THE LONDON SCHOOL OF ECONOMICS AND POLITICAL SCIENCE

Department of Statistics 2023/24

Capstone Project

Statistical Methods in A/B Testing

Industry partner: Wise

Submitted for the Master of Science, London School of Economics, University of London

Group E

By

24262

35093

21354

Acknowledgements

We want to express our sincere gratitude to our industry partners from Wise, Egor Kraev, and Alexander Polyakov and our supervisor, Professor Zoltán Szabó from the London School of Economics and Political Science for their invaluable guidance, support, and expertise throughout the research process.

Contents

List of Figures

List of Tables

Executive Summary

This capstone project, conducted in collaboration with Wise, focuses on enhancing the effectiveness and reliability of A/B testing methodologies used within the company. Wise, a fintech leader specializing in international money transfers, relies heavily on data-driven decisions to optimize their product offerings. A/B testing, a fundamental approach in this regard, faces several challenges including Sample Ratio Mismatch (SRM), anomaly detection, and the proper handling of Ratio Overall Evaluation Criteria (OECs). This study addresses these issues by proposing robust statistical methodologies and integrating them into Wise's experimentation platform.

The research problem is divided into three subsections, namely Sample Ratio Mismatch (SRM), Anomaly Detection and Ratio OECs.

SRM occurs when the actual allocation of users to different experimental variants deviates from the expected ratio. This mismatch can lead to biased results, undermining the validity of the experiment. Traditional methods like the Chi-square test identify SRM post-data collection, which can be too late to prevent invalid conclusions. Therefore, continuous monitoring and early detection of SRM using the Sequential Sample Ratio Mismatch (SSRM) approach are crucial to maintaining the integrity of A/B tests.

Anomalies, such as outliers, can skew the results of A/B tests. For example, a single large transaction can disproportionately affect revenue metrics, leading to misleading conclusions. Effective anomaly detection is essential to ensure the accuracy of experimental results and to enable timely interventions that prevent negative user experiences.

Ratio OECs, such as Click-Through Rate (CTR), are ratios of sums that are often correlated within user events (e.g., clicks and views). These correlations violate the assumptions of standard statistical tests like the t-test, complicating the analysis. Developing methods to handle these ratio OECs while preserving their statistical properties is essential for accurate A/B testing.

We developed and integrated the Sequential Sample Ratio Mismatch (SSRM) test, based on the method introduced by Lindon and Malek (2020), into Wise's experimentation platform. The SSRM test continuously monitors visitor allocation and calculates sequential p-values, allowing early detection of SRM. Our simulations and real-world data validations show that the SSRM test provides reliable and timely identification of sample mismatches, significantly improving the quality and accuracy of A/B test results.

For change point detection, we adopted the Cumulative Sum (CUSUM) algorithm, specifically Probabilistic CUSUM due to its sensitivity to small changes and its suitability for sequential data. The CUSUM algorithm accumulates deviations from a target mean and signals significant changes when thresholds are exceeded. This method is effective for detecting shifts in user behavior or financial metrics, ensuring timely responses to maintain app performance and user satisfaction.

To address the challenges of ratio OECs, we employed a linearization approach. This method transforms ratio OECs into linear metrics that preserve directionality and significance levels, allowing the use of efficient statistical tests. The linearized OECs maintain the properties of the original ratios, enabling accurate and reliable analysis without the computational burden of alternative methods like bootstrapping.

The proposed methodologies aim to eventually be integrated into Wise's existing experimentation platform, providing practical tools for continuous monitoring and analysis of A/B tests. The code for the provided solutions can be found in the GitHub repository that we have created. The codes for the SSRM test and CUSUM algorithm enhance the platform's ability to detect issues early, ensuring the validity and reliability of experimental results. These improvements help Wise make more informed decisions, optimizing their product offerings and improving user experience.

For Wise, the implementation of these advanced statistical methods offers several practical benefits. The SSRM test and CUSUM algorithm enable early identification of sample mismatches and anomalies, preventing invalid experiments and ensuring timely interventions. By handling ratio OECs more effectively, Wise can draw more accurate conclusions from A/B tests, leading to better product optimization. The integration of these methods into Wise's platform allows for seamless and efficient analysis of experimental data, supporting the company's rapid decision-making processes.

This capstone project provides and analyses comprehensives solution to some of the critical challenges faced in A/B testing. By enhancing SRM detection, anomaly detection, and the handling of ratio OECs, we have significantly improved the reliability and efficiency of Wise's experimentation platform. These advancements not only support Wise's commitment to data-driven decision-making but also contribute to the broader field of A/B testing methodologies in the fintech industry.

1 Introduction

The rise of the online economy has unlocked unprecedented opportunities for companies to enhance their product offerings. Unlike hardware products, which remain largely unchanged once they leave the factory, software products can be continuously updated and improved through over-the-air updates. However, when implementing potentially costly changes, it is crucial to assess their impact on the user experience accurately.

Ideally, a company would want to be tracking a wide selection of metrics and assess how changes to certain features affect them. This is a question of causality, which is much more difficult to answer than the one of correlation. In the field, economists and scientists exploit randomness and natural experiments to decorrelate confounding variables from the variable of interest and establish that X causes Y. There is, however, one method that allows data scientists to obtain a clear, unbiased estimate of the effect a change has on the variable of interest - it is called A/B testing.

A/B testing, also known as randomized controlled trials (RCTs) in other scientific disciplines, provides a clear and unbiased way to assess the impact of changes on specific variables. Let us consider an example of an online payments company, aiming to increase the number of money transfers between users. One idea the company come up with to achieve this goal is to make the "Send" button bigger, red, and placed above the others. To assess the impact of this change, they could simply implement it for all users and compare the average transfers made before and after the change. However, this approach is flawed due to potential confounding factors, such as seasonal variations in user behavior.

Alternatively, the company could allow users to opt-in to the new button design and compare the transfer activity of those who opted in versus those who did not. In this method, the users who opted in may simply be the ones who would make more transfers anyway and this would cause a bias, rendering the comparison invalid for assessing the true effect of the button design change. The gold standard for such evaluations, which scientists across many fields have used for years is a randomised control trial (RCT). Users are randomly allocated to either a control group (old button) or a treatment group (new button). This randomization aims to ensure that both groups are comparable in terms of demographic and behavioral characteristics, thereby isolating the effect of the new button design on transfer activity.

Is it then the case that by using a well established method such as RCTs, also called A/B testing, and applying it to any metric of interest makes answering any question of causality trivial? The reality is that there are many challenges when running A/B tests that a company needs to consider before it can believe the insights brought by them. While a proper A/B test is almost the best way to establish causality, safeguards need to be built to guarantee validity. Before we move onto the potential issues, we introduce some definitions similar to the ones used by Kohavi et al. (2020)

Overall Evaluation Criterion (OEC) is the metric of interest. It has to be a quantitative

measure of the experiment's objective. In the example above, it is the average number of transfers. It is crucial to select a good OEC that reflects the company's goals. For example, it may be tempting to select revenue as the OEC, but it would most likely lead to decisions such as increasing the number of ads, or increasing the fees - potentially increasing the revenue but decreasing user experience in the process.

While most positive changes are small and improvement is incremental, the results from OECs can be potentially lucrative. Google's famous "41 shades of blue" experiment, for example, translated into a $200 million increase in annual revenue (Hern 2014). Amazon leveraged insights from an OEC to move credit card offers from the homepage to the checkout page, resulting in tens of millions in annual profits (Kohavi and Thomke 2017). Similarly, Bing deployed an A/B test for ad displays that resulted in $100 million of additional annual revenue in the United States alone (Kohavi et al. 2020). These examples highlight the profound financial impact that well-executed A/B tests can have.

Parameter is the variable the company manipulates. In the example above, the parameter can be thought of as a dummy variable equal to one when the user has the new button and zero otherwise. A simple A/B test can be extended to an A/B/n test where the parameter can take more (discrete) values. It can even be extended further, to a Multivariable test, where multiple parameters can interact to find a global optimum (consider changing the size, colour, and position of the button separately, resulting in eight possible combinations).

Randomisation Unit is the unit at which the randomisation is performed. In the example, the randomisation unit was the user. But there are many other potential units, such as a visit, region, or page.

Running online controlled experiments (OCEs), aka A/B tests, at scale has presented a host of challenges which require new statistical methodologies to address them. Here are some of the challenges that occur while running an A/B test.

Sample Ratio Mismatch (SRM) occurs when the actual allocation of users to different variants in an A/B test deviates from the expected allocation ratio. This can lead to biased results and incorrect conclusions. For instance, if an A/B test is supposed to split users evenly between a control group and a treatment group, but instead, the treatment group ends up with significantly fewer users, the results may be misleading. Identifying and correcting SRM is essential for the accuracy of A/B testing.

Anomalies, or outliers, can significantly skew the data and impact the results of A/B tests. For example, in an online transactions company testing a new button, a single large purchase by a user could skew the revenue data, giving an unfair advantage to one variant. These anomalies must be detected and handled to prevent skewed results and ensure reliable outcomes. Effective anomaly detection helps in timely intervention, preventing negative customer experiences and ensuring the accuracy of experimental results. Change points refer to sustained changes in the data rather than abrupt anomalies. Detecting these is crucial, especially in fintech applications of A/B testing. Significant deviations from expected patterns

can indicate important events or issues that need attention.

A Ratio OEC is a metric that represents the ratio of two sums. For example, the Click-Through Rate (CTR) is a common ratio OEC that is calculated by dividing the total number of clicks by the total number of views. The challenge with ratio OECs is that the events for a single user (like their views and clicks) are often correlated. This correlation violates the assumptions required for standard statistical tests like the t-test, making it more difficult to draw accurate conclusions from the data.

Industry partner

Wise, formerly known as TransferWise, is a financial technology company headquartered in London. Wise specializes in international money transfers, offering a cost-effective and transparent alternative to traditional banking services. Utilizing a peer-to-peer system, the company minimizes currency conversion fees and provides real-time exchange rates, thereby reducing the overall cost for consumers and businesses. Wise has expanded its services to include multi-currency accounts, enabling users to hold and manage funds in multiple currencies simultaneously.

While Wise has established itself as a leader in fintech innovation, it has also contributed to the broader tech community through the development of tools like the tw-experimentation package. This package provides tools to set up, run, and analyze A/B tests efficiently. The package includes features for defining experiments, randomizing subjects into control and treatment groups, and calculating statistical metrics to evaluate experiment outcomes. For this goal one can use Jupyter notebooks or Streamlit app with user-friendly interface.

Thus, A/B testing is a fundamental method for data-driven decision-making in product development and marketing. It enables companies to experiment with new features and measure their impact on user behavior. However, ensuring the validity of these experiments is crucial, and there can be many potential problems which can invalidate these experiments. This study focuses on these three topics—Sample Ratio Mismatch, anomaly detection, and Ratio OECs—to improve Wise's experimentation platform for A/B testing.

To learn more about these potential problems and to enhance the platform, this study delves deeper into understanding and addressing these challenges to ensure reliable and accurate results from A/B testing at Wise.

2 Literature Review

This literature review section aims to provide a comprehensive overview of the existing research and methodologies used for SRM, anomaly detection, and Ratio OECs. By examining the various approaches, algorithms, and their applications, we aim to identify the strengths and limitations of each method.

2.1 Sample Ratio Mismatch

A/B testing is a fundamental methodology for evaluating the effectiveness of changes in online environments, allowing businesses to make data-driven decisions. This review synthesizes findings from various studies on Sample Ratio Mismatch (SRM) in A/B testing, highlighting its implications and detection methods. The literature covers fundamental principles to advanced methodologies, emphasizing the critical importance of SRM management to ensure the validity and reliability of experimental results.

Kohavi et al. (2020) provide a comprehensive guide on trustworthy online controlled experiments, focusing on principles and best practices. They highlight SRM, which occurs when the actual ratio of users in control and treatment groups deviates significantly from the expected ratio, leading to biased results. Practical methodologies for detecting SRM, such as statistical tests and real-time monitoring, are emphasized for maintaining experimental integrity.

Fabijan et al. (2019) describe SRM as a symptom of data quality issues in online controlled experiments (OCEs), stressing the importance of detecting and diagnosing SRM to avoid erroneous conclusions. SRM can arise from incorrect user assignment, telemetry logging errors, data processing mistakes, and analysis inaccuracies. Statistical tests like the chi-square test are recommended for detection and Fabijan et al. (2019) stresses that diagnosing the root cause of SRM requires a detailed investigation of each experiment stage.

There are various studies indicating that SRM is a significant issue in A/B testing, potentially leading to incorrect results. One such study by Esteller-Cucala et al. (2019) identifies unbalanced sampling, or SRM, as a common pitfall in A/B testing. They explain that SRM can result in biased outcomes and misinterpretations, as variations in user behavior might not be evenly distributed across test and control groups. To mitigate these risks, the authors recommend continuous validation of the environment through A/A tests, which are tests where identical groups are compared to ensure no differences exist, and repeated checks on the A/B test sample balance. They also suggest implementing robust randomization techniques and tools to minimize the risk of unbalanced sampling.

Chen et al. (2018) discusses the chi-square goodness of fit test as a method for detecting SRM, where the observed distribution is compared with the expected distribution. This test helps identify if there is bias in the triggered sample, which is crucial for maintaining

experiment integrity. The importance of automation of data quality checks like A/A tests and SRM tests to establish trust in the experimentation platform is also highlighted by Larsen et al. (2024). Expanding on the chi-square goodness of fit test as a detection method, an automated system for SRM detection and analysis was developed (Vermeer et al. 2022). The paper emphasizes that SRM checks are relatively straightforward to automate and can be valuable for data quality monitoring, particularly for companies using third-party experimentation platforms that do not provide built-in SRM checks. The studies suggest several potential improvements to their system, including the implementation of the Sequential Sample Ratio Mismatch (SSRM) approach to mitigate alpha inflation. This approach, developed by Lindon and Malek (2020), enhances the accuracy of SRM detection by providing sequential multinomial test for testing a point null based on the counts of each outcome.

The chi-square test is an example of a fixed-n test, which is a test where the sample size is predetermined and provides statistical guarantees when performed once; due to this limitation, it is typically performed after data collection and prior to analysis (Lindon and Malek 2020). Revealing a bug that renders an expensive experiment invalid, only after the experiment is finished is not ideal and to address this issue, Lindon and Malek (2020) introduce a Bayesian and sequential approach for detecting SRM as early as possible. Their method provides continuous monitoring of traffic counts to look for significant imbalances and includes sequential multinomial test accounts for optional stopping and continuation of an experiment, offering sequential guarantees of Type-I error probabilities.

In this paper, we have utilized the method introduced by Lindon and Malek (2020) to create an SRM test class that helps detect SRM in the experimental data. This approach allows for continuous monitoring and early detection of sample ratio mismatches, mitigating the risk of invalid experiments due to imbalances. Through our literature review, we compared this sequential method to the chi-square test and this comparison and the insights from existing studies guided us in adopting the sequential method, ensuring more reliable and timely identification of SRM.

2.2 Anomaly Detection

Anomaly and change point detection has a vast literature available focusing on various pre-existing and use-case-specific algorithms to detect these.

Shipmon et al. (2017) focuses on detecting anomalies in time series data that are highly periodic but noisy. Specifically, it addresses the challenge of identifying significant drops in data streams where labeled anomalies are scarce. This scenario can be thought of as similar to detecting changes in certain metrics during A/B testing, since there is a scarcity of labeled anomalous data and hence applying supervised machine learning models becomes difficult. This motivates us to search for methods that do not require a large amount of labeled data to train.

Liu et al. (2023) details the differences between anomalies and change points, stating change points as an extension and type of anomalies that are crucial to be detected. This paper provided us with a better idea as to which type of detection is more relevant to our use-case. Liu et al. (2023) compares the performance of different change point detection algorithms, including PELT, Bottom-up, and Binary segmentation, using the ruptures library. It highlights that PELT, along with the other methods, utilizes parameters such as "penalty value" and the number of data segments to perform the detection.

Granjon (2014) provides a comprehensive examination of the CUSUM change point detection algorithm, highlighting its mathematical formulation, implementation, and variations. The algorithm is designed to detect shifts in the mean of a monitored process, which is crucial for identifying changes in defined metrics during A/B testing. Granjon's review of the CUSUM algorithm underscores its robustness and adaptability, making it a suitable choice for change point detection in A/B testing for fintech applications.

Killick et al. (2012) provides a comprehensive review and comparison of changepoint detection methods, with a particular focus on the Pruned Exact Linear Time (PELT) algorithm. It compares the cost complexity of approximate methods like Binary Segmentation and exact methods like PELT, highlighting their workings. It defines a cost function that motivates and guides us to formulate our own problem mathematically. It shows us how pruning in exact methods like PELT can increase the computational efficiency. From Killick et al. (2012) we understand that PELT algorithm has a combination of exactness, efficiency, and scalability. It can be inferred that it might make it one of the superior choices for changepoint detection in fintech applications, particularly in A/B testing.

2.3 Ratio OECs

Most of the literature regarding ratio metrics focuses on bootstrapping and the delta method. Bakshy and Eckles (2013) evaluate different bootstrap methods and analyse how neglecting different levels of dependence structures affects inference. Crook et al. (2009) recommend using calculation-heavy bootstrap methods for variance estimation when the experimental unit differs from the randomisation unit. Drutsa et al. (2015)'s contribution is the introduction of novel metrics such as entropy and quantiles, which capture diversity and extreme cases of user engagement. Furthermore, they highlight the benefits of bootstrapping and p-value adjustments. CUPED (Controlled ExPeriments Using Pre-Experiment Data) method is introduced in Kohavi et al. (2014). By using covariates derived from historical data, CUPED can enhance the detection power of experiments, particularly for ratio metrics. Regarding the delta method, Deng et al. (2017) demonstrate that the delta method performs well for user-randomised experiments, providing more reliable variance estimates compared to traditional methods. Deng et al. (2013)'s paper's delta method approach allows for smaller sample sizes (or shorter experiment) durations while preserving the same statistical power, therefore

improving the efficiency.

3 Problem Formulation

In this section, we formulate our problem by addressing key concepts and challenges. We start by defining SRM and its implications on experiment results (Section 3.1). We then delve into anomaly detection, outlining its mathematical formulation and the algorithms commonly used for change point detection (Section 3.2). Finally, we discuss Ratio OECs and why traditional methods may lead to incorrect conclusions (Section 3.3).

3.1 Sample Ratio Mismatch

Sample Ratio Mismatch (SRM) is one of the major pitfalls of interpreting metrics in online controlled experiments (OCE) (Dmitriev et al. 2017). SRM is a data quality check that indicates a significant difference between expected proportions of users among experiment variants (e.g. configured before the experiment started) and the actual proportions of users observed at the end of the experiment (Fabijan et al. 2019). For instance, if a 50/50 split is expected between two experiment variants with total number of users being 1,637,070, the ratio between the number of users exposed to each of the groups at the end of the experiment is expected to be close to 1. The validity of the experiment is seriously compromised if Variant A receives 50.2% of users while Variant B receives 49.8% (821,588 versus 815,482 users). The probability of such a mismatch to occur by chance is less than 1 in 500,000. This shows how even a slight SRM in most cases can completely invalidate experiment results (Kohavi et al. 2020).

To illustrate how SRM can impact experiment outcomes in real-world scenarios, let us consider an experiment that was conducted at Microsoft. (Fabijan et al. 2020). A product team at MSN increased the number of rotating cards on the carousel from 12 to 16.

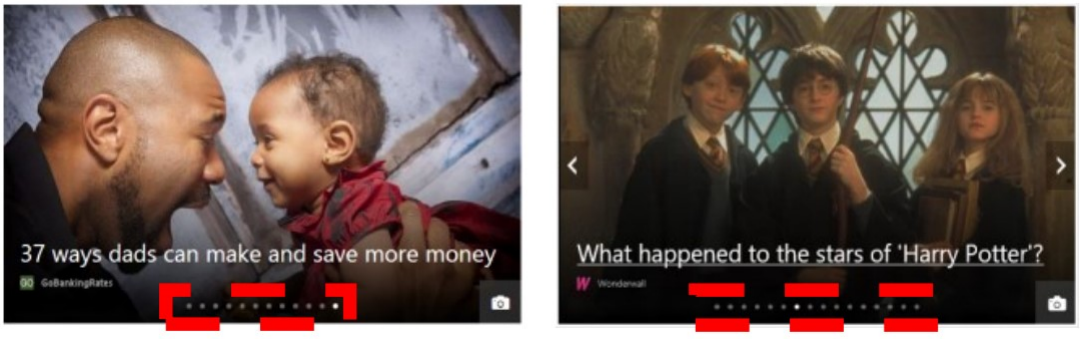

Figure 1: SRM Error.

Prior to the experiment, a rise in user engagement, i.e. a greater number of clicks and interactions with the carousel was anticipated. The experiment showed a significant drop in user engagement with the carousel in spite of having robust data collection techniques, adequate statistical power to detect modest changes, and a suitable platform that performed

reliable analysis. This outcome completely contradicted what the team was hoping for based on the learnings from related experiments that showed an increase in clicks. Users who saw more cards seemed to click less, therefore it seemed like a terrible decision to implement this modification based on the results.

The experimentation platform's data quality alert, which revealed an unexpected proportion of users in the experimental variations, was a key indicator of this unforeseen outcome. As a result of the new 16-card carousel, the treatment group in question had fewer users in the analysis than expected based on the setup of the trial. Due to the substantial statistical significance of the deviance from the configured split, the possibility of random chance was ruled out. This discrepancy is called SRM and for this reason, it is essential to continuously monitor for SRM in order to avoid experimenters from drawing incorrect findings. Once the problem generating the SRM in the above experiment was fixed, it was discovered that the adjustment was truly beneficial and greatly raised the level of product engagement.

SRM in online controlled experiments (OCEs) can occur due to several issues, including errors in random treatment assignment, lost telemetry from data logging problems, and design flaws like incorrect variant configuration. Chen et al. (2018) at LinkedIn identified SRMs occurring during triggered analysis, revealing that approximately 10% of such analyses had an SRM. Triggered analysis includes only users affected by the change and their counterfactuals, where incorrect conditions can lead to SRM. Detecting and addressing these errors early is crucial to ensuring the validity and reliability of the experiment's results.

Let $x_T^1, x_T^2, \ldots, x_T^k$ and $x_C^1, x_C^2, \ldots, x_C^k$ represent the cumulative sample counts in the treatment (T) and control (C) groups of an experiment up to time k. Suppose the assigned traffic proportions are r_T and r_C for the treatment and control groups, respectively. We aim to investigate the sample ratio mismatch through a two-sided hypothesis test under a binomial setting.

The observed sample ratio p and the expected sample ratio p_0 are defined as follows:

$$p = \frac{x_T^k}{x_T^k + x_C^k},$$

$$p_0 = \frac{r_T}{r_T + r_C}.$$

We construct the following hypotheses:

$$H_0 : p - p_0 = 0,$$
$$H_A : p - p_0 \neq 0.$$

The decision to reject the null hypothesis H_0 in favor of the alternative hypothesis H_A is based on the p-value obtained from the hypothesis test. The p-value, which measures the probability of obtaining results at least as extreme as those observed, given that the null hypothesis is true, is compared to a predetermined significance level α (commonly set at

0.05).

Decision Rule:

- If the p-value $\leqslant \alpha$, we reject the null hypothesis H_0. This indicates that there is a statistically significant SRM.
- If the p-value $> \alpha$, we do not reject the null hypothesis H_0. This indicates that there is no statistically significant SRM.

Type I Error: A Type I error occurs when we incorrectly reject the null hypothesis H_0 when it is actually true. The probability of making a Type I error is denoted by the significance level α. For us, a Type I error would mean concluding that there is a sample ratio mismatch when, in fact, the observed sample ratio is consistent with the expected sample ratio. By setting α at a conventional level (such as 0.05), we limit the probability of making this type of error to 5%.

By adhering to these decision criteria, we ensure that our hypothesis testing framework maintains a balance between detecting genuine sample ratio mismatches and minimizing the risk of false positives.

Nowadays, conducting an SRM test to confirm the experiment's execution is the recommended procedure. This procedure makes sure that the data processing stages and assignment mechanism are functioning as expected. Typically, a Chi-squared test is performed on the total units observed in each treatment group against the intended assignment probabilities at the end of data collection and prior to analysis. This method has a significant drawback even though it is statistically sound: discovering implementation problems only after data collection is complete may be too late.

The ideal scenario is to validate the implementation at the beginning. But if the Chi-squared test is run too soon, there might not be sufficient evidence to reject the null hypothesis, which would allow implementation errors to remain undetected. The Type-I error probability guarantee of the Chi-squared test is simply valid for a single execution, raising the issue of when to carry it out. The Chi-squared test is rigid due to the conflict between performing the test early enough to avoid wasting units and sufficiently late in order to have adequate power.

Due to these challenges, many practitioners make the mistake of carrying out significance tests at frequent intervals without applying the appropriate multiplicity adjustment. This raises the possibility of a Type I error. Typically, this situation leads to a team performing an experiment, doubting the validity of the experiment's implementation, and performing a Chi-squared test. The team's skepticism remains if the test fails to reject the null hypothesis. Hence, they decide to repeat the test later, concluding that the initial test lacked power. This process is repeated until the null hypothesis is finally rejected, resulting in a false positive.

According to the work by Armitage et al. (1969), after just five uses, the likelihood of receiving a false positive from a Chi-squared test set at the 0.05 threshold can rise to 0.14. Sampling to a predetermined conclusion is a risk when deciding whether to conduct a new

test based on the results of an earlier test. To address these concerns, the proposed solution (in Section 4.1) will focus on strategies to mitigate the risks of false positives and ensure robust experimental results.

3.2 Anomaly Detection

In the context of monitoring A/B tests for a fintech application, it is crucial to detect any unintended adverse effects of new features on user behavior. This involves continuously tracking key performance metrics to ensure that any significant deviations are promptly identified. In large fintech applications due to the high stakes in terms of user satisfaction, we want early detection of potential issues and maintenance of user experience. Hence, we typically want an algorithm suited for potential real-time anomaly detection. We have not implemented real-time detection in this paper, however, we aim to provide an algorithm that can be integrated into a real-time anomaly detection system. Two primary approaches for identifying deviations in data are anomaly detection and change point detection, with **change point detection** being a specialized form of anomaly detection.

Anomaly detection involves identifying data points or patterns in a dataset that significantly deviate from the expected behavior. These deviations can indicate outliers, errors, or rare events. Anomaly detection is typically used to identify single or isolated points in time that are unusual compared to the rest of the data.

Change point detection, as a subset of anomaly detection, focuses on identifying points in time where the statistical properties of a time series change significantly. While anomaly detection generally identifies isolated points, change point detection extends this by identifying intervals where the underlying process has shifted. This can include changes in mean, variance, or other distributional properties. Essentially, change point detection is used to detect anomalies that represent shifts in the underlying data distribution over time, making it essential for understanding more complex, longer-term shifts in user behavior.

In the context of A/B testing for a fintech application, change point detection is more appropriate than anomaly detection because we are interested in detecting shifts in outcome or guardrail metrics that indicate a systematic change in user behavior due to the treatment. A/B tests involve monitoring metrics over time. Change points provide actionable insights for stakeholders to make decisions about continuing or stopping the test, whereas isolated anomalies may not provide sufficient context. Change point detection can help identify significant shifts in user metrics such as user engagement, conversion rates, transaction success rates, or error occurrences that may indicate a problem with the new feature.

Now, we shift our focus entirely on change point detection, and dive into the mathematical formulation.

3.2.1 Mathematical Formulation

Given a time series $\{x_n\}_{n=1}^{N} \subset \mathbb{R}$, which is an ordered sequence of data, the objective is to identify a set of K change points whose positions are $\{t_1, t_2, \ldots, t_K\}$ where the statistical properties of the series change. Each change point position is an integer between 1 and $N-1$ inclusive. Specifically, we seek to:

1. Estimate the number of change points K.

2. Determine the locations of the change points $\{t_1, t_2, \ldots, t_K\}$.

Cost Function: The problem can be formulated as an optimization problem. Let $c(x[a : b])$ be a cost function that measures the dissimilarity within a segment of the time series from a to b. We define $t_0 = 0$ and $t_{K+1} = N$ and assume that the change points are ordered such that $t_i < t_j$ if, and only if, $i < j$. Consequently the K change points will split the data into $K + 1$ segments, with the i^{th} segment containing $x_{(t_{i-1}+1):t_i}$ points.

The objective is to minimize the sum of the costs over all segments defined by the change points:

$$(\hat{t}_1, \hat{t}_2, \ldots, \hat{t}_K) = \arg \min_{(t_1, t_2, \ldots, t_K)} \sum_{k=1}^{K+1} c(x_{(t_{k-1}+1):t_k}), \tag{1}$$

with the convention that $t_0 = 0$ and $t_{K+1} = T$.

Through this objective function we are essentially finding points in the time series where splitting the time series minimizes dissimilarities within each segment. This allows us to identify the points where the statistical properties of the data change.

3.2.2 Algorithms Used for Change Point Detection

We explored and experimented with the following algorithms for change point detection based on the literature reviewed. Here is a brief introduction for each algorithm used:

1. CUSUM (Cumulative Sum): Detects shifts in the mean by accumulating deviations from a target value.

2. PELT (Pruned Exact Linear Time): An efficient algorithm that uses pruning to reduce the computational cost while providing an exact solution.

3. Dynamic Programming: Provides an exact solution by minimizing the total cost over all possible segmentations, but is computationally expensive.

4. Binary Segmentation: An approximate method that recursively splits the series at the most significant change point.

Note that conversion rate is a key metric in A/B testing that can be directly impacted by changes in the user experience. A significant drop or increase in conversion rates usually indicates a meaningful change in user behavior.

Now coming to comparison between the algorithms implemented and their workings, CUSUM maintains a cumulative sum of deviations from a target value. For each new data point, it updates the cumulative sum and checks if it exceeds a threshold. This process involves a constant amount of work per data point, resulting in linear complexity. CUSUM's linear complexity $\mathcal{O}(n)$ is efficient for real-time applications and large datasets, offering a good balance between speed and sensitivity.

Pruned Exact Linear Time (PELT) is efficient and scalable for large datasets, capable of detecting multiple change points. While efficient, PELT requires careful selection of the penalty term β which is added to the cost function described in Equation 1 in Section 3.2.1 as a penalty term to avoid overfitting. However, the selection of β can be challenging since it requires cross-validation, domain expertise to know the significance of potential changes to help in setting a reasonable range for β, and empirical testing. Also, PELT typically requires the entire data set to be available for analysis, making it less suitable for real-time applications. PELT leverages dynamic programming to find the optimal segmentation while pruning unnecessary computations to improve efficiency. The pruning strategy reduces the problem size logarithmically, resulting in a computational complexity of $\mathcal{O}(n \log n)$. PELT is more computationally intensive than CUSUM but offers a balance between computational efficiency and the ability to detect multiple change points accurately.

Dynamic Programming involves computing the cost (cost function in Equation 1, Section 3.2.1) of segmentations for all possible pairs of start and end points in the time series. The basic dynamic programming approach has a complexity of $\mathcal{O}(n^2)$, but additional features or constraints can increase this to $\mathcal{O}(n^3)$. Dynamic programming is more computationally expensive than both CUSUM and PELT. It is suitable for scenarios where accuracy is paramount, and computational resources are less of a concern.

On the other hand, Binary Segmentation iteratively applies a single change point detection method to subdivide the data, similar to a divide-and-conquer approach. Each segmentation step involves linear scans, and the recursive nature of the algorithm leads to a logarithmic number of steps. Binary segmentation is efficient with $\mathcal{O}(n \log n)$ complexity. However, it can be less accurate in detecting multiple change points compared to PELT, due to its greedy nature.

Table 1 gives an overview of computational complexities these algorithms have, and accordingly their efficiency and suitability for certain scenarios.

For A/B testing in Wise, we need our algorithm to be suited for the following properties:

1. Sensitivity to Small Shifts: For A/B testing in a fintech company like Wise, small shifts in metrics like conversion rates can have significant implications. For example, even a slight decrease in the rate at which users complete transactions could indicate an issue

Algorithm	Complexity	Efficiency	Suitability
CUSUM (Cumulative Sum Control Chart)	$\mathcal{O}(n)$	High	Real-time monitoring, small shifts
PELT (Pruned Exact Linear Time)	$\mathcal{O}(n \log n)$	Moderate	Multiple change points, accuracy
Dynamic Programming	$\mathcal{O}(n^2)$ to $\mathcal{O}(n^3)$	Low	High accuracy, smaller datasets
Binary Segmentation	$\mathcal{O}(n \log n)$	Moderate	Quick segmentation, larger datasets

Table 1: Quantitative Comparison of Change Point Detection Algorithms.

with a new feature. Since the financial implications of such shifts can be substantial, it is crucial to detect these changes as early as possible.

2. Real-Time Detection: Critical in A/B testing for quickly identifying and reacting to issues. Wise operates in the highly competitive and fast-paced fintech industry, hence the ability to immediately detect and address problems with new features can prevent potential financial losses and maintain customer satisfaction.

3. Simplicity and Implementation: This allows for quick deployment and integration into existing systems. In a fintech company like Wise, where multiple A/B tests might be running simultaneously, it is crucial that the chosen algorithm is straightforward to implement and does not require extensive parameter tuning or new complex configurations.

4. Computational Efficiency: Wise processes large volumes of transaction data in real-time. An algorithm that is computationally efficient ensures that the system remains responsive and can handle the data throughput without lag or delay.

So, the algorithms are assessed based on the above criterion along with empirical evidence to compare and contrast the algorithms mentioned above, and a final selection is made.

3.3 Ratio OECs

When running A/B tests on continuous (e.g. time spent using the app) or discrete (number of transfers) variables, the Central Limit Theorem allows to perform statistical inference on the group means. To compute the test statistic and appropriate variances, the Student's t-test formulas are used:

$$T = \frac{\bar{X}_{treatment} - \bar{X}_{control}}{\sqrt{\mathrm{Var}(\bar{X}_{treatment}) + \mathrm{Var}(\bar{X}_{control})}}, \tag{2}$$

$$\mathrm{Var}(\bar{X}) = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2.$$

Crucially, t-test requires that the observations are independent and identically distributed. An issue arises when the OEC is a ratio OEC. A general form of this type of OEC is shown below:

$$R(U') := \frac{\sum_{u\in U'}\sum_{\omega\in\Omega_u} x(\omega)}{\sum_{u\in U'}\sum_{\omega\in\Omega_u} y(\omega)}. \tag{3}$$

In words, the OEC is defined as a ratio of sums of all events ($\omega \in \Omega_u$) summed across all units ($u \in U'$). For example, to calculate Click-Through-Rate (CTR, clicks divided by views), we first sum all of the clicks for each user, then sum those sums, then do the same calculations for views, and calculate the ratio. One can see that when the randomisation unit is the user, its events (views or clicks) will be correlated with each other, violating the t-test assumptions.

There are a couple of alternatives that try to address these issues. A simple approach is bootstrap. One can resample from the dataset many (B) times and construct multiple bootstrap samples and calculate the OEC for each. Its main drawback is that it is extremely computationally intensive, requiring $n \times B$ operations.

The second common approach is to use the Delta method. Delta method takes into consideration the within-user correlations and allows the scientist to approximate the correct variance of the ratio OEC. This approach comes from a first-order Taylor expansion. Consider a function of a random variable and it's first-order Taylor expansion around the mean:

$$f(Z) \approx f(\mu) + f'(\mu)(Z - \mu).$$

Then define:

$$g(X, Y) = \frac{X}{Y}.$$

After performing the expansion (calculations omitted), the result is:

$$g(X, Y) \approx \frac{\mu_X}{\mu_Y} + \frac{1}{\mu_Y}(X - \mu_X) - \frac{\mu_X}{\mu_Y^2}(Y - \mu_Y).$$

Then it is simply a matter of calculating the variance of this expansion using the standard properties of variance and simplifying. The resulting formula is:

$$\mathrm{Var}(R) \approx \frac{\mathrm{Var}(X)}{\mu_Y^2} + \frac{\mu_X^2 \mathrm{Var}(Y)}{\mu_Y^4} - 2\frac{\mu_X \mathrm{Cov}(X,Y)}{\mu_Y^3},$$

where R is defined as in (3).

Now the scientist can use the obtained variance to modify the t-test and obtain a statistic that is asymptotically normally distributed. While the delta method solves the issue of within-user correlation, it does not work well with sensitivity improvement techniques, resulting in the need for large sample sizes.

Yet another approach is to naively calculate the ratio for each user individually:

$$A_{X,Y}(u) := X(u)/Y(u). \tag{4}$$

Then it is simply a matter of calculating the average per group and conducting a t-test. The main issue with this approach is that an OEC constructed in this way does not preserve the directionality and the significance level of the original OEC.

Therefore, it is crucial to develop a method of handling ratio OEC that:

1. preserves the original OEC's directionality - the test statistic obtained from linearised metric should have the same sign as the test statistic of the underlying OEC,
2. allows to calculate the achieved significance level efficiently,
3. provides an OEC that allows the scientist to apply efficient sensitivity-improving approaches.

4 Proposed Solution

In this section we will detail our section-wise proposed solution. It outlines a comprehensive approach to the problems introduced by delving into SRM (Section 4.1), Anomaly Detection (Section 4.2), and Ratio OECs (Section 4.3) - each detailing the approaches used for our solution and providing justifications for those solutions.

4.1 Sample Ratio Mismatch

Just like a fever is a symptom of many illnesses, a SRM is a symptom of a variety of data quality issues (Fabijan et al. 2019). SRM occurs when there is a discrepancy between the intended and actual allocation of units in an experiment's treatment arms. Detecting these mismatches early is crucial to avoid invalid results and unnecessary costs.

The chi-square test is a popular and straightforward method for detecting SRM, but it typically identifies issues only after data collection. To address this limitation, we adopted the approach outlined by Lindon and Malek (2020), which harmonizes sequential and Bayesian methodologies to continuously monitor traffic counts and test for significant imbalances in visitor counts.

The methodology for SRM detection can be categorized into distinct stages. These steps follow the approach used by Lindon and Malek (2020):

- Step 1: Define a relevant Bayes factor.
- Step 2: Demonstrate that the Bayes Factor is a Nonnegative Supermartingale under the Null Hypothesis.
- Step 3: Construct a Test Martingale using Martingale Inequalities to Control the Type I Error Probability.
- Step 4: Invert the Sequential Test to Obtain a Confidence Sequence with a Coverage Guarantee.

In Step 1, a Bayes factor is defined to compare the null hypothesis M_0, where observations follow a Multinomial (1, θ_0) distribution, against an alternative hypothesis M_1 with a Dirichlet prior on θ. This Bayes factor measures how much more likely the data is under the alternative hypothesis than under the null hypothesis. Step 2 demonstrates that this Bayes factor forms a nonnegative supermartingale under the null hypothesis, meaning its expected value does not increase over time if M_0 is true, ensuring the test's reliability. A test martingale is constructed using martingale inequalities in Step 3, allowing the Type I error probability to be controlled. A threshold for the Bayes factor is set and this ensures that the probability of wrongly rejecting the null hypothesis remains below a chosen level u. A sequential p-value is also defined that adjusts as more data is observed, providing a stopping rule for rejecting M_0. Finally,

this test is inverted in Step 4, to create a confidence sequence for θ, updating with each new observation and maintaining a coverage probability of at least $1 - u$. In this approach, if the null hypothesis is false, the Bayes factor will eventually exceed the threshold, ensuring the test's power. By this method, a robust sequential test effectively balancing error control and detection power is created to work with count data, and this is used while detecting an SRM in the experiment data.

Lindon and Malek (2020) developed a Sequential Sample Ratio Mismatch (SSRM) test based on this method. They created a Python package, ssrm_test, which includes functions to calculate p-values according to the sequential approach they developed. We have created a class to integrate this onto Wise's experimentation platform to check for SRM. To enhance the reliability and efficiency of the experimentation platform, we developed a Python class integrating the SSRM test. This class allows continuous monitoring and early detection of SRM's in experimental data. Integrating this into Wise's experimentation platform would be beneficial for ensuring the validity and efficiency of their experiments.

In summary, the SSRM test offers a structured approach to SRM detection, by continuously monitoring the allocation of visitors to different treatment arms and calculating sequential p-values. This ensures early detection of mismatches, thus preserving the experiment's integrity and validity. The implementation of the class follows a rigorous statistical methodology while providing practical tools for visualizing and understanding the results. Integrating this test onto the experimentation platform represents a significant advancement in maintaining high-quality experimental data and delivering accurate, reliable insights from experiments. The code for the SRM_Test class can be found in our GitHub repository.

In Section 5.1, we will present numerical experiments to evaluate the effectiveness of the SSRM test in detecting sample ratio mismatches. These experiments will demonstrate how the test is applied to real-world data sets, showcasing its ability to identify and address discrepancies in treatment allocations.

4.2 Anomaly Detection

As mentioned in the Problem Formulation (Section 3.2), it is pertinent for our chosen algorithm to be suitable for real-time change point detection. We also want it to be sensitive to small changes for early detection of even minor degradations in performance or user experience to allow timely intervention. We also want it to be highly interpretable. Hence, CUSUM is the most suitable algorithm for our use-case due to its ability to quickly and sensitively detect even small changes in data, which is essential for identifying shifts in user behavior or financial metrics. This sensitivity ensures timely responses to changes, which is crucial for maintaining the app's competitiveness and user experience. Moreover, its sequential nature aligns well with real-time monitoring requirements, providing immediate feedback on ongoing tests. CUSUM is relatively simple to understand and implement. It doesn't require

complex parameter tuning or extensive computational resources. CUSUM algorithm satisfies all the properties that our algorithm requires as mentioned in Section 3.2.2.

Hence, after a thorough review we have decided to use CUSUM analysis for this problem. Let us next dive deeper into how the CUSUM algorithm operates.

4.2.1 CUSUM Algorithm

CUSUM works by accumulating deviations from the target mean over time. When the cumulative sum exceeds a predefined threshold, it signals that a significant change has occurred. The CUSUM method works by resetting the cumulative sum when it exceeds the threshold. This allows for detecting both upward and downward shifts in the mean.

A brief working of the traditional CUSUM algorithm is as follows:
1. Calculate the Target Mean: Determine the target mean conversion rate from historical data.
2. Compute Deviations: For each new data point, compute the deviation from the target mean.
3. Update Cumulative Sum: Incrementally update the cumulative sum with the computed deviations.
4. Check Threshold: Compare the cumulative sum against the predefined threshold. If it exceeds the threshold, a change point is detected.

The CUSUM algorithm does not explicitly minimize a cost function like the one in Equation 1 in the way that some other algorithms, like the PELT or Dynamic Programming methods do for change point detection. However, it can be interpreted as implicitly minimizing a detection delay cost (the time it takes to detect a change after it has occurred) under certain formulations. It can be viewed through the lens of Sequential Analysis, where the objective is to minimize the expected detection delay subject to a constraint on the false alarm rate (the frequency with which the algorithm incorrectly signals a change point when there is none).

Now, when talking about implementing CUSUM, an important distinction in the type of data to use needs to be considered- this is between **Stationary** and **Non-Stationary** time series data (see Figure 2):
1. **Stationary Time Series**: The data's statistical properties, such as mean, variance, and autocorrelation, remain constant over time. In other words, the data's behavior does not change over time, making it predictable, and there are no trend or seasonal effects.
2. **Non-Stationary Time Series**: A time series is non-stationary if its statistical properties change over time. This could be due to trends, seasonality, or other evolving structures in the data.

In a fintech company, the data collected can be both stationary and non-stationary, depending on the specific type of data and the underlying processes generating it. However, most critical datasets in fintech are typically non-stationary due to the dynamic nature of financial markets, user behaviors, and external economic factors. For example, transaction data is often non-stationary due to seasonal effects (e.g., holidays), trends (e.g., increasing use of digital payments), and external factors like economic policies or market conditions.

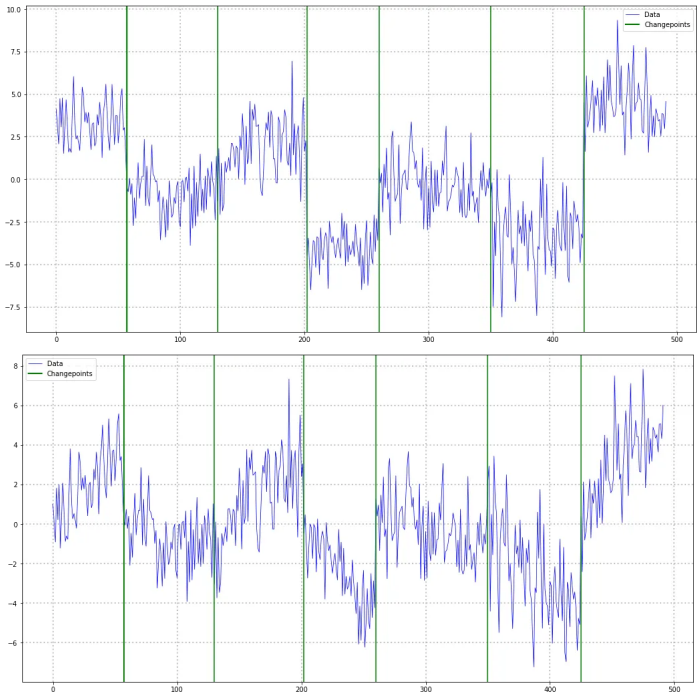

Figure 2: Example Per-Segment Stationary (Top) and Non-Stationary (Bottom) Time-Series Data (Blue) with Change Points (Straight Green Lines) (from Seitz (2022)).

Traditional CUSUM is particularly well-suited for stationary data because the baseline statistical properties (mean and variance) do not change over time. It can effectively detect small shifts in the mean as it accumulates deviations from a fixed reference point. However, when it comes to non-stationary data, traditional CUSUM might not be reliable.

Hence, it becomes extremely important to take this into account and adjust our approach accordingly.

We now introduce **Probabilistic CUSUM**, which is an extension of the traditional CUSUM algorithm that is particularly well-suited for detecting changes in non-stationary data. Its design allows it to handle variations in data properties over time, making it more robust in dynamic environments such as those encountered in fintech. This variant is better suited for non-stationary data as it incorporates a probabilistic measure, allowing for dynamic thresholding.

The algorithm's working can be summarized as follows:

1. Standardization of Observations: Given a time series $\{x_t\}_{t=1}^{T} \subset \mathbb{R}$, each observation is standardized using the estimated mean $\hat{\mu}_x$ and standard deviation $\hat{\sigma}_x$:

$$Z_t = \frac{x_t - \hat{\mu}_x}{\hat{\sigma}_x}.$$

This standardization normalizes the data, accounting for potential changes in scale.

2. Cumulative Sum Calculation: The algorithm maintains a cumulative sum S_T of the standardized observations:

$$S_T = \sum_{t=1}^{T} Z_t.$$

We heuristically employ the Central Limit Theorem (CLT) to justify the assumption that the cumulative sum of standardized observations approximates a normal distribution. Specifically, under the conditions of the CLT, as the number of observations increases, the distribution of the cumulative sum of independent and identically distributed (i.i.d.) random variables converges to a normal distribution. Although real-world data may not strictly satisfy all the assumptions of the CLT (such as perfect independence or identical distribution), we leverage the CLT heuristically to simplify our formulation. By assuming that the standardized cumulative sum follows a normal distribution, we can apply probabilistic thresholds for change detection, enabling more effective identification of significant shifts in the underlying process. Hence, the following holds approximately.

3. Normalization: To maintain comparability, the cumulative sum is normalized by the square root of the number of observations T:

$$\tilde{S}_T = \frac{1}{\sqrt{T}} S_T \sim \mathcal{N}(0, 1),$$

where $\tilde{S}_T$ is the normalised sum, $\mathcal{N}(0, 1)$ represents the standard normal distribution with mean $\mu = 0$ and variance $\sigma^2 = 1$. This normalization ensures that the cumulative sum follows a standard normal distribution under the null hypothesis of no change.

4. Probability Calculation: We define the cumulative distribution function (CDF) of the standard normal distribution as $\Phi(\cdot)$, which is used to calculate the probability that the observed cumulative sum $\tilde{S}_T$ is less than or equal to the observed value $\tilde{s}_T$:

$$\Phi(\tilde{s}_T) \approx P(\tilde{S}_T \leqslant \tilde{s}_T).$$

To consider both directions (positive and negative deviations), the two-tailed probability is calculated as:

$$P\left(|\tilde{S}_T| \geqslant |\tilde{s}_T|\right) = 2\left(1 - \Phi\left(|\tilde{s}_T|\right)\right).$$

This value represents the probability that the observed deviation is at least as extreme as the observed value, considering both positive and negative deviations.

5. Change Point Detection: A change point is detected if the probability P falls below a predefined threshold p_{limit}:

$$P\left(|\tilde{S}_T| \geqslant |\tilde{s}_T|\right) < p_{\text{limit}},$$

where p_{limit} is a real number in the range $(0, 1)$. p_{limit} is a critical threshold parameter that determines whether an observed deviation is significant enough to be considered a change

point. It is chosen empirically based on historical data or validation datasets using cross validation techniques. It could also require certain domain expertise, for example, in a highly sensitive financial environment, a lower p_{limit} might be chosen to minimize the chance of missing a critical change, even if it means accepting more false positives.

When this condition is met, the algorithm signals a change, indicating a significant shift in the data's behavior.

Here are a few advantages of using Probabilistic CUSUM over Traditional CUSUM in our scenario:

- Two-Sided Detection: Unlike traditional CUSUM, which may primarily detect changes in one direction (either an increase or decrease), Probabilistic CUSUM inherently accounts for both directions by using a two-sided test. This is achieved by evaluating the absolute deviation from the mean, thereby detecting any significant deviation regardless of its direction. This two-sided approach is essential in scenarios where both positive and negative shifts are equally important.

- Dynamic Thresholding: The probabilistic approach allows for dynamic thresholding, as the detection criterion is based on a probability calculation rather than a fixed threshold. This flexibility is crucial in non-stationary environments, where fixed thresholds might either miss changes (if set too high) or lead to excessive false positives (if set too low).

- Flexibility and Extension: The probabilistic framework allows for easy integration with other statistical models and methods, such as Bayesian updates or other probabilistic forecasting tools. This makes it a versatile tool in a data-rich environment like fintech, where combining multiple sources of information can enhance predictive power.

- Robustness to Changes in Data Distribution: By utilizing the cumulative distribution function of the standard normal distribution, Probabilistic CUSUM accounts for the overall distribution of the data, making it less sensitive to outliers or transient changes that might not indicate a true shift in the underlying process.

Thus, **our final choice of algorithm is Probabilistic CUSUM**, for which we have created a class function that can be referred to in the GitHub repository. Its probabilistic interpretation and ability to standardize and normalize observations make it particularly well-suited for non-stationary data, where the statistical properties of the data can change over time. This robustness and flexibility are critical for real-time monitoring and anomaly detection in dynamic environments, such as those found in fintech, where timely and accurate detection of changes is essential for decision-making and maintaining system integrity.

4.3 Ratio OECs

Budylin et al. (2018) propose a transformation of a ratio OEC that satisfies all three criteria

mentioned at the end of the problem formulation section. It is called linearization and it introduces the following user-level metric:

$$L_{X,Y,\kappa}(u) := X(u) - \kappa Y(u), \kappa \in \mathbb{R}, \tag{5}$$

and the corresponding OEC:

$$\mathfrak{L}_{X,Y,\kappa}(U) = \text{avg}_U L_{X,Y,\kappa}. \tag{6}$$

To guarantee the required properties, we need to impose constraints on the parameter κ.

4.3.1 Directionality preservation

Firstly, we will look at the guarantees of preservation of directionality.

Proposition 1 (Budylin et al. 2018)**.** *Let X and Y be user level metrics (such that Y_A, Y_B are positive), R be the (source) ratio OEC, and L be the linearized OEC defined by* (6)*. The parameter κ being set as $\kappa(\eta) = (1-\eta)R_A + \eta R_B, \eta \in \mathbb{R}$, implies the following identity on the OEC's differences between the control and treatment variants:*

$$\Delta(L_{X,Y,\kappa(\eta)}) = ((1-\eta)Y_B + \eta Y_A)\Delta(R). \tag{7}$$

Proof. First of all, let us show that, for $\eta = 0$ (i.e., $\kappa = R_A$),

$$\Delta(L_{X,Y,\kappa(0)}) = Y_B \Delta(R). \tag{8}$$

Plugging in the values and simplifying:

$$\begin{aligned}\Delta(L_{X,Y,\kappa(0)}) &= \Delta(X) - \kappa(0)\Delta(Y) = (X_B - X_A) - (X_A/Y_A)(Y_B - Y_A) \\ &= X_B - X_A Y_B/Y_A = Y_B(X_B/Y_B - X_A/Y_A) = Y_B(R_B - R_A) \\ &= Y_B \Delta(R).\end{aligned}$$

Similarly, one can show that $\Delta(L_{X,Y,\kappa(1)}) = Y_A \Delta(R)$ for $\eta = 1$ (i.e., $\kappa = R_B$). Finally, one can represent $L_{X,Y,\kappa(\eta)}$ as a linear combination of $L_{X,Y,\kappa(0)}$ and $L_{X,Y,\kappa(1)}$ to reduce this case to the previous ones:

$$\begin{aligned}\Delta(L_{X,Y,\kappa(\eta)}) &= \Delta(X) - ((1-\eta)R_A + \eta R_B)\Delta(Y) = (1-\eta)\Delta(L_{X,Y,R_A}) + \eta\Delta(L_{X,Y,R_B}) \\ &= (1-\eta)Y_B\Delta(R) + \eta Y_A \Delta(R).\end{aligned}$$

Hence, (7) holds. □

First, Proposition 1 establishes a clear sufficient condition on the parameter κ to maintain the directionality of R within L. Specifically, when κ lies between R_A and R_B—the source OEC values for the control and treatment versions, respectively—the average value of the user-level metric L enables an analyst to consistently infer the direction of the system quality change, in alignment with conclusions drawn from the source OEC R. The authors of the linearization method encapsulated this assertion in the following corollary:

Corollary 1. *Given the user level metrics X and Y (such that Y_A, Y_B are positive), the (source) ratio OEC R defined by* (3)*, and the linearized OEC L defined by* (6)*, if $\kappa \in [\min\{R_A, R_B\}, \max\{R_A, R_B\}]$ (in particular, if $\kappa = R_A$ or R_B), then sgn$\Delta(R)$ = sgn$\Delta(L)$.*

Second, Proposition 1 establishes the relation between the treatment effect magnitude of the source ratio OEC and the one of the linearized OEC. For instance, let $\kappa = R_A$ for each A/B test, then Proposition 1 implies a proportion between the differences $\Delta(L)$ and $\Delta(R)$:

$$\Delta(L) = Y_B \Delta(R). \tag{9}$$

It is important to note that the coefficient Y_B in (9) explicitly depends on a particular A/B experiment. However, in practice, this dependence may be significantly low, thereby rendering (9) as representing a (nearly) linear relationship between the differences $\Delta(L)$ and $\Delta(R)$ across a set of A/B experiments.

4.3.2 Significance level preservation

Secondly, we will look at the guarantees of preservation of significance level.

To measure the achieved statistical significance level of the difference $\Delta(L)$, it is necessary to construct a statistic with a known distribution under the null hypothesis. Given that the OEC L represents the average value of the metric L across users, who are assumed to be randomization units, the state-of-the-art t-statistic and Student's t-test are appropriate tools. This approach is valid when the parameter κ is set as a constant independent of any observations. However, if κ is set to R_A, two fundamental conditions of Student's t-test are violated: (a) the OEC values L_A and L_B for the treatment and control variants are not independent; and (b) the observations within $L(u)|u \in U_V, V = A, B$, are also not independent. Despite these violations, we argue below that Student's t-test with the t-statistic remains applicable for our difference $\Delta(L)$, allowing for the correct measurement of the achieved significance level (p-value).

Theorem 1 (Budylin et al. 2018)**.** *Given X and Y be user level metrics (such that Y is positive), R be the (source) ratio OEC, and L be the linearized OEC with the parameter $\kappa = R_A$. Let $T(L)$ be the t-statistic from applied to the OEC L with the metric L and $D(R) = \Delta(R)/\sqrt{\delta(R_A) + \delta(R_B)}$ be the asymptotic standard normal statistic of R obtained via the Delta method.*

1. *Then the following identity holds:*

$$T(L) = D(R)\sqrt{\frac{1-\gamma}{\delta(R_A)+\delta(R_B)}+\gamma}, \tag{10}$$

where $\gamma = (Y_A^2/Y_B^2 - 1)\delta(R_A) + \beta$ *and*

$$\beta = |U_B|^{-1}Y_B^{-2}\Delta(R)((R_A+R_B)\sigma_B^2(Y) - 2\widehat{\mathrm{Cov}}_B(X,Y)). \tag{11}$$

2. *If the sample correlation* $\widehat{\mathrm{Corr}}_B(X,Y)$ *is bounded as follows* $|\widehat{\mathrm{Corr}}_B(X,Y)| < c < 1$*, then the following inequality holds*

$$|T(L)/D(R) - 1| \leqslant C_1(c)|\Delta X/X_B| + C_2(c)|\Delta Y/Y_B| \tag{12}$$

for sufficiently small relative changes, i.e., $|\Delta X/X_B| < \epsilon_1(c)$ *and* $|\Delta Y/Y_B| < \epsilon_2(c)$*; where the constants* $C_1(c)$*,* $C_2(c)$*,* $\epsilon_1(c)$*, and* $\epsilon_2(c)$ *depend only on the bound* c*.*

3. *If* $|\mathrm{Corr}(X,Y|B)| < c < 1$ *and* $\mathrm{E}[X|A] \neq 0$*,* $\mathrm{E}[Y|A] \neq 0$*, then the* t*-statistics* $T(L)$ *is asymptotically normal under the null hypothesis that* $\mathrm{E}[\Delta X] = 0$ *and* $\mathrm{E}[\Delta Y] = 0$*.*

Proof. The claim 1 represents, in fact, an identity that directly follows from the definitions of $T(L)$ and $D(R)$ by rearrangement of their components. Let us compare the denominators of $T(L)$ and $D(R)$. The one of the t-statistic $T(L)$ is based on the standard deviation over the user samples $U_V, V = A, B$:

$$\begin{aligned}\sigma_V^2(L) \equiv \widehat{\mathrm{Cov}}_V(X - R_AY, X - R_AY) = \\ \widehat{\mathrm{Cov}}_V(X,X) + \widehat{\mathrm{Cov}}_V(R_AY, R_AY) - 2\widehat{\mathrm{Cov}}_V(X, R_AY) = \\ \sigma_V^2(X) + R_A^2\sigma_V^2(Y) - 2R_A\widehat{\mathrm{Cov}}_V(X,Y), V = A, B,\end{aligned} \tag{13}$$

where the last identity holds since R_A is constant for all users in U_V (i.e., $L(u) = X(u) - \kappa Y(u), \forall u \in U_V$, where $\kappa = R_A$) and can be thus factored out. Note that, for the variant $V = A$, the standard deviation $\sigma_A^2(L)$ is exactly $\delta(R_A)Y_A^2|U_A|$, while, for $V = B$, the standard deviation $\sigma_B^2(L)$ differs from $\delta(R_B)Y_B^2|U_B|$ in the presence of R_A in places of R_B, see the definition of β. Hence, we obtain:

$$T(L) = \frac{Y_B\Delta(R)}{\sqrt{Y_A^2\delta(R_A) + Y_B^2(\delta(R_B)+\beta)}} = \frac{\Delta(R)}{\sqrt{\delta(R_A)+\delta(R_B)+\gamma}}, \tag{14}$$

where β and γ are from the claim 1. This identity together with the definition of $D(R)$ implies (10).

The proof of Claim 2 is rather technical and therefore omitted. Claim 3 follows from Claim 2. Specifically, under the null hypothesis, the probabilities of $|\Delta X/X_B|$ and $|\Delta Y/Y_B|$

both approach zero. The inequality $|\widehat{\mathrm{Corr}}_B(X, Y)| < c + \epsilon < 1$ holds for some $\epsilon > 0$ with a probability that converges to one. Thus, using this inequality, we conclude that $T(L)/D(R)$ converges to one in probability. Consequently, $T(L)$ is asymptotically normal as $D(R)$ is. $\square$

The condition $|\widehat{\mathrm{Corr}}_B(X, Y)| < c < 1$ is necessary because it is possible to construct a scenario where the variance calculated by linearization is non-zero, while the variances calculated by the delta method and bootstrap are zero when $|\widehat{\mathrm{Corr}}_B(X, Y)| = 1$ and $R_B = 1$.

The condition $\mathrm{E}[\Delta X] = 0$ in the third claim of Theorem 1 can be substituted with $\mathrm{E}[\Delta R] = 0$. However, the authors do not yet know how to relax the condition $\mathrm{E}[\Delta Y] = 0$. In practice, it suffices if the changes in X and Y are sufficiently small.

Empirically, the mean values and variances of X and Y tend to vary within a narrow range, and the metric changes are relatively small, usually within a few percent. For instance, if the relative changes $\Delta(X)$ and $\Delta(Y)$ are no more than 2-3%, then the relative difference between $T(L)$ and $D(R)$ will be of a similar order, making these statistics comparable in terms of p-values. Therefore, in practice, the significance level achieved by $\Delta(L)$ calculated using the state-of-the-art t-test is consistent with that of $\Delta(R)$ obtained via the delta method and, consequently, the bootstrap technique.

To summarise, when the parameter is defined as the value of the ratio OEC of the control group the linearized OEC exhibits the desired properties. In the following sections, we will verify that the linearized metrics behaves as expected using synthetic datasets. We will also briefly describe the Python implementation on the Wise's tw-experimentation open-source library.

5 Results and Discussion

In this section we provide empirical evidences and numerical discussions of our results in all three core topics. This section deals with backing our solutions with experimental evidences to enhance their reliability.

5.1 Sample Ratio Mismatch

In this section, we explore the effectiveness of the Sequential SRM (SSRM) test by generating synthetic datasets with varying treatment splits to systematically analyze its performance (Section 5.1.1). To extend our analysis, we also validate the SSRM test using real-world data provided by Wise (Section 5.1.2). This approach underscores the applicability of the SSRM test in both controlled and dynamic environments.

5.1.1 Simulation Experiments

To evaluate the effectiveness of the SSRM test, we conducted simulation experiments inspired by previous studies (Kurennoy et al, 2024). Our simulations aimed to replicate real-world scenarios and the data received by the fintech company and validate the proposed methods from Section 4.1.

Experimental Setup

We developed a simulation function to generate experimental data for control and treatment groups. This function creates N observations based on a specified treatment share, which determines the split between the control and the treatment group, allowing for flexible simulation of different experimental scenarios.

This function takes the following parameters:

- N: The total number of observations.
- treatment share (p): The proportion of observations assigned to the treatment group.

In our simulations, we set N to 50,000 and varied the treatment share to generate the simulated data.

Simulation Studies

In the simulation study, we evaluated the performance of the SSRM test by integrating it into our analytical framework alongside the traditional Chi-squared test. To generate the simulated data, we used the function designed to produce variant data with $N = 50,000$ observations and different data splits. The primary objective was to compare the efficacy of the SSRM test with the traditional Chi-squared test across different data splits (e.g., 48/52, 50/50). The target significance level α was set to 5% (0.05), and we performed 1000 replications to ensure the reliability and validity of the results. We reject the null hypothesis and detect an SRM when the p-value goes below the target significance level $\alpha = 0.05$.

This setup enabled us to evaluate the performance of both the SSRM test and the Chi-square test under controlled conditions. By applying these tests to the simulated data, we were able to assess their effectiveness in detecting sample ratio mismatches and their overall performance in different scenarios.

Simulation Experiment Results

The results of our simulation experiments are illustrated in Figures 3 and 4, which present a comparison between the p-values obtained from the SSRM test and the Chi-square test as the number of observations increases. This comparison evaluates the sensitivity and robustness of the SSRM test relative to the traditional Chi-square test.

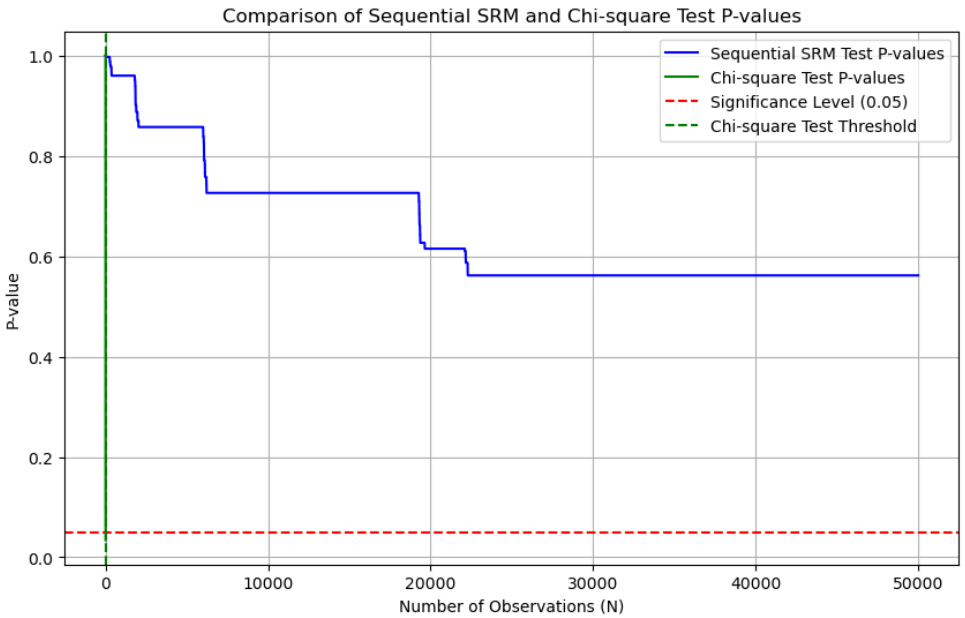

Figure 3: Comparison between p-values from SSRM test and Chi-square test in a 50-50 split (no significant imbalance).

In Figure 3, the SSRM test p-values, depicted by the blue line, demonstrate a gradual and controlled decrease as the number of observations increases. Initially, the SSRM test maintains p-values well above the significance threshold $\alpha = 0.05$ (indicated by the red dashed line), suggesting its robustness against false positives when no significant imbalance is present. Over time, as more data accumulates, the SSRM p-values decrease, reflecting the test's growing ability to detect imbalances accurately. This behavior indicates that the SSRM test is reliable in maintaining high p-values until sufficient data is collected, thereby reducing the likelihood of premature false detections.

In contrast, the Chi-square test p-values, shown by the green line, exhibit a sharp initial drop below the significance level, suggesting an early detection of imbalance. However, this sharp drop implies a higher susceptibility to false positives, as the p-values fall below the significance threshold prematurely. The rapid decrease in p-values highlights the potential for the Chi-square test to signal imbalance too early, which could lead to misinterpretations, especially during the initial stages of data collection. The green dashed line represents the

threshold for the Chi-square test, further emphasizing this point.

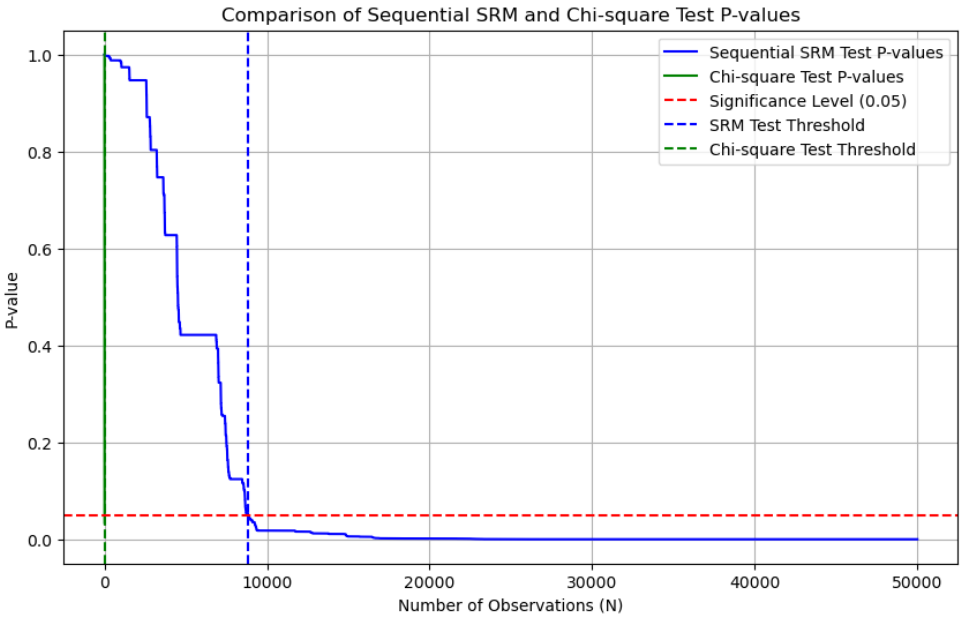

Figure 4: Comparison between p-values from SSRM test and Chi-square test in a 48-52 split (significant imbalance).

In Figure 4, the scenario of a 48/52 split with an error is analyzed. Here, the SSRM test first signals an imbalance at an observation count of 8838 with a p-value of 0.049, demonstrating its ability to detect a significant imbalance only after a substantial amount of data has been collected. This conservative approach ensures that false positives are minimized, providing a more reliable signal of imbalance. In stark contrast, the Chi-square test p-value drops below 0.05 at an observation count of merely 8, with a p-value of 0.034. This early detection by the Chi-square test underscores its propensity for false positives, flagging an imbalance prematurely with minimal data, which could lead to incorrect conclusions and potentially unnecessary corrective actions.

The red dashed line at 0.05 represents the significance level α for both tests. Comparing the performance of the two tests against this benchmark, it is evident that the SSRM test consistently maintains p-values above this significance level until sufficient data is available. This consistent performance underscores the SSRM test's ability to provide a balanced approach to detecting sample ratio mismatches, ensuring that significant imbalances are identified while minimizing the risk of false positives.

These observations indicate that the SSRM test is particularly suited for real-world applications where data accrues over time, and premature decisions based on insufficient data could be costly. The robustness and reliability of the SSRM test in identifying true imbalances without prematurely signaling false positives make it a preferable choice for monitoring sample ratios in dynamic and real-time data environments. Thus, the SSRM test proves to be an effective tool in ensuring the accuracy and integrity of A/B testing processes.

5.1.2 Validation using Real-World Data

Data Collection and Preparation

To validate our proposed approach, we utilized real-world data provided by our industrial partner, Wise, a fintech company. The dataset was designed to replicate the company's actual operational data, ensuring the validation was both realistic and relevant. The primary dataset consisted of checkout events captured over a 22-week period. Each event included details such as whether the user was in the treatment group (T), conversion status, revenue generated, pre-experiment revenue, number of actions taken, trigger dates, currency, country of origin, and various user segments. An example of the dataset is as follows:

T	conversion	revenue	pre_exp_revenue	num_actions	trigger_dates	currency	country_of_origin	segment_1	segment_2	segment_3
0	1	0	1259.336092	0.0	2022-01-01 00:16:00	EUR	US	New	Active	10- transfers
1	0	0	184.145856	0.0	2022-01-02 19:02:00	USD	UK	Old	Rare	10- transfers
2	0	0	461.242410	0.0	2022-01-12 10:55:00	EUR	USD	Old	Usual	10+ transfers
3	1	0	215.445976	0.0	2022-01-05 18:20:00	USD	USD	New	Active	10- transfers
4	1	0	214.730544	0.0	2022-01-15 23:38:00	EUR	UK	New	Rare	10- transfers

Table 2: Example of a Dataset Provided by Wise.

In addition to this primary dataset, Wise provided a supplementary dataset with various observations and detailed information on user attributes, such as account creation dates, business account status, country, platform used for creation, regional data, and product adoption metrics. We focused exclusively on the 'T' column in Table 2 or 'VARIANT' column in Table 3 for our analysis. The supplementary dataset contained extensive information, which allowed for a comprehensive analysis, ensuring that the results were robust across different user segments and behaviors.

VARIANT	DATE_CREATED	USER_DATE_CREATED	HAS_BUSINESS_ACCOUNT	ID_COUNTRY	M_CREATION_PLATFORM	...
0	2020-08-11 12:27:01.581308+00:00	2020-06-15 14:20:47+00:00	0	SEGMENT 1	SEGMENT 1	...
1	2020-08-25 09:09:13.575860+00:00	2011-07-25 14:50:03+00:00	0	SEGMENT 1	SEGMENT 2	...
1	2020-09-02 17:11:43.103826+00:00	2018-05-04 12:37:12+00:00	1	SEGMENT 1	SEGMENT 3	...
0	2020-08-29 00:12:32.359155+00:00	2015-11-29 19:05:24+00:00	0	SEGMENT 1	SEGMENT 2	...
1	2020-09-04 22:46:31.455368+00:00	2019-12-04 23:25:33+00:00	1	SEGMENT 1	SEGMENT 1	...

Table 3: Another Example of a Dataset Provided by Wise.

The extensive detail in the datasets enabled a thorough validation process, accounting for various user behaviors and characteristics. This meticulous approach ensured that our findings were not only statistically significant but also practically relevant, providing a high level of confidence in the applicability of our proposed methods to real-world scenarios.

Methodology

To validate our proposed approach, we utilized datasets based on the data collected by Wise. We developed a custom class, which is designed to prepare the data and perform the sequential SRM test. The class is structured to handle the entire process and effectively prepares the data, performs the SSRM test, and plots the results, allowing us to validate the performance of our proposed method against real-world data. The structured approach ensures that the validation process is thorough and the results are reliable, providing a robust

assessment of the SSRM test's efficacy in identifying statistically significant differences between variants.

Results

Our validation using real-world data provided significant insights into the performance of the Sequential SRM (SSRM) test compared to the traditional Chi-square test. The results are presented in Figures 5 and 6 demonstrating the behavior of p-values across different observation counts for both tests.

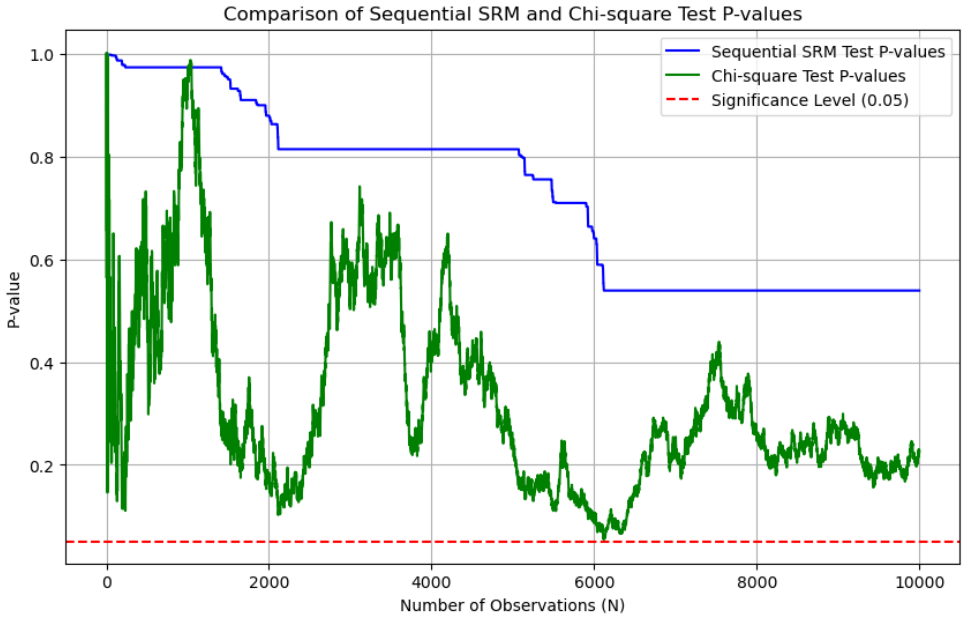

Figure 5: Comparison Between p-values from SSRM Test and Chi-Square Test for Primary Dataset Provided by Wise.

The graph in Figure 5 compares the sequential SRM test and the Chi-square test for a dataset with $N = 10,000$ observations. The Chi-square test (depicted in green) exhibited highly fluctuating p-values, frequently approaching but never falling below the significance level $\alpha = 0.05$ (marked by the red dashed line). This suggests that the Chi-square test is sensitive to variations in the data but does not conclusively detect imbalances at this sample size. The sequential SRM test (depicted in blue), on the other hand, maintained p-values well above the significance threshold for a more extended period before gradually declining, indicating a more stable performance.

The graph in Figure 6 illustrates the comparison for a larger dataset of $N = 392,501$ observations. The Chi-square test again showed a rapid decline in p-values, falling below the $\alpha = 0.05$ threshold at an observation count of 9,943. This suggests that with a larger sample size, the Chi-square test's sensitivity to detecting imbalances increases, leading to an increased likelihood of false positives. Conversely, the sequential SRM test maintained p-values above the 0.05 threshold for a significantly more extended period, indicating a more reliable performance in identifying true imbalances only when they are statistically significant.

Advantages of SSRM Over Chi-square Test

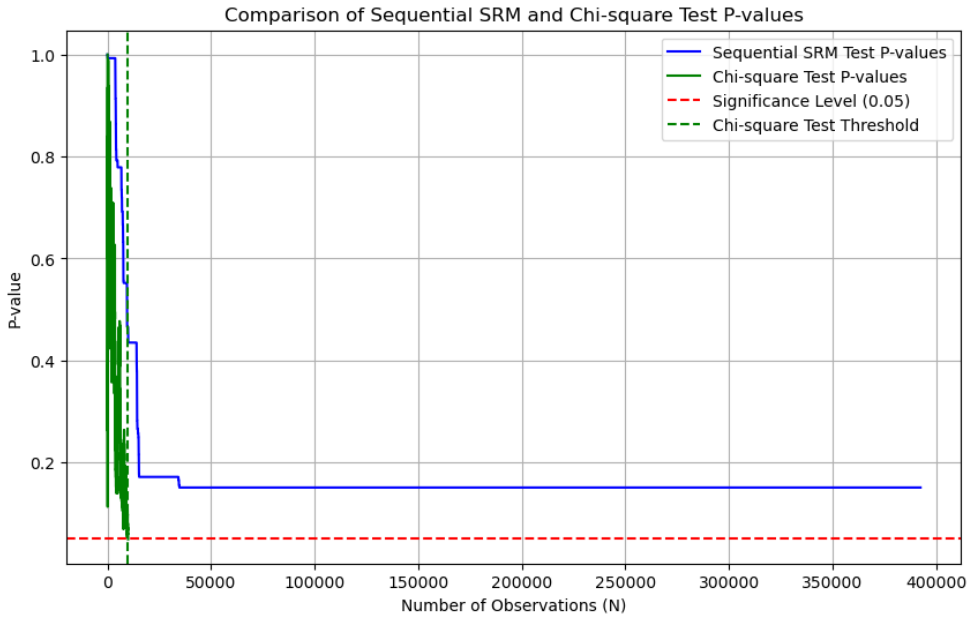

Figure 6: Comparison Between p-values from SSRM Test and Chi-Square Test for Supplementary Dataset.

The traditional Chi-square test is typically used after the completion of an experiment to evaluate the SRM. This post-hoc application makes it less suitable for real-time data monitoring, as it cannot provide timely feedback on emerging imbalances. In contrast, the SSRM test is designed for sequential analysis, allowing for continuous monitoring of data as it is collected. This capability makes the SSRM test more practical for real-time applications, enabling immediate detection and correction of SRMs during the experiment rather than after its completion. Overall, these results highlight the limitations of the Chi-square test in large-scale A/B testing scenarios where false positives can lead to erroneous conclusions. The SSRM test, with its sequential nature and robustness to early false detections, proves to be a more effective tool for maintaining the integrity of A/B testing processes. This comparison underscores the importance of selecting appropriate statistical tests based on the specific needs and design of the experiment, particularly in dynamic and high-stakes environments like e-commerce and fintech.

5.2 Anomaly Detection

5.2.1 Experiment 1: Benchmark Data

We have used five different quality control benchmarking datasets for change point detection. We have shown one of the quality control datasets with a known change point at time index 97. The data has constant Gaussian (0, 1) noise throughout, with a step change of size 1.5. It exemplifies the kind of datasets used in simulation studies of CP algorithms.

Graphs generated from comparing these algorithms using the datasets are shown below in Figures 7, 8, 9, 10. The green dotted line in each graph represents the true changepoint, while

red dotted lines indicate detected changepoints.

Here is a sample of the one of this dataset in Table 4:

Time	Value
0	-1.2250
1	-0.9146
2	-0.5910
3	0.2982
4	0.7070

Table 4: Data Sample for Change Point Detection.

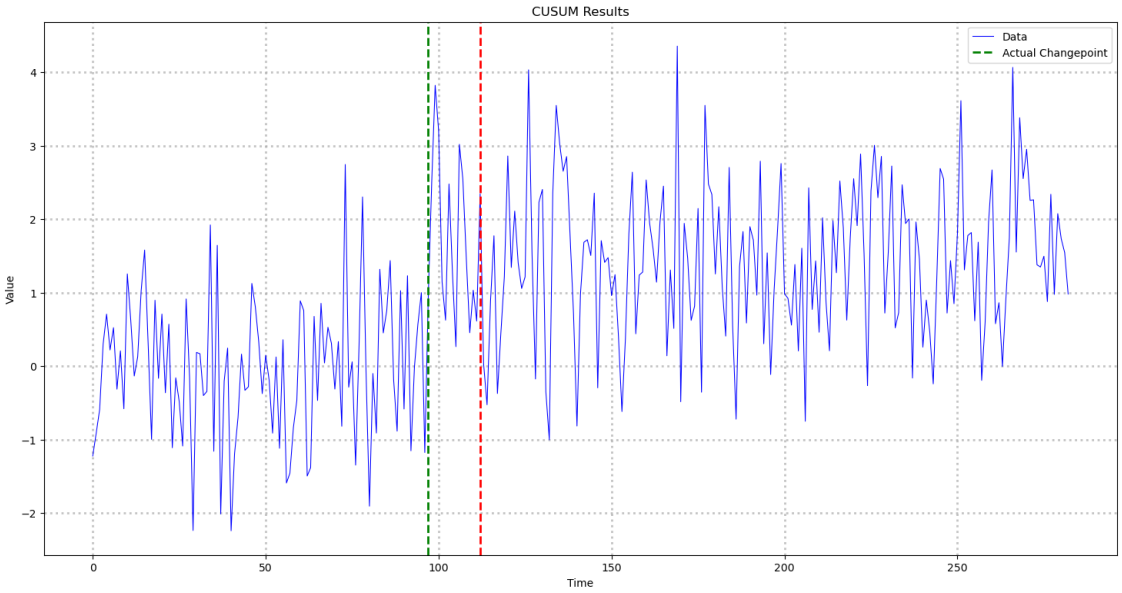

Figure 7: CUSUM Detected Change Points.

From the graph it can be seen that CUSUM effectively detected small shifts in the mean, showing high sensitivity and real-time detection capabilities. It produced fewer false positives and quickly identified changes. CUSUM effectively detected small shifts in the mean, showing high sensitivity and real-time detection capabilities. It produced fewer false positives and quickly identified changes (complexity $\mathcal{O}(n)$). On the other hand, PELT demonstrated good accuracy in detecting changepoints but tended to produce more false positives compared to CUSUM. PELT identified changes accurately but was sometimes slower due to the computational overhead of exact search, especially in real-time scenarios. PELT minimizes a penalized cost function, ensuring an optimal segmentation but at the expense of higher computational complexity $\mathcal{O}(n \log n)$. CUSUM and PELT showed higher sensitivity, with PELT detecting additional change points not present in the true signal, indicating potential false positives.

Dynamic Programming finds the optimal set of changepoints by solving subproblems and building up the solution. It ensures globally optimal detection but is computationally

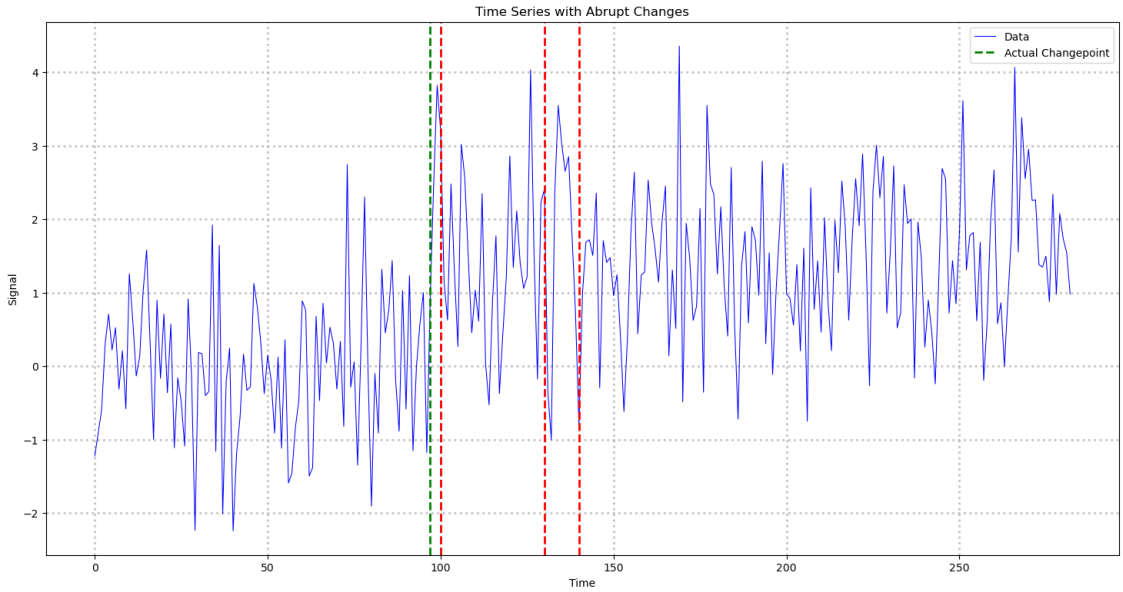

Figure 8: PELT Detected Dhange Points.

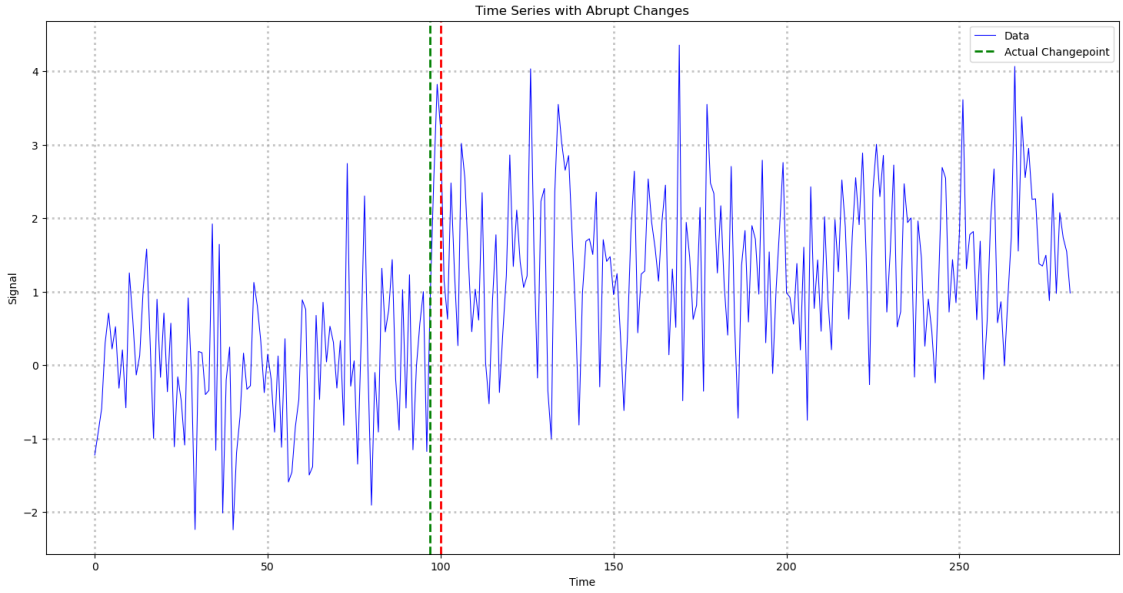

Figure 9: Dynamic Programming Detected Change Points.

intensive (complexity $\mathcal{O}(n^2) - \mathcal{O}(n^3)$). Dynamic Programming provided high accuracy but was computationally expensive, making it less suitable for real-time applications. Dynamic Programming guarantees the optimal segmentation by considering all possible solutions, which is mathematically rigorous but computationally expensive. Binary Segmentation on the other hand is a recursive method that identifies change points by splitting the data and applying the change detection method to each segment. Binary Segmentation was faster than dynamic programming (complexity $\mathcal{O}(n \log n)$) but less accurate in detecting smaller shifts, and it can be less accurate in detecting multiple change points compared to PELT, due to its greedy nature.

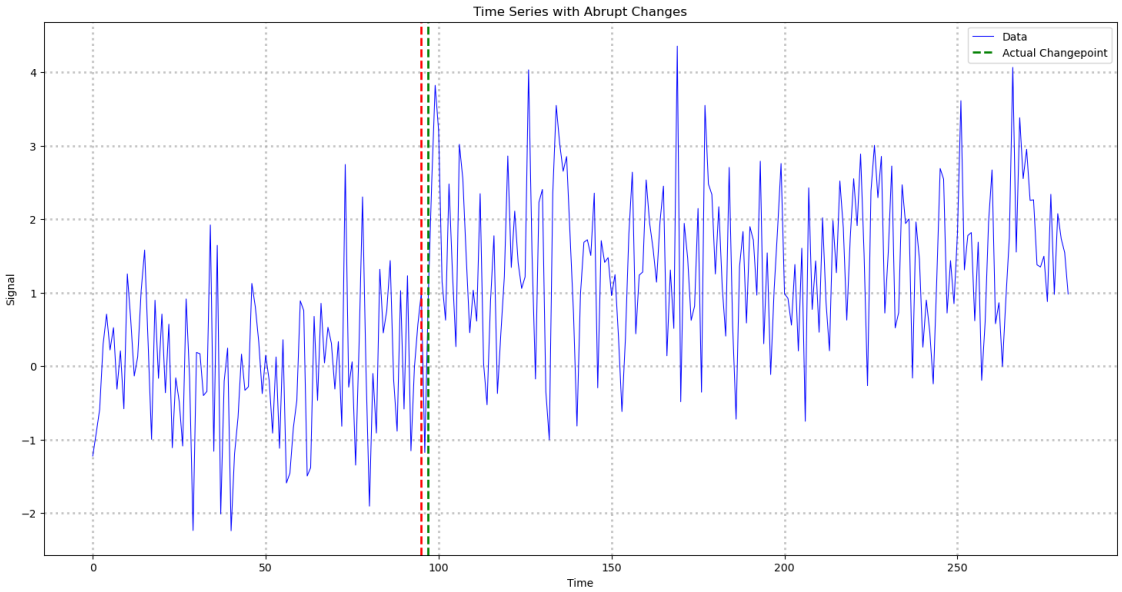

Figure 10: Binary Segmentation Detected Change Points.

5.2.2 Experiment 2: Simulated Data

We simulated a signal using the Ruptures library in Python, setting the number of samples to 1000 and setting predefined change points and 3 timestamps. We then ran 50 iterations of each of the four algorithms namely CUSUM, PELT, Dynamic Programming and Binary Segmentation on 50 simulated datasets with the same parameters. We evaluated their performance by taking an average of the True positives, true negatives, precision, recall, F1 score, and time taken. Here is a brief description of each performance metric used:

- **True Positives (TP)**: Correctly identified change points, the larger this value is the better our model is performing.
- **False Positives (FP)**: Incorrectly identified change points, we .
- **False Negatives (FN)**: Missed change points.
- **Precision**: The proportion of true positives among all detected change points.

$$\text{Precision} = \frac{TP}{TP + FP}$$

- **Recall**: The proportion of true positives among all actual change points.

$$\text{Recall} = \frac{TP}{TP + FN}$$

- **F1 Score**: Harmonic mean of precision and recall, balancing both metrics.

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

- **Mean Absolute Error (MAE)**: The average absolute difference between detected ($\hat{y}_i$) and true change (y_i) points.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i|$$

- **Average Time (s)**: Average time taken per iteration.

We have aggregated the average results of 50 iterations in Table 5:

Method	True Positives	False Positives	False Negatives	Precision	Recall	F1 Score	Mean Absolute Error	Average Time (s)
CUSUM	2.80	4.00	2.20	0.4190	0.5600	0.4763	22.9263	0.2609
PELT	5.16	22.28	0.04	0.1933	0.9927	0.3218	47.1897	0.2314
Dynp	2.94	0.06	2.06	0.9800	0.5880	0.7350	2.6400	0.4070
Binseg	4.00	0.00	1.00	1.0000	0.8000	0.8889	1.3900	0.2395

Table 5: Average Metrics for Change Point Detection Methods.

It can be observed from the table, that CUSUM shows moderate recall (0.56) and efficiency (0.26 s) but suffers from low precision and high MAE, leading to many false positives and less accurate change point locations. PELT has the highest recall (0.99) but extremely low precision (0.19) and high MAE (47.19), making it unsuitable due to the high number of false positives and inaccuracies. Dynamic Programming (Dynp) provides a good balance of high precision (0.98) and recall (0.59) with low MAE (2.64), but it is quite slower compared to others (0.40 s). Binary segmentation (Binseg) demonstrates excellent performance with perfect precision, high recall (0.8), highest F1 score (0.88), and lowest MAE (1.4), along with good efficiency (0.24 s).

In theory, in terms of accuracy and efficiency, binary segmentation seems like the superior choice. However, in real-world applications many practical factors need to be considered. In terms of algorithms simplicity, ease of implementation, and integeration into existing systems, CUSUM surpasses binary segmentation. It doesn't require complex parameter tuning beyond setting the threshold. Implementing CUSUM is straightforward, with minimal computational requirements and a clear step-by-step process. It's a well-documented algorithm with many resources available. So while CUSUM may not perform the best in all metrics, its simplicity and ease of implementation make it a strong candidate for practical use in many scenarios, particularly when ease of understanding and rapid deployment are prioritized.

We also want to consider applicability for real-time applications. CUSUM has a constant time complexity, each update is processed in $\mathcal{O}(1)$, ensuring minimal computational overhead. It has low latency, and its immediate detection capabilities are critical for applications requiring prompt responses. On the other hand, traditional binary segmentation has a complexity of $\mathcal{O}(n \log n)$, which can be less efficient for real-time processing compared to CUSUM's $\mathcal{O}(1)$. This complexity arises because binary segmentation typically processes the entire dataset to identify change points. Binary segmentation is not inherently designed for incremental updates, which can be a limitation for real-time applications. The need to

periodically re-segment the data can introduce latency makes binary segmentation slower to react to changes compared to CUSUM.

However, based on our experiments, binary segmentation demonstrated a lower average processing time per iteration compared to CUSUM, with binary segmentation taking 0.239517 seconds and CUSUM taking 0.260908 seconds on average. This practical observation, while seemingly counterintuitive given the theoretical complexities, can be attributed to several factors:

- Data Segment Size: Binary segmentation may benefit from processing relatively small segments efficiently, minimizing the impact of its $\mathcal{O}(n \log n)$ complexity.
- Implementation Optimizations: The specific implementation of binary segmentation in the ruptures library includes optimizations that enhance its practical performance.
- Experimental Conditions: The controlled environment and fixed dataset size used in the experiments may not fully capture the dynamic nature of real-time data streams.

Despite these observations, the constant time complexity ($\mathcal{O}(1)$ per update) of CUSUM offers significant theoretical advantages for real-time applications. CUSUM's simplicity, low latency, and predictable performance make it inherently well-suited for continuous monitoring and immediate response scenarios. This is crucial in fintech applications where data points arrive rapidly and decisions must be made in real-time.

Therefore, while binary segmentation shows promising performance in our experiments, CUSUM remains a robust choice for real-time anomaly detection due to its theoretical guarantees and proven reliability in handling streaming data.

5.3 Ratio OECs

In order to account for ratio OECs in Wise's **`ExperimentDataset`** class we added an optional argument ratio targets that accepts a Python dictionary as input. The user can then specify the name of the linearized OEC as the key and provide the numerator (X) and denominator (Y) as a tuple of column names. We also implemented this new feature in the streamlit app's user interface. As shown on Figure 11, the user can specify that they want to include ratio metrics and use the graphical UI to specify the ratio OEC. The underlying data model then takes care of the linearization automatically and the newly created column can be treated as an ordinary continuous column (together with sensitivity-enhancing techniques).

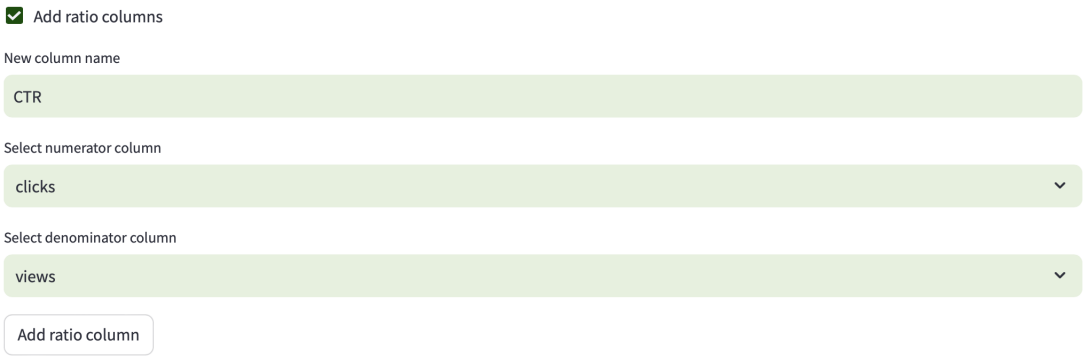

Figure 11: Graphical UI for ratio OECs.

In order to verify the consistency of the linearized OECs we have generated a thousand synthetic A/A tests and calculated the p-values for the following methods:

- bootstrap with $B = 1000$ (the 'correct' p-values and our baseline),
- naive transformation,
- linearization.

By plotting the p-values of the latter two methods against the p-values of bootstrap we can investigate whether the linearized metric behaves similarly to the true underlying OEC. Indeed, the p-values obtained from linearization are almost identical to the ground truth.

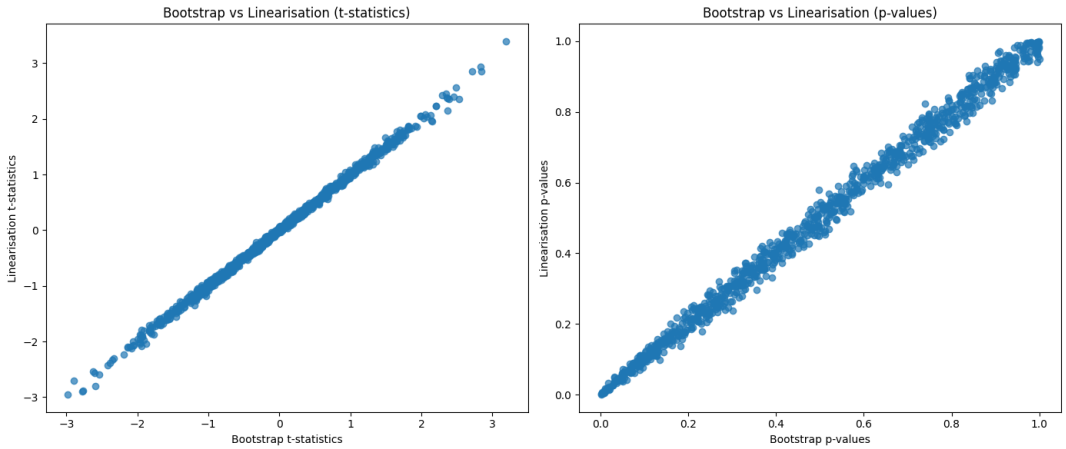

Figure 12: Comparison of t-statistics and p-values from 1000 A/A tests obtained via linearization against bootstrap.

The same cannot be said about the naive transformation's p-values, which show fewer similarities. Many points lie on or very close to the 45-degree line, but their dispersion is much larger than previously.

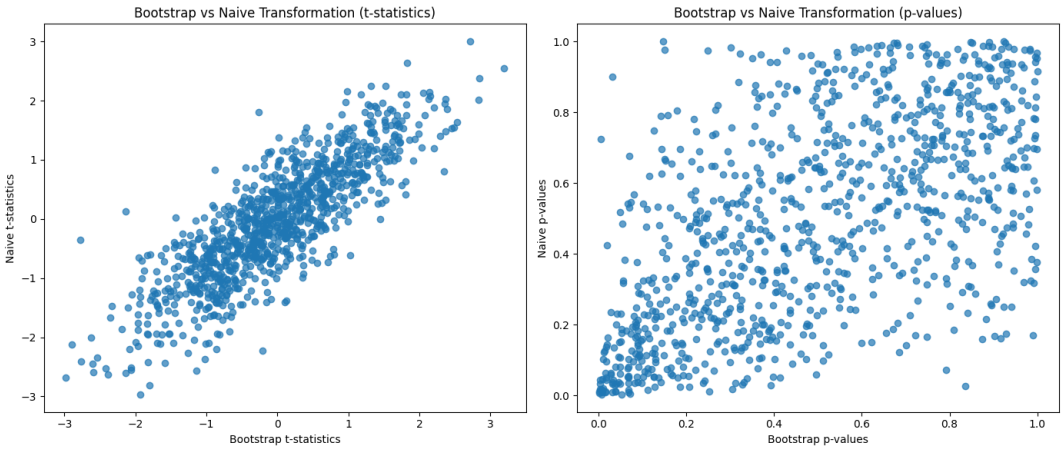

Figure 13: Comparison of t-statistics and p-values from 1000 A/A tests obtained via naive transformation against bootstrap.

Therefore, we can confirm that the theoretical guarantees from Section 4.3. Linearization is a method that is consistent, efficient, and can be further improved via sophisticated sensitivity-improving techniques. Our findings are summarised in table 6.

Method	**Consistent**	**Efficient**	**Sensitivity Improvement Techniques**
Naive	X	✓	✓
Bootstrap	✓	X	✓
Delta Method	✓	✓	Only simple ones
Linearization	✓	✓	✓

Table 6: Comparison of different methods.

6 Conclusion and Potential Future Work

The analysis of Sample Ratio Mismatch (SRM), anomaly detection, and Ratio OECs in the context of A/B testing for fintech applications has yielded several significant findings. These findings have important implications for designing and conducting robust and reliable experiments. This conclusion synthesizes the major insights from the research, ties them together, and suggests potential extensions for future work.

The SRM section highlights the importance of detecting SRM to ensure the validity of A/B tests. Methods like the Sequential SRM (SSRM) test were emphasized for their ability to provide real-time monitoring and early detection, thus preventing the invalidation of experiments due to sample imbalances. The need for continuous validation of randomization and assignment processes was underlined to avoid SRM. Techniques like the Chi-square test and its limitations were discussed, leading to the recommendation of more robust sequential methods like the SSRM test.

In the Anomaly Detection section, the CUSUM algorithm was identified as a robust method for real-time anomaly detection due to its simplicity, constant time complexity, and immediate response capabilities. The Probabilistic CUSUM variant of CUSUM enhances robustness by accounting for the distribution of data, making it more resilient to outliers and non-stationary conditions. Its probabilistic interpretation and dynamic thresholding improve its adaptability in dynamic environments. Despite some limitations in precision, its ability to promptly detect changes in streaming data makes it particularly suitable for fintech applications where quick decision-making is crucial.

In Ratio OECs, the study proposed and validated the linearization method for ratio OECs, demonstrating its consistency and efficiency in handling user-level metrics. This method addresses the challenges posed by the ratio nature of some metrics and ensures accurate statistical inference. The linearization approach was compared with bootstrap and delta methods, showing superior performance in terms of consistency and computational efficiency. This method allows for better sensitivity and accuracy in detecting true effects in A/B tests.

The integration of these findings provides a comprehensive framework for improving the reliability and accuracy of A/B testing in fintech applications. Anomaly detection, particularly with CUSUM and its probabilistic variant, ensures timely identification of significant changes, safeguarding the user experience and operational integrity. Continuous monitoring for SRM is crucial to maintain the validity of test results, while the linearization of ratio OECs enhances the robustness of statistical analyses.

Future work could explore developing more advanced methods for real-time SRM detection and correction can further ensure the integrity of online experiments. For Anomaly detection, it could also experiment with hybrid approaches that combine the strengths of different change point detection algorithms to enhance accuracy and computational efficiency. Investigating and implementing more sophisticated sensitivity-improving techniques for ratio

OECs could also be visited leading to even more reliable results.

In conclusion, the methodologies and findings discussed in this report provide a robust foundation for conducting reliable A/B tests in fintech applications. By leveraging advanced anomaly detection techniques, continuous SRM monitoring, and innovative approaches to ratio OECs, the proposed solutions enhance the validity and accuracy of experimental results. Future research should continue to build on these foundations, exploring new methods and technologies to further improve the robustness of online experimentation.

References

P. Armitage, C. K. McPherson, and B. C. Rowe. Repeated Significance Tests on Accumulating Data. *Journal of the Royal Statistical Society. Series A (General)*, 132(2):235–244, 1969. URL http://www.jstor.org/stable/2343787.

E. Bakshy and D. Eckles. Uncertainty in Online Experiments with Dependent Data: An Evaluation of Bootstrap Methods. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2013. URL http://dx.doi.org/10.1145/2487575.2488218.

R. Budylin, A. Drutsa, I. Katsev, and V. Tsoy. Consistent Transformation of Ratio Metrics for Efficient Online Controlled Experiments. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, page 55–63. Association for Computing Machinery, 2018. URL https://doi.org/10.1145/3159652.3159699.

Nanyu Chen, Min Liu, and Ya Xu. Automatic Detection and Diagnosis of Biased Online Experiments. Technical report, 2018. URL https://api.semanticscholar.org/CorpusID:88523885.

T. Crook, B. Frasca, R. Kohavi, and R. Longbotham. Seven Pitfalls to Avoid when Running Controlled Experiments on the web. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 1105–1114. Association for Computing Machinery, 2009. URL https://doi.org/10.1145/1557019.1557139.

A. Deng, Y. Xu, R. Kohavi, and T. Walker. Improving the Sensitivity of Online Controlled Experiments by Utilizing pre-experiment Data. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, page 123–132. Association for Computing Machinery, 2013. URL https://doi.org/10.1145/2433396.2433413.

A. Deng, J. Lu, and J. Litz. Trustworthy Analysis of Online A/B Tests: Pitfalls, Challenges and Solutions. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, page 641–649. Association for Computing Machinery, 2017. URL https://doi.org/10.1145/3018661.3018677.

P. Dmitriev, S. Gupta, D. Woo Kim, and G. Vaz. A Dirty Dozen: Twelve Common Metric Interpretation Pitfalls in Online Controlled Experiments. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 1427–1436. Association for Computing Machinery, 2017. URL https://doi.org/10.1145/3097983.3098024.

A. Drutsa, A. Ufliand, and G. Gusev. Practical aspects of sensitivity in online experimentation with user engagement metrics. 2015.

M. Esteller-Cucala, V. Fernandez, and D. Villuendas. Experimentation Pitfalls to Avoid in A/B Testing for Online Personalization. In *Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization*, page 153–159. Association for Computing Machinery, 2019. URL https://doi.org/10.1145/3314183.3323853.

A. Fabijan, J. Gupchup, S. Gupta, J. Omhover, W. Qin, L. Vermeer, and P. Dmitriev. Diagnosing Sample Ratio Mismatch in Online Controlled Experiments: A Taxonomy and Rules of Thumb for Practitioners. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, page 2156–2164. Association for Computing Machinery, 2019. URL https://doi.org/10.1145/3292500.3330722.

A. Fabijan, T. Blanarik, M. Caughron, K. Chen, R. Zhang, A. Gustafson, V. Kavitha Budumuri, and S. Hunt. Diagnosing Sample Ratio Mismatch in A/B Testing - Microsoft Research, 2020. URL https://www.microsoft.com/en-us/research/group/experimentation-platform-exp/articles/diagnosing-sample-ratio-mismatch-in-a-b-testing/.

P. Granjon. The CUSUM Algorithm - A Small Review. *HAL Archives*, 2014. URL https://hal.science/hal-00914697.

A. Hern. Why Google has 200M reasons to put Engineers over Designers. *The Guardian*, 2014.

R. Killick, P. Fearnhead, and I. A Eckley. Optimal Detection of Changepoints with a Linear Computational Cost. *Journal of the American Statistical Association*, 107(500):1590–1598, 2012.

R. Kohavi and S. Thomke. The Surprising Power of Online Experiments. *Harvard Business Review*, 95(5):74–82, 2017.

R. Kohavi, A. Deng, R. Longbotham, and Y. Xu. Seven Rules of Thumb for Web Site Experimenters. 2014.

R. Kohavi, D. Tang, and Y. Xu. *Trustworthy Online Controlled Experiments: A Practical Guide to A/B Testing*. Cambridge University Press, 1st edition, 2020.

N. Larsen, J. Stallrich, S. Sengupta, A. Deng, R. Kohavi, and N. T. Stevens. Statistical Challenges in Online Controlled Experiments: A Review of A/B Testing Methodology. *The American Statistician*, 78(2):135–149, 2024. URL https://doi.org/10.1080/00031305.2023.2257237.

M. Lindon and A. Malek. Anytime-Valid Inference For Multinomial Count Data. In *Neural Information Processing Systems*, 2020. URL https://api.semanticscholar.org/CorpusID:249191266.

J. Liu, D. Yang, K. Zhang, H. Gao, and J. Li. Anomaly and Change Point Detection for time series with Concept Drift. *World Wide Web*, 26:1–24, 2023.

S. Seitz. Probabilistic CUSUM for Change point Detection, 2022. URL https://sarem-seitz.com/posts/probabilistic-cusum-for-change-point-detection/.

D. T. Shipmon, J. M. Gurevitch, P. M. Piselli, and S. Edwards. Time Series Anomaly Detection: Detection of Anomalous Drops with Limited Features and Sparse Examples in Noisy Highly Periodic Data. *arXiv preprint arXiv:1708.03665*, 2017.

L. Vermeer, K. Anderson, and M. Acebal. Automated Sample Ratio Mismatch (SRM) Detection and Analysis. In *Proceedings of the 26th International Conference on Evaluation and Assessment in Software Engineering*, page 268–269. Association for Computing Machinery, 2022. URL https://doi.org/10.1145/3530019.3534982.