UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

LUAN MARTINS FELIX

ENTENDENDO E EVITANDO O SLIPPAGE PROBLEM NA WEB3.0

RIO DE JANEIRO
2023

LUAN MARTINS FELIX

ENTENDENDO E EVITANDO O SLIPPAGE PROBLEM NA WEB3.0

Trabalho de conclusão de curso de graduação apresentado ao Instituto de Computação da Universidade Federal do Rio de Janeiro como parte dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Daniel Sadoc Menasché

RIO DE JANEIRO
2023

CIP - Catalogação na Publicação

F316e

Felix, Luan Martin
Entendendo e evitando o slippage problem na Web 3.0 / Luan Martin Felix. -- Rio de Janeiro, 2023.
32 f.

Orientador: Daniel Sadoc Menasché.
Trabalho de conclusão de curso (graduação) - Universidade Federal do Rio de Janeiro, Instituto de Computação, Bacharel em Ciência da Computação, 2023.

1. Slippage. 2. Estatística. 3. Web 3.0. 4. Algoritmo. I. Menasché, Daniel Sadoc, orient. II. Título.

Elaborado pelo Sistema de Geração Automática da UFRJ com os dados fornecidos pelo(a) autor(a), sob a responsabilidade de Miguel Romeu Amorim Neto - CRB-7/6283.

LUAN MARTINS FELIX

ENTENDENDO E EVITANDO O SLIPPAGE PROBLEM NA WEB3.0

Trabalho de conclusão de curso de graduação apresentado ao Instituto de Computação da Universidade Federal do Rio de Janeiro como parte dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Aprovado em 5 de SETEMBRO de 2023

BANCA EXAMINADORA:

Prof. Daniel Sadoc Menasché (IC/UFRJ)

Prof. Rodrigo de Souza Couto (Programa Eng. Elétrica/COPPE/UFRJ)

Prof. João Ismael D. Pinheiro (Instituto de Matemática/UFRJ)

Prof. Vinicius Gusmão Pereira de Sá (IC/UFRJ)

À minha família, especialmente minha mãe, Leila, e irmão, Lucca, que sempre me educaram e incentivaram a concluir minha graduação.

À minha namorada, Camila, e a todos com quem compartilhei minha jornada na UFRJ.

AGRADECIMENTOS

Agradeço ao Henrique Caixeiro, graduado na UFRJ e meu colega de trabalho da Parfin, que me propôs o problema deste artigo. Agradeço também à Parfin, por apoiar e colaborar com o estudo. Agradeço ao meu orientador do TCC, Prof. Daniel Sadoc, pela mentoria nesse projeto, bem como ao Prof. João Paixão, que nos ajudou a modelar o problema.

Agradeço ao meu orientador da graduação, Hugo Nóbrega, por me guiar no desafiador caminho que é o ensino superior. Agradeço também a todos os professores com quem tive o prazer de dividir experiências ao longo da minha trajetória.

"Lost Time is never found again."

Benjamin Franklin

RESUMO

Estratégias de compra e venda de ativos são cada vez mais influenciadas por algoritmos. Tais algoritmos precisam levar em conta diversos fatores, sendo a volatilidade do mercado um dos principais elementos. Além disso, a precisão na avaliação da volatilidade é vital, pois permite uma adaptação mais ágil e informada, proporcionando vantagens competitivas aos investidores que adotam essas abordagens algorítmicas. A interseção entre finanças e tecnologia nunca foi tão evidente quanto agora, destacando a necessidade contínua de desenvolver algoritmos capazes de traduzir o mercado em decisões financeiras bem fundamentadas.

O trabalho apresentado descreve um estudo estatístico para um algoritmo que soluciona o problema de slippage, no contexto de um serviço (RFQ) oferecido por uma empresa (Parfin). Slippage é o termo dado à diferença do preço esperado e o preço executado de uma cota de compra ou venda de uma quantia em um par de moedas qualquer. A solução considerada é aumentar o preço esperado em uma margem (spread) que cubra o potencial prejuízo causado pelo slippage, bem como cancelar cotas que causem muito prejuízo. A solução é construída na coleta e processamento de dados para inferir, estatisticamente, o spread ideal que não perca um preço competitivo e não cause prejuízos à empresa. Os resultados foram a base crucial para o algoritmo desejado, capaz de, ao vivo, calcular o spread baseado nas últimas duas horas de dados do par de moedas em questão, e cancelar cotas que causem muito prejuízo, sem cancelar mais do que 1% das cotas, de acordo com as leis de SLA.

Palavras-chave: Slippage; estatística; Web3.0; algoritmo.

ABSTRACT

Strategies for buying and selling assets are increasingly influenced by algorithms. Such algorithms need to take into account several factors, with market volatility being one of the main elements. Furthermore, precision in assessing volatility is crucial, as it allows for a more agile and informed adaptation, providing competitive advantages to investors who adopt these algorithmic approaches. The intersection between finance and technology has never been as evident as it is now, underscoring the ongoing need to develop algorithms capable of translating the market into well-founded financial decisions.
The work presented here describes a statistic study for an algorithm that solves the slippage problem, in the context of a service (RFQ) offered by a company (Parfin). Slippage is the term given to the difference between expected price and executed price of a bid or ask quote of an amount in a given pair of currencies. The solution considered is to raise the expected price by a margin (spread) that covers the potential loss caused by the slippage, and also to cancel quotes that cause a big loss. The solution is constructed by collecting and processing data to infer, statistically, the ideal spread that does not give up a competitive price and does not cause loss to the company. The results were the crucial base for the intended algorithm, which is capable of live calculating the spread based on the last two hours of data in a given pair of currencies, and cancelling quotes that cause big losses, without cancelling more than 1% of the quotes, following the SLA rules.

Keywords: Slippage; statistics; Web3.0; algorithm.

SUMÁRIO

1 INTRODUÇÃO

Nessa tese, consideramos um problema do mercado de trabalho, nos colocando na perspectiva da Parfin (Parfin, 2023). Essa escolha claramente não é coincidência; O autor, na metade da sua graduação, começou um estágio na empresa, onde tem a oportunidade de aplicar seus conhecimentos teóricos e revertê-los em produtos diferenciais do mercado. A Parfin busca fornecer a adoção da *Web 3.0* (o mundo das criptomoedas e economia descentralizada) para seus clientes *fiat* (o mundo das moedas da economia tradicional). A diferença entre fiat e crypto pode ser melhor entendida em (Max Crawford, Hiro, 2023). Um dos principais serviços oferecidos pela Parfin é fornecer para seus clientes (empresas) um mecanismo de venda de criptomoedas para seus usuários (os clientes das empresas). Nós adotaremos o termo "usuários" para nos referir aos clientes das empresas, e o termo "clientes" para nos referir às empresas, que são clientes da Parfin. Dessa forma, essa tese se propõe a manter esse paradigma: Aplicar os conhecimentos teóricos da academia na construção da solução de um problema prático do mercado de trabalho.

O serviço que dá origem ao problema que estudaremos se chama *Request for Quote* (pedido de cotação), ou *RFQ*, e trata-se de um modelo de mercado, aplicado no contexto da Web3.0 (Kat Fore, 2023). As cotações são preços dados para um determinado par de moedas em um determinado tempo. O par de moedas BTC (Bitcoin) e USD (Dólar americano), representado por BTC/USD, estará presente ao longo de todo o trabalho. O lado de compra do par significa comprar BTC usando USD, e o lado de venda significa vender BTC recebendo em troca USD. Outros modelos de mercado para compra ou venda de pares de moedas existem, como o chamado *Automated Market Makers* (criador de mercado automatizado), ou *AMM* (Cryptopedia Staff, 2023), assim como o modelo de *order book* (livro de compras), utilizados por *exchanges* (corretoras que disponibilizam trocas de criptomoedas por moedas fiat) mais centralizadas, que entenderemos melhor mais à frente, mas o serviço de nosso estudo é baseado no modelo RFQ.

O problema em questão é chamado de *slippage* (traduzido literalmente como escorregão), retratado aqui como o *slippage problem*. Ele acontece quando há uma variação no preço da moeda entre o momento em que é feito um pedido de uma operação de troca de moedas, e o momento em que essa operação é de fato executada (Anton Boykov, 2023). Esse problema é de grande interesse econômico, já que lidar com ele significa economizar dinheiro evitando prejuízos. Em (ALDRIDGE, 2022) e (LIM, 2022) encontramos estudos sobre slippage, mas direcionados ao modelo AMM. (GRAMLICH; ANGERER; HANKE, 2021) analisa a liquidez de diferentes exchanges, usando o slippage como sua principal medida. Mais sobre exchanges e liquidez pode ser encontrado em (Rashi Maheshwari, 2023) e (Benjamin Crowell, 2023), respectivamente.

1.1 CALCULANDO O PREÇO DE UMA CRIPTOMOEDA

As exchanges são instituições que atuam como uma ponte entre o mundo fiat e o mundo Web 3.0. As mais centralizadas funcionam com base em um livro de ordens, que registra todas as ordens de compra e ordens de venda para as criptomoedas que elas oferecem. Vamos considerar o par BTC/USD. Uma ordem de compra é uma oferta proposta por alguém para comprar uma determinada quantia de BTC por um determinado preço em USD. A ordem de venda é análoga.

Aplicar ofertas de compra ou ofertas de venda não é a única opção. As ordens de mercado são ordens para comprar ou vender a criptomoeda ao preço atual de mercado. O preço de mercado é o melhor preço disponível no livro de ordens. Repare que sempre é possível aceitar ofertas piores (comprar por mais ou vender por menos), mas não há sentido lógico para isso. Assim, para vender (ao contrário de propor uma oferta de venda), deve-se aceitar uma oferta de compra (logicamente, a maior), e a compra é análoga.

Quantidade	Preço
0.150000 BTC	26345 USD/BTC
0.350000 BTC	26346 USD/BTC
1.200000 BTC	26347 USD/BTC

Tabela 1 - Maiores ordens de venda do par BTC/USD

Vamos ilustrar essas definições com um exemplo. Vamos comprar 1 BTC usando o par BTC/USD. Vamos supor que as ordens de venda do livro de ordens de uma determinada exchange sejam representadas pela Tabela 1. Então, para comprar 1 BTC, sequencialmente vamos selecionar as melhores ofertas até atingirmos a nossa quantidade desejada. Assim, ficamos com $0.15 \cdot 26345 + 0.35 \cdot 26346 + 0.5 \cdot 26347 = 26346.35$ USD. Repare que na última oferta só pegamos 0.5 BTC dos 1.2 BTC disponíveis, porque era o necessário para completar nossa compra.

1.2 ENTENDENDO O SLIPPAGE PROBLEM

O RFQ oferece cotações em certos intervalos de tempo para os usuários. Consideraremos que esse intervalo de tempo é de 5 segundos. Dessa forma, a cotação oferece, para algum lado de algum par de moedas, um preço que fica fixo durante a janela de tempo após ser oferecido, com o usuário podendo aceitar a cotação a qualquer momento nessa janela. Após o fim da janela, caso a cotação não seja aceita, ou após a cotação ser aceita, uma nova cotação será gerada. O RFQ calcula o preço com base nos preços de diferentes exchanges, mas neste trabalho focaremos nos preços da Bitstamp, uma exchange que

disponibiliza diversos dados de forma grátis que nos auxiliarão na pesquisa (Bitstamp, 2023a).

Vamos considerar um cenário de compra de 1 BTC do par de moedas BTC/USD. O usuário pede por uma cotação para comprar 1 BTC. Em um tempo t_1, o RFQ consulta o preço na exchange, P_1, e mostra para o usuário. Contudo, ainda dentro da janela de tempo de 5 segundos, ele leva algum tempo para considerar e aceitar a cotação. Vamos dizer que ele aceita a cotação em um tempo t_2. O preço da cotação, dado ao usuário, ainda é o mesmo. Contudo, devido à natureza volátil das criptomoedas, os preços nas exchanges costumam variar rapidamente. Dessa forma, no momento em que o RFQ de fato executa a operação, t_2, o preço na exchange variou para P_2.

Usuário	RFQ
Pede uma cota de compra de 1 BTC	Consulta o preço na exchange em tempo t_1 e devolve o preço P_1
Pondera sobre a oferta e aceita em tempo t_2	Executa a operação na exchange em tempo t_2 pelo preço P_2

Tabela 2 - Cenário de Compra de 1 BTC no RFQ

O preço P_1 é chamado de preço esperado (porque o cliente espera por esse preço), e por isso o nomearemos P_{esp}, e o P_2, preço executado (porque é o preço executado na exchange), e o chamaremos de P_{exec}. A diferença entre o preço esperado e o preço executado, $(P_{\text{esp}} - P_{\text{exec}})$, é o slippage. O slippage é positivo quando $P_{\text{esp}} > P_{\text{exec}}$, ou seja, o preço esperado foi maior que o preço executado. No caso de uma compra, isso significa que o usuário pagou mais do que o RFQ pagou à exchange, dando lucro à Parfin. O slippage é negativo quando $P_{\text{esp}} < P_{\text{exec}}$, ou seja, o preço esperado foi menor que o preço executado. No caso de uma compra, isso significa que o usuário pagou menos do que o RFQ pagou à exchange, dando prejuízo à Parfin. Repare que no caso de venda, o oposto é verdade: Para slippage positivo, o usuário vendeu por mais do que o RFQ vendeu à exchange, dando prejuízo à Parfin, e para slippage negativo, o usuário vende por menos do que o RFQ vende à exchange, dando lucro à Parfin. A Tabela 2 resume o cenário que observamos, e a Tabela 3 classifica em lucro ou prejuízo as diferentes possibilidades de slippage nos casos de compra e venda.

1.3 EVITANDO O SLIPPAGE PROBLEM

Vamos começar mergulhando mais a fundo na Tabela 2. Para isso, vamos observar a Figura 1. Nesse cenário mais detalhado, conseguimos ver o prejuízo do RFQ: O usuário pede uma cotação, o RFQ consulta a exchange, que retorna o preço de 30000 USD (P_{esp}) para 1 BTC, que por sua vez repassa o preço para o usuário de 30000 USD. Em seguida,

Lado	Slippage	Resultado
Compra	Positivo ($P_{\text{esp}} \geq P_{\text{exec}}$)	Lucro
Compra	Negativo ($P_{\text{esp}} < P_{\text{exec}}$)	Prejuízo
Venda	Positivo ($P_{\text{esp}} \geq P_{\text{exec}}$)	Prejuízo
Venda	Negativo ($P_{\text{esp}} < P_{\text{exec}}$)	Lucro

Tabela 3 - Classificação das Slippages

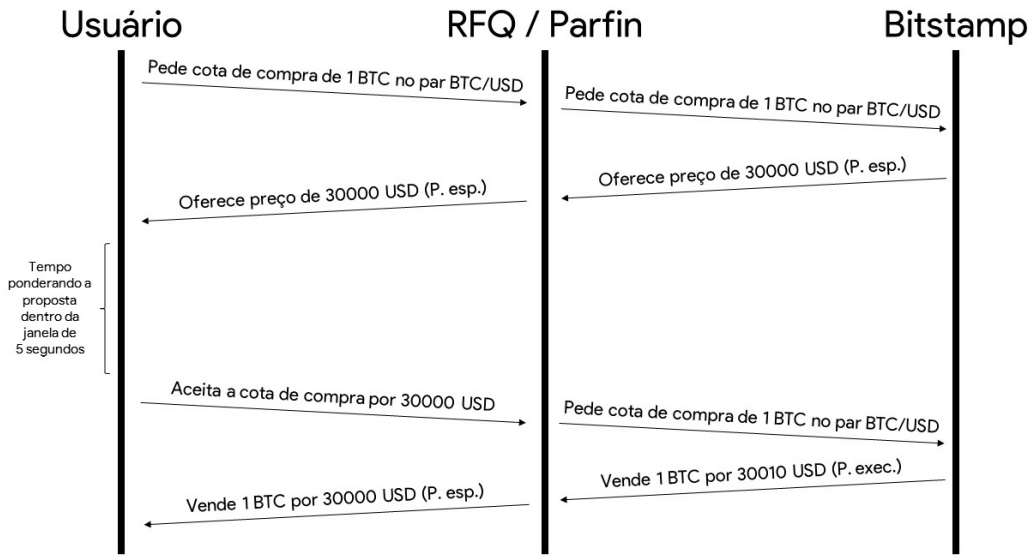

Figura 1 – Cenário de Compra de 1 BTC no RFQ detalhado

o usuário pondera e aceita o preço, o RFQ prossegue para comprar 1 BTC na exchange, que agora mudou de preço para 30010 USD (P_{exec}), e então vende o BTC por 30000 USD para o usuário (P_{esp}). Nesse caso, o RFQ perde 10 USD devido à flutuação do BTC entre o P_{esp} e o P_{exec}. Em outras palavras, o RFQ perde 10 USD devido ao slippage negativo de 10 USD.

Uma solução ingênua é, nos cenários de prejuízo, cancelar a cota. Isso pode ser visualizado na Figura 2. Embora isso resolva o problema, há um impacto negativo na experiência e satisfação do cliente. Basta imaginar um cenário, por exemplo, em que três cotas consecutivas pedidas e aceitas pelo usuário sejam recusadas, uma atrás da outra, pelo mesmo motivo. Para evitar esse problema, nós consideraremos a lei de *Service Level Agreement* (acordo a nível de serviço), ou *SLA*. Nela, estabelecemos que o RFQ possui o direito de cancelar cotas, mas não mais que 1% delas. Dessa forma, a experiência do usuário é satisfeita, mas o RFQ consegue evitar os cenários de maior prejuízo.

Outra solução é implementar uma variação no preço esperado P_{esp}, que chamaremos de *spread*. A ideia do spread é modificar o preço esperado, mostrado para o usuário nas

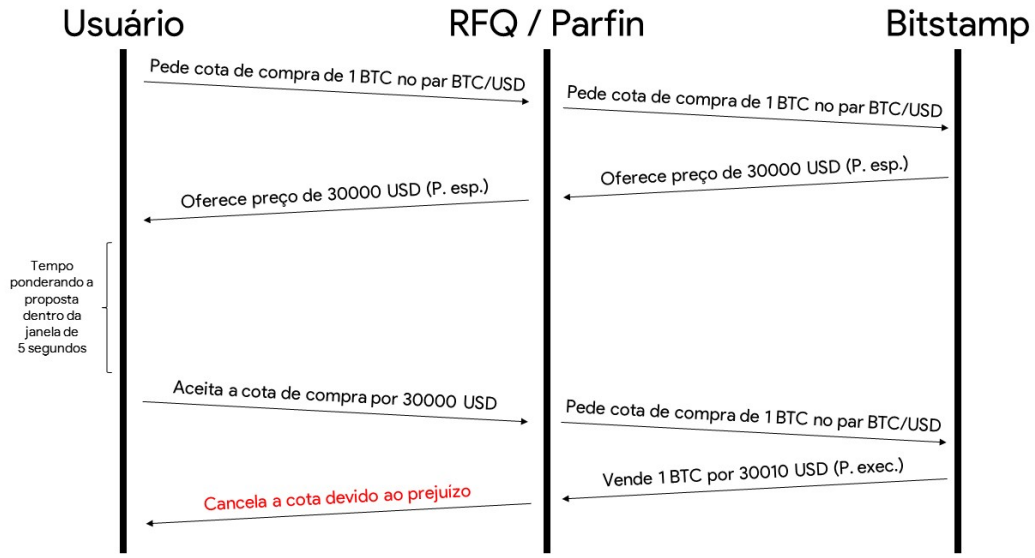

Figura 2 – Cenário de Compra de 1 BTC no RFQ com cancelamento de cota

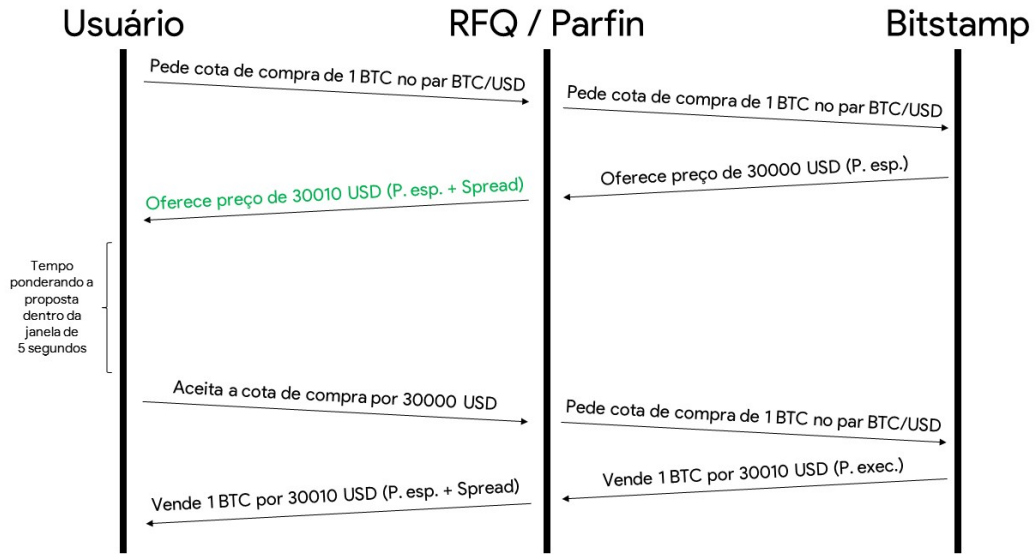

Figura 3 – Cenário de Compra de 1 BTC no RFQ com uso de spread

cotas, de forma a cobrir o slippage. Dessa forma, o usuário é quem arca com o possível prejuízo que o slippage pode causar, tirando essa responsabilidade do RFQ. Contudo, aumentar o valor de spread implica em um valor alto da cota, fazendo com que o RFQ perca a competitividade devido a um preço fraco quando comparado ao preço de mercado. Em contrapartida, um valor baixo de spread nos leva de volta ao problema original, tendo que lidar com os slippages. O conceito de spread pode ser visualizado na Figura 3.

Repare que nos cenários de compra, a estratégia é aumentar o valor da cota (fazer o usuário pagar mais) de forma que $P_{\text{esp}} + \text{spread} \geq P_{\text{exec}}$, resultando em lucro para o RFQ

(no caso de $P_{\text{esp}} + \text{spread} = P_{\text{exec}}$, o não prejuízo será considerado como lucro a longo prazo). Já para os cenários de venda, nós teremos que abaixar o valor da cota (fazer o usuário receber menos) de forma que $P_{\text{esp}} - \text{spread} < P_{\text{exec}}$, também resultando em lucro para o RFQ.

Nosso objetivo para evitar o slippage começa a ficar evidente: Queremos achar o spread ideal, levando em consideração a taxa de cancelamento máxima de 1%, o preço competitivo de mercado e a volatilidade da moeda (para entender como P_{exec} se comportará em relação a P_{esp} e então calcular o spread) mas que cubra os cenários de prejuízo para o RFQ. Portanto, a ideia é calcular um spread tal que em 99% dos casos, teremos lucro para o RFQ, e em 1% deles (os de maior prejuízo) as cotas serão canceladas.

É importante reforçar que outras estratégias são viáveis de forma a reduzir o prejuízo causado pelos slippages. Por exemplo, podemos abdicar parte do lucro para buscar um preço ainda mais competitivo com um spread menor (ou seja, arcando com alguns cenários de prejuízo, e cobrindo apenas, digamos, 70% dos prejuízos, e não 99%). Outra opção interessante seria aumentar o SLA de 1%, fazendo que seja possível cancelar mais cotas e contribuindo novamente pra um preço mais competitivo, sacrificando um pouco da experiência do usuário. Contudo, nesse estudo inicial, tal qual a janela de 5 segundos e o uso do cenário de compra e venda de 1 BTC, assumiremos o SLA de 1% e a estratégia de "zero prejuízo como premissas durante o estudo.

No capítulo 2, começaremos coletando os dados com os quais trabalharemos, para em seguida, no capítulo 3, processá-los e gerar as estatísticas que precisamos para encontrar a resposta que desejamos. O capítulo 4 elabora em cima das escolhas de amostras que devemos fazer para gerar um resultado consistente. Finalmente, o capítulo 5 conclui incluindo possíveis continuações para um trabalho futuro.

2 COLETANDO DADOS

Nosso primeiro passo em busca do spread ideal é coletar dados com os quais trabalharemos. Todas as exchanges costumam prover APIs para que seus usuários possam integrá-las aos seus sistemas, e, portanto, esses dados costumam ser gratuitos e de fácil acesso. Durante nossas pesquisas, como mencionado anteriormente, escolhemos a Bitstamp como nosso primeiro recurso. O motivo ficará claro com o decorrer do capítulo: A Bitstamp satisfaz todos os critérios que estabeleceremos na nossa coleta de dados.

A documentação da API da Bitstamp (Bitstamp, 2023b) contém informações diversas. Embora muitos endpoints estejam disponíveis, o que nos interessa é o livro de ordens. Felizmente, existe um endpoint do livro de ordens que nos retorna o livro completo, com todas as *bids* (ofertas de compra) e todas as *asks* (ofertas de venda) presentes no momento. O retorno para o par BTC/USD é apresentado no Código 1. As ofertas foram omitidas porque, para esse retorno, existem 10473 ofertas de compra e 15541 ofertas de venda.

Código 1 – Livro de Ordens

```
{
    "timestamp": "1688917128",
    "microtimestamp": "1688917128476792",
    "bids": [["30342","0.31934050"],["30341","0.37958590"], ...],
    "asks": [["30347","0.57666325"],["30348","0.05000000"], ...]
}
```

Vamos começar interpretando esse retorno. O campo *timestamp* nos mostra o momento, com precisão de segundos, em que a foto do livro de ordens foi tirada, ou seja, o estado do livro de ordens naquele exato segundo. Nesse caso, o valor é 1688917128, representando a data 9 de julho de 2023 às 12:38:48 (no fuso horário brasileiro, GMT-3). O campo *microtimestamp* é uma alternativa para essa data, mas com precisão de microssegundos. O campo bids mostra as ofertas de compra, ordenadas do maior valor para o menor valor, no formato de pares [Valor (em USD), Quantia (de BTC)]. O campo asks, analogamente, mostra as ofertas de venda ordenadas do menor valor para o maior valor.

Embora esses dados sejam úteis, existem dois problemas: uma quantia desnecessária de bids e asks (dado que vamos trabalhar com cenários de compra e venda de 1 BTC) e o timestamp, já que queremos comparar janelas de 5 segundos. Isso exigiria que fizéssemos duas chamadas à API com intervalo preciso de 5 segundos, existindo margem para erros, atrasos, etc. Como fica evidente pelo campo microtimestamp, a atualização do livro de ordens por vezes ocorre mais frequentemente do que a taxa de uma modificação por segundo.

Por isso, fomos motivados a buscar outro método de coleta de dados e encontramos em (Bitstamp, 2023c) um webapp que demonstra as 100 maiores ofertas de compra e venda

ao vivo, com todas as atualizações sendo instantaneamente refletidas. Isso nos leva a crer que a Bitstamp disponibiliza também um método de obter todas as atualizações do livro de ordens ao vivo, o que resolveria o nosso segundo problema, e com um simples corte dos dados nos livraríamos do primeiro problema também.

Não obstante, encontramos em (Bitstamp, 2023d) a documentação de uma API *WebSocket* da Bitstamp. Conforme descrito na própria página, o WebSocket é um protocolo que provê uma comunicação completa de mão dupla por meio de uma única conexão TCP. A página também afirma que a Bitstamp usa sua implementação do servidor WebSocket para compartilhamento de dados em tempo real. Com isso, podemos nos inscrever no canal do livro de ordens e obter atualizações em tempo real dele.

3 PROCESSANDO OS DADOS COLETADOS

Após coletados os dados, temos que processá-los para buscar o resultado que almejamos. Como já mencionado, trabalharemos com o cenário de compra e venda de 1 BTC do par BTC/USD. Dessa forma, como mostrado na introdução, temos que calcular o preço médio de compra e de venda para um dado livro de ordens. O Código 2 mostra essa operação, dado que o formato do livro de ordens é o mesmo do Código 1.

Código 2 – Calculando o Preço Médio

```
def calculate_average_price(offers, amount):
  total_price = 0
  current_amount = 0

  for offer in offers:
    offer_price = float(offer[0])
    offer_amount = float(offer[1])

    if current_amount + offer_amount >= AMOUNT:
      amount_left = AMOUNT - current_amount
      total_price += amount_left * offer_price
      break
    else:
      total_price += offer_price * offer_amount
      current_amount += offer_amount

  average_price = total_price/amount

  return average_price
```

Código 3 – Dados Processados

```
[
    {
      "timestamp": "1686859875",
      "microtimestamp": "1686859875734534",
      "average_bids_price": 25460.62790724,
      "average_asks_price": 25466.89550043
    },
    ...
]
```

Dessa forma, podemos nos concentrar apenas nos registros das microtimestamps asso-

ciadas aos preços médios de compra e de venda de 1 BTC. Também é importante notar que só precisamos registrar as mudanças que afetam esse preço médio, visto que o livro de ordens pode ser atualizado devido a ofertas de preços não relevantes para o cálculo de compra ou venda de 1 BTC. Um exemplo dos dados processados é representado no Código 3.

3.1 CALCULANDO OS SLIPPAGES

A próxima etapa é pegar os dados processados e calcular os slippages. Existem diversas maneiras com que podemos abordar esse cálculo. Por exemplo, se coletarmos 10 segundos de dados (digamos, de t_1 a t_{10}), podemos obter dois slippages (o slippage do intervalo $[t_0, t_5]$ e o do intervalo $[t_5, t_{10}]$), cinco slippages (nos intervalos $\big[[t_1, t_6], [t_2, t_7], \ldots, [t_5, t_{10}]\big]$), ou realisticamente quantos slippages quisermos, se considerarmos intervalos de tempo em microssegundos (digamos, da microtimestamp 1686859875734534 a 1686859875734539, da 1686859875734535 a 1686859875734540 e assim sucessivamente).

Claramente temos que fazer uma escolha, e a melhor maneira de realizá-la é pensar no nosso caso de uso. Embora o RFQ ofereça cotas de 5 em 5 segundos (conforme estamos considerando no artigo), diferentes usuários podem pedir cotas paralelamente e uma cota pode ser aceita logo após ser gerada, tendo que gerar outra em seguida. Dessa forma, considerar intervalos exclusivos de 5 segundos seria um desperdício de dados e estaria em desacordo com o nosso objetivo. Em contrapartida, intervalos de microssegundos são um exagero dado que teríamos muitos intervalos com slippages iguais, visto que a média de ofertas de compra e vendas colocadas no livro de ordens é perto de 1 oferta por segundo.

Assim, vamos escolher considerar o espaçamento entre os intervalos de um segundo. Repare que isso é quase equivalente a responder à pergunta: "Se a todo segundo um usuário pedisse uma cota para o RFQ, e a aceitasse, qual seria o slippage associado a essa cota?". Essa escolha é uma premissa para simplificar o processamento dos dados e o estudo, tal qual todas as outras que assumimos até aqui, e não necessariamente representa a completa realidade do RFQ, mas nos dá uma base pertinente.

A Figura 4 demonstra 15 minutos de dados convertidos em slippages. A melhor interpretação do gráfico é descrita como "Se o cliente pedisse uma cota (de 1 BTC do par BTC/USD) no tempo t, o gráfico mostra o slippage que o RFQ sofreria para aquela cota".

Vamos parar um momento para observar melhor a Figura 7. Primeiro, vemos que os picos de slippage, tanto positivo quanto negativo, estão na casa dos 10 a 15 USD. Comparado aos 30.000 USD (preço aproximado de 1 BTC no momento de escrita do artigo), pode não parecer um número expressivo, mas considerando que centenas de usuários transacionem dezenas de BTC por mês, estamos falando de uma possível economia (ou prejuízo) de 10 a 15 mil USD (no melhor ou pior cenário possível).

Outra observação é que, apesar dos cenários de compra e venda serem opostos em

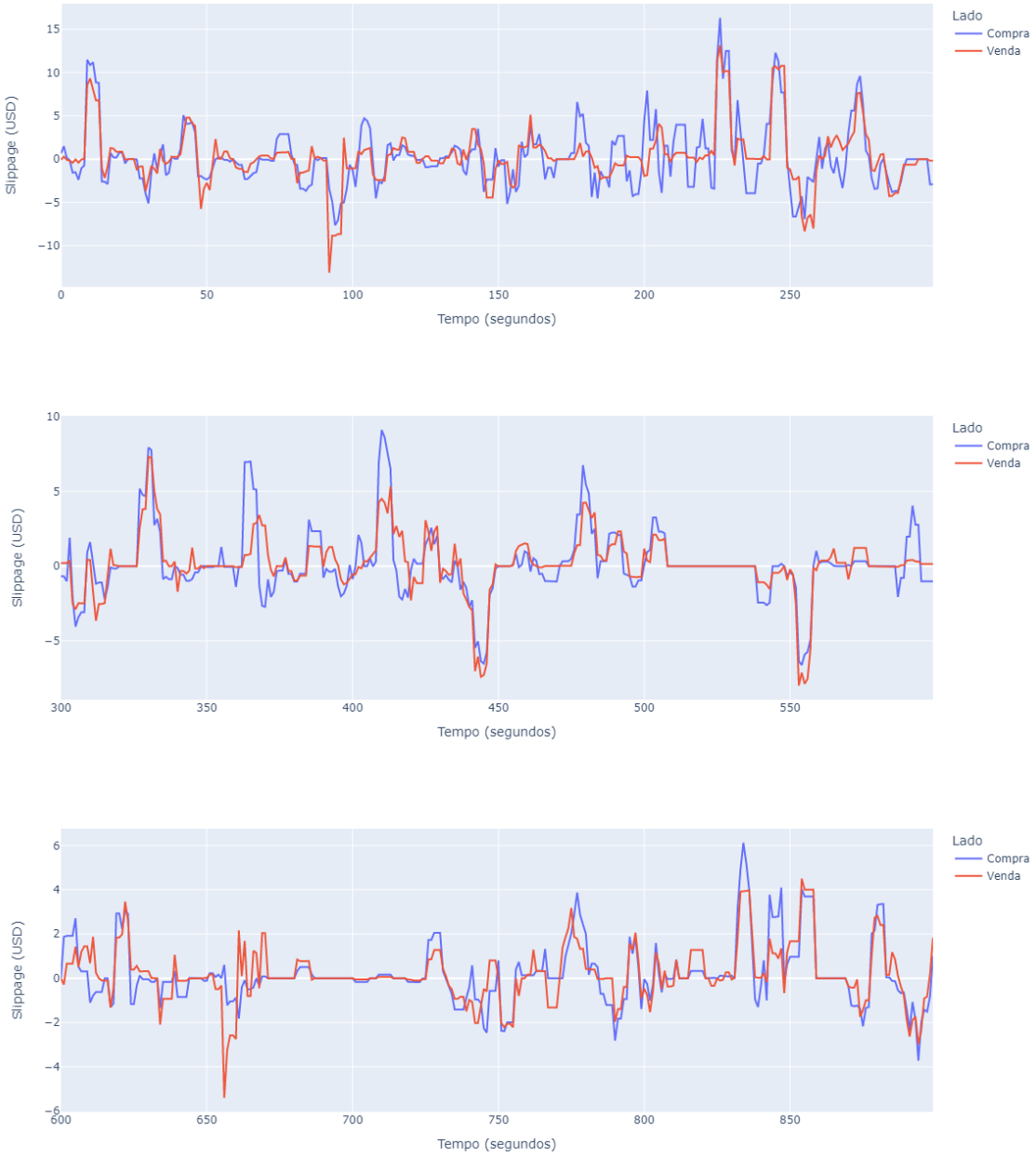

Figura 4 – Slippages ao longo de 15 minutos

relação aos slippages (como vimos na Tabela 3), os slippages costumam estar sempre no mesmo sentido (apesar de em valores diferentes) independente do lado em um determinado tempo. Por exemplo, no maior pico do primeiro gráfico, vemos que os picos de ambos os lados são parecidos; em contrapartida, no maior pico do segundo gráfico, o valor de compra é quase o dobro do valor de venda, mas é raro que um pico de um lado seja positivo e o do outro seja negativo, ou vice-versa.

Isso nos leva ao entendimento de que a qualquer momento t, ou ele é um bom momento para comprar, ou é um bom momento para vender, e nunca os dois simultaneamente. Essa

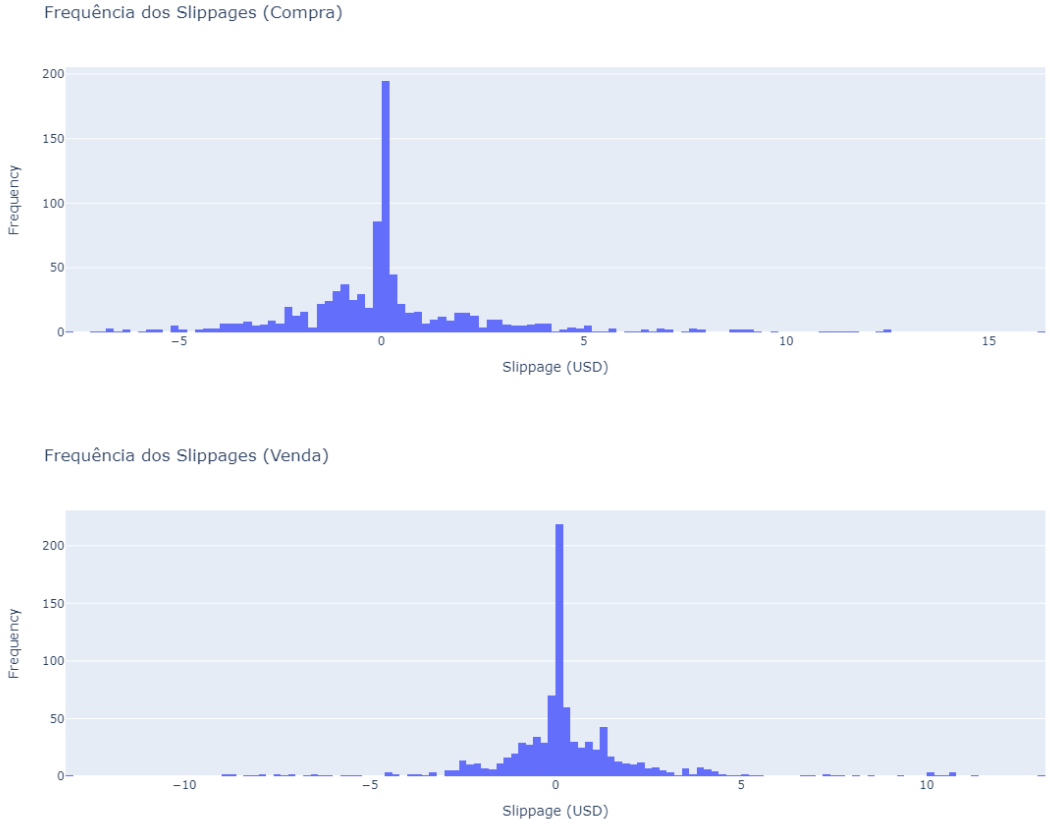

Figura 5 – Frequência dos Slippages de Compra e Venda ao longo de 15 minutos

relação antagônica nos traz a vantagem de que basta saber o slippage de um dos lados, e teremos uma boa noção do comportamento do outro. Todas essas observações foram feitas com base em uma pequena amostra de dados, portanto, servem apenas para criarmos intuições, e não para assumi-las como verdade absoluta. Continuaremos trabalhando com ambos os lados, mas sempre investigando e ponderando essas observações.

A Figura 8 nos traz a frequência dos slippages. Embora 15 minutos de dados sejam poucos, conseguimos observar que ambos os gráficos (tanto o lado de compra quanto o lado de venda) mostram semelhança com a curva normal e uma média muito próxima de 0. Junto dela, podemos observar os Códigos 4 e 5, que trazem todas as estatísticas do lado de compra e venda da mesma amostra, respectivamente.

Código 4 – Estatísticas dos Slippages de Compra ao longo de 15 minutos


```
{
    "confidence_interval": [
        -0.03645850061936415,
        0.413182033841588
    ],
    "confidence_interval_without_sample_size": [
        -6.55624625030317,
        6.932969783525394
    ],
    "mean": 0.18836176661111192,
    "sample_size": 900,
    "standard_deviation": 2.614189153842745,
    "variance": 6.833984932069046
}
```


Código 5 – Estatísticas dos Slippages de Venda ao longo de 15 minutos


```
{
    "confidence_interval": [
        0.0030131641462451675,
        0.3987534945648694
    ],
    "confidence_interval_without_sample_size": [
        -5.735221626923806,
        6.136988285634921
    ],
    "mean": 0.20088332935555728,
    "sample_size": 900,
    "standard_deviation": 2.300815874526885,
    "variance": 5.293753688474916
}
```


De todas as estatísticas, a mais importante é o intervalo de confiança. Como mencionamos ao restringir o problema, estamos buscando 99% de confiança. Portanto, usamos a tabela z avaliada em 2.58 (que em uma curva normalizada é equivalente a 99% da área do gráfico). O que o Código 4 nos traz é a primeira resposta após tantas perguntas: Durante a amostra de 15 minutos, se recusássemos os slippages de compra menores que -6.56 USD, bem como os slippages de venda maiores que 6.14 USD, estaríamos recusando apenas 1% das cotas dos usuários (assumindo que uma cota seja requisitada a todo segundo).

Repare também que o lado positivo do intervalo de confiança dos slippages de compra e o lado negativo dos slippages de venda são desconsiderados porque qualquer slippage positivo de compra e qualquer slippage negativo de venda resulta em um cenário lucrativo, e portanto não é algo com que temos que nos preocupar.

Uma observação crucial é que a quantidade de dados faz com que a confiança aumente

muito devido ao tamanho da amostra. Essa questão é discutida em (Stack Exchange, 2013), onde uma das respostas clarifica que dividir o intervalo de confiança pela raíz do tamanho da amostra é uma espécie de penalidade por usar uma amostra, e não todos os dados existentes. Contudo, em nosso contexto, não é nem possível imaginar a amostra completa, e esse modificador apenas faria com que o intervalo de confiança se aproximasse da média com o aumento do tamanho da amostra e em nada nos contribuiria na busca pelo spread ideal. Portanto, consideraremos o intervalo de confiança sem dividir pelo tamanho da amostra.

Por mais interessante que o resultado que temos seja, ele reflete apenas a verdade dessa amostra de 15 minutos, abrindo margem para muitas outras perguntas: O que acontece a longo prazo (amostras de tempo maiores)? Devemos considerar amostras de longo prazo? Se sim, o quão longo deve ser esse prazo? Desde o começo do dia, da semana, do mês, do ano, ou da história do Bitcoin? Se não, o quão curto deve ser o tempo que vamos considerar? O último minuto? Os últimos dez minutos? A última meia hora?

É importante mencionar que nosso objetivo final é fazer um estudo que consiga determinar o necessário para que o RFQ, automaticamente e algoritmicamente, preveja se o slippage com o qual se deparou é prejudicial o suficiente para que possa ser cancelado e representar apenas 1% dos cancelamentos. Ele consegue calcular o slippage, afinal possui o preço estimado e o preço executado; O que precisamos ensiná-lo (e primeiramente aprender) é a interpretar esse valor.

4 ANALISANDO DIFERENTES AMOSTRAS

O primeiro experimento que vamos realizar é coletar uma amostra maior. Existem diversas opções de volume de dados que podemos escolher: ano, mês, semana, etc. Contudo, optamos por trabalhar com dados recentes e, portanto, preferimos coletá-los ao vivo, como descrito no capítulo 2. Dessa forma, a primeira amostra que analisaremos será de 8 horas totais (que chamaremos de amostra A).

O gráfico de slippages ao longo do tempo, como vimos para a amostra de 15 minutos, é muito poluído para a amostra de 8 horas e, portanto, não nos traz nenhuma intuição. O nosso interesse está na Figura 6, que traz o histograma das frequências dos slippages em USD. Como é possível observar, o pico em torno de 0 é muito grande, dificultando a visualização do gráfico. Por isso, a Figura 7 traz os mesmos gráficos, mas com um zoom perto da origem.

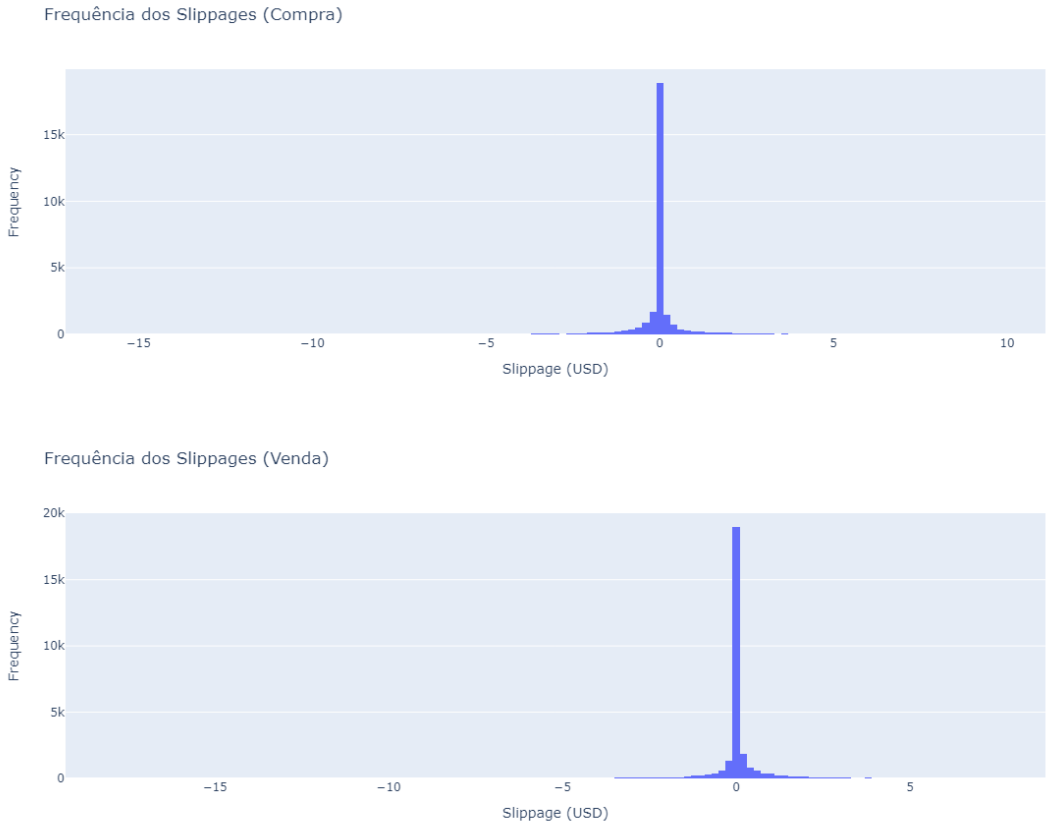

Figura 6 – Frequência dos Slippages de Compra e Venda ao longo de 8 horas

Nessa amostra maior, principalmente na Figura 7, fica ainda mais evidente a semelhança do gráfico com a curva normal. Em seguida, observamos os Códigos 6 e 7, que trazem as estatísticas desses slippages. A média é ainda mais próxima de 0, e o desvio

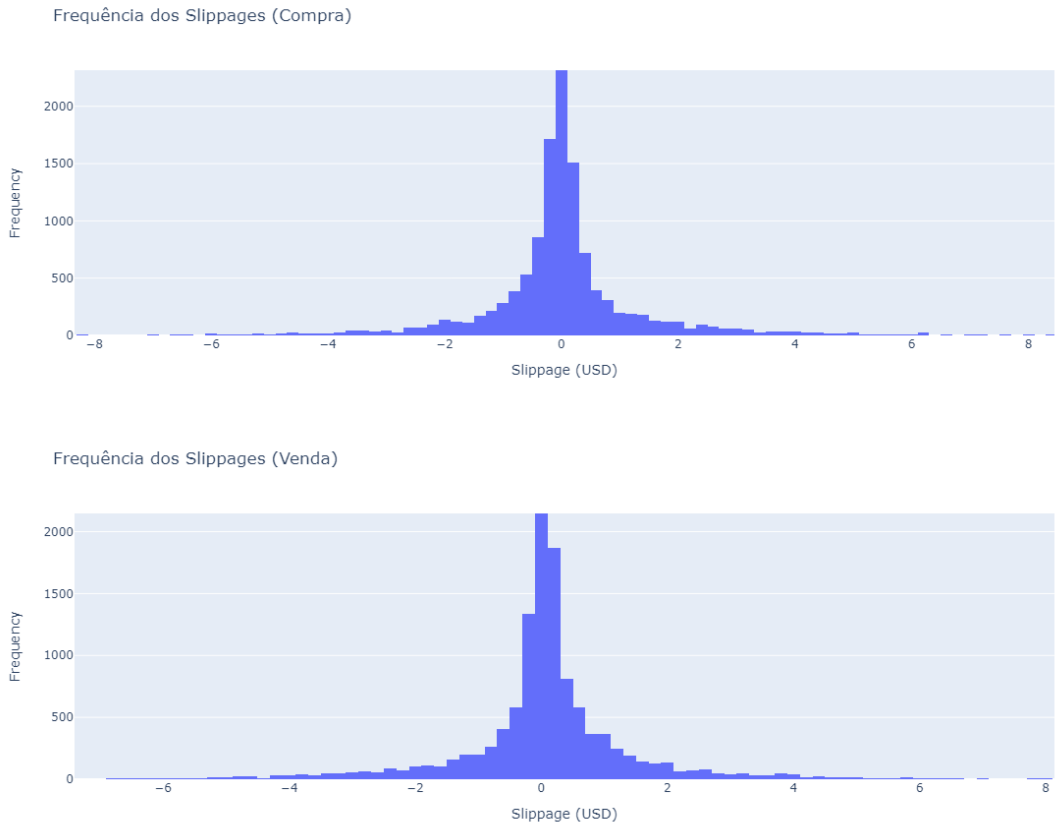

Figura 7 – Frequência dos Slippages de Compra e Venda ao longo de 8 horas em detalhe

padrão é próximo de 1. Isso nos leva à observação mais importante: a longo prazo, os slippages de fato tendem a uma curva normal.

Repare que essa intuição é, basicamente, mais uma premissa: A de que os slippages se comportam, a longo prazo, como uma curva normal. Diversos conceitos teóricos devem ser aplicados para justifcar isso: Hipóteses de normalidade, testes de aderência e até mesmo testes de outras possíveis curvas. Contudo, como reiteraremos na conclusão, isso será deixado para estudos futuros, visto que a natureza dos dados nos deixa confiante de que o comportamento normal seja razoável.

Assim, surge uma motivação para realizar um processo importante em cima dos dados sem distorcê-los tanto: a normalização. Vamos pegar todos os dados, subtraí-los pela média e dividir pelo desvio padrão. Dessa forma, o desvio padrão será 1 e a média será 0, nos levando à curva normal da tabela z. Finalmente, conseguimos descobrir exatamente onde cai o ponto de 99% nos dados, isto é, o equivalente a $z = 2.58$. Com isso, revertemos o processo e descobrimos o valor limite exato de slippage que estamos procurando.

Porém, precisamos considerar outro problema: a flutuação da moeda ao longo do tempo. Estamos calculando slippages diretamente em USD, mas não sabemos o quanto esse valor é significante em relação ao preço da moeda. O BTC vale hoje em torno dos 30.000 USD, mas um dia já valeu muito menos que uma caixa de pizza (CHOHAN,

2022). Para lidar com isso, vamos passar a tratar os slippages em porcentagens: calculado o slippage, dividimos pelo preço esperado P_{esp} e achamos a quantia.

Código 6 – Estatísticas dos Slippages de Compra ao longo de 8 horas


```
{
    "confidence_interval": [
        -0.01442270680674026,
        0.018613654449734812
    ],
    "confidence_interval_without_sample_size": [
        -2.800889388256002,
        2.805080335898997
    ],
    "mean": 0.0020954738214972756,
    "sample_size": 28795,
    "standard_deviation": 1.08642824111531,
    "variance": 1.1803263230929057
}
```


Código 7 – Estatísticas dos Slippages de Venda ao longo de 8 horas


```
{
    "confidence_interval": [
        -0.014490171366709548,
        0.018831083146532535
    ],
    "confidence_interval_without_sample_size": [
        -2.8249863021123964,
        2.8293272138922196
    ],
    "mean": 0.002170455889911493,
    "sample_size": 28795,
    "standard_deviation": 1.0957971930241504,
    "variance": 1.2007714882396072
}
```


Código 8 – Normalizando a Distribuição de Slippages

```
def standardize_slippages(slippages, statistics):
    mean = statistics['mean']
    sd = statistics['standard_deviation']
    standardized_slippages = [(x - mean)/sd for x in slippages]

    return standardized_slippages
```

Começaremos criando a função que normaliza as distribuições de slippages, visualizada no Código 8. Em seguida, consideraremos os slippages em porcentagens no lugar de em

valor absoluto (USD). Finalmente, conseguimos observar que as porcentagens são bem baixas (visto que os slippages de 0 a 20 dólares em relação ao preço de 1 BTC, 30000 USD, é uma fração da ordem de 10^{-1}% a 10^{-2}%).

O único detalhe que nos falta é determinar o tamanho da amostragem que vamos considerar. Como vimos, a amostra de 15 minutos possui um resultado bem diferente da amostra de 8 horas, que se aproximou muito de uma distribuição normalizada. Isso nos leva a inferir que 8 horas já é um intervalo de tempo grande demais; provavelmente queremos considerar apenas os últimos minutos da moeda para inferir o próximo slippage.

Nossa estratégia então será coletar dados, calcular os slippages (tanto em valor absoluto quanto em porcentagem) e então trabalhar com eles da seguinte forma: primeiro pegamos uma certa quantidade de dados, usamos os, digamos, 5 primeiros minutos para treinamento (calculando as estatísticas de slippage) e simulamos o RFQ aplicando nosso critério para decidir recusar ou aceitar o slippage (equivalente a uma cota) do próximo segundo. Anotamos esse resultado, e então, recalculamos as estatísticas removendo o primeiro slippage dos 5 minutos de dados de treinamento e adicionando o próximo, mantendo 5 minutos de dados totais. Em seguida, simulamos o RFQ novamente aplicando o critério de decisão de aceitação e assim sucessivamente até acabarem os dados.

Novamente, nosso objetivo é achar qual a melhor quantidade de dados para utilizar nos cálculos, de forma a usar o menor limite de slippage possível mantendo a taxa de rejeição de cotas perto dos 1%. A Figura 8 traz os gráficos com a abordagem descrita anteriormente feita para z = 2.58, usando a amostra de 8 horas, onde o eixo x é a quantidade de tempo utilizada no cálculo das estatísticas (treinamento) em minutos e o eixo y representa a taxa de rejeição de cotas, tanto para a compra como para a venda, da simulação do RFQ feita com o resto dos dados.

Conforme podemos observar, a função exibe um comportamento geralmente decrescente, e a partir de duas horas de dados usados para treino, começa a oscilar em torno de 1,5%. É importante destacar que esse resultado é meramente uma observação estatística, e não o resultado esperado de todo experimento. Temos que explorar mais amostras de dados para obter uma conclusão mais sólida.

Antes disso, podemos notar que todas as taxas estão acima de 1%. Isso é uma consequência de como a estatística frequentista funciona: Para esses dados, a rejeição foi um pouco maior que 1%, tal qual para outros dados poderia ser um pouco menor. Isso é equivalente ao motivo pelo qual jogar um dado justo seis vezes consecutivas não implica que cada número aparecerá uma vez. A estatística frequentista é também a base para algumas das conclusões e premissas que assumimos aqui (ISABELLA et al., 2022).

A Figura 9 exibem mais gráficos relacionados a uma nova amostra coletada, de 4 horas totais, que chamaremos de amostra B. É notável que a função também apresenta um comportamento decrescente nas primeiras duas horas de uso de dados para treinamento. Também observamos que a taxa de rejeição ficou abaixo de 1% com os 120 minutos de

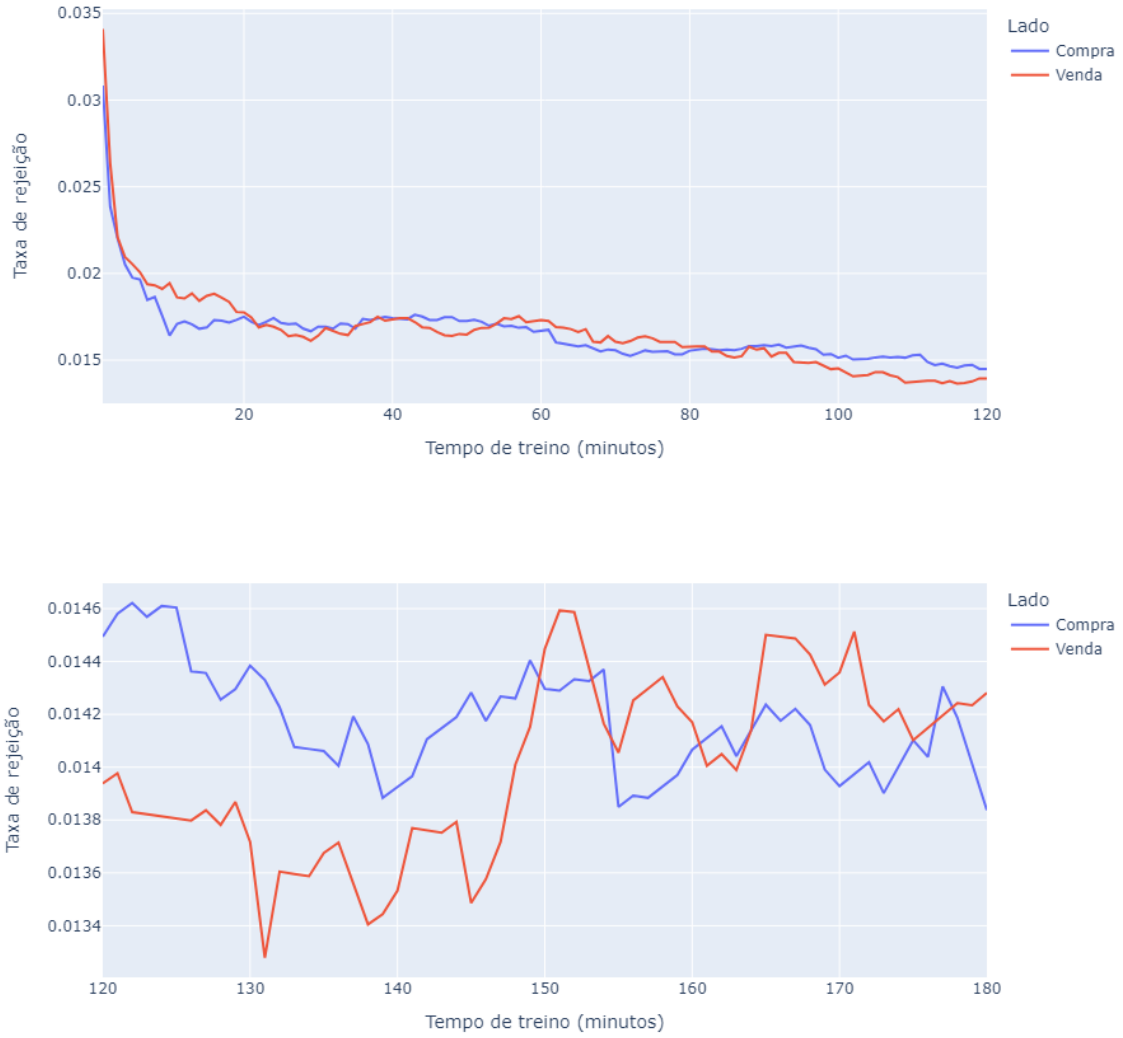

Figura 8 – Taxa de Rejeição das Cotas para diferentes tempos de treino (amostra A)

treino, em contraste com a amostra A.

Finalmente, para visualizarmos o valor de spread escolhido durante a simulação, basta revertermos o processo de normalização. Isso significa multiplicar o z escolhido, 2.58, pelo desvio padrão e somar a média (da estatística da distribuição antes de ser normalizada). É importante lembrar que esses valores mudam a cada segundo conforme um novo slippage é adicionado ao conjunto de treinamento, e um slippage antigo é retirado. As Figuras 10 e 11 mostram esses comportamentos para as amostras A e B, respectivamente. Com essa abordagem, conseguimos simular a execução do RFQ de forma a manter a taxa de cancelamento abaixo de 1% e, ao mesmo tempo, cancelar as cotas mais prejudiciais, atingindo nosso objetivo.

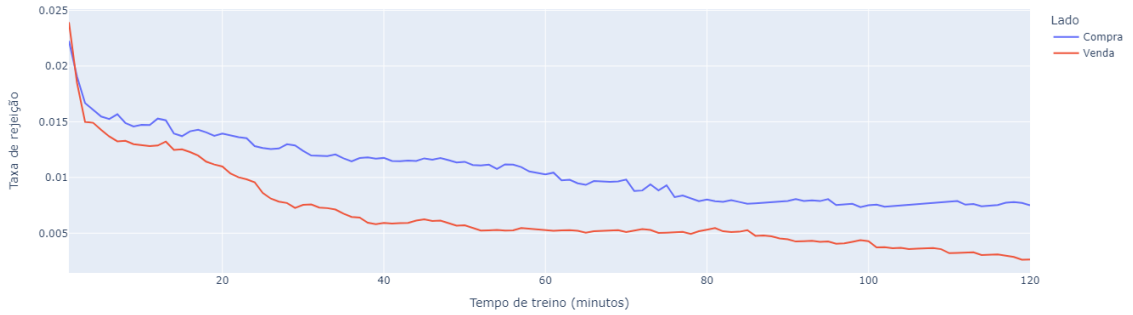

Figura 9 – Taxa de Rejeição das Cotas para diferentes tempos de treino (amostra B)

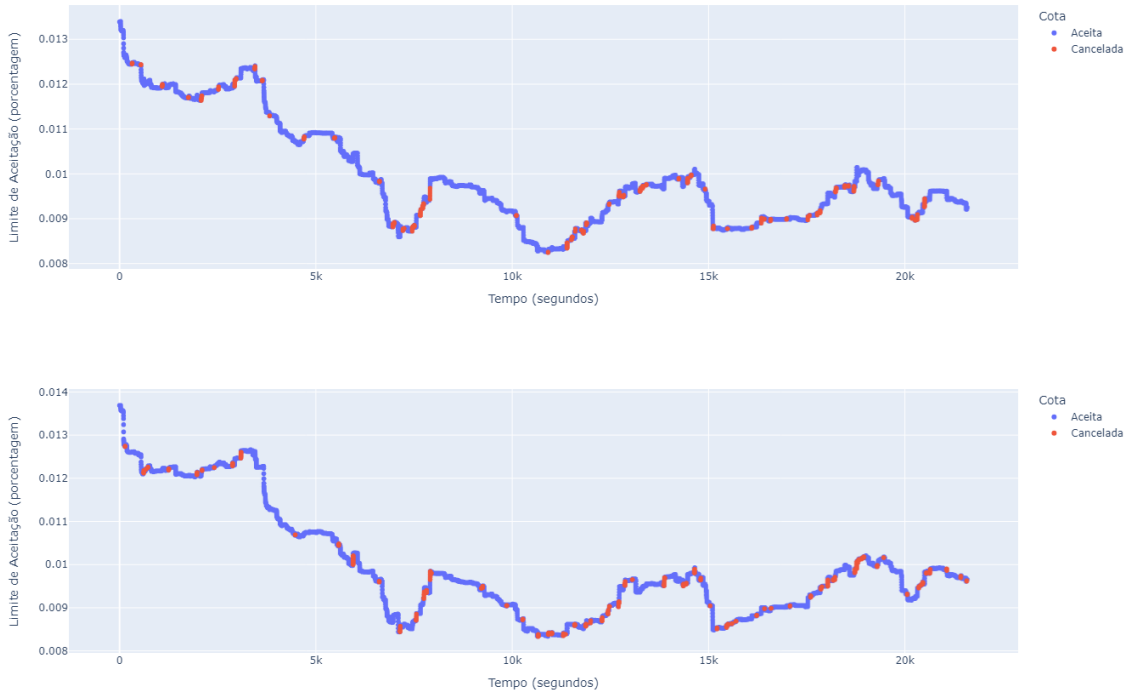

Figura 10 – Valor limite para aceitação de cotas durante a simulação do RFQ na amostra A usando 120 minutos de treino no lado de Compra e Venda, respectivamente

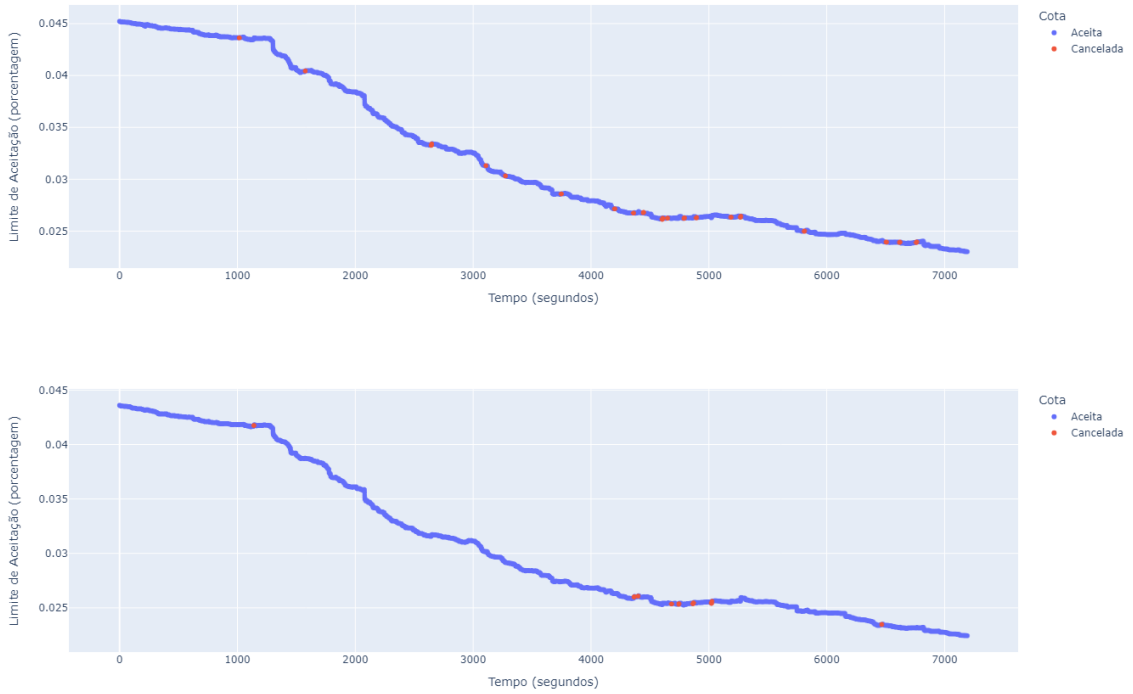

Figura 11 – Valor limite para aceitação de cotas durante a simulação do RFQ na amostra B usando 120 minutos de treino no lado de Compra e Venda, respectivamente

5 CONCLUSÃO

Após muita coleta e processamento de dados, conseguimos simular o papel do RFQ em diferentes cenários e analisar, baseado na estatística frequentista, como podemos criar um algoritmo para determinar, ao vivo, o spread ideal para uma cotação seguindo todas as premissas que assumimos. Contudo, é importante entender que justamente por conta de algumas dessas premissas, o estudo foi realizado com uma visão muito simplificada. Nós nos restringimos ao par BTC/USD e à compra e venda de 1 BTC, aos dados exclusivamente da Bitstamp e a uma janela de tempo de 5 segundos, ao SLA de 1% entre outras simplificações. Embora seja possível desenvolver um algoritmo com base no que foi apresentado no artigo, é necessário aumentar a robustez dele considerando diferentes cenários.

Os resultados obtidos foram satisfatórios e servem como uma base crucial para a expansão do trabalho. Trabalhos futuros podem utilizar os testes de aderëncia, de normalidade e experimentar outras curvas no gráfico para justificar a escolha da curva normal. Também é possível trabalhar com diversas modificações de variáveis e premissas, como diferentes SLAs, um paradigma de lucro diferente (como aceitar mais prejuízos para obter um preço mais competitivo), difererentes janelas de tempo, outros pares de moedas, outras quantidades de compra e venda e também diferentes exchanges.

REFERÊNCIAS

ALDRIDGE, I. Slippage in amm markets. **SSRN**, 2022.

Anton Boykov. **What Does Slippage Stand For in Crypto?** 2023. [Online; accessed 2-September-2023]. Disponível em: https://b2broker.com/news/what-does-slippage-stand-for-in-crypto/.

Benjamin Crowell. **Crypto Exchange Liquidity, Explained**. 2023. [Online; accessed 3-September-2023]. Disponível em: https://cointelegraph.com/explained/crypto-exchange-liquidity-and-why-it-matters-explained.

Bitstamp. **API**. 2023. [Online; accessed 30-August-2023]. Disponível em: https://www.bitstamp.net/.

Bitstamp. **API**. 2023. [Online; accessed 30-August-2023]. Disponível em: https://www.bitstamp.net/api/.

Bitstamp. **Webapp**. 2023. [Online; accessed 30-August-2023]. Disponível em: https://www.bitstamp.net/s/webapp/examples/order_book_v2.html.

Bitstamp. **Websocket**. 2023. [Online; accessed 30-August-2023]. Disponível em: https://www.bitstamp.net/websocket/v2/.

CHOHAN, U. W. A history of bitcoin. **SSRN**, 2022.

Cryptopedia Staff. **What Are Automated Market Makers?** 2023. [Online; accessed 2-September-2023]. Disponível em: https://www.gemini.com/cryptopedia/amm-what-are-automated-market-makers.

GRAMLICH, M.; ANGERER, M.; HANKE, M. Order book liquidity on crypto exchanges. **AWG2021**, 2021.

ISABELLA, F.-W. et al. Understanding the differences between bayesian and frequentist statistics. **Redjournal**, 2022.

Kat Fore. **Wtf is RFQ on-chain?** 2023. [Online; accessed 1-September-2023]. Disponível em: https://medium.com/bebop-dex/wtf-is-rfq-on-chain-19560e00058b.

LIM, T. Predictive crypto-asset automated market making architecture for decentralized finance using deep reinforcement learning. **arXiv**, 2022.

Max Crawford, Hiro. **Crypto vs. Fiat: Why Trustlessness Is a Competitive Advantage for Web3**. 2023. [Online; accessed 1-September-2023]. Disponível em: https://www.hiro.so/blog/crypto-vs-fiat-why-trustlessness-is-a-competitive-advantage-for-web3.

Parfin. **Parfin, The world on blockchain rails**. 2023. [Online; accessed 30-August-2023]. Disponível em: https://parfin.io/.

Rashi Maheshwari. **What Are Crypto Exchanges And How Do They Work**. 2023. [Online; accessed 2-September-2023]. Disponível em: https://www.forbes.com/advisor/in/investing/cryptocurrency/what-is-a-crypto-exchange/.

Stack Exchange. **Why is the formula for standard error the way it is?** 2013. [Online; accessed 3-September-2023]. Disponível em: https://stats.stackexchange.com/questions/60484/why-is-the-formula-for-standard-error-the-way-it-is#:~:text=By%20dividing%20by%20the%20square,of%20the%20%E2%80%9Cpenalty%E2%80%9D).