Econ 144 Project 2

Nicholas Cassol-Pawson

May 17, 2024

Contents


```
library(forecast)
library(vars)
library(tidyverse)
library(strucchange)
```

I. Introduction

In this project we will take advantage of 2 data sets.

The first data set is a series of observations from January 2000 to December 2023 of the deviation in average global monthly surface temperature from the average value between the baseline period from 1951 to 1980. We are interested in seeing if we can predict future increases in the monthly surface temperature.

```
surf_temp <- read.csv("MeanGlobalMonthlyTemperature.csv") |>
  dplyr::select(Year:Dec) |>
  pivot_longer(cols = Jan:Dec, names_to = "Month", values_to = "AvTemp") |>
  dplyr::filter(Year >= 2000) |>
  dplyr::select(AvTemp) %>%
  .[[1]] |>
  ts(start = c(2000, 1), freq = 12)
```

The other data set is a series of monthly observations from January 2000 to December 2023 of the average sea surface temperature (SST) over a region of the ocean from the international dateline to the coast of South America. This is a measure of the El Niño effect over the region defined as 3.4. When the SST in this region are more than 0.4°C over or under the average temperature for 6 months, scientists define an El Niño or La Niña event. We wish to predict future values of the SST, so that we can determine whether an El Niño phenomenon will occur in the coming year. The data are merely an average of the surface temperature in degrees Celsius. We import it:

```
elnino <- read.csv("ElNino.csv") |>
  pivot_longer(cols = Jan:Dec, names_to = "Month", values_to = "ElNino") |>
```

```
  dplyr::filter(Year >= 2000) |>
  dplyr::select(ElNino) %>%
  .[[1]] |>
  ts(start = c(2000, 1), freq = 12)
```

II. Results

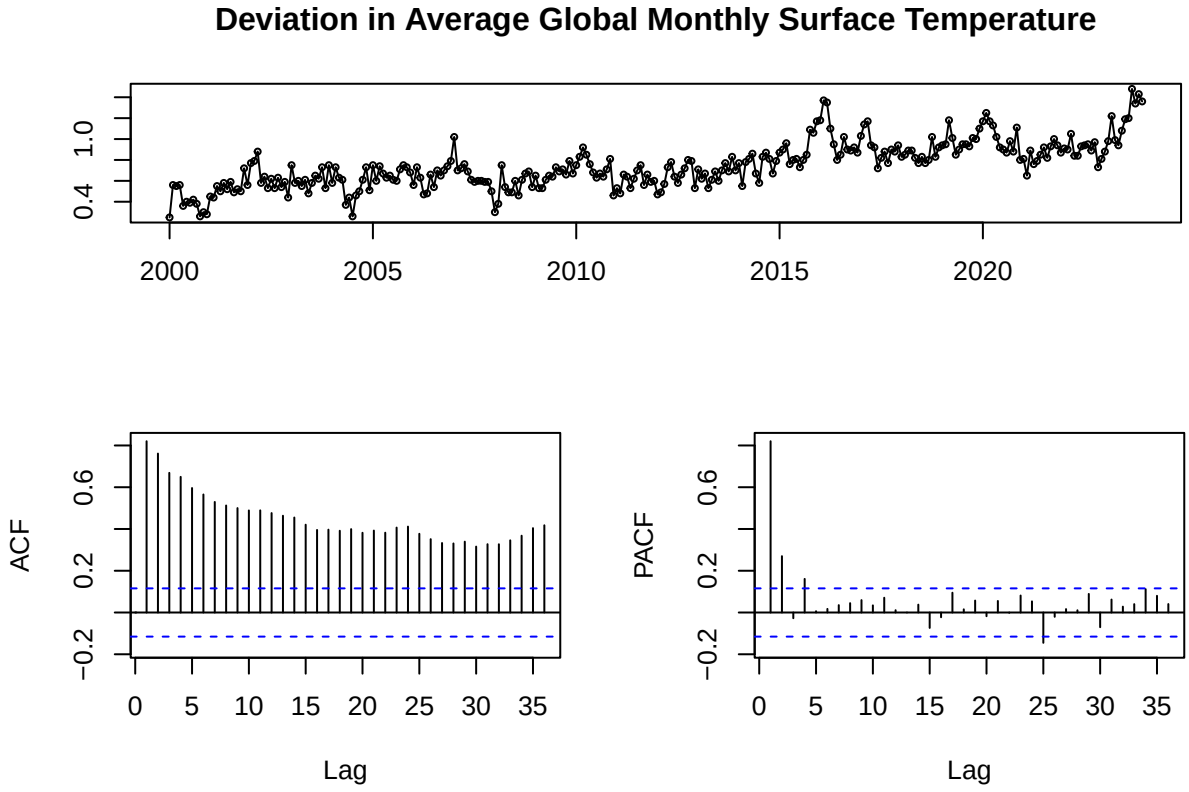
We will first fully model the global surface temperature data set, then move on to the El Niño data set, before exploring the two together.

Global Average Surface Temperature Deviation

(a)

We begin the analysis by examining the plot of the data, its ACF, and its PACF:

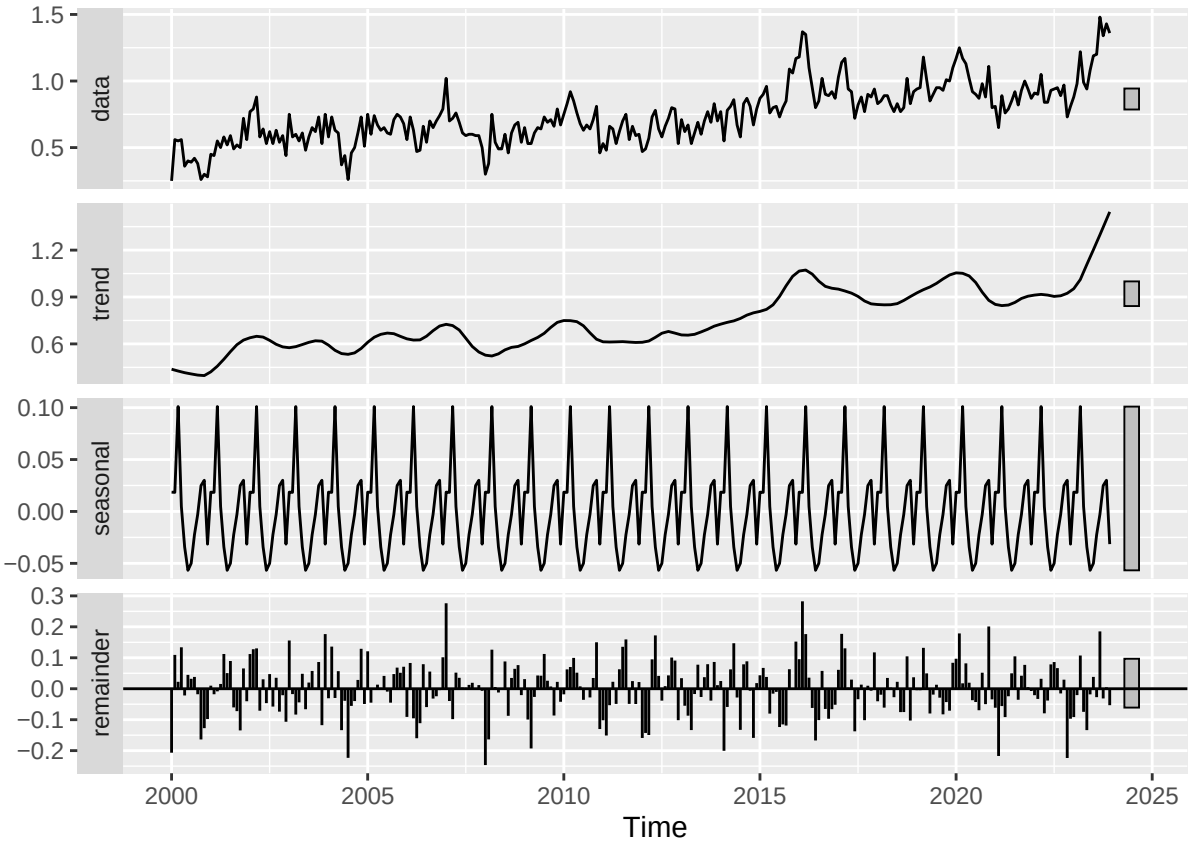
```
tsdisplay(surf_temp, main = "Deviation in Average Global Monthly Surface Temperature")
```



The preliminary overview indicates the presence of some interesting dynamics in the data: there appears to be some sort of underlying AR process, with significant spikes on the first two and potentially the fourth lag, and the data displaying some degree of persistence. Notice in the ACF plot the peaks in lags near the 12th and 24th lag, this suggests that there may be some sort of S-MA(q) process underlying the data as well.

(b)

```
decomposed_surf <- stl(surf_temp, s.window = "periodic")
autoplot(decomposed_surf)
```



We can see that there is a very clear trend to the data, it appears to be somewhat linear. The remainder has a larger range than the seasonal component, covering the values of (-0.2, 0.3), while the seasonality is about a third of the size, over values of (-0.05, 0.1), but since they are within an order of magnitude of each other, we can conclude that they are both significant.

(c)

We now need to model the time series. We will model each component individually. We start with the trend:

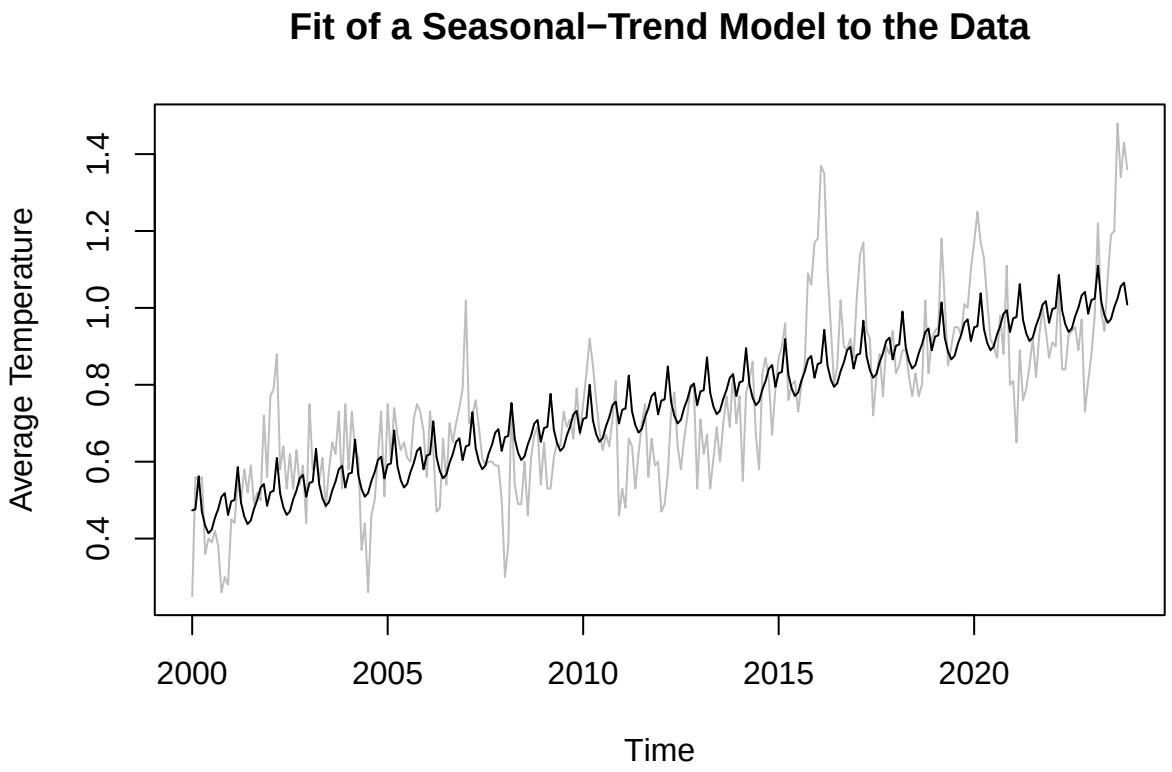
First, we create a time dummy variable to do this modeling, with a value for every month included in the data set.

```
t <- seq(2000, 2024 - 1/12, by = 1 / 12)
```

We now examine the trend component from the decomposition. It appears to be vaguely linear, although it has a non-linear increase towards the last few years. However, we choose to stick with a linear model for the data.

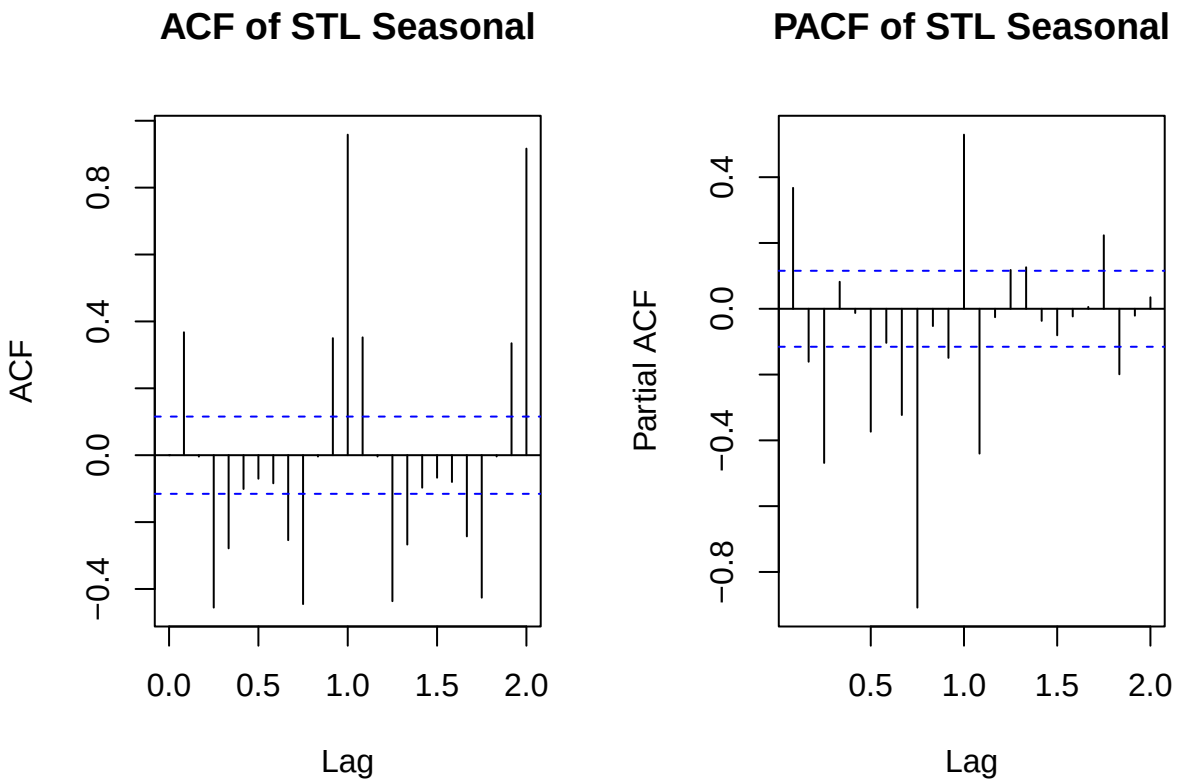
When we model the linearity, we will also model the seasonality. We will do this with a time dummy variable through the built in function `tslm` utilizing the `seasonal` component. We plot the model to see how poorly it fits the data:

```
prelim_model_surf <- tslm(surf_temp ~ t + season)
plot(surf_temp, col = "gray", main = "Fit of a Seasonal-Trend Model to the Data",
     ylab = "Average Temperature")
lines(prelim_model_surf$fitted.values, col = "black")
```



We can see that we need to capture the cyclical components of the data in the model still. We will therefore transition to using a seasonal ARIMA model, where we set the I component to be 1. We now try to determine orders for the ARMA and S-ARMA components of the model, using the STL decomposition from part (b). First, the seasonal components:

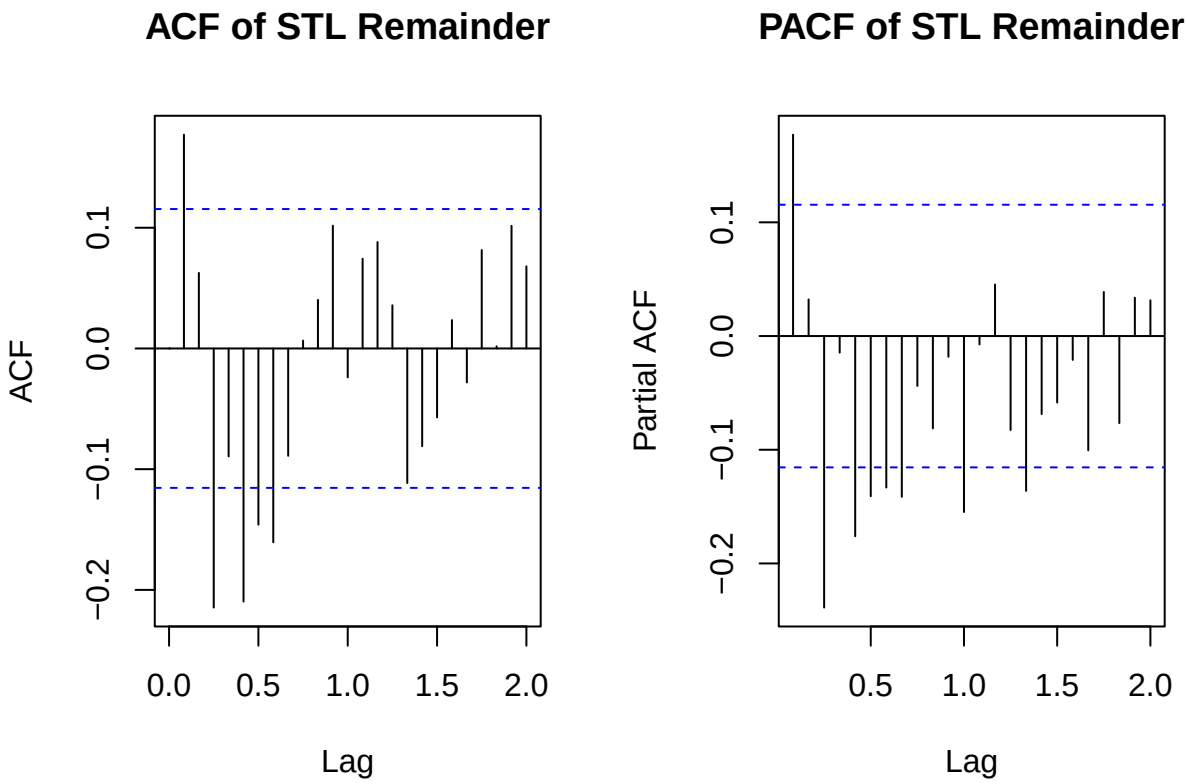
```
par(mfrow = c(1, 2))
seasonal_acf_plot_surf <- acf(decomposed_surf$time.series[, "seasonal"], plot = FALSE)
seasonal_acf_plot_surf$acf[1, 1, 1] <- 0
plot(seasonal_acf_plot_surf, main = "ACF of STL Seasonal")
pacf(decomposed_surf$time.series[, "seasonal"], main = "PACF of STL Seasonal")
```



Looking at the plots of the ACF and PACF of the STL seasonal component, we see that there are many significant lags, including at the first and second lag of the ACF and the first on the PACF. This is indicative of using an S-ARMA(1, 2) model. Hopefully this will be able to deal with most of the funky components of the data.

We now carry out the same procedure for the remainder component, to approximate cycles:

```
par(mfrow = c(1, 2))
remainder_acf_plot_surf <- acf(decomposed_surf$time.series[, "remainder"], plot = FALSE)
remainder_acf_plot_surf$acf[1, 1, 1] <- 0
plot(remainder_acf_plot_surf, main = "ACF of STL Remainder")
pacf(decomposed_surf$time.series[, "remainder"], main = "PACF of STL Remainder")
```



We see that the PACF plot, while messy, appears to decay faster than the ACF, which has significant lags through lag 7 or so. We propose modeling the remainder with a MA(7) model, although there likely is a better model including an AR process, but I am not sure how to find it.

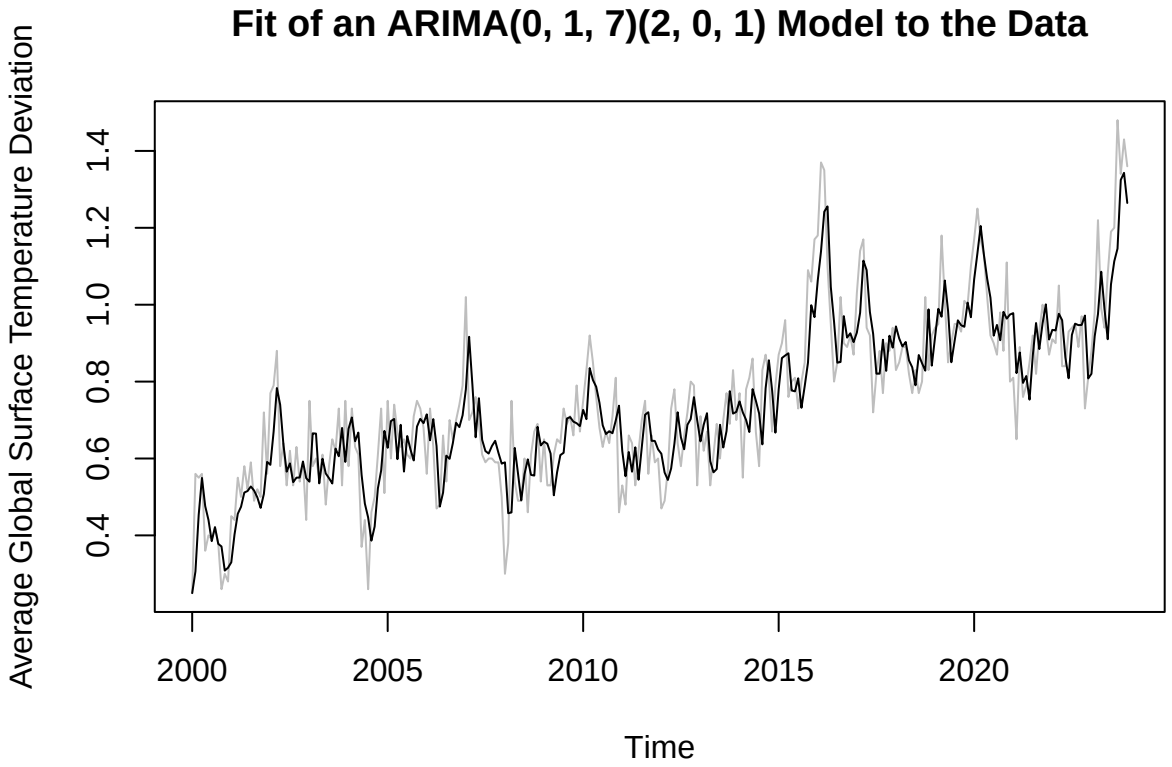
We now build our ARIMA(0, 1, 7)(1, 2) model:

```
manual_arima_model_surf <- arima(surf_temp, order = c(0, 1, 7),
                                 seasonal = list(order = c(1, 0, 2)))
summary(manual_arima_model_surf)
```

```
##
## Call:
## arima(x = surf_temp, order = c(0, 1, 7), seasonal = list(order = c(1, 0, 2)))
##
## Coefficients:
##           ma1     ma2      ma3     ma4      ma5     ma6      ma7    sar1
##       -0.4379  0.0115  -0.2247  0.1294  -0.1559  0.0217  -0.1445  0.9993
## s.e.   0.0600  0.0647   0.0636  0.0681   0.0700  0.0749   0.0662  0.0026
##          sma1    sma2
##       -1.0607  0.0752
## s.e.   0.0655  0.0601
##
## sigma^2 estimated as 0.01039:  log likelihood = 239.6,  aic = -457.19
##
## Training set error measures:
##                       ME      RMSE        MAE        MPE     MAPE      MASE
## Training set 0.006881308 0.1017726 0.08049806 -0.9703451 11.66137 0.8277657
##                     ACF1
## Training set 0.007190136
```

Most of the lags in this model appear significant. We plot the updated model:

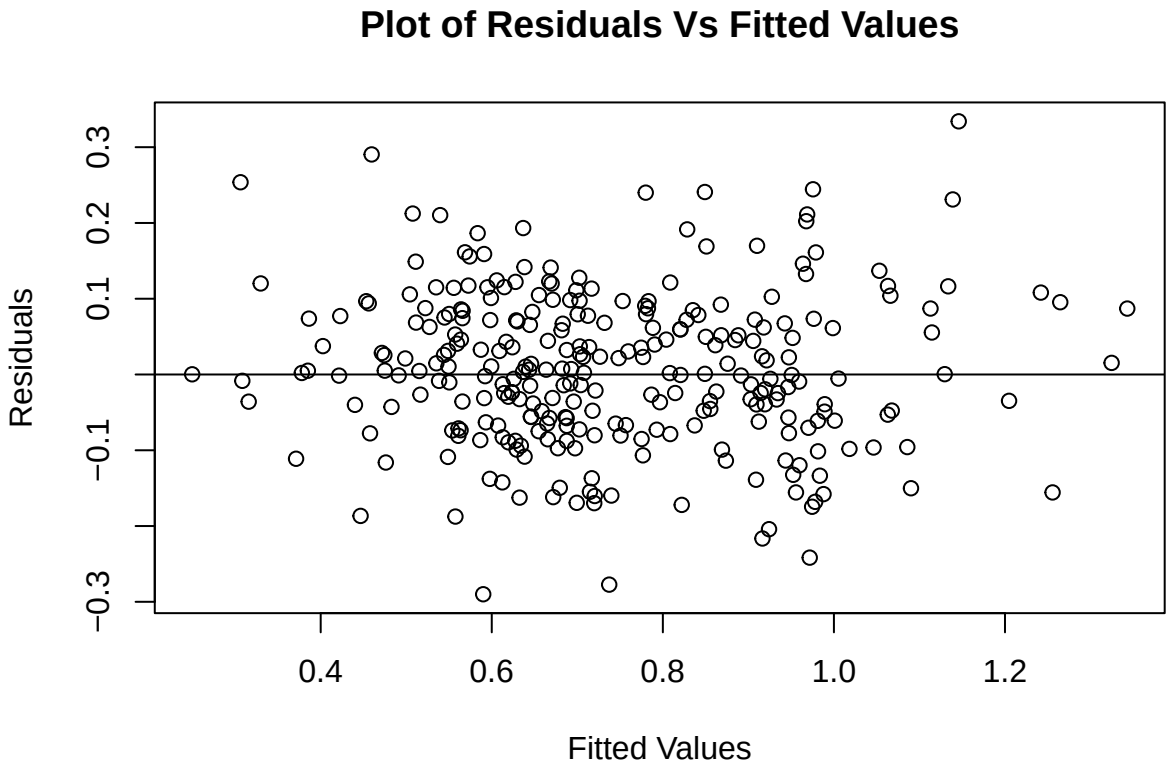
```r
plot(surf_temp, col = "gray", main = "Fit of an ARIMA(0, 1, 7)(2, 0, 1) Model to the Data",
     ylab = "Average Global Surface Temperature Deviation")
lines(fitted(manual_arima_model_surf), col = "black")
```



We see that the ARIMA model provides a rather strong fit to the data, tracing it quite well. However, the model is not very parsimonious, implying there is a better model out there.

(e)

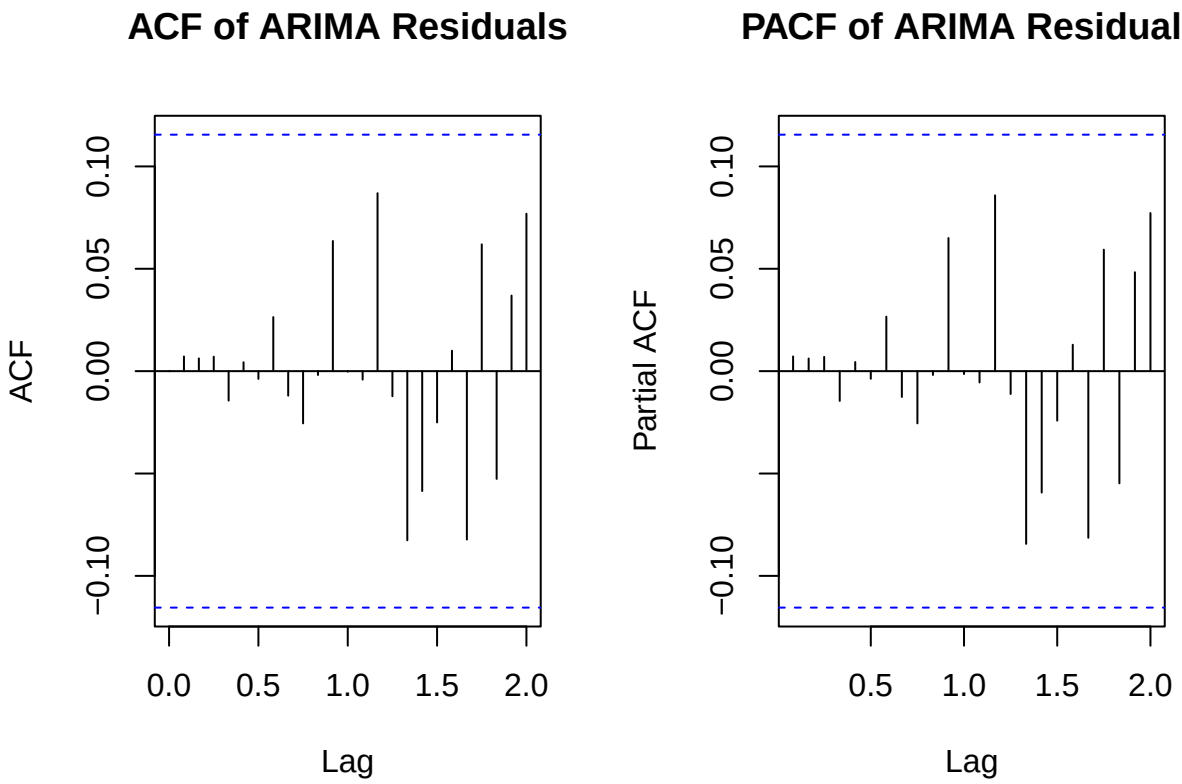
```r
plot(fitted(manual_arima_model_surf), manual_arima_model_surf$resid,
     main = "Plot of Residuals Vs Fitted Values", ylab = "Residuals", xlab = "Fitted Values")
abline(h = 0, col = "black")
```



We see that the residuals appear to be randomly scattered with constant variance about a mean of 0, indicating that the assumption that the error is distributed as $WN(0, \sigma^2)$ is satisfied.

(f)

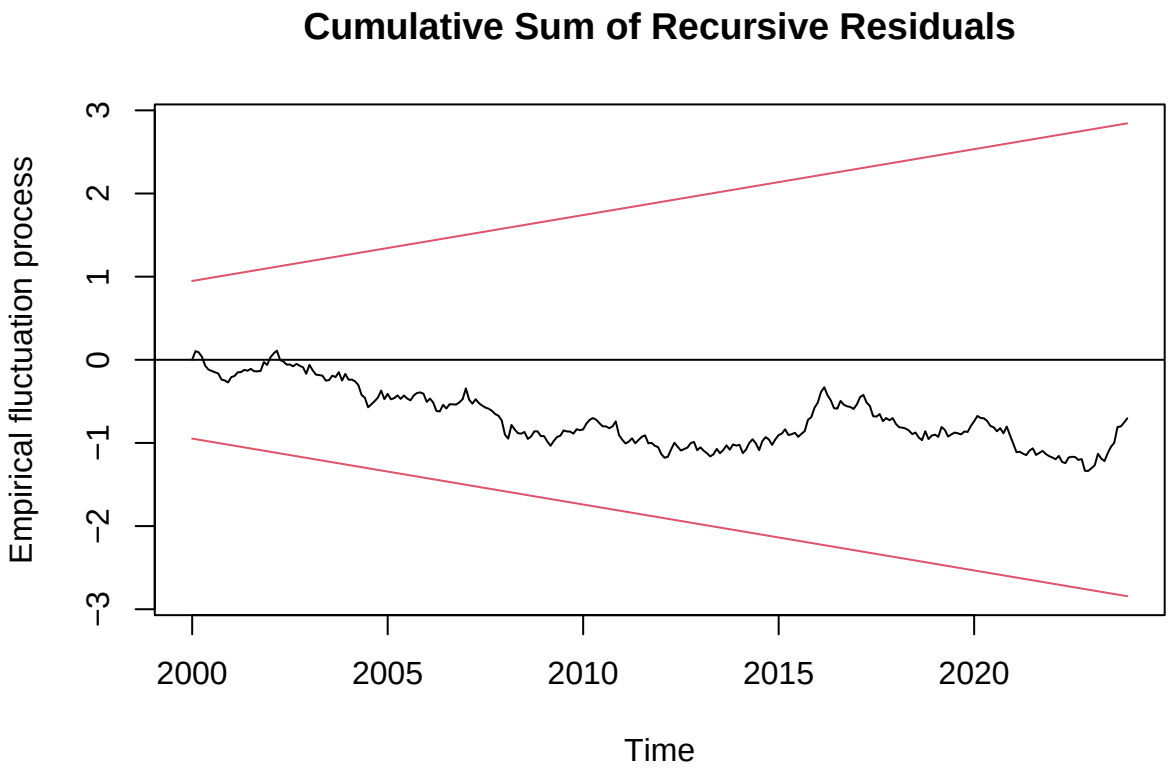
```
par(mfrow = c(1, 2))
resid_acf_plot_surf <- acf(manual_arima_model_surf$residuals, plot = FALSE)
resid_acf_plot_surf$acf[1, 1, 1] <- 0
plot(resid_acf_plot_surf, main = "ACF of ARIMA Residuals")
pacf(manual_arima_model_surf$residuals, main = "PACF of ARIMA Residuals")
```



We see that the model has accounted for any sort of significant lag in the ACF and PACF of the residuals, implying the model is capturing any sort of dynamics in the data.

(g)

```
plot(efp(manual_arima_model_surf$residuals ~ 1, type = "Rec-CUSUM"),
     main = "Cumulative Sum of Recursive Residuals")
```



Examining the cumulative sum of the recursive residuals, we see that the model is robust, with the cumulative sum of the residuals indicating that any shocks to the data never cause the model's parameters to exceed the confidence bands. This indicates that the model is robust.

(h)

We will examine a few of the accuracy statistics of the model to determine how good of a fit it is to the data.

```
accuracy(manual_arima_model_surf)
```

```
##                       ME      RMSE        MAE        MPE     MAPE      MASE
## Training set 0.006881308 0.1017726 0.08049806 -0.9703451 11.66137 0.8277657
##                      ACF1
## Training set 0.007190136
```

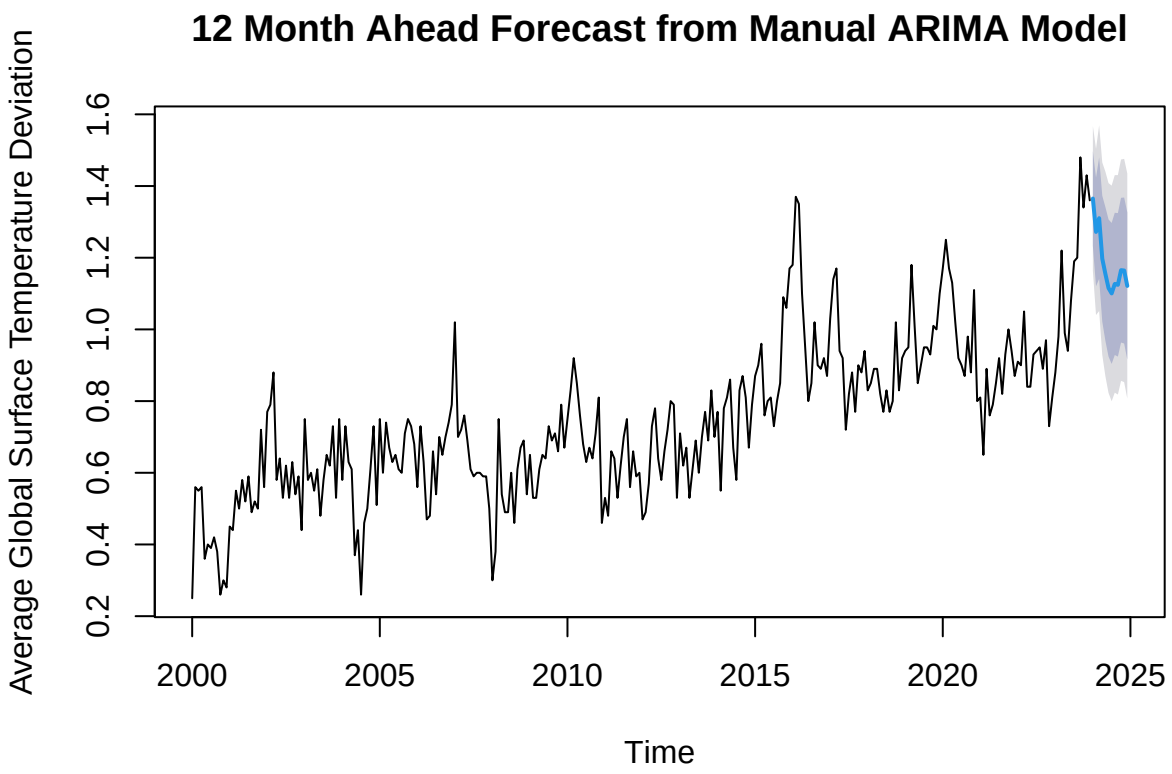
There are a few interesting and easy to interpret statistics in here.

The RMSE indicates that the model's predictions of average temperature deviations, on average, are 0.10°C from the true value. This is decent, although as indicated by the MAPE, this is a deviation of about 11% from the true values.

The other statistics say similar things or are not very useful in the context of the data, but overall, they indicate that the model provides a decent fit to the data.

(i)

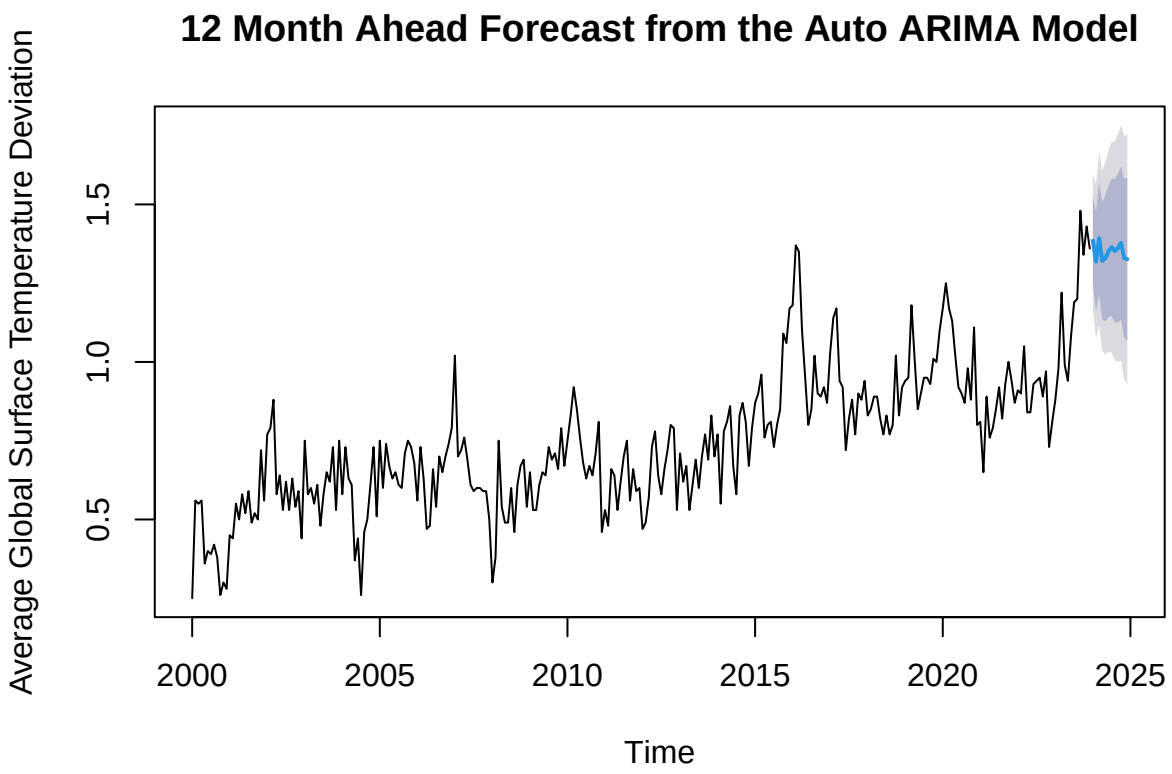
```
manual_forecast_surf <- forecast(manual_arima_model_surf, h = 12)
plot(manual_forecast_surf, ylab = "Average Global Surface Temperature Deviation",
     xlab = "Time", main = "12 Month Ahead Forecast from Manual ARIMA Model")
```



We see that the forecast from the manual ARIMA model expects the average global surface temperature deviations to decline over the next 12 months.

(j)

```
auto_arima_model_surf <- auto.arima(surf_temp)
auto_arima_forecast_surf <- forecast(auto_arima_model_surf, h = 12)
plot(auto_arima_forecast_surf, main = "12 Month Ahead Forecast from the Auto ARIMA Model",
     ylab = "Average Global Surface Temperature Deviation", xlab = "Time")
```



The auto ARIMA model expects a much smaller decline in the average global surface temperature deviations than the manual one does.

We now compute the MAPE for the auto ARIMA model:

```
cat("Model          MAPE", "\nARIMA      ", accuracy(manual_arima_model_surf)[5],
    "\nAuto ARIMA ", accuracy(auto_arima_model_surf)[5])
```

```
## Model          MAPE
## ARIMA       11.66137
## Auto ARIMA  12.15229
```

```
cat("Model       Predictors", "\nARIMA      ", length(manual_arima_model_surf$coef),
    "\nAuto ARIMA ", length(auto_arima_model_surf$coef))
```

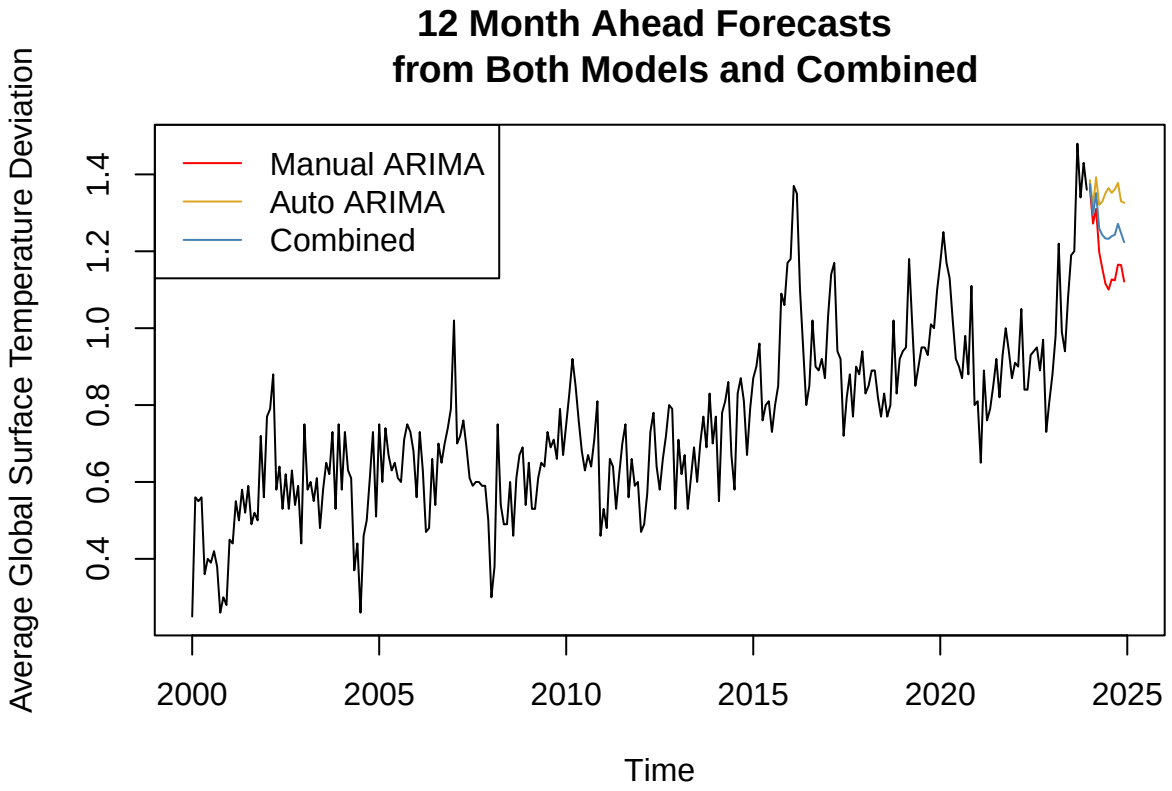
```
## Model       Predictors
## ARIMA       10
## Auto ARIMA  7
```

We see that the MAPE of the auto ARIMA model is higher than that from the ARIMA model defined in part (c). However, it is also less complex, with 7 instead of 10 terms, so it is preferable, even if a lower MAPE is better.

(k)

```
combined_forecast_surf <- (auto_arima_forecast_surf$mean + manual_forecast_surf$mean) / 2
plot(surf_temp, xlim = c(2000, 2025), main = "12 Month Ahead Forecasts
     from Both Models and Combined", ylab = "Average Global Surface Temperature Deviation",
     xlab = "Time")
lines(seq(2024, 2025 - 1/12, by = 1/12), manual_forecast_surf$mean, col = "red")
lines(seq(2024, 2025 - 1/12, by = 1/12), auto_arima_forecast_surf$mean,
      col = "goldenrod")
```

```
lines(seq(2024, 2025 - 1/12, by = 1/12), combined_forecast_surf, col = "steelblue")
legend("topleft", legend = c("Manual ARIMA", "Auto ARIMA", "Combined"),
       lty = c(1, 1, 1), col = c("red", "goldenrod", "steelblue"))
```



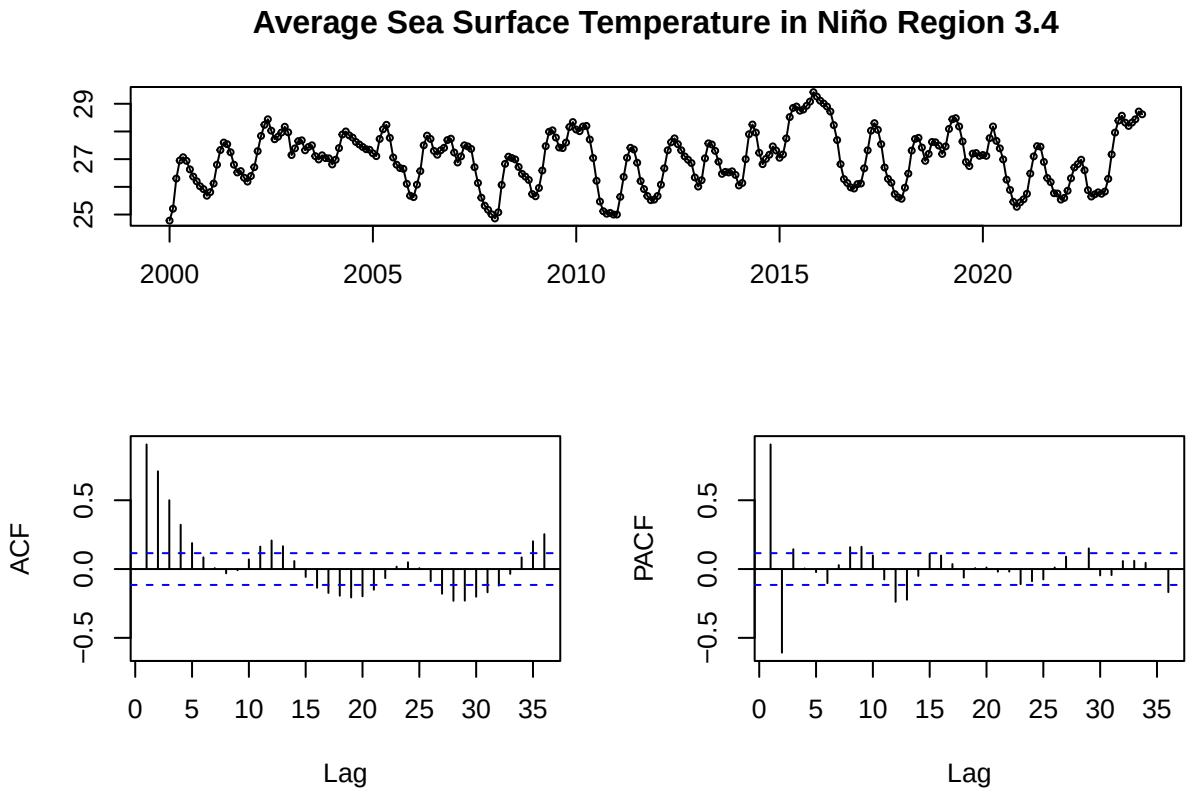
We see that the forecasted value lies between the two forecasts, not falling as quickly as the manual model wishes, but not staying as high as the auto ARIMA wishes either. The MAPE of the combined forecast will be the average of the MAPE of the two forecasts, as this forecast is merely the average of the other two forecasts.

Average Sea Surface Temperature in Niño Region 3.4

(a)

We begin the analysis by examining the plot of the data, its ACF, and its PACF:

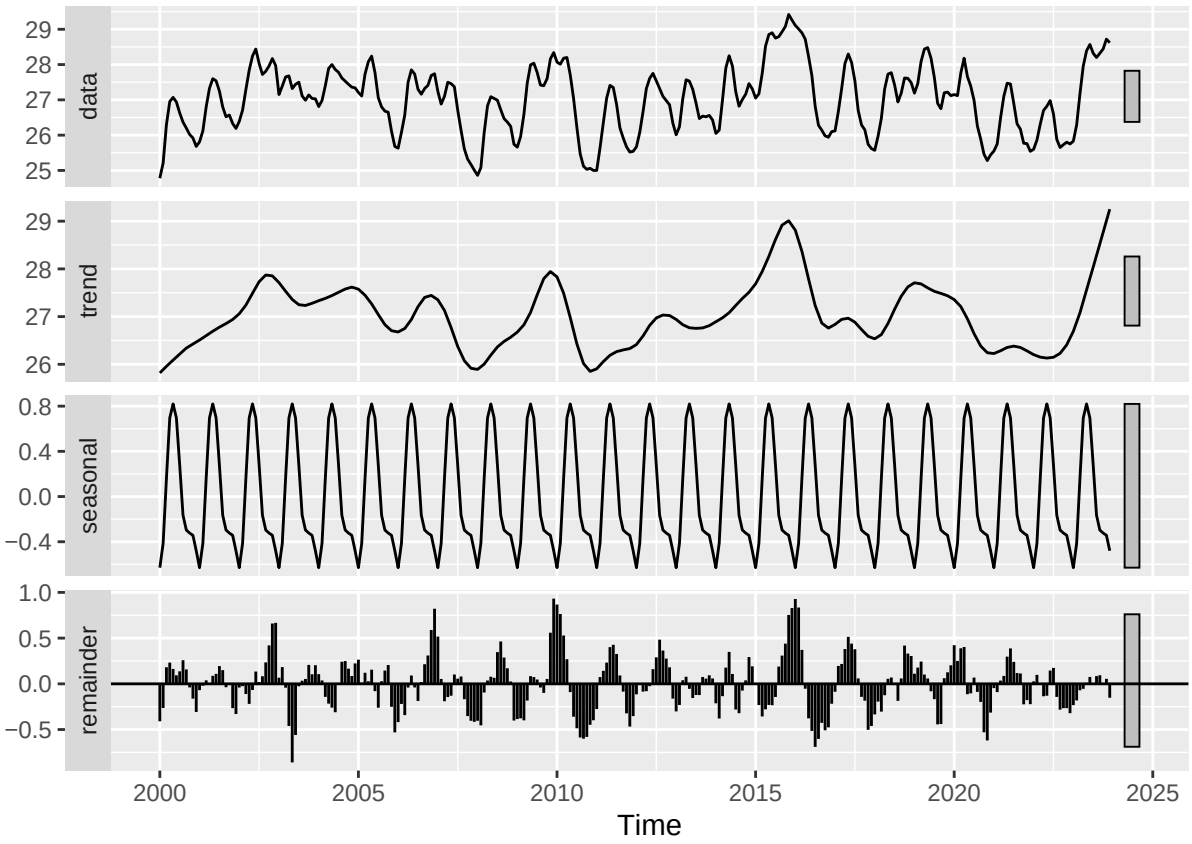
```
tsdisplay(elnino, main = "Average Sea Surface Temperature in Niño Region 3.4")
```



The preliminary overview indicates the presence of some interesting dynamics in the data: there appears to be some sort of underlying AR process underlying, with significant spikes on the first two lags, and the data displaying some degree of persistence. Notice in the ACF plot the cyclical nature of the decay in the lags, this suggests that there may be some sort of underlying MA process in the data as well.

(b)

```
decomposed_nino <- stl(elnino, s.window = "periodic")
autoplot(decomposed_nino)
```

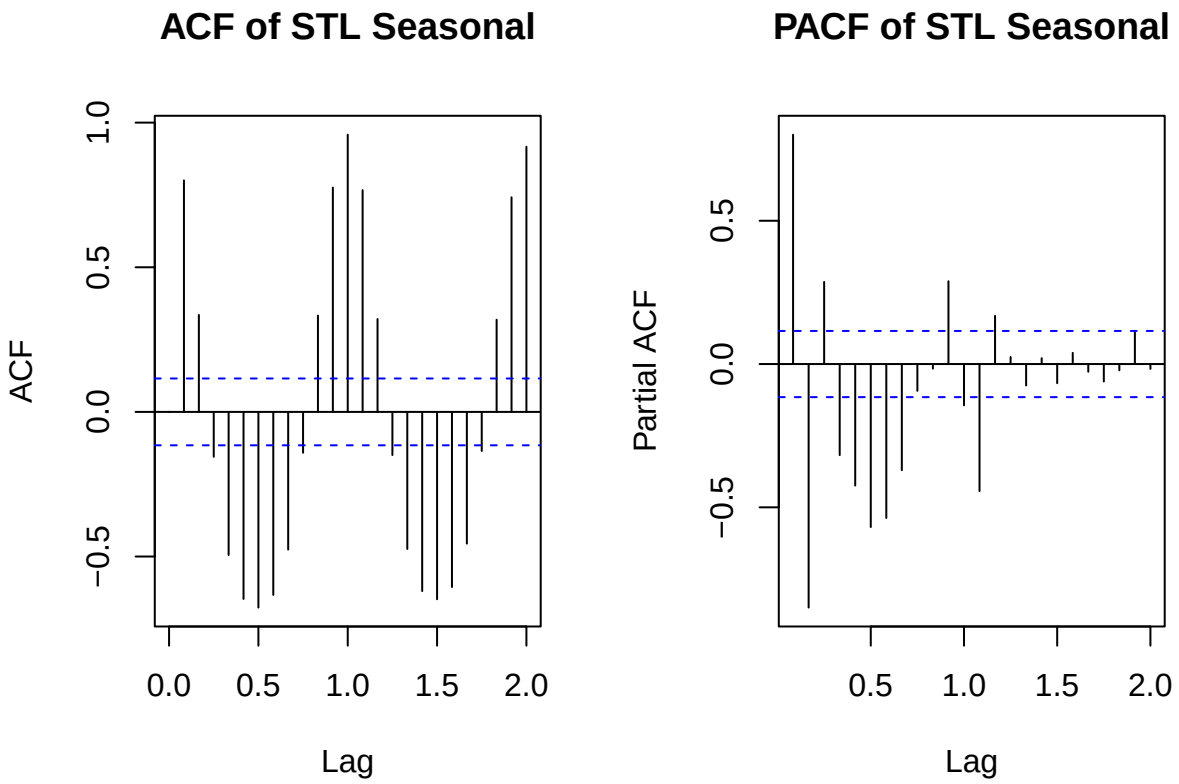


We can see that there is a slightly linear trend to the data, although it is somewhat constant over time. The remainder and seasonal components have similar ranges, with the seasonal covering the values of (-0.4, 0.8), and the remainder, values of (-0.5, 1.0). We can conclude that they are both significant.

(c)

We now need to model the time series. We will follow the ARIMA procedure that we ran for the global surface temperature data set. We first will take the first difference of the data to ensure stationarity. Now we examine the seasonal component's ACF and PACF to determine the order of the S-ARMA we wish to fit:

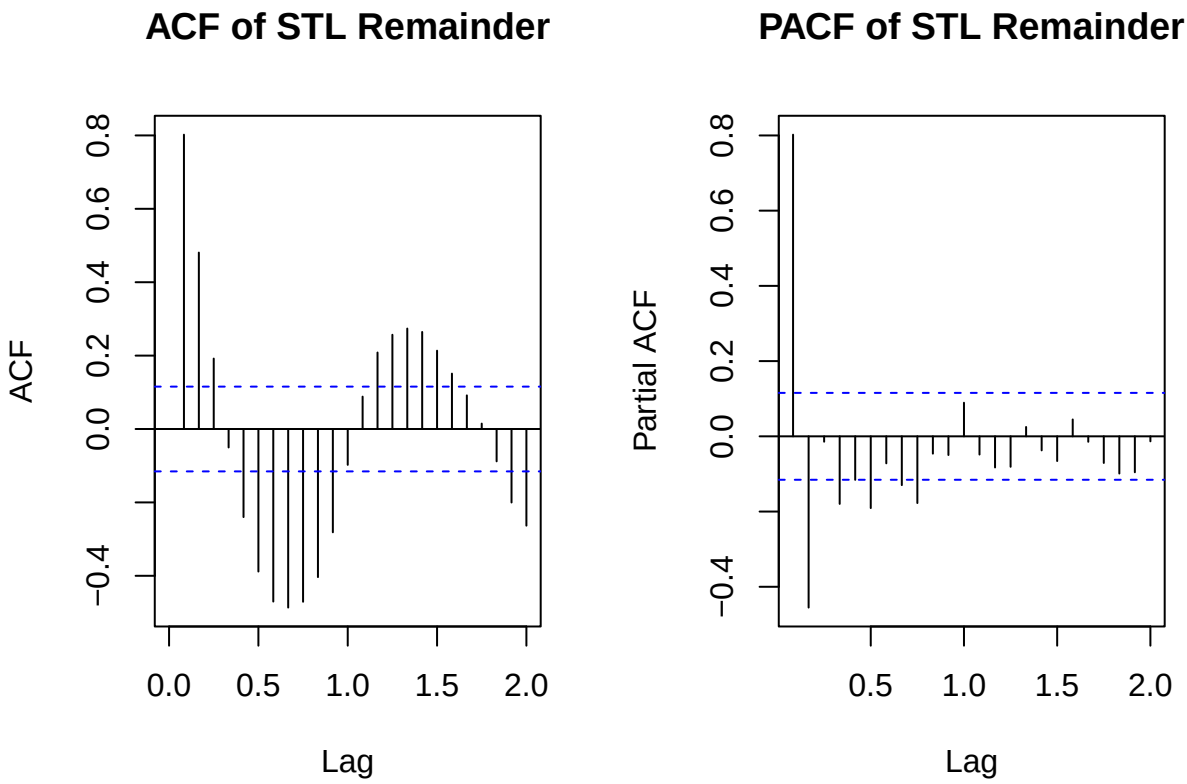
```
par(mfrow = c(1, 2))
seasonal_acf_plot_nino <- acf(decomposed_nino$time.series[, "seasonal"], plot = FALSE)
seasonal_acf_plot_nino$acf[1, 1, 1] <- 0
plot(seasonal_acf_plot_nino, main = "ACF of STL Seasonal")
pacf(decomposed_nino$time.series[, "seasonal"], main = "PACF of STL Seasonal")
```



Looking at the plots of the ACF and PACF of the STL seasonal component, we see that there are many significant lags. It is difficult to know what type of S-ARMA model to fit to the data, but we see that the first two lags on the PACF are significant, while those on the ACF appear to be related to some sort of cycle in the remainder component. Ideally, an S-ARMA(2,0) model should be enough to capture most of the dynamics present in the data.

We now carry out the same procedure for the remainder component, to approximate cycles:

```
par(mfrow = c(1, 2))
remainder_acf_plot_nino <- acf(decomposed_nino$time.series[, "remainder"], plot = FALSE)
remainder_acf_plot_nino$acf[1, 1, 1] <- 0
plot(remainder_acf_plot_nino, main = "ACF of STL Remainder")
pacf(decomposed_nino$time.series[, "remainder"], main = "PACF of STL Remainder")
```



This looks simpler. Looking at the PACF plot, we see that the first two lags are highly significant, while the fourth, sixth, and eighth may be as well. The ACF plot has a sinusoidal look to it, which may be partially due to a seasonal component coming into play; however the first spike seems taller than would be expected so we think there may be some MA dynamics involved as well. The rest of the ACF plots seem to be decaying consistently, so we will model this with a ARMA(8, 1) model.

We now build our ARIMA(8, 1, 1)(2, 0) model:

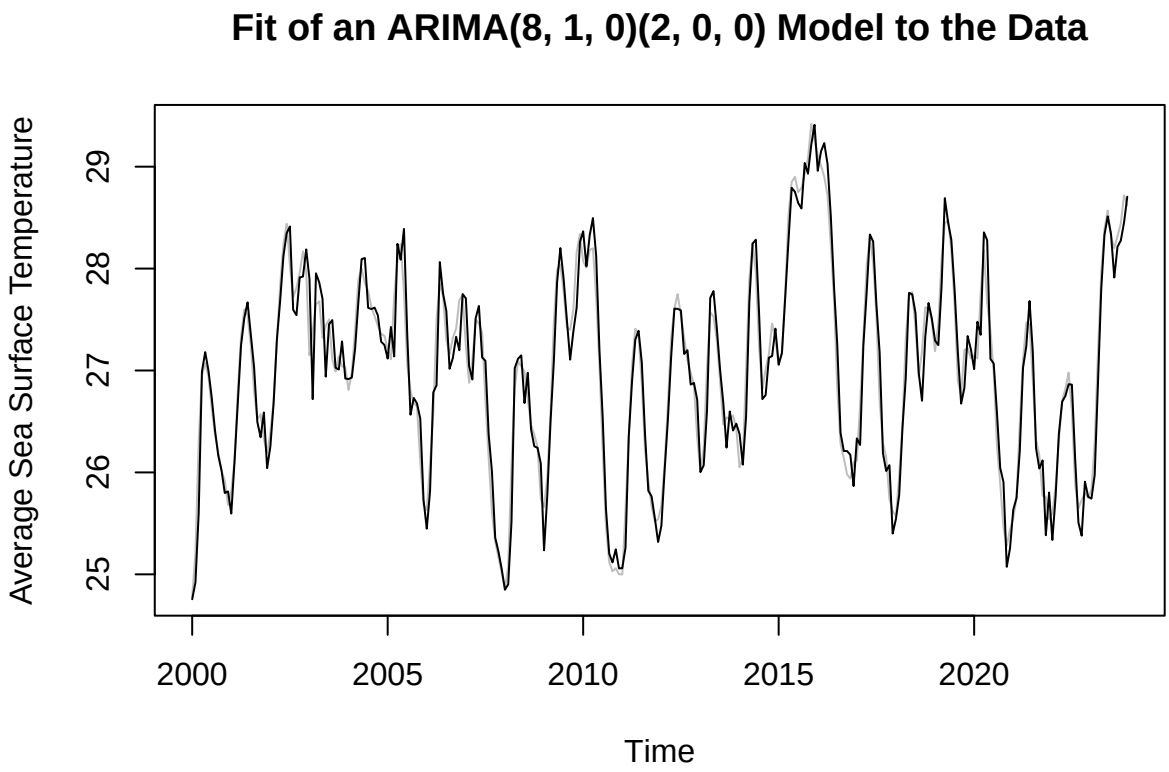
```
manual_arima_model_nino <- arima(elnino, order = c(8, 1, 0),
                                 seasonal = list(order = c(2, 0, 0)))
summary(manual_arima_model_nino)
```

```
##
## Call:
## arima(x = elnino, order = c(8, 1, 0), seasonal = list(order = c(2, 0, 0)))
##
## Coefficients:
##          ar1      ar2     ar3      ar4     ar5      ar6      ar7      ar8
##       0.6923  -0.1948  0.0387  -0.0759  0.0807  -0.0971  -0.0367  -0.1116
## s.e.  0.0629   0.0767  0.0754   0.0759  0.0731   0.0729   0.0715   0.0605
##         sar1    sar2
##       0.3301  0.2239
## s.e.  0.0663  0.0654
##
## sigma^2 estimated as 0.05463:  log likelihood = 7.68,  aic = 6.64
##
## Training set error measures:
##                       ME      RMSE       MAE        MPE      MAPE      MASE
## Training set 0.006913682 0.2333376 0.1842115 0.02501128 0.6822468 0.5968469
##                      ACF1
```

```
## Training set -0.01263946
```

Most of the lags in this model appear highly significant. We plot the model:

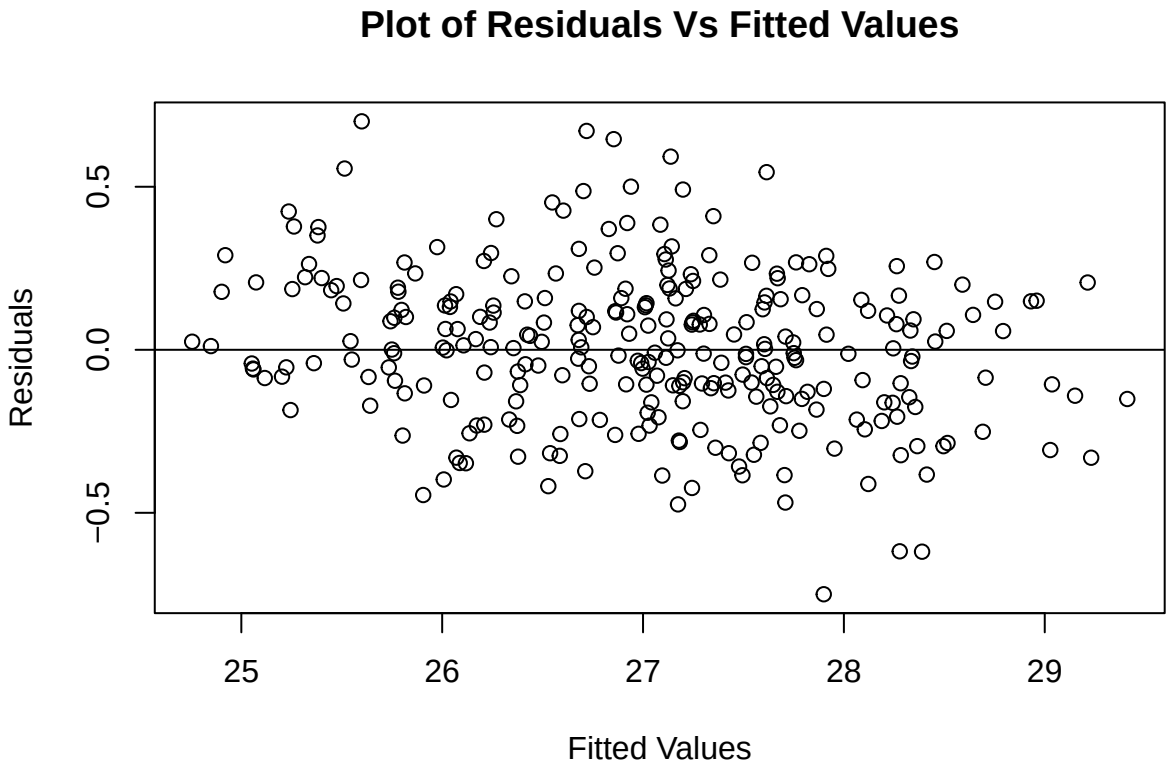
```
plot(elnino, col = "gray", main = "Fit of an ARIMA(8, 1, 0)(2, 0, 0) Model to the Data",
     ylab = "Average Sea Surface Temperature")
lines(fitted(manual_arima_model_nino), col = "black")
```



We see that the ARIMA model is almost a perfect fit to the data.

(e)

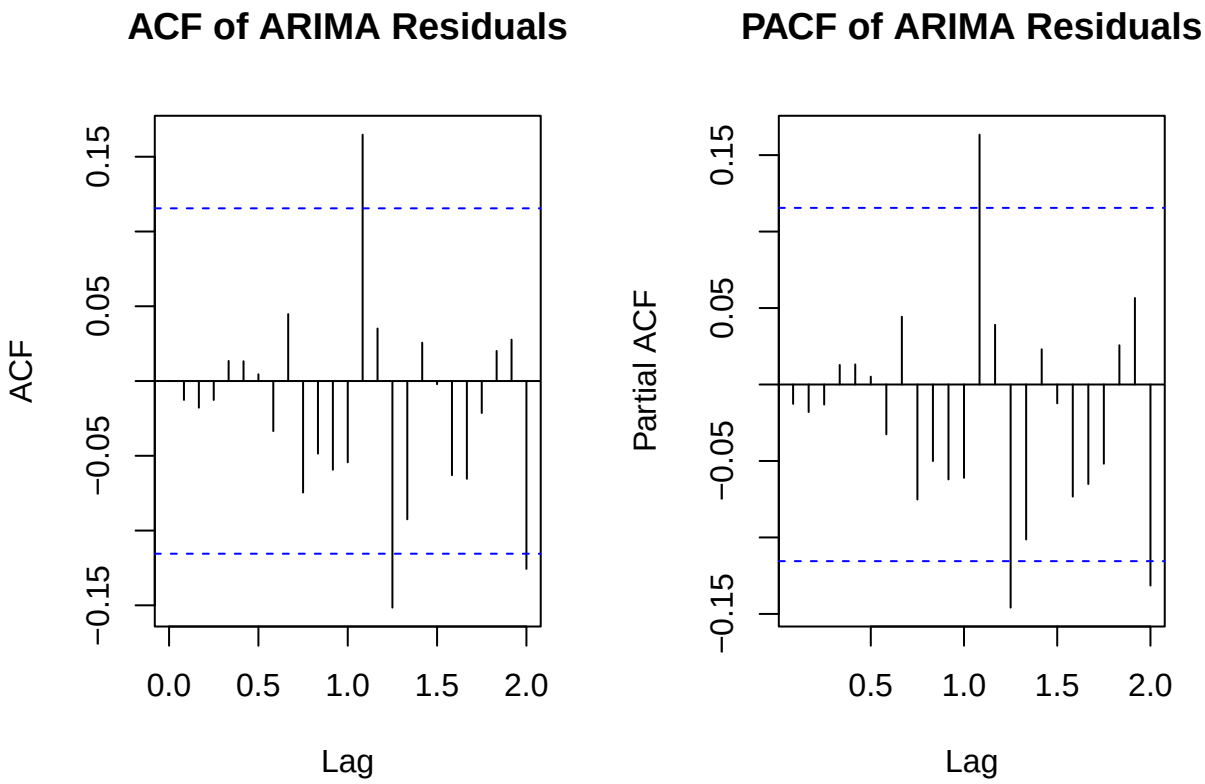
```
plot(fitted(manual_arima_model_nino), manual_arima_model_nino$resid,
     main = "Plot of Residuals Vs Fitted Values", ylab = "Residuals", xlab = "Fitted Values")
abline(h = 0, col = "black")
```



We see that the residuals appear to be randomly scattered with constant variance about a mean of 0, indicating that the assumption that the error is distributed as $WN(0, \sigma^2)$ is satisfied.

(f)

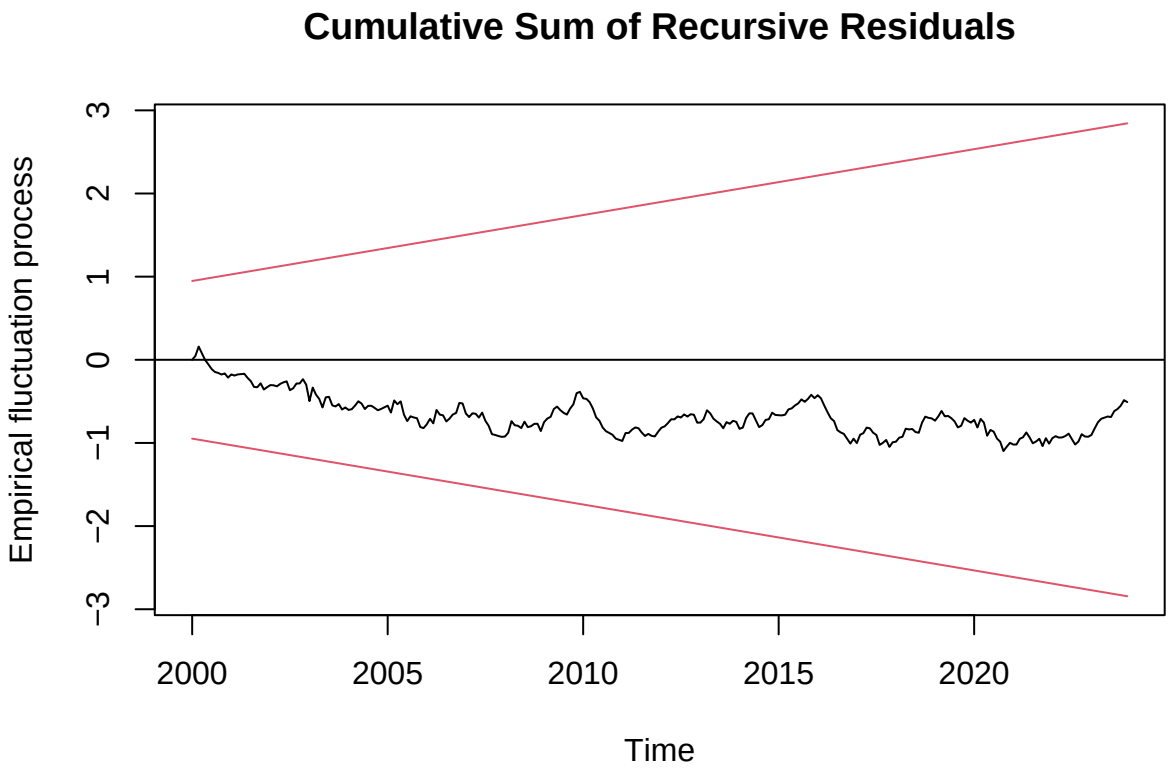
```
par(mfrow = c(1, 2))
resid_acf_plot_nino <- acf(manual_arima_model_nino$residuals, plot = FALSE)
resid_acf_plot_nino$acf[1, 1, 1] <- 0
plot(resid_acf_plot_nino, main = "ACF of ARIMA Residuals")
pacf(manual_arima_model_nino$residuals, main = "PACF of ARIMA Residuals")
```



We see that the model has missed a couple of slightly significant lags higher up in the PACF and ACF of the residuals. This indicates that the model does not account for all of the dynamics in the data, but we keep the model as is, as they seem to be just significant. We will see what the auto ARIMA model suggests and see how its plots look like.

(g)

```
plot(efp(manual_arima_model_nino$residuals ~ 1, type = "Rec-CUSUM"),
     main = "Cumulative Sum of Recursive Residuals")
```



Examining the cumulative sum of the recursive residuals, we see that the model is robust, with the cumulative sum of the residuals indicating that any shocks to the data never cause the model's parameters to exceed the confidence bands. This indicates that the model is robust.

(h)

We will examine a few of the accuracy statistics of the model to determine how good of a fit it is to the data.

```
accuracy(manual_arima_model_nino)
```

```
##                       ME      RMSE       MAE        MPE      MAPE      MASE
## Training set 0.006913682 0.2333376 0.1842115 0.02501128 0.6822468 0.5968469
##                      ACF1
## Training set -0.01263946
```

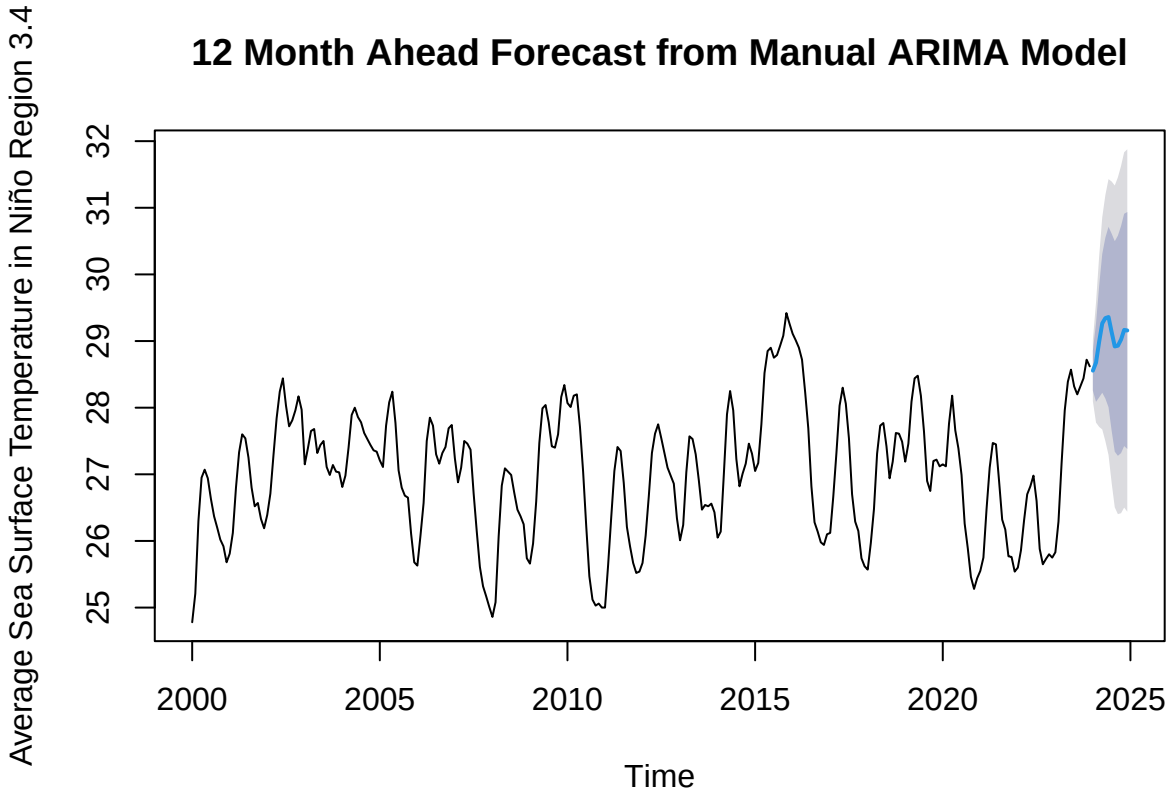
There are a few interesting and easy to interpret statistics in here.

The RMSE indicates that the model's predictions of average temperature deviations, on average, are 0.23°C from the true value. This is not great, as indicated by the MAPE, this is a deviation of about 68% from the true values.

The other statistics say similar things or are not very useful in the context of the data, but overall, they indicate that the model does not provide a very strong fit to the data.

(i)

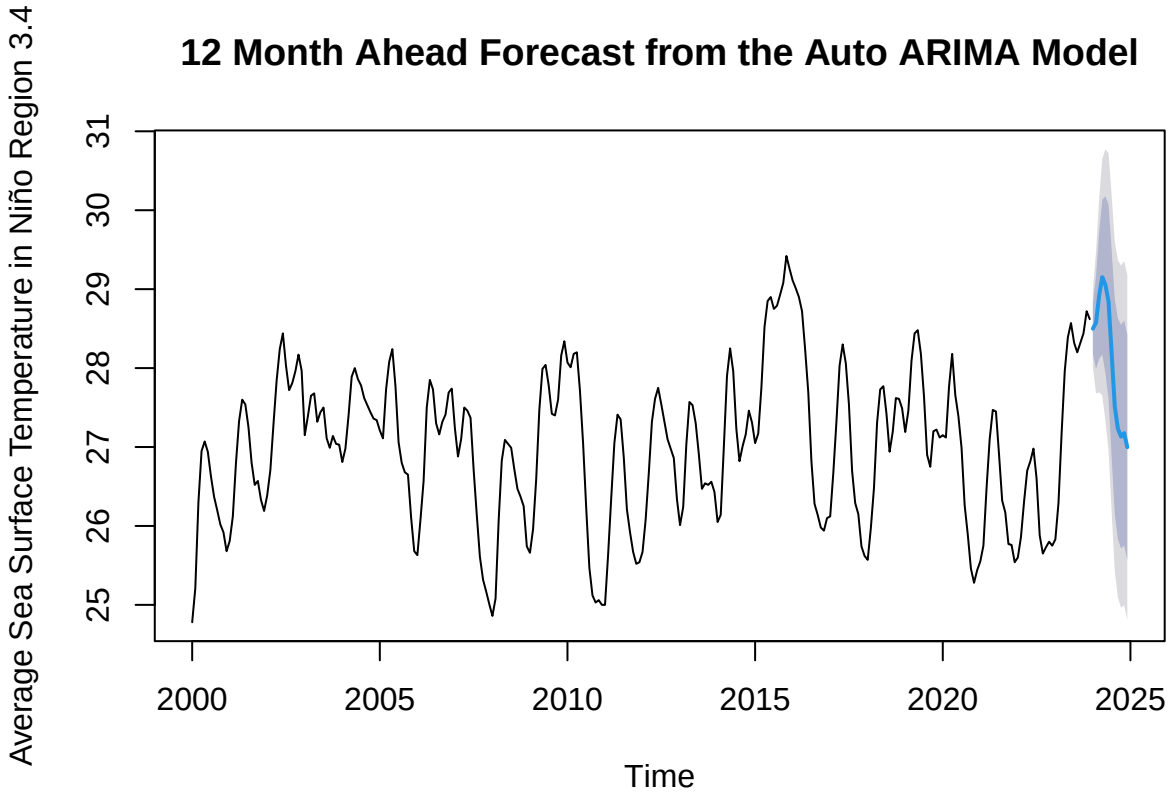
```
manual_forecast_nino <- forecast(manual_arima_model_nino, h = 12)
plot(manual_forecast_nino, ylab = "Average Sea Surface Temperature in Niño Region 3.4",
     xlab = "Time", main = "12 Month Ahead Forecast from Manual ARIMA Model")
```



We see that the forecast from the manual ARIMA model expects the average average SST to increase over the next 12 months, although it is not very certain about this.

(j)

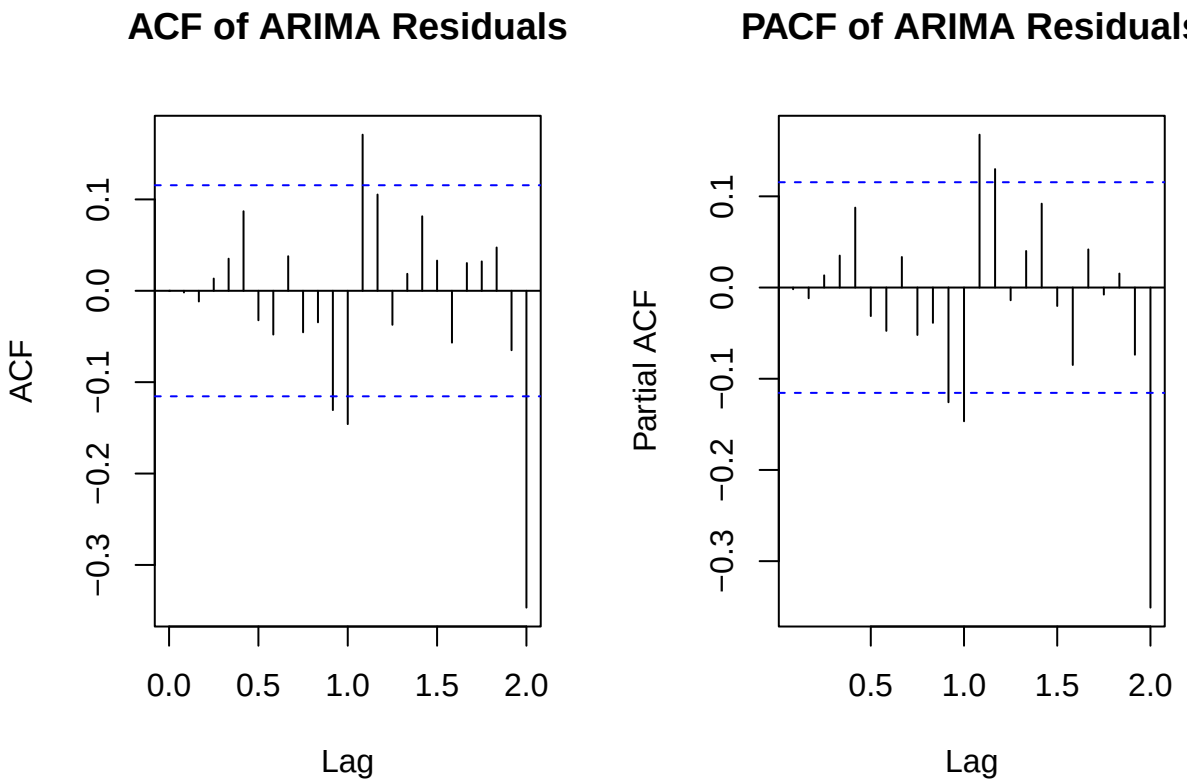
```
auto_arima_model_nino <- auto.arima(elnino)
auto_arima_forecast_nino <- forecast(auto_arima_model_nino, h = 12)
plot(auto_arima_forecast_nino, main = "12 Month Ahead Forecast from the Auto ARIMA Model",
     ylab = "Average Sea Surface Temperature in Niño Region 3.4", xlab = "Time")
```



The auto ARIMA model expects an increase followed by a decline in the average SST global surface temperature. It is more certain about the prediction, but also indicates that there is 80% confidence for an increase.

We quickly glance at the auto ARIMA model's plots:

```
par(mfrow = c(1, 2))
resid_acf_plot_nino_auto <- acf(auto_arima_model_nino$residuals, plot = FALSE)
resid_acf_plot_nino_auto$acf[1, 1, 1] <- 0
plot(resid_acf_plot_nino_auto, main = "ACF of ARIMA Residuals")
pacf(auto_arima_model_nino$residuals, main = "PACF of ARIMA Residuals")
```



We see that the auto ARIMA also struggled with this data.

We now compute the MAPE for the auto ARIMA model:

```
cat("Model          MAPE", "\nARIMA      ", accuracy(manual_arima_model_nino)[5],
    "\nAuto ARIMA ", accuracy(auto_arima_model_nino)[5])
```

```
## Model          MAPE
## ARIMA       0.6822468
## Auto ARIMA  0.6882924
```

```
cat("Model       Predictors", "\nARIMA      ", length(manual_arima_model_nino$coef),
    "\nAuto ARIMA ", length(auto_arima_model_nino$coef))
```

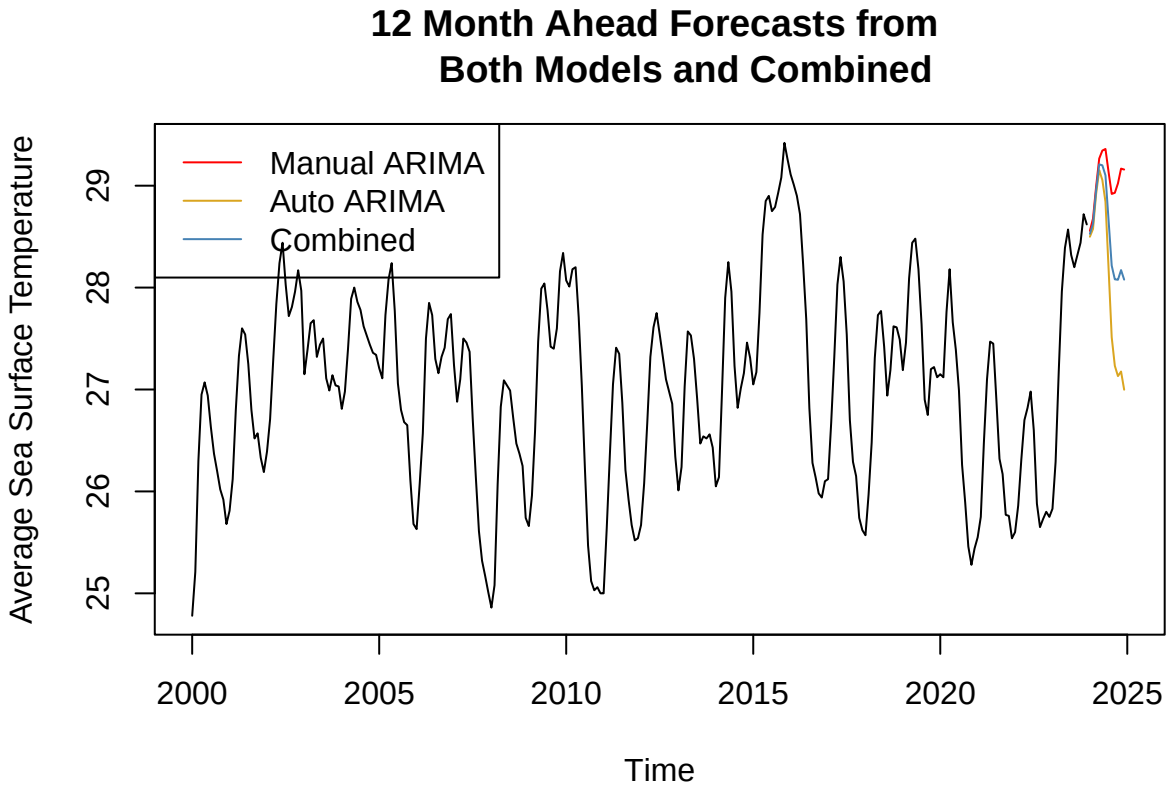
```
## Model       Predictors
## ARIMA       10
## Auto ARIMA  6
```

We see that the MAPE of the auto ARIMA model is slightly higher than that from the ARIMA model defined in part (c). However, it is also less complex, with 6 instead of 10 terms, so it is preferable, even if a lower MAPE is better.

(k)

```
combined_forecast_nino <- (auto_arima_forecast_nino$mean + manual_forecast_nino$mean) / 2
plot(elnino, xlim = c(2000, 2025), main = "12 Month Ahead Forecasts from
     Both Models and Combined", ylab = "Average Sea Surface Temperature", xlab = "Time")
lines(seq(2024, 2025 - 1/12, by = 1/12), manual_forecast_nino$mean, col = "red")
lines(seq(2024, 2025 - 1/12, by = 1/12), auto_arima_forecast_nino$mean, col = "goldenrod")
lines(seq(2024, 2025 - 1/12, by = 1/12), combined_forecast_nino, col = "steelblue")
legend("topleft", legend = c("Manual ARIMA", "Auto ARIMA", "Combined"),
```

```
       lty = c(1, 1, 1), col = c("red", "goldenrod", "steelblue"))
```



The MAPE of the combined forecast will be the average of the MAPE of the two forecasts, as this forecast is merely the average of the other two forecasts.

Exploring the Dynamics of the Models

(l)

We build the model by first running `VARselect`:

```
data_frame <- cbind(surf_temp, elnino)
VARselect(data_frame, lag.max = 10)
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##     10      3      3     10
##
## $criteria
##                  1             2             3             4             5
## AIC(n) -6.207123996 -7.0394533158 -7.1057002580 -7.1237976288 -7.1173664086
## HQ(n)  -6.175713055 -6.9871017464 -7.0324080608 -7.0295648037 -7.0021929558
## SC(n)  -6.128830015 -6.9089633477 -6.9230143027 -6.8889156861 -6.8302884787
## FPE(n)  0.002015028  0.0008766125  0.0008204325  0.0008057377  0.0008109666
##                  6             7             8             9            10
## AIC(n) -7.1053133711 -7.1184395430 -7.1553976897 -7.19762578 -7.2147653307
## HQ(n)  -6.9691992905 -6.9613848347 -6.9774023536 -6.99868982 -6.9948887390
## SC(n)  -6.7660394539 -6.7269696387 -6.7117317981 -6.70176390 -6.6667074646
## FPE(n)  0.0008208445  0.0008101998  0.0007808776  0.00074868  0.0007360676
```

Note that 2 of the lags are at 10. We skip the lag max to 30 to see what is suggested:

```r
VARselect(data_frame, lag.max = 30)
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##     16     13      3     16
##
## $criteria
##                  1             2            3             4             5
## AIC(n) -6.191743143 -6.9870360776 -7.051819712 -7.0644138853 -7.0575654900
## HQ(n)  -6.158518443 -6.9316615780 -6.974295413 -6.9647397861 -6.9357415910
## SC(n)  -6.109116176 -6.8493244658 -6.859023456 -6.8165329840 -6.7545999440
## FPE(n)  0.002046261  0.0009237895  0.000865855  0.0008550443  0.0008609604
##                  6             7             8             9            10
## AIC(n) -7.0430638528 -7.0564019156 -7.0882123191 -7.1313267535 -7.146327049
## HQ(n)  -6.8990901540 -6.8902784169 -6.8999390206 -6.9209036552 -6.913754150
## SC(n)  -6.6850136621 -6.6432670801 -6.6199928389 -6.6080226286 -6.567938279
## FPE(n)  0.0008735956  0.0008620999  0.0008352079  0.0008000851  0.000788321
##                 11            12            13            14            15
## AIC(n) -7.1621466238 -7.2271724146 -7.2936705771 -7.2802631544 -7.282438408
## HQ(n)  -6.9074239259 -6.9502999169 -6.9946482796 -6.9590910570 -6.939116511
## SC(n)  -6.5286732094 -6.5386143555 -6.5500278733 -6.4815358059 -6.428626415
## FPE(n)  0.0007761246  0.0007274594  0.0006808747  0.0006903201  0.000689113
##                 16            17            18            19            20
## AIC(n) -7.3312710345 -7.3037702170 -7.2869512092 -7.3066523303 -7.284684496
## HQ(n)  -6.9657993375 -6.9161487201 -6.8771799125 -6.8747312338 -6.830613600
## SC(n)  -6.4223743965 -6.3397889342 -6.2678852817 -6.2325017581 -6.155449279
## FPE(n)  0.0006565884  0.0006752658  0.0006871422  0.0006742007  0.000689702
##                 21            22            23            24            25
## AIC(n) -7.2881558126 -7.2714432302 -7.2532055589 -7.2719862687 -7.2515615883
## HQ(n)  -6.8119351164 -6.7730727343 -6.7326852632 -6.7293161731 -6.6867416929
## SC(n)  -6.1038359509 -6.0320387239 -5.9587164078 -5.9224124729 -5.8469031478
## FPE(n)  0.0006878925  0.0007001357  0.0007137481  0.0007012478  0.0007165846
##                 26            27            28            29            30
## AIC(n) -7.2347757164 -7.2183482661 -7.1946106209 -7.2171984435 -7.228838002
## HQ(n)  -6.6478060212 -6.6092287711 -6.5633413261 -6.5637793488 -6.553269108
## SC(n)  -5.7750326311 -5.7035205361 -5.6246982462 -5.5922014240 -5.548756338
## FPE(n)  0.0007296721  0.0007428124  0.0007618232  0.0007460386  0.000738714
```

We see that the suggest model is of order 16, which makes sense seeing as this data is highly complex.

```r
var_model <- VAR(data_frame, p = 16)
summary(var_model)
```

```
##
## VAR Estimation Results:
## ========================
## Endogenous variables: surf_temp, elnino
## Deterministic variables: const
## Sample size: 272
## Log Likelihood: 296.64
## Roots of the characteristic polynomial:
## 0.9958 0.9876 0.9876 0.983 0.983 0.917 0.917 0.909 0.909 0.9083 0.9083 0.9005 0.9005 0.8997 0.8997 0
## Call:
## VAR(y = data_frame, p = 16)
##
```

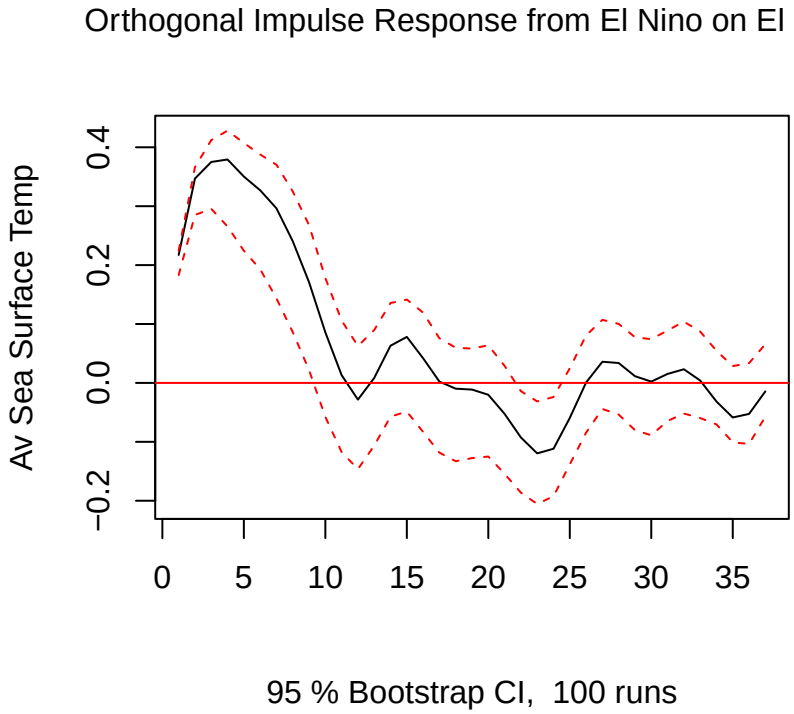
```
##
## Estimation results for equation surf_temp:
## ==========================================
## surf_temp = surf_temp.l1 + elnino.l1 + surf_temp.l2 + elnino.l2 + surf_temp.l3 + elnino.l3 + surf_ter
##
##                 Estimate Std. Error t value Pr(>|t|)
## surf_temp.l1   0.4304769  0.0648086   6.642 2.06e-10 ***
## elnino.l1      0.0836565  0.0297903   2.808 0.005394 **
## surf_temp.l2   0.2682164  0.0705403   3.802 0.000182 ***
## elnino.l2     -0.0698448  0.0557263  -1.253 0.211301
## surf_temp.l3  -0.1133939  0.0715854  -1.584 0.114508
## elnino.l3     -0.0237361  0.0605633  -0.392 0.695465
## surf_temp.l4   0.1572999  0.0726584   2.165 0.031384 *
## elnino.l4      0.1075494  0.0604098   1.780 0.076292 .
## surf_temp.l5  -0.0112220  0.0729010  -0.154 0.877792
## elnino.l5     -0.0818449  0.0605449  -1.352 0.177716
## surf_temp.l6  -0.0432508  0.0727853  -0.594 0.552925
## elnino.l6      0.0647338  0.0611711   1.058 0.291014
## surf_temp.l7   0.0240750  0.0725987   0.332 0.740468
## elnino.l7     -0.1030333  0.0614117  -1.678 0.094704 .
## surf_temp.l8   0.0463268  0.0732854   0.632 0.527898
## elnino.l8      0.0737183  0.0618679   1.192 0.234622
## surf_temp.l9   0.0459760  0.0726768   0.633 0.527593
## elnino.l9      0.0097455  0.0620703   0.157 0.875371
## surf_temp.l10  0.0787279  0.0739217   1.065 0.287943
## elnino.l10    -0.0447994  0.0619846  -0.723 0.470539
## surf_temp.l11  0.0752217  0.0740753   1.015 0.310906
## elnino.l11     0.0471696  0.0619046   0.762 0.446828
## surf_temp.l12  0.0132756  0.0738186   0.180 0.857430
## elnino.l12    -0.0003045  0.0616175  -0.005 0.996062
## surf_temp.l13  0.0340501  0.0726739   0.469 0.639831
## elnino.l13    -0.0918351  0.0616756  -1.489 0.137805
## surf_temp.l14  0.1458271  0.0725620   2.010 0.045588 *
## elnino.l14    -0.0169723  0.0617718  -0.275 0.783740
## surf_temp.l15 -0.0405464  0.0709911  -0.571 0.568437
## elnino.l15     0.0973573  0.0566657   1.718 0.087073 .
## surf_temp.l16 -0.1250678  0.0645794  -1.937 0.053967 .
## elnino.l16    -0.0432866  0.0306237  -1.413 0.158810
## const         -0.2034240  0.3469007  -0.586 0.558157
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.1031 on 239 degrees of freedom
## Multiple R-Squared: 0.7827,  Adjusted R-squared: 0.7536
## F-statistic:  26.9 on 32 and 239 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation elnino:
## =======================================
## elnino = surf_temp.l1 + elnino.l1 + surf_temp.l2 + elnino.l2 + surf_temp.l3 + elnino.l3 + surf_temp.l
##
##                 Estimate Std. Error t value Pr(>|t|)
## surf_temp.l1  -0.055121   0.138537  -0.398  0.69107
```

```
## elnino.l1      1.597579   0.063681  25.087  < 2e-16 ***
## surf_temp.l2   0.029001   0.150789   0.192  0.84765
## elnino.l2     -0.820599   0.119122  -6.889 4.95e-11 ***
## surf_temp.l3  -0.014161   0.153023  -0.093  0.92635
## elnino.l3      0.301465   0.129462   2.329  0.02072 *
## surf_temp.l4   0.052868   0.155317   0.340  0.73386
## elnino.l4     -0.239047   0.129134  -1.851  0.06538 .
## surf_temp.l5   0.095600   0.155835   0.613  0.54015
## elnino.l5      0.225842   0.129423   1.745  0.08227 .
## surf_temp.l6   0.068939   0.155588   0.443  0.65811
## elnino.l6     -0.199964   0.130761  -1.529  0.12753
## surf_temp.l7  -0.131813   0.155189  -0.849  0.39653
## elnino.l7      0.013522   0.131275   0.103  0.91805
## surf_temp.l8  -0.245678   0.156657  -1.568  0.11814
## elnino.l8     -0.008792   0.132251  -0.066  0.94705
## surf_temp.l9  -0.064902   0.155356  -0.418  0.67650
## elnino.l9     -0.016129   0.132683  -0.122  0.90335
## surf_temp.l10  0.036895   0.158017   0.233  0.81559
## elnino.l10     0.078320   0.132500   0.591  0.55502
## surf_temp.l11  0.207358   0.158345   1.310  0.19161
## elnino.l11     0.022157   0.132329   0.167  0.86717
## surf_temp.l12  0.196400   0.157797   1.245  0.21448
## elnino.l12     0.250929   0.131715   1.905  0.05797 .
## surf_temp.l13 -0.009565   0.155350  -0.062  0.95096
## elnino.l13    -0.304838   0.131839  -2.312  0.02162 *
## surf_temp.l14 -0.034428   0.155111  -0.222  0.82453
## elnino.l14    -0.030353   0.132045  -0.230  0.81839
## surf_temp.l15  0.267981   0.151753   1.766  0.07869 .
## elnino.l15    -0.127450   0.121130  -1.052  0.29378
## surf_temp.l16 -0.410413   0.138047  -2.973  0.00325 **
## elnino.l16     0.173855   0.065462   2.656  0.00844 **
## const          2.268114   0.741545   3.059  0.00248 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.2204 on 239 degrees of freedom
## Multiple R-Squared: 0.9539,  Adjusted R-squared: 0.9477
## F-statistic: 154.6 on 32 and 239 DF,  p-value: < 2.2e-16
##
##
##
## Covariance matrix of residuals:
##           surf_temp   elnino
## surf_temp  0.010630 0.003878
## elnino     0.003878 0.048575
##
## Correlation matrix of residuals:
##           surf_temp elnino
## surf_temp    1.0000 0.1707
## elnino       0.1707 1.0000
```

We see that lags on the first couple orders of each are significant explanatory variables for both series, while the highest order lags of surface temperature seem to explain the Average SST.

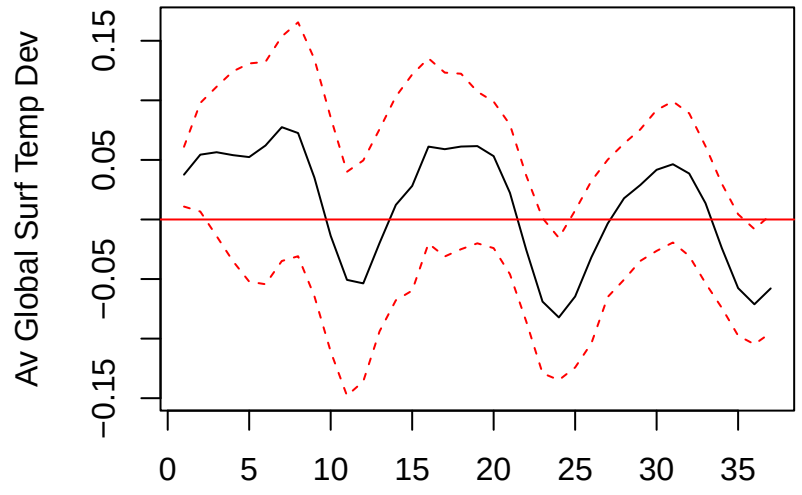
(m)

```
plot(irf(var_model, n.ahead = 36, impulse = "elnino", response = "elnino"),
     main = "Orthogonal Impulse Response from El Nino on El Nino",
     ylab = "Av Sea Surface Temp")
```




```
plot(irf(var_model, n.ahead = 36, impulse = "surf_temp", response = "elnino"),
     main = "Orthogonal Impulse Response from El Nino on Surface Temperature",
     ylab = "Av Global Surf Temp Dev")
```

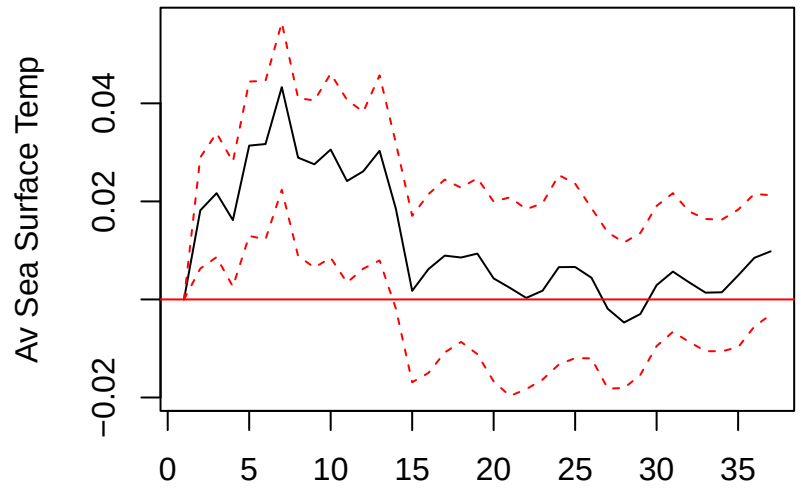
Orthogonal Impulse Response from El Nino on Surface Temperature



95 % Bootstrap CI, 100 runs

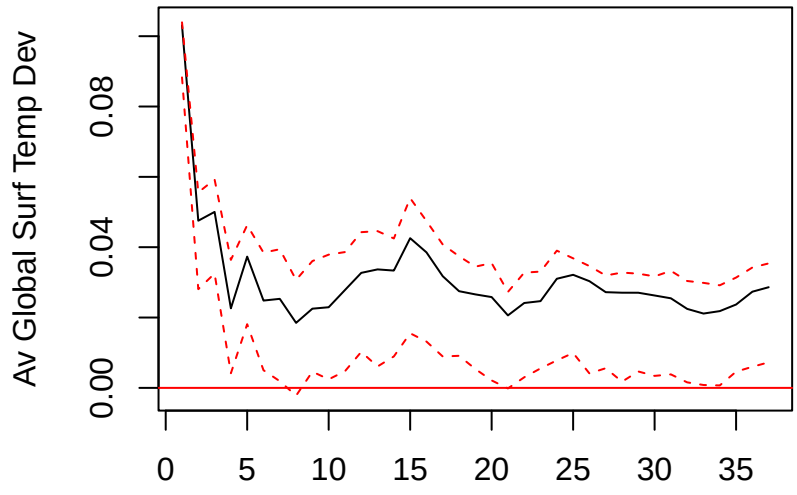
```
plot(irf(var_model, n.ahead = 36, impulse = "elnino", response = "surf_temp"),
     main = "Orthogonal Impulse Response from Surface Temperature on El Nino",
     ylab = "Av Sea Surface Temp")
```

Orthogonal Impulse Response from Surface Temperature on El Nino



95 % Bootstrap CI, 100 runs

```
plot(irf(var_model, n.ahead = 36, impulse = "surf_temp", response = "surf_temp"),
     main = "Orthogonal Impulse Response from Surface Temperature on Surface Temperature",
     ylab = "Av Global Surf Temp Dev")
```



We see that a unit shock of average SST on itself leads to a spike in average SST over the region after about 6 months, followed by a decline to an insignificant level by 10 months. A unit shock of average SST appears to have an insignificant effect on average global surface temperature. A unit shock of average global surface temperature appears to cause an increase in the average SST until a spike about 8 months in, which is followed by a decline to insignificance after about a year. A unit shock of average global surface temperature on itself appears to lead to a quick decline to insignificance after about 8 months or so.

(n)

```
grangertest(surf_temp ~ elnino, order = 16)
```

```
## Granger causality test
##
## Model 1: surf_temp ~ Lags(surf_temp, 1:16) + Lags(elnino, 1:16)
## Model 2: surf_temp ~ Lags(surf_temp, 1:16)
##   Res.Df  Df      F    Pr(>F)
## 1    239
## 2    255 -16 3.6691 5.196e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
grangertest(elnino ~ surf_temp, order = 16)
```

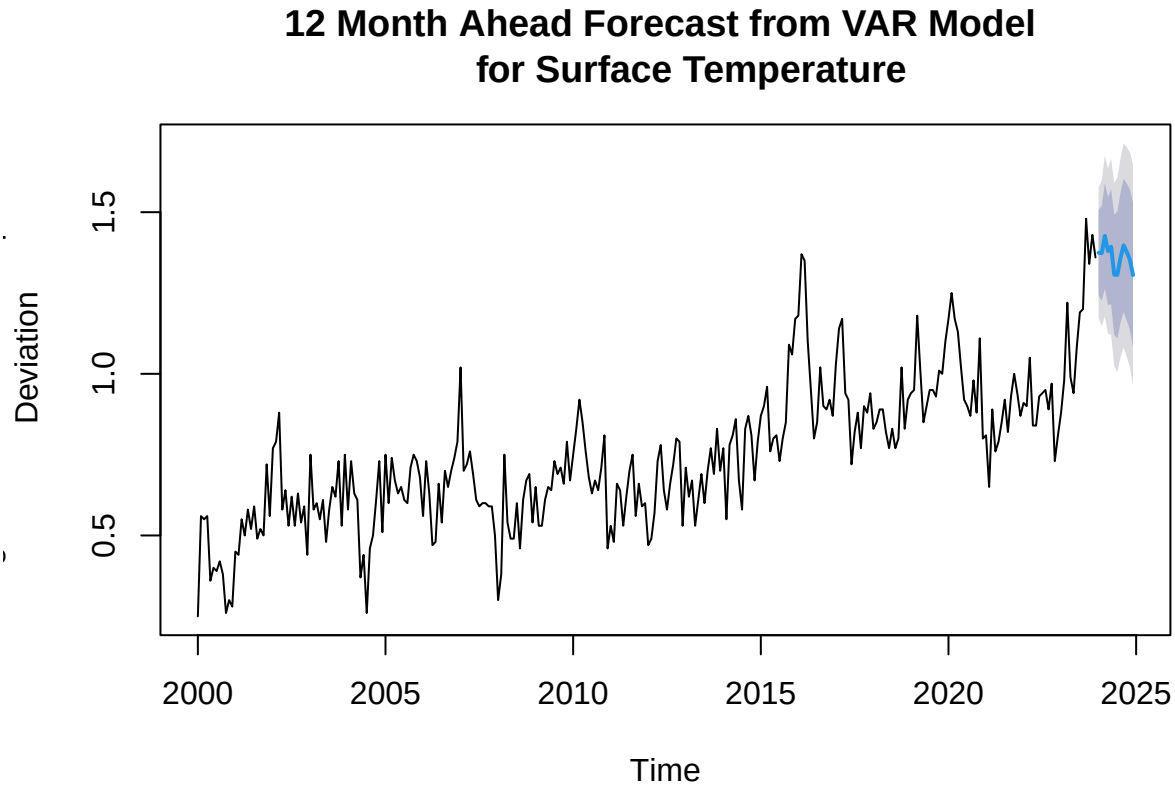
```
## Granger causality test
##
## Model 1: elnino ~ Lags(elnino, 1:16) + Lags(surf_temp, 1:16)
## Model 2: elnino ~ Lags(elnino, 1:16)
##   Res.Df  Df     F Pr(>F)
## 1    239
## 2    255 -16 1.373 0.1558
```

The test shows that the average SST (from the El Niño data) appears to Granger cause deviations in the

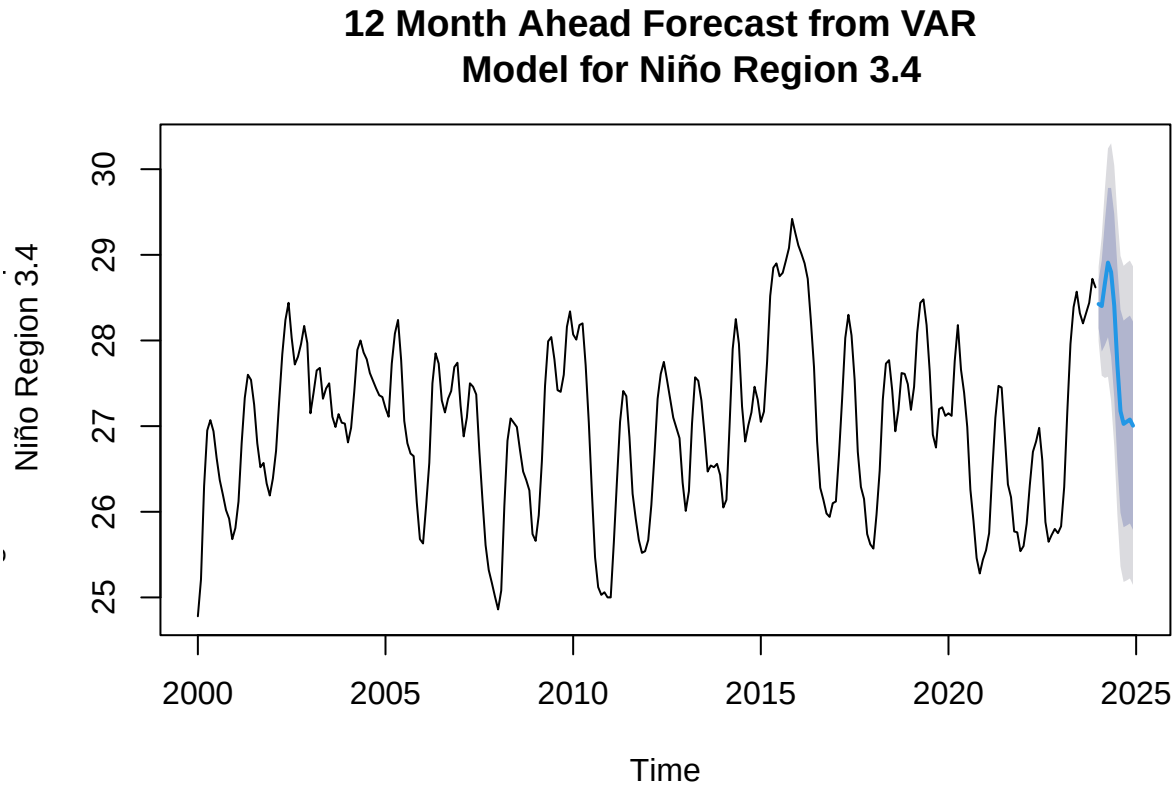
average global surface temperature at a significance level of 5%, but the converse is not true. This makes sense, as the El Niño phenomenon is known to be linked to warming cycles, so we would expect an indicator of El Niño's increase to be linked to an increase down the line in surface temperatures.

(o)

```
var_forecast <- forecast(var_model, h = 12)
plot(var_forecast$forecast$surf_temp, ylab = "Average Global Surface Temperature
     Deviation", xlab = "Time", main = "12 Month Ahead Forecast from VAR Model
     for Surface Temperature")
```




```
plot(var_forecast$forecast$elnino, ylab = "Average Sea Surface Temperature in
     Niño Region 3.4", xlab = "Time", main = "12 Month Ahead Forecast from VAR
     Model for Niño Region 3.4")
```



The VAR forecast for the surface temperature has a slightly wider estimated range than the auto ARIMA one but looks very similar to it. This makes sense, as it runs an AR model under the hood that takes into account some of the dynamics present in the average SST data. This is good, as we have determined that the average SST information should Granger cause the global surface temperature.

The Niño Region 3.4 model also does something similar, where the forecast seems to be a more flattened version of the auto ARIMA forecast. This also makes sense, as the VAR model for it is similar under the hood but takes into account the data from the surface temperature model.

III. Conclusions and Future Work

We have managed to make a very good model for the average deviation in surface temperature, although the El Niño average SST data was harder to predict. We can conclude from the Granger causality tests and the IRF plot of the response of the average deviation in surface temperature to a unit shock in the average SST for Niño region 3.4 that average SST change may cause, at least to an extent, a proportional change in the average global surface temperature, although the converse is not true.

In the future, we can further refine the models, especially the one for the average SST, to account for more dynamics, as we saw in the ACF and PACF of the residuals of all of the models for that data set that dynamics remained present. Some sort of ETS model could come in handy here.

IV. References

For the El Niño SST data:

Trenberth, Kevin & National Center for Atmospheric Research Staff (Eds). Last modified 2024-03-20 “The Climate Data Guide: Nino SST Indices (Nino 1+2, 3, 3.4, 4; ONI and TNI).” Retrieved from

https://climatedataguide.ucar.edu/climate-data/nino-sst-indices-nino-12-3-34-4-oni-and-tni on 2024-05-16.

- Data set is data set 1: from NOAA/PSL, Nino 3.4, https://psl.noaa.gov/data/correlation/nina34.data

For the average monthly global temperature deviation data:

Goddard Institute for Space Studies. "GISS Surface Temperature Analysis (GISTEMP v4)". NASA. Retrieved from: https://data.giss.nasa.gov/gistemp/ on 2024-05-16.

- Description of data: Hansen, J., et al.. "Global Surface Temperature Change". Reviews of Geophysics, 48, RG4004 / 2010, American Geophysical Union.
  https://pubs.giss.nasa.gov/docs/2010/2010_Hansen_ha00510u.pdf
- Data: https://data.giss.nasa.gov/gistemp/tabledata_v4/GLB.Ts+dSST.csv