WA Precipitation from 1895-2017 Report

Initially we import WA precipitation data with date from 1895 to 2017 and monthly average precipitation (inches).

```
library('ggplot2')
library('forecast')
```

```
## Warning: package 'forecast' was built under R version 3.4.4
```

```
library('tseries')
```

```
## Warning: package 'tseries' was built under R version 3.4.4
```

```
library('lubridate')
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

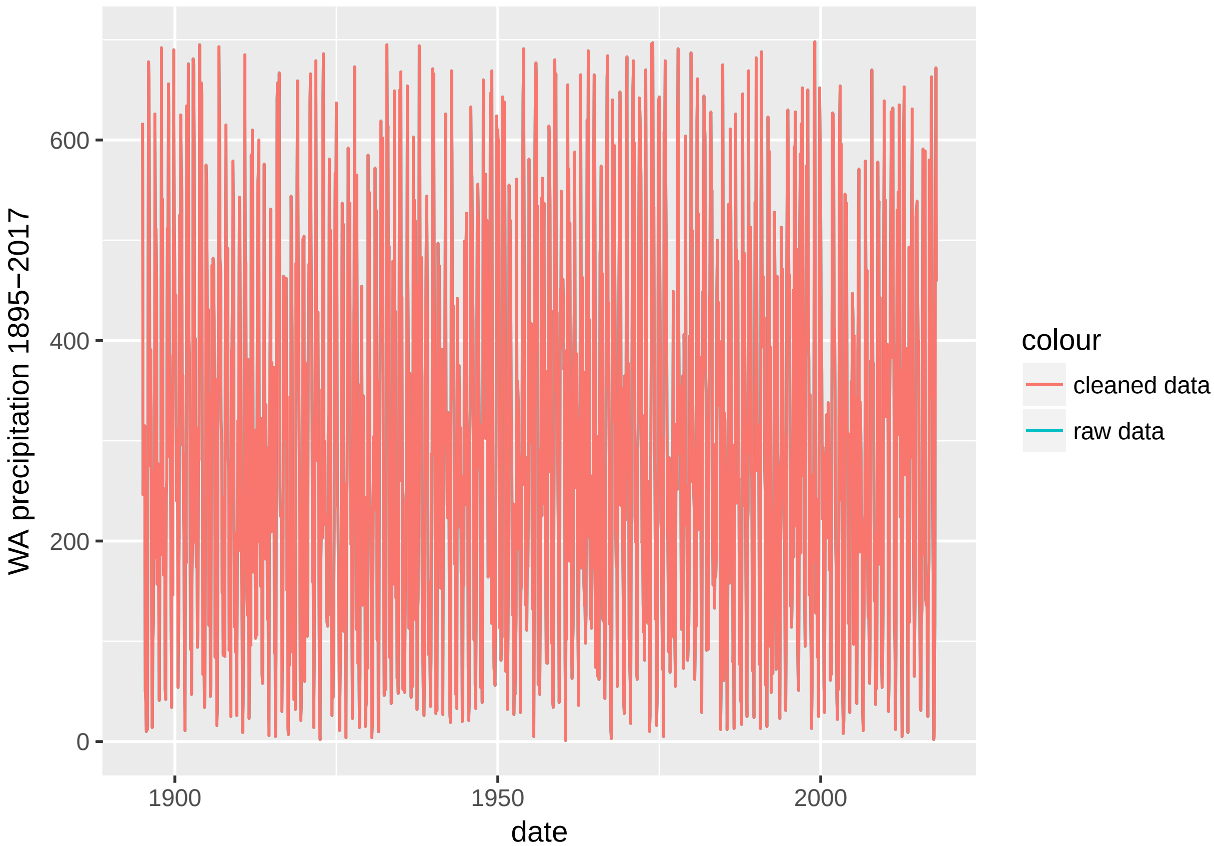
```
data = read.csv('https://www.ncdc.noaa.gov/cag/statewide/time-series/45-pcp-all-1-1895-2017.csv', header
date <- data$Washington[-c(1,2,3)]
value <- data$Precipitation[-c(1,2,3)]
df <- data.frame(date, value)
df$date <- ymd(paste0(as.character(df$date), '01'))
head(df)
```

```
##         date value
## 1 1895-01-01  7.53
## 2 1895-02-01  2.54
## 3 1895-03-01  2.95
## 4 1895-04-01  3.00
## 5 1895-05-01  3.29
## 6 1895-06-01  0.62
```

We then use ts() function to convert the the numeric vector, "precipitation", to time series object. After that, we plot the time series of out raw data. We then use tsclean() to identify and replace outliers using series smoothing and decomposition. However, the cleaned version seems not any different. (Probably because no outliers).

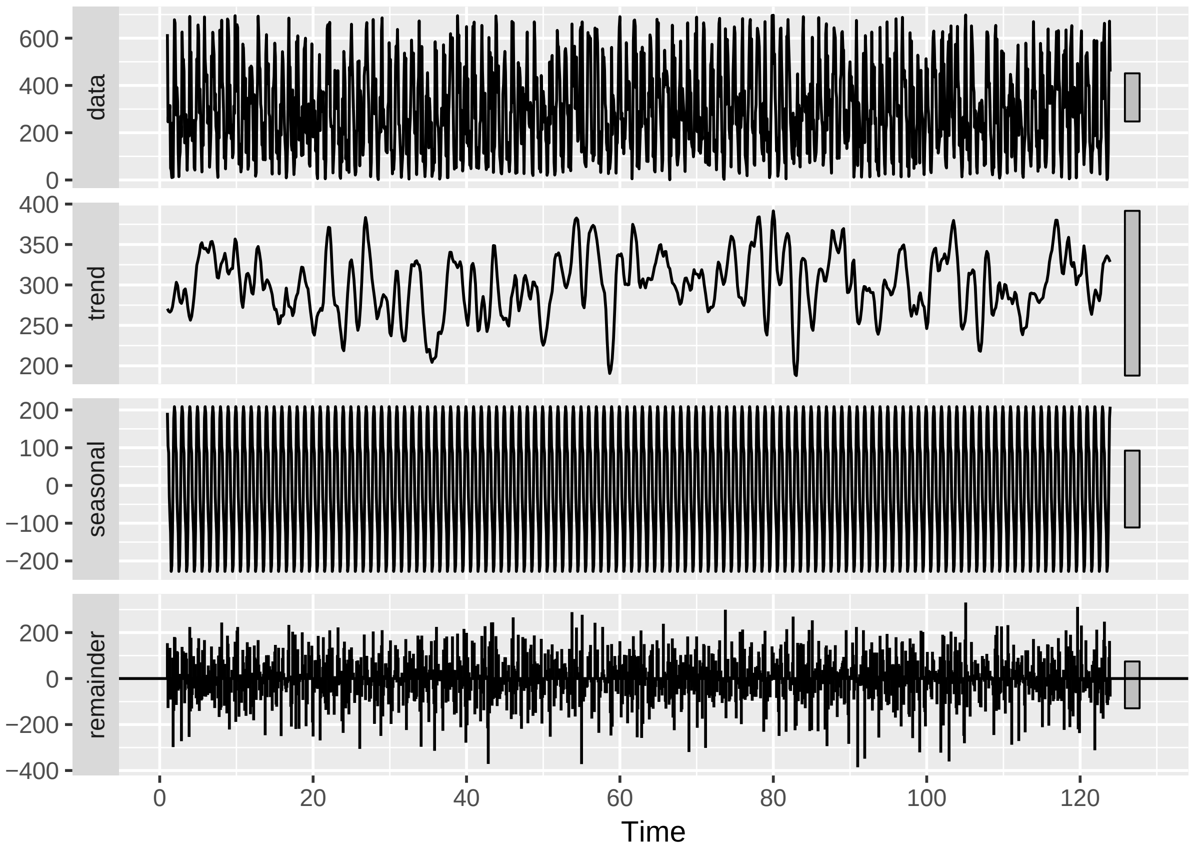
```
value_ts = ts(df[, c('value')])
df$clean_value = tsclean(value_ts)
ggplot() +
  geom_line(data = df, aes(x = date, y = value_ts, color = "raw data" ))  +
  geom_line(data = df, aes(x = date, y = clean_value, color = "cleaned data")) + ylab('WA precipitation
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```



Since we use monthly data and precipitation is related to season, we might consider to decompose the seasonal component.

```
value_ma <- ts(na.omit(df$clean_value), frequency=12)
decomp <- stl(value_ma, s.window="periodic")
deseasonal_value <- seasadj(decomp)
autoplot(decomp)
```



Then we use Dickey-Fuller test to determine whether the time series is stationary.

```
adf.test(value_ma, alternative = "stationary")
```

```
## Warning in adf.test(value_ma, alternative = "stationary"): p-value smaller
## than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  value_ma
## Dickey-Fuller = -9.2022, Lag order = 11, p-value = 0.01
## alternative hypothesis: stationary
```

Our null hypothesis is "the time series is not stationary". Notice the p-value is less than 0.05, we can reject our null hypothesis. Thus the time series is stationary.
We will run the auto.arima function to see which ARIMA model fits our data best.

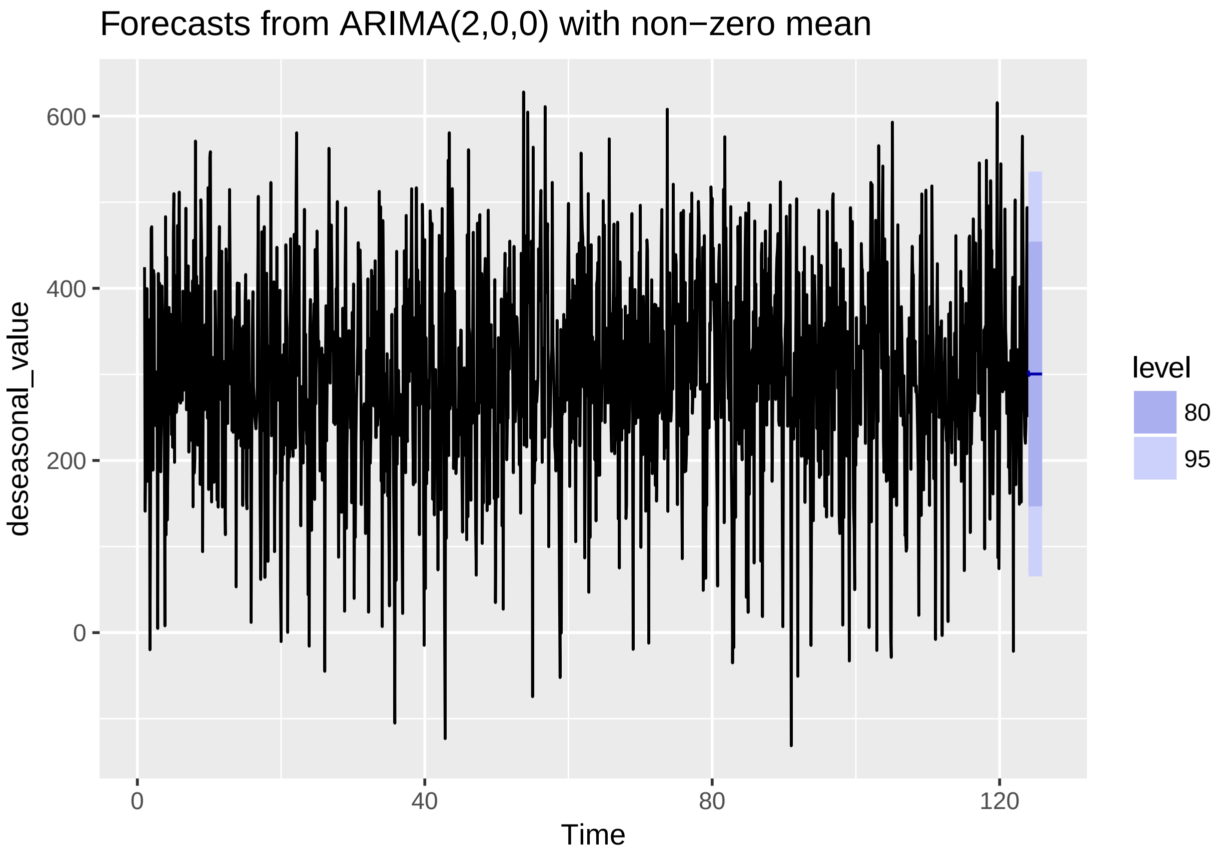
```
fit <- auto.arima(deseasonal_value, seasonal=FALSE)
fit
```

```
## Series: deseasonal_value
## ARIMA(2,0,0) with non-zero mean
##
## Coefficients:
##         ar1    ar2      mean
##       0.079  0.041  300.4608
## s.e.  0.026  0.026    3.5287
##
## sigma^2 estimated as 14268:  log likelihood=-9152.4
```

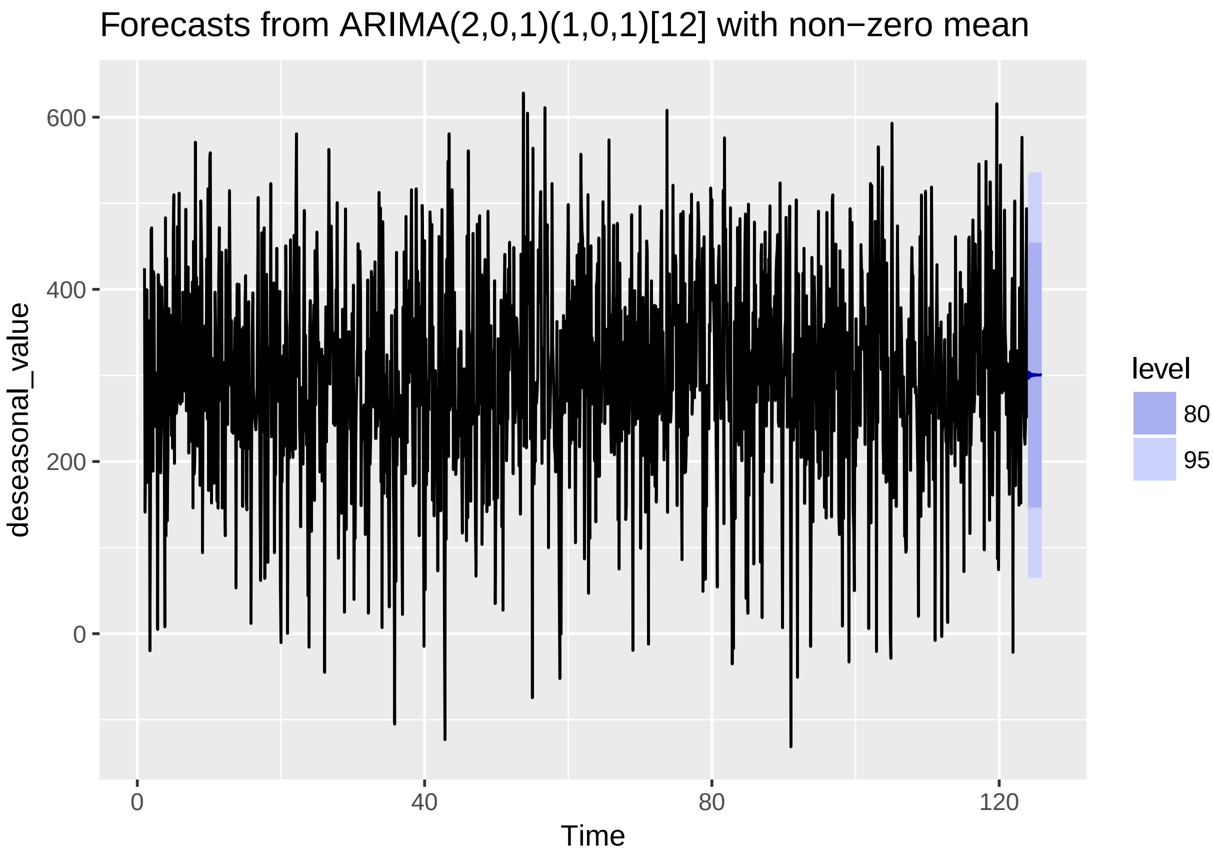
```
## AIC=18312.8   AICc=18312.83   BIC=18333.99
```

Finally we can use this model to do the forecast.

```
fcast <- forecast(fit)
autoplot(fcast)
```




```
fitwithseason <- auto.arima(deseasonal_value, seasonal=TRUE)
fcast2 <- forecast(fitwithseason)
autoplot(fcast2)
```


Forecasts from ARIMA(2,0,1)(1,0,1)[12] with non−zero mean
deseasonal_value
600
400
200
0
0
40
80
120
Time
level
80
95