Time Series Project: Prediction of Temperature in Houston

Yujie Li

1. Abstract

Weather is always related to our daily life, especially temperature, which tells us what to wear every day. Temperature of a specific place is a typical time series which may have seasonality and trend as we learned in PSTAT 174. In this project, I explored the dataset of average temperature of the city Houston in a time period of 20 years to do temperature prediction for the future. Since this is a time series with no obvious trend and difference in variance, I just difference it at lag 12 before modeling. Then I fitted several models based on the ACF and PACF of the data, and chose the one with the smallest AICc to go through the diagnostic tests. The model passed every test expect Mcleod-Li test, so the data has non-linear dependency. I use the model with the smallest AICc, SARIMA(1,0,1)(0,1,1) of seasonality of 12, to do the prediction and compare the prediction with the real-time test data I split at first. The result of the prediction is kind of good. It follows the real data well, and the real data falls in the confidence interval.

2. Introduction

Nils Bohi once said, "Prediction is very difficult, especially about the future." However, prediction of the future is actually meaningful in our real life. With accurate prediction, we can prepare for the future events. For example, whether prediction is one of the questions which is most relevant to our daily life. Specifically, temperature prediction can be the most useful prediction in the area of whether prediction.

In this project, I am going to use the "Daily Temperature of Major City" dataset from Kaggle to explore the time series data of the temperature in the city of Houston, from 1995 to 2020. The link of the dataset is https://www.kaggle.com/sudalairajkumar/daily-temperature-of-major-cities, and I will basically use R programming to do the project. After preprocessing this dataset, I will fit a SARIMA model to the dataset and do forecast for the future using this model to see the general changes of the monthly average temperature of the city of Houston.

In section 3, I will talk about the data preprocessing and visualization; in section 4, I will do transformation and differencing to make the data stational. Section 5 will be the part of model identification and section 6 will be the diagnostic testing. Then I will finally do forecasting in section 7 and conclusion in section 8.

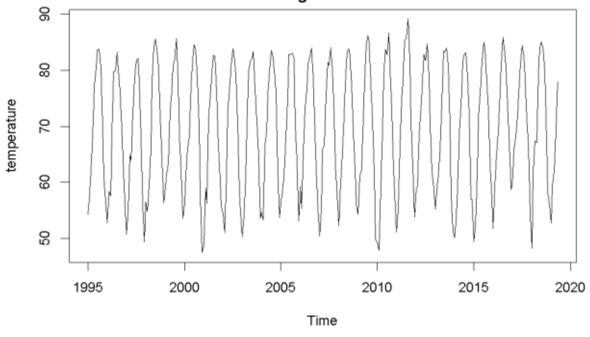
3. Data Preprocessing and Visualization

The original data from Kaggle is a really large dataset, which include daily average temperature since 1995 for a lot of major cities all over the world. So, the first step I did for the dataset is to select a city from the dataset. I chose, Houston as the city because the temperature values for Houston contains only a few missing data. The next step I did for the data preprocessing is to remove the missing data.

In this dataset, all the missing data are given the value of -99 instead of NA, which may cause a problem we I calculate the monthly average for the data. So, I first check that there are 26 missing values of -99, and then I changed all the -99 to NA. After that, I change the daily data to monthly data by taking the average temperature of each month ignoring the NA values, because this can decrease the period of my data to 12 from 365, which can make the later modeling process easier. The final step for my data preprocessing is to separate the data into train and test set, and I leave one year of data as the test set.

After the data preprocessing, I plot the time series of training data as Figure 3.1. Also, I plot the ACF and PACF of the training data as Figure 3.2 and Figure 3.3 as well. Based on the Figures, we can clearly see that the time series is a seasonal one with a period of every year. The variance of the series is relatively stable and there is no obvious trend in the time series. Then, I plot the decomposition of the time series.

Figure 3.1



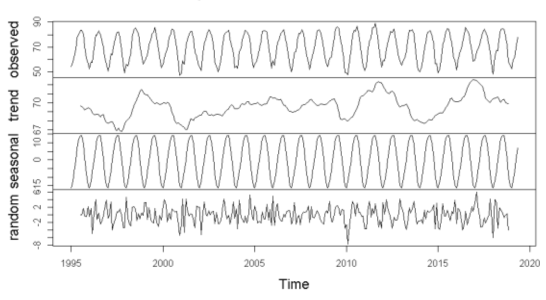


Figure 3.2

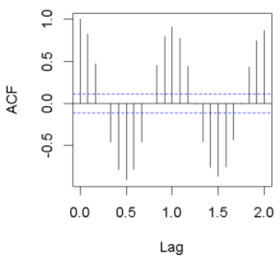
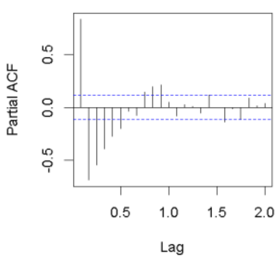


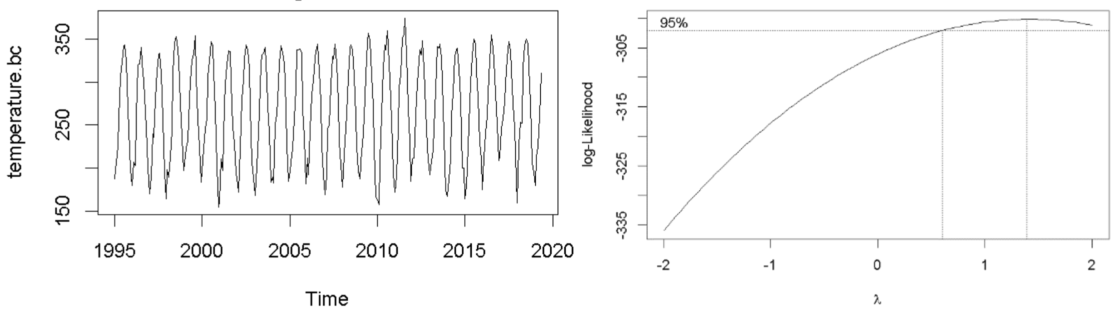
Figure 3.3



4. Transformation and Differencing

Based on the Figures shown in section 3, the time series is not likely to have a change in variance. However, I still tried a Boxcox transformation, and the data after transformation looks like Figure 4.1. The figure did not change much, so it confirms that we do not need a Boxcox transformation to stable variance here.

Figure 4.1



Then I tried to difference the data to make it stationary. I first difference the data at lag 12, since there is a clear yearly period for our monthly data. After differencing at lag 12, I got Figure 4.2. Based on this figure and the original Figure 3.1, we cannot easily discover an obvious trend. However, I still tried to difference it one more time at lag 1, and get Figure 4.3. Indeed, the data does not change much after differencing. Then I check the variance before and after the difference of lag 1. It increases from 12.9 to 18.9, which also indicates that the difference is not necessary. As a result, I only do a difference at lag 12 for my data.

Figure 4.2

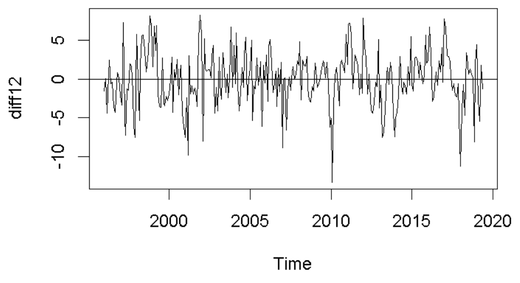
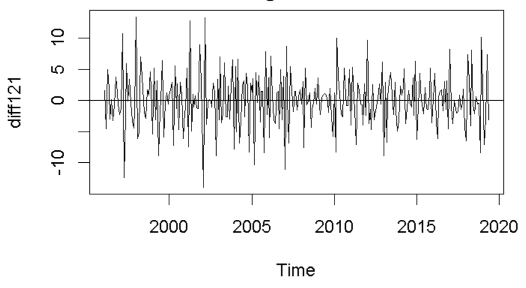


Figure 4.3



5. Model Identification

After transformation and differencing, I plotted the ACF and PACF of the data (Figure 5.1 and Figure 5.2). Based on the ACF and PACF of the data I then identified several possible pairs of coefficients for the SARIMA models.

Figure 5.1

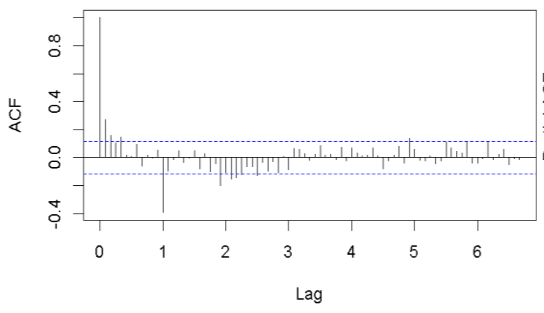
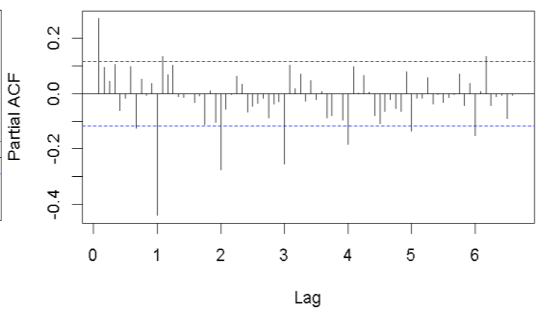


Figure 5.2



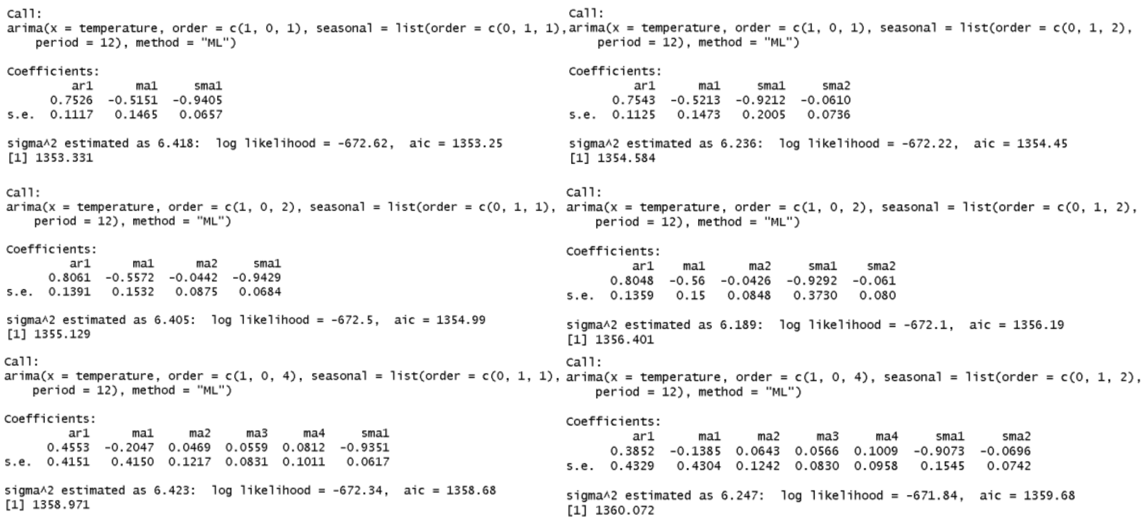
Since I only difference the data at lag 12 once, so s is 12, D is 1 and d is 0.

Based on the ACF graph (Figure 5.1), there is an obvious high value at lag 12, which is the period, and a smaller value out of the interval at lag 24, which is two period, so that the value of Q is most likely to be 1 but may be 2. For the first period, there are three values out of the interval at lag 1, 2, and 4, and the value at lag 2 and 4 are relatively small, so the value of q is most likely to be 1 but may also be 2, 4.

Based on the PACF graph (Figure 5.2), there are large values out of the interval at each of the period points, but the values tail off. Also, values at period points cut off at lag 1 or 2 for the ACF graph (Figure 5.1), so this is most likely to be an MA(q) model for the seasonal part, so P is 0. For the first period, all the values are within the interval except at lag 1, so p is 1.

Above all the model should be SARIMA(1,0,1)(0,1,1), SARIMA(1,0,2)(0,1,1), SARIMA(1,0,4)(0,1,1), or SARIMA(1,0,1)(0,1,2), SARIMA(1,0,2)(0,1,2), SARIMA(1,0,4)(0,1,2) of seasonality of 12.

Then I look at the output of the model fitting, and the fitting result as the graph below. Based on the results, there are no trivial solutions for all the six models, so they can be considered valid.



```
Call:
arima(x = temperature, order = c(1, 0, 1), seasonal = list(order = c(0, 1, 1),
    period = 12), method = "ML")

Coefficients:
         ar1      ma1     sma1
      0.7526  -0.5151  -0.9405
s.e.  0.1117   0.1465   0.0657

sigma^2 estimated as 6.418:  log likelihood = -672.62,  aic = 1353.25
[1] 1353.331

Call:
arima(x = temperature, order = c(1, 0, 2), seasonal = list(order = c(0, 1, 1),
    period = 12), method = "ML")

Coefficients:
         ar1      ma1      ma2     sma1
      0.8061  -0.5572  -0.0442  -0.9429
s.e.  0.1391   0.1532   0.0875   0.0684

sigma^2 estimated as 6.405:  log likelihood = -672.5,  aic = 1354.99
[1] 1355.129

Call:
arima(x = temperature, order = c(1, 0, 4), seasonal = list(order = c(0, 1, 1),
    period = 12), method = "ML")

Coefficients:
         ar1      ma1     ma2     ma3     ma4     sma1
      0.4553  -0.2047  0.0469  0.0559  0.0812  -0.9351
s.e.  0.4151   0.4150  0.1217  0.0831  0.1011   0.0617

sigma^2 estimated as 6.423:  log likelihood = -672.34,  aic = 1358.68
[1] 1358.971
```

```
Call:
arima(x = temperature, order = c(1, 0, 1), seasonal = list(order = c(0, 1, 2),
    period = 12), method = "ML")

Coefficients:
         ar1      ma1     sma1     sma2
      0.7543  -0.5213  -0.9212  -0.0610
s.e.  0.1125   0.1473   0.2005   0.0736

sigma^2 estimated as 6.236:  log likelihood = -672.22,  aic = 1354.45
[1] 1354.584

Call:
arima(x = temperature, order = c(1, 0, 2), seasonal = list(order = c(0, 1, 2),
    period = 12), method = "ML")

Coefficients:
         ar1    ma1      ma2     sma1    sma2
      0.8048  -0.56  -0.0426  -0.9292  -0.061
s.e.  0.1359   0.15   0.0848   0.3730   0.080

sigma^2 estimated as 6.189:  log likelihood = -672.1,  aic = 1356.19
[1] 1356.401

Call:
arima(x = temperature, order = c(1, 0, 4), seasonal = list(order = c(0, 1, 2),
    period = 12), method = "ML")

Coefficients:
         ar1      ma1     ma2     ma3     ma4     sma1     sma2
      0.3852  -0.1385  0.0643  0.0566  0.1009  -0.9073  -0.0696
s.e.  0.4329   0.4304  0.1242  0.0830  0.0958   0.1545   0.0742

sigma^2 estimated as 6.247:  log likelihood = -671.84,  aic = 1359.68
[1] 1360.072
```

Then I find out that some of the coefficients might be fixed to zero to make because zero is in the confidence interval. For example, in SARIMA(1,0,1)(0,1,2), the sma2 coefficient, and ma2 coefficients in SARIMA(1,0,2)(0,1,1) have zero in their confidence interval. So, I try to fix those coefficients to zero and tried to calculated AICc of them again, and I get the result as following.

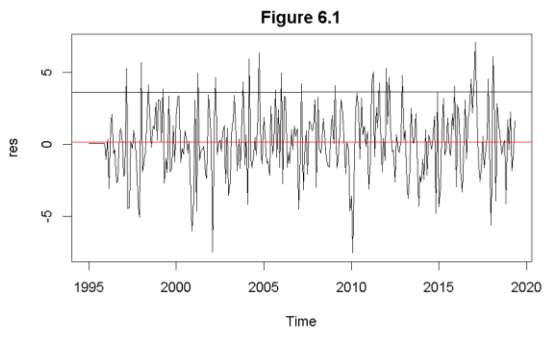
```
Call:
arima(x = temperature, order = c(1, 0, 2), seasonal = list(order = c(0, 1, 1),
    period = 12), fixed = c(NA, NA, 0, NA), method = "ML")

Coefficients:
         ar1      ma1  ma2     sma1
      0.7526  -0.5151    0  -0.9405
s.e.  0.1117   0.1465    0   0.0657

sigma^2 estimated as 6.418:  log likelihood = -672.62,  aic = 1353.25
[1] 1353.387

Call:
arima(x = temperature, order = c(1, 0, 2), seasonal = list(order = c(0, 1, 2),
    period = 12), fixed = c(NA, NA, 0, NA, 0), method = "ML")

Coefficients:
         ar1      ma1  ma2     sma1  sma2
      0.7526  -0.5151    0  -0.9405     0
s.e.  0.1117   0.1465    0   0.0657     0

sigma^2 estimated as 6.418:  log likelihood = -672.62,  aic = 1353.25
[1] 1353.457

Call:
arima(x = temperature, order = c(1, 0, 4), seasonal = list(order = c(0, 1, 2),
    period = 12), transform.pars = FALSE, fixed = c(0, 0, 0, 0, 0, NA, 0),
method = "ML")

Coefficients:
      ar1  ma1  ma2  ma3  ma4     sma1  sma2
        0    0    0    0    0  -1.0857     0
s.e.    0    0    0    0    0   0.0587     0

sigma^2 estimated as 6.218:  log likelihood = -689.79,  aic = 1383.58
[1] 1383.975
```

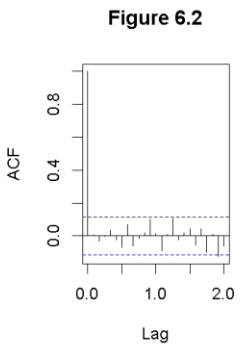
```
Call:
arima(x = temperature, order = c(1, 0, 1), seasonal = list(order = c(0, 1, 2),
    period = 12), fixed = c(NA, NA, NA, 0), method = "ML")

Coefficients:
         ar1      ma1     sma1  sma2
      0.7526  -0.5151  -0.9405     0
s.e.  0.1117   0.1465   0.0657     0

sigma^2 estimated as 6.418:  log likelihood = -672.62,  aic = 1353.25
[1] 1353.387

Call:
arima(x = temperature, order = c(1, 0, 4), seasonal = list(order = c(0, 1, 1),
    period = 12), transform.pars = FALSE, fixed = c(0, 0, 0, 0, 0, NA), method =
"ML")

Coefficients:
      ar1  ma1  ma2  ma3  ma4     sma1
        0    0    0    0    0  -1.0857
s.e.    0    0    0    0    0   0.0587

sigma^2 estimated as 6.218:  log likelihood = -689.79,  aic = 1383.58
[1] 1383.876
```

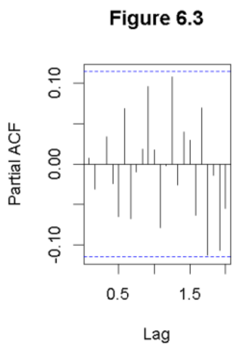
After fixing those values zero, there are only two models left, which are SARIMA(1,0,1)(0,1,1) and SARIMA(1,0,4)(0,1,1) of fixing five zeros. As a result, I choose the model with the lowest AICc: SARIMA(1,0,1)(0,1,1) of seasonality 12 to do the diagnostic testing.

6. Diagnostic Testing

After choosing the models with the lowest AICc, I did diagnostic testing on it to test whether this model is valid. I first calculate the residual and plot it as Figure 6.1. Both of the plots have mean of zero and fixed variance, so they look like white noise.







Then I plot the ACF and PACF of the residuals as Figure 6.2 and 6.3. All the values of ACF and PACF are inside the confidence interval, so it can be considered as the pattern of white noise.

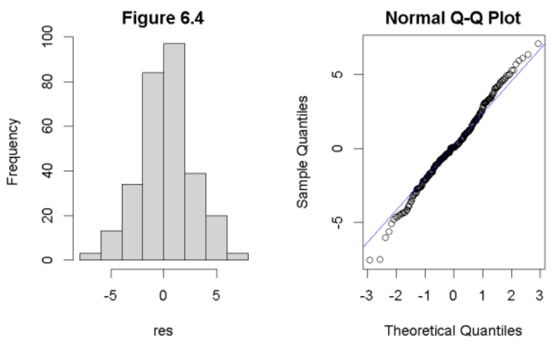
Then I ran the Box-Pierce, Ljung-Box, McLeod-Li, and Shapiro-Wilks test for the model, the results are shown below.

```
        Box-Pierce test

data:  res
X-squared = 14.027, df = 13, p-value = 0.3719


        Box-Ljung test

data:  res
X-squared = 14.669, df = 13, p-value = 0.3285


        Box-Ljung test

data:  res^2
X-squared = 54.233, df = 17, p-value = 9.092e-06


        Shapiro-Wilk normality test

data:  res
W = 0.99349, p-value = 0.2365
```

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


Order selected 0  sigma^2 estimated as  6.146
```

Based on the results, we can see that model passed all the test except McLeod-Li test. Only the McLeod-Li test result is significant, which might indict non-linear dependency. So I then use the ar() function in R, and the result is zero, which means that there is non-linear dependency in the case.

Finally, I plot the histogram and Q-Q Plot for the residuals as Figure 6.4. The histograms look like normal distribution and the Q-Q Plots follows the straight line.



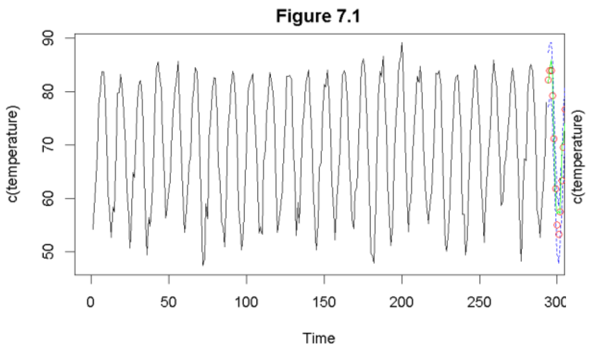
Since the first model does not pass McLeod-Li test, I tried the other model in section 5, which is SARIMA(1,0,4)(0,1,1) of fixing five zeros. The tests results of the second model is as following.

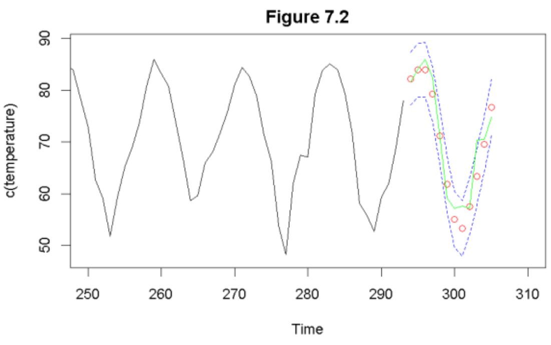
```
        Box-Pierce test

data:  res
X-squared = 61.248, df = 13, p-value = 3.139e-08


        Box-Ljung test

data:  res
X-squared = 62.428, df = 13, p-value = 1.925e-08


        Box-Ljung test

data:  res^2
X-squared = 62.24, df = 17, p-value = 4.465e-07


        Shapiro-Wilk normality test

data:  res
W = 0.98849, p-value = 0.02002
```

Based on the results, all the second model does not pass all the tests, so I will still use the first model for further forecasting.

7. Forecasting

After the diagnostic testing, I chose the model of SARIMA(1,0,1)(0,1,1) of seasonality of 12, for further steps. I tried to fit the model and use it to predict the data of year 2020, and plot the predicted data as Figure 7.1 and Figure 7.2. The black line is the training data, the green line is the test data, the blue dash line is the predicted confident interval, and the red points are the predicted 12 datapoints for 2020.





As we can see in the figures, the predicted values generally fit the real test data well as the green line. Also, all the green line falls in the confidence interval of the blue dash lines. This indicates that the model works for the problem well.

Finally, I identified the model coefficients for the model, so the model should be $(1-0.7526B)(1-B^{12})X_t = (1-0.5151\text{B})(1-0.9405B^{12})Z_t$. The non-seasonal part is AR(1) and MA(1), and the absolute values of the coefficients are less than one, so it should be stationary and invertible.

```
Coefficients:
         ar1      ma1     sma1
      0.7526  -0.5151  -0.9405
s.e.  0.1117   0.1465   0.0657
```

8. Conclusion

Above all, I fit a SARIMA(1,0,1)(0,1,1) of seasonality of 12 to the data, and predicted one year temperature later. The exact model is $(1-0.7526B)(1-B^{12})X_t = (1-0.5151\text{B})(1-0.9405B^{12})Z_t$.The prediction fits the real data well, which means that the model is suitable for the data, and it can be used to predict the monthly average of temperature of Houston city.

9. Appendix (R Code)

```
# load packages
        library(lubridate)
        library(MASS)
        library(dplyr)
        library(forecast)
        library(qpcR)
# Load the data
        data = read.csv("city_temperature.csv")
```

```
# Select city Houston, Date, Temp
        d1 = data[data$City=="Houston",]
        data1 = data.frame(Month=d1$Month, Year=d1$Year, Temp =
    d1$AvgTemperature)
# Check missing values
        sum(data1$Temp == -99)
# replace them with NA
        for (i in (1:length(data1$Temp))){
          if ( data1$Temp[i]== -99){
            data1$Temp[i]= NA
          }
          if (nchar(as.character(data1$Month[i]))==1){
            data1$Month[i]=paste0("0",as.character(data1$Month[i]))
          }
        }
        sum(data1$Temp == -99)
        sum(is.na(data1))
# Change to monthly average ignoring the NAs
        data_houston=data1 %>%
          mutate(Date = as.Date(paste0(as.character(data1$Year),
                                       as.character(data1$Month),"01"),
    "%Y%m%d"))%>%
          group_by(Date)%>%
         summarise(Avg=mean(Temp,na.rm =TRUE))
# train/test seperation
        train=data_houston[data_houston$Date<"2019-06-01",]
       test=data_houston[data_houston$Date>="2019-06-01",][1:12,]
# Plot the original data
        temperature= ts(train$Avg, start = c(1995,1), end = c(2019,5),frequency
    = 12)
        ts.plot(temperature, main = "Figure 3.1")
        plot(decompose(temperature), sub = "Figure 3.1")
# Plot the original ACF PACF
        par(mfrow=c(1,2))
        acf(temperature,main = "Figure 3.2")
        pacf(temperature,main = "Figure 3.3")
       par(mfrow=c(1,1))
# Boxcox transformation
        t = 1:length(temperature)
        fit = lm(temperature ~ t)
        bcTransform = boxcox(temperature ~ t,plotit = TRUE)
        lambda = bcTransform$x[which(bcTransform$y ==
    max(bcTransform$y))]
        temperature.bc = (1/lambda)*(temperature^lambda-1)
        ts.plot(temperature.bc, main = "Figure 4.1")
# Difference at 12
        diff12=diff(temperature,12)
        ts.plot(diff12, main = "Figure 4.2")
        abline(h=0)
# Difference at 1
        diff121=diff(diff12,1)
        ts.plot(diff121, main = "Figure 4.3")
        abline(h=0)
        var(diff12)
        var(diff121)
# Plot new ACF PACF
        acf(diff12,lag.max = 80, main = "Figure 5.1")
        pacf(diff12,lag.max = 80, main = "Figure 5.2")
# modeling
        arima(temperature, order=c(1,0,1), seasonal = list(order = c(0,1,1),
    period = 12), method="ML")
        AICc(arima(temperature, order=c(1,0,1), seasonal = list(order = c(0,1,1),
    period = 12), method="ML"))
        arima(temperature, order=c(1,0,1), seasonal = list(order = c(0,1,2),
    period = 12), method="ML")
        AICc(arima(temperature, order=c(1,0,1), seasonal = list(order = c(0,1,2),
    period = 12), method="ML"))
        arima(temperature, order=c(1,0,2), seasonal = list(order = c(0,1,1),
    period = 12), method="ML")
        AICc(arima(temperature, order=c(1,0,2), seasonal = list(order = c(0,1,1),
    period = 12), method="ML"))
        arima(temperature, order=c(1,0,2), seasonal = list(order = c(0,1,2),
    period = 12), method="ML")
        AICc(arima(temperature, order=c(1,0,2), seasonal = list(order = c(0,1,2),
    period = 12), method="ML"))
        arima(temperature, order=c(1,0,4), seasonal = list(order = c(0,1,1),
    period = 12), method="ML")
        AICc(arima(temperature, order=c(1,0,4), seasonal = list(order = c(0,1,1),
    period = 12), method="ML"))
        arima(temperature, order=c(1,0,4), seasonal = list(order = c(0,1,2),
    period = 12), method="ML")
        AICc(arima(temperature, order=c(1,0,4), seasonal = list(order = c(0,1,2),
    period = 12), method="ML"))
# fixing some of the values 0
        arima(temperature, order=c(1,0,1), seasonal = list(order = c(0,1,2),
    period = 12), fixed = c(NA,NA,NA,0),method="ML")
        AICc(arima(temperature, order=c(1,0,1), seasonal = list(order = c(0,1,2),
    period = 12), fixed = c(NA,NA,NA,0), method="ML"))
```

```
        arima(temperature,  order=c(1,0,2),  seasonal  =  list(order  =  c(0,1,1),
    period = 12), fixed = c(NA,NA,0,NA), method="ML")
        AICc(arima(temperature, order=c(1,0,2), seasonal = list(order = c(0,1,1),
    period = 12), fixed = c(NA,NA,0,NA), method="ML"))
        arima(temperature,  order=c(1,0,2),  seasonal  =  list(order  =  c(0,1,2),
    period = 12), fixed = c(NA,NA,0,NA,0), method="ML")
        AICc(arima(temperature, order=c(1,0,2), seasonal = list(order = c(0,1,2),
    period = 12), fixed = c(NA,NA,0,NA,0), method="ML"))
        arima(temperature,  order=c(1,0,4),  seasonal  =  list(order  =  c(0,1,1),
    period = 12), fixed = c(0,0,0,0,0,NA), method="ML",transform.pars = FALSE)
        AICc(arima(temperature, order=c(1,0,4), seasonal = list(order = c(0,1,1),
    period  =  12),  fixed  =  c(0,0,0,0,0,NA),  method="ML",transform.pars  =
    FALSE))
        arima(temperature,  order=c(1,0,4),  seasonal  =  list(order  =  c(0,1,2),
    period  =  12),  fixed  =  c(0,0,0,0,0,NA,0),  method="ML",transform.pars  =
    FALSES)
       AICc(arima(temperature, order=c(1,0,4), seasonal = list(order = c(0,1,2),
    period  =  12),  fixed  =  c(0,0,0,0,0,NA,0),  method="ML",transform.pars  =
    FALSE))
# diagnostic testing
        fit1= arima(temperature, order=c(1,0,1), seasonal = list(order = c(0,1,1),
    period = 12), method="ML")
        res <- residuals(fit1)
        mean(res); var(res)
# Plot res
        par(mfrow=c(1, 1))
        ts.plot(res, main="Figure 6.1")
        t <- 1:length(res)
        fit.res = lm(res~ t)
        abline(fit.res)
        abline(h = mean(res), col = "red")
# plot acf pacf of res
        par(mfrow=c(1, 2))
        acf(res, main="Figure 6.2")
        pacf(res, main="Figure 6.3")
# independency    test
        l=round(sqrt(nrow(train)))
        Box.test(res, lag = l, type = c("Box-Pierce"), fitdf = 4)
        Box.test(res, lag = l, type = c("Ljung-Box"), fitdf = 4)
        Box.test(res^2, lag = l, type = c("Ljung-Box"), fitdf = 0)
# normal test
        shapiro.test(res)
        par(mfrow=c(1,2))
        hist(res,main = "Figure 6.4")
```

```
            qqnorm(res)
            qqline(res,col ="blue",main = "Figure 6.5")
           ar(res, aic =TRUE, order.max = NULL, method = "yule-walker")
            fit1= arima(temperature, order=c(1,0,1), seasonal = list(order = c(0,1,1),
      period = 12), method="ML")
# diagnostic testing2
            fit1= arima(temperature, order=c(1,0,4), seasonal = list(order = c(0,1,1),
      period = 12), fixed = c(0,0,0,0,0,NA), method="ML",transform.pars = FALSE)
            res <- residuals(fit1)
            mean(res); var(res)
# Plot res
            par(mfrow=c(1, 1))
            ts.plot(res, main="Figure 6.1")
            t <- 1:length(res)
            fit.res = lm(res~ t)
            abline(fit.res)
            abline(h = mean(res), col = "red")
# plot acf pacf of res
            par(mfrow=c(1, 2))
            acf(res, main="Figure 6.2")
            pacf(res, main="Figure 6.3")
# independency   test
            l=round(sqrt(nrow(train)))
            Box.test(res, lag = l, type = c("Box-Pierce"), fitdf = 4)
            Box.test(res, lag = l, type = c("Ljung-Box"), fitdf = 4)
            Box.test(res^2, lag = l, type = c("Ljung-Box"), fitdf = 0)
# normal test
            shapiro.test(res)
            par(mfrow=c(1,2))
            hist(res,main = "Figure 6.4")
            qqnorm(res)
            qqline(res,col ="blue",main = "Figure 6.5")
            ar(res, aic =TRUE, order.max = NULL, method = "yule-walker")
# plot data
            pred.tr <- predict(fit1, n.ahead = 12)
            U.tr= pred.tr$pred + 2*pred.tr$se
            L.tr= pred.tr$pred - 2*pred.tr$se
            ts.plot(c(temperature),main ="Figure 7.1")
            lines((length(temperature)+1):(length(temperature)+12),c(U.tr),
      col="blue", lty="dashed")
            lines((length(temperature)+1):(length(temperature)+12),c(L.tr),
      col="blue", lty="dashed")
            points((length(temperature)+1):(length(temperature)+12),  pred.tr$pred,
      col="red")
```

```
            lines((length(temperature)+1):(length(temperature)+12),c(test$Avg),
        col="green")
# plot zoom-in data
            ts.plot(c(temperature),main ="Figure 7.2" , xlim = c(250,310))
            lines((length(temperature)+1):(length(temperature)+12),c(U.tr),
        col="blue", lty="dashed")
            lines((length(temperature)+1):(length(temperature)+12),c(L.tr),
        col="blue", lty="dashed")
            points((length(temperature)+1):(length(temperature)+12),  pred.tr$pred,
        col="red")
            lines((length(temperature)+1):(length(temperature)+12),c(test$Avg),
        col="green")
           fit1
```