Princess Nourah bint Abdulrahman University
College of Computer & Information Sciences
Computer Science Department

Time Series and Forecast, DS424- Project

Analysis and Forecasting of COVID-19 New Cases in South America

ID	Name	Task
442002988	Monerah Almobarak	Did the Preparing Data, helped in Data Visualization and Performance Evaluation
442005420	Alaa Alshuaibi	Data Visualization, Producing Forecast
442000786	Sarah Altaweel	Data Visualization, Producing Forecast
442005104	Sarah Aljuhani	Specifying Model , Model Estimation
442003374	Nada Alotaibi	Specifying Model, Model Estimation

1. Preparing Data

Objective:

The objective of this data preparation process is to transform COVID-19 data for South America from a raw CSV file into a structured and clean dataset suitable for further analysis. By addressing missing values, selecting pertinent columns, summarizing data, and converting it into a tsibble object, we aim to prepare the data for predictive analysis, allowing us to anticipate and be better prepared for future cases and their implications [1].

preparation process:

The data preparation process begins by reading the raw COVID-19 data for South America from a CSV file and creating two separate datasets for further analysis. The first dataset, referred to as "Data," focuses on the continent level, and the second dataset, named "Datawithlocation," includes location-specific information. This separation was made to address potential issues related to location data [2].

For both datasets, we started by checking for missing values using the "is.na" function and stored the count of missing values in the variable "missing_values." If there were any missing values, we printed a message indicating the number of missing values; otherwise, we printed that the dataset had no missing values. This step is essential to ensure data completeness and quality [3].

In the case of missing values, we used the "na.omit" function to remove the rows containing missing data. This step is necessary to prevent errors and maintain data integrity [3].

For the "Data" dataset, we selected specific columns, including "continent," "new_cases," and "date," using the "select" function. These columns are relevant for our analysis. Subsequently, we grouped the data by "continent" and "date" using the "group_by" function to facilitate summarization [3].

To summarize the data, we used the "summarise" function with the "sum" function to calculate the total new cases for each combination of continent and date. This aggregation step helps provide a concise view of the data and is important for time series analysis [3].

The "Datawithlocation" dataset followed a similar process, but in this case, we included the "location" column when selecting columns. The data was grouped by "continent," "date," and "location" to capture location-specific information, and the total new cases were summarized accordingly [2].

Finally, to prepare the data for time series analysis, we converted the "date" column from character format to a date format using the "as_date" function. This ensures that the date data can be used effectively in time series analysis. Both datasets were then transformed into tsibble objects, with the key set as "continent" for "Data" and as a combination of "continent" and "location" for "Datawithlocation." [4]

Note:

That we are converting the data to a daily frequency instead of a monthly frequency for better insights because our dataset will contain only 38 rows if it's aggregated on a monthly basis. Converting to a daily frequency provides more granular insights into the data, allowing for a more detailed analysis.

```
> Data
# A tibble: 1,357 × 3
# Groups:   continent [1]
   continent     date       total_new_cases
   <chr>         <date>               <dbl>
 1 South America 2020-01-03               0
 2 South America 2020-01-04               0
 3 South America 2020-01-05               0
 4 South America 2020-01-06               0
 5 South America 2020-01-07               0
 6 South America 2020-01-08               0
 7 South America 2020-01-09               0
 8 South America 2020-01-10               0
 9 South America 2020-01-11               0
10 South America 2020-01-12               0
# … with 1,347 more rows
```

Figure 1. Data tibble - First 10 rows after preprocessing.

```
> Datawithlocation
# A tibble: 19,089 × 4
# Groups:   continent, date [1,364]
   continent     date       location         total_new_cases
   <chr>         <date>     <chr>                      <dbl>
 1 South America 2020-01-03 Argentina                      0
 2 South America 2020-01-03 Bolivia                        0
 3 South America 2020-01-03 Brazil                         0
 4 South America 2020-01-03 Chile                          0
 5 South America 2020-01-03 Colombia                       0
 6 South America 2020-01-03 Ecuador                        0
 7 South America 2020-01-03 Falkland Islands               0
 8 South America 2020-01-03 French Guiana                  0
 9 South America 2020-01-03 Guyana                         0
10 South America 2020-01-03 Paraguay                       0
# … with 19,079 more rows
```

Figure 2. Datawithlocation tibble - First 10 rows after preprocessing with location.

```
> Data_tsibble
# A tsibble: 1,364 x 3 [1D]
# Key:       continent [1]
# Groups:    continent [1]
   continent     date       total_new_cases
   <chr>         <date>               <dbl>
 1 South America 2020-01-03               0
 2 South America 2020-01-04               0
 3 South America 2020-01-05               0
 4 South America 2020-01-06               0
 5 South America 2020-01-07               0
 6 South America 2020-01-08               0
 7 South America 2020-01-09               0
 8 South America 2020-01-10               0
 9 South America 2020-01-11               0
10 South America 2020-01-12               0
# … with 1,354 more rows
```

Figure 3. Data_tsibble tsibble - First 10 rows after preprocessing.

```
> tsibblewithlocation
# A tsibble: 19,089 x 4 [1D]
# Key:       continent, location [14]
# Groups:    continent @ date [1,364]
   continent     date       location  total_new_cases
   <chr>         <date>     <chr>               <dbl>
 1 South America 2020-01-03 Argentina               0
 2 South America 2020-01-04 Argentina               0
 3 South America 2020-01-05 Argentina               0
 4 South America 2020-01-06 Argentina               0
 5 South America 2020-01-07 Argentina               0
 6 South America 2020-01-08 Argentina               0
 7 South America 2020-01-09 Argentina               0
 8 South America 2020-01-10 Argentina               0
 9 South America 2020-01-11 Argentina               0
10 South America 2020-01-12 Argentina               0
# … with 19,079 more rows
```

Figure 4. Tsibblewithlocation tsibble - First 10 rows after preprocessing with location.

These figures display the two tables and two tsibble objects after preprocessing the data as daily records, showcasing the first 10 rows of each.

2. Data Visualization

Lag plot:

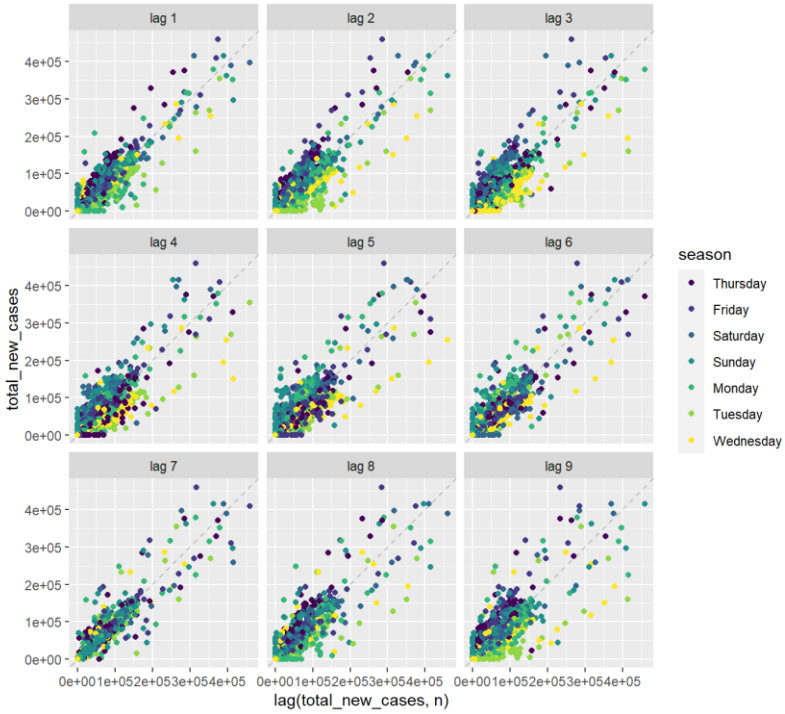

Figure 5. Gg_lag plot using data_tsibble object.

Figure 5, shows the scatterplots of daily COVID total new cases, which shows each y_t plotted against y_{t-k} where k represents the lag number. This plot colors the seasonal period, in this case is daily, to identify how each season correlates with others [5].

From the plot, it is obvious that there is a strong positive correlation between seasons indicating that there is strong seasonality.

Autocorrelation Function plots:

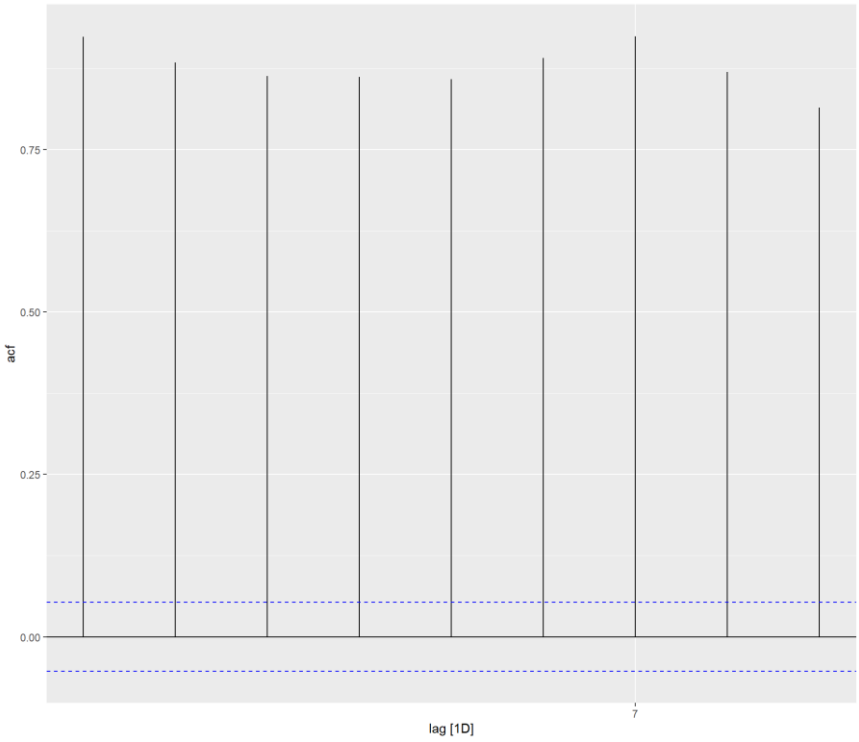

Figure 6. Acf plot with 9 using data_tsibble object.

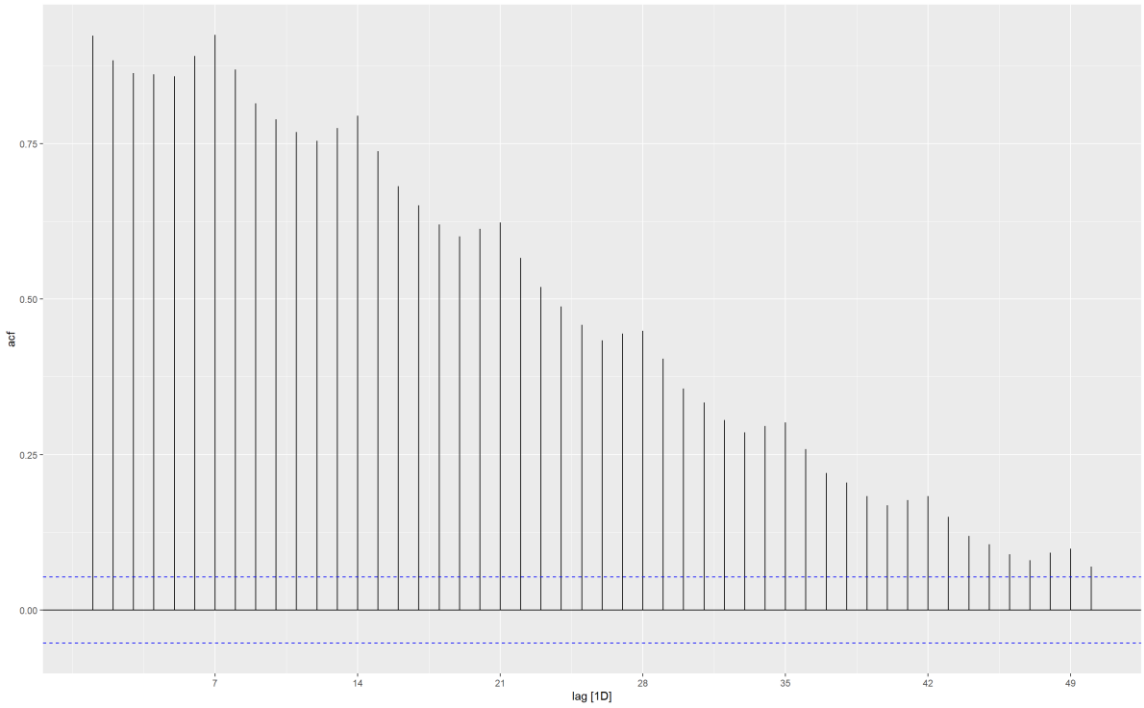

Figure 7. Acf plot with 50 lags using data_tsibble object.

Figure 6 and Figure 7 show the function autocorrelation function (acf), which computes estimates of the autocorrelation function [6]. The plots show the autocorrelation of a variable,

in this case is the total new cases of COVID, with itself separated by different lags. These plots show that there are strong positive correlations, and since all the autocorrelation values are higher than the boundary, then the correlations are significantly different from zero, which indicates that the data is not white noise. [5] In our experiment we created two plots, where Figure 6 shows 9 lags being similar to Figure 5, while Figure 7 shows 50 lags, which shows the decrease trend and seasonality.

Check for seasonality, trend, and cyclic:

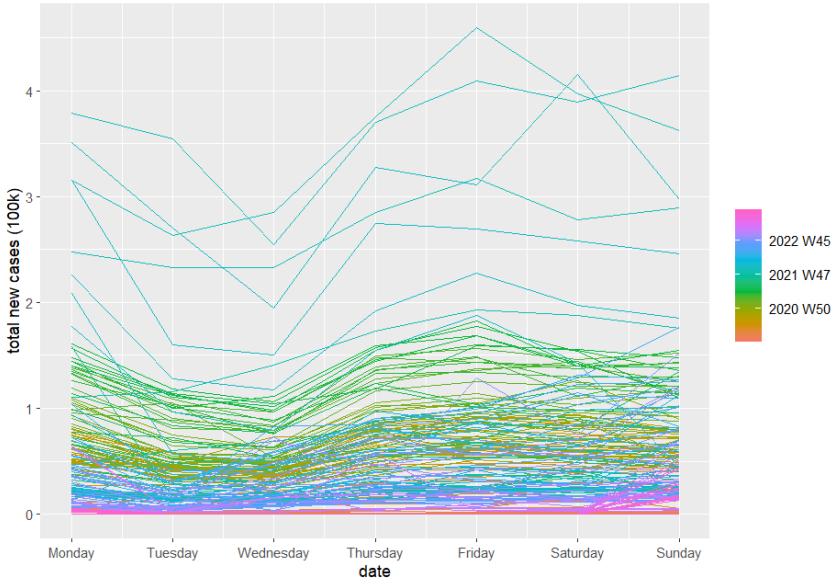

Figure *8*. Weekly season plot without location.

Figure 8, shows the seasonal plot that shows any seasonality that may exist in the time series [7]. This plot shows the weekly number of total new cases for the continent on scale of 100k. And it shows that Friday usually has the highest number of new cases, and it also shows that seven weeks in the year 2022 have the highest number of new cases.

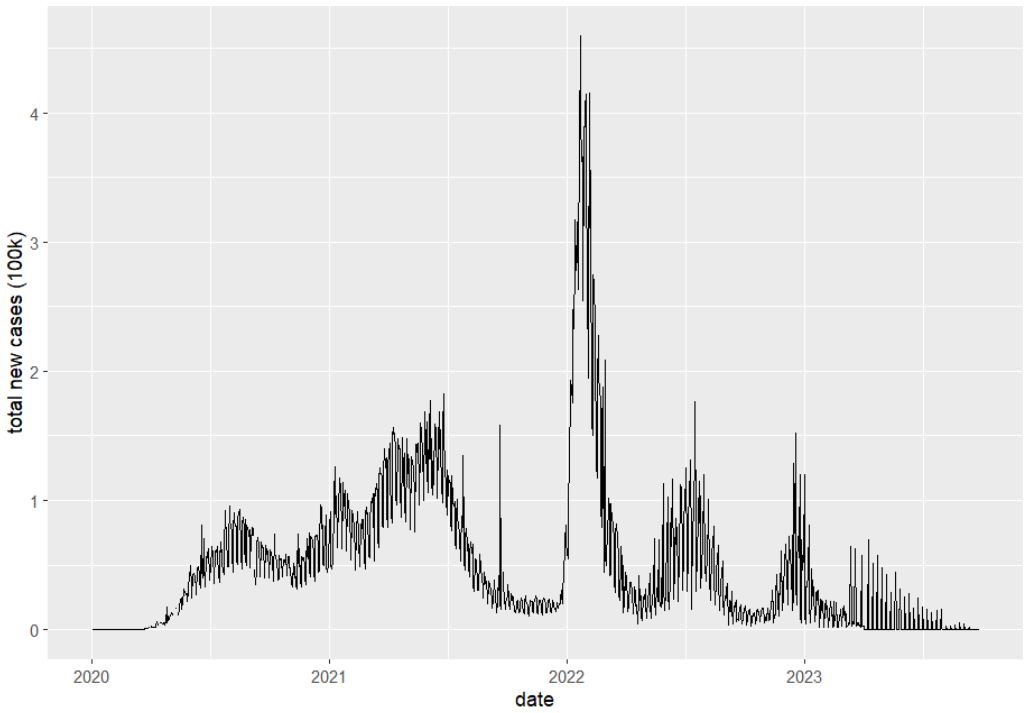

Figure 9. Yearly time plot without location.

Figure 9, this plot shows yearly number of total new cases for the continent on scale of 100k. and it shows that there is an increasing trend starting from mid-2020 until mid-2021, then the number of new cases starts going down, and at the beginning of 2022 there is a spike followed by a trough, then the number of new cases goes up on mid-2022 and on the beginning of 2023.

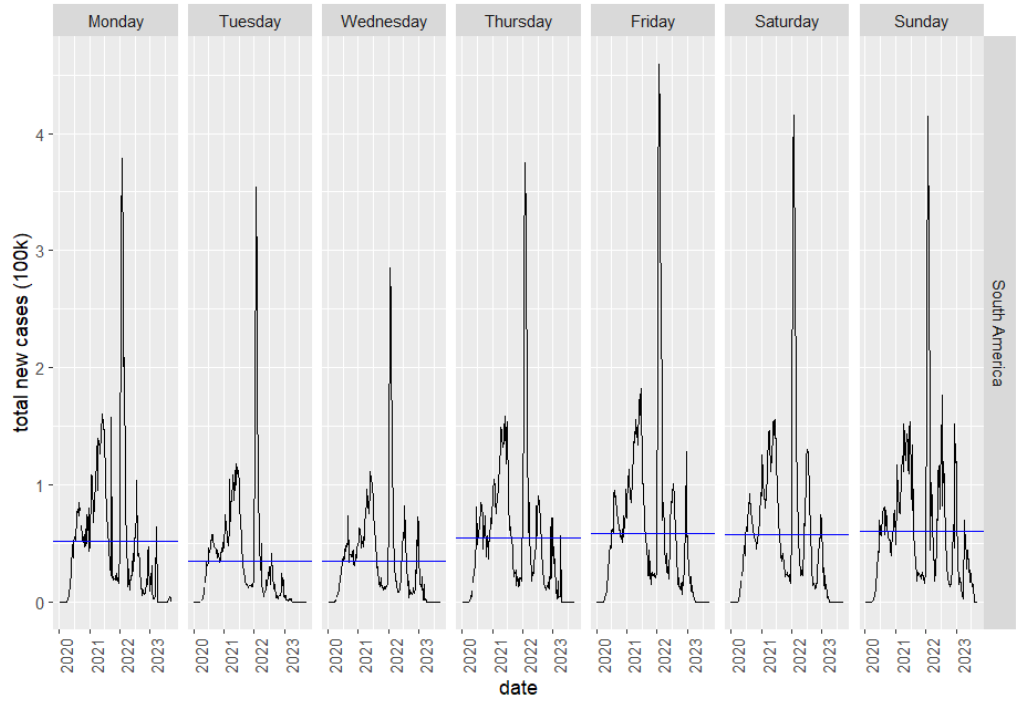

Figure 10. Weekly subseries plot without location.

Figure 10, shows the subseries plot, which facets the time series by each day in the week. They show the data from each day with the blue line as the mean [8]. This plot shows the weekly number of total new cases for the continent on scale of 100k. In this plot we can see that 2022 has the highest number of new cases. It also shows that the average number of new cases for each day is similar.

Mathematical transformations, such as logarithmic, square root, cube root, inverse, and the Box-Cox lambda transformation, offer valuable tools to enhance data visualization in time series analysis. Logarithmic transformations [9] can linearize trends, making them more apparent in plots, while square root and cube root transformations can stabilize variance and highlight cyclic behaviors. The inverse transformation helps to reverse effects and interpret data in its original scale. Applying these transformations allows for clearer visualization of trend, seasonality, and cyclic patterns and aids in the selection of appropriate modeling techniques [10].

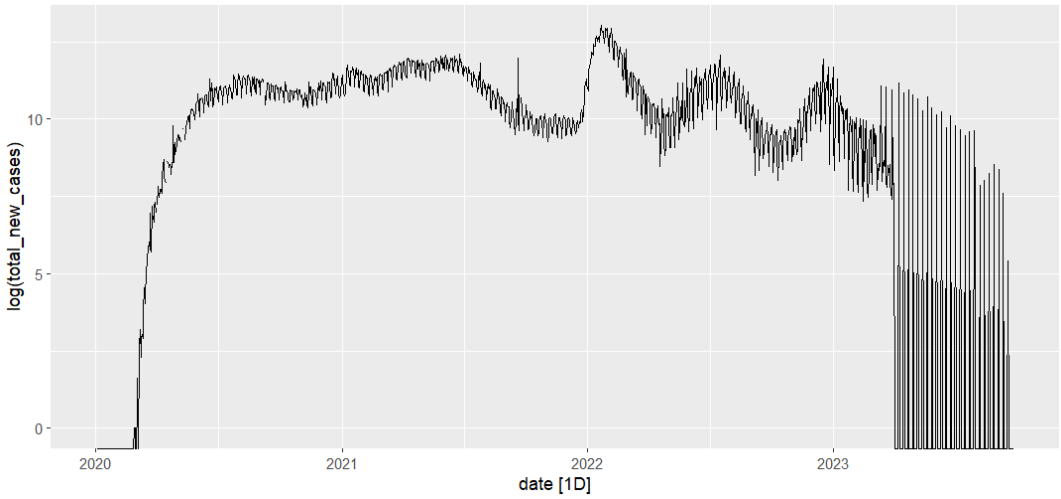

Figure 11. Log transformation.

Figure 11, after applying log transformation, we can see that it is not a good transformation for our data, because it changes it a lot.

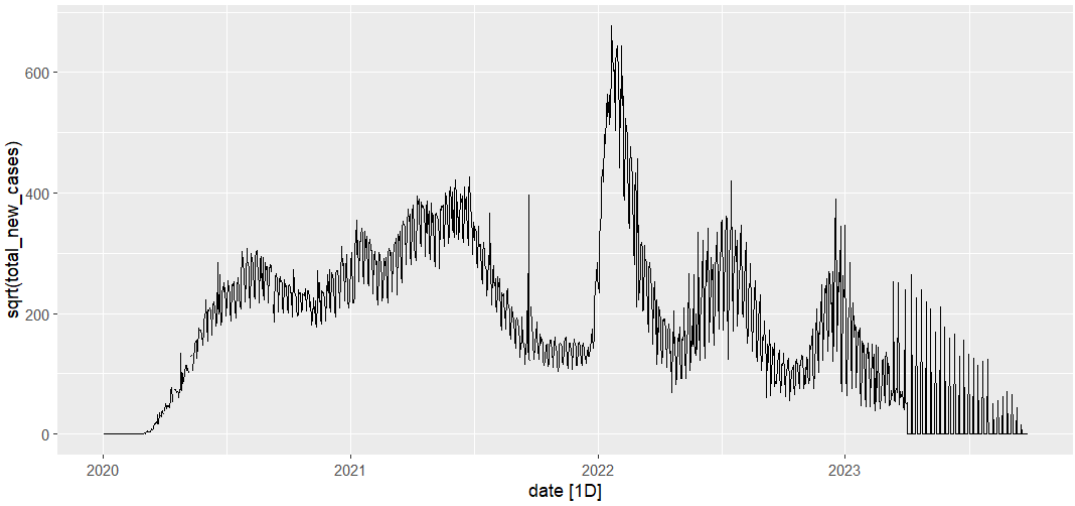

Figure 12. Square root transformation.

Figure 12, after applying square root transformation, we can see that this is better than the log transformation.

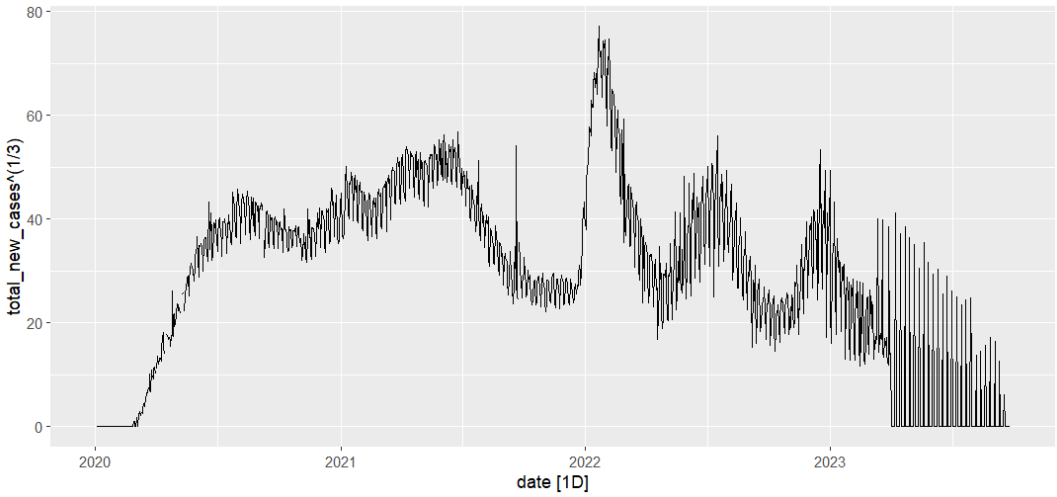

Figure 13. Cube root transformation.

Figure 13, after applying cube root transformation, we can see that the variation in the data set is less than before.

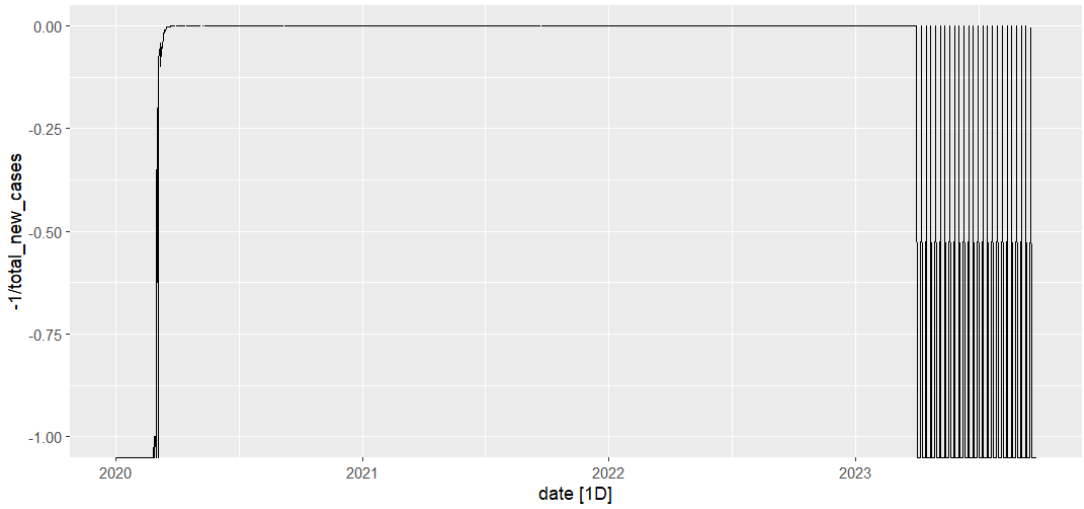

Figure 14. Inverse transformation.

Figure 14, after applying inverse transformation, this transformation does not suit our data set and it completely changes it.

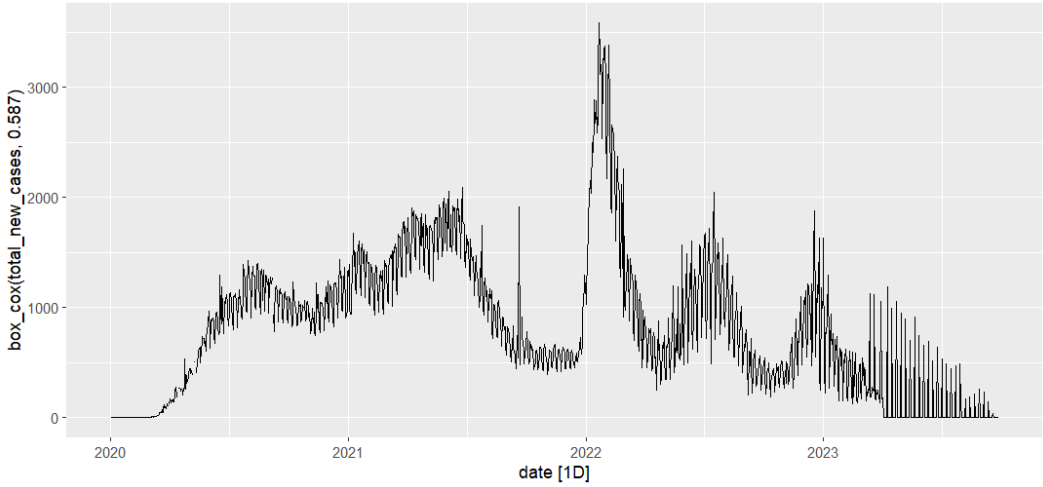

Figure 15. Time plot with the best value of lambda

Figure 15, after applying the best value of lambda which is 0.587 found using “Guerrero” method, this balances the seasonal fluctuation and random variation.

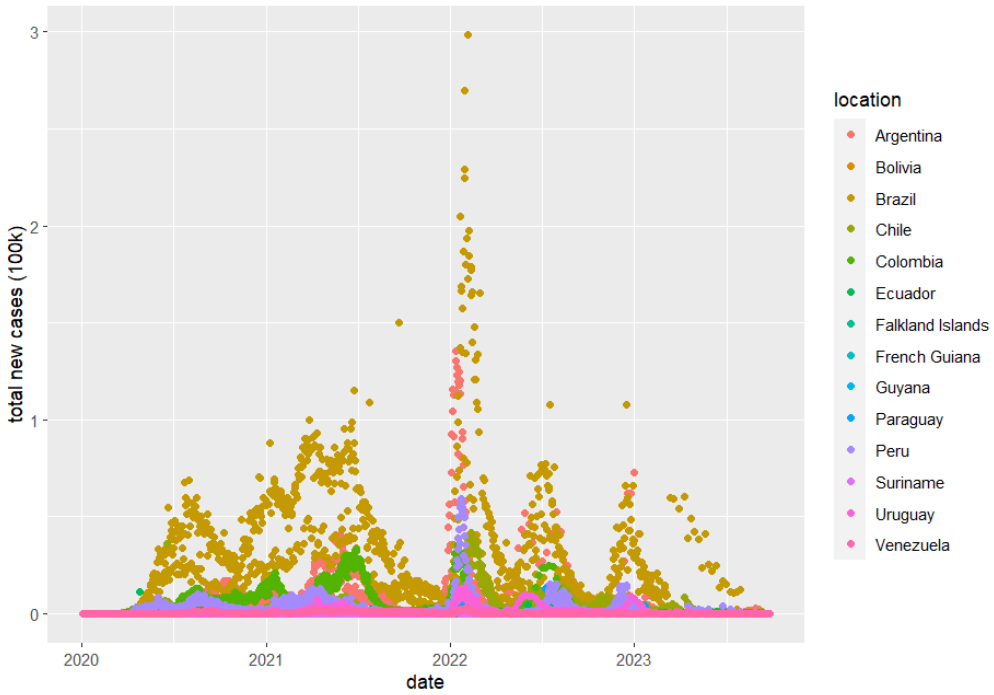

Figure 16. Point gg plot with location.

Figure 16, this plot shows the number of new cases for each country on scale of 100k, we can see that Brazil has the highest number of new cases across the years, and countries like Suriname, Uruguay, and Venezuela have the lowest number of new cases.

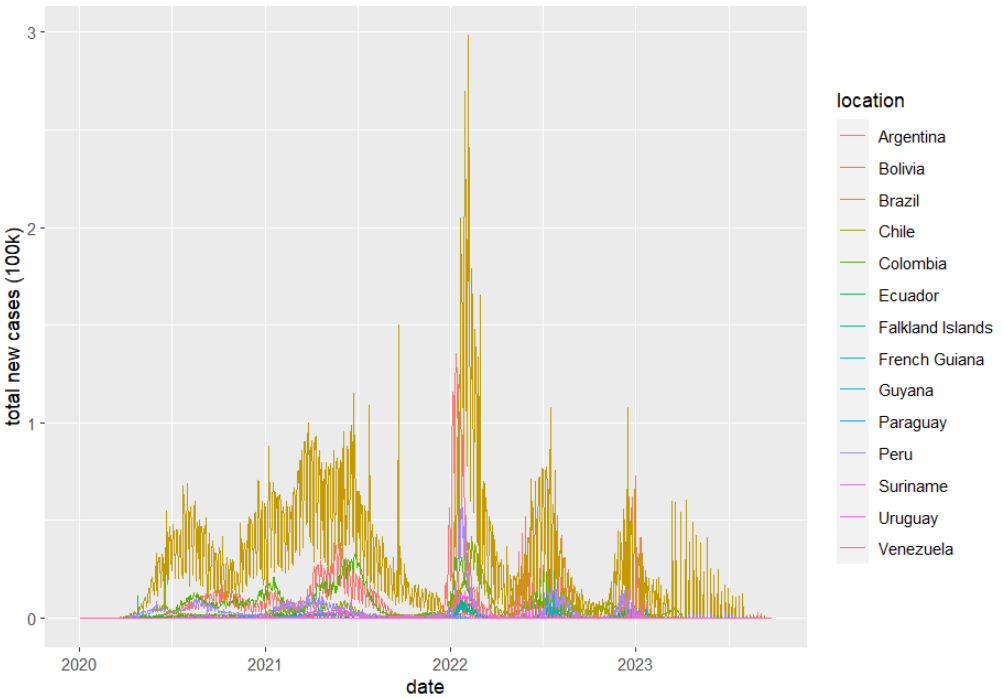

Figure 17. Line gg plot with location.

Figure 17, this is similar to the previous plot; it shows the number of new cases for each country on scale of 100k but with lines instead of points. Brazil having the highest number of new cases, and Suriname, Uruguay, and Venezuela having the lowest number of new cases.

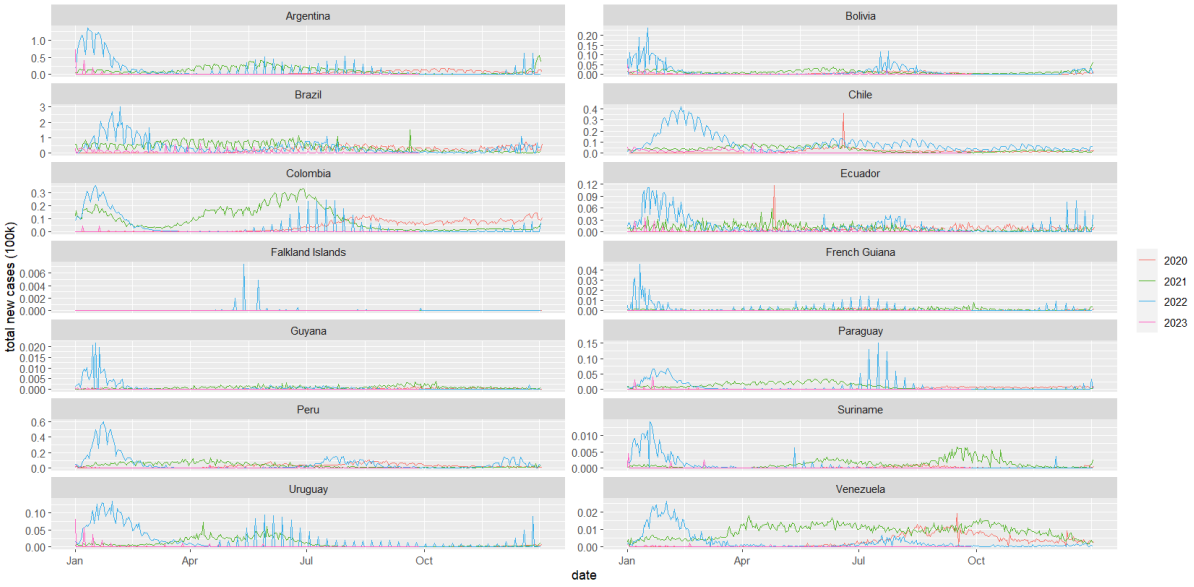

Figure 18. Gg season plot with location.

Figure 18, this plot shows the monthly number of new cases for each country each year on scale of 100k, we can see that for the country Falkland Islands has new cases on 2022 only and it is in May and July, and for Guyana there was new cases at the beginning of 2022. And like the previous plots Brazil has the highest number of cases. And for the rest of the countries, it varies from year to year.

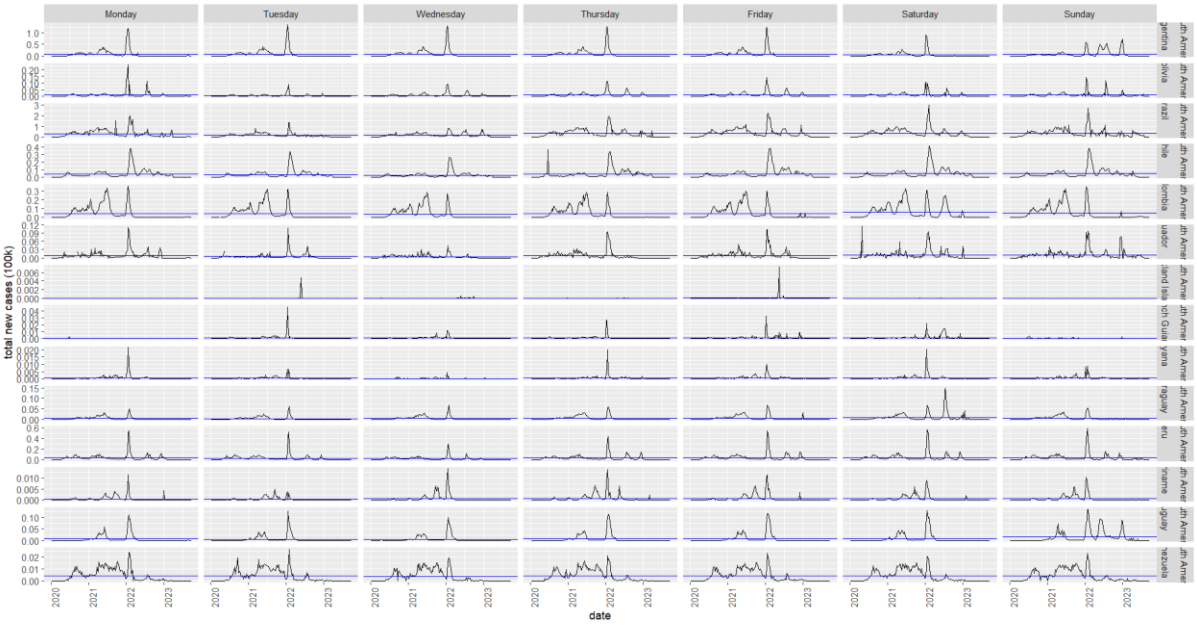

Figure 19. Gg subseries with locations.

Figure 19, this plot shows the weekly number of new cases for each country each month of each year on scale of 100k. Falkland Islands shows the least number of new cases, and Brazil the highest number of new cases. Most countries have higher number of new cases in 2022.

Note:

In our project, we opted not to employ moving average plots, as our dataset lacked a cyclical component, rendering such analysis unnecessary for our purposes.

3. Specifying Model

Seasonal Naive (Snaive) Model:

The "Seasonal Naive" or "Snaive" method is a simple and straightforward time series forecasting technique used for data that exhibits strong seasonality. This method involves making forecasts based on the most recent observed value from the same season in the previous year [7].

Justification: The model is chosen based on the robust seasonal patterns evident in the data, reflected in the significant positive correlation between seasons. It operates on the premise that forthcoming values will closely mirror the most recent observation within the same season, adeptly encapsulating recurring patterns. This choice is particularly suitable for the COVID-19 dataset, which exhibits pronounced seasonality.

Naive Method:

The "Naive Method" is one of the simplest and most straightforward time series forecasting techniques. It involves making forecasts based solely on the most recent observation in the time series. The Naive Method assumes that future values will be the same as the most recent observed value [7].

Justification: The Naive Method can serve as a straightforward benchmark and provide a quick reference point for assessing the performance of more complex models. While it does not consider seasonality, trends, or external factors, it can be useful in situations where the data is relatively stable over time or when you need a basic, rapid forecast to gain a preliminary understanding of future trends.

Regression model:

A regression model is a statistical approach used to examine and quantify the relationship between a dependent variable and one or more independent variables. It aims to model the dependency between these variables and make predictions or estimate the value of the dependent variable based on the values of the independent variables [7].

Justification: A regression model can be a suitable choice for forecasting COVID-19 data, considering the evident trends and seasonality patterns observed in the provided information. The scatterplots (Figure 5) reveal strong positive correlations between daily total new cases, highlighting significant seasonality. Figures 6 and 7 display autocorrelation functions, indicating the presence of strong positive correlations, confirming non-white noise characteristics in the data. The presence of lagged variables further supports the temporal dependencies. Furthermore, the weekly and yearly plots depict distinct patterns, such as the highest new cases on Fridays and fluctuations over time. By incorporating these features and patterns as explanatory variables, a regression model can effectively capture and forecast the complex dynamics of COVID-19 data.

TSLM (Time Series Linear Model):

A Time Series Linear Model (TSLM) is a statistical modeling approach used to analyze and forecast time series data, which is a sequence of data points collected at successive, equally spaced time intervals. TSLM is a linear regression model designed to capture and describe the underlying linear relationships within time series data. It is a straightforward method that assumes that the time series data can be represented as a linear combination of its past values [7].

Justification: The model is a flexible approach that can accommodate various components of COVID-19 data, including seasonality, trends, and external factors. The Time Series Linear Model (TSLM) effectively extends the concept of simple linear regression within the domain of time series, enabling the inclusion of both time-dependent and non-time-dependent covariates within the model. This makes TSLM an appropriate choice when distinct patterns of seasonality and cyclic behavior have been identified in the data, and when there is an anticipation of the impact of external variables on COVID-19 cases.

4. Model Estimation

To forecast the future cases of COVID-19 in South America, we employed several models and followed a structured process:

Data Splitting:

The first step in our forecasting process was to split the data into training and test sets.
This data division is crucial for time series forecasting.
We calculated the split_point to represent 80% of the total number of rows in the Data_tsibble.
This division ensured that 80% of the data was allocated to the training set, while the remaining 20% was reserved for testing the model's performance [7].

Model Training:

We then proceeded to train four different models:
Naive Model: This model, trained using the NAIVE function, assumed that the future value of the total new cases of COVID-19 is equal to the most recent observed value [11].
Seasonal Naive Model: The SNAIVE function was used to train this model, which takes into account the seasonality of the data. It predicts future values based on the corresponding values from the same season in previous years [12].
Regression Model: In this model, we employed a regression model with a constant term (total_new_cases ~ 1). This model captured both the trend and seasonality of the data [12].
Time Series Linear Model (TSLM): The TSLM function was utilized to train this model, which considered the linear relationship between the total new cases of COVID-19 and time. It could capture trend, seasonality, and other time-dependent patterns [12].

Generating Forecasts:

After training the models, we generated forecasts using the test data for evaluation purposes. These forecasts were stored in separate variables (naive_forecast, snaive_forecast, regression_forecast, tslm_forecast) for further analysis and comparison.
This comprehensive approach allowed us to assess the performance of multiple forecasting models and select the most suitable one for predicting future COVID-19 cases in South America [13].

5. Performance Evaluation

Selecting the optimal model necessitates careful consideration of the specific needs of the application and which performance metrics are of utmost importance. The assessment involves four key metrics: ME (Mean Error), RMSE (Root Mean Square Error), MAE (Mean Absolute Error), and ACF1 (Auto-Correlation Function at lag 1).

Table 1. Models' Performance

ML Algorithms	Models' Performance			
	ME	RMSE	MAE	ACF1
Naive model	-61193	63257	61651	0.325
Snaive model	-49288	60385	50358	0.0867
Regression model	-61638	73228	62660	0.149
TSLM model	-53222	55583	53887	0.325

- **ME:** A lower absolute ME is preferable, indicating less systematic bias. The Snaive model has the smallest absolute ME, suggesting more unbiased forecasts.
- **RMSE:** A lower RMSE indicates better performance. The TSLM model leads in this category with the lowest RMSE, suggesting it handles large errors more effectively.
- **MAE:** A lower MAE indicates superior performance. Here, the Snaive model reports the lowest MAE, highlighting its effectiveness in minimizing average errors.
- **ACF1:** A lower absolute value of ACF1 is desirable, implying less autocorrelation in prediction errors. The Snaive model scores the lowest ACF1.

Justification for Model Performance:

- **Snaive Model:** Despite the pronounced seasonality in COVID-19 data, the Snaive model was not the overall best performer. Although it has the lowest MAE and ACF1, indicating effectiveness in minimizing average errors and error autocorrelation, its higher RMSE compared to the TSLM model suggests limitations in handling complex interdependencies and larger errors.
- **Naive Method:** Useful as a benchmark for stable data or initial trend analysis but lacks sophistication for the complexities of COVID-19 data.
- **Regression Model:** Suitable for capturing trends and seasonality but may not fully encapsulate COVID-19 data dynamics as effectively as the TSLM model.
- **TSLM (Time Series Linear Model):** Used to fit a linear model including trend and seasonal components. Therefore, it emerges as the top contender with the lowest

RMSE, indicating its superior capability in capturing both the linear trends and the complex dynamics of COVID-19 data, including seasonality and other factors.

In summary, for overall predictive precision and handling of larger errors, the TSLM model is recommended, particularly for COVID-19 data exhibiting complex patterns of seasonality and cyclic behavior. However, if the primary focus is on reducing systematic bias or minimizing average error autocorrelation, the Snaive model's performance in these areas makes it a strong contender [14].

6. Producing Forecast

In the final step of our analysis, we generated forecasting plots for each of the models to assess their performance in relation to the actual data. To accomplish this, we utilized a forecast plot function that effectively combines historical data with forecasts and prediction intervals. [15] The forecast plot function provided a clear and intuitive visualization that allowed us to assess the alignment between the predicted values and the actual data points. Additionally, the inclusion of prediction intervals enabled us to gauge the uncertainty associated with the forecasts and assess the models' variability in capturing future outcomes.

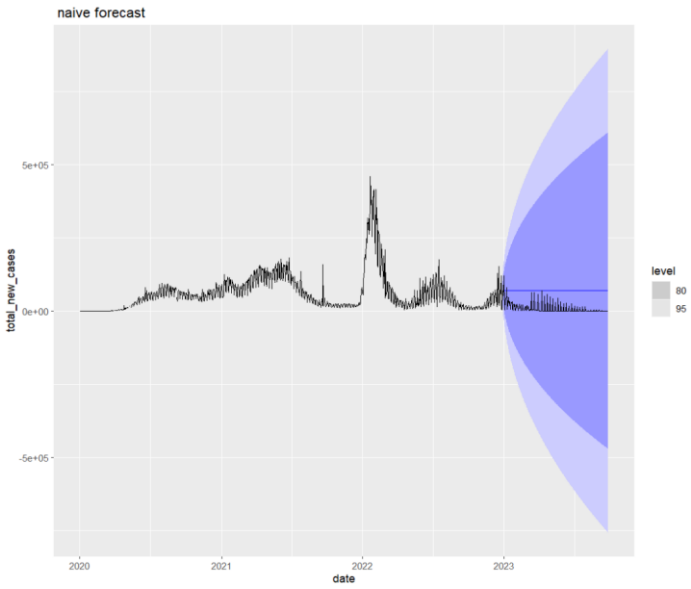

Figure 20. Naive model forecast plot

Figure 20, shows a naive forecast of average values over time, along with the actual average values in 2023. The plot indicates an increasing trend in the averages leading up to 2023, and

the naive forecast anticipates a continuation of this trend with a straight horizontal line. However, the observed average values in 2023 deviate slightly below the naive forecast, suggesting a potential overestimation in the projected rate of increase. The negative ME implies that the forecast model did not estimate the actual values, supporting the observation that the 2023 actual values are marginally lower than predicted. This hints at a tendency of the naive model to exaggerate the expected rate of increase [7].

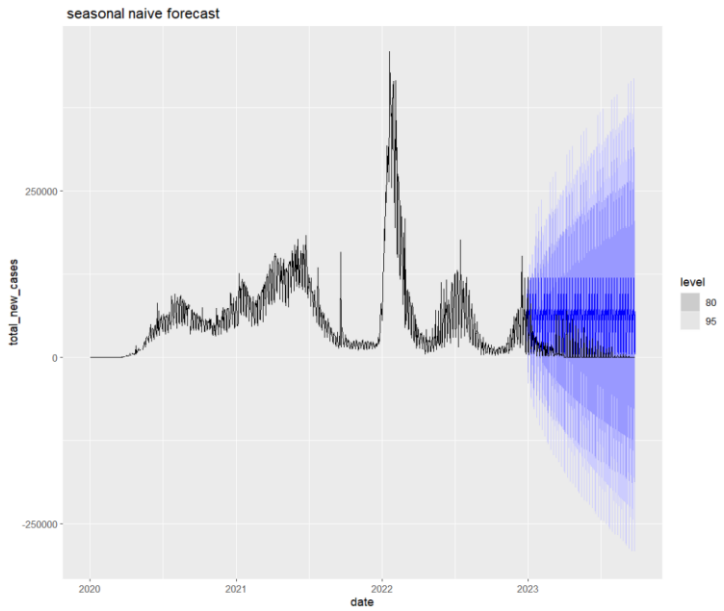

Figure 21. Seasonal naive model forecast plot

Figure 21, shows a seasonal naive forecast of COVID-19 cases in 2023 alongside the actual case numbers. The seasonal naive forecast adeptly captures a recurring pattern and provides a glimpse of the fluctuations in the data. However, a comprehensive evaluation using performance metrics shows notable discrepancies. The negative ME indicates that the estimated values are not the same as the actual values, suggesting a systemic tendency to fall short in predicting actual case numbers. The RMSE and MAE values offer insights into the overall predictive accuracy, with the former emphasizing larger errors. Additionally, the ACF value, a measure of the model's ability to account for temporal dependencies, suggests a relatively weak correlation [7].

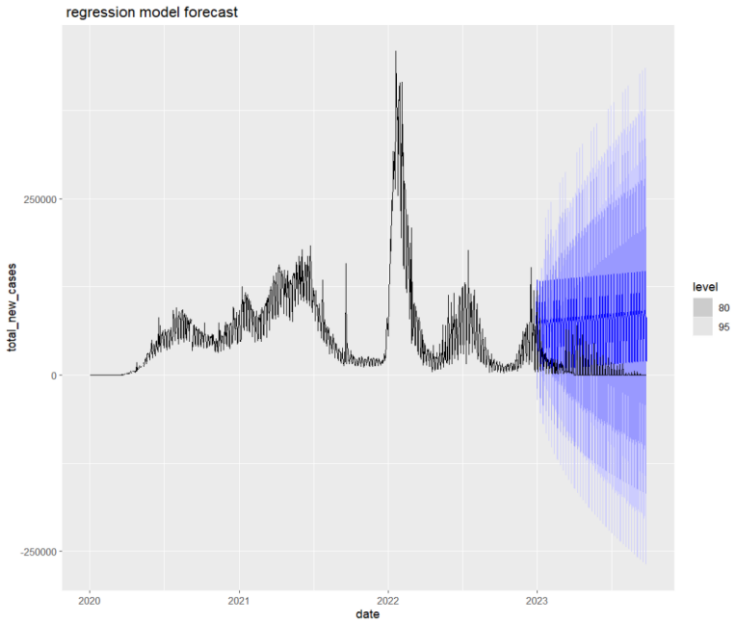

Figure 22. Regression model forecast plot

Figure 22, presents the regression model's forecast for a gradual increase in COVID-19 cases in 2023. A close examination of the actual data unveils a subtle decrease in cases, hinting at a potential overestimation in the upcoming months by the forecast. The model evaluation metrics offer valuable insights, with the Mean Error (ME) indicating a consistent deviation between the forecasted and actual values, the Root Mean Square Error (RMSE) emphasizes larger errors, and the Mean Absolute Error (MAE) offers context to the average magnitude of errors, providing a comprehensive view of the model's performance. Additionally, the Autocorrelation Function (ACF) value of 0.149 indicates a relatively weak correlation in capturing temporal dependencies [7].

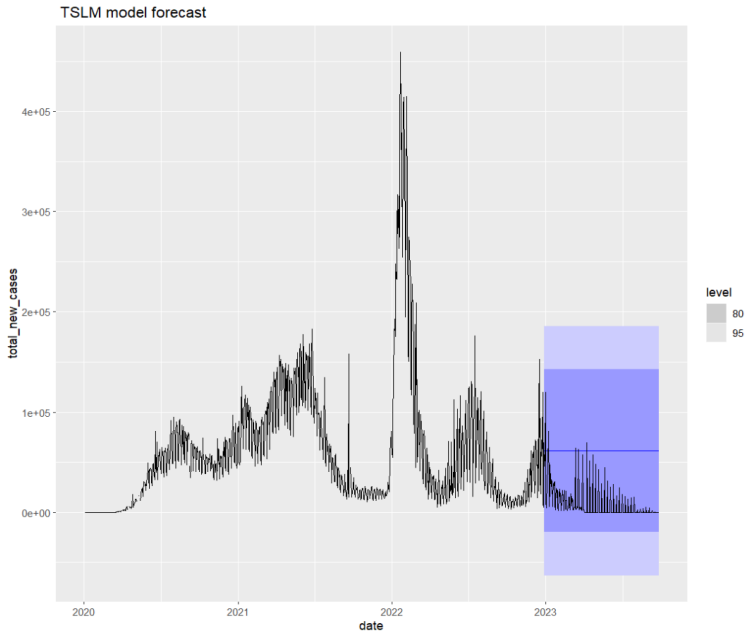

Figure 23. TSLM model forecast plot

Figure 23, displays the forecast plot of the TSLM model alongside the actual values. The plot reveals a notable divergence between the forecasted values and the actual values, as indicated by the negative Mean Error (ME) value. Despite this discrepancy, the TSLM model stands out as the best among the proposed models.

The plot suggests that the TSLM model exhibits lower uncertainty and anticipates less variability in future outcomes compared to the other models. This observation is supported by the fact that the TSLM model has the lowest Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) values when compared to the other models.

The lower uncertainty and improved accuracy of the TSLM model, as reflected in the MAE and RMSE values, contribute to its superiority among the proposed models. These metrics demonstrate that the TSLM model provides more precise predictions and aligns more closely with the actual data [7].

References

[1] TY - JOUR, AU - Cheng, Chieh, AU - Jiang, Wei-Ming, AU - Fan, Byron, AU - Cheng, Yu-Chieh, AU - Hsu, Ya-Ting, AU - Wu, Hsiao-Yu, AU - Chang, Hsiao-Han and AU - Tsou, Hsiao-Hui, " Real-time forecasting of COVID-19 spread according to protective behavior and vaccination: autoregressive integrated moving average models," *BMC Public Health*, 2023.

[2] "Tibbles," rstudio, [Online]. Available: https://cran.r-project.org/web/packages/tibble/vignettes/tibble.html. [Accessed 25 10 2023].

[3] M. Almobarak, S. Aljuhani, N. Alotaibi and S. Altaweel, github, [Online]. Available: https://github.com/monerahalmobarak/Faculty-Data-Analysis. [Accessed 25 10 2023].

[4] E. Wang, "Introduction to tsibble," rstudio, [Online]. Available: https://cran.rstudio.com/web/packages/tsibble/vignettes/intro-tsibble.html#:~:text=Built%20on%20top%20of%20the,makes%20heterogeneous%20data%20structures%20possible.. [Accessed 25 10 2023].

[5] "R: Lag plots," 25 10 2023. [Online]. Available: https://search.r-project.org/CRAN/refmans/feasts/html/gg_lag.html.

[6] "acf function - RDocumentation," [Online]. Available: https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/acf.

[7] R. J. Hyndman and G. Athanasopoulos, Forecasting : Principles and Practice, vol. 2, Heathmont, Vic.: Otexts, 2018.

[8] "Seasonal subseries plots — gg_subseries," [Online]. Available: https://feasts.tidyverts.org/reference/gg_subseries.html.

[9] D. Curran-Everett, "Explorations in statistics: the log transformation," p. 5, 2018.

[10] D. Neufeld, "Visualization Methods for Periodic Time Series Data," p. 10, 2021.

[11] "Pluralsight," [Online]. Available: https://www.pluralsight.com/guides/time-series-forecasting-using-r.

[12] "TimeSeries," [Online]. Available: https://github.com/mshanker1/60095_201810/blob/master/TimeSeries.Rmd.

[13] "rpubs," [Online]. Available: https://rpubs.com/Bassel2014/382416.

[14] R. J. Hyndman, "Accuracy measures for a forecast model," forecast, 2018. [Online]. Available: https://pkg.robjhyndman.com/forecast/reference/accuracy.default.html.

[15] "R: Forecast plot," [Online]. Available: https://search.r-project.org/CRAN/refmans/forecast/html/plot.forecast.html.

Appendices:

Appendix A:

```
# Step is to load necessary libraries
library(tsibble)
library(lubridate)
library(fpp3)
library(dplyr)
library(ggplot2)
library(scales)
library(fable)
library(fabletools)


###################### Start of the Preparing ######################
# Read the data
Data <- read.csv("/Users/alaaalshuaibi/Desktop/TimeSeries/project/owid-covid-data.csv")

Data <- Data %>%
  filter(continent == "South America") %>%
  select(continent, new_cases, date) %>%
```

```
  group_by(continent, date) %>%
  summarise(total_new_cases = sum(new_cases))
Data

# Check if there is missing values for Data
missing_values <- sum(is.na(Data))

if (missing_values > 0) {
  print(paste("The dataset contains", missing_values, "missing value(s)."))
} else {
  print("The dataset does not have any missing values.")
}

# If need to remove rows with missing values
Data <- na.omit(Data)

# update a date column format from char to date as day format to can be used in Time Series tsibble
Data <- Data %>%
  mutate( date = as_date(date))

# Create a tsibble
Data_tsibble <- as_tsibble(Data, key = continent, index = date)

# View the tsibble
Data_tsibble


###################### End of the Preparing ######################
###################### Start of the Preparing for Datawithlocation ######################
# Read the data
```

```
Datawithlocation  <-  read.csv("/Users/alaaalshuaibi/Desktop/TimeSeries/project/owid-covid-data.csv")

Datawithlocation <- Datawithlocation %>%
  filter(continent == "South America") %>%
  select(continent, new_cases, date,location) %>%
  group_by(continent, date,location) %>%
  summarise(total_new_cases = sum(new_cases))
Datawithlocation

# update a date column format from char to date as day format to can be used in Time Series tsibble
Datawithlocation <- Datawithlocation %>%
  mutate( date = as_date(date))

# Check for missing values
missing_values <- sum(is.na(Datawithlocation))

# Print the result
if (missing_values > 0) {
  print(paste("The dataset contains", missing_values, "missing value(s)."))
} else {
  print("The dataset does not have any missing values.")
}

# Remove rows with missing values
Datawithlocation <- na.omit(Datawithlocation)

# Create a tsibble with a location
tsibblewithlocation <- as_tsibble(Datawithlocation, key = c(continent,location), index = date)

# View the tsibble
```

tsibblewithlocation

###################### End of the Preparing for Datawithlocation ######################

###################### Start of the Visualization ######################

```
# autocorrelation
Data_tsibble %>% gg_lag(total_new_cases, geom = "point")

# fill gaps for ACF plot
Data_tsibble <- fill_gaps(Data_tsibble)

# Plot the ACF
Data_tsibble |> ACF(total_new_cases, lag_max = 9) |> autoplot()

# Plot the ACF
Data_tsibble |> ACF(total_new_cases, lag_max = 50) |> autoplot()

#season plot weekly
Data_tsibble |>
  gg_season(total_new_cases ,period = "week")

#time plot using ggplot
Data_tsibble |>
  ggplot(aes(x=date , y=total_new_cases))+
  geom_line()

#time plot using autoplot
Data_tsibble |>
  autoplot(total_new_cases)
```

```
## sub series
Data_tsibble |>
  gg_subseries(total_new_cases ,period = "week")

####this the way of mathematical transformation####

#using log
Data_tsibble |> autoplot(log(total_new_cases))

#using log square root
Data_tsibble |> autoplot(sqrt(total_new_cases))

#using cube root
Data_tsibble |> autoplot(total_new_cases^(1/3))

#using inverse
Data_tsibble |> autoplot(-1 / total_new_cases)

#to fine the best value of lambda
Data_tsibble |> features(total_new_cases, feature = guerrero)

#this is the plot with the best value of lambda
Data_tsibble |> autoplot(box_cox(total_new_cases,  0.587))

#this is the time plot to compare
Data_tsibble |> autoplot(total_new_cases)


###################### End of the Visualization ######################
```

```
#################### Start of the Visualization with Datawithlocation####################
#gg plot with location (points)
tsibblewithlocation |>
  ggplot(aes(x=date , y=total_new_cases  , colour= location))+
  geom_point()

#gg plot with location (lines)
tsibblewithlocation |>
  ggplot(aes(x=date , y=total_new_cases  , colour= location))+
  geom_line()

# fill gaps
tsibblewithlocation<- fill_gaps(tsibblewithlocation)

#gg seasonal
tsibblewithlocation |>
  gg_season(total_new_cases) +
  facet_wrap(vars(location), nrow = 7, scales = "free_y" )

## sub series with location
tsibblewithlocation |>
  gg_subseries(total_new_cases  ,period = "week")
#################### End of the Visualization with Datawithlocation ####################

### Start Specifying Model, Model Estimation,Performance Evaluation##########


# Splitting the data
split_point <- floor(nrow(Data_tsibble) * 0.8) # 80% for training
train_data <- head(Data_tsibble, split_point)
```

```
test_data <- tail(Data_tsibble, -split_point)


# Training the naive model
naive_model <- train_data %>%
  model(naive = NAIVE(total_new_cases))

# Training the snaive model
snaive_model <- train_data %>%
  model(snaive = SNAIVE(total_new_cases))

# Training the regression model
regression_model <- train_data %>%
  model(regression = ARIMA(total_new_cases ~ 1))

# Training the TSLM model
tslm_model <- train_data %>%
  model(tslm = TSLM(total_new_cases))

# Generating forecasts with the naive model
naive_forecast <- naive_model %>%
  forecast(new_data = test_data)

# Generating forecasts with the snaive model
snaive_forecast <- snaive_model %>%
  forecast(new_data = test_data)


# Generating forecasts with the regression model
regression_forecast <- regression_model %>%
  forecast(new_data = test_data)
```

```
# Generating forecasts with the TSLM model
tslm_forecast <- tslm_model %>%
  forecast(new_data = test_data)

# Evaluating the models
naive_accuracy <- accuracy(naive_forecast, test_data)
snaive_accuracy <- accuracy(snaive_forecast, test_data)
regression_accuracy <- accuracy(regression_forecast, test_data)
tslm_accuracy <- accuracy(tslm_forecast, test_data)

# Printing the accuracies
print(naive_accuracy)
print(snaive_accuracy)
print(regression_accuracy)
print(tslm_accuracy)

# forecast plots:
library(ggfortify)
naive_forecast |> autoplot(Data_tsibble) +
  labs(title=" naive forecast")

snaive_forecast |> autoplot(Data_tsibble) +
  labs(title=" seasonal naive forecast")

regression_forecast |> autoplot(Data_tsibble) +
  labs(title=" regression model forecast")

tslm_forecast |> autoplot(Data_tsibble) +
  labs(title=" TSLM model forecast")
```