliographic review, we will begin by a more formal definition of the bandit problem and some classic methods as well. Then we will focus on the black box optimization, for which we will present two kinds of methods: Bayesian optimization and hierarchical optimistic optimization. Finally, we will discuss about how can these techniques be used in the context of hyperparameter optimization for machine learning problems before concluding.

1 Multi-armed Bandits Problem

In this section, we will formulate the bandit problem in a more rigorous way. Then, a simple but remarkable algorithm, which is the Upper-Confidence Bound (UCB) algorithm Auer et al. 2002, will be presented. The UCB strategy aims at minimizing the cumulative regret (i.e. reward maximization), which is not a good way to base the strategies on in some circumstances. Thus, an alternative philosophy, say best arm identification, could be preferred. The last subsection will be dedicated to best arm identification, and some discussions on the difference between best arm identification and reward maximization (which is referred to as regret minimization later) will be given as well.

1.1 Problem Formulation

We consider K arms that follow K unknown $[0,1]$-valued distributions $(\nu_k)_{1\leq k\leq K}$. At time t, a player plays one arm $k_t \in \{1,\ldots,K\}$ and receives a reward $x_t \sim \nu_{k_t}$. Here x_t is an independent observation of the distribution corresponding to the chosen arm.

Let μ_k be the expectation of the unknown distribution ν_k and μ^* be the expectation of the optimal arm. Since the laws of each arm are unknown, one must explore all the arms to collect information (which refers to as exploration) and then takes the most profitable arm to act (which refers to as exploitation). This is the famous *exploration-exploitation dilemma.*

A *policy* or an *allocation strategy* is an algorithm which chooses at step n an arm k_n to play based on the past plays. Sometimes, this allocation strategy is coupled with a recommendation strategy, that selects an arm j_n as a guess for the best arms (note that j_n can surely be different from k_n). A simple example of recommendation could be the arm with the highest empirical mean until now.

In order to evaluate the performance of a given policy, two criteria related to the notion of regret are proposed, which allow us to define the speed of convergence of the average reward obtained by the policy towards the average optimal reward.

Definition 1 (Simple regret)**.** *At time n, a given policy which observes a sequence of rewards $(x_t)_{1\leq t\leq n}$ and recommendations $(j_t)_{1\leq t\leq n}$ suffers from a simple regret:*

$$S_n = \mu^* - \mu_{j_n}.$$

The simple regret measures the difference between the expected reward of the optimal arm and the recommended arm at this moment. In practice, another criterion is often considered as well.

Definition 2 (Cumulative regret)**.** *At time n, a given policy which observes a sequence of rewards $(x_t)_{1\leq t\leq n}$ suffers from a cumulative regret:*

$$R_n = n\mu^* - \sum_{1\leq t\leq n} \mu_{k_t}.$$

The cumulative regret measures the difference between the expected cumulative reward obtained by the optimal arm and the cumulative reward returned by the given policy.

We are particularly interested by the expectation of the cumulative regret $\mathbb{E}[R_n]$. Let us denote $\Delta_k = \mu^* - \mu_k$ as the difference between the optimal arm and arm k and $T_k(n)$ as the number of times that arm k has been played until time n. Then the expectation of the cumulative regret can be reformulated as follows:

$$\mathbb{E}[R_n] = \mathbb{E}\left[\sum_{k=1}^{K} T_k(n)\Delta_k\right].$$

Basically speaking, a good policy for (cumulative) regret minimization should choose any sub-optimal arm as rarely as possible.

Remark 1. *It is easy to see that for any regret minimization strategy, results on cumulative regret can be readily extended to simple regret. For instance, at time n, the algorithm recommends arm h with a probability of $T_h(n)/n$ (no matter what regret minimization is), then we have:*

$$\mathbb{E}[S_n] \leq \mathbb{E}\left[\frac{R_n}{n}\right].$$

1.2 The Upper-Confidence Bound Algorithm

The UCB algorithm, popularized by Auer et al. 2002, is one of the first strategies that achieves a uniform logarithmic regret over n and it follows the optimism in the face of uncertainty principle. That is to say, despite lack of knowledge in which actions are best, we can still construct an optimistic guess that picks an optimal arm in the most favorable environments that are compatible with the observations. Here by "compatible environments" we mean the set of possible distributions of the arms that are likely to have generated the observed rewards. The simplest version of UCB is given below.

Algorithm 1 (UCB$_1$)**.**
Initialization: *At first, the UCB policy pulls each arm once.*
Loop: *At time $n+1$, the UCB policy pulls the arm with largest B-values:*

$$x_{n+1} \in \arg\max_{1 \leq k \leq K} B_{(n+1), T_k(n)(k)},$$

where the B-value of an arm k is defined as:

$$B_{t,s}(k) = \hat{\mu}_{k,s} + \sqrt{\frac{3 \log t}{2s}},$$

where $\hat{\mu}_{k,s}$ is the average reward of the first s rewards obtained by arm k.

The policy above follows the optimism in the face of uncertainty principle and the B-values defined are actually upper-confidence bounds on μ_k. Formally speaking,

$$\mathbb{P}(B_{t,s}(k) \geq \mu_k) \geq 1 - t^{-3}, \forall s \in \{1, \ldots, t\}.$$

This property is insured by the famous Chernoff-Hoeffding inequality Hoeffding 1963. We can then deduce an upper bound on the average number of times that a sub-optimal arm is played.

Proposition 3. *Each sub-optimal arm k is played at most*

$$\mathbb{E}[T_n(k)] \leq 6\frac{\log n}{\Delta_k^2} + \frac{\pi^2}{3} + 1$$

times on average.

The expectation of the cumulative regret can then be easily bounded as shown in the corollary below.

Corollary 4. *The cumulative regret of the UCB policy is bounded as*

$$\mathbb{E}[R_n] = \mathbb{E}\left[\sum_{k=1}^{K} T_k(n)\Delta_k\right] = \sum_{k=1}^{K} \Delta_k \mathbb{E}[T_k(n)] \leq 6 \sum_{\Delta_k > 0} \frac{\log n}{\Delta_k} + K(\frac{\pi^2}{3} + 1).$$

Here, we notice that any sub-optimal arm is pulled only $O(\log n)$ times.

1.3 An Example of Best Arm Identification

Until now we mostly talked about one way to find an appropriate policy which is the cumulative regret minimization. And we explained briefly how the classic policy UCB works. This setting seems to be particularly appropriate in the context of medical trials for instance, where choosing a wrong treatment may cause severe problems.

Nevertheless, minimizing the cumulative regret is not always a good way to base the policies on. One example could be the pure exploration problem in which only exploration accounts for the goal. A pure exploration problem takes place when the exploration is only constrained by resources rather than rewards (here resources can be the number of rounds for example). This kind of problem can sometimes be referred to as *budgeted multi-armed bandits problem.*

A real life example could be the AI of Go. In a given amount of CPU times, the AI has to explore the possible sequences of plays and then recommend a final decision. In this situation, cumulative regret is not quite suitable since the costs of evaluating good and bad options are the same, while we only have a limited amount of resources.

Thus, in this subsection, we will mainly discuss about another type of multi-armed bandit problem: best arm identification. Usually, the objective of best arm identification is to minimize the simple regret. But other performance measures have been investigated as well, in particular the *fixed confidence setting*, see fro example Mannor and Tsitsiklis 2004, Even-Dar et al. 2002. In a fixed confidence setting, a precision δ is given, and one aims at finding an optimal arm, while sampling as few as possible.

An exemple of best arm identification algorithm is LUCB Kalyanakrishnan et al. 2012. LUCB tries to solve the EXPLORE-m problem, of which the goal is to find a subset of m arms with the highest expected rewards.

Remark 2. *LUCB is a strategy for m-best arm identification. But we can easily apply it to a best arm identification problem by taking $m = 1$.*

LUCB consists of two elements: a stop criterion to indicate whether the algorithm should stop; and a sampling strategy to decide which arms to pick when the algorithm continues.

Stopping criterion The stopping criterion at step n in LUCB depends on a confidence interval for each mean μ_k: a lower confidence bound for arm k during round n which is $\hat{\mu}_{k,T_k(n)} - \beta(T_k(n), n)$ and an upper confidence bound which is $\hat{\mu}_{k,T_k(n)} + \beta(T_k(n), n)$. Let us denote High_n as the set of m arms with the highest empirical means during round n, and Low_n the set of $n - m$ arms with the lowest empirical means. And we define h_n^* and l_n^* as follows,

$$h_n^* = \arg\min_{h \in \text{High}_n} \hat{\mu}_{h,T_h(n)} - \beta(T_h(n), n),$$

$$l_n^* = \arg\max_{l \in \text{Low}_n} \hat{\mu}_{l,T_l(n)} + \beta(T_l(n), n).$$

Then the stopping criterion is we stop iff,

$$(\hat{\mu}_{l_n^*, T_{l_n^*}(n)} + \beta(T_{l_n^*}(n), n)) - (\hat{\mu}_{h_n^*, T_{h_n^*}(n)} - \beta(T_{h_n^*}(n), n)) < \epsilon,$$

where $\epsilon \in (0, 1)$ is a parameter. It is demonstrated that when β is sufficiently large, the mistake probability for LUCB is small.

Sampling strategy The sampling strategy for LUCB is a very simple greedy approach. Indeed, is is natural to think that h_n^* and l_n^* are more likely to lead to mistake, thus the sampling strategy for round n is simply to pick arms h_n^* and l_n^*.

To briefly summarize this section, we can notice that best arm identification and regret minimization are two different kinds of multi-armed bandits problem. In the next, we will focus on a more general problem which is the sequential global optimization or black box optimization.

2 Black Box Optimization

In this section, we will first reformulate the definitions given previously in the context of multi-armed bandits problem in a more general way. We consider the problem of optimizing sequentially an unknown noisy function $f : \mathcal{X} \to \mathbb{R}$ of which the cost of function evaluation is high.

At each step t, a strategy picks an action $\mathbf{x} \in \mathcal{X}$ and receives a reward $r_t = f(\mathbf{x_t}) + \epsilon_\mathbf{t}$ where ϵ_t is the noise. After n steps, the strategy will return a guess for a point maximizing f, denoted by $\mathbf{x}(n)$ (which is the counterpart of the recommended arm j_n in the bandit case, which corresponds to a finite set $\mathcal{X}$). We can then define the simple regret, which is also called *optimization error* in an optimization setting, in the same way as in the bandit setting,

$$S_n = \sup_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) - f(\mathbf{x}_n).$$

Suppose that f reaches its upper bound, thus we can denote $f(\mathbf{x}^*) = \sup_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$ throughout the rest of this review. The cumulative regret is simply defined as

$$R_n = \sum_{1 \le t \le n} (f(\mathbf{x}^*) - f(\mathbf{x}_t)).$$

Note that a desirable property of an optimization algorithm would be *no-regret*, i.e. $\lim_{n \to \infty} R_n / n = 0$.

In the rest of this section, two dominant approaches for this problem will be introduced, i.e. hierarchical optimistic optimization (HOO) and Bayesian optimization. We will concentrate especially on one specific method for each of those two kinds: Parallel Optimistic Optimization (POO) which is a modified version of HOO, and Gaussian Process Upper Confidence Bound (GP-UCB) for Bayesian optimization.

2.1 Hierarchical Optimistic Optimization

The main idea of HOO is to exploit as much knowledge of f as possible around its maxima, while it does not care a lot about other parts of the space. For that purpose, a tree-formed partitioning is proposed such that nodes which are deeper in the tree represent smaller measurable sub-regions of $\mathcal{X}$.

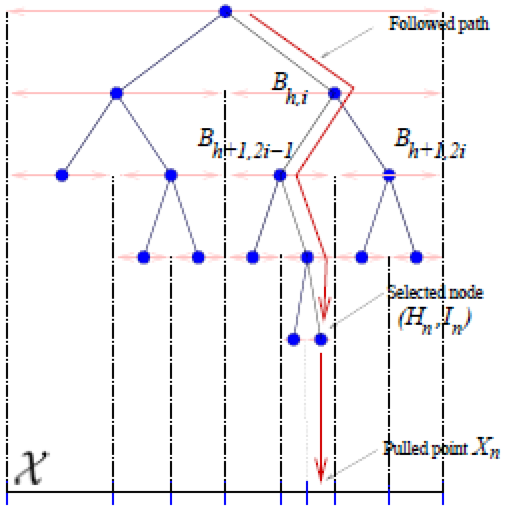

Figure 1: Extracted from Bubeck et al. 2011, illustration of the node selection in round n. In this example, $B_{h+1,2i-1}(t) > B_{h+1,2i}(t)$, thus the path includes the node $(h+1, 2i-1)$ rather than the node $(h+1, 2i)$.

2.1.1 Hierarchical Optimistic Optimization in General

HOO is an algorithm that relies on a tree-formed hierarchical partitioning $\mathcal{P} = \{\mathcal{P}_{h,i}\}$ defined recursively as follows,

$$\mathcal{P}_{0,1} = \mathcal{X},$$

$$\mathcal{P}_{h,i} = \mathcal{P}_{h+1,2i-1} \cup \mathcal{P}_{h+1,2i}.$$

As a corollary, at every depth $h \geq 0$, all the nodes form a partition of the input space $\mathcal{X}$,

$$\mathcal{X} = \bigcup_{i=1}^{2^h} \mathcal{P}_{h,i}.$$

Remark 3. *Here a simple example of binary tree partitioning is given, note that each $\mathcal{P}_{h,i}$ can also be split into several regions of the same size. In the rest of this section, we remain in this binary tree situation for the sake of simplicity.*

At each round of HOO, the function is evaluated at one point within an unexplored node at some level h. And it always chooses the node whose B-value $B_{h,i}(t)$ is the highest. $B_{h,i}(t)$ is defined as follows,

$$B_{h,i}(t) = \begin{cases} \min\{U_{h,i}(t), \max\{B_{h+1,2i-1}(t), B_{h+1,2i}(t)\}\} & \text{if } \ (h,i) \in \mathcal{T}_n \\ \infty & \text{otherwise} \end{cases}$$

where $\mathcal{T}_n$ is the sub-tree explored at step n and $U_{h,i}(t)$ is defined like,

$$U_{h,i}(t) = \begin{cases} \hat{\mu}_{h,i}(t) + \sqrt{\frac{2\ln(t)}{N_{h,i}(t)}} + \nu\rho^h & \text{if } \ N_{(h,i)}(t) > 0 \\ \infty & \text{otherwise} \end{cases}$$

where $\hat{\mu}_{h,i}(t)$ is the empirical mean of all evaluations done in the cell $\mathcal{P}_{h,i}$ (and its descendants), and $N_{h,i}$ is the number of these evaluations. Here $U_{h,i}(t)$ can be interpreted as an upper confidence bound on $f(\mathbf{x})$ where $\mathbf{x} \in \mathcal{P}_{h,i}$.

Basically speaking, HOO chooses a path from root to leaf that maximizes the minimum value $U_{h,i}(t)$ among all cells at level h. Figure 1 illustrated how HOO chooses its path. Here the third term $\nu\rho^h$ in $U_{h,i}(t)$ is supposed to be a bound on the difference $f(\mathbf{x}^*) - f(\mathbf{x})$ over a region at depth h that contains one of the maxima of the function. As a consequence, the HOO algorithm needs some prior knowledge on the smoothness of the function as it needs some information on $\nu\rho^h$.

Indeed, a global optimization problem without any assumptions on the regularity of f would be almost a "mission impossible". Most of the algorithms make a very weak hypothesis that f possesses at least some local smoothness. This smoothness is often quantified by a certain semi-metric l, e.g. $l(\mathbf{x}, \mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|^\alpha$ with $\alpha < 1$. One simple example of smoothness assumption could be:

$$f(\mathbf{x}^*) - f(\mathbf{x}) \leq l(\mathbf{x}^*, \mathbf{x})$$

for x close to one of the function maxima.

2.1.2 Parallel Optimistic Optimization

POO Grill et al. 2015 is a modified version of HOO. Indeed, the semi-metric assumption made in HOO is not necessary. HOO does not exploit full information of the metric value, but only uses ν and ρ, thus a single assumption that relates directly f to the partitioning could be preferred.

Given a global maximum $\mathbf{x}^*$, let us denote i_h^* the index of the only cell at depth h that contains $\mathbf{x}^*$. The assumption on the local smoothness of f of POO would be: there exist $\nu > 0$ and $\rho \in (0, 1)$ s.t.,

$$\forall h \geq 0, \forall \mathbf{x} \in \mathcal{P}_{h,i_h^*}, f(\mathbf{x}) \geq f(\mathbf{x}^*) - \nu\rho^h.$$

POO then uses HOO as subroutine in which several instances of HOO are run at the same time. Each instance of HOO is run with a different (ν, ρ). At the end, it chooses the (ν^*, ρ^*) that performs the best and returns one of the points chosen randomly by the corresponding HOO instance.

Finally, in order to measure the complexity of the optimization problem directly in terms of partitioning, a notion of *near-optimality dimension* is defined,

$$d(\nu, \rho) = \inf\{d' \in \mathbb{R}^+ : \exists C > 0, \forall h \geq 0, \mathcal{N}_h(2\nu\rho) \leq C\rho^{-d'h}\},$$

where $\mathcal{N}_h(2\nu\rho)$ is the number of cells $\mathcal{P}_{h,i}$ s.t.,

$$\sup_{x \in \mathcal{P}_{h,i}} f(\mathbf{x}) \geq f(\mathbf{x}^*) - 2\nu\rho.$$

Intuitively, $\mathcal{N}_h(2\nu\rho)$ represents the number of cells that any algorithm needs to sample in order to find the maximum. Thus, a small near-optimality dimension makes the function easier to optimize.

An upper bound on the simple regret of POO can also be given in terms of the near-optimality dimension.

Theorem 5. *At step n, for any (ν, ρ) that verifies the smoothness assumption for POO such that $\nu \leq \nu_{\max}$ and $\rho \leq \rho_{\max}$, there exists κ s.t.,*

$$\mathbb{E}[S_n] \leq \kappa \left(\frac{\ln^2 n}{n}\right)^{\frac{1}{d(\nu,\rho)+2}},$$

where $\nu_{\max}$ and $\rho_{\max}$ are two optional parameters for the POO algorithm that can be set automatically as functions of n.

2.2 Bayesian Optimization

As we already mentioned in the introduction that the objective of sequential global optimization is to approximate the optimum as quickly as possible. Bayesian optimization technique is another powerful tool to achieve this goal. It is called Bayesian optimization since it is derived from the famous "Bayes' theorem":

$$P[M|E] \propto P[E|M]P[M],$$

where M represents the model and E represents the given evidence. Hence, Bayesian optimization depends on some prior distribution over f, and in the next, we will examine a special case of the Bayesian optimization where the prior on f is a Gaussian process.

2.2.1 Gaussian Process Prior

Unlike in the HOO or POO settings, in the Bayesian optimization setting, we can make some implicit assumptions on the smoothness without explicit parametric assumptions. In a Gaussian process setting, we assume that the target function is a sample from a Gaussian process.

A Gaussian process $GP(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$ is characterized by its mean function $\mu(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$ and its covariance function $k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - \mu(\mathbf{x}))(f(\mathbf{x}') - \mu(\mathbf{x}'))]$. Note that the covariance function k can be replaced by a kernel function (constant, linear, Gaussian noise, Matérn, etc.), the only constraint in our case is that we enforce $k(\mathbf{x}, \mathbf{x}') \leq 1$. And in our case, we suppose that GPs are not conditioned on data, thus we can assume that $\mu \equiv 0$.

One main advantage of Gaussian process priors is that the posterior distribution has analytic expressions for mean and variance. Suppose that we use $GP(0, k(\mathbf{x}, \mathbf{x}'))$ as the prior distribution over the target function f. And we have a noisy sample $\mathbf{y}_n = [y_1, y_2, \dots, y_n]^T$ evaluated over points $A_n = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]^T$ with Gaussian noises, i.e.,

$$\forall t \in \{1, \dots, n\}, y_t = f(\mathbf{x}_t) + \epsilon_t,$$

where $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$ i.i.d., then the posterior distribution over f is still a Gaussian process distribution, and its mean, covariance and variance can be formulated as follows ,

$$\mu_n(\mathbf{x}) = \mathbf{k}_n(\mathbf{x})^T(\mathbf{K}_n + \sigma^2\mathbf{I})^{-1}\mathbf{y}^T,$$

$$k_n(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}') - \mathbf{k}_n(\mathbf{x})^T(\mathbf{K}_n + \sigma^2\mathbf{I})^{-1}\mathbf{k}_n(\mathbf{x}'),$$

$$\sigma_n^2(\mathbf{x}) = k_n(\mathbf{x}, \mathbf{x}),$$

where $\mathbf{k}_n(\mathbf{x}) = [k(\mathbf{x}_1, \mathbf{x}), \dots, k(\mathbf{x}_n, \mathbf{x})]^T$ and $\mathbf{K}_n = [k(\mathbf{x}, \mathbf{x}')]_{(\mathbf{x}, \mathbf{x}') \in A_n}$

2.2.2 Information Gain

In order to estimate globally the target function f as rapidly as possible, we need to choose carefully a set of samples $A \subset \mathcal{X}$. The way we measure the informativeness of a set of samples is the information gain which is the mutual information between f and observations $\mathbf{y}_A$,

$$I(\mathbf{y}_A; f) = H(\mathbf{y}_A) - H(\mathbf{y}_A|f).$$

Finding a subset $A \subset \mathcal{X}$ that maximizes the information gain is an NP-hard problem, however, it can be approximated by a greedy algorithm . Indeed, if we denote $F(A) = I(\mathbf{y}_A; f)$, then at step t of the algorithm, we can choose

$$\mathbf{x}_t = \arg\max_{\mathbf{x}\in\mathcal{X}} F(A_{t-1} \cup \{\mathbf{x}\}),$$

which can be shown to be equivalent to pick

$$\mathbf{x}_t = \arg\max_{\mathbf{x}\in\mathcal{X}} \sigma_{t-1}(\mathbf{x})).$$

And we can show that this greedy heuristic is guaranteed to find a near-optimal solution after n rounds,

$$F(A_n) \geq (1 - 1/e) \max_{|A|\leq n} F(A).$$

This is true mainly due to the fact that F satisfies a good property called *submodularity* Krause and Guestrin 2012. Indeed, this greedy approximation guarantee holds for any submodular function Nemhauser et al. 1978.

2.2.3 GP-UCB Algorithm and Regret Bound

In practice, it is wasteful to maximize the variance as it only concentrates on reducing the global uncertainty, but not exploiting knowledge around the maxima. Another idea is to choose points by maximizing the expected rewards, i.e. $\mathbf{x}_t = \arg\max_{\mathbf{x}\in\mathcal{X}} \mu_{t-1}(\mathbf{x})$. However, this may soon lead the algorithm to a local optimum.

A combination of these two strategies is proposed to overcome this dilemma:

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

where β_t are some well chosen constants. The GP-UCB algorithm Srinivas et al. 2009 is based on this strategy as shown below.

Algorithm 2 (GP-UCB)**.**
Input: *Input space $\mathcal{X}$; GP prior $\mu_0 = 0$, σ_0, k.*
Loop: *At time $n+1$, choose*

$$\boldsymbol{x}_{n+1} = \arg\max_{\boldsymbol{x}\in\mathcal{X}} \mu_n(\boldsymbol{x}) + \sqrt{\beta_{n+1}}\sigma_n(\boldsymbol{x}),$$

then sample

$$y_{n+1} = f(\boldsymbol{x}_{n+1}) + \epsilon_{n+1},$$

and perform Bayesian update to obtain σ_{n+1} and μ_{n+1}.

If we look closely to this algorithm, we can see that it implicitly trades off between exploration and exploitation. It explores by sampling $\mathbf{x}$ with large $\sigma_n^2(\mathbf{x})$ and it exploits by sampling $\mathbf{x}$ with large $\mu_n(\mathbf{x})$. Indeed, this GP-UCB selection is mainly motivated by the UCB algorithm, where $\mu_t(\mathbf{x}) + \sqrt{\beta_{t+1}}\sigma_t(\mathbf{x})$ is an upper confidence bound on $f(x)$.

Some cumulative regret bounds have been given for Gaussian process optimization, more precisely in the case when $\mathcal{X}$ is finite or compact and convex. Here, we will show the result for finite $\mathcal{X}$.

For that purpose, we need the notion of information gain as we defined before. We define the *maximum information gain* after n rounds as:

$$\gamma_n = \max_{A \subset \mathcal{X}, |A|=n} I(\mathbf{y}_A; f).$$

And we obtain the following bound for finite $\mathcal{X}$.

Theorem 6. *Let $\delta \in (0,1)$ and $\beta_t = 2\log(|D|t^2\pi^2/\delta)$. Running GP-UCB of a sample f of $GP(0, k(\mathbf{x}, \mathbf{x}'))$, we obtain a cumulative bound:*

$$R_n = \mathcal{O}^*(n\gamma_n \log |\mathcal{X}|)$$

with a high probability, i.e.,

$$P\{R_n \leq \sqrt{nC_1\beta_n\gamma_n}, \forall n \geq 1\} \geq 1 - \delta,$$

where $C_1 = 8/\log(1 + \sigma^{-2})$.

Remark 4. *Here regret bounds are given for the cumulative regret. This may seem to be somewhat weird since our objective is sequential global optimization. It may be more appropriate to focus on the simple regret.*

3 Applications to Hyperparameter Optimization for Machine Learning

One important application of the black box optimization or the multi-armed bandits model is the hyperparameter optimization in the context of machine learning. This task shows an increasingly importance as the quality of a machine learning model often depends strongly on its hyperparameter configuration.

A hyperparameter optimization problem can be considered as a pure exploration problem. We recall that in a pure exploration problem, the decision maker has to explore resources within a limited budget, e.g. within n rounds of exploration, then give a recommendation at the end. The quality of the decision will be evaluated only on this final recommendation. This corresponds exactly to the hyperparameter optimization situation where one hyperparameter configuration can be considered as an arm. Indeed, if we consider a function $f : \mathcal{X} \rightarrow \mathbb{R}$ and take $\mathcal{X}$ as a set of parameter configurations, we can then map one parameter configuration to its performance of the corresponding machine learning algorithm. If $|\mathcal{X}| < \infty$, we can refer to a bandit problem; otherwise, we can refer to a general sequential optimization problem.

We can cite numerous approaches that can be applied to solve this problem, such as grid search, Bayesian optimization, random search, gradient-based optimization, etc. Particularly, there exist several multi-armed bandits-inspired such as UCBE Audibert et al. 2010, UGap Gabillon et al. 2012, BayesUCB Kaufmann et al. 2012a, GP-UCB, Thompson sampling Kaufmann et al. 2012b, BayesGap Hoffman et al. 2014, etc.

Conclusion

In this bibliographic review, we studied the global optimization problem, or sometimes called the black box optimization problem, and its relationship with multi-armed bandits models. We talked about the difference between two different multi-armed bandits problems, i.e. best arm identification and regret minimization, and we concentrated especially on sequential global optimization. Two dominant approaches exist for this problem which are hierarchical optimistic optimization and Bayesian optimization. And we noticed that it seems to be particularly adapted to hyperparameter optimization in the context of machine learning.

A natural question one may ask is which one performs better, hierarchical optimistic optimization or Bayesian optimization? This can be a first task during the internship, that is to say, how can we find a way to compare these two techniques? And we are mainly interested in combining these two methods, see for example Contal et al. 2015.

The briefly summarize, the main objective of the internship is to propose new algorithms for sequential global optimization using hierarchical exploration with the help of best arm identification techniques. Then we will try to investigate their experimental performance as well as theoretical performance.

References

J.-Y. Audibert, S. Bubeck, and R. Munos. Best arm identification in multi-armed bandits. *In Proceedings of the 23th Annual Conference on Computational Learning Theory (COLT)*, 2010.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2):235–256, 2002.

E. Brochu, V. M. Cora, and N. de Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *Technical report, University of Bristish Columbia*, 2010.

S. Bubeck, R. Munos, G. Stoltz, and Szepesvári C. X-armed bandits. *Machine Learning Research*, 12:1587–1627, 2011.

E. Contal, C. Malherbe, and N. Vayatis. Optimization for gaussian processes via chaining. *In NIPS workshop on Bayesian Optimization*, 2015.

E. Even-Dar, S. Mannor, and Y. Mansour. Pac bounds for multi-armed bandit and markov decision processes. *In Proceedings of the 15th Annual Conference on Computational Learning Theory (COLT)*, pages 255–270, 2002.

C. A. Floudas and P. M. Pardalos. Optimization in computational chemistry and molecular biology: Local and global approaches. *Nonconvex Optimization and Its Applications. Springer*, 2000.

V. Gabillon, M. Ghavamzadeh, and A. Lazaric. Best arm identification: A unified approach to fixed budget and fixed confidence. *In Neural Information Processing Systems (NIPS)*, 2012.

J.-B. Grill, M. Valko, and R. Munos. Black-box optimization of noisy functions with unknown smoothness. *In Neural Information Processing Systems (NIPS)*, 2015.

W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

M. Hoffman, B. Shahriari, and N. de Freitas. On correlation and budget constraints in model-basedbandit optimization with application to automatic machine learning. *In Proceedings of the 17th International Conference on ArtiïňĄcial Intelligence and Statistics*, 2014.

S. Kalyanakrishnan, A. Tewari, P. Auer, and P. Stone. Pac subset selection in stochastic multi-armed bandits. *In International Conference on Machine Learning (ICML)*, 2012.

E. Kaufmann, O. Cappé, and A. Garivier. On bayesian upper conf. bounds for bandit problems. *In AIStats*, 2012a.

E. Kaufmann, N. Korda, and R. Munos. Thompson sampling: an asymptotically optimal finite-time analysis. *In International Conference on Algorithmic Learning Theory*, 2012b.

L. Kocsis and C. Szepesvári. Bandit based monte-carlo planning. *In Proceedings of the 17th European Conference on Machine Learning (ECML)*, pages 282–293, 2006.

A. Krause and C. E. Guestrin. Near-optimal nonmyopic value of information in graphical models. *arXiv:1207.1394*, 2012.

L. Li, K. Jamieson, G. A. Rostamizadeh, and A. Talwalkar. Efficient hyperparameter optimization and infinitely many armed bandits. *arXiv:1603.06560v1*, 2016.

S. Mannor and J. N. Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *Machine Learning Research*, 5:623–648, 2004.

C. G. Moles, P. Mendes, and J. R. Banga. Parameter estimation in biochemical pathways: a comparison of global optimization methods. *Genome Research*, 13(11):2467–2474, 2003.

G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of the approximations for maximizing submodular set functions. *Math. Prog.*, 14:265–294, 1978.

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T Graepel, and D Hassabis. Mastering the game of go with deep neuralnetworks and tree search. *Nature*, 529:484–489, 2016.

N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger. aussian process optimization in the bandit setting: No regret and experimental design. *arXiv:0912.3995*, 2009.

G. Wang and S. Shan. Review of metamodeling techniques in support of engineering design optimization. *Journal of Mechanical Design*, 129(4):370–380, 2007.

W. T. Ziemba and R. G. Vickson. Stochastic optimization models in finance. *World Scientific Singapore*, 2006.