Learning Maximin Strategies in Simulation-Based Games with Infinite Strategy Spaces

Alberto Marchesi, Francesco Trovò, Nicola Gatti
Politecnico di Milano
{alberto.marchesi,francesco1.trovo,nicola.gatti}@polimi.it

Abstract

We tackle the problem of learning equilibria in *simulation-based* games, where the players' utilities cannot be described analytically, but they are given through a black-box simulator that can be queried to obtain noisy estimates of the utilities. This is the case in many real-world games in which a complete description of the elements involved is not available upfront, such as, *e.g.*, complex military settings and online auctions. In these situations, one usually needs to run costly simulation processes to get an accurate estimate of the game outcome. As a result, solving these games begets the challenge of designing learning algorithms that can find (approximate) equilibria with high confidence, using as few simulator queries as possible. In this work, we focus on two-player zero-sum games with *infinite strategy spaces*. Drawing from the best arm identification literature, we design algorithms to learn *maximin* strategies in these games, both in the *fixed-confidence* setting and the *fixed-budget* ones. We formally prove δ-PAC theoretical guarantees for our algorithms, assuming that the utilities are drawn from a Gaussian Process.

1 Introduction

Most of the game-theoretic studies in AI focus on models where a complete description of the game is available, *i.e.*, the players' utilities can be expressed analytically. This is the case of large zero-sum recreational games such as Poker [1, 2], which are commonly used as benchmarks for evaluating algorithms to compute equilibria in games [3]. However, in many real-world problems, the players' utilities may *not* be readily available, as they are the outcome of a complex process governed by unknown parameters. This is the case in, *e.g.*, complex military settings where a comprehensive description of the environment and the units involved is not available, and online auctions in which the platform owner does not have complete knowledge of the parties involved. These scenarios can be addressed with *simulation-based games* (SBGs) [4], where the players' utilities are expressed by means of a black-box simulator that, given some players' strategies, can be queried to obtain a noisy estimate of the utilities obtained when playing such strategies. Solving these games calls for algorithms to learn (approximate) equilibria using as few queries as possible, since running the simulator is usually a costly operation. Recent works studying SBGs are only sporadic, addressing specific settings such as, *e.g.*, symmetric games with a large number of players [5, 6], empirical mechanism design [7], and two-player zero-sum finite games [8]. To the best of our knowledge, the majority of these works focus on the case in which each player has a finite number of strategies available. However, in most of the game settings in which simulations are involved, the players have an infinite number of choices available, *e.g.*, physical quantities, such as angle of movement and velocity of units on a military field, bids in auctions, and trajectories in robot planning. Dealing with infinite strategies leads to further challenges, since, being a complete exploration of the strategy space unfeasible, providing strong theoretical guarantees is in general a non-trivial task.

We study the problem of learning equilibria in *two-player zero-sum* SBGs with *infinite strategy spaces*. Specifically, we focus on *maximin* strategies for the first player, *i.e.*, those maximizing her utility

Abstract for: Smooth Games Optimization and Machine Learning Workshop (NeurIPS 2019), Vancouver, Canada.

under the assumption that the second player acts so as to minimize it, after observing the first player's course of play. When dealing with infinite strategy spaces, some regularities assumptions on the players' utilities are in order, since, otherwise, one cannot design learning algorithms with provable theoretical guarantees. In this work, we encode our smoothness assumptions on the utility function by modeling it as a sample from a *Gaussian process* (GP) [9]. We design two algorithms to learn (approximate) maximin strategies in two-player zero-sum SBGs with infinite strategy spaces, drawing from techniques used in the best arm identification literature. The first algorithm we propose, called M-GP-LUCB, is for the *fixed-confidence* setting, where the objective is to find an (approximate) maximin strategy with a given (high) confidence, using as few simulator queries as possible. Instead, the second algorithm, called SE-GP, is for the *fixed-budget* setting, in which a maximum number of queries is given in advance, and the task is to return an (approximate) maximin strategy with confidence as high as possible. First, we prove δ-PAC theoretical guarantees for our algorithms in the easiest setting in which the strategy spaces are finite. Then, we show how these results can be generalized to SBGs with infinite strategy spaces, by leveraging the GP assumption.

2 Preliminaries

A *two-player zero-sum game with infinite strategy spaces* is a tuple $\Gamma = (\mathcal{X}, \mathcal{Y}, u)$, where $\mathcal{X} \subset [0, 1]$ and $\mathcal{Y} \subset [0, 1]$ are closed intervals representing the sets of strategies available to the first and the second player, respectively, while $u : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$ is a function defining the utility for the first player. A *two-player zero-sum game with finite strategy spaces* is defined analogously, with $\mathcal{X}$ and $\mathcal{Y}$ being finite sets, *i.e.*, $\mathcal{X} := \{x^1, \ldots, x^n\}$ and $\mathcal{Y} := \{y^1, \ldots, y^m\}$. Letting $\Pi := \mathcal{X} \times \mathcal{Y}$, we denote with $\boldsymbol{\pi} := (x, y) \in \Pi$ a *strategy profile* specifying a strategy $x \in \mathcal{X}$ for the first player and a strategy $y \in \mathcal{Y}$ for the second one. We focus on the computation of *maximin* strategies for the first player, *i.e.*, those maximizing her utility assuming the opponent acts so as to minimize it after observing the first player's move. Formally, given $x \in \mathcal{X}$, we let $y^*(x) \in \arg\min_{y \in \mathcal{Y}} u(x, y)$ be a second player's best response to x. Then, $x^* \in \mathcal{X}$ is a maximin strategy if $x^* \in \arg\max_{x \in \mathcal{X}} u(x, y^*(x))$, with $\boldsymbol{\pi}^* := (x^*, y^*(x^*))$ denoting its corresponding maximin strategy profile.

Simulation-Based Games. In SBGs, the utility function u is not readily available, but it is rather specified by an exogenous simulator that provides noisy point estimates of it. Thus, the problem is to learn an (approximate) maximin strategy by sequentially querying the simulator. At each round t, the simulator is given a strategy profile $\boldsymbol{\pi}_t \in \Pi$ and returns an estimated utility $\tilde{u}_t := u(\boldsymbol{\pi}_t) + e_t$, where $e_t \sim \mathcal{N}(0, \lambda)$. The goal is to find a good approximation of a maximin strategy $x^* \in \mathcal{X}$ limiting the number of queries to the simulator. To achieve this, we propose some *dynamic querying algorithms*, which are generally characterized by the following components: a querying rule indicating which strategy profile $\boldsymbol{\pi}_t \in \Pi$ is queried at each round t; a stopping rule that determines the round T after which the algorithm terminates; and a final guess $\overline{\boldsymbol{\pi}} := (\bar{x}, \bar{y}) \in \Pi$ on the maximin strategy profile $\boldsymbol{\pi}^*$. Given an approximation $\epsilon \geq 0$, the objective of an algorithm is to find an ϵ-maximin strategy with a high accuracy, or, given $\delta \in (0, 1)$, design a δ-PAC algorithm, *i.e.*, satisfying:

$$\forall u \quad \mathbb{P}\Big(|u(\boldsymbol{\pi}^*) - u(\bar{x}, y^*(\bar{x}))| \leq \epsilon \Big) \geq 1 - \delta, \tag{1}$$

while keeping the number of rounds T as small as possible. This is known as the *fixed-confidence* setting. An alternative is to consider the *fixed-budget* case, where the maximum number of rounds T is given in advance, and the goal is to minimize the probability δ that $\bar{x}$ is not an ϵ-maximin strategy.

Gaussian Processes. To be able to design δ-PAC algorithms working with SBGs with infinite strategy spaces, we first need to introduce some regularity assumptions on the utility functions u. In this work, we model the utility as a sample from a Gaussian Process GP$(\mu(\boldsymbol{\pi}), k(\boldsymbol{\pi}, \boldsymbol{\pi}'))$ [9] over the profile $\boldsymbol{\pi} \in \Pi$, where $\mu : \Pi \mapsto \mathbb{R}$ is the *mean* function and $k : \Pi \times \Pi \mapsto \mathbb{R}$, is the *covariance* (or kernel) function (w.l.o.g., we assume that $k(\boldsymbol{\pi}, \boldsymbol{\pi}) := \sigma^2 \leq 1$ for every $\boldsymbol{\pi} \in \Pi$). Intuitively, the kernel function k determines the correlation of the utilities across the space of strategy profiles Π, thus encoding the smoothness properties of the utility functions u sampled from GP$(\mu(\boldsymbol{\pi}), k(\boldsymbol{\pi}, \boldsymbol{\pi}'))$. Our algorithms use GP$(\mathbf{0}, k(\boldsymbol{\pi}, \boldsymbol{\pi}'))$ as prior distribution over u. The advantage of working with GPs is that the posterior distribution is still a GP and admits simple formulas for its mean $\mu_t(\boldsymbol{\pi})$, covariance $k_t(\boldsymbol{\pi}, \boldsymbol{\pi}')$, and variance $\sigma_t^2(\boldsymbol{\pi})$. These relations are usually written using matrix notation [9], but they can also be expressed recursively, thus avoiding costly matrix inversions, as shown in [10].

3 Fixed-Confidence Setting

We propose a δ-PAC dynamic querying algorithm (called M-GP-LUCB) based on the M-LUCB approach introduced in [8] and provide a bound on the number of rounds T_δ it requires, as a function of the confidence level δ. For every strategy profile $\boldsymbol{\pi} \in \Pi$, the algorithm keeps track of a confidence interval $[L_t(\boldsymbol{\pi}), U_t(\boldsymbol{\pi})]$ on $u(\boldsymbol{\pi})$ built using the utility values $\tilde{u}_t$ observed from the simulator up to round t. Using GP$(\mathbf{0}, k(\boldsymbol{\pi}, \boldsymbol{\pi}'))$ as prior distribution over the utility function u, the bounds of the intervals are $L_t(\boldsymbol{\pi}) := \mu_t(\boldsymbol{\pi}) - \sqrt{b_t}\sigma_t(\boldsymbol{\pi})$ and $U_t(\boldsymbol{\pi}) := \mu_t(\boldsymbol{\pi}) + \sqrt{b_t}\sigma_t(\boldsymbol{\pi})$, where $\mu_t(\boldsymbol{\pi})$ and $\sigma_t^2(\boldsymbol{\pi})$ are the mean and the variance of the posterior distribution, while b_t is an exploration term that depends from the context. At the end of every even round t, the algorithm selects the strategy profiles to give as inputs to the simulator during the next two rounds. For every $x \in \mathcal{X}$, let $\gamma_t(x) := \operatorname{argmin}_{y \in \mathcal{Y}} L_t(x, y)$ be the second player's best response to x computed using the lower bounds L_t, and let $\bar{x}_t := \operatorname{argmax}_{x \in \mathcal{X}} \min_{y \in \mathcal{Y}} \mu_t(x, y)$ be the maximin strategy computed using the posterior mean μ_t. Then, in the following two rounds, the algorithm selects the strategy profiles $\boldsymbol{\pi}_{t+1} := (\bar{x}_t, \gamma_t(\bar{x}_t))$ and $\boldsymbol{\pi}_{t+2} := \operatorname{argmax}_{\boldsymbol{\pi} \in \{(x, \gamma_t(x))\}_{x \neq \bar{x}_t}} U_t(\boldsymbol{\pi})$. The M-GP-LUCB algorithm stops when it holds $L_t(\boldsymbol{\pi}_{t+1}) > U_t(\boldsymbol{\pi}_{t+2}) - \epsilon$. The final strategy profile recommended by the algorithm is $\overline{\boldsymbol{\pi}} := (\bar{x}_t, \gamma_t(\bar{x}_t))$. Theorem 1 shows that M-GP-LUCB is δ-PAC and provides an upper bound on the number of rounds T_δ it requires. The analysis is performed for $\epsilon = 0$, *i.e.*, with respect to an exact maximin profile. The upper bound for T_δ depends on the utility-dependent term $H^*(u)$, which, intuitively, measures the complexity of the problem instance [11].

Theorem 1. *Using a generic nondecreasing exploration term $b_t > 0$, the M-GP-LUCB algorithm stops its execution after at most T_δ rounds, where $T_\delta \leq \inf\left\{t \in \mathbb{N} \,:\, 8\,H^*(u)\,b_t\,\lambda - \frac{\lambda\,n\,m}{\sigma^2} < t\right\}$. Letting $b_t := 2\log\left(\frac{n\,m\,\pi^2\,t^2}{6\delta}\right)$, $\overline{\boldsymbol{\pi}}$ is a maximin profile with confidence at least $1 - \delta$, and:*

$$T_\delta \leq 64\,H^*(u)\,\lambda\left(\log\left(64\,H^*(u)\,\lambda\,\pi\sqrt{\frac{n\,m}{6\delta}}\right) + 2\log\left(\log\left(64\,H^*(u)\lambda\pi\sqrt{\frac{n\,m}{6\delta}}\right)\right)\right), \qquad (2)$$

where we require that $64\,\lambda\,\pi\,\sqrt{\frac{n\,m}{6\delta}} > 4.85$.

4 Fixed-Budget Setting

We propose a successive elimination algorithm (called GP-SE), which is based on an analogous method proposed in [11] for the best arm identification problem. The fundamental idea behind our GP-SE algorithm is a novel elimination rule, which is suitably defined for the problem of identifying maximin strategies. The algorithm works by splitting the number of available rounds T into $P - 1$ phases, where we let $P := |\Pi| = n\,m$ be the number of players' strategy profiles. At the end of each phase, the algorithm excludes from the set of candidate solutions the strategy profile that has the lowest chance of being maximin. Specifically, letting Π_p be the set of the remaining strategy profiles during phase p, at the end of p, the algorithm dismisses the strategy profile $\boldsymbol{\pi}_p = (x_p, y_p) \in \Pi_p$ such that $(x_p, \cdot) := \operatorname{argmin}_{\boldsymbol{\pi} \in \Pi_p} \mu_p(\boldsymbol{\pi})$ and $y_p := \operatorname{argmax}_{y \in \mathcal{Y} : (x_p, y) \in \Pi_p} \mu_p(x_p, y)$, where μ_p represents the mean of the posterior distribution computed at the end of phase p. Intuitively, the algorithm selects the first player's strategy x_p that is less likely to be a maximin one, together with the second player's strategy y_p that is the worst given x_p. At the end of the last phase, the (unique) remaining strategy profile $\overline{\boldsymbol{\pi}} = (\bar{x}, \bar{y})$ is recommended by the algorithm. Following [11], the length of the phases have been carefully chosen so as to obtain an optimal (up to a logarithmic factor) convergence rate. Specifically, letting $\overline{\log}(P) := \frac{1}{2} + \sum_{i=2}^{P} \frac{1}{i}$, we define $T_0 := 0$ and, for every $p \in \{1, \ldots, P-1\}$, $T_p := \left\lceil \frac{T-P}{\overline{\log}(P)(P+1-p)} \right\rceil$. Then, during each phase p, the algorithm selects every remaining strategy profile in Π_p for exactly $T_p - T_{p-1}$ rounds. We remark that the algorithm is guaranteed to do not exceed the number of available rounds T, as each $\boldsymbol{\pi}_p$ is selected for T_p rounds, while $\overline{\boldsymbol{\pi}}$ is chosen T_{P-1} times, and $\sum_{p=1}^{P-1} T_p + T_{P-1} \leq T$. Theorem 2 provides an upper bound on the probability δ_T that the strategy profile $\overline{\boldsymbol{\pi}}$ is not ϵ-maximin, as a function of the number of rounds T. As for the fixed-confidence setting, our result holds for the case in which $\epsilon = 0$.

Theorem 2. *Letting T be the number of available rounds, the GP-SE algorithm returns a maximin strategy profile $\boldsymbol{\pi}^*$ with confidence at least $1 - \delta_T$, where:*

$$\delta_T = 2P(n + m - 2)e^{-\frac{T-P}{8\lambda\overline{\log}(P)H_2}}, \qquad (3)$$

with $H_2 := \max_{i \in \{1,\ldots,P\}} i\Delta_{(i)}^{-2}$ and $\Delta_{(i)} := |u(\boldsymbol{\pi}^*) - u(\boldsymbol{\pi}^i)|$ so that $\Delta_{(1)} \leq \Delta_{(2)} \leq \ldots \leq \Delta_{(P)}$.

5 SBGs with Infinite Strategy Spaces

We show how the δ-PAC algorithms proposed in Sections 3 and 4 for finite SBGs can be adapted to work with infinite strategy spaces, also providing theoretical guarantees on the returned ϵ-maximin profiles. Our main result relies on our assumption that the utility function u is drawn from a GP, provided some mild technical requirements are satisfied (see Assumption 1). The idea is to work with a discretization of the players' strategy spaces, each made of at least K_ϵ equally spaced points, where $\epsilon \geq 0$ is the desired approximation level. This induces a new SBGs with finite strategy spaces, where techniques presented in the previous sections can be applied. Given an SBG with infinite strategy spaces Γ, we denote with $\Gamma(K)$ the finite SBG obtained when approximating the players' strategy spaces with K equally spaced points, *i.e.*, a game in which the players have $n = m = K$ strategies available and the utility value of each of the $n\,m$ strategy profiles is the same as that one of the corresponding strategy profile in Γ. First, let us introduce the following assumption.

Assumption 1 (Kernel Smoothness)**.** *A kernel $k(\boldsymbol{\pi}, \boldsymbol{\pi}')$ is said to be* smooth *over Π if, for each $L > 0$ and for some constants $a, b > 0$, the functions u drawn from GP$(\mathbf{0}, k(\boldsymbol{\pi}, \boldsymbol{\pi}'))$ satisfy:*

$$\mathbb{P}\left(\sup_{\boldsymbol{\pi}\in\Pi}\left|\frac{\partial u}{\partial x}\right| > L\right) \leq ae^{-\frac{L^2}{b^2}} \quad \text{and} \quad \mathbb{P}\left(\sup_{\boldsymbol{\pi}\in\Pi}\left|\frac{\partial u}{\partial y}\right| > L\right) \leq ae^{-\frac{L^2}{b^2}}. \tag{4}$$

This assumption is standard when using GPs in online optimization settings [12], and it is satisfied by many kernel functions for specific values of a and b, such as the squared exponential and the Matérn kernels. We are now ready to state our main result:

Theorem 3. *Assume that u is drawn from a GP$(\mathbf{0}, k(\boldsymbol{\pi}, \boldsymbol{\pi}'))$ satisfying Assumption 1. Given $\epsilon > 0$ and $\delta \in (0, 2)$, let $\overline{\boldsymbol{\pi}} = (\bar{x}, \bar{y}) \in \Pi$ be a maximin strategy profile for a finite game $\Gamma(K)$ where K is at least $K_\epsilon := \left\lceil \frac{b}{2\epsilon}\sqrt{\log\left(\frac{4a}{\delta}\right)}\, \right\rceil + 1$. Then, $\mathbb{P}\Big(\, |u(\boldsymbol{\pi}^*) - u(\overline{\boldsymbol{\pi}})| \leq \epsilon\Big) \geq 1 - \frac{\delta}{2}$.*

The following two results rely on Theorem 3 to show that the M-GP-LUCB and the GP-SE algorithms can find, with high confidence, ϵ-maximin strategy profiles in infinite SBGs.

Corollary 1. *Assume that u is drawn from a GP$(\mathbf{0}, k(\boldsymbol{\pi}, \boldsymbol{\pi}'))$ satisfying Assumption 1. Given $\epsilon > 0$ and $\delta \in (0, 1)$, letting $b_t := 2\log\left(\frac{nm\pi^2 t^2}{3\delta}\right)$ and $K_\epsilon := \left\lceil \frac{b}{2\epsilon}\sqrt{\log\left(\frac{4a}{\delta}\right)}\, \right\rceil + 1$, the M-GP-LUCB algorithm applied to $\Gamma(K)$ with $K \geq K_\epsilon$ returns a strategy profile $\overline{\boldsymbol{\pi}} = (\bar{x}, \bar{y})$ such that $\mathbb{P}\left(|u(\boldsymbol{\pi}^*) - u(\bar{x}, y^*(\bar{x}))| \leq \epsilon\right) \geq 1 - \delta$ after at most $T_{\delta,\epsilon}$ rounds, where:*

$$T_{\delta,\epsilon} \leq 64H^*(u)\lambda\left[\log\left(64H^*(u)\lambda\pi K_\epsilon\sqrt{\frac{1}{3\delta}}\right) + 2\log\left(\log\left(64H^*(u)\lambda\pi, K_\epsilon\sqrt{\frac{1}{3\delta}}\right)\right)\right], \tag{5}$$

where we require that $64\lambda\pi K_\epsilon\sqrt{\frac{1}{3\delta}} > 4.85$.

Corollary 2. *Assume that u is drawn from a GP$(\mathbf{0}, k(\boldsymbol{\pi}, \boldsymbol{\pi}'))$ satisfying Assumption 1. Given $\epsilon > 0$ and $\delta \in (0, 1)$, letting T be the number of available rounds and $K_\epsilon := \left\lceil \frac{b}{2\epsilon}\sqrt{\log\left(\frac{4a}{\delta}\right)}\, \right\rceil + 1$, the GP-SE algorithm applied to $\Gamma(K)$ with $K \geq K_\epsilon$ returns a profile $\overline{\boldsymbol{\pi}} = (\bar{x}, \bar{y})$ such that $\mathbb{P}\left(|u(\boldsymbol{\pi}^*) - u(\bar{x}, y^*(\bar{x}))| > \epsilon\right) < \delta_{T,\epsilon}$, where:*

$$\delta_{T,\epsilon} = 4K_\epsilon^2(K_\epsilon - 1)e^{-\frac{T - K_\epsilon^2}{8\lambda\overline{\log}(K_\epsilon^2)H_2}} + 2ae^{-\frac{b^2}{4\epsilon^2(K_\epsilon - 1)^2}}. \tag{6}$$

In the result of Corollary 2, the discretization parameter K_ϵ depends on a confidence level δ that has to be chosen in advance. Another possibility is to try to minimize the overall confidence $\delta_{T,\epsilon}$ by appropriately tuning the parameter δ. Formally, a valid confidence level can be defined as $\delta_{opt} = \inf\{\delta \in (0, 1)\ :\ \delta_{T,\epsilon}\}$, noticing that $\delta_{T,\epsilon}$ depends on δ also through the term K_ϵ. Unfortunately, this minimization problem does not admit a closed-form optimal solution. Nevertheless, we can compute an (approximate) optimal value for δ by employing numerical optimization methods [13].

6 Conclusion

We addressed the problem of learning *maximin* strategies in two-player zero-sum SBGs with *infinite strategy spaces*. To the best of our knowledge, we provided the first learning algorithms for infinite SBGs enjoying δ-PAC theoretical guarantees on the quality of the returned solutions. This significantly advances the current state of the art for SBGs, as dealing with infinite strategies paves the way to the application of such models in complex real-world settings. The fundamental ingredient of our results is the assumption that the utility functions are drawn from a GP, which allows us to encode function regularities without relying on specific parametric assumptions, such as, *e.g.*, linearity.

In future, we will extend our work along different directions, such as, *e.g.*, SBGs in which the players' strategy spaces are multi-dimensional, and empirical mechanism design problems [7].

References

[1] Noam Brown and Tuomas Sandholm. Superhuman ai for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2018.

[2] Noam Brown and Tuomas Sandholm. Superhuman ai for multiplayer poker. *Science*, 365(6456):885–890, 2019.

[3] N. Brown and T. Sandholm. Safe and nested subgame solving for imperfect-information games. In *Proceeding of the conderence on Neural Information Processing Systems (NIPS)*, pages 689–699, 2017.

[4] Y. Vorobeychik and M.P. Wellman. Strategic analysis with simulation-based games. In *Proceedings of the IEEE Winter Simulation Conference (WSC)*, pages 359–372, 2009.

[5] B. Wiedenbeck, F. Yang, and M.P. Wellman. A regression approach for modeling games with many symmetric players. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, pages 1266–1273, 2018.

[6] S. Sokota, C. Ho, and B. Wiedenbeck. Learning deviation payoffs in simulation-based games. In *AAAI Technical Track: Game Theory and Economic Paradigms*, 2019.

[7] E.A. Viqueira, C. Cousins, Y. Mohammad, and Greenwald. A. Empirical mechanism design: Designing mechanisms from data. In *Proceedings of the Conference on Uncertainty in Artificial (UAI)*, pages 1–11, 2019.

[8] A. Garivier, E. Kaufmann, and W.M. Koolen. Maximin action identification: A new bandit framework for games. In *Proceedings of the Conference On Learning Theory (COLT)*, pages 1028–1050, 2016.

[9] C.K.I. Williams and C.E. Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

[10] S.R. Chowdhury and A. Gopalan. On kernelized multi-armed bandits. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 844–853, 2017.

[11] J. Audibert, S. Bubeck, and R. Munos. Best arm identification in multi-armed bandits. In *Proceedings of the Conference On Learning Theory (COLT)*, pages 41–53, 2010.

[12] N. Srinivas, A. Krause, S.M. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1015–1022, 2010.

[13] J. Nocedal and S. Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[14] K.B. Petersen and M.S. Pedersen. The matrix cookbook. *Technical University of Denmark*, 7(15):1–25, 2008.

[15] E. Kaufmann, O. Cappé, and A. Garivier. On the complexity of best-arm identification in multi-armed bandit models. *J MACH LEARN RES*, 17(1):1–42, 2016.