Analysis and Forecasting of North and South Geomagnetic Pole Intensity

1005780558

Peerapat Phatpanichot

STA457 FINAL PROJECT

Abstract:

This study aims to develop a forecasting methodology for predicting the magnetic intensities of the North and South geomagnetic poles using the Box-Jenkins framework. The Earth's magnetic field, characterized by complex and dynamic behavior, undergoes changes in strength and orientation over time, known as geomagnetic secular variation. Recent observations reveal that the geomagnetic poles are drifting at an accelerating rate, and with further inspection we select an appropriate ARIMA models for North and South geomagnetic pole intensities after transforming the data by taking out the deterministic trend and using differencing twice to achieved stationary. The best fitted models were ARIMA(4,0,5) for North geomagnetic pole and ARIMA(5,0,5) for South geomagnetic pole. Our findings suggests that the past observation and dependences is significant to determine the current and future intensities of both poles especially the North geomagnetic pole. We forecast the future predictions of both poles 10 years ahead and found that both magnetic pole continues its trend where the North geomagnetic intensity continues to increase, and the South geomagnetic intensity continues to decrease in the next 10 years. This prediction can be crucial to navigation systems, satellite performance, and the studies in the field of Earth’s magnetic field.

Table of Contents:

Introduction 4

- Background and motivation

Data Description 5

- Methodology
- Time series decomposition

Model Specification 7

- Model selection and Transformation
    1. ACF and PACF plots
    2. ARIMA model fitting and diagnostics
- Model interpretation

Forecasting and Results 15

- North geomagnetic pole intensity forecasts
- South geomagnetic pole intensity forecasts
- Forecast accuracy and evaluation
- Comparison of North and South geomagnetic pole intensities

Discussion 19

- Summary of findings
- Limitations and potential improvements
- Future work

References 20

- List of sources

Appendices 21

- R code North Geomagnetic Pole
- R code South Geomagnetic Pole

Introduction

Background and Motivation

It is not a well-known fact that the Earth has two North pole the Geographic North pole and Magnetic North pole which always on the move which in the last 150 years the magnetic North pole is moving faster than usual 25 miles a year. This could be a sign that a significant change in coming, and the South and North pole might switch. If the Earth magnetic pole were to switch places this could be catastrophic this could weaken Earth protective magnetic pole during the flip the protective magnetic field is what protect us from radiation and might disturb internal compass to the living things on Earth. This could also affect how our stat elite up in space moves and astronauts up in space might be unsafe to more exposure to radiation. The geomagnetic pole intensities is the measure of the strength of magnetic pole as it provides information on behaviour of the Earth's Interior. The moving of these magnetic pole can be causes by various factors and in the past few years these poles are moving exponentially faster than ever. The main objective of this project is to identity the best model for our North and South geomagnetic pole intensity time series and use the best model to forecast our predictions of both pole intensity movements for the next 10 years and compare the results.

Methodology

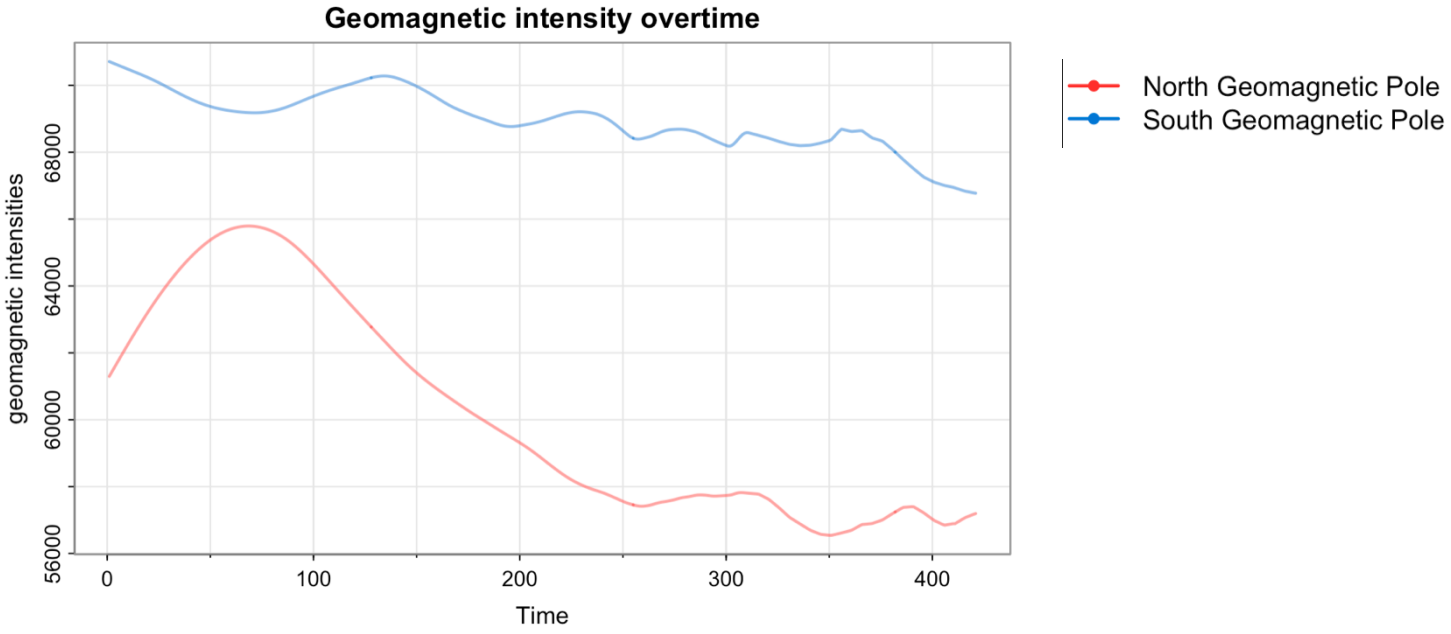
Box-Jenkins framework

We will be using Box-Jenkins framework involves a systematic approach to identifying, fitting, and validating univariate time series models, specifically ARIMA and SARIMA models.

We can clearly see that the North and South geomagnetic pole time series are non-stationary, and both depicts a negative deterministic trend. Therefore, the first transformation we did was removing this deterministic trend. After the first differencing both time series is still non-stationary therefore, we need second order differencing to make our time series stationary. However, the North geomagnetic pole time series does not pass the the KPSS test on level therefore we decided to change our index in the trained data set to start where the North geomagnetic pole intensity peaks and to certain year close to present and fortunately we managed to be achieved stationary in both time series

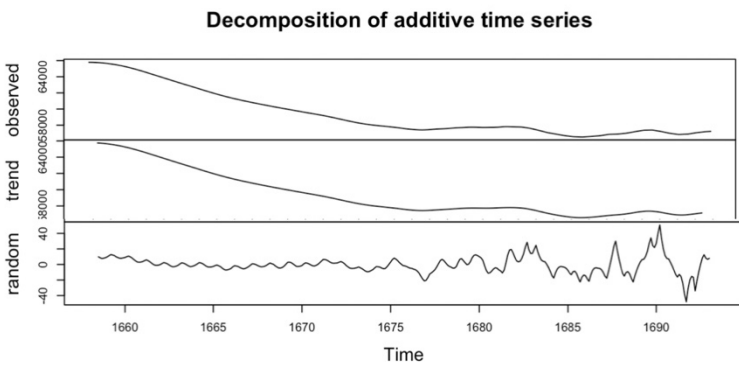
Model fitting: Next we choose our best model respectively to our geomagnetic pole using AIC and BIC this is where we must look at the residuals of ACF and PACF of many suggested models perform a Ljung-Box- test and see which suggested model from AIC and BIC is best fit. After examining the best fitted model ACF and PACF of its respective residuals and use the ensure that the residuals are white noise and that there's no remaining autocorrelation in the data. We moved on to the forecasting method where we use our best model to forecast 10 years ahead into the future where we can validate these forecasts with our test dataset to make sure our best model can accurately forecast. After ensuring that our forecasts of the train dataset is accurate and do not deviate much from the true value we use the whole dataset to forecast the 10 years ahead intensity of both magnetic pole. At the end we interpret the results from our forecasts and conclude our results and suggests any further improvement we could make in our analysis.

Time series decomposition

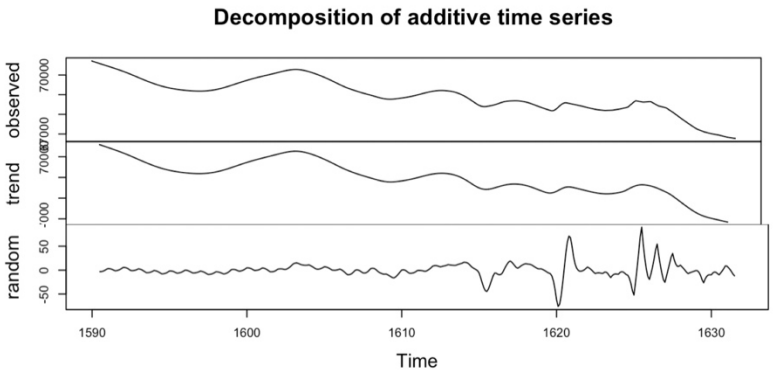


The graph above depicts the timeseries of North geomagnetic intensity and South geomagnetic intensity over the year from 1590 to 2023. Both time series shows negative trend overtime, and we can see that the North geomagnetic Pole intensity depicts lower tesla than the South geomagnetic pole, and both pole shows no seasonality over the years. After further inspection we can see that both of our data sets can be considered a random walk and transformation are needed before fitting the model. However, I decided to cut some part of the North pole dataset because with the full North geomagnetic pole data set the time series is still non-stationary after detrending and differencing twice we will discuss more of this in later part.

North Geomagnetic Pole



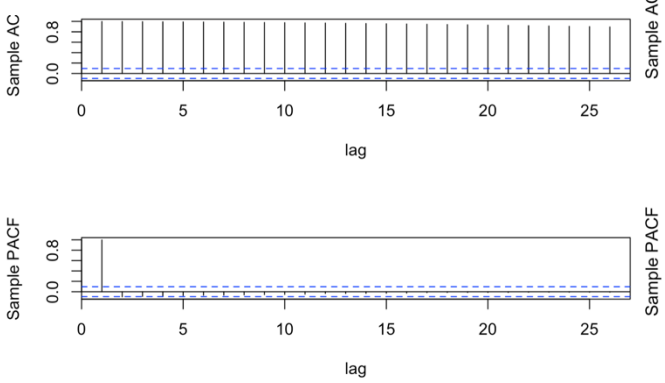
South Geomagnetic Pole



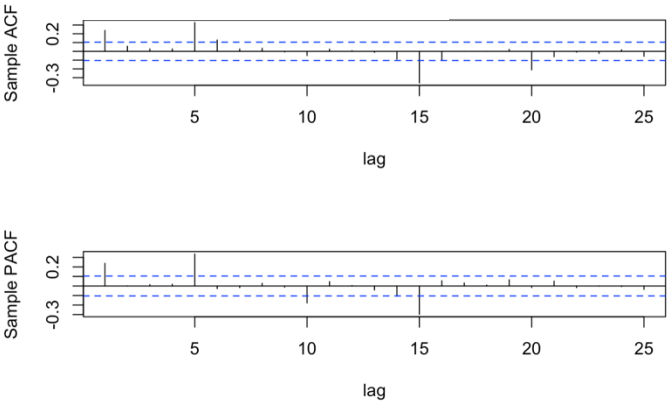
Both of the Earth magnetic pole time series seems how to have a negative trend therefore after further inspection it is a deterministic trend and I decided to remove this trend from both time series, however it was not enough to make it stationary, look at the random part of the decomposition the random components from both graphs shows the residuals are close to 0 for a certain period of time then started to fluctuate suggesting there might be an additional pattern that the data might not be fully captured by the model so after differencing once both of our time series are still non-stationary and therefore, I difference our data once again to make sure it pass all the tests needed to Box-Jenkins methodology.

Model selection and Transformation for North Geomagnetic Pole

Prior to Transformation



Post Transformation



Making sure our time series is stationary is an essential first time in the Box-Jenkins analysis method which in our case we can see that on the right-hand side the ACF and PACF graph of the time series prior to applying transformation can be seen as non-stationary visually, on the other hand after we apply the transformation our ACF and PACF looks more stationary.

We can also confirm stationarity of our time series using ADF test and KPSS test where the null hypothesis for ADF test is non-stationary and null hypothesis for KPSS is stationary

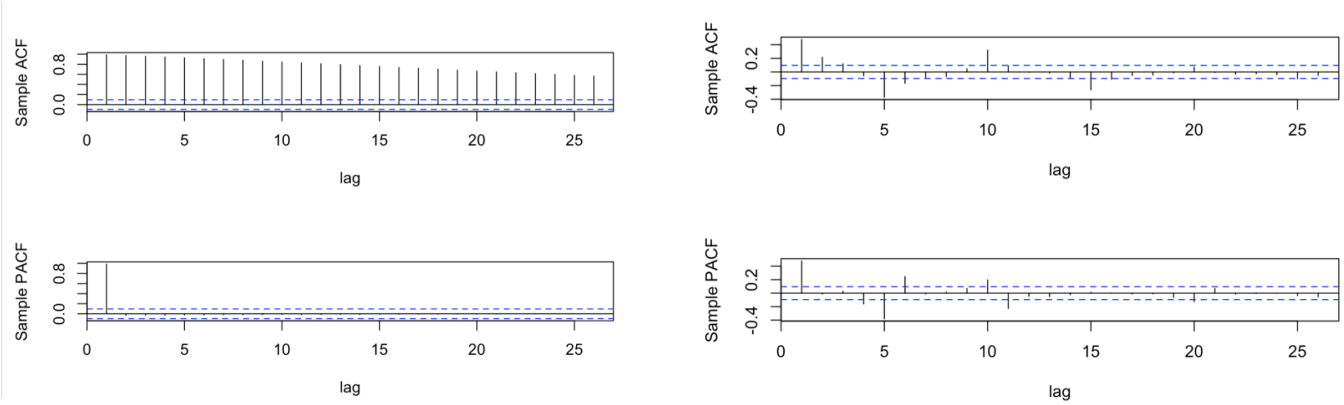
Test	Type	Dickey- Fuller Statistic	Lag Order	p-value	Interpretation
Augmented Dickey-Fuller	No Constant	-5.2808	13	0.01	Stationary at 5 %
Augmented Dickey-Fuller	Constant	-5.2852	13	0.01	Stationary
Augmented Dickey-Fuller	Constant and Trend	-5.2734	13	0.01	Stationary
KPSS	Level	0.12437	5	0.1	Fail to reject H0
KPSS	Trend	0.070833	5	0.1	Fail to reject Ho

These are AIC and BIC outputs:

AIC	Values	BIC	Values
ARMA(4,5)	2075.978	**ARMA(2,1)**	2108.89
ARMA(5,5)	2077.103	**ARMA(1,2)**	2109.60
ARMA(3,5)	2080.013	**ARMA(4,5)**	2110.80
ARMA(2,5)	2085.512	**ARMA(3,5)**	2110.96

Even though AIC and BIC does not output the same best models I decided to run each model it outputs and look at the residuals of its ACF and PACF and see if the accuracy of our forecast increases and I conclude that our best option is ARMA(4,5)

Model selection and Transformation for South Geomagnetic Pole



In our analysis, we observed that the initial dataset was non-stationary. Upon examining the trend plot, we determined that it was necessary to apply two transformations to achieve stationarity: firstly, subtracting the deterministic trend, and secondly, differencing the detrended data twice. After performing these transformations, the augmented Dickey-Fuller (ADF) tests for no constant (n), constant (c), and constant with linear trend (ct) indicated that the transformed dataset achieved stationarity. However, the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test for the presence of a deterministic trend still showed evidence of non-stationarity, even after detrending the dataset. This inconsistency requires further transformation therefore I transformed using differencing once again and from the transformed data ACF and PACF we achieved stationary

Test	Type	Dickey- Fuller Statistic	Lag Order	p-value	Interpretation
Augmented Dickey-Fuller	No Constant	-5.8896	13	0.01	Reject null hypothesis and conclude stationary
Augmented Dickey-Fuller	Constant	-5.8833	13	0.01	Reject null hypothesis and conclude stationary
Augmented Dickey-Fuller	Constant and Trend	-5.8784	13	0.01	Reject null hypothesis and conclude stationary
KPSS	Level	0.024341	5	0.1	Accept null hypothesis and conclude stationary
KPSS	Trend	0.020859	5	0.1	Accept null hypothesis and conclude stationary

These are the AIC and BIC outputs:

AIC	Values	BIC	Values
ARMA(5,5)	2480.447	**ARMA(5,5)**	2520.826
ARMA(3,5)	2498.006	**ARMA(3,5)**	2530.309
ARMA(4,5)	2498.998	**ARMA(4,5)**	2535.328
ARMA(5,4)	2510.423	**ARMA(5,4)**	2546.764

Even though AIC and BIC does output the same best models I decided to run each model it outputs and look at the residuals of its ACF and PACF and see if the accuracy of our forecast increases and I conclude that our best option is ARMA(5,5)

ARIMA model fitting and diagnostics

For North Geomagnetic Pole

After performing transformation by getting rid of the deterministic trend and applying differencing twice, and finding out best model from running AIC and BIC and comparing the best models we conclude that our best model to be ARIMA(4,0,5)

Mathematically, this can be represented as:

ARIMA (4,0,5)

To represent this model with a non-zero mean mathematically, let x_t be the North geomagnetic pole intensities dataset, the chosen ARIMA model can be expressed as:

$$x_t = u + \phi_1 x_{t-1} + \phi_2 x_{t-2} + \phi_3 x_{t-3} + \phi_4 x_{t-4} + w_t - \theta_1 w_{t-1} - \theta_2 w_{t-2} - \theta_3 w_{t-3} - \theta_4 w_{t-4} - \theta_5 w_{t-5}$$

Where,

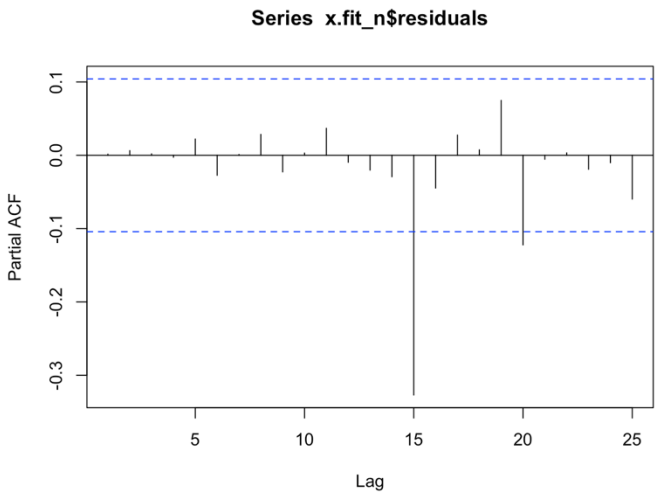
- x_t is the value of our time series at time t
- $\phi_1, \phi_2, \phi_3, \phi_4$ are the AR coefficients
- $\phi_{S_1}, \phi_{S_2}, \phi_{S_3}, \phi_{S_4}, \phi_{S_5}$ are the seasonal AR coefficients
- $\theta_1, \theta_2, \theta_3, \theta_4, \theta_5$ are the MA coefficients
- μ is the non-zero mean term
- w_t is the error term, white noise at time t

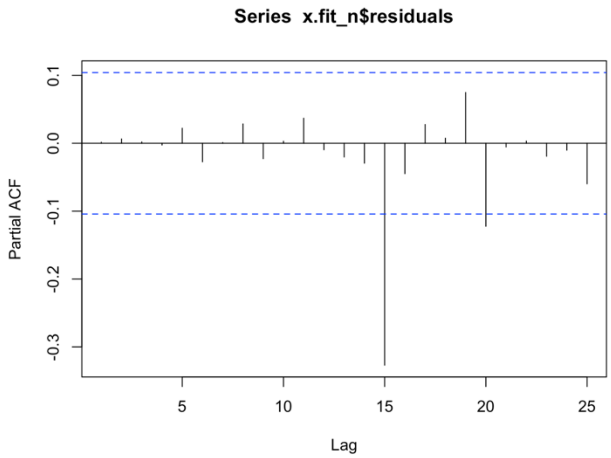
	Ar1	Ar2	Ar3	Ar4
Coefficients	0.4928	-0.1480	0.0967	0.1441
Standard Errors	0.1729	0.1931	0.1358	0.1772

	Ma1	Ma2	Ma3	Ma4	Ma5
Coefficients	-0.2569	0.0884	-0.0825	-0.2356	0.3345
Standard Errors	0.1691	0.1915	0.1219	0.1503	0.0694

Mean	
Coefficient	0.0059
Standard error	0.4667

We can also confirm that our model that we chose capture most of the underly structure of time series. After looking at the ACF and PACF of the model residuals we can see, most of the residuals don't behave like white noise, mostly all the ACF and PACF values fall within the confidence intervals. This would indicate that the model has adequately captured the underlying structure of the time series.





ARIMA Model Parameters for North Geomagnetic Pole Dataset

In this section, we present the maximum likelihood estimates (MLEs) of the AR and MA parameters, as well as the intercept parameter for the ARIMA model fitted to the North geomagnetic pole dataset.

AR Parameters (MLEs):

Ar1	Ar2	Ar3	Ar4
0.49277682	-0.14801874	0.09671711	0.14405374

MA Parameters (MLEs):

Ma1	Ma2	Ma3	Ma4	Ma5
-0.25691328	0.08841760	-0.08250507	-0.23555985	0.33448150

Intercept Parameter (MLE):

Intercept (μ): 0.005887878

Box-Ljung Test for Residual Autocorrelation:

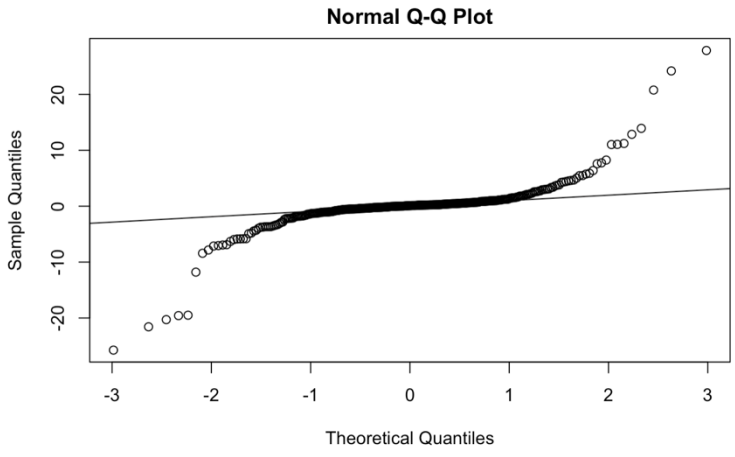
Test Statistic (X-squared): 1.5664

Degrees of Freedom: 13

P-value: 0.9999

With our intercept = 0.005887 represents the mean level of the series after accounting for the AR and MA components, considering that the data has been detrended and differenced twice. This value implies a small positive constant term in the model, which may be related to any remaining systematic patterns in the data not captured by the AR and MA components. The PACF and ACF capture most of the data but there are some that lie outside confidence interval.

With the Box-Ljung Test the test statistic (X-squared) is 1.5664 with 13 degrees of freedom, and the corresponding p-value is 0.9999. Since the p-value is greater than the significance level of 0.05, there is insufficient evidence to reject the null hypothesis, which assumes no autocorrelation in the residuals. This result indicates that the ARIMA(4, 0, 5) model has captured most of the dependence in the data, and the remaining residuals can be considered white noise. As it suggests that the model is adequately capturing the underlying dynamics of the series.



To check the normality of the distribution of our residuals we perform a Shapiro-Wilk test and the results are the our p-value is extremely small and we reject the null hypothesis that the distribution of our residuals are normally distributed. This doesn't always mean that our model is inadequate when it comes to forecast, we may have to explore some other transformation or model specification to accomplish better normality in residuals.

For South Geomagnetic Pole

After we ensure that South geomagnetic pole dataset is stationary, we sought to identify the best fitting ARIMA model. Based on the choosing of the AIC and BIC we have a few candidates for the best model but after further inspection, we selected the ARIMA (5, 2, 5) model with the following parameters:

Mathematically, this can be represented as: ARMA(5,2,5)
To represent this model with a non-zero mean mathematically, let x_t be the South geomagnetic pole intensities dataset, the chosen ARIMA model can be expressed as:

$$(1 - \phi_1 B - \phi_2 B^2 - \phi_3 B^3 - \phi_4 B^4 - \phi_5 B^5)(1 - B)^2(x_t - T_t - \mu) = (1 + \theta_1 B + \theta_2 B^2 + \theta_3 B^3 + \theta_4 B^4 + \theta_5 B^5) w_t$$

Where,

- X_t is the original time series
- T_t is the deterministic trend component at time t
- $T_t - T_t$ represents the original time series with the deterministic trend removed
- $(1 - B)^2$ represents the differencing of order 2
- B is the back shift lag operator
- $\phi_1, \phi_2, \phi_3, \phi_4, \phi_5$ are the AR coefficients
- $\theta_1, \theta_2, \theta_3, \theta_4, \theta_5$ are the MA coefficients
- w_t is the error term, white noise at time t
- μ is the non-zero mean term

AR coefficients:

	Ar1	Ar2	Ar3	Ar4	Ar5
Coefficients	-0.1819	0.0337	0.0847	0.0204	-0.6777
Standard Error	0.0719	0.0608	0.0672	0.0803	0.0569

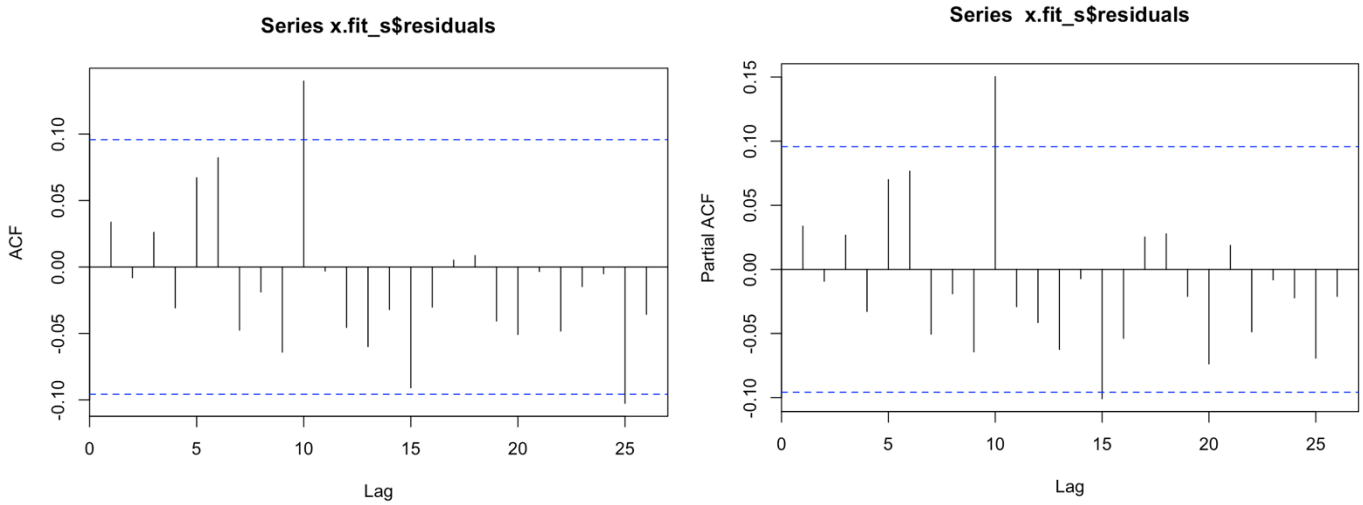
Ma Coefficients:

	Ma1	Ma2	Ma3	Ma4	Ma5
Coefficients	0.6819	0.3315	0.1206	0.1558	0.3929
Standard Error	0.0875	0.0997	0.0928	0.1251	0.1112

Mean:

	Mean
Coefficients	0.0416
Standard error	0.3390

We can also confirm that our model that we chose capture the underly structure of time series. After looking at the ACF and PACF of the model residuals we can see, the residuals don't behave like white noise, all most all the ACF and PACF values fall within the confidence intervals. This would indicate that the model has adequately captured the underlying structure of the time series



ARIMA Model Parameters for South Geomagnetic Pole Dataset

In this section, we present the maximum likelihood estimates (MLEs) of the AR and MA parameters, as well as the intercept parameter for the ARIMA model fitted to the South geomagnetic pole dataset. MLEs allow us to understand the relationship between the current value of the series and its past values, as well as the impact of past errors. This information can be useful for forecasting future values of the series and evaluating the model's performance.

AR Parameters MLEs	
Ar1	-0.18187348
Ar2	0.03367978
Ar3	0.08467727
Ar4	0.02043755
Ar5	-0.67765734

MA Parameters MLEs	
Ma1	0.6819286
Ma2	0.3315252
Ma3	0.1205640
Ma4	0.1558285
Ma5	0.3928576

Intercept Parameter (MLE):

Intercept (μ): 0.04163358

Box-Ljung Test for Residual Autocorrelation:

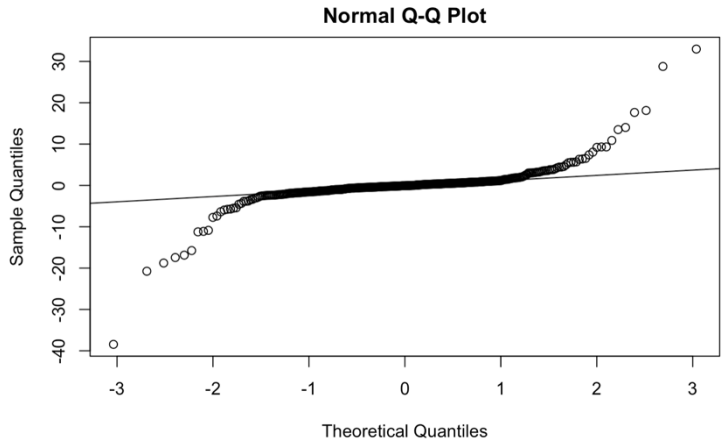
Test Statistic (X-squared): 19.795

Degrees of Freedom: 13

P-value: 0.1005

With our intercept = 0.04163 represents the mean level of the series after accounting for the AR and MA components, considering that the data has been detrended and differenced twice. This value implies a small positive constant term in the model, which may be related to any remaining systematic patterns in the data not captured by the AR and MA components. The PACF and ACF capture most of the data but there are some that lie outside confidence interval.

With the Box-Ljung Test the test statistic (X-squared) is 19.795 with 13 degrees of freedom, and the corresponding p-value is 0.1005. Since the p-value is greater than the conventional significance, there is insufficient evidence to reject the null hypothesis, which assumes no autocorrelation in the residuals. This result indicates that the ARIMA(5, 2, 5) model has captured most of the temporal dependence in the data, and the remaining residuals can be considered white noise. As it suggests that the model is adequately capturing the underlying dynamics of the series.



To check the normality of the distribution of our residuals we perform a Shapiro-Wilk test and the results are the our p-value is extremely small and we reject the null hypothesis that the distribution of our residuals are normally distributed. This doesn't always mean that our model is inadequate when it comes to forecast, we may have to explore some other transformation or model specification to accomplish better normality in residuals.

Forecasting and results

North Geomagnetic Pole

Comparing Forecasts from our train dataset (1658 to 2023) to the actual values from our test dataset (2014-2023) then using our best model to forecasts the intensities from 2023 onwards

Year	Forecasts	Lower bound	Upper bound
2014	57223.13	57224.53	57241.73
2015	57270.57	57261.73	57279.41
2016	57305.43	57296.58	57314.28
2017	57337.41	57328.56	57346.26
2018	57369.72	57360.85	57378.59
2019	57402.35	57393.02	57411.68
2020	57434.44	57424.97	57443.91
2021	57465.83	57456.36	57475.31
2022	57497.05	57487.58	57506.52
2023	57528.27	57518.78	57537.77

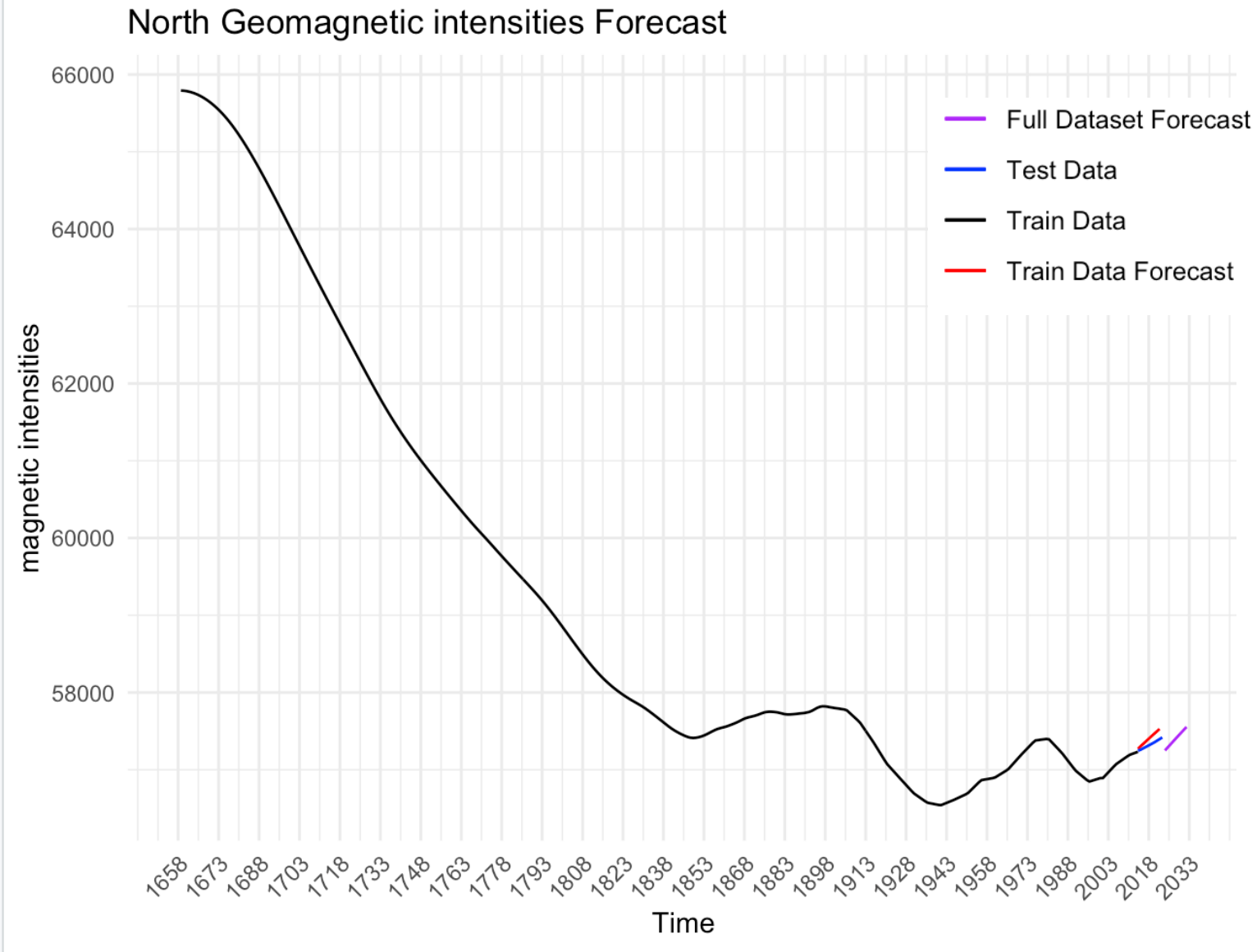
Year	Test data
2014	57246.3
2015	57260.5
2016	57278.8
2017	57297.2
2018	57315.7
2019	57334.2
2020	57353.5
2021	57375.1
2022	57396.7
2023	57418.5

We can clearly see that from the values of our forecasts from train dataset is not too far off from our actual test data, see figure below. Therefore, we can conclude that our model is valid and accurate enough to work with the whole dataset and forecast the next 10 years predictions from 2023 to 2033

	ME	RMSE	MAE	MPE	MAPE
Test set	-56.77	68.60	59.404	-0.098	0.1035

The ME mean error of -56.77 indicates that our forecasts are on average 56.77 units lower than actual values. The RMSE of 68.60 the typical error is around 68 units. The MAE or the mean absolute error of 59.404 absolute values of forecast deviates from the true value by around 59 units. The MPE of -0.098 indicates that on average the forecasts are lower than actual of value by 9.9 %. and lastly the MAPE value of 0.1035 indicates that the absolute percentage error of the forecast is 10.35%.

Comparing Forecasts from our train dataset (2014 to 2023) to the actual values from our test dataset (2014-2023) to validate our model and forecasts then using our best model to forecasts



From our plot we can see that the red line using the train data from (1658-2013) to forecasts the red line of (2014- 2023) is very similar to our actual dataset in the test sets which we can validate and conclude that the model that we build is somewhat efficient for forecasting. Therefore, we forecast 10 years ahead into the future and can see that the magnetic intensity of our North geomagnetic pole continues to rise from 2023 to 2033 which can be represented by the purple line.

Forecasting and Results

South Geomagnetic Pole

We split out data into training from year 1590 to 2010 and testing data from year 2010 to 2023 where our we use our training data to forecasts the next 13 years ahead from 2010 to 2023 and validate it with our known value from testing data (actual 2010 to 2023) and here are the results. Keep in mind these are the results after performing the forecasts manually using out best model ARIMA and transforming our data back because we transformed it in the beginning.

Year	Forecasts	Lower bounds	Upper bounds
2011	66775.8	66766.95	66784.62
2012	66769.7	66759.83	66779.58
2013	66763.5	66753.27	66773.62
2014	66757.7	66747.44	66768.01
2015	66751.5	66741.10	66761.96
2016	66743.2	66732.44	66753.88
2017	66733.1	66722.08	66744.02
2018	66726.1	66715.06	66737.11
2019	66718.6	66707.55	66729.71
2020	66709.6	66698.46	66720.76
2021	66700.9	66689.59	66712.16
2022	66694.4	66683.03	66705.71
2023	66688.7	66677.38	66700.08

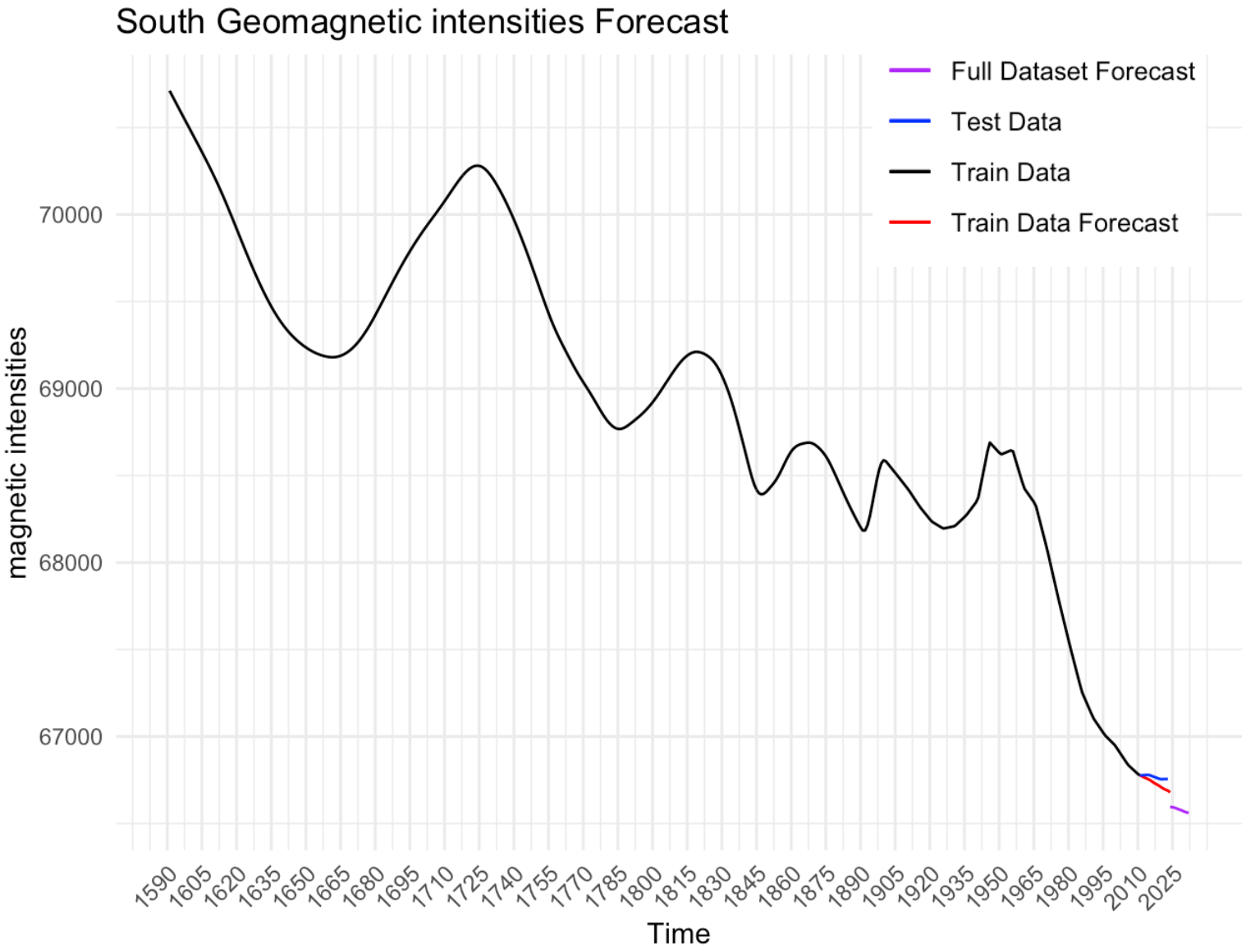
Year	Test Data
2011	66776.5
2012	66777.3
2013	66778.1
2014	66778.9
2015	66778.4
2016	66773.3
2017	66768.2
2018	66763.2
2019	66758.1
2020	66754.3
2021	66754.8
2022	66755.3
2023	66755.8

We can clearly see that from the values of our forecasts from train dataset is not too far off from our actual test data, see figure below. Therefore, we can conclude that our model is valid and accurate enough to work with the whole dataset and forecast the next 10 years

	ME	RMSE	MAE	MPE	MAPE
Test set	33.80	38.86	33.80	0.051	0.051

We can clear see from accuracy tests that our forecasts underestimate the true values by 33.80 units which is consider very small, RMSE of 38.86 suggests that the typical forecast error is around 38.86 units. the MAE of 33.80 indicates that, on average, the model's forecasts deviate by 33.80 units from the true values. MPE indicates that the model underestimates the true values by an average of 5.1%. And MAPE of 0.051 indicates that, on average, the model's forecasts deviate by 5.1% from the true values in relative terms.

Comparing Forecasts from our train dataset (2010 to 2023) to the actual values from our test dataset (2010-2023) then using our best model to forecasts the intensities from 2023 onwards



After using our train data set (1590- 2010) which is represented by the black line, t to build a model and forecasts 13 years ahead which can be represented by the red line we can compare and validate our forecasts to our actual data from 2010 to 2023 which is represents by the blue line our test data. We can conclude that our model is sufficient to make predictions into the future, and therefore we use our full dataset of South Geomagnetic Pole to forecasts predictions from 2023-2033 which are represented by the purple line, and we can see that the South geomagnetic intensity continues to drop from 2023 and onwards.

Discussion

Summary of findings

Based on the ARIMA model forecasts, both the North geomagnetic pole intensity and South geomagnetic pole intensity is expected to continue their trend from the present where the North geomagnetic pole intensities continue to increase, and South geomagnetic pole intensities continue to decrease. As shown in the section above the South geomagnetic pole intensity is predicted to gradually decrease over the next 10 years, starting at 66,595.60 in 2023 and ending at 66,553.08 in 2033. Similarly, the North geomagnetic pole intensity is projected to gradually increase over the next 10 years, starting at 57,250.34 in 2023 and ending at 57,628.98 in 2033. It is a well-known fact in that with the intensity of these pole going further apart there might be a chance that the potential change in poles location and this may influence satellite and navigation system. Lastly, the North and South geomagnetic pole intensities are growing apart, in our case, North geomagnetic pole is becoming stronger because the intensity increases, and the South geomagnetic intensity is getting weaker, and it might as well possibly impact the Earth's weather and etc.

Limitations and potential improvements

The main problem I encounter when writing this report and forecasting the Earth geomagnetic pole is making my time series stationary in particular case in the North geomagnetic pole. Because we can clearly see that our time series for the North is a random walk which in time series sense where in our case seems to be random walk without drift term. In this case, it was very complicated to make our random walk stationary taking in account if we over differencing our data, we might lose some significant outputs. My ideal approach to random walk is to use the previous values to forecast our predictions because it would be easier using our previous outputs, than following a Box-Jenkins methodology. After differencing it first time my time series still did not pass the ADF and KPSS test which I need to further difference the time series. Now with differencing twice I still fail one of the KPSS test which at this point there's a potential of over differencing. Therefore, for the North geomagnetic pole I decided to split my index of my training data differencing by starting at the year 1658 where the intensity peaks and continue my analysis.

Future work

With limited variables in our analysis, we might not be able to get the most accurate prediction of the Earth's magnetic intensity over the next 10 years. There might be factor like weather or sea level that could affect our forecasts. With Earth's geomagnetic intensity, the topic itself is very sensitive and when it comes to forecasts and prediction, we want to make sure to be as close as possible to the true value. From validating my training data forecast with the true value in the test data we can see that the forecasts prediction and true value are slightly differencing therefore with more information and variables I think that we can further improve the accuracy of our forecast predictions.

References

Stoffer, D. S. (n.d.). Time series analysis and its applications: With R examples. University of Pittsburgh Department of Statistics & Data Science. https://www.stat.pitt.edu/stoffer/tsda/

Poison, N. (n.d.). R issues: A Blog about problems I encountered with R programming. https://nickpoison.github.io/rissues

Nau, R. (2007). Autoregressive models. Duke University: The Fuqua School of Business. https://people.duke.edu/~rnau/Decision411_2007/Class10notes.htm

Rao, K. S. (2019, March 3). Financial time series and your random walk in R: ARIMA model. Medium. https://medium.com/analytics-vidhya/financial-time-series-and-your-random-walk-in-r-arima-model-a6461da78ecc

Zivot, E. (n.d.). Lecture 2: Unit root tests. University of Washington Department of Economics. https://faculty.washington.edu/ezivot/econ584/notes/unitrootLecture2.pdf

R code North Geomagnetic Pole

```{r,echo=FALSE,message=FALSE}
setwd("~/Desktop")
library(readxl)
library(dplyr)
library(ggplot2)
library(cowplot)
library(tseries)
library(forecast)
library(TSA)
library(knitr)
library(kableExtra)
library(fUnitRoots)
library(timeSeries)
library(astsa)
library(zoo)
library(MASS)
library(fracdiff)

my_data <- read_excel("Geomagnetic_Intensity_Data.xlsx")
```

```{r,echo=FALSE,message=FALSE,out.width="49%",out.height="49%",fig.show='hold',fig.align='center'}
# train_data: training set (95% of the data)
# test_data: test set (10% of the data)


train_data <- head(my_data, round(nrow(my_data) * 0.97))
h <- nrow(my_data) - nrow(train_data)
test_data <- tail(my_data, h)

train_data_north <- my_data[69:424, ]
test_data_north <- my_data[425:434, ]
whole_data_north <- my_data[69:434,]

```

```{r}
north <- train_data$North_Geomagnetic_Pole
cummeannorth <- cumsum(north)/seq_along(north)
par(mfrow = c(2,1), mar = c(4,4,4,4))
```

```
plot(north, type = 'l', xlab = "Time", ylab = "Time Series x")
plot(cummeannorth, type = "l", xlab = "Time", ylab = "Mean Level")
par(mfrow = c(2,1), mar = c(4,4,4,4))
acf(north, xlab = "lag", ylab = "Sample ACF", main = "")
acf(north, type = "partial", xlab = "lag", ylab = "Sample PACF", main= "")

south <-train_data$South_Geomagnetic_Pole
cummeansouth <- cumsum(south)/seq_along(south)
par(mfrow = c(2,1), mar = c(4,4,4,4))
plot(south, type = 'l', xlab = "Time", ylab = "Time Series x")
plot(cummeansouth, type = "l", xlab = "Time", ylab = "Mean Level")
par(mfrow = c(2,1), mar = c(4,4,4,4))
acf(south, xlab = "lag", ylab = "Sample ACF", main = "")
acf(south, type = "partial", xlab = "lag", ylab = "Sample PACF", main= "")

par(mfrow=2:1)
tsplot(cbind(north,south), spaghetti = TRUE, col = astsa.col(c(2,4), 0.5), lwd = 2, type = "l",
pch=20, ylab= "geomagnetic intensities", main = "Geomagnetic intensity overtime")
legend("topright", col=c(2,4), lty = 1, lwd = 2, pch =20, bg="white", legend = c("North
Geomagnetic Pole", "South Geomagnetic Pole"))


```

`

```{r}
#non stationary random walk use previous point to forecast
or_n <- train_data$North_Geomagnetic_Pole
summary(mod.ar <- lm(or_n[2:421] ~ or_n[1:420]))
#xt = x_{t-1} + w_t
plot.ts(train_data_north$North_Geomagnetic_Pole)
###
```

```{r}

ts_data_n <- ts(train_data_north$North_Geomagnetic_Pole, start = c(1658, 1), frequency = 10)

ts_decomposed_n <- stl(ts_data_n, s.window = "periodic")

plot(ts_decomposed_n)

ts_adj_n <- seasadj(ts_decomposed_n)
```

```
plot(ts_adj_n, main = "Geomagnetic pole data without seasonal component", ylab = "Value")

plot(decompose(ts_data_n))
```

```{r,echo=FALSE,message=FALSE,out.width="49%",out.height="49%",fig.show='hold',fig.align='center'}
n_n <- length(train_data_north$North_Geomagnetic_Pole)
time_index_n <- 1:n_n
regression_n <- lm(train_data_north$North_Geomagnetic_Pole ~ time_index_n)
detrend_north <- train_data_north$North_Geomagnetic_Pole - predict(regression_n)


time_index_n2 <- 1:length(detrend_north)
regression_n2 <- lm(detrend_north ~ time_index_n2)
detrend_north2 <- detrend_north - predict(regression_n2)

xn <- diff(diff(detrend_north))
cummeanxn <- cumsum(xn)/seq_along(xn)
par(mfrow = c(2,1), mar = c(4,4,4,4))
plot.ts(xn,type = 'l', xlab = "Time", ylab = "Time Series x")
plot(cummeanxn, type = "l", xlab = "Time", ylab = "Mean Level")
par(mfrow = c(2,1), mar = c(4,4,4,4))
acf(xn, xlab = "lag", ylab = "Sample ACF", main = "")
acf(xn, type = "partial", xlab = "lag", ylab = "Sample PACF", main= "")
```

```{r,echo=FALSE,message=FALSE, warning = FALSE}
# Model Specification NORTH
# Perform ADF test with no constant (nc)
adf_nc_n <- adfTest(xn, type = "nc",lags = 13)

# Perform ADF test with constant only (c)
adf_c_n <- adfTest(xn, type = "c", lags = 13)

# Perform ADF test with constant and trend (ct)
adf_ct_n <- adfTest(xn, type = "ct", lags = 13)

# Print the results
adf_nc_n
adf_c_n
adf_ct_n

x.kpss_level_n <- kpss.test(xn, null = "Level")
```

```
x.kpss_trend_n <- kpss.test(xn, null = "Trend")
x.kpss_level_n
x.kpss_trend_n



xn.eacf <- eacf(xn, ar.max = 5, ma.max = 5)
```

```{r,echo=FALSE,message=FALSE}

xn.aic = matrix(0,5,5,)
xn.bic = matrix(0,5,5)

for (i in 0:4) for (j in 0:4){
  xn.fit <- Arima(xn,order=c(i,0,j), method = "ML",include.mean = TRUE)
  xn.aic[i+1, j+1] <- xn.fit$aic
  xn.bic[i+1,j+1] <- BIC(xn.fit)
}
```

```{r,echo=FALSE,message=FALSE}
xn.aic.sort <- sort(xn.aic)
indices1 <- arrayInd(order(xn.aic), dim(xn.aic))
aic1.df <- data.frame(row = indices1[,1], col = indices1[,2] , value = xn.aic.sort)
aic1.v <- kable(aic1.df, row.names = FALSE, col.names = c("AR", "MA", "Value")) %>%
   kable_styling()
aic1.v
```

```{r,echo=FALSE,message=FALSE}
xn.bic.sort <- sort(xn.bic)
indices11 <- arrayInd(order(xn.bic), dim(xn.bic))
bic1.df <- data.frame(row = indices11[,1], col = indices11[,2] , value = xn.bic.sort)
bic1.v <- kable(bic1.df, row.names = FALSE, col.names = c("AR", "MA", "Value")) %>%
   kable_styling()
bic1.v
```

```{r,echo=FALSE,message=FALSE,out.width="49%",out.height="49%",fig.show='hold',fig.ali
gn='center'}
```

```
x.fit_n = Arima(xn, order = c(4,0,5) , method = "ML", include.mean = TRUE)
##############################################
##tsdia()with my own dof
residuals_arima <- residuals(x.fit_n)
acf_resid_n <- acf(residuals_arima, plot = FALSE)


plot(acf_resid_n, main = "ACF of Residuals with Custom Confidence Interval")


alpha <- 0.05  # 95% confidence interval (change alpha for other confidence levels)
dof <- 389-(10)  # Your desired degrees of freedom
critical_value <- qnorm(1 - alpha / 2) / sqrt(dof)


abline(h = c(-critical_value, critical_value), col = "blue", lwd = 2, lty = "dashed")


##############################################
#tsdiag(x.fit_n)
qqnorm(residuals(x.fit_n))
qqline(residuals(x.fit_n))

shapiro.test(residuals(x.fit_n))
hist(x.fit_n$residuals)
acf(x.fit_n$residuals)
pacf(x.fit_n$residuals)

```

```{r}
ar_param_n <- x.fit_n$coef[grepl("^ar", names(x.fit_n$coef))]
ma_param_n <- x.fit_n$coef[grepl("^ma", names(x.fit_n$coef))]
intercept_param_n <- x.fit_n$coef["intercept"]

ar_param_n
ma_param_n
intercept_param_n


residuals_north <- residuals(x.fit_n)
ljung_box_test_n <- Box.test(residuals_north, lag = 13, type = "Ljung-Box")

print(ljung_box_test_n)
```

```r
#PREDICTION NORTH
x <- xn
forecast_arima_4_0_5 <- function(x, ar_coefs, ma_coefs, n_ahead) {
  n <- length(x)
  last_non_na <- max(which(!is.na(residuals_north)))
  errors <- residuals_north[(last_non_na - 4):last_non_na]

  forecast <- numeric(n_ahead)
  for (i in 1:n_ahead) {
    new_data <- c(x, forecast[1:(i - 1)])
    data <- new_data[(n - 3):(n + i - 1)]

    forecast[i] <- sum(ar_coefs * data[1:4]) + sum(ma_coefs * errors)
    errors <- c(errors[-1], -forecast[i])
  }

  return(forecast)
}

ar_coefs <- c(0.4928, -0.1480, 0.0967, 0.1441)
ma_coefs <- c(-0.2569, 0.0884, -0.0825, -0.2356, 0.3345)

n_ahead <- 10
forecast_n <- forecast_arima_4_0_5(x, ar_coefs, ma_coefs, n_ahead)
#forecast_n
xn_predictions <- predict(x.fit_n, 10)
forecast_n <- xn_predictions$pred
################################################################################

x_n <- train_data_north$North_Geomagnetic_Pole

detrended_series_n <- residuals(lm(x_n ~ time(x_n)))
d1_series_n <- diff(detrended_series_n)
d2_series_n <- diff(d1_series_n)

rev_d2_forecasts_n <- cumsum(c(d1_series_n[length(d1_series_n)], forecast_n[-1]))

rev_d1_forecasts_n <- cumsum(c(detrended_series_n[length(detrended_series_n)],
rev_d2_forecasts_n[1:(length(rev_d2_forecasts_n) - 1)]))

forecast_dates_n <- seq(max(time(x_n)) + 1/length(x_n), length = length(rev_d1_forecasts_n),
by = 1/length(x_n))
forecast_trend_n <- coef(lm(x_n ~ time(x_n)))[1] + forecast_dates_n * coef(lm(x_n ~
time(x_n)))[2]
final_forecast_n <- rev_d1_forecasts_n + forecast_trend_n
```

```
final_forecast_n

#xn_predictions <- predict(x.fit_n, n.ahead = 10)
#point_forecasts_n <- xn_predictions$pred
#se <- xn_predictions$se
#alpha <- 0.05
#z <- qnorm(1 - alpha / 2)
#lower_bound_n <- final_forecast_n - z * se
#upper_bound_n <- final_forecast_n + z * se
#forecast_data_df_bounds_n <- data.frame(Date = time(forecast_tn),
                                #Lower = lower_bound_n,
                               # Upper = upper_bound_n)
accuracy(final_forecast_n, test_data_north$North_Geomagnetic_Pole)
```

```{r}

forecast_ts_n <- ts(final_forecast_n[-1], start = end(x_n) + 1/length(x_n), frequency =
frequency(x_n))

original_data_df_n <- data.frame(Date = time(x_n), Value = as.vector(x_n))
forecast_data_df_n <- data.frame(Date = time(forecast_ts_n), Value = as.vector(forecast_ts_n))

test_data_ts_n <- ts(test_data_north$North_Geomagnetic_Pole, start =
end(train_data_north$North_Geomagnetic_Pole) +
1/length(train_data_north$North_Geomagnetic_Pole), frequency =
frequency(train_data_north$North_Geomagnetic_Pole))

test_data_df_n <- data.frame(Date = time(test_data_ts_n), Value = as.vector(test_data_ts_n))

p_n <- ggplot() +
  geom_line(data = original_data_df_n, aes(x = Date, y = Value), color = "black") +
  geom_line(data = forecast_data_df_n, aes(x = Date, y = Value), color = "red") +
  geom_line(data = test_data_df_n, aes(x = Date, y = Value), color = "blue") +
  theme_minimal() +
  labs(title = "ARIMA Forecast (North Geomagnetic Pole)", x = "Time", y = "Value") +
  scale_x_continuous(breaks = pretty(original_data_df_n$Date, n = 10)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

p_n


```

```{r}
n_full_n <- length(whole_data_north$North_Geomagnetic_Pole)
```

```
time_index_full_n <- 1:n_full_n
regression_full_n <- lm(whole_data_north$North_Geomagnetic_Pole ~ time_index_full_n)
detrend_north_full <- whole_data_north$North_Geomagnetic_Pole - predict(regression_full_n)
x_n_full <- diff(diff((detrend_north_full)))

x.fit_n_full <- Arima(x_n_full, order = c(4, 0, 5), method = "ML", include.mean = TRUE)

n_steps_ahead_n <- 10
x_n_predictions_full <- predict(x.fit_n_full, n.ahead = n_steps_ahead_n)
point_forecasts_n_full <- x_n_predictions_full$pred


rev_d2n_full <- cumsum(c(d1_series_n[length(d1_series_n)], point_forecasts_n_full))

rev_d1n_full <- cumsum(c(detrended_series_n[length(detrended_series_n)], rev_d2n_full[-1]))

forecast_dates_n_full <- seq(max(time(whole_data_north$North_Geomagnetic_Pole)) +
1/length(whole_data_north$North_Geomagnetic_Pole), length = length(rev_d1n_full), by =
1/length(whole_data_north$North_Geomagnetic_Pole))

detrend_n1 <- coef(lm(whole_data_north$North_Geomagnetic_Pole ~
time(whole_data_north$North_Geomagnetic_Pole)))[1]

detrend_n2 <- coef(lm(whole_data_north$North_Geomagnetic_Pole ~
time(whole_data_north$North_Geomagnetic_Pole)))[2]

forecast_trend_n_full <- detrend_n1  + forecast_dates_n_full * detrend_n2


final_forecasts_n_full <- rev_d1n_full + forecast_trend_n_full

forecast_ts_n_full <- ts(final_forecasts_n_full, start =
end(whole_data_north$North_Geomagnetic_Pole) +
1/length(whole_data_north$North_Geomagnetic_Pole), frequency =
frequency(whole_data_north$North_Geomagnetic_Pole))

forecast_data_df_n_full <- data.frame(Date = time(forecast_ts_n_full), Value =
as.vector(forecast_ts_n_full))

year_label_north <- function(x) {
  return(x + 1658)
}

p_n_full <- ggplot() +
  geom_line(data = original_data_df_n, aes(x = Date, y = Value), color = "black") +
  geom_line(data = forecast_data_df_n, aes(x = Date, y = Value), color = "red") +
```

```
  geom_line(data = test_data_df_n, aes(x = Date, y = Value), color = "blue") +
  geom_line(data = forecast_data_df_n_full, aes(x = Date, y = Value), color = "purple") +
  theme_minimal() +
  labs(title = "North Geomagnetic intensities Forecast", x = "Time", y = "magnetic intensities") +
  scale_x_continuous(labels = year_label_north,
                     breaks = seq(0, 2033 - 1658, by = 15),
                     limits = c(0, 2033 - 1658)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

p_n_full

```

R code South Geomagnetic Pole

```{r}

ts_data_s <- ts(train_data$South_Geomagnetic_Pole, start = c(1590, 1), frequency =10 )
#ts_data_s <- ts(xs, start = c(1590, 1), frequency = 10)
start_date <- start(ts_data_s)
end_date <- end(ts_data_s)

plot(decompose(ts_data_s))
```

```{r,echo=FALSE,message=FALSE,out.width="49%",out.height="49%",fig.show='hold',fig.align='center'}

n <- length(train_data$South_Geomagnetic_Pole)
time_index <- 1:n
regression <- lm(train_data$South_Geomagnetic_Pole ~ time_index)
detrend_south <- train_data$South_Geomagnetic_Pole - predict(regression)
#ts(detrend_south)
xs <- diff(diff((detrend_south)))

#########################################################################################
cummeanxs <- cumsum(xs)/seq_along(xs)
par(mfrow = c(2,1), mar = c(4,4,4,4))
plot.ts(xs,type = 'l', xlab = "Time", ylab = "Time Series x")
plot(cummeanxs, type = "l", xlab = "Time", ylab = "Mean Level")
par(mfrow = c(2,1), mar = c(4,4,4,4))
acf(xs, xlab = "lag", ylab = "Sample ACF", main = "") #can add  xlim =c(1,25)
acf(xs, type = "partial", xlab = "lag", ylab = "Sample PACF", main= "")
```

```{r,echo=FALSE,message=FALSE, warning = FALSE}
# Model Specification SOUTH
# Perform ADF test with no constant (nc)
adf_nc_s <- adfTest(xs, type = "nc", lags = 13)

# Perform ADF test with constant only (c)
adf_c_s <- adfTest(xs, type = "c", lags = 13)
```

```
# Perform ADF test with constant and trend (ct)
adf_ct_s <- adfTest(xs, type = "ct", lags = 13)

# Print the results
adf_nc_s
adf_c_s
adf_ct_s

x.kpss_level_s <- kpss.test(xs, null = "Level")
x.kpss_trend_s <- kpss.test(xs, null = "Trend")
x.kpss_level_s
x.kpss_trend_s

xs.eacf <- eacf(xs, ar.max = 5, ma.max = 5)
```

```{r,echo=FALSE,message=FALSE}

xs.aic = matrix(0,5,5)
xs.bic = matrix(0,5,5)

#### check
for (i in 0:4) for (j in 0:4){
  xs.fit <- Arima(xs,order=c(i,0,j), method = "ML", include.mean = TRUE)
  xs.aic[i+1, j+1] <- xs.fit$aic
  xs.bic[i+1,j+1] <- BIC(xs.fit)
}
```

```{r,echo=FALSE,message=FALSE}
xs.aic.sort <- sort(xs.aic)
indices2 <- arrayInd(order(xs.aic), dim(xs.aic))
aic2.df <- data.frame(row = indices2[,1], col = indices2[,2] , value = xs.aic.sort)
aic2.v <- kable(aic2.df, row.names = FALSE, col.names = c("AR", "MA", "Value")) %>%
   kable_styling()
aic2.v
```

```{r,echo=FALSE,message=FALSE}
xs.bic.sort <- sort(xs.bic)
indices22 <- arrayInd(order(xs.bic), dim(xs.bic))
bic2.df <- data.frame(row = indices22[,1], col = indices22[,2] , value = xs.bic.sort)
bic2.v <- kable(bic2.df, row.names = FALSE, col.names = c("AR", "MA", "Value")) %>%
   kable_styling()
bic2.v
```

```{r,echo=FALSE,message=FALSE,out.width="49%",out.height="49%",fig.show='hold',fig.align='center'}
x.fit_s = Arima(xs, order = c(5,0,5), method = "ML", include.mean = TRUE)
##############################################
##tsdia()with my own dof
residuals_arima <- residuals(x.fit_s)
acf_resid <- acf(residuals_arima, plot = FALSE)

# Plot the ACF
plot(acf_resid, main = "ACF of Residuals with Custom Confidence Interval")

# Calculate the custom confidence interval
alpha <- 0.05  # 95% confidence interval (change alpha for other confidence levels)
dof <- 389-(3+2)  # Your desired degrees of freedom
critical_value <- qnorm(1 - alpha / 2) / sqrt(dof)

# Add the custom confidence interval lines
abline(h = c(-critical_value, critical_value), col = "blue", lwd = 2, lty = "dashed")


##############################################
#tsdiag(x.fit_s)
qqnorm(residuals(x.fit_s))
qqline(residuals(x.fit_s))

shapiro.test(residuals(x.fit_s))


acf(x.fit_s$residuals)
pacf(x.fit_s$residuals)
```

```{r}
ar_param_s <- x.fit_s$coef[grepl("^ar", names(x.fit_s$coef))]
ma_param_s <- x.fit_s$coef[grepl("^ma", names(x.fit_s$coef))]
intercept_param_s <- x.fit_s$coef["intercept"]

cat("AR Parameters (MLEs):\n")
print(ar_param_s)
cat("\nMA Parameters (MLEs):\n")
print(ma_param_s)
cat("\nIntercept Parameter (MLE):\n")
print(intercept_param_s)
```

```
residuals_south <- residuals(x.fit_s)
# Perform the test
ljung_box_test_s <- Box.test(residuals_south, lag = 13, type = "Ljung-Box")

# Print the test results
print(ljung_box_test_s)
```

```{r}
####SOUTH
n_steps <- 13
xs_predictions <- predict(x.fit_s, n.ahead = n_steps)
#point_forecasts_s <- xs_predictions$pred
point_forecasts_s <- forecasts
point_forecasts_ss <- final_forecasts
se <- xs_predictions$se
alpha <- 0.05
z <- qnorm(1 - alpha / 2)
#lower_bounds <- point_forecasts_ss - z * se
#upper_bounds <- point_forecasts_ss + z * se
#forecast_data_df_bounds <- data.frame(Date = time(forecast_ts),
                                      # Lower = lower_bounds,
                                      # Upper = upper_bounds)


xss <- train_data$South_Geomagnetic_Pole
xs_ts <- ts(xss, start = 1, frequency = 1)

detrended_series <- residuals(lm(xss ~ time(xss)))
d1_series <- diff(detrended_series)
d2_series <- diff(d1_series)

rev_d2_forecasts <- cumsum(c(d1_series[length(d1_series)], point_forecasts_s))

rev_d1_forecasts <- cumsum(c(detrended_series[length(detrended_series)], rev_d2_forecasts[-
1]))

forecast_dates <- seq(max(time(xss)) + 1/length(xss), length = length(rev_d1_forecasts), by =
1/length(xss))
forecast_trend <- coef(lm(xss ~ time(xss)))[1] + forecast_dates * coef(lm(xss ~ time(xss)))[2]
final_forecasts <- rev_d1_forecasts + forecast_trend

forecast_ts <- ts(final_forecasts[-1], start = end(xss) + 1/length(xss), frequency = frequency(xss))

original_data_df_s <- data.frame(Date = time(xs_ts), Value = as.vector(xs_ts))
```

```
forecast_data_df_s <- data.frame(Date = time(forecast_ts), Value = as.vector(forecast_ts))

test_data_ts <- ts(test_data$South_Geomagnetic_Pole, start =
end(train_data$South_Geomagnetic_Pole) + 1/length(train_data$South_Geomagnetic_Pole),
frequency = frequency(train_data$South_Geomagnetic_Pole))

test_data_df_s <- data.frame(Date = time(test_data_ts), Value = as.vector(test_data_ts))

p_s <- ggplot() +
  geom_line(data = original_data_df_s, aes(x = Date, y = Value), color = "black") +
  geom_line(data = forecast_data_df_s, aes(x = Date, y = Value), color = "red") +
  geom_line(data = test_data_df_s, aes(x = Date, y = Value), color = "blue") +
  theme_minimal() +
  labs(title = "ARIMA Forecast", x = "Time", y = "Value") +
  scale_x_continuous(breaks = pretty(original_data_df$Date, n = 10)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

p_s

accuracy(final_forecasts, test_data$South_Geomagnetic_Pole)

```

```{r}
###### PREDICTION SOUTH
ar_coefs <- c(-0.1819, 0.0337, 0.0847, 0.0204, -0.6777)
ma_coefs <- c(0.6819, 0.3315, 0.1206, 0.1558, 0.3929)
intercept <- 0.0416

forecast_arima_5_0_5 <- function(data, ar_coefs, ma_coefs, intercept, errors) {
  ar_term <- sum(ar_coefs * data[(length(data)-4):(length(data))])
  ma_term <- sum(ma_coefs * errors)
  return (ar_term + ma_term + intercept)
}

n <- length(xs)
h <- 13 # Number of steps ahead you want to forecast
forecasts <- numeric(h)
forecasts_diff <- numeric(h)
forecasts_cum_diff <- numeric(h)

data <- xs
errors <- c(rep(0, 5))
```

```
for (i in 1:h) {
  forecasts[i] <- forecast_arima_5_0_5(data, ar_coefs, ma_coefs, intercept, errors)
  data <- c(data, forecasts[i])
  errors <- c(errors[-1], forecasts[i] - data[length(data) - 1])
}

forecasts
```

```
```

##

TESTING

```
```{r}
south_full <- my_data$South_Geomagnetic_Pole
n_full <- length(south_full)
time_index_full <- 1:n_full
regression_full <- lm(south_full ~ time_index_full)
detrend_south_full <- south_full - predict(regression_full)
xs_full <- diff(diff((detrend_south_full)))

x.fit_s_full <- Arima(xs_full, order = c(5, 0, 5), method = "ML", include.mean = TRUE)

n_steps_ahead <- 10
xs_predictions_full <- predict(x.fit_s_full, n.ahead = n_steps_ahead)
point_forecasts_s_full <- xs_predictions_full$pred

rev_d2_forecasts_full <- cumsum(c(d1_series[length(d1_series)], point_forecasts_s_full))

rev_d1_forecasts_full <- cumsum(c(detrended_series[length(detrended_series)],
rev_d2_forecasts_full[-1]))

forecast_dates_full <- seq(max(time(south_full)) + 1/length(south_full), length =
length(rev_d1_forecasts_full), by = 1/length(south_full))
forecast_trend_full <- coef(lm(south_full ~ time(south_full)))[1] + forecast_dates_full *
coef(lm(south_full ~ time(south_full)))[2]
final_forecasts_full <- rev_d1_forecasts_full + forecast_trend_full

forecast_ts_full <- ts(final_forecasts_full, start = end(south_full) + 1/length(south_full),
frequency = frequency(south_full))
```
```

```
forecast_data_df_s_full <- data.frame(Date = time(forecast_ts_full), Value =
as.vector(forecast_ts_full))

year_label <- function(x) {
  return(x + 1590)
}

p_s_full <- ggplot() +
  geom_line(data = original_data_df_s, aes(x = Date, y = Value), color = "black") +
  geom_line(data = forecast_data_df_s, aes(x = Date, y = Value), color = "red") +
  geom_line(data = test_data_df_s, aes(x = Date, y = Value), color = "blue") +
  geom_line(data = forecast_data_df_s_full, aes(x = Date, y = Value), color = "purple") +
  theme_minimal() +
  labs(title = "South Geomagnetic intensities Forecast", x = "Time", y = "magnetic intensities") +
  scale_x_continuous(labels = year_label,
                     breaks = seq(0, 2033 - 1590, by = 15),
                     limits = c(0, 2033 - 1590)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

p_s_full
```