Forecasting The Next Solar Maximum

Judith Victoria, Aaron Jones, & Greg Johnson

30 June 2016

Introduction

A sunspot is a temporary dark (as in color) spot on the surface of the sun that is associated with lower temperatures. These dark spots result from high concentrations of magnetic flux that interfere with convention, which is essentially the process of energy transportation. Sunspots are of interest because they are related to many other solar phenomena, including solar flares. A solar cycle is approximately 11 years, which we confirmed when plotting the data. However, as became clear to us as we worked through the various analyses, the word approximately had a huge impact. The inconsistency (non-perfection) of the seasonal component caused problems in modeling fitting, residual diagnostics, and in forecasting. The "solar maximum" and "solar minimum" are respectively the maximum and minimum number of sunspots in one solar cycle. Our goal is to predict the monthly solar maximum value and time, which would be the month and year that will have the highest mean number of sunspots during an 11 year solar cycle.

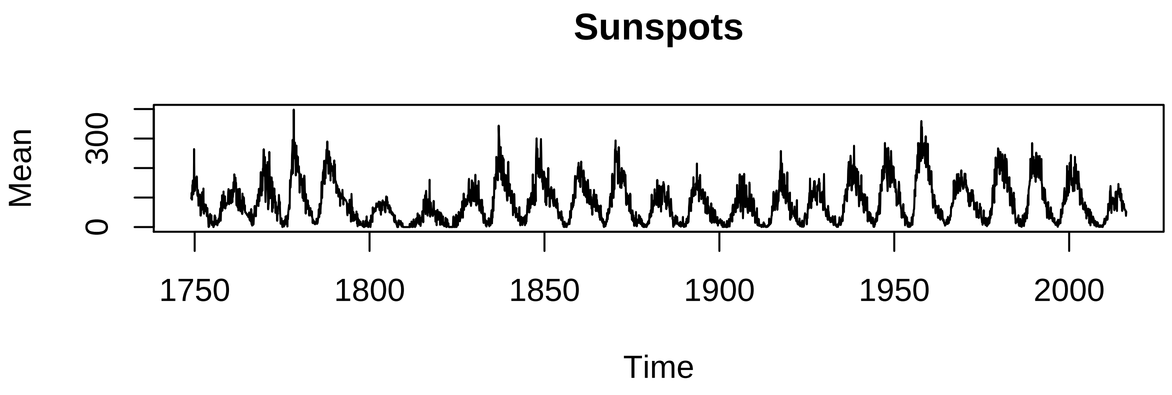
The data we have chosen to analyze comes from the Sunspot Index and Long-Term Solar Observations website. We are looking at the mean monthly number of sunspots running from January 1749 to May 2016. While there are several advanced quantitative methods for counting the number of sunspots, this data was collected by astronomers manually counting the number of sunspots. This methodology was employed because the more quantitative methodologies are fairly new (within the last few decades) whereas scientisits have been counting sunspots by eye since Galileo. Thus, in order to utilize data going back centuries and to maintain legitimacy, the Sunspot Index has continued to use the more primitive methodology. Obviously, though, manually counting has some inherent problems, the least of which is simply counting incorrectly. Although we cannot verify this fact, it seems possible that some of the inconsistency in the seasonal component might be the result of counting errors.

As far as how we divided the project up, Judith took the lead on regression, Aaron took the lead on Holt-Winters smoothing, and Greg took the lead on Box-Jenkins. The writing of the report and the development of the presentation were joint efforts.

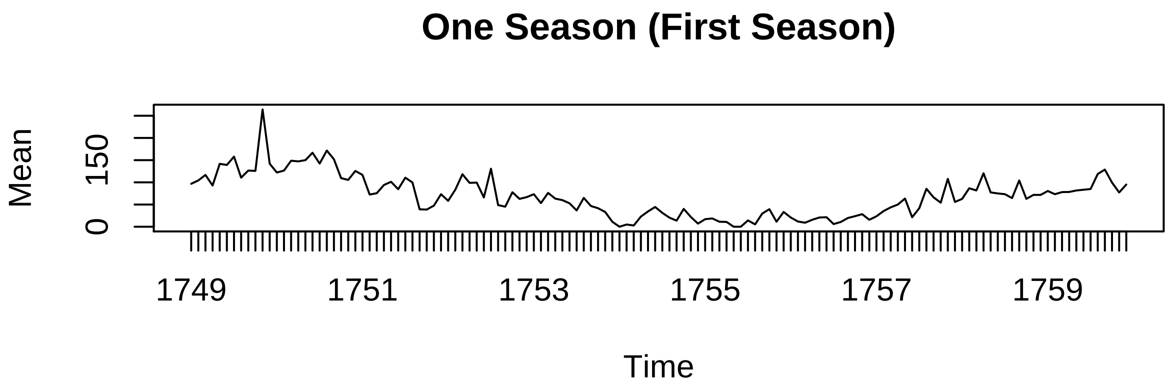
Exploratory Analysis

To begin, we performed some exploratory analyses, including some basic plotting in order to get a better understanding of the data.

```r
plot(sunspots,type = 'l',main = 'Sunspots',xlab = 'Time',ylab = 'Mean')
```

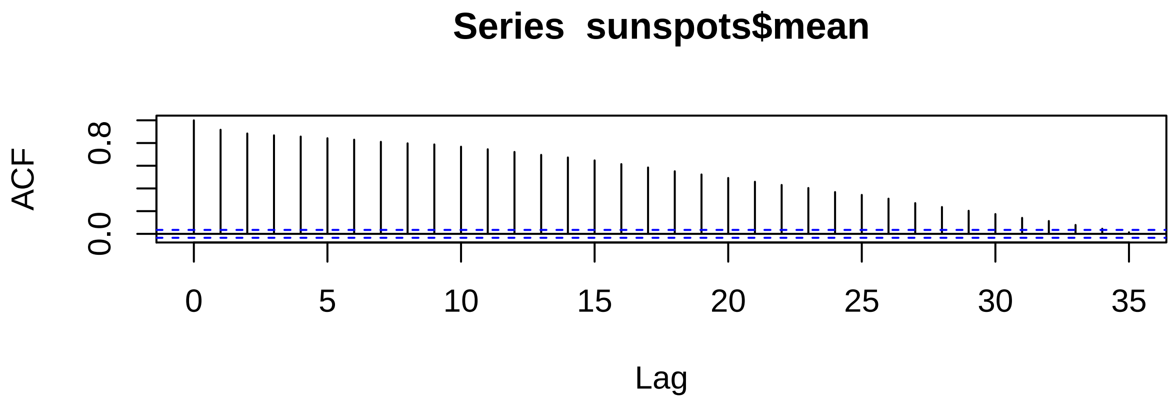


```r
plot(sunspots[1:132,],type='l',main='One Season (First Season)',xlab = 'Time',ylab = 'Mean')
```

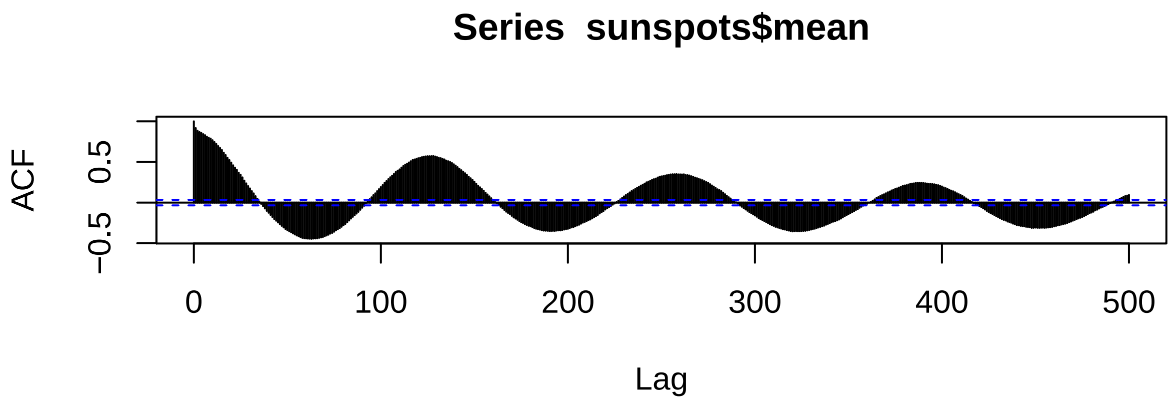


The first plot is of the complete data. From the visualization, we clearly see the inconsistencies in the seasonal componenet. It is not consistent in length or amplitude. The second plot is of one 11 year season (132 data points).

```r
acf(sunspots$mean)
```

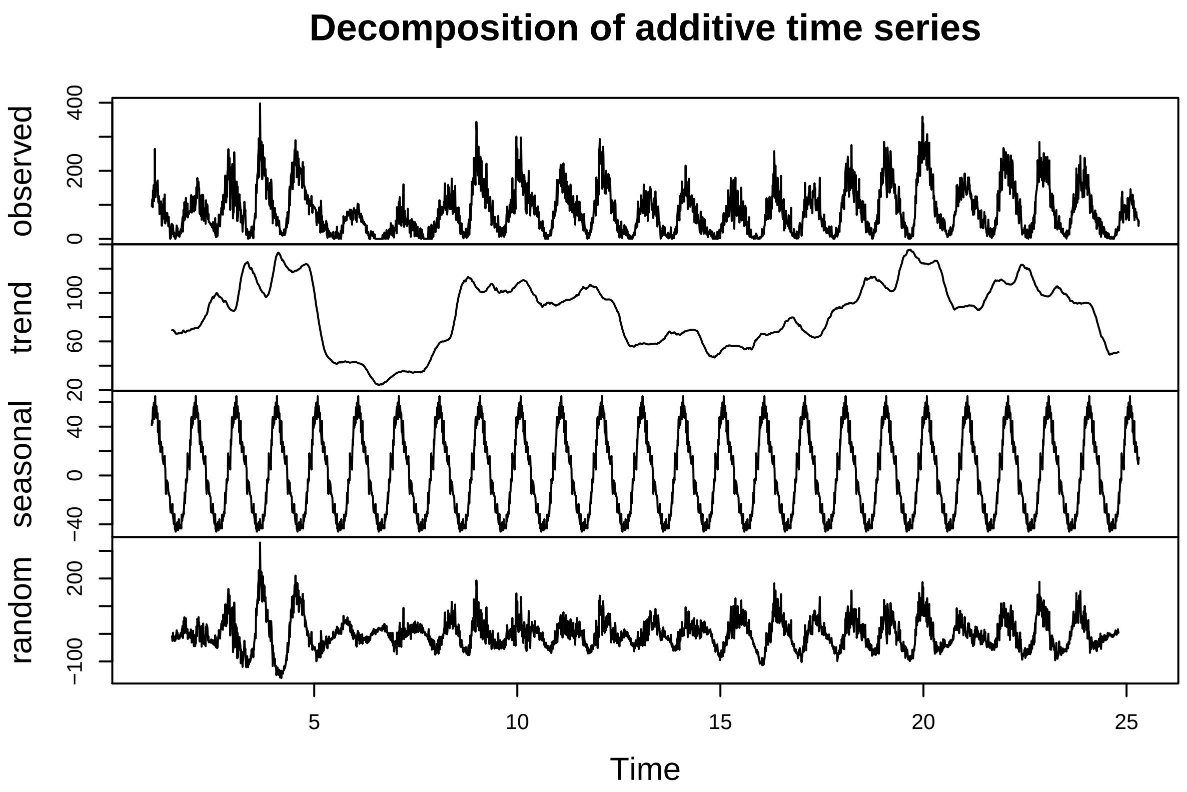


```r
acf(sunspots$mean,lag.max=500)
```



Here, we've plotted the autocorrelation function of the data. The first plot is the R default acf plot. The second plot goes up to lag 500. We extended the lag in order to see the seasonality in the data. The autocorrelation function shows that this data is not stationary because there is a clear seasonal component.

```r
plot(decompose(full.ts))
```



The final plot of our exploratory analysis is the decomposition of the data. What really sticks out of this series of plots is that the random component is dominating both the trend and seasonal components. The random component ranges from -100 to 300 while the trend component ranges from 20 to 140 and the seasonal component ranges from -40 to 60. The domination by the random component provides some explanation for the problems we faced when trying to fit various models to the data.

In order to perform the analyses, we split the data into training and validations sets. The training was all the data except for the last 65 points, which were defined as the validation set. Because of the inconsistent seasonal component, we also tried shortening the training data in an effort to get rid of some of the fluctuations in seasonality. Included in this shortened training data were the last 12 seasons of the full training data.

We begin with regression.

Regression

We approach regression two different ways, we tried fitting models using all of our seasons, but we also tried fitting models only considering the 12 most recent seasons. Since we had a lot of data, we tried fitting models both ways to see which model gave us a smallest SSPr and thus provide a better prediction on the next solar maximum. So the first model that we tried was a full model, which include up to degree 7 along with a seasonal component. However, when we calculated its AIC, it wasn't able to calculate the values for degree 3 or above because there was some correlation between them. So the other models we fitted were with no more than degree 2. That being said, our second model was with a quadratic trend and seasonal component. We consider doing models based on all of our seasons, but we were not getting a great prediction. So we decided to focus on the 12 most recent seasons to see how the SSPr changed. This wouldn't cause any complications with our prediction since we would still have enough history to do analysis. Our period is roughly 11 years and in each season we had 132 points, so there was enough data into the history, thus, we decided to not look before that. The other model was based only on the 12 most recent seasons. For this third model we considered again a quadratic trend along with a seasonal component. Since this model had non-constant variance, the next model we proposed was a log transformation model. We wanted to see how it would affect $SSPr$. It turned out that the log transformation fixed the issue of non-constant variance, and it still had smaller SSPr compared to quadratic and seasonal model; however, it wasn't the model with the smallest SSPr.

All the models that we tried in regression were not good in terms of fit. However, below we show the results for fitting $X_t = \beta_0 + Season + e$. This model is not the best fit, but it has the smallest SSPr compare to the rest.

```r
model6=lm(training2~season2)
season.new <- factor(rep(1:132,25))[3145:3209]
data.new <- data.frame(season2=season.new)
pred6 <- predict.lm(model6,data.new,interval="prediction")
pred6 <- pred6[,1] #extracting the predicted values
sum((validation-pred6)^2) #calculating SSpr
```

```
## [1] 193452.4
```

```r
model2=lm(training~tim+tim2+season)
tim.1new <- time(validation) #65 points in my validation
tim2.1new <- tim.1new^2;
season.new <- factor(rep(1:132,25))[3145:3209]
data.new <- data.frame(tim=tim.1new,tim2=tim2.1new,season=season.new)
pred2 <- predict.lm(model2,data.new,interval="prediction")
pred2 <- pred2[,1] #extracting the predicted values
sum((validation-pred2)^2) #calculating SSpr
```

```
## [1] 7.338912e+12
```

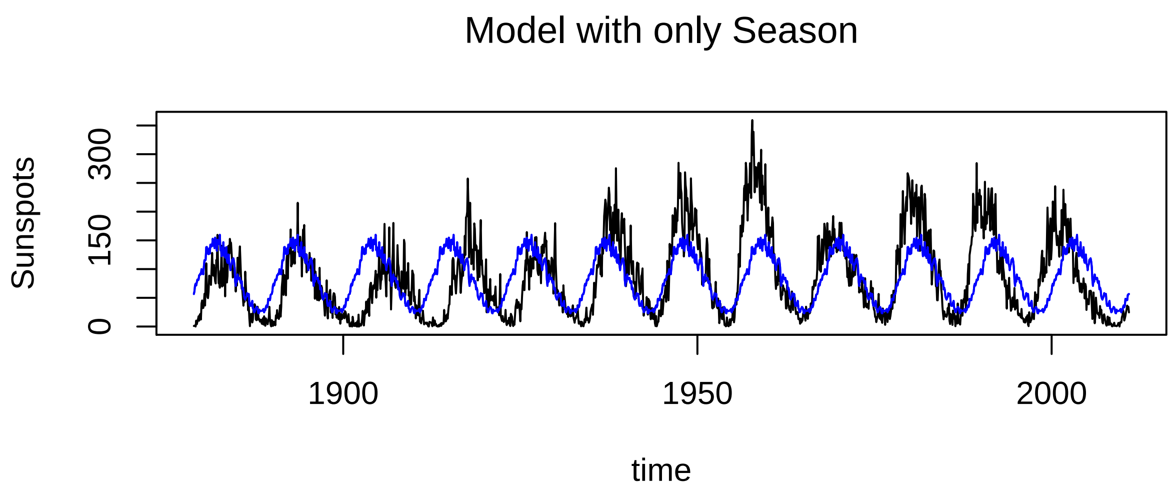
```r
model4=lm(training2~tim.2+tim2.2+season2)
timm.new <- time(validation)
timm2.new <- timm.new^2
season.new <- factor(rep(1:132,25))[3145:3209]
data.new <- data.frame(tim.2=timm.new,tim2.2=timm2.new,season2=season.new)
pred4 <- predict.lm(model4,data.new,interval="prediction")
pred4 <- pred4[,1]
sum((validation-pred4)^2)
```

```
## [1] 7.81559e+14
```

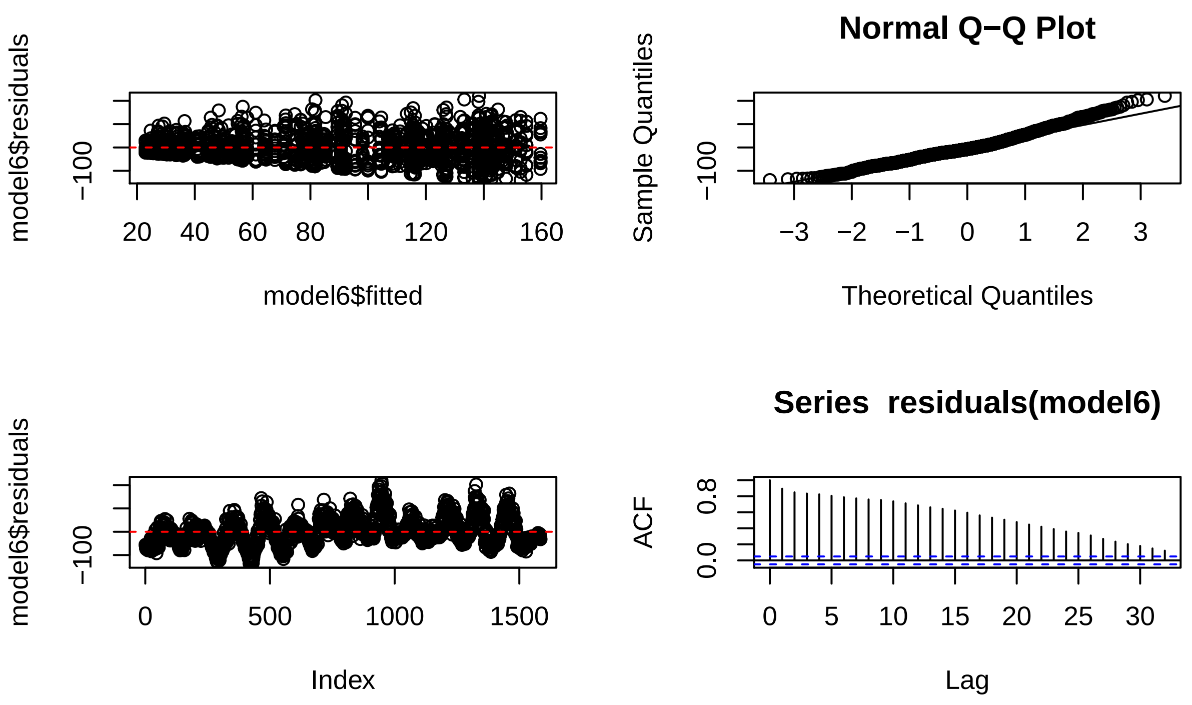
```
t.training=log(training2+1)
model5=lm(t.training~tim.2+tim2.2+season2)
data.new <- data.frame(tim.2=timm.new ,tim2.2=timm.new ,season2=season.new)
pred5 <- exp(predict.lm(model5,data.new,interval="prediction"))
pred5 <- pred5[,1]
sum((validation-pred5)^2)
```

```
## [1] 522286.9
```

```
plot(sunspots[["date"]][1560:3144], training2 , type="l", main=expression
      ("Model with only Season"),xlab="time",ylab="Sunspots")
points(sunspots[["date"]][1560:3144],predict.lm(model6),type="l",col="blue")
points(sunspots[["date"]][1560:3144],fitted(model6),col="blue",type='l')
```

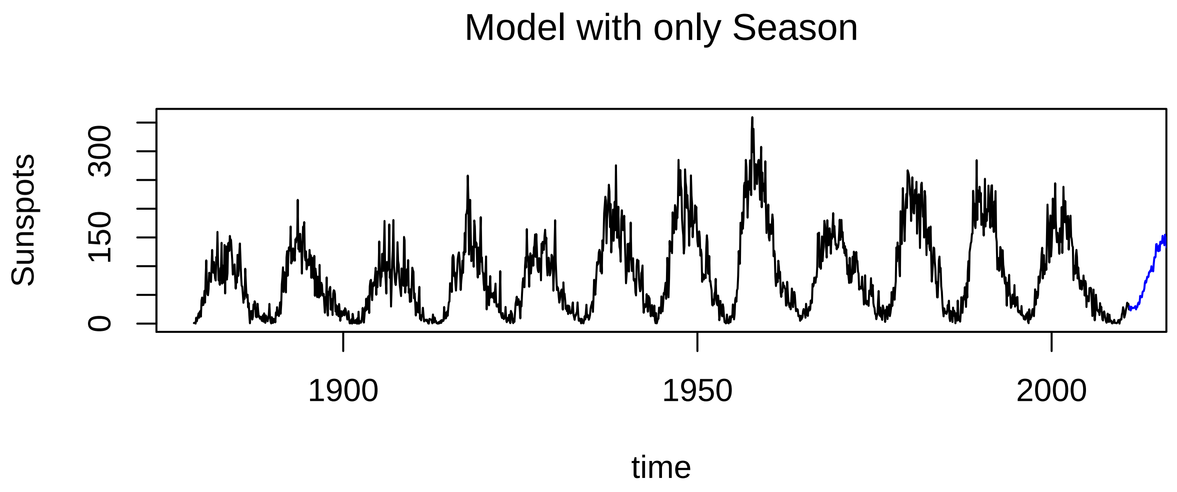


```
par(mfrow=c(2,2))
plot(model6$fitted, model6$residuals)
abline(h=0,lty=2,col="red")
qqnorm(model6$residuals)
qqline(model6$residuals)
plot(model6$residuals)
abline(h=0,lty=2,col="red")
acf(residuals(model6))
```



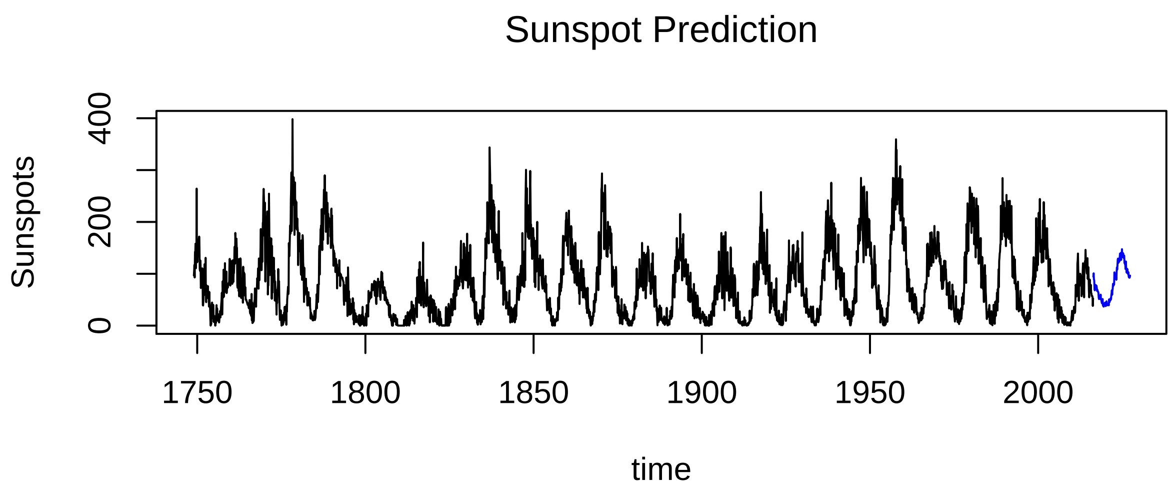
We can see that this model does not have the best fit nor the best diagnostics. We can see from the first plot in the diagnostics that we don't have constant variance, normality doesn't seem too bad, but there is an obvious problem with our residuals. The problem is that we have seasonality in the residuals, but we weren't able to fix that doing regression. Since our interest of our project was not to come up with a prediction interval for the solar maximum, but rather to predict the solar maximum, we were still able to use this model for that. Nonetheless, although this model didn't provide the best fit, it did have the "best" predicting power.

```r
plot(sunspots[["date"]][1560:3144], training2 , type="l", main=expression
     ("Model with only Season"),xlab="time",ylab="Sunspots")
points(sunspots[["date"]][3145:3209],pred6,col="blue",type='l')
```



Because the seasonal component model provided us with the best predicting power, we went back to our original data to fit a model with just a seasonal component and then used it to predicted the next solar maximum.

```
plot(newdata$d[1:3209], mean.sun , type="l", main=expression
     ("Sunspot Prediction"),xlim=c(1749,2027),xlab="time",ylab="Sunspots")
points(newdata$d[3210:dim(newdata)[1]],newdata$mean[3210:dim(newdata)[1]],col="blue",type='l')
```



It turns out the the next solar maximum will be 147.412 in December 2024. Last minute we thought of trying to do a rolling average in order to smooth out some of the variation and then fitting the rolling averages. We think this might have helped with forescating, but we didn't have enough time to do so.

Holt-Winters

Next, we attempted to fit Holt-Winters models to the sunspot data. Forseeing potential problems resulting from a lengthy data set and an inconsistent periodic component, we ran each one of the four Holt-Winters models with the full training data and the abbreviated training data, which includes the last 12 seasons of the full training data. Listed below are the SS_{pr} values for each of the eight models, and the plots and residuals diagnostics for the model that has the lowest SS_{pr}, which is the additive model using the complete training data. What the residual diagnostics show for the additive, complete training data model as well as for the other 7 models (whose residual diagnostics are in the appendix) is that each of the models is bad. Each model violates more the one of the fundamental assumptions. In general, the residuals are correlated, non-normal, and feature some sort of trend. There were some challenges in computing these models, which will be discussed as we move through the models.

Exponential

```
exp <- HoltWinters(sunspots.ts,gamma=F,beta=F)
exp.predict <- predict(exp,n.ahead=65)
exp.sspr <- sum((validation$mean - exp.predict)^2)
exp.sspr
```

```
## [1] 255537.2
```

Even though we don't yet have another Holt-Winters models to compare this predicted sum of squares to, it seems large. The SS_{pr} is 255,537.2. We assume that the large sum of squares predicted is the result of an inconsistent period component and the limitation of the exponential smoothing model.

Double Exponential

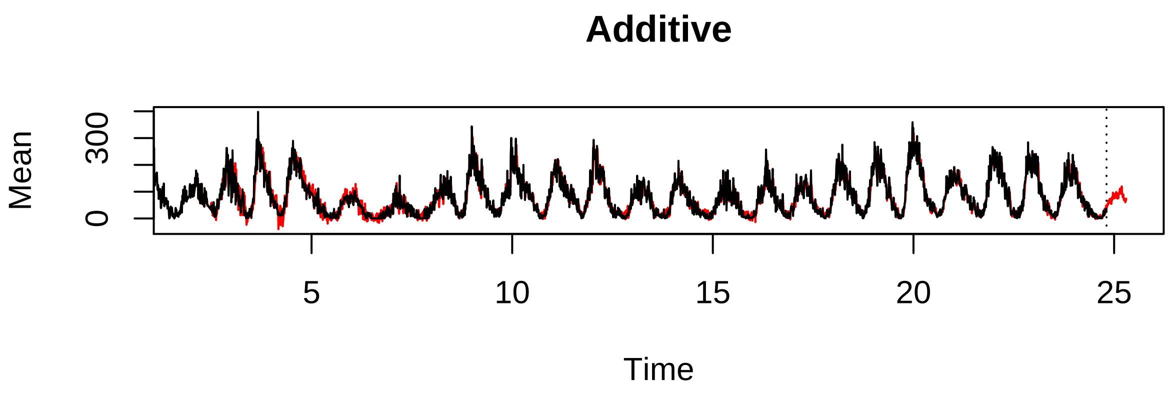
```
dexp <- HoltWinters(sunspots.ts,gamma=F)
dexp.predict <- predict(dexp,n.ahead=65)
dexp.sspr <- sum((validation$mean - dexp.predict)^2)
dexp.sspr
```

```
## [1] 345068.3
```

By applying the double exponential model, the predictions gain linearity, but still lack any seasonality. The SS_{pr} is again quite large at 345,068.3. So far, the exponential model would be better for making predictions.

Additive

```
add <- HoltWinters(sunspots.ts,seasonal='additive')
plot(add,predict(add,n.ahead=65),type='l',main = 'Additive',ylab='Mean')
```



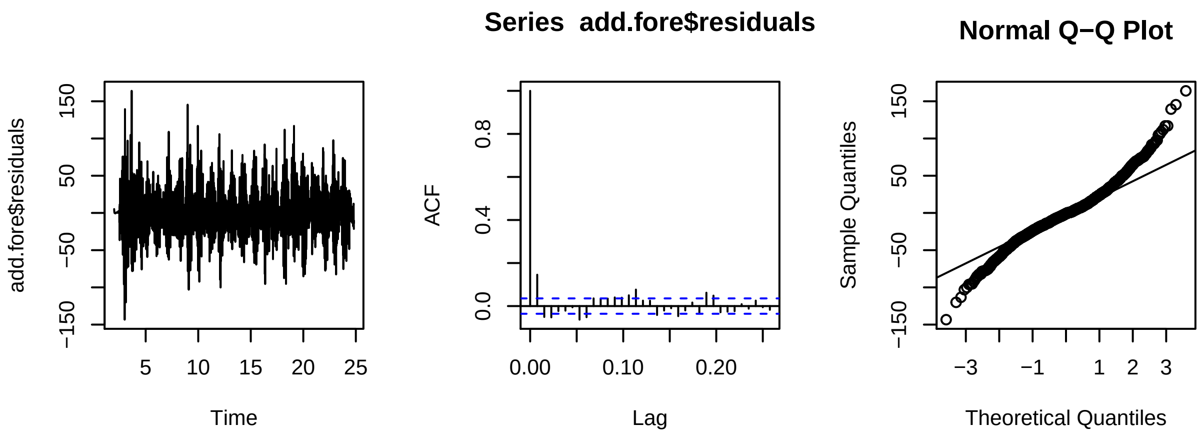
We are going to explore this additive model in more deal, since it is the one with the smallest sum of squares predicted. With the periodicity included, in this case added, our predictions seem slightly more reasonable visually too.

```
add.predict <- predict(add,n.ahead=65)
add.sspr <- sum((validation$mean - add.predict)^2)
add.sspr
```

```
## [1] 45444.55
```

As the visualization suggests, our predictions are better than when using either of the two previous models. Using the additive model, the SS_{pr} equals 45,444.55. This predicted sum or squares is significantly lower than the ones seem previously.

```
add.fore <- forecast.HoltWinters(add,h=65)
par(mfrow=c(1,3))
plot.ts(add.fore$residuals)
acf(add.fore$residuals)
qqnorm(add.fore$residuals)
qqline(add.fore$residuals)
```



```
Box.test(add.fore$residuals,type='Ljung-Box')
```

```
##
##  Box-Ljung test
##
## data:  add.fore$residuals
## X-squared = 64.07, df = 1, p-value = 1.221e-15
```

From the residual diagnostics, it is clear that even our best model, in terms of sum of squares predicted, is a bad model best it violates the fundamental assumptions. If the first plot were stretched out a bit more there would be an obvious trend to the residuals. In addition, the residuals are clearly correlated and non-normal. The lack of normality is established two ways: the first is visually with a QQ-plot and the second is the Ljung-Box test, which returned a small p-value suggesting a lack of normality.

Multiplicative

```r
training.plusone <- sapply(training$mean,function(x) x+1) #Adding 1.
sunspotsplusone.ts <- ts(training.plusone,start=1,frequency=12*11)
multi <- HoltWinters(sunspotsplusone.ts,seasonal='multiplicative')
multi.predict <- predict(multi,n.ahead=65)
multi.predict <- sapply(multi.predict,function(x) x-1) #Removing 1.
multi.sspr <- sum((validation$mean - multi.predict)^2)
multi.sspr
```

```
## [1] 6205204
```

Here, the periodicity of the data was included via multiplication. Despite including the periodicity, the predictions are, compared to the previous three Holt Winters models, not good. The multiplicative Holt-Winters model was a bit more difficult to execute. In order to use the multiplicative model, the data needs to be strictly positive; however, that is not the case with our sunspot data. We have zeros. So, to make the data strictly positive, we added one to each data point. Once the predictions were computed, we subtracted the one that was added and then computed the sum of squares predicted. The one needed to be subtracted out in order to get a valid sum of squares predicted. As we mentioned above, despite including, by multiplication, the seasonal component, the sum of squares predicted is very large. The SS_{pr} is 6,205,204.

We also ran these four models using the abbreviated training data, which, as was mentioned earlier, is only the last twelve seasons of the full training data. However, while the sum of squares predicted did decrease, substantially in some cases, none of the models using this reduced training data produced a sum of squares predicted smaller than that of the additive method with full training data. The recomputed models also did not fair any better in terms of residual diagnostics.

Exponential (Abbreviated Training Data)

```r
exp2 <- HoltWinters(sunspots.ts2,gamma=F,beta=F)
exp.predict2 <- predict(exp2,n.ahead=65)
exp.sspr2 <- sum((validation$mean - exp.predict2)^2)
exp.sspr2
```

```
## [1] 256372.3
```

Double Exponential (Abbreviated Training Data)

```r
dexp2 <- HoltWinters(sunspots.ts2,gamma=F)
dexp.predict2 <- predict(dexp2,n.ahead=65)
dexp.sspr2 <- sum((validation$mean - dexp.predict2)^2)
dexp.sspr2
```

```
## [1] 240727.4
```

Additive (Abbreviated Training Data)

```r
add2 <- HoltWinters(sunspots.ts2,seasonal='additive')
add.predict2 <- predict(add2,n.ahead=65)
add.sspr2 <- sum((validation$mean - add.predict2)^2)
add.sspr2
```

```
## [1] 176423.9
```

Multiplicative (Abbreviated Training Data)

```r
training.plusone2 <- sapply(training2$mean,function(x) x+1) #add one because the data had zeros
sunspotsplusone.ts2 <- ts(training.plusone2,start=1,frequency=12*11)
multi2 <- HoltWinters(sunspotsplusone.ts2,seasonal='multiplicative',optim.start=c(alpha=1,beta=1,gamma=
multi.predict2 <- predict(multi2,n.ahead=65)
multi.predict2 <- sapply(multi.predict2,function(x) x-1)
multi.sspr2 <- sum((validation$mean - multi.predict2)^2)
multi.sspr2
```

```
## [1] 492631.7
```

We ran into some problems when trying to compute the multiplicative, abbreviated training data model. When we initially tried running computing this Holt-Winters model, R returned an error that essentially said the process of optimizing the parameters of the model failed. We believe this problem resulted from the shortened data since this was not an issue when using the full training data. Using this realization, we added a few more points to the abbreviated data set, but that did not work. Next we tried adjusting the optimization method and/or initial values. When the initial values were set to $\alpha = \beta = \gamma = 1$, the parameters converged and the model computed successfully. However, looking at the visualization of the model, there still seems to be an issue with this model that we cannot at this point solve.

Here is a table of all the sums of squares predicted for easy comparison.

```r
sspr <- rbind(exp.sspr,dexp.sspr,add.sspr,multi.sspr,
              exp.sspr2,dexp.sspr2,add.sspr2,multi.sspr2)
rownames(sspr) <- c('Exponential','Double Exponential','Additive','Multiplicative',
                    'Exponential - 12 Seasons','Double Exponential - 12 Seasons',
                    'Additive - 12 Seasons','Multiplicative - 12 Seasons')
colnames(sspr) <- 'Sum of Squares Predicted'
sspr
```

```
##                                 Sum of Squares Predicted
## Exponential                                    255537.23
## Double Exponential                             345068.31
## Additive                                        45444.55
## Multiplicative                                6205203.67
## Exponential - 12 Seasons                       256372.35
## Double Exponential - 12 Seasons                240727.44
## Additive - 12 Seasons                          176423.87
## Multiplicative - 12 Seasons                    492631.65
```

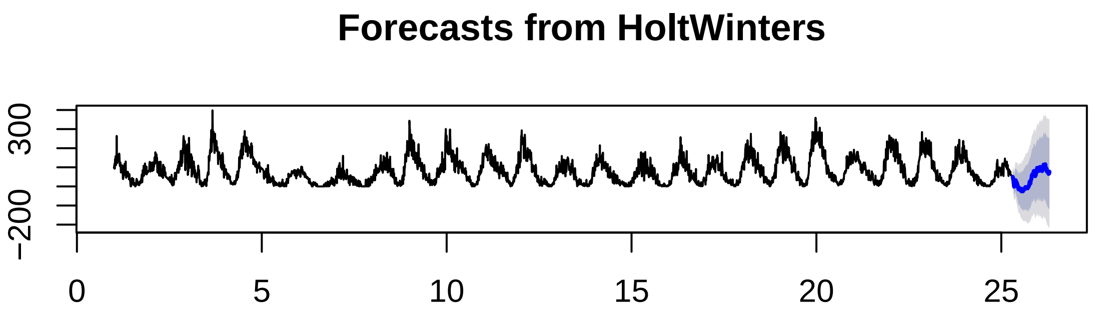
We also decided to forecast using the Holt-Winters additive model featuring the full data, despite the fact that it failed the fundamental assumptions.

Forecasting

```r
full.ts <- ts(sunspots$mean,start=1,frequency=12*11)
add.refit <- HoltWinters(full.ts,seasonal='additive')
forecast <- predict(add.refit,n.ahead=11*12)
forecast.max=max(forecast)
forecast.max
```

```
## [1] 116.0764
```

```r
forecastwithint <- predict(add.refit,n.ahead=11*12,prediction.interval=TRUE)
forecast.plot=forecast.HoltWinters(add.refit,h=11*12)
plot(forecast.plot)
```



The forecasting of one solar cycle (132 months or ll years) produces some interesting revelations. According to this model, the next maximum mean is 116.0764 and that maximum will occur February of 2026. One indication that this model might not be the best for forecasting is that some of the forecasted averages are negative numbers, which is not possible. Finally, we plotted the forecasted values above with prediction intervals despite knowing that the model violated the fundamental assumptions and is therefore not an appropriate model for doing anything other than producing predictions. However, out of curiousity we created the prediction intervals and were struck by the size of the intervals. We will not comment anymore on the intervals since we know that the intervals are not valid.

Box-Jenkins

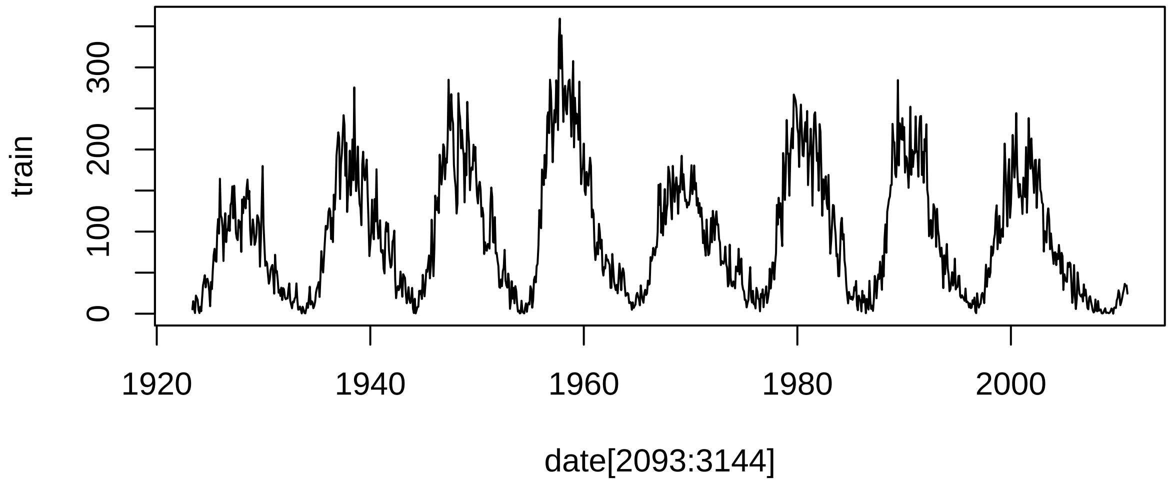
Box-Jenkins was the last class of models we attempted to fit to the data. Because a vital part of specifying SARIMA models is correctly specifying the seasonal components, it's important to have stable estimates of the large-lage autocorrelations. However Box-Jenkins models with seasonal components applied to large amounts of data can take unreasonable amounts of time to compute (especially in the context of this summer class). Thus we struck a balance between the two by retaining 8 seasons (88 years) of our data.

As an additional technical aside, while we explored many SARIMA models, we only report on one here again because of the lengthy computation time that these models ended up requiring. SARIMA's took anywhere from 40 minutes to two hours, so in the interest of keeping the knitting of this document under two hours, we elected to only explore the one model.

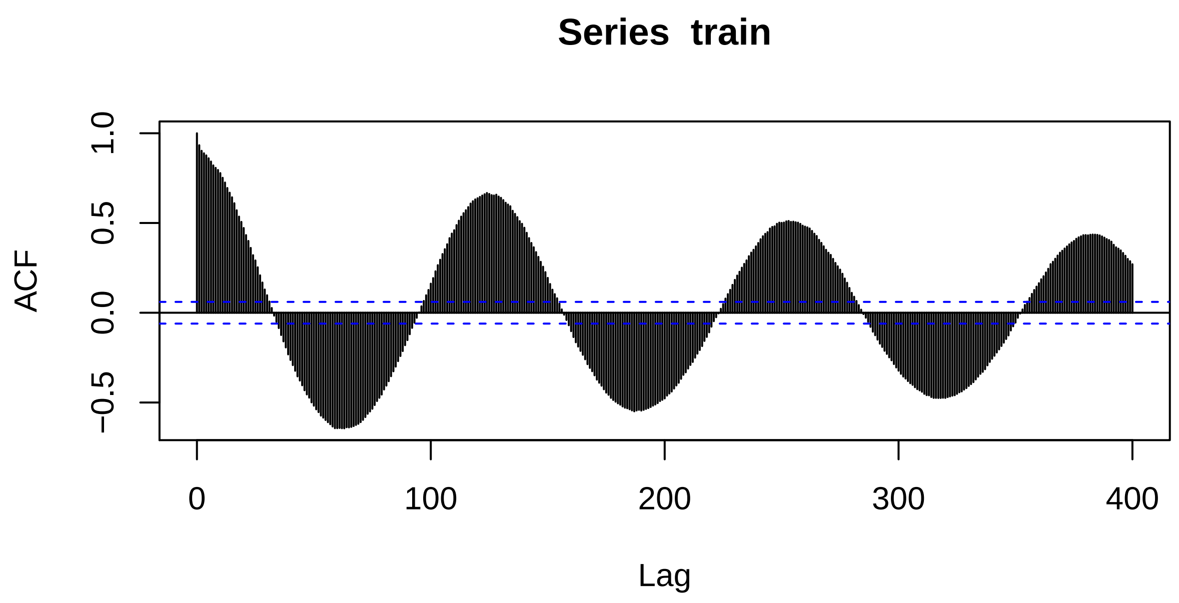
Achieving Stationarity

The first step in the Box-Jenkins model-building process is achieving stationarity in the data through differencing i.e. choosing d, D, and S. We want to remove any slow-changing trend in the mean or any seasonal component. Just from domain knowledge we know that our data is in fact seasonal. Since it's seasonal, it's nonstationary. We can see this in the ACF and PACF plots of the data:

```r
plot(date[2093:3144],train,type="l")
```



```r
acf(train,lag=400) #obvious seasonality
```

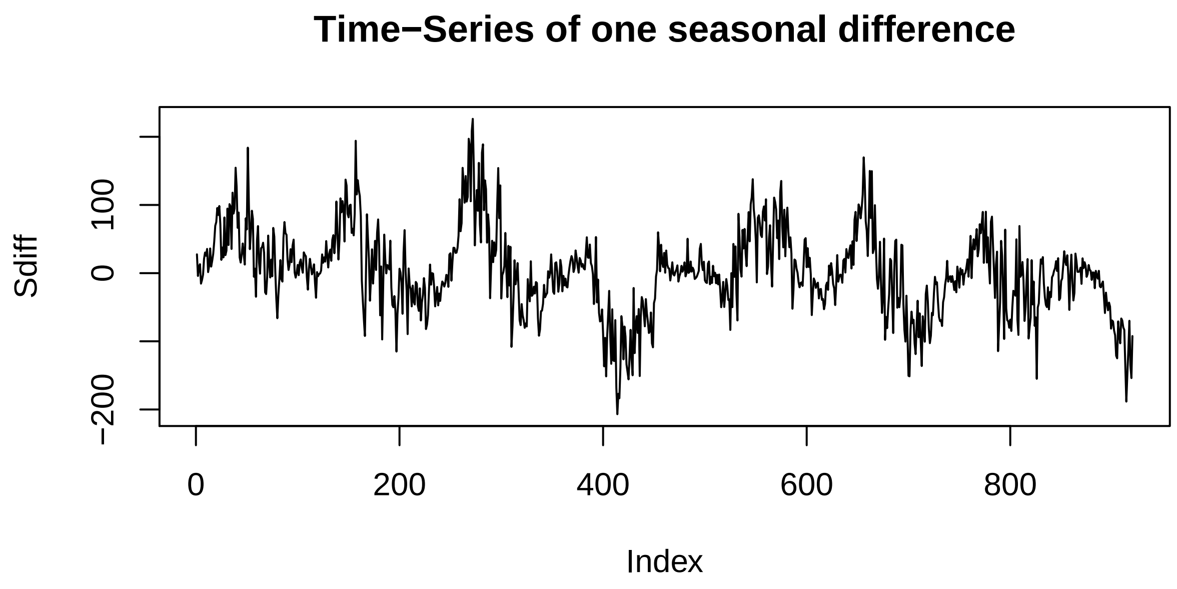


Domain knowledge also tells us that there is no slow-change trend in the mean so we shouldn't have to worry about first-order differencing. We do, however, have to difference at the lag equal to the length of the season of our data. Fortunately we know from the literature that the average period is 11 years and since our data is monthly, one period consists of 132 data points. We can confirm this by looking back at the ACF plot and noticing that the period of the autocorrelation function is ~130. Theoretical and empirical evidence suggest that the S in our SARIMA model should be 132.

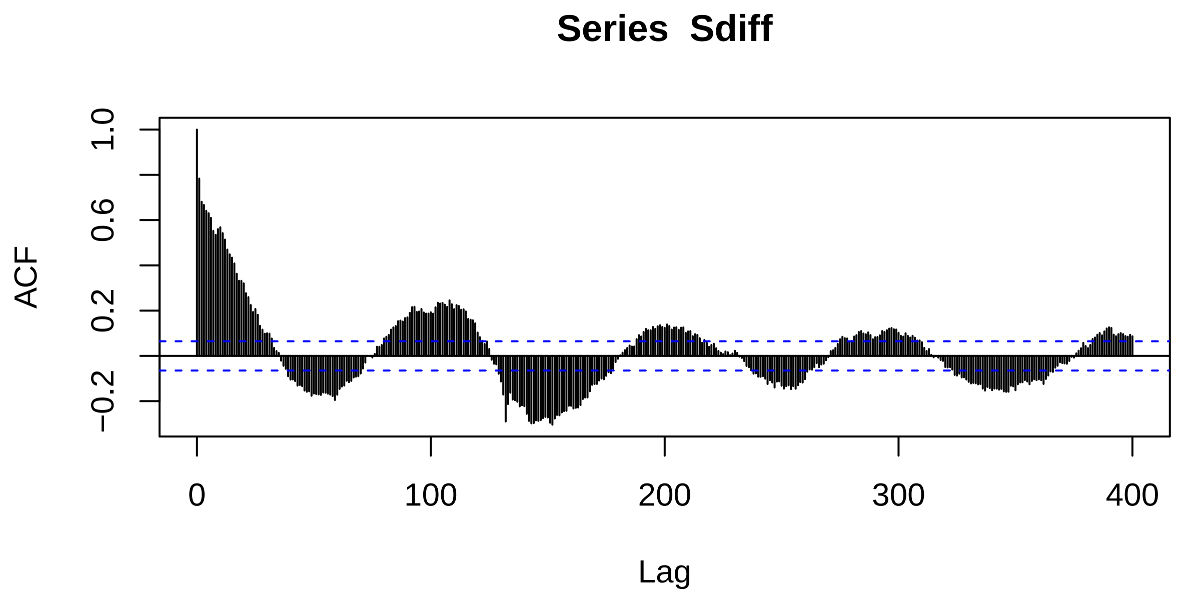
It's important to note that our period is variable - it ranges from 9.0 to 13.7 which is a huge range. Box-Jenkins assumes a constant period so the models that we produce from the Box-Jenkins class of models will automatically suffer from lack-of-fit due to the non-constant period.

Now let's do seasonal differencing and re-evaluate the seasonality of the data.

```r
Sdiff<-diff(train,132)
plot(Sdiff,type="l",main="Time-Series of one seasonal difference")
```



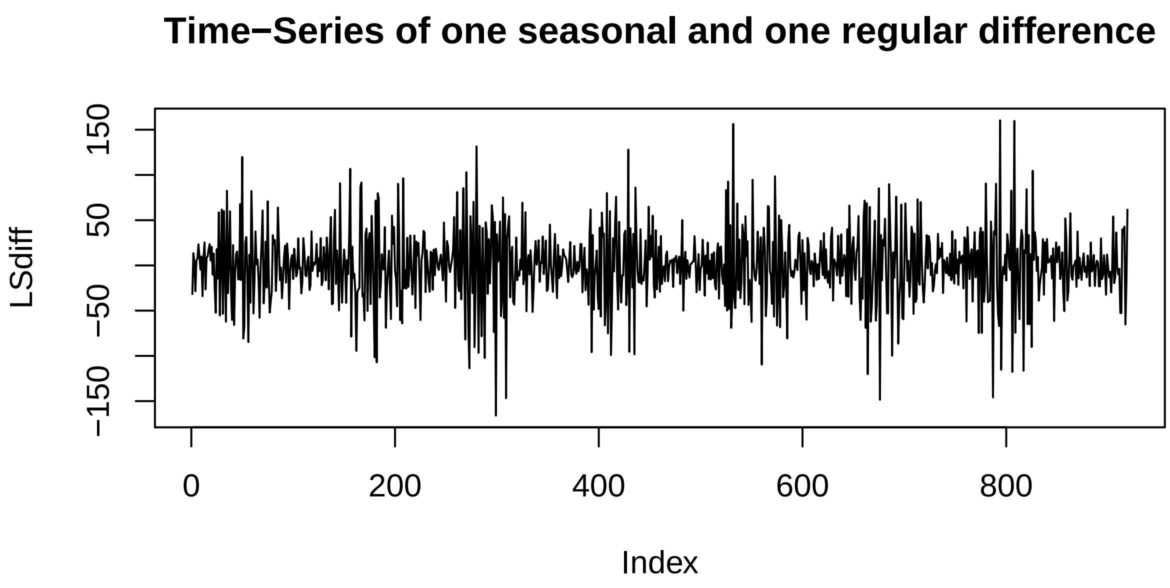
```r
acf(Sdiff,lag=400)
```



There appears to still be a seasonal component in the data. Our theory is that this “residual seasonality” is due to our imperfect season. If our season was perfectly 11 years, a seasonal difference would completely remove the effect. However since our seasons vary, the seasonal differencing didn’t work (at least to it’s fullest effect).

There are two steps from here: a seasonal differencing or a differencing at lag 1. We tried both and found that the latter resulted in stationarity while the former did not. Below is the time-series plot of the seasonal-differenced, then first-order differenced training data:

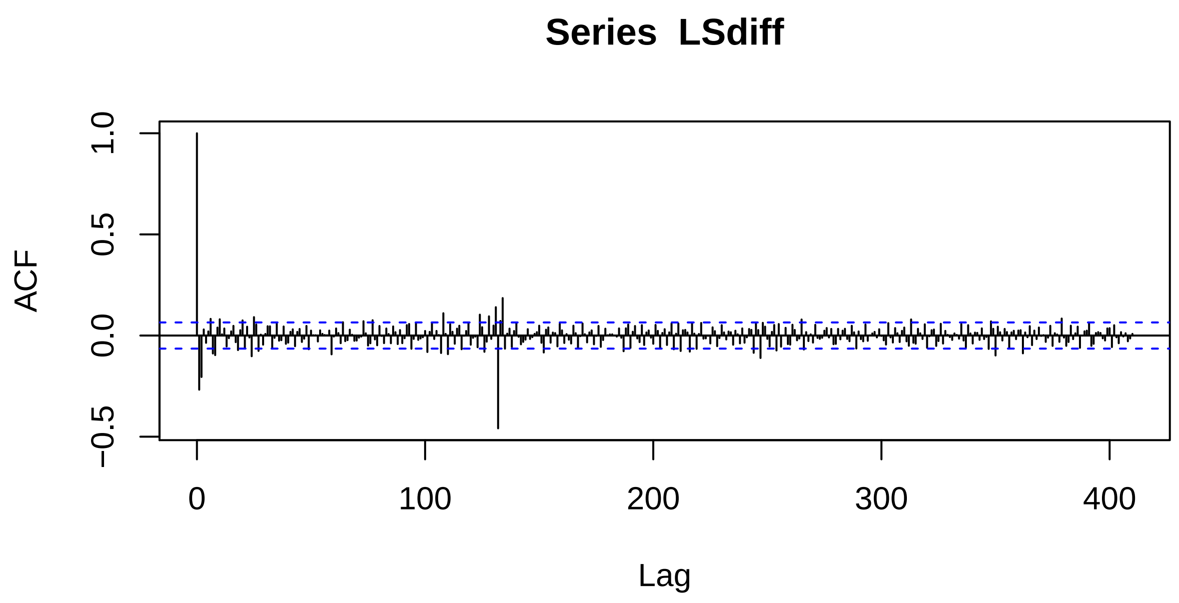
```r
LSdiff<-diff(diff(train,132))
plot(LSdiff,type="l",main="Time-Series of one seasonal and one regular difference")
```



There is no straightforward reason for why lag 1-differencing, something normally used to remove a linear trend in the mean, removed this "residual seasonality" but suffice to say that with a $d = 1$ and $D = 1$, our data has become stationary and we can now proceed with fitting SARIMA models.

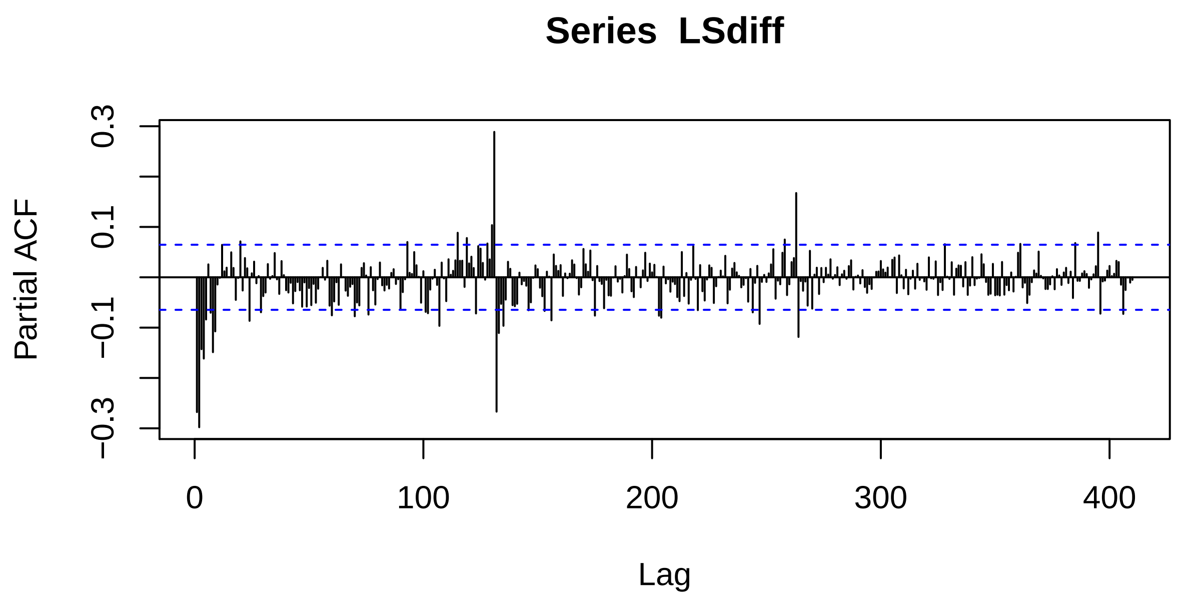
SARIMA Model Selection

```r
acf(LSdiff,lag=410) #lagof 410 so we can see 3 seasons and a little more
```



It looks like at the local level there's either a lag of 2 or a very steep exponential decay. At the seasonal level there's a lag of 1 and we can see some frequency leakage around the first seasonal lag.

```r
acf(LSdiff,lag=410,type="partial")
```



At the local level, the PACF appears to show sinusoidal decay. At the seasonal level, there's an exponential decay. It could be linear but the PACF of twice-seasonal differenced data doesn't make the seasonal lags go to zero any faster.

Based on our interpretations of the ACF and PACF plots, we have two basic types of models to try:

1. $SARIMA(0,1,2)*(0,1,1)_{132}$ which is an $MA(2)$ at the local level and an $MA(1)$ at the seasonal level.
2. $SARIMA(p,1,q)*(0,1,1)_{132}$ which is an $ARMA(p,q)$ at the local level with p,q to be determined and an $MA(1)$ at the seasonal level.

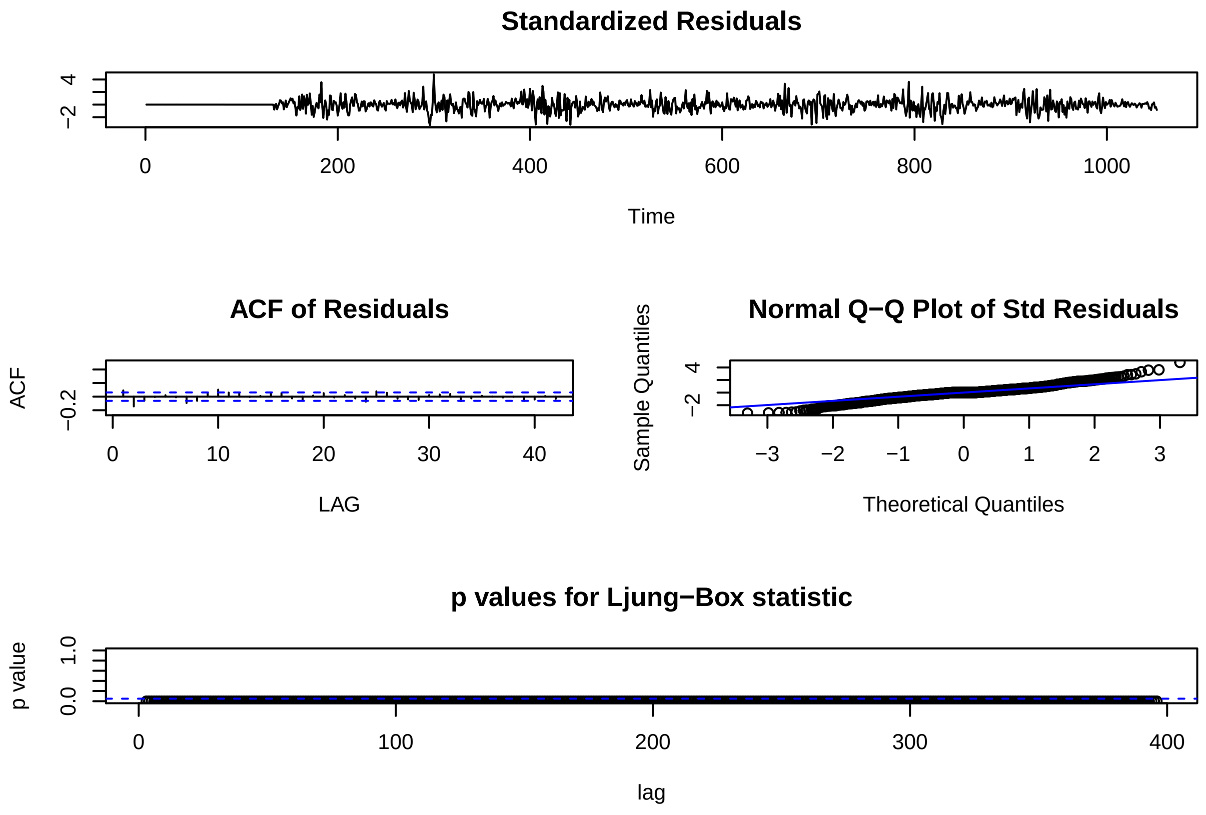
Since SARIMA models took so long to run, we'll only review one model in this report (to cut knitting time down), but we did run and evaluate 11 different models. All of their residual diagnostics and forecast graphs will be presented in the appendix.

The residual diagnostic graphs were pretty much the same for all of the models with small, neglible differences. Thus our deciding criterion was sum of squared prediction error:

Model	SS_{pr}
(0,1,1)(0,1,1)	61837
(2,1,1)(0,1,1)	68431
(1,1,2)(0,1,1)	69838
(2,1,2)(0,1,1)	68542
(3,1,2)(0,1,1)	68564
(0,1,2)(0,1,1)	70926
(3,1,1)(0,1,1)	68603
(2,1,3)(0,1,1)	68620
(3,1,3)(0,1,1)	68514
(4,1,3)(0,1,1)	68193
(5,1,3)(0,1,1)	70462

Our best performing model turned out to be $SARIMA(0,1,1)*(0,1,1)_{132}$. Let's look at the residual diagnostics.

```r
sarima(xdata=train,p=0,d=1,q=1,P=0,D=1,Q=1,S=132,details=FALSE)
```



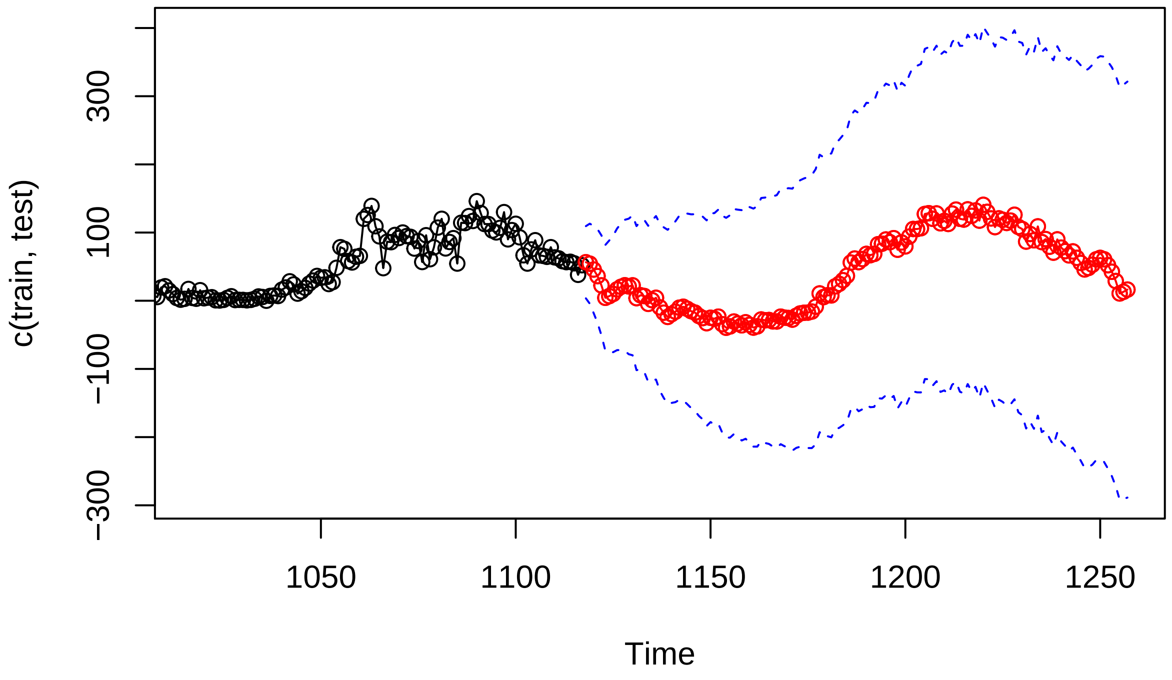
```
## $fit
##
## Call:
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,
##     Q), period = S), include.mean = !no.constant, optim.control = list(trace = trc,
##     REPORT = 1, reltol = tol))
##
## Coefficients:
##           ma1     sma1
##       -0.5215  -0.9974
## s.e.   0.0342   0.0756
##
## sigma^2 estimated as 634.5:  log likelihood = -4408.73,  aic = 8823.46
##
## $AIC
## [1] 7.456624
##
## $AICc
## [1] 7.458547
##
## $BIC
## [1] 6.466051
```

Normality is not great (which is why we won't put confidence bands on our forecast from this model). The ACF is pushing the limit of how many autocorrelations should be popping out of the confidence bands but the ones that do do appear to be at random places. Finally, the Ljung-Box statistic is atrocious. We believe the statistical significance of the Ljung-Box hypothesis tests to be mostly an artefact of our gigantic sample size. With a large enough n, it's possible to find any small effect size (especially small ones totally due to randomness) statistically significant. Thus we took the Ljung-Box statistic with a grain of salt.

As mentioned earlier, since all 11 models had similar residual diagnostics, we ended up choosing this model based on how well it forecasts the testing data (as measured by SS_{pr}).

Now that we have our SARIMA model, let's retrain on the the training and validation data and forecast the next solar maximum.

```r
pred<-sarima.for(xdata=c(train,test),n.ahead=140,p=0,d=1,q=1,P=0,D=1,Q=1,S=132)
```



The next solar maximum is predicted to be 140.7039593 .

Conclusion

Comparing the best models of the three methods: Regression, Holt-Winters, and Sarima on forecasting i.e. SS_{pr} we find that additive Holt-Winters clearly comes out on top:

Model	SS_{pr}
Regression (Season)	61837
Holt-Winters (Additive)	68431
SARIMA(0,1,1)(0,1,1)132	69838

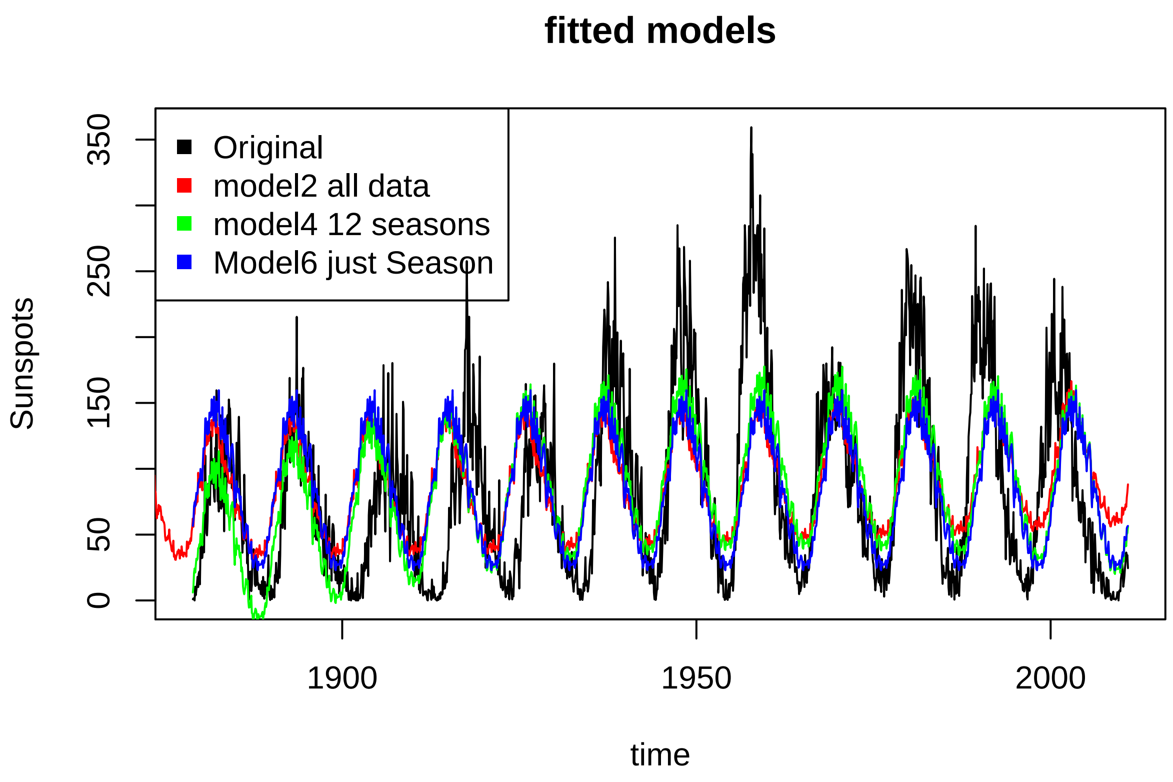
Thus for forecasting the next solar maximum, we used additive Holt-Winters and predicted 116 sunspots occurring on February 2026. As mentioned in the Holt-Winters subsection, prediction intervals were not valid so we only give the point estimate here.

As a general conclusion, our models were across-the-board pretty poor. Few if any assumptions were met, prediction error was fairly high, and our models suffered from some egregious errors like predicting negative sunspots. While there will always be problems with fitting theoretical models to real-world data, we believe our data was especially complicated for time-series data. We confirmed this by exploring some of the literature surrounding sunspot prediction and found that modern methods for forecasting sunspot data are fantastically complex, including such things as chaotic time series analysis and neural network ARIMA models. Bearing that in mind, we understand the limitations of the approaches we took and hopefully sometime in the future on our own time we can explore some of the more complex time-series methods that exist out there.

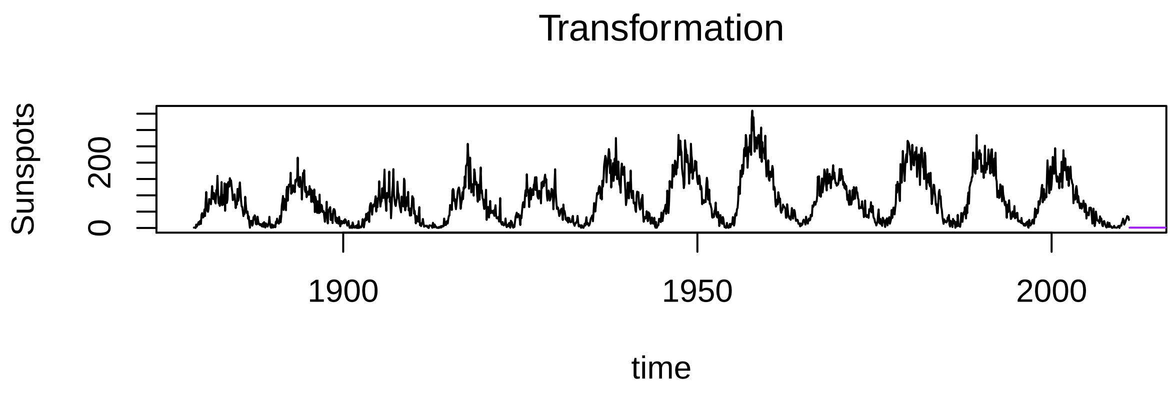
Appendix

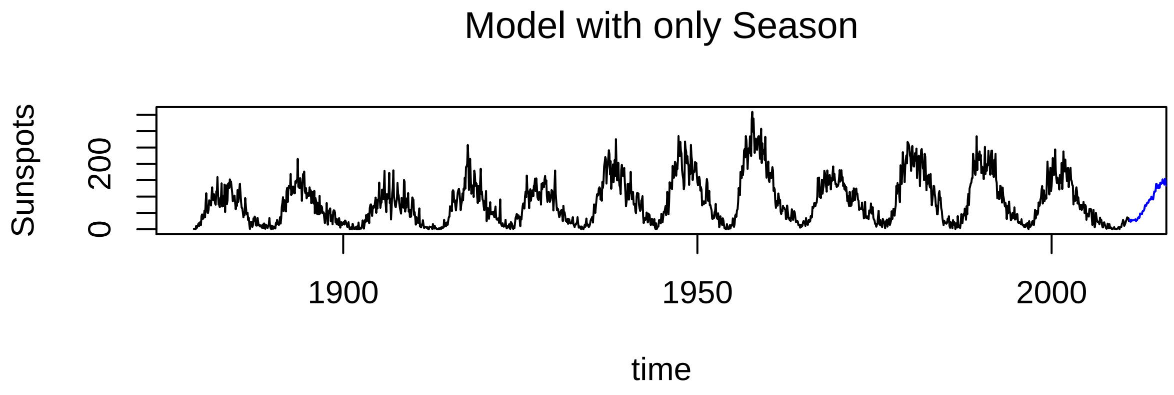
Below we show the plots of the models that we tried doing regression on. All the models do a bad job in terms of fitting our data, however we were able to find a model that gave us the smallest SSP. We can also see that the model that provides the a better fit to the data is labeled as "model 4 12 seasons", but this model didnt give us the smallest SSPr. So the best prediction power is not always going to come from best fit.

```r
plot(sunspots[["date"]][1560:3144], training2, type="l", main=
       "fitted models",xlab="time",ylab="Sunspots")
points(sunspots[["date"]][1:3144],fitted(model2),col="red",type='l')
points(sunspots[["date"]][1560:3144],fitted(model4),col="green",type='l')
points(sunspots[["date"]][1560:3144],predict.lm(model6),type="l",col="blue")
legend("topleft",c("Original","model2 all data","model4 12 seasons","Model6 just Season"),
       pch= c(15,15,15), col=c("black","red","green","blue"))
```



```r
par(mfrow=c(2,1))
plot(sunspots[["date"]][1560:3144], training2, type="l", main =
       expression("Transformation"),xlab="time",ylab="Sunspots")
points(sunspots[["date"]][3145:3209],exp(pred5),col="purple",type='l')
plot(sunspots[["date"]][1560:3144], training2 , type="l",
main=expression("Model with only Season"),xlab="time",ylab="Sunspots")
points(sunspots[["date"]][3145:3209],pred6,col="blue",type='l')
```



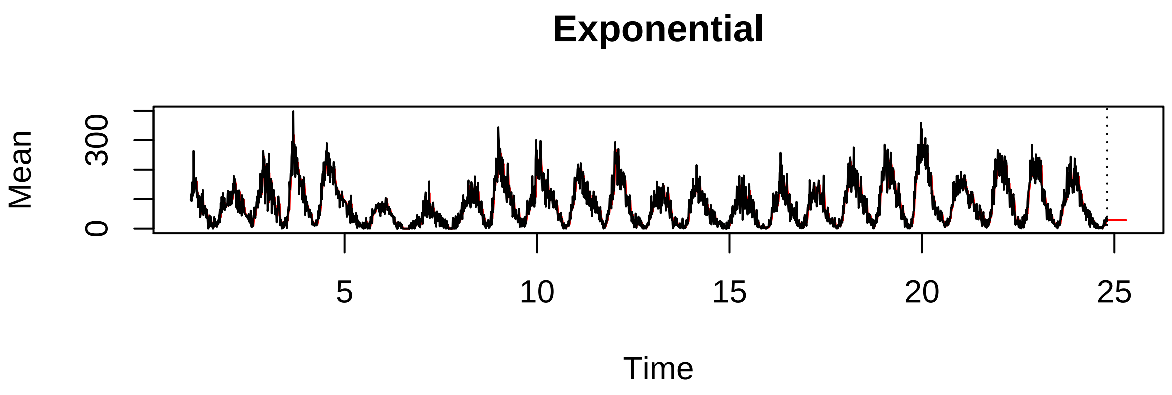


We included a plot of the two models that had the smallest SSPr, so you can visually see that indeed the model that provides a better prediction is the model with a seasonal component.

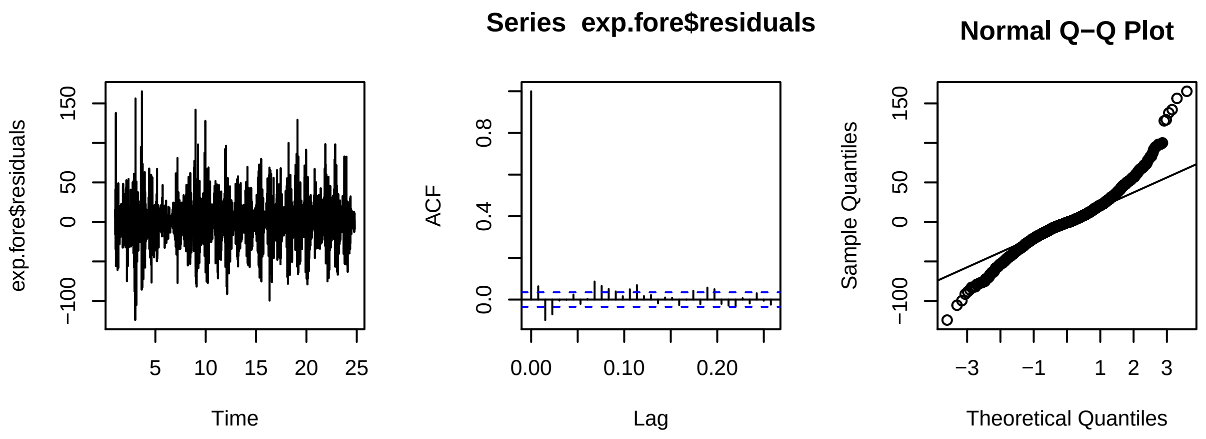
Holt-Winters Plots

Exponential

```
plot(exp,predict(exp,n.ahead=65),type='l',main = 'Exponential',ylab='Mean')
```



```
exp.fore <- forecast.HoltWinters(exp,h=65)
par(mfrow=c(1,3))
plot.ts(exp.fore$residuals)
acf(exp.fore$residuals)
qqnorm(exp.fore$residuals)
qqline(exp.fore$residuals)
```

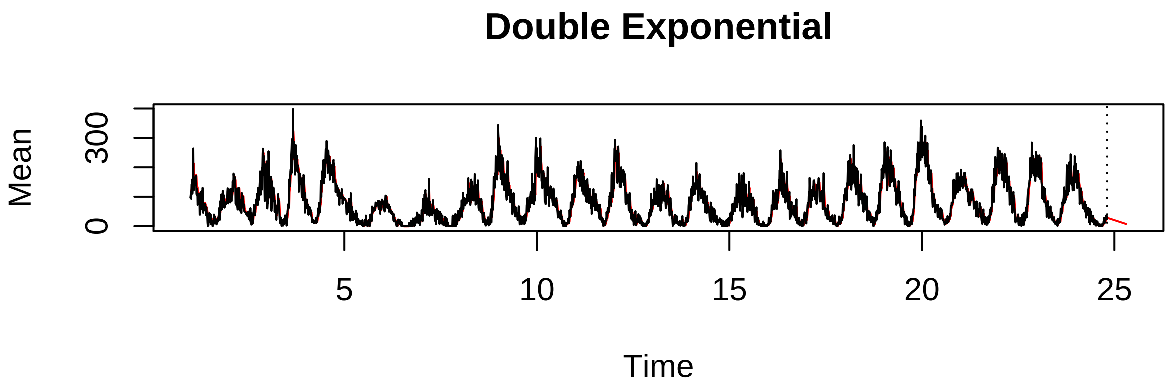


```
Box.test(exp.fore$residuals,type='Ljung-Box')
```

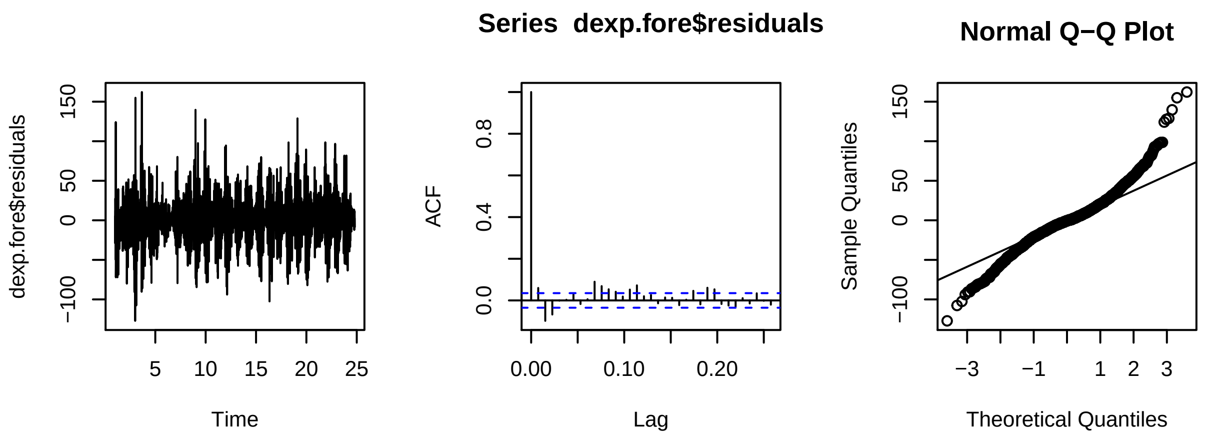
```
##
##  Box-Ljung test
##
## data:  exp.fore$residuals
## X-squared = 12.7, df = 1, p-value = 0.0003656
```

Double Exponential

```
plot(dexp,predict(dexp,n.ahead=65),type='l',
     main = 'Double Exponential',ylab='Mean')
```



```
dexp.fore <- forecast.HoltWinters(dexp,h=65)
par(mfrow=c(1,3))
plot.ts(dexp.fore$residuals)
acf(dexp.fore$residuals)
qqnorm(dexp.fore$residuals)
qqline(dexp.fore$residuals)
```

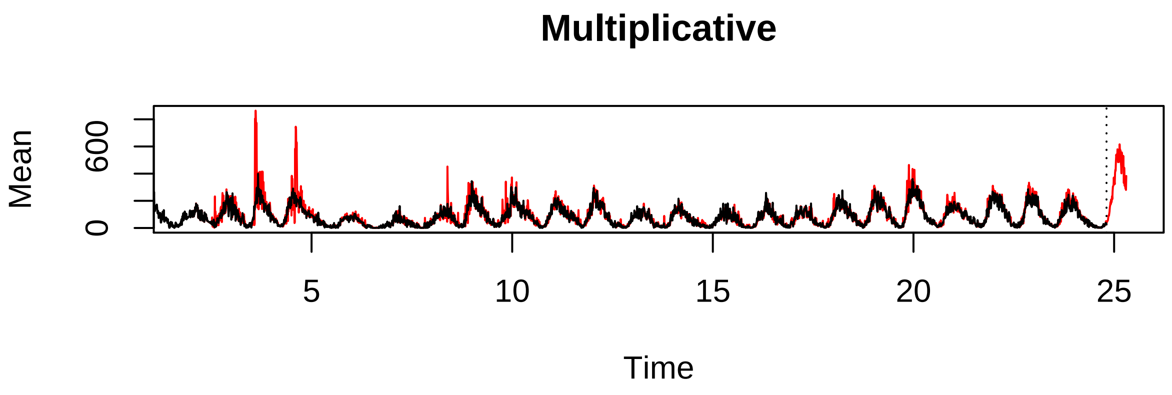


```
Box.test(dexp.fore$residuals,type='Ljung-Box')
```

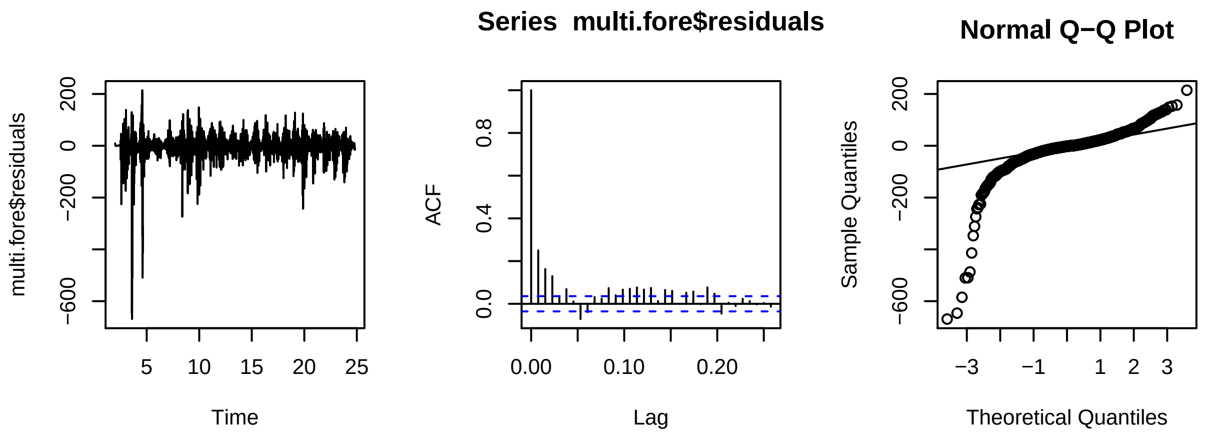
```
##
##  Box-Ljung test
##
## data:  dexp.fore$residuals
## X-squared = 11.267, df = 1, p-value = 0.000789
```

Multiplicative

```
plot(multi,predict(multi,n.ahead=65),type='l',main = 'Multiplicative',ylab='Mean')
```



```r
multi.fore <- forecast.HoltWinters(multi,h=65)
par(mfrow=c(1,3))
plot.ts(multi.fore$residuals)
acf(multi.fore$residuals)
qqnorm(multi.fore$residuals)
qqline(multi.fore$residuals)
```

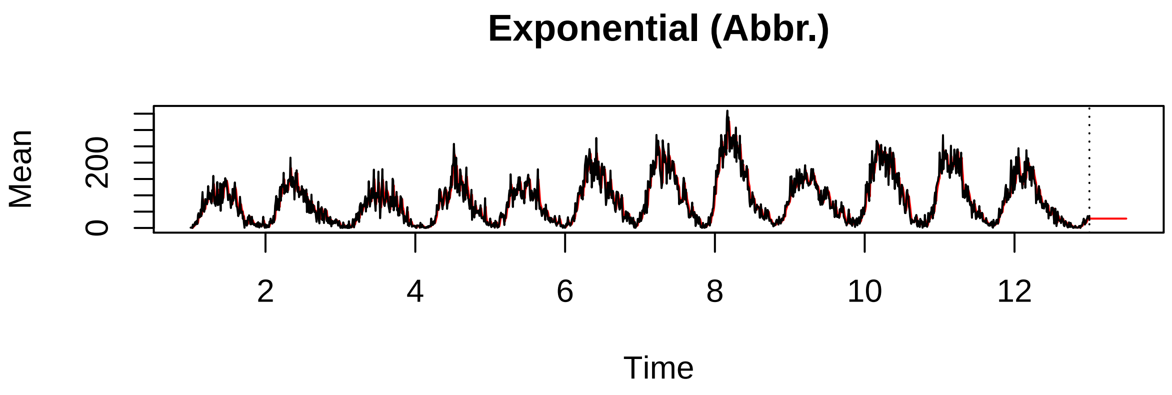


```r
Box.test(multi.fore$residuals,type='Ljung-Box')
```

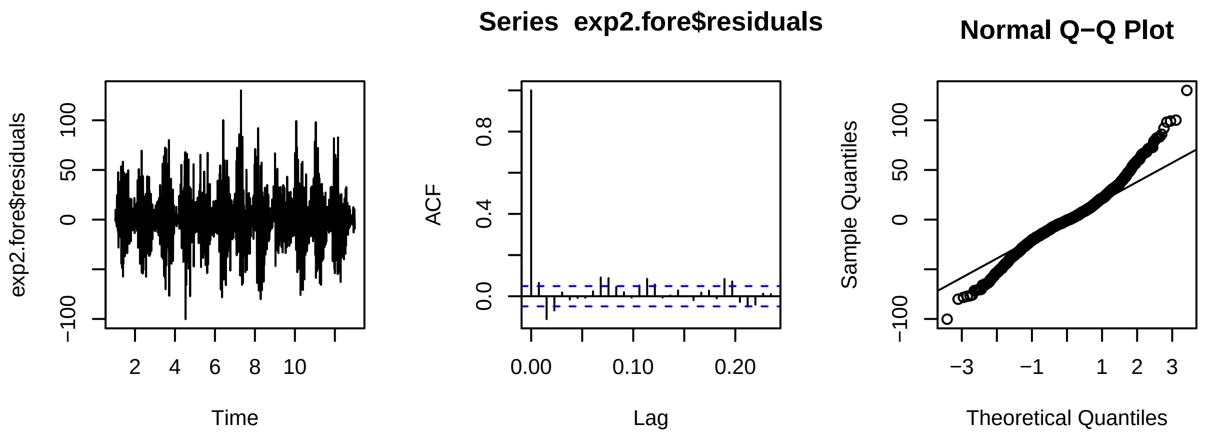
```
##
##  Box-Ljung test
##
## data:  multi.fore$residuals
## X-squared = 188.63, df = 1, p-value < 2.2e-16
```

Exponential (Abbreviated Training Data)

```r
plot(exp2,predict(exp2,n.ahead=65),type='l',main = 'Exponential (Abbr.)',ylab='Mean')
```



```r
exp2.fore <- forecast.HoltWinters(exp2,h=65)
par(mfrow=c(1,3))
plot.ts(exp2.fore$residuals)
acf(exp2.fore$residuals)
qqnorm(exp2.fore$residuals)
qqline(exp2.fore$residuals)
```

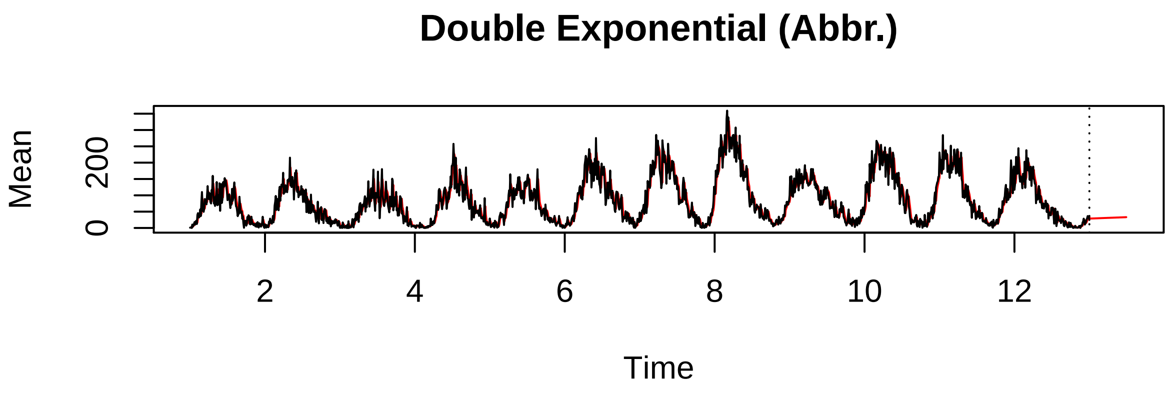


```r
Box.test(exp2.fore$residuals,type='Ljung-Box')
```

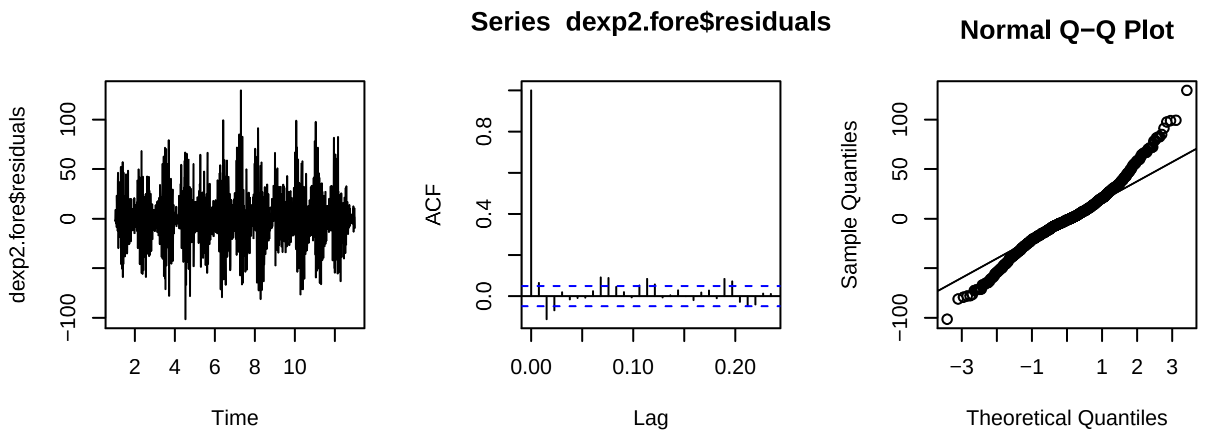
```
##
##  Box-Ljung test
##
## data:  exp2.fore$residuals
## X-squared = 6.6973, df = 1, p-value = 0.009656
```

Double Exponential (Abbreviated Training Data)

```r
plot(dexp2,predict(dexp2,n.ahead=65),type='l',main = 'Double Exponential (Abbr.)',ylab='Mean')
```



```r
dexp2.fore <- forecast.HoltWinters(dexp2,h=65)
par(mfrow=c(1,3))
plot.ts(dexp2.fore$residuals)
acf(dexp2.fore$residuals)
qqnorm(dexp2.fore$residuals)
qqline(dexp2.fore$residuals)
```

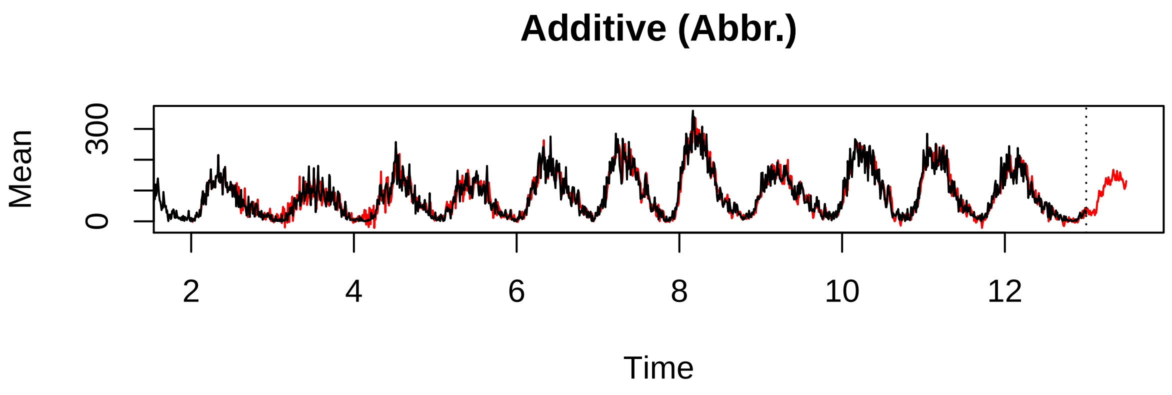


```r
Box.test(dexp2.fore$residuals,type='Ljung-Box')
```

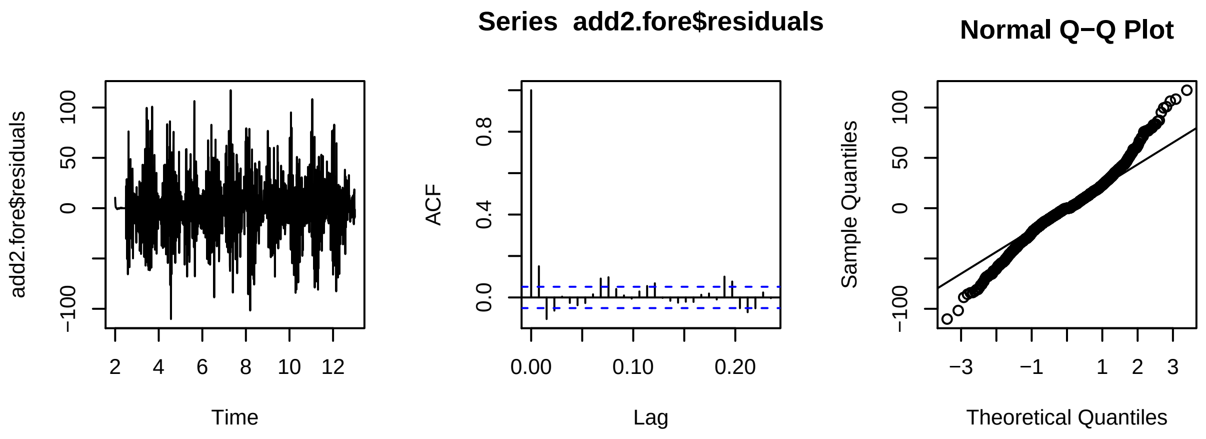
```
##
##  Box-Ljung test
##
## data:  dexp2.fore$residuals
## X-squared = 6.4818, df = 1, p-value = 0.0109
```

Additive (Abbreviated Training Data)

```r
plot(add2,predict(add2,n.ahead=65),type='l',main = 'Additive (Abbr.)',ylab='Mean')
```



```r
add2.fore <- forecast.HoltWinters(add2,h=65)
par(mfrow=c(1,3))
plot.ts(add2.fore$residuals)
acf(add2.fore$residuals)
qqnorm(add2.fore$residuals)
qqline(add2.fore$residuals)
```

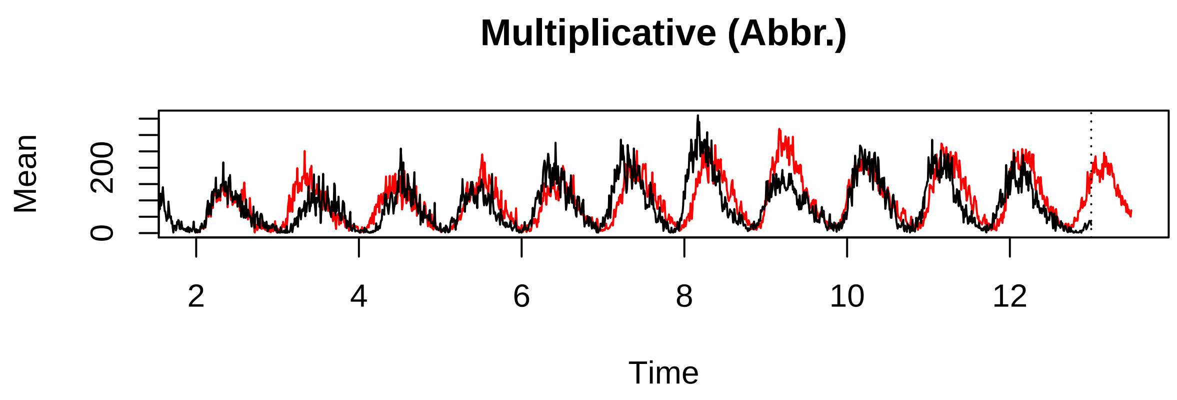


```r
Box.test(add2.fore$residuals,type='Ljung-Box')
```

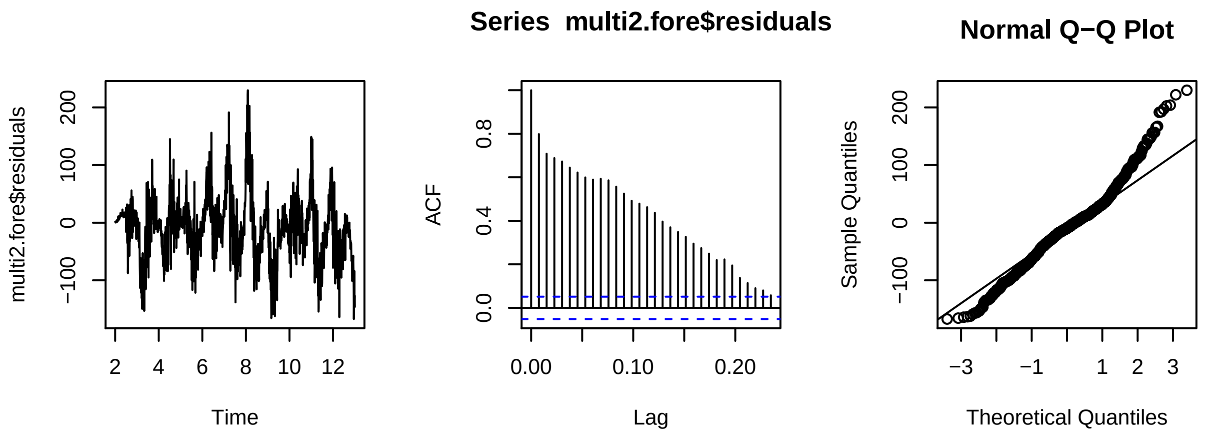
```
##
##  Box-Ljung test
##
## data:  add2.fore$residuals
## X-squared = 33.021, df = 1, p-value = 9.117e-09
```

Multiplicative (Abbreviated Training Data)

```r
plot(multi2,predict(multi2,n.ahead=65),type='l',main = 'Multiplicative (Abbr.)',ylab='Mean')
```



```r
multi2.fore <- forecast.HoltWinters(multi2,h=65)
par(mfrow=c(1,3))
plot.ts(multi2.fore$residuals)
acf(multi2.fore$residuals)
qqnorm(multi2.fore$residuals)
qqline(multi2.fore$residuals)
```

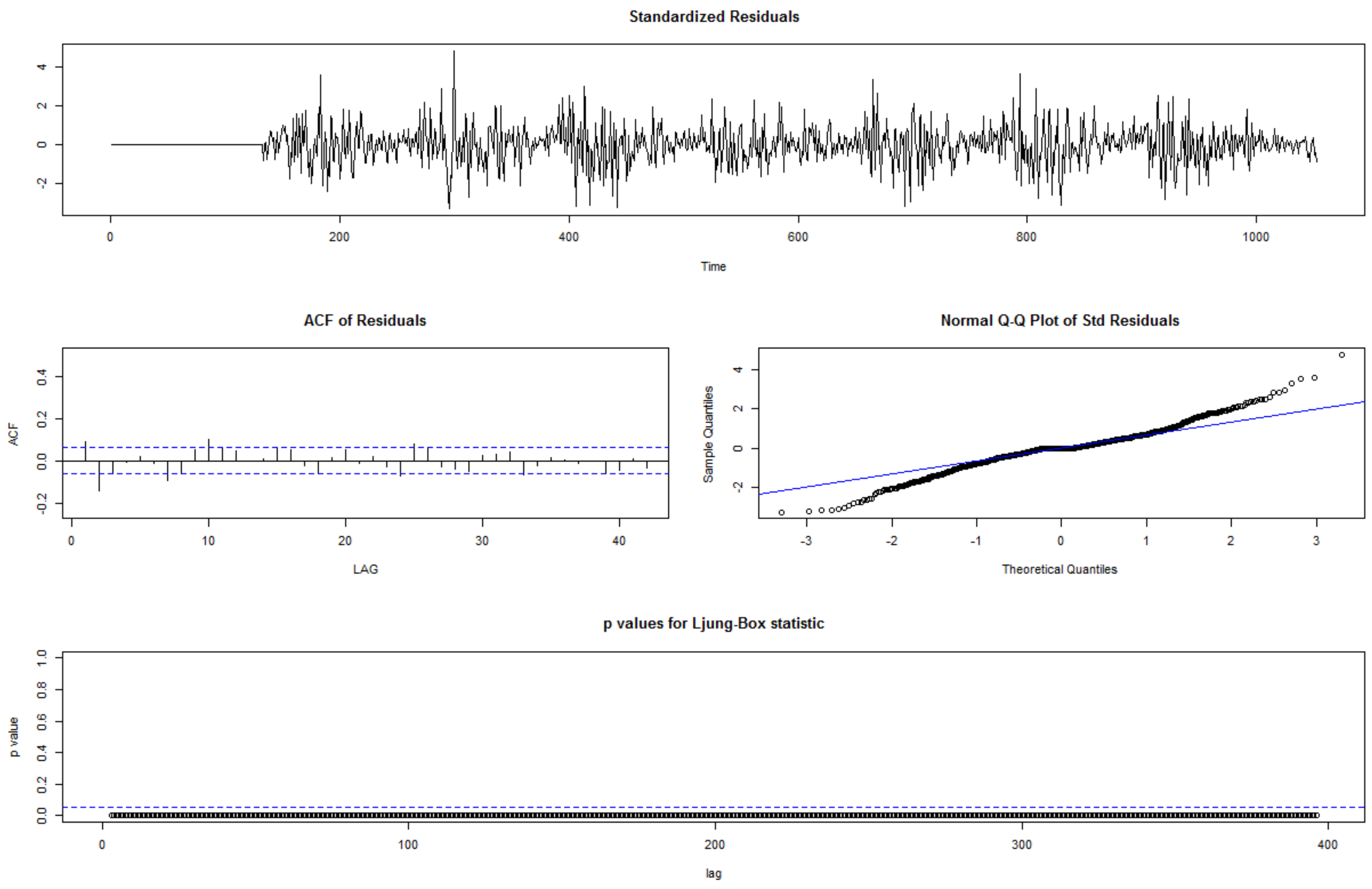


```r
Box.test(multi2.fore$residuals,type='Ljung-Box')
```

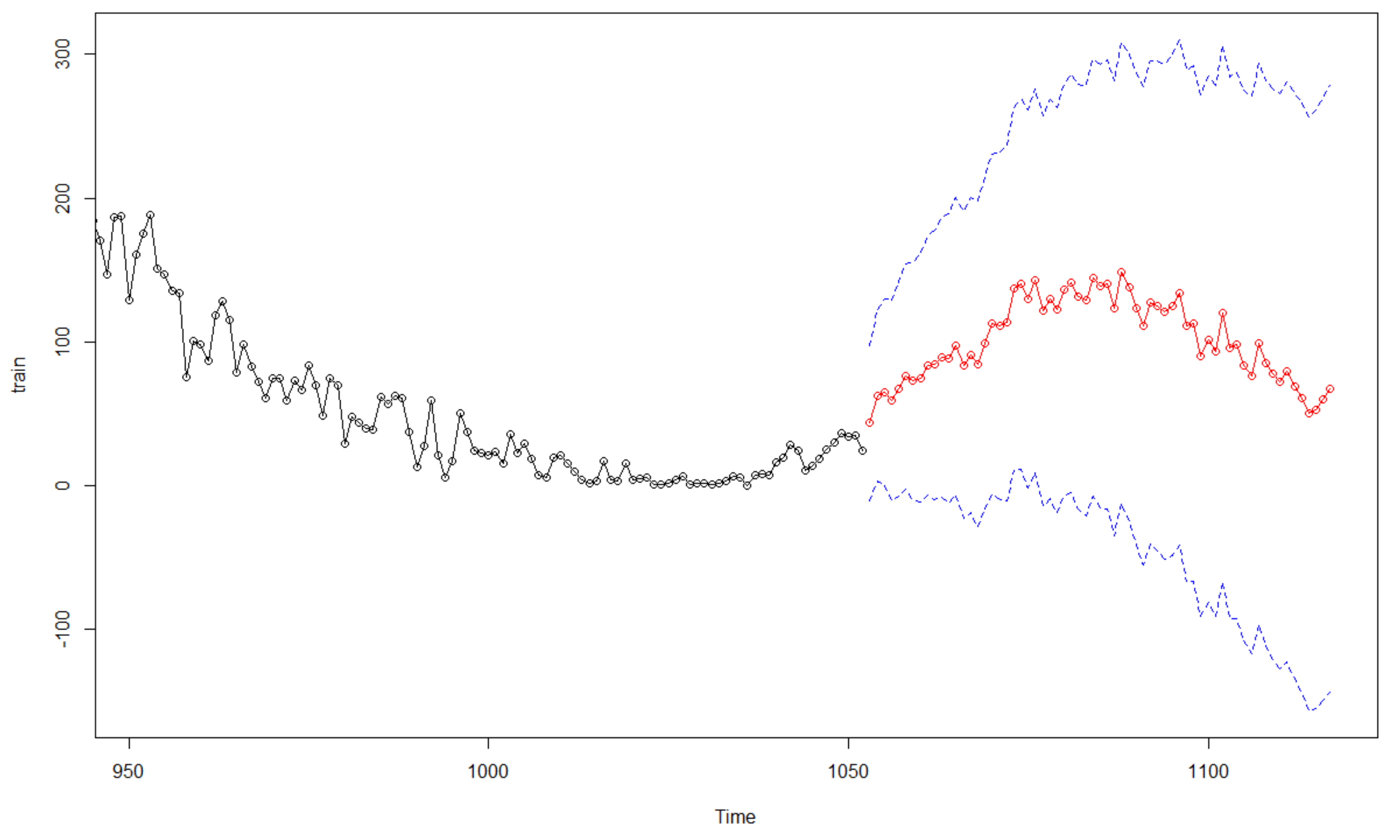
```
##
##  Box-Ljung test
##
## data:  multi2.fore$residuals
## X-squared = 927.94, df = 1, p-value < 2.2e-16
```

SARIMA(0,1,1)*(0,1,1)132

Residual Diagnostics

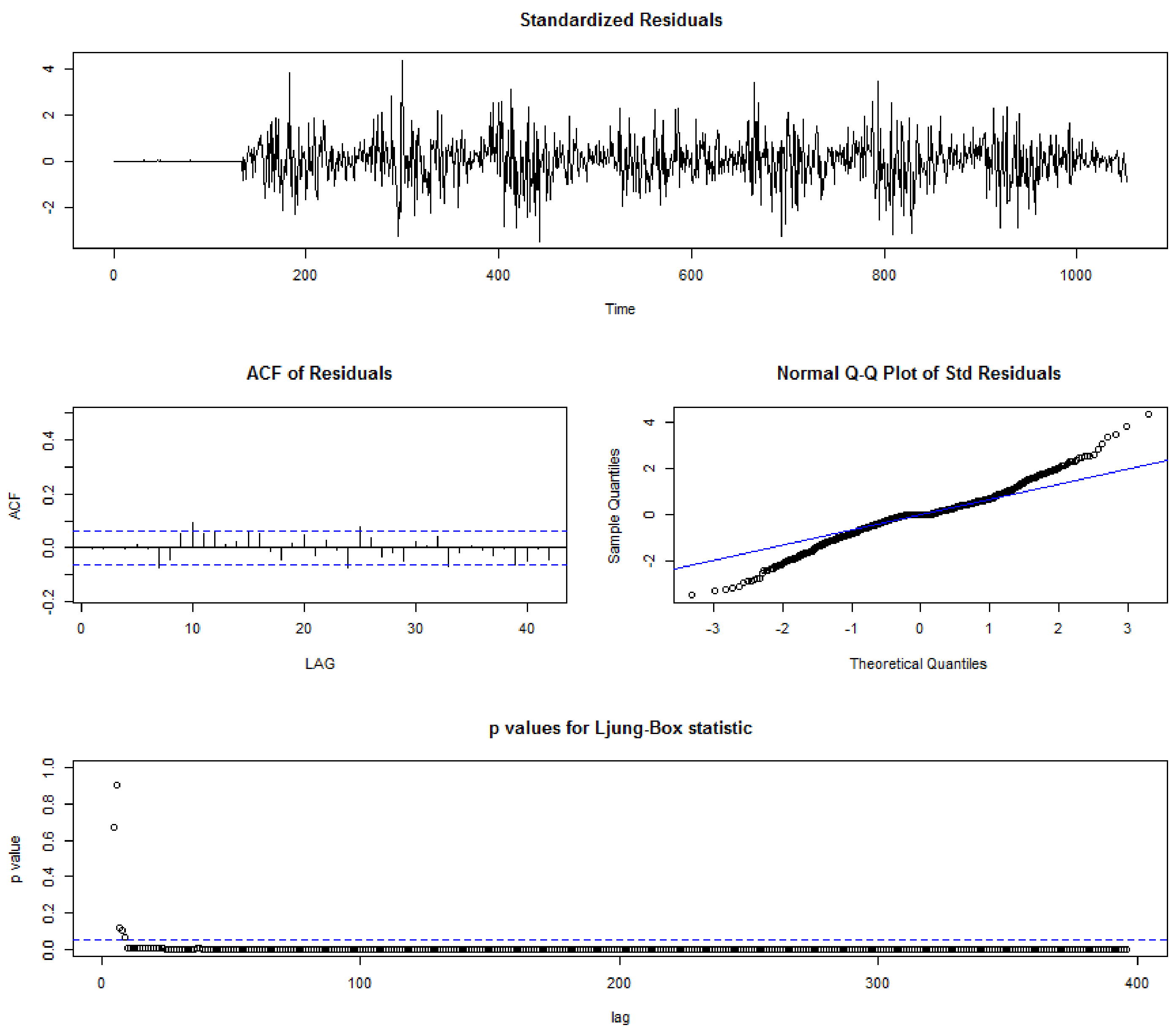


Forecasting Validation Set

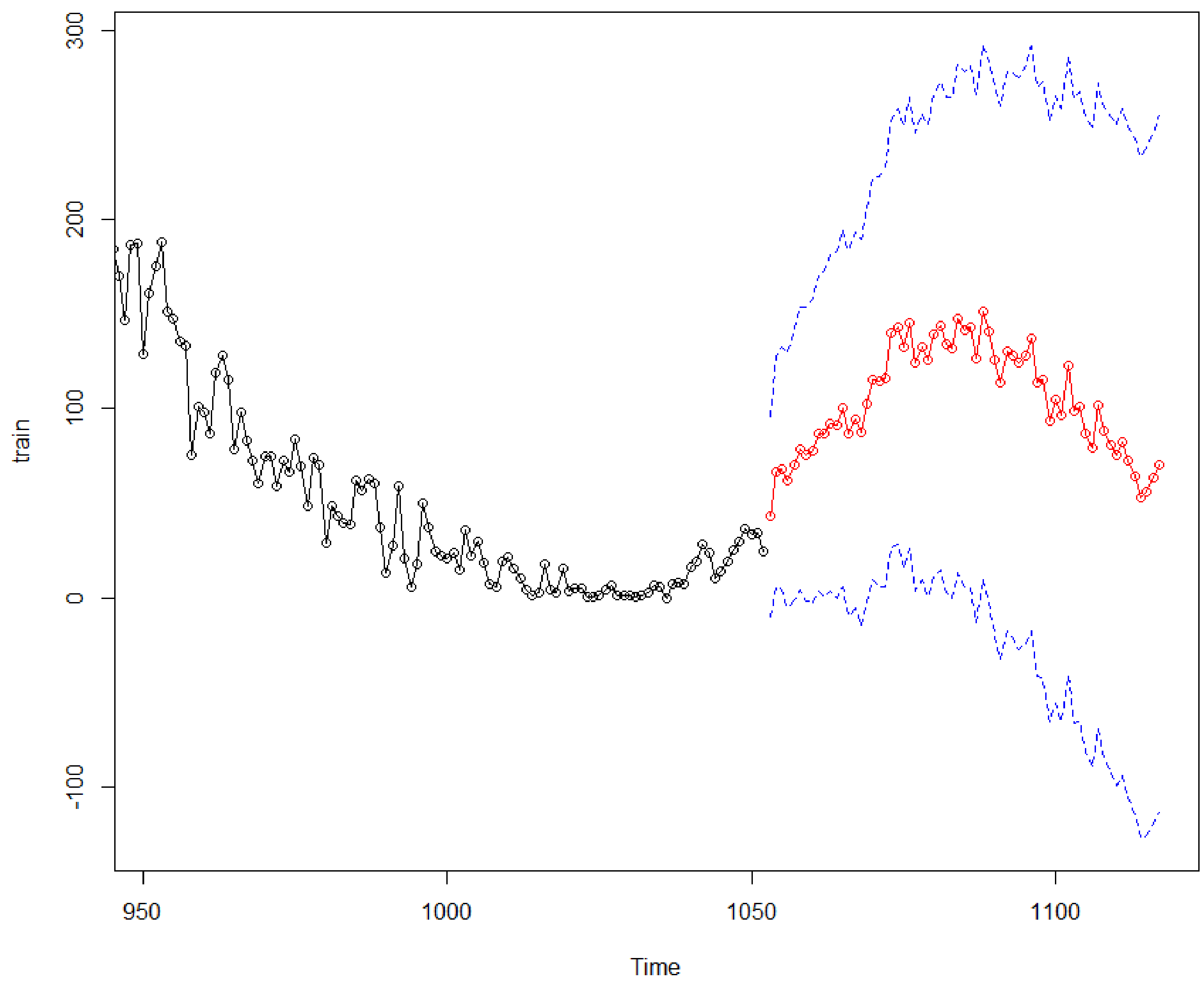


SARIMA(2,1,1)*(0,1,1)132

Residual Diagnostics

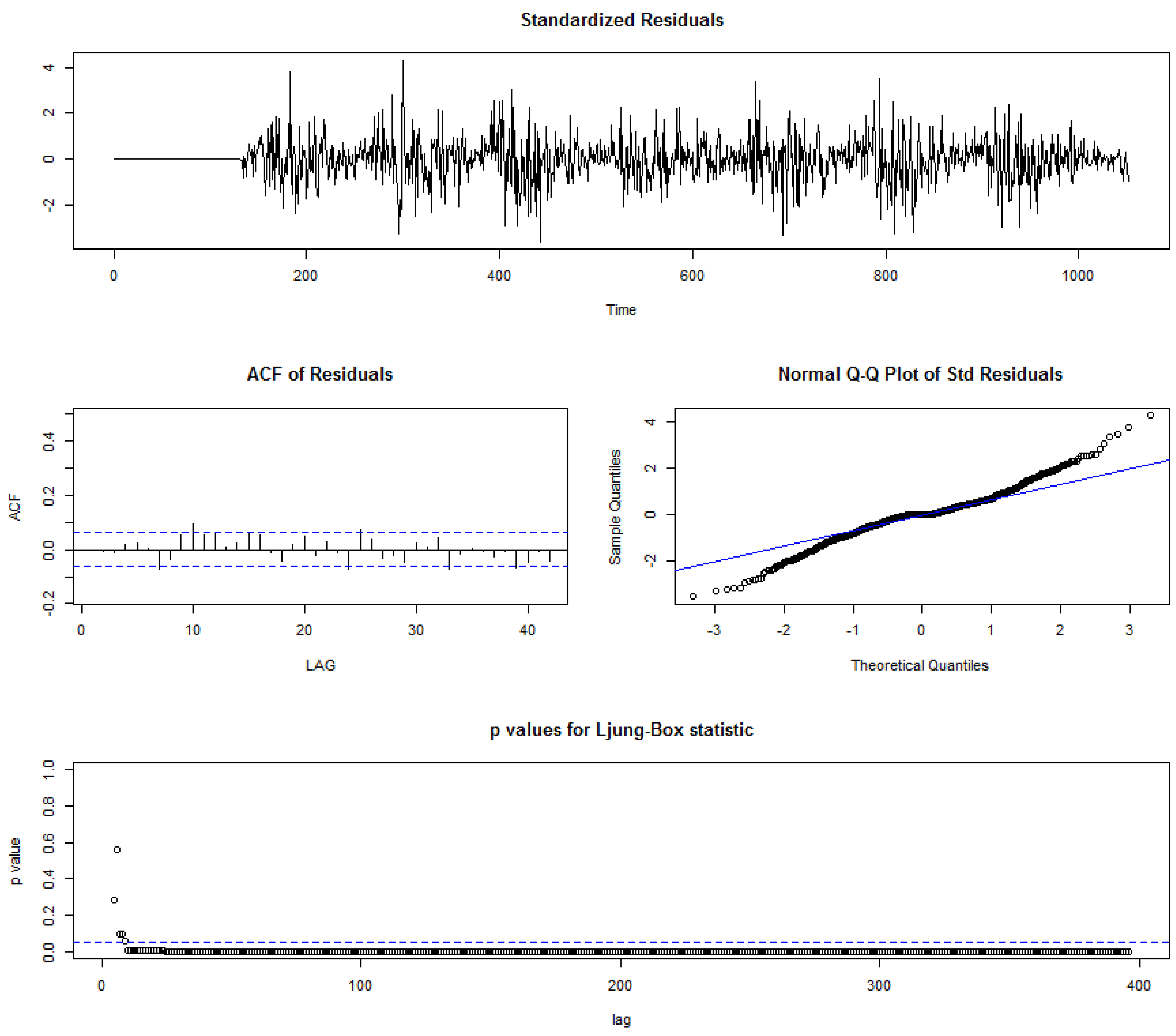


Forecasting Validation Set

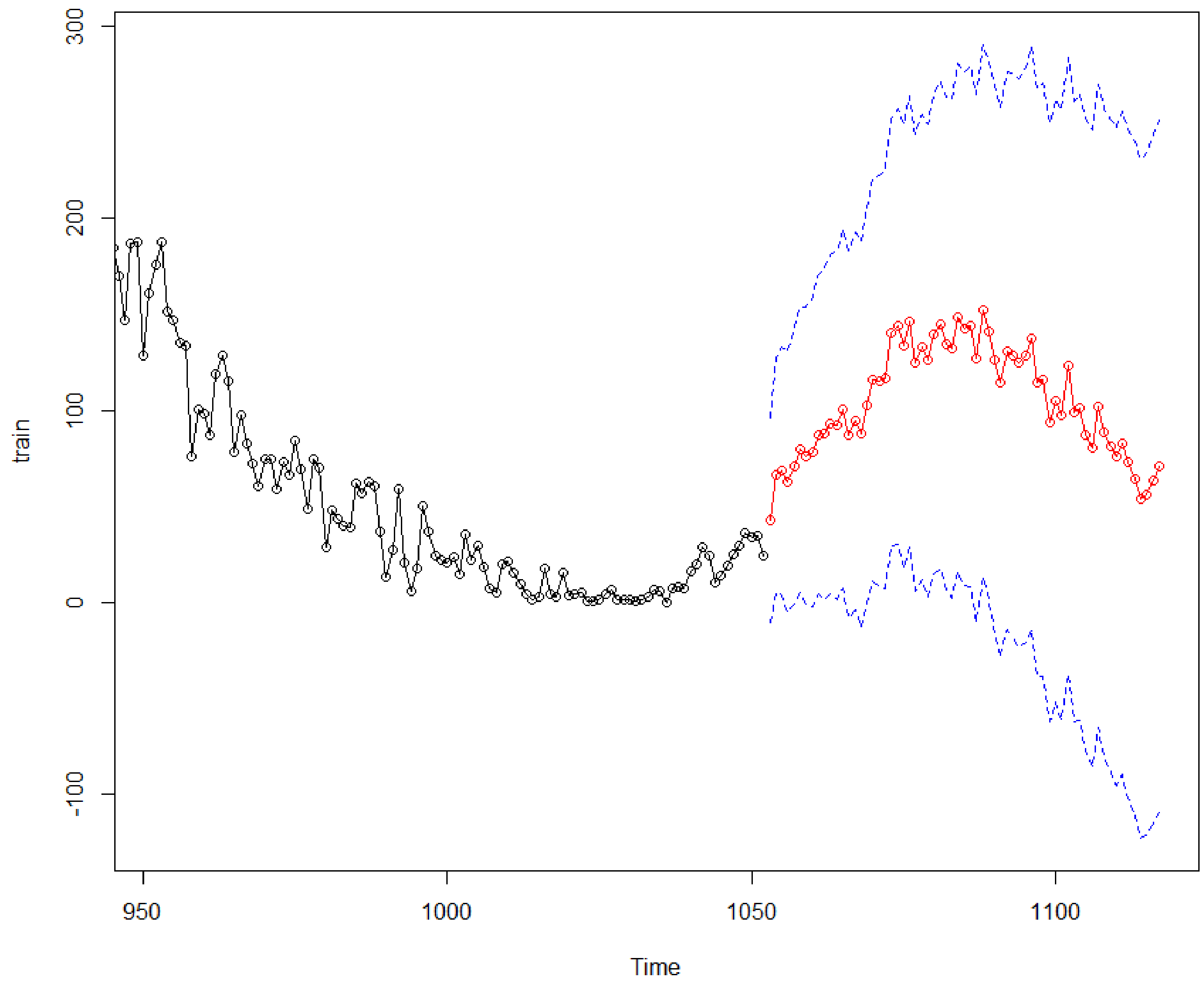


SARIMA(1,1,2)*(0,1,1)132

Residual Diagnostics

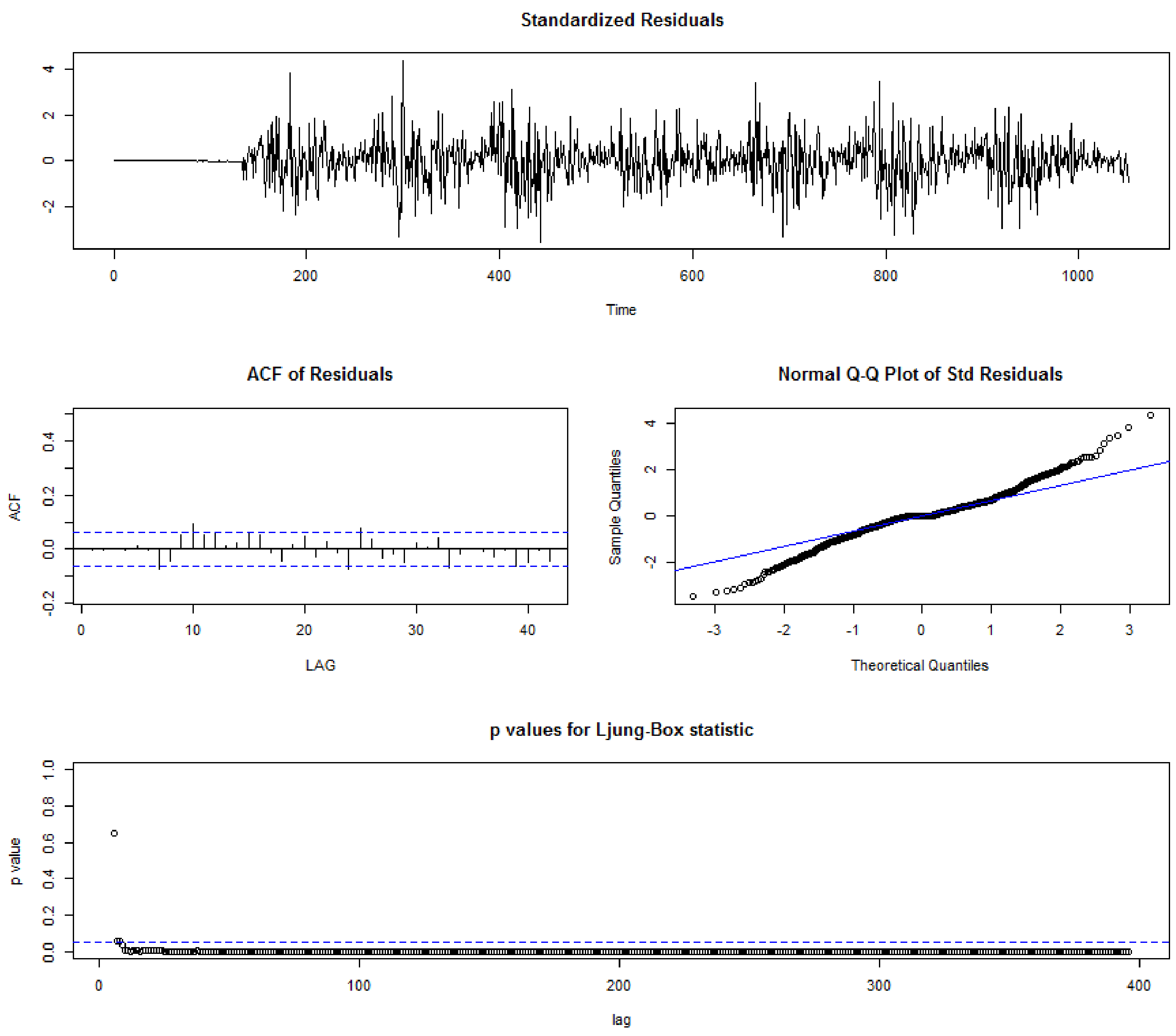


Forecasting Validation Set

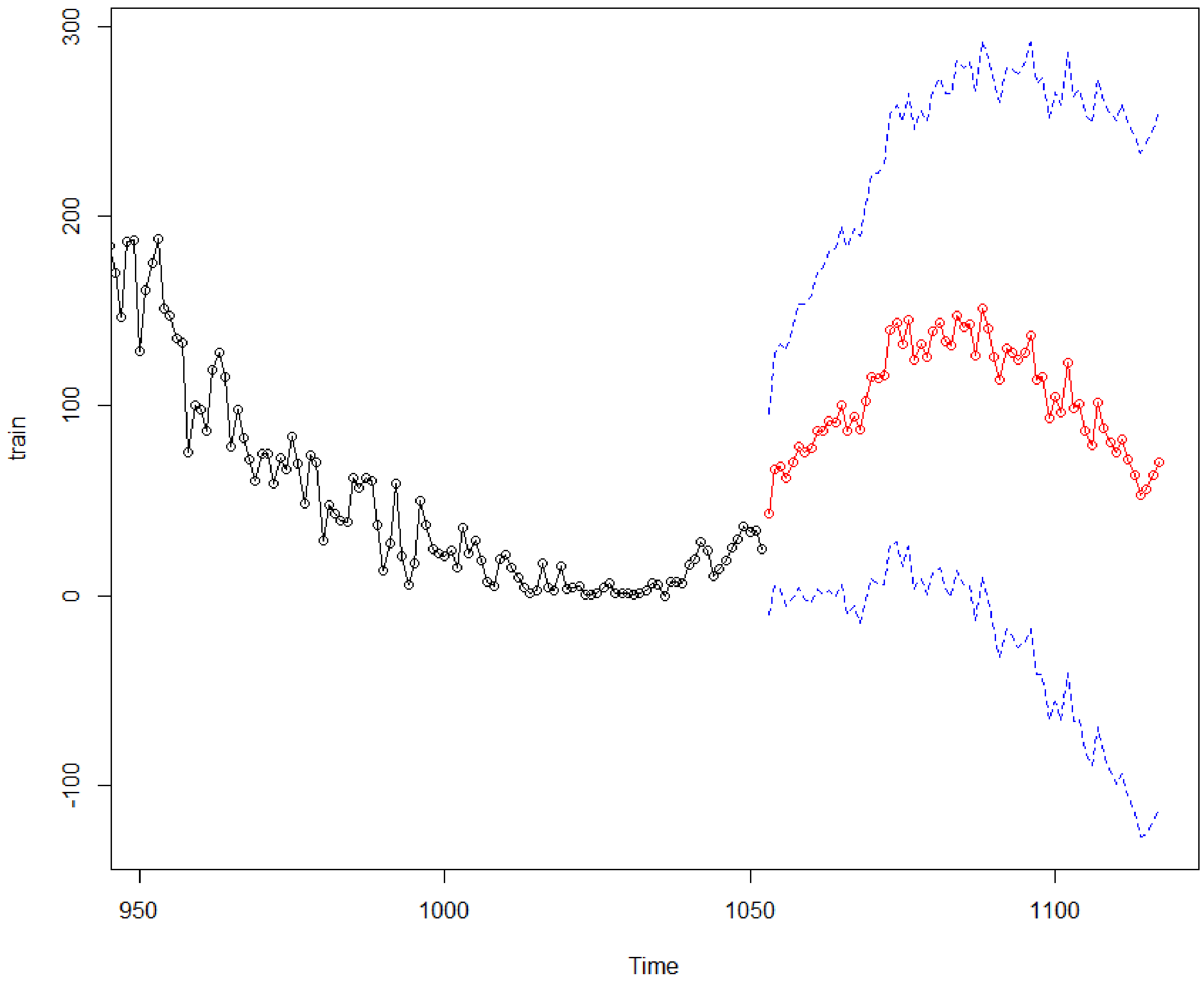


SARIMA(2,1,2)*(0,1,1)132

Residual Diagnostics

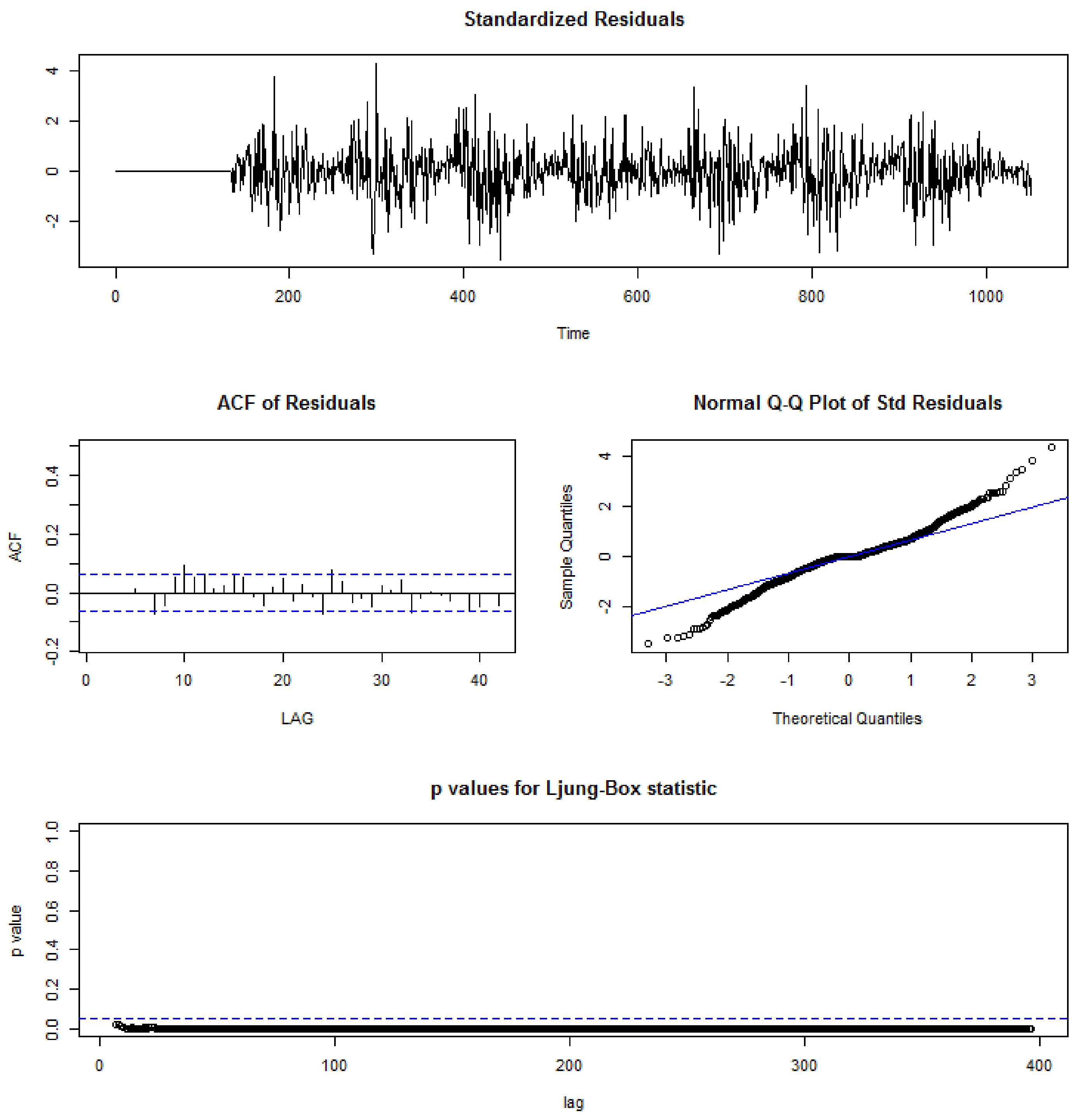


Forecasting Validation Set

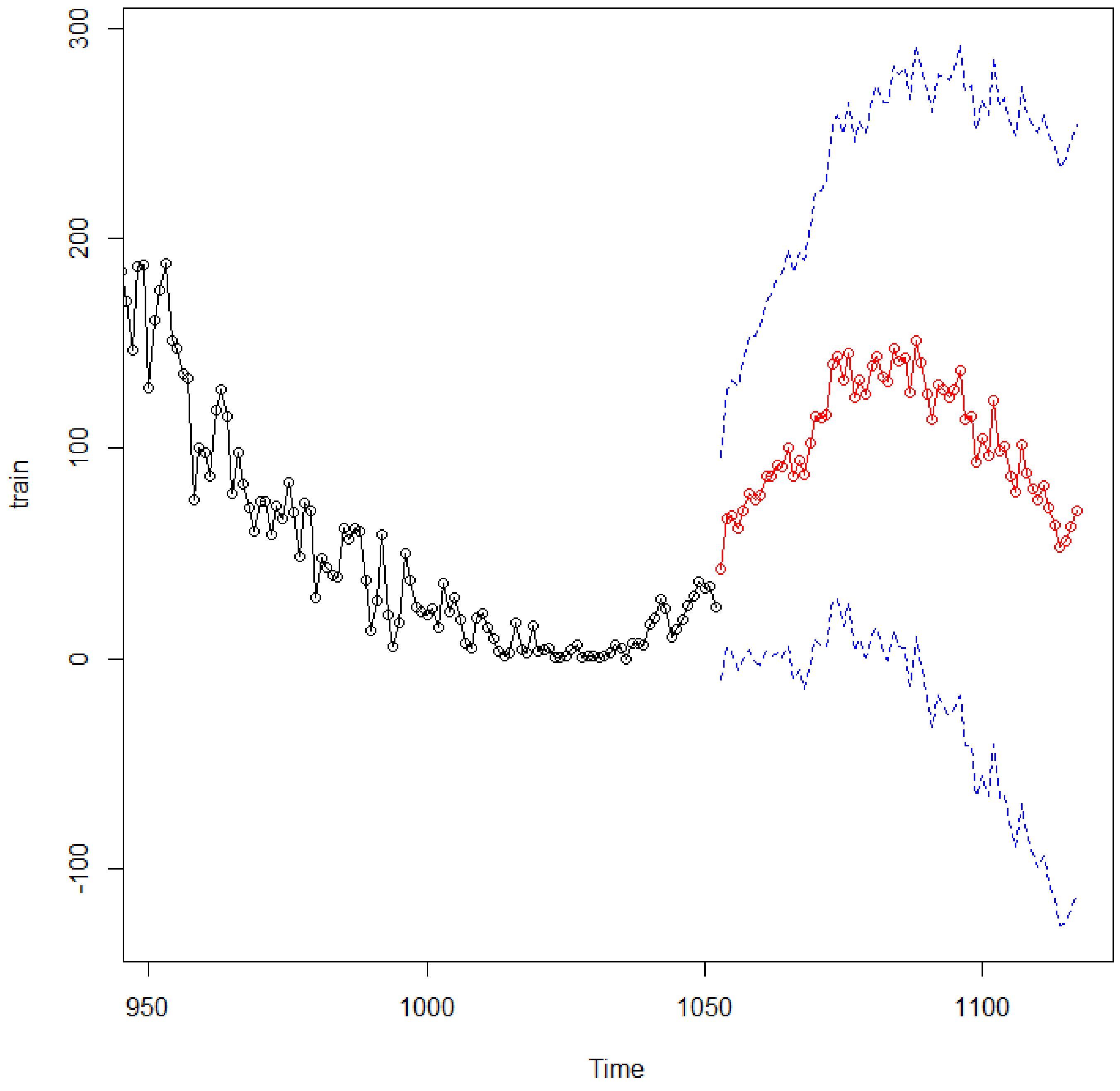


SARIMA(3,1,2)*(0,1,1)132

Residual Diagnostics

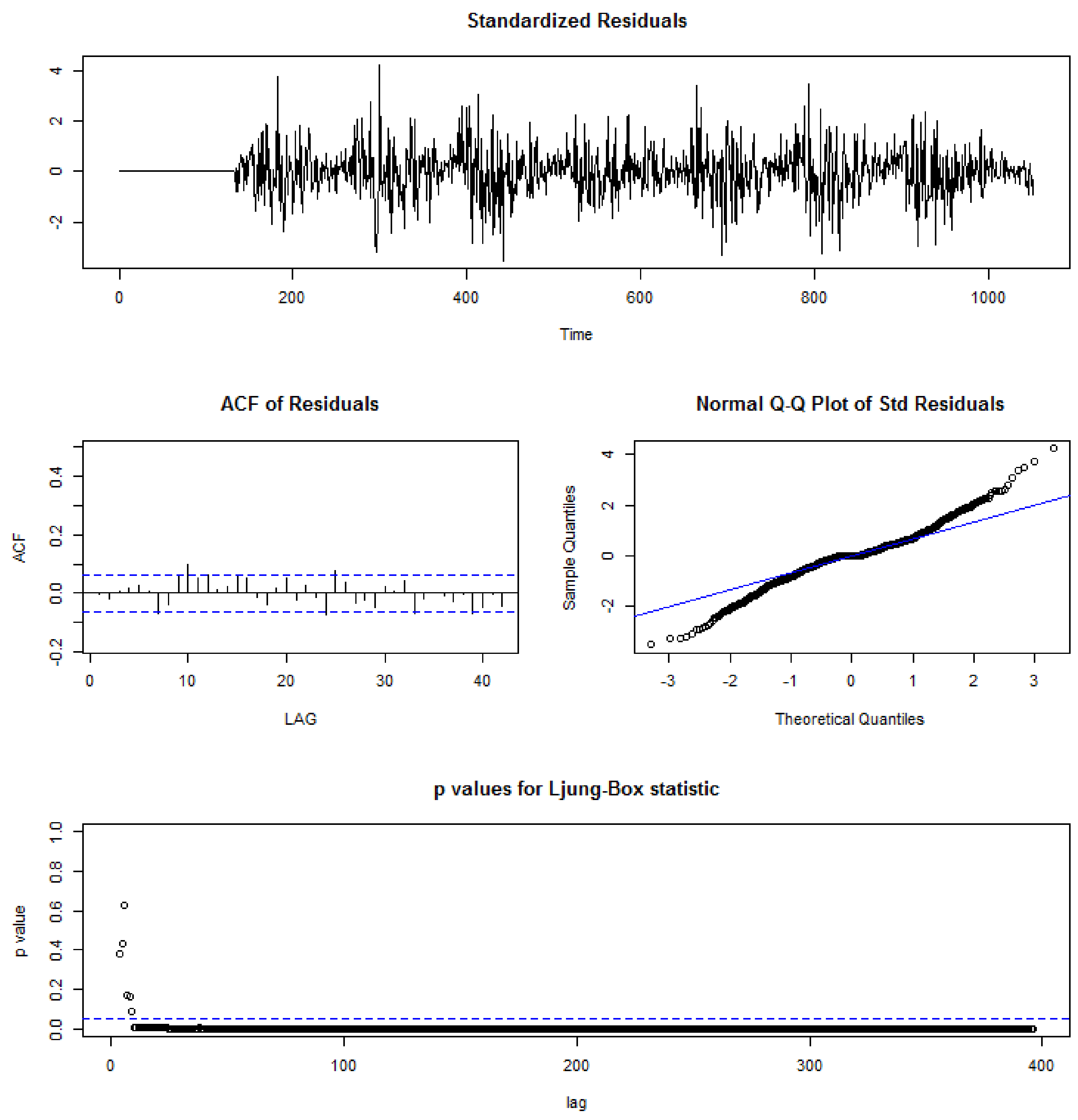


Forecasting Validation Set

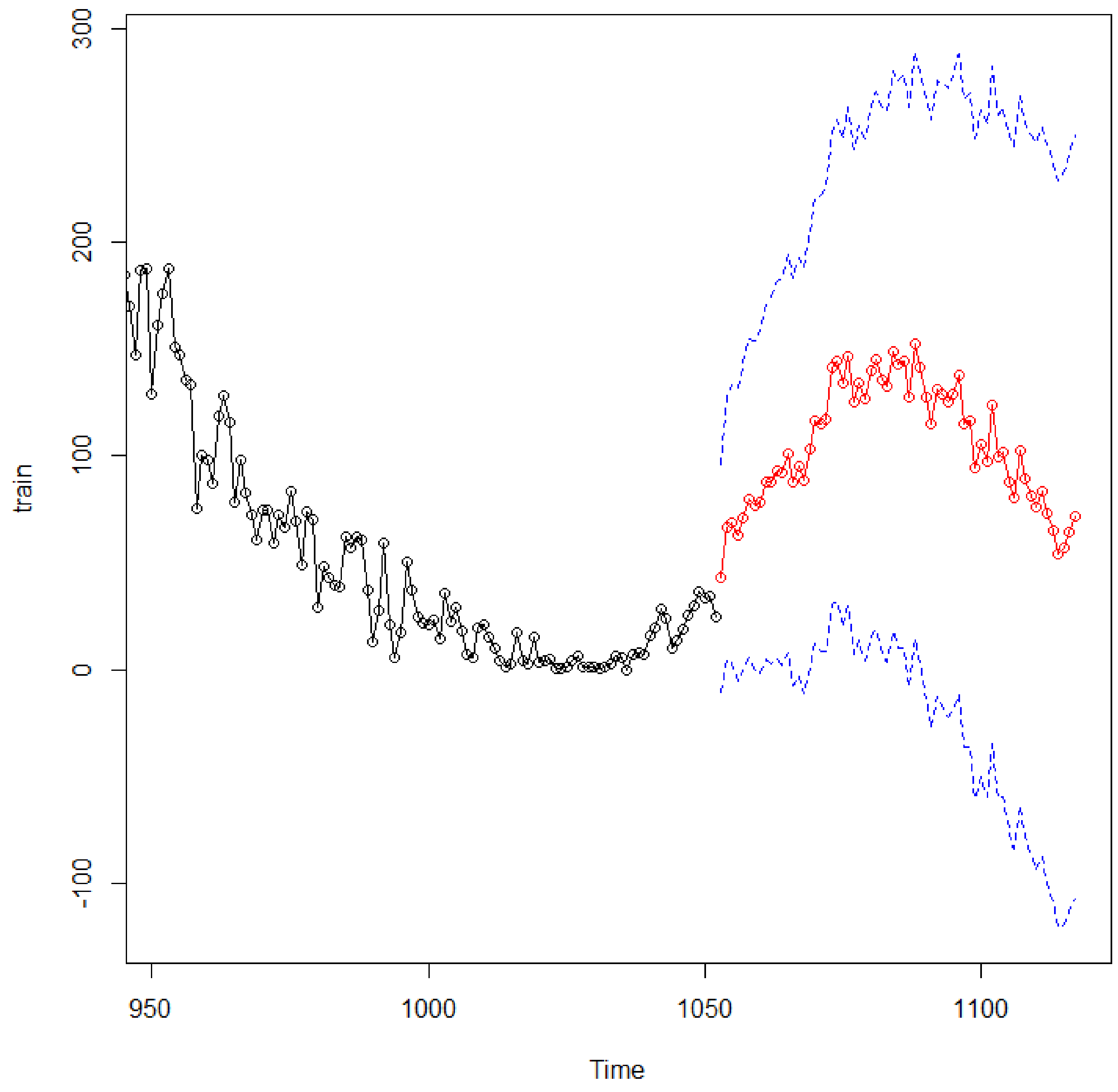


SARIMA(0,1,2)*(0,1,1)132

Residual Diagnostics

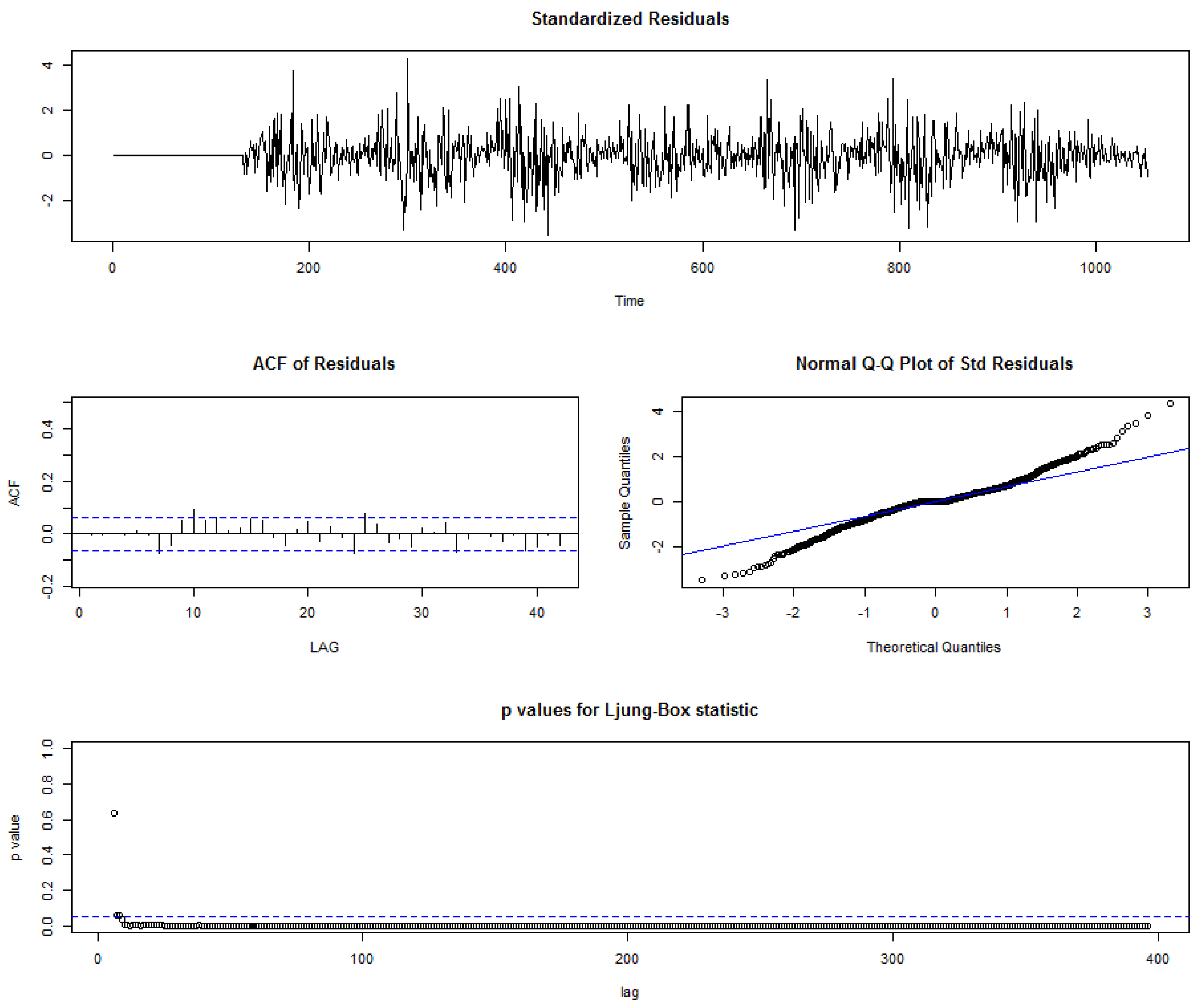


Forecasting Validation Set

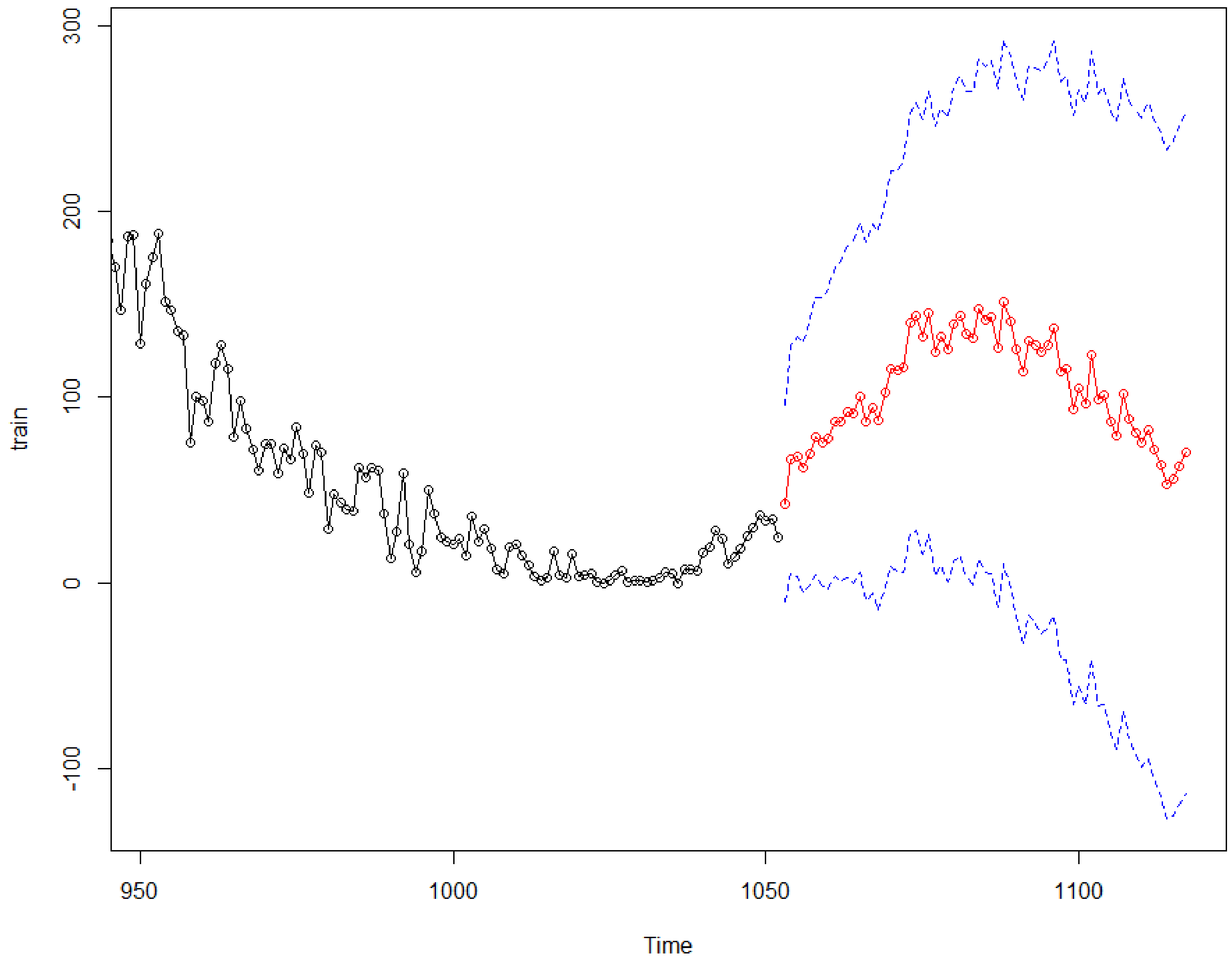


SARIMA(3,1,1)*(0,1,1)132

Residual Diagnostics

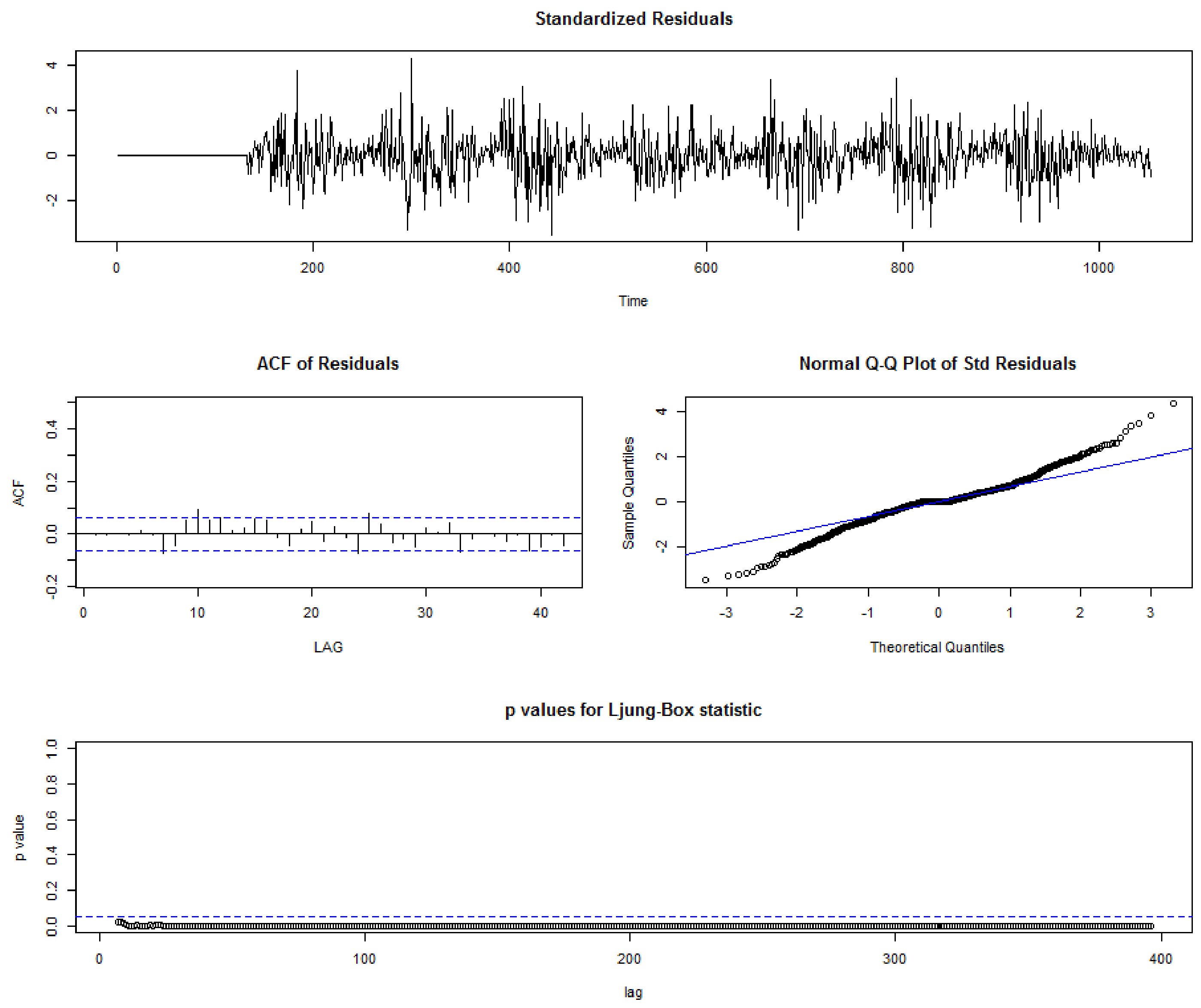


Forecasting Validation Set

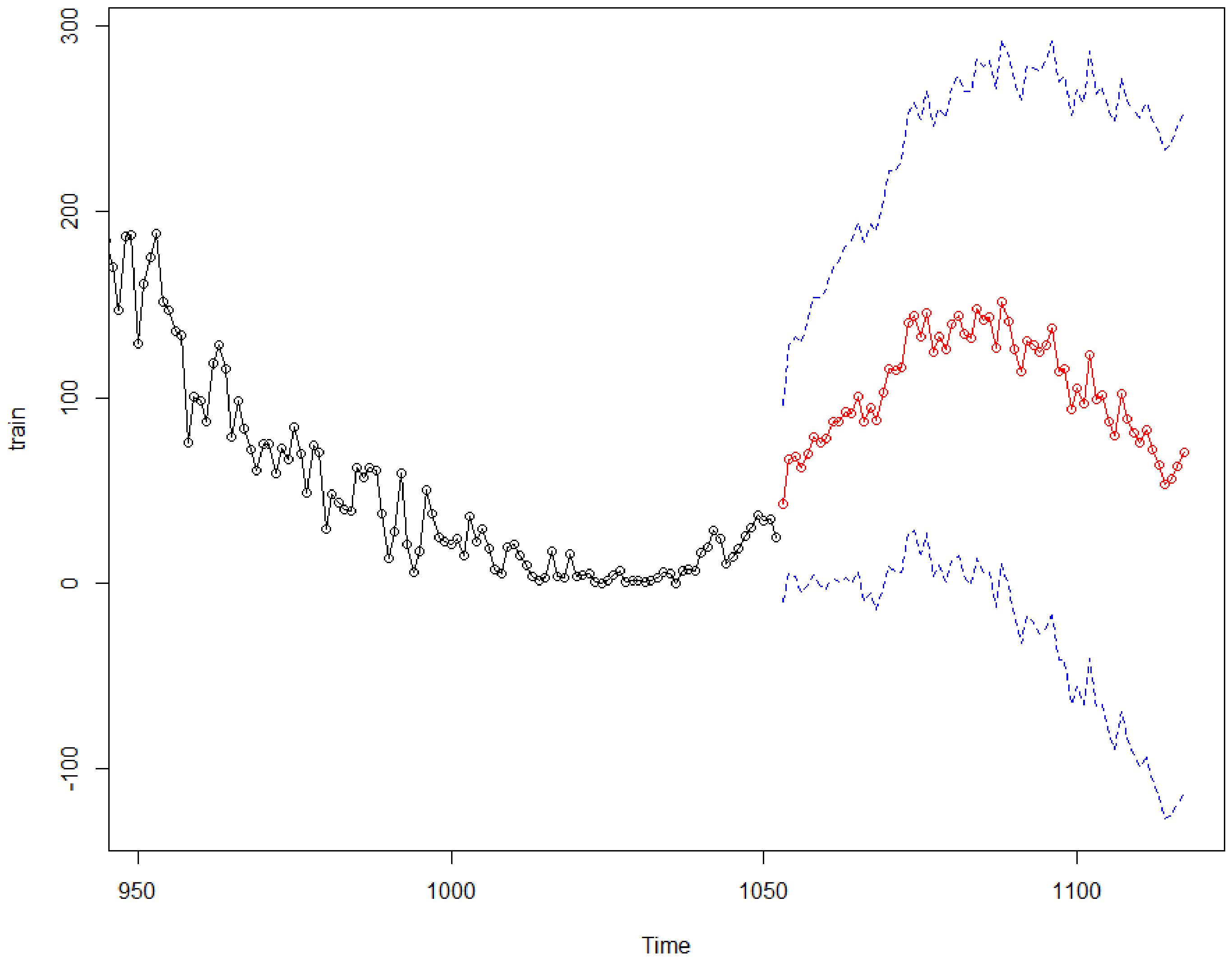


SARIMA(2,1,3)*(0,1,1)132

Residual Diagnostics

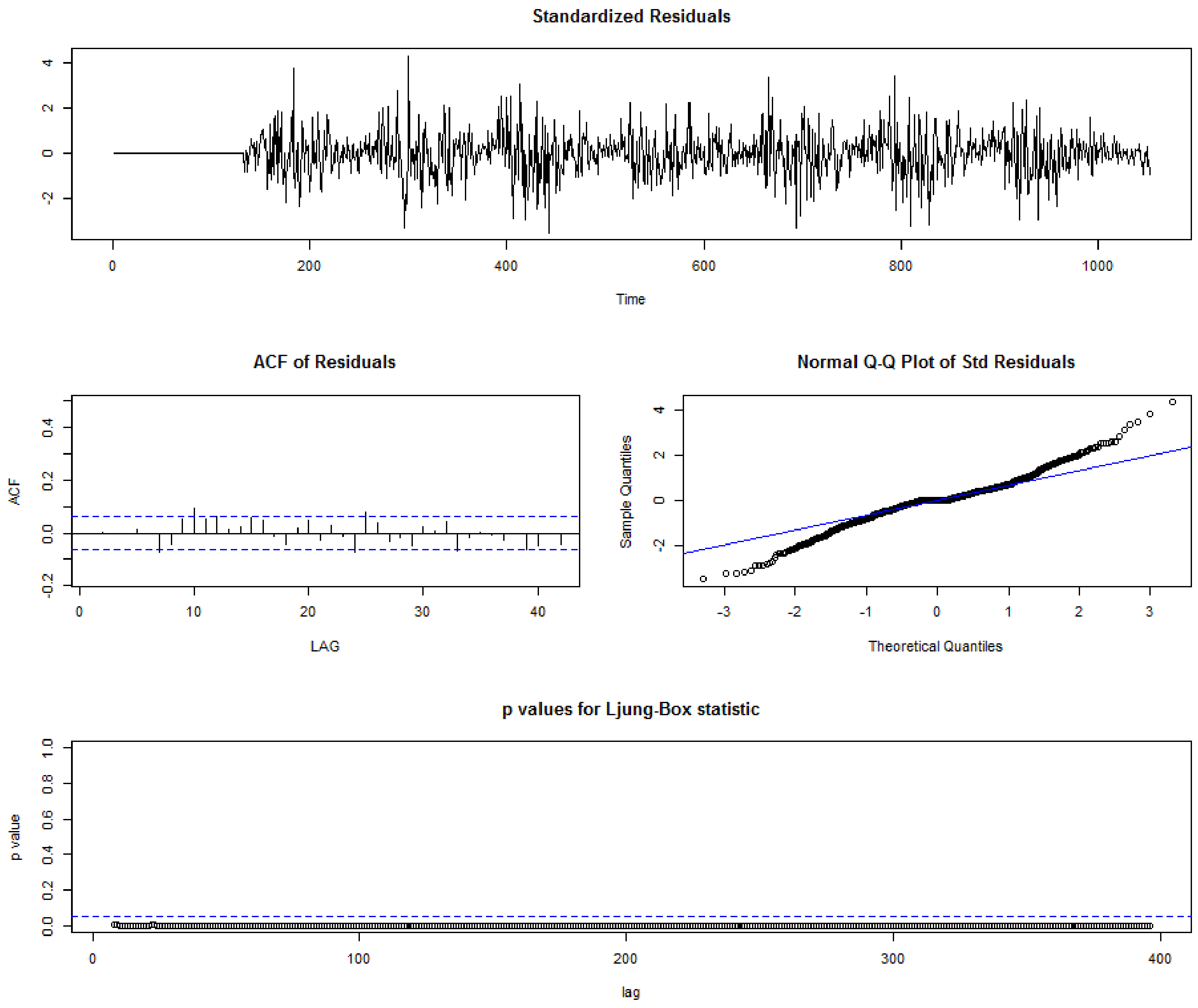


Forecasting Validation Set

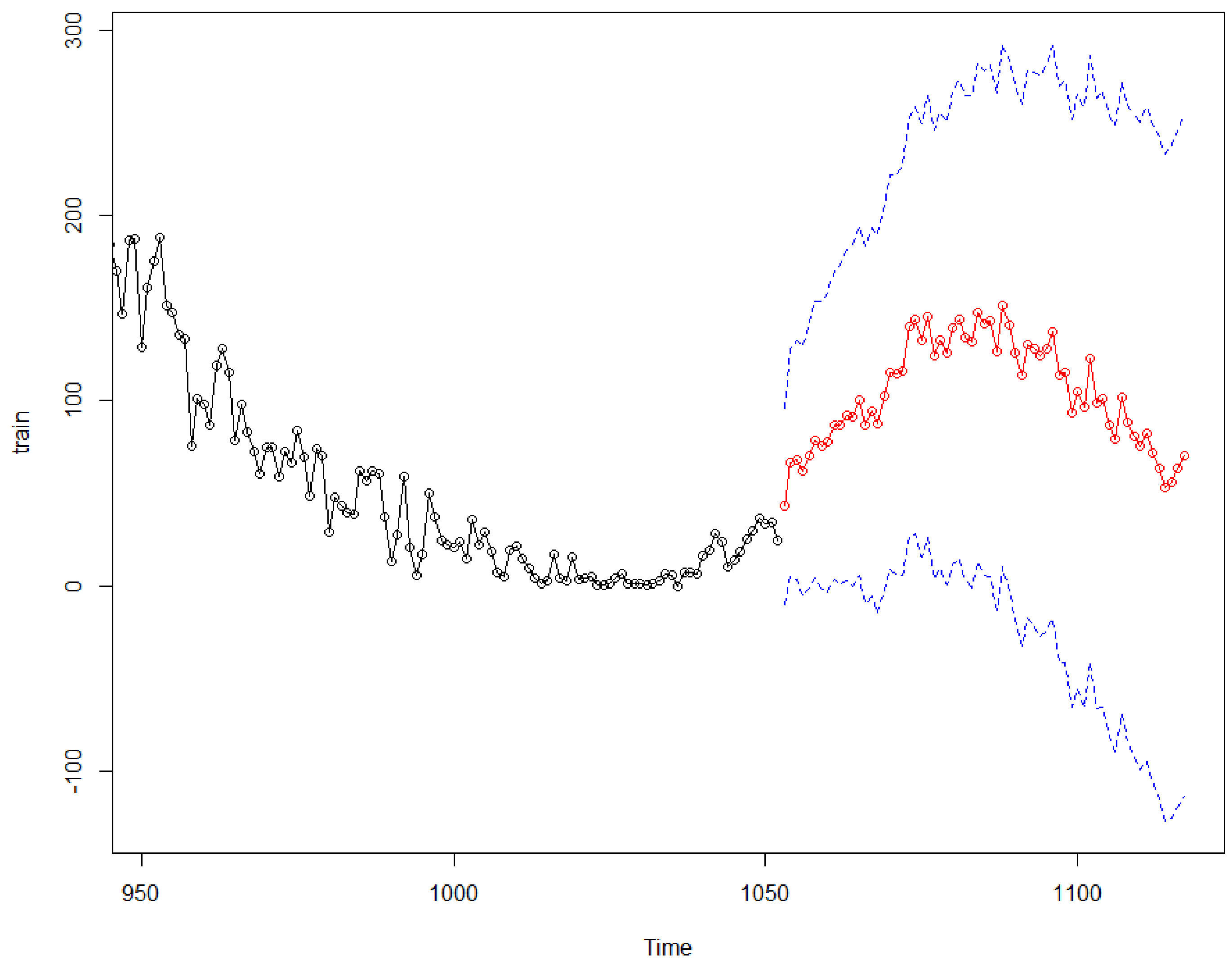


SARIMA(3,1,3)*(0,1,1)132

Residual Diagnostics

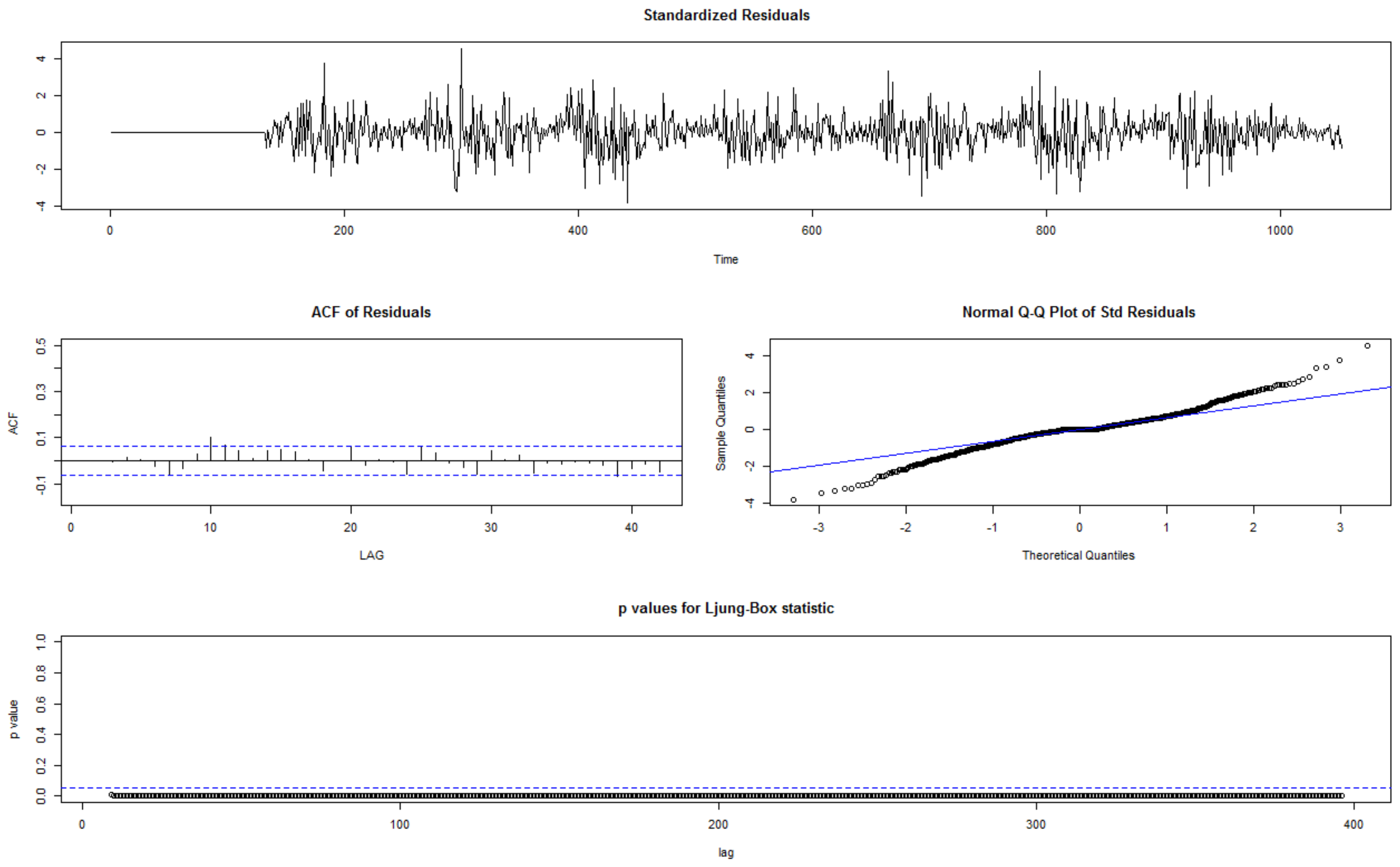


Forecasting Validation Set

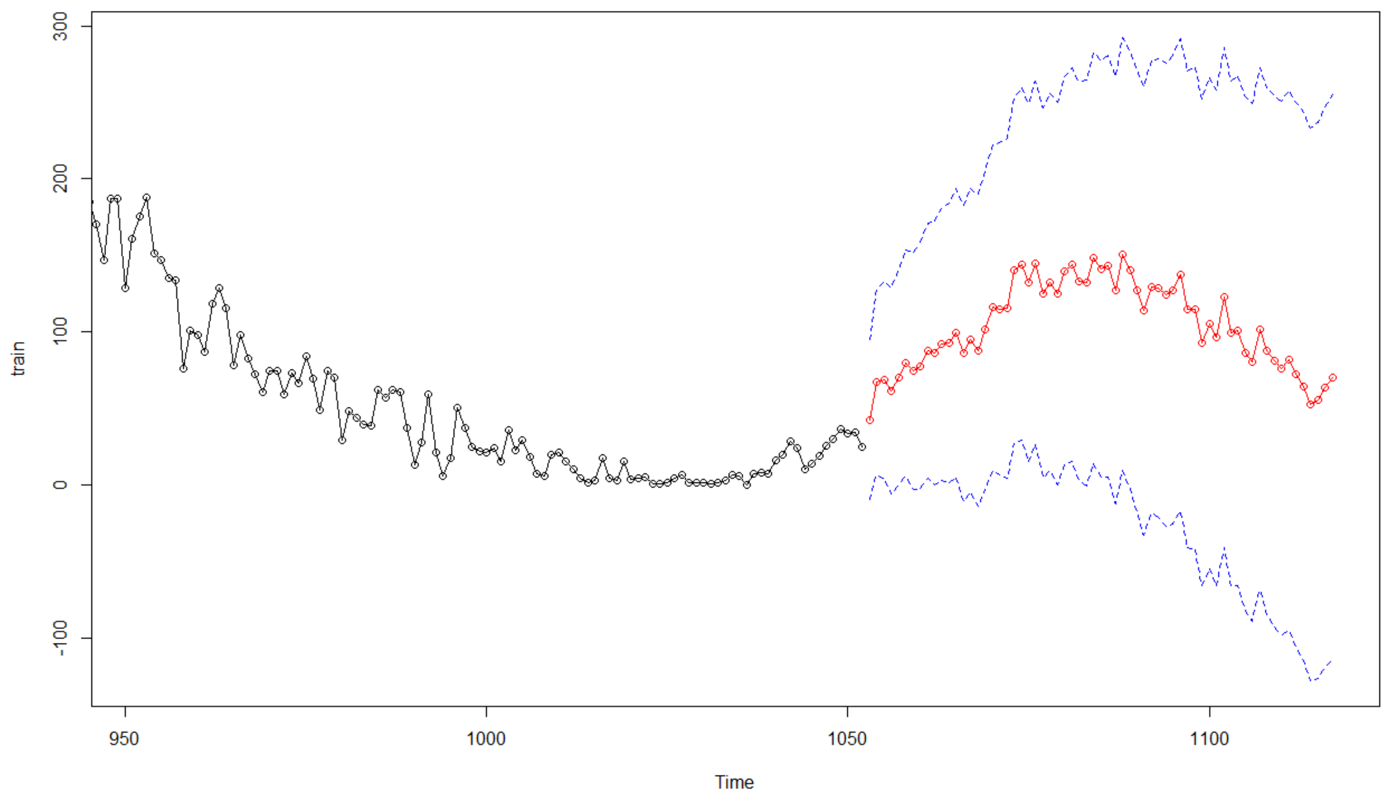


SARIMA(4,1,3)*(0,1,1)132

Residual Diagnostics

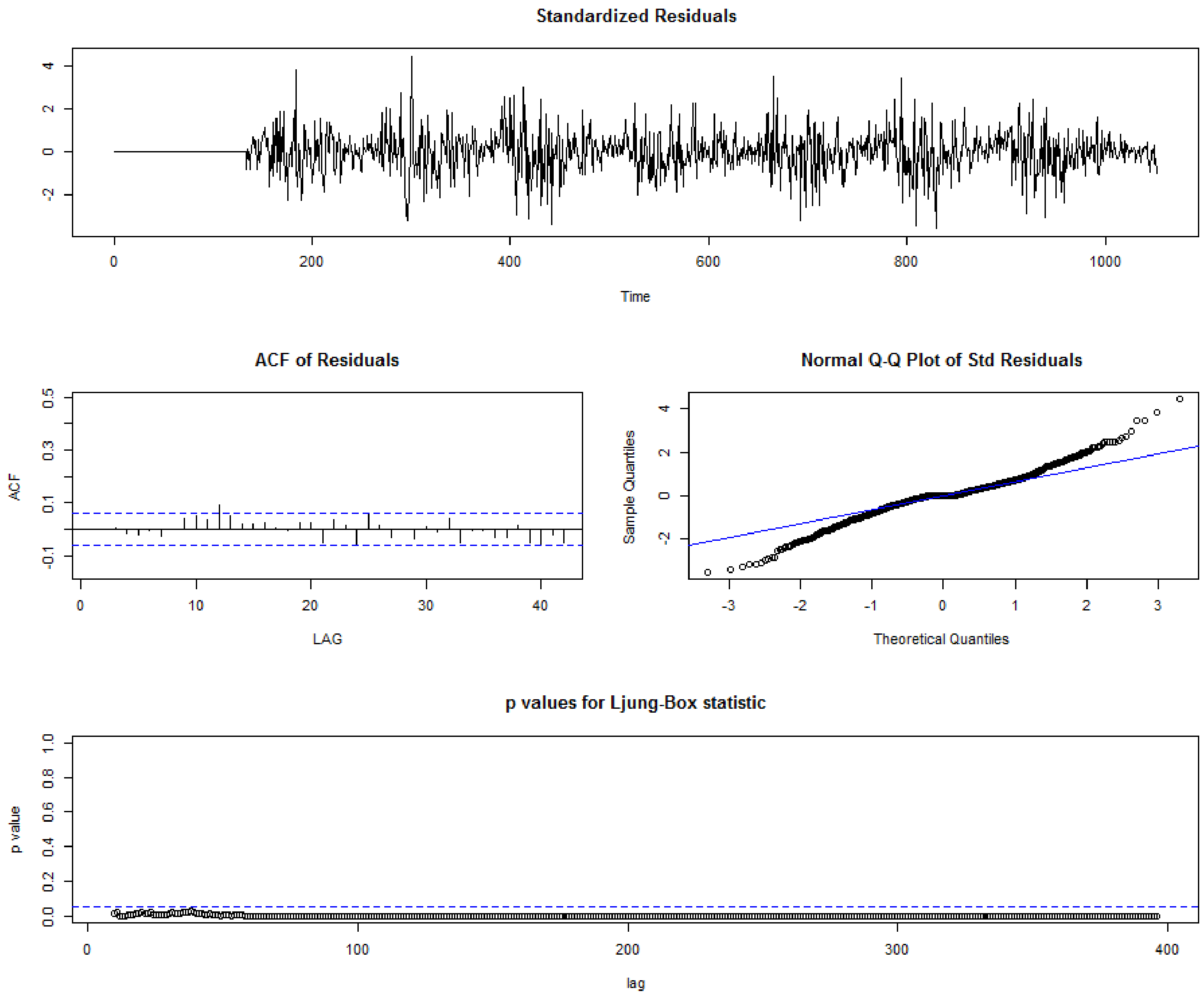


Forecasting Validation Set

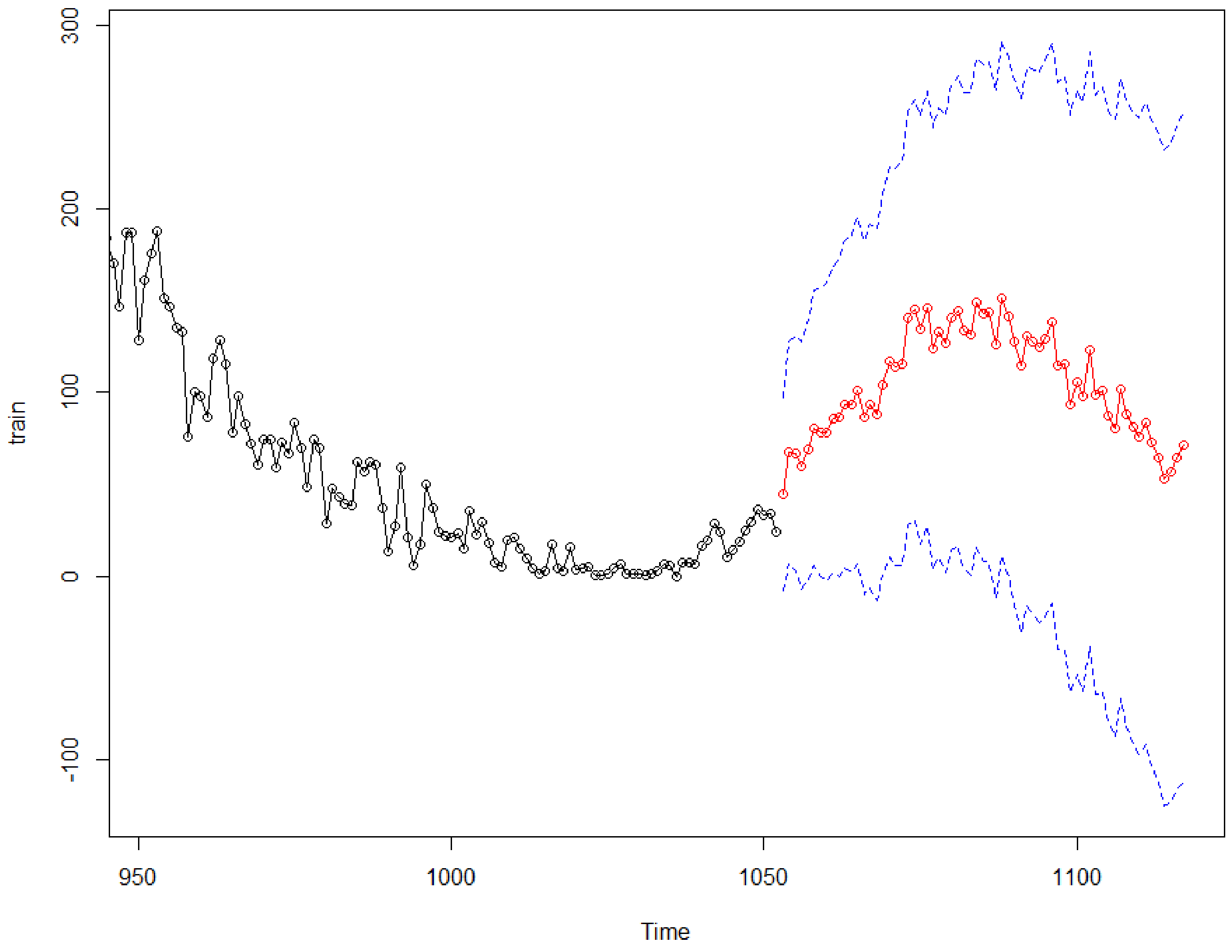


SARIMA(5,1,3)*(0,1,1)132

Residual Diagnostics



Forecasting Validation Set


300
200
100
0
-100
train
950
1000
1050
1100
Time