# Timeseries

*Tiancheng hou*

*February 10, 2019*

## Time series analysis

Abstract: In this note, I mainly used the ARIMA model and EMD decomposition to predict the stock price of American Electric Power. And I used crawler technology to download the stock price that I want to analyze from the web.

```
library(dplyr)
library(forecast)
library(xts)
library(EMD)
library(ggplot2)
library(jsonlite)
library(tseries)
setwd("C:/Users/Arche/graduate/study/github/Time-series")
```

First load the required packages and set the working path

```
Get_price <- function(symbol="MSFT",f="TIME_SERIES_DAILY"){
url <- paste("https://www.alphavantage.co/query?function=",f,"&symbol=",symbol,"&outputsize=full&apikey=
data <- fromJSON(url)
if(f=="TIME_SERIES_DAILY"){
  temp_data <- data$`Time Series (Daily)`
} else if(f=="TIME_SERIES_WEEKLY"){
  temp_data <- data$`Weekly Time Series`
} else if(f=="TIME_SERIES_MONTHLY"){
  temp_data <- data$`Monthly Time Series`
}
day <- names(temp_data)
close_price <- array()
for (i in 1:length(day)) {
  d <- day[i]
    close_price[i] <- as.numeric(temp_data[[d]]$'4. close')
}
as.numeric()
    mydata <- data.frame(symbol,day,close_price)
    return(mydata)
}
```

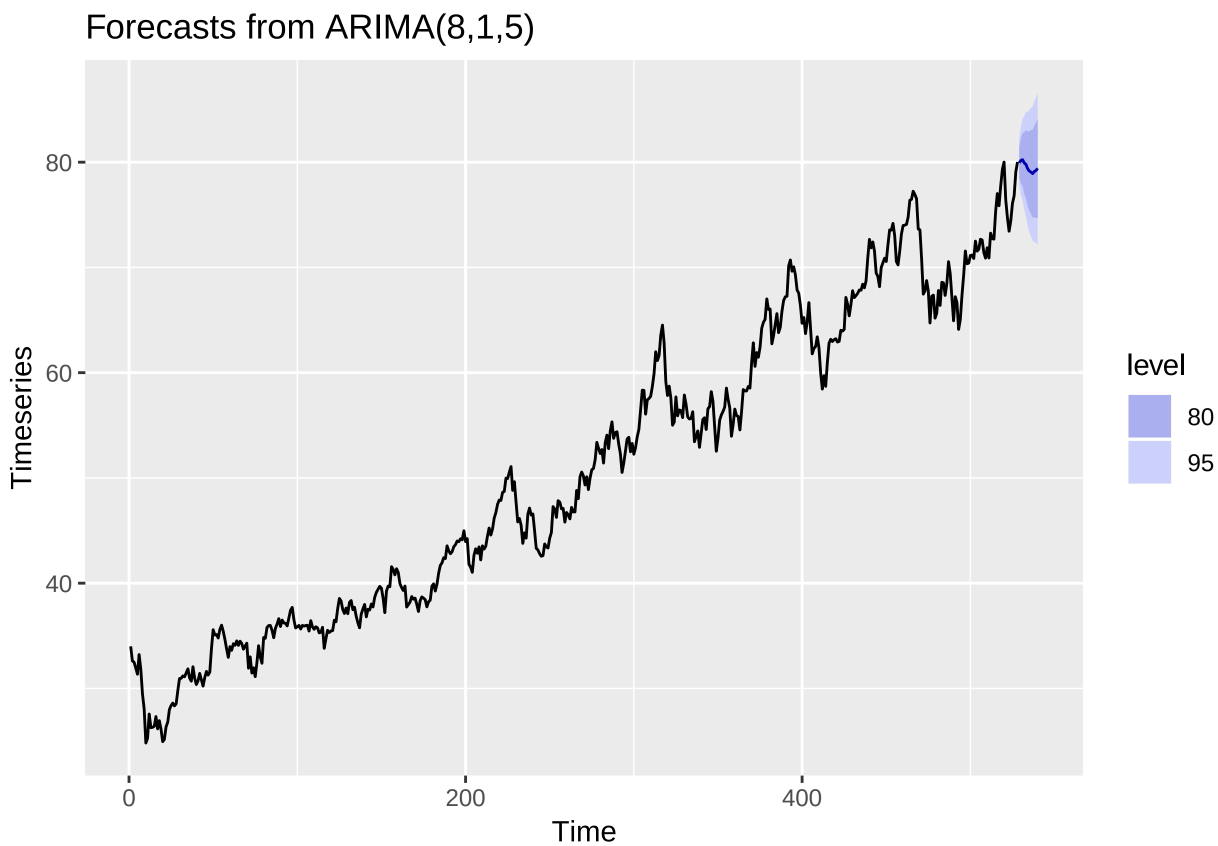
The above function will be used to download stock price data from the network. Please note that I have hidden my APIkey here and the user needs to set their own APIkey to access the data. It is actually very easy to apply for an API key. You only need to login on "https://www.alphavantage.co" to apply with a e-mail. In this function, there are two parameters. The first one is "symbol", which is the short name of the company you need to query. The default value for "symbol" is Microsoft Corporation. The second parameter is "f". By changing the value of f we can choose to query daily data, weekly data or monthly data.

```
AEP <- Get_price("AEP","TIME_SERIES_WEEKLY")[1:528,]
AEP$day <- as.Date(AEP$day,format ="%Y-%m-%d")
Timeseries <- xts(x = AEP$close_price, order.by = AEP$day)
head(Timeseries)
```

```
##              [,1]
## 2009-01-02 34.00
## 2009-01-09 32.61
## 2009-01-16 32.50
## 2009-01-23 31.94
## 2009-01-30 31.35
## 2009-02-06 33.22
```

Call the function I just wrote to download the data. After getting our data, we want to convert it to xts format, because it will be convenient for future analysis. In addition, I only want to analyze the data of the last ten years, so I removed the earlier data.

```
mod_arima <- arima(Timeseries,order = c(8,1,5))
perdict_arima <- forecast(mod_arima,h=12)
autoplot(perdict_arima)
```




We first use the ARIMA model to fit the model. And use the obtained model to make predictions. Because our sequence frequency is relatively high, we also choose larger AR and MA parameters when building the model. At the same time, we noticed that the data has some upward trend, so we choose to make a first-order difference on the data to make the data stationary. Drawing the predicted results, we see that the big ARIMA model predicts that AEP's share price will decline slightly in the next 12 weeks.

```
emd_predict <- function(X=AEP){
  a <- emd(X$close_price)
  emd_price <- as.data.frame(a$imf)
  emd_price$residue <- a$residue
  n_imfs <- length(names(emd_price))
  emd_price$day <- as.Date(X$day,format ="%Y-%m-%d")
```

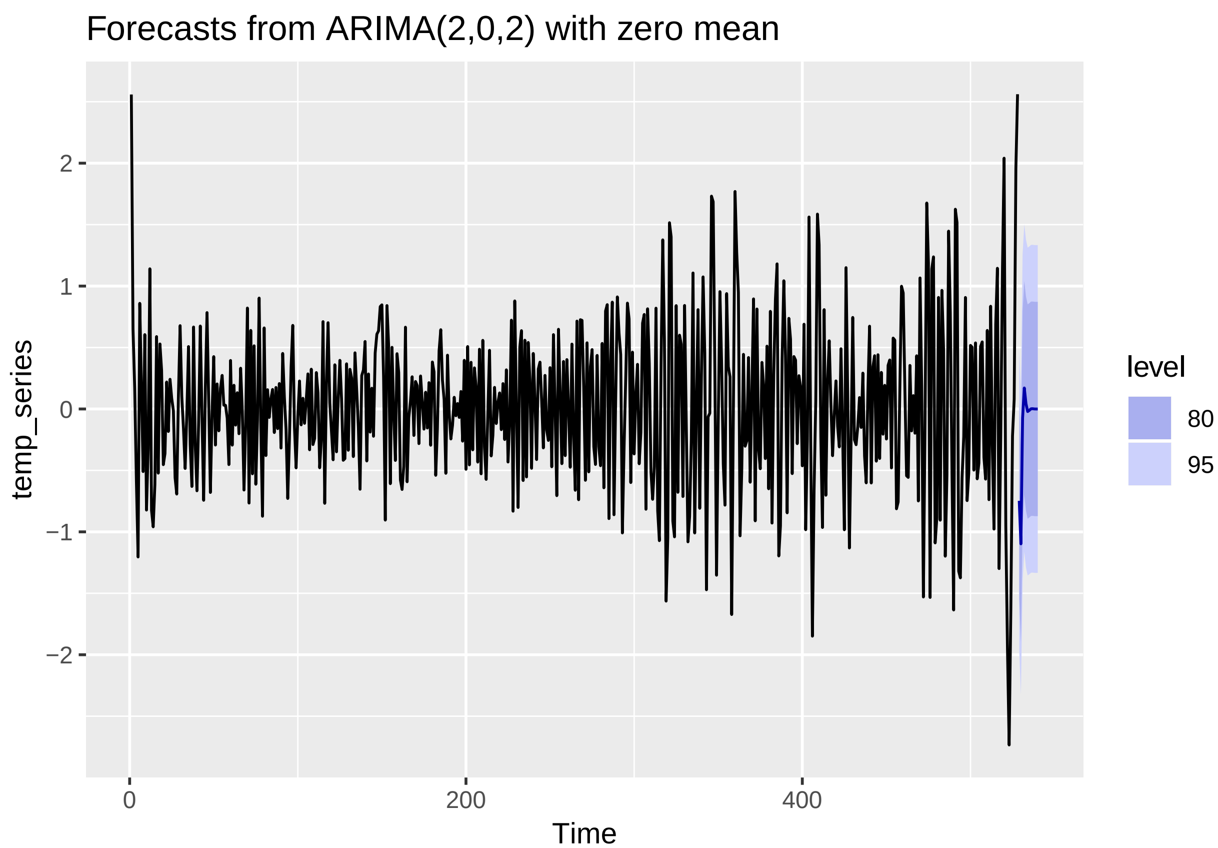
```r
  p <- list()
  fft_plot <- list()
  predict_price <- rep(0,12)
  period <- array()
  for (i in 1:n_imfs) {
      temp_series <- xts(x = emd_price[i], order.by = X$day)
      mod_emd_arima <- auto.arima(temp_series)
      predict_emd_arima <- forecast(mod_emd_arima,h=12)
      p[[i]] <- autoplot(predict_emd_arima)
      predict_price <- predict_price+predict_emd_arima$mean

      ######
      temp_series <- as.numeric(unlist(emd_price[i]))
      N=length(temp_series)
      dt=1
      f  = (0:(N-1))/N * (1/dt);
      freq<-fft(temp_series,inverse = T)
      period[[i]] <- N/(which.max(abs(freq)[1:N/2])-1)
      temp_data <- data.frame(x=f[1:N/2],y=abs(freq)[1:N/2])
      fft_plot[[i]] <- ggplot(data = temp_data,aes(x=x,y=y))+
        geom_line()
      ######
  }
  return(list(p,predict_price,period,fft_plot))
}
```

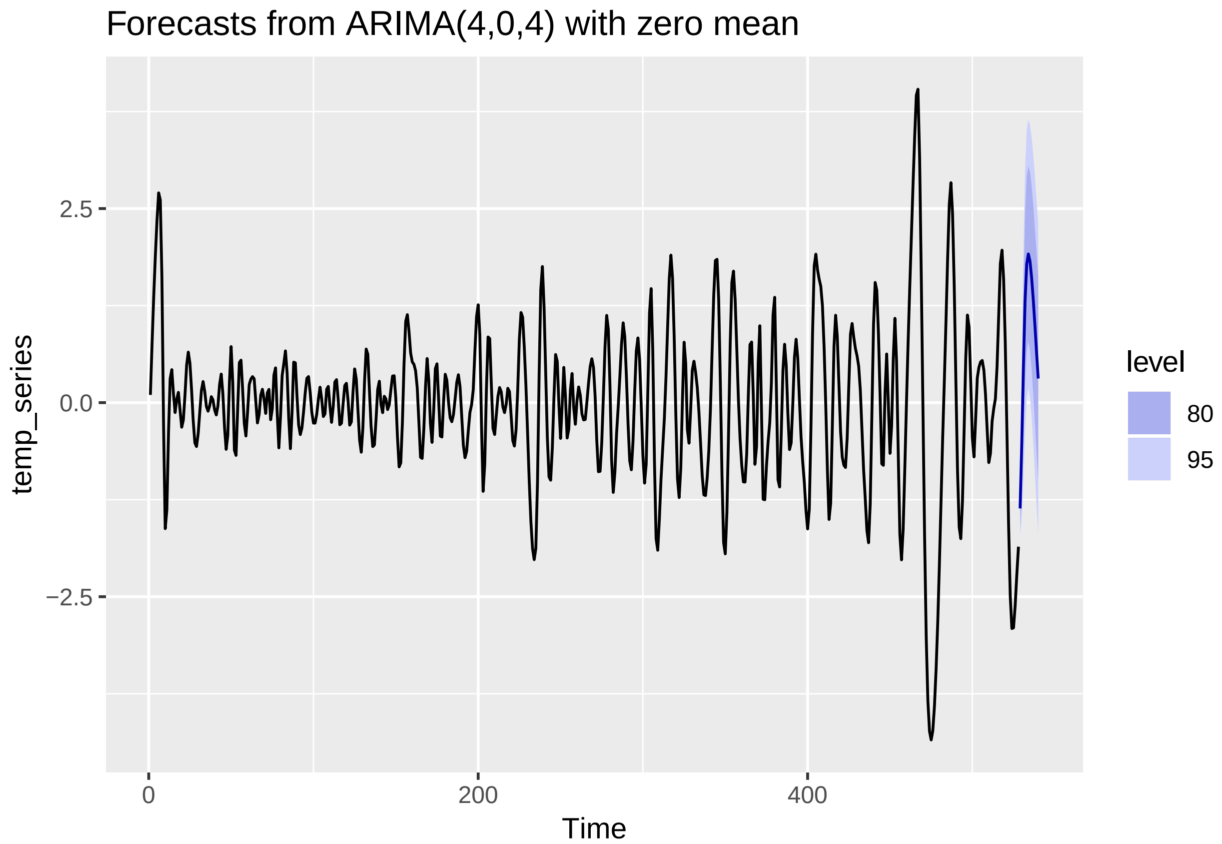
Next, I wrote a function that combines EMD decomposition with ARIMA. After decomposing the original time series using EMD, I performed ARIMA modeling on each component separately. I use the EMD-ARIMA model to predict AEP's stock price for the next 12 weeks.

```r
result <- emd_predict(AEP)
result[[1]]
```

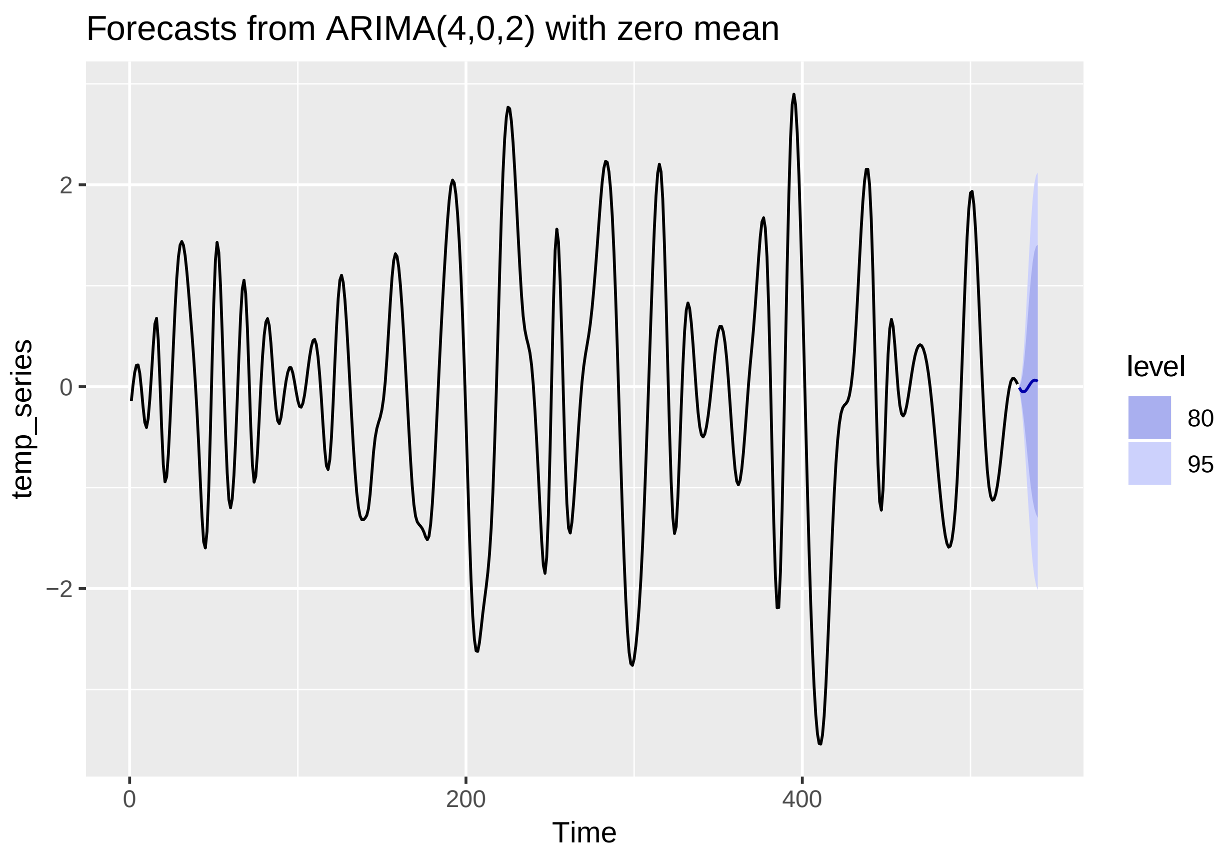
```
## [[1]]
```


Forecasts from ARIMA(2,0,2) with zero mean
temp_series
Time
level
80
95
2
1
0
−1
−2
0
200
400

```
##
## [[2]]
```

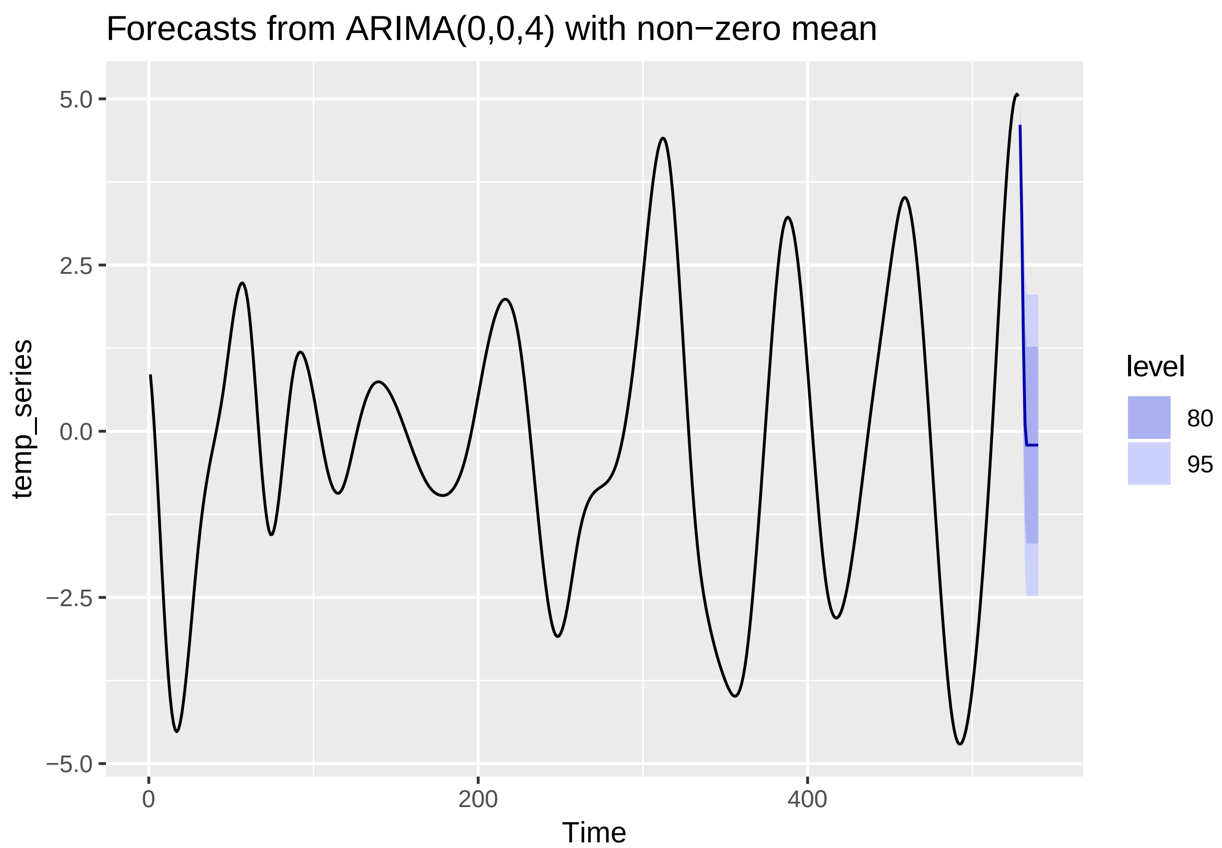

Forecasts from ARIMA(4,0,4) with zero mean
temp_series
2.5
0.0
−2.5
0
200
400
Time
level
80
95

```
##
## [[3]]
```

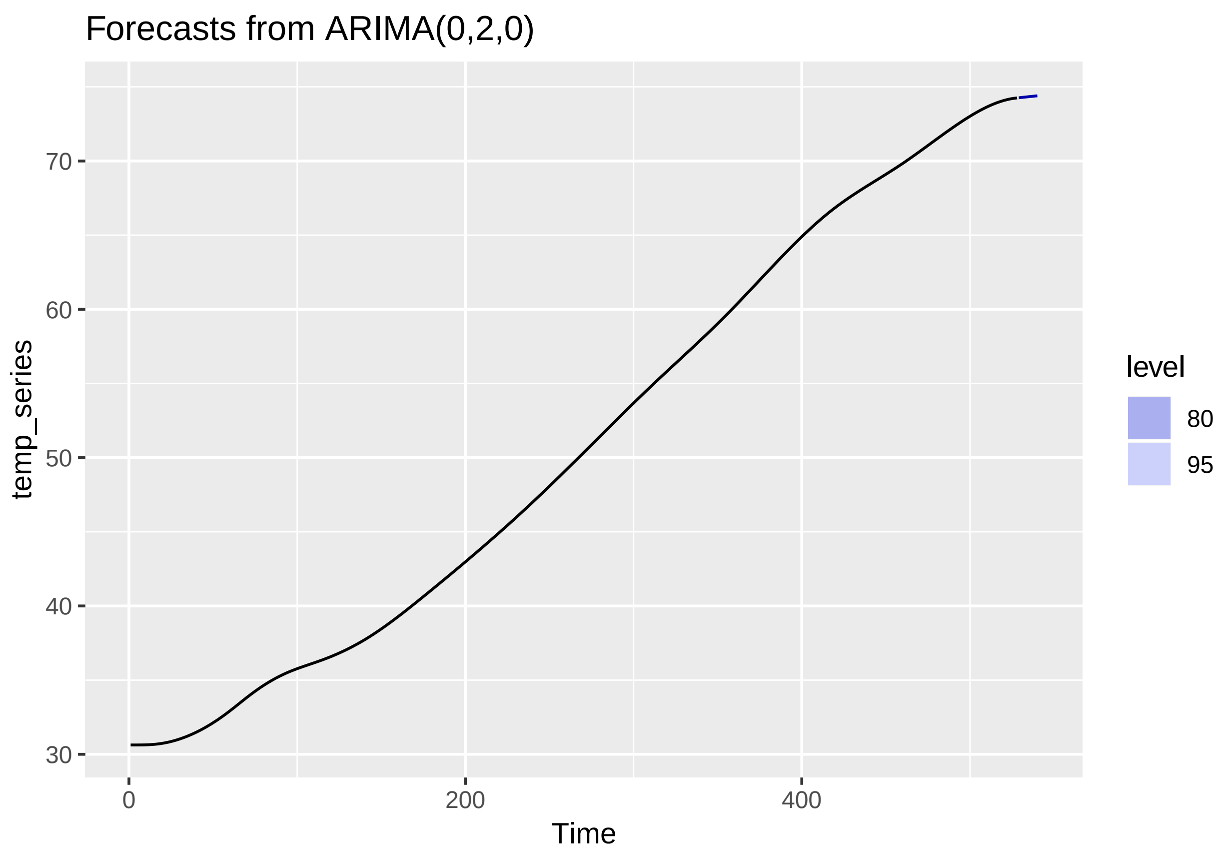


Forecasts from ARIMA(4,0,2) with zero mean
2
0
−2
temp_series
0
200
400
Time
level
80
95


```
##
## [[4]]
```


Forecasts from ARIMA(0,0,4) with non−zero mean
5.0
2.5
0.0
−2.5
−5.0
temp_series
0
200
400
Time
level
80
95

```
##
## [[5]]
```




We can see that the ARIMA model fits better after being decomposed by EMD.

```
predict_price <- result[[2]]
predict_price_xts <- xts(predict_price,
                         order.by =as.Date(c("2019-2-15","2019-2-22","2019-3-1","2019-3-8",
                                             "2019-3-15","2019-3-22","2019-3-29","2019-4-5",
                                            "2019-4-12","2019-4-19","2019-4-26","2019-5-3"),
                                           format ="%Y-%m-%d"))

df_history <- fortify(Timeseries)
df_predict <- fortify(predict_price_xts)
ggplot()+
  geom_line(aes(x=Index,y=Timeseries),data=df_history, col="gray2")+
  geom_line(aes(x=Index,y=predict_price_xts),data=df_predict,col="red")
```


```r
df_predict
```

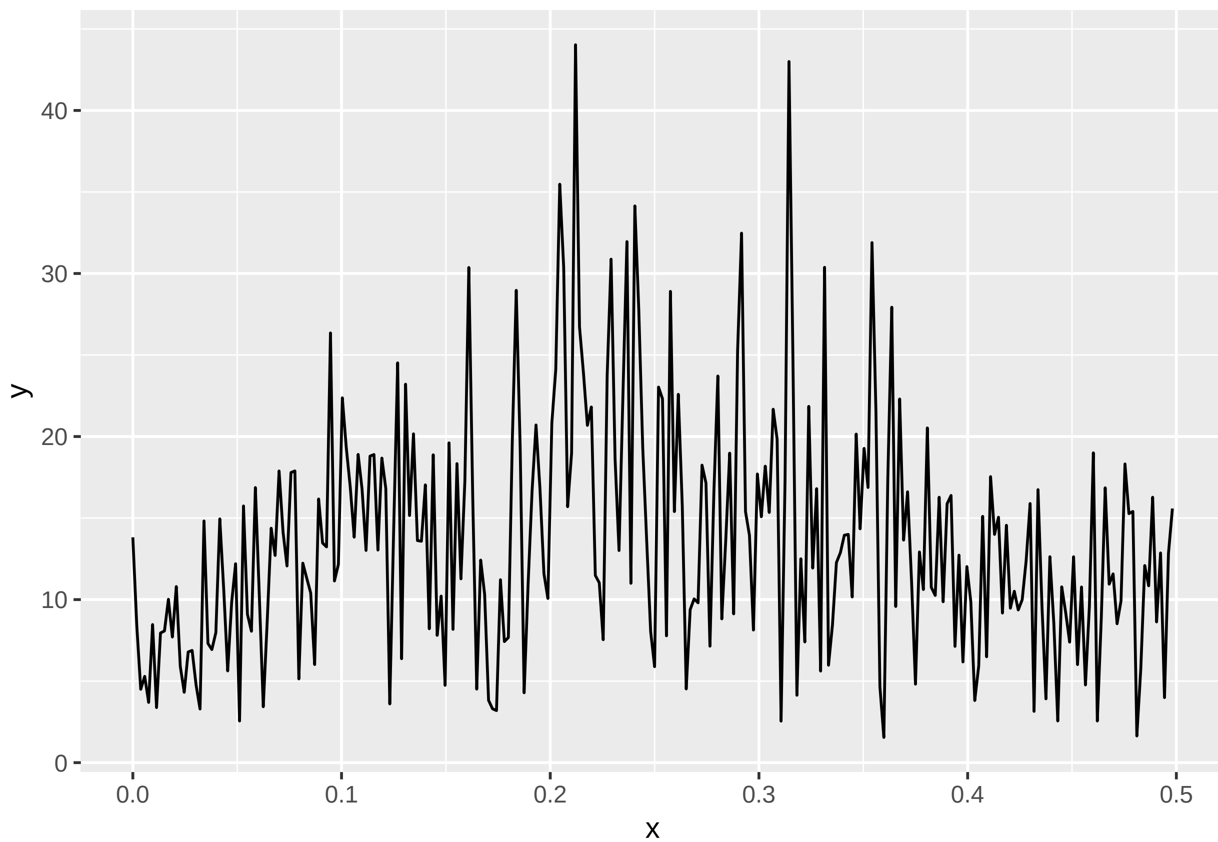
```
##          Index predict_price_xts
## 1  2019-02-15          76.75230
## 2  2019-02-22          75.83208
## 3  2019-03-01          75.95455
## 4  2019-03-08          75.82541
## 5  2019-03-15          75.89267
## 6  2019-03-22          75.99508
## 7  2019-03-29          75.95029
## 8  2019-04-05          75.78477
## 9  2019-04-12          75.53021
## 10 2019-04-19          75.21856
## 11 2019-04-26          74.88068
## 12 2019-05-03          74.55056
```

Then, we add each predicted model to get the final predicted price. The red line in the figure represents the data we forecast for the next 12 weeks. We see that, compared to the original ARIMA model, the new model predicts a relatively larger decline in AEP's stock prices over the next 12 weeks.

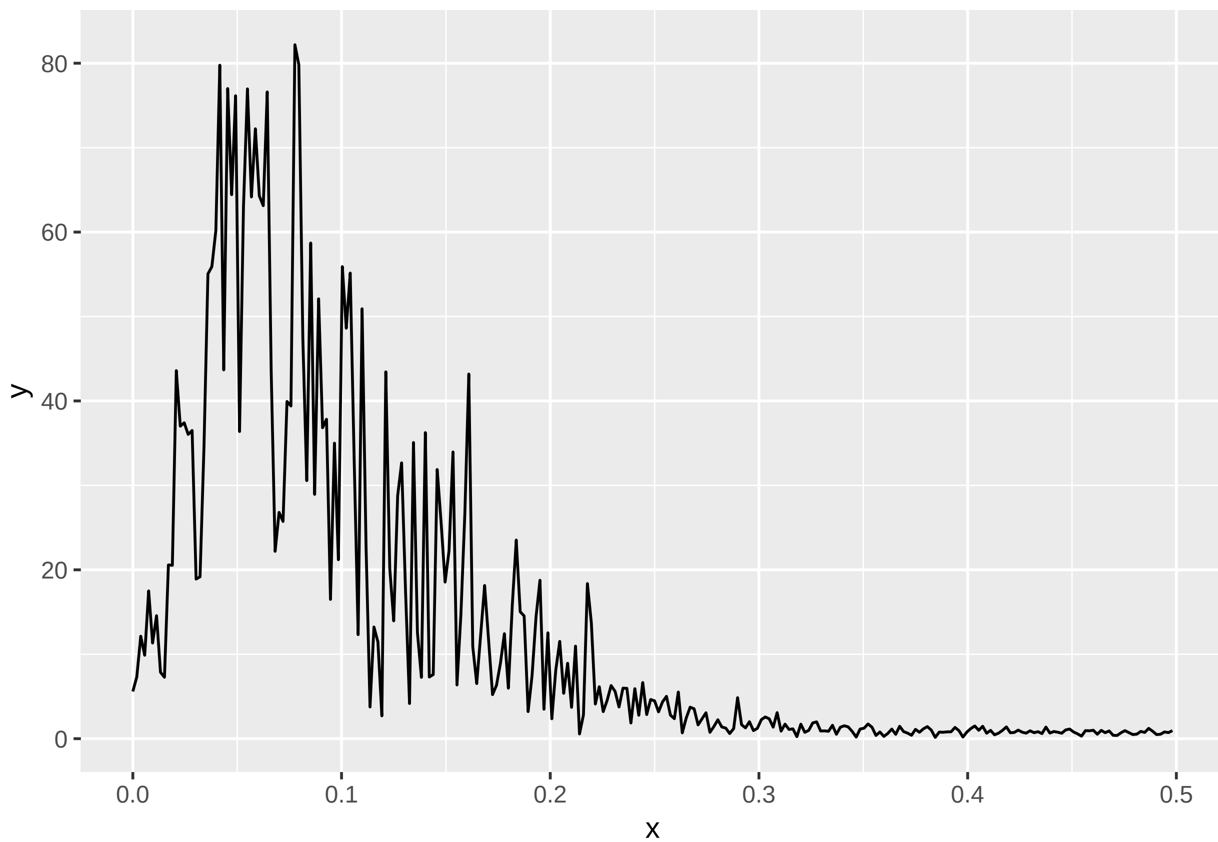
## Fourier transformation

```r
result[[4]]
```

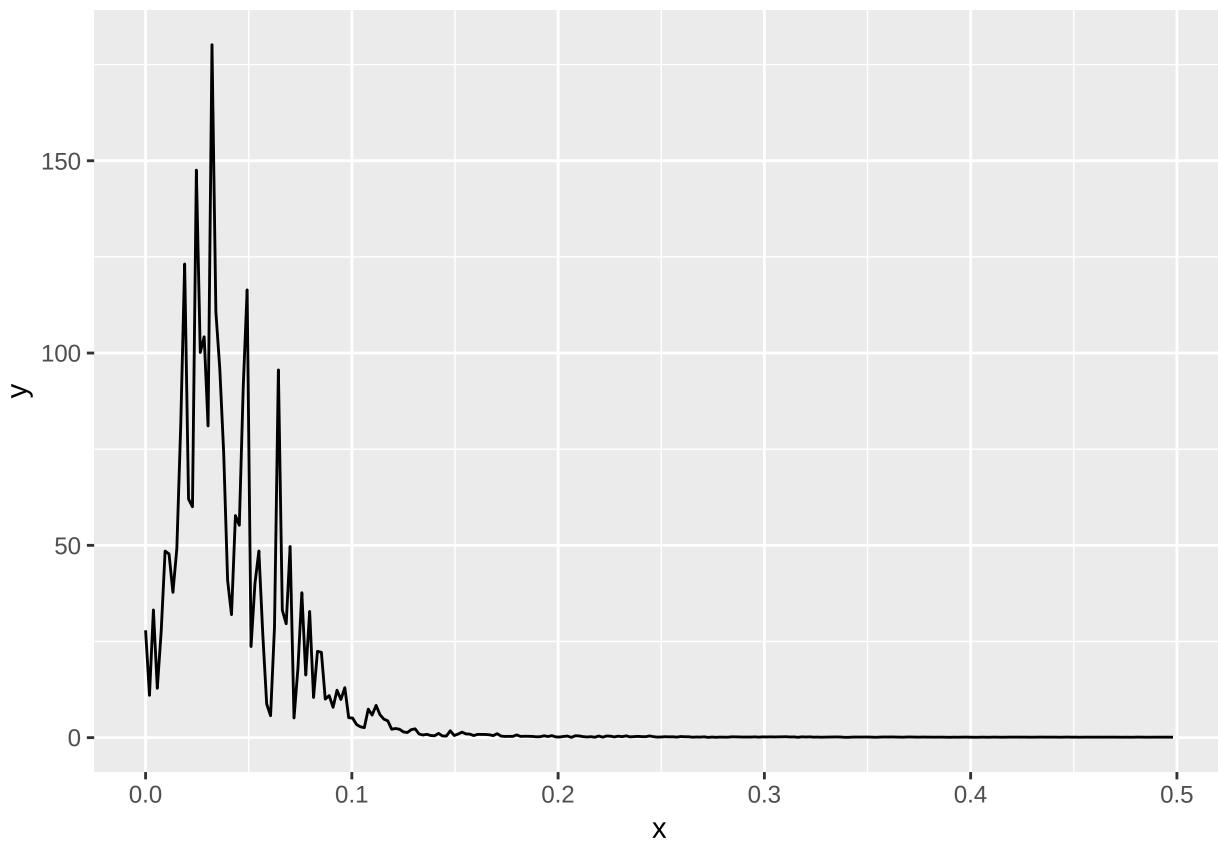
```
## [[1]]
```



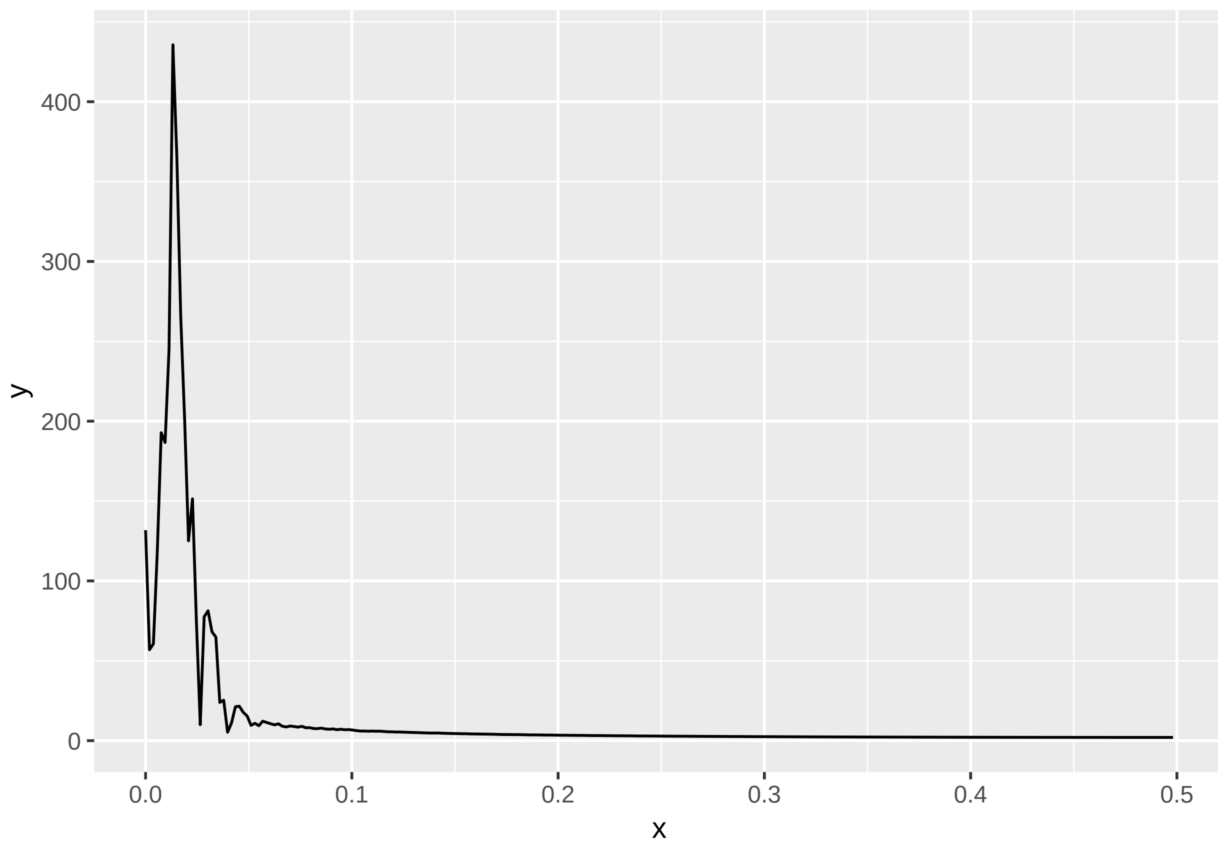

```
##
## [[2]]
```



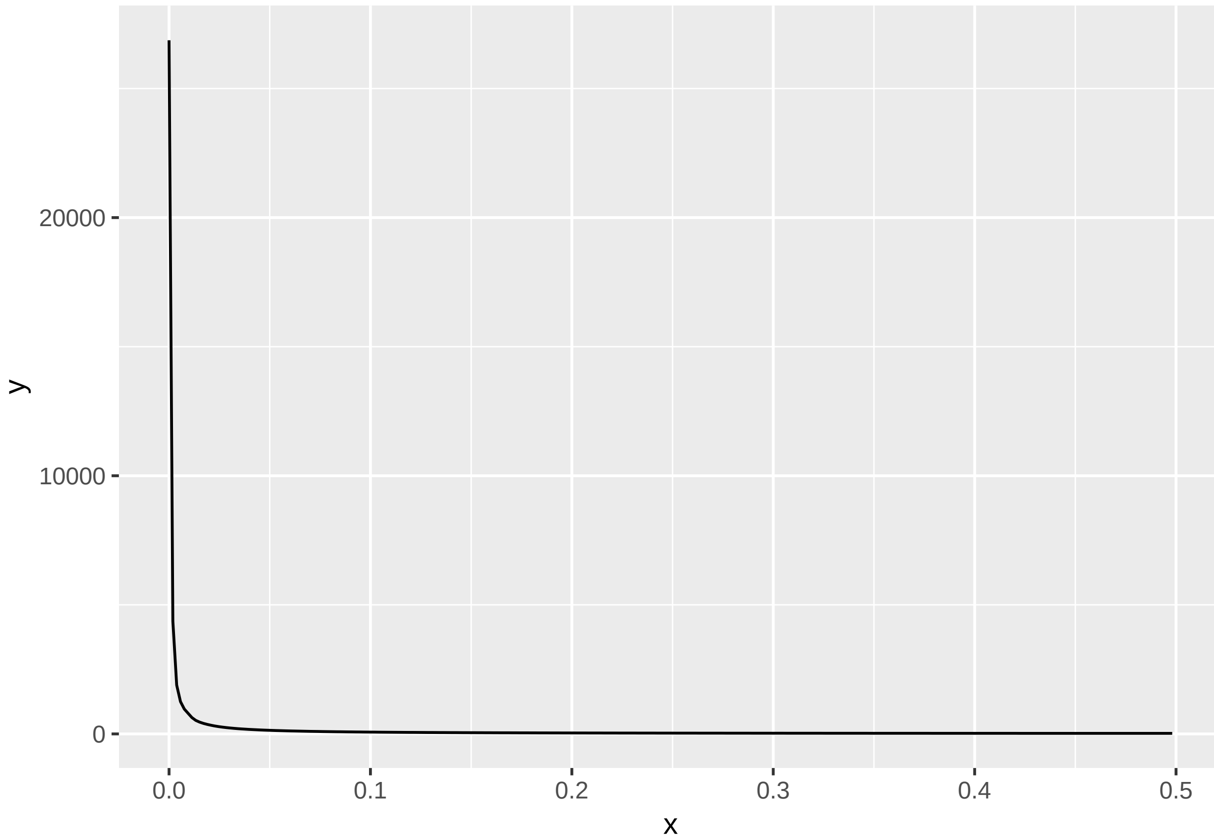

```
##
## [[3]]
```




```
##
## [[4]]
```




```
##
## [[5]]
```




```r
result[[3]]
```

```
## [1]  2.357143  6.439024 15.529412 37.714286       Inf
```

Finally, we use the Fourier transform to analyze the individual components that I decomposited from the original sequence. By observing the plot and calculating, we can get the cycle of different components. According to the calculation, we noticed that AEP's stock has fluctuation cycles in 2.357143, 6.439024, 15.529412 and 37.714286 weeks.