Time Series project

Qin Pu

3/26/2018

```r
if(!require("readxl")){
  install.packages("readxl")
  library("readxl")
}
```

```
## Loading required package: readxl
```

```r
setwd("~/Dropbox/MSP/2nd semester/time series/Fitbit project/FitBit Data")
filenames = list.files(pattern="*.xls")
df = lapply(filenames, function(x) read_xls(x, sheet=2))
fitbit = do.call(rbind, df)
```

```r
if(!require("forecast")){
  install.packages("forecast")
  library("forecast")
}
```

```
## Loading required package: forecast
```

```
## Warning: package 'forecast' was built under R version 3.4.2
```

```r
if(!require("ggplot2")){
  install.packages("ggplot2")
  library("ggplot2")
}
```

```
## Loading required package: ggplot2
```

```r
if(!require("vars")){
  install.packages("vars")
  library("vars")
}
```

```
## Loading required package: vars
```

```
## Loading required package: MASS
```

```
## Loading required package: strucchange
```

```
## Loading required package: zoo
```

```
## Warning: package 'zoo' was built under R version 3.4.3
```

```
##
## Attaching package: 'zoo'
```

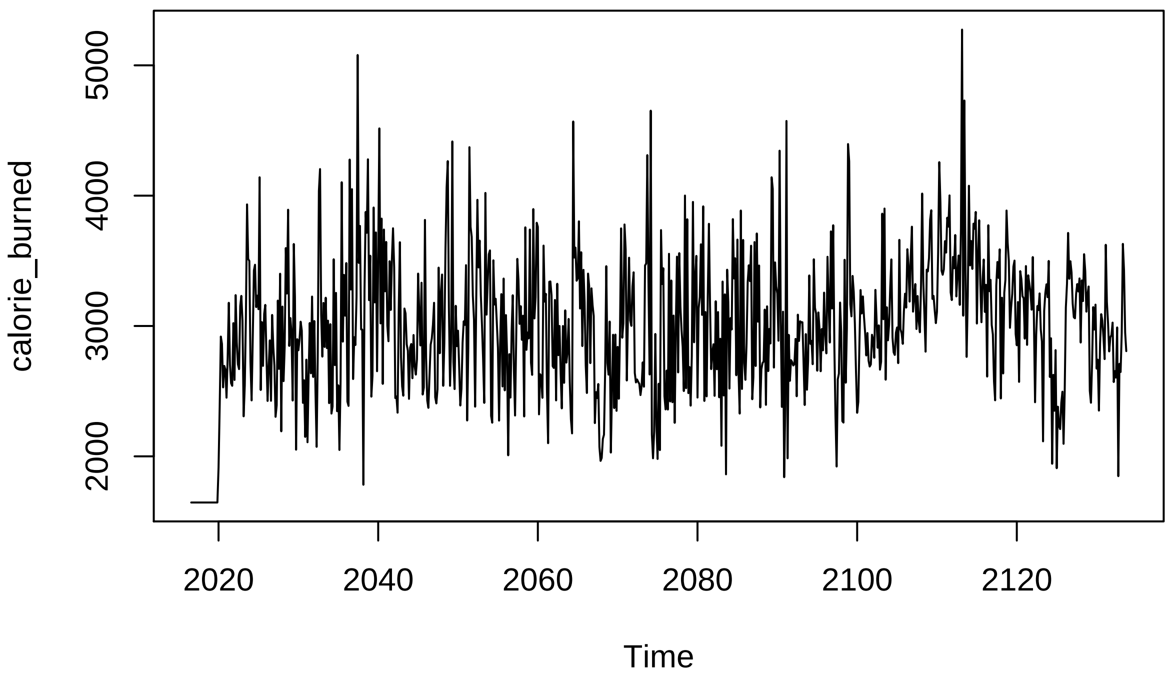
```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sandwich
```

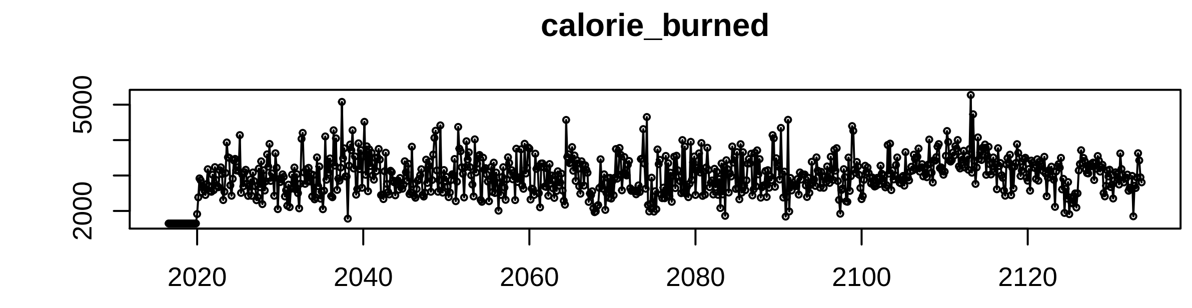
```
## Loading required package: urca
```

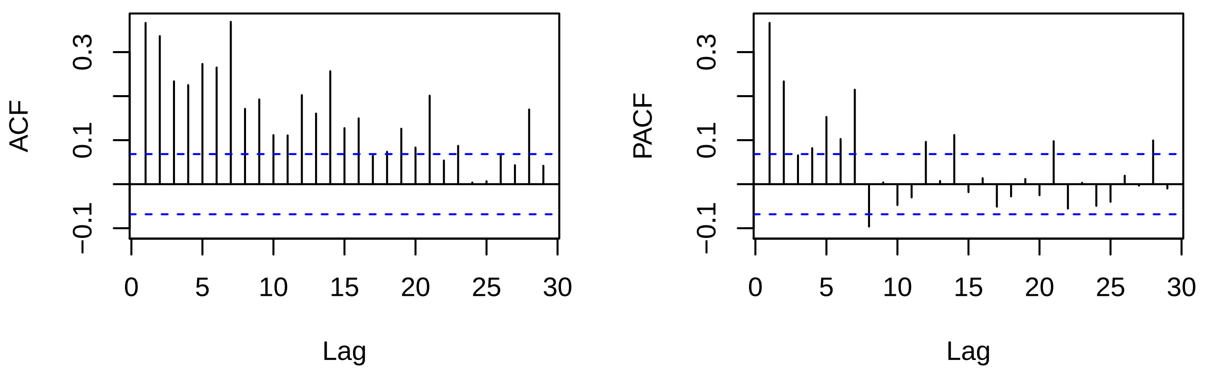
```
## Loading required package: lmtest
```

```
library(tseries)
for(i in 2:ncol(fitbit)){
  fitbit[i] = lapply(fitbit[i], function(x) gsub("[,]", "", x))
  fitbit[i] = lapply(fitbit[i], function(x) as.numeric(x))
}


calorie_burned = ts(fitbit$`Calories Burned`, start=c(2015,12,1), frequency=7)
plot.ts(calorie_burned)
```

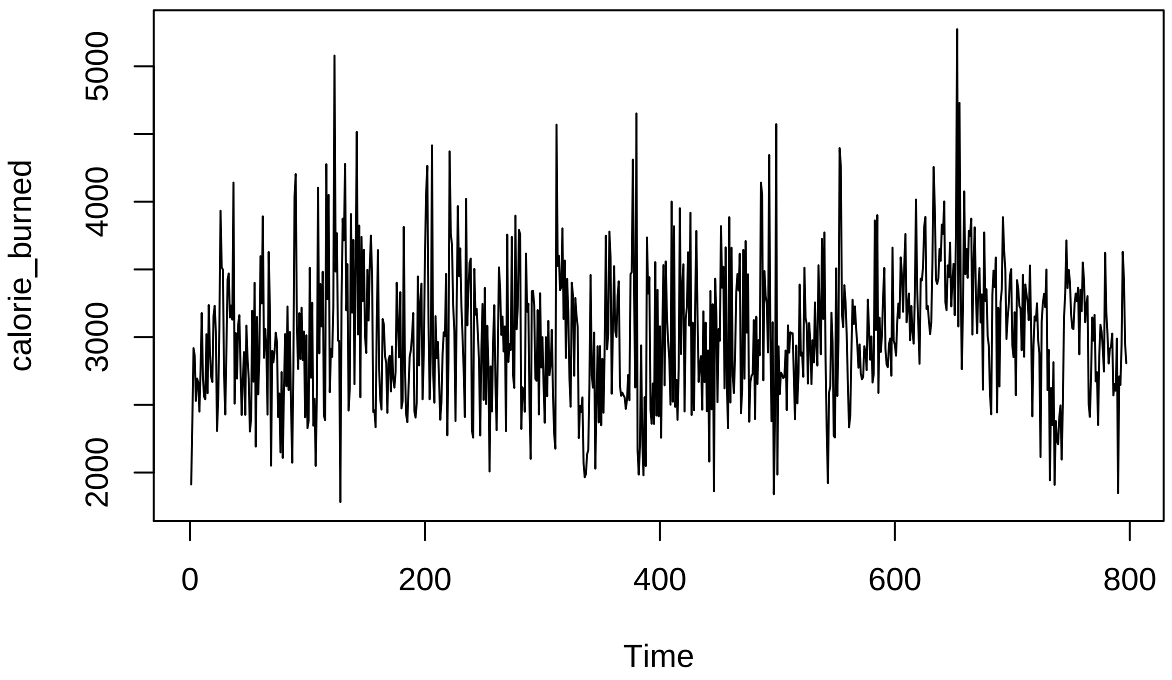


```
tsdisplay(calorie_burned)
```

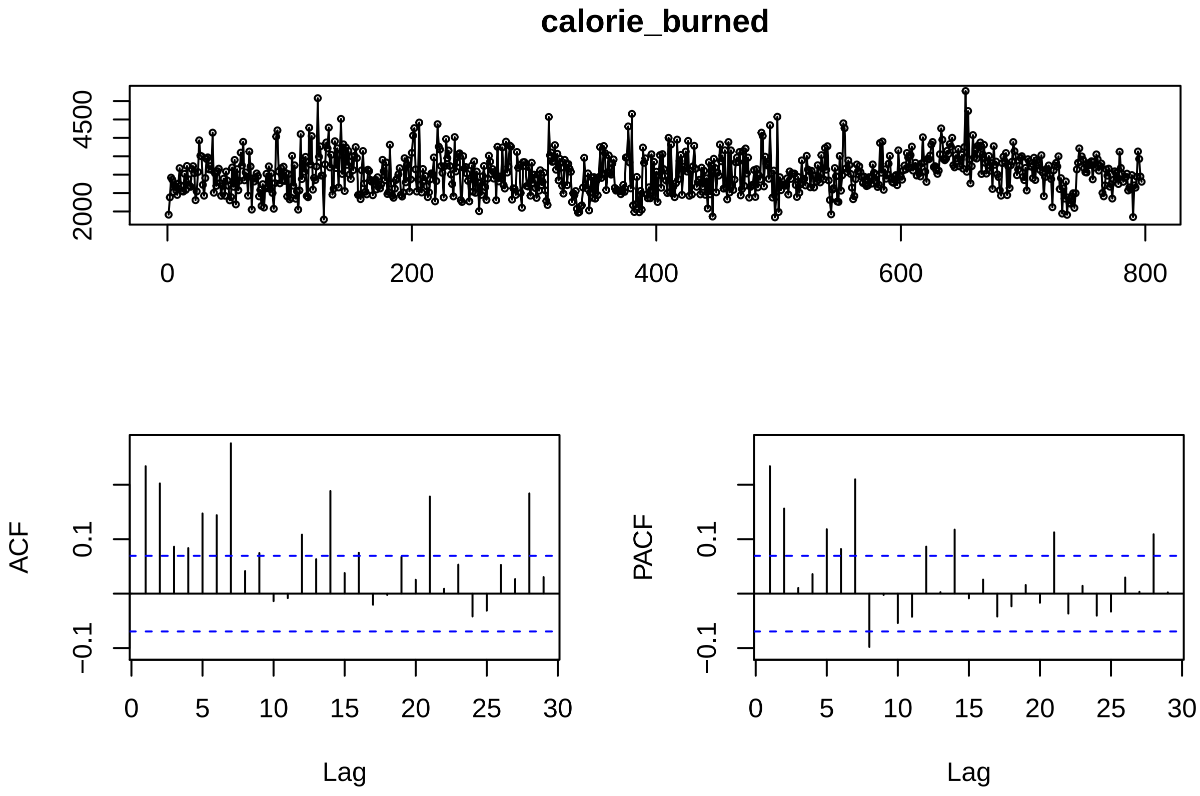




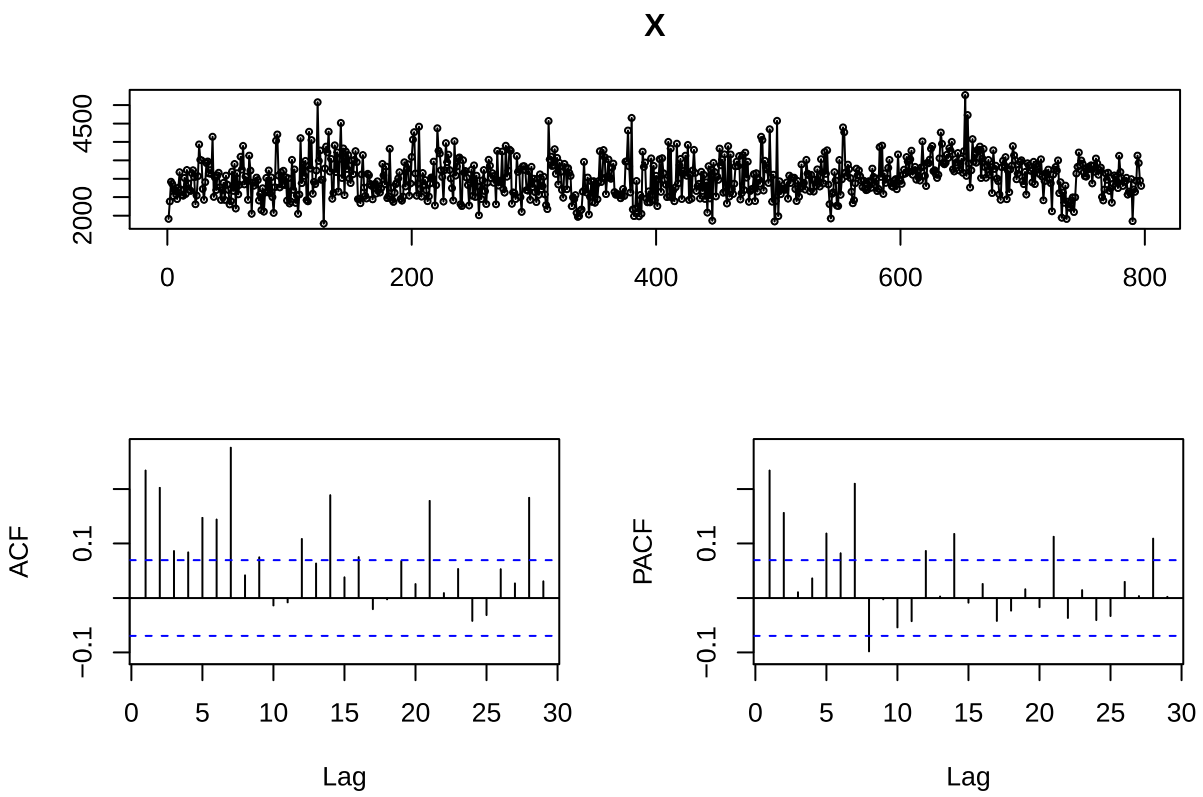
```r
# steps more than 20000
step.2000 = fitbit[which(fitbit$Steps>20000),]
# steps = 0
step.0 = fitbit[which(fitbit$Steps==0),]
#start to wear fitbit on 12.25

# delete data before then
fitbit.new = fitbit[which(fitbit$Steps!=0),]
calorie_burned = ts(fitbit.new$`Calories Burned`)
plot.ts(calorie_burned)
```



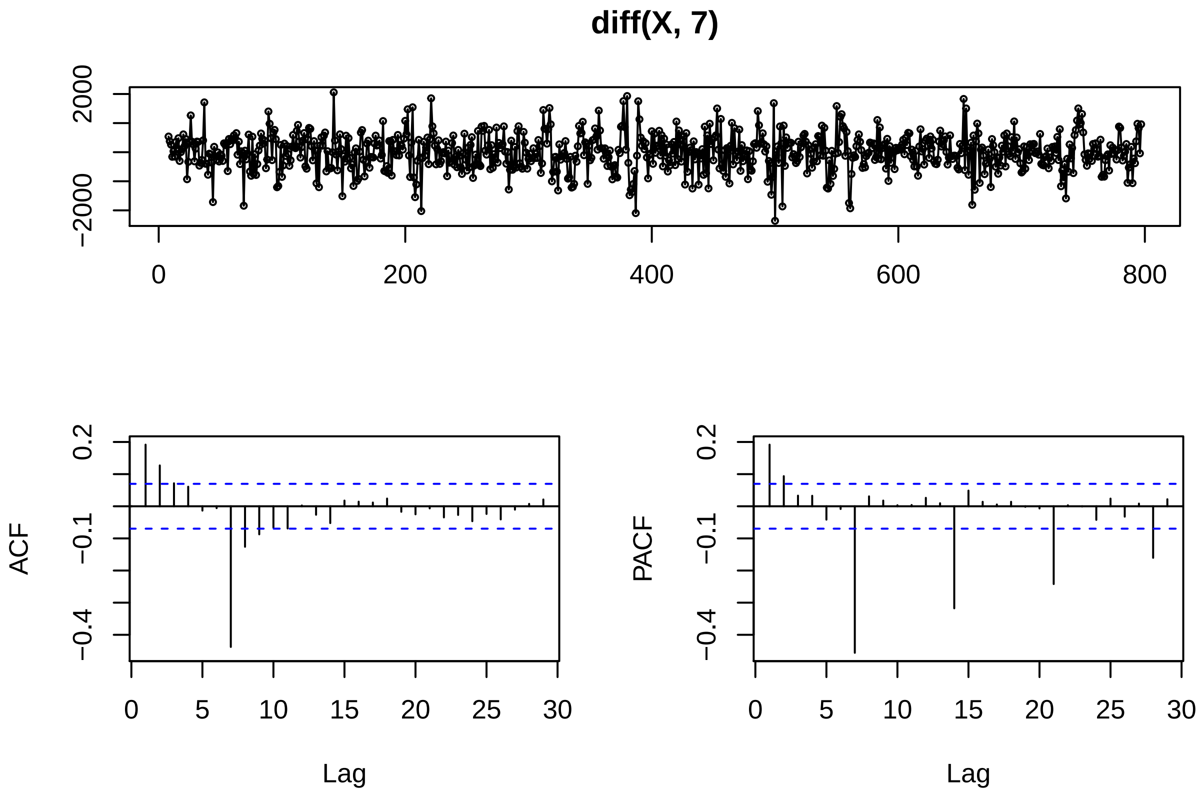
```r
tsdisplay(calorie_burned)
```



```r
X = calorie_burned
tsdisplay(X)
```



```r
tsdisplay(diff(X, 7)) # better
```



```r
adf.test(diff(X,7), alternative="stationary", k=0)
```

```
## Warning in adf.test(diff(X, 7), alternative = "stationary", k = 0): p-value
## smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(X, 7)
## Dickey-Fuller = -23.051, Lag order = 0, p-value = 0.01
## alternative hypothesis: stationary
```

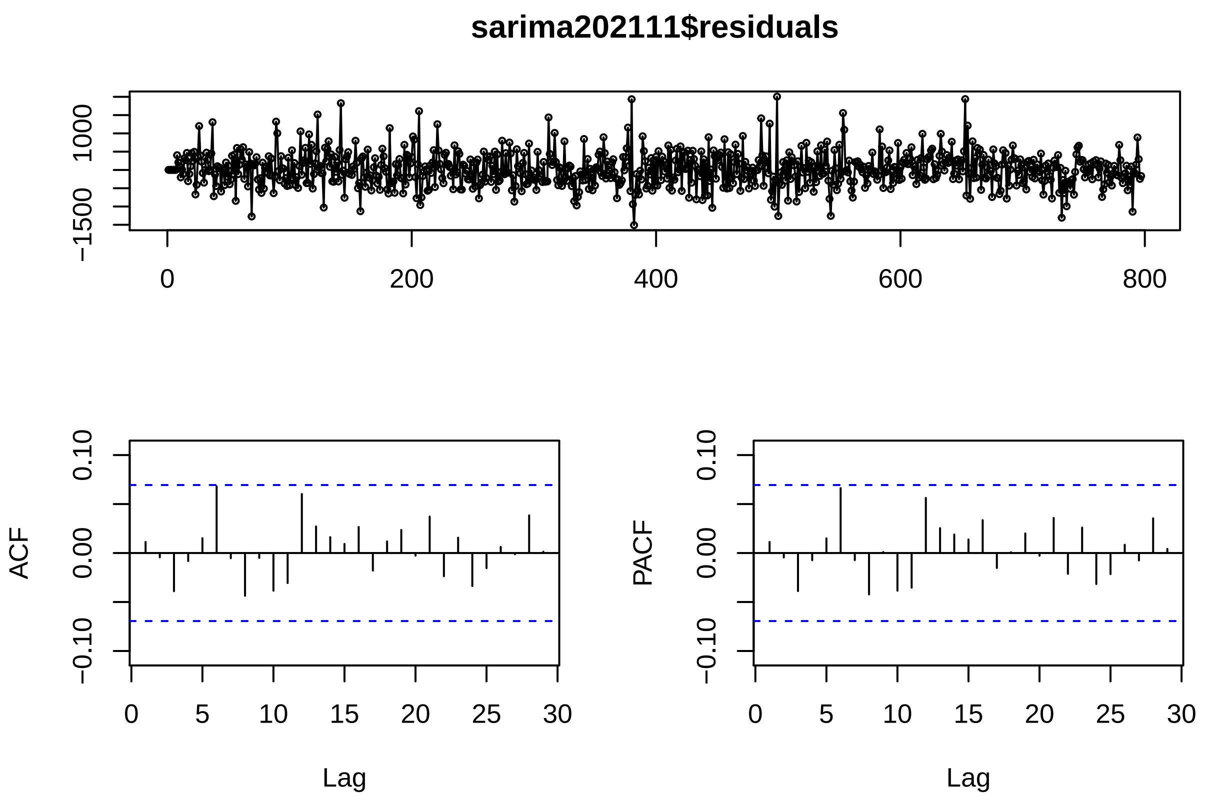
```r
sarima202112 = Arima(X, order = c(2, 0, 2), seasonal=list(order=c(1,1,2), period=7))
sarima202111 = Arima(X, order = c(2, 0, 2), seasonal=list(order=c(1,1,1), period=7))
sarima202113 = Arima(X, order = c(2, 0, 2), seasonal=list(order=c(1,1,3), period=7))
AIC(sarima202111, sarima202112, sarima202113) #sarima202112
```

```
##              df      AIC
## sarima202111  7 11967.51
## sarima202112  8 11966.14
## sarima202113  9 11971.59
```

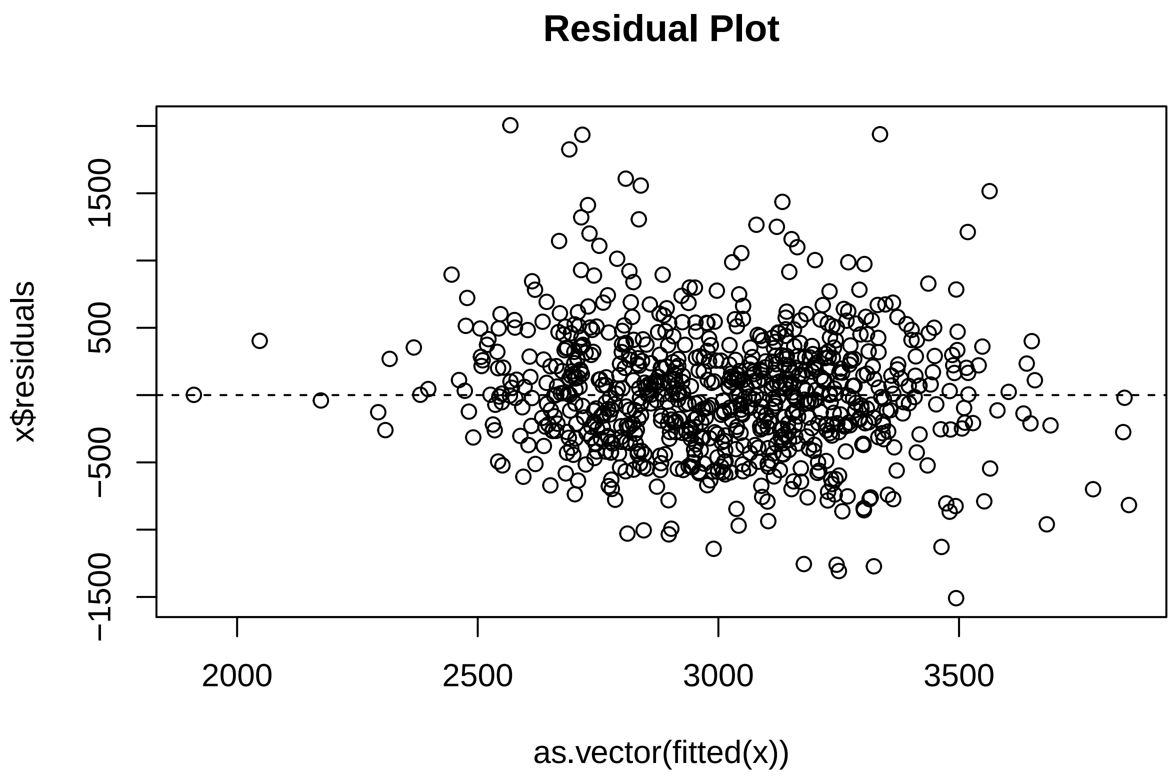
```r
BIC(sarima202111, sarima202112, sarima202113) #sarima202111
```

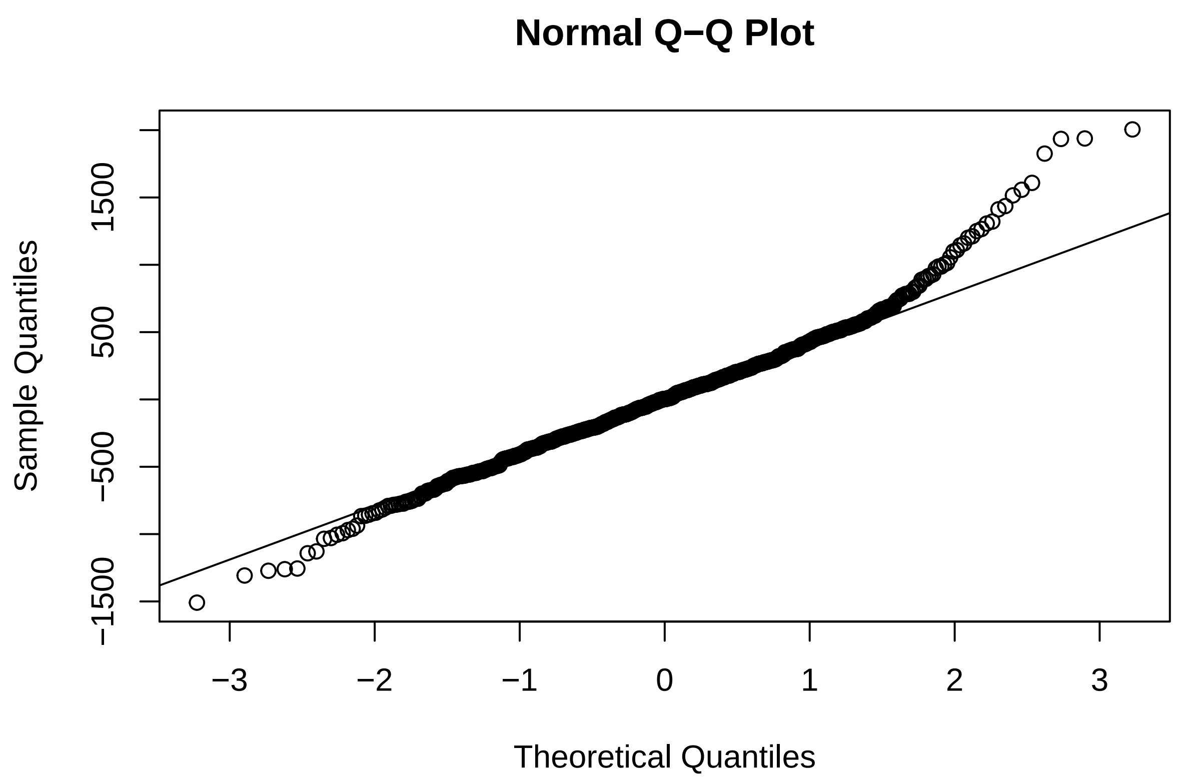
```
##              df      BIC
## sarima202111  7 12000.22
## sarima202112  8 12003.52
## sarima202113  9 12013.64
```

```r
# choose sarima101111
tsdisplay(sarima202111$residuals)
```



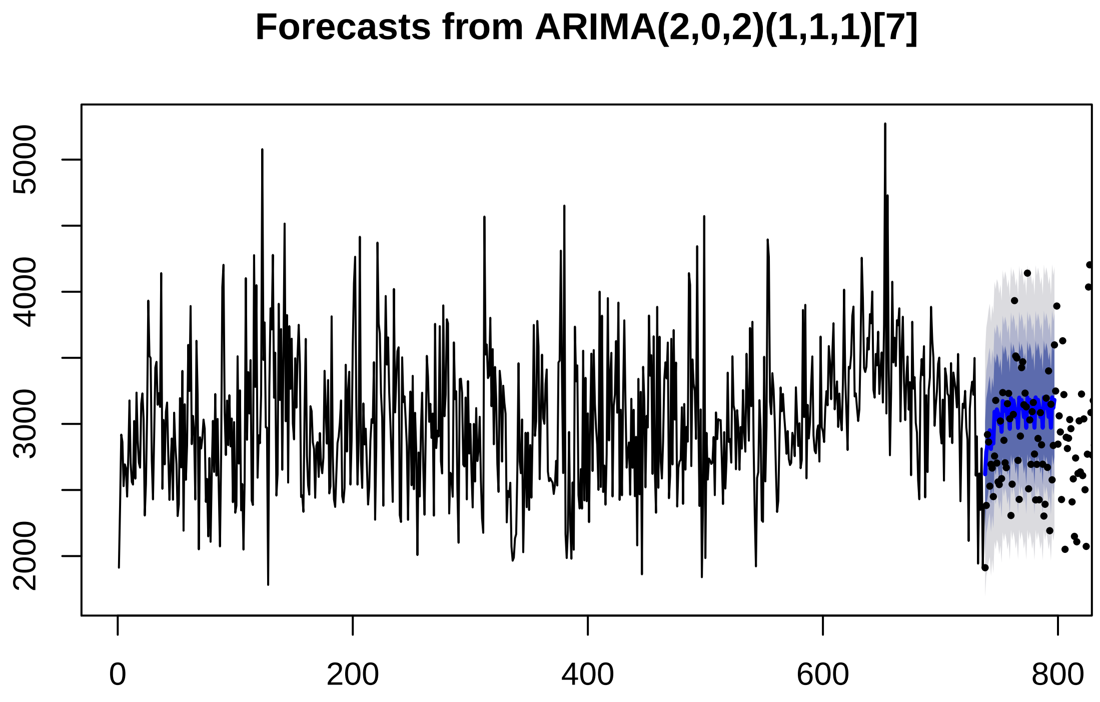
```r
diag = function(x){
  # test for constant variance
  plot(as.vector(fitted(x)), x$residuals, main="Residual Plot")
  abline(h=0, lty=2)
  # assessing normality
  qqnorm(x$residuals)
  qqline(x$residuals)
}
diag(sarima202111)
```





```r
#Box.test(sarima202111$residuals, 1, type="Ljung-Box", fitdf=14)
sarima_2 = Arima(X[1:(length(X)-60)], order = c(2, 0, 2), seasonal=list(order=c(1,1,1), period=7))
last60 = forecast(sarima_2, 60, level = c(60, 80, 95))
plot(last60)
points(ts(X, start=length(calorie_burned)-60+1), end=length(calorie_burned), pch=16, cex=.5)
```

```
## Warning in plot.xy(xy.coords(x, y), type = type, ...): "end" is not a
## graphical parameter
```



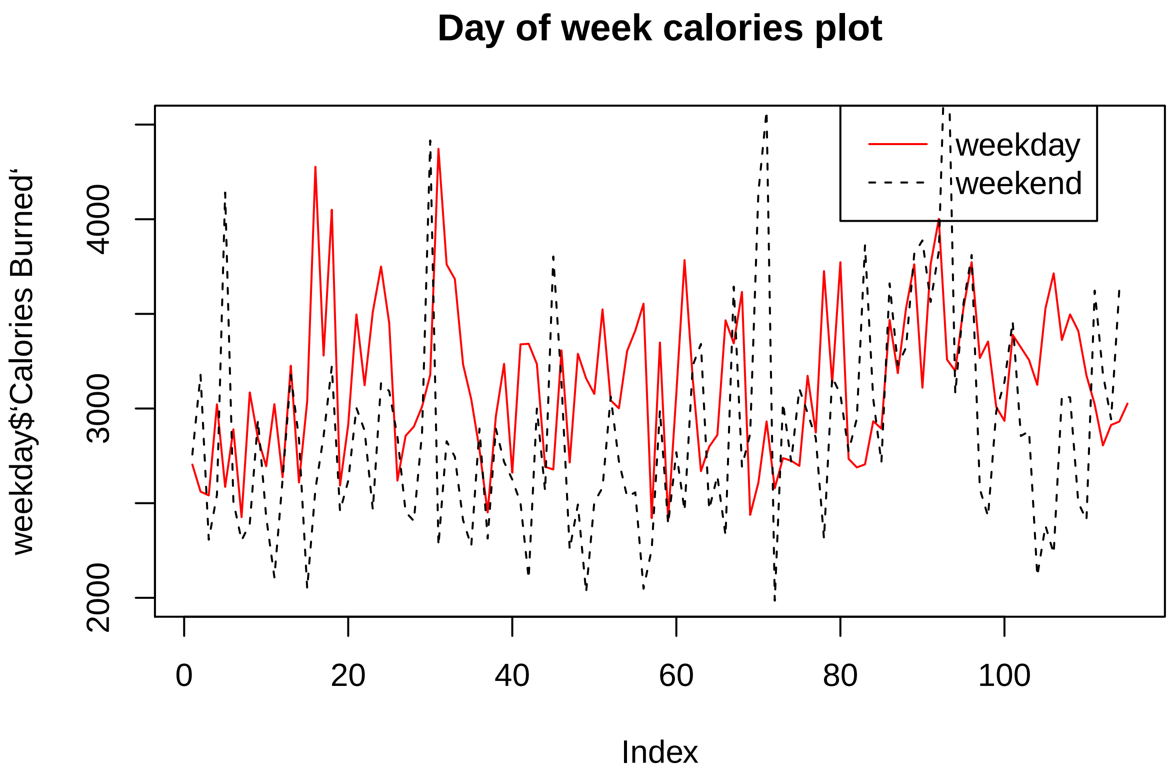
```r
# set up time
fitbit.new$Date = as.Date(fitbit.new$Date)
fitbit.new$day_of_week = weekdays(fitbit.new$Date)
fitbit.new$week = strftime(fitbit.new$Date, format = "%V")
fitbit.new$month = months(fitbit.new$Date)
fitbit.new$year =  as.numeric(substr(fitbit.new$Date,1,4))
fitbit.new$quarters = paste(fitbit.new$year,quarters(fitbit.new$Date),sep = "-")
weekday = fitbit.new[which(fitbit.new$day_of_week==c("Monday","Tuesday","Wednesday", "Thursday", "Frida
```

```
## Warning in fitbit.new$day_of_week == c("Monday", "Tuesday", "Wednesday", :
## longer object length is not a multiple of shorter object length
```

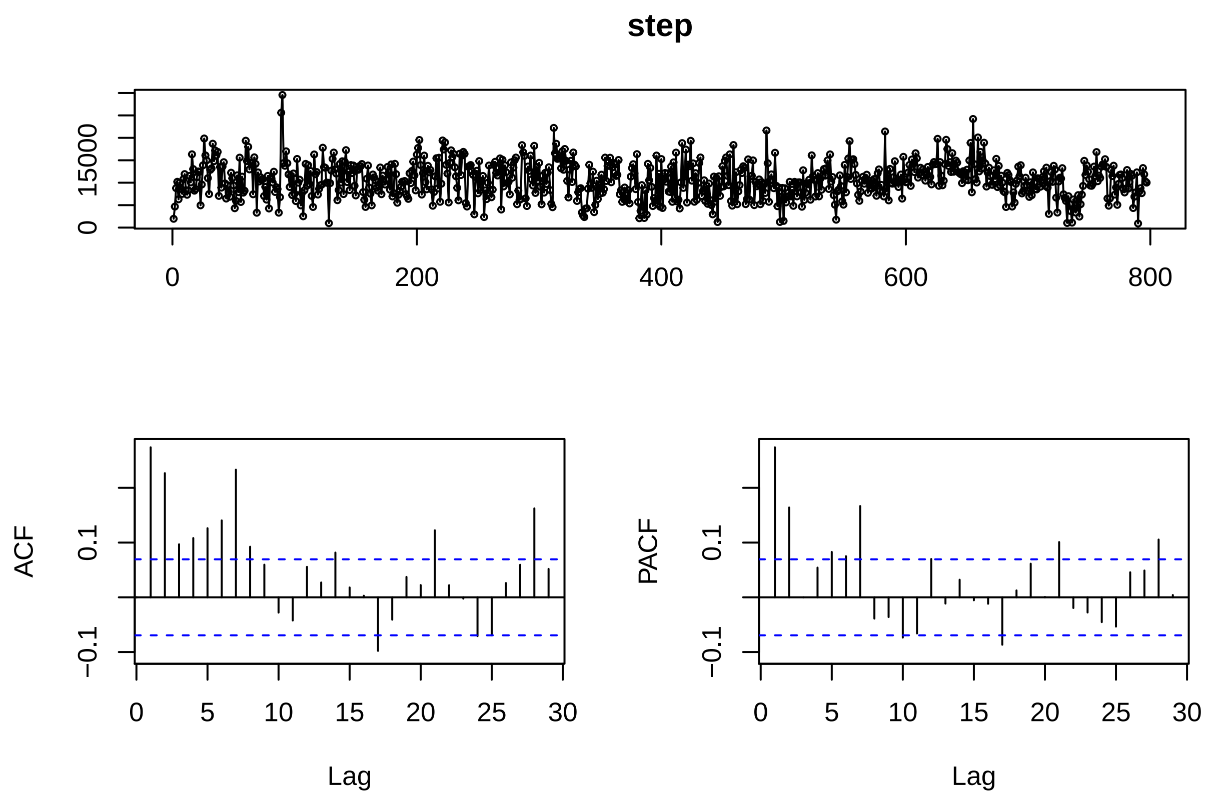
```r
weekend = fitbit.new[which(fitbit.new$day_of_week==c("Saturday", "Sunday")),]
```

```
## Warning in fitbit.new$day_of_week == c("Saturday", "Sunday"): longer object
## length is not a multiple of shorter object length
```

```r
plot(weekday$`Calories Burned`, type = "l", col = "red", lty = 1,
     ylim = c(2000,4500), main = "Day of week calories plot")
lines(weekend$`Calories Burned`, col = "black", lty = 2)
legend(80,4600, legend = c("weekday", "weekend"), col =c("red", "black"), lty =1:2)
```



```r
#test stationary
step = ts(fitbit.new$Steps)
tsdisplay(step)
```



```r
adf.test(step)
```

```
## Warning in adf.test(step): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  step
## Dickey-Fuller = -7.1728, Lag order = 9, p-value = 0.01
## alternative hypothesis: stationary
```

```r
summary(step)
```

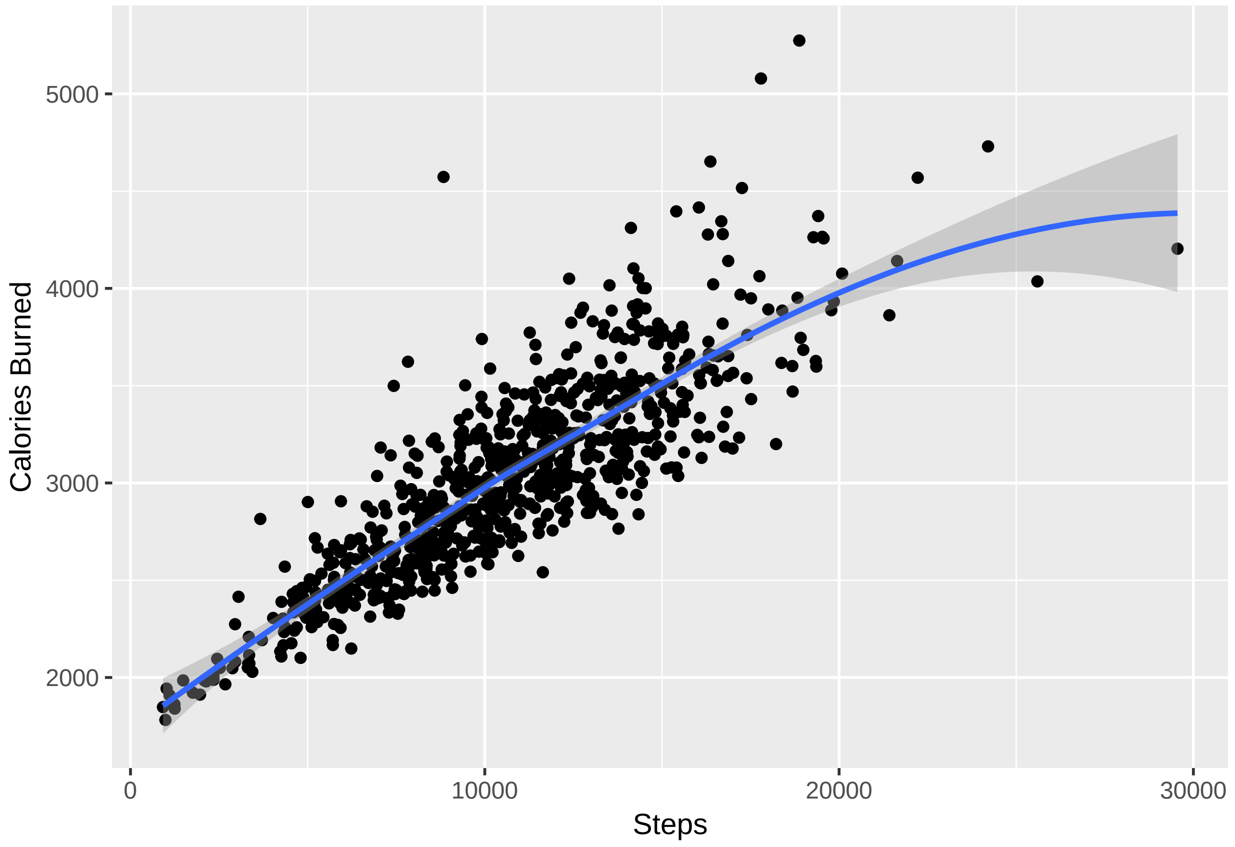
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     917    7900   10459   10610   13365   29552
```

```r
summary(lm(fitbit.new$`Calories Burned`~fitbit.new$Steps))
```

```
##
## Call:
## lm(formula = fitbit.new$`Calories Burned` ~ fitbit.new$Steps)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -921.85 -169.27  -32.47  143.78 1746.10
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      1.846e+03  2.774e+01   66.56   <2e-16 ***
## fitbit.new$Steps 1.110e-01  2.452e-03   45.25   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 271.5 on 795 degrees of freedom
## Multiple R-squared:  0.7203, Adjusted R-squared:   0.72
## F-statistic:  2048 on 1 and 795 DF,  p-value: < 2.2e-16
```

```
ggplot(fitbit.new, aes(x=Steps, y=`Calories Burned`)) + geom_point() + geom_smooth()
```

```
## `geom_smooth()` using method = 'loess'
```


Calories Burned
Steps