PSTAT 274 Final Project

Zheng Jing

1. Abstract

I conduct time series analysis to model the mean maximum temperature in Melbourne, Australia from 1971 to 1990. Melbourne has its unique temperature distribution yearly. The weather is warm at the beginning and end of the year, and turns cold in the middle of the year. By studying the change of temperature in Melbourne in the past 20 years, we can study the change of temperature in Melbourne throughout the years, and study the influence of global warming on Melbourne’s climate, and also fit a model to predict the future temperature in Melbourne. As a result, I fit a SARIMA model and make accurate predictions with reasonable error bound for the future temperature in Melbourne.

2. Introduction

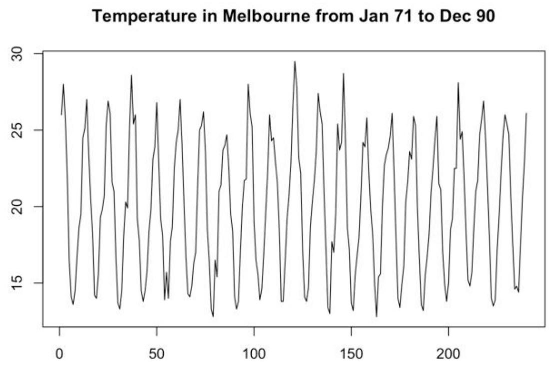
The dataset is released by “Australian Bureau of Meteorology”, and it describes the mean maximum temperature in Melbourne. My goal is to study the pattern of temperatures in Melbourne over the years and make predictions for the next few years.

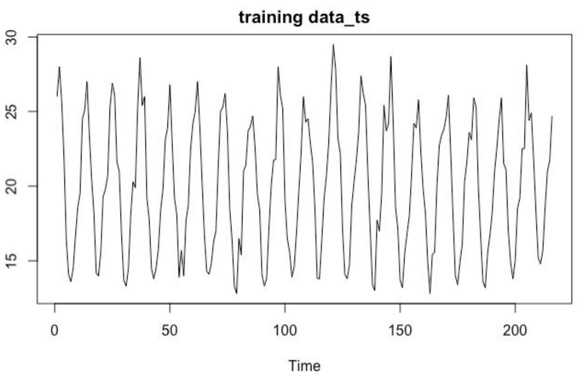
The dataset is documented by Rob Hyndman and Yangzhouran Yang at tsdl: Time Series Data Library (https://pkg.yangzhuoranyang./tsdl/). The dataset is released by Australian Bureau of Meteorology. The dataset records the monthly data of mean maximum temperature in Melbourne (Celsius) from January 1971 to December 1990. The dataset contains 240 observations for 20 consecutive years. An obvious seasonarity with s = 12 can be observed from the time series plot and no obvious trend exists in the original data. Moreover, employing the idea from machine learning class, I divided the dataset into training and testing. I used the training set which consists of the first 216 observations to fit a model, and use the last 24 observations to test our model.

I used R software to draw time series plot, acf/pacf, and perform data transformation. In particular, I performed Box-Cox transformation and differencing to obtain stationarity. Due to the existence of seasonality, I fit SARIMA models. The best model was selected in terms of AICc, the significance of coefficients, root behavior, rule of parsimony, and residuals plot. Eventually, I obtained two appropriate models, which can give accurate predictions on 1991 temperatures at Melbourne.

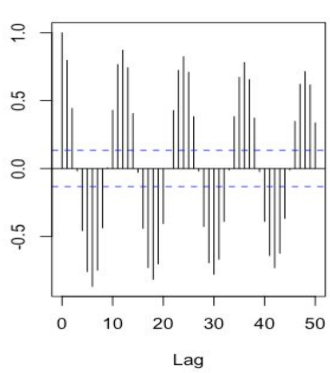
3. Preliminary Analysis

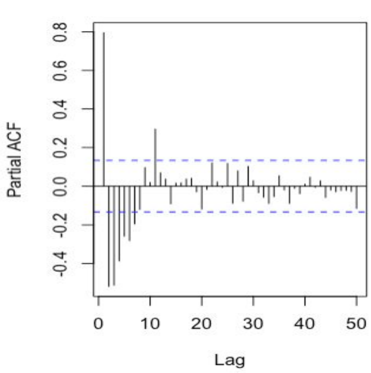
Plot the time series of original data (240 observations) and training dataset (216 observations).





According to the original time series plot, the temperature from 1971 to 1990 are bounded between 5 and 30 Celsius. There does not exist any obvious trend, but a seasonality exists with s = 12. A slight change of variance exists on the plot, but can be ignored. There does not exist any sharp or extreme values in the original time series plot. Moreover, the acf/pacf of the original data is shown below:


ACF

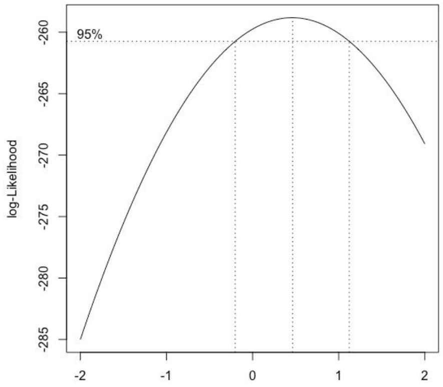

PACF

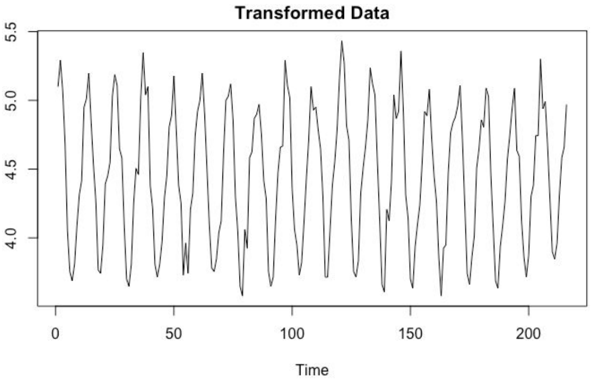
The ACF shows a seasonal component with s =12. The acf remains nonzero for a long period, so a trend may exist.

4. Obtain Stationarity

4.1 Box-Cox Transformation

Perform Box-Cox to reduce the variance and make the data more normal. The optimal lamda returned by box-cox is 0.464, but will use 0.5 for the convenience of transforming the data back.

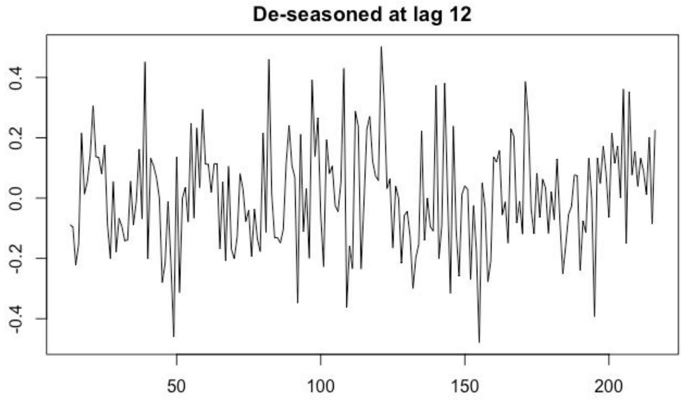


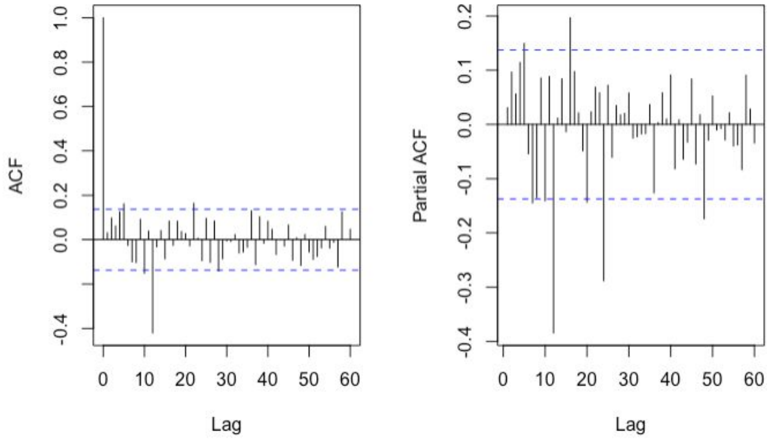


After the transformation, the variance reduces significantly from 19.5087 to 0.2481.

4.2 Differencing the data

In order to reach the stationarity, we need difference the data. Due to the existence of seasonarity, I differenced the data at lag = 12 to eliminate the seasonality first .





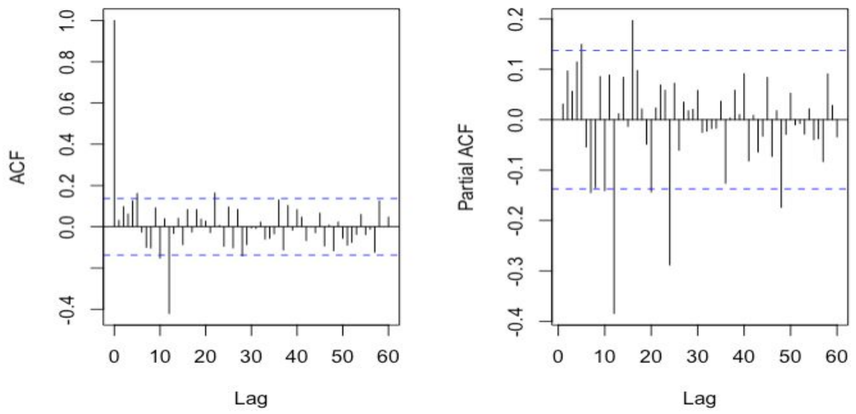
After the differencing, the variance decreases from 0.248 to 0.03, which indicates the step is significant. The acf/pacf are shown above. Acf falls inside the C.I. after lag 10, so the trend has been eliminated. But in case, I will further difference the data at lag = 1. It turns out the variance increases *from 0.03 to 0.06* after de-trend, which indicates we overdifferenced the data.

5. Model Identification & Parameter Estimation

5.1 Model Identification

Since the seasonality exists in my dataset, I will fit SARIMA model. Since I only differenced the data once at lag =12, the parameter d = 0 and D = 1. For the other parameters p, q, P, Q, I am going to run a for loop to test all the combinations, and then use MLE method to estimate the parameters. (Recall acf/pacf of the stationary data)

ACF **PACF**



In the following, I will identify P and Q first, and then identify p and q.

Seasonal Part (P&Q)

Look at ACF at lags that are multiples of 12 to identify Q. The ACF is significant only at lag = 12, and lie inside the confidence interval at lag = 24,36,48….. So we suspect a MA seasonal component exists and Q = 1.

Then look at the PACF at lags that are multiples of 12 to identify P. The pacf decreases exponentially from lag = 12 to 24 to 36 to 48 (tails off), so we suspect P = 0 and SMA(1) model for the seasonal part. Another interpretation is that pacf cuts off at lag = 24. In other words, we assume the pacf has a spike at lag 24 and then lie inside the C.I. boundaries for all lags after 24. Therefore, P = c(0,2).

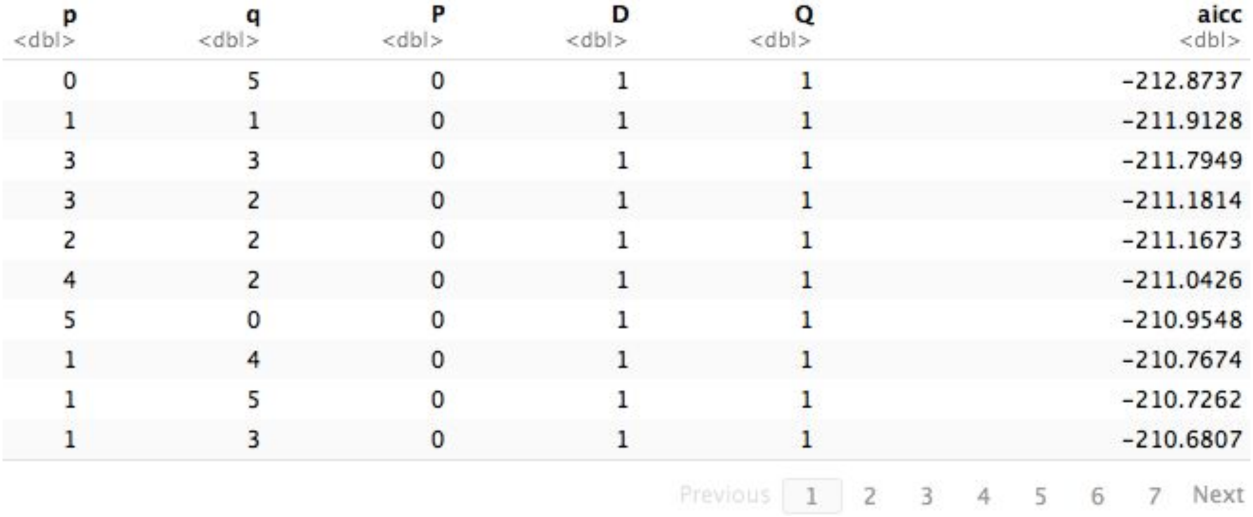
Non-seasonal Part (p&q)

Look at the acf/pacf of a single period to identify p & q. Since the first period is always influenced by white noise, we look at not only the first but also the next few periods to identify p & q. Unfortunately, acf/pacf has no apparent signs of cutting/tailing off, so I decided to run a loop by choosing q from c(0,1,2,3,4,5) and p from c(0,1,2,3,4,5). Both p&q are up to five, considering the rule of parsimony. If none of them work well, I will come back to this step and fit more complex models.

5.2 Fit Models

In order to select the best model efficiently from the big pool, I created a “for loop” by setting Q = 1 and P = c(0,2) and choosing q from c(0,1,2,3,4,5) and p from c(0,1,2,3,4,5). In total, there are sixty nine models. In the dataframe below, the models are ranked in terms of their AICc from the lowest to highest. (only the top 10 candidates are shown in the table, who have the lowest AICc)

There are only five parameters in the data frame because we know d=0 for all models.



p <dbl>	q <dbl>	P <dbl>	D <dbl>	Q <dbl>	aicc <dbl>
0	5	0	1	1	-212.8737
1	1	0	1	1	-211.9128
3	3	0	1	1	-211.7949
3	2	0	1	1	-211.1814
2	2	0	1	1	-211.1673
4	2	0	1	1	-211.0426
5	0	0	1	1	-210.9548
1	4	0	1	1	-210.7674
1	5	0	1	1	-210.7262
1	3	0	1	1	-210.6807

Previous
1
2
3
4
5
6
7
Next

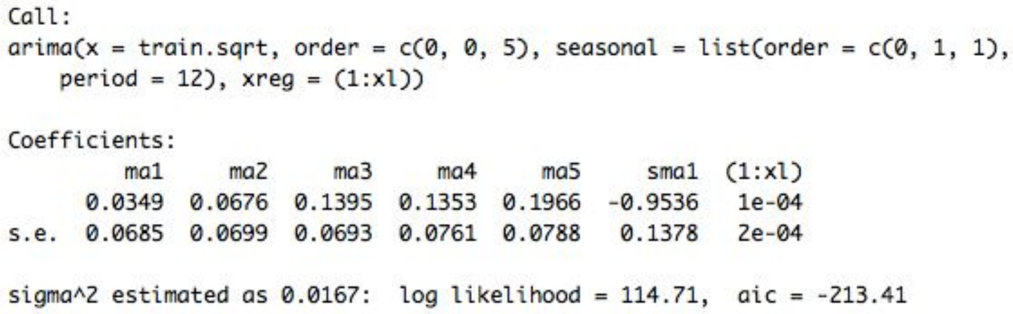
The top seven candidates are listed as follows:

1. SARIMA(0,0,5)*(0,1,1)
2. SARIMA(1,0,1)*(0,1,1)
3. SARIMA(3,0,3)*(0,1,1)
4. SARIMA(3,0,2)*(0,1,1)
5. SARIMA(2,0,2)*(0,1,1)
6. SARIMA(4,0,2)*(0,1,1)
7. SARIMA(5,0,0)*(0,1,1)

5.3 Coefficient Estimation

For each model listed above, I use the “MLE” method to estimate the coefficients. Then we check whether 0 is contained in the 95% confidence intervals, in order to determine the significance level of the coefficients. If a coefficient is not significant, I will fix that coefficient to be zero and refit the model.

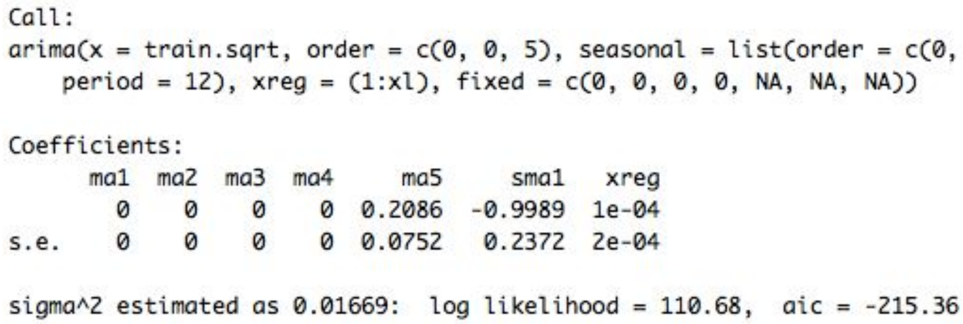
SARIMA(0,0,5)(0,1,1)*



```
Call:
arima(x = train.sqrt, order = c(0, 0, 5), seasonal = list(order = c(0, 1, 1),
    period = 12), xreg = (1:xl))

Coefficients:
         ma1     ma2     ma3     ma4     ma5     sma1   (1:xl)
      0.0349  0.0676  0.1395  0.1353  0.1966  -0.9536   1e-04
s.e.  0.0685  0.0699  0.0693  0.0761  0.0788   0.1378   2e-04

sigma^2 estimated as 0.0167:  log likelihood = 114.71,  aic = -213.41
```

The coefficients of ma1, ma2, ma3, ma4 are not significant. To improve the model, I will refit the model as follows, by fixing those coefficients to be zero.

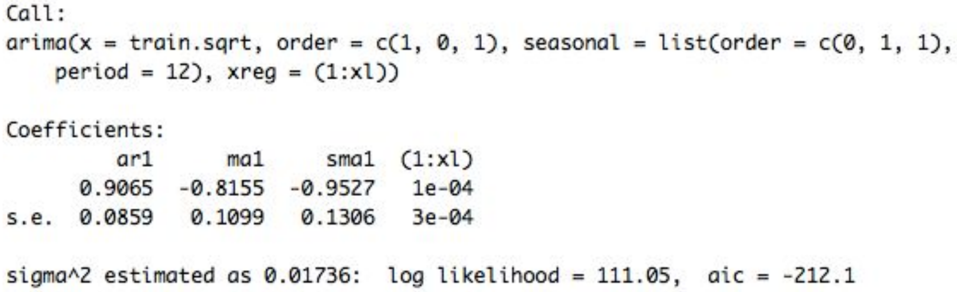
SARIMA(0,0,5)(0,1,1) -- fixed the coefficients of ma1,2,3,4 as zero*



```
Call:
arima(x = train.sqrt, order = c(0, 0, 5), seasonal = list(order = c(0, 1, 1),
    period = 12), xreg = (1:xl), fixed = c(0, 0, 0, 0, NA, NA, NA))

Coefficients:
      ma1  ma2  ma3  ma4     ma5     sma1   xreg
        0    0    0    0  0.2086  -0.9989  1e-04
s.e.    0    0    0    0  0.0752   0.2372  2e-04

sigma^2 estimated as 0.01669:  log likelihood = 110.68,  aic = -215.36
```

The AICc decreases, which indicates the model with fixed coefficients performs better. However, since the coefficient of sma part is very close to 1, I suspect a unit root exist in the model. But at this moment, I will save this model for later diagnostic and abandon the previous one.

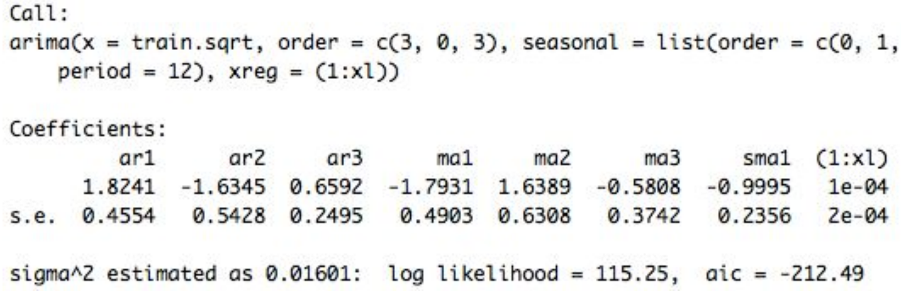
SARIMA(1,0,1)(0,1,1)*



```
Call:
arima(x = train.sqrt, order = c(1, 0, 1), seasonal = list(order = c(0, 1, 1),
    period = 12), xreg = (1:xl))

Coefficients:
         ar1      ma1     sma1  (1:xl)
      0.9065  -0.8155  -0.9527   1e-04
s.e.  0.0859   0.1099   0.1306   3e-04

sigma^2 estimated as 0.01736:  log likelihood = 111.05,  aic = -212.1
```

All the three coefficients are significant, so this is a good candidate for our final model. This is a good model also in terms of parsimony. So I will save this model for later diagnostic.

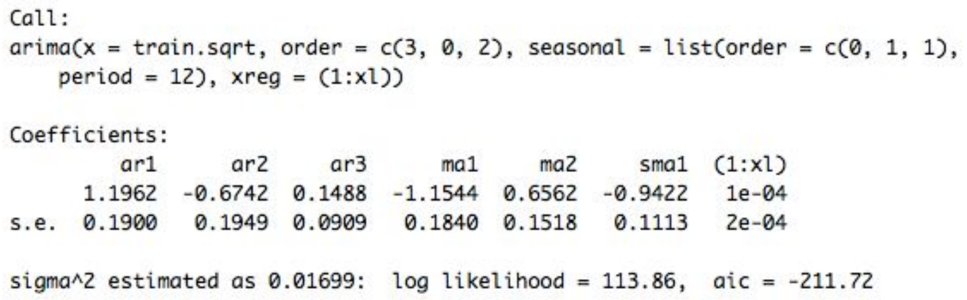
SARIMA(3,0,3)(0,1,1)*



```
Call:
arima(x = train.sqrt, order = c(3, 0, 3), seasonal = list(order = c(0, 1, 1),
    period = 12), xreg = (1:xl))

Coefficients:
         ar1      ar2     ar3      ma1     ma2      ma3     sma1  (1:xl)
      1.8241  -1.6345  0.6592  -1.7931  1.6389  -0.5808  -0.9995   1e-04
s.e.  0.4554   0.5428  0.2495   0.4903  0.6308   0.3742   0.2356   2e-04

sigma^2 estimated as 0.01601:  log likelihood = 115.25,  aic = -212.49
```

All the coefficients are significant except ma3, which indicates that we should try the model with q = 2 instead of 3: *SARIMA(3,0,2)*(0,1,1).* That happens to be our next model to be discussed in terms of AICc.

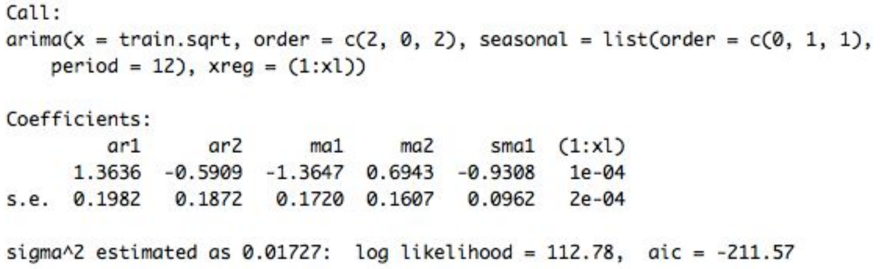
SARIMA(3,0,2)(0,1,1)*



```
Call:
arima(x = train.sqrt, order = c(3, 0, 2), seasonal = list(order = c(0, 1, 1),
    period = 12), xreg = (1:xl))

Coefficients:
         ar1      ar2     ar3      ma1     ma2     sma1  (1:xl)
      1.1962  -0.6742  0.1488  -1.1544  0.6562  -0.9422   1e-04
s.e.  0.1900   0.1949  0.0909   0.1840  0.1518   0.1113   2e-04

sigma^2 estimated as 0.01699:  log likelihood = 113.86,  aic = -211.72
```

All the coefficients are significant except ar3. which indicates that we should try the model with qp = 2 instead of 3: *SARIMA(2,0,2)*(0,1,1)*. That happens to be our next model to be discussed in terms of AICc.

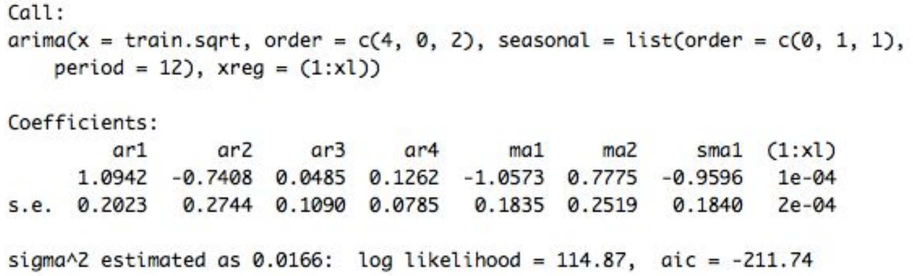
SARIMA(2,0,2)(0,1,1)*



```
Call:
arima(x = train.sqrt, order = c(2, 0, 2), seasonal = list(order = c(0, 1, 1),
    period = 12), xreg = (1:xl))

Coefficients:
         ar1      ar2      ma1     ma2     sma1  (1:xl)
      1.3636  -0.5909  -1.3647  0.6943  -0.9308   1e-04
s.e.  0.1982   0.1872   0.1720  0.1607   0.0962   2e-04

sigma^2 estimated as 0.01727:  log likelihood = 112.78,  aic = -211.57
```

All the coefficients are significant, so this is a good model, and I will save it for later diagnostic.

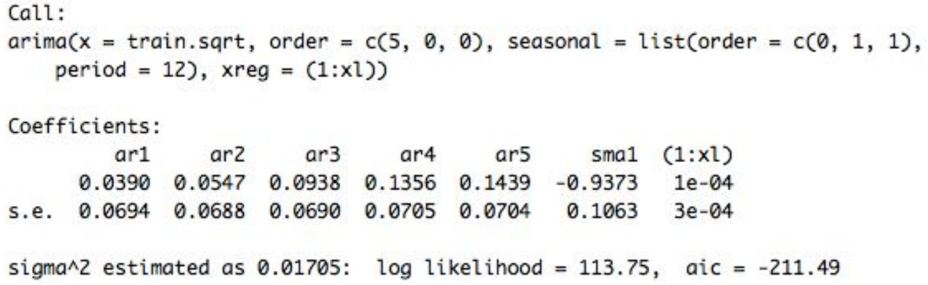
SARIMA(4,0,2)(0,1,1)*



```
Call:
arima(x = train.sqrt, order = c(4, 0, 2), seasonal = list(order = c(0, 1, 1),
    period = 12), xreg = (1:xl))

Coefficients:
         ar1      ar2     ar3     ar4      ma1     ma2     sma1  (1:xl)
      1.0942  -0.7408  0.0485  0.1262  -1.0573  0.7775  -0.9596   1e-04
s.e.  0.2023   0.2744  0.1090  0.0785   0.1835  0.2519   0.1840   2e-04

sigma^2 estimated as 0.0166:  log likelihood = 114.87,  aic = -211.74
```

Both ar3 and ar4 are insignificant, so this is not a good model and filter it out.

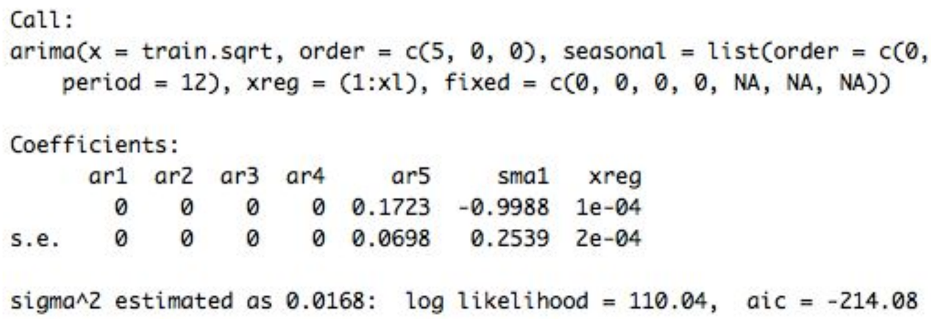
SARIMA(5,0,0)(0,1,1)*



```
Call:
arima(x = train.sqrt, order = c(5, 0, 0), seasonal = list(order = c(0, 1, 1),
    period = 12), xreg = (1:xl))

Coefficients:
         ar1     ar2     ar3     ar4     ar5     sma1  (1:xl)
      0.0390  0.0547  0.0938  0.1356  0.1439  -0.9373   1e-04
s.e.  0.0694  0.0688  0.0690  0.0705  0.0704   0.1063   3e-04

sigma^2 estimated as 0.01705:  log likelihood = 113.75,  aic = -211.49
```

The first four coefficients ar1, ar2, ar3, ar4 are not significant, so we should refit the model by fixing those insignificant coefficients as zero.

SARIMA(5,0,0)(0,1,1) --- fixed coefficients*



```
Call:
arima(x = train.sqrt, order = c(5, 0, 0), seasonal = list(order = c(0, 1, 1),
    period = 12), xreg = (1:xl), fixed = c(0, 0, 0, 0, NA, NA, NA))

Coefficients:
      ar1  ar2  ar3  ar4     ar5     sma1   xreg
        0    0    0    0  0.1723  -0.9988  1e-04
s.e.    0    0    0    0  0.0698   0.2539  2e-04

sigma^2 estimated as 0.0168:  log likelihood = 110.04,  aic = -214.08
```

The AICc decreases, which indicates the model with fixed coefficients performs better. However, since the coefficient of sma part is very close to 1, I suspect a unit root exist in the model. But at this moment, I will save this model for later diagnostic and abandon the previous one.

(Notice that intercepts are not discussed above because they are so close to zero for all models and zero is contained in all the confidence intervals, so we can just ignore the intercepts.)

5.4 Model Summary

Now we have four models left to choose from:
Model 1: *SARIMA(1,0,1)*(0,1,1)*
$(1 - 0.9065 B) \nabla 12 X_t = (1 - 0.8155 B)(1 - 0.9527 B^{12}) Z_t$

Model 2: *SARIMA(2,0,2)*(0,1,1)*
$(1 - 1.3636 B - 0.5909 B^2) \nabla 12 X_t = (1 - 1.3647 B + 0.6943 B^2)(1 - 0.9308 B^{12}) Z_t$

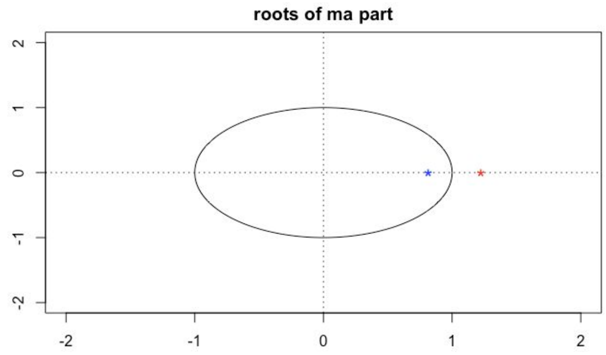
Model 3: *SARIMA(0,0,5)*(0,1,1) --- fixed coefficients*
$\nabla 12 X_t = (1 - 0.2086 B^5)(1 - 0.9989 B^{12}) Z_t$

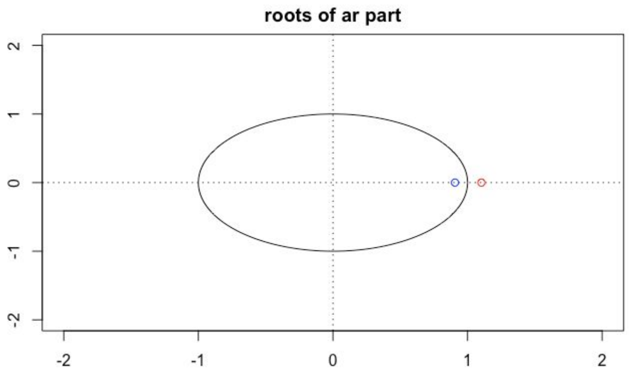
Model 4: *SARIMA(5,0,0)*(0,1,1) --- fixed coefficients*
$(1 - 0.1723 B^5) \nabla 12 X_t = (1 - 0.9988 B^{12}) Z_t$

6. Diagnostic Checking

6.1 Check Unit Roots for Invertibility & Causality

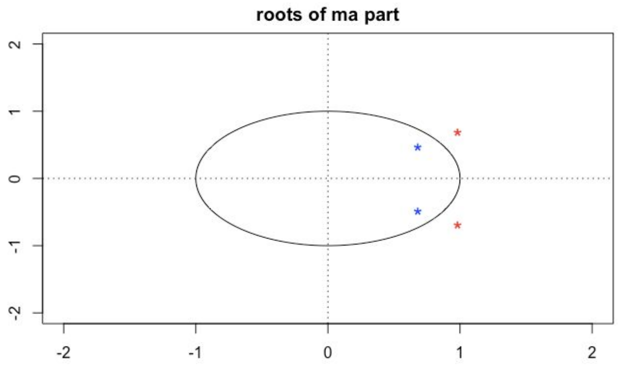
Model 1:

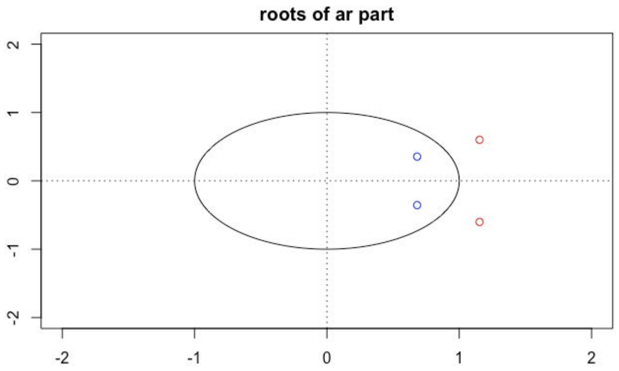




The red dots represent the roots and the blue dots represent their inverse.
All the roots of non-seasonal part lie outside the unit circle. Although the root of sma part = 1.004046 is very close to one, it is not considered as a unit root using 10^{-3} as the error bound condition. Hence, we conclude that model 1 is both invertible and causal.

Model 2:





The red dots represent the roots and the blue dots represent their inverse.
All the roots of non-seasonal part lie outside the unit circle. Although the root of sma part = 1.005994 is very close to one, it is not considered as a unit root using 10^{-3} as the error bound condition. Hence, we conclude that model 2 is both invertible and causal.

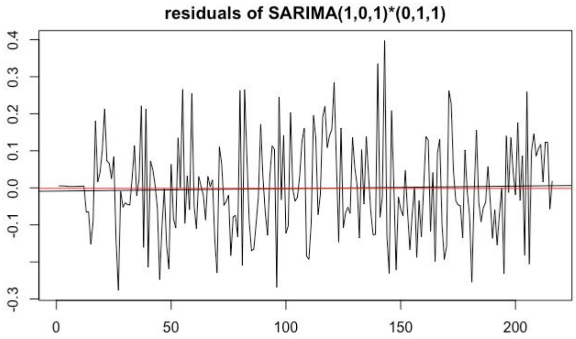
Model 3:
Model 3 has a unit root = 1.0001 in sma part, using 10^{-3} as the error bound.

Model 4:
Model 4 has a unit root = 1.00009 in sma part, using 10^{-3} as the error bound.

As a summary, model 3 and model 4 are not invertible due to the existence of unit roots in the SMA part. Therefore, we only two models left to choose from, which are model 1 and model 2.

6.2 Check Residuals

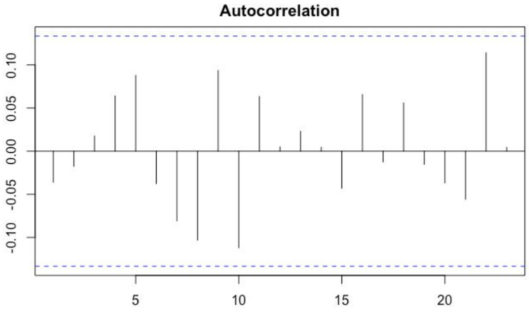
Check the residual plot and its acf/pacf, as well as the autoregression auto-fit.
Model 1



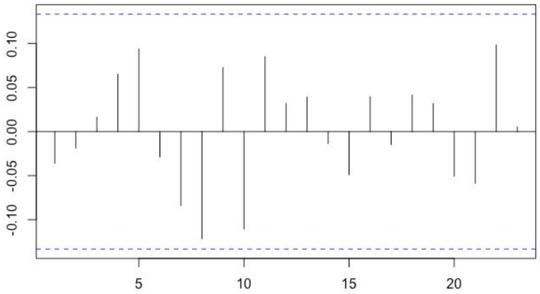
The mean of residuals is -0.00148 and the variance is 0.01647. There is no change of variance, trend, or seasonality in the residual plot. Moreover, the trend and mean overlap. Thus, the residuals of model 1 has similar behavior to the white noise.

Plot residuals acf/pacf:

ACF

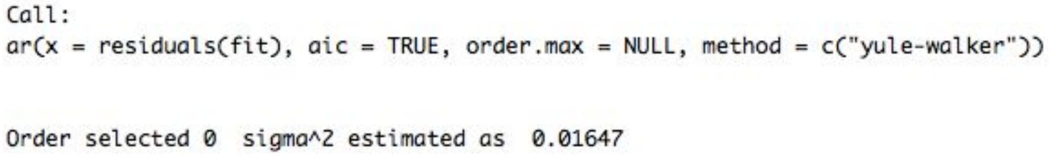


PACF



As shown in the figure above, acf/pacf lie inside the confidence interval at all lags.

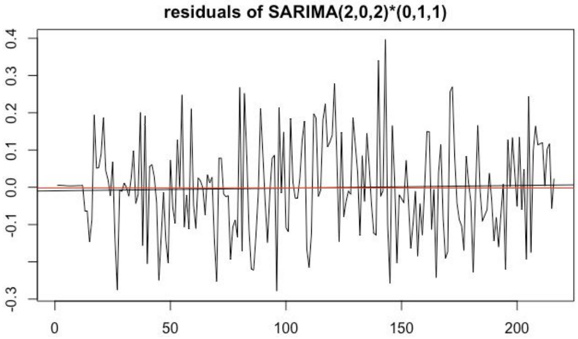
As shown in the figure below, autoregression auto-fit returns AR(0) model for the residuals, which further confirms that the residuals behave similarly to white noise.



```
Call:
ar(x = residuals(fit), aic = TRUE, order.max = NULL, method = c("yule-walker"))


Order selected 0  sigma^2 estimated as  0.01647
```

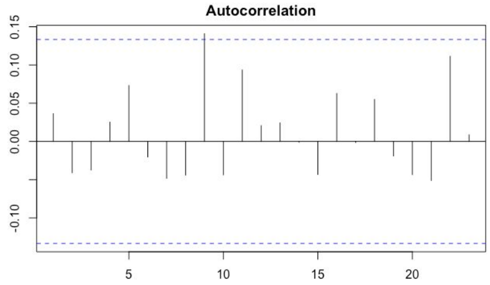
Model 2:

Plot residuals:

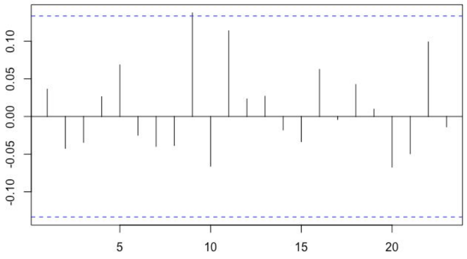


The mean of residuals is - 0.0017 and the variance is 0.016385.There is no change of variance, trend, or seasonality in the plot. Moreover, the trend and mean (red curve) overlap given the small sample size.

ACF

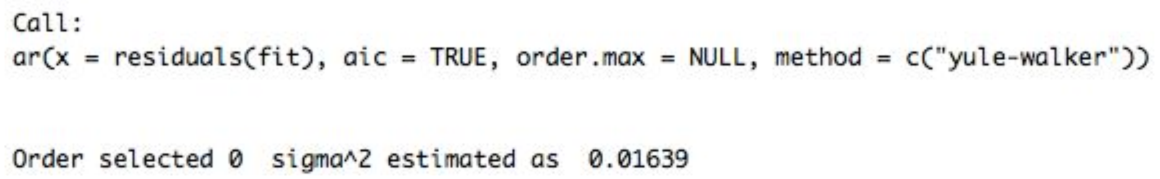


PACF



As shown in the figure above, the acf/pacf lie inside the confidence interval at all lags except lag = 9. But the acf/pacf at lag = 9 are very close to the C.I. boundary.

As shown in the figure below, autoregression auto-fit returns AR(0) model for the residuals, which further confirms residuals behave similarly to white noise.



```
Call:
ar(x = residuals(fit), aic = TRUE, order.max = NULL, method = c("yule-walker"))


Order selected 0  sigma^2 estimated as  0.01639
```

6.3 Independence Checking

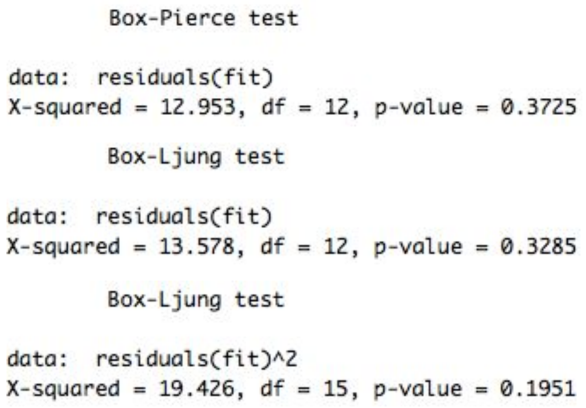
I will perform three tests: box-pierce, box-Ljung, McLeod Li, to check independence.

Model 1:

df = p+q+P+Q = 3

h = sqrt(n) = 15, where n = total number of observations = 240

As shown in the figure below, the p-values for all three tests are greater than 0.05, so we fail to reject the null hypothesis. In other words, model 1 passed all the independence tests.



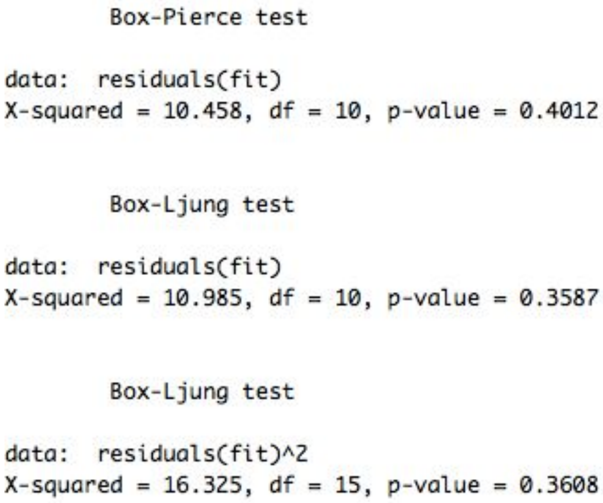
```
        Box-Pierce test

data:  residuals(fit)
X-squared = 12.953, df = 12, p-value = 0.3725
        Box-Ljung test

data:  residuals(fit)
X-squared = 13.578, df = 12, p-value = 0.3285
        Box-Ljung test

data:  residuals(fit)^2
X-squared = 19.426, df = 15, p-value = 0.1951
```

Model 2:

df= p+q+P+Q = 5

h = sqrt(n) = 15

As shown in the figure below, all the p-values are greater than 0.05, so model 2 also passes the independence tests.

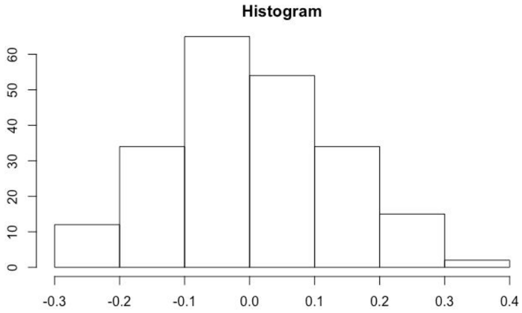


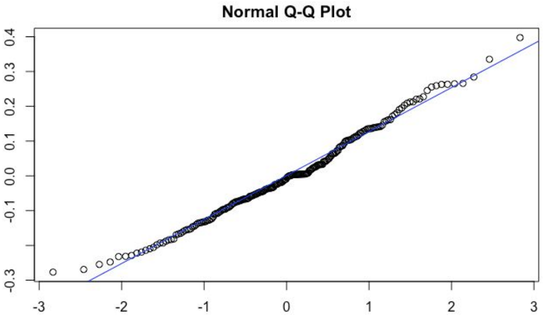
```
        Box-Pierce test

data:  residuals(fit)
X-squared = 10.458, df = 10, p-value = 0.4012


        Box-Ljung test

data:  residuals(fit)
X-squared = 10.985, df = 10, p-value = 0.3587


        Box-Ljung test

data:  residuals(fit)^2
X-squared = 16.325, df = 15, p-value = 0.3608
```

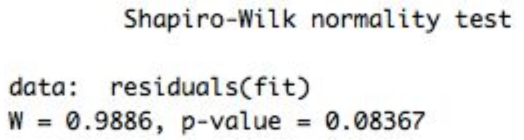
6.4 Normality Checking

I will use histogram, QQ-plot, and Shapiro-Wilk test

<u>Model 1:</u>



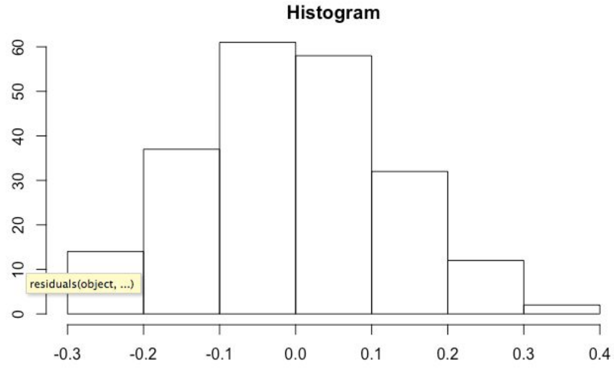


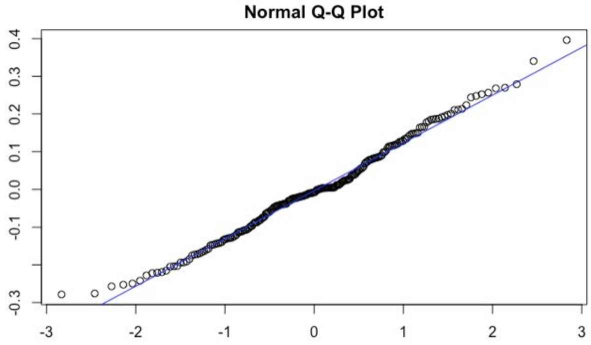


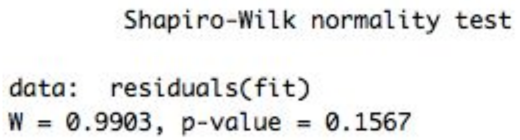
```
        Shapiro-Wilk normality test

data:  residuals(fit)
W = 0.9886, p-value = 0.08367
```

Given a small sample size, the histogram looks symmetric enough, but it skews to the right a bit.
The qq-line follows a linear trend, but it deviates a bit in the middle.
The p-value for Shapiro-Wilk test is 0.08367, which is greater than 0.05.
Hence, Model 1 passes all the normality test

<u>Model 2:</u>







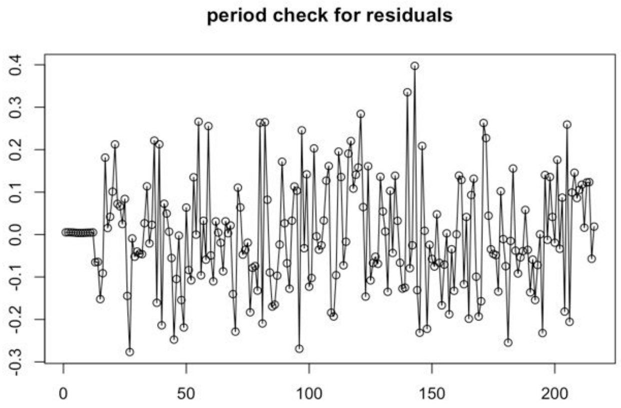
```
        Shapiro-Wilk normality test

data:  residuals(fit)
W = 0.9903, p-value = 0.1567
```

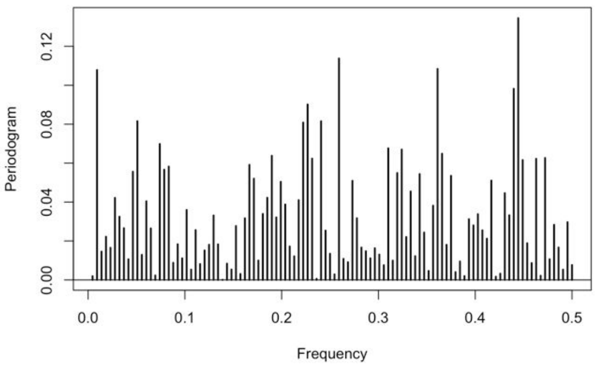
The histogram looks symmetric and the QQ-plot attaches closely to the linear trend. Moreover, model 2 passes the Shapiro-Wilk test with a larger p-value = 0.1567. Therefore, model 2 passes all the normality checking.

6.5 Spectral Analysis

I will check the periodicity of the residuals, by drawing periodogram plot and performing fisher's test and Kolmogorov-Smirnov test.

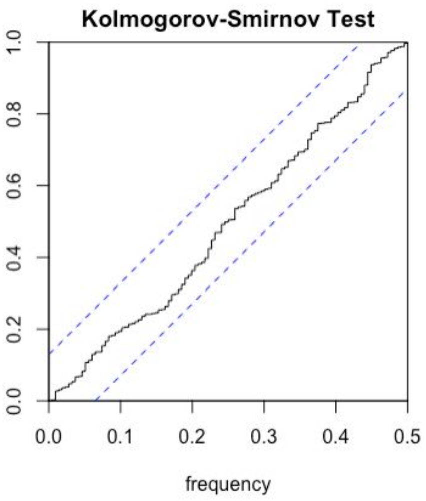
Model 1:



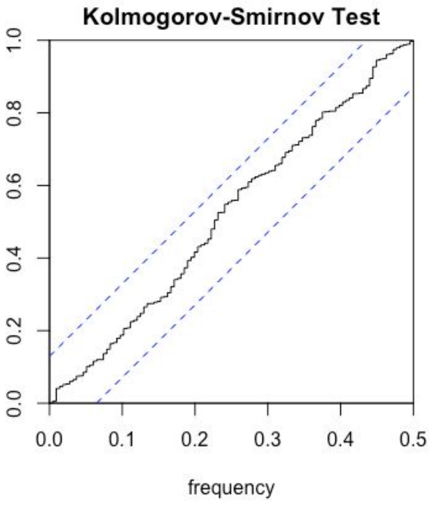


The periodogram are bounded above by 0.12, which are approximately zero for all frequencies. The p-value for fisher's test is 0.875, which is greater than 0.05. Therefore, we conclude that no periodicity exists in the residuals of model 1.

Model 1

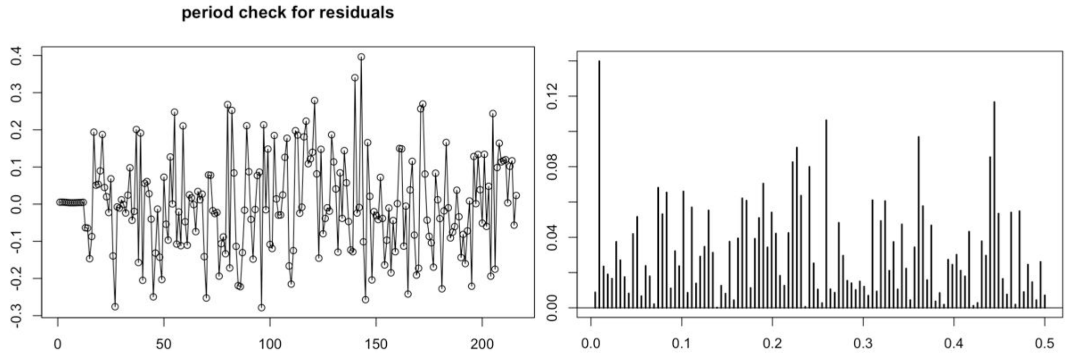


Model 2



As shown in the figure above (left), the curve does not exceed the boundaries. Thus, it further confirms that model 1 passes the Kolmogorov-Smirnov test. Hence, the residuals have no periodicity.

Model 2:



The periodogram are bounded above by 0.13 at all frequencies, which are approximately zero. Moreover, the p-value for fisher's test is 0.875, which is greater than 0.05, so no periodicity exists. Therefore, we conclude that no periodicity exists in the residuals of model 2. Moreover, as shown in the figure on the top of the page (right), the curve does not exceed the boundaries. Thus, it further confirms that model 2 passes the Kolmogorov-Smirnov test.

As a result, both model 1 and model 2 are satisfactory.

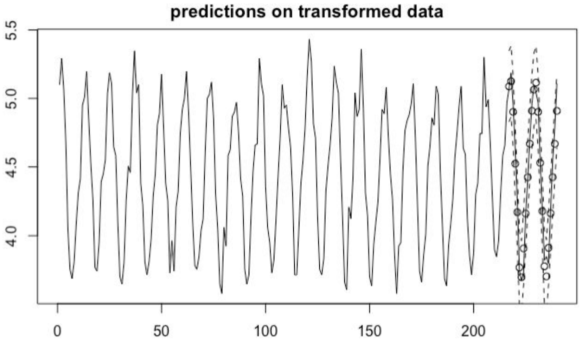
7. Final Model Selection

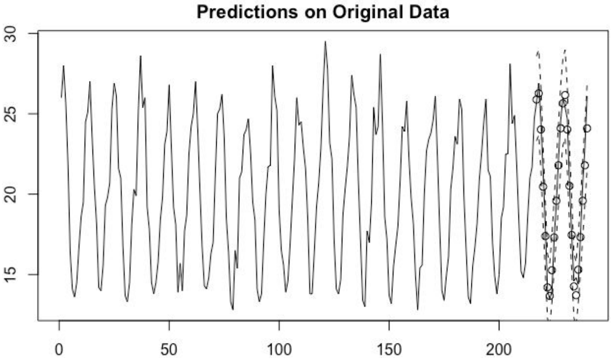
Both model 1 and model 2 pass all the diagnostic checkings. All the coefficients of both model 1 and model 2 are significant. Either one can be used as the final model for forecasting. As a result, I choose model 2 over model 1 because of its better performance in normality test.

Final Model: SARIMA(2,0,2)(0,1,1)*

$(1 - 1.3636 B - 0.5909 B^2) \nabla 12 X_t = (1 - 1.3647 B + 0.6943 B^2)(1 - 0.9308 B^{12}) Z_t$

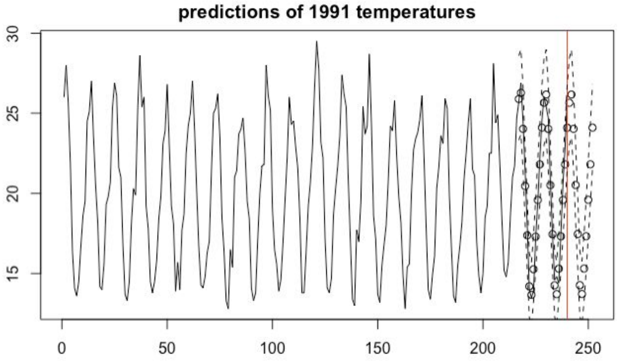
8. Evaluate the final model on Testing Set

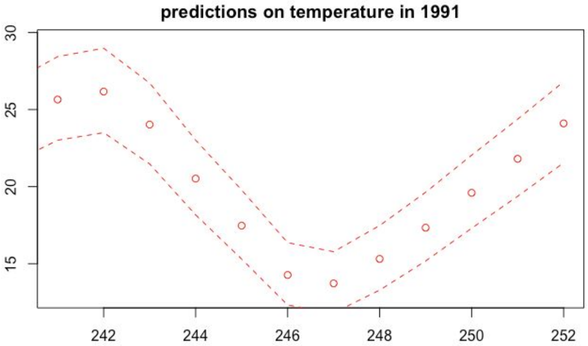




All the prediction points lie closely to the true values. Moreover, the confidence intervals of the predictions contain all the true values. Therefore, the model 2 is ready for forecasting.

9. Forecasting





I employed my final model to make predictions on 1991 temperature in Melbourne.
As shown in the figure above, the red points represent the predictions for the monthly temperature at Melbourne in 1991. The figure on the right enlarges those prediction points for 1991 temperature.

10. Conclusion

The final model we obtained is:

$$\boldsymbol{(1 - 1.3636 B - 0.5909 B^2) \nabla 12 X_t = (1 - 1.3647 B + 0.6943 B^2)(1 - 0.9308 B^{12}) Z_t}$$

Based on our times series model, we predict that the temperatures in 1991 in Melbourne will follow the same pattern as the previous years. The weather will get warm at the beginning and end of the year, and get cold in the middle of the year. In particular, we can also accurately predict the temperature for each month in 1991 with a reasonable error bound. For example, we predict the temperature in January 1991 in Melbourne to be 26 Celsius (plus minus 4), and 10 Celsius in July (plus minus 3).

11. Acknowledgment

I appreciate all the help from Professor Raya Feldman and my teaching assistant Yuanbo, who provided me many constructive suggestions in office hours. I would also like to thank my friend - Catherine Miao, who provided valuable advice for my project formatting and style.

12. References

Rob Hyndman and Yangzhuoran Yang (2018). tsdl: Time Series Data Library. v0.1.0. https://pkg.yangzhuoranyang./tsdl/.

"Mean maximum temperature in Melbourne from Jan 1971 to Dec 1990", *Australian Bureau of Meteorology.*

13. Appendix

Rmd file (in pdf format)

~~~
---
title: "PSTAT 274 Final Project"
author: "Zheng Jing (8675738)"
output:
  pdf_document: default
  html_document:
    df_print: paged
---

```{r}
#install.packages("devtools")
#devtools::install_github("FinYang/tsdl")
library(tsdl)
library(qpcR)
```

Data markets from Gauchospace
```{r}
tsdl
```



```{r}
k = 90
length(tsdl[[k]])
attr(tsdl[[k]], "subject")
attr(tsdl[[k]], "source")
attr(tsdl[[k]], "description")
```


Plot the time series data
```{r}
data = tsdl[[k]]
data = array(data)
data_ts <- ts(data)
ts.plot(data_ts,main  = "Temperature in Melbourne from Jan 71 to Dec 90")
ts.plot(data_ts,main  = "Temperature in Melbourne in 1971",xlim = c(0,12))
mean(data_ts)
var(data_ts)
```


Divide the data into training and testing set
```{r}
cut_point = length(data)*0.9

# 90 percent training
train_data = data[1:cut_point]
train_ts = ts(train_data)

# 10 percent testing
test_data = data[(cut_point+1):240]
test_ts = ts(test_data)
```
~~~

~~~
Plot training dataset
```{r}
ts.plot(train_ts,main  = "training data_ts")
mean(train_ts)
var(train_ts)
```


Plot the acf and pacf of training data
```{r acf_pacf}
op <- par(mfrow = c(1,2))
acf(train_ts,lag.max = 50)
pacf(train_ts,lag.max = 50)
par(op)
```

Box Cox Transformation
```{r message=FALSE,fig.height=6,fig.width=6,fig.show='hold'}
# Transform data using boxcox()
require(MASS)
bcTransform <- boxcox(train_ts ~ as.numeric(1:length(train_ts)))
lambda = bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
lambda
```

Plot sqrt data
```{r sqrt_tranform}
train.sqrt = sqrt(train_ts)
ts.plot(train.sqrt,main = "Transformed Data")
var(train.sqrt)
```

Plot acf and pacf of the sqrt data
```{r}
op <- par(mfrow = c(1,2))
acf(train.sqrt,lag.max = 50)
pacf(train.sqrt,lag.max = 50)
par(op)
```

Difference at lag 12
```{r}
train.season.diff12 <- diff(train.sqrt,12)
ts.plot(train.season.diff12,main = "De-seasoned at lag
12",ylab=expression(paste(nabla,12,Y)))
var(train.season.diff12)
```

Plot the acf and pacf of de-seasoned data
```{r}
op <- par(mfrow = c(1,2))
acf(train.season.diff12,lag.max =60, main = "De-Seasoned")
pacf(train.season.diff12,lag.max = 60, main = "De-Seasoned")
par(op)
```

difference at lag 1
~~~

~~~
```{r}
train.trend.diff1 <- diff(train.season.diff12,1)
ts.plot(train.trend.diff1,main = "De-trend Once",
ylab=expression(paste(nabla,1,nabla,12,Y)))
var(train.trend.diff1)
```

Plot the acf and pacf of stationary data
```{r}
op <- par(mfrow = c(1,2))
acf(train.season.diff12,lag.max =60, main = "de-seasoned")
pacf(train.season.diff12,lag.max = 60, main = "de-seasoned")
par(op)
```

Model identification
```{r}
P = c(0,2)
Q = 1
p = c(0,1,2,3,4,5)
q = c(0,1,2,3,4,5)
aicc_array = c()

# create a dataframe to compare AICC
aicc_data = data.frame()
col.P = c() # e
col.p = c() # i
col.q = c() # j
col.aicc = c()
xl = length(train.sqrt)
for (e in P){
  for (i in p){
    for (j in q){
      if (!(i==5 & j==5 & e==2) & !(i==3 & j==5 & e==2)){
      #if (!(i==5 & j==5) & !(i==3 & j==4 & e==2)){
        SARIMA.i.j.e.1 <- arima(train.sqrt,order = c(i,0,j),seasonal = list(order
= c(e,1,Q), period =12), xreg = (1:xl))
        col.P = c(col.P,e)
        col.p = c(col.p,i)
        col.q = c(col.q,j)
        col.aicc = c(col.aicc, AICc(SARIMA.i.j.e.1))
      }
    }
  }
  }

aicc_data = data.frame(p = col.p, q = col.q, P = col.P,D = rep(1,length(col.q)), Q
= rep(1,length(col.q)), aicc = col.aicc)
head(aicc_data[order(aicc_data$aicc),]) # from smallest to largest
```


```{r}
arima(train.sqrt,order = c(0,0,5),seasonal = list(order = c(0,1,1), period = 12),
xreg = (1:xl))
# coefficients not significant
```
~~~

~~~~
```{r}
arima(train.sqrt,order = c(0,0,5),seasonal = list(order = c(0,1,1), period = 12),
xreg = (1:xl),fixed = c(0,0,0,0,NA,NA,NA))
```

```{r}
arima(train.sqrt,order = c(1,0,1),seasonal = list(order = c(0,1,1), period = 12),
xreg = (1:xl))
# good model - parsimony & coefficient
```

```{r}
arima(train.sqrt,order = c(3,0,3),seasonal = list(order = c(0,1,1), period = 12),
xreg = (1:xl))
# unit root may exist
```

```{r}
arima(train.sqrt,order = c(3,0,2),seasonal = list(order = c(0,1,1), period = 12),
xreg = (1:xl))
```

```{r}
arima(train.sqrt,order = c(2,0,2),seasonal = list(order = c(0,1,1), period = 12),
xreg = (1:xl))
```


```{r}
arima(train.sqrt,order = c(4,0,2),seasonal = list(order = c(0,1,1), period = 12),
xreg = (1:xl))
```

```{r}
arima(train.sqrt,order = c(5,0,0),seasonal = list(order = c(0,1,1), period = 12),
xreg = (1:xl))
```


```{r}
arima(train.sqrt,order = c(5,0,0),seasonal = list(order = c(0,1,1), period = 12),
xreg = (1:xl), fixed = c(0,0,0,0,NA,NA,NA))
```
~~~~

Source for plotting roots

~~~~
```{r}
plot.roots <- function(ar.roots=NULL, ma.roots=NULL, size=2, angles=FALSE,
special=NULL, sqecial=NULL,my.pch=1,first.col="blue",second.col="red",main=NULL)
{xylims <- c(-size,size)
      omegas <- seq(0,2*pi,pi/500)
      temp <- exp(complex(real=rep(0,length(omegas)),imag=omegas))

plot(Re(temp),Im(temp),typ="l",xlab="x",ylab="y",xlim=xylims,ylim=xylims,main=main)

      abline(v=0,lty="dotted")
      abline(h=0,lty="dotted")
~~~~

~~~
      if(!is.null(ar.roots))
        {
          points(Re(1/ar.roots),Im(1/ar.roots),col=first.col,pch=my.pch)
          points(Re(ar.roots),Im(ar.roots),col=second.col,pch=my.pch)
        }
      if(!is.null(ma.roots))
        {
          points(Re(1/ma.roots),Im(1/ma.roots),pch="*",cex=1.5,col=first.col)
          points(Re(ma.roots),Im(ma.roots),pch="*",cex=1.5,col=second.col)
        }
      if(angles)
        {
          if(!is.null(ar.roots))
            {
              abline(a=0,b=Im(ar.roots[1])/Re(ar.roots[1]),lty="dotted")
              abline(a=0,b=Im(ar.roots[2])/Re(ar.roots[2]),lty="dotted")
            }
          if(!is.null(ma.roots))
            {
              sapply(1:length(ma.roots), function(j)
abline(a=0,b=Im(ma.roots[j])/Re(ma.roots[j]),lty="dotted"))
            }
        }
      if(!is.null(special))
        {
          lines(Re(special),Im(special),lwd=2)
        }
      if(!is.null(sqecial))
        {
          lines(Re(sqecial),Im(sqecial),lwd=2)
        }
        }
```


Plot roots of model 1
```{r}
SARIMA.1.0.1.0.1.1 <- arima(train.sqrt,order = c(1,0,1),seasonal = list(order =
c(0,1,1), period = 12), xreg = (1:xl))
SARIMA.1.0.1.0.1.1

# sar part not existing (P = 0)

# ar part
ar_roots =polyroot(c(1,-0.9065))
plot.roots(ar.roots = polyroot(c(1,-0.9065)), NULL, main = "roots of ar part")
Mod(ar_roots)

# ma part
ma_roots = polyroot(c(1,-0.8155))
plot.roots(ma.roots = polyroot(c(1,-0.8155)), NULL, main = "roots of ma part")
Mod(ma_roots)

# sma part
sma_roots = polyroot(c(1,0,0,0,0,0,0,0,0,0,0,0,-0.9527))
Mod(sma_roots)
```
~~~

~~~
Plot roots of model 2
```{r}
SARIMA.2.0.2.0.1.1 <- arima(train.sqrt,order = c(2,0,2),seasonal = list(order =
c(0,1,1), period = 12), xreg = (1:xl))
SARIMA.2.0.2.0.1.1

# sar part not existing (P = 0)

# ar part
ar_roots =polyroot(c(1, -1.3636, 0.5909))
plot.roots(ar.roots = polyroot(c(1, -1.3636, 0.5909)), NULL, main = "roots of ar
part")
Mod(ar_roots)

# ma part
ma_roots = polyroot(c(1, -1.3647, 0.6943))
plot.roots(ma.roots = polyroot(c(1, -1.3647, 0.6943)), NULL, main = "roots of ma
part")
Mod(ma_roots)

# sma part
sma_roots = polyroot(c(1,0,0,0,0,0,0,0,0,0,0,0, -0.9308))
Mod(sma_roots)
```

Diagnostic Checking of model 1
```{r}
fit = SARIMA.1.0.1.0.1.1
df = 3 # 1+1+1 = 3
h = 15

# plot residuals
ts.plot(residuals(fit), main = "residuals of SARIMA(1,0,1)*(0,1,1)")
abline(lm(residuals(fit)~as.numeric(1:length(residuals(fit)))))
abline(h = mean(residuals(fit)),col = "red")

# residual mean & variance
mean(residuals(fit))
var(residuals(fit))

# AR auto-fit - AR(0) expected
ar(residuals(fit), aic = TRUE, order.max = NULL, method = c("yule-walker"))

# test correlation among residuals & residual squares
Box.test(residuals(fit), lag = h, type=c("Box-Pierce"),fitdf = df)
Box.test(residuals(fit), lag = h, type=c("Ljung-Box"),fitdf = df)
Box.test(residuals(fit)**2, lag = h, type=c("Ljung-Box"),fitdf = 0)



#install.packages("TSA")
#require(TSA)
#plot(residuals(fit),type='o',ylab = expression(y[t]), main= "period check for
residuals")
#periodogram(residuals(fit))
#abline(h=0)
~~~

~~~~
# fisher test - periodicity
#install.packages("GeneCycle")
library("GeneCycle")
fisher.g.test(residuals(fit))

# Kolmogorov-Smirnov test
cpgram(residuals(fit), main = "Kolmogorov-Smirnov Test")
# spectral analysis

# normality test
shapiro.test(residuals(fit))
# residual Acf/pacf + Histogram + QQ-plot
par(mfrow=c(1,2),oma=c(0,0,2,0))
op <- par(mfrow=c(2,2))
acf(residuals(fit),main = "Autocorrelation")
pacf(residuals(fit),main = "Partial Autocorrelation")
hist(residuals(fit),main = "Histogram")
qqnorm(residuals(fit))
qqline(residuals(fit),col ="blue")
title("Fitted Residuals Diagnostics", outer=TRUE)
par(op)
```

Diagnostic Checking of model 2
```{r}
fit = SARIMA.2.0.2.0.1.1
df = 5 # p+q+P+Q = 2+2+0+1 = 5
h = 15

# plot residuals
ts.plot(residuals(fit), main = "residuals of SARIMA(2,0,2)*(0,1,1)")
abline(lm(residuals(fit)~as.numeric(1:length(residuals(fit)))))
abline(h = mean(residuals(fit)),col = "red")
mean(residuals(fit))
var(residuals(fit))

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

# chi-square tests
Box.test(residuals(fit), lag = h, type=c("Box-Pierce"),fitdf = df)
Box.test(residuals(fit), lag = h, type=c("Ljung-Box"),fitdf = df)
Box.test(residuals(fit)**2, lag = h, type=c("Ljung-Box"),fitdf = 0)
# normality test
shapiro.test(residuals(fit))

# spectral analysis
# periodicity detect
fisher.g.test(residuals(fit))

# Kolmogorov-Smirnov test - Normal Gaussian
cpgram(residuals(fit), main = "Kolmogorov-Smirnov Test")


# acf/pacf; histogram; qq-plot
par(mfrow=c(1,2),oma=c(0,0,2,0))
~~~~

~~~~
op <- par(mfrow=c(2,2))
acf(residuals(fit),main = "Autocorrelation")
pacf(residuals(fit),main = "Partial Autocorrelation")
hist(residuals(fit),main = "Histogram")
qqnorm(residuals(fit))
qqline(residuals(fit),col ="blue")
title("Fitted Residuals Diagnostics", outer=TRUE)
par(op)
```


Test the final model on the last 24 observations
Make predictions on transformed data
```{r}
fit = SARIMA.2.0.2.0.1.1

# plot the transformed data
ts.plot(sqrt(data_ts), xlim = c(0,240), main = "predictions on transformed data")

# 24-step ahead predictions
xl = length(train.sqrt)
mypred <- predict(fit, 24, newxreg = (xl+1):(xl+24)) # including the intercept

# plot prediction points & C.I.
points(217:240, mypred$pred)
lines(217:240, mypred$pred + 1.96*mypred$se,lty=2)
lines(217:240, mypred$pred - 1.96*mypred$se,lty=2)
```

Make predictions on original data
```{r}
# plot original data
ts.plot(data_ts, xlim = c(0,240), main = "Predictions on Original Data")

# plot predictions points & C.I. Boundary
points(217:240, mypred$pred**2)
lines(217:240, (mypred$pred + 1.96*mypred$se)**2,lty=2)
lines(217:240, (mypred$pred - 1.96*mypred$se)**2,lty=2)

# prediction plot
ts.plot(data_ts, xlim = c(210,240), main = "Predictions on Original Data")
points(217:240, mypred$pred**2)
lines(217:240, (mypred$pred + 1.96*mypred$se)**2,lty=2)
lines(217:240, (mypred$pred - 1.96*mypred$se)**2,lty=2)
```

Make predictions for 1991 temperatures in Melbourne
```{r}
mypred <- predict(fit, 36, newxreg = (xl+1):(xl+36))
ts.plot(data_ts, xlim = c(0,252), main = "predictions of 1991 temperatures")
points(217:252,(mypred$pred)**2)
abline(v = 240, col = "red")
lines(217:252, (mypred$pred + 1.96*mypred$se)**2,lty=2)
lines(217:252, (mypred$pred - 1.96*mypred$se)**2,lty=2)

ts.plot(data_ts, xlim = c(241,252), main = "predictions on temperature in 1991")
points(217:252,(mypred$pred)**2, col = "red")
~~~~

~~~
lines(217:252, (mypred$pred + 1.96*mypred$se)**2,lty=2, col = "red")
lines(217:252, (mypred$pred - 1.96*mypred$se)**2,lty=2, col = "red")
```
~~~