UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Kevin Gabriel Ramisch Pergher

Aplicações de Reservoir Computing e Sistemas Não-Lineares para Forecasting

Porto Alegre

2022

Kevin Gabriel Ramisch Pergher

Aplicações de Reservoir Computing e Sistemas Não-Lineares para Forecasting

Trabalho de Diplomação em Engenharia Física, realizado sob orientação do Prof. João Henrique Ferreira Flores e apresentado à Universidade Federal do Rio Grande do Sul como requisito parcial para a obtenção do título de Bacharel em Engenharia Física.

Orientador: Prof. João Henrique Ferreira Flores

Porto Alegre

2022

AGRADECIMENTOS

Aos meus pais, Yudi Cristina Ramisch e Ricardo Daguer Pergher, pelo apoio incondicional e presença constante bem como ao meu irmão João Miguel Ramisch Pergher pela companhia de cada dia.

Um agradecimento especial à todos os meus familiares que compartilharam destes anos comigo e sempre estiveram ao meu lado, com um especial agradecimento ao meu avô Roque Pergher (*in memoriam*) pelo tempo e dedicação empregados em prol da minha educação e estudos.

De maneira especial à todos os meus colegas de curso, de forma especial aos meus bons amigos do Instituto de Física com os quais vivi anos maravilhosos e espero ainda muitos outros anos. Também àqueles espalhados por toda Porto Alegre, Ivoti, Bento Gonçalves, Caxias do Sul, no resto do Brasil e no além-mar.

Ao meu orientador, professor João Henrique Ferreira Flores pela atenção especial que tem tido em cada etapa deste trabalho, possibilitando este trabalho de atingir níveis cada vez mais elevados de qualidade, bem como todo o grande conhecimento que nunca deixou de compartilhar na área.Igualmente dedicado ao professor Carlo Requião da Cunha que em muito incentivou e tornou possível este projeto, tendo sempre compartilhado do seu conhecimento e experiência, bem como por me haver introduzido ao brilhante campo da Econofísica e Física de Sistemas Econômicos.

Aos meus colegas da Fundamenta Investimentos, pela dedicação constante em entender cada vez mais o mercado financeiro e por todo o conhecimento compartilhado no mercado de fundos de investimentos, renda variável e macroeconomia.

"Qu'est-ce que le bonheur
sinon l'accord vrai entre un homme
et l'existence qu'il mène ?"
(Albert Camus)

RESUMO

Em meio a um cenário de forte competição impulsionado por tecnologias disruptivas, novos paradigmas técnicos e operacionais e o advento de aplicações em ciência de dados e inteligência artificial faz-se cada vez mais necessário o desenvolvimento de produtos sofisticados, fortemente embasados do ponto de vista teórico e científico e capazes de agregar valor a processos em empresas atuantes no mercado financeiro.

Com o desafio de alocação de capitais no mercado de ações se mostrando cada vez maior, com a utilização em larga escala de algoritmos de *trading* de alta frequência e análise em tempo real, otimização, eficiência e robustez são cada vez mais características imprescindíveis de qualquer produto que se proponha a ser competitivo e atrativo. Em decorrência desta situação e com consciência da capacidade do profissional de engenharia física para agregar valor a negócios aliando tecnologia com vasta formação teórica propusemos o desenvolvimento de um produto. Partindo do estado da arte e elaborando uma solução concisa e aplicável para a análise de *forecasting* de cotações de diversos ativos financeiros com o uso de conceitos sólidos da física de sistemas não-lineares e a implamentação de modelos de redes neurais.

O desenvolvimento deste projeto passa pela implementação de redes neurais do tipo *Echo State* baseadas em paradigmas de *Reservoir Computing* para redes neurais e a análise de sistemas não-lineares já bastante conhecidos na literatura. O desempenho da rede neural no que tange a sua capacidade de predição em séries temporais será avaliado com base nas medições do erro de predição dos modelos.

A ideia conceitual será ampliada visando a análise de sistemas caóticos sobre efeito de diversas fontes de ruído e analisando a consequência destes nas capacidades preditivas e na acurácia. Por fim, o sistema será aplicado para séries temporais de ativos financeiros de frequência diária, alta liquidez e interesse prático. O produto será então amplamente avaliado, documentado e seus resultados devidamente apresentados do ponto de vista técnico.

Ao final, teremos desenvolvido um produto completo munido do rigor científico adequado, robusto, estruturado e escalonável possibilitando sua aplicação prática por empresas no setor tanto como um algoritmo *solo* e integrado a sistemas de maior complexidade.

Palavras-chave: Forecasting algorithms, reservoir computing, redes neurais, quantitative finance, echo state neural networks, física de sistemas não-lineares, física de sistemas caóticos.

ABSTRACT

Due to a highly competitive market driven by disruptive technologies, it has become necessary to develop novel products well supported by current scientific theories. Furthermore, high-frequency trading algorithms and real-time analysis tools have become the standard approach for problems of capital allocation in the stock market.

Given this scenario, this work proposes the conceptualization of a forecasting tool based on reservoir computing. This tool will be implemented using echo state networks and tested using standard non-linear systems. The performance of the neural network regarding its predictability power will be measured according to several error metrics in order to evaluate the model's capacity.

This concept will be extended to chaotic systems under the influence of strong noise sources. Finally, the system will be applied to real time-series previously selected according to several criteria such as liquiduty, data frequency and practical interest. The tool will then be adequately evaluated, documented and presented.

We aim at developing a scientifically designed forecasting tool that can be readily used by the market at the end of this project.

Keywords: Forecasting algorithms, reservoir computing, neural networks, quantitative finance, echo state neural network, nonlinear systems physics, chaotic systems physics.

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

AI - Artificial Intelligence

ML - Machine Learning

RNN - Recurrent Neural Network

RC - Reservoir Computing

ESN - Echo State Network

ESP - Echo State Property

MSE - Mean Squared Error

RMSE - Root Mean Squared Error

MAPE - Mean Absolute Percentage Error

1 INTRODUÇÃO

Nas últimas décadas o mercado financeiro internacional tem experimentado um aumento abrupto de sua complexidade e interconectividade. O advento de meios de comunicação eficazes, rápidos e acessíveis à população como um todo, democratizaram o acesso a uma variada gama de ativos financeiros em constante expansão. Mercados internacionais tornaram-se acessíveis para investidores físicos e institucionais e a competitividade entre empresas de corretagem atingiu níveis de elevado grau de qualificação profissional.

Nem cem anos se passaram desde que Benjamin Graham publicou sua grande obra intitulada *Security Analysis* [1], considerada um marco para o mercado financeiro, que propôs pela primeira vez muitos dos principais conceitos em análises de ações e *securities*, o compilado destas viria a compor a estruturação do *Value Investing*. Estas análises atingiriam níveis cada vez mais elevados de sofisticação e complexidade nas décadas seguintes. A popularização do *data science* como motor inovativo da indústria no século XXI, e não apenas como disciplina de caráter exclusivamente acadêmico, trouxe às grandes casas do mercado financeiro extenso arsenal de técnicas e tecnologias, hoje vitais para o exercício das diversas atividades do setor além de determinantes nas relações de competitividade.

Diversos pacotes cada vez mais acessíveis ao público em geral e a fluência cada vez maior em linguagens de programação, tornaram de livre acesso diversos algoritmos e métodos antes confinados entre as páginas das mais diversas revistas acadêmicas. O termo *finanças quantitativas* invadiu o mercado financeiro, com cada vez mais bancos, instituições financeiras, gestoras, investidores ou simplesmente entusiastas clamando o desenvolvimento de algoritmos mais precisos, mais rápidos e robustos.

A genuína preocupação pela utilização correta de métodos e tecnologias, a busca por uma análise séria e cética dos dados e o desenvolvimento de metodologias de pesquisa realmente aplicáveis ao mercado financeiro com significância e propriedade científica foram os principais motivadores deste trabalho.

Tendo como principal proposta a aplicação de arquiteturas de redes neurais derivadas do grande ramo da computação intitulado *Reservoir Computing*, algoritmos baseados neste paradigma têm obtido sucesso frente a sistemas de natureza caótica [2] e diversas pesquisas já se propuseram a analisar, do ponto de vista acadêmico, problemas semelhantes [3]. Buscaremos entender a relação deste tipo de algoritmo com propriedades de sistemas caóticos já conhecidos na literatura, ampliaremos o nosso raciocínio inicial de modo a tentar entender o impacto do ruído na capacidade preditiva destes e, por fim, estenderemos nossas ideias para séries temporais reais de ativos financeiros, obtendo assim

uma aplicação prática, concisa e sólida de nossas pesquisas, podendo ser amplamente utilizada por empresas e profissionais atuantes no setor.

1.1 Situação Socieconômica

O produto que buscamos propor visa servir à casas de investimentos bem como a negócios relacionados. Em última análise, ele pode vir a ser utilizado por qualquer empresa que busque aprimorar suas capacidades em *analytics*, com ênfase específica para dados advindos do mercado financeiro.

Buscamos a criação de um software conciso e enxuto que permita analisar séries temporais financeiras, aplicando os métodos e técnicas desenvolvidas a partir do conhecimento da física de sistemas não-lineares e da implementação de arquiteturas de redes neurais.

Diversas plataformas oferecem serviço de prospecção para uma série de ativos financeiros, do mercado de capitais a criptomoedas. Todavia, muitos destes algoritmos tendem a ser simplórios demais, ou alicerçados em técnicas de eficácia dúbia ou não comprovada, como a análise gráfica de ativos, bastante popular entre muitos investidores. Algoritmos mais sofisticados são por vezes disponibilizados em plataformas de *trading*, frequentemente figurando como caixas pretas de constituição desconhecida e disponíveis para o público em geral.

Da mesma forma, nosso produto não se encaixa diretamente com as chamadas *casas de research*, instituições especializadas em analisar e pesquisar o mercado, recebendo remuneração através de relatórios de análises e *insights*. Entretanto, não descartamos que o mesmo possa ter aplicações posteriores neste campo em especial.

Entendemos nosso software como uma solução direta para gestores de investimentos e instituições que busquem acompanhar o comportamento de dado ativo financeiro, podendo constituir, por si só, uma análise de *forecasting* ou uma tecnologia adicional a ser acoplada em outros processos de decisão quantitativamente embasada, bem como em uma variada gama de estratégias de investimentos.

As principais razões pelas quais entendemos este produto e seu conceito como válidos e inovadores, será detalhada adiante, mas é possível afirmar que há um baixíssimo custo computacional em termos de processamento, o que impacta em economia de energia, ganho de tempo no processo, viabilidade para sistemas do tipo *real time* e escalabilidade do algoritmo. Sua precisão e acurácia, especialmente para sistemas de natureza caótica, bastante comentada na literatura, e o domínio de técnicas computacionais já consagradas em física de sistemas não-lineares torna o conceito inteiramente promissor.

Entendemos, portanto, que temos um produto de forte base teórica-computacional,

factível, de baixo custo de desenvolvimento e produção, além de ser proeminente em resultados, pronto para ser posto e testado em um ambiente real.

1.2 Problema Técnico

Em contrapartida ao embasamento teórico e acadêmico, faz-se necessário explicitar, a complexidade técnica e os principais desafios do ponto de vista prático, que constituem as oportunidades por excelência a serem abordadas no desenvolvimento deste trabalho. Entendemos os desafios técnicos do produto como pertencentes a uma ou mais das seguintes categorias a serem explicadas adiante:

1. Bases de dados,
2. Otimização e poder computacional,
3. Precisão e acurácia,
4. Viabilidade de implementação.

O primeiro ponto diz respeito a qualidade e confiabilidade dos dados, a quantidade de dados e ao tratamento necessário. Apesar de a quantidade total de dados a serem processados, para este projeto, não apresentar, *à priori*, um empecilho ao desenvolvimento do trabalho, é necessário o desenvolvimento de técnicas para o devido tratamento e organização dos dados. Os protocolos desenvolvidos devem seguir conceitos de escalabilidade e apresentar a dinâmica necessária para serem revisitados e reavaliados. Nossa base inicial constitui-se de dados financeiros de grande interesse do mercado, obtidos através dos provedores pagos da Refinitiv [1], fornecidos pelo *London Stock Exchange Group*. A empresa oferece dados de excelente qualidade para diversos ativos e mercados, especialmente dados do mercado de capitais norte-americano e de preços de negociação de *commodities*, nos quais temos um interesse em especial. No decorrer deste trabalho, dedicaremos especial atenção às nossas bases de dados e às metodologias para tratá-los devidamente.

O segundo ponto conecta-se diretamente com diversos tópicos apresentados neste trabalho. Quando nos referimos a otimização e ao poder computacional estamos devidamente nos preocupando com dois tópicos essenciais e suas implicações, *tempo de execução* e *complexidade computacional*. Dado que pretendemos implementar um produto viável além de um algoritmo por si só, o tempo de execução e sua dependência com os parâmetros do algoritmo devem ser de conhecimento e avaliados constantemente. Reduzir o tempo de processamento não representa apenas uma vantagem do ponto de vista operacional, como também torna o produto mais atrativo e competitivo, além de reduzir os possíveis

[1] Site oficial da Refinitiv, em português: *https://www.refinitiv.com/pt*

custos associados com a utilização de algoritmos complexos de *deep learning* como o alto consumo de energia elétrica. O produto proposto é, portanto, mais econômico e ecológico.

Em situações limites, onde lidamos com grande necessidade do ponto de vista de processamento de dados ou sinais e esperamos um tempo de resposta ínfimo, algoritmos não otimizados devidamente podem inviabilizar o projeto. Seja por meio da otimização do código ou da análise da tecnologia utilizada, buscaremos tornar o processo o mais viável possível para aplicação e o mais econômico possível sem perda de robustez. O terceiro item diz respeito diretamente a um dos grandes pontos deste projeto. Testar a eficácia de nossa ideia embasará todo o nosso estudo de viabilidade técnica. Buscaremos entender as nuances e os limites de nossas capacidades preditivas e de modelagem, bem como estabelecer margens de segurança para a operação deste produto e limites com relação aos tempos de estimativa e dos tamanhos das bases de *treino* a serem utilizadas, falaremos em muito sobre este ponto nas próximas seções.

O último item espelha a conclusão almejada a ser obtida ao término deste trabalho, esperamos que, tendo cumprido com os itens anteriormente informados, havemos de ter suficiente propriedade técnica para embasar a testagem, prototipação e implementação em maior escala de um sistema mais robusto e/ou em maior escala. Buscamos não apenas atingir determinado grau elevado em termos de desenvolvimento do ponto de vista acadêmico, mas principalmente que o resultado seja de fato aplicável e cumpra com as expectativas comerciais e técnicas que delimitamos à ponto de ser não só comercialmente viável como competitivo no mercado financeiro nacional.

1.3 Ideias

Neste capítulo, adentraremos de forma concisa e expositiva nos principais conceitos, técnicas e metodologias a serem utilizadas no decorrer deste trabalho. Reiteramos que muitos itens serão revisitados e discutidos sob outros prismas em seções posteriores, tendo em vista familiarizar o leitor com conceitos e ideias chave neste domínio do conhecimento. As próximas seções versarão acerca de tópicos em redes neurais, *reservoir computing* e física de sistemas não-lineares, bem como outros assuntos correlacionados.

1.3.1 Redes Neurais

Nas últimas décadas, o expressivo advento de métodos matemáticos e computacionais aplicáveis e implementáveis em contextos de pesquisas acadêmicas em diversas áreas e em diversos setores da cadeia produtiva industrial ou de serviços tem sido alvo de forte atenção e discussão. Associados inerentemente ao processo de desenvolvimento tecnológico e inovativo, os conceitos de *Machine Learning* (ML) uma partição do conceito

de *Artificial Intelligence* (AI) têm estado no cerne das recentes evoluções tecnológicas, com grande ênfase e destaque nas chamadas redes neurais.

Algoritmos deste gênero foram originalmente inspirados como modelos matemáticos de analogia ao sistema nervosos central, tendo suas unidades fundamentais de processamento recebido a nomenclatura de neurônios e suas conexões entre unidades de processamento denominadas conexões sinápticas.

Dentro desta classe de algoritmos, podemos destacar a existência de dois grupos em particular, sendo estes chamados *feedforward neural networks* e *recurrent neural networks* (RNN). Tais categorias distinguem-se entre si pela topologia das conexões entre os neurônios. Neste trabalho estaremos diretamente interessados na segunda categoria.

Dentro de uma análise de escopo superficial, a grande diferença, em termos de topologia, é a existência de ciclos de realimentação em redes do tipo RNN. A existência desses ciclos dá origem a profundas alterações de comportamento, tendo uma alta gama de aplicações e de relevância no meio científico. Do ponto de vista matemático, segundo Lukovsevivcius et al. [2] RNNs se assimilam mais a *sistemas dinâmicos* enquanto *feedforward networks* apresentam a capacidade de generalização de funções.

Tomando esta classe de redes neurais, faremos ainda uma distinção posterior em duas ramificações principais exatamente como proposto por Lukosevicius e Jaeger [2]. A primeira sendo constituída por algoritmos que têm como princípio uma simetria entre suas conexões neurais e uma dinâmica estocástica de minimização de energia e tendo como principal alicerce o campo da física estatística.

A segunda categoria é caracterizada por conexões direcionadas e regras determinísticas para a dinâmica de atualização dos pesos da rede, utilizando-se de filtros não-lineares de maneira a operar uma transformação em uma série de dados inicial de modo a obtermos uma série de saída. Esta categoria tem muito da sua fundação matemática na área de sistemas dinâmicos não-lineares e será desta que derivará toda a nossa discussão e da qual nos referiremos na sequência deste trabalho.

1.3.1.1 Reservoir Computing

Algoritmos do tipo RNN tem cativado há tempos a atenção de pesquisadores e engenheiros, dado sua aparente similaridade com o sistemas nervosos de ordem biológica e capacidade de generalização [4]. Apesar de seu grande potencial, diversos usos práticos deste tipo de tecnologia são limitados por uma série de empecilhos.

Uma das grandes ideias bem sucedidas que endorsa as RNNs é a existência da chamada retro-propagação de erro ou *error backpropagation* (BP) que pode ser pensado de maneira simplória como o uso generalizado de um gradiente visando buscar o mínimo de uma certa função desconhecida.

Apesar de interessante do ponto de vista de aplicações e de angariar bons resultados para diversos casos, o uso de algoritmos baseados em gradientes enfrentavam intrinsecamente problemas dado a possibilidade de existência de mínimos locais na topologia do problema.

O custo computacional elevado podia acarretar em diversos ciclos de atualização e longos períodos de tempo de treinamento, inviabilizando a aplicação de RNNs de dimensões consideráveis, especialmente para problemas em que esperava-se soluções *real time*. Alguns problemas requeriam uma certa experiência por parte do desenvolvedor para uma boa implementação bem como para a melhor escolha dos parâmetros. Somamos a isto o fato de que a convergência do algoritmo não é necessariamente garantida dado, por exemplo, pela existência de bifurcações. [5, 2].

Visando resolver estes problemas, um paradigma mais comumente chamado *Reservoir Computing* (RC), também referido como *Backpropagation-Decorrelation* foi primeiramente proposto em meados de 2001 em duas implementações, uma delas intitulada *Liquid States Machines* (LSM) e proposta por Wolfgang Maass [6] e a outra denotada por *Echo State Networks* (ESN) e proposta por Herbert Jaeger [7].

O centro do paradigma RC reside, justamente, na ideia de uma unidade intitulada reservatório (*reservoir*). De um ponto de vista bastante objetivo, o reservatório consiste de uma RNN que é criada aleatoriamente, ou seja, que tem seus pesos relativos as conexões neurais definidos logo no início do programa através de uma certa distribuição estatística (frequentemente Gaussiana), tendo igualmente uma quantidade pré-definida de pesos de valor zero. Esta matriz de pesos define a topologia da rede e é imutável durante toda execução do algoritmo. Outra característica marcante do paradigma é o uso de regressões lineares visando obter o valor de saída com base nos estados atuais de cada neurônio que compõem a rede [7, 8, 2, 9].

Dentre as principais vantagens da utilização deste paradigma, elencadas ao longo de anos de estudo em estado da arte e aplicações de algoritmos derivados deste modelo inicial, é possível ressaltar a boa acurácia do modelo com relação a algoritmos de propósitos similares, sua fácil implementação, sua simplicidade conceitual, seu baixo custo computacional, sua similaridade e sua notável capacidade de previsão em processos de longa memória e/ou sistemas de natureza caótica [5, 2, 9]. Estas duas últimas propriedades, em particular, serão de vital importância para o embasamento teórico de todo este trabalho, importante destacar que as arquiteturas utilizadas apresentam inspiração biológica em arquiteturas neurais de mamíferos.

1.3.1.2 Echo State Networks

Redes neurais do tipo ESN, tornaram-se, no decorrer de décadas de pesquisa, a arquitetura dominante na literatura científica em RC e em aplicações diversas. Com uma

estruturação matemática bastante sólida e bem definida e bons resultados empíricos e experimentais, esta arquitetura ganhou destaque rapidamente e serviu como base para diversos modelos posteriores em RC [5].

A ideia inicial por trás desta arquitetura residiria na premissa de que a camada oculta, ou *reservoir*, atuaria como um reservatório de dinâmicas capazes que seriam devidamente excitados (em analogia à excitação de neurônios no sistema nervoso central) quando expostos a determinado *input* ou *feedback* [8].

Modelos de arquitetura tipo ESN devem apresentar a chamada *Echo State Property* (ESP), essa propriedade é o que garante que o efeito das condições iniciais do problema desapareça após um determinado transiente finito, impactando no fato de que *inputs* de trajetórias recentes similares evocarão estados de eco próximos, havendo assim uma estabilidade dinâmica por parte do reservatório [5].

Condição ESP: Seguindo a definição de Sun et al.[5], é possível, de forma simplificada, descrever a propriedade de ESP. Sendo F a equação de transição de estados e tendo esta a ESP, para uma dada sequência de padrões de entrada $U = [u(1), u(2), ..., u(N)]$ e duas configurações possíveis distintas de estados iniciais dos neurônios da camada oculta x e x', é possível afirmar que:

$$||F(U, x) - F(U, x')|| \to 0 \; se \; N \to \infty \tag{1.1}$$

De forma geral, a ESP é tida como atingida quando satisfeita a condição de $\rho(\boldsymbol{W}_{res}) < 1$, onde $\rho(\boldsymbol{W}_{res})$ é definido como o raio espectral da matriz de pesos da camada oculta, todavia, a garantia da propriedade de ESP é ainda um problema aberto na comunidade científica [9]. Quando devidamente implementada, esta propriedade garante a estabilidade dinâmica do reservatório. Possibilitando a obtenção de convergência assintótica para os estados dos neurônios da camada oculta dado iterações suficientes. Pesquisas e discussões posteriores tem questionado a suficiência ou abrangência da condição acima, buscando condições mais gerais ou mais matematicamente estritas para a ESP. Como não constitui nossa pretensão adentrar nesta discussão, utilizaremos uma abordagem mais livre neste trabalho, investigando valores abaixo a acima do limite de uma unidade para o raio espectral.

Diversas referências oferecem definições matemáticas para algoritmos básicos de ESNs além de sutis variações posteriores. Um modelo simples pode ser descrito da maneira como fora proposta teoricamente por Jaeger [7, 8], em 2001 e 2002 respectivamente, e trabalhado posteriormente por diversos autores [9, 5]. Assim, definimos para o escopo deste trabalho $\boldsymbol{u}(n) \in \mathbb{R}^{D\times 1}$ como o sinal de entrada no instante n, $\boldsymbol{x}(n) \in \mathbb{R}^{N\times 1}$ são os estados do reservatório em n, $\boldsymbol{y}(n) \in \mathbb{R}^{N\times D}$ é a saída conhecida deste processo. $\boldsymbol{W}_{in} \in \mathbb{R}^{N\times(D+1)}$ são os pesos fixos entre a camada de entrada e a camada oculta (reservatório), $\boldsymbol{W}_{res} \in \mathbb{R}^{N\times N}$

são os pesos das conexões entre neurônios no reservatório e $\boldsymbol{W}_{out}$ são os pesos das conexões entre os neurônios e a saída.

Tomando arbitrariamente os estados iniciais como nulos, i.e. $\boldsymbol{x}(n) = [0, 0, ...0]$, partimos para a primeira etapa do processo de aprendizado da ESN, o processo de *harvesting*. O estado relativo a cada *step* $n+1$ é então calculado com base no estado anterior, relativo a n. Para controlar a velocidade de transição dos estados, ou seja, o quão rapidamente incorporamos novas informações ao estado atual às custas de informações previamente incorporadas, é praxe a introdução de uma variável auxiliar $\alpha_{leaking}$ denominada *taxa de vazamento espectral*. Visando garantir que os estados de cada neurônio tenham módulo inferior ou igual a 1 introduzimos também uma função não-linear f sobre cada um dos neurônios constituintes do estado $\boldsymbol{x}(n+1)$ (vetorizada), neste trabalho em particular faremos uso da função *logit*, outra opção seria o uso da função tangente hiperbólica. Formulamos então a equação de atualização de estados (abaixo) que define qualquer estado $\boldsymbol{x}(n+1)$ como a soma uma razão de $1 - \alpha_{leaking}$ dos seus estados prévios $\boldsymbol{x}(n)$ e a soma de uma razão $\alpha_{leaking}$ das projeções de $\boldsymbol{W}_{in}$ sobre o *input* e de $\boldsymbol{W}_{res}$ sobre os estados anteriores, de modo que garantimos que cada estado é resultado de uma componente passada e de novas atualizações (na proporção da constante $\alpha_{leaking}$) bem como que cada estado tem módulo inferior ou igual a 1.

$$\boldsymbol{x}(n+1) = (1 - \alpha_{leaking})\boldsymbol{x}(n) + \alpha_{leaking} f(\boldsymbol{W}_{in}[\mathbf{1}; \boldsymbol{u}(n+1)] + \boldsymbol{W}_{res}\boldsymbol{x}(n)), \tag{1.2}$$

Seguindo o padrão da literatura em ESN, utilizamos sempre a notação $[;]$, neste caso $[\mathbf{1}; \boldsymbol{u}(n+1)]$, para denotar a concatenação do vetor de *inputs* $\boldsymbol{u}(n+1)$ com um vetor de valores unitários e mesma dimensão $\mathbf{1}$. Dessa forma $\boldsymbol{W}_{in}$ não apenas projeta as variáveis de *input* como também projeta constantes sobre os estados do reservatório, essa operação equivale ao *bias* aplicado sobre os *inputs* em arquiteturas comuns de RNNs e tende a melhorar a performance e generalização do modelo.

É possível descartar deliberadamente uma porcentagem ou número fixo de estados iniciais, tais estados são considerados parte do chamado transiente dado que a Equação 1.2 exige um determinado número de iterações para atingir determinado grau de estabilidade, o descarte desses valores têm impacto positivo nas capacidades preditivas do modelo. De posse da equação de atualização de estados, é possível formular uma equação que projeta o *input* $\boldsymbol{u}$ e os estados $\boldsymbol{x}$ a cada *step* $n+1$ na saída desejada do processo $\boldsymbol{y}$.

$$\boldsymbol{y}(n+1) = \boldsymbol{W}_{out}[1; \boldsymbol{u}(n+1); \boldsymbol{x}(n+1)]. \tag{1.3}$$

A função *logit* é reservada apenas à transição de estados na Equação 1.2, sendo a função identidade aplicada na Equação 1.3, a qual omitiremos da notação. Ao final

do processo, nos resta apenas o fato de os pesos de saída $\boldsymbol{W}_{out}$ não terem sido treinados (são os únicos pesos que são alterados na execução do programa). São criadas então duas matrizes, a matriz de estados $\boldsymbol{X}$, que é concatenação de todos os estados obtidos no período de *harvesting*, e a matriz de *outputs* esperados $\boldsymbol{Y}$. Buscamos a matriz $\boldsymbol{W}_{out}$ que projete os estados em $\boldsymbol{X}$ de modo a obter o menor erro quadrático possível, isso é:

$$W_{out} = \underset{W_{out}}{argmin} \; ||\boldsymbol{W}_{out}\boldsymbol{X} - \boldsymbol{Y}||_2^2. \tag{1.4}$$

Soluções possíveis para a equação acima se dão na forma de simples regressões, o primeiro método proposto na bibliografia consistia no uso de matrizes pseudo-inversas (pseudo-inversa de Moore-Penrose) para a solução da equação linear, solução esta que pode apresentar sérios problemas de convergência bem como *overfitting*. Soluções exploradas na bibliografia atual incluem a regressão de Ridge, o Lasso e Elastic Net, neste trabalho focaremos na primeira solução (mais popular na literatura), usando de sua implementação chamada *regularização de Tikhonov*, de modo que, sendo $\boldsymbol{X}^T$ a transposta de $\boldsymbol{X}$, $\boldsymbol{I}$ a matriz identidade e β o *coeficiente de regularização* a ser definido, temos:

$$\boldsymbol{W}_{out} = \boldsymbol{Y}\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{X}^T + \beta\boldsymbol{I})^{-1}. \tag{1.5}$$

As principais características, do ponto de vista estatístico, das ESNs implementadas são dadas pelos chamados hiper-parâmetros α, ω_{in} e $\rho(\boldsymbol{W}_{res})$ além do já mencionado $\alpha_{leaking}$, que devem ser definidos no início da execução do algoritmo. Na definição mais amplamente utilizada, por convenção, α é denominada *sparsity* e delimita a porcentagem de elementos da matriz $\boldsymbol{W}_{res}$ a serem deliberadamente atribuídos como tendo valor igual a zero, ω^{in} define o intervalo $[-\omega^{in}, \omega^{in}]$ da distribuição uniforme que definirá aleatoriamente os pesos de $\boldsymbol{W}_{in}$. E $\rho(\boldsymbol{W}_{res})$ é chamado de raio espectral da matriz de conexões entre os neurônios e representa, do ponto de vista matemático, o maior autovalor em módulo de $\boldsymbol{W}_{res}$.

Visando garantir a ESP, utilizando a convenção já mencionada anteriormente, a matriz $\boldsymbol{W}_{res}$, inicialmente gerada por meio de uma distribuição uniforme de valores no intervalo [-1, 1], têm seus valores alterados antes da execução de qualquer etapa relacionada ao processo de *learning*. Para tal calculamos o raio espectral da matriz de pesos da camada oculta (original) que a literatura denomina $\lambda_{max}(W_{res})$ e aplicamos a seguinte correção com base em $\rho(\boldsymbol{W}_{res})$, que é o valor desejado de raio espectral:

$$\boldsymbol{W}_{res} \leftarrow \frac{\rho(\boldsymbol{W}_{res})}{\lambda_{max}(\boldsymbol{W}_{res})} \, \boldsymbol{W}_{res}. \tag{1.6}$$

Quando todas as condições foram satisfeitas, os hiper-parâmetros definidos e a matriz $\boldsymbol{W}_{out}$ devidamente treinada, é possível utilizar a ESN em modo *free-running*, recalculando os estados com base nas entradas e estados anteriores e projetando o *output*

desejado, o processo pode se dar indefinidamente conforme o número de *steps* desejados, de maneira geral neste trabalho estamos interessados no caso de um *step* de predição à frente.

Extensão do modelo: Uma alternativa igualmente utilizada na literatura original consiste em acrescentar uma matriz de pesos de *feedback*, isto é, $\boldsymbol{W}_{back} \in \mathbb{R}^{N \times D}$, desta forma evidenciamos mais fortemente as características do tipo RNN da arquitetura. Todavia o formato mais simples permaneceu como o mais utilizado, inclusive na bibliografia recente, as equações de atualização de estados e de predição são reescritas na Equação 1.8 apresentada abaixo, sem a implementação de vazamento espectral e f a chamada função de ativação (*logit*, tangente hiperbólica ou identidade).

$$\boldsymbol{x}(n+1) = f(\boldsymbol{W}_{in}\boldsymbol{u}(n+1) + \boldsymbol{W}_{res}\boldsymbol{x}(n) + \boldsymbol{W}_{back}\boldsymbol{y}(n)), \tag{1.7}$$

$$\boldsymbol{y}(n+1) = \boldsymbol{W}_{out}[\boldsymbol{u}(n+1); \boldsymbol{x}(n+1); \boldsymbol{y}(n)], \tag{1.8}$$

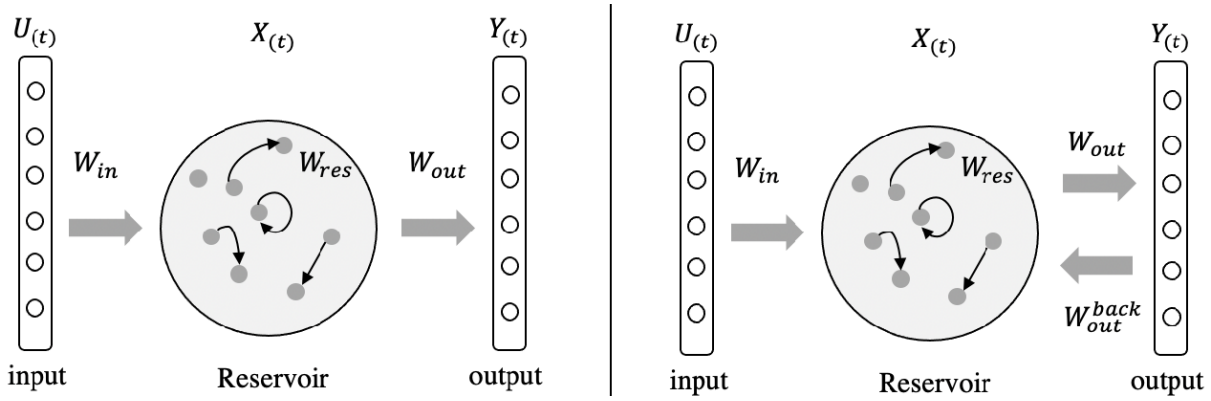

Figura 1 – Demonstrativo de uma ESN com detalhe para o reservatório, *input data* e *output data* com os pesos associados a cada etapa do processo. No esquema à esquerda encontra-se o modelo básico de ESN e a direita uma ESN acrescida de *feedback*. Extraído de Sun et al. [5]

1.3.2 Sistemas não-lineares

Sistemas não lineares constituem alguns dos mais fascinantes problemas encontrados na física, matemática e engenharia, por se tratar de uma área bastante ativa e vívida do ponto de vista científico. Tendo transcorrido menos de um século desde os trabalhos pioneiros como os de Edward Lorenz acerca de sistemas de natureza caótica [10], a física de sistemas caóticos em muito evoluiu, estabelecendo amplas bases teóricas das quais faremos uso. Diversos trabalhos desenvolvidos, demonstram o potencial das ESNs para a predição de séries temporais nestes sistemas [5], de modo que pretendemos utilizar alguns

dos mais comumente citados na área como *benchmark* inicial de avaliação da performance, robustez e capacidade de predição das redes neurais a serem desenvolvidas.

2 ORGANIZAÇÃO & METODOLOGIA

Neste capítulo, desenvolveremos as ideias principais elaboradas no decorrer deste trabalho em termos de organização das etapas envolvidas e cronograma. Tendo como principal objetivo expor devidamente os procedimentos e as premissas adotadas para a obtenção dos resultados expostos na sequencia deste trabalho.

2.1 Metodologia

Tendo empenhado grande parte de nosso esforço no desenvolvimento completo de redes ESN robustas e alinhadas à ampla bibliografia da área. As ESNs desenvolvidas, e cujos resultados serão aqui expostos, foram implementadas em paradigma de programação orientado a objeto (OOP) e implementada em linguagem Python [11].

Nosso trabalho será então dividido em duas partes essenciais, sendo a primeira delas as simulações em séries temporais caóticas e a segunda relativa às séries temporais financeiras de criptoativos. Na primeira etapa estaremos interessados em avaliar as capacidades das ESNs desenvolvidas no que tange tanto as séries univaridas como as séries multivariadas. Testaremos seu potencial como generalizadoras de funções por meio da inserção de ruído gaussiano e comparação de diferentes modelos em critérios de treino e validação e avaliaremos os resultados obtidos. Na segunda etapa, utilizaremos o processo anteriormente desenvolvido de maneira aplicada sobre séries temporais financeiras de ativos reais (criptoativos) e discutiremos os resultados obtidos.

2.1.1 Séries Temporais Caóticas

Para a execução da primeira etapa deste trabalho, selecionamos as principais séries temporais caóticas para fins de avaliação do modelo e discussão: a equação de Mackey-Glass [12], com duas variações de coeficientes $\tau = 17$ e $\tau = 22$, respectivamente, o mapa de Hénon [13] e o atrator de Lorenz [10]. Tendo sido o principal critério utilizado o fato de se tratarem de séries temporais caóticas com características determinísticas já amplamente citadas na literatura e utilizadas com frequência na aplicação e avaliação de ESNs [5]. O problema da análise de séries caóticas consiste, principalmente, na instabilidade das trajetórias com relação ao ponto inicial, de difícil predição.

Mackey-Glass (unidimensional): Descrita matematicamente por Mackey e Glass em 1977 [12]. Tem aplicações em fisiologia, relacionando a densidade de células $P(t)$

com cinco parâmetros denotados β_0, θ, n, γ e τ:

$$\frac{dP(t)}{dt} = \frac{\beta_0 \theta^n P(t-\tau)}{\theta^n + P(t-\tau)} - \gamma P(t). \tag{2.1}$$

É possível igualmente reescrever a equação tomando a variável $Q(t) = P(t)/\theta$, de modo que θ é ignorado de forma explícita. Assim, é possível escrever:

$$\frac{dQ(t)}{dt} = \frac{\beta_0 Q(t-\tau)}{1 + Q^n(t-\tau)} - \gamma Q(t), \tag{2.2}$$

Neste sistema a variável τ é de extrema importância para o grau de caoticidade esperado da série temporal, de modo que quanto maior o valor desta variável, o será a dificuldade de predição do sistema. Nas séries temporais a serem utilizadas neste estudo utilizaremos a equação 2.1 com os parâmetros $\gamma = 0.9$, $\beta_0 = 0.2$, $n = 10$ e $\theta = 0.8$ e a condição inicial única $P(0) = 0.1$. Sendo a série Mackley-Glass 17 constituída com $\tau = 17$ e a série Mackley-Glass 22 constituída com $\tau = 22$, estes dois valores encontrados com certa frequência em trabalhos similares.

Mapa de Hénon (bidimensional): Proposto pelo francês Michel Hénon no ano de 1976 [13], o mapa é frequentemente descrito na literatura em variações tridimensionais e bidimensionais, para a realização deste trabalho optamos pela análise do Mapa de Hénon bidimensional, mais comumente discutido em análises similares, sendo descrito pela seguinte equação:

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \begin{pmatrix} y_n + 1 - a x_n^2 \\ b x_n \end{pmatrix}. \tag{2.3}$$

Para fins de análise, os dois parâmetros serão definidos de tal maneira que $a = 1.4$ e $b = 0.3$, parâmetros bastante populares na literatura. Como condição inicial da trajetória, definimos o ponto dado por $x_0 = 0$ e $y_0 = 0$, outra combinação bastante usual para este sistema.

Atrator de Lorenz (tridimensional): Um sistema não-linear bastante conhecido e talvez a equação mais comumente utilizada em caos determinístico, o atrator de Lorenz foi proposto por Edward Lorenz em 1963 [10]. O sistema tridimensional é ditado por três equações diferenciais, tais equações são, por sua vez, descritas através de três parâmetros chamados de (i) *Número de Prandtl* σ, (ii) o *Número de Rayleigh* ρ e (iii) um parâmetro adicional (sem nome) β, de modo que:

$$\frac{dx}{dt} = \sigma(y - x), \tag{2.4}$$

$$\frac{dy}{dt} = x(\rho - z) - y, \tag{2.5}$$

$$\frac{dz}{dt} = xy - \beta z. \tag{2.6}$$

Uma combinação convencional dos três parâmetros, e que utilizaremos no presente trabalho, consiste em $\rho = 28$, $\sigma = 10$ e $\beta = 8/3$. Tendo como ponto inicial da trajetória $x_0 = 1$, $y_0 = 1$ e $z_0 = 1$.

2.1.2 Séries temporais financeiras

Para avaliação do nosso modelo, optamos por desenvolver nossas análises para dados de fechamento, de frequência diária, de criptoativos. Para a realização deste trabalho, optamos pelos seguintes ativos financeiros: Bitcoin (BTC), Ethereum (ETH), Cardano (ADA) e Dogecoin (DOGE). Sendo ambos ativos de negociação contínua (24 horas de negociação em todos os 7 dias da semana), o critério de seleção deu-se por questões de maiores valores de *market cap* e popularidade na comunidade *crypto*.

Para obtenção das séries de frequência diária, utilizamos o provedor Eikon da Refinitiv (*London Stock Exchange Group*). A requisição foi feita de modo a obter a maior série possível das disponíveis nas bases de dados dos provedores. As séries apresentam quantidades distintas de observações, associadas a distintos intervalos de tempo.

A análise de dados de criptoativos é um tema de relevância crescente e foco de grande volume de discussões envolvendo suas perspectivas sociológicas, econômicas, ambientais, tecnológicas e financeiras. A presença de caos em séries temporais de ativos financeiros é ainda um tema em discussão no meio acadêmico e esperamos comparar os resultados obtidos com as séries caóticas. Assim, a escolha deste grupo de ativos pode em muito contribuir para o debate de ideias bem como possibilitar aplicações diretas do método na indústria financeira. Creditamos também o conjunto de quatro criptoativos a serem utilizados como sendo bons representantes da classe de ativos em questão em toda a sua extensão.

2.1.3 Métricas

Para avaliação dos modelos, serão empregadas o *root mean squared error* (RMSE), uma modificação do *mean squared error* (MSE), e o *mean percentage absolute error* (MAPE), todas as medidas empregadas neste trabalho são para *forecasting in-sample*.

As medidas de MSE, sua variação de RMSE e o MAPE são obtidas através das equações abaixo, onde N é o número de observações de uma dada série de dados, y_i é o valor de saída desejado e $\hat{y}_i$ são os valores preditos para os dados de saída.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2, \tag{2.7}$$

$$RMSE = \sqrt{MSE}, \tag{2.8}$$

$$MAPE = \frac{100\%}{N}\sum_{i=1}^{N}\left|\frac{y_i - \hat{y}_i}{y_i}\right|. \tag{2.9}$$

2.1.4 Metodologia de pesquisa: Séries Temporais Caóticas

A metodologia de avaliação empregada para este conjunto de séries, consistirá na comparação de três métodos, denominados I, II e III. Estes modelos nos auxiliarão a compreender o processo de aprendizagem e generalização de padrões resultante da aplicação das ESN, bem como avaliar a eficiência do modelo implementado em modelagem de dados univariados e multivariados e a sensibilidade do sistema quando da incidência deliberada de ruído gaussiano.

Dadas nossas séries temporais, passaremos inicialmente ao processo de normalização dos dados utilizados, isso é, buscamos uma distribuição de dados de entrada relativamente simétrica, de média 0 e desvio padrão 1, para este fim utilizaremos a implementação do método de Yeo-Johnson [14], amplamente utilizado na comunidade de ML, na biblioteca *scikit-learning* [15] em Python.

A análise apresentada neste trabalho fará uso de conjuntos de dados de 5000 observações para cada uma das séries temporais avaliadas, sendo as 3500 observações iniciais separadas como conjunto de treino e as 1500 posteriores pertencentes ao conjunto de validação do modelo, em uma configuração convencional de proporção de treino e validação de 70% e 30% do conjunto total. Restringiremos nossa análise a previsões de um passo a frente *in-sample* (*one steap ahead*). Para avaliação dos diferentes métodos (I, II e III), para alguns dos conjuntos de treino e/ou validação serão submetidos à incidência deliberada de ruído gaussiano, de modo a possibilitar a avaliação de robustez. Os três modelos avaliados serão:

I Método treinado sobre conjunto de treino original, sem adição de ruído gaussiano, e executado sobre conjunto de validação original, igualmente sem adição de quaisquer fontes de ruído;

II Método treinado sobre conjunto de treino original, sem adição de ruído, mas executado sobre conjunto de validação original acrescido de ruído gaussiano;

III Método treinado sobre conjunto de treino original acrescido de ruído gaussiano e executado sobre conjunto de validação original acrescido de ruído gaussiano;

O foco principal de nossa análise, no que concerne o conjunto de dados de treinamento, orbita em torno da descoberta dos chamados hiperparâmetros ótimos (raio espectral, escala, vazamento espectral e esparsidade do reservatório). Para tal, toda a etapa de *learning* consistirá no levantamento dos erros dos modelos (RMSE e MAPE) para cada uma das combinações possíveis de hiperparâmetros escolhidas. Optamos, com base em exploração empírica assim como por algumas bases literárias [5, 9], por avaliar todas as combinações possíveis entre 10 valores de esparsidade e taxa de vazamento espectral, igualmente espaçados no intervalo $[0.05; 0.95]$ e 10 valores de raio espectral e escala, igualmente espaçados no intervalo $[0.1; 1.9]$. O número de neurônios na camada oculta é deliberadamente fixado em $N = 20$, a proporção de valores dedicados ao transiente, ou seja, valores que serão descartados no processo de *learning*, é fixado em 10% das observações do conjunto de treino e o tamanho de janela para os padrões de entrada é determinado como 1.

Como as matrizes pseudo-aleatórias podem distinguir entre si na qualidade da predição, optamos pela avaliação de 10 ESNs distintas para cada combinação de hiperparâmetros de modo a avaliar os resultados obtidos de cada combinação sob a ótica de uma medida estatística de tendência central. Totalizaram-se assim 10^5 rodagens distintas de ESN para o conjunto de treino. O processo é realizado duas vezes, uma para o conjunto de treino original e outro para o conjunto de treino acrescido de um ruído gaussiano estimado empiricamente.

Para cada etapa de *learning*, é obtido um arquivo com todos os parâmetros e características das redes neurais utilizadas, os dados são então avaliados por meio de uma medida de tendência central, neste caso a mediana dos erros. Para visualização dos erros, faremos uso de *Heat Maps* de duas variáveis e das medidas de MAPE associadas aos métodos. Assim, para cada etapa de *learning* são gerados 6 *Heat Maps*, contendo todas as combinações em pares dos 4 hiperparâmetros avaliados (i.e. raio espectral vs. esparsidade, raio espectral vs. vazamento espectral, etc.), de modo que cada ponto de um *Heat Map* equivale a mediana dos valores de MAPE obtidos para aquela combinação fixa específica de dois hiperparâmetros (sendo os outros dois variáveis), um exemplo em separado pode ser visualizado abaixo.

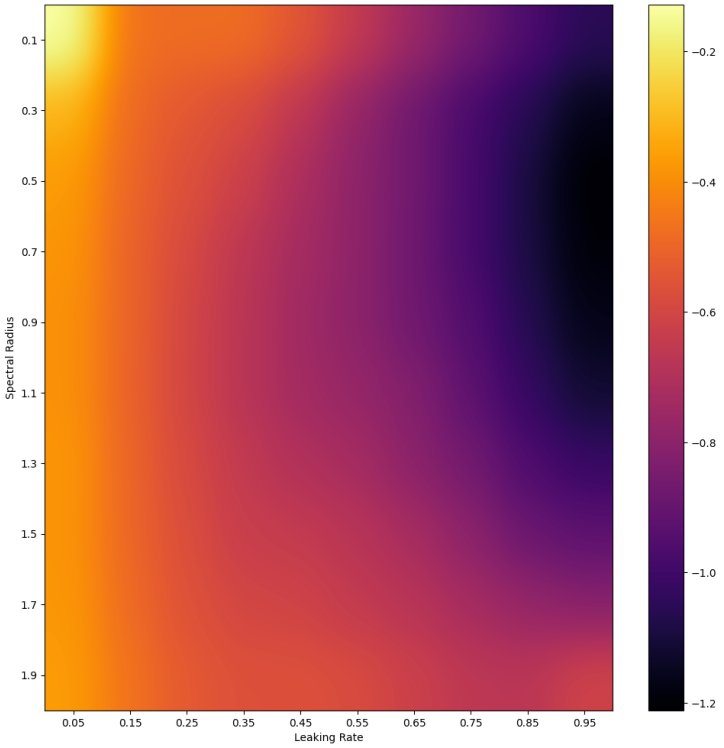

Figura 2 – Um exemplo desconexo de *Heat Map*, mostrando diferentes valores obtidos para o caso das variáveis raio espectral e taxa de vazamento espectral. Cada ponto apresentado equivale a uma medida de tendência central (média ou mediana) de um conjunto de medições para os quais os valores de raio espectral e taxa de vazamento espectral são fixos em detrimento dos outros hiperparâmetros, que são variáveis, e das arquiteturas, que são distintas entre si. Na escala de cores padrão utilizada, quanto mais amarelado (claro) um ponto, maior o valor da variável em questão e quanto mais arroxado (escuro) um ponto, menor o valor da variável apresentada. Um sucinta barra de valores, com as respectivas cores, é apresentada no canto direito da imagem.

Pela observação dos *Heat Maps* para cada procedimento de *learning* é possível estimar aproximadamente os hiperparâmetros que melhor minimizam o MAPE do conjunto de treino, esses hiperparâmetros são então denominados *hiperparâmetros ótimos*.

Executamos então as mesmas ESNs com o uso dos hiperparâmetros ótimos e com mesmas matrizes pseudo-aleatórias (utilizadas com estes hiperparâmetros no processo de *learning*) no conjunto de validação. O conjunto de hiperparêmtros selecionados tem 10 modelos ESNs distintos associados e a validação é feita para todos os valores de N múltiplos de 20 no intervalo $[20; 500]$. O processo é realizado para os três modelos separadamente e os resultados são avaliados conjuntamente de forma a explicitar a comparação dos resultados e possibilitar as discussões acerca das capacidades de generalização dos conjuntos de dados propostos mesmo sob a influência de ruído exógeno ao sistema.

2.1.5 Metodologia de pesquisa: Séries temporais financeiras

A análise das séries temporais financeiras se dará em etapas de tratamento, teste e validação dos modelos. Tendo como base o processo desenvolvido anteriormente em análise de séries temporais caóticas.

A primeira etapa, de tratamento dos dados, consiste inicialmente no nivelamento das bases de dados a serem utilizadas de modo a garantir que todos os conjuntos de dados tenham o mesmo número de observações e que essas observações sejam relativas ao mesmo período de tempo de frequência diária. Garantida esta condição, tomamos o log-retorno dos dados de fechamento para o período analisado, os log-retornos apresentam propriedades estatísticas relevantes ao nosso estudo e que serão melhor discutidas posteriormente, além de serem um procedimento de praxe na área de séries temporais financeiras e econometria.

As séries de log-retornos serão então normalizadas por meio do método de Yeo-Johnson [14], e as distribuições finais serão verificadas quanto as suas propriedades estatísticas. O conjunto resultante de log-retornos normalizados de frequência diária será então dividido cronologicamente, de modo que 70% das observações consistirão os conjuntos de treino (sendo 10% deste o transiente) e os 30% remanescentes consistirão nos conjuntos de validação. As previsões serão sempre realizadas um passo a frente.

A etapa de *learning* será similar a realizada anteriormente, o tamanho da janela para os padrões de entrada será mantido como 1 e o número de neurônios na camada oculta será fixado em $N = 20$. Serão avaliadas todas as combinações possíveis obtidas através de 10 valores de esparsidade e taxa de vazamento espectral, igualmente espaçados no intervalo $[0.05; 0.95]$ e 10 valores de raio espectral e escala, igualmente espaçados no intervalo $[0.1; 1.9]$. Para cada combinação de hiperparâmetros, serão implementados 10 modelos distintos de ESNs pseudo-aleatórias, de modo a obter estatísticas para a análise dos modelos, totalizando 10^5 execuções.

Seguindo o protocolo adotado anteriormente, a análise de cada ativo resultará em conjunto de 6 *Heat Maps* constituídos através das medianas do MAPE para cada par fixo dentro das combinações possíveis de hiperparâmetros. A visualização do conjunto permitirá a determinação aproximada dos hiperparâmetros ótimos associados a cada conjunto de dados.

As arquiteturas selecionadas e os hiperparâmetros ótimos serão implementados sobre ambos os conjuntos, teste e validação. Faremos uso de análise de autocorrelação (ACF) e de autocorrelação parcial (PACF) de modo a compreender as relações de memória presentes na série, apesar de amplamente utilizados em modelos de séries temporais clássicos da literatura, como ARMA e ARIMA, a aplicação desta metodologia para redes neurais não é ainda tão convencional, apesar de diversas literaturas demonstrarem seus benefícios em termos de avaliação quanto a modelagem no conjunto de dados de testes

[16].

Tais medidas serão realizadas junto ao conjunto de treino, de modo a identificar as relações relevantes entre as variáveis e adquirir um ideia intuitiva acerca do tamanho provável de janela a ser almejado, aumentaremos gradativamente o tamanho de janela de entrada do modelo e analisaremos o ACF e o PACF dos resíduos obtidos através da subtração da série original pela modelada.

Uma vez que todas as relações relevantes forem eliminadas pelo modelo este é considerado capaz e seu tamanho de janela é então fixado. As arquiteturas selecionadas são então implementados no conjunto de dados de validação com a fixação dos hiperparâmetros ótimos e do tamanho da janela de padrões de entrada. O número de neurônios na camada oculta é variado e os resultados em termos de RMSE e MAPE são analisados.

2.2 Cronograma

Seguindo a linha de desenvolvimento inicialmente pensada, apresentamos o cronograma final do desenvolvimento realizado para a primeira e segunda etapa deste trabalho de conclusão de curso, respectivamente nas figura 3 e 4 à seguir.

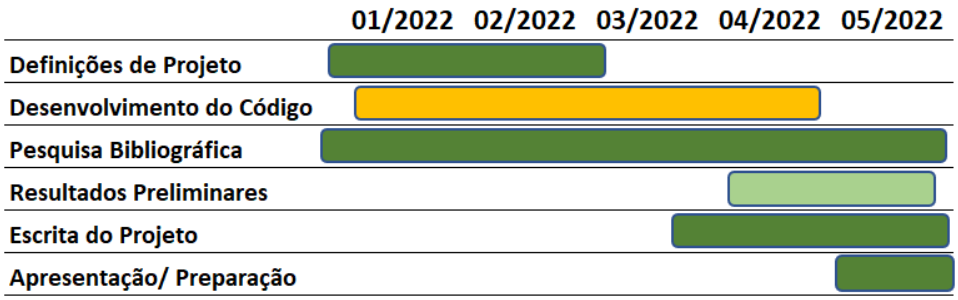

Figura 3 – Cronograma de atividades realizadas na primeira etapa do trabalho de conclusão de curso.

Como demonstrados, os desenvolvimentos na primeira etapa do trabalho de conclusão de curso ocorreram de forma rápida e concisa, sem grandes contratempos em nenhuma das etapas cruciais do processo, o que permitiu uma escrita fluida e apresentação subsequente.

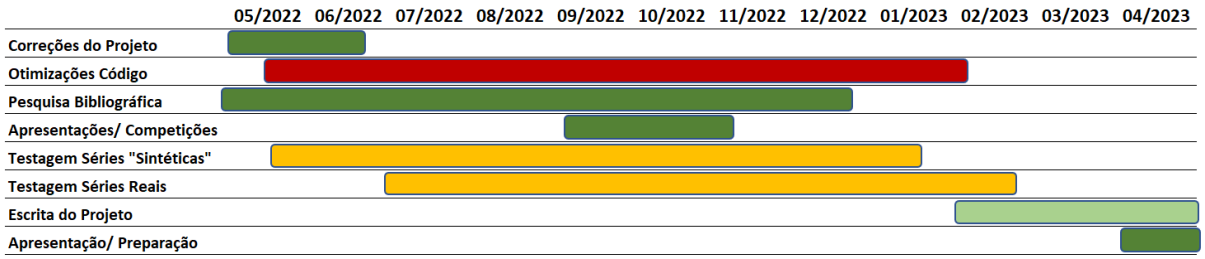

Figura 4 – Cronograma de atividades realizadas na segunda etapa do trabalho de conclusão de curso.

Na segunda etapa, contratempos foram adquiridos através das otimizações e extensões do código tanto por questões de otimização da ESN como pela reinterpretação dos trabalho conforme este se desenvolvia, pelo maior conhecimento da bibliografia e da implementação destas redes. Um pequeno atraso foi deliberadamente ocasionado em meados de outubro por conta da participação e apresentação deste projeto em competições e palestras (mais informações ao final deste trabalho), todavia é de consenso de todos os envolvidos na produção deste trabalho que tais apresentações e competições apenas reforçaram o projeto em termos técnicos e aumentaram seu impacto.

3 RESULTADOS & DISCUSSÃO

Neste capítulo exibiremos os principais resultados obtidos, bem como aprofundaremos a discussão acerca da aplicação das ESN, da viabilidade do sistema, da robustez do método e das métricas obtidas.

3.1 Séries Temporais Caóticas

Seguindo a metodologia proposta, as séries caóticas foram normalizadas através do método de Yeo-Johnson [14] e repartidas em conjunto de treino, com 3500 observações (70%), e teste, com 1500 observações (30%). Iniciamos as testagens dos três modelos propostos, variando os quatro hiperparâmetros de interesse (raio espectral, escala, taxa de vazamento espectral e esparsidade).

O procedimento, no conjunto de treino, foi feito duas vezes para todas as séries caóticas determinísticas: Mackey Glass-17 ($\tau = 17$), Mackey Glass-22 ($\tau = 22$), Mapa de Hénon e Atrator de Lorenz. Sendo a primeira vez no conjunto limpo de dados e a segunda na presença de ruído, de modo a avaliar os três métodos distintos.

Tendo cada combinação de quatro hiperparâmetros sido testada com quatro distintas arquiteturas pseudo-aleatórias, construímos os *Heat Maps* através das medianas dos erros (RMSE e MAPE) em cada uma das combinações 2 x 2 de hiperparâmetros. Para fins de visualização, os resultados são expostos em base logarítmica de 10, de modo a ressaltar as diferenças e a dinâmica dos erros encontrados.

Na sequência é possível visualizar todos os resultados obtidos para os *Heat Maps* associados ao Mapa de Hénon. Todos os *Heat Maps* adicionais estão presentes no apêndice A (5). Os valores aproximados para os hiperparâmetros ótimos, foram compilados em uma tabela.

Hiperparâmetros	**MackeyGlass-17**	**MackeyGlass-22**	**Mapa de Hénon**	**Atrator de Lorenz**
Raio Espectral	0.70	0.70	0.10	0.10
Escala da Entrada	0.30	0.30	1.50	0.10
Esparsidade	0.75	0.75	0.95	0.95
Vazamento Espectral	0.95	0.95	0.95	0.55

Tabela 1 – Hiperparâmetros ótimos aproximados obtidos via análise de Heat Maps para séries temporais caóticas.

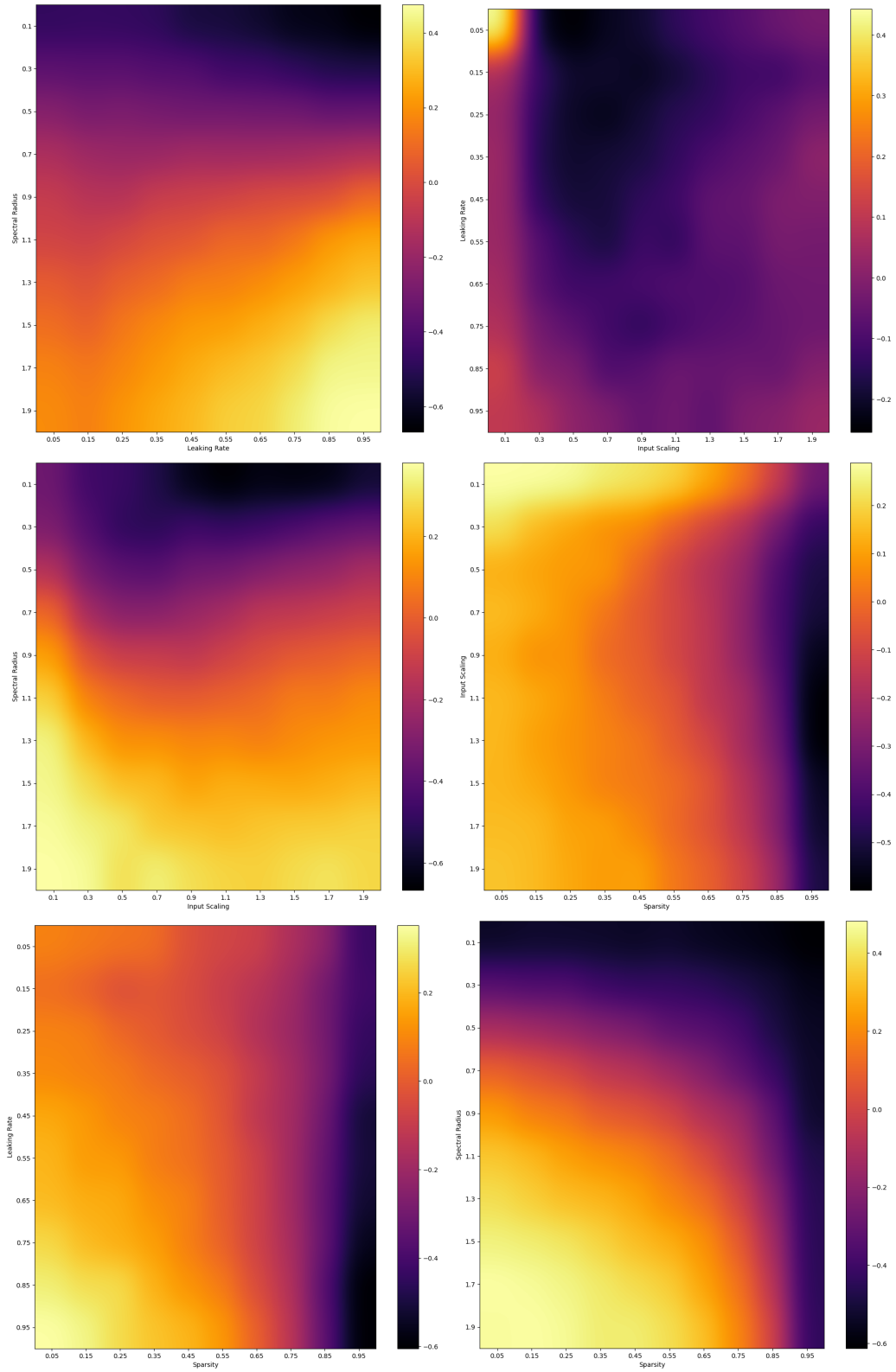

Figura 5 – Configurações de *Heat Maps* com medianas do MAPE em escala log10, para o mapa de Hénon evidenciando comportamento das ESNs em conjunto de teste face alterações nos hiperparêmtros

Os *Heat Maps* encontrados para a aplicação do modelo no Mapa de Hénon, expostos na figura 5, demonstram a dinâmica dos hiperparâmetros associados a rede com relação ao erro obtido. Neste caso em especial, é possível visualizar um claro desenho de gradiente, com zonas de minimização de medianas de erros (cores mais escuras) bem como zonas de maximização de mediana de erros (cores mais claras). Há transição entre estes pontos extremos dá-se de forma gradual e homogênea, de modo que é simples identificar os hiperparâmetros ótimos para o sistema. Testagens similares foram realizadas com adição de ruído gaussiano de média $\mu = 0$ e desvio-padrão σ, de modo a analisar a construir os modelos I, II e III e analisar a robustez dos modelos empregados.

O nível de ruído, disposto na Tabela 2, dado através de σ, foi determinado empiricamente para cada série temporal, de modo a não ser tão extenso a ponto de impossibilitar a comparação entre os diferentes modelos nem tão pequeno que pudéssemos ressaltar a dinâmica de generalização das ESNs. Dessa forma, os hiperparâmetros ótimos obtidos para os dados sob influência de ruído não diferem dos encontrados na Tabela 1 de hiperparâmetros, dado que os *Heat Maps* (dispostos no apêndice A, capítulo 5) não apresentaram desvios significativos no seu comportamento.

Nível de Ruído	**MackeyGlass-17**	**MackeyGlass-22**	**Mapa de Hénon**	**Atrator de Lorenz**
σ	2.5×10^{-3}	2.5×10^{-3}	1.0×10^{-4}	5.0×10^{-5}

Tabela 2 – Nível de ruído incidido sobre os conjuntos normalizados, dado através do desvio-padrão σ de uma distribuição gaussiana de média $\mu = 0$, obtido empiricamente para séries temporais caóticas.

Definidos os hiperparâmetros ótimos e os níveis de ruído, executamos as comparações entre os métodos I, II e III. Para melhor compreensão da dinâmica dos ruídos, comparamos ambas as medidas de erros (RMSE e MAPE) para cada eixo separadamente. Os modelos foram reproduzidos com tamanho de camada oculta N variando entre 20 e 500 em múltiplos de 20. Sendo possível avaliar a distinção das 10 ESNs pseudo-aleatórias via utilização de Box Plots, os resultados visuais mostram o comportamento para o métodos I (cor vermelha), II (cor azul) e III (cor verde).

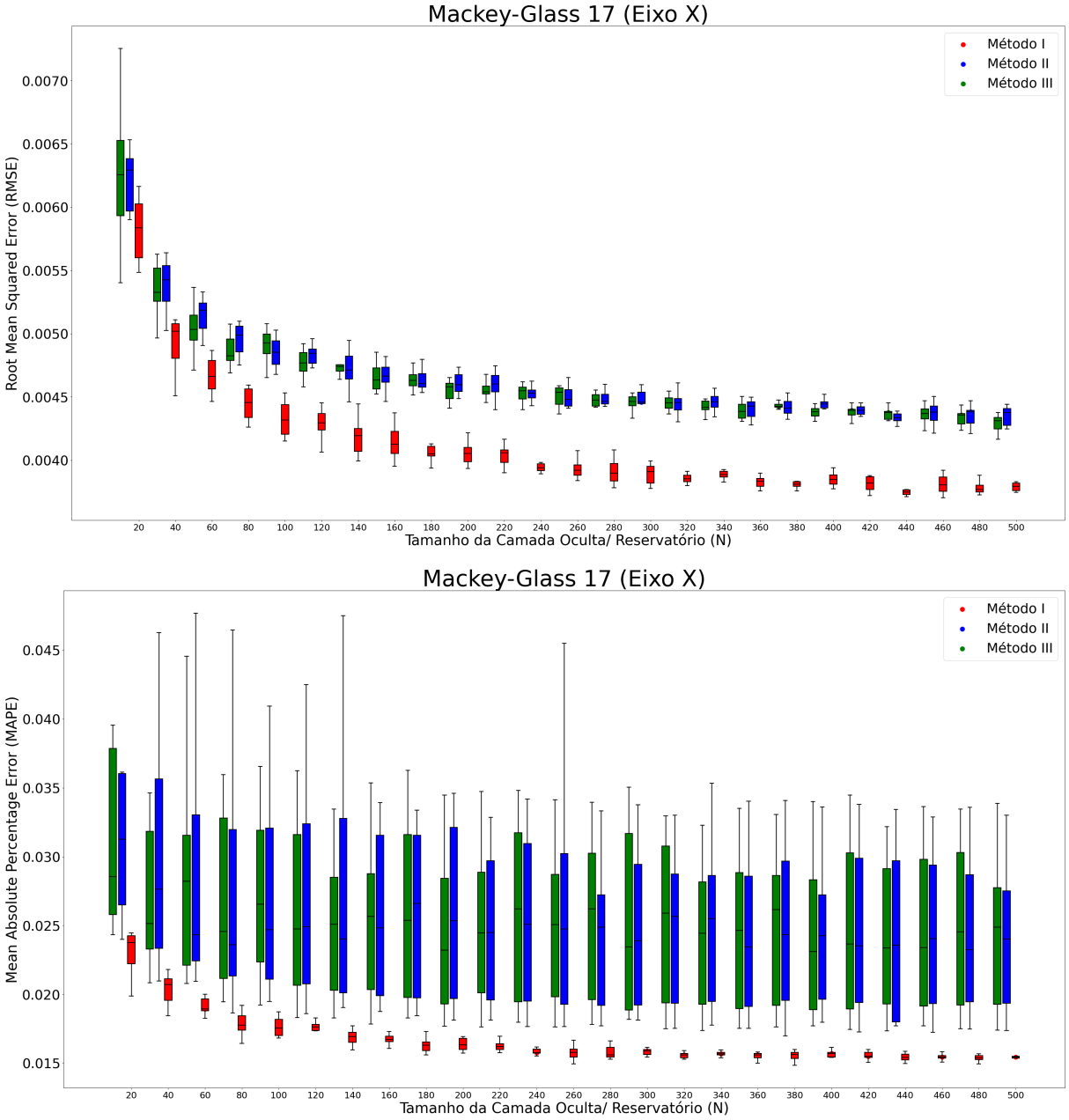

Figura 6 – Comparação entre os métodos I (vermelho), II (azul) e III (verde) para o Mackey Glass-17. São apresentados os valores de RMSE (acima) e os valores do MAPE (abaixo). A série temporal em questão é unidimensional.

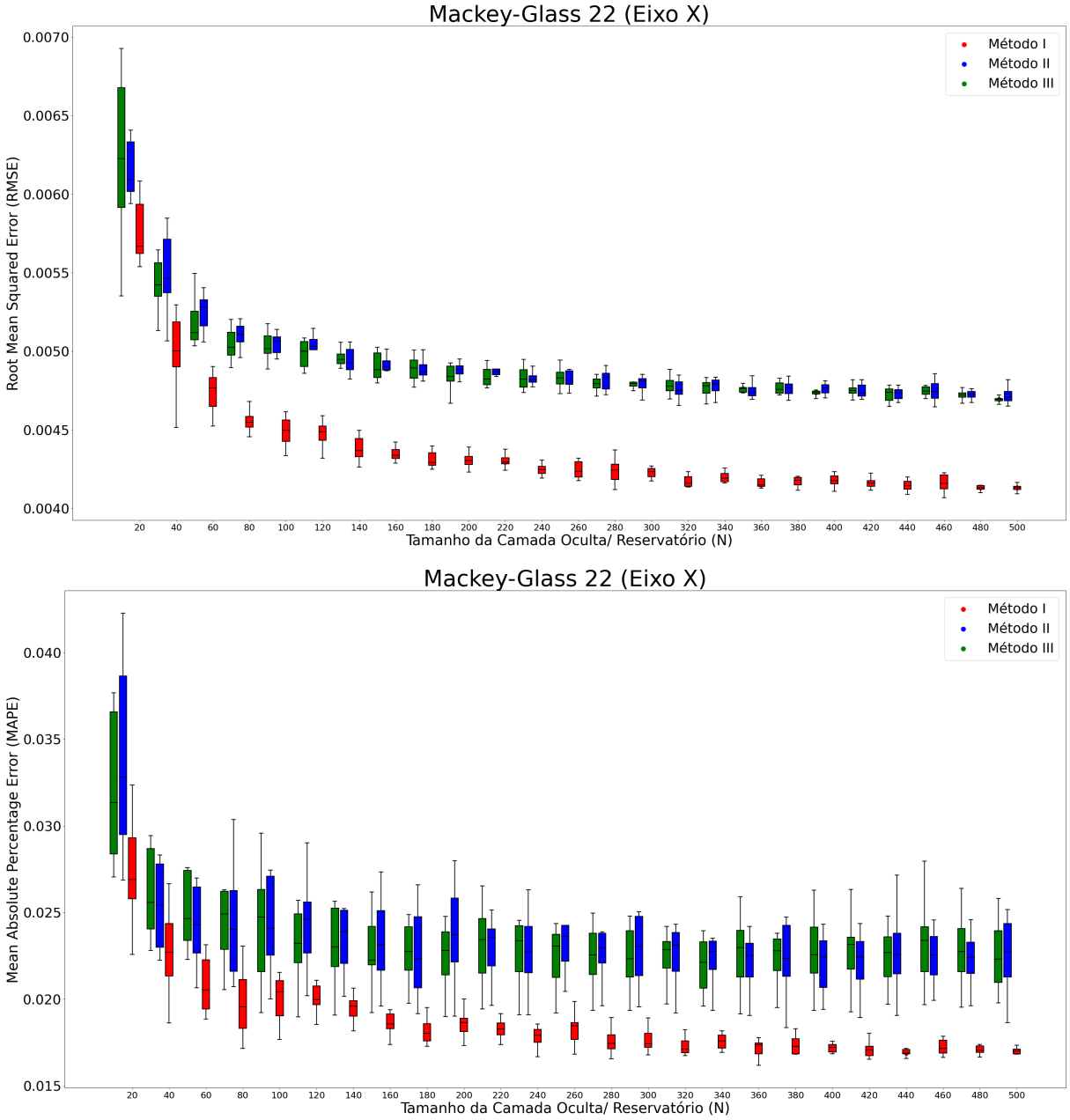

Figura 7 – Comparação entre os métodos I (vermelho), II (azul) e III (verde) para o Mackey Glass-22. São apresentados os valores de RMSE (acima) e os valores do MAPE (abaixo). A série temporal em questão é unidimensional.

Séries unidimensionais como o Mackey Glass-17, presente na figura 6, e Mackey Glass-22, na figura 7, apresentam lento decaimento de erros (RMSE e MAPE) conforme aumento da camada oculta (reservatório), essa curva evolui para uma diferença quase constante entre o método I e suas versões com adição de ruído (II e III), uma certa tendência assintótica torna-se visível com valores de camada oculta elevados ($N > 200$).

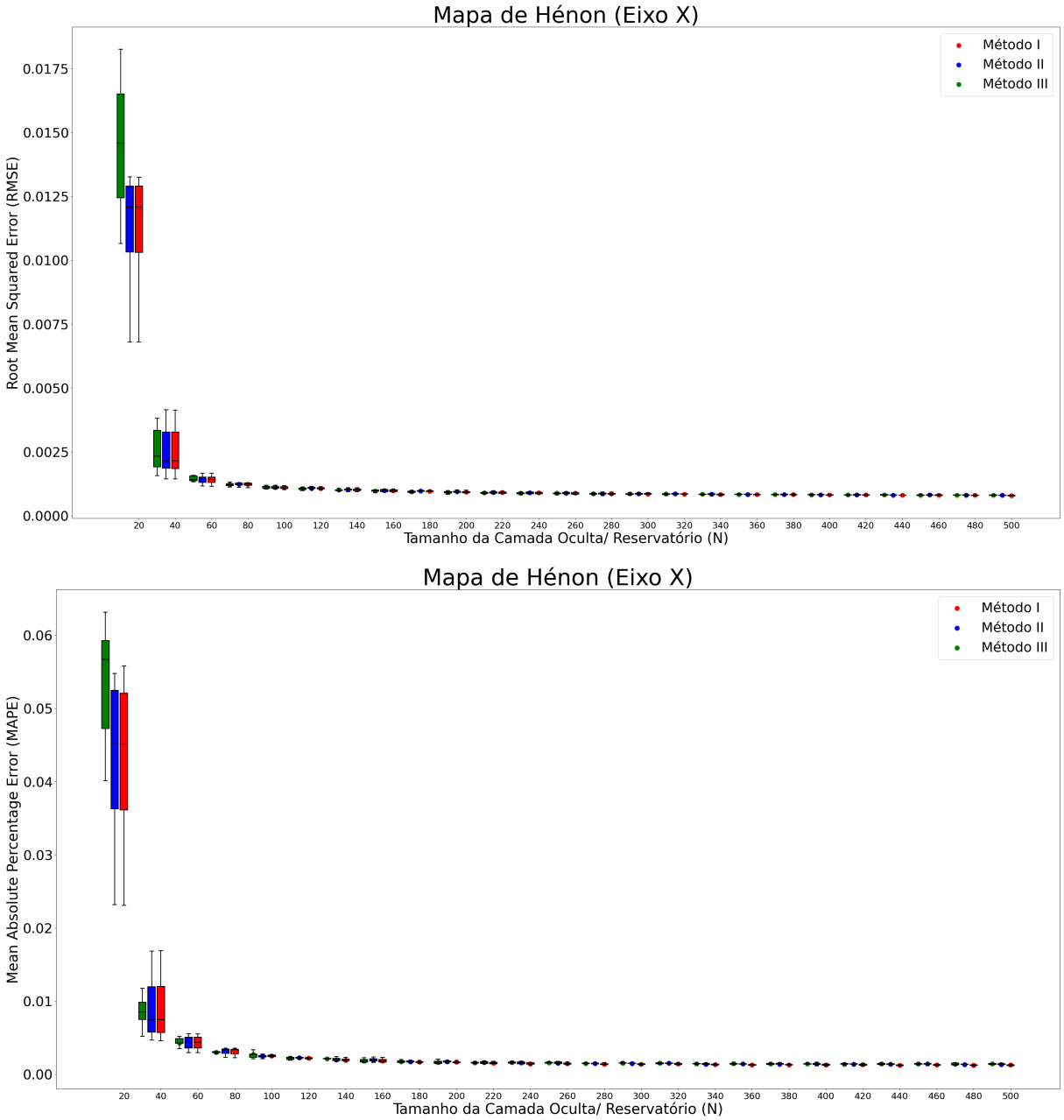

Figura 8 – Comparação entre os métodos I (vermelho), II (azul) e III (verde) para o eixo X do Mapa de Hénon. São apresentados os valores de RMSE (acima) e os valores do MAPE (abaixo).

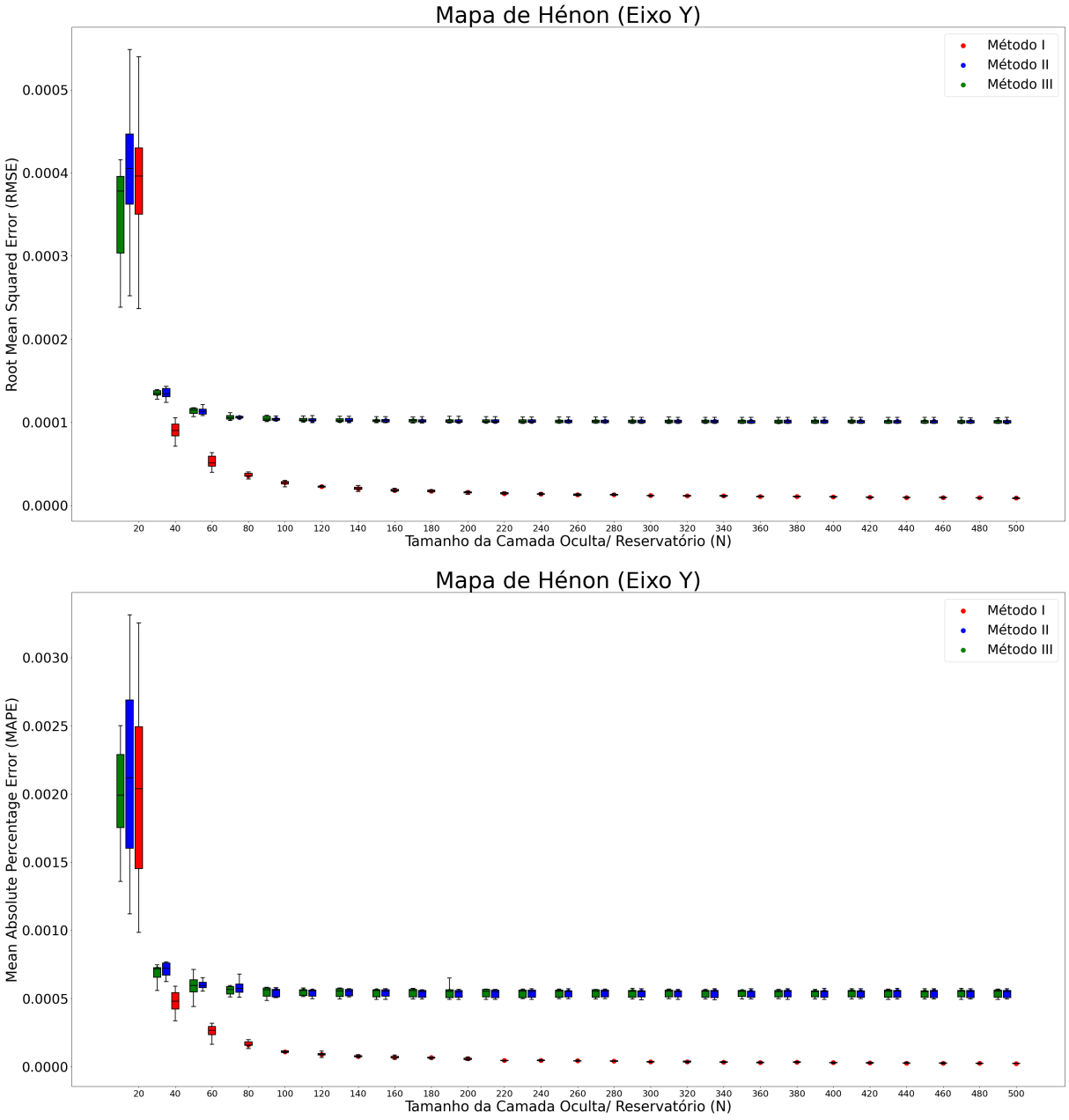

Figura 9 – Comparação entre os métodos I (vermelho), II (azul) e III (verde) para o eixo Y do Mapa de Hénon. São apresentado os valores de RMSE (acima) e os valores do MAPE (abaixo).

Séries temporais multidimensionais, por outro lado, apresentam comportamento distinto conforme o eixo de análise. Para o Mapa de Hénon, o eixo X tem valores associados de erros praticamente indistinguíveis entre os três métodos enquanto. O eixo Y apresenta comportamento similar ao dos séries unidimensionais, com comportamento assintótico bem estabelecido para valores em $N > 100$, de modo que todo o ganho do modelo se apresenta através do eixo Y.

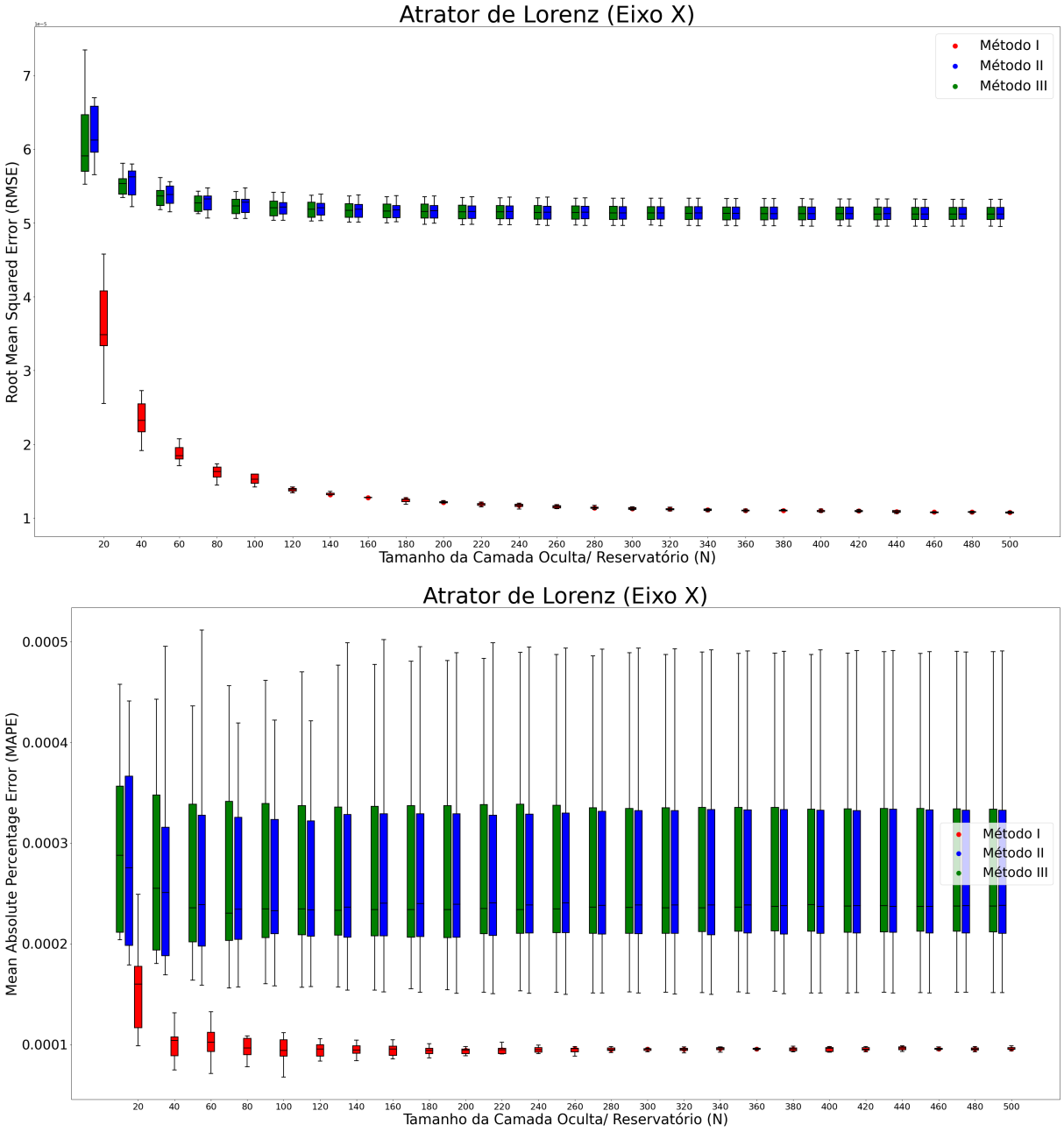

Figura 10 – Comparação entre os métodos I (vermelho), II (azul) e III (verde) para o eixo X do Atrator de Lorenz. São apresentados os valores de RMSE (acima) e os valores do MAPE (abaixo).

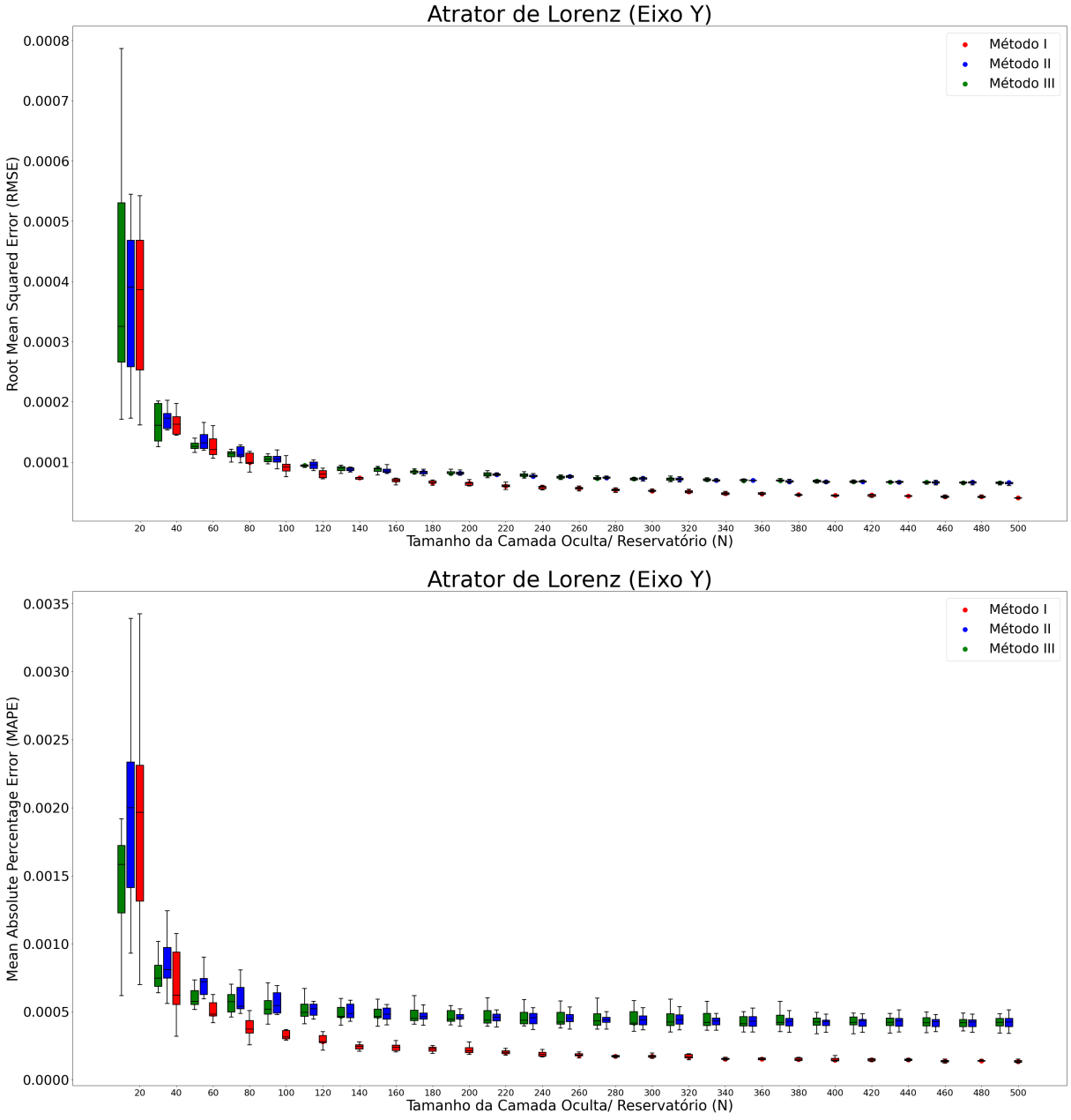

Figura 11 – Comparação entre os métodos I (vermelho), II (azul) e III (verde) para o eixo Y do Atrator de Lorenz. São apresentados os valores de RMSE (acima) e os valores do MAPE (abaixo).

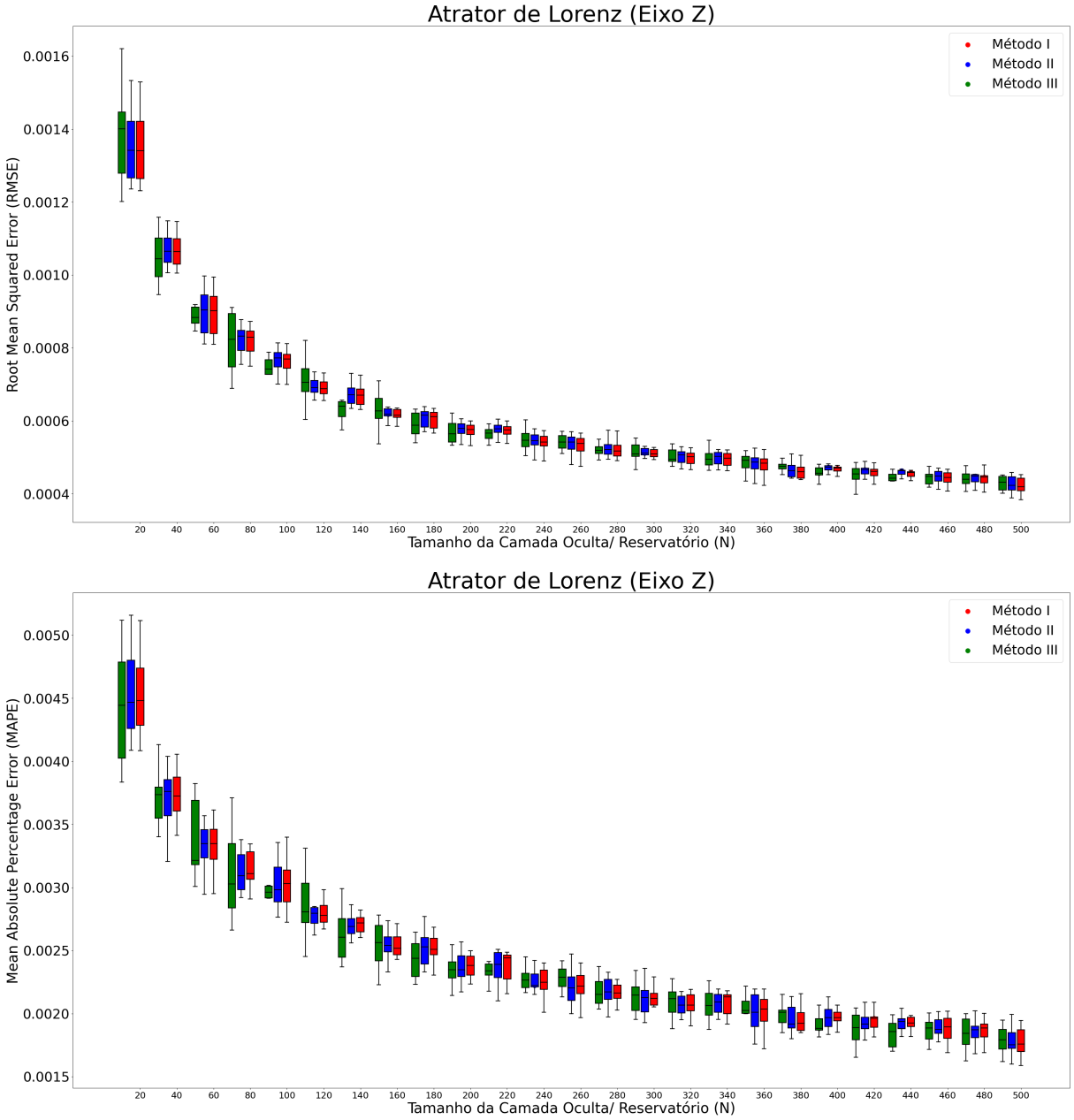

Figura 12 – Comparação entre os métodos I (vermelho), II (azul) e III (verde) para o eixo Z do Atrator de Lorenz. São apresentados os valores de RMSE (acima) e os valores do MAPE (abaixo).

O Atrator de Lorenz apresenta bom comportamento e demonstra ganhos substanciais do método I em detrimento dos métodos II e III para o eixo X além de ganhos modestos, mas com menor dispersão, para o eixo Y. Destaca-se o fator de o eixo Z apresentar alta similaridade de comportamento entre os modelos, bem como não apresentar, aparentemente, uma condição de equilíbrio estabelecida no que se refere à dinâmica das medidas de erro.

Tão importante quanto o formato das curvas é atentar-se à escala dos valores

apresentados em cada um dos eixos, para o eixo X do Atrator de Lorenz a diferença entre o método I e os métodos II e III, após estabilização do erro, aproxima-se de quatro unidades no gráfico de RMSE, várias ordens de grandeza superior ao apresentado nos gráficos de RMSE para os eixos Y e Z, o que evidencia a maior dificuldade de modelagem deste eixo bem como os ganhos que o modelo é capaz de obter através do aumento da camada oculta com e sem adição de ruído.

Os diferentes padrões apresentados, especialmente para sistemas multidimensionais, evidencia em parte a complexidade de se modelar tais sistemas caóticos tendo em vista a dinâmica de suas equações, as relações sinérgicas entra as variáveis, as transições de fase e os graus de liberdade apresentados.

3.2 Séries Temporais Financeiras

Realizamos as requisições de séries de preço de fechamento de ativo para Bitcoin (BTC), Ethereum (ETH), Cardano (ADA) e Dogecoin (DOGE), com frequência diária, de maior extensão possível via terminal Eikon da Refinitv e data limite definida em 14 de fevereiro de 2023. As séries obtidas foram temporalmente alinhadas, de modo que estaremos tratando dos preços de fechamento relativos aos mesmos dias e o vetor de dados terá o mesmo tamanho final.

A base de dados resultante contém as quatro séries temporais de dados de preço de fechamento (BTC, ETH, ADA e DOGE), cada qual com 1537 observações, sendo os 70% reservados para conjunto de dados de treino e os 30% posteriores para conjunto de validação conforme demonstrado na tabela 3.

Base de Dados	**Data de Ínicio**	**Data de Término**	**# Observações (Treino)**	**# Observações (Validação)**
Criptoativos	13/02/2018	14/02/2023	1076	461

Tabela 3 – Datas e quantidades de observações para o conjunto de dados de criptoativos.

Por questões de operacionalidade das ESNs, buscamos um conjunto de dados de entrada o mais normalizado o possível, isso é, estacionário de média $\mu = 0$ e desvio-padrão $\sigma = 0$, além de apresentar simetria considerável. Como primeira abordagem, trabalharemos com as séries de log-retornos em contrapartida das séries de preço.

As séries de log-retorno apresentam melhores estatísticas descritivas conforme os requisitos desejados para o bom funcionamento dos algoritmos. Sendo t o tempo de registro de um dado e $P(t)$ o preço de fechamento de dado ativo no tempo t, o log-retorno $r(t)$, relativo a este tempo, é dado por:

$$r(t) = log(P(t)) - log(P(t-1)). \tag{3.1}$$

As estatísticas descritivas demonstraram porém que as séries de log-retornos para os ativos selecionados apresentavam média distinta de zero, desvio-padrão distinto de um, forte assimetria para a direita e a existência de caudas pesadas (*fat-tails*).

Apesar de não previstas na Hipótese de Mercados Eficientes, bem como na Teoria de Portfólios Moderna, a existência de *fat-tails* é evidência de um fator de risco e volatilidade adicional constantemente encontrado em distribuições de séries de retornos de ativos financeiros. Essa temática constitui um tema de grande discussão no meio dos financistas e foi amplamente desenvolvida sob a óptica de sistemas caóticas, dinâmicas não lineares e, especialmente, fractalidade por Benoît B. Mandelbrot em seus artigos acerta da distribuição de preços de *commodities*, especialmente algodão, e ações [17, 18].

A assimetria à direita é resquício do fenômeno conhecido como *Boom das Cryptos*, onde diversos criptoativos experimentaram um aumento de preço significativo por conta da alta demanda por estes ativos atribuída a entrada de investidores institucionais e pessoas físicas em larga escala. Como as séries não são suficientemente longas, tais dados acabam sendo significativos na distribuição em termos de afastamento da média, de modo que há uma assimetria resultante evidente.

Como evidenciado pelas estatísticas descritivas apresentadas na tabela 4, as séries de log-retornos não apresentaram resultados totalmente satisfatórios, de modo que optou-se por uma normalização adicional via método de Yeo-Johnson [14]. Os resultados da normalização adicional, presentes na mesma tabela, apresentam melhoras significativas, especialmente no que concerne a correção da assimetria e padronização dos dados. Nas figuras 13, 14, 15 e 16 é possível visualizar as distribuições de log-retornos original (em azul) e após normalização via método de Yeo-Johnson (em vermelho).

Uma vez que as séries tratadas apresentam os requisitos de normalização desejados, partimos ao período de *learning*, para o qual utilizamos o mesmo procedimento adotado anteriormente com as séries temporais caóticas. Utilizando as mesmas configurações de rede e variando os hiperparâmetros sobre os mesmos intervalos. Similarmente, cada combinação de hiperparâmetros foi utilizada para dez ESNs distintas, de modo a obter-se estatísticas centrais acerca do comportamento dos erros (RMSE e MAPE).

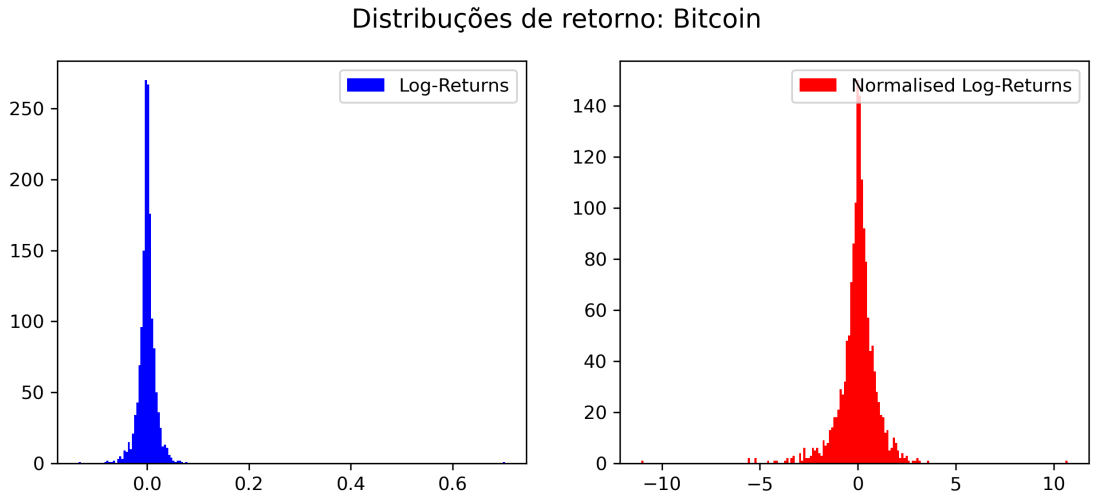

Figura 13 – Comparação visual da distribuição de log-retornos (em azul) e log-retornos normalizados via Yeo-Johnson (em vermelho) para Bitcoin (BTC).

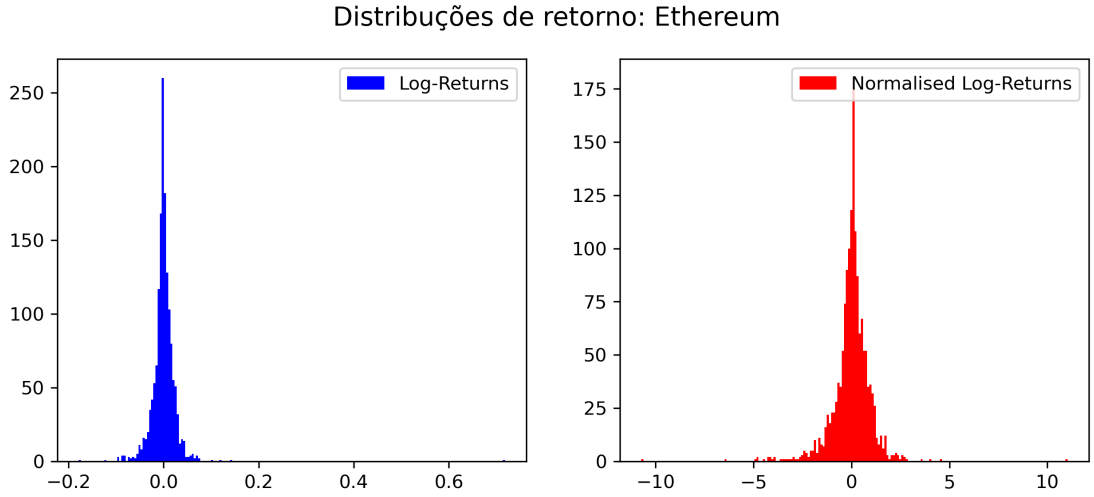

Figura 14 – Comparação visual da distribuição de log-retornos (em azul) e log-retornos normalisados via Yeo-Johnson (em vermelho) para Ethereum (ETH).

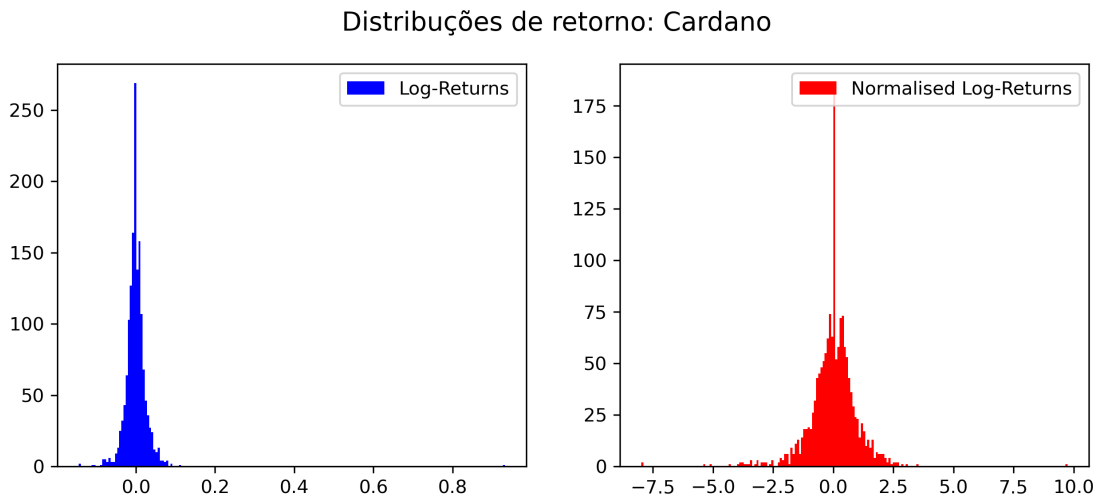

Figura 15 – Comparação visual da distribuição de log-retornos (em azul) e log-retornos normalisados via Yeo-Johnson (em vermelho) para Cardano (ADA).

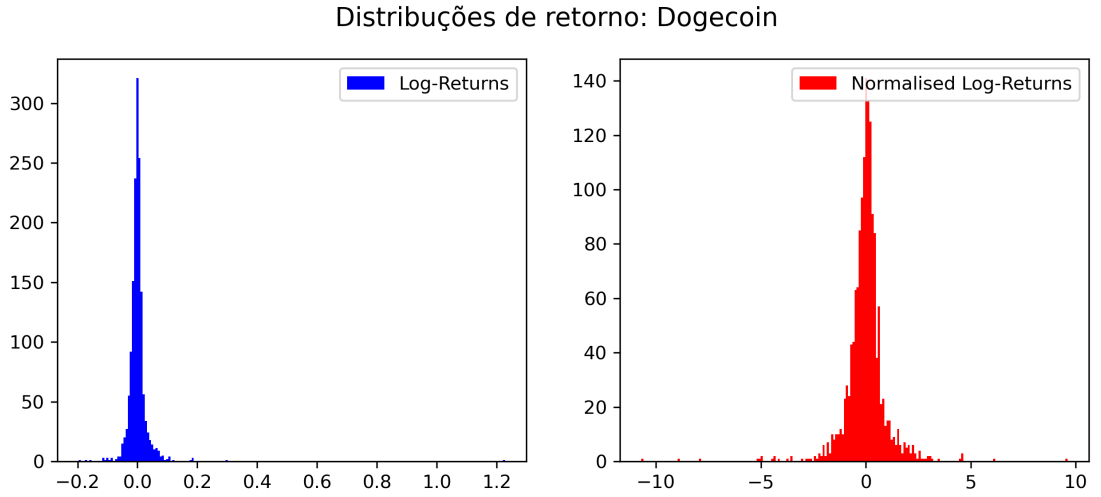

Figura 16 – Comparação visual da distribuição de log-retornos (em azul) e log-retornos normalisados via Yeo-Johnson (em vermelho) para Dogecoin (DOGE).

Estatística	BTC	**BTC (N)**	ETH	ETH (N)	ADA	**ADA (N)**	DOGE	**DOGE (N)**
Média	0.0	**0.0**	0.0	**0.0**	0.0	**0.0**	0.0	**0.0**
σ	$2.4\ 10^{-2}$	**1.0**	$2.9\ 10^{-2}$	**1.0**	$3.4\ 10^{-2}$	**1.0**	$4.2\ 10^{-2}$	**1.0**
Mínimo	-0.1	**-11.1**	-0.2	**-10.7**	-0.1	**-7.9**	0.2	**-10.7**
25%	$-6.9\ 10^{-3}$	**-0.3**	$-8.9\ 10^{-3}$	**-0.3**	$-1.2\ 10^{-2}$	**-0.4**	$-1.1\ 10^{-2}$	**-0.4**
50%	0.0	**0.05**	0.0	**0.05**	0.0	**0.07**	$-7.7\ 10^{-4}$	**-0.02**
75%	$7.3\ 10^{-3}$	**0.45**	$1.0\ 10^{-2}$	**0.5**	$1.1\ 10^{-2}$	**0.5**	$8.6\ 10^{-3}$	**0.4**
Máximo	0.703	**10.70**	0.718	**11.00**	0.932	**9.75**	1.23	**9.61**
Curtose	$4.5\ 10^{2}$	**21.3**	$2.6\ 10^{2}$	**21.8**	$3.9\ 10^{2}$	**13.0**	$4.9\ 10^{2}$	**24.5**
Assimetria	15.5	**-0.9**	10.1	**-0.8**	14.0	**-0.6**	16.9	**-1.0**

Tabela 4 – Principais estatísticas descritivas para as distribuições das séries de log-retornos para Bitcoin (BTC), Ethereum (ETH), Cardano (ADA) e Dogecoin (DOGE). Os valores em negrito, com uso da marcação (N) no cabeçalho indicam as séries tratadas via método de Yeo-Johnson. Valores de ordem igual ou menos a 10^{-5} foram deliberadamente considerados como zero.

Os dados obtidos com as 10^5 rodagens são organizados em *Heat Maps* de duas variáveis, onde cada ponto equivale à mediana dos valores presentes, tendo como premissa as variáveis em questão constantes e as outras variando. Para facilitação da visualização dos dados, optamos por expor na figura 17 o logaritmo em base dez dos valores de medianas obtidos.

Os *Heat Maps* resultantes, são exemplificados pelo resultado obtido para Bitcoin (BTC) na figura 17 abaixo. Em contrapartida aos gráficos obtidos para as séries temporais caóticas, os gráficos relativos as combinações de hiperparâmetros para criptoativos apresentam maior grau de dificuldade de interpretação. Os gradientes difusos dificultam o processo de leitura de hiperparâmetros ótimos.

Os picos e vales, muito bem definidos nas séries anteriormente utilizadas, apresentam-se na forma de áreas ou de múltiplos pontos. Nos casos como os da imagem (iv) e (v), situadas à direita no centro e à esquerda na terceira linha, há áreas de menor erro. A figura (vi), situada à direita na última linha, mostra áreas de maior e menor erro mediano se intercalando em padrões de difícil leitura. Como esperado, as séries temporais financeiras são relativamente complexas do ponto de vista de modelagem, há dificuldade em obter-se modelos concisos e precisos, os valores assumidos para cada criptoativo são apresentados na tabela 5.

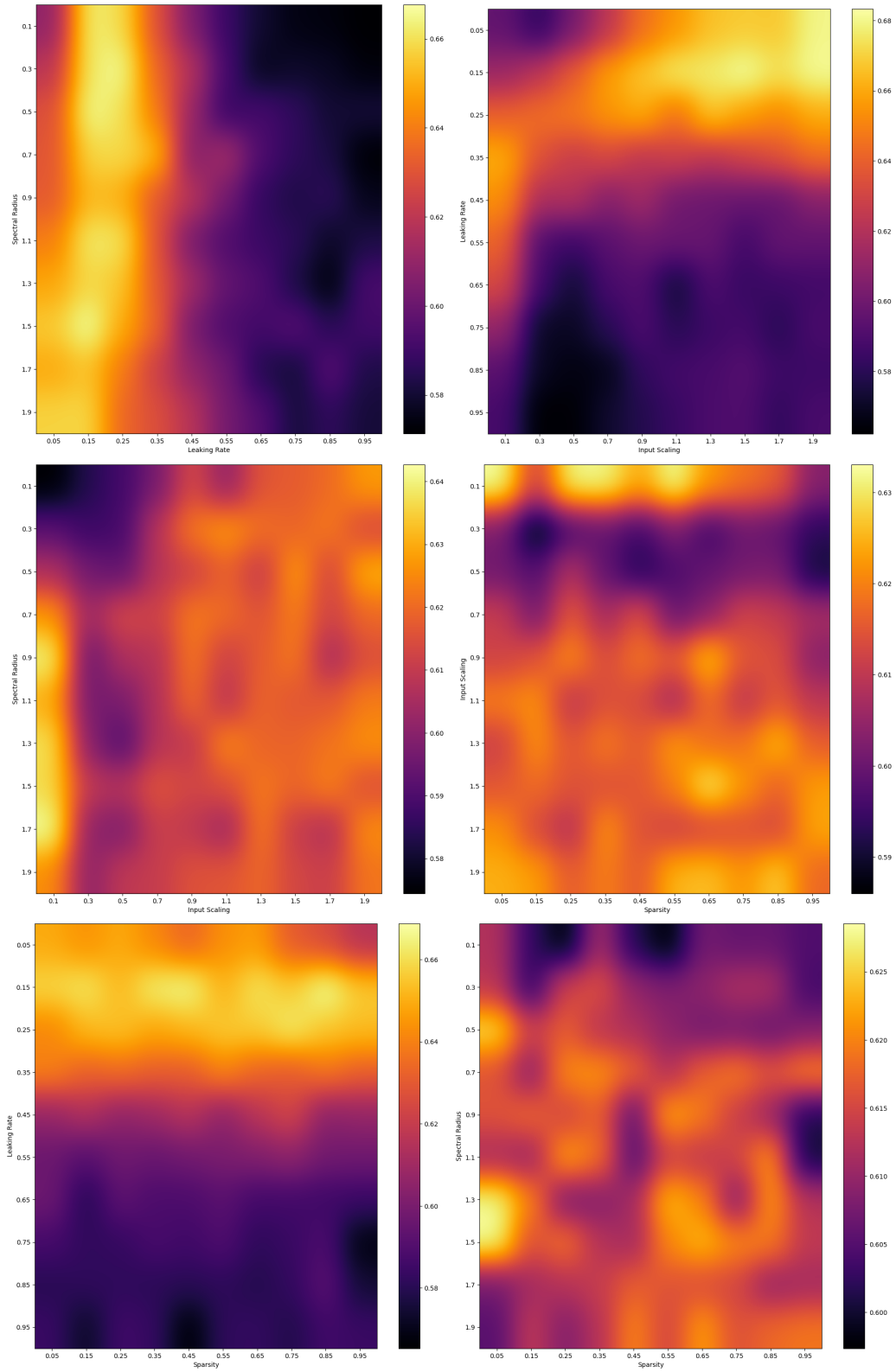

Figura 17 – Configurações de *Heat Maps* com medianas do MAPE em escala log10, para Bitcoin (BTC) evidenciando comportamento das ESNs em conjunto de teste face alterações nos hiperparâmetros

Hiperparâmetros	BTC	ETH	ADA	DOGE
Raio Espectral	0.10	0.10	0.10	0.10
Escala da Entrada	0.10	0.10	0.10	0.10
Esparsidade	0.95	0.95	0.95	0.95
Vazamento Espectral	0.95	0.95	0.15	0.95

Tabela 5 – Hiperparâmetros ótimos aproximados obtidos via análise de Heat Maps para séries de log-retornos de criptoativos.

A tabela 5 apresenta os resultados aproximados obtidos via *Heat Maps* para os hiperparâmetros das criptomoedas analisadas. Os *Heat Maps* dos ativos financeiros apresentam dinâmicas de mediana de erros de leitura mais complexa, isso se deve em parte à presença de ruído (estocasticidade) na série mas também à capacidade elevada de eficiência em termos de mercado destes ativos, de modo que oportunidades de modelar comportamento ou tendência de preços são rapidamente arbitradas pelos agentes econômicos envolvidos. O sistema acaba por minimizar a capacidade de quaisquer agentes de inferir com muita precisão os estados seguintes do mesmo, de modo que é difícil ao modelo, mesmo em conjunto de treino, de estabelecer gradientes suaves em torno de um único mínimo de erro. Os ativos convergiram em termos de hiperparâmetros ótimos aproximados, demonstrando uma certa preferência, em geral, para este modelo, à exceção da taxa de vazamento espectral de Cardano (ADA) que desviou-se dos demais, como explicitado na Equação 1.2, uma taxa de vazamento espectral menor equivale a uma tendência geral de retenção ou conservação de estados anteriores, pode-se então concluir de antemão que o Cardano difere essencialmente das outras séries temporais financeiras ao apresentar tendências de preço mais acentuadas, no que os financistas apontam como o chamado *fator momento*.

Para certificar-nos que os modelos obtidos via processo de *learning* explicavam todas as relações possivelmente existentes dentro do conjunto de treino, realizamos a verificação dos modelos via função de autocorrelação (ACF) e autocorrelação parcial (PACF). A ACF constitui uma representação gráfica da autocorrelação em relação à defasem (chamada de *lag*). A autocorrelação parcial por sua vez, realiza procedimento similar descontando correlações implícitas adjacentes, de modo a mostrar apenas o valor da correlação *de facto*.

Nas figuras 18, 19, 20 e 21 é possível visualizar os gráficos de autocorrelação (ACF) e autocorrelação parcial (PACF) para as séries de log-retornos normalizados dos ativos analisados, com *lag* máximo de 25. Logo em sequência, nas figuras 22, 23, 24 e 25 é possível verificar o ACF e PACF para o resíduo obtido da diferença entre as mesmas séries e as séries de valores preditos, considerando melhor e pior modelo.

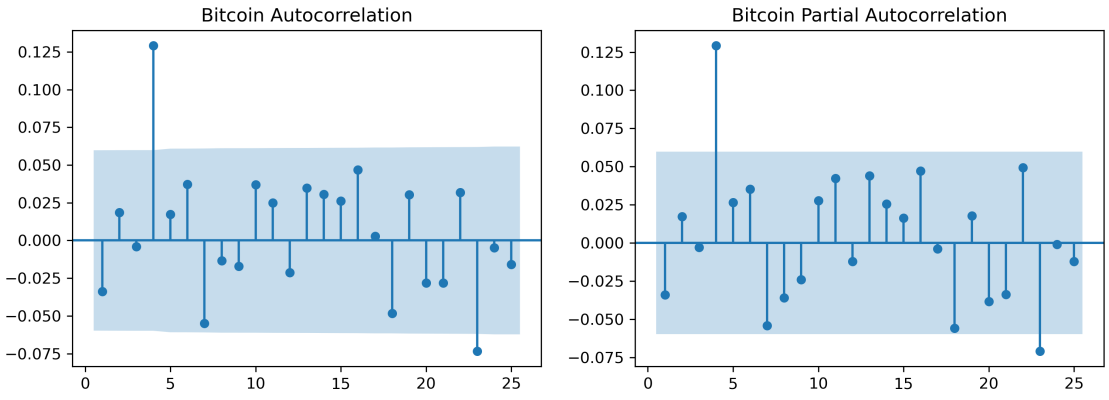

Figura 18 – A coluna da esquerda representa os resultados de autocorrelação (ACF) e a da direita os de autocorrelação parcial (PACF) para o conjunto de dados de treino de log-retornos normalizados de Bitcoin (BTC). Quantidade de *lags* observados foi de 25.

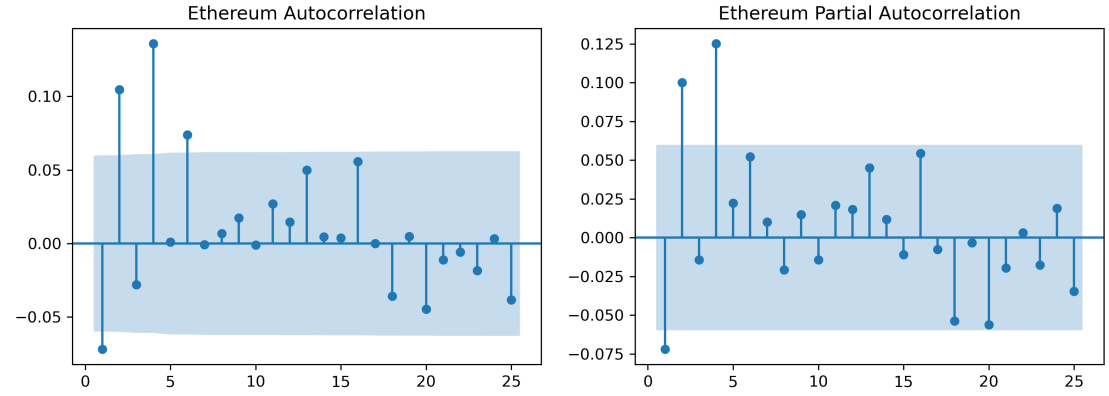

Figura 19 – A coluna da esquerda representa os resultados de autocorrelação (ACF) e a da direita os de autocorrelação parcial (PACF) para o conjunto de dados de treino de log-retornos normalizados de Ethereum (ETH). Quantidade de *lags* observados foi de 25.

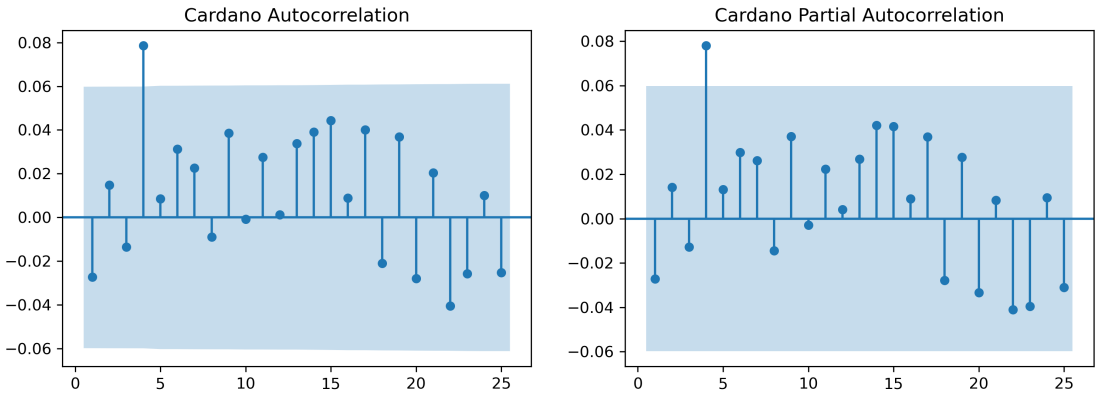

Figura 20 – A coluna da esquerda representa os resultados de autocorrelação (ACF) e a da direita os de autocorrelação parcial (PACF) para o conjunto de dados de treino de log-retornos normalizados de Cardano (ADA). Quantidade de *lags* observados foi de 25.

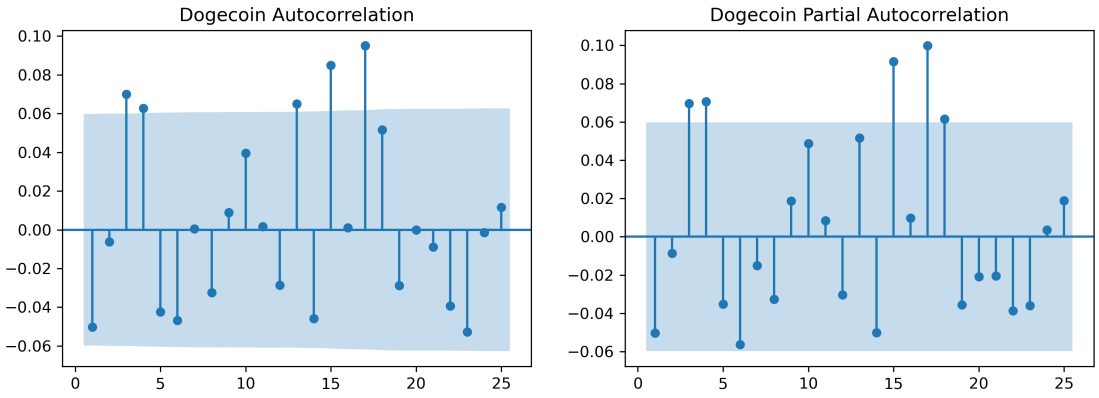

Figura 21 – A coluna da esquerda representa os resultados de autocorrelação (ACF) e a da direita os de autocorrelação parcial (PACF) para o conjunto de dados de treino de log-retornos normalizados de Dogecoin (DOGE). Quantidade de *lags* observados foi de 25.

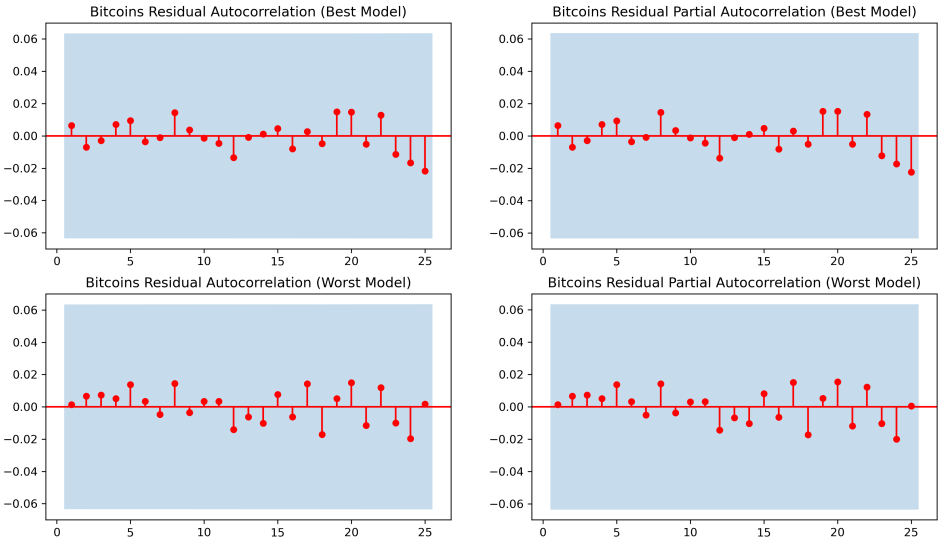

Figura 22 – A coluna da esquerda representa os resultados de autocorrelação (ACF) e a da direita os de autocorrelação parcial (PACF) para cada os resíduos dos melhores e piores modelos, em condição pré-definida, os modelos em questão são relativos às séries do Bitcoin (BTC). Quantidade de *lags* observados foi de 25.

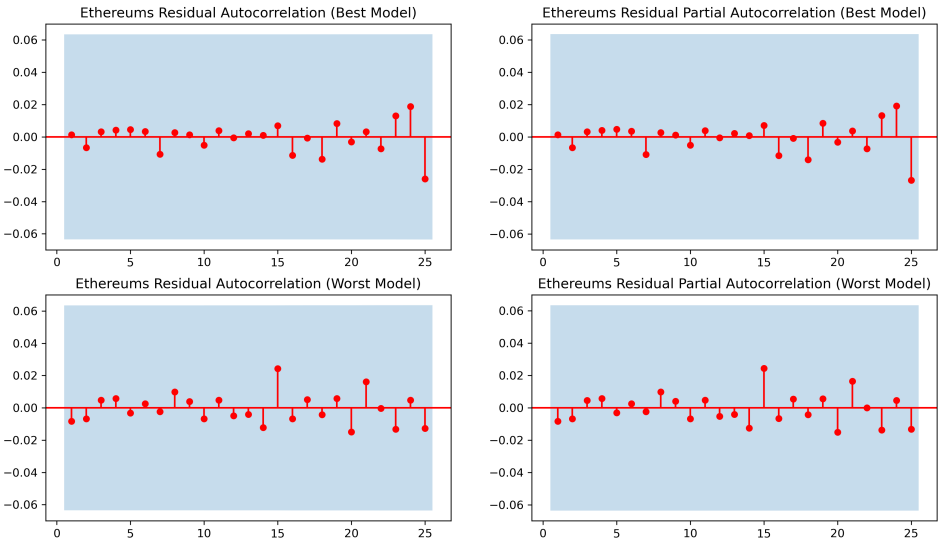

Figura 23 – A coluna da esquerda representa os resultados de autocorrelação (ACF) e a da direita os de autocorrelação parcial (PACF) para cada os resíduos dos melhores e piores modelos, em condição pré-definida, os modelos em questão são relativos às séries do Ethereum (ETH). Quantidade de *lags* observados foi de 25.

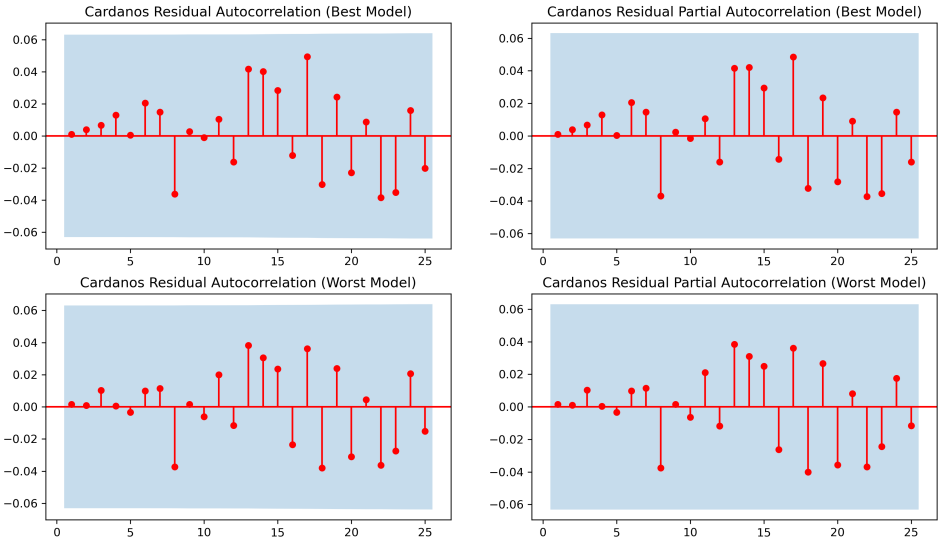

Figura 24 – A coluna da esquerda representa os resultados de autocorrelação (ACF) e a da direita os de autocorrelação parcial (PACF) para cada os resíduos dos melhores e piores modelos, em condição pré-definida, os modelos em questão são relativos às séries do Cardano (ADA). Quantidade de *lags* observados foi de 25.

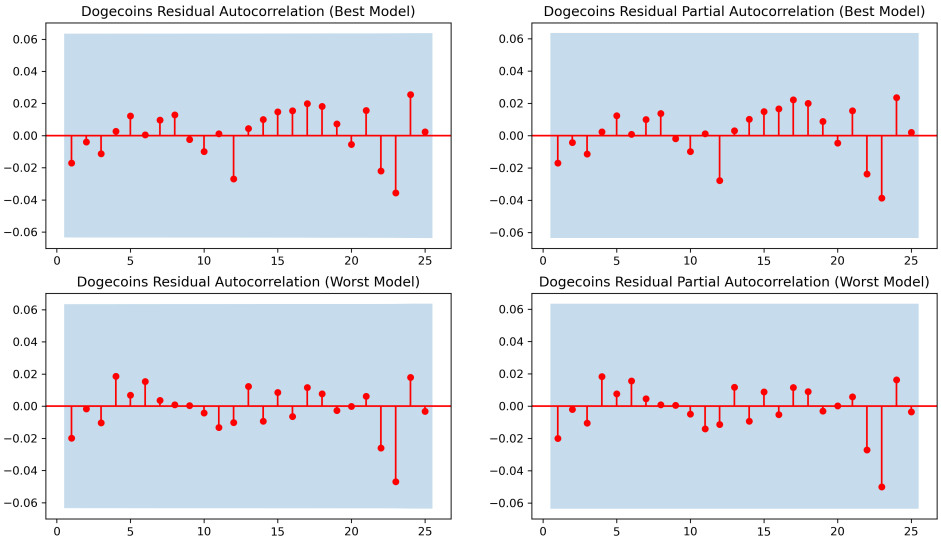

Figura 25 – A coluna da esquerda representa os resultados de autocorrelação (ACF) e a da direita os de autocorrelação parcial (PACF) para cada os resíduos dos melhores e piores modelos, em condição pré-definida, os modelos em questão são relativos às séries do Dogecoin (DOGE). Quantidade de *lags* observados foi de 25.

Análises de gráficos de ACF e PACF permitem compreender visualmente se existe alguma forma de relação relevante entre as variáveis e seus valores anteriores em um forma de memória de processo. Valores plotados acima ou abaixo da área hachurada indicam correlações para as quais não se pode afirmar a hipótese nula dos testes de ACF e PACF, isso é, são significativamente distintas de zero. Definimos o *lag* máximo de interesse como 25 e identificamos as principais componentes relevantes para o modelo via ACF e PACF para cada um dos ativos, como mostrado nas figuras 18, 19, 20 e 21. Tendo os hiperparâmetros ótimos, iniciamos o processo de aumentar paulatinamente o tamanho de janela do modelo, inicialmente definido de forma padrão como $n = 1$.

Conforme aumentávamos o tamanho de janela utilizada, subtraíamos os valores obtidos através da predição das ESNs sobre o próprio conjunto de treinamento dos valores ali presentes, avaliávamos então os gráficos de ACF e PACF destes resíduos obtidos. Como mais de um modelo era gerado para as mesmas condições, os gráficos eram feitos para o melhor e pior modelo (em termos de RMSE e MAPE total), considerando mesmo tamanho de camada oculta.

Ao final do processo, considerávamos então que o modelo havia explicado com sucesso as relações de dependência dos conjuntos. Os resultados de ACF e PACF para os piores e melhores modelos estão expostos nas figuras 22, 23, 24 e 25. Os resultados obtidos podem ser visualizados na tabela 6.

Tamanho de janela	**BTC**	**ETH**	**ADA**	**DOGE**
n	24	22	4	20

Tabela 6 – Tamanhos de janela ótimos para cada conjunto de dados.

Tendo validado nosso modelo de ESN quanto às capacidades de explicar as séries de dados e suas relações e selecionados os parâmetros ótimos, que minimizaram os erros totais no conjunto de treino, passamos ao conjunto de validação. Nossos principais objetivos se concentram em (I) mensurar os erros totais do modelo sobre a base de dados de validação com relação a quantidade de neurônios na camada oculta e (II) comparar o modelo contra um preditor lógico e simples, neste caso o preditor ingênuo.

O preditor ingênuo é dado de tal modo que, sendo S_t a observação de uma série temporal qualquer no tempo t, temos que:

$$S_{t+1} = S_t \tag{3.2}$$

Apesar de simples, o preditor ingênuo têm forte embasamento na Hipótese de Mercado Eficiente, para a qual, em sua hipótese mais fraca, os preços dos ativos já espelhariam todas as informações disponíveis publicamente bem como todas as expectativas

envolvidas na precificação, desta forma, modelos matemáticos não deveriam ser melhores que simples predições baseadas na própria série. Outro fator envolvido é a existência de processos retroalimentados, onde vemos valorização e desvalorização de ativos de forma sequencial. Há também a já documentada existência do chamado fator momento, onde ativos com certa tendência de crescimento ou queda tendem a continuar assim por um período suficientemente grande.

Assim, o uso do preditor ingênuo é justificável e engloba fatores suficientes para validar o modelo em termos de utilidade. O modelo é considerado válido quando apresentar erro de validação (RMSE e MAPE) inferior ao do preditor ingênuo. As linhas d'água do nosso modelo, que constituem os *benchmarks* de utilidade são portanto os erros totais do conjunto de validação contra ele mesmo um passo à frente, os valores calculados são mostrados na tabela 7.

Métricas (Naive)	**BTC**	**ETH**	**ADA**	**DOGE**
MSE	1.5102	1.3507	1.2635	1.4931
RMSE	0.8406	0.8210	0.8054	0.8198
MAPE	4.9218	5.5987	8.1154	7.5029

Tabela 7 – Medidas de erros, RMSE e MAPE, relativas ao conjunto de dados de validação e seu preditor ingênuo.

São computados os valores de RMSE e MAPE para cada uma das séries de dados de validação. Variando-se o número de neurônios na camada oculta, mas mantendo-se fixos os hiperparâmetros ótimos bem como o tamanho de janela ótima para cada ativo, os resultados obtidos para as séries de log-retornos normalizados de BTC, ETH, ADA e DOGE estão expostos nas figuras 26, 27, 28 e 29 respectivamente.

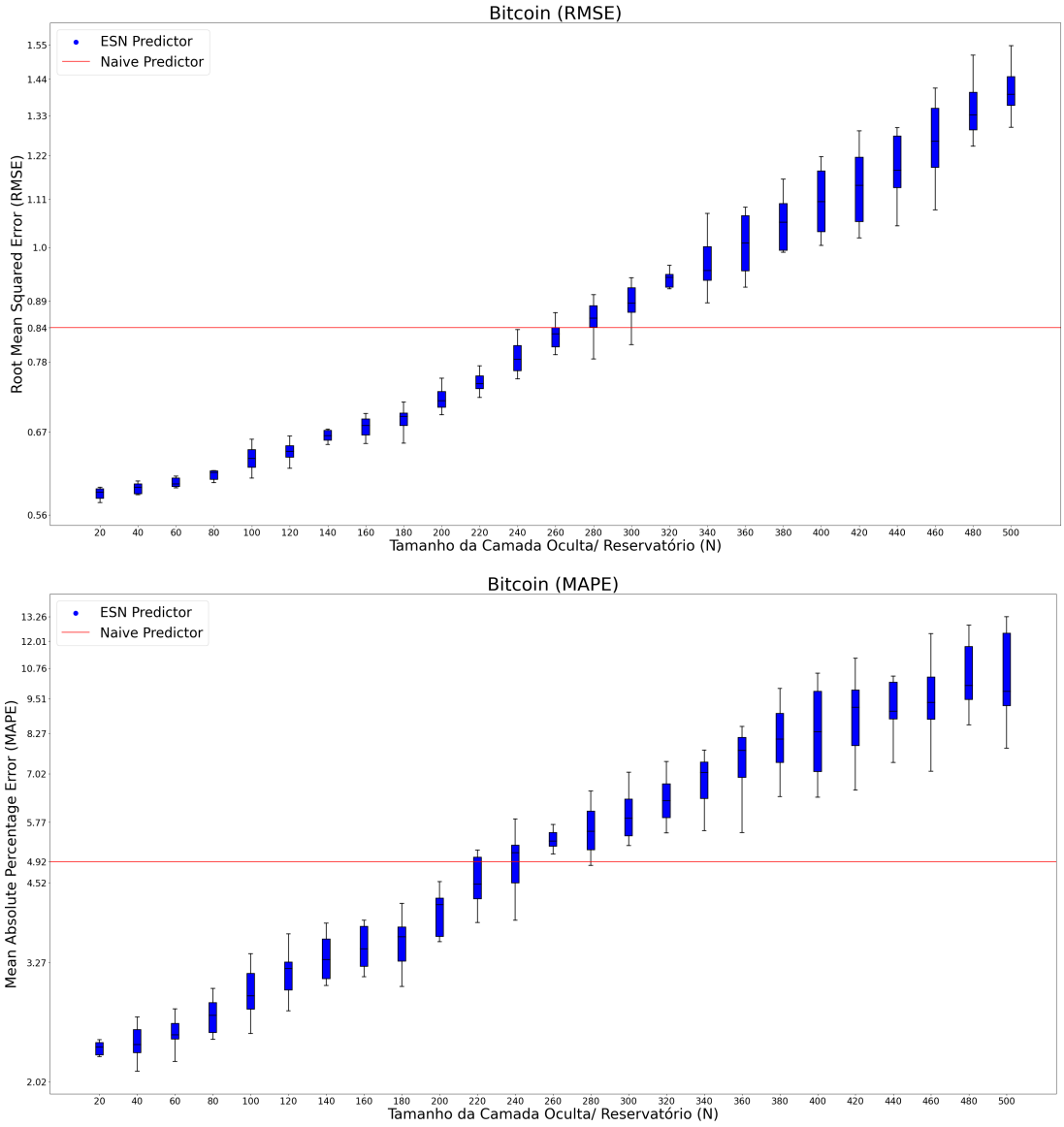

Figura 26 – Comparação em escala logarítmica de RMSE (esquerda) e MAPE (direita) para modelo (azul) vs. preditor ingênuo (linha vermelha) para predições um passo a frente na série de retornos de BTC. Considerando ESNs como camada oculta N entre 20 e 500.

Para o caso do Bitcoin (BTC), na figura 26 é possível visualizar uma curva quase exponencial, com relação ao aumento de erro, perceptível tendo em vista uma tendência linear na escala logarítmica. Esse comportamento exemplifica o fato de modelos com menores camadas ocultas (aqueles com valores baixos de N) apresentam desempenho assaz superior, em termos de erros, quando comparados a modelos de maior camada oculta. Tal fato corrobora o fato de uma maior eficiência de mercado na negociação do ativo, de modo que modelos de grande camada oculta acabam assumindo conexões espúrias em detrimento de conexões fundamentais associadas a padrões de relevância estatística.

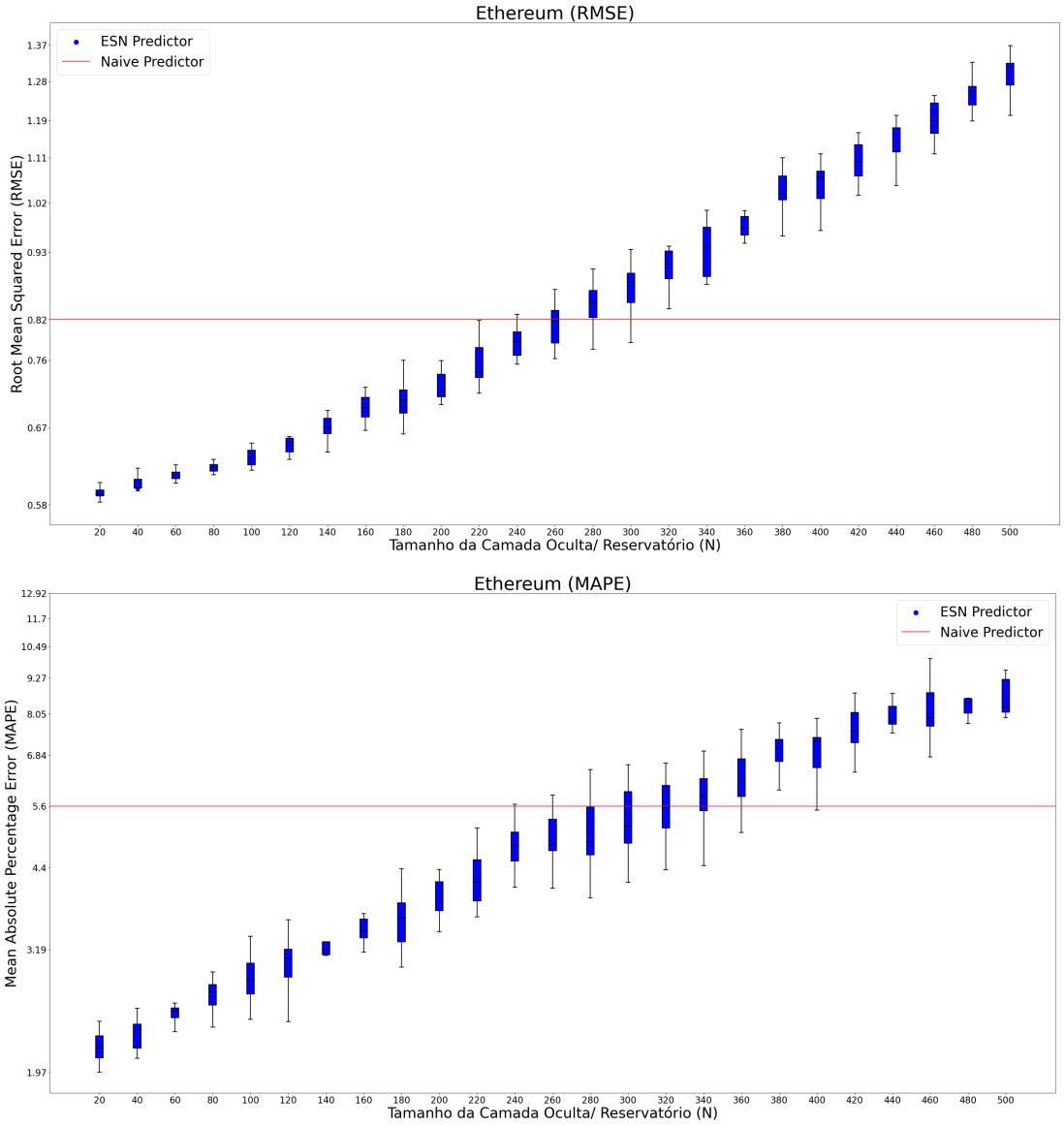

Figura 27 – Comparação em escala logarítmica de RMSE (esquerda) e MAPE (direita) para modelo (azul) vs. preditor ingênuo (linha vermelha) para predições um passo a frente na série de retornos de ETH. Considerando ESNs como camada oculta N entre 20 e 500.

Como apresentado na figura 27, o comportamento do Ethereum (ETH) se assemelha em muito com o apresentado pelo Bitcoin em termos de formato de curva em relação ao tamanho da camada oculta. Atribuímos este comportamento à similaridade com relação ao Bitcoin, dado que ambos figuram entre as maiores captações de mercado em criptoativos, em termos de captação e negociação de modo que ambos compartilham de uma certa eficiência de mercado.

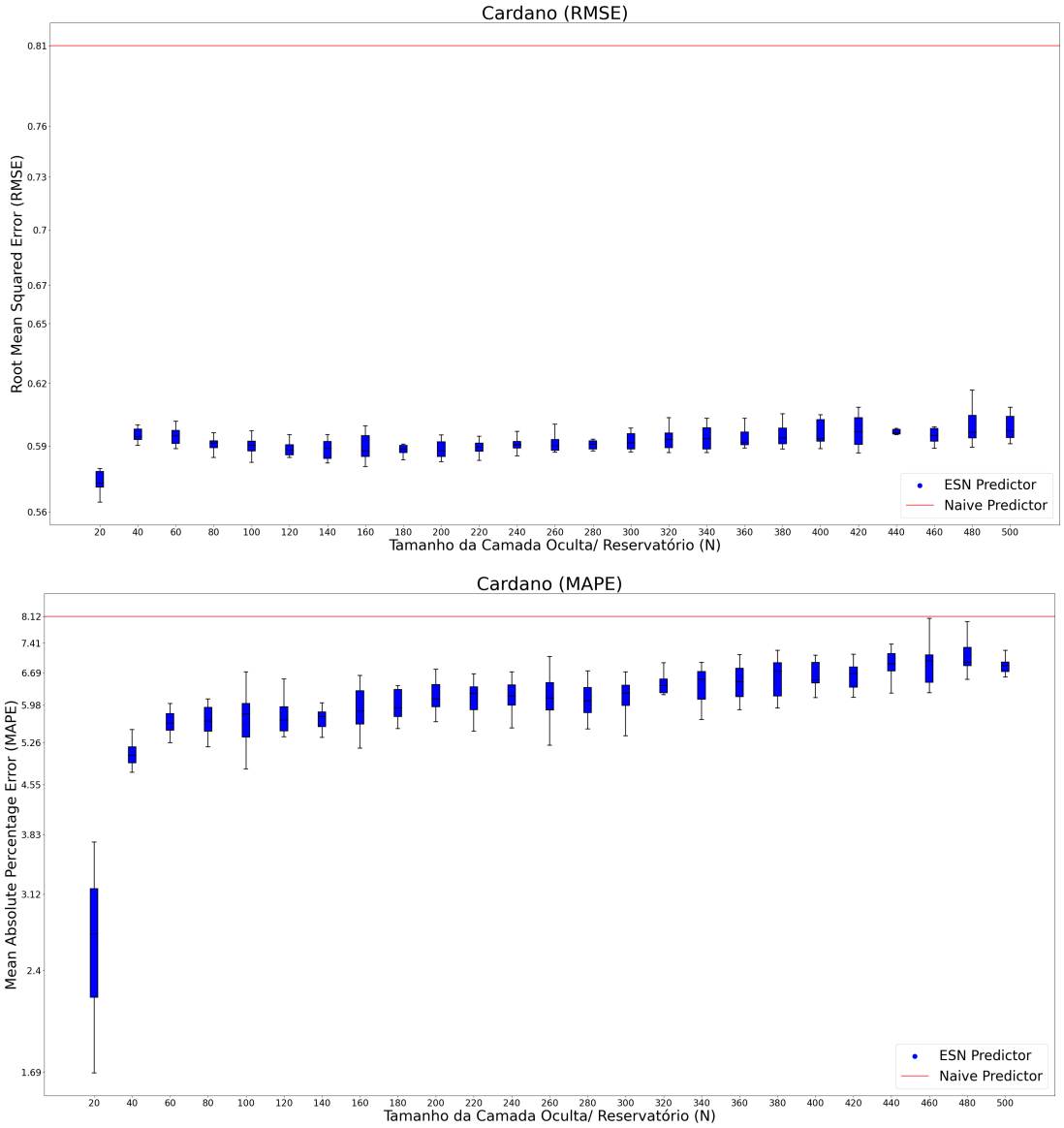

Figura 28 – Comparação em escala logarítmica de RMSE (esquerda) e MAPE (direita) para modelo (azul) vs. preditor ingênuo (linha vermelha) para predições um passo a frente na série de retornos de ADA. Considerando ESNs como camada oculta N entre 20 e 500.

Em termos de eficiência de modelo, Cardano (ADA) se destaca por seus excelentes resultados apresentados na figura 28. Destacamos também que Cardano teve um tamanho de janela ótimo substancialmente inferior àqueles necessários aos outros ativos, com $n = 4$. Tendo um incremento substancial de erros na transição entre os modelos com tamanho de camada oculta $N = 20$ e $N = 40$, os modelos posteriores apresentaram resultados consistentemente abaixo do preditor ingênuo, o que demonstra que modelos simples são capazes de capturar padrões estatísticos de curta memória (tamanho de janela $n = 4$) que apresentam bons resultados de forma constante.

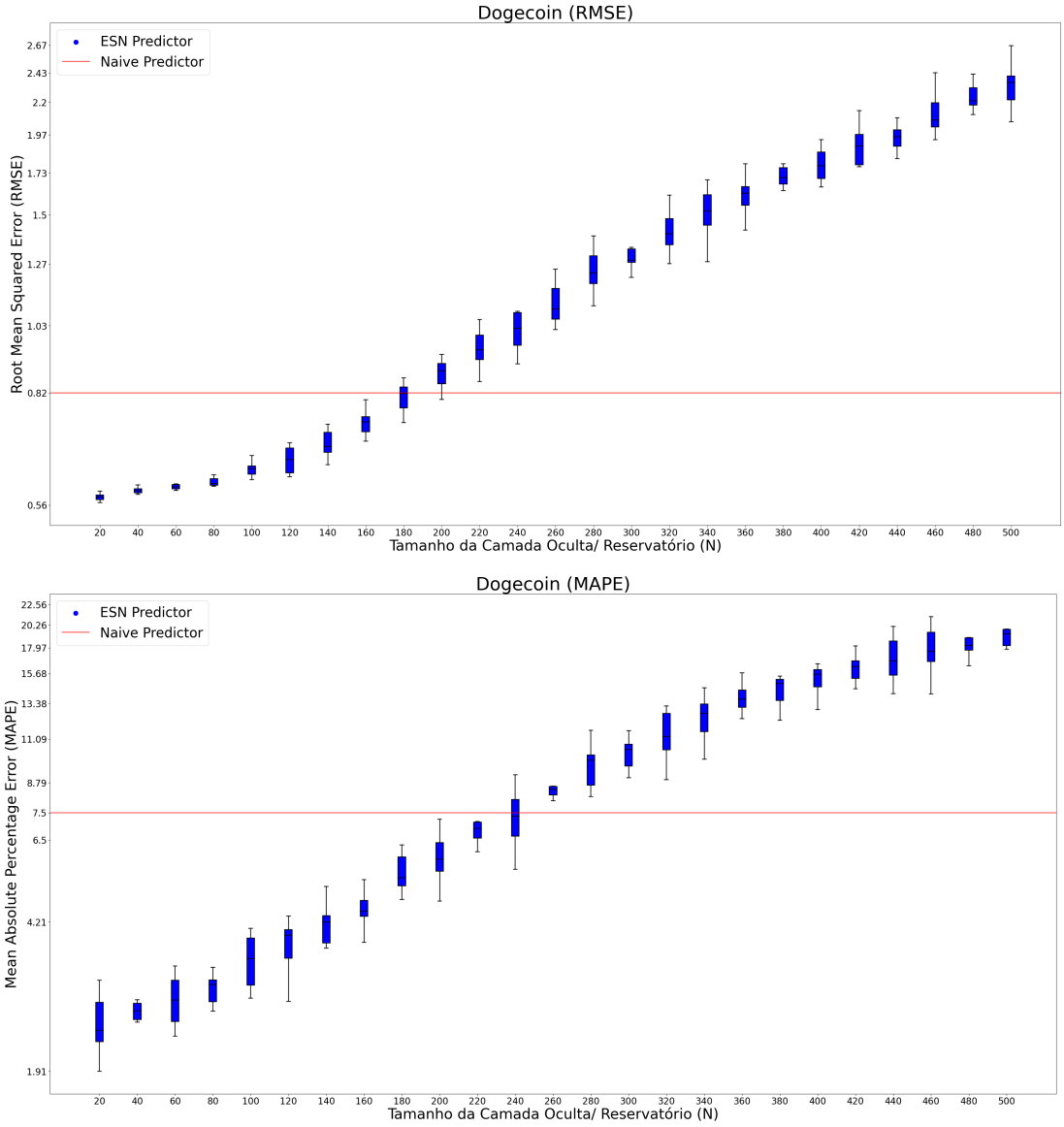

Figura 29 – Comparação em escala logarítmica de RMSE (esquerda) e MAPE (direita) para modelo (azul) vs. preditor ingênuo (linha vermelha) para predições um passo a frente na série de retornos de DOGE. Considerando ESNs como camada oculta N entre 20 e 500.

O comportamento geral do modelo aplicado a Dogecoin (DOGE) apresenta similaridades com os apresentados por Bitcoin (BTC) e Ethereum (ETH), demonstrando menores variações relativas, em termos de erros para modelos com tamanho de camada oculta pequeno ($N < 120$) bem como modelos de tamanho de camada oculta elevada ($N > 400$), assumindo uma curva do tipo "S". É importante destacar que mesmo com grande captação de mercado e volume de negociação, Dogecoin é um projeto de objetivos bastante distintos em relação a outras criptomoedas, sendo referenciado na comunidade como uma *meme coin*. Apesar de aparentemente eficiente, este ativo é mais suscetível a fenômenos coletivos como bolhas e *crashs*, bem como manipulações de mercado.

De forma geral, os modelos mais simples (com valores de N relativamente baixos). Superaram o preditor ingênuo em ambas as medidas de erro utilizadas. Isto se deve em parte ao fato de que menores reservatórios são capazes de melhor generalizar sem *overfitting* as relações essenciais entre as variáveis dos padrões de entrada, de modo a estabelecer um padrão estatístico relevante que é capaz de predizer com algum grau de certeza o próximo movimento do ativo. Reservatórios grandes por sua vez tendem a modelar padrões mais sofisticados porém por vezes espúrios, abrindo mão da capacidade de generalização.

Conforme esperado através de conhecimento empírico de mercado, Bitcoin (BTC) e Ethereum (ETH) obtiveram curvas de características semelhantes, sendo os dois maiores ativos financeiros deste mercado em termos de *market cap*. Há uma tendência de movimentos correlacionados. São projetos consolidados, de mercados mais eficientes e pouca oportunidade de arbitragem de preço.

Dogecoin (DOGE) apresenta leves alterações com relação a sua curva de erros no conjunto de dados de validação. Apesar de focada em outro nicho de mercado é um projeto igualmente antigo e com muita movimentação. Cardano (ADA) acabou mostrando-se um ativo diferenciado quando analisado por vias do nosso modelo em parte por requerer um tamanho de janelas bastante enxuto quando comparado com seus similares ($n = 4$), mas especialmente pelo seu excelente desempenho no conjunto de dados de validação, superando o preditor ingênuo constantemente e com amplas margens para valores de N pequeno, acreditamos que por tratar-se de uma criptomoeda de menor captação e movimentação Cardano torna-se suscetível a movimentações de mercado mais previsíveis, abrindo-se uma possibilidade real de arbitragem de preço de ativo por meios algorítmicos.

4 CONCLUSÃO

Este capítulo destina-se à conclusão deste trabalho de conclusão de curso em Engenharia Física e versará sobre as conclusões finais acerca do modelo, tendo em vista a discussão elaborada no decorrer deste documento, bem como os próximos passos sugeridos. Dedicaremos também breves parágrafos aos agradecimentos técnicos, premiações, palestras e à implementação deste projeto nas gestoras.

4.1 Comentários Finais

Desde o início das atividades constituintes deste trabalho, visamos a criação de um produto, na forma de um algoritmo, aplicável ao mercado de fundos de investimentos, com foco especial ao mercado nacional. A premissa para as definições de projeto e objetivos deste trabalho deu-se através que uma grande troca de experiências e intercâmbio de ideias com praticantes, *traders*, professores de economia, assessores de investimento e gestores certificados com grande experiência na gestão de fundos de renda variável e multimercados. Constatamos uma grande oportunidade aberta para o desenvolvimento de produtos estruturados quantitativos e algorítmicos em gestão de recursos, tendo conhecido duas grandes gestoras independentes de Porto Alegre, vemos uma grande movimentação em torno da busca por melhores provedores de dados de mercado (nacional e internacional), infraestrutura de armazenamento em nuvem e principalmente capital humano qualificado, capaz de unir conhecimento técnico em análise e modelagem de dados através de métodos quantitativos e de inteligência artificial com grande *expertise* no mercado financeiro local.

Buscamos uma forma de processar uma grande quantidade de dados em *streaming*, de modo a arbitrar quaisquer descompassos, bem como utilizar-se de padrões estatísticos recorrentes da série temporal de modo a obter *alpha*, isso é, retorno sem agregação de risco. Nosso conhecimento empírico de mercado demonstra também que estratégias híbridas com o uso de variáveis de balanço financeiro, macroeconômicos, juros, câmbio e sentimento de mercado, têm grande potencial em termos de performance esperada. Nosso produto final foi inicialmente pensado para aplicação em quaisquer séries temporais relativos a renda variável, para o caso do Brasil isto inclui ações, *units*, *Brazilian Depositary Receipts* (BDRs), Fundos de Investimentos Imobiliários (FIIs), ETFs e criptoativos, este último, constituiu o escopo deste trabalho. Muitos destes ativos apresentam suficiente liquidez, na forma de alto volume de negociação diária e baixo *bid-ask spread* apresentado no *book* de negociação, fenômeno que se deve em parte à popularização e acessibilidade destes investimentos, cada vez mais presentes na vida da pessoa física brasileira, em taxas acessíveis.

Entendemos que as ESNs se enquadram, do ponto de vista técnico, nas demandas do nosso projeto. Primeiramente, por serem modelos de paradigma e performance consolidada na literatura, com vasta produção científica, boa capacidade preditiva e tempo de execução esperado notadamente baixo além de boa capacidade de generalização face à ruído em parte por conta das características da regressão linear utilizada. Uma vez que o modelo é treinado, sua execução é praticamente instantânea, especialmente considerando períodos de operação de ordens superiores a um minuto, ou seja, é possível predizer o próximo valor esperado e disparar uma ordem de negociação em tempo suficientemente hábil do ponto de visto operacional. Por fim, as ESNs são atrativas quando vistas através do prisma da Econofísica, pela sua grande capacidade de predição, amplamente citada na literatura, face a sistemas dinâmicos não-lineares com características caóticas e complexas, pontos estes coincidentes com características presentes dos mercados, bastante enfatizadas por físicos envolvidos na área.

Os primeiros testes, foram deliberadamente realizados tendo em vista séries caóticas unidimensionais e multidimensionais. Os objetivos de tais testes eram claros, avaliar a capacidade das redes neurais implementadas em modelar dados em uma ou mais dimensões, verificar seu desempenho face à aplicação de ruído de modo a compreender a capacidade de generalização das redes neurais bem como verificar o erro de validação destes modelos, avaliamos o Mackey-Glass 17 ($\tau = 17$), Mackey-Glass 22 ($\tau = 22$), o Mapa de Hénon e o Atrator de Lorenz. Os resultados obtidos demonstram uma excelente capacidade de generalização de tais funções, otimizamos os hiperparâmetros (taxa de vazamento espectral, raio espectral, esparsidade e escala dos dados de entrado) via análise dos *Heat Maps*, a visualização gráfica dos resultados permite identificar gradientes claros de hiperparâmetros que levavam à um estado de maior otimização do modelo, durante o processo mantivemos fixos o tamanho da janela de entrada ($n = 1$) e o tamanho da camada oculta ($N = 20$) e utilizando-nos de mais de um modelo pseudo-aleatório de ESN, levantamos os erros de validação para cada conjunto. Nossa análise quanto ao comportamento dos erros (MAPE e RMSE) levou-nos a dissecar tais erros por eixo (X, Y e Z) de modo a melhor compreender as dinâmicas presentes em tais resultados.

Nossos resultados evidenciaram que a característica pseudo-aleatória das matrizes de pesos, afeta o desempenho e resultado esperado, em termos de erro, nos conjuntos de testes e validação, todavia, não diverge suficientemente a ponto de que um comportamento comum não seja evidente. Apesar de que os valores tenham sido empiricamente determinados, são suficientemente representativos à ponto de distorcer as séries temporais apresentadas, os métodos I, II e III foram gerados de modo a verificar a robustez da solução, ou seja, a capacidade de generalização das ESNs face à adição de tais ruídos. Os resultados para o conjunto de testes não demonstraram variações perceptíveis via *Heat Maps* de hiperparâmetros ótimos de modo que a diferença entre os métodos II e III, tato em termos de conjunto de testes como de validação, deu-se exclusivamente por conta das diferenças

entre os modelos em termos das suas matrizes de pesos.

O método I é insuperável, como esperado, em todas as séries testadas, tanto em termos de RMSE quanto de MAPE. A dinâmica do método I em contrapartida dos métodos II e III não é evidente em todos os eixos dos sistemas, sendo o Atrator de Lorenz o mais claro exemplo, neste sistema é possível perceber uma clara e relevante superioridade do modelo no eixo X em face de uma alta similaridade quando avaliado no eixo Z. Ruídos gaussianos de mesma intensidade (desvio-padrão equivalente), quando aplicado sobre diferentes dimensões da série de dados normalizada acaba por produzir efeitos totalmente distintos, ora tais efeitos mostram-se aparentemente irrelevante em termos de impacto nos resultados dos métodos, ora estes ruídos acarretam erros de ordem significativos evidenciados na comparação dos métodos. De forma geral todos os modelos convergem ou aproximam-se de uma convergência para os dados de validação conforme aumentamos o tamanho da camada oculta (N) o que se deve ao fato de que, mesmo possuindo características caóticas, essas séries apresentam padrões de comportamento que podem ser encontrados pelo algoritmo com certa facilidade. Ademais, as ESNs implementadas demonstraram capacidade de convergência de solução, tempo de execução conciso e foram capazes de processar sistemas com mais de uma dimensão (mais de uma observação por tamanho de janela), o resultado foi considerado satisfatório e passamos à análise de ativos reais de modo a, definitivamente, validar o produto.

Adquirimos os dados de valor de fechamento, em dólares, com frequência de amostragem diária via provedor Eikon/Refinitiv, apesar de o histórico de criptoativos ser mais recente quando comparado a *commodities* e *securities* há uma quantidade suficiente de dados para fins práticos, em parte por se tratar de um mercado de negociação ininterrupta, com negociações em finais de semana e feriado além de boa liquidez. Dado que pretendíamos não apenas avaliar o desempenho dos modelos em cada série mas também compará-los, vimos como necessário garantir que tais dados referiam-se ao mesmo período de tempo, assim, os *datasets* foram nivelados de modo que descartamos dados associados a dias que não eram comuns a todos os ativos.

Seguimos a risca a premissa da estacionariedade da série e da necessidade de sua normalização para o bom funcionamento das ESNs, iniciamos através da tomada dos log-retornos dos dados, uma abordagem tradicional em trabalhos do setor financeiro. Constatamos via análise dos dados de estatística descritiva e observação das distribuições sua não normalidade evidenciada através da assimetria latente do conjunto de dados de retorno, aplicamos o algoritmo de Yeo-Johnson visando obter uma melhor normalização das séries temporais e obtivemos bons resultados.

Passamos à análise do conjunto de dados de treino via método de ACF e PACF, constatamos a existência de relações de memória entre os dados que deveriam constituir parte da explicação do modelo, bem como evidenciar sua capacidade de modelagem de

dados. Realizamos as diversas rodadas de combinações de hiperparâmetros e modelos distintos e levantamos os *Heat Maps* para os conjuntos de hiperparâmetros analisados. Foi observado um maior grau de ruído nos *Heat Maps* resultantes, de modo que não era possível visualizar gradientes bem definidos de minimização de erro, à exceção do Cardano, que mostrou-se mais dependente da conservação de estados anteriores, via baixa taxa de vazamento espectral, todos os ativos têm seu mínimo aproximado, em dados de treino, em baixos valores relativos de raio espectral e escala da entrada bem como valores elevados de esparsidade e taxa de vazamento espectral.

Os modelos foram gerados e os valores preditos foram descontados da base de dados de treino, de modo a gerar diversas séries de resíduos a serem analisadas. Para evidenciar a capacidade do modelo em modelar as relações de dependência e memória na série de dados passamos à análise do ACF e PACF de tais ruídos, tendo como base o melhor e pior modelo em termos de RMSE para $N = 20$. Esperamos que bons modelos sejam capazes de reduzir todos os gráficos de ACF e PACF para a zona hachurada, isto é, eliminar qualquer forma de dependência estatisticamente distinta de zero, caso o modelo não fosse capaz de modelar todas as relações presentes, o tamanho de janela (n) era incrementado de modo a obter um ajuste fino de tamanho de janelas. O tamanho ideal de janelas ultrapassou $n = 20$ dias para todos os ativos à exceção de Cardano, que foi devidamente modelo em $n = 4$ dias.

Verificamos o desempenho dos modelos gerados no conjunto de dados de validação em termos de RMSE e MAPE para diferentes tamanhos de camada oculta (N) e decidimos verificar sua usabilidade e aplicabilidade face ao, assim chamado, *preditor ingênuo* que consistia em simplesmente pressupor o resultado esperado no próximo passo como idêntico ao anteriormente realizado. Apesar de simples, o preditor ingênuo tem forte arcabouço teórico na teoria dos mercado eficientes e demonstra bons resultados em ativos sobre efeitos de crescimento/decrescimento constantes, o assim chamado *fator momento*. Os resultados apresentados são considerados úteis e aceitáveis, demonstrando boa capacidade de superar, em muito, o preditor ingênuo em ambas as medidas de erro, especialmente em modelos de tamanho de camada oculta reduzida.

Os gráficos verificados são promissores para a aplicação do modelo e tem forte intuição lógica e teórica, dado que os modelos de menor tamanho de camada oculta, estes acabam limitando-se a modelar relações mais genéricas, do ponto de vista estatístico, que evidenciam tendências reais de comportamento de mercado. Modelos de maior envergadura acabam por modelar relações entre as variáveis não tão essenciais ou significativas e correm grandes riscos de *overfitting* além de interpretar sinais espúrios como ruídos ou eventos isolados como padrões constantes de negociação, por isso desempenham muito pior, com aumento de erro praticamente exponencial com relação ao aumento de camada, conforme constatação empírica via visualização dos gráficos. A grande exceção foi Cardano, que

não apenas possuía o menor modelo em termos de tamanhos de janela, como também apresentou constantemente um superioridade marcante em ambas as medidas de erro, o ativo apresentou igualmente um padrão peculiar, distinto de seus semelhantes, já na análise do conjunto de dados de treino, evidenciado pelos *Heat Maps* e pelo ACF e PACF, todavia, mais testes seriam necessários para associar tais características à eficiência de predição do modelo.

Nossos resultados são promissores, entendemos as limitações do método e de nossos processos de otimização, todavia, ressaltamos que bons resultados foram obtidos em ambos os ativos e que as redes neurais implementados se mostraram de rápida execução, apresentam pouca exigência em termos de memória e podem ser facilmente escalonadas para modelos mais complexos e otimizados. Ressaltamos que os ativos utilizados apresentam alta liquidez, valor de captação e volume de negociação esperado, o que impacta em uma alta eficiência de mercado em torno da precificação o que dificulta ganhos relevantes via análise de padrões de negociação, dado que pressupõem-se que tais padrões não deveriam existir, ou seriam devidamente arbitrados. Os resultados do modelo contestam a impossibilidade da modelagem de tais tendências e são devidamente factíveis de implementação.

Salientamos também que mais testes hão de ser realizados, visando compreender o desempenho deste sistema em outros mercados, com especial ênfase no mercado de capitais brasileiro, o qual muito nos interessa. Também não é possível relacionar os comportamentos observados com relação a liquidez dos ativos, dado que os dados utilizados já pressupõem a existência de alto volume de negociação, de modo que permanece em dúvida a extensão da relação entre os resultados de predição e as características intrínsecas associadas as negociações dos ativos. Compreendermos que a classe de criptoativos apresenta elevada volatilidade, que se manifesta em variações abruptas de preço, este fenômeno está amplamente associado às características do mercado, aos diferentes agentes econômicos envolvidos, às expectativas em torno dos ativos e o relativo consenso acerca das características arriscadas de tais ativos, sendo este um dos grandes motivos que levaram-nos à escolha de tais ativos.

Por fim, é importante compreender que a precificação, como processo econômico, é tão mais eficiente conforme a capacidade dos agentes de receber, transmitir e interpretar as informações econômicas bem como sua capacidade de atuação via colocação de ordens, além de fenômenos comportamentais e características próprias do sistema financeiro de negociação e regulamentação. Deste modo, espera-se maior desempenho em torno da precificação do ativo, via processo de arbitragem, quanto menor a frequência dos dados do sistema. Partindo-se da constatação de que nosso modelo apresenta capacidade de arbitrar anomalias estatísticas consistes no processo de negociação esperamos um desempenho superior face a dados de frequência mais elevada, onde menos agentes são capazes de exercer poder de arbitragem e correção de preço. Sistemas de operação em mais alta frequência

exigirão maior grau de desenvolvimento em termos de otimização e custo de infraestrutura, além de expertise em técnicas de paralelização, sendo demasiadamente complexas para o escopo deste trabalho mas não para implementação em empresa especializada.

Este produto foi apresentado foi apresentado para gestores de fundos de investimento em renda variável e multimercados da cidade de Porto Alegre, tendo sido considerado por estes como promissor. A metodologia deste pesquisa tornou-se referência para outros projetos internos de capital privado e o produto encontra-se em fases de testes interna já bastante avançada para implementação em estratégias de negociação, visualização de risco de mercado e alocação de carteira em mercado nacional.

4.2 Próximos Passos

No decorrer deste trabalho, foi possível apropriar-se da técnica e adquirir experiência prática suficiente a ponto de elencar-se tópicos relevantes em termos de otimização, melhorias diversas e implementação, dentre as quais sugerimos.

Aprimoramento de Design & Projetos das ESNs: Diversas arquiteturas vinculadas ao paradigma de *Reservoir Computing* foram posteriormente sugeridas e discutidas na literatura. Dentre as principais elencamos simplificações de arquitetura de reservatório em termos de conexões de neurônios, em detrimento das conexões pseudo-aleatórias utilizadas, utilização de vetores de pesos de *feedback*, modelos híbridos com utilização de camada oculta de redes neurais do tipo recursiva e paradigmas como o *DeepESN* que visa a conexão de diferentes reservatórios [5].

Otimização de hiperparâmetros: Implementações de meta-heurísticas via máquinas de Monte Carlo e/ou algoritmos genéticos são uma provável solução para o problema de otimização automatizada dos hiperparâmetros. São possíveis também soluções via gradiente e otimizadores bastante utilizados em *frameworks* abertos de *machine learning*.

Paralelização: Em voga no mercado de tecnologia, técnicas de paralelização via GPU são um possível caminho visando ganho vital em termos de tempo de processamento.

Outros ativos: Acreditamos que muito pode ser encontrado em termos de séries temporais de ativos financeiros. Diferentes mercados são mais ou menos arbitrados do ponto de vista quantitativo. O tempo de operação também é uma informação importantíssima, dado que menor tempo de operação reduz o número de agentes capazes de compreender as relações dos padrões estatísticos presentes nas séries temporais, por conta do custo dos dados, infraestrutura e expertise necessárias para tais operações, desenvolvimento e implementação de algorítimos.

Expoente de Lyapunov e Expoente de Hurst: As características caóticas e fractais do mercado financeiro, bem como a capacidade de generalização de estados de

sistemas dinâmicos das ESNs, levantam a questão acerca da existência de relações de desempenho das redes face determinado grau de caoticidade e/ou fractalidade que podem ser estimados via expoente de Lyapunov e expoente de Hurst, respectivamente.

4.3 Agradecimentos & Considerações Finais

Uma versão inicial deste trabalho foi uma das apresentações centrais do **Scitek**, evento presencial de *data science* focado em interação entre empresas e meio acadêmico, o evento foi organizado pela Poatek, empresa de tecnologia de Porto Alegre, e contou com grande participação em termos de público.

Este trabalho foi premiado com o segundo lugar na *Datathon* nacional da **Escola de Economia de São Paulo da FGV (FGV-EESP)** focado em soluções de *data science* para criptoativos e *blockchain*. Nosso trabalho foi apresentado em *webinar* via canal da FGV-EESP no YouTube.

Agradecemos ao corpo técnico da **Fundamenta Investimentos** pelo conhecimento técnico provido em mercados financeiros, bem como no setor de negociação, fundos, análise e operações automatizadas e quantitativas. Agradecemos igualmente à **Quantitas Asset Management**, por propiciar expertise, infraestrutura de dados e suporte para o desenvolvimento de algoritmos de *machine learning* de alto impacto para fundos de investimentos multimercados no mercado nacional.

5 APÊNDICE A - HEAT MAPS

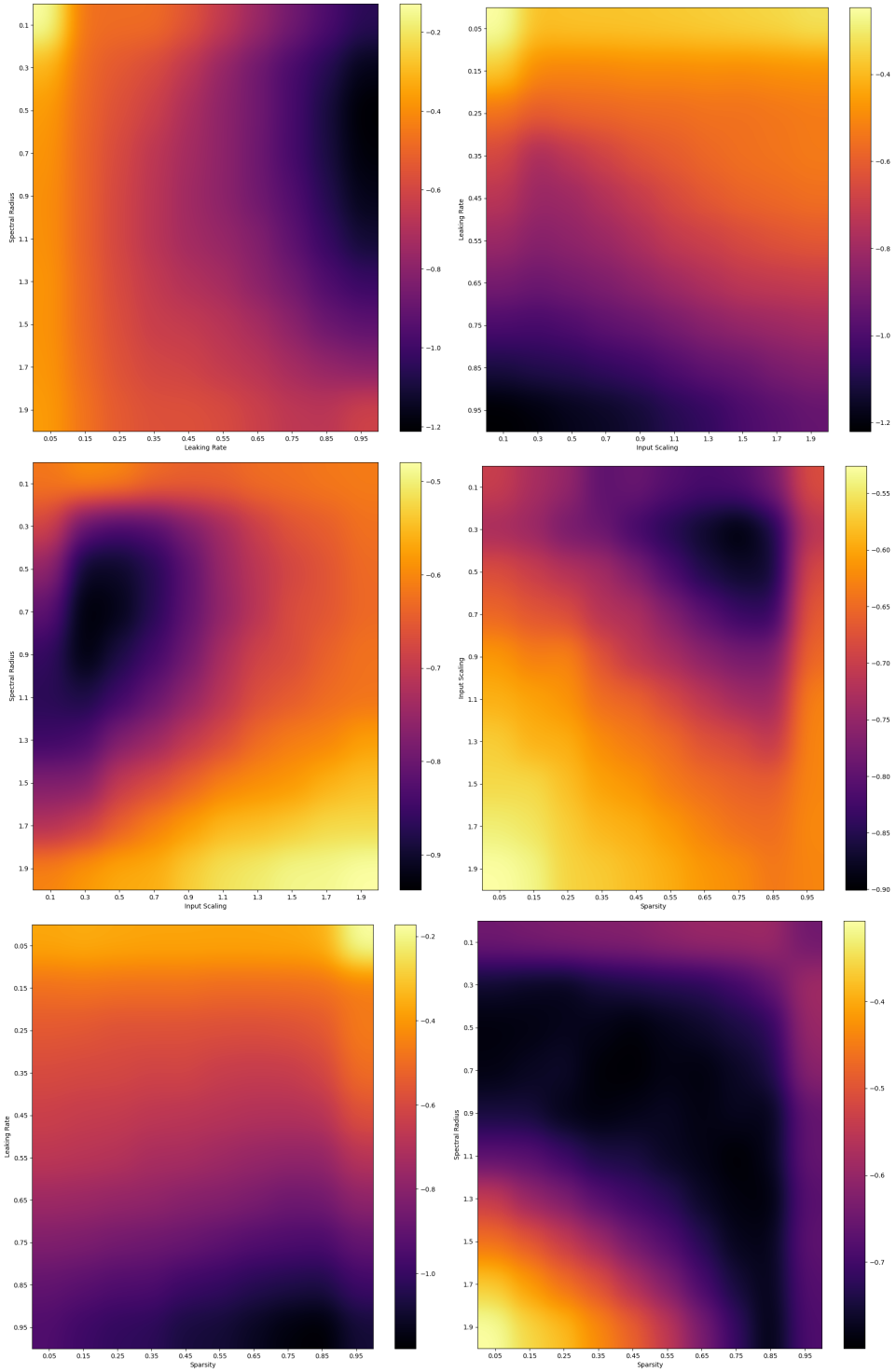

Figura 30 – Configurações de *Heat Maps* com medianas do MAPE em escala log10, para o Mackey Glass ($\tau = 17$) evidenciando comportamento das ESNs em conjunto de teste face alterações nos hiperparâmetros.

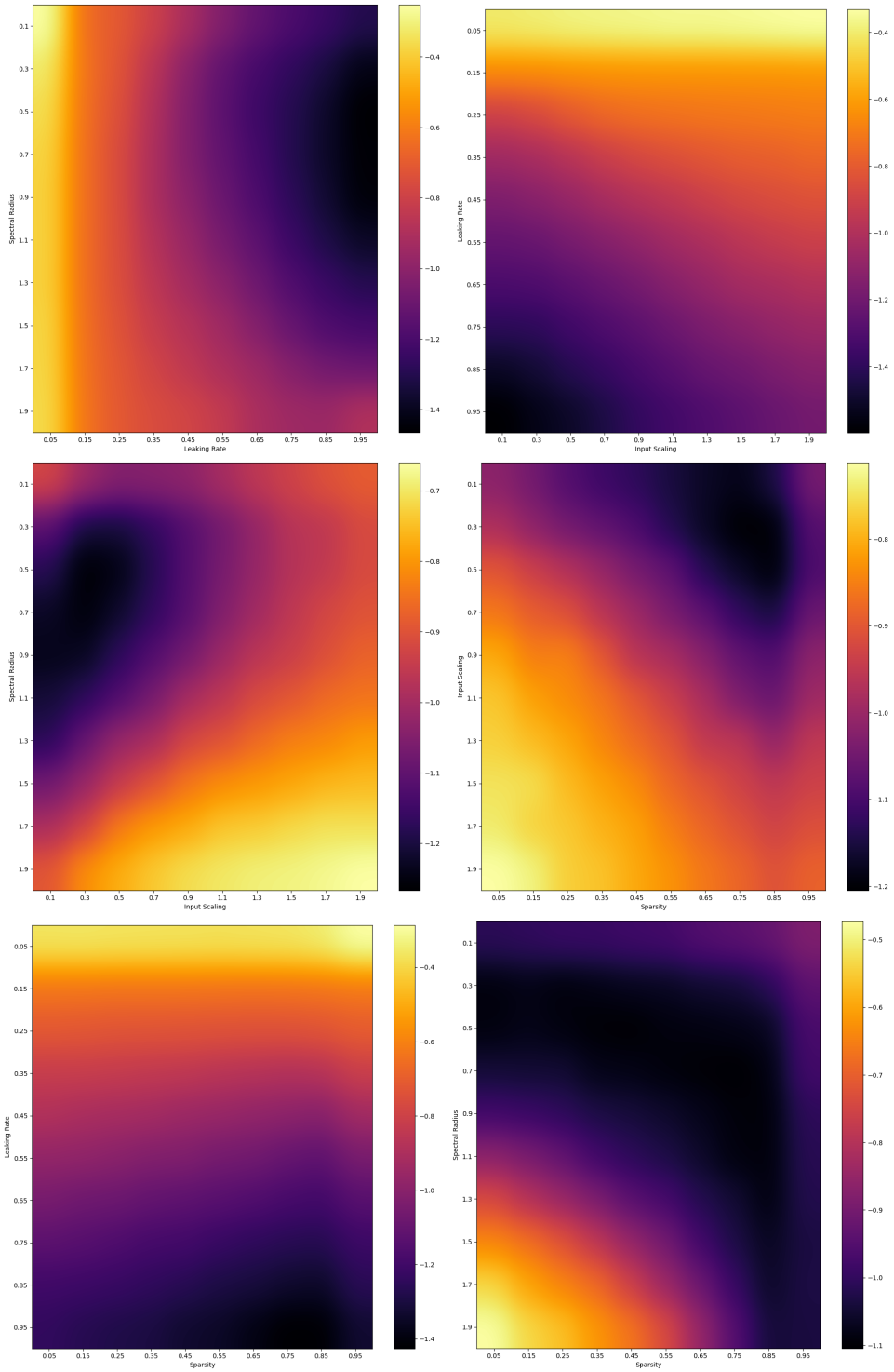

Figura 31 – Configurações de *Heat Maps* com medianas do MAPE em escala log10, para o Mackey Glass ($\tau = 22$) evidenciando comportamento das ESNs em conjunto de teste face alterações nos hiperparâmetros.

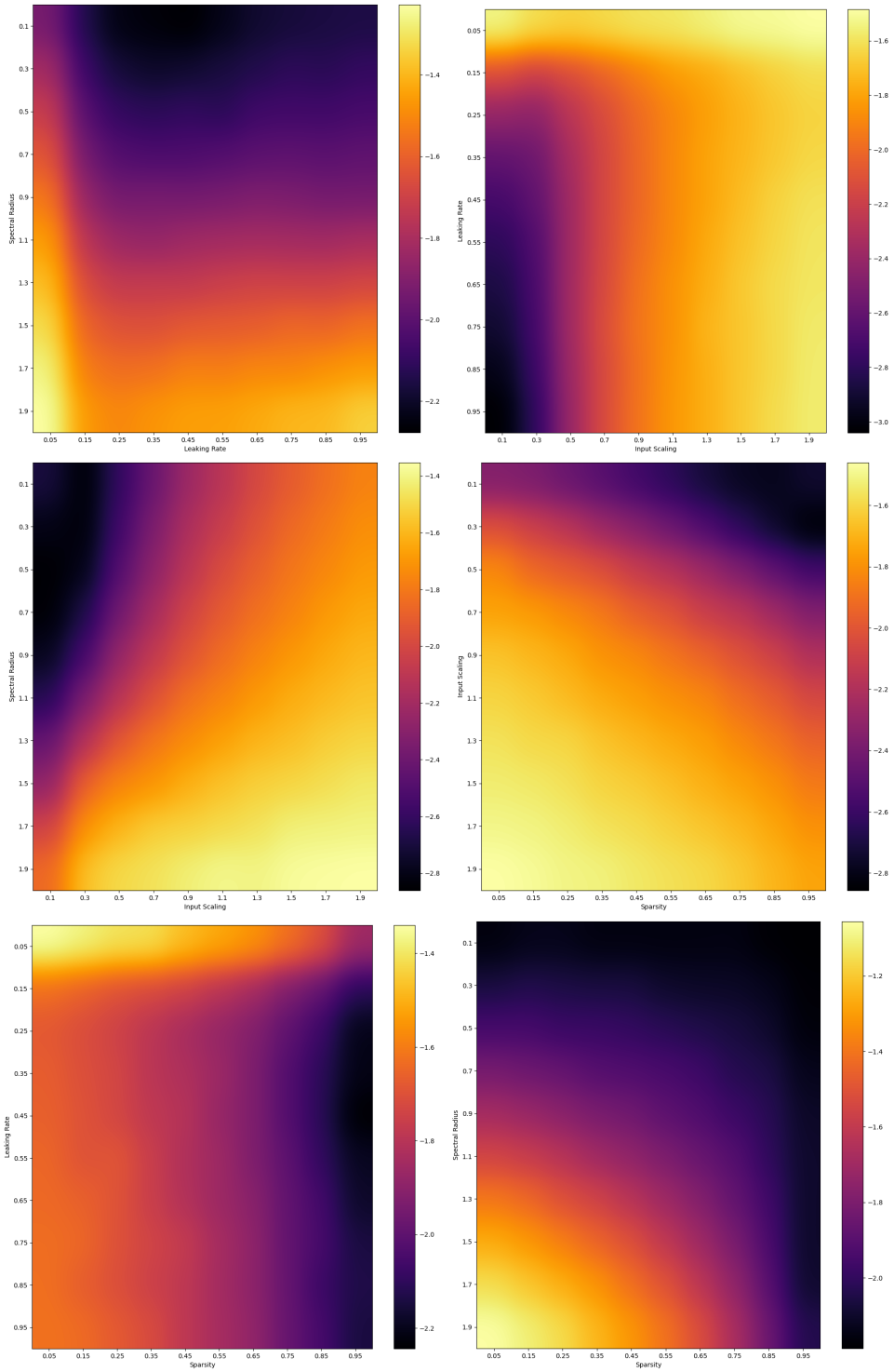

Figura 32 – Configurações de *Heat Maps* com medianas do MAPE em escala log10, para o atrator de Lorenz evidenciando comportamento das ESNs em conjunto de teste face alterações nos hiperparâmetros.

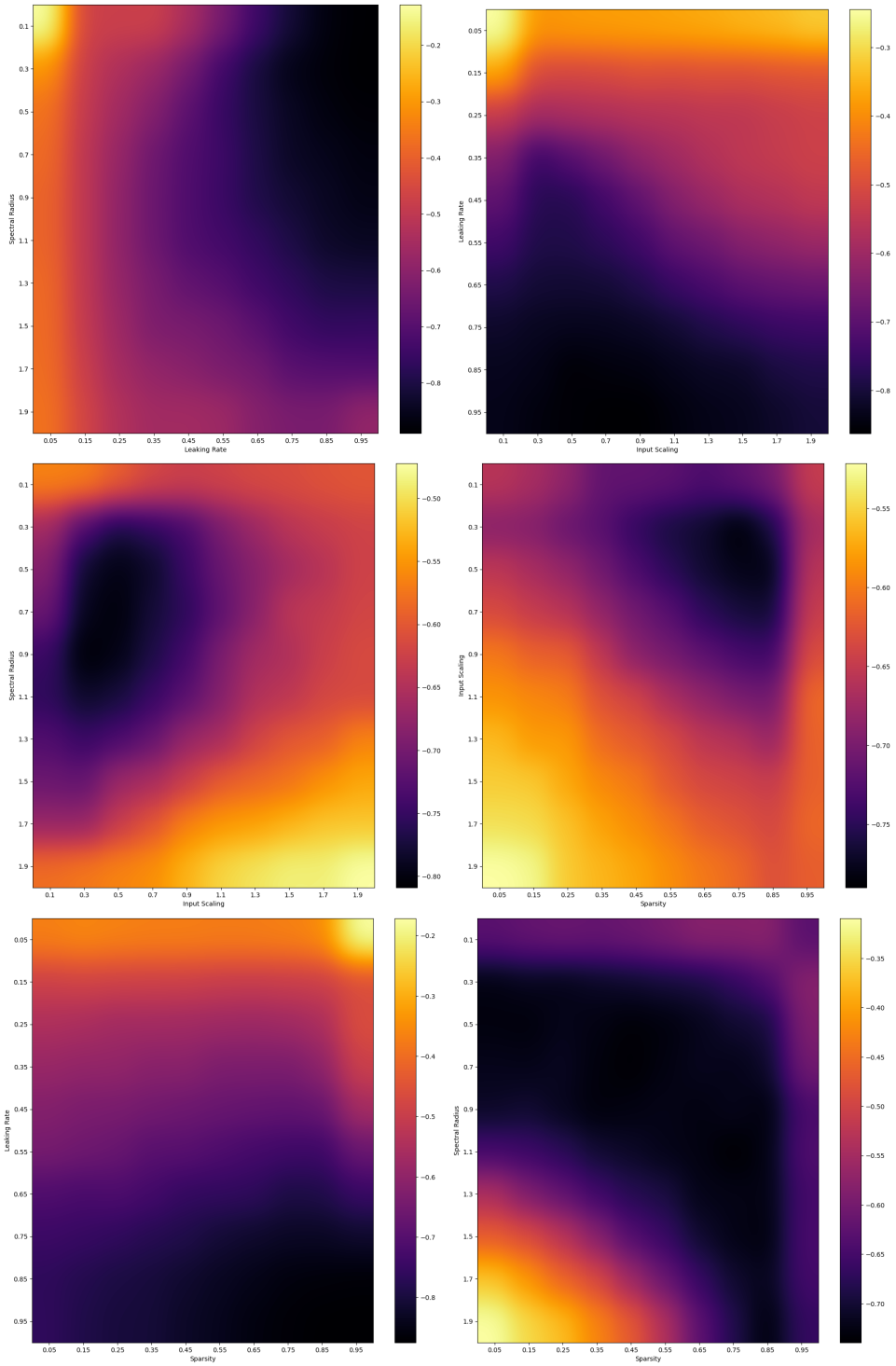

Figura 33 – Configurações de *Heat Maps* com medianas do MAPE em escala log10, para o Mackey Glass ($\tau = 17$) acrescido de ruído evidenciando comportamento das ESNs em conjunto de teste face alterações nos hiperparâmetros.

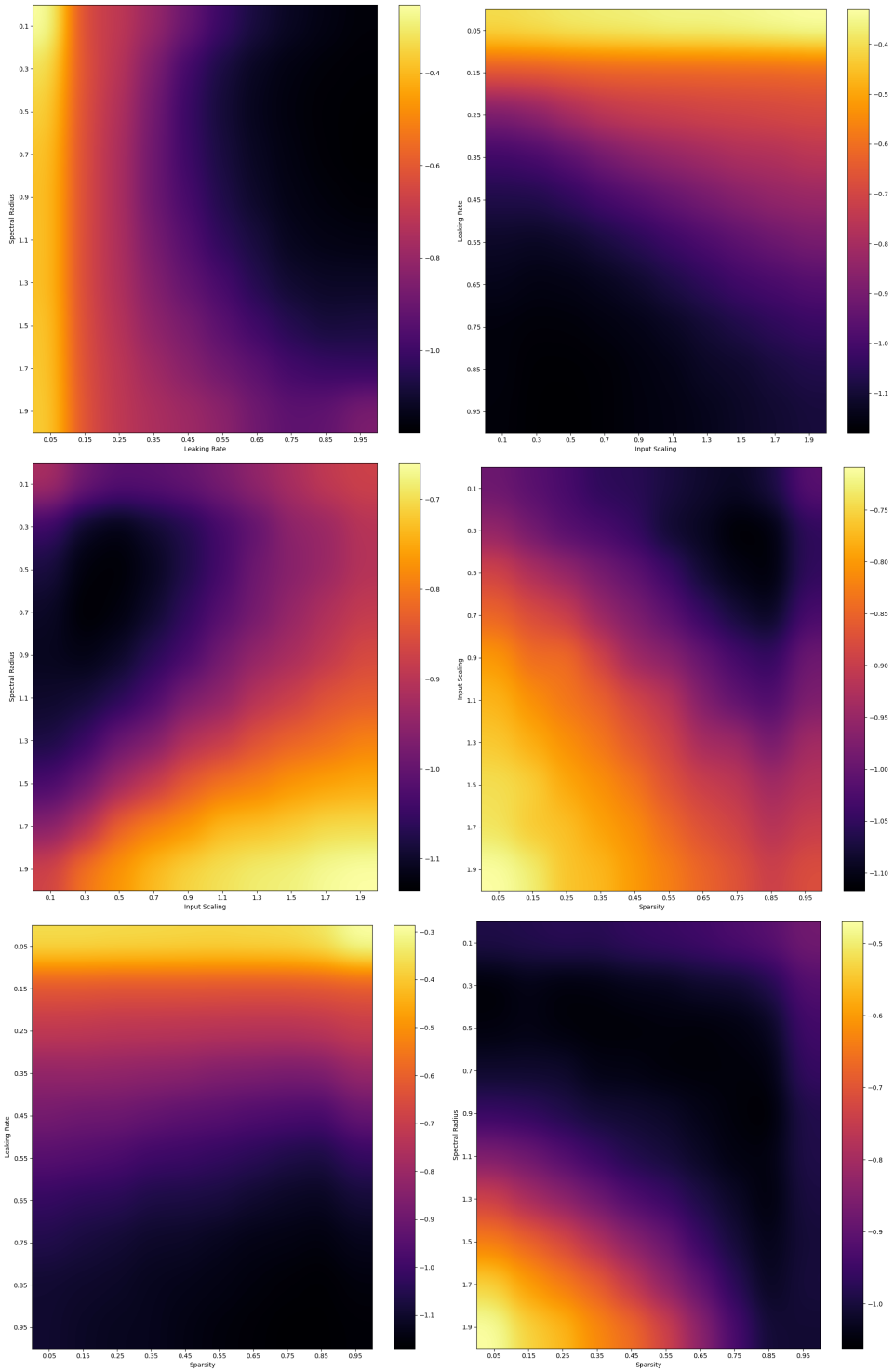

Figura 34 – Configurações de *Heat Maps* com medianas do MAPE em escala log10, para o Mackey Glass ($\tau = 22$) acrescido de ruído evidenciando comportamento das ESNs em conjunto de teste face alterações nos hiperparâmetros.

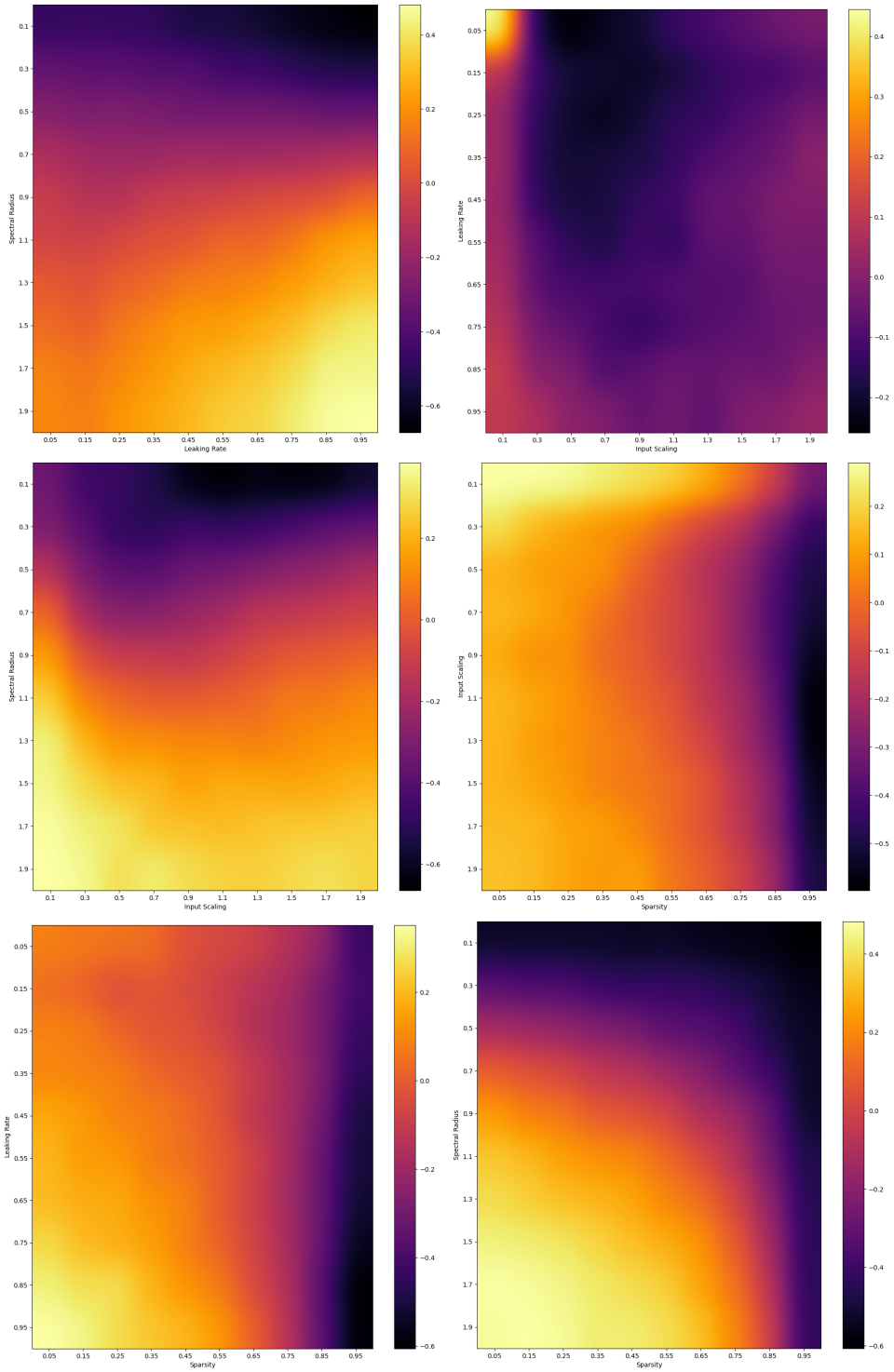

Figura 35 – Configurações de *Heat Maps* com medianas do MAPE em escala log10, para o Mapa de Hénon acrescido de ruído evidenciando comportamento das ESNs em conjunto de teste face alterações nos hiperparâmetros.

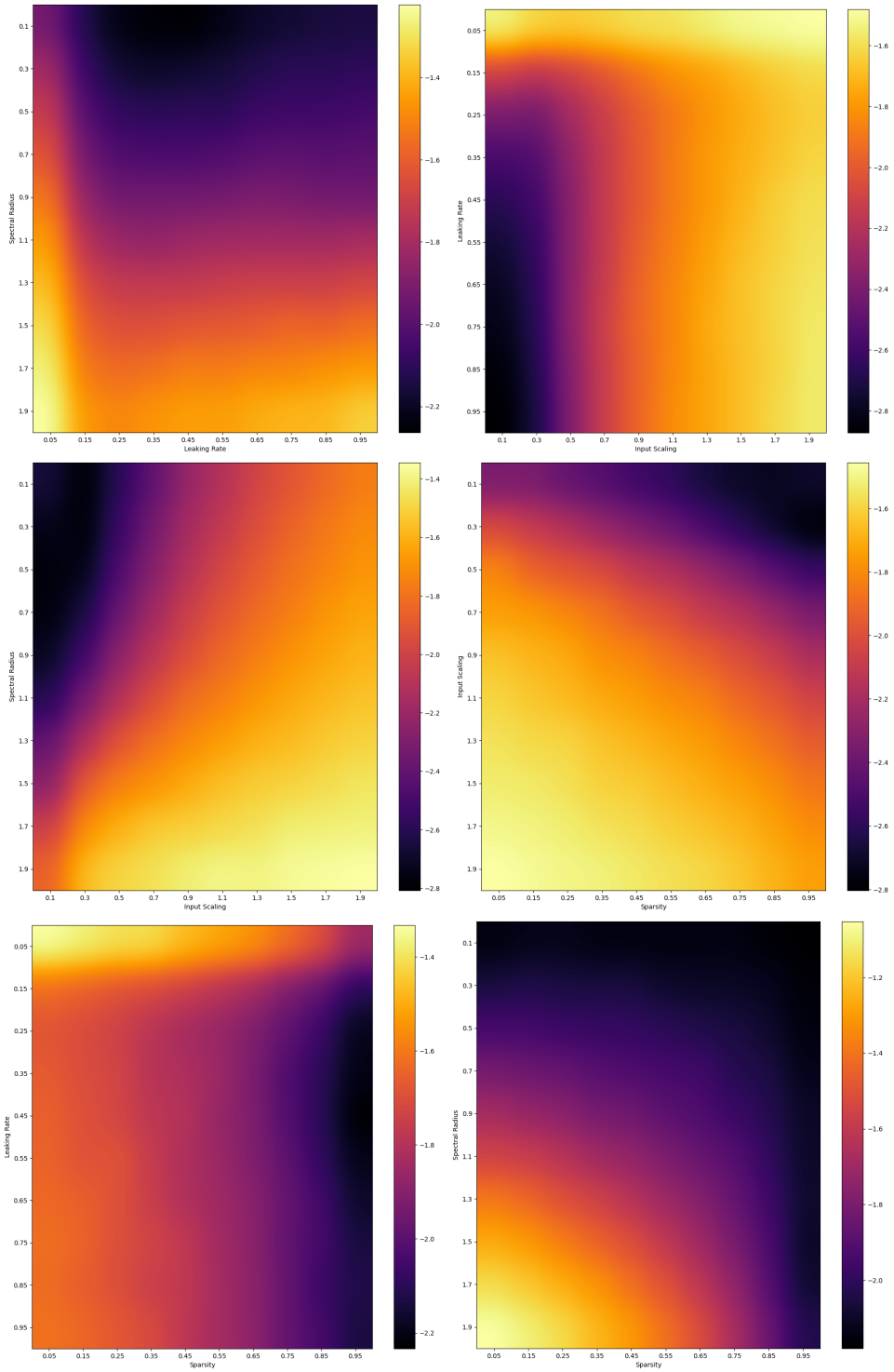

Figura 36 – Configurações de *Heat Maps* com medianas do MAPE em escala log10, para o Atrator de Lorenz acrescido de ruído evidenciando comportamento das ESNs em conjunto de teste face alterações nos hiperparâmetros.

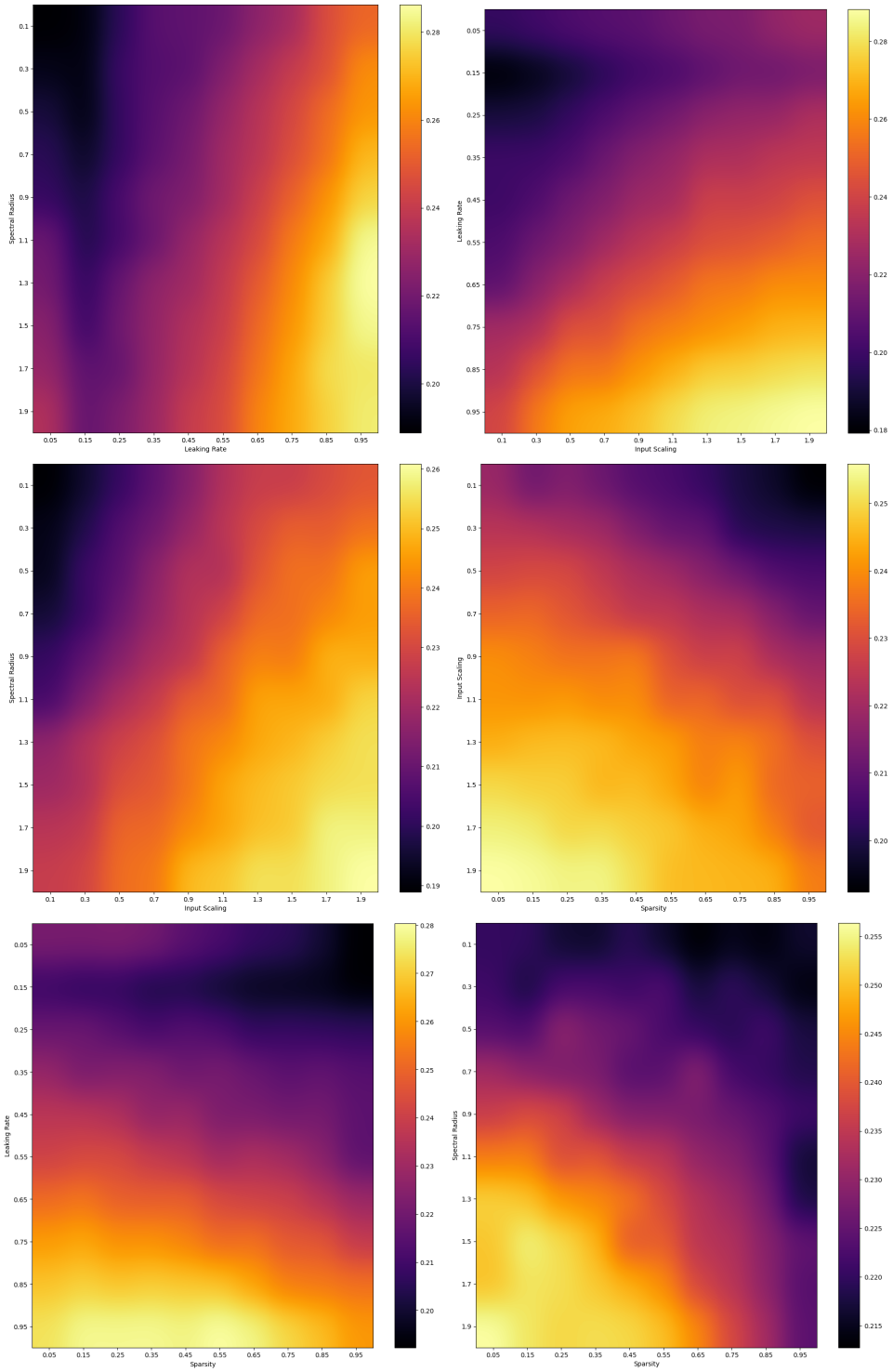

Figura 37 – Configurações de *Heat Maps* com medianas do MAPE em escala log10, para o Cardano evidenciando comportamento das ESNs em conjunto de teste face alterações nos hiperparâmetros.

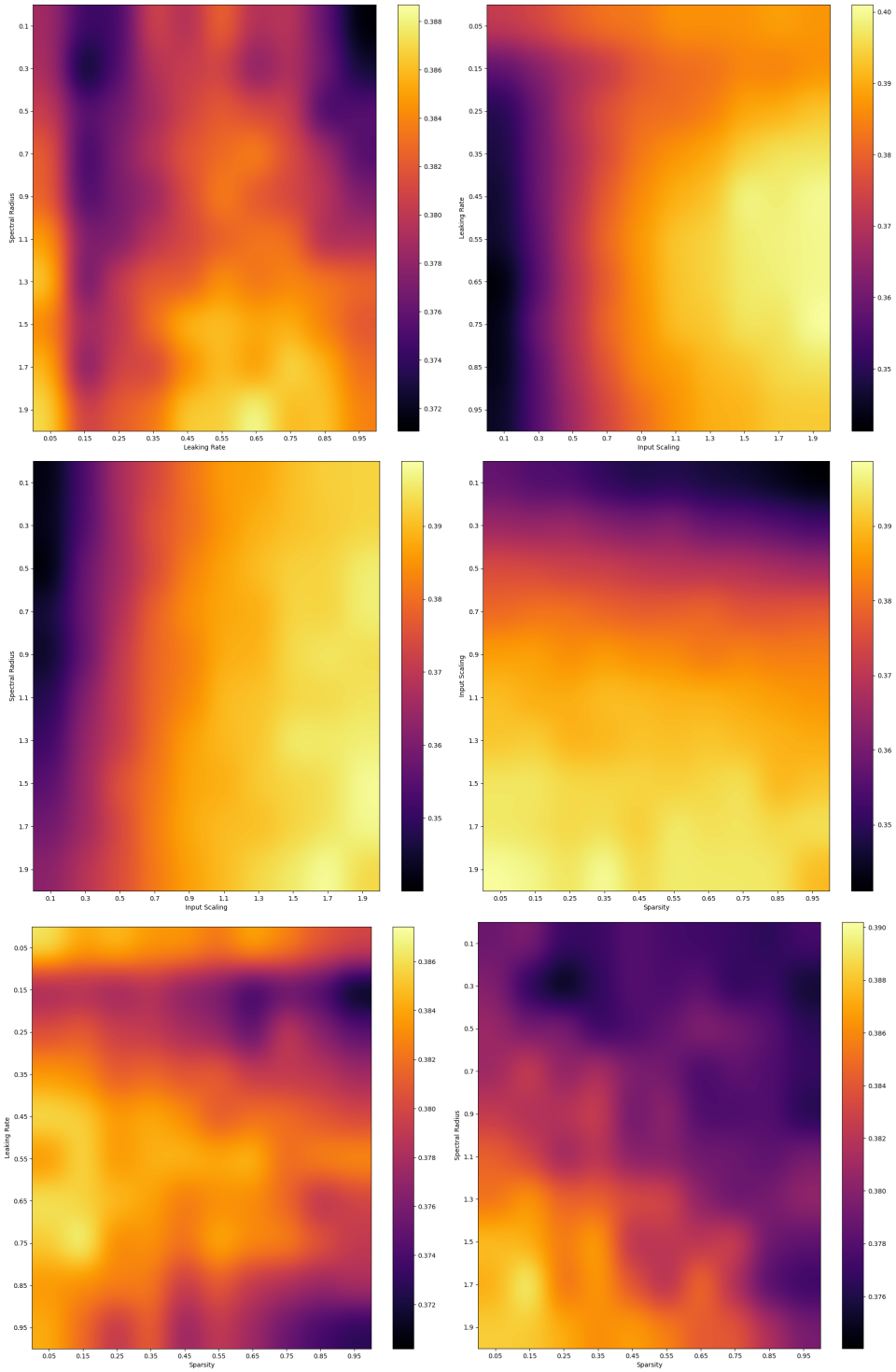

Figura 38 – Configurações de *Heat Maps* com medianas do MAPE em escala log10, para o Dogecoin evidenciando comportamento das ESNs em conjunto de teste face alterações nos hiperparâmetros

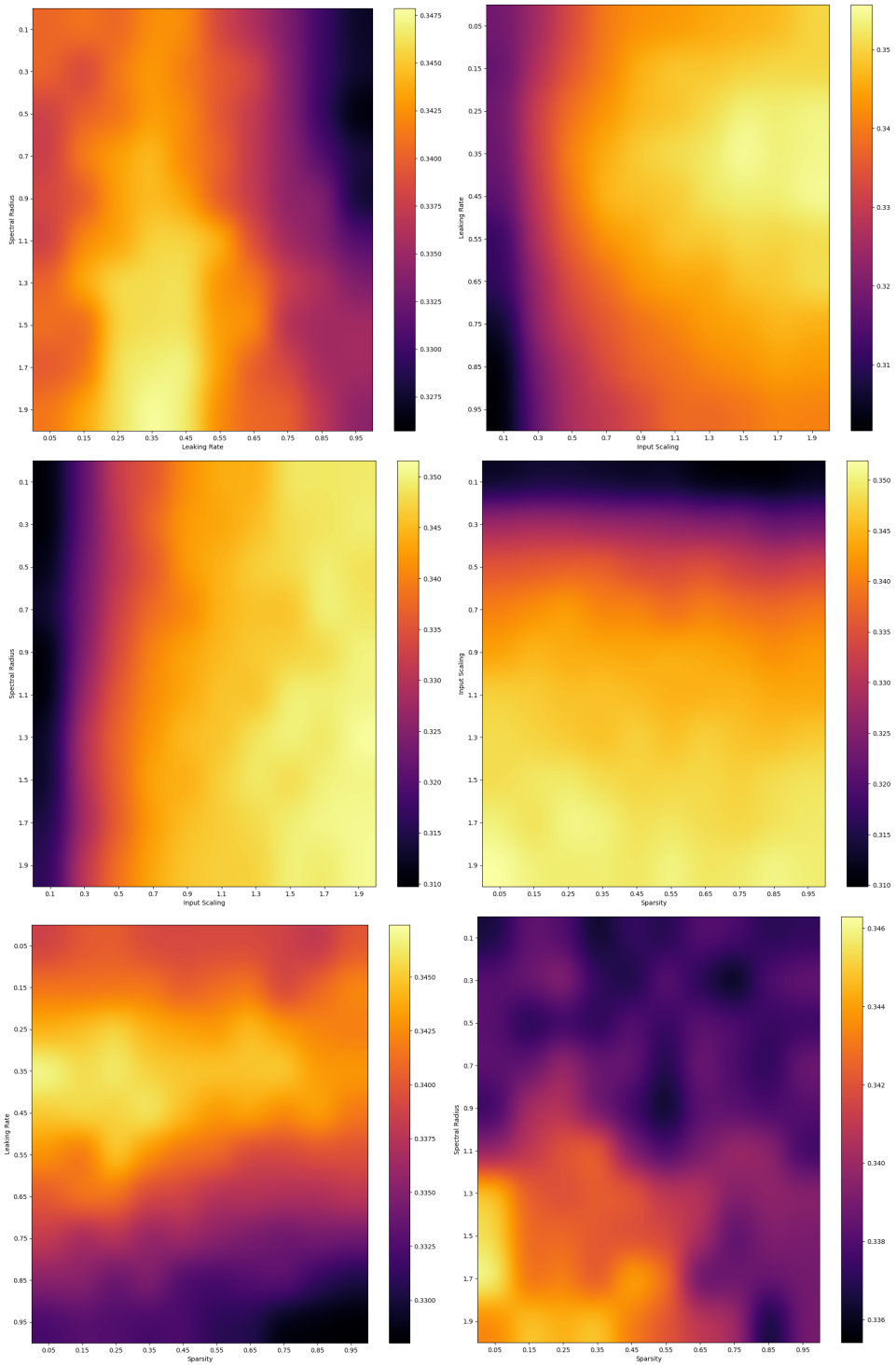

Figura 39 – Configurações de *Heat Maps* com medianas do MAPE em escala log10, para o Ethereum evidenciando comportamento das ESNs em conjunto de teste face alterações nos hiperparâmetros

6 APÊNDICE B - CÓDIGOS

Códigos completos e mais informações úteis acerca da implementação podem ser encontradas no repositório **Reservoir Computing** presente no GitHub do desenvolvedor. O algoritmo abaixo demonstra uma implementação completa de uma ESN para processamento de um arquivo de duas colunas (X e Y) contendo os dados relativos ao Mapa de Hénon.

```
// EchoState.py
import numpy
from scipy import linalg
import matplotlib.pyplot as plt

# Garante o formato das colunas para arrays unidimensionais
def reshape(df):
    if len(df.shape) < 2:
        return numpy.reshape(df, (df.shape[0], 1))
    else:
        return df

dataset = reshape(numpy.loadtxt('NormHenon.txt', delimiter = ","))

K = dataset.shape[1]
L = K

steps = 1

train_length = 3000
test_length = 1000

transiente = 500

train_dataset = dataset[0 : train_length, :]
train_target = dataset[None, steps + transiente : train_length + steps, :]

test_dataset = dataset[train_length : train_length + test_length, :]
test_target = dataset[None, train_length + steps : train_length + test_length
    + steps, :]

N = 10
```

```
leaking_rate = 0.3
input_scaling = 1
spectral_radius = 1.25

numpy.random.seed(42)

W_input = (numpy.random.rand(N, 1 + K) - 0.5) * input_scaling
W_reservoir = numpy.random.rand(N, N) - 0.5

rhoReservoir = numpy.max(numpy.abs(linalg.eig(W_reservoir)[0]))
W_reservoir *= spectral_radius / rhoReservoir

state_matrix = numpy.zeros((1 + K + N, train_length - transiente))

state = numpy.zeros((N, 1))

n = 0
u = train_dataset[n,:]
state = (1 - leaking_rate)*state + leaking_rate*numpy.tanh(numpy.dot(W_input,
    reshape(numpy.hstack((1, u)))) + numpy.dot(W_reservoir, state))

state_matrix[:, n] = numpy.hstack((1, u, state[:,0]))

reg = 1e-8
W_output = linalg.solve(numpy.dot(state_matrix, state_matrix.T) + reg *
    numpy.eye(1 + K + N),
    numpy.dot(state_matrix, train_target.T) ).T[0]

state = reshape(state_matrix[K + 1::,-1])

state_matrix = numpy.zeros((1 + K + N, test_length))

for n in range(test_dataset.shape[0]):
    u = test_dataset[n,:]
    state = (1 - leaking_rate)*state +
        leaking_rate*numpy.tanh(numpy.dot(W_input, reshape(numpy.hstack((1,
        u)))) + numpy.dot(W_reservoir, state))
    state_matrix[:, n] = numpy.hstack((1, u, state[:,0]))

predict = numpy.dot(W_output, state_matrix)

plt.figure(figsize=(12,12))
```

```
plt.scatter(test_target[0, :, 0], test_target[0, :, 1], color = "blue",
    linewidth = 2, label = "Expected")
plt.scatter(predict[0,:], predict[1,:], color = "red", linewidth = 1, label =
    "Predict")
plt.legend()
plt.show()
```

REFERÊNCIAS

[1] Benjamin Graham, David Le Fevre Dodd, Sidney Cottle, et al. *Security analysis*, volume 452. McGraw-Hill New York, 1934.

[2] Mantas Lukoševičius and Herbert Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, 2009.

[3] Zhiwei Shi and Min Han. Support vector echo-state machine for chaotic time-series prediction. *IEEE transactions on neural networks*, 18(2):359–372, 2007.

[4] Ken-ichi Funahashi and Yuichi Nakamura. Approximation of dynamical systems by continuous time recurrent neural networks. *Neural networks*, 6(6):801–806, 1993.

[5] Chenxi Sun, Moxian Song, Shenda Hong, and Hongyan Li. A review of designs and applications of echo state networks. *arXiv preprint arXiv:2012.02974*, 2020.

[6] Wolfgang Maass, Thomas Natschläger, and Henry Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation*, 14(11):2531–2560, 2002.

[7] Herbert Jaeger. The "echo state" approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, 148(34):13, 2001.

[8] Herbert Jaeger. Adaptive nonlinear system identification with echo state networks. *Advances in neural information processing systems*, 15, 2002.

[9] Mantas Lukoševičius. A practical guide to applying echo state networks. In *Neural networks: Tricks of the trade*, pages 659–686. Springer, 2012.

[10] Edward N Lorenz. Deterministic nonperiodic flow. *Journal of atmospheric sciences*, 20(2):130–141, 1963.

[11] Guido Van Rossum and Fred L Drake Jr. *Python tutorial*. Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands, 1995.

[12] Michael C Mackey and Leon Glass. Oscillation and chaos in physiological control systems. *Science*, 197(4300):287–289, 1977.

[13] Michel Hénon. A two-dimensional mapping with a strange attractor. In *The theory of chaotic attractors*, pages 94–102. Springer, 1976.

[14] In-Kwon Yeo and Richard A Johnson. A new family of power transformations to improve normality or symmetry. *Biometrika*, 87(4):954–959, 2000.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[16] Joao Henrique F Flores, Paulo Martins Engel, and Rafael C Pinto. Autocorrelation and partial autocorrelation functions to improve neural networks models on univariate time series forecasting. In *The 2012 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2012.

[17] Benoit B Mandelbrot. The variation of the prices of cotton, wheat, and railroad stocks, and of some financial rates. *Fractals and Scaling in Finance: Discontinuity, Concentration, Risk. Selecta Volume E*, pages 419–443, 1997.

[18] Benoit Mandelbrot. The variation of some other speculative prices. *The Journal of Business*, 40(4):393–413, 1967.