Average Monthly Temperatures (°F) at Nottingham Castle Time Series Final Project

Elizabeth Forney

Fall 2021

Abstract

In this paper I address the problem of if is possible to forecast average monthly temperatures (°F) at Nottingham Castle in England using past data. My goal is to use the Box-Jenkins methodology to build a seasonal autoregressive moving average (SARIMA) model and then used this model to forecast 12 future data points. After making the original data stationary and examining several models identified based on the autocorrelation function (ACF) and partial autocorrelation function (PACF) graphs, I identified two models that had both a low Akaike Information Criterion, Corrected for Bias (AICc) and were stationary and invertible. Both of these models passed diagnostic checking, so I used the Principle of Parsimony to choose the model with the lowest coefficients to be my final model used for forecasting. The original data fell within my prediction interval, supporting the validity of my final model and suggesting that it is possible to forecast the average monthly temperature at Nottingham.

Introduction

The dataset I used is average monthly temperatures (°F) at Nottingham Castle in England from 1920 through 1939. This data is interesting because we can see if there is any trend or pattern in the temperature over time. Being able to forecast future monthly temperatures here could help staff and visitors prepare accordingly for their visit. It could also potentially give climate scientists a snapshot of how temperature is changing in the area over time. I obtained this dataset from a collection of R datasets maintained by Vincent Arel-Bundock on GitHub. The particular dataset I used from this collection is sourced from *Time Series Analysis and Forecasting: The Box-Jenkins Approach* (1976) by Oliver Duncan Anderson. Furthermore, I have used the software R for my coding and statistical computing.

I applied the Box-Jenkins to build a SARIMA model for the average monthly temperatures (°F) at Nottingham Castle and forecast 12 future data points. I first partitioned the original dataset into a training dataset (228 observations from 1920 through 1938) used for building a SARIMA model and a testing dataset (12 observations in the year 1939) to check the validity of my final model. To begin building my model I first visualized the original data to identify any key features such as trend, seasonality, and sharp changes in behavior. I then examined the ACF and PACF graphs to preliminarily identify appropriate SARIMA$(p,d,q)\times(P,D,Q)_s$ models. In order to narrow down the number of candidate models, I computed the AICc for each model. I chose the six models that produced the lowest AICc because a lower AICc indicates a better fit. Following, I estimated the coefficients of each candidate model using maximum likelihood method. In addition, I checked each fitted model for stationarity and invertibility.

Out of the six models I estimated the coefficients for, only two were stationary and invertible. I proceeded with only these two models to perform diagnostic checking on. The diagnostic checks included checking that the fitted residuals for each model were Gaussian White Noise, and using Portmanteau tests to check correlation of residuals, linear dependence of residuals, and non-linear dependence of residuals. Since both

models passed diagnostic checking, I applied the Principle of Parsimony to choose my final model since one had more coefficients than the other. The final model I chose estimates 4 coefficients and follows a SARIMA$(1,0,0) \times (1,1,2)_{12}$ model:

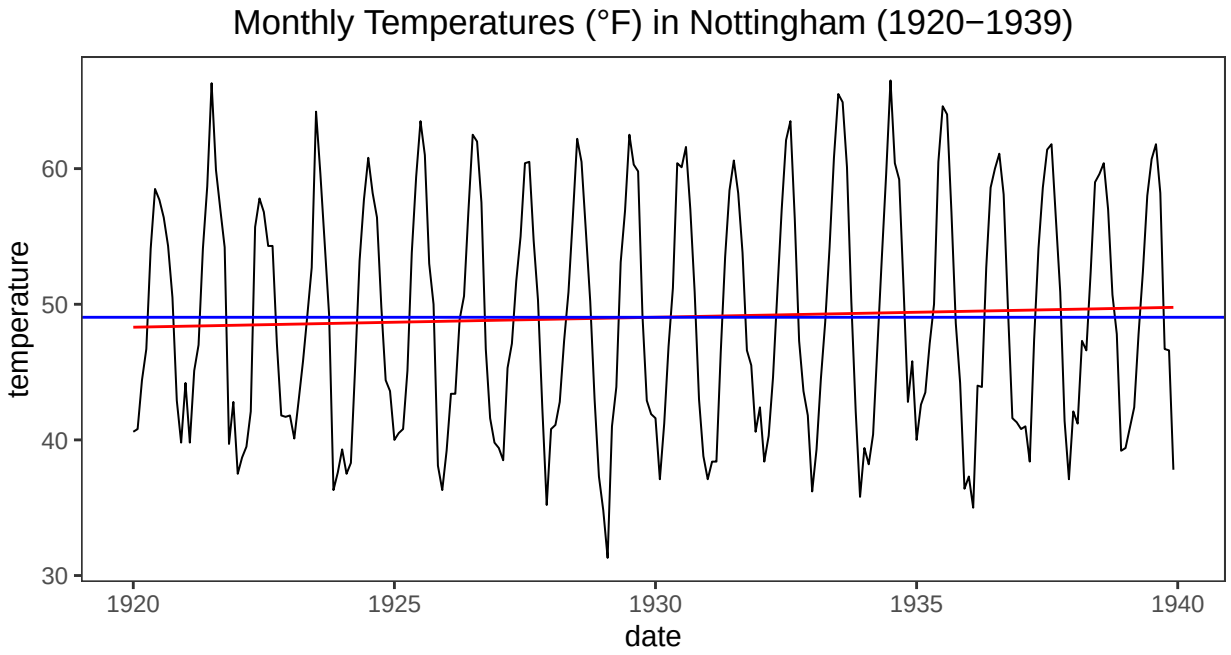
$$(1-0.3083_{(0.0677)}B)(1+0.369_{(0.0979)}B^{12})\nabla_{12}X_t = (1-1.639_{(0.2735)}B^{12}+0.6397_{(0.2067)}B^{24})Z_t, Z_t \sim \text{WN}(0, 5.427)$$

. I used this model to forecast 12 observations and then used the test dataset from the year 1939 to validate my model. The test data fell within my 95% prediction interval, further supporting my final model.

Model Building

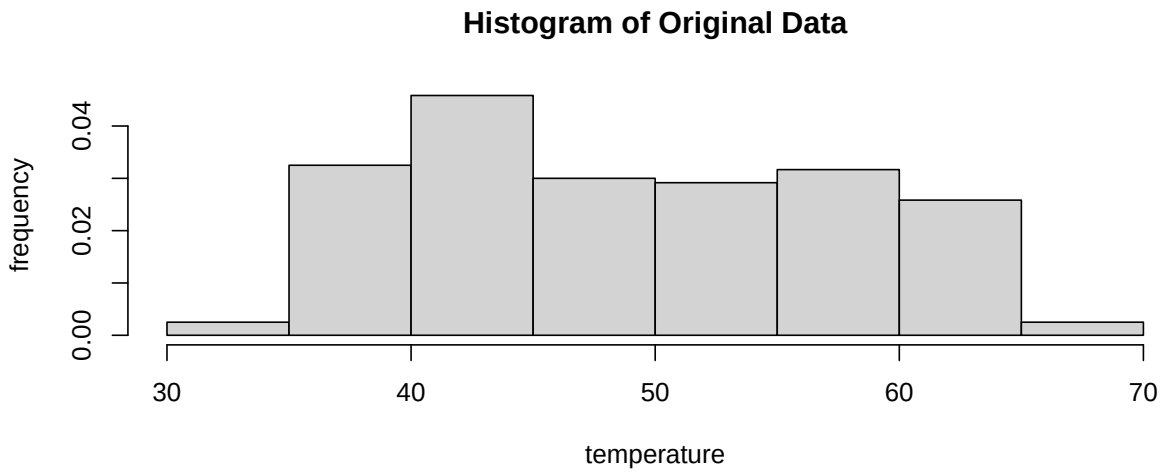
First, I will visualize the raw time series data by plotting the average monthly temperatures in Nottingham from 1920 through 1939 as well as a histogram of the data, the autocorrelation function (ACF), and partial autocorrelation function (PACF) of the raw data in order to identify the main features of the data.

Figure 1: Original time series data of average monthly temperatures in Nottingham from 1920 through 1939 with mean line (blue) and trend line (red).



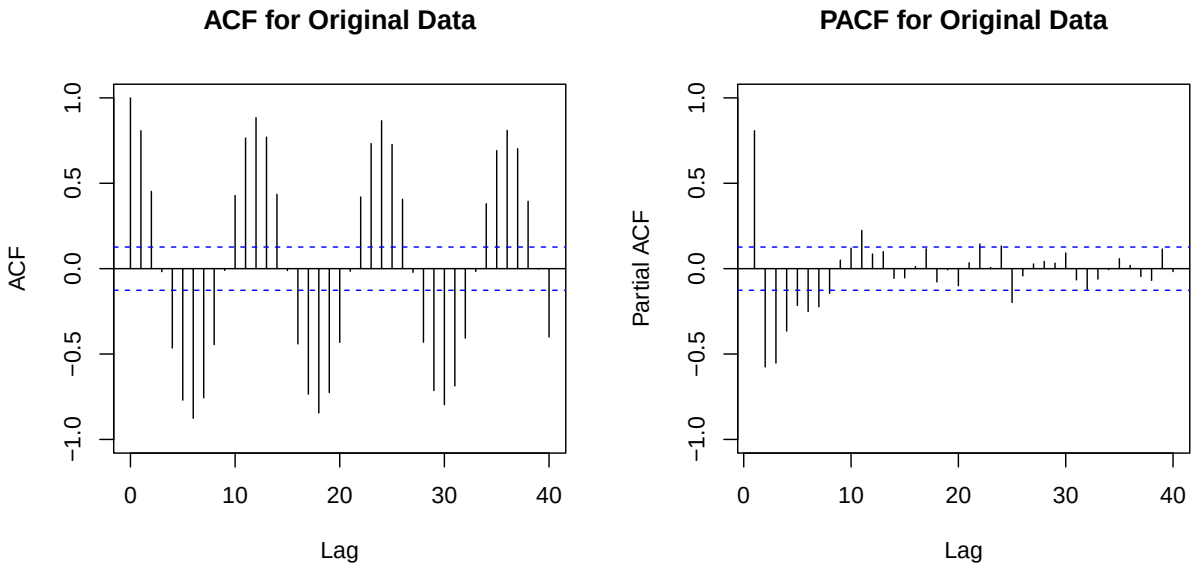
In figure 1 above, the black line is the original data, the blue line represents the mean of the data, and the red line indicates the trend line. Based on this plot, there appears to be a slight upward trend. Additionally, it is evident that there is seasonality because each year roughly follows the same cycle of increasing in temperature for the first half of the year then decreasing in temperature for the second half of the year. There do not appear to be any sharp changes in behavior and the variance looks stable.

Figure 2: Histogram of original data.



The data does not appear to be skewed, which suggests constant variance.

Figure 3: ACF and PACF of original time series data.



The ACF plot in figure 3 shows strong seasonality and both the ACF and PACF slowly decay, further supporting the behavior seen in figure 1. Therefore, this time series data is not stationary because there is seasonality and what appears to be a slight trend. I will need to apply differencing in order to make the data stationary.

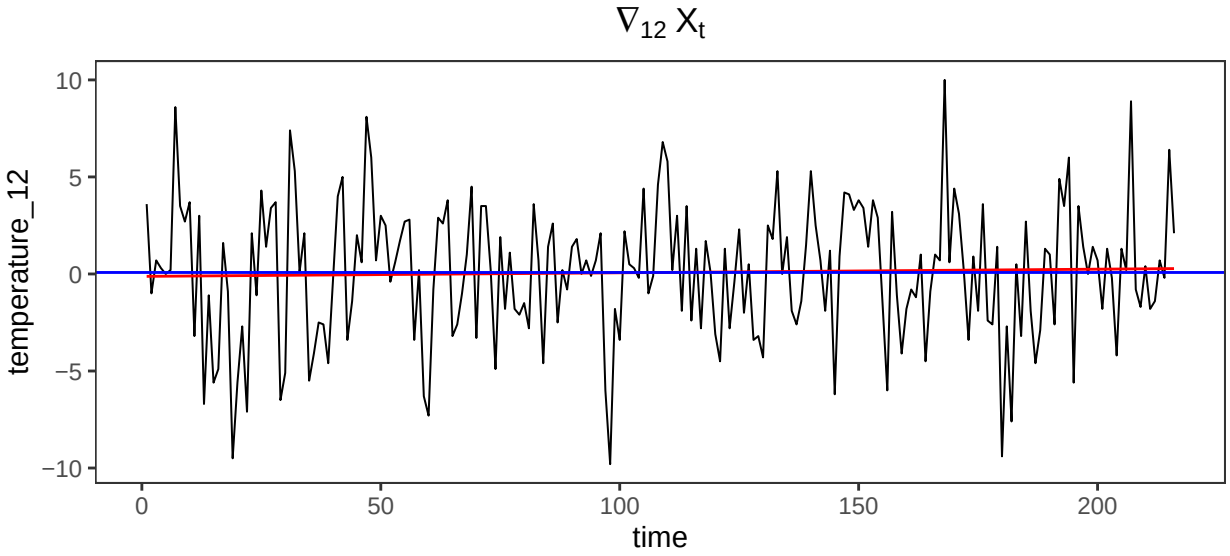
First, I partitioned the original dataset that contains a total of 240 observations into a training dataset to use for model building and a test dataset for model validation at the end via the following code:

```
train <- temp[c(1:228)] # train
test <- temp[c(229:240)] # test
```

The training dataset (X_t) consists of the first 228 observations and the test set consists of the remaining 12 observations.

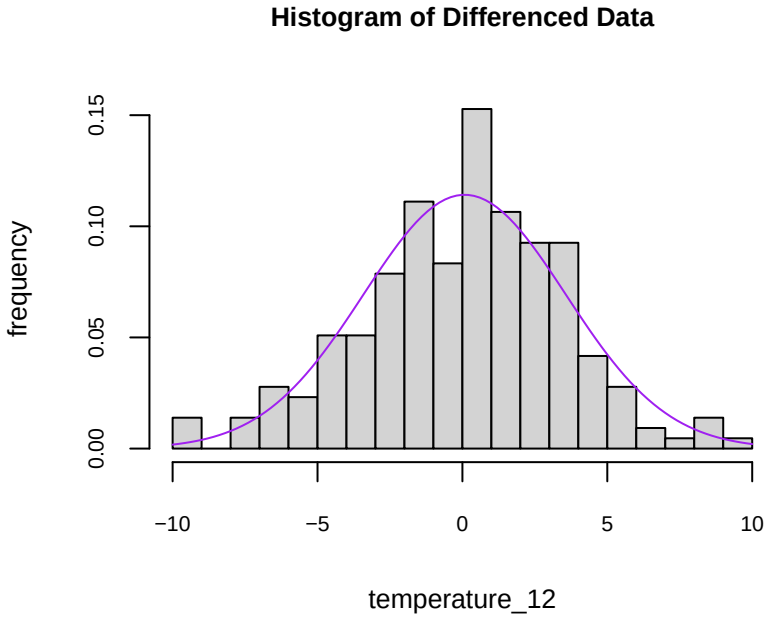
Now, I will proceed with model building using the training set. I begin by differencing the data in order to achieve a stationary time series. In order to remove seasonality in the data, I differenced once at lag 12. Lag 12 is chosen because I am working with monthly data.

Figure 4: Plot of data after removing seasonality, where X_t is the original training data..



The variance of the data, X_t, without any transformations or differencing is 73.771. After differencing once at lag 12, the variance has now decreased to 12.207. Next, I attempted to difference once at lag 1, but it increased the variance to 19.606. This is a sign of overdifferencing, so I will not be differencing at lag 1. The data appears to be stationary now.

Figure 5: Histogram of differenced data $\nabla_{12} X_t$ with Normal curve.

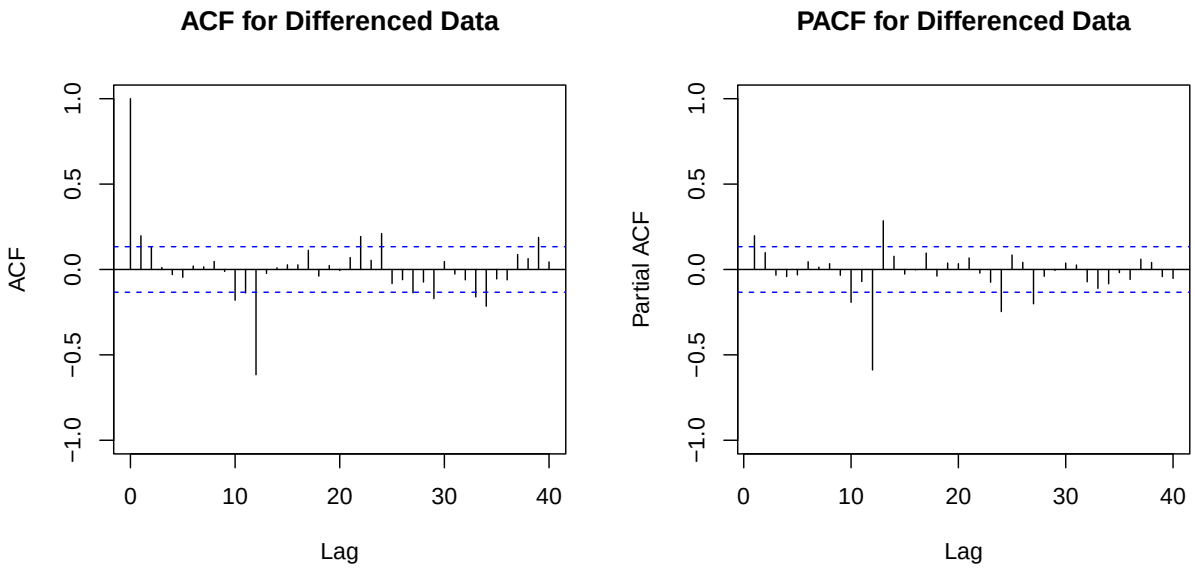


After differencing at lag 12, the data now looks approximately symmetric and Gaussian. Thus, just differencing once at lag 12 produces a stationary time series and can be used to identify a model.

Preliminary Model Identification (Choosing p's and q's)

Since the data I am working with is seasonal, I will examine the sample ACF and PACF graphs of the stationary data in order to preliminarily identify SARIMA$(p,d,q)\times(P,D,Q)_s$ model(s).

Figure 6: Sample ACF and PACF graphs.



Since I applied seasonal differencing once at lag 12, $D=1$ and $s=12$. I did not apply any differencing at lag 1, so $d=0$. The seasonal part can be modeled by looking at the seasonal lags $1s, 2s, 3s, ...$ in both the sample ACF and PACF plots in figure 6 above. The plot of the ACF of the differenced data shows a strong peak at lag $1s=12$ and a smaller peak at lag $2s=24$. Therefore, good choices for Q would be $Q=1$ or $Q=2$. The PACF plot of the differenced data shows a strong peak at lag $1s=12$ and a smaller peak at lag $2s=24$. Thus, good choices for P would also be $P=1$ or $P=2$. Additionally, to model the non-seasonal part, I can focus on the ACF/PACF between the seasonal lags (i.e. 1,...,11). The ACF shows a small peak at lag 1 which suggests $q=1$. However, the ACF also appears to tail off at lag 0, which suggests another choice can be $q=0$. The PACF shows a small peak at lag 1, thus a good choice for p is $p=1$.

Candidate Models

After identifying p's and q's based on the ACF and PACF plots, I propose the following appropriate candidate models to try:

- SARIMA$(1,0,0)\times(1,1,1)_{12}$
- SARIMA$(1,0,1)\times(1,1,1)_{12}$
- SARIMA$(1,0,0)\times(2,1,1)_{12}$
- SARIMA$(1,0,0)\times(1,1,2)_{12}$
- SARIMA$(1,0,1)\times(2,1,1)_{12}$
- SARIMA$(1,0,1)\times(1,1,2)_{12}$

- SARIMA$(1,0,0) \times (2,1,2)_{12}$
- SARIMA$(1,0,1) \times (2,1,2)_{12}$

I will first examine the Akaike Information Criterion, Corrected for Bias (AICc) for each of the candidate models in order to narrow down the number of models to perform diagnostic checking on. I generated the AICc for each model using the following code:

```
AICc(arima(train.d, order = c(1,0,q),seasonal = list(order = c(P,1,Q), period = 12),
method = 'ML'))
```

Table 1: AICc for each candidate model ($s = 12$, $d = 0$, $D = 1$), sorted by AICc in ascending order.

p	q	P	Q	num_coeffs	aicc
1	0	2	2	5	1010.19693736837
1	0	1	2	4	1011.13875797589
1	1	2	2	6	1012.11301373133
1	1	1	2	5	1012.9303696064
1	0	2	1	4	1016.80412836741
1	1	2	1	5	1018.33804246399
1	0	1	1	3	1028.59708576191
1	1	1	1	4	1030.48569938415

As seen in Table 1, the models with the lowest AICc are:

- SARIMA$(1,0,0) \times (2,1,2)_{12}$
- SARIMA$(1,0,0) \times (1,1,2)_{12}$
- SARIMA$(1,0,1) \times (2,1,2)_{12}$
- SARIMA$(1,0,1) \times (1,1,2)_{12}$

Next, I will also look at the estimates of the model coefficients to see if 0 falls within the interval

$$[\text{coefficient} - 2(\text{standard error}), \text{coefficient} + 2(\text{standard error})]$$

. If 0 falls within the interval, it is possible that coefficient can be taken as 0, thus I will generate the AICc for the modified model to see if it produces a lower AICc.

The following models have coefficients that can be taken as 0 and if replaced by 0, will not be a duplicate of a model already examined in table 1.

Output 1: Estimates and standard errors for the coefficients for SARIMA$(1,0,0) \times (2,1,2)_{12}$.

```
##
## Call:
## arima(x = train.d, order = c(1, 0, 0), seasonal = list(order = c(2, 1, 2), period = 12),
##     method = "ML")
##
## Coefficients:
##          ar1     sar1    sar2     sma1    sma2
##       0.3042  -0.0766  0.2483  -1.9461  0.9998
## s.e.  0.0677   0.0827  0.0862   0.2683  0.2751
##
## sigma^2 estimated as 5.162:  log likelihood = -498.89,  aic = 1009.77
```

As seen in output 1, 0 falls within the interval

$$\begin{aligned}[\Phi_1 - 2 * \text{s.e.}, \Phi_1 + 2 * \text{s.e.}] &= [-0.0766 - 2(0.0827), -0.0766 + 2(0.0827)] \\ &= [-0.242, 0.0888]\end{aligned}$$

. Therefore, Φ_1 may be 0 and thus another candidate model to try would be SARIMA$(1,0,0) \times (2,1,2)_{12}$ with $\Phi_1 = 0$.

Output 2: Estimates and standard errors for the coefficients for SARIMA$(1,0,1) \times (2,1,2)_{12}$.

```
##
## Call:
## arima(x = train.d, order = c(1, 0, 1), seasonal = list(order = c(2, 1, 2), period = 12),
##     method = "ML")
##
## Coefficients:
##          ar1      ma1     sar1    sar2     sma1    sma2
##       0.3943  -0.0999  -0.0741  0.2467  -1.9452  0.9991
## s.e.  0.1923   0.2075   0.0829  0.0861   0.2682  0.2751
##
## sigma^2 estimated as 5.163:  log likelihood = -498.77,  aic = 1011.54
```

As seen in output 2, 0 falls within the interval

$$\begin{aligned}[\Phi_1 - 2 * \text{s.e.}, \Phi_1 + 2 * \text{s.e.}] &= [-0.0741 - 2(0.0829), -0.0741 + 2(0.0829)] \\ &= [-0.2399, 0.0917]\end{aligned}$$

. Therefore, Φ_1 may be 0. 0 also falls within 2 standard errors from the coefficient estimates for θ_1. However, if I take just $\theta_1 = 0$ in this model, this model would be the same as the model SARIMA$(1,0,0) \times (2,1,2)_{12}$ with no modifications. Additionally, if I take both $\theta_1 = 0$ and $\Phi_1 = 0$ in this model, it would result in the same model as that in output 1. Therefore SARIMA$(1,0,1) \times (2,1,2)_{12}$ with $\Phi_1 = 0$ is another model to try.

Output 3: Estimates and standard errors for the coefficients for SARIMA$(1,0,1) \times (1,1,1)_{12}$.

```
##
## Call:
## arima(x = train.d, order = c(1, 0, 1), seasonal = list(order = c(1, 1, 1), period = 12),
##     method = "ML")
##
## Coefficients:
##          ar1      ma1     sar1     sma1
##       0.3977  -0.1055  -0.6627  -1.0000
## s.e.  0.2078   0.2260   0.0522   0.0568
##
## sigma^2 estimated as 6.695:  log likelihood = -510.09,  aic = 1030.18
```

Output 3 suggests that ϕ_1 and θ_1 could possibly be 0 because 0 falls within 2 standard errors of the coefficient estimates. Therefore, I will look at the AICc for this SARIMA$(1,0,1) \times (1,1,1)_{12}$ model while (1) fixing just $\phi_1 = 0$, (2) fixing just $\theta_1 = 0$, and (3) fixing both $\phi_1 = 0$ and $\theta_1 = 0$.

In summary, outputs 1-3 suggest examining the AICc for the following modified models:

- SARIMA$(1,0,0) \times (2,1,2)_{12}$ with $\Phi_1 = 0$

- SARIMA$(1,0,1) \times (2,1,2)_{12}$ with $\Phi_1 = 0$
- SARIMA$(1,0,1) \times (1,1,1)_{12}$ with $\phi_1 = 0$
- SARIMA$(1,0,1) \times (1,1,1)_{12}$ with $\theta_1 = 0$
- SARIMA$(1,0,1) \times (1,1,1)_{12}$ with $\phi_1 = 0$ and $\theta_1 = 0$

Table 2: Modified models and their AICc (where $s = 12$, $d = 0$, $D = 1$).

p	q	P	Q	fixed	num_coeffs	aicc
1	0	2	2	c(NA, 0, NA, NA, NA)	4	1008.89881701252
1	1	2	2	c(NA, NA, 0, NA, NA, NA)	5	1010.73700495537
1	1	1	1	c(NA, 0, NA, NA)	3	1028.59708576197
1	1	1	1	c(0, NA, NA, NA)	3	1031.06953954285
1	1	1	1	c(0, 0, NA, NA)	2	1045.71061385672

The AICc for SARIMA$(1,0,0) \times (2,1,2)_{12}$ is 1010.197, but when fixing $\Phi_1 = 0$ the AICc decreases to 1008.899. Similarly, for the model SARIMA$(1,0,0) \times (2,1,2)_{12}$ the AICc is 1012.113, but fixing $\Phi_1 = 0$ results in a lower AICc of 1010.737.

Thus, I will proceed with estimating the coefficients of the following candidate models because they have the lowest AICc:

- Model 1: SARIMA$(1,0,0) \times (2,1,2)_{12}$
- Model 2: SARIMA$(1,0,0) \times (2,1,2)_{12}$ with $\Phi_1 = 0$
- Model 3: SARIMA$(1,0,0) \times (1,1,2)_{12}$
- Model 4: SARIMA$(1,0,1) \times (2,1,2)_{12}$
- Model 5: SARIMA$(1,0,1) \times (2,1,2)_{12}$ with $\Phi_1 = 0$
- Model 6: SARIMA$(1,0,1) \times (1,1,2)_{12}$

Model Estimation

Now, I need to fit the models by estimating the coefficients of the 6 candidate models. I will be using the `arima()` function and the maximum likelihood method here. Additionally, I will check each model to see if it is stationary and/or invertible by examining unit roots of the seasonal and non-seasonal AR and MA parts. To check stationarity for a SARIMA model, $\phi(z)$ and $\Phi(z)$ must both not have any unit roots. Moreover, to check invertibility of the model, $\theta(x)$ and $\Theta(x)$ must both not have unit roots. It is ideal that the model is both stationary and invertible.

Model 1: SARIMA$(1,0,0) \times (2,1,2)_{12}$

Output 4: Estimates of coefficients for model 1.

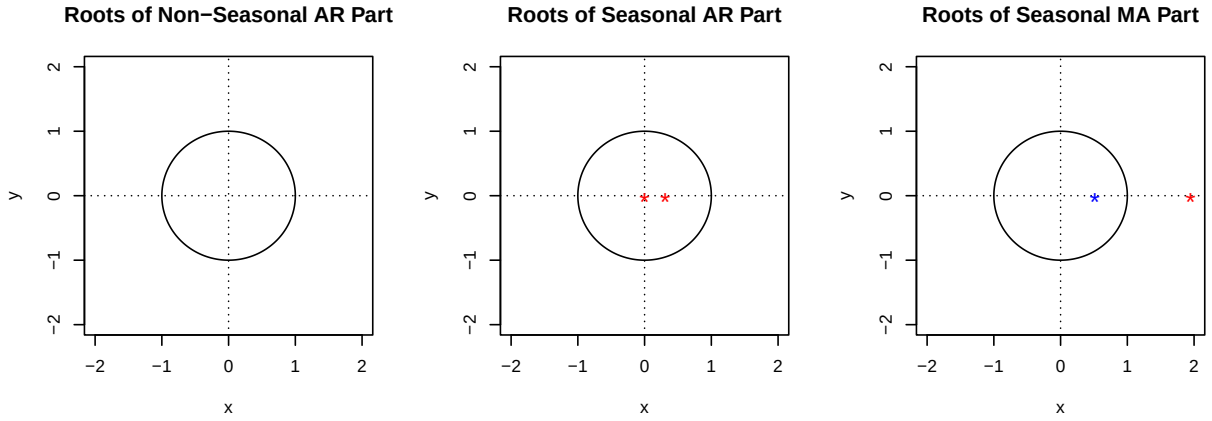
```
##
## Call:
## arima(x = train.d, order = c(1, 0, 0), seasonal = list(order = c(2, 1, 2), period = 12),
##     method = "ML")
##
## Coefficients:
##          ar1     sar1    sar2     sma1    sma2
##       0.3042  -0.0766  0.2483  -1.9461  0.9998
## s.e.  0.0677   0.0827  0.0862   0.2683  0.2751
##
## sigma^2 estimated as 5.162:  log likelihood = -498.89,  aic = 1009.77
```

This means that the estimates for the coefficients for model 1 are: $\phi_1 = 0.3042$, $\Phi_1 = -0.0766$, $\Phi_2 = 0.2483$, $\Theta_1 = -1.9461$, and $\Theta_2 = 0.9998$. The estimate for the variance of Z_t is $\sigma_Z^2 = 5.162$. This suggests that the estimate for model 1 is:

$(1 - 0.3042_{(0.0677)}B)(1 + 0.0766_{(0.0827)}B^{12} - 0.2483_{(0.0862)}B^{24})\nabla_{12}X_t = (1 - 1.9461_{(0.2683)}B^{12} + 0.9998_{(0.2751)}B^{24})Z_t$, $Z_t \sim \text{WN}(0, 5.162)$

To check stationarity for this model, $\phi(z) = 1 - 0.3042z$ and $\Phi(z) = 1 + 0.0766z - 0.2483z^2$ must both not have unit roots. Moreover, to check invertibility of the model, $\Theta(x) = 1 - 1.9461x + 0.9998x^2$ must not have a unit root.

Figure 7: Plots of roots for model 1.



For these plots of the unit roots, the red points are the unit roots and the blue points are the inverse of the unit root. As seen in figure 7, the plot for the roots for the seasonal AR part of model 1 show that there are roots inside of the unit circle. This reveals that model 1 is not stationary. However, this model is invertible because the plot of the roots for the seasonal MA part do not show any roots inside of the unit circle.

Model 2: SARIMA$(1, 0, 0) \times (2, 1, 2)_{12}$ **with** $\Phi_1 = 0$

Output 5: Estimates of coefficients for model 2.

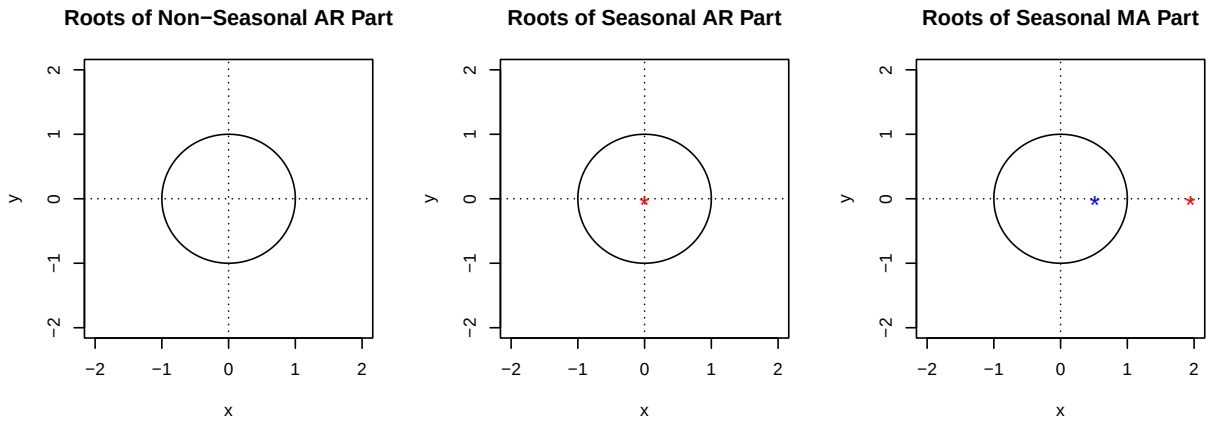
```
##
## Call:
## arima(x = train.d, order = c(1, 0, 0), seasonal = list(order = c(2, 1, 2), period = 12),
##     fixed = c(NA, 0, NA, NA, NA), method = "ML")
##
## Coefficients:
##          ar1  sar1    sar2     sma1    sma2
##       0.2979     0  0.2833  -1.9483  1.0001
## s.e.  0.0675     0  0.0801   0.1222  0.1242
##
## sigma^2 estimated as 5.304:  log likelihood = -499.3,  aic = 1008.6
```

This means that the estimates for the coefficients for model 2 are: $\phi_1 = 0.2979$, $\Phi_1 = 0$, $\Phi_2 = 0.2833$, $\Theta_1 = -1.9483$, and $\Theta_2 = 1.0001$. The estimate for the variance of Z_t is $\sigma_Z^2 = 5.304$. This suggests that the estimate for model 2 is:

$(1-0.2979_{(0.0675)}B)(1-0.2833_{(0.0801)}B^{24})\nabla_{12}X_t = (1-1.9483_{(0.1222)}B^{12}+1.0001_{(0.1242)}B^{24})Z_t, Z_t \sim \text{WN}(0, 5.304)$

To check stationarity for this model, $\phi(z) = 1 - 0.2979z$ and $\Phi(z) = 1 - 0.2833z^2$ must both not have unit roots. Moreover, to check invertibility of the model, $\Theta(x) = 1 - 1.9483x + 1.0001x^2$ must not have a unit root.

Figure 8: Plots of roots for model 2.



Model 2 is also not stationary because the seasonal AR part has a unit root, but this model is invertible because the MA part does not have any unit roots.

Model 3: SARIMA$(1,0,0) \times (1,1,2)_{12}$

Output 6: Estimates of coefficients for model 3.

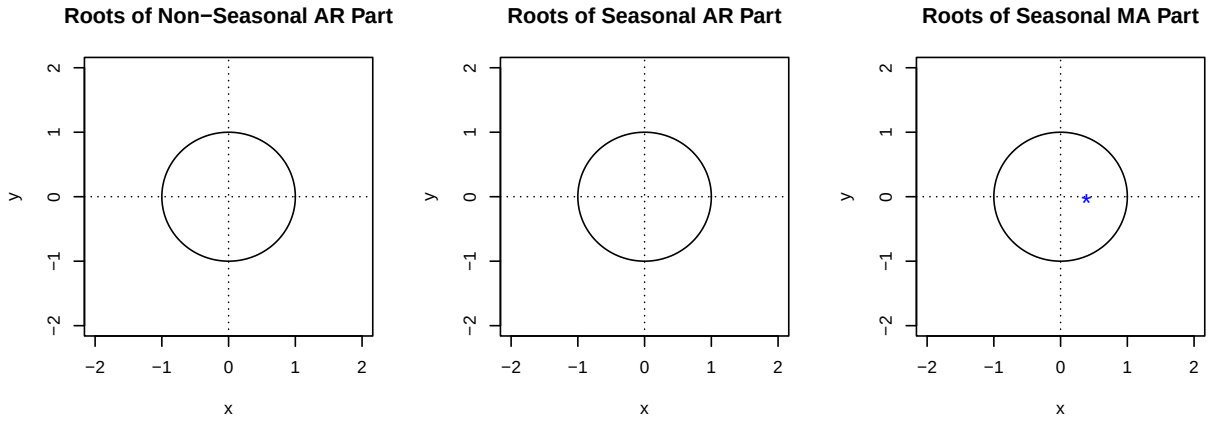
```
##
## Call:
## arima(x = train.d, order = c(1, 0, 0), seasonal = list(order = c(1, 1, 2), period = 12),
##     method = "ML")
##
## Coefficients:
##          ar1     sar1     sma1    sma2
##       0.3083  -0.3690  -1.6397  0.6397
## s.e.  0.0677   0.0979   0.2735  0.2067
##
## sigma^2 estimated as 5.427:  log likelihood = -500.42,  aic = 1010.84
```

This means that the estimates for the coefficients for model 3 are: $\phi_1 = 0.3083$, $\Phi_1 = -0.3690$, $\Theta_1 = -1.6397$, and $\Theta_2 = 0.6397$. The estimate for the variance of Z_t is $\sigma_Z^2 = 5.427$. This suggests that the estimate for model 3 is:

$$(1-0.3083_{(0.0677)}B)(1+0.369_{(0.0979)}B^{12})\nabla_{12}X_t = (1-1.639_{(0.2735)}B^{12}+0.6397_{(0.2067)}B^{24})Z_t, Z_t \sim \text{WN}(0, 5.427)$$

To check stationarity for this model, $\phi(z) = 1 - 0.3083z$ and $\Phi(z) = 1 + 0.369z$ must both not have unit roots. Moreover, to check invertibility of the model, $\Theta(x) = 1 - 1.639x + 0.6397x^2$ must not have a unit root.

Figure 9: Plots of roots for model 3.



Model 3 is both stationary and invertible because it does not have any unit roots in either the AR part or MA part.

Model 4: SARIMA$(1,0,1) \times (2,1,2)_{12}$

Output 7: Estimates of coefficients for model 4.

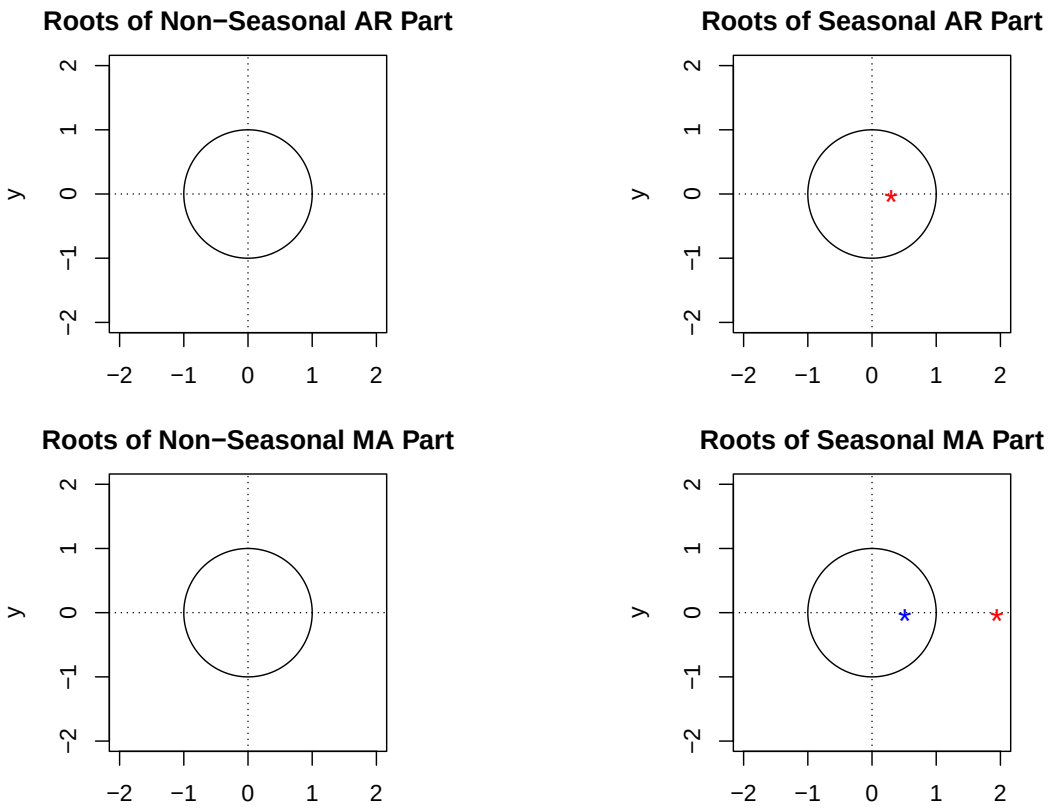
```
##
## Call:
## arima(x = train.d, order = c(1, 0, 1), seasonal = list(order = c(2, 1, 2), period = 12),
##     method = "ML")
##
## Coefficients:
##          ar1      ma1     sar1    sar2     sma1    sma2
##       0.3943  -0.0999  -0.0741  0.2467  -1.9452  0.9991
## s.e.  0.1923   0.2075   0.0829  0.0861   0.2682  0.2751
##
## sigma^2 estimated as 5.163:  log likelihood = -498.77,  aic = 1011.54
```

This means that the estimates for the coefficients for model 4 are: $\phi_1 = 0.3943$, $\Phi_1 = -0.0741$, $\Phi_2 = 0.2467$, $\theta_1 = -0.0999$, $\Theta_1 = -1.9452$, and $\Theta_2 = 0.9991$. The estimate for the variance of Z_t is $\sigma_Z^2 = 5.163$. This suggests that the estimate for model 4 is:

$(1 - 0.3943_{(0.1923)}B)(1 + 0.0741_{(0.0829)}B^{12} - 0.2467_{(0.0861)}B^{24})\nabla_{12}X_t = (1 - 0.0999_{(0.2075)}B)(1 - 1.9452_{(0.2682)}B^{12} + 0.9991_{(0.2751)}B^{24})Z_t, Z_t \sim \text{WN}(0, 5.163)$

To check stationarity for this model, $\phi(z) = 1-0.3943z$ and $\Phi(z) = 1+0.0741z-0.2467z^2$ must both not have unit roots. Moreover, to check invertibility of the model, $\theta(x) = 1-0.0999x$ and $\Theta(x) = 1-1.9452x+0.9991x^2$ must not have any unit roots.

Figure 10: Plots of roots for model 4.



Model 4 is not stationary because it has a unit root in the seasonal AR part. It is invertible, though, because it does not have any unit roots in the MA part.

Model 5: SARIMA$(1,0,1)\times(2,1,2)_{12}$ **with** $\Phi_1 = 0$

Output 8: Estimates of coefficients for model 5.

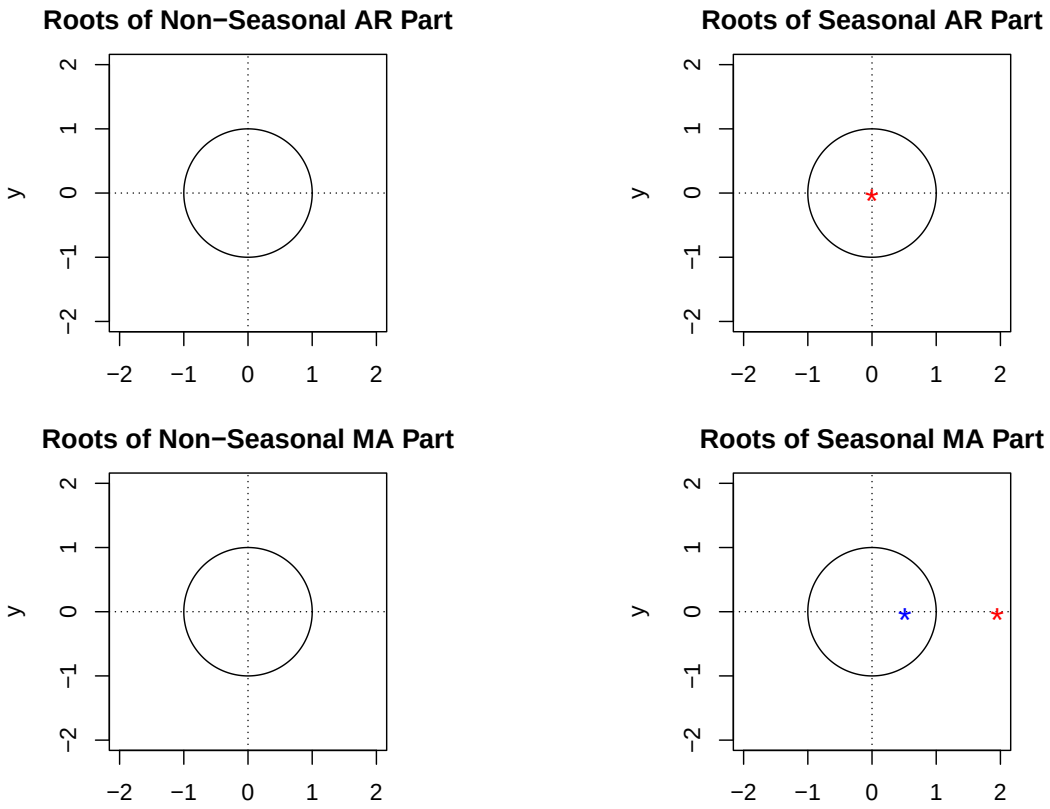
```
##
## Call:
## arima(x = train.d, order = c(1, 0, 1), seasonal = list(order = c(2, 1, 2), period = 12),
##     fixed = c(NA, NA, 0, NA, NA, NA), method = "ML")
##
## Coefficients:
##          ar1      ma1  sar1    sar2     sma1    sma2
##       0.4012  -0.1138     0  0.2801  -1.9479  1.0000
## s.e.  0.1954   0.2108     0  0.0801   0.1241  0.1261
##
## sigma^2 estimated as 5.296:  log likelihood = -499.16,  aic = 1010.31
```

This means that the estimates for the coefficients for model 4 are: $\phi_1 = 0.4012$, $\Phi_1 = 0$, $\Phi_2 = 0.2801$, $\theta_1 = -0.1138$, $\Theta_1 = -1.9479$, and $\Theta_2 = 1.0000$. The estimate for the variance of Z_t is $\sigma_Z^2 = 5.296$. This suggests that the estimate for model 5 is:

$(1 - 0.4012_{(0.1954)}B)(1 - 0.2801_{(0.0801)}B^{24})\nabla_{12}X_t = (1 - 0.1138_{(0.2108)}B)(1 - 1.9479_{(0.1241)}B^{12} + 1_{(0.1261)}B^{24})Z_t$,
$Z_t \sim \text{WN}(0, 5.296)$

To check stationarity for this model, $\phi(z) = 1 - 0.4012z$ and $\Phi(z) = 1 - 0.2801z^2$ must both not have unit roots. Moreover, to check invertibility of the model, $\theta(x) = 1 - 0.1138x$ and $\Theta(x) = 1 - 1.9479x + x^2$ must not have any unit roots.

Figure 11: Plots of roots for model 5.



Model 5 is not stationary because it has a unit root in the seasonal AR part. It is invertible, though, because it does not have any unit roots in the MA part.

Model 6: SARIMA$(1, 0, 1) \times (1, 1, 2)_{12}$

Output 9: Estimates of coefficients for model 6.

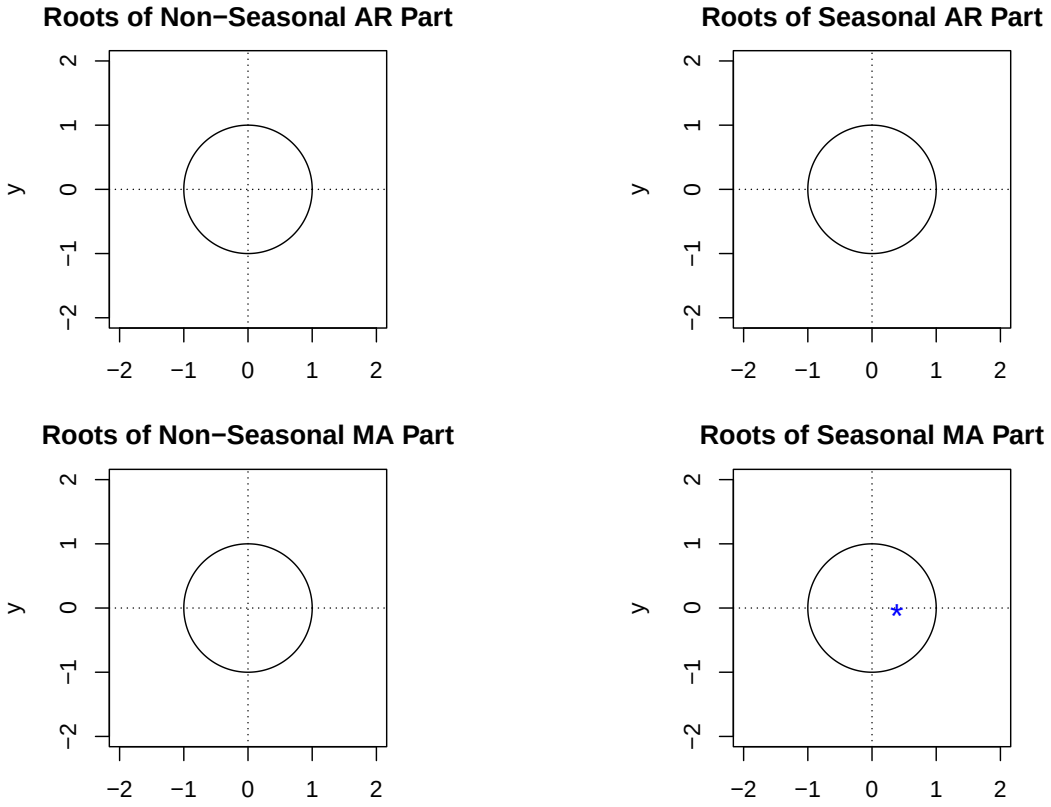
```
##
## Call:
## arima(x = train.d, order = c(1, 0, 1), seasonal = list(order = c(1, 1, 2), period = 12),
##     method = "ML")
##
## Coefficients:
##          ar1      ma1     sar1     sma1    sma2
##       0.4106  -0.1132  -0.3667  -1.6375  0.6375
## s.e.  0.1816   0.1959   0.0980   0.2778  0.2086
##
## sigma^2 estimated as 5.425:  log likelihood = -500.25,  aic = 1012.5
```

This means that the estimates for the coefficients for model 4 are: $\phi_1 = 0.4012$, $\Phi_1 = 0$, $\Phi_2 = 0.2801$, $\theta_1 = -0.1138$, $\Theta_1 = -1.9479$, and $\Theta_2 = 1.0000$. The estimate for the variance of Z_t is $\sigma_Z^2 = 5.296$. This suggests that the estimate for model 5 is:

$(1 - 0.4012_{(0.1954)}B)(1 - 0.2801_{(0.0801)}B^{24})\nabla_{12}X_t = (1 - 0.1138_{(0.2108)}B)(1 - 1.9479_{(0.1241)}B^{12} + 1_{(0.1261)}B^{24})Z_t$, $Z_t \sim \text{WN}(0, 5.296)$

To check stationarity for this model, $\phi(z) = 1 - 0.4012z$ and $\Phi(z) = 1 - 0.2801z^2$ must both not have unit roots. Moreover, to check invertibility of the model, $\theta(x) = 1 - 0.1138x$ and $\Theta(x) = 1 - 1.9479x + x^2$ must not have any unit roots.

Figure 12: Plots of roots for model 6.



Model 6 is both stationary and invertible because it has no unit roots in either the AR part or the MA part.

I will now proceed to diagnostic checking with the two candidate models that are both stationary and invertible:

- Model 3:

  $(1-0.3083_{(0.0677)}B)(1+0.369_{(0.0979)}B^{12})\nabla_{12}X_t = (1-1.639_{(0.2735)}B^{12}+0.6397_{(0.2067)}B^{24})Z_t, Z_t \sim \text{WN}(0, 5.427)$

- Model 6:

  $(1 - 0.4012_{(0.1954)}B)(1 - 0.2801_{(0.0801)}B^{24})\nabla_{12}X_t = (1 - 0.1138_{(0.2108)}B)(1 - 1.9479_{(0.1241)}B^{12} + 1_{(0.1261)}B^{24})Z_t$, $Z_t \sim \text{WN}(0, 5.296)$

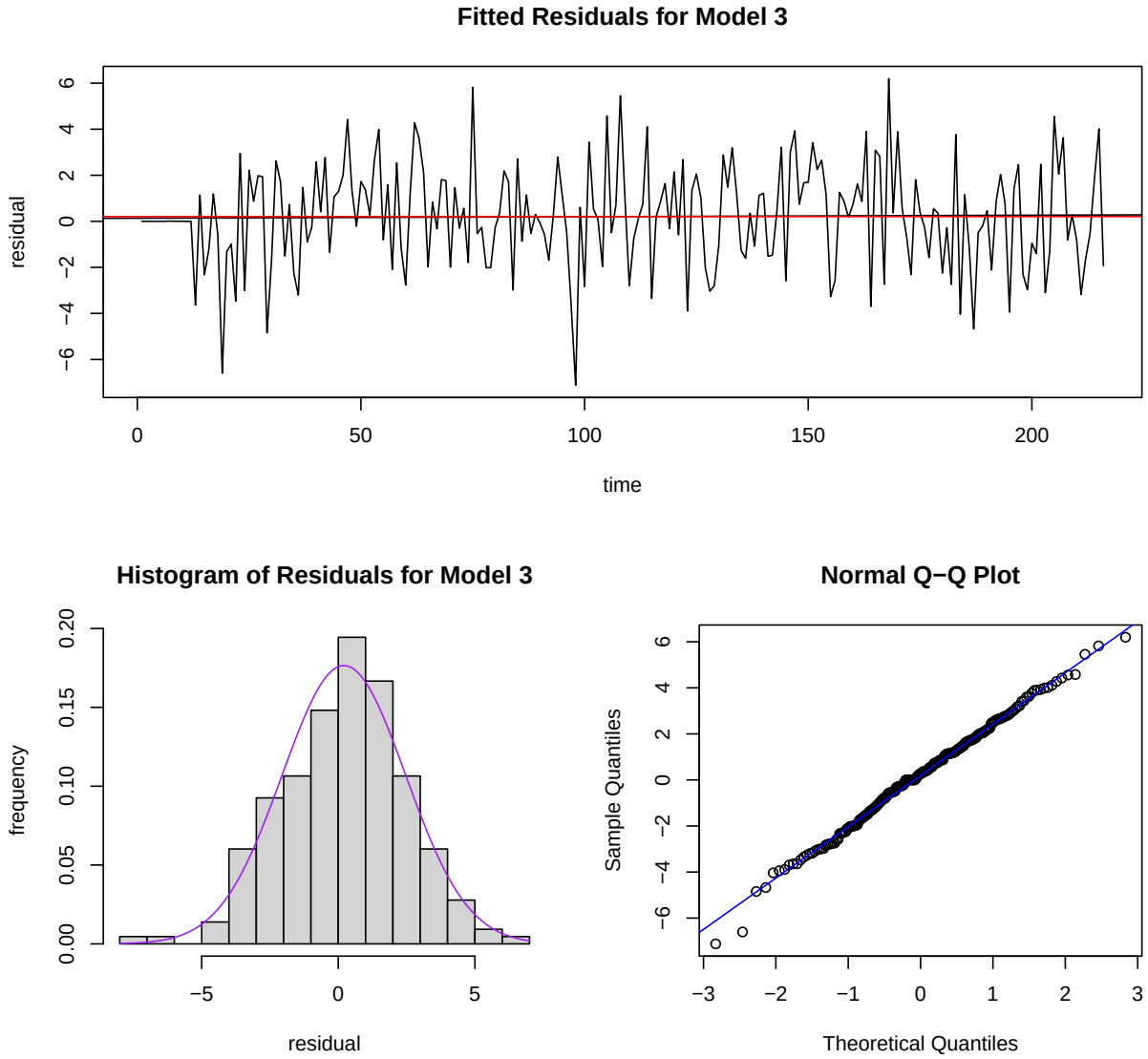
Diagnostic Checking

For each candidate model that is stationary and invertible, I will first extract the residuals from the fitted model and then plot the fitted residuals, histogram of the residuals, and the Normal Q-Q plot in order to check resemblance of Gaussian White Noise. Then, I will plot the ACF and PACF of the residuals to ensure they resemble White Noise. Following, I will perform Shapiro-Wilk test for normality as well as Portmanteau tests. The Potmanteau tests I will run are Box-Pierce to test if the residuals are correlated, Box-Ljung to test linear dependence of the residuals, and Mc-Leod Li to test non-linear dependence of the residuals.

Model 3

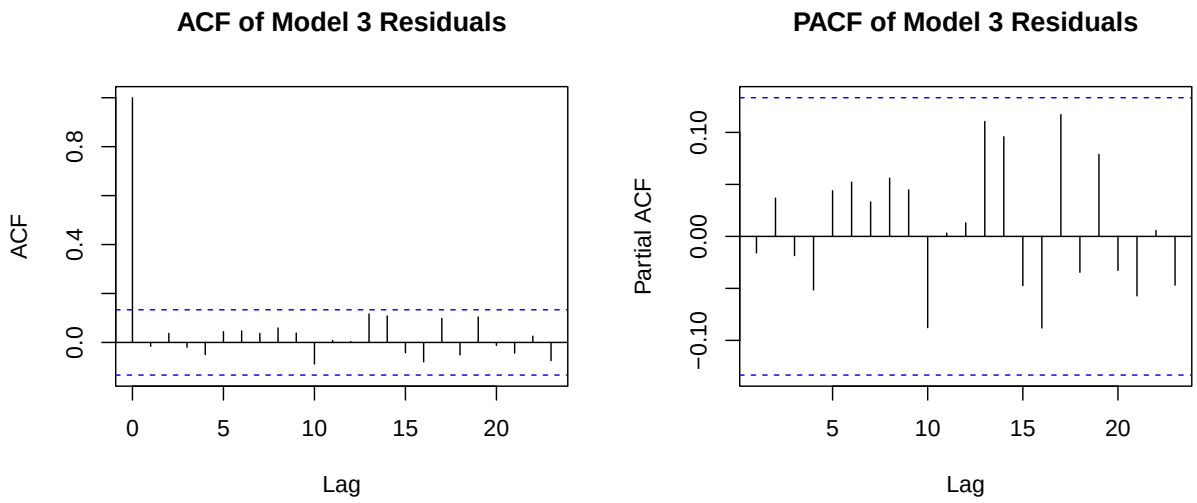
$$(1-0.3083_{(0.0677)}B)(1+0.369_{(0.0979)}B^{12})\nabla_{12}X_t = (1-1.639_{(0.2735)}B^{12}+0.6397_{(0.2067)}B^{24})Z_t, Z_t \sim \text{WN}(0, 5.427)$$

Figure 13: Plots of fitted residuals, histogram of residuals, and Normal Q-Q plot of residuals for model 3.



The mean of the residuals is 0.205, which is close to 0 and, based on the plot of the fitted residuals above, also appears to be approximately constant. The histogram of the residuals approximately resembles a Normal distribution, and all the points in the Normal Q-Q plot are very close to the blue line. All of these are good indicators that the residuals are Gaussian White Noise.

Figure 14: ACF and PACF graphs of the residuals for model 3.



All ACF and PACF of the residuals for model 3 fall within the confidence intervals and can be counted as 0, which suggests that the residuals are White Noise. Next, I will perform the Shapiro-Wilk Normality test, Box-Pierce test, Ljung-Box test, and Mc-Leod Li test on the residuals. I will use `lag=15` because the square-root of the number of observations in the training dataset is $\sqrt{228} \approx 15$. Additionally, since model 3 estimates 4 coefficients, I will set `fitdf=4` for the Box-Pierce test and Ljung-Box test. For the Mc-Leod Li test, `fitdf=0`.

Output 10: Outputs for the Shapiro-Wilk Normality test, Box-Pierce test, Ljung-Box test, and Mc-Leod Li test on the residuals for model 3.

```
shapiro.test(res3) # shapiro wilk
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res3
## W = 0.99553, p-value = 0.7818
```

```
Box.test(res3, lag = 15, type = c('Box-Pierce'), fitdf = 4) # box-pierce
```

```
##
##  Box-Pierce test
##
## data:  res3
## X-squared = 10.736, df = 11, p-value = 0.4657
```

```r
Box.test(res3, lag = 15, type = c('Ljung-Box'), fitdf = 4) # ljung-box
```

```
##
##  Box-Ljung test
##
## data:  res3
## X-squared = 11.403, df = 11, p-value = 0.4101
```

```r
Box.test(res3^2, lag = 15, type = c('Ljung-Box'), fitdf = 0) # mc-leod li
```

```
##
##  Box-Ljung test
##
## data:  res3^2
## X-squared = 13.037, df = 15, p-value = 0.5994
```

As seen in output 10, all the p-values are greater than $\alpha = 0.05$, which supports the assumption that the residuals are approximately Gaussian and do not have linear or non-linear dependence. Now, I will fit the residuals to an AR model using `ar()` with the method Yule-Walker estimation.

Output 11: Order of AR model for the residuals of model 3.

```
##
## Call:
## ar(x = res3, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  5.107
```

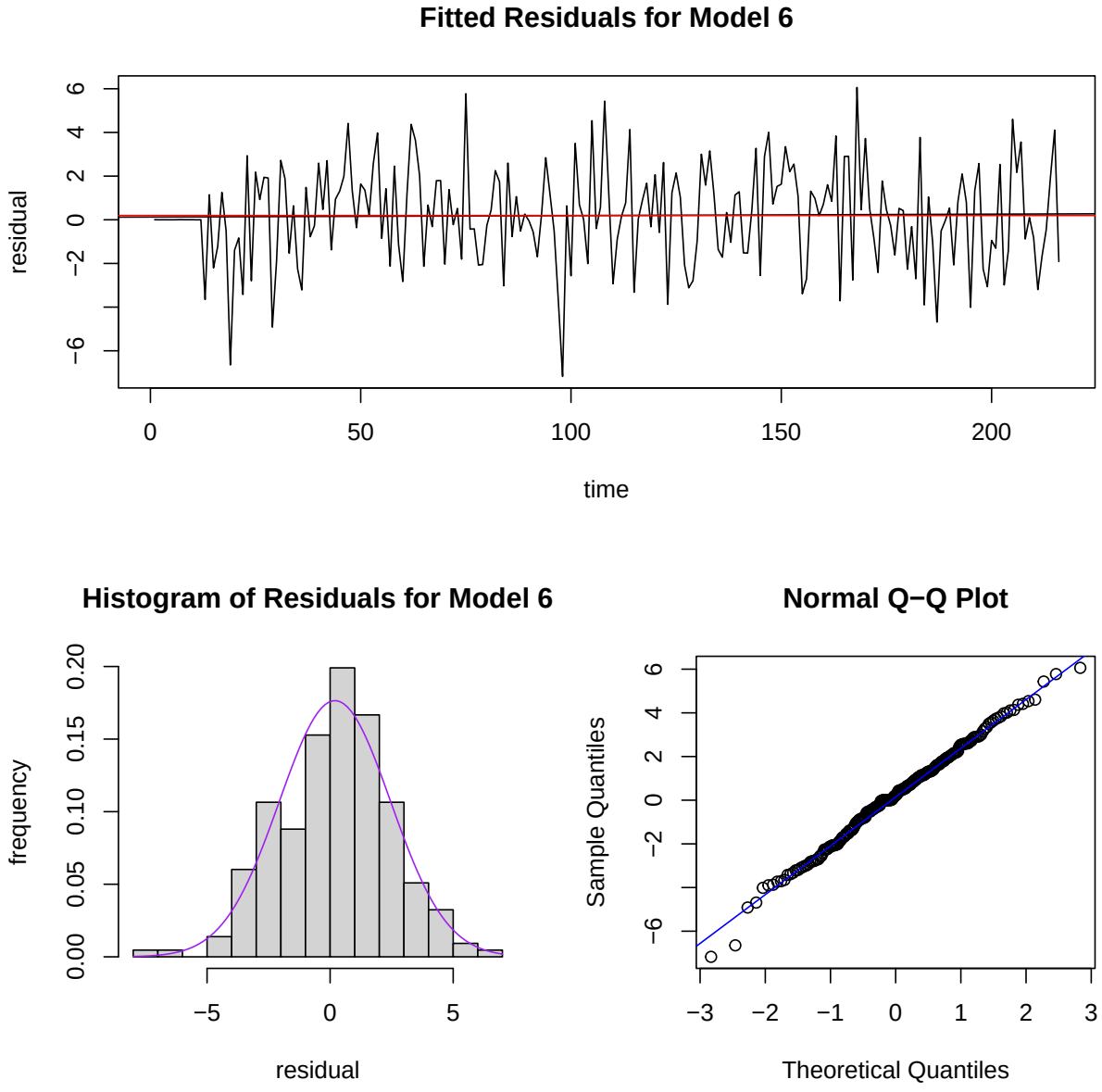
The residuals were fit to AR(0), which is White Noise. Model 3 has passed all diagnostic checks and can be used for forecasting.

Next, I will perform diagnostic checks on model 6.

Model 6

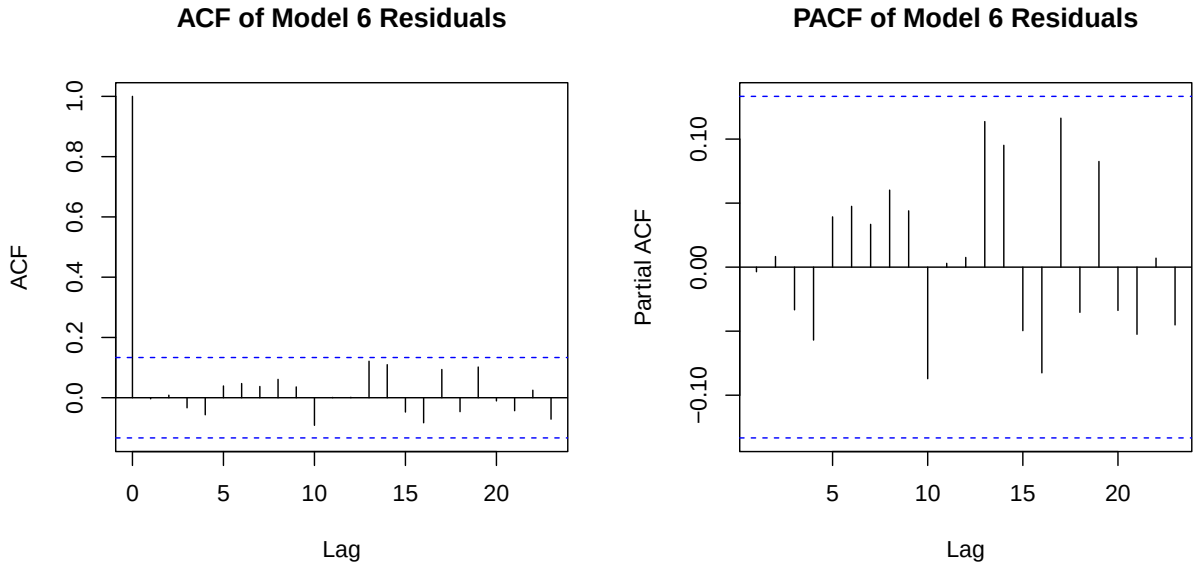
$(1 - 0.4012_{(0.1954)}B)(1 - 0.2801_{(0.0801)}B^{24})\nabla_{12}X_t = (1 - 0.1138_{(0.2108)}B)(1 - 1.9479_{(0.1241)}B^{12} + 1_{(0.1261)}B^{24})Z_t$,
$Z_t \sim \text{WN}(0, 5.296)$

Figure 15: Plots of fitted residuals, histogram of residuals, and Normal Q-Q plot of residuals for model 6.



The mean of the residuals for model 6 is also 0.205, which is close to 0 and also appears to be approximately constant based on the plot of the fitted residuals above in figure 13. The histogram of the residuals approximately resembles a Normal distribution, and all the points in the Normal Q-Q plot are very close to the blue line. All of these are good indicators that the residuals are Guassian White Noise.

Figure 16: ACF and PACF graphs of the residuals for model 6.



As seen in figure 16, all ACF and PACF of the residuals for model 6 fall within the confidence intervals and can be counted as 0, which is a good sign that the residuals are White Noise. Next, I will perform the Shapiro-Wilk Normality test, Box-Pierce test, Ljung-Box test, and Mc-Leod Li test on the residuals. I will use `lag=15` because the square-root of the number of observations in the training dataset is $\sqrt{228} \approx 15$. Additionally, since model 6 estimates 5 coefficients, I will set `fitdf=5` for the Box-Pierce test and Ljung-Box test. For the Mc-Leod Li test, `fitdf=0`.

Output 12: Outputs for the Shapiro-Wilk Normality test, Box-Pierce test, Ljung-Box test, and Mc-Leod Li test on the residuals for model 6.

```
shapiro.test(res6) # shapiro-wilk
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res6
## W = 0.99503, p-value = 0.7034
```

```
Box.test(res6, lag = 15, type = c('Box-Pierce'), fitdf = 5) # box-pierce
```

```
##
##  Box-Pierce test
##
## data:  res6
## X-squared = 11.187, df = 10, p-value = 0.3431
```

```
Box.test(res6, lag = 15, type = c('Ljung-Box'), fitdf = 5) # ljung-box
```

```
##
##  Box-Ljung test
##
## data:  res6
## X-squared = 11.894, df = 10, p-value = 0.2922
```

```r
Box.test(res6^2, lag = 15, type = c('Ljung-Box'), fitdf = 0) # mc-leod li
```

```
##
##  Box-Ljung test
##
## data:  res6^2
## X-squared = 13.6, df = 15, p-value = 0.556
```

Ass seen in output 12, all the p-values are greater than $\alpha = 0.05$, which supports the assumption that the residuals are approximately Gaussian and do not have linear or non-linear dependence. Now, I will fit the residuals to an AR model using `ar()` with the method Yule-Walker estimation.

Output 13: Order of AR model for the residuals of model 6.

```
##
## Call:
## ar(x = res6, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  5.109
```

The residuals were fit to AR(0), which is White Noise. Model 6 has passed all diagnostic checks and can be used for forecasting.

Both model 3 and model 6 passed diagnostic checking and are satisfactory to be used for forecasting. Model 3 estimates 4 coefficients while model 6 estimates 5 coefficients. Thus, by the Principle of Parsimony, I will proceed with using model 3 for forecasting since it has fewer coefficients.

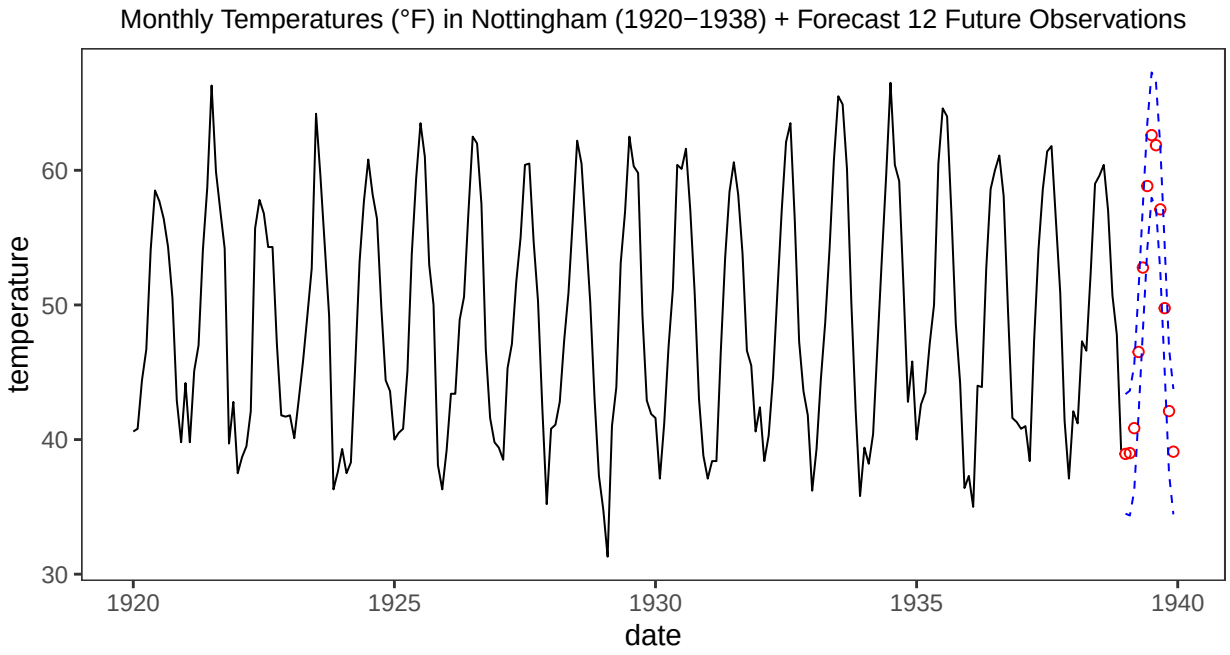
Therefore, **my final model is model 3, which follows a SARIMA**$(1,0,0) \times (1,1,2)_{12}$ **model**:

$$(1-0.3083_{(0.0677)}B)(1+0.369_{(0.0979)}B^{12})\nabla_{12}X_t = (1-1.639_{(0.2735)}B^{12}+0.6397_{(0.2067)}B^{24})Z_t, Z_t \sim \text{WN}(0, 5.427)$$

Forecasting

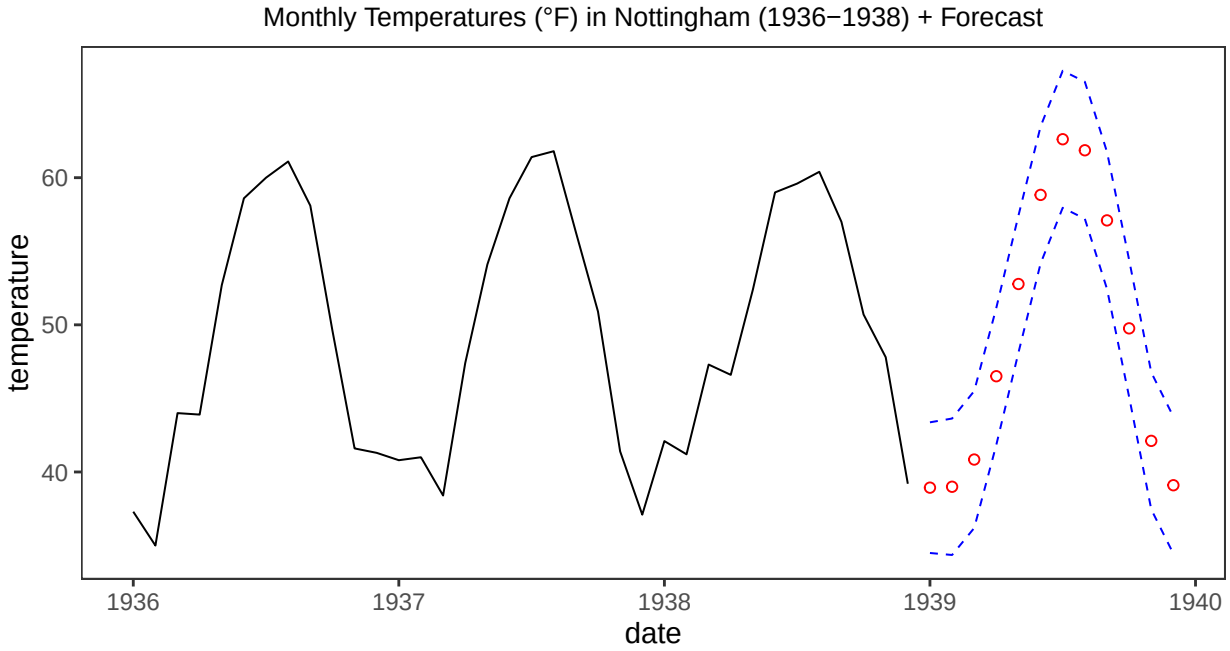
Since I did not perform any transformations, other than differencing, I will use model 3 to forecast 12 future observations on the the original data.

Figure 17: Forecast for 12 future observations on original training data using model 3.



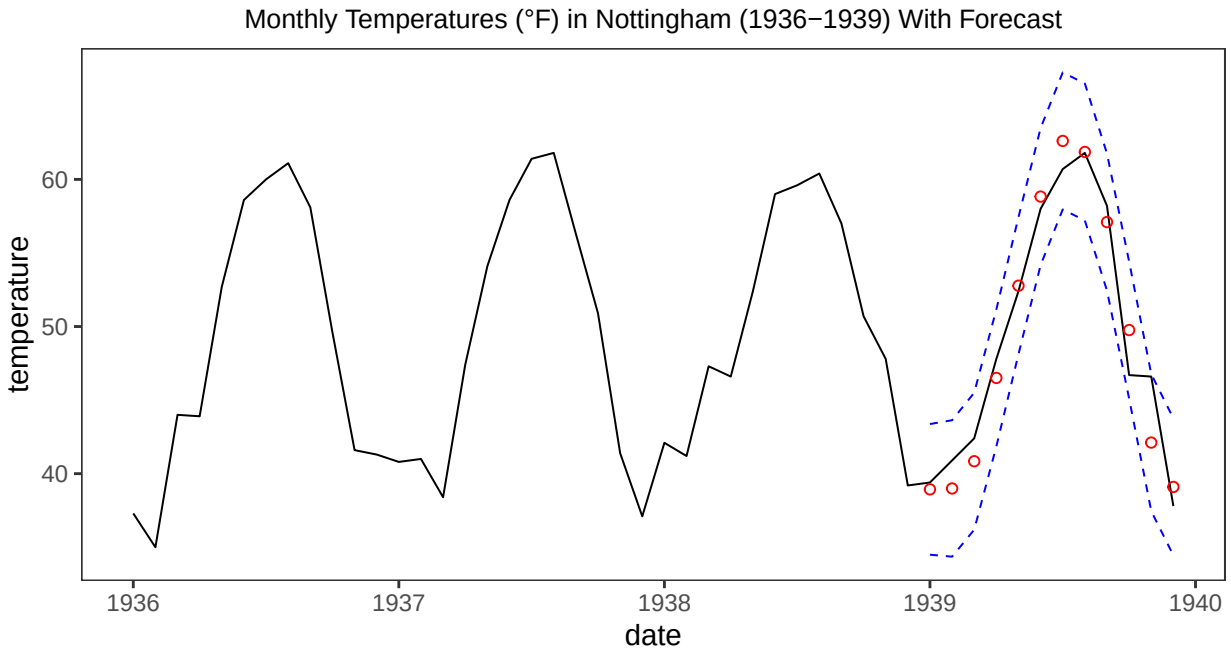
In figure 17, the red points indicate the predicted point and the dashed blue lines indicate the lower and upper bounds of the 95% confidence interval for the prediction. The interval is computed via the formula [prediction−1.96(standard error), prediction+1.96(standard error)].

Figure 18: Zoomed in graph of the forecast, starting from 1936.



Below I have added the test data to the zoomed in plot.

Figure 19: Zoomed in graph of the forecast with test data, starting from 1936.



In figure 19 above, the black line is the original data while the red points are the forecasts and the dashed blue lines are the lower and upper bounds for the 95% confidence interval of the forecasts. As seen in this graph, the test data falls within the 95% confidence interval. This supports the validity of my final model.

Conclusion

The goal of forecasting 12 average monthly temperatures (°F) at Nottingham Castle using past data from 1920 through 1938 was achieved. I found 2 suitable models for forecasting, but ultimately chose the one with less coefficients due to the Principle of Parsimony. My final model follows a $\text{SARIMA}(1,0,0) \times (1,1,2)_{12}$ model and has the following mathematical formula:

$$(1-0.3083_{(0.0677)}B)(1+0.369_{(0.0979)}B^{12})\nabla_{12}X_t = (1-1.639_{(0.2735)}B^{12}+0.6397_{(0.2067)}B^{24})Z_t, Z_t \sim \text{WN}(0, 5.427)$$

. The actual data fell between the 95% prediction interval, validating the model I identified.

I would like the acknowledge my professor, Dr. Raisa Feldman, for this class, PSTAT 174: Time Series, for equipping me with sufficient background and knowledge to be able to apply Box-Jenkins methodology in order to build a model for a dataset and use it for forecasting.

References

Dataset used for this project was found here: https://vincentarelbundock.github.io/Rdatasets/csv/datasets/nottem.csv.

All learning material was attained through Professor Feldman's lectures.

Apendix (Full Code)

```r
knitr::opts_chunk$set(echo = TRUE)
library(ggplot2) # for plotting
library(ggfortify) # for getting CI value from acf() and pacf()
library(MASS) # for box cox
library(MuMIn) # for AICc()
library(forecast) # for auto.arima()
library(pander) # for pander()
library(dplyr)

# read in data
temp.csv <- read.table("temp.csv", header = T,
                       sep = ',', col.names = c('index', 'date','temp'))
temp.csv <- temp.csv[2:3]

# create time series object
temp <- ts(temp.csv[,2], start = c(1920), frequency = 12)

# plot data
ggplot(temp.csv, aes(x = date, y = temp)) +
  geom_line(lwd = 0.35) +
  geom_smooth(method = lm, se = FALSE, col='red', size = 0.5) +
  geom_hline(yintercept = mean(temp.csv[,2]), col = 'blue') +
  ggtitle('Monthly Temperatures (°F) in Nottingham (1920-1939)') +
  ylab('temperature') +
  theme_bw() +
  theme(plot.title = element_text(hjust = 0.5),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())

# histogram of original data
hist(temp, main = "Histogram of Original Data", xlab = 'temperature',
     ylab = 'frequency', prob = TRUE, breaks = 10)

op <- par(mfrow = c(1,2))

# ACF
acf <- acf(temp.csv[,2], lag.max = 40,
    ylim = c(-1,1),
    main = 'ACF for Original Data',
    plot = T)

# PACF
pacf <- pacf(temp.csv[,2], lag.max = 40,
    ylim = c(-1,1),
    main = 'PACF for Original Data',
    plot = T)
train <- temp[c(1:228)] # train
test <- temp[c(229:240)] # test

# remove seasonality
train.d <- diff(train, 12)

# plot data after removing seasonality
```

```r
data.frame(time = c(1:length(train.d)),
           temp.stat = train.d) %>%
ggplot(aes(x = time, y = temp.stat)) +
  geom_line(lwd = 0.35) +
  geom_smooth(method = lm, se = FALSE, col='red', size = 0.5) +
  geom_hline(yintercept = mean(train.d), col = 'blue') +
  ggtitle(expression(nabla[12] ~ X[t])) +
  ylab('temperature_12') +
  theme_bw() +
  theme(plot.title = element_text(hjust = 0.5),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())
var(train) # variance of original training data
var(train.d) # variance after differencing at lag 12

# difference at lag 1
train.dd <- diff(train.d, 1)

var(train.dd) # variance after differencing at lag 1

# histogram of original data
hist(train.d, main = "Histogram of Differenced Data", xlab = 'temperature_12',
     ylab = 'frequency', prob = TRUE, breaks = 15, cex.main = 0.85, cex.lab = 0.8, cex.axis = 0.7)
curve(dnorm(x, mean(train.d), sqrt(var(train.d))), add = TRUE, col = 'purple')

op <- par(mfrow = c(1,2))

# ACF
acf <- acf(train.d, lag.max = 40,
    ylim = c(-1,1),
    main = 'ACF for Differenced Data',
    plot = T)

# PACF
pacf <- pacf(train.d, lag.max = 40,
    ylim = c(-1,1),
    main = 'PACF for Differenced Data',
    plot = T)

# data frame to store AICc values with the corresponding p's and q's
aic <- data.frame(p = c(''), q = c(''), P = c(''), Q = c(''), num_coeffs = c(''), aicc = c(''))

# generate AICc for each candidate model
for(i in 0:1) {
  for(j in 1:2) {
    for(k in 1:2) {
      aic <- rbind(aic, c(1, i, j , k, 1 + i + j + k,
                          AICc(arima(train.d, order = c(1,0,i),
                                     seasonal = list(order = c(j,1,k), period = 12), method = 'ML'))))
    }
  }
}
```

```r
aic <- aic[order(aic$aicc),][-1,] # sort AICc in ascending order
rownames(aic) <- 1:nrow(aic) # re-index data frame
aic %>% pander() # print AICc's in nice table

# examine coefficients
for(i in 0:1) {
  for(j in 1:2) {
    for(k in 1:2) {
      print(paste('p = 1', 'q =', i, 'P =', j, 'Q =', k))
      print(arima(train.d, order = c(1,0,i),
                  seasonal = list(order = c(j,1,k), period = 12), method = 'ML'))
    }
  }
}

# coeff estimates for  p=1, q=0, P=2, Q=2
arima(train.d, order = c(1,0,0), seasonal = list(order = c(2,1,2), period = 12), method = 'ML')

# coeff estimates for  p=1, q=1, P=2, Q=2
arima(train.d, order = c(1,0,1), seasonal = list(order = c(2,1,2), period = 12), method = 'ML')

# coeff estimates for  p=1, q=1, P=1, Q=1
arima(train.d, order = c(1,0,1), seasonal = list(order = c(1,1,1), period = 12), method = 'ML')

# data frame to store AICc values with the corresponding p's and q's
aic_mod <- data.frame(p = c(''), q = c(''), P = c(''), Q = c(''), fixed = c(''),
                      num_coeffs = c(''), aicc = c(''))

# compute AICc for each modified model
aic_mod <- rbind(aic_mod, c(1, 0, 2 , 2, 'c(NA, 0, NA, NA, NA)', 4,
                            AICc(arima(train.d, order = c(1,0,0),
                                       seasonal = list(order = c(2,1,2), period = 12),
                                       fixed = c(NA, 0, NA, NA, NA), method = 'ML'))))
aic_mod <- rbind(aic_mod, c(1, 1, 1 , 1, 'c(0, 0, NA, NA)', 2,
                            AICc(arima(train.d, order = c(1,0,1),
                                       seasonal = list(order = c(1,1,1), period = 12),
                                       fixed = c(0, 0, NA, NA), method = 'ML'))))
aic_mod <- rbind(aic_mod, c(1, 1, 1 , 1, 'c(0, NA, NA, NA)', 3,
                            AICc(arima(train.d, order = c(1,0,1),
                                       seasonal = list(order = c(1,1,1), period = 12),
                                       fixed = c(0, NA, NA, NA), method = 'ML'))))
aic_mod <- rbind(aic_mod, c(1, 1, 1 , 1, 'c(NA, 0, NA, NA)', 3,
                            AICc(arima(train.d, order = c(1,0,1),
                                       seasonal = list(order = c(1,1,1), period = 12),
                                       fixed = c(NA, 0, NA, NA), method = 'ML'))))
aic_mod <- rbind(aic_mod, c(1, 1, 2 , 2, 'c(NA, NA, 0, NA, NA, NA)', 5,
                            AICc(arima(train.d, order = c(1,0,1),
                                       seasonal = list(order = c(2,1,2), period = 12),
                                       fixed = c(NA, NA, 0, NA, NA, NA), method = 'ML'))))

aic_mod <- aic_mod[order(aic_mod$aicc),][-1,] # sort AICc in ascending order
rownames(aic_mod) <- 1:nrow(aic_mod) # re-index data frame
aic_mod %>% pander() # print AICc's in nice table
```

```r
# estimate model 1 coeffs
arima(train.d, order = c(1,0,0), seasonal = list(order = c(2,1,2), period = 12), method = 'ML')

op <- par(mfrow = c(1,3))

# plot roots for model 1
source("plot.roots.R")
plot.roots(NULL, polyroot(c(-0.3042)), main = 'Roots of Non-Seasonal AR Part')
plot.roots(NULL, polyroot(c(0,0.0766,-0.2483)), main = 'Roots of Seasonal AR Part')
plot.roots(NULL, polyroot(c(-1.9461, 0.9998)), main = 'Roots of Seasonal MA Part')

# estimate model 2 coeffs
arima(train.d, order = c(1,0,0), seasonal = list(order = c(2,1,2), period = 12),
      fixed = c(NA, 0, NA, NA, NA), method = 'ML')

op <- par(mfrow = c(1,3))

# plot roots for model 2
source("plot.roots.R")
plot.roots(NULL, polyroot(c(-0.2979)), main = 'Roots of Non-Seasonal AR Part')
plot.roots(NULL, polyroot(c(0,-0.2833)), main = 'Roots of Seasonal AR Part')
plot.roots(NULL, polyroot(c(-1.9483, 1.0001)), main = 'Roots of Seasonal MA Part')
# estimate model 3 coeffs
arima(train.d, order = c(1,0,0), seasonal = list(order = c(1,1,2), period = 12), method = 'ML')

op <- par(mfrow = c(1,3))

# plot roots for model 3
source("plot.roots.R")
plot.roots(NULL, polyroot(c(-0.3083)), main = 'Roots of Non-Seasonal AR Part')
plot.roots(NULL, polyroot(c(0.369)), main = 'Roots of Seasonal AR Part')
plot.roots(NULL, polyroot(c(-1.639, 0.6397)), main = 'Roots of Seasonal MA Part')

# estimate model 4 coeffs
arima(train.d, order = c(1,0,1), seasonal = list(order = c(2,1,2), period = 12), method = 'ML')

op <- par(mfrow = c(1,2), mai = c(0.5,1.25,0.5,1.25))

# plot roots for model 4
source("plot.roots.R")
plot.roots(NULL, polyroot(c(-0.3943)), main = 'Roots of Non-Seasonal AR Part')
plot.roots(NULL, polyroot(c(0.0741, -0.2467)), main = 'Roots of Seasonal AR Part')
plot.roots(NULL, polyroot(c(-0.0999)), main = 'Roots of Non-Seasonal MA Part')
plot.roots(NULL, polyroot(c(-1.9452, 0.9991)), main = 'Roots of Seasonal MA Part')

# estimate model 5 coeffs
arima(train.d, order = c(1,0,1), seasonal = list(order = c(2,1,2), period = 12),
      fixed = c(NA, NA, 0, NA, NA, NA), method = 'ML')

op <- par(mfrow = c(1,2), mai = c(0.5,1.25,0.5,1.25))

# plot roots for model 5
source("plot.roots.R")
```

```r
plot.roots(NULL, polyroot(c(-0.4012)), main = 'Roots of Non-Seasonal AR Part')
plot.roots(NULL, polyroot(c(0, -0.2801)), main = 'Roots of Seasonal AR Part')
plot.roots(NULL, polyroot(c(-0.1138)), main = 'Roots of Non-Seasonal MA Part')
plot.roots(NULL, polyroot(c(-1.9479, 1)), main = 'Roots of Seasonal MA Part')

# estimate model 6 coeffs
arima(train.d, order = c(1,0,1), seasonal = list(order = c(1,1,2), period = 12), method = 'ML')

op <- par(mfrow = c(1,2), mai = c(0.5,1.25,0.5,1.25))

# plot roots for model 6
source("plot.roots.R")
plot.roots(NULL, polyroot(c(-0.4106)), main = 'Roots of Non-Seasonal AR Part')
plot.roots(NULL, polyroot(c(0.3667)), main = 'Roots of Seasonal AR Part')
plot.roots(NULL, polyroot(c(-0.1132)), main = 'Roots of Non-Seasonal MA Part')
plot.roots(NULL, polyroot(c(-1.6375, 0.6375)), main = 'Roots of Seasonal MA Part')

# model 3 residuals
fit3 <- arima(train.d, order = c(1,0,0), seasonal = list(order = c(1,1,2), period = 12),
              method = 'ML')
res3 <- residuals(fit3)
# mean(res3)
# var(res3)

layout(matrix(c(1,1,2,3),2,2,byrow = T))

# plot residuals
ts.plot(res3, main = "Fitted Residuals for Model 3", ylab = 'residual', xlab = 'time')
t <- 1:length(res3)
fit.res3 <- lm(res3~t)
abline(fit.res3)
abline(h = mean(res3), col = "red")

# histogram of residuals
hist(res3, main = "Histogram of Residuals for Model 3", xlab = 'residual',
     ylab = 'frequency', prob = TRUE, breaks = 10)
curve(dnorm(x, mean(res3), sqrt(var(res3))), add = TRUE, col = 'purple')

# q-q plot of residuals
qqnorm(res3)
qqline(res3, col ="blue", main = 'Normal Q-Q Plot for Model 3')

op <- par(mfrow = c(1,2))

# acf of the residuals
acf <- acf(res3, main = "ACF of Model 3 Residuals", plot = T)

# pacf of the residuals
pacf <- pacf(res3, main = "PACF of Model 3 Residuals", plot = T)

shapiro.test(res3) # shapiro wilk
Box.test(res3, lag = 15, type = c('Box-Pierce'), fitdf = 4) # box-pierce
Box.test(res3, lag = 15, type = c('Ljung-Box'), fitdf = 4) # ljung-box
```

```r
Box.test(res3^2, lag = 15, type = c('Ljung-Box'), fitdf = 0) # mc-leod li

# fit residuals to AR model
ar(res3, aic = TRUE, order.max = NULL, method = c("yule-walker"))

# model 6 residuals
fit6 <- arima(train.d, order = c(1,0,1), seasonal = list(order = c(1,1,2), period = 12), method = 'ML')
res6 <- residuals(fit6)
# mean(res3)
# var(res3)

# plot residuals
ts.plot(res6, main = "Fitted Residuals for Model 6", ylab = 'residual', xlab = 'time')
t <- 1:length(res6)
fit.res6 <- lm(res6~t)
abline(fit.res6)
abline(h = mean(res6), col = "red")

op <- par(mfrow = c(1,2))

# histogram of residuals
hist(res6, main = "Histogram of Residuals for Model 6", xlab = 'residual',
     ylab = 'frequency', prob = TRUE, breaks = 10)
curve(dnorm(x, mean(res6), sqrt(var(res6))), add = TRUE, col = 'purple')

# q-q plot of residuals
qqnorm(res6)
qqline(res6, col ="blue", main = 'Normal Q-Q Plot for Model 6')

op <- par(mfrow = c(1,2))

# acf of the residuals
acf <- acf(res6, main = "ACF of Model 6 Residuals", plot = T)

# pacf of the residuals
pacf <- pacf(res6, main = "PACF of Model 6 Residuals", plot = T)

shapiro.test(res6) # shapiro-wilk
Box.test(res6, lag = 15, type = c('Box-Pierce'), fitdf = 5) # box-pierce
Box.test(res6, lag = 15, type = c('Ljung-Box'), fitdf = 5) # ljung-box
Box.test(res6^2, lag = 15, type = c('Ljung-Box'), fitdf = 0) # mc-leod li

# fit residuals to AR model
ar(res6, aic = TRUE, order.max = NULL, method = c("yule-walker"))

# fit model 3 to training data
fit3 <- arima(train, order = c(1,0,0), seasonal = list(order = c(1,1,2), period = 12),
              method = 'ML')

# forecast 12 future observations
forecast(fit3)

# predict 12 future observations
```

```r
mypred <- predict(fit3, n.ahead = 12)
pred.df <- data.frame(date = temp.csv[229:240,]$date,
                      pred = mypred$pred,
                      se = mypred$se)

# plot original training data with forecast
ggplot(temp.csv[1:228,], aes(x = date, y = temp)) +
  geom_line(lwd = 0.35) +
  geom_point(data = pred.df, mapping = aes(x = date, y = pred), col = 'red', shape = 1) +
  geom_line(data = pred.df, mapping = aes(x = date, y = pred+1.96*se),
            col = 'blue', linetype = 'dashed', lwd = 0.35) +
  geom_line(data = pred.df, mapping = aes(x = date, y = pred-1.96*se),
            col = 'blue', linetype = 'dashed', lwd = 0.35) +
  ggtitle('Monthly Temperatures (°F) in Nottingham (1920-1938) + Forecast 12 Future Observations') +
  ylab('temperature') +
  theme_bw() +
  theme(plot.title = element_text(hjust = 0.5, size = 10),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())

# zoomed in forecast plot
ggplot(temp.csv[193:228,], aes(x = date, y = temp)) +
  geom_line(lwd = 0.35) +
  geom_point(data = pred.df, mapping = aes(x = date, y = pred), col = 'red', shape = 1) +
  geom_line(data = pred.df, mapping = aes(x = date, y = pred+1.96*se),
            col = 'blue', linetype = 'dashed', lwd = 0.35) +
  geom_line(data = pred.df, mapping = aes(x = date, y = pred-1.96*se),
            col = 'blue', linetype = 'dashed', lwd = 0.35) +
  ggtitle('Monthly Temperatures (°F) in Nottingham (1936-1938) + Forecast') +
  ylab('temperature') +
  theme_bw() +
  theme(plot.title = element_text(hjust = 0.5, size = 10),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())

# zoomed in forecast plot with test data
ggplot(temp.csv[193:240,], aes(x = date, y = temp)) +
  geom_line(lwd = 0.35) +
  geom_point(data = pred.df, mapping = aes(x = date, y = pred), col = 'red', shape = 1) +
  geom_line(data = pred.df, mapping = aes(x = date, y = pred+1.96*se),
            col = 'blue', linetype = 'dashed', lwd = 0.35) +
  geom_line(data = pred.df, mapping = aes(x = date, y = pred-1.96*se),
            col = 'blue', linetype = 'dashed', lwd = 0.35) +
  ggtitle('Monthly Temperatures (°F) in Nottingham (1936-1939) With Forecast') +
  ylab('temperature') +
  theme_bw() +
  theme(plot.title = element_text(hjust = 0.5, size = 10),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())
```