To Massimo Natale,

who first introduced me to the world of bandits.

Ringraziamenti

Trarre le conclusioni di un percorso non è mai facile, soprattutto se questo segna l'inizio del resto della propria vita. Eppure, per quanto la fine dell'università possa essere difficile da processare a caldo nella mente di un ventiquattrenne, vale la pena prendere quest'occasione per guardarsi indietro una volta per tutte, prima di scegliere quale sentiero percorrere nel bosco intricato descritto da Robert Frost nella poesia che apre questa tesi. E d'altro canto (bandit algorithms a parte), la stessa *The Road Not Taken* non avrebbe mai visto la luce se Frost non avesse stretto un legame così forte con Edward Thomas, suo collega e amico, caduto tragicamente in battaglia durante la prima guerra mondiale. Forse è proprio questa la giusta chiave di lettura da dare alla poesia nel mio caso. Come non si riesce ad affrontare un semplice sentiero senza avere una spalla, un amico o un supporto su cui fare affidamento, così, in ambito universitario, risulta impossibile superare i numerosi ostacoli che continuano a presentarsi se non si è circondati dalle persone giuste. Per mia fortuna, queste persone sono molte, ognuna delle quali merita un doveroso ringraziamento.

La prima persona che voglio ringraziare è il professor Daniele Durante, il mio attento relatore, senza le cui intuizioni, scoperte e numerosissime correzioni questa tesi non sarebbe mai esistita. La sensazione a fine lavoro è di aver imparato moltissimo su come si ragiona nella ricerca in ambito statistico, e in generale su come vanno presentate le proprie idee. Oltre a Daniele, meritano il primo posto in questi ringraziamenti i professori Igor Pruenster e Antonio Lijoi, i miei mentori al BIDSA, vere e proprie guide in questi ultimi due anni e mezzo. Grazie soprattutto per la fiducia che mi avete dato e per avermi aperto la strada al di là dell'oceano.

In secondo luogo, questa tesi non sarebbe mai esistita se non avessi trascorso sei mesi della mia vita con il team di data scientists più figo d'Italia. Un grazie di cuore a Davide Consiglio e a tutto i team Advanced Analytics di Generali Italia per la fiducia e il supporto di questi mesi (e per avermi concesso di utilizzare tutte quelle ore sulla VM). Grazie a Massimo Natale (a cui questa tesi è dedicata), Federica Palorini, Isabella David, Stefano Frigerio, Alessandro Colombo, Valeria Danese, Andrea Giussani, Daniele

Maccagnola, Costanza Bossi, Giorgio Conte, Cesare Bassu, Damian Eduardo Taranto, Virginia Scarabattoli, Valeria Verzi, Diego Paladini, Andrea Lombardi, Andrea Tirrelli, Edoardo Legnani e, ultimo ma non meno importante, Antonio "il Principe" Bencini.

Tanto per rimanere in tema di poesie, se come diceva John Donne "no man is an island", di certo neanche uno studente può esserlo. E cosí, sento di dover ringraziare quelle persone con le quali ho condiviso di più noie, paure, gioie, dolori e tanta, tantissima ansia pre-esame, a cominciare da Claudia Marangon. Da *Dead Or Alive* di Tom Cruise in Rock of Ages a *My Life is Going On* de La Casa de Papel, dagli Avengers a Game of Thrones, da Diritto Privato a Macroeconomics, sempre insieme a tirare fuori l'uno il peggio dell'altro (o il meglio, a seconda dei punti di vista). Ti voglio bene. E poi ancora grazia a Leonardo D'Amico, Ludovica Ciasullo, Valentina Santeusanio, Chiara Gardenghi, Francesco Fabbri, Federica Ricci, Andrew Funk, Francesca Montrasio, Leonardo Stiz e Edoardo Cogliati. Grazie anche e soprattutto ai miei compagni d'ufficio, cioè Vincenzo Loffredo, Alice Cima, Giacomo Mantegazza e Pierfrancesco Mei Innocenti. Senza di voi tutto questo non avrebbe avuto lo stesso sapore.

Università a parte, a rendere speciali questi 5 lunghi anni è stata anche la mia vita parallela da artista. Senza l'energia del teatro, il calore, gli affetti e le esperienze al limite davanti al palcoscenico, non avrei avuto la stessa grinta nell'affrontare gli esami. Voglio ringraziare dunque tutta la Proxima Res, inclusi Carmelo Rifici e Omar Nedjari per avermi insegnato così tanto di questo mestiere (e per aver perdonato le tante assenze causate dallo studio). Inoltre, devo ringraziare in particolar modo il maestro Guido Buganza. Coi tuoi insegnamenti sono riuscito ad esporre alla Triennale di Milano partendo da zero. La pittura mi ha insegnato a trovare della poesia anche in un bagno, in una balena o nei cavalli di una giostra. È una dimensione nuova, che spero di abitare per il resto della mia vita. Grazie anche alle ragazze del corso per ogni martedì sera trascorso insieme.

Veniamo ora agli amici veri, quelli di vecchia data, quelli che ormai sono 10 anni almeno che prendiamo una birra assieme senza stancarci mai. Quelli che hanno sopportato per anni il mio dover stare a casa per preparare l'esame con tre settimane di anticipo. Grazie ai pochi prescelti, orgoglioso di essere quel +1. Grazie a Francesco Azzigana, Lorenzo

Pavone, Lorenzo Zambon, Francesco Bonafè e a Luca Brazzola. Grazie anche alle loro ragazze, soprattutto a Giulia Guidicini per essere una forza della natura. E poi, grazie a Martina Tremolada, la quale, spero, prenderà questa benedetta laurea prima del mio dottorato. Grazie anche a Costanza Ferrero, a Enea Malfatto, Alessandro Rampoldi e Edoardo Nicodemi. Un grazie speciale va infine a Lorenzo Diaferia, da 10 anni ormai la voce consigliera dall'altra parte del telefono, che sia per la versione di latino che per i dubbi sul mondo della consulenza. Grazie per le sciate, per le serate e per le numerose ore di studio assieme.

Ma una persona è niente senza una famiglia alle spalle. Il ringraziamento più grande per cui va a loro. Grazie a Mamma e a Papà, che non hanno mai smesso di lottare e sacrificarsi per me ed Elena. Grazie per avermi dato tutto senza aspettarvi niente in cambio (a parte una media alta e una borsa di studio per il master). Della miriade di regali che mi avete fatto, la Bocconi è quello più importante di tutti. Grazie poi a zia Grazia, a zio Nicola, a Nonno Pietro e Nonna Lina e a Michela. Grazie a mio cugino Giorgio e alle ore di cammino passate assieme, grazie ai miei cugini Tata e Marco, a Marta e Giulio, a Chicco e a Valentina. Grazie infine a mia sorella Elena, per avermi sopportato durante le mie giornate nervose di studio.

Infine, grazie a Laura Gorla, che da un'anno e mezzo ormai chiamo Amore. Grazie per esserci sempre, per sostenermi e per darmi ogni giorno la sicurezza di andare avanti. Quest' ultimo anno è stato il più bello della mia vita, e qualsiasi sfida il futuro ci riserverà, riusciremo ad affrontarla insieme. Lo devo a te, Amore Mio.

Contents

Two roads diverged in a yellow wood,
And sorry I could not travel both
And be one traveler, long I stood
And looked down one as far as I could
To where it bent in the undergrowth;

Then took the other, as just as fair,
And having perhaps the better claim,
Because it was grassy and wanted wear;
Though as for that the passing there
Had worn them really about the same,

And both that morning equally lay
In leaves no step had trodden black.
Oh, I kept the first for another day!
Yet knowing how way leads on to way,
I doubted if I should ever come back.

I shall be telling this with a sigh
Somewhere ages and ages hence:
Two roads diverged in a wood, and I -
I took the one less traveled by,
And that has made all the difference.

The Road Not Taken (1916)
Robert Frost

1 Introduction

In 1916, Robert Frost published one of his most famous poems, titled *The Road Not Taken* (Frost, 1916). It describes a man (the poet) in front of a yellow wood, who has to decide which road to take between two diverging ones. Even though he tries to look as further as possible, he cannot see where each of the two separate paths would eventually lead him. He then chooses to take one, and while hoping one day to return to that same wood and make a different choice, he doubts he will ever be able to come back: he had chosen a path and *that has made all the difference.*

It is known that Frost wrote this poem to gently poke his friend Edward Thomas, who would constantly regret not having taken a different road during his long walks with the poet in the english countryside. And yet, this apparently easy choice of whether to take

the left or the right path in a wood becomes a powerful metaphor for life itself. Seventeen years later, interestingly, William R. Thompson published an article on Biometrika, titled *On the likelihood that one unknown probability exceeds another in view of the evidence of two samples* (Thompson, 1933). The paper proposes a treatment allocation method which adapts the probability of assigning each treatment to a group of individuals based on its previously observed rates of success and failure. In other words, he suggested a sampling rule, which will later inherit his name (the so called Thompson Sampling): as the probability of success of a treatment in spite of the other increases, then the same treatment should be applied to a larger and larger fraction of individuals. This intuitive and yet powerful heuristic allows for a reduction of the regret associated to the assignment choice (eg. apply treatment 1 to a patient when treatment 2 would have worked) as the experiment proceeds over time. It almost seems that Thompson and Frost indirectly shared the same desire to learn which road was "best". However, the scientist attempts to propose a first solution to such a dilemma: keep on adjusting your expectations based on what you have observed in the past, and then follow the road that has a higher probability of ending wherever you believe to be optimal according to such expectations. Even though life is unpredictable (and, unfortunately, one), this should work as a rule of thumb to minimize the regret of the possible choices one can make, especially if such choices are irreversible. If the poet and his friend had known and had applied the theory behind bandit algorithms to their leisurely strolls, *The Road Not Taken* would have probably never seen the light as we know it, and humanity would have missed one of the most famous poems of all time.

While hinted in Thompson (1933), bandit algorithms[1] where first formally introduced by Bush and Mosteller (1953). The two researchers carried on two separate repeated experiments on mice and on humans to understand their learning behavior. In particular, some mice were put at the bottom of a T-shaped maze, and had to decide whether to go left or right to find food. In particular, food showed up at each endpoint of the T with a different fixed probability, so that mice had to estimate which direction was more

[1] Often referred as Multi-armed bandit algorithms in the literature.

likely to deliver their supper. Similarly, the researchers placed individuals in front of a specially designed slot machine endowed with two different arms. To echo the nickname of the classical Las Vegas slot machine (the one-armed "bandit" that steals your money), such a device was endearingly called "two-armed bandit". As in the T maze, each arm of the slot generated a positive reward with a certain probability when pulled, such that one had a higher rate of success than the other. After several trials of exploration, eventually every individual in the experiment understood which arm was the most rewarding one, and kept on pulling it for the remaining trials.

Nowadays, this simple framework can be extended to a variety of situations (e.g. recommender systems, online advertising and production choices) where the pool of arms among which to chose is potentially so huge and the available data are so immense that it becomes almost necessary to have automatic optimization methods at hand. For this reason, the literature on bandit algorithms is growing at an exponential rate (Lattimore and Szepesvari, 2018), thanks to the increasing computational capacity of modern computers[2]. This should come as no surprise to the business educated readers: testing and updating new marketing strategies is an absolute must for every company that wishes to sustain a desirable level of profitability over time. However, these policies need to be carefully designed; abrupt changes in the product features, untested modifications of the website front page or even ads not targeted to a specific subgroup of the customer base may exibith undetected high opportunity costs. In addition, developing efficient marketing solutions requires a huge amount of time and resources, to the point that companies may often decide to stick to the *status quo* and opt for more secure and well-established policies. This trade-off is also referred as the "exploration-exploitation" dilemma (Sutton and Barto, 1998). If exploiting the "best" available option is less costly and generates a higher profit in the short run, changes in consumers' taste or strengthening competitors may harm the effectiveness of such policy in the long run. On the other hand, exploring new solutions increases the regret in the short run, but it is necessary to maintain high

[2] Google scholar reports less then 1000 paper at the voice "bandit algorithm" for the period 2001-2005, 2700 for the period 2006-2010 and 7000 in 2011-2015 and from 2016 to today the number of papers published is around 5600.

margins later in time. Ideally, the most effective way to conduct a business is to strongly focus on exploitation, but still leave enough room for the exploration of new options. Bandit algorithms are, as of today, the best solution to the exploration-exploitation dilemma. As Whittle (1988) claims, *bandit problems embody in essential form a conflict evident in all human actions: information versus immediate payoff.* Their main advantage is that they are able to efficiently process information even when the available options among which to choose are large in number or deep in complexity. Moreover, they run automatically, thus allowing businessmen to save a great deal of time and resources that would otherwise be spent in the decision process.

This thesis has two main objectives. First, it introduces and reviews the main techniques that, over the years, have shaped the bandit literature. In particular, it focuses on those methodologies that have been fundamental for the overall development of the theory and its subsequent application. In doing so, we lay the fundamental focus on the case of the K-armed stochastic Bernoulli bandit. This choice is in line with the literature itself, for the simple reason that the majority of applications of bandit algorithms deals with binary distributed rewards (clicked - not clicked, bought - not bought, and so on). To allow for mathematical simplicity and aid a better intuition, we assume that the probability associated to the reward of every arm available in the pool is invariant over time. Dropping such an assumption will be the object of a future research work. Furthermore, to ensure a proper performance comparison, we centre our evaluation metric over the concept of *regret*, that is, the cumulative loss that is observed when the algorithm chooses a sub-optimal option. In particular, we compare all the algorithms by looking at the upper bound of their overall regret (whenever this is mathematically tractable), and by assessing at the average trend of the per-period regret achieved by each strategy under different simulated environments. Second, and most importantly, it expands the current literature on Bayesian bandit algorithms by applying the novel conjugacy property of the probit regression in Durante (2019) to the Fractional Factorial Thompson Sampling introduced by Scott (2010). In particular, if a normal prior is set over the parameters of the probit regression, the resulting posterior follows a unified skew-normal distribu-

tion (Arellano-Valle and Azzalini, 2006; Azzalini and Capitanio, 2014). Such a conjugacy property has, however, certain sampling issues which make its practical application impossible whenever the number of observations becomes high. To solve this problem, we present an resampling method based on a Sequential Monte Carlo technique.

The work is structured as follows: Section 2 frames the analysis by briefly introducing the bandit environment; Section 3 treats the fundamental frequentist strategies: the ε-greedy, the Softmax, and the UCB family; Section 4 goes back to Thompson idea and tackles the bandit environment from a Bayesian point of view. Section 5 presents our advances in the Fractional Factorial Thompson Sampling. Finally, Section 6 concludes[3].

[3] All the `Python` scripts to produce the pictures and the simulations in this thesis are available at the following GitHub repository: `https://github.com/alessandrozito/Bandit_Algorithms.git`

2 The Bandit Environment

A bandit problem is a sequential game between a *learner* and an *environment* (Lattimore and Szepesvari, 2018). The game is played $T \in \mathbb{N}^+$ rounds, where T is the so called *horizon*, and $\mathcal{A}_t$ is the set of actions the learner can take at every trial $t \leqslant T$. In the bandit literature, $\mathcal{A}_t$ is also referred as the set of arms the player can pull at t.[4] In general, the pool is invariant over time, that is $\mathcal{A}_t = \mathcal{A} = \{1 \dots K\}\ \forall t \in \{1, \dots, T\}$. Moreover, the general case assumes that $K < \infty$ (i.e. the number of arms is finite), even though there exists an efficient version of the bandit problem with infinite arms (Berry et al., 1997). We denote the generic element of $\mathcal{A}$ by a (an abbreviation for "arm"). At any given round t, the learner chooses an arm $a_t \in \mathcal{A}$ and observes both an outcome $y_t \in \Upsilon$ and an associated reward $r_{a_t,t} \equiv r(a_t, y_t)$, where $r : \mathcal{A} \times \Upsilon \to \mathbb{R}$ is the reward function. Note that the choice of arm a_t depends exclusively on the choice-reward history $\mathbb{H}_{t-1} = \{(a_1, r_{a_1,1}), \dots, (a_{t-1}, r_{a_{t-1},t-1})\}$. In other words, the learner cannot borrow information from the future, but decides how to proceed based on past observations only. In particular, he chooses action a_t based on a policy function $\pi : \mathbb{H}_{t-1} \to \mathcal{A}$, and learns the associated outcome according to the environment map $Z : \mathbb{H}_{t-1} \times \mathcal{A} \to \Upsilon$. The set of possible policies $\pi \in \Pi$ is called *competitor class*. The typical objective of the learner is to choose the sequence of arm pulls that lead him to the highest cumulative reward $\sum_{t=1}^{T} r_{a_t,t}$. Now, if the learner knew the environment function Z, the optimal sequence of actions would be easy to find. However, the fundamental challenge of the bandit problem is that the environment is unknown the player. All the learner knows is that $Z \in \mathcal{E}$, where $\mathcal{E}$ is called *environment class*. In general, the performance of a policy π given $\mathcal{E}$ is measured in terms of *regret*, that is, broadly speaking, the difference between the total expected cumulative reward obtained by applying policy π and the total expected cumulative reward that comes from the application of the "best" policy $\pi^* \in \Pi$.

In line with the majority of the literature on bandit algorithms, our environment of reference is a *stochastic Bernoulli bandit*, whose characteristics are detailed in definitions 1 and 2.

[4] Throughout the review, I will use the term arm, action and choice as synonyms.

Definition 1. *A K-armed stochastic bandit is a tuple of distributions* $\nu = (P_1, \dots, P_K)$ *such that* $\forall t \in \{1, \dots, T\}$*:*

(i) $r_{a_t,t} \mid \{\mathbb{H}_{t-1}, a_t\} \sim P_{a_t}$

(ii) $\forall a \in \{1, \dots, K\}$, $\mathbb{P}(a_t = a|\mathbb{H}_{t-1}) = \xi_t(\mathbb{H}_{t-1})$, *where* $\xi_1, \xi_2, \dots$ *is a sequence of functions* $\xi_t : \mathbb{H}_{t-1} \to [0, 1]$

In other words, a stochastic bandit is an environment where the rewards associated to each arm have a distribution that is independent of past observations and fixed over time (condition (i)) and the probability of selecting arm a at t does not depend on future values (condition (ii)). The mean and variance associated to the reward of the arm selected at time t are given by:

$$\begin{aligned} \text{mean:} \quad \mu_{a_t} &= \int_{-\infty}^{\infty} r_{a_t,t} dP_{a_t}(r_{a_t,t}) \\ \text{variance:} \quad \sigma^2_{a_t} &= \int_{-\infty}^{\infty} (r_{a_t,t} - \mu_{a_t})^2 dP_{a_t}(r_{a_t,t}). \end{aligned}$$

If in particular the rewards follow a Bernoulli distribution, then the environment is termed stochastic Bernoulli bandit.

Definition 2. *A K-armed stochastic bandit* $\nu = (P_a)_{a=1}^K$ *is a stochastic Bernoulli bandit if and only if* $r_{a_t,t} \sim Bernoulli(\mu_{a_t})$ *with* $\mu_{a_t} \in [0, 1]$ $\forall t \in \{1, \dots, T\}$ *and* $\forall a_t \in \{1, \dots, K\}$

In other words, stochastic Bernoulli bandits are characterized by a sequence $(\mu_a)_{a=1}^K$ where each component is the probability of arm a to return a reward equal to 1. Note that these probabilities remain constant over time for each arm. Moreover, the meaning of $r_{a_t,t} = 1$ varies depending on the application the bandit algorithm. For example, it can represent a click on an online banner, or a conversion registered by a website after the selection of arm a_t. In particular, in the first case a_t can capture the color of the banner, while in the second it can represent the choice of a particular font for the website's main page. As these probabilities are unknown, they need to be estimated. However, the purpose of the algorithm is not the one of precisely estimating the probability of success

of every arm, but rather the one of understanding which of them leads to the highest expected reward, or, equivalently, to the lowest regret. In particular, we denote by

$$\hat{n}_a(t) := \sum_{s=1}^{t-1} \mathbb{1}\{a_s = a\} \tag{2.1}$$

the number of times arm a has been selected up to time t, and by

$$\hat{\mu}_a(t) := \frac{1}{\hat{n}_a(t)} \sum_{s=1}^{t-1} r_{a_s,s} \mathbb{1}\{a_s = a\} \tag{2.2}$$

the average observed reward for every arm a up to t. Note that the summation goes from 1 to $t-1$. This is a pure choice of notation: whenever we fix a generic time period t, we mean that the algorithm has collected information for the first $t-1$ trials, and needs to choose which arm to pull at t. For example, if $t = 1$, $\hat{n}_a(1) = 0$ for every arm, and no estimate for μ_a has been computed yet.

Call $\mu^* = \arg\max_{a \in \mathcal{A}} \mu_a$ the highest mean, and a^* the corresponding arm. The regret associated to policy $\pi = \{a_1, \ldots, a_T\}$ in bandit ν is defined as

$$R_T(\pi, \nu) = T\mu^* - \mathbb{E}\left[\sum_{t=1}^{T} r_{a_t,t} \right], \tag{2.3}$$

while the *sub-optimality gap* of action a with respect to the best arm a^* is defined as

$$\forall a \in \mathcal{A} : \Delta_{a^*,a} = \mu^* - \mu_a. \tag{2.4}$$

Note that, by definition, $\Delta_{a^*,a^*} = 0$. But then, the following lemma holds:

Lemma 1. *Regret Decomposition Lemma - For any policy π and K-armed stochastic bandit environment ν with horizon T, the regret can be written as*

$$R_T(\nu, \pi) = \sum_{a=1}^{K} \Delta_{a^*,a} \mathbb{E}[\hat{n}_a(T+1)], \tag{2.5}$$

where $\hat{n}_a(T+1)$ indicates the number of times arm a has been selected once the experiment is concluded.

Finally, we define the *per-period regret* of selecting arm a_t at any given time t as

$$\rho_{a*,a_t} = \mu^* - \mu_{a_t}, \tag{2.6}$$

with $\rho_{a*,a*} = 0$ by definition. The link between equation (2.4) and (2.6) is rather subtle. While $\Delta_{a*,a}$ refers to all arms and is invariant over time, ρ_{a*,a_t} instead is specific to a time period t where the optimal arm is a^* and the chosen arm is a_t. In other words, the per-period regret can be rewritten as

$$\rho_{a*,a_t} = \sum_{a=1}^{K} \Delta_{a*,a} \mathbb{1}\{a_t = a\}. \tag{2.7}$$

Note that we focus our analysis on the concept of regret rather then on the per-period reward or on the cumulative reward. While the minimization of the former mirrors a maximization of the latter two, the regret is a better term of comparison for the efficiency of an algorithm[5].

Indeed, much of the literature on bandit algorithms is concerned with the evaluation of the dependency between either the regret or the per-period regret at time t and t itself. As a rule of thumb, if the cumulative regret increases linearly with time or the per-period regret does not eventually become null, we should change our sequential allocation policy, because we keep on selecting sub-optimal arms even when the best arm has been spotted with enough confidence. Ideally, an upper bound on the regret that evolves according to a logarithmic scale should be preferred. Lai and Robbins (1985) introduced a primer result on the asymptotic bound of the regret R_T for a generic allocation policy, given than certain mild conditions are satisfied. In particular, let

$$D_{KL}(P_{a_t} \mid\mid P^*) = \int_{-\infty}^{\infty} p_{a_t}(r_{a_t,t}) \log\left(\frac{p_{a_t}(r_{a_t,t})}{p^*(r_{a_t,t})}\right) dr_{a_t,t} \ \in \ [0, \infty] \tag{2.8}$$

[5] Per-period rewards depend on the distribution from which the responses of each arm are generated. This means that the value to which they converge depends on the specific simulation. On the other hand, a per-period regret that is not equal or close to zero always indicate that the algorithm is selecting the wrong arm irrespectively of the bandit environment. In other words, the per-period regret is an absolute measure of performance.

denote the Kullback-Leibler divergence between the distribution function associated to the reward of arm a_t, P_{a_t}, and the one associated to the best arm, P^* (i.e. the one for which $\mu_{a_t} = \mu^*$). In addition, assume that

$$\forall a_t \in \mathcal{A} \text{ s.t. } \mu^* > \mu_{a_t} : \quad 0 < D_{KL}(P_{a_t} \,||\, P^*) < \infty \tag{2.9}$$

and that

$$\forall \mu_i, \mu_j, i, j \in \{1, \ldots, K\} : \quad \mu_i \to \mu_j \implies D_{KL}(P_i \,||\, P^*) \to D_{KL}(P_j \,||\, P^*). \tag{2.10}$$

Then, the the following theorem holds:

Theorem 1. *Lai and Robbins (1985) - Let $D_{KL}(P_a \,||\, P^*)$ satisfy conditions (2.9) and (2.10). If*

$$\forall \alpha > 0 : \lim_{T \to \infty} \frac{R_T}{T^\alpha} = 0, \tag{2.11}$$

then for every ν such that the success probabilities $(\mu_a)_{a \in \mathcal{A}}$ are not all equal,

$$\liminf_{T \to \infty} \frac{R_T}{\log T} \geqslant \sum_{a : \mu_a < \mu^*} \frac{\mu^* - \mu_a}{D_{KL}(P_a \,||\, P^*)} \tag{2.12}$$

In other words, $R_T = \Omega(\log T)$[6], which means that the regret attains an asymptotic lower bound that is at least on the logarithmic scale. An algorithm solves the multi-armed bandit environment if $R_T = O(\log T)$ (Kuleshov and Precup, 2000). In this case, the algorithm is said to be *asymptotically optimal*. In the following sections, we will introduce a series of algorithms that attain this upper bound, both asymptotically and

[6] For the rest of the paper, we adopt the Bachmann-Landau notation to indicate limiting properties of function. In particular, given the functions $f, g : \mathbb{N} \to [0, \infty)$:

$$f(n) = O(g(n)) \iff \limsup_{n \to \infty} \frac{f(n)}{g(n)} < \infty$$
$$f(n) = \Omega(g(n)) \iff \limsup_{n \to \infty} \frac{f(n)}{g(n)} > 0$$
$$f(n) = o(g(n)) \iff \lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$$
$$f(n) = \Theta(g(n)) \iff f(n) = O(g(n)) \text{ and } f(n) = \Omega(g(n))$$

not.

3 Frequentist Strategies

This section presents the fundamental algorithms that have shaped the bandit theory from a frequentist point of view: the *greedy* strategies and the *upper confidence bound strategies*. In line with the existing literature, much of the attention is dedicated to the comparison of the performance of the algorithms in terms of per-period regret in a simulated environment. As this thesis is fundamentally focused on extending the Thompson Sampling literature, this section serves mainly as an introduction to the world of bandits, rather than a detailed analysis of the most recent improvements of their frequentist literature. The chapter is divided as follows. Section 3.1 deals with the most common greedy strategies: ε-greedy and Softmax exploration. Section 3.2 introduces the concept of sub-gaussian random variables and uses it to explain the concepts behind the UCB strategies.

3.1 Greedy strategies

The aim of every bandit algorithm is to understand which arm is the best among the ones in $\mathcal{A}$ as quickly as possible. However, trying them all repeatedly before deciding may lead us to play sub-optimal arms more than it is necessary. On the other hand, complete exploitation (i.e. keep on exploiting the arm with the highest estimated reward at t) may prevent us from discovering new and better arms. Every policy π that exploits the arm with the highest estimated reward at time t is called a *greedy* strategy. While the aim of a greedy strategy is to maximize the immediate reward, it proves sub-optimal in the long run (Sutton and Barto, 1998): dull exploitation does not monitor possible changes of the bandit environment (like a sudden variation of the success probabilities of each arm).

The first solution to solve the inefficiencies of such a strategy is to introduce an element of randomness in the model for the arm selection: at the beginning of the experiment, fix a probability ε of exploring one arm in $\mathcal{A}$ at random. Then, at every trial, the algorithm takes the greedy action (i.e. pull the arms with the highest estimated reward up to t, with ties broken arbitrarily) with a probability of $1 - \varepsilon$, and chooses a random arm with probability ε. This algorithm is called ε-greedy (Sutton and Barto, 1998). Algorithm 1

reports the technical details of the ε-greedy applied to the case of a K-armed stochastic Bernoulli bandit.

Algorithm 1 ε-greedy algorithm with Bernoulli rewards

Input: $\varepsilon \in [0,1]$ ▷ Set the degree of exploration
Input: $(\hat{n}_a(1))_{a=1}^K \leftarrow \mathbf{0} \in \mathbb{R}^K$ ▷ Pulls counter
Input: $(\hat{\mu}_a(1))_{a=1}^K \leftarrow \mathbf{0} \in \mathbb{R}^K$ ▷ Vector of estimated means
for $t = 1, 2, 3 \ldots, T$ **do**
    Draw $u \sim \mathcal{U}(0,1)$
    if $u > \varepsilon$ **then**
        Select arm $a_t = \arg\max_{a \in \mathcal{A}} \hat{\mu}_a(t)$ ▷ Exploit w.p. $1 - \varepsilon$
    else
        Select arm a_t at random from $\mathcal{A}$ ▷ Explore w.p. ε
    Observe reward $r_{a_t,t}$ ▷ $r_{a_t,t} \sim Bernoulli(\mu_{a_t})$
    $\hat{n}_{a_t}(t+1) \leftarrow \hat{n}_{a_t}(t) + 1$
    $\hat{\mu}_{a_t}(t+1) \leftarrow [(\hat{n}_{a_t}(t+1) - 1)\hat{\mu}_{a_t}(t) + r_{a_t,t}]/\hat{n}_{a_t}(t+1)$ ▷ Update
end for

The mechanism is rather easy: at each time period t, it draws a random number u from a uniform distribution between 0 and 1, and compares it with ε. If $\varepsilon < u$, exploit the best arm (i.e. the one with the highest $\hat{\mu}_a(t)$). Otherwise, explore one arm at random.[7] Note that if the algorithm explores, then every arm in $\mathcal{A}$ has an equal probability of being selected. This means that it can happen that the learner takes the greedy action even during exploration. After the toss and the selection of arm a_t, the algorithm observes the associated reward $r_{a_t,t}$. Following our assumptions, such a reward follows a Bernoulli distribution which returns a success (a 1) with unknown probability μ_{a_t}, and a failure (a 0) otherwise. This allows to update the estimate for μ_{a_t} itself (that is, $\hat{\mu}_{a_t}(t)$), thus increasing the amount of information the algorithm has over the environment. In particular, the method used here for updating the value of an arm is the so called *action-value* method, which estimates the mean reward of each arm by averaging the number of successes over the number of pulls the arm has received. This method gives higher weight to the observations in the first trial periods, and progressively lower weight

[7] This is equivalent to selecting the best arm with probability $1 - \varepsilon$ and exploring with probability ε. To see this, let $Y \sim Bernoulli(\varepsilon)$ and $u \sim \mathcal{U}(0,1)$. But then, it holds that

$$\mathbb{P}(u \leqslant \varepsilon) = \int_{-\infty}^{\varepsilon} \mathbb{1}_{[0;1]}(u) du = \Big[u\Big]_0^{\varepsilon} = \varepsilon = \mathbb{P}(Y = 1)$$

to new observations as t increases. Note that this updating rule is efficient if and only if $(\mu_a)_{a=1}^K$ is invariant over time. If this assumption drops, the algorithm's ability to monitor potential changes in the environment decreases as the experiment proceeds. Then, the learner should consider other updating rules to avoid the risk of getting stuck to sub-optimal arms. More in general, the updating rule for the transition between t and $t+1$ for every arm a is given by:

$$\hat{\mu}_a(t+1) \leftarrow \begin{cases} \hat{\mu}_a(t) + \alpha(t, \hat{n}_a(t+1)) \times \left[r_{a_t,t} - \hat{\mu}_a(t)\right] & \text{if } a_t = a \\ \hat{\mu}_a(t) & \text{otherwise} \end{cases}$$

where $\alpha(t, \hat{n}_a(t+1))$ measures the weight to give at time t to the reward of a after a has been pulled $\hat{n}_a(t+1)$ times[8]. Notice that only the estimated mean of the arm selected at t receives the update, whereas the values for all the other arms $a \neq a_t$ remain equal. In particular, the action-values method sets $\alpha(t, \hat{n}_a(t+1)) = 1/\hat{n}_a(t+1)$. If instead we suspect that the probability of success of an arm can vary over time, we could set $\alpha(t, \hat{n}_a(t+1)) = \alpha \in [0,1]\ \forall t$. Even though this favors exploration over exploitation, the α level needs to be tuned properly, thus adding extra computational effort to the problem. Moreover, a high α may significantly decrease the performance of the algorithm irrespective of the environment.

Figure 1 plots the average per-period regret of algorithm 1 over 500 simulations with an horizon $T = 1000$ for five different values of ε. In particular, the left panel simulates over a 10 arms test-bed, while the right one over a 30 one. In both cases, the probability of success of each arm have been randomly drawn from a $Beta(1,6)$ at the beginning of each simulation, so to average the results across 500 different bandits environments. Two important aspects are worth mentioning. First, it is clear how different degrees of exploration lead to different levels of regret in the long run. In particular, as ε increases, so does the probability of pulling sub-optimal arms, irrespective of whether the algorithm has correctly identified the best arm in the environment. On the other hand, if ε is too low,

[8] The $t+1$ component comes from the fact that at time t arm a has been selected.

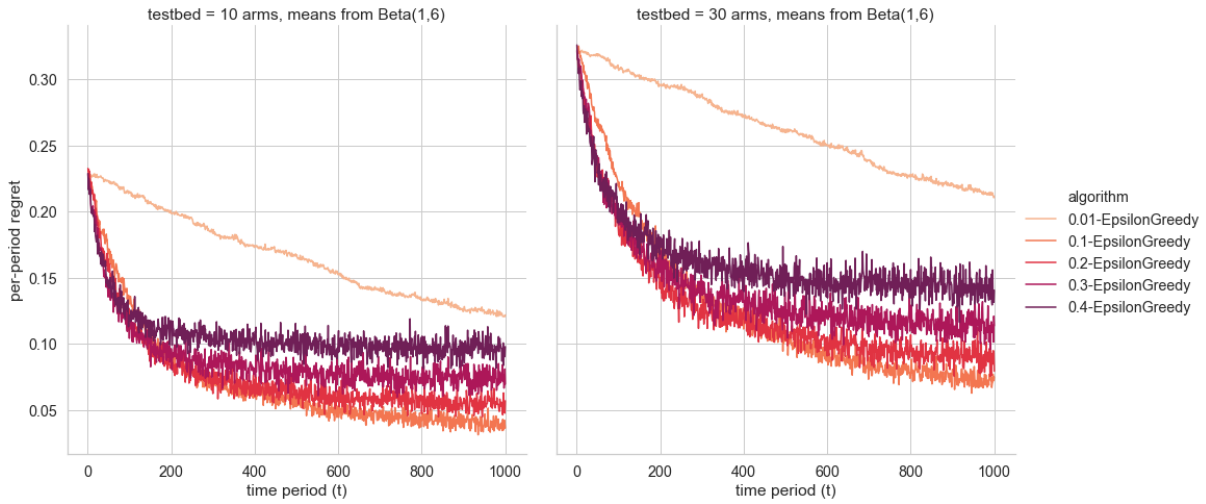


Figure 1: Average per-period regret of the ε-greedy algorithm across 500 simulations and varying ε in a 10-arms (left) and a 30-arms (right) stochastic Bernoulli bandit environment. Real success probabilities are randomly drawn from a $Beta(1, 6)$ at the beginning of each simulation.

the regret becomes very high, because the algorithm has really low chances of exploring all arms satisficingly. In other words, too much exploration and too much exploitation both increase the frequency with which sub-optimal choices are pulled. Indeed, the optimal level of exploration (i.e. the exact level of ε that minimizes the regret) is strictly dependent on the environment and needs to be tuned if one wishes to implement algorithm 1 in a real business application. Notice in particular that the performance of the algorithm decreases as the size of the arm pool increases, even though an $\varepsilon = 0.1$ achieves always the best performance. The second fundamental aspect is that average per period regret never quite goes to 0, but becomes stable after a certain time t depending on the size of ε. This means that when the ε-greedy algorithm understands what arm is the best, it is still enforced to explore sub-optimal arms. Moreover, the random design of the exploration rule can be problematic as it assigns each arm an equal probability of being explored, irrespective of its estimated success probability.

Ideally, we would like to have an algorithm that preserves the information collected on the exploited arms, so that the exploration returns at least a minimum acceptable reward. Rather than an *haphzard exploration*, what we might want is a more *structured exploration* (White, 2013). These two order of problems can be easily solved by modifying algorithm 1 by either making ε decrease over time (the so called *annealing* ε-greedy, algorithm 2), or by

selecting the arms according to a discrete probability distribution where the probability of pulling arm a at t is proportional to the estimated success of a itself (the *Softmax* algorithm, also referred as *Boltzmann exploration*).

Algorithm 2 Annealing ε-greedy algorithm with Bernoulli rewards

1: Input: $(\hat{n}_a(1))_{a=1}^K \leftarrow \mathbf{0}_K$ ▷ Pulls counter
2: Input: $(\hat{\mu}_a(1))_{a=1}^K \leftarrow \mathbf{0}_K$ ▷ Vector of estimated means
3: **for** $t = 1, 2, 3 \ldots, T$ **do**
4: $\quad \varepsilon \leftarrow g_\varepsilon(t)$ ▷ $g'_\varepsilon(t) \leqslant 0$, decreasing function over time
5: $\quad$ Perform steps 5-12 in algorithm 1
6: **end for**

The reasoning behind the annealing version of the ε-greedy is straightforward. During the first trials, the exploration rate is set high so to get a rough estimate of how each arm behaves when selected. Then, as t increases, ε decreases, which make the probability of selecting sub-optimal arms quickly reaching zero as the experiment proceeds. Cesa-Bianchi and Fischer (1998) proved that when

$$g_\varepsilon(t) = \min\left\{1, \frac{cK}{d^2 t}\right\}$$

with $d \in (0, \min_{a:\mu_a<\mu^*} \Delta_{a^*,a}]$ and $\mu_a \in [0,1]$ for every a, the annealing ε-greedy selects a sub-optimal arm with probability at most

$$\frac{c}{d^2 t} + 2\Big(\frac{c}{d^2}\log\frac{(t-1)d^2\sqrt{e}}{cK}\Big)\Big(\frac{cK}{(t-1)d^2\sqrt{e}}\Big)^{c/(5d^2)} + \frac{4e}{d^2}\Big(\frac{cK}{(t-1)d^2\sqrt{e}}\Big)^{c/2} \tag{3.1}$$

for all $t \geqslant cK/d$, with e indicating the Euler's constant. The authors also point out that, for $c > 5$, the bound is of order $\frac{c}{d^2 t} + o(\frac{1}{t})$ for $t \to \infty$, while the second and third terms' bounds are $O(\frac{1}{t^{1+\eta}})$ for some η. Even though this is a strong and non asymptotic result on the per-period regret, the exact calculation of this bound requires a prior knowledge of both the range of μ_a and of $\Delta_{a^*,a}$. More in general, they proved that the instantaneous regret bound on the finite-time multi-armed bandit problem in the case of both the ε-greedy and the Softmax is, under moderate assumptions, of the form $a + b\log T + \log^2 T$, where a, b and c are constants that do not depend on T.

However, as one can easily notice, the annealing ε-greedy suffers from the same fun-

damental problem as the simple ε-greedy does. Whenever it explores, every arm has an equal probability of being selected irrespective of its reward history. The Softmax algorithm (and its annealing version) elegantly solves this issue by choosing an arm at every round using a random draw form the Boltzmann distribution. Let $(\hat{\mu}_a(t))_{a=1}^K$ be the vector of estimated means at time t. Then, each arm receives a pull with a probability given by:

$$\hat{w}_{at} = \mathbb{P}(a_t = a) = \frac{e^{\hat{\mu}_a(t)/\tau}}{\sum_{j=1}^K e^{\hat{\mu}_j(t)/\tau}} \tag{3.2}$$

where $\tau \in [0, \infty)$ is a fixed parameter called *temperature*. The idea behind τ is pretty simple: the closer it is to 0, the less the algorithm tends to explore. On the other hand, a very high τ leads to a reduced exploitation rate.[9] To see this, consider the case where $K = 3$ and $\hat{\mu}_1(t) > \hat{\mu}_j(t), j = 2, 3$. Then, the probability of selecting arm 1 at time t is given by:

$$\begin{aligned}\tilde{w}_{1t} &= \frac{e^{\hat{\mu}_1(t)/\tau}}{\sum_{j=1}^K e^{\hat{\mu}_j(t)/\tau}} \\ &= \frac{1}{1 + e^{\frac{1}{\tau}(\hat{\mu}_2(t) - \hat{\mu}_1(t))} + e^{\frac{1}{\tau}(\hat{\mu}_3(t) - \hat{\mu}_1(t))}},\end{aligned}$$

which is equal to $\frac{1}{3}$ if $\tau \to \infty$ (unweighted random exploration) and tends to 1 if $\tau \to 0$ (pure exploitation). For values between $(0, \infty)$, arm 1 has higher chances of being pulled the lower τ is. In the annealing version of the Softmax algorithm, τ starts high and decreases over time, so to favor exploration during the first trials and exploitation in the last ones. Algorithm 3 and 4 provide a detailed description of the Boltzmann exploration with fixed and decreasing values for τ.

Figure 2 compares the simulated per-period regret of algorithms 1, 2, 3 and 4. As in the case of figure 1, the simulation is carried over a 10 and a 30 arms test-bed, each having a different probability of yielding $r_{a_t,t} = 1$ randomly drawn from a $Beta(1, 6)$. Again, the initial value of the regret is higher in the case of 30 arms, and decreases at a similar pace. Few things are worth noticing. First, the relative performances of the

[9] This behavior is similar the one displayed by the water particles as the temperature of the environment varies. A low temperature makes the particles stable, while at high degrees they tend to move more chaotically.

Algorithm 3 Softmax algorithm with Bernoulli rewards

1: Input: $\tau \geqslant 0$ ▷ Set the degree of exploration
2: Input: $(\hat{n}_a(1))_{a=1}^K \leftarrow \mathbf{0} \in \mathbb{R}^K$ ▷ Pulls counter
3: Input: $(\hat{\mu}_a(1))_{a=1}^K \leftarrow \mathbf{0} \in \mathbb{R}^K$ ▷ Vector of estimated means
4: **for** $t = 1, 2, 3 \ldots, T$ **do**
5: **for all** $a \in \mathcal{A}$ **do**
6: $\hat{w}_{at} \leftarrow \frac{e^{\hat{\mu}_a(t)/\tau}}{\sum_{j=1}^K e^{\hat{\mu}_j(t)/\tau}}$
7: **end for**
8: Select $a_t \in \mathcal{A}$ from the probability distribution $\hat{\mathbf{w}}_t = (\hat{w}_{at})_{a=1}^K$
9: Observe reward $r_{a_t,t}$ ▷ $r_{a_t,t} \sim Bernoulli(\mu_{a_t})$
10: $\hat{n}_{a_t}(t+1) \leftarrow \hat{n}_{a_t}(t) + 1$
11: $\hat{\mu}_{a_t}(t+1) \leftarrow [(\hat{n}_{a_t}(t+1) - 1)\hat{\mu}_{a_t}(t) + r_{a_t,t}]/\hat{n}_{a_t}(t+1)$ ▷ Update
12: **end for**

Algorithm 4 Annealing Softmax algorithm with Bernoulli rewards

1: Input: $(\hat{n}_a(1))_{a=1}^K \leftarrow \mathbf{0} \in \mathbb{R}^K$ ▷ Pulls counter
2: Input: $(\hat{\mu}_a(1))_{a=1}^K \leftarrow \mathbf{0} \in \mathbb{R}^K$ ▷ Vector of estimated means
3: **for** $t = 1, 2, 3 \ldots, T$ **do**
4: $\tau \leftarrow g_\tau(t)$ ▷ $g'_\tau(t) \leqslant 0$, decreasing function over time
5: Perform steps 5-11 in algorithm 3
6: **end for**

algorithms are similar: the annealing strategies correctly identify the best arm faster than the non annealing strategies. In particular, the worst algorithm in the long run is the simple Softmax with $\tau = 0.1$, but in the short run (the first 200 trials on the left, 400 on the right), its performance is superior to the simple 0.1-greedy. However, after this cutoff, the per-period regret of the Softmax becomes stationary. The reason is that, again, once the algorithm has correctly estimated the rates of success of all the arms, it is still forced to explore sub-optimal choices at a constant rate. Moreover, as the true success probabilities are similar[10], the corresponding selection probabilities will remain similar, thus making the algorithm unable to properly exploit. Second, the potential inefficiencies of the non-annealing algorithms are overcome once we set ε and τ to decrease over time. This is especially true in the case of the annealing Softmax, where a decreasing τ ensures that enough exploration takes place within the first trial so to justify pure exploitation in later ones. Third, the per-period regret of the algorithms never quite goes to zero, as the randomness of the selection leaves always some space for further exploration. In the next

[10] The $Beta(1, 6)$ distribution has mean around 0.14 and variance 0.015.

subsection, we analyze a class of algorithms which is purely deterministic and potentially can overcome this problem.

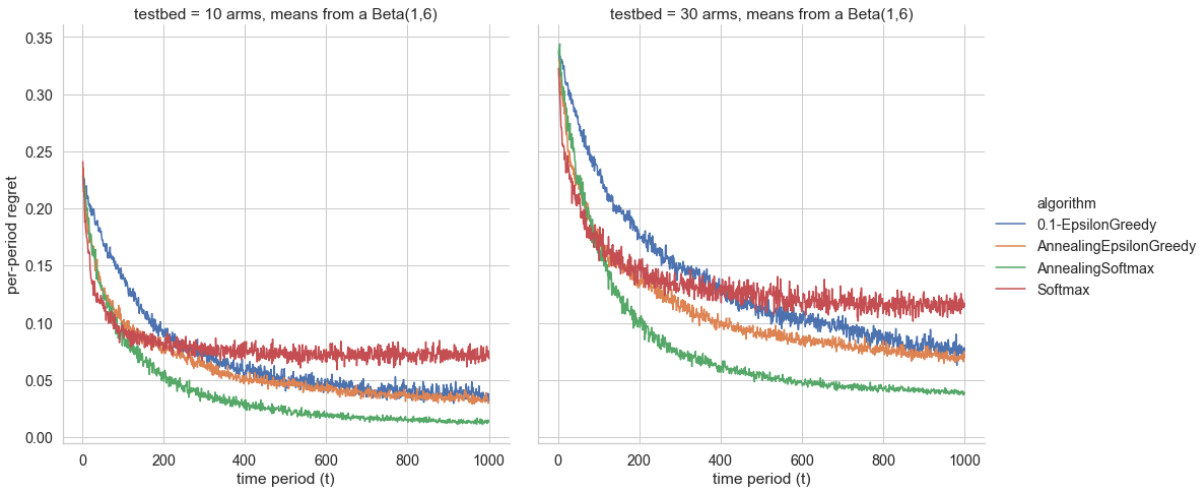


Figure 2: Algorithm comparison: ε-greedy ($\varepsilon = 0.1$), Softmax ($\tau = 0.1$), annealing ε-greedy and annealing Softmax. Per-period regret is averaged over 500 simulations, with T=1000. In the left panel, the simulations are carried over a 10 arms test-bed, while on the right on a 30 arms. True means are drawn from a $Beta(1, 6)$. In both annealing algorithms, $g_\varepsilon(t) = g_\tau(t) = 1/\sqrt{t}$ for every $t \in \{1, \dots, T\}$.

3.2 Upper Confidence Bound strategies

All the algorithms examined so far suffer from a fundamental issue: whether structured (Softmax) or unstructured (ε-greedy), their explorative rules are still bounded to a certain degree of randomness. This feature is a two-edged blade: on the one hand, it ensures that certain degree of exploration will always be maintained, while on the other hand it has a reduced control over the direction of such exploration, with a non-zero probability of exploring the worst arms. Moreover, random exploration is always *guillable* to some extent, as it can be easily fooled by a few negative experiences. In other words, the algorithm can value an arm sub-optimal only because in the first few trials its associated average reward has been low. Say for example that, in a 2-armed stochastic Bernoulli bandit, arm 1 has a probability of success of 0.6, and arm 2 of 0.5. If both arms have been pulled 3 times each, then the probability that the number of successes from arm 2 (the sub-optimal one) exceeds the ones from 1 is 0.254. Though low, there is a non negligible chance that the algorithm exploits arm 1 for several trials in a row, with a

subsequent increase of R_t. If the exploration parameters ε and τ decrease over time, there is a non-zero probability to get stuck on a sub-optimal arm because the best one has not received enough pulls. To tackle such an issue, we can opt for a fully deterministic strategy, which pays an initial extra explorative cost by pulling all of the available arms, and then allocates the subsequent pulls depending both on the number of times each arm has been selected and on its respective rate of success. This ensures that, in the long run, the algorithm correctly identifies the best arm with enough confidence.

In what follows, we introduce the most important class of frequentist deterministic algorithms, commonly known as the *Upper Confidence Bound* class (UCB for short). While there exist an increasing number of variants of the UCB, this section focuses on the algorithms presented in Auer et al. (2002), which are the building blocks of such a class. Before doing so, however, we need to review some important concentration results.

3.2.1 Sub-gaussian random variables

Following Bubeck and Cesa-Bianchi (2012), we phrase the analysis of the UCB bandit in term of the concept of sub-gaussianity, that is

Definition 3. *(Sub-gaussianity) - A random variable X with $\mathbb{E}[|X|] < \infty$ is said to be σ-sub-gaussian if $\forall \lambda \in \mathbb{R}$ it holds that $\mathbb{E}[\exp\{\lambda(X - \mathbb{E}[X])\}] \leqslant \exp\{\lambda^2\sigma^2/2\}$.*

Note in particular that, if X is σ-sub-gaussian and $0 < \sigma < \eta$, then X is also η-sub-gaussian.

Sub-gaussian random variables enjoy the following important inequality:

Theorem 2. *If X is σ-sub-gaussian, then for any $\epsilon \geqslant 0$,*

$$\mathbb{P}(X \geqslant \epsilon) \leqslant \exp\Big(-\frac{\epsilon^2}{2\sigma^2}\Big) \tag{3.3}$$

The proof follows the so called *Cramer-Chernoff method*, which consists of making the inequality depend on a certain parameter that will later be properly tuned. In general, this method is the key through which the literature proves all the regret bounds of the UCB.

Proof. Consider the following chain of inequalities for a certain $\lambda \in \mathbb{R}$:

$$\begin{aligned}\mathbb{P}(\{X - \mathbb{E}[X]\} \geqslant \epsilon) &= \mathbb{P}(\exp \lambda\{X - \mathbb{E}[X]\} \geqslant \exp\{\lambda\epsilon\}) \\ &\leqslant \mathbb{E}[\exp \lambda\{X - \mathbb{E}[X]\}]/\exp\{\lambda\epsilon\} \\ &\leqslant \exp\Big(\frac{\lambda^2\sigma^2}{2} - \lambda\epsilon\Big),\end{aligned}$$

where the first inequality follows from Markov's[11] inequality and the second from the definition of sub-gaussianity. Then, the last term can be minimized by setting $\lambda = \epsilon/\sigma^2$ □

Moreover, it holds that

Lemma 2. *Let X_1 and X_2 be two independent random variables such that X_1 is σ_1-sub-gaussian and X_2 is σ_2-sub-gaussian. Then, $X_1 + X_2$ is $\sqrt{\sigma_1^2 + \sigma_2^2}$-sub-gaussian.*

Now, for $X_1, \ldots, X_n$ independent and σ-sub-gaussian, let $\mu = \mathbb{E}[X]$ and $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n}(X_i - \mu)$, and pick a generic $\epsilon \geqslant 0$. Then, it holds that

$$\mathbb{P}(\hat{\mu} \geqslant \mu + \epsilon) = \mathbb{P}(\sum_{i=1}^{n}(X_i - \mu) \geqslant n\mu + n\epsilon) = \mathbb{P}(\sum_{i=1}^{n} X_i \geqslant n\epsilon) \leqslant \exp\Big(-\frac{n\epsilon^2}{2\sigma^2}\Big),$$

where the last inequality is a direct consequence of lemma 2, which implies that $\sum_{i=1}^{n} X_i$ is $\sqrt{n}\sigma$-sub-gaussian. In a similar fashion, it can be proved that

$$\mathbb{P}(\hat{\mu} \geqslant \mu - \epsilon) \leqslant \exp\Big(-\frac{n\epsilon^2}{2\sigma^2}\Big)$$

Finally, if one fixes $\delta \in [0, 1]$ as $\delta = \exp(-\frac{n\epsilon^2}{2\sigma^2})$, then with probability at least $1 - \delta$ it holds that

$$\mu \geqslant \hat{\mu} + \sqrt{\frac{2\sigma^2 \log(1/\delta)}{n}}. \tag{3.4}$$

The result in equation (3.4) is at the core of the UCB bandit analysis.

[11] Let X be a random variable with finite expectation. Then, $\forall \epsilon \geqslant 0$, it holds that

$$\mathbb{P}(|X| \geqslant \epsilon) \leqslant \frac{\mathbb{E}[|X|]}{\epsilon}$$

Sub-gaussianity is a property enjoyed by many random variables. In particular, Cesa-Bianchi and Lugosi (2006) proved that Bernoulli distributed random variables are 1/2-sub-gaussian. Exploiting such a result can significantly improve the performance of the simple UCB algorithm, as we illustrate in the next subsection.

3.2.2 UCB1 and UCB-Tuned

The fundamental principle that regulates every UCB algorithm is the so called "optimism in the face of uncertainty" (Lai and Robbins, 1985), which states that *a decision maker should act as if the environment is as nice as plausibly possible given the data collected prior to the decision.* In other words, if the learner observes that the average reward of an arm is low, he admits that such a bad performance can be caused by uncertainty in the estimate. As a consequence, he believes that the true rates of success are higher than those observed, and he is willing to adjust his beliefs only when he has reached a satisfying level of confidence over those same rates. To do so, he balances between exploration and exploitation by either pulling the observed highest rewarding arm, or trying those for which the estimates are more uncertain due to a low number of pulls. In particular, at the beginning of every trial t, the algorithm assigns a bonus to each arm, which is inversely related to the number of pulls that specific arm has received up to t. Then, it greedily selects the arm which has the highest Upper Confidence Bound, computed by summing the observed average reward at t and the bonus itself. To define this bonus, we just look back at equation (3.4), which ensures that, for every $\delta \in (0,1)$, for every arm $a \in \mathcal{A}$ and every time t,

$$\mathbb{P}\left(\mu_a \geqslant \underbrace{\hat{\mu}_a(t)}_{\text{estimated mean at } t} + \underbrace{\sqrt{\frac{2\sigma^2 \log(1/\delta)}{\hat{n}_a(t)}}}_{\text{bonus}}\right) \leqslant \delta. \tag{3.5}$$

Algorithm 5 reports the steps to implement the easiest version of the UCB algorithm family.

In the case of Bernoulli rewards, it holds that $\sigma = 1/2$. Nevertheless, the algorithm behaves well under $\sigma = 1$ as well.[12] Auer et al. (2002) first tested algorithm 5 on a 10-arm

[12] Bubeck and Cesa-Bianchi (2012) introduce a further, and more efficient, UCB algorithm in the case of independent Bernoulli distributed rewards, known as the KL-UCB (where KL stands for Kullback-

Algorithm 5 UCB algorithm with generic σ-sub-gaussian reward distribution

Input: $\delta \in (0,1)$ ▷ Set the width of the confidence bound
Input: $(\hat{n}_a(1))_{a=1}^K \leftarrow \mathbf{0} \in \mathbb{R}^K$ ▷ Pulls counter
Input: $(\hat{\mu}_a(1))_{a=1}^K \leftarrow \mathbf{0} \in \mathbb{R}^K$ ▷ Vector of estimated means
for $t = 1, 2, 3 \ldots, T$ **do**
    if $\exists a \in \mathcal{A}$ such that $\hat{n}_a(t) = 0$ **then**
        Select $a_t = a$ ▷ Pull each arm at least one time
    else
        for all $a \in \mathcal{A}$ **do**
            $bonus_{a,t} \leftarrow \sqrt{\frac{2\sigma^2 \log(1/\delta)}{\hat{n}_a(t)}}$
            $UCB_{a,t} \leftarrow \hat{\mu}_a(t) + bonus_{a,t}$
        end for
        Select arm $a_t = \arg\max_{a \in \mathcal{A}} UCB_{a,t}$
    Observe reward $r_{a_t,t}$ ▷ $r_{a_t,t} \sim P_{a_t}$ σ-sub-gaussian
    $\hat{n}_{a_t}(t+1) \leftarrow \hat{n}_{a_t}(t) + 1$
    $\hat{\mu}_{a_t}(t+1) \leftarrow [(\hat{n}_{a_t}(t+1) - 1)\hat{\mu}_{a_t}(t) + r_{a_t,t}]/\hat{n}_{a_t}(t+1)$ ▷ Update
end for

stochastic Bernoulli bandit by setting $\sigma = 1$ and $\delta = 1/t$. Under such a formulation, the algorithm is commonly referred as UCB1. In particular, they introduced the following theorem for the regret bound:

Theorem 3. *(Theorem 1 in Auer et al. (2002)) - Let ν be a K-armed stochastic bandit environment where P_a has support in $[0,1]$ $\forall a$. Then, for every time $t \in \{1, \ldots, T\}$, the regret for the UCB1 algorithm satisfies*

$$R_t \leqslant \left[8 \sum_{a:\mu_a < \mu^*} \left(\frac{\log t}{\Delta_{a^*,a}} \right) \right] + \left(1 + \frac{\pi^2}{3} \right) \sum_{a=1}^{K} \Delta_{a^*,a}, \tag{3.6}$$

where $(\mu_a)_{a=1}^K$ are the usual means of the rewards distributed according to $(P_a)_{a=1}^K$.

Note that the UCB1 algorithm is *optimal*, as its regret satisfies $R_T = O(\log T)$. Other versions of the above bound for cases when $\delta = 1/t^2$ and generic δ can be found in Bubeck and Cesa-Bianchi (2012) and Lattimore and Szepesvari (2018). All these bounds satisfy the optimality condition above. Auer *et al.* (2002) further improved on such an algorithm by including the rewards observed variances in the computation of the bonus. This version of algorithm 5 is called *UCB-Tuned*. In particular, at every time t, the UCB-Tuned selects

Leibler). For the purpose of this analysis, it is sufficient to focus on UCB1 and UCB-tuned.

a_t based on

$$a_t = \arg\max_a \left(\hat{\mu}_a(t) + \sqrt{\frac{\log t}{\hat{n}_a(t)} \min\left\{\frac{1}{4}, V_a(\hat{n}_a(t))\right\}} \right),$$

where

$$V_a(\hat{n}_a(t)) = \frac{1}{\hat{n}_a(t)} \sum_{s=1}^{\hat{n}_a(t)} r_{a,s}^2 - \hat{\mu}_a^2(t) + \sqrt{\frac{2\log t}{\hat{n}_a(t)}}.$$

Algorithm 6 reports the technicalities of the UCB-Tuned. While the authors could not provide a proper regret bound for this version of the algorithm, Audibert et al. (2009) extensively analyzed the long run properties of variance-enhanced UCB algorithms similar to the UCB-Tuned.

Algorithm 6 UCB-Tuned as in Auer et al. (2002)

1: Substitute steps 8-11 from algorithm 5 with the following:
2: **for all** $a \in \mathcal{A}$ **do**
3: $V_a(\hat{n}_a(t)) \leftarrow \frac{1}{\hat{n}_a(t)} \sum_{s=1}^{\hat{n}_a(t)} r_{a,s}^2 - \hat{\mu}_a(t)^2 + \sqrt{\frac{2\log t}{\hat{n}_a(t)}}$
4: $bonus_{a,t} \leftarrow \sqrt{\frac{\log(1/\delta)}{\hat{n}_a(t)} \times \min\left\{\frac{1}{4}, V_a(\hat{n}_a(t))\right\}}$
5: $UCB_{a,t} \leftarrow \hat{\mu}_a(t) + bonus_{a,t}$
6: **end for**

Let's now take a step back and explain the advantages of algorithm 5 and 6 as opposed to the other algorithms. As already stated above, all UCB strategies drive the exploration towards the arms that have either been explored less frequently than the others, or that have a high estimated probability of success (up to a certain level of confidence). These rules are purely deterministic, and allow for a lower level of explorative uncertainty than other mechanisms which include a random component in the selection. In particular, the UCB strategies presented here ensure that every arm is explored at least one time at the beginning of the experiment (otherwise, the computation of the bound is impossible), and later re-explored whenever the algorithm deems it necessary. It can happen instead that both the Softmax and the annealing ε-greedy forget to explore a certain arm, or do not explore it enough to be sure that it is indeed sub-optimal. In particular, the upper confidence bound of a sub-optimal arm may exceed the bound of the best arm for at least two reasons. First, in the case when $\delta = 1/t$, if the arm has been explored less frequently than the other arms at time t, then the bound increases, and so does the probability of

exploring that particular arm. Second, if the sub-optimal arm has returned more successes than it should have, its estimated mean is going to be high, and the algorithm explores it further to collect more information on its actual mean. But then, this type of *structured* exploration ensures that each arm is explored enough times so to understand with enough confidence which one is best according to the pre-established metric. However, such a reasoning has its benefits only when the UCB algorithm is run on a rather small set of arms, or when the horizon is sufficiently large. In the case of a large number of arms, the initial cost of exploring all arms (step 5-6 of algorithm 5) may never quite be recovered in later exploitation periods.

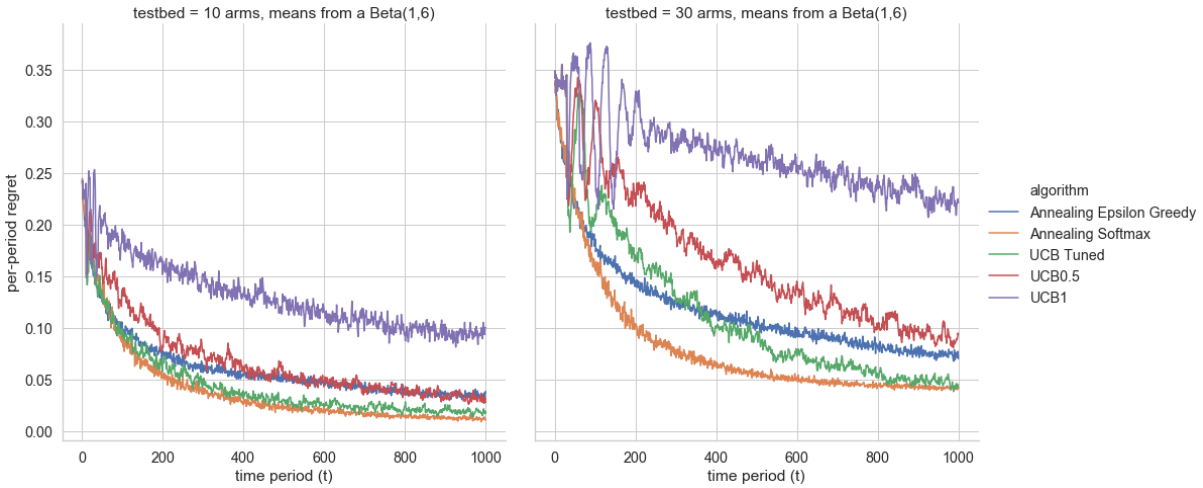


Figure 3: Average per-period regret over a 500 simulation of five algorithms: UCB1 as in Auer *et al.* (2002), UCB1 with $\sigma = 1/2$ to exploit the sub-gaussianity of the Bernoulli distribution, UCB-Tuned, annealing ε-greedy and annealing Softmax. As above, T=1000 and success probabilities are drawn from a $Beta(1, 6)$ in both panels.

Figure 3 plots the average regret obtained by the UCB algorithms described so far over 500 simulations, and compares the performances with the other algorithms. As above, means come from a $Beta(1, 6)$. The following facts are worth noticing. First, in both cases the UCB-Tuned achieves a better performance than the other two UCB algorithms. This could have easily be expected, as the UCB-Tuned exploits information that comes from both the estimated mean and variance of each arm. Second, exploiting the 1/2-sub-gaussianity in the Bernoulli distributed rewards strongly improves the performance of the variance-free UCB1. This is due to the fact that a lower σ is associated to tighter upper confidence bounds, which immediately translates into a reduction of the exploration of

low exploited arms in the long run. However, the property of the 1/2-sub-gaussianity could be efficiently used if and only if we are sure that the rewards are indeed generated from a Bernoulli distribution with a probability of success constant over time. Finally, the UCB class shows a noisy behavior, especially in the first few hundred trials. This initial noise comes from the fact that, in the first time periods, the estimates for the upper confidence bounds are imprecise due to low sample sizes. This means that the algorithm needs quite a few extra pulls to really get a correct estimate for the value of each arm. As in the first trials little information is available on the arms themselves, then the algorithm tends to explore a lot.[13] Finally, UCB-Tuned and annealing Softmax have a very similar performance in both test-beds, while annealing ε-greedy seems to achieve a better performance than the 1/2-UCB, especially under a high number of arms. Part of the reason has to do with the necessity of the UCB to explore all the arms in the test-bed at the beginning of the experiment. If the number of arms is large, then the per-period regret within the first trials will be necessarily high and noisy.

[13] The UCB family is quite efficient both in finding the actual best arm, and in ensuring a close surveillance of the environment. In other words, if there is an arm that changes the probability of generating a positive reward, eventually the UCB will detect it. However, this constant monitoring has a fundamental drawback: once the arms become huge in number, the UCB will not have enough trials left to exploit the best solution effectively.

4 Thompson Sampling

As we have already hinted in the introduction, bandit algorithms can be tackled from a Bayesian perspective as well. Indeed, the first bandit ever introduced in the reinforcement learning literature has a Bayesian nature (Thompson, 1933). However, due to the absence of closed form integrals and a subsequent computational intractability, Bayesian bandits have been largely ignored up until a few decades. In this respect, the seminal work of Gittins (1989) paved the way towards a full Bayesian implementation of an index strategy. In particular, the Gittins index is a forward looking algorithms that plays the arm that maximizes the expected present value of discounted future rewards at each round t. Though possessing explicit optimality properties, this technique is rather hard to implement, as it works only within a particular set of strict hypotheses.[14] Besides Gittins, other methods have been developed to further exploit the Bayesian prior-posterior link. For example, the Bayes-UCB (Kaufmann et al., 2012) follows the reasoning behind the UCB algorithm, but computes the upper confidence bound to the probability of success by looking at the percentiles of the posterior distribution of the reward of each arm.

For the purpose of this thesis, we decide to rather focus on Thompson Sampling, a technique that has been shown to be efficient and also rather simple to implement (Scott, 2010; Chapelle and Li, 2011). In particular, one convenient feature is that we can measure its performance in terms of the frequentist regret in equation (2.3), thus making us able to directly compare its behavior with that of the bandits explained so far (Agrawal and Goyal, 2011; Kaufmann et al., 2012). Nevertheless, Thompson Sampling has an essentially Bayesian nature: rather than selecting the arms according to their observed probability of success (or their respective upper confidence bounds), it treats such probabilities as random variables having their own distributions. Then, estimates for the success probabilities of each arm are simply computed by randomly drawing samples from such distributions, which are updated via Bayes rule as the experiment proceeds. As we will see in the following sections, such mechanism correctly identifies the best arm, while still maintaining a reasonable degree of exploration.

[14] For further details, see Weber (1992)

Scott (2010) and Chapelle and Li (2011) introduced the first version of the algorithm, and demonstrated its empirical superiority comparing it to other standard techniques. In particular, Chapelle and Li (2011) showed how Thompson sampling achieves a better performance than the Linear UCB algorithm introduced by Li et al. (2010) in the context of a news article recommendation system.[15] In addition, it is particularly efficient in dealing with the problem of delayed feedback (that is, entering in trial t and pulling an arm without having observed $r_{a_{t-1},t-1}$), which is quite common in all business applications (Vernade et al., 2018).

The purpose of this chaprter is twofold. First, it reviews the fundamentals behind Thompson Sampling and the more general *randomized probability matching* algorithm. The center of the analysis is to provide a clear framework to ensure a proper empirical comparison of their performances with the other non-Bayesian algorithms. Second, it introduces the Fractional Factorial Thompson Sampling algorithm presented in Scott (2010), which is the object this thesis intends to improve on. In particular, section 4.1 deals with the general formulation of Thompson Sampling and analyzes its simplest version (the beta-binomial case). Section 4.2 extends the analysis to a further class of algorithms, known as *randomized probability matching*. Finally, section 4.3 deals with the possibility to exploit the potential structure of the arms to increase the performance of the bandit using the algorithm proposed by Scott (2010).

4.1 Binomial Thompson Sampling

Thompson Sampling is an allocation heuristic that follows a simple reasoning: the number of observations to allocate to a treatment is set to be directly proportional to the probability of success of the treatment itself. In other words, if an option is more likely to lead to a favorable result than another, the researcher should somehow feel morally compelled to apply that option more than the others. To quote Thompson, 1933:

[15] A bandit algorithm that directly exploits information in conjunction with the observed reward from each arm is called *contextual* bandit. As an example, the UCB algorithm can be effectively improved by assuming that the reward of each arm linearly depends on the features of the users that observes the arm (such as age, or previously observed clicks). See Li et al. (2010) for detailed description of the LinUCB and its application on the Yahoo! FrontPage module for news article recommendation.

"If such a discipline were adopted, even though it were not the best possible, it seems apparent that considerable saving of individuals otherwise sacrificed to the inferior treatment might be effected. This would be important in cases where either the rate of accumulation of data is slow, or the individuals treated are valuable, or both."

Eventually, this policy leads to a low per-period regret as the experiment proceeds. In particular, Thompson Sampling has essentially a Bayesian nature. This means that we need to set a prior over the rewards' distributions. We denote such prior as $q(\cdot)$. Suppose now that ν is our usual K-armed stochastic Bernoulli bandit in the sense of definitions 1 and 2, and assume we have already observed history $\mathbb{H}_{t-1}$. Then, the true success probability of every arm $a \in \mathcal{A}$ follows a distribution

$$(\mu_a \mid \mathbb{H}_{t-1}) \sim q(\mu_a \mid \mathbb{H}_{t-1}) \tag{4.1}$$

called "posterior", obtained via Bayes' rule. In what follows we indicate both the distribution and the density function of the mean reward as $q(\cdot \mid \cdot)$. Then, the posterior for μ_a at the beginning of time t (prior to the observation of $r_{a_t,t}$) is given by

$$q(\mu_a \mid \mathbb{H}_{t-1}) = \frac{q(\mu_a) \prod_{s=1}^{t-1} p_a(r_{a_s,s} \mid \mu_a) \mathbb{1}\{a_s = a\}}{\int q(\mu_a) \prod_{s=1}^{t-1} p_a(r_{a_s,s} \mid \mu_a) \mathbb{1}\{a_s = a\} d\mu_a}, \tag{4.2}$$

where $p_a(r_{a_t,t} \mid \cdot)$ is the likelihood function that models the observed rewards of arm a_t at time t. Note that we are assuming that the reward distribution of each arm depends uniquely on μ_a, and that the arms are independent. These assumptions will be relaxed in section 4.2. Furthermore, the indicator functions simply serve to indicate that the posterior distribution of the success probability of arm a receives an update only in those trials where arm a is selected (that is, all $s < t$ for which $a_s = a$).

As in the case of frequentist bandits, the optimal policy π^* can either come from a minimization of the loss associated to the sequential choice, or over the maximization of the expected reward. Thompson Sampling achieves such a maximization by drawing a random sample from the posterior distribution of the reward of each arm, and subsequently

selecting the arm whose draw has had the highest value (Chapelle and Li, 2011; Russo et al., 2018). As we will see later on, such a procedure allocates the pulls at every trial t according to the reasoning introduced by Thompson (1933), which is known as *randomized probability matching*. Algorithm 7 reports the general formulation of Thompson Sampling.

Algorithm 7 General Thompson Sampling

Prior Input: $\forall a \in \mathcal{A} : \mu_a \sim q(\mu_a)$ ▷ Prior over the environment
for $t = 1, 2, 3 \ldots, T$ **do**
    For all $a \in \mathcal{A}$, sample $\hat{\mu}_a(t) \sim q(\mu_a|\mathbb{H}_{t-1})$, with $q(\mu_a|\mathbb{H}_0) = q(\mu_a)$
    Select arm $a_t = \arg\max_{a \in \mathcal{A}} \hat{\mu}_a(t)$
    Observe reward $r_{a_t,t}$
    Update the posterior via Bayes' rule as in equation (4.2)
end for

Note that, unlike in the frequentist cases, the estimates for the probability of success $\hat{\mu}_a(t)$ at every time t do not come from a simple average of the observed rewards, but are treated as random samples from the posterior distribution[16]. This randomness is what drives the exploration: if indeed the posterior is poorly concentrated, or only few observations have been registered for each arm, then the variance of the $\hat{\mu}_a(t)$ drawn will be quite high, allowing for sub-optimal arms to get their chance of being explored. As an example, figure 4 plots 1000 draws from a $Beta(2, 1)$ and a $Beta(20, 10)$ against 1000 draws from a $Beta(10, 20)$. While for the first two distributions the mean is $\frac{2}{3}$, the variance becomes progressively lower (0.056 and 0.0071 respectively), thus concentrating the mass around the mean. On the other hand, the distribution plotted on the horizontal axes has an inferior mean of $\frac{2}{5}$. But then, as the parameters of the beta increase (or, equivalently, as the number of observations goes up), it becomes clear that the payoff of the distribution plotted on the vertical axis is better than the one on the horizontal axis.

The easiest way to exploit this posterior updating is using the conjugacy property of certain classes of distributions[17] . In particular, in our K-armed stochastic Bernoulli bandit environment ν with mean vector $\boldsymbol{\mu} = (\mu_a)_{a=1}^K$, the choice of a $Beta(\alpha_a, \beta_a)$ prior

[16] Note that If $t = 1$, $\hat{\mu}_a(1)$ is sampled from the prior $q(\mu_a)$ for every arm.

[17] A class of distributions is conjugated with respect to a particular likelihood function if it is closed with respect to Bayesian updating. In a more formal way, let $p(\theta) \in \mathbb{P}_\theta$ be a generic element of a class of distributions $\mathbb{P}_\theta$ which all depend on a parameter $\theta \in \mathbb{R}$, and $p(x \mid \theta)$ be the likelihood that models the data. Then, $\mathbb{P}_\theta$ is said to be *conjugate* with respect to $p(x \mid \theta)$ if and only if $\forall p \in \mathbb{P}_\theta : \; p(\theta \mid x) \in \mathbb{P}_\theta$.

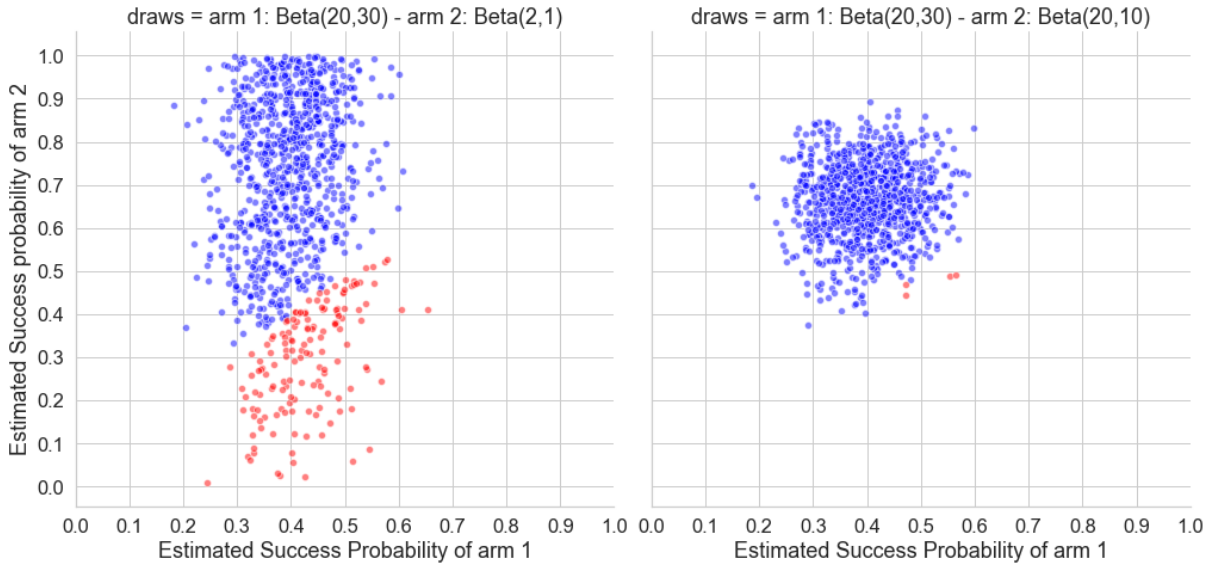


Figure 4: Scott (2010): One thousand draws from the joint distribution of independent beta distributions. Red points indicate that the draws from the distributions on the horizontal axes exceed the one on the vertical axis, while blue the opposite.

over each arm seems natural. But then, the joint prior distribution of the vector $\boldsymbol{\mu}$ is

$$q(\boldsymbol{\mu}) \propto \prod_{a=1}^{K} \mu_a^{\alpha_a - 1}(1-\mu_a)^{\beta_a - 1}.$$

Then, given the fact that for each arm a and for every time t, $(r_{a,t}|\mu_a) \sim Bernoulli(\mu_a)$, then the posterior of $\boldsymbol{\mu}$ is

$$q(\boldsymbol{\mu}|\mathbb{H}_{t-1}) \propto \prod_{a=1}^{K} \mu_a^{\alpha_a + S_a(t) - 1}(1-\mu_a)^{\beta_a + \hat{n}_a(t) - S_a(t) - 1},$$

where $\hat{n}_a(t)$ is the usual number of times arm a has been selected up to trial t, and $S_a(t) := \sum_{s=1}^{t-1} r_{a_s,s}\mathbb{1}(a_s = a)$ is the total number of successes registered for arm a up to t. But then, Thompson Sampling simply requires to draw one vector $\hat{\boldsymbol{\mu}}(t) = (\hat{\mu}_1(t), \ldots, \hat{\mu}_K(t))$ from the joint distribution above[18] at each trial, and then to select the arm that corresponds to the highest $\hat{\mu}_a(t)$. When the number of pulls allocated to an arm is low, the draws from its *beta* posterior will have a high variance. On the other hand, a high number of plays will lead to a draw that is highly concentrated around the empirical mean, allowing for a more precise ranking of the arms. In other words, the algorithm favors exploration within the first trials,

[18] Note that arms here are assumed independent.

and exploitation in the latest ones. Algorithm 8 reports in detail the mechanism behind the beta-binomial Thompson Sampling. In particular, such a formulation works both within the context of Bernoulli distributed rewards, and in the case when the responses follow more general binomial distribution. This fact comes particularly at hand when the same arm is pulled multiple times at each t, a common situation in many real business applications. In general, α_a and β_a are set to 1 $\forall a \in \{1, \dots, K\}$, which is equivalent to assigning a uniform prior on [0,1] to the success probability of each arm.

Algorithm 8 Beta-Binomial Thompson Sampling

Prior Inputs: $\forall a \in \mathcal{A} : \ \hat{\mu}_a(1) \sim Beta(\alpha_a, \beta_a)$
Input: $(\hat{n}_a(1))_{a=1}^K \leftarrow \mathbf{0} \in \mathbb{R}^K$ ▷ Pulls counter
Input: $(S_a(1))_{a=1}^K \leftarrow \mathbf{0} \in \mathbb{R}^K$ ▷ Successes counter
for $t = 1, 2, 3 \dots, T$ **do**
    for all $a \in \mathcal{A}_t$ **do**
        Draw $\hat{\mu}_a(t) \sim Beta(\alpha_a + S_a(t), \beta_a + \hat{n}_a(t) - S_a(t))$
    end for
    Select the arm $a_t = \arg\max_{a \in \mathcal{A}} \hat{\mu}_a(t)$
    Observe reward $r_{a_t,t}$ ▷ $(r_{a_t,t} \mid \mu_{a_t}) \sim Bernoulli(\mu_{a_t})$
    $\hat{n}_a(t+1) \leftarrow \hat{n}_a(t) + 1$
    $S_{a_t}(t+1) \leftarrow S_{a_t}(t) + r_{a_t,t}$
end for

While a Bayesian regret analysis is rather hard to treat, Agrawal and Goyal (2011) provide an upper bound to the regret R_T in the case of beta-binomial Thompson sampling that is directly comparable to the bounds we have seen in the sections above. In particular, recalling that the sub-optimality gap of arm a is $\Delta_{a^*,a} = \mu^* - \mu_a$, where a^* is the arm with a probability of success equal to $\mu^* = \max_a \mu_a$, the following theorem holds:

Theorem 4. *Agrawal and Goyal (2011) - Let ν be a K-armed stochastic Bernoulli bandit environment with success probability vector $\boldsymbol{\mu} = (\mu_a)_{a=1}^K$ and finite horizon T.*
If $\forall a \in \{1, \dots, K\} : \mu_a \sim Beta(1, 1)$, then $\forall t \in \{1, \dots, T\}$:

$$R_t \leqslant O\left(\left[\sum_{a:\mu_a<\mu^*} \frac{1}{\Delta_{a^*,a}^2}\right]^2 \log t\right). \tag{4.3}$$

But then, Thompson sampling is quasi-optimal up to a constant that is inversely proportional to the squared sub-optimality gap. Comparing equation (4.3) with the bound

on the UCB1 introduced by Auer *et al.* (2002) in equation (3.6), we can see that the second possesses a slightly more convenient dependency on $\Delta_{a*,a}$ (as the term is not squared), and an optimal dependency on t. However, Thompson Sampling is superior in terms of immediate exploration and early per-period regrets. Moreover, it is not forced to explore whenever it has an appropriate estimate of the probability of success of each arm, and does not suffers much efficiency loss in the case of delayed feedback (Chapelle and Li, 2011).

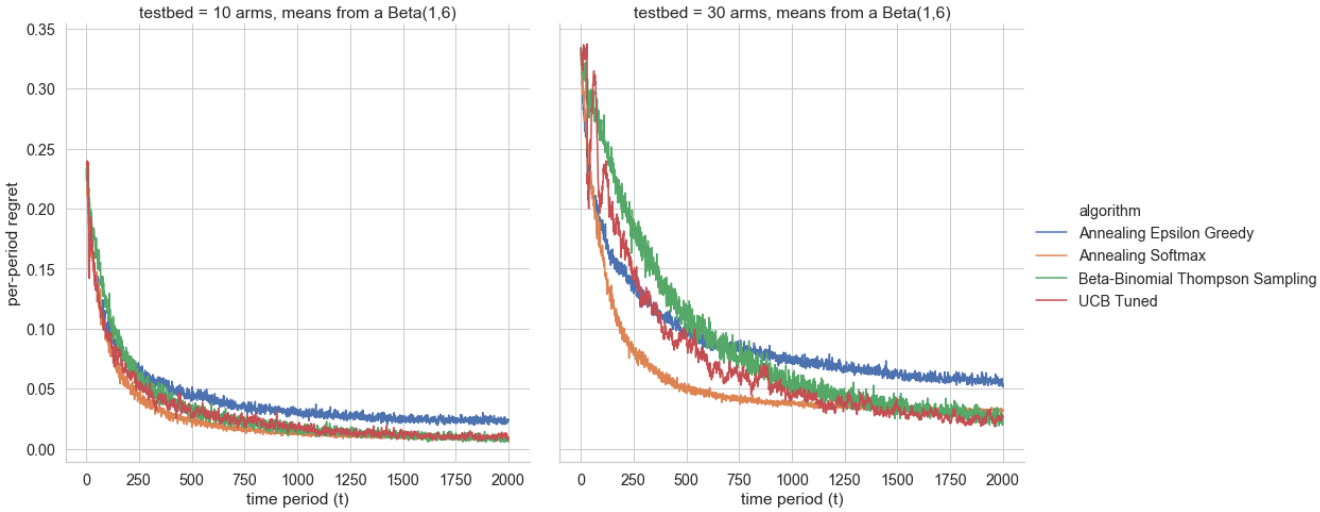


Figure 5: Average per-period regret over 500 simulation of four algorithms: annealing ε-greedy, annealing Softmax, UCB-Tuned and binomial Thompson Sampling with a uniform prior over the probability of success of each arm. Horizon is set to T=2000

Figure 5 displays the same comparison made in figure 3 but complements it with the inclusion of the results of 500 simulations from a the beta-binomial Thompson Sampling with uniform priors. To make the simulation convey more information, we have doubled the horizon, setting it to $T = 2000$. Panels descriptions are the as usual, with true means generated randomly form a $Beta(1, 6)$. It is immediately clear that the performance of the beta-binomial Thompson Sampling is preferable in long run, as no stationary behavior is shown in neither test-beds. In particular, when the number of arms in low, the overall performance of all algorithms is similar. Under many arms however, Thompson Sampling suffers more within the first trials, and recovers later on. As a general rule of thumb then, if the horizon the is low (below 1000 trials), the best performing algorithm is the annealing Softmax. Otherwise, under a large horizon, both UCB-Tuned and Thompson

Sampling are preferable.

In the following section, we present a more general formulation of Thompson Sampling, which allows to exploit the potential correlation that can exists in certain real business scenarios.

4.2 Randomized Probability Matching

Thompson sampling is an algorithm that belongs to the family of *randomized probability matching* policies (*RPM* for short). Within the context of Bayesian bandit algorithms, a *RPM* algorithm randomly selects one or more arms at every time t according to a probability vector $\mathbf{w}_t = (w_{at})_{a=1}^K$, where w_{at} is the estimated probability that arm a is the most rewarding one. More formally, let $\boldsymbol{\mu} = (\mu_a)_{a=1}^K$ denote the usual success probability vector in a K-armed stochastic Bernoulli bandit environment. The probability of arm a to be the best at the beginning of time t (that is, after having observed history $\mathbb{H}_{t-1}$) is

$$w_{at} := \mathbb{P}(\mu_a = \max\{\mu_1, \dots, \mu_K\} | \mathbb{H}_{t-1}) \tag{4.4}$$

Suppose now that the true success probabilities depend on a vector of unknown parameters $\boldsymbol{\theta} \in \Theta \subset \mathbb{R}^p$. In other words, $\forall t \in \{1, \dots, T\}$ it holds that $(r_{a_t,t} \mid \boldsymbol{\theta}) \sim Bernoulli(\mu_{a_t}(\boldsymbol{\theta}))$. Then, recalling that $p(\cdot|\mathbb{H}_{t-1})$ is the posterior distribution of $\boldsymbol{\theta}$ conditioned on history $\mathbb{H}_{t-1}$, equation (4.4) can be rewritten as

$$w_{at} = \int_\Theta \mathbb{1}_a(\boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathbb{H}_{t-1}) \mathrm{d}\boldsymbol{\theta} = \mathbb{E}[\mathbb{1}_a(\boldsymbol{\theta})|\mathbb{H}_{t-1}], \tag{4.5}$$

where $\mathbb{1}_a(\boldsymbol{\theta}) = \begin{cases} 1 & \text{if } \mu_a(\boldsymbol{\theta}) = \max\{\mu_1(\boldsymbol{\theta}), \dots, \mu_K(\boldsymbol{\theta})\} \\ 0 & \text{elsewhere} \end{cases}$.

Following the idea of Thompson (1933), the vector of "best arm" probabilities $\mathbf{w}_t$ determines the way in which the various treatments are allocated over the experimental subjects. Suppose that at each round t we have a budget of 100 units - observation that we can spend, and K arms among which to choose. In the *RPM* framework, each of the 100 units has a probability w_{at} of ending up in arm a. If a^* is the arm that has

the highest probability of success, we expect that eventually $w_{a^*,t} \to 1$ and $w_{a \neq a^*,t} \to 0$ as $t \to \infty$. In other words, once the algorithm has identified the best arm, it allocates all the available pulls to it up until the end of the experiment. The following example may further clarify the idea. Every day, a business company has to contact 100 clients to sell a particular product, and the clients are divided into K clusters according to certain socio-economic characteristics. These clusters constitute our arms set. The goal of the company is to identify the cluster that has the highest probability of accepting the offer and buy the product. Each time a cluster receives a pull, one client in that cluster is randomly called. At the beginning of the experiment then, the 100 pulls will be allocated almost uniformly across the clusters, while half way through the experiment, all the pulls will start concentrating around the one that has delivered the highest number of positive responses. Note that the number of allocations to distribute at every time t can vary depending on the experimental design. In the previous sections, we have allocated 1 pull each t. This is equivalent to a scenario where the response of the selected arm at every trial is immediate[19]. The main drawback of equation (4.5) is that the integral is often not available in closed form, so it needs to be estimated via Monte Carlo approximation. Specifically, letting $g \in \{1, \dots, G\}$ the index for a generic draw $\boldsymbol{\theta}_t^{(g)}$ from the posterior distribution $p(\boldsymbol{\theta}|\mathbb{H}_{t-1})$ available at round t, one can compute $\mathbb{1}_a(\boldsymbol{\theta}_t^{(g)})$ by simply ranking the corresponding means $\mu_a(\boldsymbol{\theta}_t^{(g)})$. Then, by the law of large numbers, for each arm $a \in \{1, \dots, K\}$

$$\frac{1}{G} \sum_{g=1}^{G} \mathbb{1}_a(\boldsymbol{\theta}_t^{(g)}) \to w_{at} \quad \text{as} \quad G \to \infty. \tag{4.6}$$

In other words, w_{at} represents the expected number of times arm a has been the best arm in a total of G simulations. Algorithm 9 reports the technical details of *RPM*. In particular, the version reported here allocates M pulls across the arms at every trial t

[19] A situation where multiple arms are selected within the same trial period t can have two possible interpretations. The first is the one in the example above: at each round, the learner has a budget of pulls to allocate within the K-armed test-bed. The second interpretation instead treats the multiple pulls as a delayed feedback. In other words, the responses come in batches based on a context-related rule. For example, the number of clicks on an ad online refreshes every hour. In such a setting, the learner has to allocate a pull without knowing the result of the previously allocated one. As Chapelle and Li (2011) point out, *RPM* strategies are particularly efficient in dealing with batch updates, while other deterministic strategies suffer more.

according to a sequence of integers $(\psi_{a,t})_{a=1}^{K}$ with $\sum_{a=1}^{K}\psi_{a,t} = M$ randomly drawn from a multinomial distribution with probabilities given by $\mathbf{w}_t$. This means that each arm a is pulled $\psi_{a,t}$ times, each with probability w_{at}, and delivers the sequence of rewards $\mathbf{r}_{a,t} = (r_{a,t}^{(1)}, \dots, r_{a,t}^{(\psi_{a,t})})$ of length $\psi_{a,t}$ (where each component is $Bernoulli(\mu_a(\boldsymbol{\theta}))$ distributed).

Algorithm 9 Randomized Probability matching with M pulls

Prior Inputs: $\boldsymbol{\theta} \sim q(\boldsymbol{\theta})$ ▷ Prior over $\boldsymbol{\theta}$
Input: $M \geqslant 1$ pulls to allocate each day, $G \geqslant 1$ samples
for $t = 1, 2, 3 \dots, T$ **do**
    for $g = 1, \dots, G$ **do**
        Draw $\boldsymbol{\theta}_t^{(g)} \sim q(\boldsymbol{\theta}|\mathbb{H}_{t-1})$ ▷ Sample from the posterior
        Observe the arm $a_t^{(g)} = \arg\max_{a\in\mathcal{A}} \mu_a(\boldsymbol{\theta}_t^{(g)})$
        $\mathbb{1}_{a_t^{(g)}}(\boldsymbol{\theta}_t^{(g)}) = 1$, $\mathbb{1}_a(\boldsymbol{\theta}_t^{(g)}) = 0 \ \forall a \neq a_t^{(g)}$
    Compute $\mathbf{w}_t = (w_{at})_{a=1}^{K}$ as in equation (4.6)
    Draw $(\psi_{a,t})_{a=1}^{K} \in \mathbb{R}^K$ from $Multinomial(M, \mathbf{w}_t)$
    For all a with $\psi_{a,t} > 0$, observe rewards sequence $(r_{a,t}^{(i)})_{i=1}^{\psi_{a,t}}$
    Update the posterior via Bayes' rule.
end for

In the beta-binomial Thompson sampling case as in algorithm 8, we can assume, without loss of generality, that $\mu_a(\boldsymbol{\theta}) = \theta_a \ \forall a \in \mathcal{A}$. In other words, the true probability of success of each arm depends exclusively on a single parameter θ_a that is specific to the arm itself. This means that the corresponding allocation probability vector $\mathbf{w}_t$ has components given by

$$w_{at} = \frac{1}{\mathrm{B}(\hat{\alpha}_t, \hat{\beta}_t)} \int_0^1 \theta_a^{\hat{\alpha}_t - 1}(1-\theta)^{\hat{\beta}_t - 1} \prod_{k\neq a} \mathbb{P}(\theta_k < \theta_a|\mathbb{H}_{t-1}) d\theta_a,$$

where $\hat{\alpha}_t := \alpha + S_a(t)$ and $\hat{\beta}_t := \beta + \hat{n}_a(t) - S_a(t)$ and $\mathrm{B}(\cdot,\cdot)$ is the normalizing beta function associated to the $Beta(\cdot,\cdot)$ distribution.

If we take a closer look at algorithm 9, two fundamental aspects are worth mentioning. The first one concerns the difference between Softmax and *RPM* allocation rules. The second one instead deals with the difference between pure Thompson Sampling and *RPM*.

4.2.1 Difference between RPM and Softmax allocation rules

Both *RPM* and Softmax algorithms randomly pull one or more arms at each trial t according to an unbalanced probability vector. Such unbalance comes from the fact that more rewarding arms have a higher probability of being selected. However, the difference between $\hat{\mathbf{w}}_t$ in algorithm 3 and $\mathbf{w}_t$ in algorithm 9 concerns the exact meaning of the components in each vector. Within the same K-armed stochastic Bernoulli bandit with $\mu_a(\boldsymbol{\theta}) = \theta_a$, the probability of selecting arm 1 after having observed history $\mathbb{H}_{t-1}$ in the *RPM* is given by

$$w_{1t} = \mathbb{P}(\theta_1 = \max\{\theta_1, \dots, \theta_K\} \mid \mathbb{H}_{t-1}). \tag{4.7}$$

In other words, arm 1 is selected with a probability which is equal to the (estimated) probability of being the best arm in the pool. The Softmax algorithm instead randomly explores arm 1 with a probability given by

$$\hat{w}_{1t} = \frac{e^{\hat{\mu}_1(t)/\tau}}{\sum_{j=1}^{K} e^{\hat{\mu}_j(t)/\tau}} \tag{4.8}$$

where $\tau > 0$ is the temperature parameter that drives the explorative component on the algorithm, and $\hat{\mu}_a(t)$ is the estimated probability of success of arm a (computed as in the frequentist case). Such a probability comes from a pre-defined reweighing rule of the average success rate of each arm. While in both cases the more rewarding the arm is, the higher the chances of being selected it has, the exploration that emerges from equation (4.7) and from (4.8) is completely different. The first obvious reason is that the *RPM* has a Bayesian nature and the selection probabilities depend on the prior $p(\boldsymbol{\theta})$. On the other hand, Softmax allocation rule is purely frequentist and solely depends on τ. The second reason deals with the way in which $\mathbf{w}_t$ and $\hat{\mathbf{w}}_t$ are designed. As the *RPM* randomly selects an arm according to its probability of being the best, at some point in time $w_{1t} \to 1$ if arm 1 is the (estimated) best arm, and $w_{1t} \to 0$ otherwise. The Softmax instead assigns a higher selection probability to the most rewarding arms, without necessarily allocating all the pulls to the highest one. Indeed, the behavior of $\hat{w}_{1t}$ strictly depends on the pre-set temperature parameter. If τ is kept constant throughout the experiment, each of

the component of $\hat{\mathbf{w}}_t$ will eventually become constant and greater than zero. Figure 6 highlights such a difference.

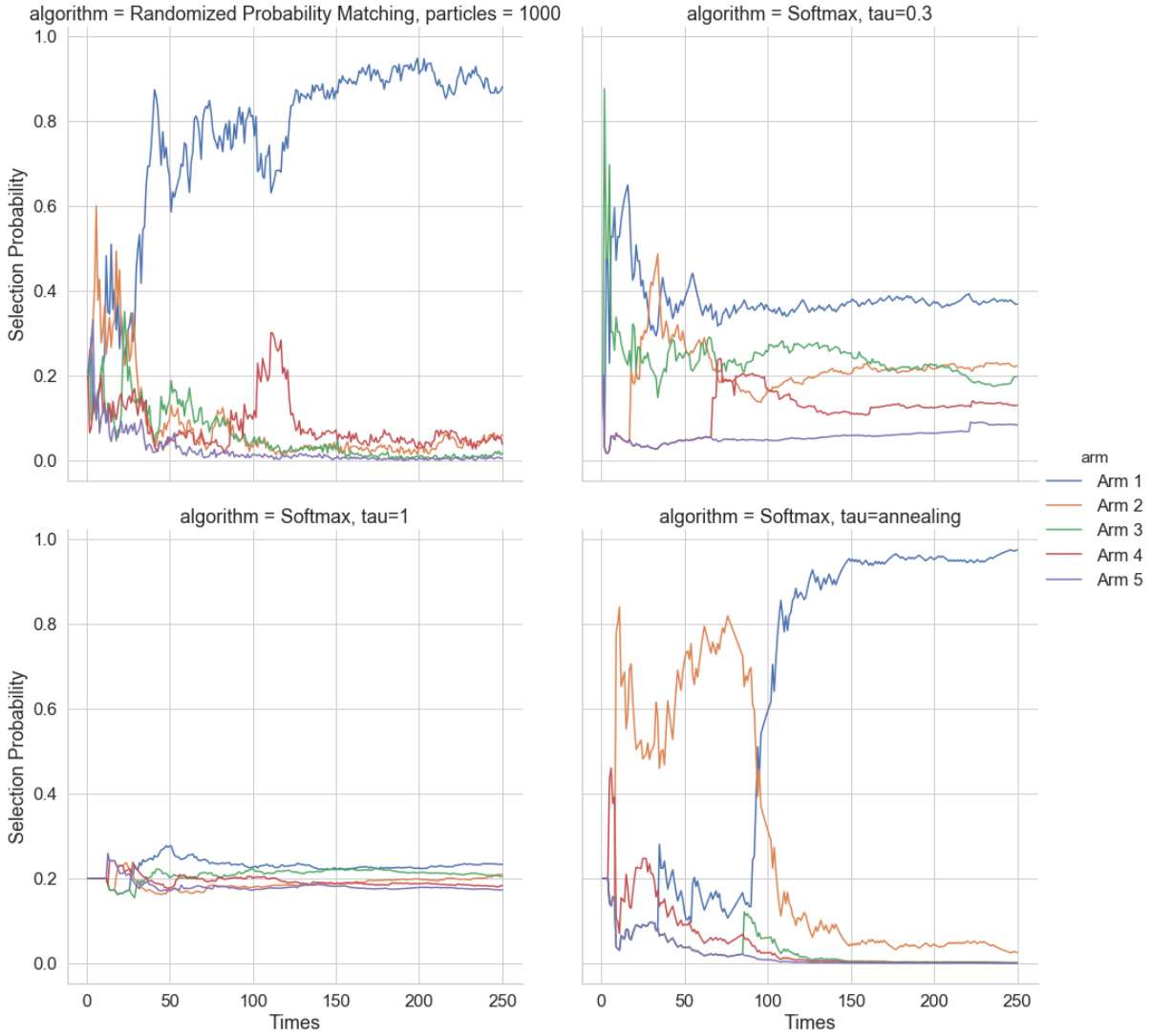


Figure 6: Allocation probabilities over a 5-arms test-bed for four algorithms: Beta-Binamial Thompson Sampling with $G = 1000$ and uniform prior over the probability of successes, annealing Softmax with $g_\tau(t) = 1/\sqrt{t}$, and Softmax with $\tau = 1$ and $\tau = 0.3$. Arm 1 has a probability of success of 0.58, and arm 2 of 0.4, arm 3 of 0.3, arm 4 of 0.2 and arm 5 of 0.1.

While the behavior of the probability schedules $\mathbf{w}_t$ and $\hat{\mathbf{w}}_t$ is similar in the annealing version of the Softmax algorithm (where the temperature is decreasing with t), the probabilities in the Softmax with constant τ become stable after a few trials. In other words, this algorithm selects sub-optimal arms with a positive probability even though it has identified the best one. A proper tuning of τ is indeed the fundamental issue of the Softmax strategy: if a too high τ risks to be costly in terms of regret, while a low

one may make the algorithm get stuck on a sub-optimal arm. This is true also in the annealing version, where the risk of getting stuck might still occur if the rate at which τ decreases is too fast. But then, it is clear how, under a reasonably distributed prior over the parameters (such as a uniform distribution), *RPM* strategies possess a desirable selection criteria.

4.2.2 Difference between RPM and pure Thompson Sampling

As already stated above, Thompson Sampling belongs to the family of *randomized probability matching* algorithms. And indeed, a careful reader will have already noticed that the general version of Thompson Sampling presented in algorithm 7 is equivalent to a *RPM* with only one draw from the posterior at each trial t (that is, when $G = 1$, Scott (2015)). As a consequence, Thompson Sampling is a lot faster than *PRM* when sampling from conjugate posterior distributions is rather simple. In this case, the empirical difference between Thompson sampling and *RPM* is minimal, as figure 7 demonstrates. On the other hand, if the posterior is not available in closed form and thus we need to

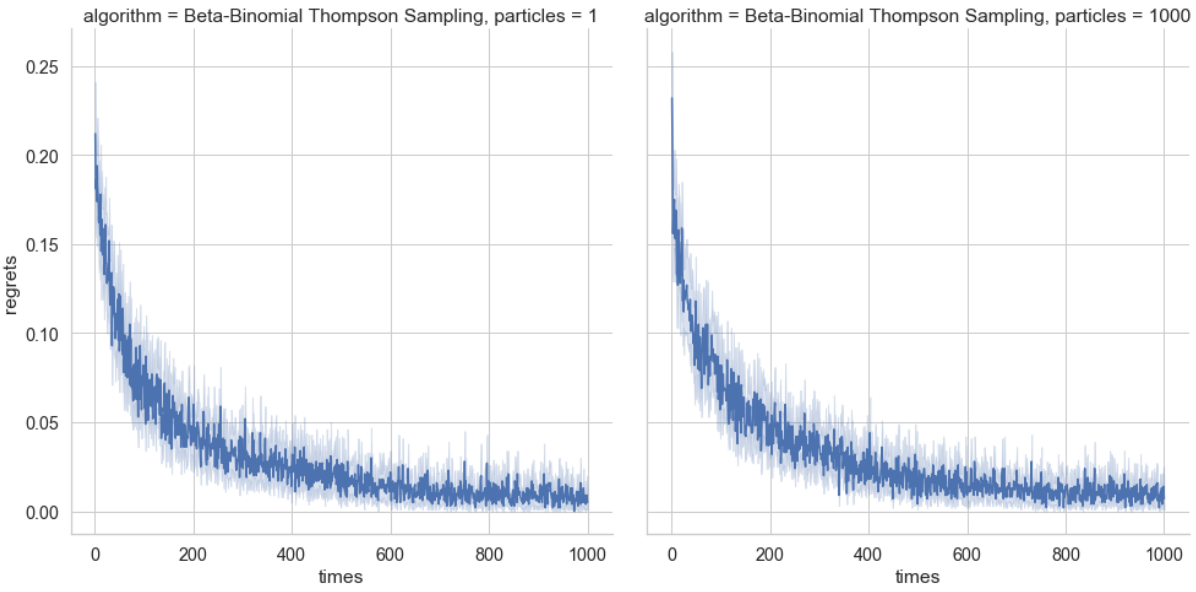


Figure 7: Average per-period regret of Binomial Thompson Sampling with $G = 1$ and with $G = 1000$ over 100 simulations of a 5-arms testbed with probability of success of arm 1 equal to 0.58, arm 2 of 0.4, arm 3 of 0.3, arm 4 of 0.2 and arm 1 of 0.1. Uniform priors were assigned over the probability of success of every arm.

apply MCMC methods to sample from it, selecting an arm based on a single draw can

be inefficient. Think for example of a standard Gibbs Sampler where the first J samples are thrown away (the so called "burn-in"). Which sample should we use at every trial t to compute the allocation probabilities? Should it be the $J + 1^{th}$, or the $J + 1000^{th}$?. As there is no pre-defined rule, the safest option is to exploit all the samples from J+1 to $J + 1000$ to compute $\mathbf{w}_t$, so to allocate the pulls according to a properly computed best arm probabilities. In line with the literature (Zhou et al., 2018; Scott, 2010, 2015; Urteaga and Wiggins, 2018; Russo et al., 2018), from now on will treat Thompson Sampling and *RPM* as synonyms.

4.3 Fractional Factorial Thompson Sampling

As seen in section 4.1, Thompson Sampling is an efficient strategy in almost every case, but eventually its performance decreases as the number of arms among which to choose becomes large. In the framework of a real business application, the number of arms can be certainly high, but can also follow a factorial structure (Scott, 2015). For example, the arms can be defined at the beginning of the experiment as a combination of several features. Rather than looking for the best arm *directly*, we can think of an algorithm that aims at finding it *indirectly* by understanding which combination of such features possesses the highest probability of success. Suppose that the same business company described before has divided its client base in mutually exclusive clusters according to three features: sex (male - female), age group (below 25, 25-34, 35-44, 45-54, 55-64, else) and geographical location (north, centre, south). Then, there are 36 different possible arm-clusters to contact for our campaign, a number that, under a sufficiently long horizon, is still manageable. However, if individuals where clustered according to age group, sex, the 20 regions in the country and whether they live in a city or in a non-urban area, the total number of arms becomes 480. Even with a long horizon, such a huge pool of arms cannot be tackled efficiently by treating each arm independently from the others.

Ideally, we wish to incorporate this factorial structure of the arms in our model so to drive the exploration towards the features that are really relevant in shaping the reward. Suppose that $\forall t \in \{1, \dots, T\}$ the probability of success for the arm chosen at t is

determined by

$$\mu_{a_t}(\boldsymbol{\theta}) = \mathbb{P}(r_{a_t,t} = 1|\boldsymbol{\theta}) = H(\mathbf{x}_{a_t}^{\mathrm{T}}\boldsymbol{\theta}), \tag{4.9}$$

where $\boldsymbol{\theta} \in \Theta \subset \mathbb{R}^p$ is a vector of parameters, $\mathbf{x}_{a_t}$ is the vector that collects all the p features of arm a_t and $H(\cdot)$ is a link function. In other words, the bandit environment $\nu = (P_a)_{a=1}^K$ is a K-armed stochastic Bernoulli bandit with $Bernoulli\big(H(\mathbf{x}_a^{\mathrm{T}}\boldsymbol{\theta})\big)$ distributed rewards. Consistent with Scott (2010), we focus here on the case in which the link function has a probit formulation with $\Phi(z)$ denoting the cumulative density function of the standard normal distribution evaluated at z. Furthermore, we assume, without loss of generality, that the vectors $\mathbf{x}_{a_t}$ are exclusively made of binary indicator variables with 0 indicating that a level of a feature is absent, and 1 if it is present in the arm pulled at time t. Note that the number of parameters in this case is different from the number of actual features. In the business company example, the number of features is 4 (age, sex, region, urban), but the associated number of parameters in the probit regression with binary regressors is $p = 28$ (constant included)[20]. As the rest of the analysis depends on p rather than on the number of real features-clusters, from now on we prefer to use the term "feature" to indicate the generic component of the vector of dummies $\mathbf{x}_{a_t} \in \mathbb{R}^p$. Specifically, we denote by $\mathbf{X}_{t-1} \in \mathbb{R}^{(t-1)\times p}$ the matrix that contains all the feature vectors of the arms selected up to the beginning of time t (that is, before action a_t has been played), and by $\mathbf{r}_{t-1}$ the associated response vector. In other words, under the case of one pull per day, $\mathbf{X}_{t-1} = (\mathbf{x}_{a_1}^{\mathrm{T}}, \ldots, \mathbf{x}_{a_{t-1}}^{\mathrm{T}})^{\mathrm{T}}$, and $\mathbf{r}_{t-1} = (r_{a_1,1}, \ldots, r_{a_{t-1},t-1})^{\mathrm{T}}$. Slightly forcing the notation, in what follows we indicate history $\mathbb{H}_{t-1}$ as the couple $(\mathbf{X}_{t-1}, \mathbf{r}_{t-1})$.

Based on the above assumptions, our bandit environment becomes the parameter space Θ itself. Denote our prior over Θ as $q(\boldsymbol{\theta})$. Then, the posterior distribution of $\boldsymbol{\theta}$ given $\mathbb{H}_{t-1}$ becomes

$$q(\boldsymbol{\theta} \mid \mathbf{X}_{t-1}, \mathbf{r}_{t-1}) = \frac{q(\boldsymbol{\theta}) \prod_{s=1}^{t-1} \Phi(\mathbf{x}_{a_s}^{\mathrm{T}}\boldsymbol{\theta})^{r_{a_s,s}} (1 - \Phi(\mathbf{x}_{a_s}^{\mathrm{T}}\boldsymbol{\theta}))^{1-r_{a_s,s}}}{\int_\Theta q(\boldsymbol{\theta}) \prod_{s=1}^{t-1} \Phi(\mathbf{x}_{a_s}^{\mathrm{T}}\boldsymbol{\theta})^{r_{a_s,s}} (1 - \Phi(\mathbf{x}_{a_s}^{\mathrm{T}}\boldsymbol{\theta}))^{1-r_{a_s,s}} \mathrm{d}\boldsymbol{\theta}} \tag{4.10}$$

[20] This comes from a simple combination. The feature "sex" has 2 levels, and so its associated dummy is $x_{sex} = 1$ if male, 0 if female. Equivalently, "region" has 20 levels that can be discretized in 19 dummies. Then, the total number of parameters is 1 ("sex") + 5 ("age group") + 3 ("geographical location") + 1 ("urban area") +1 (the constant) = 28.

where $r_{a_s,s}$ is our usual Bernoulli distributed reward. Note that, despite the fractional structure of the configurations, the distribution of the rewards coming from each arm is assumed independent of the other arms, conditionally on $\boldsymbol{\theta}$. Algorithm 10 reports the general mechanism of the Fractional Factorial Thompson Sampling.

Algorithm 10 Fractional Factorial Thompson Sampling with M pulls

1: Input: $\mathbf{X}_0 \in \mathbb{R}^{0\times p}$, $\mathbf{r}_0 \in \mathbb{R}^0$ ▷ Empty feature matrix and outcomes vector
2: Input: $M \geqslant 1$ $G \geqslant 1$.
3: **for** $t = 1, 2, 3 \ldots, T$ **do**
4: **for** $g \in 1, 2, 3, \ldots, G$ **do**
5: Sample $\boldsymbol{\theta}_t^{(g)}$ from $q(\boldsymbol{\theta} \mid \mathbf{X}_{t-1}, \mathbf{r}_{t-1})$.
6: Get $a_t^{(g)} = \arg\max_{a\in\mathcal{A}} \mathbf{x}_a^{\mathrm{T}} \boldsymbol{\theta}_t^{(g)}$
7: $\mathbb{1}_{a_t^{(g)}}(\boldsymbol{\theta}_t^{(g)}) = 1$, $\mathbb{1}_a(\boldsymbol{\theta}_t^{(g)}) = 0$ $\forall a \neq a_t^{(g)}$
8: **end for**
9: Compute $\mathbf{w}_t = (w_{at})_{a=1}^K$ as in equation (4.6)
10: Draw $(\psi_{a,t})_{a=1}^K \in \mathbb{R}^K$ from $Multinomial(M, \mathbf{w}_t)$
11: **for all** $a \in \mathcal{A}$ **where** $\psi_{a,t} > 0$ **do**
12: Observe the response vector $\mathbf{r}_{a,t}$ of size $\psi_{a,t}$.
13: $\mathbf{r} \leftarrow (\mathbf{r}^{\mathrm{T}}, \mathbf{r}_{a,t}^{\mathrm{T}})^{\mathrm{T}}$ ▷ Extend the outcomes vector
14: $\mathbf{X} \leftarrow [\mathbf{X}^{\mathrm{T}}, \underbrace{\mathbf{x}_a, \ldots, \mathbf{x}_a}_{\psi_{a,t} \text{ times}}]^{\mathrm{T}}$ ▷ Extend the feature matrix
15: **end for**
16: **end for**

At time 0, the algorithm requires to initialize an empty feature matrix $\mathbf{X}_0$ and an empty vector of responses $\mathbf{r}_0$. At each trial t the pulls are distributed across all the possible (or available) combinations according to the usual probability probability vector $\mathbf{w}_t$, computed via Monte Carlo approximation with the draws from the posterior distribution $q(\boldsymbol{\theta} \mid \mathbf{X}_{t-1}, \mathbf{r}_{t-1})$. Notice in particular that, as $\Phi(\cdot)$ in an increasing function, it is sufficient to increase the best arm counter at every trial g of the arm with the highest $\mathbf{x}_a^{\mathrm{T}} \boldsymbol{\theta}_t^{(g)}$ (point 6 in the algorithm). This allows to save some computational time, especially under the case of a high p. Finally, the matrix $\mathbf{X}_{t-1}$ and the vector $\mathbf{r}_{t-1}$ are updated by simply appending the M feature vectors of the selected arms and the M responses observed, respectively.

A careful reader may argue that such a setting, while being reasonable in terms of assumptions, is rather impractical to apply. Indeed, as we have seen in section 4.1, the most efficient way to deal with Bayesian models is by exploiting the conjugacy property

of certain distributions. However, such a property is rather rare and, at first glance, it does not seem to apply to our case. Not surprisingly, the absence of a conjugate family of distributions has lead to the design of several approximations methods to sample from the posterior in equation (4.10), especially in the case of a normal prior over each parameter of the probit regression. These methods mainly deal with standard data augmentation procedures via Markov Chain Monte Carlo (Albert and Chib, 1993), adaptive Metropolis-Hastings algorithms (Roberts and Rosenthal, 2001) and generalizations of the Hamiltonian Monte Carlo (Hoffman and Gelman (2014)). All of these methods present their strengths and their liabilities, elegantly summarized by Chopin and Ridgway (2017). In particular, in the case of large n and moderate p (that is, high number of observations and reasonable amount of parameters in the probit regression), standard data augmentation suffers more than the Hamiltonian sampler and exhibits mixing problems in the case of small n and large p as well. Moreover, MCMC and Hamiltonian-based methods are generally computationally inefficient for our purposes, as they require a non-negligible amount of time to be implemented when p is large. As a consequence, these liabilities have limited the practical use of Thompson Sampling in real business applications (especially within the context of online settings, where immediate elaboration of the feedback is necessary and the overall amount of observations is very high).

However, the absence of a conjugate family of distribution under the case of a probit likelihood has been recently denied by Durante (2019), where it is shown that, under a normal prior over the regression parameters, the posterior follows a unified skew-normal distribution (Arellano-Valle and Azzalini, 2006; Azzalini and Capitanio, 2014). While this novel result offers the opportunity to enlarge the literature on Thompson Sampling in the case of contextual bandit algorithms, it still faces computational bottlenecks in large n studies. The solution presented in this work consists of a Sequential Monte Carlo approach. These two advances in the fractional factorial literature are presented in Section 5. The remaining part of this Section instead summarizes the MCMC sampling method used in Scott (2010).

4.3.1 Posterior approximation via Gibbs sampling

The Gibbs sampling technique (Gelfand and Smith, 1990) is a well known Markov Chain Monte Carlo (MCMC) algorithm that is largely used to sample from the joint posterior distribution of a set of parameters in the absence of direct conjugacy between the prior and the likelihood. Again, let $\boldsymbol{\theta} = (\theta_1, \dots, \theta_p)$ with $p > 1$ be the vector of parameters of interest, and fix a generic time period t within the horizon. To simplify the analysis, we set the number of pulls allocated each day equal to 1. Then, our history $\mathbb{H}_{t-1}$ consists of the couple $(\mathbf{X}_{t-1}, \mathbf{r}_{t-1})$, with $\mathbf{X}_{t-1} = (\mathbf{x}_{a_1}^{\mathrm{T}}, \dots, \mathbf{x}_{a_{t-1}}^{\mathrm{T}})^{\mathrm{T}}$ and $\mathbf{r}_{t-1} = (r_{a_1,1}, \dots, r_{a_{t-1},t-1})^{\mathrm{T}}$. Suppose now that drawing $\theta_l \sim q(\theta_l \mid \{\theta_j,\ j \neq l\}, \mathbb{H}_{t-1})$ - the full conditional distribution of a single component given the observed history - is rather simple. Initializing the Markov Chain at an arbitrary value $\boldsymbol{\theta}^{(0)} = (\theta_1^{(0)}, \dots, \theta_p^{(0)})$, at each round $g = 1, \dots, G$ the sampler requires to simulate:

$$
\begin{aligned}
&\theta_1^{(g)} \quad \text{from} \quad q(\theta_1 \mid \{\theta_j^{(g-1)},\ j \neq 1\}, \mathbb{H}_{t-1}) \\
&\theta_2^{(g)} \quad \text{from} \quad q(\theta_2 \mid \{\theta_1^{(g)}, \theta_j^{(g-1)},\ j \geqslant 2\}, \mathbb{H}_{t-1}) \\
&\qquad \vdots \\
&\theta_p^{(g)} \quad \text{from} \quad q(\theta_p \mid \{\theta_j^{(g)},\ j \leqslant p\}, \mathbb{H}_{t-1}).
\end{aligned}
$$

Then, it can be shown that as $G \to \infty$, then $\boldsymbol{\theta}^{(G)}$ is a sample from the the joint posterior distribution of $(\boldsymbol{\theta} \mid \mathbb{H}_{t-1})$. Within the context of probit regression, Albert and Chib (1993) introduced a data augmentation approach based on the Gibbs sampler to provide an efficient approximation of the distribution in equation (4.10).

Define now a vector of latent variables $\mathbf{z} = (z_{a_1}, \dots, z_{a_s})$ where $z_{a_s} \sim \mathcal{N}(\mathbf{x}_{a_s}^{\mathrm{T}}\boldsymbol{\theta}, 1)$ for every $s \in \{1, \dots, t-1\}$. Following Albert and Chib (1993), we rewrite the components of $\mathbf{r}$ as $r_{a_s,s} = \mathbb{1}\{z_{a_s} > 0\}$, where $\mathbb{1}$ denotes the usual indicator variable[21]. Then, the conditional distribution of each z_{a_s} given $\mathbf{X}_{t-1}$, $\mathbf{r}_{t-1}$, and $\boldsymbol{\theta}$ is a truncated normal. To see

[21] It follows from simple calculations that $\forall s \in \{1, \dots, t-1\} : r_{a_s,s} \sim \mathit{Bernoulli}(\Phi(\boldsymbol{\theta}^{\mathrm{T}}\mathbf{x}_{a_s}))$. In fact:

$$\mathbb{P}(r_{a_s,s} = 1) = \mathbb{P}(z_{a_s} > 0) = 1 - \Phi(-\mathbf{x}_{a_s}^{\mathrm{T}}\boldsymbol{\theta}) = \Phi(\mathbf{x}_{a_s}^{\mathrm{T}}\boldsymbol{\theta})$$

this, notice that the joint posterior density of $\boldsymbol{\theta}$ and $\mathbf{z}$ given $\mathbf{r}_{t-1}$ and $\mathbf{X}_{t-1}$ is

$$q(\boldsymbol{\theta}, \mathbf{z} \mid \mathbf{X}_{t-1}, \mathbf{r}_{t-1}) \propto q(\boldsymbol{\theta}) \prod_{s=1}^{t-1} \big(\mathbb{1}\{z_{a_s} > 0, r_{a_s,s} = 1\} + \mathbb{1}\{z_{a_s} \leqslant 0, r_{a_s,s} = 0\} \big) \phi(z_{a_s}; \mathbf{x}_{a_s}^{\mathrm{T}} \boldsymbol{\theta}, 1), \tag{4.11}$$

where $\phi(\cdot, \mu, \sigma^2)$ denotes the density function of the normal distribution with mean μ and variance σ^2.

Based on the above results, samples from $q(\boldsymbol{\theta}|\mathbf{X}_{t-1}, \mathbf{r}_{t-1})$ can be obtained via a Gibbs sampler which iterates between the two full conditionals $q(\boldsymbol{\theta}|\mathbf{z}, \mathbf{X}_{t-1}, \mathbf{r}_{t-1})$ and $q(\mathbf{z} \mid \boldsymbol{\theta}, \mathbf{X}_{t-1}, \mathbf{r}_{t-1})$. Recalling the above discussion, both are available in closed-form. Indeed, from equation (4.11) it holds that

$$q(\boldsymbol{\theta} \mid \mathbf{z}, \mathbf{X}_{t-1}, \mathbf{r}_{t-1}) \propto q(\boldsymbol{\theta}) \prod_{s=1}^{t-1} \phi(z_{a_s}; \mathbf{x}_{a_s}^{\mathrm{T}} \boldsymbol{\theta}, 1). \tag{4.12}$$

Then, by assigning a multivariate normal prior over $\boldsymbol{\theta}$ as $\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\xi}, \Sigma)$ with $\boldsymbol{\xi} \in \mathbb{R}^p$ and $\Sigma \in \mathbb{R}^{p \times p}$, we can easily exploit the conjugacy property of the normal model. In particular, according to standard linear regression results it holds that $(\boldsymbol{\theta} \mid \mathbf{z}, \mathbf{X}_{t-1}, \mathbf{r}_{t-1}) \sim \mathcal{N}(\tilde{\boldsymbol{\theta}}, \Omega)$, with

$$\Omega = (\Sigma^{-1} + \mathbf{X}_{t-1}^{\mathrm{T}} \mathbf{X}_{t-1})^{-1} \quad \text{and} \quad \tilde{\boldsymbol{\theta}} = \Omega(\mathbf{X}_{t-1}^{\mathrm{T}} \mathbf{z} + \Sigma^{-1} \boldsymbol{\xi}). \tag{4.13}$$

On the other hand, the components of $\mathbf{z}$ are conditionally independent random variables with full conditional distribution as

$$z_{a_s} \mid \mathbf{r}_{t-1}, \mathbf{x}_{a_s}, \boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\theta}^{\mathrm{T}} \mathbf{x}_{a_s}, 1) \begin{cases} \text{truncated at the left by 0 if } r_{a_s,s} = 1 \\ \text{truncated at the right by 0 if } r_{a_s,s} = 0. \end{cases} \tag{4.14}$$

Algorithm 11 summarizes the steps to follow to apply the Albert and Chib (1993) algorithm to our case (step 5 in algorithm 10). To shorten the notation, from now on we denote the truncated normal distribution with support in $[0, \infty)$ as $\mathcal{N}_0^+$, and $\mathcal{N}_0^-$ in the other case. Note that the sampler does not change much under $M > 1$. The only difference is the dimension of the matrix $\mathbf{X}_{t-1}$, which lives in $\mathbb{R}^{M(t-1) \times p}$, and the vector $\mathbf{r}_{t-1}$, that is in $\mathbb{R}^{M(t-1)}$.

Algorithm 11 Albert and Chib (1993) Gibbs sampler (with $M = 1$)

Prior Input: $\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\xi}, \Sigma)$
Fix $t \in \{1, \dots T\}$
$\mathbb{H}_{t-1} = (\mathbf{X}_{t-1}, \mathbf{r}_{t-1}) \in \mathbb{R}^{t-1} \times \mathbb{R}^{(t-1)\times p}$ ▷ Observed History
Input: $\boldsymbol{\theta}^{(0)} \in \mathbb{R}^p$
for $g \in 1, 2, 3, \dots, G$ **do**
    for $s \in 1, \dots, t-1$ **do**
        if $r_{a_s,s} = 1$ **then**
            Sample $z_{a_s}^{(g)} \sim \mathcal{N}_0^+(\mathbf{x}_{a_s}^{\mathrm{T}} \boldsymbol{\theta}^{(g-1)})$
        else
            Sample $z_{a_s}^{(g)} \sim \mathcal{N}_0^-(\mathbf{x}_{a_s}^{\mathrm{T}} \boldsymbol{\theta}^{(g-1)})$
    end for
    $\mathbf{z}^{(g)} \leftarrow (z_{a_1}^{(g)}, \dots z_{a_{t-1}}^{(g)})^{\mathrm{T}} \in \mathbb{R}^{t-1}$
    Sample $\boldsymbol{\theta}_t^{(g)} \sim \mathcal{N}(\tilde{\boldsymbol{\theta}}^{(g)}, \Omega)$ with $\tilde{\boldsymbol{\theta}}^{(g)}$ and Ω as in (4.13)
end for

After convergence, the quantities $\boldsymbol{\theta}_t^{(g)}$ can be viewed as samples from the posterior $q(\boldsymbol{\theta} \mid \mathbf{X}_{t-1}, \mathbf{r}_{t-1})$. This method allows Monte Carlo inference on the functionals of the posterior in an effective way, but becomes slower as t increases. Moreover, it loses effectiveness under a high number of parameters (Chopin and Ridgway, 2017). The main disadvantage of algorithm 11 within the context of bandit algorithms is the sequentiality of the draws at each round, which require a non negligible amount of time (especially under high G and t). Indeed the MCMC sampler needs to be implemented from zero at the beginning of each trial t and its overall duration increases as the experiment proceeds. This computational burden significantly lowers the applicability of MCMC methods in the context of real applications of Bayesian bandits. In the next section we introduce a result that allows us to draw i.i.d. samples from the posterior without relying on MCMC methods or data augmentations.

5 Advances in Fractional Factorial Thompson Sampling

As already hinted in the introduction, this work intends to extend the current literature on Thompson Sampling in two directions. First, it applies the novel conjugacy property for the regression coefficients in the probit likelihood with Gaussian priors presented in Durante (2019) to the Fractional Factorial algorithm in Scott (2010). This conjugacy allows to draw i.i.d. samples from the exact posterior, without the need to run time-consuming MCMC data augmentation algorithms. Second, it presents a first solution to the scalability associated with the proposed i.i.d. sampler, by combining this routine with Sequential Monte Carlo solutions. The resulting performance of these two advances is almost equal to the one obtained by applying Albert and Chib (1993) Gibbs sampler, but the overall running time of the new algorithm is significantly lower, thus increasing its overall applicability.

The chapter is divided as follows: Section 5.1 introduces the Independent Additive Sampler in Durante (2019) and applies it to the Fractional Factorial Thompson Sampling scheme, Section 5.2 describes a Sequential Monte Carlo procedure which we propose to improve scalability, and Section 5.3 compares the performance of all the algorithms presented in this thesis.

5.1 The unified skew-normal and the Independent Additive Sampler

Durante (2019) proved that, under a normal prior over each parameter of the probit regression in equation (4.9), the joint posterior distribution $q(\boldsymbol{\theta} \mid \mathbf{X}_{t-1}, \mathbf{r}_{t-1})$ is a unified skew-normal (Arellano-Valle and Azzalini, 2006; Azzalini and Capitanio, 2014). In line with the literature, we indicate such a distribution as $\text{SUN}_{p,t-1}$, where p is the number of parameters, and $t-1$ is the usual number of observations up to the beginning of time t. This distribution is a generalization of the skew-normal distribution, and has the following probability density function:

Definition 4. - *Unified skew-normal - Let* $\boldsymbol{\theta} \sim SUN_{p,t-1}(\boldsymbol{\xi}, \Sigma, \Delta, \boldsymbol{\gamma}, \Gamma)$*, where* $\boldsymbol{\xi} \in \mathbb{R}^p$*,* $\Sigma \in \mathbb{R}^{p \times p}$*,* $\Delta \in \mathbb{R}^{p \times t-1}$*,* $\boldsymbol{\gamma} \in \mathbb{R}^{t-1}$ *and* $\Gamma \in \mathbb{R}^{t-1 \times t-1}$*. Let* $\Sigma = \sigma \bar{\Sigma} \sigma$*, where* $\sigma = diag\{\sqrt{\Sigma_{[1,1]}}, \ldots, \sqrt{\Sigma_{[p,p]}}\}$ *and* $\bar{\Sigma}$ *is a correlation matrix. Then, the density function of* $\boldsymbol{\theta}$ *is equal to*

$$\phi_p(\boldsymbol{\theta} - \boldsymbol{\xi}; \Sigma) \frac{\Phi_{t-1}(\boldsymbol{\gamma} + \Delta^T \bar{\Sigma}^{-1} \sigma^{-1}(\boldsymbol{\theta} - \boldsymbol{\xi}); \Gamma - \Delta^T \bar{\Sigma}^{-1} \Delta)}{\Phi_{t-1}(\boldsymbol{\gamma}; \Gamma)}, \tag{5.1}$$

where $\phi_p(\boldsymbol{\theta} - \boldsymbol{\xi}; \Sigma)$ *denotes the density function of a p-variate normal with mean* $\boldsymbol{\xi}$ *and variance-covariance matrix* Σ *and* $\Phi_k(\cdot; A)$ *denotes the cumulative density function of* $\mathcal{N}_k(\mathbf{0}_k, A)$*, that is k-variate normal with variance-covariance matrix* A *and mean* $\mathbf{0}_k$ *evaluated at* $\cdot$*.*

The intuition behind the shape of the $\text{SUN}_{p,t-1}$ is rather simple: it consists of a multivariate normal distribution that is "skewed" by a component defined as a ratio of two cumulative distribution functions of a multivariate normal. In particular, the amount of the skewness is determined mainly by Δ, whereas $\boldsymbol{\gamma}$ and Γ serve as parameters controlling dependence and departures from normality. Notice in fact that if $\Delta = 0_{p \times t-1}$, then $\text{SUN}_{p,t-1}(\boldsymbol{\xi}, \Sigma, 0_{p \times t-1}, \boldsymbol{\gamma}, \Gamma) = \mathcal{N}_p(\boldsymbol{\xi}, \Sigma)$. For further properties of the SUN, see Azzalini and Capitanio (2014), chapter 7.

To see how and why the above distribution is the posterior in a probit regression with Gaussian priors for the coefficients, we first restrict the analysis to a simple $\text{SUN}_{1,1}$. Let $\mathbb{P}(r_{a_1,1} = 1 \mid \theta) = \Phi(\theta x_{a_1})$ be our probit regression, where $x_{a_1} \in \mathbb{R}$ indicates the generic unique feature of the arm chosen at $t = 1$ (not necessarily a dummy). If we set $\theta \sim \mathcal{N}(0, 1)$ as a prior, the posterior distribution of θ given x_{a_1} and $r_{a_1,1}$ is

$$\begin{aligned} p(\theta \mid x_{a_1}, r_{a_1,1}) &\propto \phi(\theta) \times \{\Phi(\theta x_{a_1})^{r_{a_1,1}} (1 - \Phi(\theta x_{a_1}))^{1 - r_{a_1,1}}\} \\ &\propto \phi(\theta) \Phi((2r_{a_1,1} - 1)\theta x_{a_1}) \\ &\propto \phi(\theta) \Phi((2r_{a_1,1} - 1) x_{a_1} (x_{a_1}^2 + 1)^{-1/2} \theta; (x_{a_1}^2 + 1)^{-1}) \end{aligned}$$

Notice that the first proportionality comes from the application of Bayes' rule (normal prior and Bernoulli likelihood), whereas the second follows form the fact that $r_{a_1,1} \in \{0, 1\}$ and that $(1 - \Phi(\theta x_{a1})) = \Phi(-\theta x_{a_1})$. If we set $\gamma = 0$, $\xi = 0$, $\Sigma = 1$, $\Delta = (2r_{a_1,1} -$

$1)x_{a_1}(x_{a_1}^2+1)^{-1/2}$ and $\Gamma = (x_{a_1}^2+1)^{-1} + \Delta^2 = 1$[22] and $\Phi(\gamma;\Gamma) = \Phi(0;1) = 1/2$. This means that

$$p(\theta \mid x_{a_1}, r_{a_1,1}) \propto \phi(\theta)\frac{\Phi(\Delta\theta; 1-\Delta^2)}{\Phi(0;1)}$$

which is the kernel of a unified skew-normal. In other words, $(\theta \mid x_{a_1}, r_{a_1,1}) \sim \text{SUN}_{1,1}(0, 1, (2r_{a_1,1}-1)x_{a_1}(x_{a_1}^2+1)^{-1/2}, 0, 1)$. Figure 8 shows how the variations of $r_{a_1,1}$ and x_{a_1} influence the shape of the unified skew-normal density function. Note that when $x_{a_1} = 0$, then $\Delta = 0$ and the distribution is a standard normal. If the response of the arm is positive ($r_{a_1,1} = 1$), then the density is skewed towards the right for positive values of the real line, and towards the left for negative ones. If instead $r_{a_1,1} = 0$, the opposite happens.

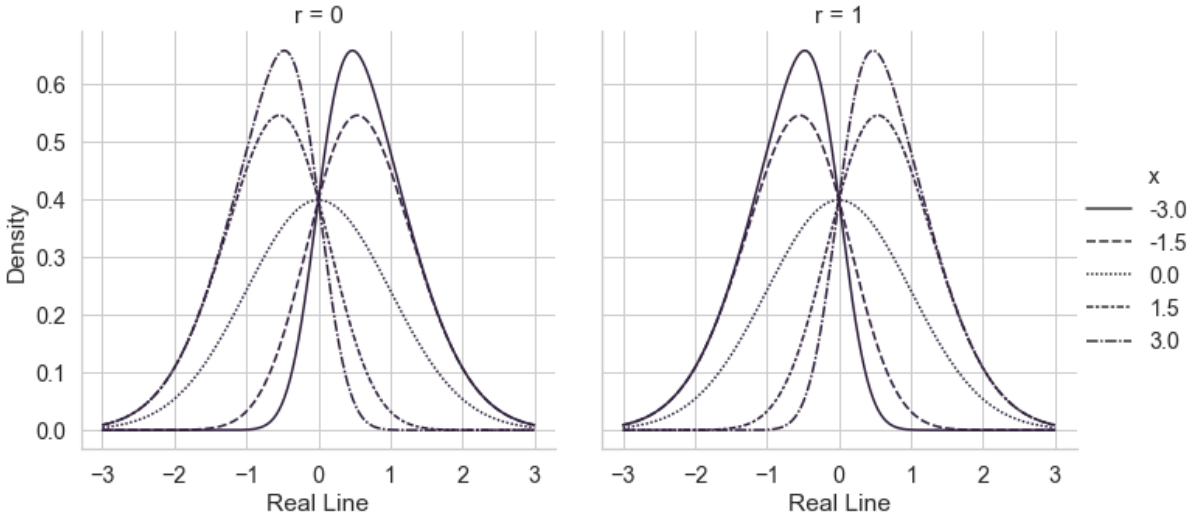


Figure 8: Probability density function of a $\text{SUN}_{1,1}(0, 1, (2r_{a,1}-1)x_a(x_a^2+1)^{-1/2}, 0, 1)$

Theorem 5 generalizes the conjugacy property under a generic history $\mathbb{H}_{t-1}$ with $\boldsymbol{\theta} \in \mathbb{R}^p$. For a proper proof, see Durante (2019)[23].

Theorem 5. *Let $\mathbb{H}_{t-1} = (\mathbf{X}_{t-1}, \mathbf{r}_{t-1})$ be the history observed up to the beginning of time t. If $\forall a \in \mathcal{A}\ \mathbb{P}(r_a = 1 \mid \boldsymbol{\theta}) = \Phi(\mathbf{x}_a^T\boldsymbol{\theta})$ holds with $\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\xi}, \Sigma)$ as a prior, then*

$$(\boldsymbol{\theta} \mid \mathbf{X}_{t-1}, \mathbf{r}_{t-1}) \sim SUN_{p,t-1}\{\boldsymbol{\xi}, \Sigma, \bar{\Sigma}\sigma S^T D^{-1/2}, D^{-1/2}S\boldsymbol{\xi}, D^{-1/2}(S\Sigma S^T + \mathbf{I}_{t-1})D^{-1/2}\}, \quad (5.2)$$

[22] $(x_{a_1}^2+1)^{-1} + \underbrace{(2r_{a_1,1}-1)^2}_{=1} x_{a_1}^2(x_{a_1}^2+1)^{-1} = (x_{a_1}^2+1)^{-1} \times (x_{a_1}^2+1) = 1$

[23] Note that the notation reported here is slightly different than that presented in the published version of Durante (2019). In particular, it corresponds to an earlier unpublished version of the paper.

where $S = diag\{2r_{a_1,1} - 1, \ldots, 2r_{a_{t-1},t-1} - 1\}\mathbf{X}_{t-1} \in \mathbb{R}^{t-1\times p}$ *and* $D = diag\{\mathbf{s}_1^T\Sigma\mathbf{s}_1 + 1, \ldots, \mathbf{s}_{t-1}^T\Sigma\mathbf{s}_{t-1} + 1\} \in \mathbb{R}_+^{t-1\times t-1}$. *Note that* $\mathbf{s}_i^T$ *is the* i^{th} *row of* S, *and* $\mathbf{I}_{t-1} \in \mathbb{R}^{t-1\times t-1}$ *is an identity matrix.*

In terms of density function, the distribution in equation 5.2 is such that

$$p(\boldsymbol{\theta}|\mathbf{X}_{t-1}, \mathbf{r}_{t-1}) = \phi_p(\boldsymbol{\theta} - \boldsymbol{\xi}; \Sigma)\frac{\prod_{s=1}^{t-1}\Phi((2r_{a_s,s} - 1)\mathbf{x}_{a_s}^{\mathrm{T}}\boldsymbol{\theta})}{\Phi_{t-1}(S\boldsymbol{\xi}; S\Sigma S^{\mathrm{T}} + \mathbf{I}_{t-1})}, \tag{5.3}$$

where $\Phi(\cdot)$, $\Phi(\cdot;\cdot)$ and $\phi_p(\cdot;\cdot)$ have the usual meaning. Notice that the product in the numerator is the probit likelihood for the observed rewards.

Thompson Sampling can elegantly exploit such a conjugacy property to acquire i.i.d. samples directly from the $\mathrm{SUN}_{p,t-1}$. The best way to do so is by exploiting a convenient property of the unified skew-normal which states that, for every p and $t-1$, the distribution in (5.3) can be rewritten as a linear combination of a p-variate normal distribution, and a $t-1$-variate truncated normal distribution. Then, it is sufficient to draw one sample from the normal, one from the truncated normal and compute a linear combination among these two samples. Following Durante (2019), we call such a sampler as *independent additive sampler*. In particular, this technique relies on the following corollary, which is a direct modification of the result in Azzalini and Capitanio (2014), chapter 7. Recalling that, given a generic vector $\boldsymbol{\eta} \in \mathbb{R}^n$, the term $\mathcal{N}_{\boldsymbol{\eta}}^+(\mathbf{a}, A)$ denotes a generic n-variate truncated normal with mean $\mathbf{a}$, correlation matrix A and truncation from below $\boldsymbol{\eta}$, it holds that:

Corollary 1. *Let* $(\boldsymbol{\theta} \mid \mathbf{X}_{t-1}, \mathbf{r}_{t-1}) \sim p(\boldsymbol{\theta} \mid \mathbf{X}_{t-1}, \mathbf{r}_{t-1})$ *as in equation* (5.2). *Then*

$$(\boldsymbol{\theta} \mid \mathbf{X}_{t-1}, \mathbf{r}_{t-1}) \overset{\mathrm{d}}{=} \boldsymbol{\xi} + \Sigma\{V_0 + S^T(S\Sigma S^T + \mathbf{I}_{t-1})^{-1}D^{1/2}V_1)\} \tag{5.4}$$

where

(i) $V_0 \sim \mathcal{N}_p(\mathbf{0}_p, \Sigma^{-1} - S^T(S\Sigma S^T + \mathbf{I}_{t-1})^{-1}S)$ *, a p-variate normal distribution*

(ii) $V_1 \sim \mathcal{N}^+_{-D^{-1/2}S\boldsymbol{\xi}}(\mathbf{0}_{t-1}, D^{-1/2}(S\Sigma S^T + \mathbf{I}_{t-1})D^{-1/2})$, *a* $(t-1)$*-variate truncated normal, with truncation below* $-D^{-1/2}S\boldsymbol{\xi}$

Finally, given the results listed so far, we are able to implement point 5 in algorithm 10 by simply exploiting the conjugacy property of the probit regression with normal priors. Algorithm 12 reports the technical details of the *independent additive sampler*.

Algorithm 12 Independent Additive Sampler for the unified skew-normal posterior (Durante, 2019) with $M = 1$ pulls

Prior Input: $\boldsymbol{\theta} \sim \mathcal{N}_p(\boldsymbol{\xi}, \Sigma)$
Fix $t \in \{1, \dots T\}$
$\mathbb{H}_{t-1} = (\mathbf{X}_{t-1}, \mathbf{r}_{t-1}) \in \mathbb{R}^{t-1} \times \mathbb{R}^{(t-1)\times p}$ ▷ Observed History
if $t = 1$ **then** ▷ Sample from the Prior
    $\forall g \in \{1, \dots, G\}$ sample $\boldsymbol{\theta}_t^{(g)} \sim \mathcal{N}_p(\boldsymbol{\xi}, \Sigma)$
else
    $S \leftarrow \text{diag}\{2r_{a_1,1} - 1, \dots, 2r_{a_{t-1},t-1} - 1\}\mathbf{X}_{t-1}$
    $D \leftarrow \text{diag}\{\mathbf{s}_{a_1}^{\mathrm{T}}\Sigma\mathbf{s}_{a_1} + 1, \dots, \mathbf{s}_{a_{t-1}}^{\mathrm{T}}\Sigma\mathbf{s}_{a_{t-1}} + 1\}$
    for all $g \in \{1, \dots, G\}$ **do**
        Sample $V_0^{(g)} \sim \mathcal{N}_p(\mathbf{0}_p, \Sigma^{-1} - S^{\mathrm{T}}(S\Sigma S^{\mathrm{T}} + \mathbf{I}_{t-1})^{-1}S)$
        Sample $V_1^{(g)} \sim \mathcal{N}^{+}_{-D^{-1/2}S\boldsymbol{\xi}}(\mathbf{0}_{t-1}, D^{-1/2}(S\Sigma S^{\mathrm{T}} + \mathbf{I}_{t-1})D^{-1/2})$
        $\boldsymbol{\theta}_t^{(g)} = \boldsymbol{\xi} + \Sigma\{V_0^{(g)} + S(S\Sigma S^{\mathrm{T}} + \mathbf{I}_{t-1})^{-1}D^{-1/2}V_1^{(g)}\}$
    end for

Given the availability of a closed form posterior, we can easily replicate the same reasoning presented for the simple beta-binomial Thompson Sampling. By increasing the number of observations (that is, the number of rows in $\mathbf{X}_{t-1}$), the variance of the draws from the posterior decreases significantly. In other words, the posterior concentrates around a certain value for each parameter. Figure 9 mirrors the comparison scheme proposed in figure 4. Assume that, in our fractional factorial framework, we have a 2-arms test-bed, with a single binary variable x_a that equals 1 if the selected arm is 1, and 0 if not. The probit regression is then

$$\mathbb{P}(r_{a_t,t} = 1 \mid \alpha, \beta) = \Phi(\alpha + \beta x_{a_t}).$$

In other words, the probability of success of arm 1 is $\Phi(\alpha + \beta)$, and the one of arm 2 is $\Phi(\alpha)$, and the generic s^{th} row of the matrix $\mathbf{X}_{t-1}$ is the vector $(1, x_{a_s})$. For the sake of simplicity, rows in $\mathbf{X}_{t-1}$ have been generated randomly, with an equal probability of selecting arm 1 and arm 2. In particular, in the left panel matrix $\mathbf{X}_{t-1}$ is made up by 10 rows only, whilst in the right one by 100. We have set $\alpha = -0.5$ and $\beta = 0.5$ as

true values for the parameters, so that $\Phi(\alpha + \beta) = 0.5$ and $\Phi(\alpha) \approx 0.3$. Notice that the draws from the posterior under few observations are widespread across the whole probability space. In other words, Thompson Sampling estimates that arm 1 and arm 2 have an almost equal probability of being the best arm. The situation changes when the number of observations increases: the draws start concentrating around the true values for the probability of success, and Thompson Sampling starts allocating its pulls with a probability vector that is unbalanced towards arm 1.

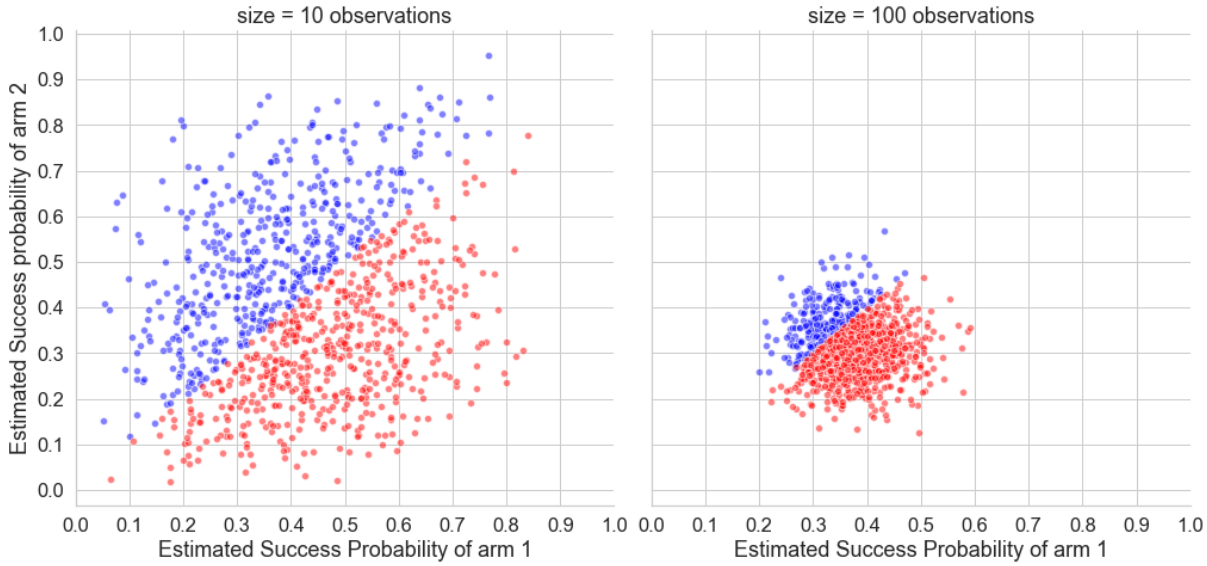


Figure 9: Success probabilities of arm 1 (horizontal ax) and arm 2 (vertical ax) resulting form 1000 draws from the posterior distribution in 5.4 with $t-1$=10 (on the left), and $t-1 = 100$ on the right. Real success probability come from the probit regression $\mathbb{P}(r_{a_t,t} = 1 \mid \alpha, \beta) = \Phi(\alpha + \beta x_a)$, with $x_{a_t} = 1$ if the selected arm is 1, and 0 otherwise. True values of α and β are set to -0.5 and 0.5, so that $\mathbb{P}(r_{1,t} = 1 \mid \alpha, \beta) = \Phi(\alpha+\beta) = 0.5$ and $\mathbb{P}(r_{2,t} = 1 \mid \alpha, \beta) = \Phi(\alpha) \approx 0.3$. Red points indicate that the draws from the posterior estimate a higher probability of success for arm a_1, blue points the opposite. The sequence of pulls is drawn at random, with an equal probability of choosing arm 1 or arm 2.

As already stated before, algorithm 12 is a tool that requires a twofold sampling: one sample from a p-variate normal, and another from a $(t-1)$-variate truncated normal. Besides the conjugacy property, the main advantage of the algorithm is that the resulting samples are independent by definition, which is not guaranteed with other sampling methods (Durante, 2019). A careful reader might have already noticed how this type of sampling procedure strongly depends on the dimensions of $\mathbf{X}_{t-1}$. Indeed, while p is constant throughout the bandit experiment, $t-1$ keeps on increasing by definition. If we were

to apply such an algorithm to a real business scenario, we would face significant issues right form the start of our experiment (roughly when $t-1 \geqslant 1000$). The reason is that, though theoretically unquestionable, sampling from a generic n-variate truncated normal becomes either very slow when n is small-to-medium, and computationally intractable when n is large. Indeed, the most efficient way to sample i.i.d. values from a multivariate truncated normal is to use the accept-reject algorithm of Botev (2017)[24], which becomes slow in high dimensions. If we were to estimate the posterior a few times only, this time requirement would be bearable. However, the bandit experiment requires to sample from the posterior after every trial t. As we will see in the results section below, this sampling effort does not justify the performance increase that derives from the conjugacy property, especially when compared to the standard Gibbs sampling technique of Albert and Chib (1993). In the following subsection we present a primer solution to this problem, which still relies on the unified skew-normal posterior but sequentially approximates it under large t. In this way, the resulting algorithm ensures both a high performance in terms of regret and a significant reduction in the completion time.

5.2 Sequential Monte Carlo

A useful solution to the Bayesian updating in problems where the data stream-in at different times is to adapt ideas from Sequential Monte Carlo (Doucet et al., 2010). In particular, our goal is to sample from the posterior of $\boldsymbol{\theta}$ at a generic time t by re-drawing the previously produced samples at $t-1$ with probabilities depending on the information provided by the new data available at t. Let again $\mathbf{X}_{t-1}$ be the feature matrix at the beginning of time t (before the arm is selected), $\mathbf{r}_{t-1}$ the associated response vector, $\mathbf{x}_{a_t}$ the feature vector of the arm selected at time t and $r_{a_t,t}$ its usual response in the case of $M=1$. Then, $\mathbf{r}_t = (\mathbf{r}_{t-1}^{\mathrm{T}}, r_{a_t,t})^{\mathrm{T}}$ and $\mathbf{X}_t = [\mathbf{X}_{t-1}^{\mathrm{T}}, \mathbf{x}_{a_t}]^{\mathrm{T}}$. Suppose now that at time t we have drawn $(\boldsymbol{\theta}_t^{(g)})_{g=1}^{G}$ i.i.d. samples from $p(\boldsymbol{\theta} \mid \mathbf{X}_{t-1}, \mathbf{r}_{t-1})$. These samples are often called

[24] Botev (2017) algorithm can be applied simply by calling in R the package `TruncatedNormal`. However, the equivalent version of such algorithm in Python does not exists yet, so that it is only possible to sample from the univariate truncated normal. As the codes for this thesis are deveoped in Python, the solution applied consisted on calling the R commands from Python using the package `rpy2`. See the codes section in the appendix.

particles in the Sequential Monte Carlo literature. Note that these particles are always drawn *in order to select* arm a_t, which means that $r_{a_t,t}$ has not yet been observed. Then, the posterior at time t (prior to the observation of $r_{a_t,t}$) can be approximated via

$$\hat{p}_G(\boldsymbol{\theta} \mid \mathbf{X}_{t-1}, \mathbf{r}_{t-1}) = \frac{1}{G} \sum_{g=1}^{G} \delta_{\boldsymbol{\theta}_t^{(g)}}, \tag{5.5}$$

where $\delta_{\theta^{(g)}}$ is the delta-Dirac mass located at $\boldsymbol{\theta}^{(g)}$. As always, our goal is to update the posterior $p(\boldsymbol{\theta} \mid \mathbf{X}_{t-1}, \mathbf{r}_{t-1})$ with the reward $r_{a_t,t}$ that follows form the selection of arm a_t. Adapting Sequential Monte Carlo ideas, one approach is to use the distribution in equation (5.5) as an approximate prior when applying Bayes' rule. In particular, the update that comes from the choice of a_t and the observation of $r_{a_t,t}$ can be written as

$$\begin{aligned} p(\boldsymbol{\theta} \mid \mathbf{X}_t, \mathbf{r}_t) &\propto p(\mathbf{r}_t \mid \boldsymbol{\theta}, \mathbf{X}_t) p(\boldsymbol{\theta}) \\ &\propto p(\mathbf{r}_{t-1}, r_{a_t,t} \mid \boldsymbol{\theta}, \mathbf{X}_{t-1}, \mathbf{x}_{a_t}^{\mathrm{T}}) p(\boldsymbol{\theta}) \\ &\propto p(r_{a_t,t}, \mid \mathbf{x}_{a_t}^{\mathrm{T}}, \boldsymbol{\theta}) p(\mathbf{r}_{t-1} \mid \boldsymbol{\theta}, \mathbf{X}_{t-1}) p(\boldsymbol{\theta}) \\ &\propto p(r_{a_t,t}, \mid \mathbf{x}_{a_t}^{\mathrm{T}}, \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \mathbf{X}_{t-1}, \mathbf{r}_{t-1}). \end{aligned} \tag{5.6}$$

Notice that the first line in the equation follows form the fact that $p(\boldsymbol{\theta} \mid \mathbf{X}_t) = p(\boldsymbol{\theta})$ (the parameters in the probit regression do not depend on the features of the selected arms). The third line in the equation instead follows from the fact that the reward at time t is conditionally independent from both the reward at previous times and the previously selected arms. In other words, equation (5.6) points out that the posterior at the end of trial t (after $r_{a_t,t}$ has been observed) can be obtained by updating the posterior at the beginning of time t with the associated reward $r_{a_t,t}$ itself. But then, as the rewards are Bernoulli distributed, it holds that

$$\begin{aligned} p(r_{a_t,t} \mid \mathbf{x}_{a_t}^{\mathrm{T}}, \boldsymbol{\theta}) &= \Phi(\mathbf{x}_{a_t}^{\mathrm{T}} \boldsymbol{\theta})^{r_{a_t,t}} (1 - \Phi(\mathbf{x}_{a_t}^{\mathrm{T}} \boldsymbol{\theta}))^{1 - r_{a_t,t}} \\ &= \Phi((2 r_{a_t,t} - 1) \mathbf{x}_{a_t}^{\mathrm{T}} \boldsymbol{\theta}), \end{aligned}$$

and, following equation (5.5), the posterior at the beginning of t can be approximated via Monte Carlo as

$$p(\boldsymbol{\theta} \mid \mathbf{r}_{t-1}, \mathbf{X}_{t-1}) \approx \hat{p}_G(\boldsymbol{\theta} \mid \mathbf{r}_{t-1}, \mathbf{X}_{t-1}) = \frac{1}{G} \sum_{g=1}^{G} \delta_{\boldsymbol{\theta}_t^{(g)}}.$$

These two results can be easily exploited to approximate the updated posterior distribution (or, equivalently, the posterior distribution at the beginning of time $t+1$) in equation (5.6). Specifically,

$$\begin{aligned} p(\boldsymbol{\theta} \mid \mathbf{r}_t, \mathbf{X}_t) &\propto \sum_{g=1}^{G} \delta_{\theta_t^{(g)}} \Phi((2r_{a_t,t} - 1)\mathbf{x}_{a_t}^{\mathrm{T}} \boldsymbol{\theta}_t^{(g)})) \\ &\propto \sum_{g=1}^{G} \delta_{\theta_t^{(g)}} \frac{\Phi((2r_{a_t,t} - 1)\mathbf{x}_{a_t}^{\mathrm{T}} \boldsymbol{\theta}_t^{(g)})}{\sum_{g=1}^{G} \Phi((2r_{a_t,t} - 1)\mathbf{x}_{a_t}^{\mathrm{T}} \boldsymbol{\theta}_t^{(g)})} \\ &\propto \sum_{g=1}^{G} \delta_{\theta_t^{(g)}} \pi_t^{(g)}, \end{aligned} \tag{5.7}$$

where for every g, $\pi_t^{(g)} = \frac{\Phi((2r_{a_t t}-1)\mathbf{x}_{a_t}^{\mathrm{T}} \boldsymbol{\theta}_t^{(g)})}{\sum_{g=1}^{G} \Phi((2r_{a_t,t}-1)\mathbf{x}_{a_t}^{\mathrm{T}} \boldsymbol{\theta}_t^{(g)})}$. Note that the approximate proportionality that results from the sum of particles allows us to normalize the values that come from the likelihood (the second line in the equation above). In other words, we are re-writing the posterior we have to sample from at the beginning of time $t+1$ as a sum of the particles drawn at t, weighted by the likelihood. As a consequence, in order to choose a_{t+1}, the samples $(\boldsymbol{\theta}_{t+1}^{(g)})_{g=1}^{G}$ from the posterior at the beginning of time $t+1$ can either be directly sampled from $p(\boldsymbol{\theta} \mid \mathbf{X}_t, \mathbf{r}_t)$, or from the approximate posterior in equation (5.7). In this last case, it will be sufficient to compute the importance weights $\pi_t^{(g)}$ and use them to resample the particles $\boldsymbol{\theta}_t^{(g)}$ drawn from $p(\boldsymbol{\theta} \mid \mathbf{X}_{t-1}, \mathbf{r}_{t-1})$, which act, here, as an approximate prior. Such a reasoning in particular assigns higher weights to the particles for which the value of $\Phi((2r_{a_t,t} - 1)\mathbf{x}_{a_t}^{\mathrm{T}} \boldsymbol{\theta}_t^{(g)})$ is high.

The fundamental advantage of this approach is that is requires only the calculation of the sampling weights by evaluating cumulative distribution functions of univariate normals. This benefit overcomes the previous computational bottlenecks of sampling from t-variate truncated normals. However, it relies on discrete approximations of the target

distributions which may deteriorate as the resampling technique progresses with t (Zhou et al., 2018). To partially address this issue we combine this scalable routine with the proposed i.i.d. sampler from the $SUN_{p,t-1}$ posterior in algorithm 12. In particular, we propose to set a cut-off period $\bar{t}$ at the beginning of the experiment, such that for the time periods $t < \bar{t}$ the algorithm samples directly from the $\text{SUN}_{p,t-1}$ via algorithm 12, and then it applies the re-sampling approach to the draws produced by the i.i.d. sampler at $\bar{t}$ until the end of the experiment. In such a way, we consider an i.i.d sampler from the exact posterior during the first trials, and then switch to an approximate sampler when the posterior becomes more concentrated and, typically, easier to approximate in a reliable way. Algorithm 13 summarizes the procedures described so far.

Algorithm 13 Sequential Monte Carlo Sampler

Prior Input: $\boldsymbol{\theta} \sim \mathcal{N}_p(\boldsymbol{\xi}, \Sigma)$
Fix $t, \bar{t} \in \{1, \dots T\}$
$\mathbb{H}_{t-1} = (\mathbf{X}_{t-1}, \mathbf{r}_{t-1}) \in \mathbb{R}^{t-1} \times \mathbb{R}^{(t-1)\times p}$ ▷ Observed History
if $t < \bar{t}$ **then**
    $\forall g \in \{1, \dots, G\}$ sample $\boldsymbol{\theta}_t^{(g)} \sim \text{SUN}_{p,t-1}$ via algorithm 12
else
    Compute $\boldsymbol{\pi}_t = (\pi_t^{(g)})_{g=1}^G$, where $\pi_t^{(g)} = \frac{\Phi((2r_{a_t,t}-1)\mathbf{x}_{a_t}^{\mathrm{T}}\boldsymbol{\theta}_t^{(g)})}{\sum_{g=1}^G \Phi((2r_{a_t,t}-1)\mathbf{x}_{a_t}^{\mathrm{T}}\boldsymbol{\theta}_t^{(g)})}$
    Draw G samples with replacement from $(\boldsymbol{\theta}_t^{(g)})_{g=1}^G$ using $\boldsymbol{\pi}_t$ as sampling weights

Note that, in the case of multiple pulls within the same trial period t, the sampling weights can be easily computed as

$$\pi_t^{(g)} = \frac{\prod_{s=1}^M \Phi((2r_{a_t^s,t} - 1)\mathbf{x}_{a_t^s}^{\mathrm{T}}\boldsymbol{\theta}_t^{(g)})}{\sum_{g=1}^G \prod_{s=1}^M \Phi((2r_{a_t^s,t} - 1)\mathbf{x}_{a_t^s}^{\mathrm{T}}\boldsymbol{\theta}_t^{(g)})}, \tag{5.8}$$

where M is the number of observations in the batch at time $t-1$ and $(r_{a_t^s,t})_{s=1}^M$ is the reward sequence associated to the selection of arms $(a_t^s)_{s=1}^M$. The essential advantage of such a procedure is its speed and its computational simplicity. The main disadvantage however is that the best resampling cutoff needs to be carefully tuned in order to balance accuracy in the approximation and computational scalability. The following section is dedicated to the empirical comparison of all the bandit algorithms listed so far in a simulated environment, which allows to clearly highlight the main advantages and drawbacks of algorithm 13.

5.3 Empirical Comparison

Let our arms in $\mathcal{A}$ have a factorial structure. In particular, suppose that they are the result of a combination of three different features: one with 2, one with 3 and one with 5 levels. Then, the resulting number of arms is 30, with a total number of parameters in equation (4.9) equal to 8 (constant included). Following Scott (2010), we independently generate the true values for the parameters at random from a normal distribution at the beginning of each simulation. In particular, the constant follows a normal distribution with mean $\Phi^{-1}(0.05)$ and variance equal to 0.1, while all the other parameters are normally distributed with mean 0 and variance 0.25. Then, the resulting mean success probability of every arm is around 0.05, with a standard deviation of around 0.8 on the probit scale. Figure 10 reproduces the true success probability schedule for one simulated environment with the features listed so far.

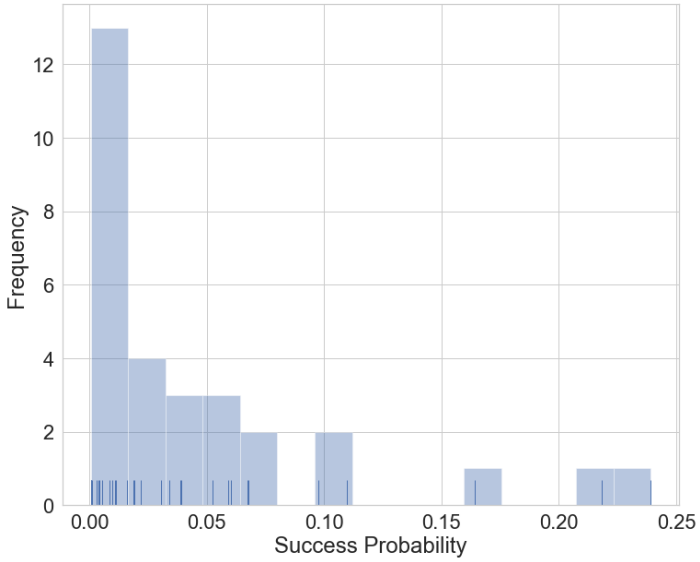
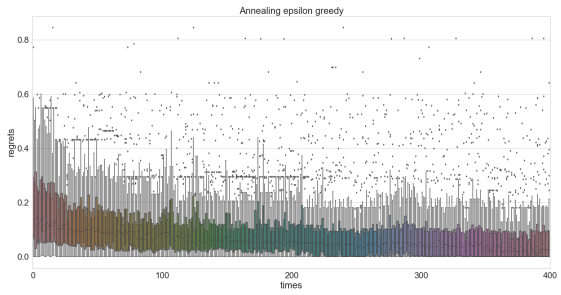


Figure 10: True success probabilities of the 30 arms in one simulated environment of the fractional factorial structure.

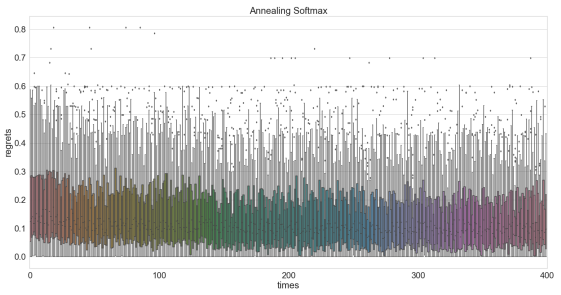
In general, such a configuration produces one or two arms that clearly stand out in terms of true success probability, whereas the others have a success rate that concentrates around 0.05. Figure 11 reports the results of 40 simulations carried over the 30-arms test-bed described above, with $M = 1$ pulls allocated each time t. Specifically, every subfigure consists of a series of box-plots (one for each t) which allow to monitor the behavior of the algorithms under every simulated environment. In particular, panel (a)

and (b) test the performance of the annealing ε-greedy in algorithm 2 and the annealing Softmax in algorithm 4 with $g_\varepsilon(t) = g_\tau(t) = 1/\sqrt{t}$. Panel (c) instead shows the results obtained by the UCB Tuned (algorithm 6), and panel (d) the ones resulting from the beta-binomial *RPM* with 5000 samples drawn from the posterior at each trial and with uniform prior over the probability of success of every arm. Finally, panels (e) to (f) illustrate the Fractional Factorial Thompson Sampling (*FFTS* for short) in algorithm 10 with the different methods listed to sample from the posterior and a $\mathcal{N}(0, 1)$ prior over each parameter: Albert and Chib (1993) Gibbs sampler (panel (e), algorithm 11), Durante (2019) Independent Additive Sampler (panel (f), algorithm 12), and Sequential Monte Carlo sampler with cutoffs at $\bar{t} = 0$ and at $\bar{t} = 200$ (algorithm 13). Each algorithm computes the best arm probability vector $\mathbf{w}_t$ by relying on 5000 samples at each trial (with a burn-in of 2000 in the Gibbs sampler).

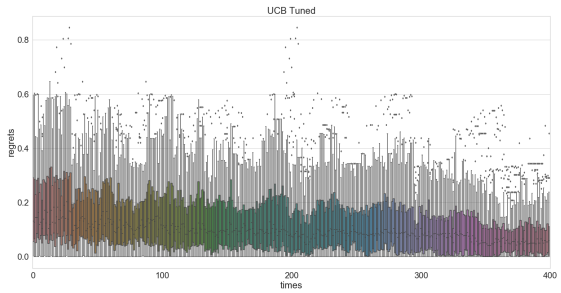
The graphs clearly point out two aspects. First, accounting for a fractional factorial structure in the bandit environment improves the performance in terms of per-period regret. The reason is simple: by including external features we are borrowing information across the different arms, rather than estimating their success probabilities separately (8 parameters to estimate as opposed to 30 means). In particular, panel (a) to (d) exhibit box-plots with a rather large width and long whiskers throughout the whole horizon. This happens mainly due to the fact that $T = 400$ and $K = 30$ do not allow for a proper assessment of the probability of success of every arm, and thus for a proper exploitation. On the other hand, whiskers at t close to the end of the horizon in panels (e), (f) and (h) are shorter and more compressed around zero, with only a few outliers above 0.2. In other words, in more than 3/4 of the simulated environments, the fractional factorial Thompson Sampling achieves a per-period regret below 0.1. This means that, when it does not correctly identify the best arm, it still allocates its pulls on a sub-optimal arm whose success probability is slightly lower than the actual best arm.



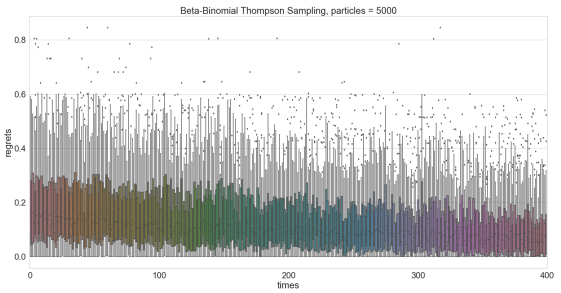
(a) Annealing ε-greedy



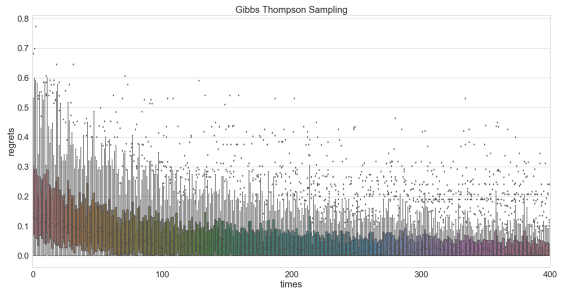
(b) Annealing Softmax



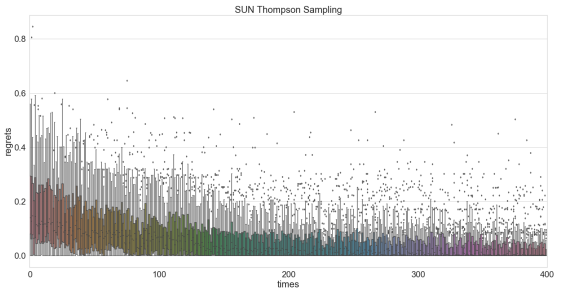
(c) UCB Tuned



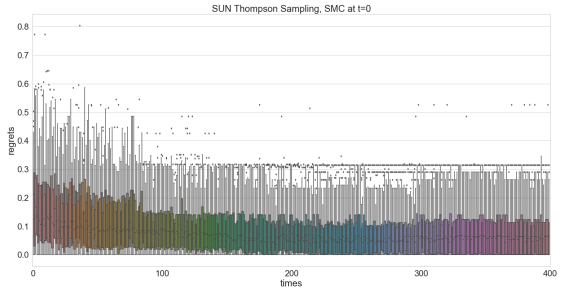
(d) Beta-Binomial Thompson Sampling



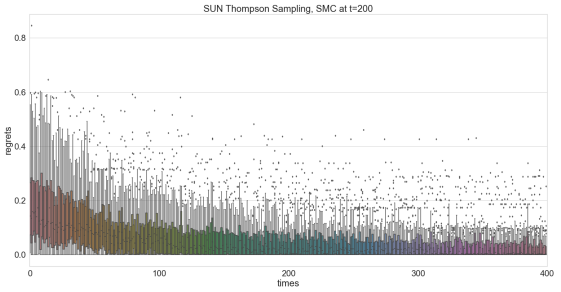
(e) FFTS via Gibbs Sampler



(f) FFTS via Independent Additive Sampler



(g) FFTS via Sequential Monte Carlo, cutoff at $\bar{t} = 0$



(h) FFTS via Sequential Monte Carlo, cutoff at $\bar{t} = 200$

Figure 11: Per-period regrets over 40 simulations of the 30-arms test-bed with normally distributed parameters.

The second fundamental aspect of figure 11 concerns the difference between the performance of the four fractional factorial Thompson Sampling algorithms. As could be

expected, sampling directly form a unified skew-normal posterior throughout the whole horizon is the best choice: as t increases, the median per-period regret quickly reaches 0, while the length of the box-plots decreases progressively. Albert and Chib (1993) Gibbs sampler has a similar performance, albeit a slight higher number of outliers and longer whiskers. This is due to the fact that, unlike the proposed i.i.d. sampler from the exact posterior, the Gibbs algorithm is an MCMC which may suffer from mixing and convergence issues affecting the quality of the inference. Finally, in the case of panels (g) and (h) it is clear how applying the approximate re-sampling strategy from the beginning leads to a poor performance. On the other hand, a resampling cutoff at $\bar{t} = 200$ allows for a proper learning of the parameters in the probit regression, so that the particles drawn from the posterior are precise enough to identify the best arm in the majority of the environments. This is due to the fact that the particles at $\bar{t} = 200$ are i.i.d. samples from the exact unified skew-normal posterior, whereas with a cutoff at $\bar{t} = 0$, the particles at $t = 200$ come from a sequential re-sampling of draws from the prior distribution. While the median in this last case is slightly above 0, the time needed to conclude the bandit experiment significantly decreases. This generates a sort of *trade-off*: while the quality of the particles coming from the exact SUN additive sampler increases with $\bar{t}$, so it does the computational burden required to sample from $\bar{t}$-variate truncated normals under such an i.i.d. additive sampler. Then, if $\bar{t}$ is set early, the algorithm is subject to a high risk of getting stuck on suboptimal arms. If instead $\bar{t}$ is high, such a risk is lower, but the associated computational effort significantly reduces its applicability over real business scenarios. Finally, if $\bar{t}$ is too high, the algorithm may even fail to reach the end of the horizon.

Figure 12 summarizes this trade-off by showing the cumulative regret density resulting from the 40 simulations (left panel) and the average completion time in seconds (right panel) of the four sampling methods for the posterior in the fractional factorial Thompson Sampling. Notice that the mean of each distribution is approximately the same (roughly 25 conversions over 400 trials). However, the mass of the pure Sequential Monte Carlo is widespread across the real line. Though being efficient in term of elapsed time, such a low

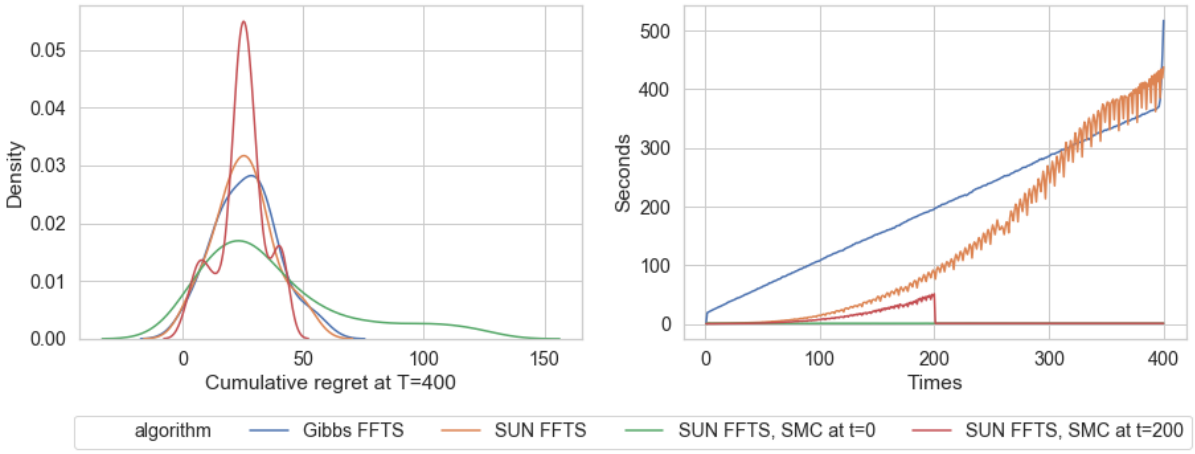


Figure 12: Cumulative regret density at the end of the bandit experiment ($T = 400$) and average completion time across 40 simulated environment of the 4 fractional factorial algorithms in figure 11

resampling cutoff leads to more uncertain results. The most interesting aspects emerge however if we look at the other three methods. The cumulative regret distribution resulting from Albert and Chib (1993) Gibbs sampler is rather similar to the one resulting from the independent additive sampler, while their associated completion time in seconds is significantly different. In particular, the elapsed time of the Gibbs sampler increases almost linearly with t. The reason is rather simple: at every new trial, the number of univariate truncated normals to sample from increases by one, but the overall computational complexity of the sampling procedure remains equal[25]. The independent additive sampler instead performs surprisingly better than the Gibbs sampler within the first trials, but becomes inefficient after t reaches 300. Part of this initial efficiency comes from the fact that we have set a burn-in of 2000 particles for the Gibbs Sampler, which increase its sampling cost in terms of time. Moreover, algorithm 13 avoids the data augmentation step typical of the Gibbs sampler, and the computational complexity of sampling from p-variate normal is limited when p is small-to-medium. However, if the experiment have had a longer horizon, the overall advantages of the pure independent additive sampler would have faded quickly. Finally, as could be expected, the average completion time of the independent additive sampler with the Sequential Monte Carlo starting at $\bar{t} = 200$

[25] The time required to draw one sample from the univariate truncated normal with the Python command `scipy.stats.truncnorm.rvs()` is always the same irrespective of the given parameters.

quickly reaches 0 seconds per trial after the cutoff[26]. This computational efficiency still comes with a risk of getting stuck on sub-optimal arms, as can be seen from the three modes of the cumulative regret distribution. If by chance the algorithm fails to correctly learn the true best arm before the resampling cutoff, then it will resample from "bad" particles only and thus would not be able to later refine the learning properly. Indeed, Sequential Monte Carlo updating has the inconvenient issue that, after a while, the particles in the process all identify one arm as the best one, irrespective of the associated reward. But then, the further the trial $t > \bar{t}$ is from $\bar{t}$, the lower the chances are that the algorithm explores different arms (which is the core idea behind bandit algorithms). In other words, repeated re-sampling makes the algorithm blind. To see this, look at the behavior of the outliers and of the whiskers in panel (g): after a while, the per-period regret becomes constant. This means that all the re-sampled particles identify one or two arms as best arm and assign to the others a probability of being selected equal to zero, but such identification does not change in the case of negative responses. Finally, a Sequential Monte Carlo approach has the further issue that there is no clear rule behind the number of particles to use in the resampling process (Zhou et al., 2018), and the optimal resampling cutoff $\bar{t}$ needs proper tuning.

[26] Note that in this particular experiment, the average seconds per trial of this last algorithm are lower than the corresponding pure independent additive sampler even before the resampling cutoff. However, as the number of simulations is somehow low and the algorithms are identical before $\bar{t}$, the increased efficiency must be attributed to chance alone.

6 Conclusions

This thesis has reviewed the major algorithms that fall within the category of bandit problems, which are an effective solution to the widespread exploration-exploitation dilemma. The first part of the thesis has been devoted to the presentation and application of the foundational frequentist bandits: the ε-greedy, the Softmax and the UCB. The second part instead tackled the problem from a Bayesian perspective and main aspects of the Thompson Sampling algorithm (equivalently referred as *randomized probability matching*), both in its simple beta-binomial formulation, and in its fractional factorial one. In particular, this last formulation, first presented in Scott (2010), has been improved by applying the novel conjugacy result in Durante (2019), which allows for i.i.d. sampling from the exact posterior without data augmentations. All the above algorithms have been tested over a K-armed stochastic bandit environment with Bernoulli distributed rewards. In all simulations, arms were assumed to have a fixed and time-invariant probability of success. Such a situation is rather uncommon in real business applications, but still, we could assume it to be true if the experiments were carried within a short period of time. After several comparisons and testing over different bandit environments, we can conclude that there is *a priori* no algorithm that works more efficiently than others, except the fractional factorial one (which however exploits more information on the arms). In fact, the efficiency of every algorithm is always strictly context related: as the true success probability distribution changes, so does the relative performance of the algorithms. In this respect, the most emblematic difference can be found on the the per-period regret of the Softmax algorithm in figure 3 and in figure 11, panel (b). When the process according to which we generate the true probabilities of success changed, the Softmax shifts from the best performing to the worst one, even though the number of arms remains equal. Thus, the application of a bandit algorithm in the real business world should require, whenever possible, a careful offline tuning period.

In the last part of the analysis, we have considered the fractional factorial Thompson Sampling. While being theoretically convenient, such an algorithm has had a limited applicability due to its computational bottlenecks. To solve them, we have first applied

the unified skew-normal posterior result in Durante (2019), and then a Sequential Monte Carlo approach. The resulting algorithm, which balances the theoretical advantages of the conjugacy property with the computational speed of the resampling algorithms is a first attempt to expand the applicability of Thompson Sampling within the context of real business scenarios, and further work is needed to improve scalability without paying excessive costs in terms of accuracy. In this respect, future research directions will include a refinement of the Sequential Monte Carlo procedure, and an approximation of the unified skew-normal with Variational Bayes methods.

References

Agrawal, S. and Goyal, N. (2011), Analysis of thompson sampling for the multi-armed bandit problem. arXiv:1111.1797.

Albert, J. H. and Chib, S. (1993), 'Bayesian analysis of binary and polychotomous response data', *Journal of the American Statistical Association* **88**(422), 669–679.

Arellano-Valle, R. B. and Azzalini, A. (2006), 'On the unification of families of skew-normal distributions', *Scandinavian Journal of Statistics* **33**(3), 561–574.

Audibert, J.-Y., Munos, R. and Szepesvari, C. (2009), 'Exploration-exploitation trade-off using variance estimates in multi-armed bandits', *Theoretical Computer Science* **410**(19), 1876–1902.

Auer, P., Fischer, P. and Kivinen, J. (2002), Finite-time analysis of the multiarmed bandit problem, *in* 'Machine Learning', Vol. 47, pp. 235–256.

Azzalini, A. and Capitanio, A. (2014), *The Skew-Normal and Related Families*, Cambridge University Press.

Berry, D. A., Chen, R. W., Zame, A., Heath, D. C. and Shepp, L. A. (1997), 'Bandit problems with infinitely many arms', *The Annals of Statistics* **25**(5), 2103–2116.

Botev, Z. I. (2017), 'The normal law under linear restrictions: simulation and estimation via minimax tilting', *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **79**(1), 125–148.

Bubeck, S. and Cesa-Bianchi, N. (2012), *Regret Analysis of Stochastic and Non-Stochastic Multi-Armed Bandit Problems*, NOW the Essence of Knowledge.

Bush, R. R. and Mosteller, F. (1953), 'A stochastic model with applications to learning', *The Annals of Mathematical Statistics* **24**(4), 559–585.

Cesa-Bianchi, N. and Fischer, P. (1998), Finite-time regret bounds for the multiarmed bandit problem, *in* 'In 5th International Conference on Machine Learning', pp. 100–108.

Cesa-Bianchi, N. and Lugosi, G. (2006), *Prediction, Learning, and Games*, Cambridge University Press.

Chapelle, O. and Li, L. (2011), An empirical evaluation of thompson sampling, *in* 'Proceedings of the 24th International Conference on Neural Information Processing Systems', NIPS'11, Curran Associates Inc., pp. 2249–2257.

Chopin, N. and Ridgway, J. (2017), 'Leave pima indians alone: Binary regression as a benchmark for bayesian computation', *Statistical Science* **32**(1), 64–87.

Doucet, A., Freitas, N. D. and Gordon, N. (2010), *Sequential Monte Carlo Methods in Practice*, Springer.

Durante, D. (2019), 'Conjugate bayes for probit regression via unified skew-normals', *Biometrika (in press.)* .

Frost, R. (1916), The road not taken, *in* 'Mountain Interval', Henry Holt.

Gelfand, A. E. and Smith, A. F. (1990), 'Sampling based approaches to calculating marginal densities', *Journal of the American Statistical Association* **85**(410), 398–409.

Gittins, J. C. (1989), *Multi-armed Bandit Allocation Indices*, Wiley.

Hoffman, M. D. and Gelman, A. (2014), 'The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo', *Journal of Machine Learning Research* **15**, 1593–1623.

Kaufmann, E., Cappe, O. and Garivier, A. (2012), On bayesian upper confidence bounds for bandit problems, *in* N. D. Lawrence and M. Girolami, eds, 'Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics', Vol. 22 of *Proceedings of Machine Learning Research*, PMLR, pp. 592–600.

Kuleshov, V. and Precup, D. (2000), 'Algorithms for the multi-armed bandit problem', *Journal of Machine Learning Research* **1**(1), 1–48.

Lai, T. and Robbins, H. (1985), 'Asymptotically efficient adaptive allocation rules', *Advances in Applied Mathematics* **6**(1), 4–22.

Lattimore, T. and Szepesvari, C. (2018), *Bandit Algorithms*, Cambridge University Press.

Li, L., Chu, W., Langford, J. and Schapire, R. E. (2010), A contextual-bandit approach to personalized news article recommendation, *in* 'Proceedings of the 19th international conference on World wide web - WWW 10'.

Roberts, G. O. and Rosenthal, J. S. (2001), 'Optimal scaling for various metropolis-hastings algorithms', *Statistical Science* **16**(4), 351–367.

Russo, D. J., Roy, B. V., Kazerouni, A., Osband, I. and Wen, Z. (2018), 'A tutorial on thompson sampling', *Foundations and Trends in Machine Learning* **11**(1), 1–96.

Scott, S. L. (2010), 'A modern bayesian look at the multi-armed bandit', *Applied Stochastic Models in Business and Industry* **26**(6), 665–667.

Scott, S. L. (2015), 'Multi-armed bandit experiments in the online service economy', *Applied Stochastic Models in Business and Industry* **31**(1), 37–45.

Sutton, R. S. and Barto, A. G. (1998), *Reinforcement Learning: an Introduction*, MIT Press.

Thompson, W. R. (1933), 'On the likelihood that one unknown probability exceeds another in view of the evidence of two samples', *Biometrika* **25**(3/4), 285.

Urteaga, I. and Wiggins, C. H. (2018), Bayesian bandits: balancing the exploration-exploitation tradeoff via double sampling. arXiv:1709.03162v2.

Vernade, C., Carpentier, A., Zappella, G., Ermis, B. and Bruckner, M. (2018), Contextual bandits under delayed feedback. arXiv:1807.02089.

White, J. M. (2013), *Bandit Algorithms for Website Optimization*, OReilly.

Whittle, P. (1988), 'Restless bandits: activity allocation in a changing world', *Journal of Applied Probability* **25**(A), 287–298.

Zhou, Y., Zhu, J. and Zhuo, J. (2018), Racing thompson: an efficient algorithm for thompson sampling with non-conjugate priors, *in* 'Proceedings of the 35th International Conference on Machine Learning', Vol. 80, pp. 6000–6008.