Erasmus University Rotterdam

Erasmus School of Economics

Master Thesis Econometrics and Management Science - Analytics and Operations Research in Logistics

Optimal Order Routing with Contextual Multi-Armed Bandits

J. (Colin) van Garderen

Supervisor:	Dr. P.C. (Paul) Bouman
Second assessor:	B.T.C. (Bart) van Rossum
Company supervisor:	Dr. M. (Martin) van der Schans
Date final version:	July 31, 2024
Student number:	575370jg

Abstract

In asset management, brokers offer algorithms for buying and selling stocks, which are used by buy-side traders. These traders distribute the orders among these algorithms using algo wheels. Initially, the cost characteristics of broker algorithms are unknown, and each order must be routed to a broker, with costs observed only for the selected broker. This results in a so called bandit problem, where the challenge is to balance exploration exploitation. This study advances the current methodologies by developing Contextual Multi-Armed Bandit (CB) algorithms, to enhance the balance between exploration and exploitation. These algorithms account for the fact that cost characteristics may depend on specific context features such as market conditions or trading volume. Linear estimation methods, including Ordinary and Regularized Least Squares, are used within CB frameworks to estimate the relationship between context and costs. Additionally, a novel non-linear CB employing Random Forest Regressors (RF) is introduced in this research. To further refine the dynamic balancing process, Thompson Sampling (TS) is incorporated into the CB frameworks. Custom metrics are developed to provide deeper insights into routing decisions based on context features, enhancing the understanding of both linear and non-linear CBs. The CBs are tested on a synthetic dataset with known broker distribution characteristics and context dependencies, as well as a real-world dataset from Robeco containing 2023 trading cost data, characterized by leptokurtic cost distributions. Results indicate that the CB with RF and TS outperforms all MABs and linear CBs on both datasets. Moreover, non-linear CBs demonstrated superior handling of leptokurtic cost distributions and provided more valuable insights into routing decisions, highlighting their practical utility. These findings suggest that the adoption of advanced CB algorithms can significantly improve order routing efficiency in asset management.

Keywords: Buy-side Trading, Order Routing, Contextual Multi-Armed Bandits, Random Forest Regressors, Thompson Sampling

Preface

This research was conducted at Robeco, a global asset manager overseeing a total of 150 to 200 billion dollars in assets. As part of the requirements for completing my Master of Science in Econometrics and Management Science at Erasmus University, I joined the Equity Trading Research Team at Robeco's Rotterdam office from April to July 2024. Through Robeco's Super Quant Program, I had the privilege of writing my thesis under the guidance of Martin van der Schans, a seasoned trading researcher.

My interest in financial markets began at the age of twelve, thanks to my father, who introduced me to this field during one of our long-distance cycling trips. This led me to recognize early on the importance of programming and statistical analysis, which ultimately guided my decision to study econometrics. During my time at Robeco, I had the opportunity to apply my mathematical, econometric, and programming skills to real-world problems at the intersection of operations research and financial trading. This experience allowed me to practically implement many theoretical aspects of econometrics, significantly enriching my learning journey.

I am immensely thankful to Martin van der Schans for his confidence in my abilities, offering me this unique opportunity, and for his expert guidance throughout the process. My heartfelt thanks also go to all my colleagues at Robeco, whose shared knowledge and expertise greatly enhanced my research experience. Additionally, I wish to express my gratitude to Dr. P.C. Bouman, my supervisor at Erasmus University, for his support, and to B.T.C. van Rossum for being my second assessor.

As this research marks the end of my academic studies, I would like to extend my deepest appreciation to my parents for their constant support. My family and friends have been pillars of encouragement throughout this journey. Finally, I owe a special thank you to my wife, Harmke, whose unwavering motivation and support, especially during the last month, have been indispensable, coinciding with our first month of marriage.

Colin van Garderen
Rotterdam, July 2024

Contents

1 Introduction

On a daily basis, portfolio managers place orders to buy or sell stocks in response to evolving market conditions, shifts in investor sentiment, or changes in fund inflows and outflows. These orders are routed to a buy-side trading desk for execution, where large trades are managed manually by traders, while smaller orders are often executed through broker algorithms provided by execution brokers. Robeco, managing between 150 and 200 billion in assets, is such a buy-side firm. Efficient execution of these orders is critical for optimizing portfolio performance and minimizing trading costs. This involves strategically routing smaller orders to broker algorithms to reduce costs. In addition to cost minimization, buy-side traders must continuously explore and evaluate alternative broker algorithms to ensure optimal trade execution strategies. This proactive approach not only monitors the performance of top brokers but also assesses whether other brokers offer improved execution under different market conditions or algorithmic advancements.

Building on the work of ter Braak & van der Schans (2023), which focused on achieving an optimal balance between exploration and exploitation for routing orders to broker algorithms in the US, this study extends the analysis to multiple regions, each with its own market dynamics. As the algorithms are offered by brokers, evaluating broker performance implicitly assesses the effectiveness of these algorithms. By considering regions as context behind the orders, we aim to diversify broker usage accros all regions. Furthermore, some brokers may perform better with low spreads or volatility, while others excel under higher values for these features. Our study investigates these potential variations in broker performance across different regional markets. This study explores variations in broker performance across regions, incorporating additional features like participation rates, spreads and volatility, to enhance order execution based on regional insights.

Expanding on established order routing frameworks such as Multi-Armed Bandits (MABs), as outlined in ter Braak & van der Schans (2023), which include methods like ϵ-greedy, Upper Confidence Bound (UCB), and Thompson Sampling (TS) for optimizing broker selection, this study advances the approach by incorporating Contextual Multi-Armed Bandits (CBs). We aim to model the relationship between order contexts, such as region and spread, and broker performance. Initially, we employ linear regression models, including Ordinary Least Squares (OLS) and Regularized Least Squares (RLS). However, recognizing the limitations of linear models in capturing complex scenarios, we develop non-linear CBs utilizing Random Forest Regressors (RF). These non-linear models may offer enhanced flexibility and accuracy in capturing intricate relationships between order contexts and broker performance. We evaluate these models using ϵ-greedy, UCB, and TS strategies to optimize order allocations based on predicted broker performance within specific contexts.

We conduct a thorough evaluation using two datasets. The first is a synthetic dataset for tuning and validation, where cost distributions and context dependencies are controlled, and real-world trading data from Robeco for 2023, spanning two regions. The synthetic dataset shows that CBs significantly outperform the MABs, with the non-linear CB based on TS demonstrating superior performance. Further validation with the Robeco data confirms that the non-linear CB

consistently outperforms both MABs and linear CBs in selecting optimal brokers. It effectively handles complex, leptokurtic cost distributions, achieving cost reductions of approximately 18% in one region and nearly 4% in another. These findings, supported by historical data, highlight the model's effectiveness in diverse market conditions.

In summary, this study contributes to the field by advancing methodologies for optimal order routing. Specifically, we introduce a novel non-linear CB framework. Additionally, our research incorporates innovative methods to enhance the transparency of decision-making processes, ensuring that the evaluation and selection of broker algorithms are both robust and interpretable. This work also provides actionable insights for optimizing order execution in diverse trading environments.

In Chapter 2, we present a detailed problem description and define our performance metrics. Chapter 3 reviews existing literature on related methods, explores potentially relevant context features for order routing, and outlines our research contributions. The methodology for MABs, linear CBs, and non-linear CBs is detailed in Chapters 4, 5, and 6, respectively. Chapter 7 describes both the synthetic dataset we constructed for hyperparameter tuning and model selection and validation, and the real-world trading cost dataset obtained from Robeco. In Chapter 9, we analyze the results from both datasets, providing insights and comparisons. Chapter 10 wraps up the findings and implications of our research.

Additionally, four appendices complement the main text. Appendix A offers a description of an additional linear CB framework. Appendix B provides an in-depth data description. Appendix C details the tuning and bandit selection process and Appendix D includes an extended analysis of the results from the Robeco dataset.

2 Problem Description

As introduced in previous chapter, Robeco employs algo wheels that distribute orders among electronic trading algorithms offered by brokers. To enhance the performance of these algo wheels, Robeco employs MAB algorithms, a subset of reinforcement learning techniques. MAB algorithms address the exploration-exploitation trade-off by systematically exploring various options to discover optimal strategies, while also exploiting known high-performing strategies to maximize cumulative rewards or minimize cumulative costs. This research builds upon the foundational work of ter Braak & van der Schans (2023), who utilized MABs to distribute orders among multiple brokers. However, MABs typically overlook contextual factors such as the geographic location of companies or the daily trading volume of specific shares. Therefore, investigating the integration of these contextual states into the decision-making process may lead to further reductions in total regret, facilitated by the adoption of CBs. Regret, defined as the difference between the total price paid for specific shares and the volume-weighted average price (VWAP), serves as a critical metric for assessing performance. Additionally, leveraging contextual states can offer insights into the effectiveness of electronic algorithms, whose operational logic remains opaque, thereby enabling the identification of algorithms that excel under specific conditions.

In the context of algo wheels, orders arrive sequentially at discrete times $t = 0, 1, 2, \ldots$. In this study, we operate under the assumption that each order is fulfilled before the subsequent one arrives. In practice, however, models may undergo retraining during market closures, such as overnight or over weekends. This choice can also be justified by the assumption that the inclusion of one or a few additional observations would not significantly alter the model's decision-making process. Each broker serves as an *arm* in the MAB framework, where A denotes the set of all arms, where K represents the total number of brokers. The cost incurred by allocating an order to a specific broker at time t is denoted as $C_t(a_t)$. In MAB theory, the objective is to minimize total regret up to time T, defined by ter Braak & van der Schans (2023) as:

$$R_T = \sum_{t=0}^{T-1} C_t(a_t) - T\mu^*, \tag{1}$$

where μ^* denotes the average cost of the best-performing broker, an unknown parameter due to our ability to observe only the selected broker's costs, rather than all possible costs. To minimize regret in the context of MAB algorithms, the optimal strategy involves routing each order arriving at time t to the broker a_t that minimizes the expected regret $E_t[R_T]$ after T orders, given all the information accumulated up to time t. This expectation, E_t, conditions on the knowledge of brokers $a_0, \ldots, a_{t-1}$ chosen up to time t, as well as the realized costs $C_0(a_0), \ldots, C_{t-1}(a_{t-1})$ up to time t. To evaluate the performance of a given MAB algorithm, one can calculate the pseudo-regret, defined as:

$$\bar{R_T} = \sum_{t=0}^{T-1} \mu_{a_t} - T\mu^*, \tag{2}$$

where μ_{a_t} denotes the unconditional mean cost for the broker a_t chosen at time t. The pseudo-regret is equivalent to the regret in expectation, but with reduced variability, as it presumes that

the cost incurred is the average cost for each chosen broker.

However, in this research, we investigate how the brokers perform in specific contexts and how we can utilize the differences in broker performances given these contexts. Therefore, we modify the regret function to a context-based regret function. We have a matrix of observed contexts, denoted as X, containing contexts $x_1, x_2, \ldots, x_n$ corresponding to the trades placed on time $t \in \{1, 2, \ldots, n\}$. The Bandit model must learn the best mapping $g : X \to \{1, \ldots, K\}$ of contexts to arms. The regret for Contextual Bandits is formulated as follows:

$$R_T = \sum_{t=0}^{T-1} C_t(a_t) - \mu^*_{g(x_t)}, \tag{3}$$

where $\mu^*_{g(x_t)}$ is the average cost of the broker with the lowest average cost given context x_t. We can analyze the pseudo-regret in the contextual case, defined as:

$$\bar{R_T} = \sum_{t=0}^{T-1} \mu_{a_t} - \mu^*_{g(x_t)}, \tag{4}$$

where μ_{a_t} denotes the unconditional mean cost for the broker a_t chosen at time t given the context x_t. In this research, we focus on routing orders to the broker with the lowest average cost for a certain context. Therefore, we utilize the pseudo-regret metric, as it provides a stable estimate of performance by assuming the incurred cost is the average cost for each selected broker. As we are minimizing the costs rather than maximizing rewards in our problem, we will discuss everything within this research in the context of minimization. However, the methods described in this paper can be easily adapted to a maximization problem where one aims to pull the arm corresponding to the highest reward.

3 Literature

In the previous chapter, we introduced the (Contextual) Multi-Armed Bandit problem. This chapter builds on that foundation by reviewing existing literature related to CBs. In Section 3.1, we examine the methods that have been developed and investigated, along with the results obtained from these approaches. Additionally, in Section 3.2, we discuss relevant literature on key features that can explain trading costs among brokers. Finally, Section 3.3 outlines the contributions of our research, highlighting how it advances the current understanding and application of CBs in this domain.

3.1 (Contextual) Multi-Armed Bandits in General

MAB problems focus on the process of optimizing decision-making over time by balancing the choice between known, high-reward options and exploring new, potentially more rewarding alternatives. Various strategies, such as ϵ-greedy, UCB, and TS, have been devised to address this challenge. The ϵ-greedy approach prioritizes exploitation while occasionally exploring random actions with a small probability ϵ. In contrast, UCB algorithms leverage uncertainty estimates to systematically balance exploration and exploitation, aiming to select actions based on high estimated reward or low estimated costs under conditions of significant uncertainty. Thompson Sampling, on the other hand, adopts a Bayesian perspective, sampling from posterior distributions to guide decision-making.

Regret analysis, as studied by Bubeck et al. (2012) and Russo et al. (2018), offers formal proofs regarding regret bounds. Regret analysis quantifies the cumulative opportunity loss due to suboptimal decision-making, providing a theoretical framework to compare exploration-exploitation algorithms and assess their effectiveness in diverse contexts. Specifically, Bubeck et al. (2012) show that for several popular bandit algorithms, the expected regret grows logarithmically in the number of rounds, implying efficient learning. This result highlights the ability of these algorithms to balance exploration and exploitation effectively. Russo et al. (2018) further explore regret analysis in the context of different problem settings, including adversarial and stochastic bandits. They discuss how regret bounds can be used to design algorithms with guaranteed performance under various assumptions about the reward distributions. Agrawal & Goyal (2012a) analyze Thompson Sampling and establish regret bounds that depend on the inherent uncertainty in the reward distributions. Their work highlights the benefits of Thompson Sampling in settings where the agent has some prior knowledge about the rewards, potentially leading to faster learning compared to other algorithms.

While traditional MAB algorithms could help balancing between exploration and exploitation, real-world scenarios often entail additional contextual information influencing decision-making processes. As such, the field has progressed towards Contextual Bandit problems, where contextual cues inform action selection strategies, offering enhanced adaptability and performance in complex decision-making environments.

The CB problem with linear payoffs is an extensively researched topic in statistics and machine learning, known by various names throughout its development. Chu et al. (2011) note that

it has been referred to as bandit problems with co-variates (Woodroofe, 1979; Sarkar, 1991), associative reinforcement learning (Kaelbling, 1994), bandit problems with expert advice (Auer, Cesa-Bianchi, Freund & Schapire, 2002), associative bandit problems (Strehl et al., 2006), and linear bandits (Dani et al., 2008; Abbasi-Yadkori et al., 2011; Bubeck et al., 2012). The term "contextual bandits" was introduced by Langford & Zhang (2007), and has since become the standard terminology in the field. Langford & Zhang (2007) analyzed an epoch-greedy approach to the contextual bandit problem.

Hence, the relationship between context and rewards can be estimated using linear estimation methods, such as OLS and RLS methods like Ridge and Lasso regression. One notable application of contextual bandits is in news article recommendation. Li et al. (2010) constructed both disjoint and hybrid LinUCB models utilizing OLS and Ridge regression, comparing these models to simple multi-armed bandits. Their findings indicate that LinUCB methods generally outperform the simpler ϵ-greedy methods. Furthermore, they concluded that LinUCB with Ridge regression demonstrates particular advantages when dealing with sparse data. Additionally, the hybrid model, which shares information across multiple arms rather than treating each arm independently, outperformed all other models in their study. Dimakopoulou et al. (2017) conducted a comparison of regularized linear estimation methods Ridge and Lasso in personalized recommendations using LinUCB and LinTS (Linear Thompson Sampling). Their findings indicate that while UCB exhibits strong theoretical guarantees, Thompson Sampling generally outperforms it in practical applications. Their results also indicate that Lasso results in lower regrets compared to Ridge.

Linear estimation methods rely on assumptions, which may occasionally result in misspecified models. To mitigate this, researchers have explored non-parametric approaches like tree-based methods and neural networks. Féraud et al. (2016) developed Bandit Forests, which utilize a group of decision trees as classifiers to draw an arm. Their comparisons with LinUCB show that Bandit Forests outperform LinUCB and other estimation methods across several datasets from the UCI Machine Learning Repository. Additionally, Elmachtoub et al. (2017) developed a decision tree-based method incorporating bootstrapping techniques and compared it to Bandit Forests and LinUCB in an offline setting. Their results demonstrate that these bootstrap-based methods outperform the others. Dimakopoulou et al. (2017) also compared Bandit Forests with LinUCB and LinTS, concluding that non-parametric models like Bandit Forests perform better in practice. Furthermore, neural network techniques have been examined by researchers such as P. Xu et al. (2020), who constructed several neural networks using ReLU, following UCB, Linear, or LinearUCB approaches. These neural network-based methods consistently outperformed the standard LinUCB approach.

However, despite their impressive performance, neural networks are resource-intensive in terms of training time. A recent study by Nilsson et al. (2024) investigated XGBoost with UCB and TS approaches, which is a tree-based method, and compared it to three models: LinUCB, NeuralUCB, and NeuralTS. Their results indicate that the ensembled tree-based method outperforms all the others. Inspired by these recent findings, we will develop and explore another ensembled tree-based method in a contextual bandit setting. We aim to examine how random forest re-

gressors perform compared to linear methods. Random Forests, developed by Breiman (2001), create a set of regression trees based on several subsets of the available features and average the predictions made by these trees. The variance in predictions made by the trees can be incorporated into bandit methods, which serves as the foundation for our developed contextual bandits using random forests and bandit methods like UCB and TS. Random Forests leverage parallelization and random feature selection for faster training, particularly on large datasets, while XGBoost prioritizes accuracy through sequential training that corrects errors from previous trees, albeit it can be slower for very large datasets. Both algorithms require hyperparameter tuning, but Random Forests are generally less sensitive, with fewer parameters to adjust. Although Nilsson et al. (2024) reported good results with XGBoost, we opt to use Random Forests. This decision stems from the fact that, on many problems, the performance of Random Forests is comparable to boosting methods, and they are simpler to train and tune (Hastie et al., 2009).

3.2 Key Features in Trading Costs

In the realm of equity trading, algorithmic strategies play a crucial role in optimizing execution outcomes while minimizing costs. Broker algorithms and strategies such as VWAP algorithms have been extensively studied for their effectiveness in achieving best execution (Madhavan, 2002; Kissell et al., 2003; Kato, 2015). These studies underline the importance of algorithmic trading in enhancing execution quality by adhering to predefined trading benchmarks.

The securities industry has undergone rapid structural, technological, and regulatory changes, significantly impacting market microstructure. Baker & Kiymaz (2013) highlights how these developments have transformed trading environments globally. These changes have also caused regional differences in trading practices, particularly between European and American stocks, as discussed by Gutiérrez et al. (2018). Regional policies and regulations further exacerbate these differences, necessitating a nuanced approach to algorithm selection and execution strategies.

Market impact, closely tied to participation rates, is another critical factor influencing trading costs. Said et al. (2017) discusses how high participation rates can significantly drive up costs, highlighting the importance of managing market impact effectively. Interestingly, this cost driver is not limited to large orders. Dani et al. (2008) demonstrate that small orders also experience notable market impact, suggesting that all trade sizes require careful algorithmic consideration.

The spread of a security, indicative of its liquidity, is another vital determinant of trading costs. Hasbrouck (2004) provides empirical evidence on how the spread reflects the liquidity of a stock, impacting the ease of executing trades without adverse price movements. Understanding and managing spread dynamics are crucial for optimizing trading strategies.

Volatility might also be an informative feature for VWAP algorithm trading costs, as it can significantly impact market conditions and, consequently, the execution performance of trading algorithms Domowitz et al. (2001). Some brokers might perform well in markets with lower volatility, while others may excel during periods of high volatility. Understanding and anticipating these differences can help in selecting the appropriate broker algorithm to optimize trading costs.

To have an indication of the volatility at the time an order was placed, we use the close of the volatility measure from the day before the order was executed. There are multiple volatility measures, representing certain regions, sectors or markets. For example the VIX, which focuses specifically on the US market. The VIX is widely regarded as a good indicator of current market volatility Whaley (2000). Another measure is the VSTOXX, which is more oriented on the Euopean market, based on the EURO STOXX 50 index, providing a more relevant measure of market volatility within the European context.

3.3 Our Contribution

Our contribution to the contextual bandit problem extends beyond existing literature in several key aspects. Firstly, in contrast to the prevailing approach of employing separate estimation models for each arm as seen in studies like (Li et al., 2010; Dimakopoulou et al., 2017), we adopt a unified estimation model. This model provides deeper insights into the decisions and estimations made by the models. Moreover, our singular model construction allows for the detection of similarities between arms, a feature lacking in previous methodologies. Additionally, while decision tree-based methods are examined in CB frameworks, our research distinguishes itself by incorporating the Random Forest Regressors approach. We leverage the flexibility of RF, known for their ability to handle continuous state spaces and predict numerical values, offering advantages over traditional classification-based approaches in certain contexts.

Moreover, in our research, we utilize and develop additional techniques to gain deeper insights into the model its performance, its decision-making process, and the importance of various features. By employing the RF, we may identify which features are most significant and which are less influential, by measuring the feature importances. This analysis allows us to discern which variables contribute valuable information and which ones merely introduce noise, thereby optimizing the allocation of orders to brokers. Additionally, since we integrate estimation methods such as RF and OLS within the Thompson Sampling-based CB framework, we can illustrate the probabilities associated with routing orders to specific brokers given certain contexts. This approach provides insights into the model its certainty regarding the expectation that a particular broker will incur the lowest costs compared to others.

Lastly, we build upon the work of ter Braak & van der Schans (2023) by extending their model to a context-based model. Furthermore, we enhance their methodology by testing it on a larger dataset, providing a more comprehensive evaluation of its effectiveness in real-world applications. Together, these contributions enhance our understanding of CBs and offer practical solutions with broader applicability.

4 Multi-Armed Bandits

In Chapter 2 we introduced the MAB problem and we described what we want to optimize. In this Chapter we describe three MABS that are often used to manage the exploration-exploitation trade-off and form the basis for the CBs we discuss in next chapters. The first strategy is the ϵ-greedy approach, a probabilistic method that incorporates a degree of randomness, as detailed in Section 4.1. The second strategy involves algorithms that adopt an optimistic stance in the face of uncertainty, exemplified by the Upper Confidence Bound algorithms, discussed in Section 4.2. The third strategy, described in Section 4.3, utilizes Bayesian techniques, with a particular focus on Thompson Sampling, which leverages probability distributions to guide decision-making in a adaptive manner.

4.1 ϵ-Greedy

In this section, we describe the ϵ-greedy strategy. This approach strikes a balance between exploiting the best-known arm with a probability of $1 - \epsilon$ and exploring other arms randomly with a probability of ϵ, which must lie between 0 and 1. The method for estimating the cost of selecting arm a up to a given time t is captured mathematically by $\hat{\mu}_a(t)$, which is computed as follows:

$$\hat{\mu}_a(t) = \frac{1}{n_a(t)} \sum_{\tau=0}^{t-1} C_\tau(a_\tau) 1_{a_\tau = a}, \tag{5}$$

where $n_a(t)$, representing the total number of selections of arm a by time t, is expressed by:

$$n_a(t) = \sum_{\tau=0}^{t-1} 1_{a_\tau = a}. \tag{6}$$

In the ϵ-greedy strategy, the algorithm for arm selection is structured to balance exploration and exploitation, determined through the following decision-making formula:

$$a_t = \begin{cases} \arg\min_{a \in A} \hat{\mu}_a(t) & \text{with probability } 1 - \epsilon, \\ \text{random } a \in A & \text{with probability } \epsilon. \end{cases} \tag{7}$$

These formulations enable a systematic approach to balancing risk and reward by dynamically adjusting the exploration-exploitation trade-off based on accumulated experience and observed outcomes.

However, it is recognized that these algorithms can sometimes converge to a sub-optimal arm. Additionally, the ϵ-greedy approach does not always offer an ideal or dynamic balance between exploitation and exploration, potentially leading to less efficient learning. Given that the literature has demonstrated that other strategies provide more robust performance guarantees compared to the ϵ-greedy method, we have opted not to pursue other random strategies beyond ϵ-greedy in this research.

4.2 Upper-Confidence Bound

The UCB approach operates on an optimistic assumption about the outcomes when an arm is pulled, particularly favoring arms about which there is considerable uncertainty. This optimism is reflected in the strategy's preference for selecting arms with the greatest uncertainty in their reward distribution. The rationale behind this is that exploring arms with high uncertainty potentially maximizes the rewards, dynamically balancing exploration and exploitation, providing a systematic method to focus on potentially under-explored yet rewarding options. In the context of MABs, the UCB strategy selects the arm with the highest upper confidence bound, that we subtract from the average reward since we are minimizing, which is mathematically represented as:

$$a_t = \arg\min_{a \in A} \left(\hat{\mu}_a(t) - \hat{\sigma}_a(t) \sqrt{\frac{2 \log t}{n_a(t)}} \right), \tag{8}$$

where $\hat{\sigma}_a(t)$ represents the standard deviation of the observed costs up to time t for arm a. This formula ensures that the arm selected not only has a high estimated reward but also takes into account the uncertainty or variability in its reward distribution, represented by the square root term.

While the standard UCB incorporates a measure of variability directly into its confidence calculation, UCB-1, introduced by Auer, Cesa-Bianchi & Fischer (2002), opts for a simplified exploration term:

$$a_t = \arg\min_{a \in A} \left(\hat{\mu}_a(t) - c \sqrt{\frac{\log t}{n_a(t)}} \right). \tag{9}$$

Here, c is a hyperparameter that scales the exploration term, $\sqrt{\frac{\log t}{n_a(t)}}$, in UCB1. The main distinction lies in UCB1 its simplified approach, which utilizes a constant factor c to modulate the exploration intensity. This simplification does not account for the variability of rewards as directly as the standard UCB approach, potentially making UCB1 less sensitive to rapid changes in the variance among different arms their rewards. Although this may result in a more straightforward and potentially less computationally intensive calculation, it could overlook critical risk factors associated with arms whose rewards are more volatile, thus affecting the robustness of decision-making in environments with fluctuating reward distributions.

4.3 Thompson Sampling

Thompson Sampling, a strategy first proposed in 1933 and further developed in the work of Agrawal & Goyal (2012b), applies a Bayesian framework to make probabilistic decisions in multi-armed bandit problems. At its core, Thompson Sampling constructs a probability distribution f_t for the estimated mean costs of each arm, $\mu = [\mu_1, \ldots, \mu_K]$, using all available information up to time t. This distribution is then used to select a broker by modeling the probability that any given broker has the lowest mean cost. Mathematically, the selection process is described as:

$$P(\text{broker } a \text{ has the lowest mean cost at time } t) = \int 1_{a = \arg\min_a \mu_a} f_t(\mu)\, d\mu, \tag{10}$$

where μ_a is the component of the vector μ corresponding to arm a. This approach effectively balances the exploration of brokers with potentially optimal but uncertain performance against the exploitation of brokers whose performance is well-determined and likely superior.

In practical terms, the actual computation of the integral is circumvented by sampling directly from the distribution f_t. At each decision point t, we simply draw a sample $\tilde{\mu}_a(t)$ for each arm and choose the arm a_t with the lowest sampled mean cost:

$$a_t = \arg\min_a \tilde{\mu}_a(t), \tag{11}$$

where $\tilde{\mu}_a(t)$ denotes the drawn mean cost for arm a at time t.

For constructing the distribution f_t, we employ a bootstrapping method as outlined by Kveton et al. (2019) and enhanced by the re-weighted bootstrap technique discussed by L. Xu et al. (2020). This bootstrap technique is also used in Appendix A.2, where we describe the procedure. These methods allow for a robust estimation of the mean distributions by incorporating variations and uncertainty directly into the decision-making process, thus adhering to the principles of Thompson Sampling by effectively using historical data to inform current choices.

5 Linear Contextual Bandits

In the previous chapter we explained three variants of MABs. These form also the basis for the CBs that we describe in this chapter and Chapter 6, where we introduce the non-linear CB. In this Chapter, we describe the linear methods utilized to estimate the relationship between contexts and rewards and how these methods are leveraged within the framework of CBs. Section 5.1 introduces the notations and definitions used throughout our study. We explore several linear methods for estimating the relationship between the costs of an arm and the context. We start with a basic linear approach using OLS, which is described in Section 5.2. As discussed in Chapter 4, there are multiple strategies to balance exploration and exploitation in MABs, which similarly apply to CBs. Consequently, this section elaborates on how these strategies function within the Linear CB setting. In Section 5.3, we describe the approaches for OLS based CBs. Since it is known OLS models might be sensitive for overfitting, we also explore some RLS based methods, like Ridge and Lasso, as detailed in Appendix A.1. And Appendix A.2 discusses CB approaches using the RLS methods.

Successful application of linear models generally requires adherence to certain assumptions. Although it is not always guaranteed that these prerequisites are met in our context, we have opted to incorporate these estimation methods due to their interpretability and their utility in providing a solid benchmark for our non-linear models, which are discussed in Chapter 6.

5.1 Context Matrix Definition

Traditional research often builds separate models for each arm in contextual bandits, a method referred to as disjoint models Li et al. (2010). In contrast, our research utilizes a unified model across all arms within the multi-armed bandit framework. We enhance the context matrix by duplicating feature columns for each arm, resulting in a matrix expansion to $k \times l$ columns for a model with k arms and l features. In our disjoint setup, the context vector $\mathbf{x}(t)$ at any given time $t \in 1, \ldots, T$ for the selected arm a from the set of arms A is constructed to reflect the arm-specific activation within a contextual multi-armed bandit framework. The feature matrix $\mathbf{X}(\mathbf{T})$ representing all context vectors for the allocations to the arms in A up to and including time T can be described as follows:

$$\mathbf{X}(\mathbf{T}) = \begin{bmatrix} \mathbf{x}(1) \\ \vdots \\ \mathbf{x}(T) \end{bmatrix},$$

where the context vector for any time $t \in \{1, \ldots, T\}$, denoted as $\mathbf{x}(t)$, is structured as follows:

$$\mathbf{x}(t) = [S_1(t)\mathbb{1}_{a_t=1}, \ldots, S_j(t)\mathbb{1}_{a_t=a}, \ldots, S_l(t)\mathbb{1}_{a_t=k}],$$

where $S_j(t)$ denotes the value of feature j at time t, and $\mathbb{1}_{a_t=a}$ is an indicator function that is zero if arm a was not selected at time t, and otherwise, it is equal to $S_j(t)$. This configuration ensures that for each time t when arm a is selected, the vector $\mathbf{x}(t)$ contains non-zero values only in the subsection that corresponds to the selected arm a, while all other sections corresponding

to unselected arms are set to zero. This selective activation allows the model to learn distinct feature weights and interactions specific to each arm, which enhances the model's predictive accuracy and specificity by focusing learning on the activated arm's context.

Building on our disjoint model, we further refine our approach by introducing a hybrid model where context features are shared across all arms. This model integrates additional columns that correspond to shared features across all arms, alongside arm-specific columns. The context vector $\mathbf{x}(t)$ for the hybrid models at any given time t is can be described as follows:

$$\mathbf{x}(t) = [H_1(t), \ldots, H_j(t), \ldots, H_l(t), S_1(t)\mathbb{1}_{a_t=1}, \ldots, S_j(t)\mathbb{1}_{a_t=a}, \ldots, S_l(t)\mathbb{1}_{a_t=k}].$$

In this configuration, $H_1(t), \ldots, H_j(t), \ldots, H_l(t)$ represent the shared feature values across all arms, enhancing the ability to detect similarities and interactions between different arms. Each context vector $\mathbf{x}(t)$ in the matrix $\mathbf{X}$ has $(k+1) \times l$ columns under the initial setup.

To address potential issues of multicollinearity and the dummy variable trap commonly encountered with this expanded feature set in linear models, we can apply a dimensionality reduction by removing columns corresponding to one arbitrarily chosen arm. This results in the feature matrix, with $k \times l$ columns, where the context vectors are being adjusted to:

$$\mathbf{x}(t) = [H_1(t), \ldots, H_j(t), \ldots, H_l(t), S_1(t)\mathbb{1}_{a_t=1}, \ldots, S_j(t)\mathbb{1}_{a_t=a}, \ldots, S_l(t)\mathbb{1}_{a_t=k-1}].$$

In our hybrid model for contextual bandits, the feature vector $\mathbf{x}(t)$ at each time step t incorporates both arm-specific and shared features across contexts. This setup optimizes the learning process by focusing on the relevant features for the selected arm, while neutralizing others.

In our contextual bandit model, estimating the reward for a specific arm $a \in A$ requires constructing a context vector $\mathbf{x}_a(t)$ tailored to the model's configuration. The context vector is defined such that each element $x_a(t)$ is:

$$x_a(t) = [\ldots, H_j(t), \ldots, S_j(t)\mathbb{1}_{a_t=a}, \ldots].$$

This setup ensures that only the features associated with the selected arm or shared across all arms (represented by z) are active. For estimating the expected reward for arm $a \in A$ at any time t, we utilize the estimation methods we describe in the next sections, and denote the expected reward for arm a as $\hat{\mu}_a(t)$. This value is derived from one of the estimation models applied within the framework, capturing the expected payoff based on the current context and learned parameters.

In the contextual bandit framework, our goal is to estimate the expected reward for each arm in set A based on the given context. Each iteration of the process involves constructing and training a model that captures the relationship between the context and the performance of each arm given the observations up to time T.

Each iteration $t \in \{t, \ldots, \}$ introduces a new observation from iteration $t-1$, necessitating an update to the model to include this new data. The model is typically updated by retraining

to incorporate the latest observations. However, retraining the model for each new observation can be computationally intensive. To manage this, we often choose to retrain the model only after every t_{ret} number of iterations, balancing the need for model accuracy with computational efficiency.

Based on the updated model and the chosen strategy, an arm a_t is selected, typically the one that matches the lowest predicted cost or highest reward for the given context. A reward is received only for the selected arm, reinforcing the learning specific to that arm's context-performance relationship. This iterative updating and selection process refines our understanding and predictions over time, optimizing decision-making in dynamic environments.

5.2 Ordinary Least Squares

The linear estimation methods, OLS and RLS, aim to estimate the expected reward given the context, expressed mathematically as $E_{t-1}[r_t|x(t)] = x(t)^\top \theta^*$, where θ^* is the optimal coefficient vector which is unknown. To apply these methods effectively, we estimate the parameter vector $\hat{\theta}$, and subsequently use it to compute the expected reward for each arm. In the framework of CBs, the OLS approach provides a fundamental method for estimating the relationship between the context and rewards. Utilizing OLS, we model this relationship through the regression coefficient vector θ, which quantifies the influence of each context variable on the expected reward. The regression formula is expressed as:

$$\mathbf{r} = \mathbf{X}\theta,$$

where $\mathbf{r}$ represents the predicted rewards, and $\mathbf{X}$ denotes the matrix of context variables. This method assumes that the errors are normally distributed and independent of the context, providing a straightforward estimation approach that is particularly valued for its simplicity and interpretability. Using this estimation approach, we can incorporate three exploration-exploitation balancing methods. In all cases, the vector $\hat{\theta}$ is estimated using OLS, with the estimation formula given by:

$$\hat{\theta} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{r}.$$

This formula represents the solution to the normal equations derived from setting the gradient of the least squares cost function to zero. This method of estimating $\hat{\theta}$ allows us to predict rewards for arms $a \in A$ for any time t:

$$\hat{\mu}_a(t) = \mathbf{x}_a(t)^\top \hat{\theta}.$$

For further clarification we can express the expected reward of arm a at time t, conditioned on the context vector $x(t)$, as follows:

$$E_{t-1}[r_t \mid x(t), a_t = a] = \sum_{i=1}^{L} \alpha_i H_i(t) + \sum_{a' \in A} \sum_{i=1}^{L} \beta_{i,a'} S_i(t) \mathbb{1}_{a_t = a'} = \sum_{i=1}^{L} \alpha_i H_i(t) + \beta_{i,a} S_i(t)$$
$$= \alpha H(t) + \beta_a S(t),$$

where $H_i(t)$ represents hybrid features and $S_i(t)$ denotes standard features at time t. Here, $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_L)$ and $\beta_a = (\beta_{1,a}, \beta_{2,a}, \ldots, \beta_{L,a})$ are vectors of coefficients derived from the

estimated coefficient vector $\hat{\theta}$. The vector β_a specifically denotes the coefficients corresponding to arm a. The context vector $x(t)$ facilitates the extraction of $a_t = a$, ensuring that only the feature columns corresponding to arm a are active. This formulation allows for predicting the expected reward based on the available context at time t, integrating both hybrid and standard features into the OLS estimation model.

5.3 Linear Contextual Bandit Variants

In the ϵ-greedy variant of MABs, the decision rule operates by selecting, with probability $1 - \epsilon$, the arm that corresponds to the lowest predicted value. Conversely, with a probability of ϵ, a random arm is drawn:

$$a_t = \begin{cases} \arg\min_{a \in A} \hat{\mu}_a(t) & \text{with probability } 1 - \epsilon, \\ \text{random } a \in A & \text{with probability } \epsilon. \end{cases}$$

This method maintains a fixed balance between exploration (testing various arms) and exploitation (using the best-performing arm), which typically does not provide a strong regret bound. Furthermore, it is not guaranteed that this strategy will converge to the optimal arm, raising concerns about its effectiveness in certain scenarios.

To dynamically balance exploration and exploitation, we incorporate the uncertainty in our parameter estimates in the arm scores, which in turn reflects the uncertainty in the cost or reward estimation for a specific arm. We quantify this uncertainty for a given context, denoted as $x(t)$, which also indicates the arm to be considered based on our previously defined context matrix.

When employing the UCB approach in a minimization problem, which adopts optimism in the face of uncertainty, we select the arm that offers the lowest value of the combination of expected reward minus the uncertainty in this prediction. This uncertainty is typically represented by the standard deviation of the predicted reward, scaled by a predefined constant c. The mathematical formulation of the UCB valuation is given by:

$$UCB_a(t) = \hat{\mu}_a(t) - c \cdot \hat{\sigma}_a(t),$$

where $\hat{\mu}_a(t)$ represents the estimated mean reward for arm a at time t, and $\hat{\sigma}_a(t)$ represents the estimated standard deviation of the reward for arm a at time t, calculated using the following equation:

$$\hat{\sigma}_a(t) = \sqrt{\mathbf{x}_a(t)^\top (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{x}_a(t)}.$$

The arm with the lowest $UCB_a(t)$ value is selected, thereby integrating both the uncertainty component and the expected reward into the decision-making process.

While the UCB approach requires selecting a hyperparameter c, we further automate the exploration-exploitation balancing by implementing a Thompson Sampling based variant. This method employs a Bayesian framework, constructing a probability model for each arm's performance based on the estimated parameters and their uncertainty. Specifically, we assume a normal distribution

for the parameter vector $\hat{\theta}$, with the covariance matrix denoted as $\Sigma_{\hat{\theta}}$.

This setup allows us to use the distribution of $\hat{\theta}$ to estimate the expected reward of arms. The key advantage of Thompson Sampling is its ability to use the entire posterior distribution of the parameters to make decisions. By sampling from the normal distribution defined by $\hat{\theta}$ and $\Sigma_{\hat{\theta}}$, we can compute the probability that an arm has the lowest cost. The formula for this Bayesian probability estimation is given by:

$$P(a^* = a|x(t), \hat{\theta}, \Sigma_{\hat{\theta}}) = \int \mathbb{1}_{\{\theta: x_a(t)^\top \theta \leq x_b(t)^\top \theta, \forall b \neq a\}} p(\theta|\hat{\theta}, \Sigma_{\hat{\theta}}) d\theta, \tag{12}$$

where a^* is the optimal arm, $x_a(t)$ is the context vector for arm a, and $\mathbb{1}$ is the indicator function that evaluates to 1 if the condition within the braces is true, and 0 otherwise. This integral represents the probability that the linear combination $x_a(t)^\top \theta$ under the posterior distribution of θ results in the lowest cost compared to all other arms.

In the implementation of Thompson Sampling for CBs, we utilize a Bayesian approach to update our beliefs about the parameters. Rather than explicitly evaluating the probability integral described previously in equation 12, practical applications often simplify the process. Parameters θ_{TS}, which represent the coefficients in our model, are drawn once from the multivariate normal distribution $N(\hat{\theta}, \Sigma_{\hat{\theta}})$. Using these drawn parameter coefficients, we predict the reward for a specific arm given the context vector $x(t)$ and choose the arm with the lowest predicted costs. The decision criterion can be mathematically represented as:

$$a_t = \arg\min_a \mathbf{x}_a(t)^\top \theta_{TS},$$

where a_t is the arm selected at time t, and $x(t)$ is the context vector associated with that time step. It is crucial that the covariance matrix, $\Sigma_{\hat{\theta}}$, is positive semidefinite to ensure the validity of the drawn samples. If $\Sigma_{\hat{\theta}}$ is not positive semidefinite, the procedure defaults to randomly selecting an arm, bypassing the potential numerical issues that arise from an invalid covariance structure. This approach leverages the probabilistic nature of the Thompson Sampling algorithm to balance exploration and exploitation dynamically, based on the updated distributions of the estimated parameters.

6 Non-Linear Contextual Bandits

In the previous chapter, we examined the linear CBs investigated in this research, which also serve as benchmarks for the non-linear CBs introduced in this chapter. Here, we describe the non-linear estimation methods used to model the relationship between context and costs for arms within the CB framework. This approach eliminates the need for the assumptions required by linear models, allowing for better identification of important features and the uncovering of non-linear relationships in rewards.

Section 6.1 discusses the Random Forest Regressor and its application in our context. In Section 6.2, we present our initial methods for balancing exploration and exploitation based on predictions from the RF trees. We also explore alternative approaches to balancing exploration and exploitation based on the TS method. The first alternative approach involves estimating uncertainty using the subsamples used for prediction from individual trees, as detailed in Section 6.3. The second alternative method assesses prediction uncertainty based on the observed cost distributions of the arms, as described in Section 6.4. To gain deeper insights into the RF model, including identifying relevant versus redundant features, we explain the methodology for calculating feature importance in Section 6.5. Additionally, we introduce an evaluation metric in Section 6.6 to assess the performance of the RF model based on the data it is trained on.

In the contextual bandit framework, our objective is to estimate the expected reward conditioned on the context at each time t, denoted as $\hat{\mu}_t$. Instead of a linear dependency, the expected reward $\hat{\mu}_t$ is modeled using a nonlinear function f, which takes the context vector $x(t)$ and potentially other features or historical data into account. Formally, this relationship is expressed as:

$$\hat{\mu}_t = E[r_t|x(t)] = f(x(t), \Theta),$$

where $x(t)$ represents the context at time t, and Θ encapsulates the parameters or the learned aspects of the model up to time $t-1$, which may include historical context matrices and observed returns. This nonlinear approach allows for a more flexible and potentially more accurate estimation of rewards based on the given context.

6.1 Random Forest Regressor

In this paper, we propose and develop an alternative CB using a RF to estimate the relationship between the cost of the arm and the context. RF employs an ensemble of trees, with each tree acting as a weak learner. By aggregating the predictions of these individual trees, we aim to achieve accurate cost estimates for the arms.

Each tree in the RF is trained independently. With a total of M trees, the construction of each tree (T_b) involves several steps. First, a bootstrap sample Z^* of size T is drawn from the original training data. Then, the tree is grown iteratively by splitting each terminal node until the minimum node size $n_{\min}$ is reached. At each split, a random subset of m variables is selected from the total p variables, and the best variable and split point among them are determined. This process is repeated for each terminal node until the stopping criteria, such as maximum

depth or minimum samples per leaf node, are met.

In the RF construction, each tree is trained on a bootstrap sample drawn from the original training data, creating diversity among trees through random sampling with replacement. This process ensures robustness by allowing some data points to be repeated while others may be left out in the bootstrap sample. Utilizing the bootstrap sample, each tree is independently grown by considering random subsets of features at split points until reaching predefined stopping criteria like maximum depth or minimum samples per leaf node, contributing to the RF its predictive power and generalization ability. In RF, stopping the tree growth based on the minimum number of samples per leaf node makes the model less sensitive to dataset size changes and helps prevent overfitting. This approach also makes the model easier to understand and more adaptable to different types of data compared to limiting the tree depth.

The output of the RF model is the ensemble of trees $\{T_b\}_1^M$, which collectively make predictions based on the input features. In our application of RF, we utilize the ensemble of trees to predict the reward conditioned on a given context. This involves taking the average of the predictions generated by all weak learners, which collectively contribute to the model its predictive power. The ensemble prediction from a RF model can be represented as follows:

$$\hat{\mu}_a(t) = f(x_a(t)) = \frac{1}{M}\sum_{b=1}^{M} f_b(x_a(t)),$$

where $\hat{\mu}_a(t)$ represents the ensemble prediction of the reward for arm a given context $x(t)$, M is the total number of trees in the RF, and $f_b(a, x(t))$ is the prediction of the reward for arm a given context $x(t)$ by the b-th tree in the ensemble. Based on the leaf sample in each tree given the context $x_a(t)$, the reward for arm a is estimated by taking the mean of all observations in this leaf sample. Mathematically, for the b-th tree, if $L_b(x_a(t))$ denotes the leaf containing the context $x_a(t)$, and $\mathcal{D}_b(L_b(x_a(t)))$ represents the set of all observations in this leaf, the prediction $f_b(x_a(t))$ is given by:

$$f_b(x_a(t)) = \frac{1}{|\mathcal{D}_b(L_b(x_a(t)))|}\sum_{(x,r)\in\mathcal{D}_b(L_b(x_a(t)))} r,$$

where $|\mathcal{D}_b(L_b(x_a(t)))|$ is the number of observations in the leaf $L_b(x_a(t))$, and r represents the reward associated with each observation (x, r).

6.2 Random Forest Contextual Bandit Variants

To balance between exploration and exploitation, we employ the three approaches discussed in Section 4: ϵ-greedy, UCB and TS. The ϵ-greedy strategy follows a simple approach similar as explained for the linear methods described in Section 5. In the ϵ-greedy approach, the arm with the lowest estimated cost will be selected with a probability of $1-\epsilon$, while with a probability of

ϵ, a random arm is chosen. This selection process can be formulated as:

$$a_t = \begin{cases} \arg\min_{a \in A} \hat{\mu}_a(t) & \text{with probability } 1-\epsilon, \\ \text{random } a \in A & \text{with probability } \epsilon. \end{cases}$$

However, obtaining uncertainty estimates and standard errors per variable in a RF model is not directly available and can be challenging or is even impossible to compute. To address this limitation, we explored several alternative approaches. The other approaches, inspired by the UCB and TS ideas, aim to incorporate uncertainty into the predictions of the ensemble of weak learners.

In the TS approach, we model the reward for each arm as a normal distribution with the mean equal to the predicted reward from the RF model and the standard deviation given by the standard error of the mean of the predictions across the trees. Denoting the standard errors of the predictions for all arms with a vector SEM_t, we draw samples from a normal distribution with mean $\hat{\mu}_a(t)$ and standard deviation $\text{SEM}_t(a)$ for each arm a. The probability that arm a has the lowest cost is estimated by sampling from the posterior distributions of all arms and selecting the arm with the lowest sampled cost. Mathematically, this process can be represented as:

$$P(a^* = a | x(t), \text{SEM}_t) = \int \mathbb{1}_{\{\boldsymbol{\epsilon}: f(x_a(t)) + \boldsymbol{\epsilon}(a) \leq f(x_b(t)) + \boldsymbol{\epsilon}(b), \forall b \neq a\}} p(\boldsymbol{\epsilon} | \mathbf{0}, \text{SEM}_t) d\boldsymbol{\epsilon},$$

where $f(x_a(t))$ denotes the estimated reward for arm a given the context $x(t)$, and SEM_t is the vector of standard errors for the reward predictions up to time t.

Mathematically, the standard error of the mean for the reward predictions of arm a is calculated as:

$$\text{SEM}_t(a) = \frac{\sigma_t(a)}{\sqrt{M}},$$

where M is the total number of trees in the RF, and $\sigma_t(a)$ is the sample standard deviation of the reward predictions for arm a across the M trees. Specifically, $\sigma_t(a)$ is given by:

$$\sigma_t(a) = \sqrt{\frac{1}{M-1} \sum_{b=1}^{M} \left(f_b(x_a(t)) - \hat{\mu}_a(t) \right)^2},$$

where $f_b(x_a(t))$ is the reward prediction for arm a given the context $x(t)$ by the b-th tree in the ensemble. By incorporating the standard error into the TS approach, we can effectively balance exploration and exploitation based on the underlying uncertainty in the predictions.

6.3 Thompson Sampling Utilizing Leaf Samples of Random Forests

In this section, we describe an enhanced TS approach that leverages the individual trees within the RF model. Unlike the previous methods that consider the ensemble predictions and their associated uncertainties, this approach delves deeper into the structure of each tree to estimate the reward of each arm. For each tree in the RF, the reward prediction for an arm a given the context $x(t)$ is derived from the sample in the leaf node corresponding to $x(t)$. Let $L_b(x_a(t))$

denote the leaf node in the b-th tree that corresponds to the context $x(t)$ for arm a. The samples in this leaf node, $\mathcal{S}_b(x_a(t))$, are used to estimate the reward. The characteristics of this sample, specifically the mean $\mu_b(x_a(t))$ and the standard deviation $\sigma_b(x_a(t))$, are computed as follows:

$$\mu_b(x_a(t)) = \frac{1}{|\mathcal{S}_b(x_a(t))|} \sum_{r \in \mathcal{S}_b(x_a(t))} r,$$

$$\sigma_b(x_a(t)) = \sqrt{\frac{1}{|\mathcal{S}_b(x_a(t))| - 1} \sum_{r \in \mathcal{S}_b(x_a(t))} (r - \mu_b(x_a(t)))^2},$$

where r represents the rewards in the sample $\mathcal{S}_b(x_a(t))$, and $|\mathcal{S}_b(x_a(t))|$ is the number of observations in the leaf node. To incorporate the uncertainty in these predictions, we draw a sample $\tilde{r}_b(x_a(t))$ from a normal distribution with the mean $\mu_b(x_a(t))$ and standard deviation $\sigma_b(x_a(t))$:

$$\tilde{r}_b(x_a(t)) \sim \mathcal{N}(\mu_b(x_a(t)), \sigma_b(x_a(t))).$$

Once we have the sampled predictions from all trees, we aggregate these to form the final prediction for the reward of arm a by taking the mean of all sampled predictions:

$$\hat{\mu}_a^{\mathrm{TS}}(t) = \frac{1}{M} \sum_{b=1}^{M} \tilde{r}_b(x_a(t)),$$

where $\hat{\mu}_a^{\mathrm{TS}}(t)$ represents the TS-based reward prediction for arm a at time t.

This tree-based TS approach provides a more granular way of incorporating uncertainty by focusing on the characteristics of the samples within the leaf nodes of individual trees. By drawing from the distributions defined by these samples, we capture the variability inherent in the data, leading to more robust and informative predictions. This method not only considers the average predictions but also the spread of the rewards within each leaf node, enabling better exploration-exploitation trade-offs.

6.4 Cost Distributions Based Thompson Sampling

The uncertainty in predictions can be estimated using the predictions generated by all the independent weak learners, which constitute the RF. However, since these trees are weak learners, they may exhibit large standard deviations in their predictions, particularly when dealing with large feature spaces. Each tree focuses on its own subset of features and training data, leading to significant exploration and randomness in the predictions.

Therefore, we propose an alternative approach that focuses more on the underlying cost or reward distributions of the arms, which can be estimated based on the observations made up to the current time t. We predict the mean reward of each arm with the RF model and calculate the standard error of the mean reward for each arm using the observations made up to time t. Denoting the standard errors of all arms with a vector SEM_t, we can draw from a normal distribution with mean zero and standard deviation SEM_t.

This approach allows us to sample from all observations where a specific arm is drawn, resulting in a context-free algorithm variant of this method. However, to better balance between exploration and exploitation and account for potential context-dependencies in the cost distributions, we aim to adapt this approach to consider specific contexts. Therefore, in addition to the context-free variant of calculating the standard error of the mean for all observations made for arm a, we also introduce a context-based variant. In this variant, we calculate the standard error of the mean of the n closest observations made for arm a in terms of context vectors. The determination of the n closest observations follows the same procedure as described in Section 8.2. This concept aligns with the principles of Thompson Sampling, where we leverage the posterior distribution of the arms' costs to inform decision-making. By drawing samples from the normal distribution, we can estimate the probability that an arm possesses the lowest costs. The Bayesian probability estimation in this method is expressed by the following equation:

$$P(a^* = a|x(t), \text{SEM}_t) = \int \mathbb{1}_{\{\boldsymbol{\epsilon}: f(x_a(t))+\boldsymbol{\epsilon}(a) \leq f(x_b(t))+\boldsymbol{\epsilon}(b), \forall b \neq a\}} p(\boldsymbol{\epsilon}|\mathbf{0}, \text{SEM}_t) d\boldsymbol{\epsilon}, \tag{13}$$

where $f(x_a(t))$ denotes the estimated reward associated with arm a given the context $x(t)$, while SEM_t refers to the standard error of the mean vector encompassing the costs of all arms up to time t. This vector is derived from all observations conducted up to time t in the context-free version, whereas in the context-based approach, it draws from the n closest observations for each specific arm.

Mathematically, if $\mathcal{D}(a, t)$ represents the set of rewards observed for arm a up to time t, then the standard error of the mean for arm a is calculated as:

$$\text{SEM}_t(a) = \frac{\sigma_t(a)}{\sqrt{|\mathcal{D}(a, t)|}},$$

where $\sigma_t(a)$ is the sample standard deviation of the rewards observed for arm a up to time t, and $|\mathcal{D}(a, t)|$ is the number of observations for arm a up to time t.

In the context-based variant, if $\mathcal{D}_n(a, t)$ represents the set of n closest observations in terms of context for arm a up to time t, then the standard error of the mean is:

$$\text{SEM}_{n,t}(a) = \frac{\sigma_{n,t}(a)}{\sqrt{n}},$$

where $\sigma_{n,t}(a)$ is the sample standard deviation of the rewards in $\mathcal{D}_n(a, t)$.

6.5 Feature Importance

The RF is an ensemble learning method that aggregates the predictions of multiple decision trees to enhance predictive accuracy and reduce overfitting. One of the key advantages of the RF is its ability to quantify the importance of each feature in predicting the target variable. For each tree T_b, let $I_j^{(b)}$ represent the importance score of feature j in tree b. The overall importance

score of feature j in the RF is then computed as:

$$I_j = \frac{1}{M} \sum_{b=1}^{M} I_j^{(b)}.$$

The importance score $I_j^{(b)}$ for a single tree b is typically calculated based on the total reduction in the Mean Squared Error (MSE) attributed to splits involving feature j across all nodes where j is used. This can be mathematically represented as:

$$I_j^{(b)} = \sum_{t \in \mathcal{T}_b} \Delta\text{MSE}_t \cdot \mathbb{1}_{j \text{ is used in node } t},$$

where $\mathcal{T}_b$ denotes the set of all nodes in tree b, ΔMSE_t is the reduction in MSE at node t, and $\mathbb{I}\{\cdot\}$ is the indicator function. The reduction in MSE ΔMSE_t at node t is calculated as the difference between the MSE of the parent node and the weighted sum of the MSE of the child nodes:

$$\Delta\text{MSE}_t = \frac{N_t}{N_p}\text{MSE}(p) - \left(\frac{N_L}{N_p}\text{MSE}(L) + \frac{N_R}{N_p}\text{MSE}(R)\right),$$

where $\text{MSE}(p)$ is the MSE of the parent node, $\text{MSE}(L)$ and $\text{MSE}(R)$ are the MSE of the left and right child nodes, respectively, N_p, N_L, N_R, and N_t are the number of samples in the parent, left child, right child nodes, and node t, respectively.

To ensure that the importance scores $I_j^{(b)}$ for each feature in tree b are on a comparable scale, they are typically normalized such that the sum of the importances across all features for each tree equals 1. This can be expressed as:

$$\sum_{j=1}^{p} I_j^{(b)} = 1,$$

where p is the total number of features. Consequently, the overall importance scores for the RF are also normalized such that:

$$\sum_{j=1}^{p} I_j = 1.$$

This normalization ensures that the feature importances are expressed as proportions of the total importance, facilitating easier interpretation and comparison across different features. Feature importance provides valuable insights into which variables contribute most significantly to the predictions made by the RF. This information can be used to interpret the model's decisions, identify key drivers of the target variable, and potentially reduce the dimensionality of the dataset by removing less important features.

6.6 Trustworthiness of Weak Learners

In our RF model, each tree T_b is considered a weak learner, trained on a bootstrap sample of the training data and a random subset of features. Given the nature of weak learners, we calculate the in-sample MSE for the entire RF as well as for each individual tree. Let MSE_{RF} represent

the in-sample MSE of the entire RF, and MSE_b the in-sample MSE of the b-th tree:

$$\text{MSE}_{\text{RF}} = \frac{1}{N}\sum_{t=1}^{N}(y_t - f(x_a(t)))^2,\ \text{MSE}_b = \frac{1}{N_b}\sum_{t\in\mathcal{S}_b}(y_t - f_b(x_a(t)))^2,$$

where N is the total number of samples, y_i is the actual value, $\hat{y}_i$ is the prediction by the RF, N_b is the number of samples in the bootstrap sample $\mathcal{S}_b$ used to train tree b. To evaluate the contribution of each tree, we select the k trees with the lowest in-sample MSE and compute the ensemble prediction using only these k trees. The resulting MSE for the subsample of k trees, denoted MSE_k, is given by:

$$\text{MSE}_k = \frac{1}{N}\sum_{t=1}^{N}\left(y_t - \frac{1}{k}\sum_{b\in\mathcal{K}} f_b(x_a(t))\right)^2,$$

where $\mathcal{K}$ denotes the set of indices of the k selected trees. We then normalize these MSE values by the MSE of the full RF to assess the performance:

$$\text{Normalized MSE}_k = \frac{\text{MSE}_k}{\text{MSE}_{\text{RF}}}.$$

This process is repeated for $k \in \{1, \ldots, M\}$ to determine the number of trees necessary for a robust model.

In-sample MSE may not fully capture the model's robustness. Therefore, we also evaluate the out-of-sample MSE for the best k trees, still selected based on their in-sample MSE. Let $\text{MSE}_k^{\text{out}}$ denote the out-of-sample MSE for the k selected trees. We compare the normalized out-of-sample MSE values against those obtained from a random ordering of trees:

$$\text{Normalized MSE}_k^{\text{out}} = \frac{\text{MSE}_k^{\text{out}}}{\text{MSE}_{\text{RF}}^{\text{out}}},$$

where $\text{MSE}_{\text{RF}}^{\text{out}}$ is the out-of-sample MSE for the entire RF. This comparison reveals whether selecting a fraction of trees based on in-sample MSE can maintain or improve model robustness out-of-sample, or if utilizing all weak learners is necessary to ensure generalization.

7 Data

In previous chapters, we explain the bandit algorithms we examine in this research. To examine the performance and robustness of these Bandit algorithms, we present in this chapter the datasets that we use in our study. Given that the specific cost characteristics of brokers and their dependence on context features and spaces are unknown in the real-world trading cost dataset gathered by Robeco, and we are uncertain whether our developed (non-)linear CBs can detect these relationships, we first test and examine our CBs on a synthetic dataset with known cost distribution characteristics. We introduce our synthetic dataset in Section 7.1. This dataset is constructed to determine robust hyperparameters, perform model selection, and evaluate the strengths and weaknesses of the non-linear TS-based CB framework compared to other Bandit algorithms. The synthetic data allows us to simulate various scenarios and understand the behavior of our models under controlled conditions.

Additionally, we describe the Robeco dataset in Section 7.2. We outline the specific features of this dataset, including regional distinctions and continuous features. Given the variability in market characteristics across different regions, our research analyzes the performance of our models both globally and on a regional basis, resulting in one major Robeco dataset and two sub-datasets. Further detailed analysis of the Robeco dataset, including additional statistics and insights into specific context feature spaces, is provided in the Appendix (Section B).

By utilizing both synthetic and real-world data, we aim to comprehensively evaluate our models and ensure their robustness and applicability across diverse scenarios, including various cost distributions. Furthermore, our contribution includes the capability of our models to handle leptokurtic distributions, which are common in real-world trading cost data.

7.1 Synthetic Data

It is not guaranteed that the models we develop, which we explain in the methodology sections, will perform well in reality. To assess their performance, we propose testing the models on a set of brokers with known and predefined cost distributions. By evaluating the models on synthetic data, we can observe whether they effectively capture the underlying patterns in the data and make accurate predictions of the brokers expected costs.

The scenario we consider involves three brokers, denoted as A, B, and C, operating in several regions. The costs associated with these brokers follow Laplace distributions, to replicate the cost distributions of the brokers in real life. The broker costs are visualized in Figure 1. Each broker has distinct means and standard deviations depending on several context features, detailed in Table 1. From this table, it is evident that broker A performs best in the North region, while in the South region, brokers B and C exhibit similar mean costs, with broker A performing comparatively worse. Moreover, the varying standard deviations for brokers B and C in the South region enable us to observe how different bandit strategies react to these differences. These distinctions are visualized in Figures 2a and 2b.

Additionally, an examination of these distributions reveals insights into the hybrid version of

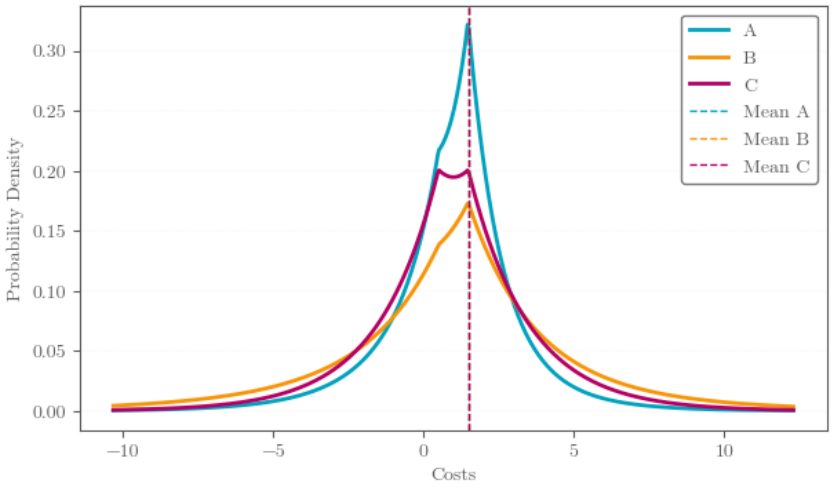


Figure 1: Visualization of the general performance of the synthetic brokers

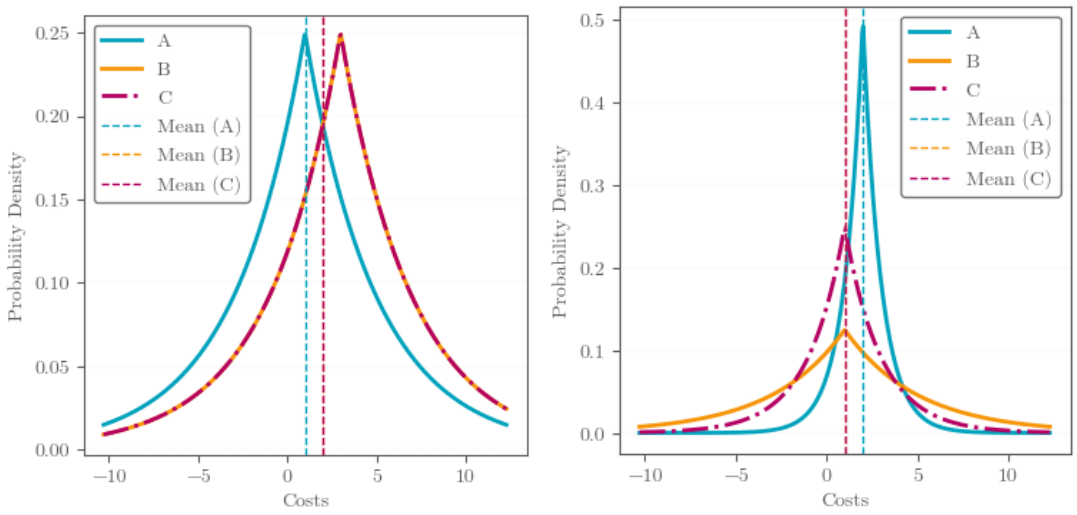
Broker	North	South	East	West
A	1 (2)	2 (1)	0 (1) + 0.2VOL	$\begin{cases} 2(1) & \text{if } VOL < 20\%, \\ 5(1) & \text{if } VOL \geq 20\%. \end{cases}$
B	2 (2)	1 (4)	2 (1) + 0.1VOL	$\begin{cases} 3(1) & \text{if } VOL < 30\%, \\ 4(1) & \text{if } VOL \geq 30\%. \end{cases}$
C	2 (2)	1 (2)	3.5 (1) + 0.05VOL	$\begin{cases} 4(1) & \text{if } VOL < 30\%, \\ 1(1) & \text{if } VOL \geq 30\%. \end{cases}$

Table 1: Represents the cost distributions of brokers A, B and C in regions North, South, East and West, which follow Laplace distributions, where the standard deviations are denoted within brackets

the model. Notably, the costs for brokers B and C in the North region can theoretically be represented solely by the hybrid column, rather than utilizing separate columns for each broker. We can explore how regularization techniques influence the model's feature selection process, potentially reducing the number of utilized features. Furthermore, given that the region feature space is one-hot encoded, the model only needs to identify a linear relationship between region and broker performance. Consequently, all models employing the same bandit algorithm, such as ϵ-greedy, UCB, or TS, may theoretically perform similarly.

Besides discrete-valued features, our synthetic dataset includes continuous features. To account for this, we extend the synthetic broker performance data to include trading in region East. However, in this region, broker performance is not solely determined by the region of the stock company but also by the percentual global volatility, denoted as VOL, which accounts for all regions. In our synthetic dataset, we consider a range of percentual volatility values from 10% to 40%. The relationship between broker performance and context in region East is defined such that broker A performs best in the low volatility interval $[10\%, 20\%]$, broker B excels in $[20\%, 30\%]$, and broker C dominates in $[30\%, 40\%]$, representing a high-volatility market.

Despite the inclusion of the VOL feature, there is no direct relationship between VOL and broker performance in our synthetic dataset. A mathematical expression of the broker costs in region East in terms of VOL can be found in Table 1, where the constant term is drawn from the



(a) Broker costs for region North *(b) Broker costs for region South*

Figure 2: Visualization of broker costs for regions North and South

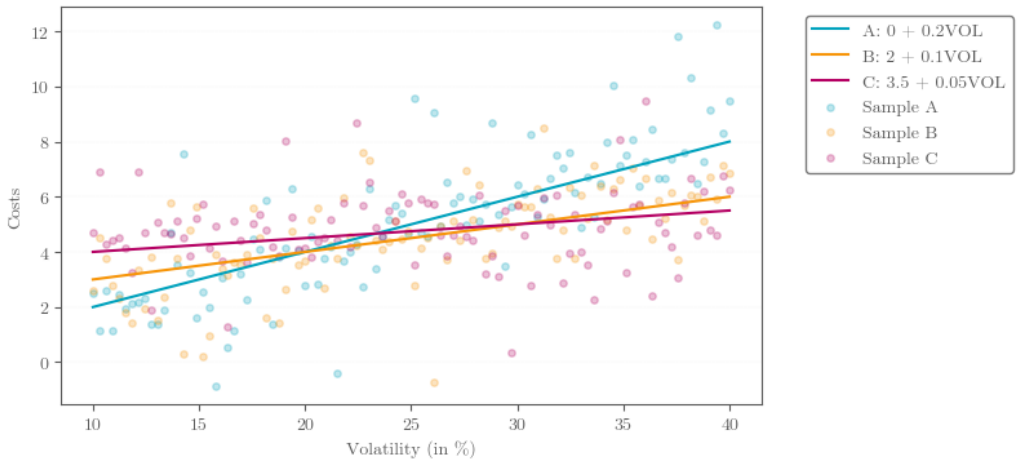


Figure 3: Visualization of the relationship between synthetic broker performances and VOL in region East

Laplace distribution. Additionally, we provide visualizations of broker performances in region East in Figure 3, showcasing both mean broker performance and a sampled version. Since we have defined linear relationships, theoretically, all estimation methods, like (Reguralized) Least Squares and RF, employing the same bandit approach could perform similarly in this context.

However, in real data, it is not guaranteed that the relationship between broker performance and context is linear. To explore this non-linearity in our synthetic scenario, we express broker performances in region West as step functions in terms of VOL, as defined in Table 1. We consider the same interval for the feature VOL as in region East and define our step functions such that brokers perform best on the same sub-intervals as in region East. Furthermore, we design the step functions to ensure that linear estimation methods would perform poorly. Figure 4 provides a visualization of the step functions and how a sample would look. Additionally, we illustrate how a linear fit on this sample appears, indicating that linear estimation methods in the CB would be inappropriate for this region. This visualization suggests that a limited number

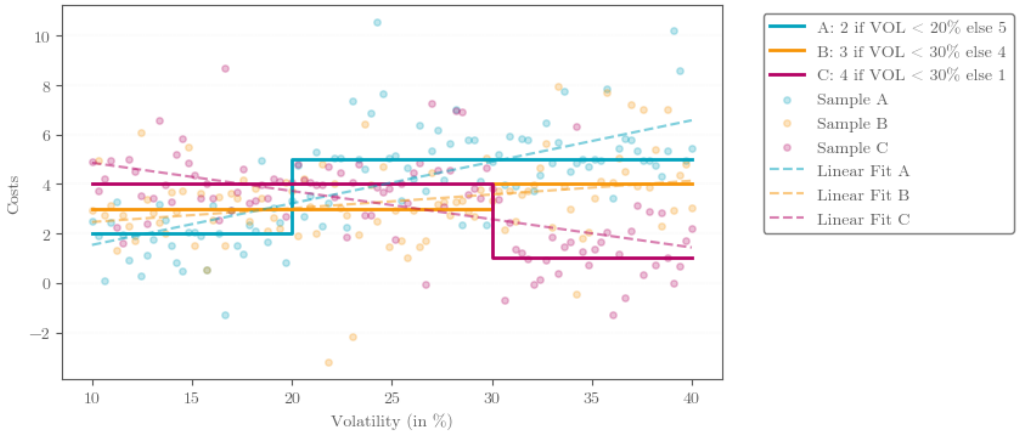


Figure 4: Visualization of the relationship between synthetic broker performances and VOL in region West

of orders would likely be allocated to broker B when using a linear model. However, it is evident from this visualization that broker B performs best within a certain interval.

Lastly, we introduce some noise features to the synthetic dataset, each following arbitrarily chosen distributions. This addition is crucial for evaluating how well the algorithms handle noisy data. The noise features are generated according to the following distributions: Gamma$(2, 2)$, Beta$(2, 5)$, Exponential(2), Weibull(2), and Lognormal$(0, 1)$. By including these diverse noise features, we aim to understand the robustness and effectiveness of the algorithms in discerning relevant information from irrelevant noise, thereby providing insights into their performance in real-world scenarios where noise is prevalent.

By defining multiple features, both discrete and continuous, that depend on each other, we can examine how our models interact with complex relationships. It is important to note that we define broker costs in a manner that ensures there is no universally "best" or "cheapest" broker across all regions. This deliberate design choice allows us to investigate whether CBs outperform the MABs. Through this comparison, we aim to gain insights into the relative efficacy of contextual information in decision-making and its impact on optimizing resource allocation strategies.

7.2 Robeco Data

The Robeco dataset used in this study comprises trading cost data for four anonymized brokers, denoted as A, B, C, and D, operating in both region one and region two, denoted as region $R1$ and $R2$ respectively. The data is collected and provided by Robeco, covering the entire year of 2023, from January 3rd to December 29th. The dataset includes a total of 24,254 observations, each representing a distinct trade. Of these, 6,708 trades were conducted in R1 markets, while the remaining 17,546 trades took place in region $R2$. Alongside region-specific data, the dataset provides several features that contextualize each order, including the participation rate, indicating the realized percentage of the average daily volume involved in each trade, the spread of security, and a volatility index corresponding to the security that is traded. This dataset offers a comprehensive view of trading activities across diverse markets and brokers, facilitating in-depth

Region	Broker	Mean	St.Dev	Skewness	Kurtosis	Median	Observations
R1	A	0.101	0.375	0.066	1.236	0.099	953
	B	0.153	0.509	0.300	1.115	0.132	857
	C	0.119	0.391	0.308	1.019	0.104	1135
	D	0.078	0.365	−0.330	1.132	0.095	1575
R2	A	0.123	0.342	0.648	3.953	0.097	1086
	B	0.137	0.430	−0.086	4.092	0.124	2394
	C	0.096	0.482	0.022	6.135	0.082	5640
	D	0.083	0.342	−0.200	11.351	0.073	4960

Table 2: Cost characteristics for brokers trading in regions $R1$ and $R2$ based on the filtered Robeco dataset. Statistics are obtained by bootstrapping mean costs through Dirichlet-distributed weights, summing scaled samples, and repeating this process $1,000,000$ times per broker. The table presents the mean, standard deviation, skewness, kurtosis, and median of the bootstrapped distributions, along with the number of data points in the filtered dataset. The inclusion of the median provides a more robust measure of central tendency, addressing data skewness.

analysis of market behavior, trading patterns, and broker performance. For further insights into the frequency of observations for specific features see Section B.1 in the Appendix, where we provide a more detailed overview of the dataset. Additionally, we filter the data, since we have many observations of orders corresponding to low values of participation rate and spread, but for higher values of these contexts the number of observations are very scattered. Therefore we decide to only consider orders with a participation rate lower than 0.3 and spread lower than 0.1, resulting in a filtered dataset. This filtering further illustrated in Section B.1. The filtered Robeco dataset includes $18,610$ observations, with $4,530$ corresponding to region $R1$ and $14,080$ to region $R2$.

In Table 2, we present the cost characteristics for brokers in regions $R1$ and $R2$ based on the filtered dataset. The statistics in the table are derived from bootstrapped estimates of mean costs. For each bootstrap iteration, we draw a sample of 50 observations from the dataset for a specific broker. The bootstrapping process involves assigning weights drawn from the Dirichlet distribution to each sample, summing the scaled samples, and repeating this procedure $1,000,000$ times for each broker. The table provides the mean, standard deviation, skewness, kurtosis, and median of the bootstrapped distributions. The observations column indicates the number of data points in the filtered dataset, not the number of bootstrapped samples. While the mean provides an average cost measure, it may not fully capture the distribution characteristics due to potential skewness in the data. Therefore, we also include the median, which acts as a skewness-adjusted mean and offers a more robust measure of central tendency, especially in the presence of skewed data distributions. The additional statistics, such as skewness and kurtosis, further describe the shape and dispersion of the cost distributions, providing a comprehensive view of the cost characteristics for each broker.

Based on the skewness-adjusted mean, indicated by the median, brokers perform more similarly in region $R1$ compared to region $R2$. In region $R2$, we expect that most orders will be allocated to broker D. However, broker C has a high standard deviation and the second lowest median, suggesting that a TS-based CB might also allocate a substantial fraction of orders to this broker. In region $R1$, it is likely that most orders will be routed to brokers A or D due to their lower

median costs. Broker A has a slightly higher standard deviation, which could result in a higher fraction of orders being routed to it. Additionally, despite broker B having the highest median costs, its higher standard deviation indicates that a TS-based CB may still allocate a significant fraction of orders to this broker. This allocation strategy takes advantage of the inherent variability in cost distributions to potentially discover lower costs for certain contexts.

After routing an order, the broker its performance is sampled from its empirical cost distribution. This sampling assumes that brokers perform in line with their historically observed cost distributions. The cost, quantified as the observed slippage relative to the VWAP as a fraction of the bid-ask spread, is modeled using an asymmetric generalized normal distribution. Such distributions are characterized by their asymmetry and leptokurtic nature. Understanding these distributions is crucial for accurately modeling and predicting broker performance. How we do that in our research is described in Sections 8.1 and 8.2. From the characteristics of the Robeco dataset as presented in table 2, we can conclude that the cost distributions are asymmetric and leptokurtic, as indicated by the skewness and kurtosis values.

However, the values of the context features are highly region-specific, indicating distinct market dynamics in each region. For instance, trading activity is generally lower in region $R1$, resulting in higher average participation rates for Robeco compared to region $R2$. Additionally, increased trading activity in region $R2$ can lead to higher volatility and wider spreads, which might explain why the average spreads in region $R2$ are approximately five times higher than those in region $R1$. Sections B.2 and B.3 provide more detailed insights into these region-specific feature characteristics. Given these market differences, we decided to evaluate the Bandits their performance and decision-making from three perspectives. First, at a global level, where we analyze the entire filtered Robeco dataset and examine the models their performance and allocation decisions by considering the region an order belongs to as context. This approach allows us to determine whether our models can detect the brokers their distribution characteristics as described above. Subsequently, we analyze our models separately for regions $R1$ and $R2$, resulting in the Robeco $R1$ and $R2$ datasets, respectively. As noted earlier and illustrated in Figure 22b, spreads are lower in region $R1$. Therefore, the Robeco $R1$ dataset includes only trades with a participation rate lower than 0.3 and a spread lower than 0.002, resulting in a dataset containing 4237 observations. The Robeco $R2$ dataset comprises 14080 observations, maintaining a broader range of trading activity and higher spread values.

In conclusion, this data Chapter highlights the dual approach of examining both synthetic and real-world datasets to evaluate the strengths and weaknesses of our developed Bandit algorithms, particularly the non-linear TS-based CB using RF. This comprehensive approach not only allows us to test our models their robustness and decision-making capabilities but also to observe how they handle costs that follow non-normal, leptokurtic distributions. This is evident from the characteristics of the Robeco dataset presented in Table 2, where high skewness and kurtosis values confirm the asymmetry and leptokurtic nature of the cost distributions. By leveraging these datasets, we aim to ensure that our Bandit algorithms are not only effective in various scenarios but also capable of accurately modeling and predicting broker performance in the presence of complex cost distributions.

8 Experimental Setup

To examine and evaluate the performance of various bandit models, as detailed in Chapters 5 and 6, we conduct a series of simulations. As previously noted in Chapter 7, our investigation encompasses two distinct datasets, which serve as the basis for model testing and evaluation. In order to accurately estimate the costs associated with the arms, we introduce a clustering and sampling technique in Section 8.2. Subsequently, Section 8.3 outlines our simulation methodology, designed to yield valid and comparable results. Finally, in Section 8.4 we describe how we achieve the routing probabilities for bandits that follow a TS-based approach.

8.1 (Asymmetric) Generalized Normal Distribution

As concluded and visualized in Section 7.2 the data seems to be Laplace distributed. Laplace distributions can be modelled using generalized normal distributions. Moreover, our data might always be symmetric. Therefore, we need a generic framework that is able to fit various distributions, be it symmetric or asymmetric.

The use of a generalized normal distribution to model data that exhibits Laplace-like characteristics, possibly with asymmetry, offers significant advantages due to its inherent flexibility in capturing various distribution shapes. This distribution can effectively accommodate heavier or lighter tails and incorporate skewness, making it well-suited for achieving a more accurate empirical fit. This flexibility enhances the reliability and robustness of statistical analyses and resampling procedures by providing a closer representation of real-world data characteristics. The probability density function (PDF) of the generalized normal distribution with location parameter μ and scale parameter α is defined as:

$$f(x;\beta,\mu,\alpha) = \frac{\beta}{2\alpha\Gamma\left(\frac{1}{\beta}\right)} \exp\left[-\left(\frac{|x-\mu|}{\alpha}\right)^{\beta}\right],$$

where x is a real number, $\beta > 0$, and Γ is the gamma function. The parameter β controls the shape of the distribution, determining whether it has heavier or lighter tails compared to a normal distribution ($\beta = 2$) or a Laplace distribution ($\beta = 1$). For $\beta = 1$, the distribution simplifies to the Laplace distribution, known for its peakedness around the mean.

To introduce asymmetry into the generalized normal distribution, a method similar to the asymmetric Laplace distribution is applied, as presented in Arellano-Valle et al. (2005). The PDF of the asymmetric generalized normal distribution is given by:

$$g(x;\beta,\kappa) = \begin{cases} \frac{2\kappa}{\kappa+\kappa^{-1}} f(x^{\kappa},\beta) & \text{for } x \geq 0, \\ \frac{2\kappa}{\kappa+\kappa^{-1}} f(-x^{\kappa},\beta) & \text{for } x < 0, \end{cases}$$

where $\kappa > 0$ is the asymmetry parameter. This formulation allows the distribution to capture asymmetry by adjusting the shape of the PDF differently for positive and negative values of x. When $\kappa = 1$, the distribution becomes symmetric, while for $\beta = 1$, it reduces to the asymmetric Laplace distribution. The moments, cumulative distribution function, and its inverse can be

derived directly from the generalized normal distribution framework Arellano-Valle et al. (2005).

8.2 Cost Bootstrapping Method

In Chapter 7, we provide a comprehensive description of the datasets utilized in our study, encompassing both the data gathered by Robeco and our synthetic dataset. In our synthetic dataset, containing observations for three brokers that are operating in four distinct regions, we know the relationships between order contexts and broker costs. However, in reality, we lack knowledge about the underlying distributions, dependencies between features, and the significance of certain features in specific regions. Consequently, we refrain from directly utilizing the known distributions of our synthetic data to replicate real-world scenarios. We further enhance this synthetic dataset by bootstrapping the costs for each broker given a specific context using a clustering technique. This approach enables us to assess the performance of our models on the synthetic dataset, even after sampling and bootstrapping costs for specific or closely related contexts. When our models perform well under these conditions, we can confidently apply the same technique to the Robeco data, facilitating robust analysis and comparison.

To sample a cost for a broker $a \in A$ for a specific context $x(t)$, we first estimate the distance between context $x(t)$ and the contexts in our historical dataset. We achieve this by computing the Euclidean distance. However, since not all features have the same bounds and to ensure each feature contributes equally to the distance calculation, we standardize the data beforehand. This standardization transforms the feature space so that the lower bound is zero and the upper bound is one.

We select M observations from the historical data that are closest to $x(t)$, resulting in a set of cost observations denoted by $\mathcal{C}(x(t))$. In the Results section, M is referred to as n_{closest}. If more than M observations share the same distance as the M-th closest observation, we randomly select n_{closest} observations from those within this distance range to form the set $\mathcal{C}(x(t))$. We assume that the costs in this set follow an asymmetric generalized normal distribution (AGN). For broker a, we then draw the cost for context $x(t)$ from an AGN with parameters estimated from the costs in the sample $\mathcal{C}(x(t))$, specified by:

$$c_a(x(t), \mathcal{C}(x(t))) \sim \text{AGN}(\mu_{\mathcal{C}(x(t))}, \alpha_{\mathcal{C}(x(t))}, \beta_{\mathcal{C}(x(t))}, \kappa_{\mathcal{C}(x(t))}),$$

where $c_a(x(t))$ represents the cost for broker a given context $x(t)$ and the M closest observations in the set $\mathcal{C}(x(t))$. Here, $\mu_{\mathcal{C}(x(t))}$ is the location parameter, $\alpha_{\mathcal{C}(x(t))}$ is the scale parameter, $\beta_{\mathcal{C}(x(t))}$ is the shape parameter, and $\kappa_{\mathcal{C}(x(t))}$ is the asymmetry parameter of the AGN derived from the costs in $\mathcal{C}(x(t))$.

It is important to consider the trade-off involved in selecting the size M of the set $\mathcal{C}(x(t))$. A larger M might include observations that are not very similar to each other in terms of contexts, thereby introducing additional noise into the bandit. This increased noise can make it more difficult to detect differences in context-related costs. On the other hand, choosing a smaller M could lead to an underrepresentation of the true cost distribution for a given context $x(t)$. Striking the right balance is crucial to ensure that the sampled costs accurately reflect the real

cost distribution while maintaining sufficient context similarity to minimize noise.

8.3 Simulation

In our research, we test our models with simulations. A simulation consists of two phases. The first phase is a warm-up phase, which results in t_{warmup} observations, where t_{warmup} orders are allocated equally among the brokers. Following this phase, the simulation continues allocating the orders for $T - t_{warmup}$ timestamps using a specific bandit. Since the start or warm-up phase can significantly influence the performance of the models and, since orders are allocated randomly, we perform the same simulation for T timestamps for the same model multiple times. Therefore, for one test, we perform N trials, allowing us to measure the averages and standard deviations in terms of (pseudo-)regret and the fraction of orders routed to the cheapest broker. This approach also provides insights into how the models evolve over time. By running the simulation over T timestamps, we can observe after how many observations the model achieves stable performance. This thorough testing ensures a robust evaluation of the models, accounting for variability and providing a clearer picture of their effectiveness.

We can determine a sufficient time horizon size from our simulations, which indicates the number of observations required for the model to perform well. It is important to note that there is a trade-off between the number of observations used to train the models and the model's responsiveness to market shocks or changes in market structure, which can significantly impact broker performances. We will test this trade-off in one of the scenarios introduced in the next subsection.

Since the start of the simulation involves a high degree of randomness, and to avoid overfitting to the initial few observations—which could lead to incorrect convergence—we provide the algorithms with a warm start, consisting of t_{warmup} time steps. Additionally, some of the estimation techniques employed in our models require a minimum number of observations before they can accurately estimate the model parameters, making this warm-up phase essential. After the t_{warmup} observations, we conduct another T runs for the main simulation. This approach ensures that the models have a stable foundation before the actual testing phase begins, improving the reliability of our evaluation.

We can test the CBs in two different ways using the synthetic and Robeco datasets. The first approach involves drawing a context from the observed contexts in the dataset and subsequently bootstrapping the broker cost using the method described in Section 8.2. However, this approach is not entirely practical, because we still need to bootstrap the broker performance. This necessity arises from the fact that we lack observations of costs for all brokers for specific contexts in the Robeco dataset, since Robeco is only able to observe the costs for the broker to which the order was allocated. Therefore, even when drawing a context from the dataset, we must employ bootstrapping to estimate the costs for brokers that were not selected for that context.

Moreover, the distribution of feature occurrences, and consequently the distribution of the context space, may not be uniformly distributed. This non-uniformity can lead to an overfitted model that performs well only in specific context regions while failing to generalize across the

entire context space. To address this issue and to ensure that edge cases for specific features or the entire context space are adequately represented, we decide to uniformly draw feature values randomly from predefined ranges. This approach helps in creating a more balanced and comprehensive dataset, allowing us to evaluate the model's performance more effectively across various contexts. More mathematically, we sample a context vector $\mathbf{x}(t)$ for time t such that each feature value $S_i(t)$ for $i \in L$, is drawn uniformly from its predefined range. This ensures that the context space for each context feature is evenly explored, preventing overfitting to specific context regions and allowing us to adequately test the model's performance across a diverse set of contexts.

8.4 Achieving Routing Probabilities

In Sections 4.3 and 5.3, and in Chapter 6, we describe various approaches for routing orders using a TS-based method. Since these approaches are Bayesian, the same order $\mathbf{x}(t)$ might not always be allocated to the same broker a_j each time. The TS-based bandits route orders with a certain probability $\mathbb{P}(a_j|\mathbf{x}(t))$, depending on how well broker a_j distinguishes itself from other brokers and outperforms them. The algorithm allocates to broker a_j with a higher probability if it is more certain that this allocation is optimal.

These allocation probabilities are not directly visible in a contextual environment based on historical orders. To examine how the CBs act for specific context variables and to estimate the routing probabilities, we generate R new orders, each with its own context. As previously noted, the same order can be routed to different brokers upon repeated trials with the TS-based CB. To estimate the routing probability $\mathbb{P}(a_j|\mathbf{x}(t))$, we route each order P times, resulting in a total of $R \times P$ orders. For each context variable, we classify the routed orders into n buckets, $\{B_1, B_2, \ldots, B_n\}$, each representing a range of values of the context variable. For each bucket B_k, we calculate the routing probability for each broker a_j by determining the fraction of orders in bucket B_k routed to a_j, denoted as $\hat{\mathbb{P}}(a_j|B_k)$. Mathematically, this can be expressed as:

$$\hat{\mathbb{P}}(a_j|B_k) = \frac{1}{|B_k|} \sum_{\mathbf{x}(t) \in B_k} \mathbb{1}(\mathbf{x}(i) \to a_j),$$

where $\mathbb{1}(\mathbf{x}(t) \to a_j)$ is an indicator function that equals 1 if order $\mathbf{x}(t)$ is routed to broker a_j, and 0 otherwise. Since we test our Bandits over N trials, each Bandit acts differently in each trial. Thus, for each bucket B_k, we obtain N series of routing probabilities $\{\hat{\mathbb{P}}_1(a_j|B_k), \hat{\mathbb{P}}_2(a_j|B_k), \ldots, \hat{\mathbb{P}}_N(a_j|B_k)\}$. From these, we can derive the mean routing probability $\bar{\mathbb{P}}(a_j|B_k)$ and the standard deviation bounds for each bucket:

$$\bar{\mathbb{P}}(a_j|B_k) = \frac{1}{N} \sum_{t=1}^{N} \hat{\mathbb{P}}_t(a_j|B_k),$$

$$\sigma_{\mathbb{P}(a_j|B_k)} = \sqrt{\frac{1}{N} \sum_{t=1}^{N} \left(\hat{\mathbb{P}}_t(a_j|B_k) - \bar{\mathbb{P}}(a_j|B_k) \right)^2}.$$

These results allow us to understand the routing behavior of the CBs concerning specific context variables and evaluate the robustness of the routing decisions.

9 Results

Building on the foundational concepts and methodologies outlined in earlier chapters, and utilizing the experimental setup detailed in Chapter 8, we now present the results of our study. This chapter focuses on evaluating our proposed algorithms and their performance on both synthetic and real-world datasets.

Prior to this evaluation, we conducted hyperparameter tuning for specific estimation techniques and bandit algorithms, with detailed results provided in Appendix C. The tuning process, performed on the synthetic dataset described in Section 7.1, involved selecting a subset of models from various bandit algorithms. Based on the tuning outcomes, we identified three bandit algorithms employing a Thompson Sampling-based approach to balance exploration and exploitation. The first bandit algorithm is the MAB, labeled as $ThompsonWheel$, which is discussed in Section 4.3 and serves as our benchmark for comparison with the CBs. The second algorithm is a linear CB utilizing OLS, denoted as LTS. The third algorithm is our own developed non-linear CB employing Random Forest Regressors, referred to as $RFTS$.

Additionally, our tuning analysis reveals that the steepest learning curve occurs within the initial 500 observations. Recognizing that early observations significantly impact (pseudo-)regret per order on the long-term, we have adjusted the warmup phase to $T_{warmup} = 500$ to allow sufficient exploration, particularly for the non-linear CBs. To ensure robust long-term performance evaluation, we extend the total simulation time to $T = 4000$. We test each bandit algorithm with specific settings thirty times, hence $N = 30$, adhering to the Central Limit Theorem. To estimate and approximate the costs for a specific broker in a given context, we use the observed costs from the closest fifty observations in the historical data, as explained in Section 8.2, hence $n_{closest} = 50$. The synthetic dataset is designed to consist of 20,000 trades, which closely matches the total number of trades in the entire Robeco dataset.

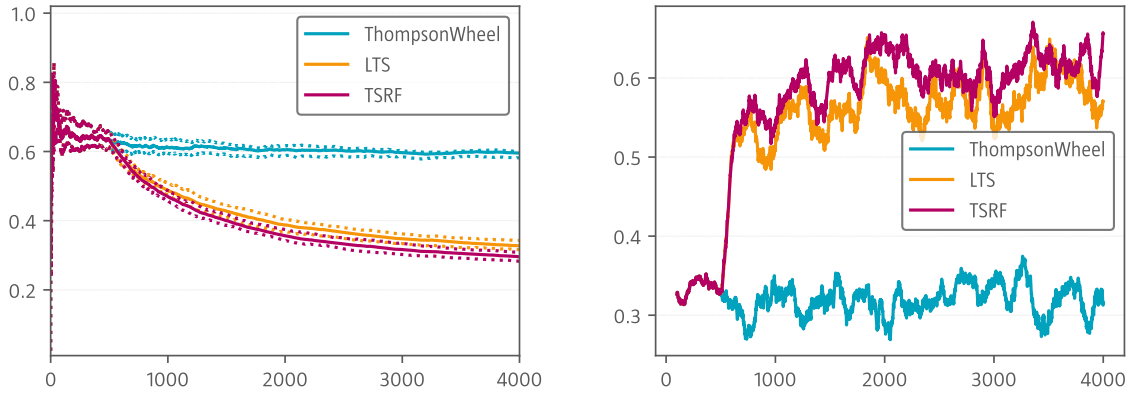
Subsequently, we assess the performance of the three Bandits on this synthetic dataset in Section 9.1. We analyze the strengths and weaknesses of the selected bandit algorithms based on our understanding of the dataset. With this insight, and having demonstrated that the models perform adequately on noisy data, we then investigate whether our CB algorithms can also surpass the MAB algorithms on the Robeco dataset. This investigation is presented in Section 9.2.

9.1 Results on Synthetic Data

In Appendix C, the hyperparameters are tuned and three Bandit algorithms are selected. In this section, we test the different models with various settings on a fictitious scenario, where we know the distributions in advance, allowing us to observe how the different models perform on this data. Section 9.1.1 evaluates the Bandits' performance and provides insights into the important features of the $TSRF$. Then, in Section 9.1.2, we visualize the routing probabilities based on selected context variables.

9.1.1 Performance Evaluation

In this section, we evaluate the performance of the Bandits and provide insights into feature importances. Figure 5 illustrates the performance of the Bandits, revealing that both the LTS and $TSRF$ significantly outperform the $ThompsonWheel$. This superior performance can be attributed to the dependencies present in the synthetic dataset between the costs and the contexts, which vary across brokers. Additionally, it is evident that the $TSRF$ outperforms the LTS. This can be explained by the presence of non-linear dependencies in region $WEST$, which $TSRF$ can capture more effectively. Furthermore, the linear dependencies in region $EAST$ influence decision-making and thus the performance of the LTS in other regions. These dependencies are further illustrated in Section 9.1.2.



(a) Pseudo-regret (per order) over time

(b) Fraction routed to best broker

Figure 5: Pseudo-regret and fraction routed to best broker of the Bandits on the synthetic dataset.

Figure 6 provides an overview of the feature importance of all features in the $TSRF$. The feature names are labeled on the x-axis, while the feature importance score is indicated on the y-axis. Features starting with Z refer to hybrid columns, which may contain non-zero values for each arm an order is routed to. Features starting with A, B, or C correspond to the respective brokers, and features with numbers in their names denote noise features.

Among the features, the hybrid feature Z_East has the highest importance score, indicating that splitting on Z_East significantly improves the RF its predictions, which can be explained by the higher costs in region $East$ compared to regions $North$ and $South$. Additionally, Z_VOL is another variable with a high importance score, suggesting that splitting on volatility substantially reduces the MSE in the children nodes. Features Z_North and Z_South also hold notable importance, enabling the Random Forest to distinguish orders based on regions. Notably, if these feature values are zero for a given context and for Z_East, the orders are categorized as belonging to region $WEST$.

It is also observed that there is some ordering in selecting the noise features. A reason for selecting a hybrid noise feature could be the absence of a better alternative, i.e., no informative feature in the subset of m drawn features. When splitting on a noise feature reduces the MSE in the children nodes, this could be due to pure luck or might indicate slight overfitting. A noise

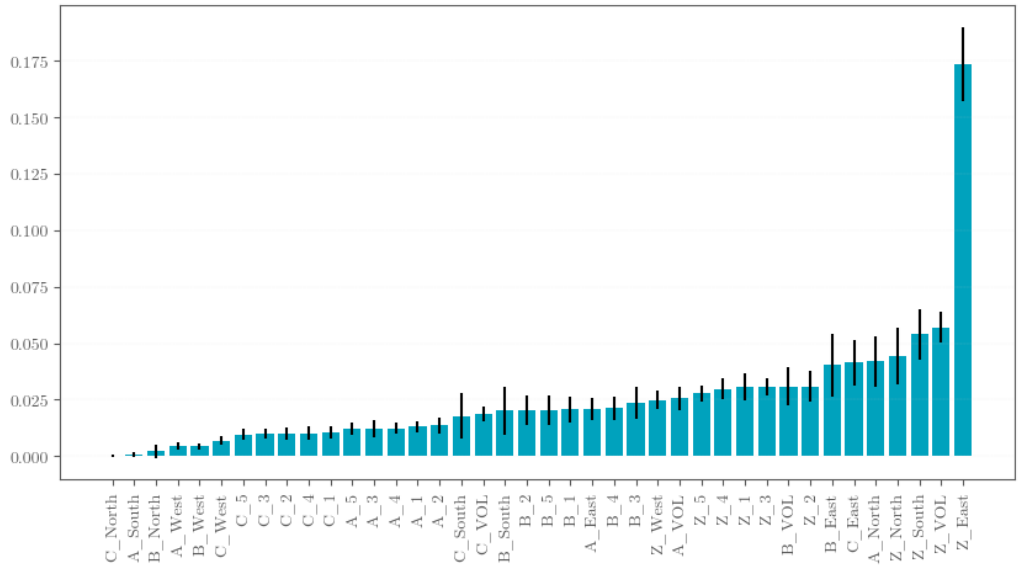


Figure 6: Feature importance for TSRF on the synthetic dataset. The y-axis represents the importance score, and the x-axis lists the features.

feature associated with broker B might be selected because it helps the algorithm determine whether an order in the data was routed to broker B by checking if the noise feature is non-zero since the feature value can only be non-zero if the order was routed to that specific broker. The fact that regional features corresponding to a specific broker have very low scores suggests redundancy, as the hybrid columns already provide sufficient information. This indicates that the model efficiently uses hybrid features to capture the necessary context for routing decisions.

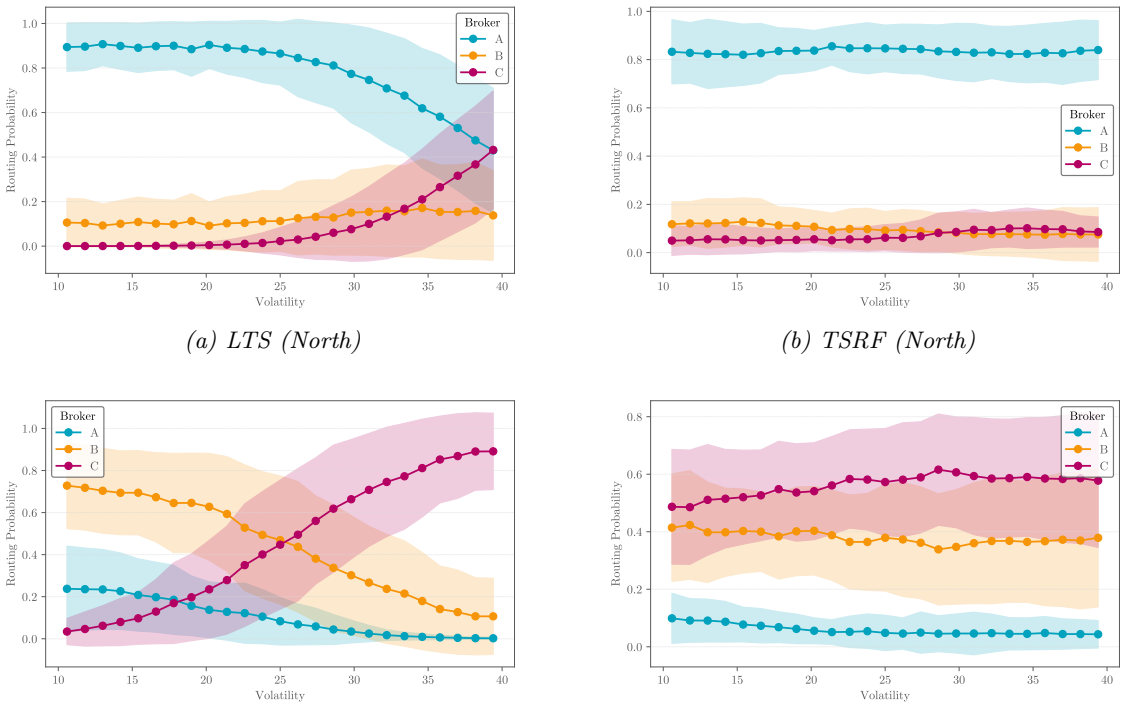
9.1.2 Routing Probabilities

In this section, we will further explore the LTS and $TSRF$ models by investigating their routing probabilities per region, as described in Section 8.4. We set $R = 10,000$ and $P = 100$, meaning that we route R orders P times to the CB, resulting in a total of $1,000,000$ observations. This extensive sampling allows us to accurately estimate the routing probabilities. To provide a precise yet clear overview of these probabilities, we aggregate the routed orders into 25 buckets. This approach ensures that our figures reflect the routing tendencies of the models in a comprehensible manner.

In region $North$, broker A is on average the cheapest. This is also detected well by both CBs as can be observed in Figures 7a and 7b. In region $South$, brokers B and C have on average the lowest costs. This is detected effectively by the $TSRF$, as shown in Figure 7d. The costs for broker B have a standard deviation that is twice as high as for broker C, which likely explains why the probabilities for broker C are higher. This indicates that the $TSRF$ prefers more certainty about the mean costs, aligning with our preference.

The LTS also performs well in region $South$, where most orders are routed to the cheaper brokers. However, while in region $North$ the probabilities for broker A decline with increasing volatility, in region $South$ the routing probabilities are unstable when plotted against volatility. In reality, there is no relationship between volatility and costs in regions $North$ and $South$. The typical

slopes of the probabilities can be attributed to the cost logic in region *East*, which influences the models' decisions for other regions.



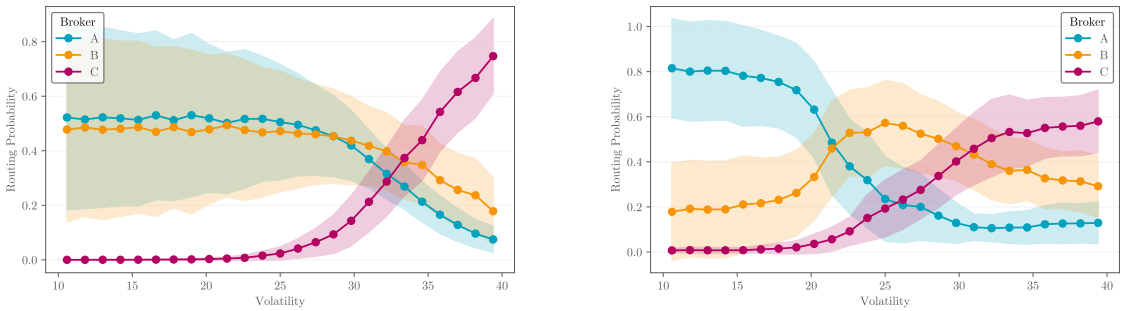
(a) LTS (North) *(b) TSRF (North)*

(c) LTS (South) *(d) TSRF (South)*

Figure 7: Routing probabilities of the LTS and TSRF in the North and South regions, based on the synthetic dataset. The plot displays the average probabilities across 30 models with shaded areas indicating standard deviation bounds, plotted against the volatility context feature. In the North region, broker A has the lowest average costs, while in the South region, brokers B and C exhibit the same lowest average costs, with broker C having lower standard deviations in costs.

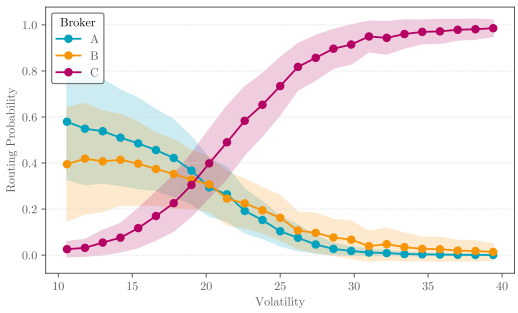
For regions *East* and *West*, it is known that there is a relationship between costs and volatility: broker A is the cheapest at low volatility, broker C is the best at high volatility, and broker B is optimal when volatility is between twenty and thirty. Figures 8b and 8d illustrate that the $TSRF$ effectively captures this relationship. The algorithm adjusts its routing probabilities aggressively in response to increasing volatility, with the non-linearity in its decisions clearly visible. It is worth noting that the transition from low to average volatility is somewhat smoothed due to our cost estimation technique, which simply takes the n closest observations based on context and samples a cost for a specific context. Using a more sophisticated clustering technique capable of better handling non-linear data might result in a more aggressive $TSRF$ response.

From Figures 8a and 8c can be observed that the LTS performs worse compared to the $TSRF$, but it is still capable of increasing the probability of routing to broker C for higher volatility, while maintaining high routing probabilities for broker A and B at lower and average volatility levels. It is important to note that, for the LTS, capturing the relationships between volatility and costs in regions *East* and *West* also affects routing decisions in regions where volatility is essentially a noise feature. Consequently, the LTS is not very robust against significant differences between regions with distinct characteristics.

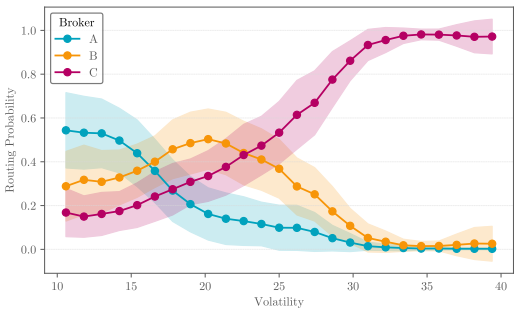


(a) LTS (East)

(b) TSRF (East)



(c) LTS (West)



(d) TSRF (West)

Figure 8: Routing probabilities of the LTS and TSRF in the East and West regions, based on the synthetic dataset. The plot shows the average probabilities across 30 models, with shaded areas representing standard deviation bounds, plotted against the volatility context feature. In both the East and West regions, broker A is the most cost-effective for low volatility markets, broker C for high volatility markets, and broker B for markets with average volatility.

These results also shows that using the closest fifty observations, so $n_{closest} = 50$, for bootstrapping and sampling the costs for each broker given a certain context, provides sufficient insights given specific context regions, since they are in line with the known broker behaviour characteristics for specific context regions. Since the results indicate that the routing probabilities of the LTS are sensitive to and influenced by certain linearities in the data and the model, these probabilities are less informative. Therefore, we will only demonstrate the routing probabilities of the $TSRF$ in the rest of our research. The $TSRF$ has proven to be effective in handling different types of data, whether linear or non-linear, and can distinguish various subsets of the data. Moreover, the routing probabilities provide a very clear overview of its decisions, which align with the known cost distributions. In the next section, Section 9.2, we apply the same Bandits to the real data to examine whether we can also detect similar relationships.

9.2 Results on Robeco Data

In the previous section, we demonstrated that our developed CBs can outperform MABs when there are relationships between costs and order contexts. Moreover, our methods for obtaining insights into the decisions of the CBs align with the known characteristics of the synthetic data, indicating that we have developed a non-linear CB that not only works but is also transparent and understandable. In this section, we will apply the TS-based Bandits to the data of Robeco, detailed in Section 7.2. Besides, Sections B.2 and B.3 offer detailed insights into region-

specific feature characteristics, highlighting differences in market dynamics across regions $R1$ and $R2$. Given these market differences, we decided to evaluate the Bandits their performance and decision-making from three perspectives.

First, at a global level, we evaluate the performance of the Bandits in Section 9.2.1, examining whether we can outperform the $ThompsonWheel$, which is the TS MAB, while only using regional context features, indicating whether orders belong to region $R1$ or $R2$. This approach allows us to determine whether our models can detect the brokers their cost distribution characteristics, including some leptokurtic characteristics, which is one of the contributions of our research.

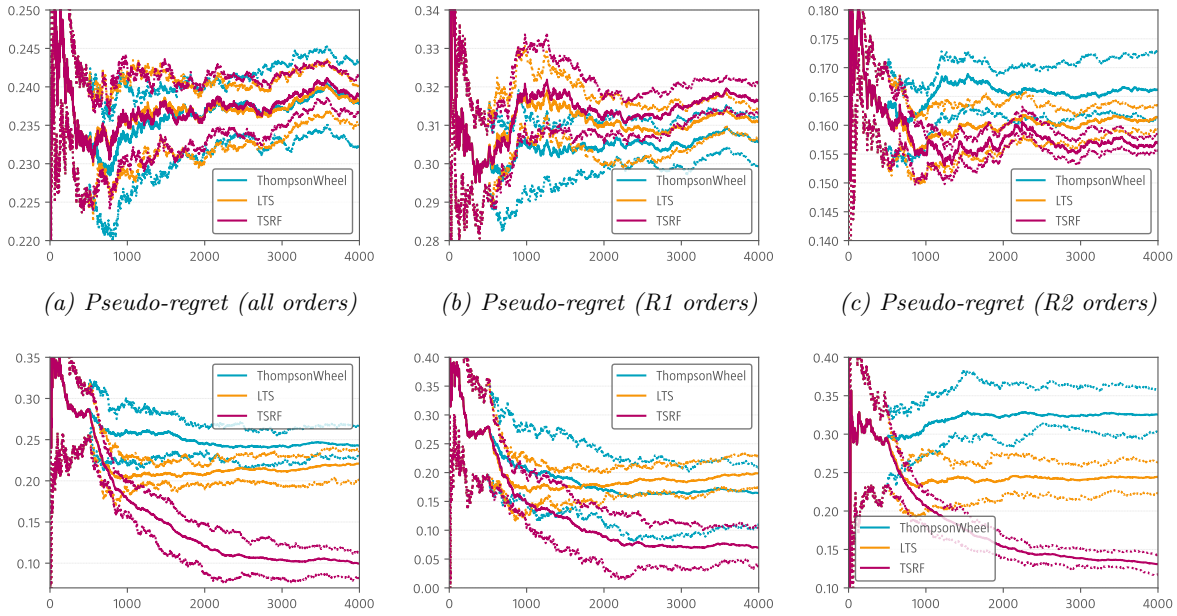
Subsequently, we analyze our models separately for regions $R1$ and $R2$, which we describe in Sections 9.2.2 and 9.2.3 respectively. In these sections, we examine whether our developed CBs can outperform the $ThompsonWheel$ using context features such as participation rate, spread, and volatility. Additionally, we investigate whether the broker performance is stable over time by including the date of an order as a context feature. By doing this, we can determine whether our CBs are able to dynamically select a range of observations that are useful for a specific time period, where one would have to determine a window size when using a MAB. Finally, in Section 9.2.4, we will provide some recommendations for Robeco.

9.2.1 Global Level Performance Analysis

In this section, we assess the performance of the TS-based Bandits from a global perspective by simulating them on the global Robeco dataset, which only includes the context of whether an order belongs to region $R1$ or $R2$. Figure 9 presents the performance, measured by (pseudo-)regret per order, for the three Bandits across both regions, with results split by region. The pseudo-regret figures do not immediately reveal a clear advantage of one Bandit over the others. It is important to note that, as detailed in Section 7.2, the brokers their cost distributions exhibit leptokurtic behavior. This leptokurtic nature means that the mean costs used to calculate pseudo-regret may not fully represent the underlying cost distributions due to the inherent uncertainty in the bootstrapped mean cost of a broker.

Despite the real-world challenge of observing regret directly, we can track its progression over time since we sample the regrets in our simulations. The subplots in Figure 9, which show regret per order, demonstrate that the $TSRF$, our developed non-linear TS-based CB, significantly outperforms the other Bandits. This suggests that $TSRF$ is better equipped to handle the leptokurtic cost distributions, unlike the LTS method. The regret plots underscore that relying solely on mean costs to calculate pseudo-regret may not be optimal for leptokurtic distributions. While $TSRF$ shows the highest average pseudo-regret across regions, it also outperforms the $ThompsonWheel$ in terms of regret by nearly 60%. Figure 40 further illustrates the fraction of orders routed to the optimal broker based on pseudo-regrets. Although this fraction is informative in non-leptokurtic contexts, it may not fully capture the impact of leptokurtic distributions on decision-making, potentially limiting the interpretability of the results.

Figure 10 shows the routing probability to a certain broker, estimated using the $TSRF$ CB, based



(a) Pseudo-regret (all orders) *(b) Pseudo-regret (R1 orders)* *(c) Pseudo-regret (R2 orders)*

(d) Regret (all orders) *(e) Regret (R1 orders)* *(f) Regret (R2 orders)*

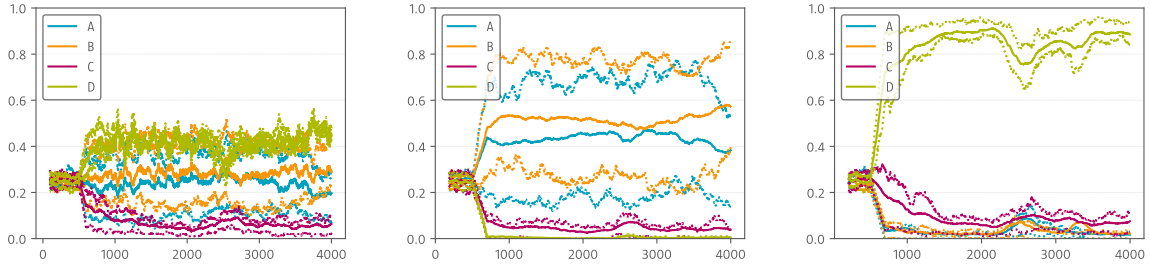
Figure 9: (Pseudo-)Regret per order over time, as shown by the x-axis (time) and y-axis (regret). This data is obtained from simulations on the global Robeco dataset, where brokers exhibit leptokurtic cost distributions. The discrepancy between pseudo-regret and regret is attributed to the leptokurtic nature of the cost distributions.

on the foregoing 100 orders for each timestamp, both from a global perspective and splitted up for both regions. It can be observed that the routing probabilities for all orders are meaningless, since the routing probabilities based on the orders corresponding to a specific region really differs per region. In region $R2$ most of the orders are routed to broker D, which meets our expectations, since the costs are the lowest for this broker in this region. The reason why a certain fraction is allocated to broker C in region $R2$ is that this broker has a larger standard deviation in its costs, which is intersting for a Bandit following a TS approach, since there might be a probability that broker C has low costs.

Most of the orders belonging to region $R1$ are routed to broker B, which has the largest standard deviation in its cost. Also broker A has a high routing probability, which can be explained by the fact that it is one of the cheapest brokers in this region. This high probability for broker A might be the reason why broker D has a probability of almost zero in this region.

Figure 10 illustrates the routing probabilities to various brokers, as estimated by the $TSRF$ CB, based on the preceding 100 orders at each timestamp. The figure provides both a global perspective and a breakdown by region. It is evident that routing probabilities across all orders are not very informative, as they vary significantly between regions. In region $R2$, most orders are routed to broker D, aligning with expectations given that D has the lowest median costs in this region. However, a non-negligible fraction of orders is also allocated to broker C, which, despite having higher costs on average, exhibits a larger standard deviation in costs. This variability makes broker C a potential candidate for exploration by the Thompson Sampling-based approach, as there is a chance it could have low costs. Conversely, in region $R1$, a majority

of orders are routed to broker B, which has the highest standard deviation in costs. Broker A also receives a significant share of the routing probability, likely due to its status as one of the most cost-effective brokers in this region. The relatively high probability assigned to broker A may explain the near-zero routing probability for broker D in this region. However, the low routing probability for broker D could also be attributed to the negative skewness in its cost distribution, whereas the other brokers exhibit positive skewness in region $R1$.



(a) All orders *(b) Region R1 orders* *(c) Region R2 orders*

Figure 10: Routing probability to a broker, estimated using the TSRF, based on the preceding 100 orders for each timestamp, with time shown on the x-axis and routing fraction on the y-axis. The data is derived from simulations on the global Robeco dataset. In region R2, broker D has the lowest median costs, while in region R1, the cost distributions of the brokers are less distinct.

Figure 11 illustrates the feature importance scores for the TSRF model on the global Robeco dataset. The figure highlights that, in addition to the hybrid features denoted by a Z in their names, the features A_R1, B_R1, and D_R2 exhibit the highest average importance. This is consistent with the observed high routing probabilities for brokers A, B, and D in the $R1$ and $R2$ regions, respectively. The substantial importance of the hybrid features Z_R1 and Z_R2 underscores the benefit of incorporating regional splits in reducing the model its variance in mean squared error.

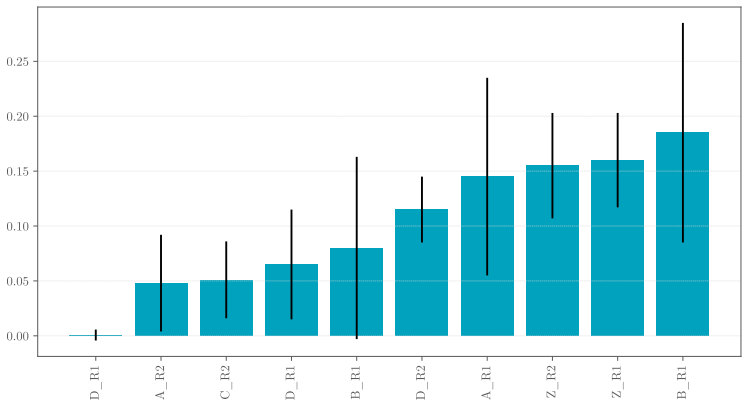


Figure 11: Feature importance for the TSRF model on the global Robeco dataset. The y-axis displays the importance scores, while the x-axis lists the features. Features with a Z in their names represent hybrid features, whereas the remaining features correspond to brokers A, B, C, and D.

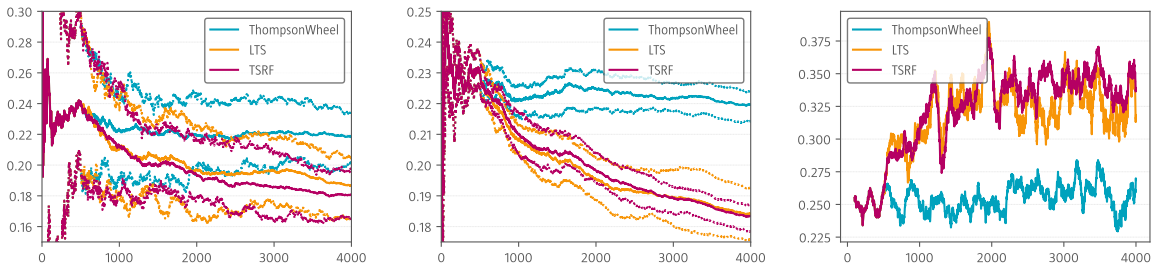
The limitations of the pseudo-regret may stem from the method used to bootstrap the cost distribution for each specific context. We derive the cost distribution by selecting the $n_{closest}$

observations based on the context of an order. In cases where context is defined solely by region, there are numerous historical observations within the same distance range, i.e., within the same region. Consequently, we draw $n_{closest}$ observations from these closest records to estimate the cost distribution, as detailed in Section 8.2. For this study, $n_{closest}$ is set to fifty, which effectively captures the cost distributions for specific context regions, especially when multiple context features are considered, as demonstrated in Section 9.1 with the synthetic dataset. However, given the abundance of observations in the global dataset, it might be prudent to increase the $n_{closest}$ value. Despite this, we achieved sufficient results in terms of regret with the current setting and will explore regional variations by incorporating additional context features in future analyses.

From this section, we conclude that our methods can effectively detect differences in broker performance across specific contexts, where each context region exhibits its own leptokurtic cost distribution. These findings align with our expectations. Given that market dynamics vary by region, we will proceed with a regional-focused investigation of the Bandits on the Robeco data.

9.2.2 Region R1 Performance Analysis

In this section, we present the results from simulations performed on the Robeco $R1$ dataset, which includes historical trading cost observations from region $R1$. These simulations account for context features such as participation rate, spread, volatility, and date. Figure 12 illustrates both the (pseudo-)regret per order and the fraction of orders routed to the best broker over time. The results indicate that both the LTS and the $TSRF$ outperform the $ThompsonWheel$. After routing 4000 orders across 30 trials, the $TSRF$ demonstrates an average cost reduction of nearly 18%, as assessed by the regret metric. Additionally, the observed decline in the regret per order suggests that the cost reduction could potentially be even greater. Despite the LTS and $TSRF$ showing similar performance on average based on pseudo-regrets, as illustrated in Figure 12b, the $TSRF$ exhibits more stability, reflected by narrower standard deviation bounds represented by the dotted lines. Furthermore, Figure 15a reveals that the $TSRF$ outperforms the LTS when evaluating real regret. The performance of the $TSRF$ suggests that it may be more effective in handling the leptokurtic cost distributions observed among brokers compared to the LTS.



(a) Regret (per order) *(b) Pseudo-regret (per order)* *(c) Fraction routed to best broker*

Figure 12: (Pseudo-)Regret per order over time, as shown by the x-axis (time) and y-axis (regret), and the fraction of routing to the best broker over time for the Bandits, based on the Robeco dataset from region R1.

Figure 13 displays the feature importances for the $TSRF$ on the Robeco $R1$ dataset. The ana-

lysis reveals that all hybrid features, denoted with a Z prefix in their names, are informative, with the spread feature emerging as the most significant. In contrast, the differences in importance scores among the broker-specific features, indicated by the anonymized broker names, are relatively minor. Notably, the *DateNumber* columns corresponding to a specific broker, which represent the dates on which the orders were placed, show the highest average importance scores. This suggests that the decision-making process for broker allocation may have exhibited some instability over time. Additionally, features specific to broker D consistently score the lowest in terms of importance, indicating that these features contribute less to the model's routing decisions compared to others.

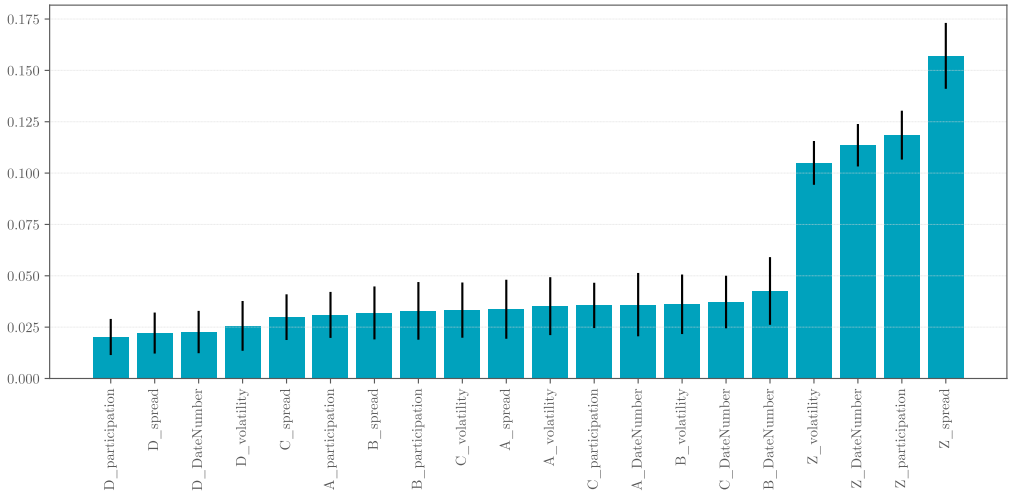
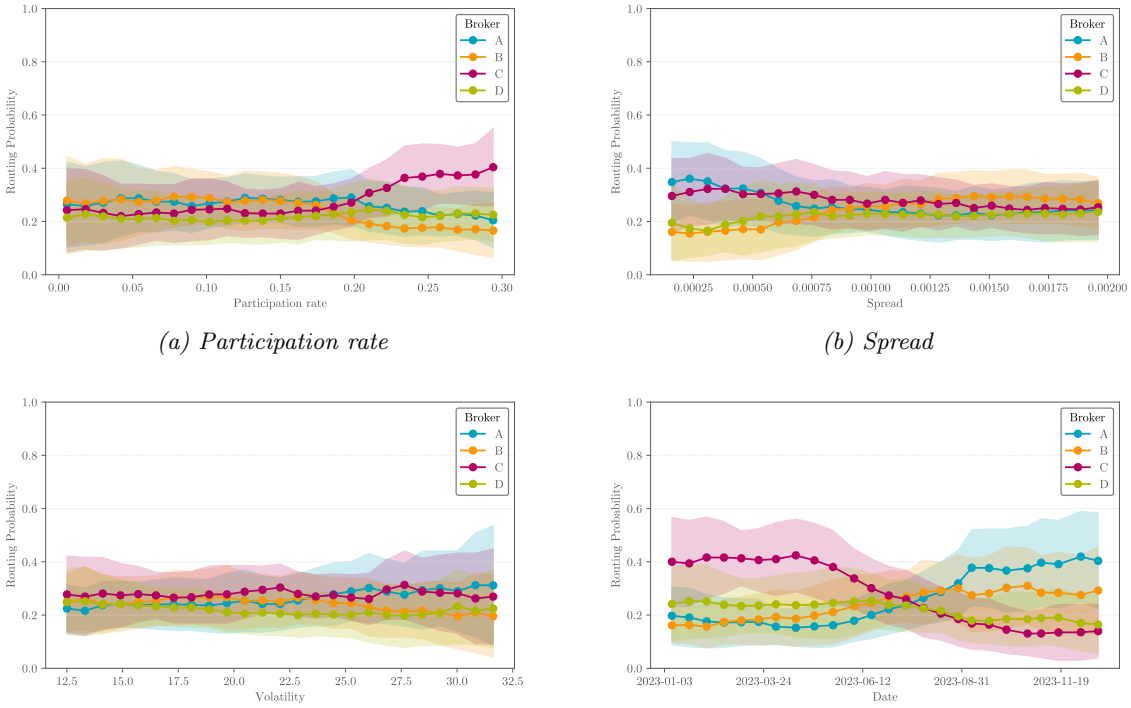


Figure 13: Feature importance scores for the TSRF on the Robeco dataset for region R1. The y-axis represents the importance score, while the x-axis lists the features. Hybrid features are denoted with a Z prefix, whereas the remaining features correspond to brokers A, B, C, or D.

Figure 14 demonstrates the average routing probabilities for region $R1$ plotted against the context features. In Appendix D.2, we provide an in-depth analysis of these routing probabilities by comparing them to cost distributions of brokers for specific ranges of context features. Figure 14 suggests dependencies between context features and broker performance, as reflected in the routing probabilities. Notably, we observe that the routing probability for broker C increases with an increasing participation rate. Additionally, the routing probabilities for brokers A and C decrease with an increasing spread. Furthermore, broker C becomes less favored in the second half of 2023, as evidenced by a significant decrease in its routing probabilities over the year. All these findings align with the statistics obtained from the Robeco $R1$ dataset, as described in Appendix D.2. Moreover, the routing probabilities appear less sensitive to volatility, as they remain stable across the entire volatility range. This observation is consistent with the feature importance scores, where volatility-related features have lower average importance compared to other features, as illustrated in Figure 13.

The findings based on the Robeco $R1$ dataset show that using a CB like the $TSRF$ can significantly reduce trading costs compared to using a *Thompsonwheel*. Our simulation observed a relative decrease of 18% in trading costs. Additionally, the CB provides valuable insights into which features are important for each broker in this region, and further reveals which brokers



(a) Participation rate

(b) Spread

(c) Volatility

(d) Date

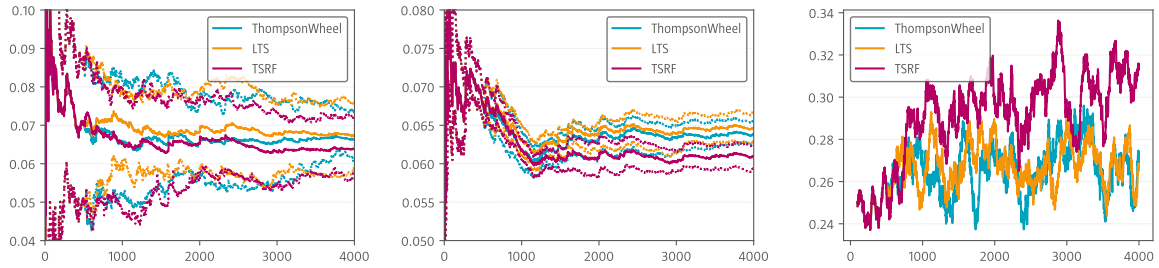
Figure 14: Routing probabilities of the TSRF algorithm in region R1, based on the Robeco trading data from 2023, showing the average probabilities across 30 models with shaded areas for standard deviation bounds, plotted against four context features.

outperform others in specific context regions. These findings align with the in-depth analysis described in Appendix D.2, which also demonstrates that the $TSRF$ effectively handles leptokurtic distributions.

9.2.3 Region R2 Performance Analysis

In this section, we present the results from simulations conducted on the Robeco $R2$ dataset, incorporating context features such as participation rate, spread, volatility, and date. Figure 15 illustrates both the (pseudo-)regret per order and the fraction of orders routed to the best broker over time. The results indicate that only the $TSRF$ outperforms the $ThompsonWheel$. At the end of the simulation, the $TSRF$ achieves an average cost reduction of nearly 4%, as measured by the regret metric. Furthermore, the observed decline in the regret per order suggests that the cost reduction could potentially be even greater. Additionally, Figure 12c reveals that the $TSRF$ consistently outperforms both the LTS and the $ThompsonWheel$ in terms of the fraction of orders routed to the best broker. The underperformance of the LTS compared to the $ThompsonWheel$ suggests that it is not effective in handling the leptokurtic cost distributions observed among brokers.

Figure 16 illustrates the feature importances for the $TSRF$ in region $R2$. Similar to region $R1$, the hybrid spread feature has the highest importance score. However, in this region, the spread feature scores relatively higher compared to other columns, contrasting with the feature importances for region $R1$ shown in Figure 13. The hybrid column representing the participation



(a) Regret (per order) *(b) Pseudo-regret (per order)* *(c) Fraction routed to best broker*

Figure 15: (Pseudo-)Regret per order over time, as shown by the x-axis (time) and y-axis (regret), and the fraction of routing to the best broker over time for the Bandits, based on the Robeco dataset from region R2.

rate is the second most important feature, but the other features score lower, indicating that in this region, spread and participation rates are relatively more important. Consequently, broker costs depend less on other context features.

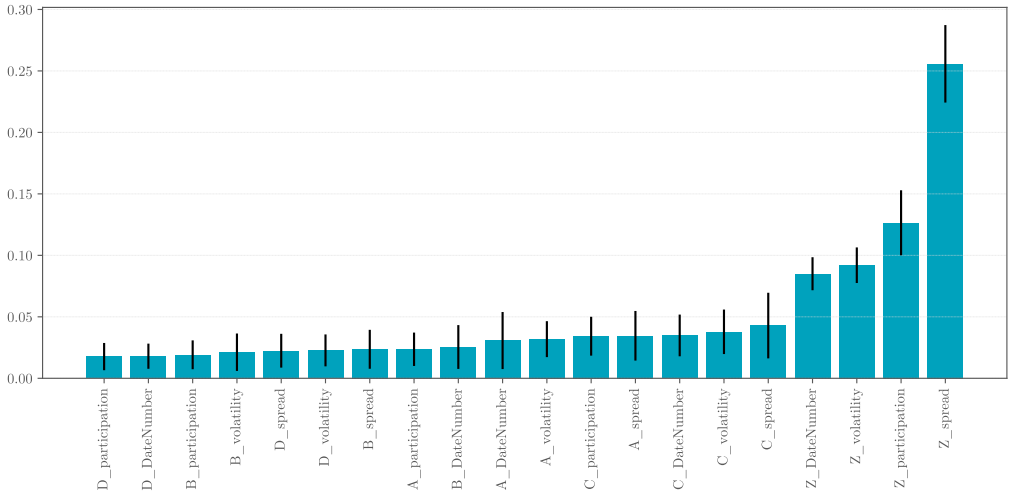
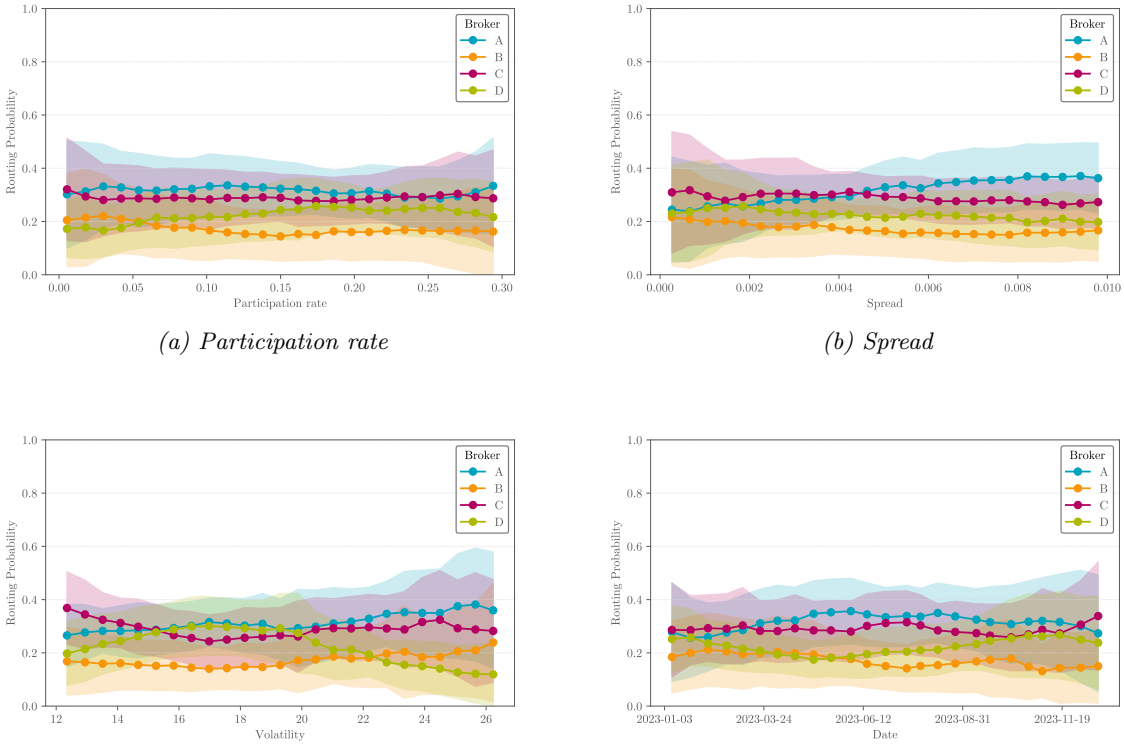


Figure 16: Feature importance for TSRF on the Robeco dataset for region R2. The y-axis represents the importance score, and the x-axis lists the features.

This finding in feature importance is also reflected in the routing probabilities for the four context features, as depicted in Figure 17. In this figure, we observe notable slopes in the routing probabilities for the brokers, primarily in relation to spread, but also for participation rate. However, the routing probabilities over the year appear more stable, and there is less variation observed in routing probabilities given the volatility, except for the extremes of low and high volatility where historical observations are sparse. It appears that broker C outperforms other brokers for low spreads and broker A excels for high spreads, based on the routing probabilities depicted in Figure 17b. Additionally, broker D seems to perform slightly better for higher participation rates. These routing probabilities are validated with the Robeco $R2$ dataset in Appendix D.3, where we conclude that, although the routing probabilities are less interpretable compared to region $R1$, they are still consistent with the underlying data.

We can conclude that the $TSRF$ outperforms its benchmark, $ThompsonWheel$, in region $R2$,



(a) Participation rate

(b) Spread

(c) Volatility

(d) Date

Figure 17: Routing probabilities of the TSRF algorithm in region R2, based on the Robeco trading data from 2023, showing the average probabilities across 30 models with shaded areas for standard deviation bounds, plotted against four context features.

achieving an approximate cost reduction of 4%. Although identifying dependencies between context features and routing probabilities in this region is more challenging, the observed dependencies align with both the underlying data and the feature importance scores.

9.2.4 Recommendations for Robeco

Based on our findings, we offer the following recommendations for Robeco to effectively utilize the methods, results, and insights from our research. Our research demonstrates that CBs, particularly the Thompson Sampling-based Random Forest, significantly outperform traditional Multi-Armed Bandits in scenarios where there are relationships between costs and order contexts. We recommend Robeco to adopt the $TSRF$ for order routing decisions as it provides robust and reliable performance, as well as transparency into the routing decisions it makes. Moreover, understanding the importance of each feature can help in refining the models and removing redundant features, thus improving efficiency and interpretability.

In our analysis, we examine broker performance from both a global perspective and a more localized regional perspective. This dual approach was necessary because each region possesses unique market dynamics and structures. However, since these market dynamics can also vary significantly at a country level, being subsets of the larger regions, we recommend that Robeco explore additional features, such as country-specific data. Our findings demonstrate that the $TSRF$ effectively identifies broker differences both globally and regionally. Given the substantial

variations in market dynamics across regions, leveraging the $TSRF$ for regional analysis remains crucial. Nevertheless, incorporating finer features such as countries within each region could offer deeper insights.

Furthermore, our research underscores that, in general, the spread is the most significant context feature, with participation rate being the second most important. In our study, we sampled order contexts independently for all context features, using uniform distributions to ensure an even evaluation of the bandit algorithms across different feature regions. This method provided a balanced view, but it does not fully capture the real-world distribution of orders, where lower participation rates and spreads are more common compared to higher values. Additionally, there are notable relationships between context features. For example, spreads in region $R1$ are approximately five times lower than those in region $R2$, while participation rates are higher in $R1$ compared to $R2$. Further investigation into context features with dependent sampling could provide valuable insights into the performance of CBs in real-world scenarios.

The current method of estimating costs by bootstrapping from the nearest observations based on context provides a smoothed transition between different volatility levels. However, exploring advanced clustering techniques that can better handle non-linear data might result in more aggressive and precise routing decisions. Robeco could experiment with various clustering algorithms to enhance the performance of their CBs further.

By implementing these recommendations, Robeco can leverage the full potential of the CBs to optimize their order routing process, reduce costs, and improve overall trading efficiency.

10 Conclusion

In this research, we explore the use of Contextual Bandits (CBs) to enhance order routing within financial trading. In trading, buy-side traders typically use algo wheels to allocate smaller orders among multiple electronic trading algorithms offered by execution brokers. Our study focuses on developing and evaluating CB algorithms, with a particular emphasis on Thompson Sampling (TS)-based methods. We introduce a non-linear CB utilizing Random Forest Regressors, termed $TSRF$, and compare its performance against a linear CB, LTS, which employs Ordinary Least Squares. Both CB algorithms are designed to balance exploration and exploitation through TS-based approaches and are assessed in comparison to a traditional TS-based Multi-Armed Bandit (MAB), referred to as $ThompsonWheel$.

We perform a thorough evaluation using two distinct datasets. The first is a synthetic dataset, specifically designed for hyperparameter tuning, model selection, and performance validation. It provides a controlled dataset where the cost distributions of broker algorithms and their dependencies on specific context features are known. The second dataset is real-world data provided by Robeco, which includes trading costs for various broker algorithms in 2023, spanning regions $R1$ and $R2$. The results from the synthetic dataset demonstrate that CBs notably outperform traditional MABs, with the $TSRF$ showing exceptional robustness and effectiveness compared to the LTS. Furthermore, the analysis of feature importances and routing probabilities, derived from context features, confirms that the $TSRF$ aligns well with the expected outcomes based on the synthetic data, thereby validating its effectiveness and transparency, despite its non-linear approach.

Further validation with the Robeco dataset confirms that the $TSRF$ consistently outperforms both the $ThompsonWheel$ and the LTS by effectively selecting optimal broker algorithms for specific regions. The $TSRF$ excels in managing leptokurtic cost distributions, providing more reliable and transparent decision-making compared to LTS. The analysis of routing probabilities under various market conditions offers actionable insights into broker performance. Region-specific evaluations reveal that the $TSRF$ reduces costs by approximately 18% in region $R1$ and nearly 4% in region $R2$, with features such as spread and participation rate emerging as highly informative. These findings align with historical data, further validating the $TSRF$'s capability to handle complex cost distributions effectively.

However, this study does have limitations. The synthetic dataset, while useful for tuning and validation, may not fully capture all real-world complexities, and assumptions made during the modeling may not always hold in practice. Additionally, the focus on only two datasets might limit the generalizability of the findings. Future research should address these limitations by incorporating a broader range of datasets and exploring additional market conditions.

The practical implications of this research are significant for Robeco and similar firms. By adopting the $TSRF$, firms can potentially enhance their order execution, achieving more precise and cost-effective order routing. This could lead to improved trading outcomes and reduced transaction costs. Beyond finance, the methodologies developed in this study have broader applications. For example, the techniques used for handling complex distributions and balancing exploration

and exploitation can be applied in marketing to optimize ad placements or in operations research to enhance decision-making in logistics and resource allocation.

Future research should focus on establishing analytical regret bounds for non-linear CBs leveraging Random Forests to provide theoretical performance guarantees. Additionally, exploring alternative clustering and bootstrapping metrics for order cost estimation and investigating sampling methods that account for dependencies between context features could further refine the model. Such advancements would enhance both the accuracy and interpretability of CBs, leading to more robust applications in various fields. This research provides a comprehensive framework for implementing and evaluating CBs, highlighting their advantages over traditional methods and offering practical recommendations for effective application.

References

Abbasi-Yadkori, Y., Pál, D. & Szepesvári, C. (2011). Improved algorithms for linear stochastic bandits. *Advances in neural information processing systems*, *24*.

Agrawal, S. & Goyal, N. (2012a). Analysis of thompson sampling for the multi-armed bandit problem. , *23*, 39.1–39.26.

Agrawal, S. & Goyal, N. (2012b). Analysis of thompson sampling for the multi-armed bandit problem. , 39–1.

Arellano-Valle, R. B., Gómez, H. W. & Quintana, F. A. (2005). Statistical inference for a general class of asymmetric distributions. *Journal of Statistical Planning and Inference*, *128*(2), 427–443.

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

Auer, P., Cesa-Bianchi, N., Freund, Y. & Schapire, R. E. (2002). The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, *32*(1), 48–77.

Baker, H. K. & Kiymaz, H. (2013). Market microstructure. *Market Microstructure in Emerging and Developed Markets: Price Discovery, Information Flows, and Transaction Costs*, 1–16.

Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5–32.

Bubeck, S., Cesa-Bianchi, N. et al. (2012). Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, *5*(1), 1–122.

Chipman, H. A., George, E. I. & McCulloch, R. E. (2010). Bart: Bayesian additive regression trees.

Chu, W., Li, L., Reyzin, L. & Schapire, R. (2011). Contextual bandits with linear payoff functions. , 208–214.

Dani, V., Hayes, T. P. & Kakade, S. M. (2008). Stochastic linear optimization under bandit feedback. In *Colt* (Vol. 2, p. 3).

Danyliv, O. (2022). Market impact of small orders. *arXiv preprint arXiv:2201.02983*.

Dimakopoulou, M., Zhou, Z., Athey, S. & Imbens, G. (2017). Estimation considerations in contextual bandits. *arXiv preprint arXiv:1711.07077*.

Domowitz, I., Glen, J. & Madhavan, A. (2001). Global equity trading costs. *ITG White Paper, May*, *8*.

Elmachtoub, A. N., McNellis, R., Oh, S. & Petrik, M. (2017). A practical method for solving contextual bandit problems using decision trees. *arXiv preprint arXiv:1706.04687*.

Féraud, R., Allesiardo, R., Urvoy, T. & Clérot, F. (2016). Random forest for the contextual bandit problem. In *Artificial intelligence and statistics* (pp. 93–101).

Gutiérrez, G., Philippon, T. et al. (2018). *How eu markets became more competitive than us markets: A study of institutional drift* (No. w24700). National Bureau of Economic Research New York,.

Hasbrouck, J. (2004). Empirical market microstructure. *Economic and Statistical Perspectives on the Dynamics of Trade in Securities Markets*.

Hastie, T., Tibshirani, R., Friedman, J. H. & Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction* (Vol. 2). Springer.

Kaelbling, L. P. (1994). Associative methods in reinforcement learning: An emirical study.

Kato, T. (2015). Vwap execution as an optimal strategy. *JSIAM Letters*, *7*, 33-36. doi: 10.14495/jsiaml.7.33

Kissell, R., Glantz, M. & Malamut, R. (2003). Optimal trading strategies: quantitative approaches for managing market impact and trading risk. *(No Title)*.

Kveton, B., Szepesvari, C., Vaswani, S., Wen, Z., Lattimore, T. & Ghavamzadeh, M. (2019). Garbage in, reward out: Bootstrapping exploration in multi-armed bandits. In *International conference on machine learning* (pp. 3601–3610).

Langford, J. & Zhang, T. (2007). The epoch-greedy algorithm for contextual multi-armed bandits. *Advances in neural information processing systems*, *20*(1), 96–1.

Li, L., Chu, W., Langford, J. & Schapire, R. E. (2010). A contextual-bandit approach to personalized news article recommendation. , 661–670.

Madhavan, A. (2002). Vwap strategies. *Trading*, *1*, 32–39.

Nilsson, H., Johansson, R., Åkerblom, N. & Chehreghani, M. H. (2024). Tree ensembles for contextual bandits. *arXiv preprint arXiv:2402.06963*.

Quiroga, M., Garay, P. G., Alonso, J. M., Loyola, J. M. & Martin, O. A. (2022). Bayesian additive regression trees for probabilistic programming. *arXiv preprint arXiv:2206.03619*.

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

Said, E., Ayed, A. B. H., Husson, A. & Abergel, F. (2017). Market impact: A systematic study of limit orders. *Market microstructure and liquidity*, *3*(03n04), 1850008.

Sarkar, J. (1991). One-armed bandit problems with covariates. *The Annals of Statistics*, 1978–2002.

Strehl, A. L., Mesterharm, C., Littman, M. L. & Hirsh, H. (2006). Experience-efficient learning in associative bandit problems. In *Proceedings of the 23rd international conference on machine learning* (pp. 889–896).

ter Braak, L. & van der Schans, M. (2023). Optimal order routing with reinforcement learning. *The Journal of Financial Data Science*, *6*(2), 10-23.

Whaley, R. E. (2000). The investor fear gauge. *Journal of portfolio management*, *26*(3), 12.

Woodroofe, M. (1979). A one-armed bandit problem with a concomitant variable. *Journal of the American Statistical Association*, *74*(368), 799–806.

Xu, L., Gotwalt, C., Hong, Y., King, C. B. & Meeker, W. Q. (2020). Applications of the fractional-random-weight bootstrap. *The American Statistician*, *74*(4), 345–358.

Xu, P., Wen, Z., Zhao, H. & Gu, Q. (2020). Neural contextual bandits with deep representation and shallow exploration. *arXiv preprint arXiv:2012.01780*.

A Additional Linear Contextual Bandits

In Chapter 5 we describe linear CBs using OLS for estimation the relationship between context and rewards in the CB framework. However, it is known that OLS models might be sensitive for overfitting. Therefore, we explore some Regularized Least Squares based methods in a CB framework in this setting. We first describe some RLS estimation methods, like Ridge and Lasso, in detailed in Section A.1. As discussed in Chapter 4, there are multiple strategies to balance exploration and exploitation in MABs, which similarly apply to CBs. Consequently, this section elaborates on how these strategies function within the Linear CB setting. Section A.2 discusses CB approaches for the Regularized Linear methods.

A.1 Regularized Least Squares

Besides employing OLS in our CB framework, which may lead to overfitting on training data, we can also utilize Regularized Least Squares techniques such as Ridge, Lasso, and Elastic Net. These methods are particularly advantageous over OLS as they help prevent overfitting, manage multicollinearity, and enhance prediction accuracy, which is crucial in scenarios involving high-dimensional data or highly correlated predictors. Given that our model features duplicated columns for each arm, which can introduce multicollinearity, regularized techniques are invaluable. In the hybrid context model, where context states are shared among the arms, the Lasso method could be particularly insightful. It not only simplifies the model by performing variable selection, removing irrelevant predictors but also highlights differences and commonalities among the arms by emphasizing the hybrid features and limiting the overall sum of the coefficients. This approach allows Lasso to provide a clear insight into how various arms might differ or overlap in influencing the outcomes, facilitating a more nuanced understanding of arm-specific effects within the shared context.

A critical component common to Ridge, Lasso, and Elastic Net regularization techniques is the hyper-parameter λ, which represents the extent of regularization applied to the model coefficients. This parameter λ balances the trade-off between fitting the model accurately to the training data and maintaining a level of generalization that is robust to new data. Specifically, Ridge regression utilizes λ to shrink all coefficients towards zero uniformly, thereby reducing model complexity and managing multicollinearity effectively. This is particularly beneficial in scenarios with highly correlated predictors, as it helps stabilize the model's predictions. Conversely, Lasso goes a step further by not only shrinking coefficients but also setting some of them to zero. This ability to perform variable selection simplifies the model significantly, removing non-informative predictors and resulting in a model that is easier to interpret and manage, especially beneficial when dealing with high-dimensional datasets where feature selection becomes crucial.

Ridge regression utilizes the L2 norm to impose a penalty on the coefficients of the model, which helps control overfitting by penalizing the sum of the squares of the coefficients, effectively shrinking them towards zero. This approach is implemented through the estimation formula for the coefficient vector θ, given by $\hat{\boldsymbol{\theta}} = \left(\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I}\right)^{-1} \mathbf{X}^\top \mathbf{r}$. In this formula, $\mathbf{X}$ represents the context matrix containing the predictors, $\mathbf{r}$ is the vector of obtained rewards that are the arm

related costs to be predicted, λ is the regularization parameter that determines the degree of shrinkage of the coefficients, and $\mathbf{I}$ is the identity matrix scaled by λ. The larger the value of λ, the greater the shrinkage applied to the coefficients.

Lasso regression employs the L1 norm for regularization, providing a robust method for estimating the coefficient vector $\boldsymbol{\theta}$. The formulation of Lasso aims to minimize the objective function given by:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \left(\frac{1}{2n} \|\mathbf{r} - \mathbf{X}\boldsymbol{\theta}\|^2 + \lambda \|\boldsymbol{\theta}\|_1 \right),$$

The parameter λ controls the strength of the L1 penalty, which encourages sparsity in the coefficients. Sparsity implies that some coefficients can be driven to zero, effectively performing variable selection as part of the regression process.

Elastic Net is a hybrid regularization technique that incorporates the benefits of both Ridge and Lasso regularization methods. This approach is particularly useful for models where both the reduction of feature dimensionality and the stabilization of the model's prediction are crucial. The unique characteristic of Elastic Net is its ability to balance the feature selection capabilities of Lasso and the regularization strength of Ridge through a hyper-parameter α, which should be in the interval $[0, 1]$, such that an alpha equal to 0 gives the Ridge. The parameter α helps dictate the mix between the Ridge and Lasso penalties, providing a flexible control over the balance between coefficient shrinkage and feature selection. The mathematical formulation of Elastic Net is given by:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \left(\frac{1}{2n} \|\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\|^2 + \lambda \left((1-\alpha) \|\boldsymbol{\theta}\|_2^2 + \alpha \|\boldsymbol{\theta}\|_1 \right) \right),$$

where α is the balance parameter between Ridge ($\|\boldsymbol{\theta}\|_2^2$) and Lasso ($\|\boldsymbol{\theta}\|_1$) penalties. This formulation allows Elastic Net to enjoy the robustness against multicollinearity provided by Ridge while also benefiting from the variable selection properties of Lasso, making it exceptionally suited for dealing with practical challenges in predictive modeling where features are numerous and highly correlated.

Elastic Net is a hybrid regularization technique that incorporates the benefits of both Ridge and Lasso regularization methods. This approach is particularly useful for models where both the reduction of feature dimensionality and the stabilization of the model's prediction are crucial. The unique characteristic of Elastic Net is its ability to balance the feature selection capabilities of Lasso and the regularization strength of Ridge through a hyper-parameter α, which should be in the interval $[0, 1]$. Specifically, an α equal to 0 corresponds to Ridge regression, while an α equal to 1 would signify Lasso regression, providing a flexible control over the balance between coefficient shrinkage and feature selection. The mathematical formulation of Elastic Net is given by:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \left(\frac{1}{2n} \|\mathbf{r} - \mathbf{X}\boldsymbol{\theta}\|^2 + \lambda \left((1-\alpha) \|\boldsymbol{\theta}\|_2^2 + \alpha \|\boldsymbol{\theta}\|_1 \right) \right),$$

where α is the balance parameter between Ridge ($\|\boldsymbol{\theta}\|_2^2$) and Lasso ($\|\boldsymbol{\theta}\|_1$) penalties.

A.2 Regularized Linear Contextual Bandit Variants

To balance exploration and exploitation in our model, we utilize the same approaches as those described for OLS in Section 5.3. However, unlike OLS and Ridge regression, which provide closed-form solutions for estimating the parameter vector $\hat{\theta}$, Lasso and Elastic Net do not yield an exact calculation for the variance-covariance matrix of the parameters. This variance-covariance matrix is essential for quantifying uncertainty in both the UCB and TS methods. Given the lack of a direct solution in these instances, we must employ approximation methods to assess the uncertainty in predictions and parameter estimates. These approximations are critical for effectively implementing UCB and TS approaches, as they rely on a robust understanding of uncertainty to balance the dual needs of exploring new options and exploiting known rewards.

In the estimation of the variance-covariance matrix for our contextual bandit model, we employ a method as proposed by L. Xu et al. (2020), using the Dirichlet distribution. This approach is recognized for its efficiency over traditional sampling methods, which often require multiple draws from a dataset. Such repeated sampling not only consumes significant time but also may not be feasible given the typically limited number of observations available in contextual bandit scenarios. Drawing repeatedly from a small dataset tends to produce unreliable uncertainty estimates, which can be disproportionately influenced by outliers. Instead, we opt for a more efficient approach by drawing n observations just once from the entire sample, giving us the sample context matrix $\mathbf{X}'$ corresponding to the observed costs $\mathbf{r}'$. We then draw k series of weights from the Dirichlet distribution and use these weights to estimate the parameter coefficients with the weighted observations corresponding to each series. This method yields k vectors of parameter estimates, from which we can accurately derive both the mean and the variance-covariance matrix.

The Dirichlet distribution is particularly suited for this task because it ensures that the sum of the sampled weights equals one, allowing for easy rescaling if only a subset of weights is used. The probability density function of a Dirichlet distribution of order n with parameters $\alpha_1, \ldots, \alpha_n$ is given by:

$$f(w_1, \ldots, w_n; \alpha_1, \ldots, \alpha_n) = \frac{1}{B(\alpha_1, \ldots, \alpha_n)} \prod_{i=1}^{n} w_i^{\alpha_i - 1},$$

subject to the constraints $\sum_{i=1}^{n} w_i = 1$ and $w_i \geq 0$ for all i, where $B(\alpha_1, \ldots, \alpha_n)$ is the normalizing constant. In our application, we use a special case of the Dirichlet distribution where all parameters α_i are equal to 1 for $i = 1, \ldots, n$, simplifying the distribution to the uniform distribution over the simplex defined by these constraints.

In our estimation process, we first draw k series of weights, each series having n weights, collectively denoted as $\mathbf{W} = \{\mathbf{w_1}, \ldots, \mathbf{w_k}\}$. For each weight series $\mathbf{w_i}$ in $\mathbf{W}$, we rescale the context matrix $\mathbf{X}$ and the response vector $\mathbf{r}$, denoted $\mathbf{X}'$ and $\mathbf{r}'$ respectively, according to the weights. These rescaled matrices and vectors are then used to perform regularized regression, applying pre-specified values of α and λ, to derive the coefficient estimates θ_i for each series. Our estimated

parameter vector $\hat{\theta}$ is the average of these θ_i values, calculated as:

$$\hat{\theta} = \frac{1}{k} \sum_{i=1}^{k} \theta_{\mathbf{i}}.$$

To quantify the uncertainty associated with these estimates, we compute the variance-covariance matrix, represented as:

$$\mathbf{\Sigma}_{\hat{\theta}} = \frac{1}{k-1} \sum_{i=1}^{k} (\theta_i - \hat{\theta})(\theta_i - \hat{\theta})^\top.$$

This matrix provides a measure of the spread of the coefficient estimates, indicating the level of variability in the parameters derived from the weighted regression models.

B Robeco Data Insights

In this chapter, we discuss the trading cost dataset provided by Robeco in greater detail. We offer further insights into the observations, including detailed statistics on the costs and analyses based on specific context feature spaces. We begin by filtering and cleaning the data based on these insights. Section B.1 presents a global perspective on the data, providing a broad overview.

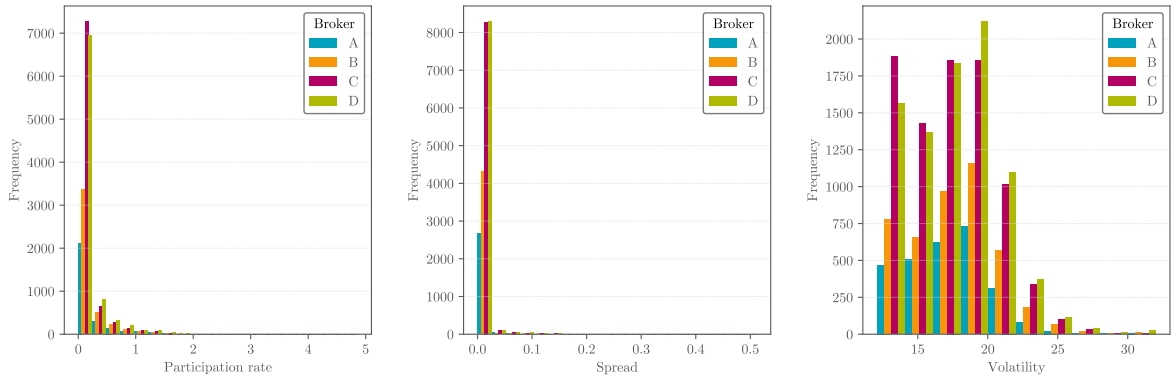
However, a closer examination of the data from regions $R1$ and $R2$ reveals that each region exhibits distinct characteristics. This suggests that applying global filtering and context simplification might not be appropriate for regional-level investigations. Therefore, we analyze and filter the data specific to region $R1$ in Section B.2, and illustrate the data for region $R2$ in Section B.3.

B.1 Insights into the Entire Dataset

In this section, we provide some insights into the entire trading cost dataset gathered by Robeco. Figure 18 shows the frequencies of observations for the context features participation rate, spread, and volatility individually. From these figures, we can observe that we have a large number of observations corresponding to low values for participation rates and spreads. This suggests that when we uniformly draw context values for these features across their ranges, only the results corresponding to very low feature values will be valid, as we have a sufficient number of observations for them. Moreover, since we sample feature values for each context feature individually, there is a possibility that we might sample order contexts that do not exist in the data. Consequently, the closest observations may not provide meaningful information about the costs for such sampled contexts. Additionally, the Bandit algorithms will focus on learning how to handle contexts that are rare or unlikely to occur in the real world, which is not desired and results from these algorithms will be useless. Although there are fewer observations available for high values of volatility, it is crucial for our algorithm to learn how to manage higher volatility. In scenarios with limited historical information, this implies greater uncertainty in cost estimations for brokers. As a result, there is a higher likelihood that a Thompson Sampling-based Contextual Bandit will explore these less frequently observed contexts, potentially gaining valuable insights for future use.

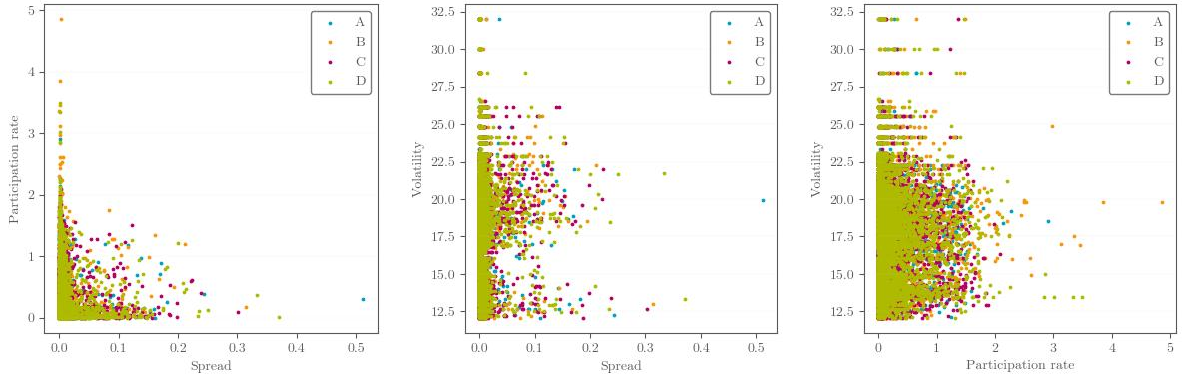
To provide further insights into the occurrence of context features in the dataset, Figure 19 shows scatter plots of pairs of variables plotted against each other. These scatter plots further illustrate why sampling feature values uniformly from a range between the minimum and maximum observed values for each feature individually would be problematic. The patterns observed in these plots suggest that the relationship between features is not uniformly distributed across their ranges, highlighting potential issues with uniform sampling from these wide ranges.

Therefore, we decided to include only the trades in the dataset that correspond to a participation rate of less than 0.3 and a spread lower than 0.01. Figure 20 shows the frequencies of the remaining data. This filtered dataset includes $18,610$ observations, with $4,530$ corresponding to region $R1$ and $14,080$ to region $R2$. Additionally, the scatter plots in Figure 21 support the notion that these filters for participation rate and spread result in a more sufficient context feature space for sampling order contexts. Although there are fewer observations for higher



(a) *(b)* *(c)*

Figure 18: Subplots visualizing the frequency of specific context feature values, including participation rate, spread, and volatility, in the Robeco trading dataset for trades from all regions across all four brokers. The data shown is unfiltered.



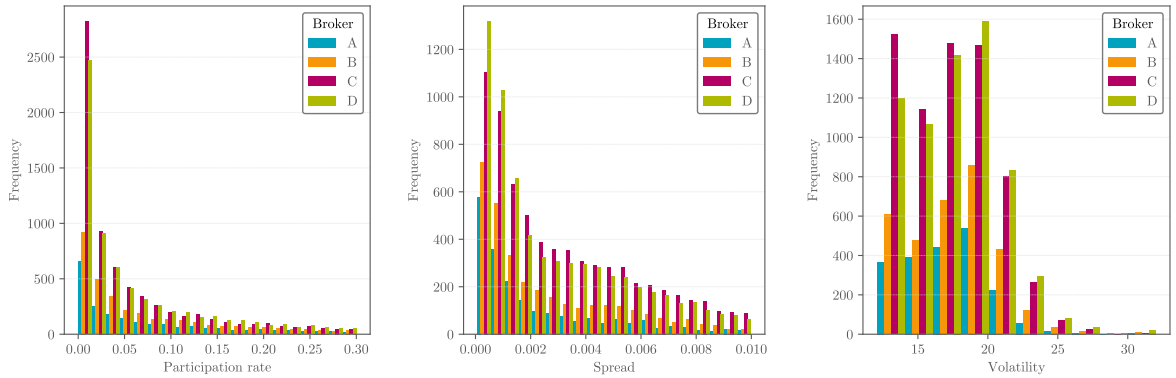
(a) *(b)* *(c)*

Figure 19: Scatter plots representing occurrences of context feature pairs in the Robeco trading dataset for trades from all regions across all four brokers. The data shown is unfiltered.

values of the context features, there are still enough observations in specific context regions to gain valuable insights. Furthermore, if real-life scenarios present even less data, the bandit algorithm will continue to explore as long as there is insufficient certainty.

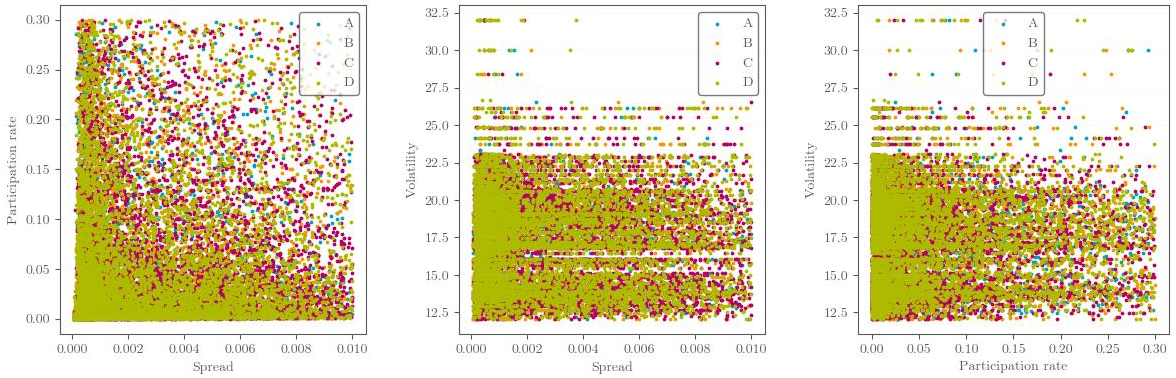
B.2 Regional Analysis: Insights for Region $R1$

A closer examination of the data from region $R1$ reveals distinct characteristics. This subsection focuses on specific insights and details related to this region. In region $R1$, the spreads are on average lower compared to the global perspective, which includes observations from region $R2$. This difference is illustrated in Figure 22b, showing the observed frequencies of orders with spreads lower than 0.002 in region $R1$. In region $R1$, there are only a few observations of orders with spreads higher than 0.0015. Training the model on context regions with such few observations is not ideal. Although the CB algorithm can account for uncertainties in observations, it is better to filter the Robeco $R1$ dataset to include only orders with a participation rate lower than 0.3 and spreads lower than 0.002. Additionally, results are less interpretable when considering orders with spreads higher than 0.002 in this region. These results would be based



(a) *(b)* *(c)*

Figure 20: Subplots visualizing the frequency of specific context feature values, including participation rate, spread, and volatility, in the Robeco trading dataset for trades from all regions across all four brokers. The data shown is filtered to include only trades with a participation rate lower than 0.3 and spreads lower than 0.01.

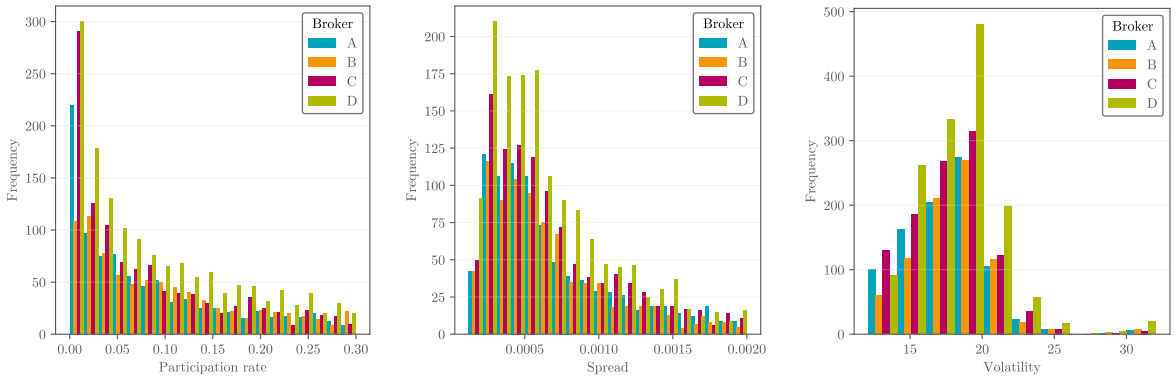


(a) *(b)* *(c)*

Figure 21: Scatter plots representing occurrences of context feature pairs in the Robeco trading dataset for trades from all regions across all four brokers, after applying the filter: participation rate lower than 0.3 and spreads lower than 0.01.

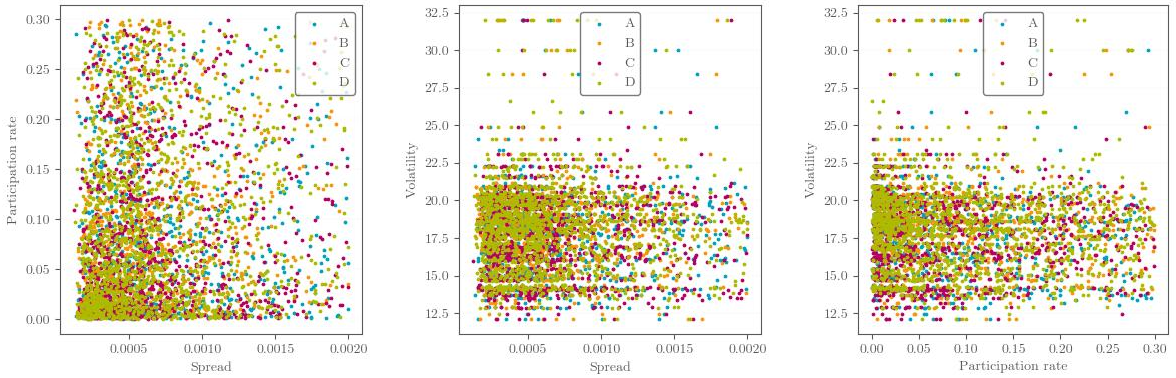
on a small fraction of the data, making it more interesting to see how the CB algorithm handles lower spread values and to determine any differences in low spread values. As mentioned in the previous section, we are interested in how the CB algorithm deals with volatile markets, which occur infrequently. Therefore, we do not filter the dataset based on volatility, but this should be considered when evaluating the CB's performance in high volatility scenarios, as these results are based on only a few observations.

Figure 23, which illustrates the occurrence of observations given a combination of two context features, demonstrates that there are sufficient observations within the specified range of context feature values for participation rate and spread. Additionally, for lower volatility, i.e., volatility below 21, there are adequate observations given the ranges for the context features participation rate and spread.



(a) *(b)* *(c)*

Figure 22: Subplots visualizing the frequency of specific context feature values, including participation rate, spread, and volatility, in the Robeco trading dataset for trades from region R1 across all four brokers. The data shown is filtered to include only trades with a participation rate lower than 0.3 and spreads lower than 0.002.

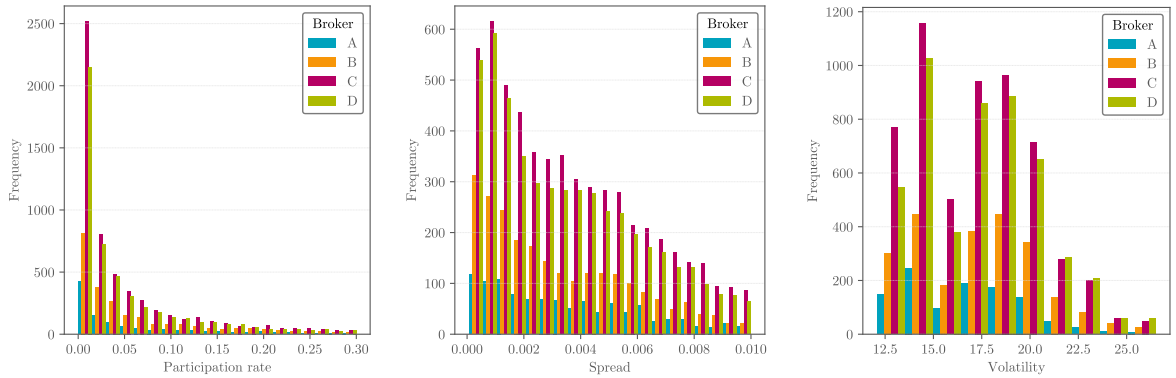


(a) *(b)* *(c)*

Figure 23: Scatter plots representing occurrences of context feature pairs in the Robeco trading dataset for trades from region R1 across all four brokers, after applying the filter: participation rate lower than 0.3 and spreads lower than 0.002.

B.3 Regional Analysis: Insights for Region $R2$

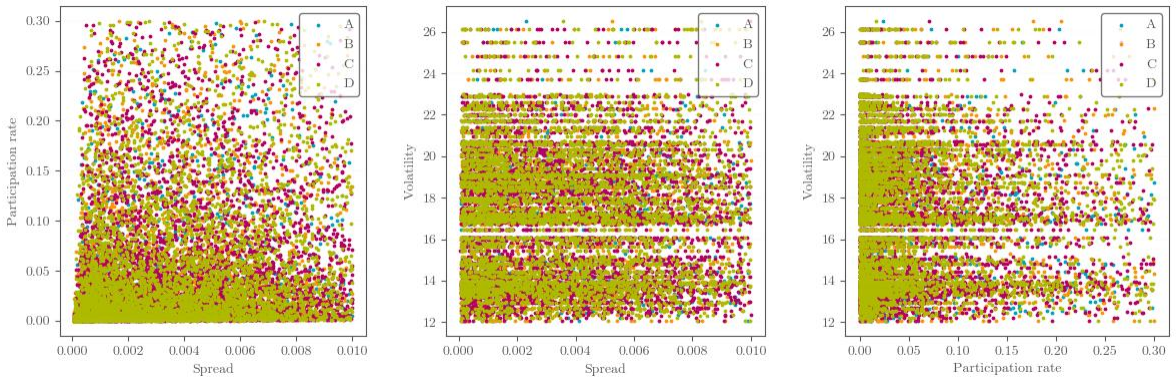
Similarly to Section B.2, we analyze the dataset from region $R2$, highlighting key observations and characteristics unique to this region. Figure 24a shows that the number of observed historical trades for participation rates larger than 0.2 is limited. However, since there are a few hundred orders annually with participation rates higher than 0.2, it is interesting to see how the CBs handle this historical data. From Figure 24b, which depicts the frequency of observations given the spread, we observe that a few hundred orders have corresponding spreads higher than 0.008. This indicates that, unlike region $R1$, the spreads in this region are higher on average, and further filtering of observations is unnecessary. Additionally, for higher volatilities, there are fewer historical observations, as illustrated in Figure 24c, since periods of high volatility, i.e., volatility higher than 22, occur less frequently. In comparison to region $R1$, the volatility values observed in region $R2$ are less scattered. This difference may be attributed to variations in the volatility measure or differences in the underlying indices used for each region.



(a) *(b)* *(c)*

Figure 24: Subplots visualizing the frequency of specific context feature values, including participation rate, spread, and volatility, in the Robeco trading dataset for trades from region R2 across all four brokers. The data shown is filtered to include only trades with a participation rate lower than 0.3 and spreads lower than 0.01.

The sufficiency of observations for each combination of context feature values is further corroborated by Figure 25, which illustrates the frequencies of observations for combinations of two context features. This figure demonstrates that, even when considering multiple context features simultaneously, there are still ample data points to facilitate effective learning. The distribution of observations across various combinations ensures that the Contextual Bandit algorithms can adequately explore and exploit the different scenarios.



(a) *(b)* *(c)*

Figure 25: Scatter plots representing occurrences of context feature pairs in the Robeco trading dataset for trades from region R2 across all four brokers, after applying the filter: participation rate lower than 0.3 and spreads lower than 0.01.

C Parameter Tuning and Model Selection

In this chapter, we describe the tuning of hyperparameters and the selection of specific models to be analyzed in subsequent result sections. We will perform this using the synthetic dataset, which represents various non-trivial characteristics of datasets, such as linear relationships between variables and costs, non-linear relationships, and, importantly, includes some noise features. Since it is impractical to consider and describe all possible combinations of parameter settings, we will balance between optimizing specific parameters and providing rationale for selecting certain parameters. This approach ensures that our parameter selection is not exclusively tailored to the synthetic dataset, but results in robust models with robust settings, potentially applicable to the Robeco dataset. Given that the pseudo-regret is a valid and smoothed approximation of the regret, which converges to the true regret over the long term, we will visualize the pseudo-regret for each comparison of different models with various hyperparameter settings. In addition, we will illustrate the learning progression of the bandits by presenting the fraction of orders routed to the cheapest broker. This fraction is based on the sampled pseudo-regret for a given order context and is calculated over the most recent 100 routed orders up to time t.

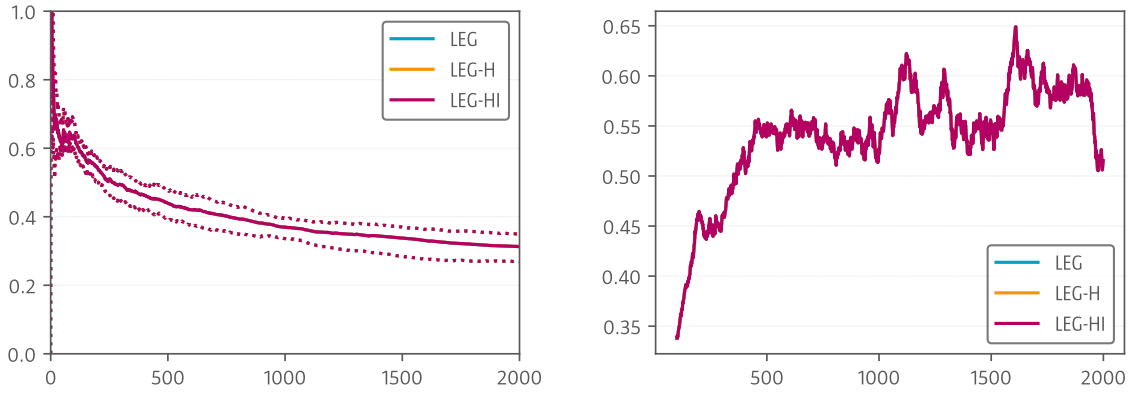
We conduct our tests on 2000 subsequent orders, thus $T = 2000$. This duration provides a sufficient overview of the bandits' evolution over time. Since in real-life scenarios each model requires some experience and observations to learn from, we choose to route the first hundred orders randomly to the brokers, so $n_{warmup} = 100$. To ensure the significance of model tuning, selection, and evaluation, and to mitigate the impact of the initial n_{warmup} randomly routed orders, we test each bandit algorithm with specific settings thirty times, hence $N = 30$, adhering to the Central Limit Theorem. To estimate and approximate the costs for a specific broker in a given context, we use the observed costs from the closest fifty observations in the historical data, as explained in Section 8.2, hence $n_{closest} = 50$. The synthetic dataset is designed to consist of 20,000 trades, which closely matches the total number of trades in the entire Robeco dataset.

In Section C.1, we tune the hyperparameters for several Linear CBs, specifically those using the ϵ-greedy or UCB approaches. We also investigate whether CBs utilizing Regularized Least Squares outperform those based on OLS estimation methods. Following this, in Section C.2, we tune the parameters for the Non-Linear CBs, including some hyperparameters of a RF, which serves as our estimation method in these models. With the hyperparameters of all our CBs fixed, we then perform a selection between these algorithms, as detailed in Section C.3.

C.1 Tuning Linear Bandits

In this section, we tune some hyperparameters of the CB algorithms using a linear context estimation method, as explained in Section 5. We begin by determining whether there are differences in the performances and routing decisions of the algorithms depending on the context matrix provided to the model. We consider three types of context matrices: disjoint, hybrid, and a hybrid matrix where columns corresponding to one specific arm are ignored to avoid multicollinearity, as mathematically described in Section 5.1. Since the linear ϵ-greedy bandit algorithm (LEG) is the simplest, we use it for this analysis. Using an LEG with $\epsilon = 0.05$,

which we will tune later, we evaluate whether a disjoint or hybrid model performs better. The performances of the three models, namely Disjoint (LEG), Hybrid (LEG-H), and Hybrid without columns of one specific arm (LEG-HI), are presented in Figure 26. In this figure, only the result of one model is visible, as all models perform similarly. This indicates that this OLS-based bandit is not sensitive to multicollinearity. Therefore, we decide to continue with the $LEG-H$ model, as it is the most flexible due to its extra columns in the context matrix, making the context matrix more informative while training the model. Henceforth, we will refer to the linear ϵ-greedy CB with a hybrid context matrix as LEG for simplicity in notation. As referred to earlier, we are

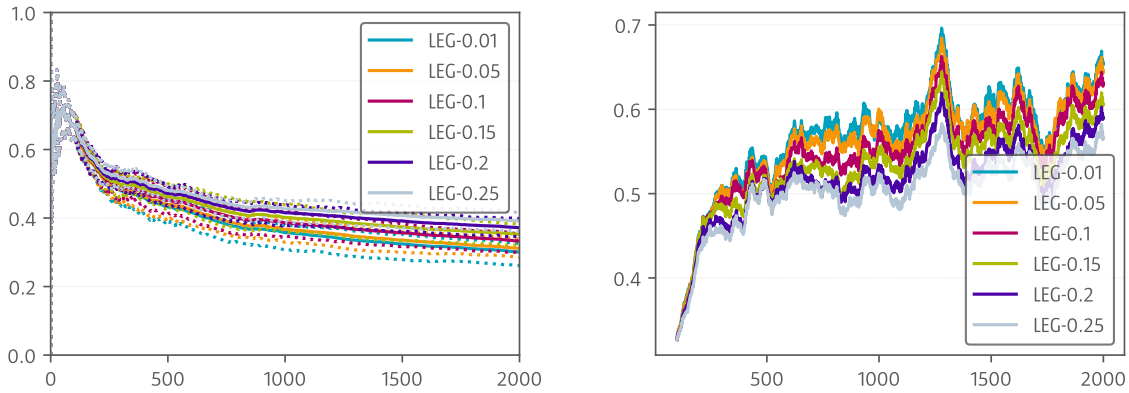


(a) Pseudo-regret (per order) over time

(b) Fraction routed to best broker

Figure 26: Pseudo-regret and fraction routed to best broker for linear ϵ-greedy CBs given different context matrices, namely Disjoint (LEG), Hybrid (LEG-H), Hybrid without columns of one specific arm (LEG-HI).

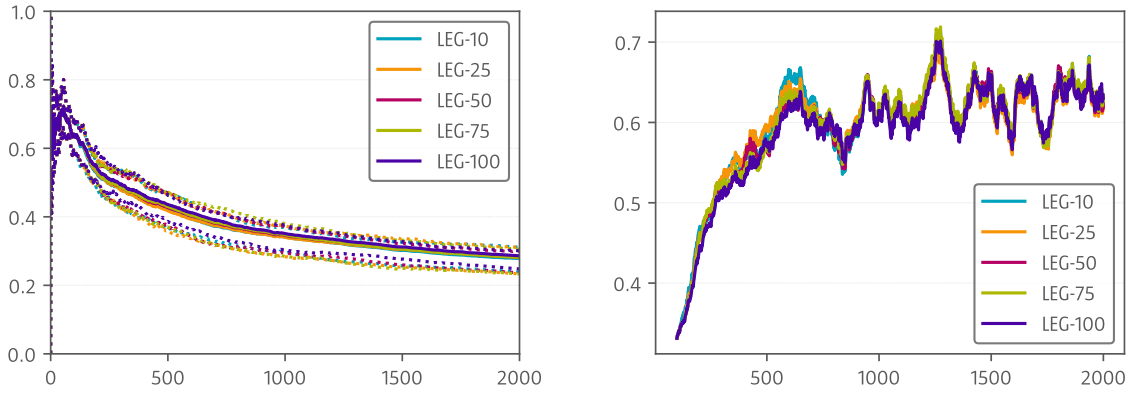
also interested in determining an appropriate value for ϵ in the LEG algorithms. This ϵ should strike a balance between exploring new brokers and exploiting the best-known broker. It is reasonable to expect that a higher ϵ will result in higher regret, as increased exploration comes with a cost. We test the LEG bandit with several values of ϵ: $\{0.01, 0.05, 0.10, 0.15, 0.20, 0.25\}$. The results, shown in Figure 27, clearly illustrate the trade-off between exploration and regret. From this figure, it can also be observed that LEG CBs with ϵ values of 0.01 and 0.05 perform quite similarly. Since we aim to include some level of exploration, we decide to continue this research with all CBs using an ϵ value of 0.05, ensuring a 5% exploration rate. Another parameter of interest is the frequency of retraining the estimation model. While it is possible to retrain the model after every new observation, this approach incurs significant computational costs. Additionally, over time, the impact of a single new observation on the decision-making process diminishes. Moreover, in real-world scenarios, multiple orders might be placed and executed within the same timeframe, making it more practical to retrain the model during off-market hours, such as overnight or during the weekend. To illustrate how this parameter influences model performance, we examine the LEG model with retraining occurring after each n subsequent executed orders, where $n \in \{10, 25, 50, 75, 100\}$. The performance of these different settings is visualized in Figure 28. It can be observed that the retraining frequency has minimal impact on regret. However, pseudo-regret tends to be highest with a retraining frequency of 100 and lowest with frequencies of 10 and 25. Figure 28b supports this, indicating that retraining once per



(a) Pseudo-regret (per order) over time *(b) Fraction routed to best broker*

Figure 27: Pseudo-regret and fraction routed to best broker for different values of ϵ of a linear ϵ-greedy CB, indicated behind the dash.

hundred trades is less precise than more frequent retraining. On the other hand, more frequent retraining, every 10 or 25 iterations, requires more computational time, and the performance differences are not substantial. To balance these considerations, we decide to retrain all models in our simulations once per 50 iterations.

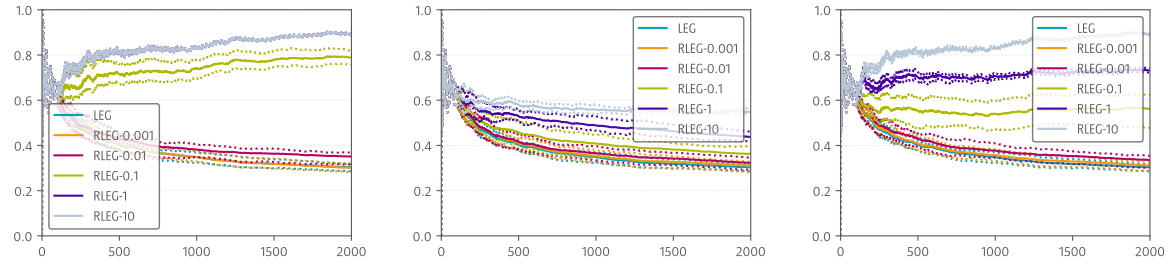


(a) Pseudo-regret (per order) over time *(b) Fraction routed to best broker*

Figure 28: Pseudo-regret and fraction routed to best broker for different values of retrain frequencies of a linear ϵ-greedy CB, indicated behind the dash.

Besides estimating the relationship in a linear way using OLS, we also describe in Section A.2 how Regularized Least Square methods can be utilized in CBs. We examine three regularization methods: Lasso, Ridge, and Elastic Net, with the Elastic Net having α set to 0.5, indicating that both Lasso and Ridge penalty terms equally impact the estimation of parameter coefficients. We compare the OLS and RLS based CBs used in the ϵ-greedy Bandit variant, referring to them as *LEG* and *RLEG* respectively. The results of this comparison are presented in Figure 29, which shows the pseudo-regret per order over time for each of the three RLEG CBs with various values of λ, such that $\lambda \in \{0.001, 0.01, 0.1, 1, 10\}$. From all the subplots corresponding to each regularization method, it can be observed that the lower the value of λ, indicating lower

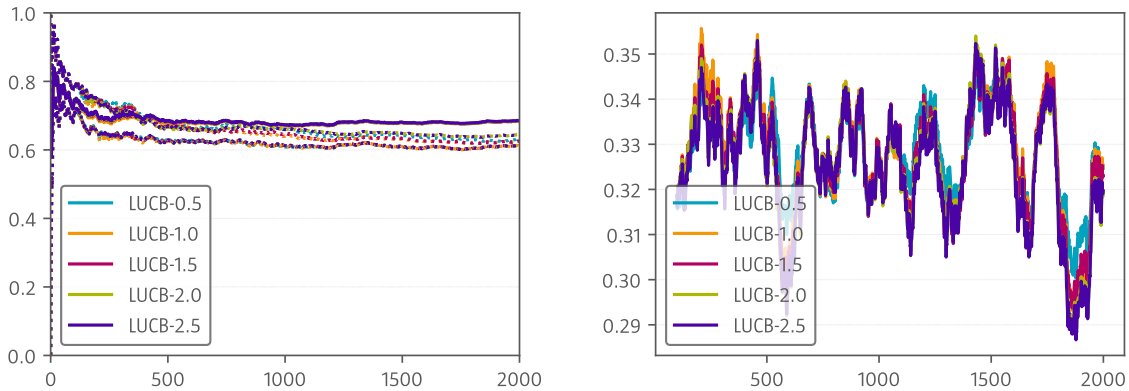
penalization, the lower the pseudo-regret per order. Additionally, the *LEG* model performs the best, suggesting that regularization-based CBs increase the bias too much, leading to suboptimal performance. Therefore, we decide not to further explore these regularized CBs.



(a) Lasso *(b) Ridge* *(c) Elastic-NET ($\alpha = 0.5$)*

Figure 29: Pseudo-regret (per order) over time to compare OLS based LEG to Regularized LEG (RLEG) CBs for several values for the penalization term λ indicated after the dash.

Next to linear CB with an ϵ-greedy approach, we investigate a linear CB with the UCB approach, referred to as *LUCB*, as described in Section 4.2. The *LUCB* algorithm includes one hyperparameter to tune, the scalar C introduced in Section 5.3, which determines the rate of exploration by scaling the upper-confidence bound. We examine several values of C to understand how it influences the performance of our *LUCB* algorithm. Specifically, we analyze the pseudo-regret per order and the fraction of orders routed to the best broker, as visualized in Figure 30. We test C values in the range $\{0.5, ..., 2.5\}$. From Figure 30a, which shows the pseudo-regret for all *LUCB* models, it is difficult to determine a clear winner. However, Figure 30b suggests that a lower C value slightly increases the fraction of orders routed to the cheapest broker, though the differences are not substantial. Therefore, for the remainder of our research, we set C to 1.0. Note that the *LUCB* has significant higher pseudo-regrets and lower fractions of orders routed to the best broker compared to *LEG* algorithms.



(a) Pseudo-regret (per order) over time *(b) Fraction routed to best broker*

Figure 30: Pseudo-regret and fraction routed to best broker for different values of for the hyperparameter C of linear UCB Contextual Bandit, indicated behind the dash.

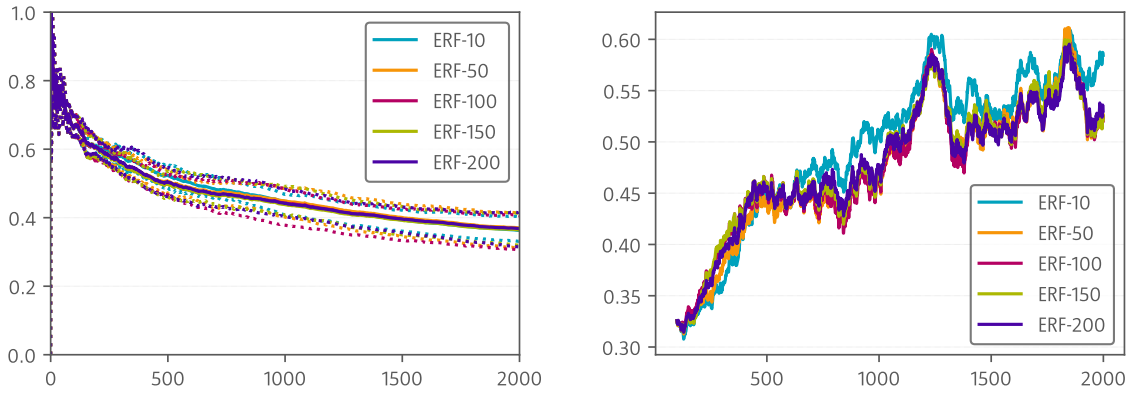
Having tuned all the Linear CBs with hyperparameters, we have settled on an *LEG* with $\epsilon = 0.05$, an *LUCB* with $C = 1.0$, and an *LTS* (Linear CB following the Thompson Sampling approach)

that requires no hyperparameter tuning. In Section C.3, we perform a selection between these CBs to identify the best-performing linear CB for the remainder of our analysis.

C.2 Tuning Non-Linear Bandits

In this section, we tune the hyperparameters of the CBs that use a RF as the estimation method to find the relationship between the context and broker costs. As mentioned in the Literature and Methodology section, RF have proven to be very robust models and are less sensitive to hyperparameter settings compared to models like XGBoost and Regularized Linear models. Therefore, we have decided to examine the impact of three key hyperparameters on the CB performance. These three hyperparameters, as detailed in the Methodology section about RF, are the number of trees, the minimum sample size of each leaf node, and the size of the random subset of features considered for determining the optimal split. In this section, we first examine the impact of the number of trees in the RF, followed by an analysis of the minimum sample size for leaf nodes. Subsequently, we investigate the effect of the size of the subset of m features considered at each split. Thereafter, we visualize how the trees, which are all weak learners, influence the Mean Squared Error. Finally, we investigate which TS based approach works the best for a CB utilizing a RF. These TS based approaches are described in Sections 6.2, 6.3 and 6.4.

We begin by finding a suitable value for the number of trees, denoted as n_{est}, in the RF. This is done using a RF CB following an ϵ-greedy approach, denoted as ERF. We set ϵ equal to 0.05, consistent with the LEG setting discussed in Section C.1, to ensure a sufficient balance between exploration and exploitation. Figure 31 visualizes the pseudo-regrets per order over time and the fraction of orders routed to the best broker for ERFs with $n_{est} \in \{10, 50, 100, 150, 200\}$. These figures indicate that the number of trees does not significantly influence the RF's performance. Generally, a higher number of trees makes the RF more robust, albeit at the cost of increased computational expense. To balance robustness and computational cost, we will proceed with RFs with $n_{est} = 100$ in this research. The second hyperparameter for which we seek a robust

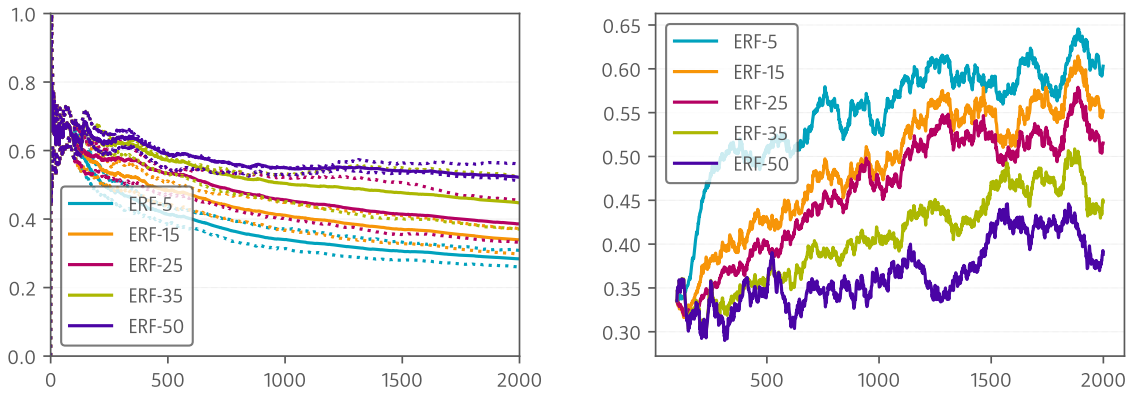


(a) Pseudo-regret (per order) over time

(b) Fraction routed to best broker

Figure 31: Pseudo-regret and fraction routed to best broker for different values of retrain frequencies of a ERF, indicated behind the dash.

value is the minimum size for a leaf sample, denoted as n_{ls}. A lower n_{ls} generally yields more precise predictions from the RF, but it also increases the risk of overfitting. Figure 32 presents the results of our tests for $n_{ls} \in \{5, 15, 25, 35, 50\}$, showing that a lower n_{ls} improves the ERF performance based on pseudo-regret and the fraction of orders routed to the best broker. This would suggest setting n_{ls} to 5. However, to mitigate the risk of overfitting, we decide to set $n_{ls} = 15$, which also appears to be sufficient as the number of observations increases over time.



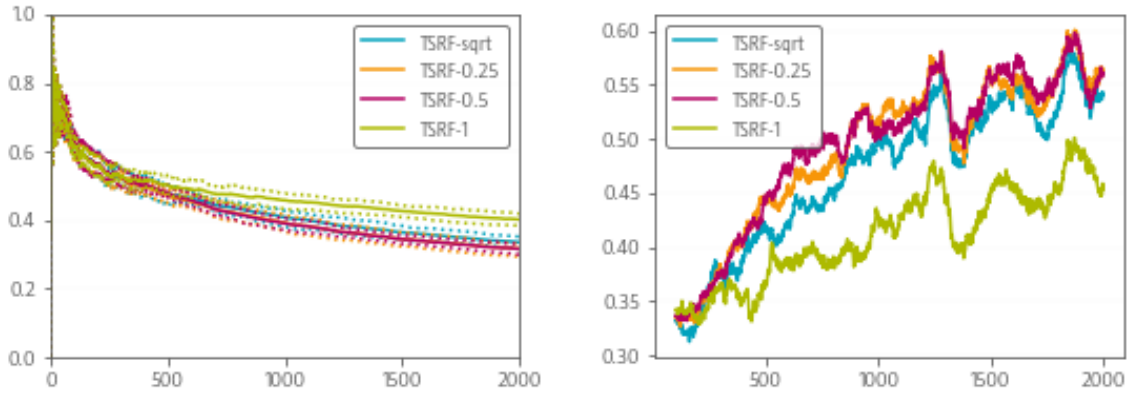
(a) Pseudo-regret (per order) over time

(b) Fraction routed to best broker

Figure 32: Pseudo-regret and fraction routed to best broker for different values for the minimum size of the leaf sample of a ERF, indicated behind the dash.

The final hyperparameter we want to tune for the RF is the number of features to be drawn and considered at each splitting point, denoted as m. To find a robust setting, we examine $m \in \{\sqrt{M}, 0.25M, 0.5M, 1.0M\}$, where M represents the total number of features provided to the RF. Figure 33 visualizes the performances of the different ERF CBs. It is evident that using a large subset, i.e., $m = 1.0M$, results in the worst performance, indicating a loss of robustness due to less independent trees. The ERF CBs with the other three values for m perform similarly. Therefore, we prefer $m = \sqrt{M}$, as this configuration provides the RF with the most independent trees, resulting in a more robust CB.

To further support our choice of m, we also examine the feature importances of the different ERFs, calculated as explained in Section 6.5. The fifteen most important features are visualized for each CB in Figure 34. We observe in Figures 34b and 34c that the importance score of the most important feature is higher compared to the model shown in Figure 34a. This indicates that the score of the most important feature is positively correlated with m, suggesting that a higher m results in less independent and diverse trees being generated. Since we test each CB for N trials, resulting in N different models, we also provide the one standard deviation bounds of the importance scores, indicated by the vertical black bars. Furthermore, it appears that the higher m, the more frequently and quickly the ERF selects noise features, indicated by a number in the feature name. Although, a smaller m increases the probability of drawing noise features, leading to their selection as splitting features, RFs with larger m values do not seem to ignore these variables. Thus, increasing m does not improve the interpretability of the feature importance.



(a) Pseudo-regret (per order) over time

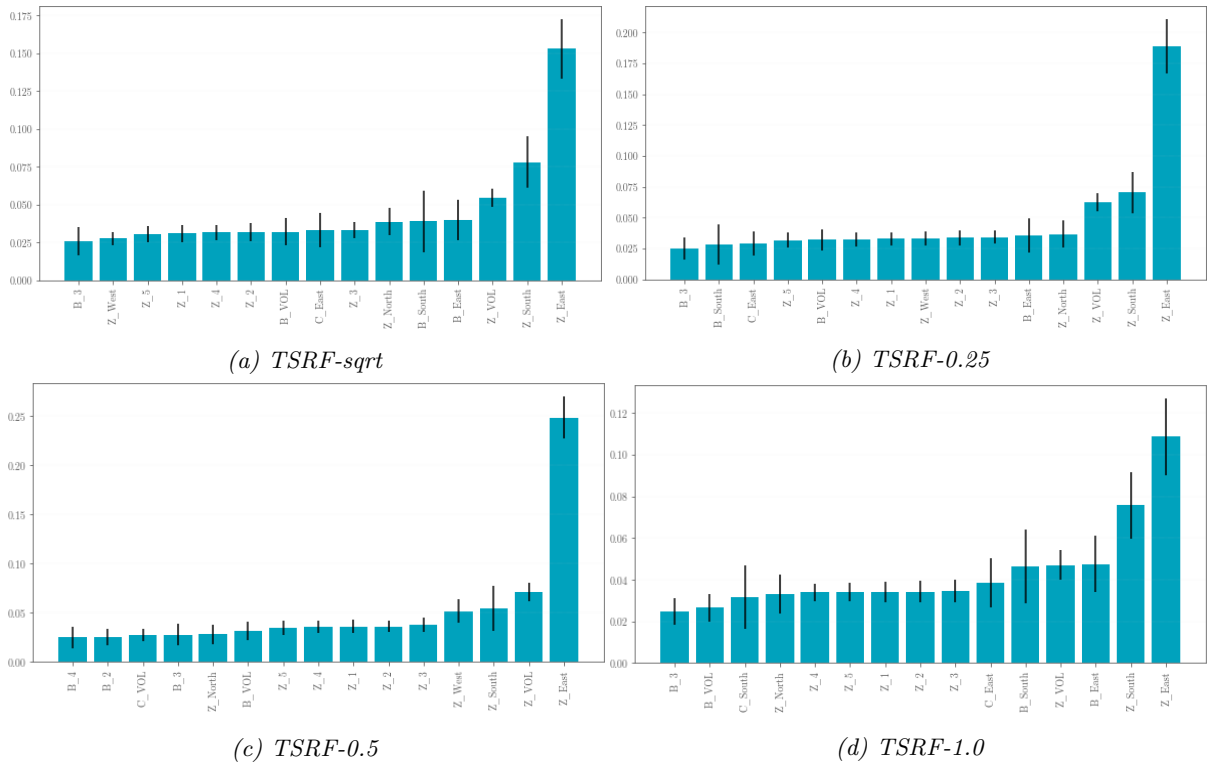
(b) Fraction routed to best broker

Figure 33: Pseudo-regret and fraction routed to best broker for different sizes for the subset of features to consider at each splitting point of a ERF, indicated behind the dash.

To further validate the effectiveness of using a low number of m for weak learners, we follow the approach described in Section 6.6, where we order the trees based on their individual MSE, calculated using the in-sample data on which they were trained. The results of this procedure are shown in Figure 35. By ordering these trees, we can make predictions and calculate the MSE for the k best trees, resulting in MSE_k. We perform this analysis across all N trials, providing us with a mean MSE_k for the k best trees in each of the N RFs, along with the standard deviation bounds. Based on these MSE_k scores, we determine the number of best trees, k, required to achieve optimal in-sample predictions. Figure 35a shows that, on average, only the best 16 trees are needed to achieve the lowest in-sample MSE, which is not significantly different from the MSE obtained using all n_{est} trees.

Since in-sample evaluation alone is not sufficient, we generate an out-of-sample dataset with 2000 additional orders and calculate the out-of-sample MSE_k using the same tree ordering as in the in-sample data. For comparison, we also calculate the MSE_k based on a random order of the trees. Figure 35b indicates that, on average, the best 71 trees are needed to achieve optimal out-of-sample results when using the in-sample tree order, while 94 trees are needed when using a random order. These findings suggest that using the in-sample MSE of each tree to select a subset for out-of-sample predictions can enhance the RF's performance based on MSE. However, determining the optimal k requires out-of-sample evaluation. Despite the slight improvement in MSE, using all the trees can also be adequate, providing more stable and robust predictions. This analysis demonstrates that removing the $n_{est} - k$ worst trees based on their in-sample MSE is unnecessary for making predictions. Additionally, it shows that even the worst-performing trees, with the highest in-sample MSE, do not significantly degrade the overall model performance.

Now, having determined a sufficient setting of hyperparameters for the RF model, we compare four non-linear CBs following a Thompson Sampling based approach, denoted as $TSRF$. The first approach is based on the individual predictions of the trees, as described in Section 6.2, and is referred to as $TSRF\_TB$. Another approach delves further into the individual trees by using, for each tree, the sample in the leaf node that is used for the prediction, as described in Section



(a) TSRF-sqrt *(b) TSRF-0.25*

(c) TSRF-0.5 *(d) TSRF-1.0*

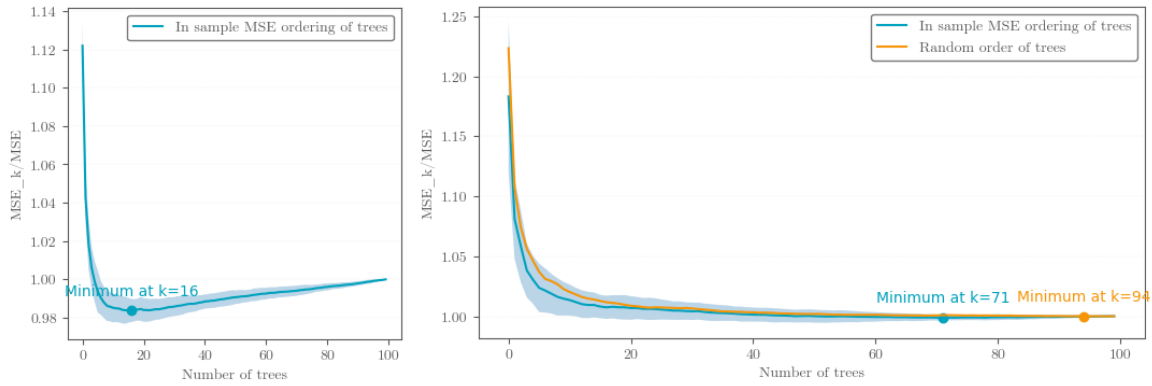
Figure 34: Mean feature importance based on N with standard deviations RFs per CB for different sizes for the subset of features to consider at each splitting point of a ERF, only the fifteen most important features are visualized in each subplot

6.3, denoted as $TSRF\_LB$. The final approach uses the RF to predict the mean costs of a broker and incorporates the uncertainty in the observed costs for each broker, as described in Section 6.4. This can be done by considering all orders routed to one specific broker without looking at the context, resulting in a context-free distribution-based CB, denoted as $TSRF\_DB\_CF$, and its variant that considers the context, denoted as $TSRF\_DB$. The performances of these four models are shown in Figure 36, from which it can be observed that both distribution-based CBs perform the best, namely $TSRF\_DB\_CF$ and $TSRF\_DB$. Although $TSRF\_DB\_CF$ has a slightly lower pseudo-regret per order, we prefer $TSRF\_DB$ since it explores more precisely based on the contexts. Therefore, in this research, we will use only $TSRF\_DB$, to which we will refer from now on as $TSRF$.

C.3 Bandit Selection

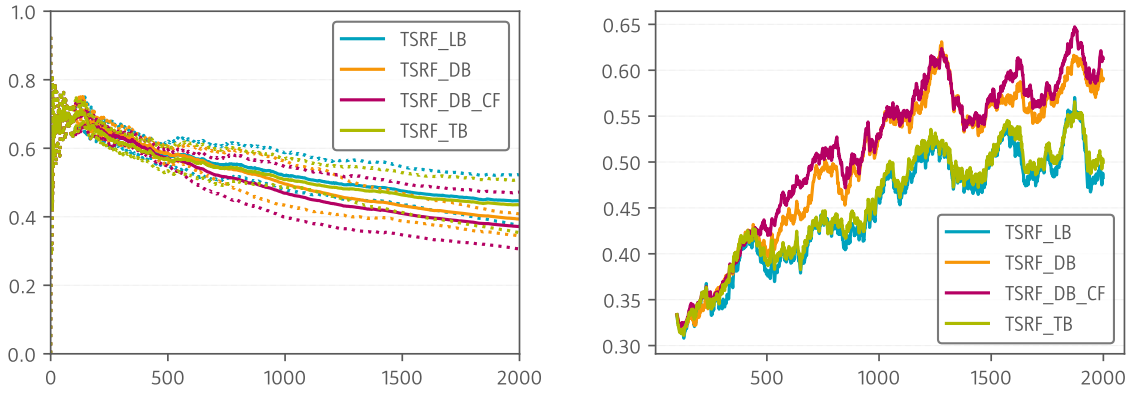
In the previous sections, we tuned the parameters for the linear CBs (Section C.1) and for the non-linear CBs (Section C.2). In this section, we aim to select three bandit algorithms, one MAB, one linear CB, and one non-linear CB, to continue our research.

First, we compare the MAB algorithms with each other, as introduced in Section 4, namely ϵ-greedy (EG), Upper-Confidence Bound (UCB), and Thompson Sampling (TS). The performance of these MABs is shown in Figure 37, where, for comparison, the performance of LEG is also shown. From this figure, it can be observed that the MABs perform almost the same. Additionally, it is evident that on the synthetic dataset, the MABs significantly underperform



(a) In sample MSE evaluation *(b) Out-of-sample MSE evaluation*

Figure 35: Mean in-sample and out-of-sample MSE scores based on the k best trees estimated for all N RF in the ERF. The blue shading represents the standard deviation bounds.



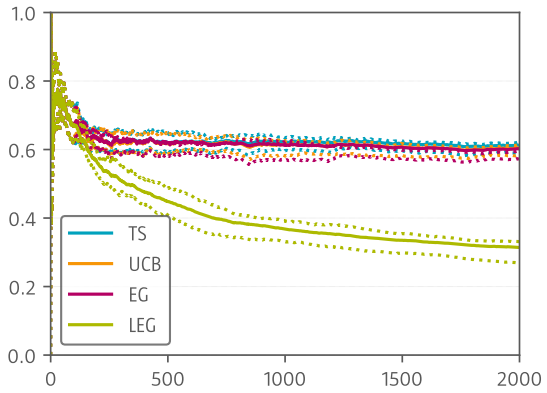
(a) Pseudo-regret (per order) over time *(b) Fraction routed to best broker*

Figure 36: Pseudo-regret and fraction routed to best broker for four different TS based approaches for a TSRF, indicated behind the dash.

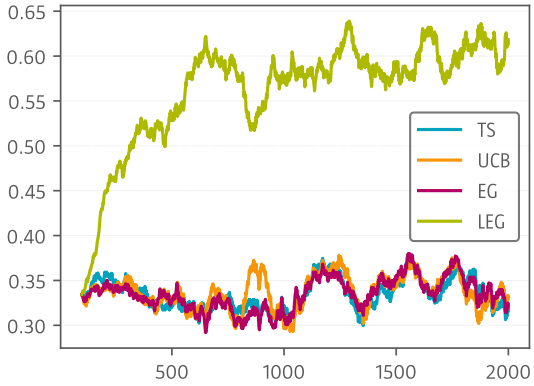
compared to the *LEG*. Since we want one MAB as a benchmark for the CBs, and given that we do not know whether the CBs will also outperform the MABs on the Robeco data, we will use TS for the rest of our research. Furthermore, ter Braak & van der Schans (2023) express their preference for TS in their paper, citing its dynamic exploration and exploitation balance method, which does not require any hyperparameter setting.

Among the three linear CBs (*LEG*, *LUCB*, and *LTS*), Figure 38 illustrates that *LEG* performs the best on the synthetic data, closely followed by *LTS*, which shows competitive performance. While *LEG* demonstrates superiority over these 2000 orders, *LTS* exhibits potential to outperform *LEG* over longer periods. This difference can be attributed to the exploration strategies employed by each algorithm. *LEG* explores at a fixed rate, whereas *LTS* follows a more dynamic exploration procedure, adapting its exploration based on observed outcomes. Based on these observations, we will continue our research with *LTS* as our linear CB.

Finally, our selection process includes choosing one non-linear CB. In Section C.2, we evaluated two specific TS-based approaches: *ERF* and *TSRF*. Figure 39 demonstrates that both *ERF* and

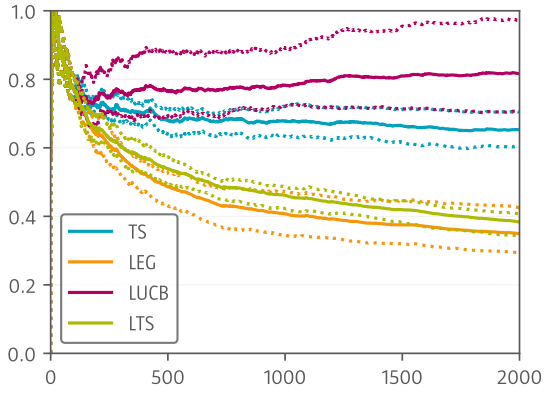


(a) Pseudo-regret (per order) over time

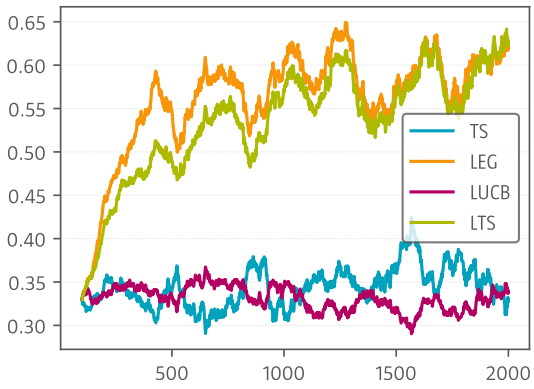


(b) Fraction routed to best broker

Figure 37: Pseudo-regret and fraction routed to best broker for EG, UCB, TS and LEG.



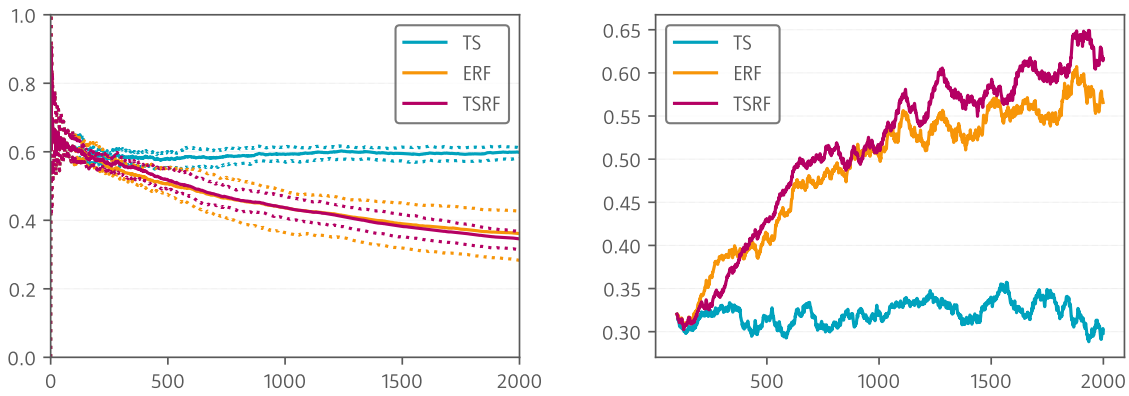
(a) Pseudo-regret (per order) over time



(b) Fraction routed to best broker

Figure 38: Pseudo-regret and fraction routed to best broker for the linear Contextual Bandits.

$TSRF$ exhibit competitive performance initially, but as the number of observations increases, $TSRF$ shows lower pseudo-regret compared to ERF. This mirrors the pattern observed in the selection of linear CBs. Based on these findings, we will proceed with $TSRF$ as our non-linear CB for the remainder of our research. As a result, our research will proceed with three Bandit algorithms, all employing a TS-based approach: TS, LTS, and $TSRF$. Additionally, our analysis of the tuning sections reveals that the steepest learning curve occurs within the initial 500 observations. Given that early observations and routing decisions can significantly impact pseudo-regret over the long term, we have decided to adjust the warmup phase to $T_{warmup} = 500$. This modification allows sufficient exploration for the CBs, particularly the non-linear CBs, following the initial warmup period. Furthermore, to ensure a comprehensive evaluation of the CBs their long-term performance, we extend the total simulation time to $T = 4000$.



(a) Pseudo-regret (per order) over time

(b) Fraction routed to best broker

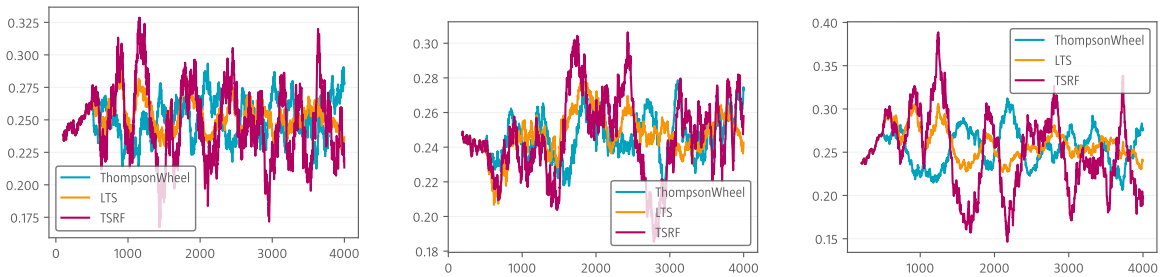
Figure 39: Pseudo-regret and fraction routed to best broker for the non-linear Contextual Bandits.

D Additional Results and Analysis

In this chapter, we present supplementary results and conduct an extended analysis of these findings. Section D.1 includes additional data from the global Robeco dataset, encompassing both region $R1$ and region $R2$. For each region, we provide an in-depth statistical analysis of costs within specific context spaces. The insights are based on critical thresholds derived from routing probabilities discussed in Sections 9.2.2 and 9.2.3, as depicted in Figures 14 and 17. Detailed statistical results for region $R1$ and region $R2$ are provided in Sections D.2 and D.3, respectively.

D.1 Supplementary Global Results

Figure 40 illustrates the fraction of orders routed to the optimal broker based on pseudo-regrets, derived from simulations of the TS-based bandits applied to the global Robeco dataset. This analysis considers only the regional context, specifically whether orders are from region $R1$ or $R2$. However, as detailed in Section 7.2, the brokers their cost distributions are characterized by leptokurtic behavior. This implies that the mean costs, used to calculate the pseudo-regrets, may not provide a comprehensive representation of the underlying cost distributions. Consequently, while the fraction of orders routed to the best broker is informative, it may not fully capture the impact of the leptokurtic distributions on decision-making, potentially limiting the interpretability of the results.



(a) Fraction routed to best (R1 and R2)

(b) Fraction routed to best (R1)

(c) Fraction routed to best (R2)

Figure 40: Fraction of orders routed to the optimal broker, based on pseudo-regrets, obtained by simulating the Thompson Sampling-based Bandits on the global Robeco dataset, considering only the regional context of whether orders belong to region $R1$ or $R2$.

D.2 Extended Analysis for Region $R1$ Results

In this section, we analyze the routing probabilities for region $R1$, as depicted in Figure 14 from Section 9.2.2. We assess whether notable variations in routing probabilities, particularly those that appear to be influenced by context features, can be explained by data from specific context regions. Our analysis indicates that routing probabilities in region $R1$ show limited sensitivity to volatility. However, we observe an increase in the routing probability for broker C as participation rates rise and when spreads decrease. Additionally, given the routing probabilities for specific dates it seems that broker C was the optimal choice during the first half of 2023 according to

the $TSRF$, but other brokers became more competitive in the latter half of the year.

To compare brokers, we utilize bootstrapping techniques to estimate mean costs. This process involves drawing fifty observations from a selected subset of the dataset for each bootstrap sample, and repeating this procedure 1,000,000 times per broker using Dirichlet-distributed weights. The statistics of these bootstrapped mean cost distributions are summarized in tables, including the mean, standard deviation, skewness, kurtosis, and median of the distributions, as well as the number of observations for each period. The median is particularly valuable as it provides a robust measure of central tendency, effectively mitigating the impact of skewness and leptokurtic tendencies in the cost distributions.

We focus on the median of the bootstrapped means for broker comparison because it accounts for the skewness and kurtosis in the cost distributions, offering a more reliable measure than the sample mean alone, which can be unduly influenced by outliers. By using bootstrapped means, we obtain a more accurate reflection of cost distributions and avoid the potential bias introduced by outliers. Additionally, comparing brokers based on the median of the bootstrapped means allows us to assess their long-term performance more effectively.

We begin by examining whether the observed higher routing probability for broker C at higher participation rates is consistent with the data. To do this, we compare the costs per broker for orders in 2023 with participation rates below 0.1 to those with participation rates above 0.2. The results, summarized in Table 3, reveal that broker C has the lowest median costs for high participation rates. Additionally, broker C exhibits the highest standard deviation in costs, which may further contribute to its higher selection probability by the CB algorithm. Conversely, for lower participation rates, the cost differences between brokers become less pronounced, and the standard deviations are larger. This increased overlap in cost distributions among brokers makes them all more competitive and less distinct, which can influence the CB's selection process.

Participation rate	Broker	Mean	St.Dev	Skewness	Kurtosis	Median	Observations
< 0.1	A	0.158	0.429	0.271	0.926	0.143	607
	B	0.126	0.586	0.351	0.860	0.096	491
	C	0.094	0.421	0.303	1.091	0.080	747
	D	0.115	0.415	−0.109	0.600	0.123	916
> 0.2	A	0.156	0.237	0.023	0.228	0.155	104
	B	0.246	0.299	−0.059	0.358	0.248	120
	C	−0.042	0.357	0.188	0.731	−0.046	114
	D	0.018	0.259	−0.609	1.272	0.039	204

Table 3: Overview of cost distributions for brokers for high and low participation rates, based on the Robeco R1 dataset. The table shows statistics obtained by bootstrapping mean costs using Dirichlet-distributed weights, summing scaled samples, and repeating this process 1,000,000 *times per broker. It includes the mean, standard deviation, skewness, kurtosis, and median of the bootstrapped distributions, as well as the number of observations for each period. The median offers a robust measure of central tendency, mitigating the impact of data skewness.*

Conversely, the routing probabilities for broker C, as well as broker A, decrease as the spread increases. Table 4 provides the cost distribution statistics for high and low spreads, indicating

that broker C consistently has the lowest median costs for both low and high spreads. This suggests that broker C is a preferred choice for low spreads. While broker D appears competitive with broker C in terms of median costs, its negative skewness and higher kurtosis may contribute to its lower selection frequency by the CB algorithm. The CB algorithm may favor broker C due to these higher statistical measures of variability. Broker A, despite having higher median costs than brokers C and D, exhibits higher routing probabilities for lower spreads. This is likely due to its slightly higher standard deviation, which increases its competitiveness in situations with lower spreads. Although broker B has the highest median costs for both high and low spreads, it becomes more competitive for higher spreads. This can be attributed to broker B's higher standard deviation and positive skewness, which make it a more attractive option in scenarios with high spreads.

Spread	Broker	Mean	St.Dev	Skewness	Kurtosis	Median	Observations
<0.00075	A	0.142	0.470	0.089	0.823	0.135	595
	B	0.223	0.571	0.271	0.726	0.201	573
	C	0.107	0.437	0.136	0.450	0.100	730
	D	0.091	0.433	−0.194	0.619	0.103	992
>0.00100	A	0.046	0.162	−0.495	2.120	0.052	192
	B	0.130	0.300	1.476	4.973	0.088	138
	C	0.041	0.173	0.002	0.503	0.040	225
	D	0.049	0.162	−0.625	3.016	0.054	281

Table 4: Overview of cost distributions for brokers for high and low spreads, based on the Robeco $R1$ dataset. The table shows statistics obtained by bootstrapping mean costs using Dirichlet-distributed weights, summing scaled samples, and repeating this process 1,000,000 *times per broker. It includes the mean, standard deviation, skewness, kurtosis, and median of the bootstrapped distributions, as well as the number of observations for each period. The median offers a robust measure of central tendency, mitigating the impact of data skewness.*

Table 5 presents the cost characteristics of the brokers for the first and last four months of 2023. This analysis is particularly relevant because routing probabilities suggest that broker C was the optimal choice during the first half of the year according to the TSRF algorithm, whereas other brokers became more competitive in the latter half. In the initial period, broker C ranks as the second cheapest based on both mean and median costs, with broker D being the cheapest. However, broker D exhibits lower skewness and higher kurtosis, which might affect its cost statistics and contribute to broker C being favored. In contrast, during the last four months, broker A is selected the most, which aligns with its lowest median costs for this period. Additionally, broker B sees an increase in selection frequency compared to the earlier months, which may be attributed to its higher standard deviation relative to other brokers.

To conclude, this in-depth analysis of the routing probabilities, reflecting the decisions of the $TSRF$ algorithm, aligns well with the historical data for region $R1$. These findings validate the effectiveness of the $TSRF$ algorithm in adapting to varying market conditions and underscore its robustness in handling diverse cost distributions and volatility. This analysis not only provides insights into the broker selection process but also demonstrates the algorithm its ability to optimize routing strategies based on evolving market dynamics.

Daterange	Broker	Mean	St.Dev	Skewness	Kurtosis	Median	Observations
Before 2023-05	A	0.082	0.411	0.054	0.633	0.085	220
	B	0.131	0.494	0.327	1.053	0.112	322
	C	0.041	0.320	−0.068	0.391	0.044	244
	D	−0.007	0.388	−0.321	0.885	0.010	655
After 2023-08	A	0.088	0.417	0.286	1.169	0.072	387
	B	0.136	0.594	0.188	0.875	0.122	266
	C	0.133	0.446	0.357	0.900	0.111	516
	D	0.158	0.392	−0.057	0.574	0.161	365

Table 5: Overview of cost distributions for brokers across specific time periods, based on the Robeco $R1$ dataset. The table shows statistics obtained by bootstrapping mean costs with Dirichlet-distributed weights, summing scaled samples, and repeating this process 1,000,000 *times per broker. It includes the mean, standard deviation, skewness, kurtosis, and median of the bootstrapped distributions, along with the number of observations for each period. The median provides a robust measure of central tendency, addressing data skewness.*

D.3 Extended Analysis for Region $R2$

In this section, we analyze the routing probabilities for region $R2$, as depicted in Figure 17 from Section 9.2.3, following the same approach as in the previous section, Section D.3. We investigate whether notable variations in routing probabilities, particularly those influenced by context features, can be explained by data from specific context regions.

The figures demonstrating the routing probabilities for region $R2$ show that the brokers performed more consistently over the year. Another observation is that the routing probabilities in this region seem to be more sensitive to volatility, especially for higher values of volatility, compared to the models for region $R1$. However, this is less significant due to the limited number of observations for high volatility.

More interestingly, we observe an increasing routing probability for broker D with higher participation rates. Additionally, there is a noticeable increase in routing probability for broker A with increasing spreads. These trends warrant further investigation to understand the underlying reasons behind the algorithm's decisions.

Table 6 presents the cost characteristics of the brokers for high and low participation rates in region $R2$. Based on the median costs, it is not immediately clear why broker A has the highest average routing probability. However, broker A also exhibits the highest skewness, which may not be adequately reflected in the median and could explain its frequent selection by the algorithm. It is also worth noting that the number of observations for broker A is relatively low. For broker C, the higher routing probabilities are justified as it has the lowest average costs. Broker D becomes more competitive at higher participation rates, which can be attributed to its lower skewness and higher kurtosis for orders with low participation rates.

Figure 17 shows an increasing routing probability for broker A with higher values of spreads. This trend aligns with the data in Table 7, which presents the cost characteristics of brokers for both low and high spreads. Broker A has the lowest median costs for high spreads, justifying its higher routing probability. Conversely, broker C has the highest median costs for high spreads,

Participation rate	Broker	Mean	St.Dev	Skewness	Kurtosis	Median	Observations
<0.1	A	0.155	0.372	0.375	2.974	0.132	842
	B	0.140	0.452	0.041	3.758	0.126	1883
	C	0.091	0.535	0.086	5.331	0.083	4718
	D	0.094	0.367	−0.352	9.488	0.089	4132
>0.2	A	0.067	0.075	0.336	0.939	0.063	78
	B	0.115	0.115	0.674	1.590	0.103	151
	C	0.030	0.124	0.011	2.828	0.027	285
	D	0.051	0.106	−0.022	1.785	0.049	227

Table 6: Overview of cost distributions for brokers for high and low participation rates, based on the Robeco R2 dataset. The table shows statistics obtained by bootstrapping mean costs using Dirichlet-distributed weights, summing scaled samples, and repeating this process 1,000,000 *times per broker. It includes the mean, standard deviation, skewness, kurtosis, and median of the bootstrapped distributions, as well as the number of observations for each period. The median offers a robust measure of central tendency, mitigating the impact of data skewness.*

which contradicts its second-highest routing probability. This discrepancy may be explained by broker C's relatively high skewness, which might not be adequately reflected in the median costs. For orders with low spreads, broker C receives the highest routing probabilities, consistent with its lowest median costs. However, it should be noted that for low spreads, the cost characteristics among brokers are less distinct.

Spread	Broker	Mean	St.Dev	Skewness	Kurtosis	Median	Observations
<0.004	A	0.218	0.443	0.366	2.125	0.192	660
	B	0.198	0.509	−0.010	3.110	0.180	1546
	C	0.097	0.631	−0.245	4.114	0.100	3445
	D	0.139	0.436	−0.197	7.521	0.132	3071
>0.006	A	0.007	0.025	0.119	0.891	0.007	213
	B	0.016	0.039	−0.661	2.373	0.018	389
	C	0.029	0.043	0.582	2.156	0.026	1123
	D	0.013	0.024	−0.147	1.503	0.014	928

Table 7: Overview of cost distributions for brokers for high and low spreads, based on the Robeco R2 dataset. The table shows statistics obtained by bootstrapping mean costs using Dirichlet-distributed weights, summing scaled samples, and repeating this process 1,000,000 *times per broker. It includes the mean, standard deviation, skewness, kurtosis, and median of the bootstrapped distributions, as well as the number of observations for each period. The median offers a robust measure of central tendency, mitigating the impact of data skewness.*

To conclude, the routing probabilities for region $R2$ are more challenging to interpret due to their lesser dependence on specific context features compared to region $R1$. The spread emerges as the most significant feature, which is consistent with its importance score depicted in Figure 16. This lesser dependence between costs and context variables might also explain the relatively smaller decrease in (pseudo-)regret observed for region $R2$.