Análise de Aplicação de Modelos ARIMA e de Redes Neurais em Dados de COVID-19

Uriel Cairê Balan Calvi, Dérick Wellman Brock Rangel

[1]Aeronautics Institute of Technology (ITA) – Computer Science Division
São José dos Campos – SP – Brazil

uriel.caire@gmail.com, derickwelman@gmail.com

Abstract. *With the first cases reported in December 2019 in Wuhanna China, the SARS-CoV-2 (COVID-19) virus pandemic was officially recognized by the World Health Organization (WHO) on March 11, 2020. At the last week of April of the same year, Brazil, with a total of 3670 deaths, already occupied the 11th place in the ranking of countries with more deaths. Despite the scenario, in Brazil the restrictive measures were and still are a matter of debate and the federal government's decision on the subject varies until today. Considering the contribution of forecasting tools in decision-making situations, the objective of this work is to evaluate the performance of the ARIMA and Neural Networks models in predicting the number of deaths through a COVID-19 time series. In conclusion, we observed that the model that best adapted to the data was ARIMA Seasonal, but the data can still be improved to improve the training of the models.*

Resumo. *Com os primeiros casos reportados em Dezembro de 2019 em Wuhan na China, a pandemia do vírus SARS-CoV-2 (COVID-19) foi oficialmente reconhecida pela Organização Mundial da Saúde (OMS) em 11 de março de 2020. Na última semana de Abril do mesmo ano, o Brasil, com um total de 3670 mortes, já ocupava o 11º lugar no ranking dos países com mais mortes registradas. Apesar do cenário, no Brasil as medidas restritivas eram e ainda são motivos de debate e a decisão do governo federal em relação ao tema oscila até hoje. Tendo em vista a contribuição das ferramentas de previsão em situações de tomada de decisão, o objetivo deste trabalho é avaliar o desempenho dos modelos ARIMA e de Redes Neurais na previsão do número de óbitos através de uma série temporal de COVID-19. Como conclusão, observamos que o modelo que melhor se adaptou ao dado foi o de ARIMA Sazonal, mas o dado ainda pode ser melhorado para aperfeiçoar o treinamento dos modelos.*

1. Introdução

Em estatística, séries temporais podem ser descritas como uma sequência de observações de uma variável ao longo de intervalos regulares de tempo [Ehlers 2007]. Uma característica forte de uma série temporal é a dependência de ordem dos dados, portanto a alteração dessa ordem muda totalmente os resultados das análises ou o comportamento dos modelos de previsão.

A análise de uma série temporal pode revelar algumas características comportamentais do dado. Entre estas características, podemos destacar três:

- Tendência: em geral, uma série temporal pode apresentar uma tendência de crescimento ou queda com alguns possíveis padrões. Entre os padrões mais conhecidos estão: o crescimento linear, onde o valor observado tem um aumento estável acrescido por uma constante fixa; e o crescimento exponencial, onde ao invés de uma adição simples de uma constante há uma interação de multiplicação entre uma constante de crescimento e uma variável.
- Sazonalidade: um padrão sazonal ocorre quando uma série temporal é afetada por fatores externos que resultam em movimentos temporais regulares. Uma época do ano (e.g., natal) ou dia específico da semana são exemplos destes fatores externos.
- Ciclo: um ciclo ocorre quando os dados da série temporal exibem subidas e descidas que, diferentemente da sazonalidade, não possuem frequência fixa. Estas flutuações podem estar relacionadas a, por exemplo, fatores econômicos.

Além da análise descritiva dos dados, as séries temporais também podem ser utilizadas para realizar previsões. Utilizam-se os valores passados para predizer os valores futuros. Assumindo que o futuro envolve incerteza, essas previsões não são perfeitas, porém estes erros podem ser minimizados levando a resultados próximos aos valores reais.

Séries temporais estão presentes em diferentes áreas de conhecimento, como na Economia (e.g., variação dos preços de ações durante o dia), Medicina (e.g., eletrocardiograma), Epidemiologia (e.g., número diário de novas contaminações), etc. Neste artigo, foram utilizados dados da pandemia de SARS-CoV-2 (COVID-19) obtidos através do portal Our World in Data, um projeto mantido por um time da Universidade de Oxford.

Com os primeiros casos reportados em Dezembro de 2019 em Wuhanna China, a pandemia do vírus COVID-19 foi oficialmente reconhecida pela Organização Mundial da Saúde (OMS) em 11 de março de 2020. Na última semana de Abril do mesmo ano, o Brasil, com um total de 3670 mortes, já ocupava o 11º lugar no ranking global dos países com mais mortes registradas [Souza et al. 2020].

Em contraste com países que conseguiram eliminar os casos de COVID-19 através do distanciamento social ou até mesmo *lockdown*, no Brasil ainda se discute a estratégia mais adequada para o combate a pandemia. O debate fica em torno da adoção do chamado ”isolamento vertical”(apenas indivíduos dos grupos de risco se isolam) ou ”isolamento horizontal”(válido para todos os indivíduos) [Werneck and Carvalho 2020]. Além disso, o governo tende a uma postura reativa, fortificando ou afrouxando as medidas restritivas de acordo com as altas e baixas do número de infectados.

2. Justificativa

O processo de tomada de decisão para fortificar ou não as medidas restritivas no Brasil é um exemplo de processo que pode ser auxiliado por modelos de previsão de dados. Portanto, este artigo busca analisar os dados de séries temporais do COVID-19 no país, com foco no número de óbitos; utilizar estes dados para treinar dois modelos de previsão distintos; e, por fim, avaliar o desempenho destes modelos.

A comparação será realizada entre um modelo ARIMA (*”Autoregressive integrated moving average”*), que é um dos mais populares modelos lineares para previsão de séries temporais, e um modelo de Redes Neurais Artificiais, que pode ser uma alternativa promissora aos modelos lineares tradicionais [Zhang 2003].

3. Pergunta(s) de Pesquisa

Os estudos desenvolvidos neste artigo são guiados pelas seguintes questões:

- Quais as características da série temporal de óbitos por COVID-19 no Brasil? Isto é, os dados possuem efeitos sazonais, tendências ou ciclos?
- Entre o modelo ARIMA e o modelo de Redes Neurais, qual deles obteve melhor acurácia nas previsões para esta série temporal?

4. Objetivos Gerais

O objetivo deste trabalho é realizar uma análise da série temporal de óbitos por COVID-19 no Brasil e realizar uma avaliação comparativa dos resultados da utilização destes dados em dois modelos de previsão, sendo eles ARIMA e Redes Neurais.

5. Objetivo Específicos

Os objetivos específicos que conduziram o desenvolvimento deste artigo estão listados a seguir:

- Analisar e pré-processar os dados brutos;
- Realizar uma análise exploratória da série temporal;
- Selecionar métodos, treinar, e avaliar os modelos treinados;
- Executar previsões, avaliar e comparar os resultados obtidos.

6. Metodologia

Nesta seção serão expostas todas as tecnologias, técnicas e dados utilizados no desenvolvimento deste trabalho e que levaram aos experimentos e resultados obtidos. Esta explanação estará contida nas seguintes subseções: Ferramentas; Conjunto de Dados; Modelos ARIMA; Modelos de Redes Neurais; Métricas de Avaliação.

6.1. Ferramentas

Para o desenvolvimento deste artigo, utilizou-se o R com apoio do *software* R Studio em todos os processos.

O R é um sistema para computação estatística e gráficos, que fornece uma linguagem de programação, gráficos de alto nível e interfaces com outras linguagens [Team 2000].

Já o R Studio é um *software* aberto, de ambiente de desenvolvimento, que fornece um conjunto de ferramentas e facilidades para a utilização do R [Van der Loo 2012].

6.2. Conjunto de Dados

Esta subseção contém a descrição do conjunto de dados utilizado, apresentando sua origem, momento de coleta e algumas características destes dados.

6.2.1. Origem e Momentos de Coleta

Os dados foram obtidos através do portal Our World In Data, uma plataforma especializada em publicar e expor dados de maneira interativa utilizando ferramentas *online* de visualização de dados através de mapas e gráficos. O portal foi originalmente desenvolvido por Max Roser, historiador social e economista, e é mantido por uma equipe de pesquisadores sob tutela da Universidade de Oxford. O *site* pode ser acessado através do endereço https://ourworldindata.org, onde todo e qualquer dado pode ser baixado gratuitamente.

Neste trabalho utilizou-se o conjunto de dados *Coronavirus Pandemic (COVID-19)* [Ritchie et al. 2020]. Uma vez que os dados são atualizados diariamente, o *download* dos mesmos foi dividido em dois momentos diferentes: *download* no dia 22 de Maio de 2021, e no dia 01 de Julho de 2021. Os dados do primeiro momento (22 de Maio) foram utilizados apenas para o treinamento dos modelos, enquanto que os dados do segundo momento (01 de Julho) foram utilizados como conjunto de testes.

6.2.2. Características Gerais dos Dados de Treino

O conjunto de dados de treino, baixado em 22 de Maio de 2021, contém um total de 90532 observações com 59 variáveis cada. Todas as observações estão relacionadas a uma data e localização específicas, e associadas a variáveis com informações econômicas (e.g. renda per capita), sociais (e.g. Índice de Desenvolvimento Humano) e epidemiológicas (e.g. quantidade de novas mortes, quantidade de novos infectados).

Entre todas as 59 variáveis, destacam-se as seguintes:

- *location*: corresponde ao país em que a observação foi feita - 226 países ao todo;
- *date*: corresponde a data da observação;
- *total_cases*: número total de casos de covid-19 reportados em uma determinada *location* até a data correspondente;
- *new_cases*: número de novos casos de covid-19 reportados em uma determinada *location* na data correspondente;
- *total_deaths*: número total de óbitos por covid-19 reportados em uma determinada *location* até a data correspondente;
- *new_deaths*: número de novos casos de óbito por covid-19 reportados em uma determinada *location* na data correspondente;
- *total_vaccinations*: número total de vacinas de covid-19 apicadas em uma determinada *location* até a data correspondente;
- *new_vaccinations*: número total de vacinas de covid-19 apicadas em uma determinada *location* na data correspondente.

Filtrando somente as instâncias referentes ao Brasil e descartando as variáveis não mencionadas acima, utilizou-se a função *vis_guess()* do R passando o *dataframe* de COVID-19 como parâmetro para observar os tipos e os valores faltantes nas observações. O resultado pode ser observado na Figura 1.

É possível notar que os tipos inferidos pela função do R estão de acordo com o tipo de informação que essas variáveis fornecem. As variáveis *total_cases*, *new_cases*, *total_deaths*, *new_deaths*, *total_vaccinations*, *new_vaccinations* foram identificadas como

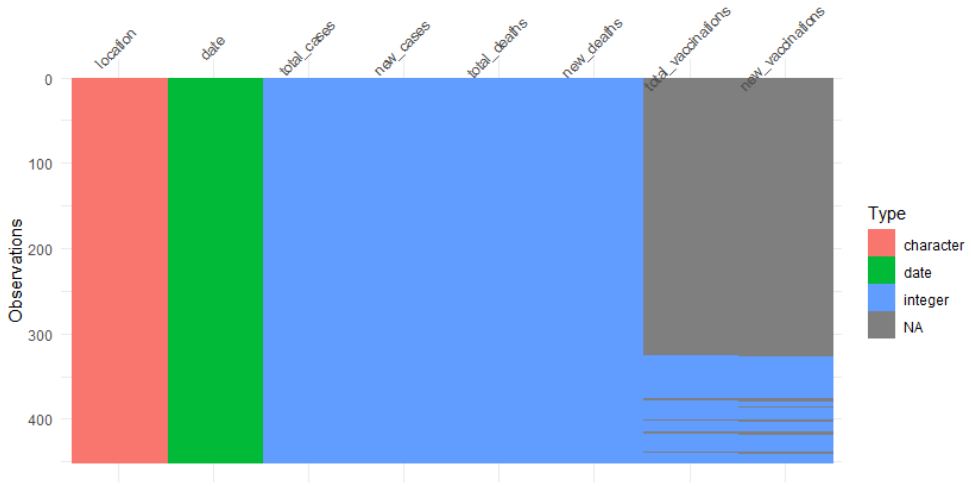


Figura 1. Resultado da execução da função *vis_guess()* sobre o *dataframe* de COVID-19 no Brasil. Foram selecionadas somente um conjunto de 8 variáveis.

integer (tipo inteiro); a variável *date* foi corretamente identificada como do tipo data (*date*) no padrão americano (e.g. mês-dia-ano); e, por fim, a variável *location* foi identificada como cadeia de caracteres.

Além dos tipos identificados, observa-se que a maior presença de dados faltantes (*NA*) estão nas variáveis relacionadas a vacinação. Isso faz bastante sentido, uma vez que a vacinação no Brasil só teve início em 17 de Janeiro de 2021.

Para observar a intersecção entre variáveis com os valores faltantes, utilizou-se a função *gg_miss_upset()* do R passando o mesmo *dataframe* anterior. O resultado pode ser observado na Figura 2.

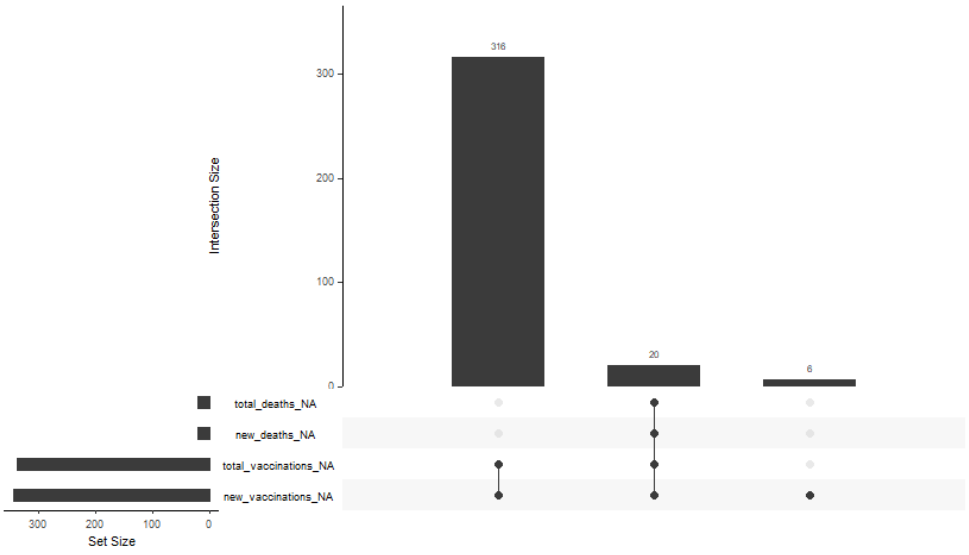


Figura 2. Resultado da execução da função *gg_miss_upset()* sobre o *dataframe* de COVID-19 no Brasil. Foram selecionadas somente um conjunto de 8 variáveis.

O resultado apresentado confirma a observação realizada anteriormente sobre as variáveis de vacinação. Além disso, também é possível notar que existem apenas 20 observações em que estão faltantes os valores das variáveis *total_deaths*, *new_deaths*, *total_vaccinations* e *new_vaccinations* simultaneamente.

Após este levantamento, e tendo em vista que a variável *new_deaths* seria a variável alvo desta pesquisa, foi realizada a substituição dos valores "*NA*"dessa variável por 0. Esse processo foi realizado no R, através do código exibido na Figura 3, onde *covid19_brazil* é o *dataframe* que utilizamos nas análises até agora.

```
# Get dates with NA in 'new_deaths'
new_deaths_na_dates <- covid19_brazil[is.na(covid19_brazil$new_deaths), 'date']
# Replace NA with 0's in 'new_deaths'
covid19_brazil$new_deaths <- ifelse(covid19_brazil$date %in% new_deaths_na_dates,
                                    0, covid19_brazil$new_deaths)
# Also for 'total_deaths'
covid19_brazil$total_deaths <- ifelse(covid19_brazil$date %in% new_deaths_na_dates,
                                      0, covid19_brazil$total_deaths)
```

Figura 3. Código utilizado para substituir os valores *NA* por 0 na variável *new_deaths*

6.2.3. Características Gerais dos Dados de Teste

Diferentemente dos dados de treino, que foram coletados logo no início deste trabalho, o conjunto de teste foi obtido somente na etapa final de desenvolvimento. Isso correu porque, em sua origem, o *dataset Coronavirus Pandemic* é atualizado diariamente com novas observações, mas seguindo sempre a mesma estrutura (organização e número de variáveis).

Para este conjunto não realizamos um pré-processamento e análises tão fortes quanto para o conjunto anterior, visto que neste momento já tínhamos como objetivo a utilização somente da variável *new_deaths* e que a estrutura dos dados novos eram exatamente iguais a dos dados de treino. Portanto, já conhecíamos os dados. As únicas alterações realizadas foram para filtrar somente os casos do Brasil, ordenar os dados através da data de obervação e substituir os valores faltantes por 0 na variável *new_deaths*.

Os dados de teste possuem observações coletadas a partir de 23 de Maio de 2021 até 01 de Julho de 2021. Compreendendo cinco semanas inteiras, este conjunto é grande o suficiente para os fins deste trabalho e foi nomeado *new_deaths_test*.

6.3. Modelos ARIMA

Auto Regressive Integrated Moving Average (ARIMA) - em tradução livre, Modelo Autorregressivo Integrado Média Móvel - é a combinação do modelo autorregressivo diferenciado com o modelo de média móvel. Estes modelos visam descrever as autocorrelações presentes nos dados para realizar previsões em séries temporais [Kotu and Deshpande 2019].

Um modelo autorregressivo (o AR da sigla ARIMA) realiza previsões sobre uma determinada variável de interesse utilizando uma combinação linear dos valores passados dessa mesma variável. O termo "autocorrelação"indica uma regressão da variável com base nela mesma [Siegel 2016].

Sendo assim, podemos descrever um modelo autorregressivo de ordem p como:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + ... + \phi_p y_{t-p} + \varepsilon_t$$

Onde y_t e ε_t são, respectivamente, o valor atual e o erro aleatório em um intervalo de tempo t; c é uma constante associada aos dados; e $\phi i, i = 1, ..., p$ são os parâmetros do modelo AR(p), onde p determina o número de observações passadas necessárias para realizar a previsão do valor atual.

Já um modelo de média móvel (MA da sigla ARIMA) de ordem q, MA(q), é conceitualmente uma regressão linear do valor atual da série contra os erros de previsão anteriores. Este pode ser descrito da seguinte maneira:

$$y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + ... + +\theta_q \varepsilon_{t-q}$$

Onde novamente temos y_t e ε_t como, respectivamente, o valor atual e o erro aleatório em um intervalo de tempo t; uma constante c associada aos dados; e $\theta i, i = 1, ..., q$ como os parâmetros do modelo MA(q).

Portanto, o modelo ARIMA completo pode ser escrito da seguinte maneira:

$$y_t = I + \phi_1 y_{t-1} + \phi_2 y_{t-2} + ... + \phi_p y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + ... + \theta_q \varepsilon_{t-q}$$

Nesta descrição acrescentamos o *I*, que representa a quantidade d de diferenciações aplicadas sobre os dados com o objetivo de se obter uma série estacionária - um requisito para os modelos ARIMA [Zhang 2003].

6.4. Modelos de Redes Neurais Auto Regressivas

Rede Neural Artificial é um método de previsão construído com base em um modelo matemático inspirado na estrutura neural de organismos inteligentes. Este método permite a criação de modelos com relacionamentos não lineares complexos entre a variável de resposta e seus preditores [Hammerstrom 1993].

Um modelo de rede neural artificial pode ser entendido como uma rede de unidades de processamento (neurônios), organizada em camadas. Os preditores (*inputs*) ou neurônios de entrada formam as camadas inferiores, e as previsões (*outputs*) ou nerônios de saída formam a camada superior. Além destas, existem camadas intermediárias contendo os chamados ”neurônios ocultos”que buscam facilitar o processamento.

Cada neurônio da rede recebe um dado de entrada com os respectivos pesos, processa estes dados localmente através de uma função de ativação e então produz uma saída para a próxima camada ou com as previsões [Zhang et al. 1998].

Um modelo de rede neural simples, sem camadas ocultas, possui comportamento semelhante a uma regressão linear. Para realizar previsões com dados de uma determinada série temporal, utilizam-se dados defasados desta mesma série como entrada para o modelo de redes neurais [Hyndman 2018]. Esta técnica é conhecida como Rede Neural Autorregressiva e será utilizada neste artigo.

6.5. Métricas de Avaliação de Modelos

Nesta subseção serão apresentadas as métricas utilizadas para avaliar os modelos gerados neste artigo. O texto está dividido em dois tópicos: Análise de Resíduos e Acurácia.

6.5.1. Análise de Resíduos

Para muitos modelos de série temporal, os resíduos correspondem à diferença entre as observações e os valores ajustados. Estes resíduos são úteis para avaliar se um modelo capturou adequadamente as informações contidas nos dados. Um bom modelo produzirá resíduos com as seguintes características:

- Não correlacionados: se existe correlação entre os resíduos, então há chances de existirem informações deixadas nos resíduos que podem ser usadas nos cálculos das previsões;
- Média zero: se os resíduos não tem média zero, então poderá ocorrer algum viés nas previsões.

Qualquer modelo de previsão que não satisfaça essas características pode ser melhorado. Entretanto, isso não significa que os modelos que as satisfaçam estejam perfeitos. Além dessas características, outras duas propriedades que indicam que um modelo faz bom uso dos dados de treino são:

- Resíduos com variância constante;
- Resíduos possuem distribuição normal.

Quando atendidas, estas propriedades tornam o cálculo dos intervalos de previsão mais fáceis. Porém, um modelo que não atinja essas características não necessariamente pode ser melhorado ou apresenta algum problema.

6.5.2. Acurácia

A análise de resíduos nos dá informações relevantes sobre a qualidade de um modelo, mas não é suficiente para avaliar a qualidade das previsões. A acurácia das previsões deve ser determinada considerando o desempenho de um modelo em novos dados que não foram usados ao treiná-lo.

Portanto, após a escolha de um modelo é comum separar os dados disponíveis em dois conjuntos diferentes: conjunto de treino, utilizado para treinar o modelo e ajustar os parâmetros do método de previsão; e conjunto de testes (*hold-out set*), que é utilizado para avaliar a precisão do modelo desenvolvido. Uma vez que os dados de teste não são utilizados durante o processo de treinamento, estes devem fornecer uma indicação confiável do desempenho do modelo.

De modo geral, o tamanho do conjunto de testes corresponde a cerca de 20% do conjunto total disponível. Entretanto isso não é uma regra, e vai variar de acordo com a quantidade de dados disponíveis ou o quão distante no tempo seu modelo precisa realizar as previsões. Para este trabalho, os conjuntos foram divididos em dois momentos diferentes de coleta, conforme explanado na seção 6.2.1.

Ao analisar a atuação do modelo com os dados de teste, é preciso ter em mente os seguintes aspectos:

- Um modelo que atinge bom desempenho com dados de treino não necessariamente fará boas previsões;
- Existe risco de um sobreajuste do modelo caso muitas variáveis sejam utilizadas (não será uma preocupação neste trabalho, uma vez que apenas uma variável será utilizada);
- Produzir um modelo que se ajusta perfeitamente ao dado é tão ruim quanto falhar em identificar os padrões do mesmo.

Além destes aspectos, é preciso avaliar os erros das previsões geradas pelos modelos desenvolvidos. O ”erro”de uma previsão de um modelo é a diferença entre o valor observado e a previsão, e são portanto a parcela imprevisível da observação. Estes erros podem ser escritos da seguinte maneira:

$$e_{T+h} = y_{T+H} - \hat{y}_{T+h|T}$$

Onde os dados de treino correspondem a $y_1, ..., y_T$ e os de teste a $y_{T+1}, y_{T+2}, ...$

Tendo como base estes erros, existem diferentes maneiras de mensurar o desempenho de uma previsão, sendo alguns exemplos: *Root Mean Squared Error* (RMSE), *Mean Absolute Error* (MAE), *Root Relative Squared Error*(RRSE), etc [Mehdiyev et al. 2016]. Neste artigo, utilizaremos duas dessas métricas:

- $MAE = mean(|e_t|)$;
- $RMSE = \sqrt{mean(e_t^2)}$

Ambas se tratam de métricas populares devido a facilidade de interpretação de seus resultados. Um método ou modelo de previsão que minimiza a taxa MAE levará a previsões da mediana, enquanto que minimizar o RMSE levará a previsões da média.

7. Análise de Resultados/Discussões

Nesta seção serão exibidos os detalhes dos processos desenvolvidos e os resultados obtidos. Essa seção está dividida da seguinte maneira: Definição da Série Temporal e Análises Exploratórias; Pré-processamento dos Dados; Desenvolvimento dos Modelos; e, Comparação de Resultados.

Todos os procedimentos aqui descritos foram realizados através da linguagem R com apoio do *software* R Studio. Para assegurar a reprodutibilidade deste experimento e dos resultados aqui expostos, todos os *scripts* e *datasets* utilizados neste artigo estão publicamente disponíveis em um repositório do GitHub e podem ser acessados através deste link https://github.com/urielcaire/covid19-statistics.

7.1. Análise Exploratória da Série Temporal

Esta seção descreve não somente o processo de análise exploratória da série temporal utilizada neste trabalho, mas também o processo de definição da mesma.

7.1.1. Definição da Série Temporal

Como já foi mencionado na seção 6.2.1, utilizaremos a variável *new_deaths* como a variável alvo deste trabalho. Portanto, nossa série temporal foi criada a partir dessa única variável.

Inicialmente carregamos os dados de treino em um *dataframe* nomeado *covid19*, e em seguida ordenamos estes dados pela data de observação *date*. Após isso, capturamos a data do primeiro óbito registrado (17 de Março de 2020) e a armazenamos em uma variável chamada *day_zero*. Por fim, foi aplicado um filtro no *dataframe* para que fossem mantidas apenas as observações dessa data em diante.

A declaração de uma série temporal no R exige explicitamente uma frequência, que é o número de observações considerados até que um determinado padrão sazonal se repita. Uma vez que os nossos dados possuem observações diárias, nós podemos trabalhar com eles com uma frequência sazonal semanal (frequência=7) ou até mesmo anual (frequência=365).

A nossa série temporal foi criada com uma frequência igual a 7, ou seja, com um período sazonal semanal. Embora nossos dados não iniciem exatamente no 1º dia da semana (Domingo), podemos definir a exata semana (em relação ao ano) e dia de início (em relação a semana) ao declarar nossa série temporal no R.

A partir da variável *day_zero* obtivemos as variáveis *start_week* (semana de início) e *start_day* (dia de início) e então as utilizamos para criar a série temporal *new_deaths*, conforme código exposto na Figura 4.

```
22  # Getting the initial week
23  start_week <- as.numeric(strftime(as.Date(day_zero), format = '%U'))
24  start_week
25  # Getting initial day position in week range (1:7)
26  start_day <- as.numeric(strftime(as.Date(day_zero), format = '%w')) + 1
27  start_day
28  # Transforms data into daily TimeSeries with a weekly frequency
29  #https://otexts.com/fpp2/tspatterns.html
30  new_deaths <- ts(covid19[,c('new_deaths')], start = c(start_week, start_day),
31                   frequency = 7)
```

Figura 4. Código utilizado para declarar a série temporal *new_deaths* no R, através da função *ts()*.

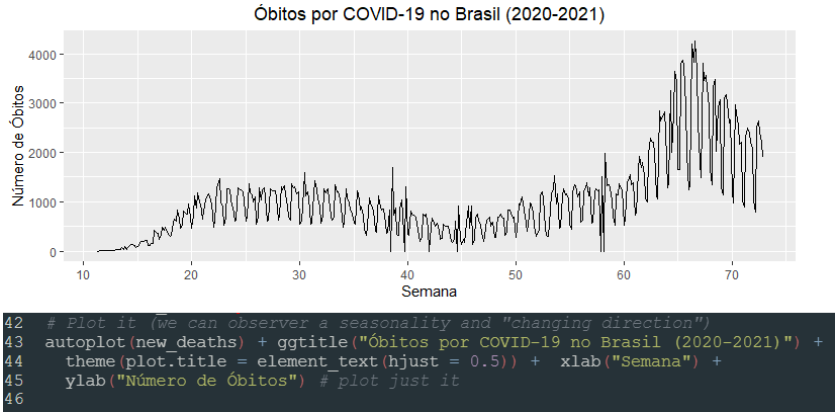
7.1.2. Análise Exploratória

Para compreender melhor a série temporal *new_deaths*, utilizamos a função *autoplot()* para visualizá-la. Tanto o código quanto o gráfico resultante podem ser vistos na Figura 5. No eixo x o gráfico apresenta as semanas, iniciando na 11ª semana de 2020 até a 72ª semana (22 de Março de 2021); enquanto que o eixo y apresenta o número de óbitos, em uma escala de 0 até mais de 4000.

Neste mesmo gráfico conseguimos observar uma possível sazonalidade, visto que existem picos e quedas que se mantém do início ao fim da série temporal e que aparentam possuir algum tipo de padrão. Além disso, podemos notar que por mais de uma vez ocorrem mudanças na direção desta série temporal. Da 11ª semana até pouco depois da 20ª há um crescimento, entre a 20ª até a 30ª ela se mantém, após isso começa a cair, e a partir da

45ª ela volta a crescer novamente.

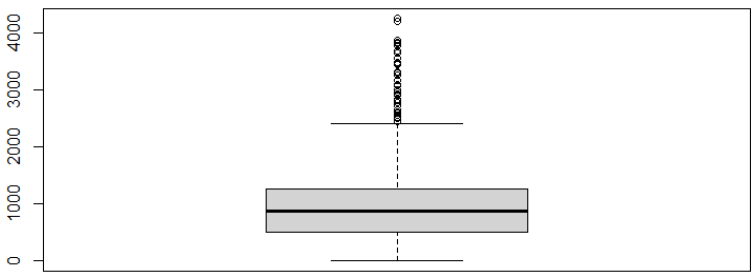
Outra observação interessante são os picos e quedas bruscas que ocorrem próximos a 40ª e 60ª semanas. É possível observar que próximos a estes pontos existem dias que atingem 0 ou até 2000 mil mortes, enquanto que os registros vizinhos apresentam comportamentos bem diferentes. Estes dados podem indicar a presença de algum tipo de coleta incorreta, para os casos de 0, ou feriados para os picos.




```
42  # Plot it (we can observer a seasonality and "changing direction")
43  autoplot(new_deaths) + ggtitle("Óbitos por COVID-19 no Brasil (2020-2021)") +
44    theme(plot.title = element_text(hjust = 0.5)) +  xlab("Semana") +
45    ylab("Número de Óbitos") # plot just it
46
```

Figura 5. A primeira imagem, no topo, apresenta o gráfico correspondente à série temporal. Logo abaixo está o código utilizado para gerar o mesmo.

Voltando o foco para os números, através do gráfico da Figura 6 conseguimos observar que valores acima de aproximadamente 2500 foram julgados como *outliers* nesta série temporal. E que o 0, apesar de provavelmente ser um erro, não foi julgado da mesma forma.




```
> summary(new_deaths)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    0.0   503.2   871.5  1037.5  1271.2  4249.0
>
```

```
47  # Print some infos about new_deaths
48  summary(new_deaths)
49  # Plot a box plot
50  boxplot(new_deaths)
```

Figura 6. A primeira imagem, no topo, apresenta o gráfico exibido pela função *boxplot()* do R. Logo abaixo, a esquerda estão o valor mínimo, o primeiro quartil, a mediana, a média, o 3 quartil e o valor máximo. Na direita é possível observar o código utilizado para gerar estes dados e o gráfico.

Com a finalidade de observar de maneira isolada os componentes da série temporal *new_deaths*, fizemos a decomposição da série utilizando a função *stl()* e a exibimos graficamente no R. A Figura 7 exibe a série já decomposta. Nesta figura podemos perceber uma sazonalidade constante, sem alterações. Além disso, percebemos também que mudanças bruscas de direção na tendência.

Embora a sazonalidade já esteja bastante clara, utilizamos a função *ggAcf()* para

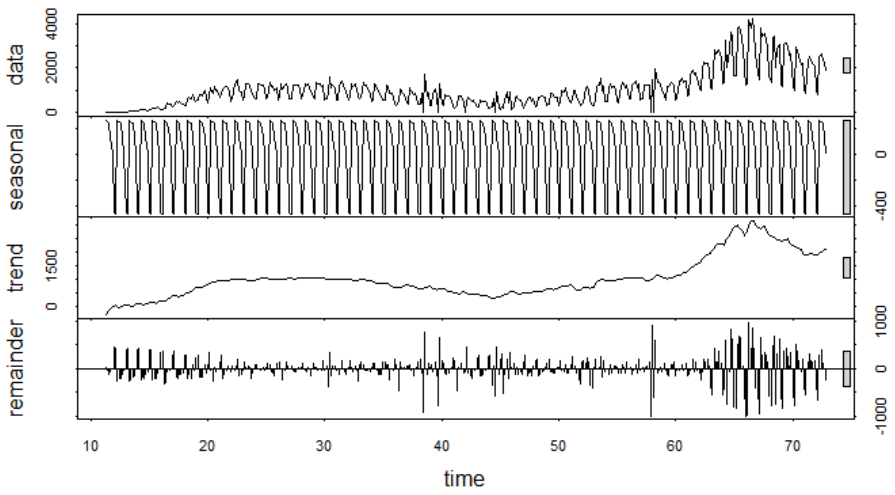


Figura 7. Resultado da decomposição da série temporal *new_deaths*.

exibir um gráfico de autocorrelação dos dos dados. Observa-se na Figura 8 que as pontas ou picos ultrapassam todos os limites em azul, o que confirma a autocorrelação dos dados. Além disso, essa autocorrelação é mais forte em múltiplos de 7 e vai diminuindo conforme os atrasos (*lags*) aumentam, o que indica, respectivamente, sazonalidade e alguma tendência.

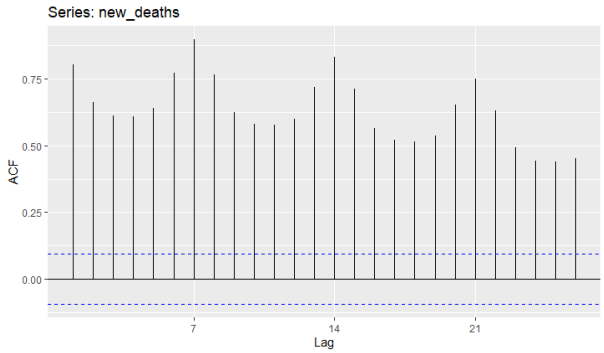


Figura 8. Correlograma da série temporal *new_deaths*.

7.2. Pré-processamento da Série Temporal

Com o objetivo de remover os zeros, visto que é bastante provável que sejam valores incorretos, os mesmos foram substituídos pela média das 14 observações mais próximas. Dessas observações, 7 eram de dias anteriores e as outras 7 de dias posteriores. Inicialmente essa alteração foi realizada em uma cópia da série temporal, nomeada *new_deaths_p*. O código R utilizado para realizar esta alteração é exibido na Figura 9.

Além dessa alteração, uma nova série temporal chamada *diff_new_deaths_train* foi criada com base na série *new_deaths_p* para testar a utilização de um modelo ARIMA não sazonal. Uma vez que os dados possuem sazonalidade, existe a necessidade de aplicar algum processo de diferenciação para chegar a um dado adequado a este método. Porém, não vamos utilizar somente este método, e por isso criamos a segunda variável.

Ao final, a série *new_deaths_p* foi renomeada para *new_deaths_train*.

```
# Replace zeros with 14-day average deaths
0 %in% new_deaths_test # FALSE
0 %in% new_deaths # TRUE
length(new_deaths)
new_deaths_p <- new_deaths
for(i in 1:length(new_deaths_p)) {
  if(new_deaths_p[i]==0){
    past <- sum(new_deaths_p[(i-7):(i-1)])
    post <- sum(new_deaths_p[(i+1):(i+7)])
    new_deaths_p[i] <- as.integer((past + post)/14)
  }
}
0 %in% new_deaths_p # FALSE
```

Figura 9. Código utilizado para remover os zeros da série temporal, criando valores a partir da média de 14 observações próximas.

O processo de diferenciação foi aplicado considerando que o teste *KPSS Unit Root Test* [Shin and Schmidt 1992] indicou que os dados não eram estacionários. Além disso, a função *nsdiffs()* do R também indicou a necessidade de aplicar diferenciação sobre a sazonalidade. Todo o código utilizado para para o processo de diferenciação e análise pode ser visto na Figura 10.

```
# kpss.test() also useful
new_deaths_train %>% ur.kpss() %>% summary() # p-value: 3.2307; not stationary.
#apply diff and test again
diff(new_deaths_train) %>% ur.kpss() %>% summary() # p-value: 0.0357; stationary.
#determining whether seasonal differencing is required
nsdiffs(new_deaths_train)# returned 1; so it is required too.
diff(diff(new_deaths_train),7) %>% ur.kpss() %>% summary()
#get diff data
diff_new_deaths_train <- diff(diff(new_deaths_train),7)
```

Figura 10. Código com o processo de teste de estacionariedade e aplicação de diferenciação.

7.3. Desenvolvimento dos Modelos

Três modelos foram criados para comparação de resultados, sendo eles: modelo ARIMA não Sazonal, modelo ARIMA Sazonal e modelo de Redes Neurais.

7.3.1. Modelo Arima Não Sazonal

Este modelo foi criado utilizando a série temporal diferenciada *diff_new_deaths_train*. Os parâmetros *stepwise* e *approximation* foram ambos configurados como falso, para ter certeza de que o modelo selecionado pela função *auto.arima()* seria o melhor entre os casos possíveis. O código para a criação do modelo pode ser observado na Figura 11, bem como algumas métricas do modelo resultante.

```
#Non-Seasonal Arima Model
#R uses maximum likelihood estimation (MLE) to estimate ARIMA models
#https://otexts.com/fpp2/arima-estimation.html
ns_arima_model <- auto.arima(diff_new_deaths_train, seasonal=FALSE, stepwise=FALSE,
                             approximation = FALSE)
ns_arima_model
> ns_arima_model
Series: diff_new_deaths_train
ARIMA(2,0,3) with zero mean

Coefficients:
         ar1      ar2     ma1      ma2      ma3
      -1.4547  -0.6241  0.8565  -0.6616  -0.8789
s.e.   0.0457   0.0467  0.0298   0.0440   0.0271

sigma^2 estimated as 71697:  log likelihood=-2971.24
AIC=5954.47   AICc=5954.67   BIC=5978.77
>
```

Figura 11. Criação do modelo ARIMA Não Sazonal.

Através da função *checkresiduals()* nós avaliamos os dados residuais do modelo gerado. Os resíduos possuíam média zero, porém falhou no teste Ljung-Box [Burns 2002] com um *p-value* de 1.346e-10 indicando a presença de autocorrelação. Os gráficos resultantes da aplicação dessa função podem ser observados na Figura; 12.

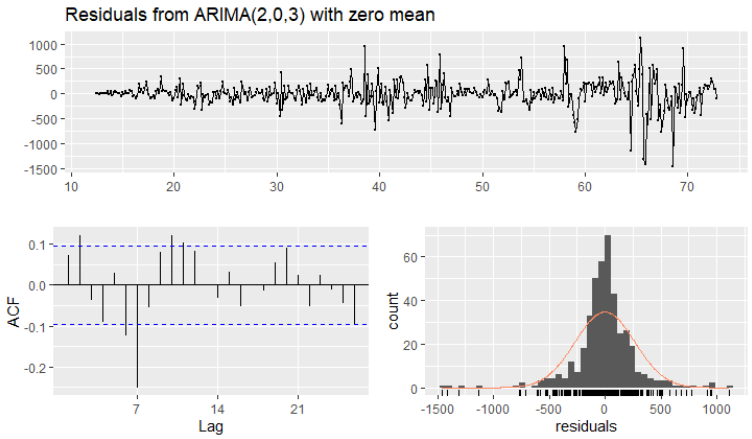


Figura 12. Análise de resíduos do modelo ARIMA não sazonal. Gráficos gerados pela função *checkresiduals()*.

7.3.2. Modelo Arima Sazonal

Diferentemente do modelo descrito na subseção anterior, neste modelo utilizou-se a série temporal principal *new_deaths_train*. Além disso, ainda configuramos o parâmetro *seasonal* como verdadeiro. Assim como no anterior, parâmetros *stepwise* e *approximation* foram ambos configurados como falso pelos mesmos motivos. A Figura 13 exibe o código e as métricas do modelo resultante.

```
272  #Seasonal Arima Model
273  sea_arima_model <- auto.arima(new_deaths_train, seasonal=TRUE, stepwise=FALSE,
274                                approximation = FALSE)
275  sea_arima_model
> sea_arima_model
Series: new_deaths_train
ARIMA(1,0,2)(1,1,1)[7]

Coefficients:
         ar1      ma1      ma2    sar1     sma1
      0.9798  -0.5643  -0.2081  0.1344  -0.7024
s.e.  0.0115   0.0475   0.0432  0.0822   0.0627

sigma^2 estimated as 60220:  log likelihood=-2940.96
AIC=5893.92   AICc=5894.12   BIC=5918.24
>
```

Figura 13. Criação do modelo ARIMA Sazonal.

Através da função *checkresiduals()* nós avaliamos os dados residuais do modelo ARIMA sazonal. Este modelo não apresentou média 0 em seus residuais, porém passou no teste Ljung-Box com um *p-value* de 0.07 indicando que podemos descartar a hipótese de que os resíduos estejam correlacionados. Os gráficos resultantes da aplicação dessa função podem ser observados na Figura 14.

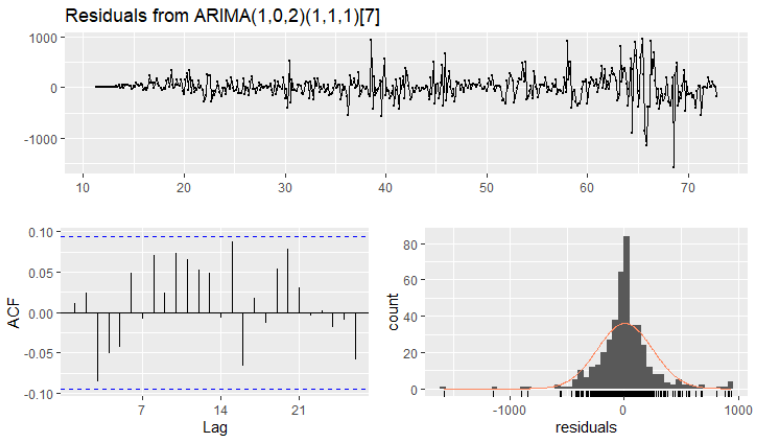


Figura 14. Análise de resíduos do modelo ARIMA Sazonal. Gráficos gerados pela função *checkresiduals()*.

7.3.3. Modelo de Redes Neurais

Neste modelo utilizou-se a série temporal principal. O modelo foi criado através da função *nnetar()*, conforme Figura 15.

```
290 ############################################################
291   # Neural network models
292   #https://otexts.com/fpp2/nnetar.html
293
294   neural_model <- nnetar(new_deaths_train)
295   neural_model
```

```
> neural_model <- nnetar(new_deaths_train)
> neural_model
Series: new_deaths_train
Model:  NNAR(21,1,11)[7]
Call:   nnetar(y = new_deaths_train)

Average of 20 networks, each of which is
a 21-11-1 network with 254 weights
options were - linear output units

sigma^2 estimated as 4942
>
```

Figura 15. Criação do modelo de Redes Neurais.

Apesar de exibir um ponto fora dos limites no gráfico de correlações, o mesmo atingiu um *p-value* de 0.2 no teste Box-Ljung, o que nos possibilita assumir que os resíduos podem não estar correlacionados. A Figura 16 exibe a análise residual deste modelo através da função *checkresiduals()*.

7.4. Comparação dos Resultados

Os modelos ARIMA Sazonal e de Redes Neurais não apresentaram média zero, mas demonstraram que não há correlação entre seus resíduos. De modo oposto, o modelo ARIMA não Sazonal apresentou média zero mas possui correlação. Isso sugere que ainda existem possíveis melhorias que podemos fazer nas séries temporais geradas.

Para comparar a capacidade de previsão dos três modelos criados até agora, ainda que não sejam os melhores possíveis, realizamos a previsão de uma semana para todos. A Figura 17 exibe as respectivas previsões. É bastante perceptível que o modelo ARIMA Não Sazonal não se adequou aos dados e fez uma previsão bem fora do padrão. Por outro

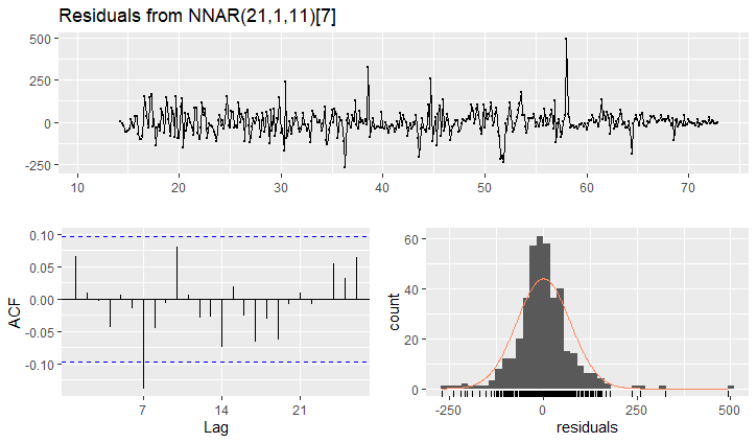


Figura 16. Análise de resíduos do modelo de Rede Neural. Gráficos gerados pela função *checkresiduals()*.

lado, os modelos ARIMA Sazonal e de Redes Neurais obtiveram resultados bem próximos.

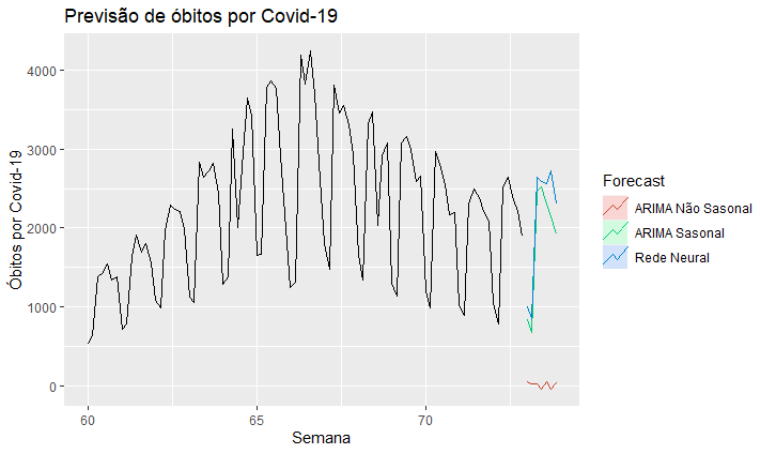


Figura 17. Comparação das previsões realizadas pelos três modelos desenvolvidos.

Para desempatar, observamos as métricas AME e RMSE dos dois modelos através da função *accuracy()*. Conforme Tabela 1, o modelo ARIMA Sazonal obteve os melhores resultados, ainda que estes valores estejam muito altos.

Modelo	MAE	RMSE
Redes Neurais	262.40233	291.70732
ARIMA Sazonal	132.8881	161.2301

Tabela 1. Métricas de avaliação dos modelos de Redes Neurais e ARIMA Sazonal.

Exibindo na Figura 18 a previsão realizada pelo modelo (em vermelho) em comparação com o dado real, podemos perceber que ele flutuou próximo ao valor real (em verde). Porém, como já foi evidenciado, ainda existem pontos de melhoria para este modelo.

8. Considerações Finais/Conclusões

O melhor desempenho ficou com o modelo ARIMA Sazonal. Entretanto, até mesmo este modelo apresenta taxas ainda muito altas de MAE e RMSE. E além disso, seus resíduos indicam que ainda existem pontos de melhorias.

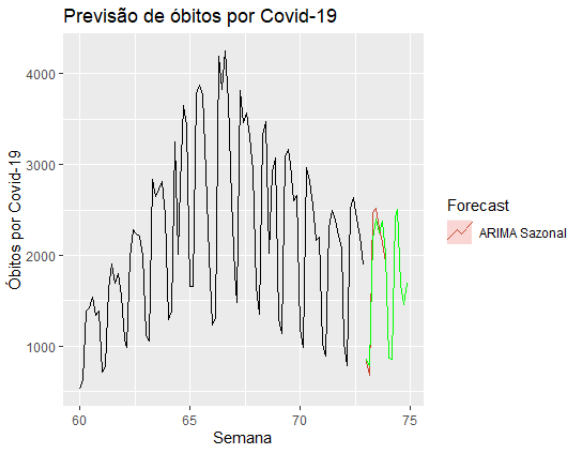


Figura 18. Análise de resíduos do modelo de Rede Neural. Gráficos gerados pela função *checkresiduals()*.

Como continuação deste trabalho, poderiam ser inclusas novas etapas na fase de pré-processamento. Tais como: alguma suavização dos dados, com o objetivo de remover os picos que encontramos e até aqueles que não visualizamos tão claramente nos gráficos; redução do tamanho da série para que os dados fiquem melhor distribuídos, criando uma série com somente os dados do ano de 2021, por exemplo.

Estas alterações poderiam contribuir para uma melhor adaptação dos métodos aos dados, e consecutivamente para a criação de previsões ainda mais satisfatórias.

Referências

[Burns 2002] Burns, P. (2002). Robustness of the ljung-box test and its rank equivalent. *Available at SSRN 443560*.

[Ehlers 2007] Ehlers, R. S. (2007). Análise de séries temporais. *Laboratório de Estatística e Geoinformação. Universidade Federal do Paraná*, 1:1–118.

[Hammerstrom 1993] Hammerstrom, D. (1993). Working with neural networks. *IEEE Spectrum*, 30(7):46–53.

[Hyndman 2018] Hyndman, R. J. (2018). George athanasopoulos. *Forecasting: Principles and Practice. Monash University, Australia*.

[Kotu and Deshpande 2019] Kotu, V. and Deshpande, B. (2019). Chapter 12 - time series forecasting. In Kotu, V. and Deshpande, B., editors, *Data Science (Second Edition)*, pages 395–445. Morgan Kaufmann, second edition edition.

[Mehdiyev et al. 2016] Mehdiyev, N., Enke, D., Fettke, P., and Loos, P. (2016). Evaluating forecasting methods by considering different accuracy measures. *Procedia Computer Science*, 95:264–271. Complex Adaptive Systems Los Angeles, CA November 2-4, 2016.

[Ritchie et al. 2020] Ritchie, H., Ortiz-Ospina, E., Beltekian, D., Mathieu, E., Hasell, J., Macdonald, B., Giattino, C., Appel, C., Rodés-Guirao, L., and Roser, M. (2020). Coronavirus pandemic (covid-19). *Our World in Data*. https://ourworldindata.org/coronavirus.

[Shin and Schmidt 1992] Shin, Y. and Schmidt, P. (1992). The kpss stationarity test as a unit root test. *Economics Letters*, 38(4):387–392.

[Siegel 2016] Siegel, A. F. (2016). Chapter 14 - time series: Understanding changes over time. In Siegel, A. F., editor, *Practical Business Statistics (Seventh Edition)*, pages 431–466. Academic Press, seventh edition edition.

[Souza et al. 2020] Souza, C. D. F. d., Paiva, J. P. S. d., Leal, T. C., Silva, L. F. d., and Santos, L. G. (2020). Evolução espaçotemporal da letalidade por covid-19 no brasil, 2020.

[Team 2000] Team, R. C. (2000). R language definition. *Vienna, Austria: R foundation for statistical computing*.

[Van der Loo 2012] Van der Loo, M. P. (2012). *Learning RStudio for R statistical computing*. Packt Publishing Ltd.

[Werneck and Carvalho 2020] Werneck, G. L. and Carvalho, M. S. (2020). A pandemia de covid-19 no brasil: crônica de uma crise sanitária anunciada.

[Zhang 2003] Zhang, G. (2003). Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, 50:159–175.

[Zhang et al. 1998] Zhang, P., Patuwo, E., and Hu, M. (1998). Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting*, 14:35–62.