Introduction to R timeseries

Quan Nguyen

May 7, 2016

We will use the Canadian car sales data to study a time series.

The data has been extracted from:

http://www.goodcarbadcar.net/2012/10/canada-overall-auto-industry-sales-figures.html

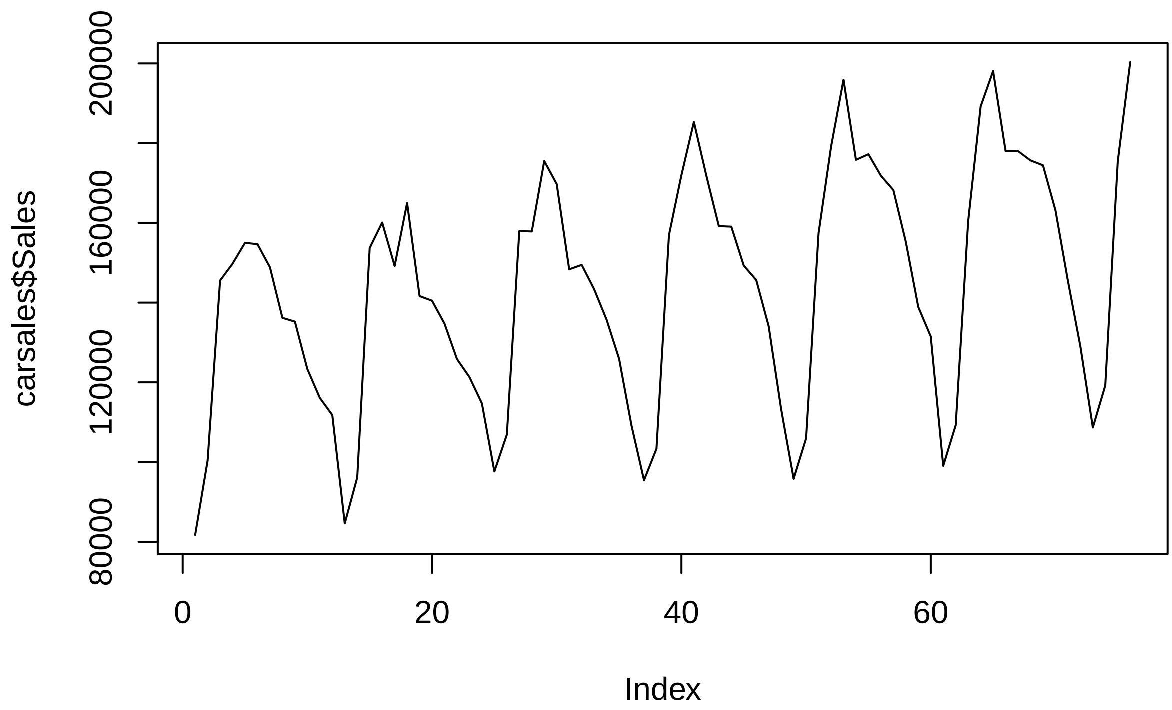
It contains monthly sales data of number of cars sold in Canada since January 2010.

Prepare the data

```r
carsales = read.csv("canadian_carsales.csv", header=T, stringsAsFactors = F)
str(carsales)
```

```
## 'data.frame':    76 obs. of  3 variables:
##  $ Month: chr  "January" "February" "March " "April " ...
##  $ Year : int  2010 2010 2010 2010 2010 2010 2010 2010 2010 2010 ...
##  $ Sales: chr  "81,693" "100,352" "145,539" "149,757" ...
```

```r
# convert Sales column to numeric
carsales$Sales = as.numeric(gsub(",", "", carsales$Sales))
# the x-axis does not show the year/month
plot(carsales$Sales, type='l')
```

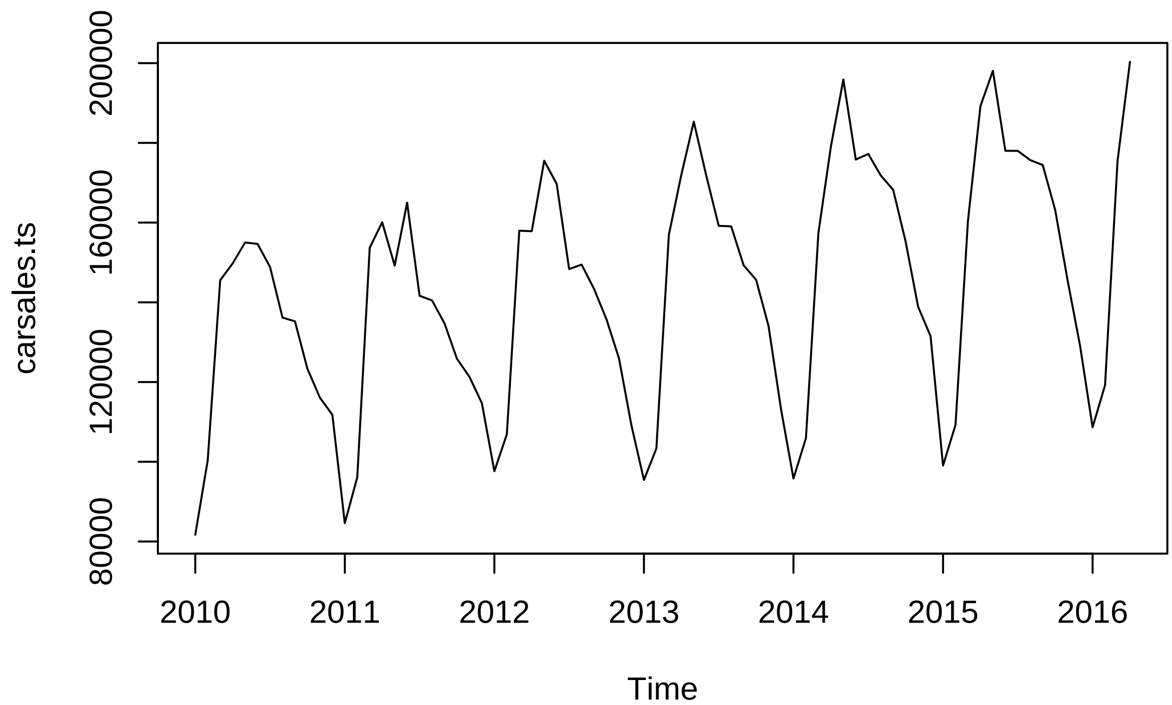


Convert data frame into a timeseries using ts() function

```
carsales.ts = ts(carsales$Sales, frequency=12, start=c(2010,1))
str(carsales.ts)
```

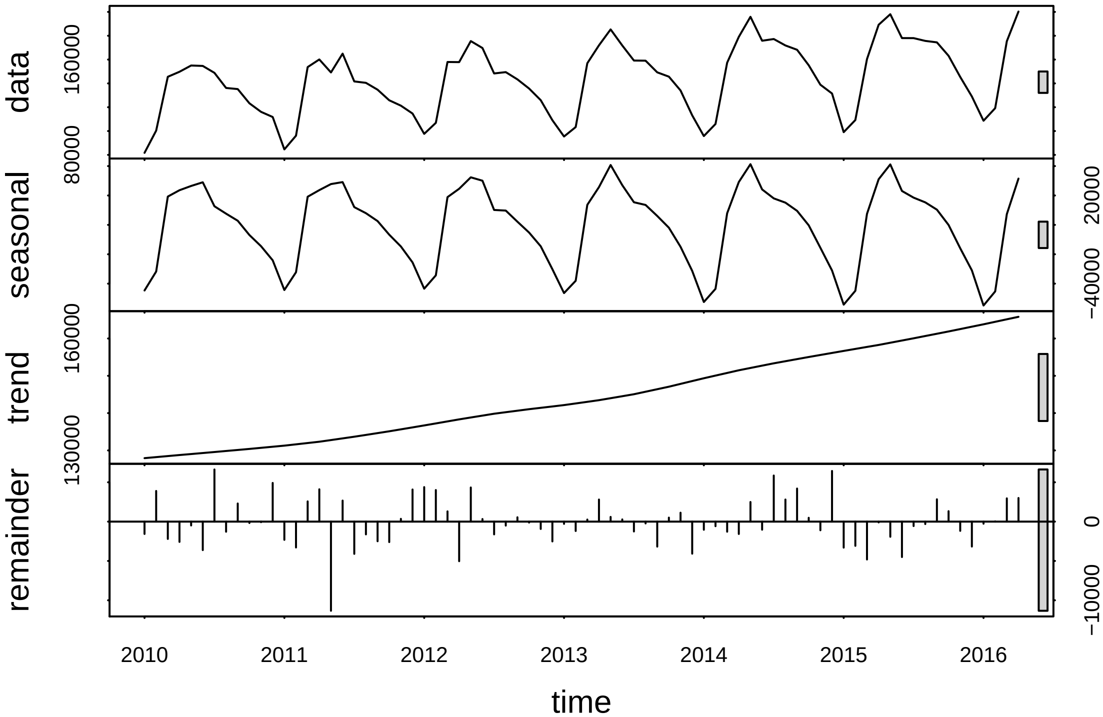
```
##  Time-Series [1:76] from 2010 to 2016: 81693 100352 145539 149757 155008 ...
```

```
# the x-axis should now show the year/month
plot(carsales.ts)
```



Time series decomposition using stl() function

```
carsales.stl = stl(carsales.ts, s.window = 4)
plot(carsales.stl)
```

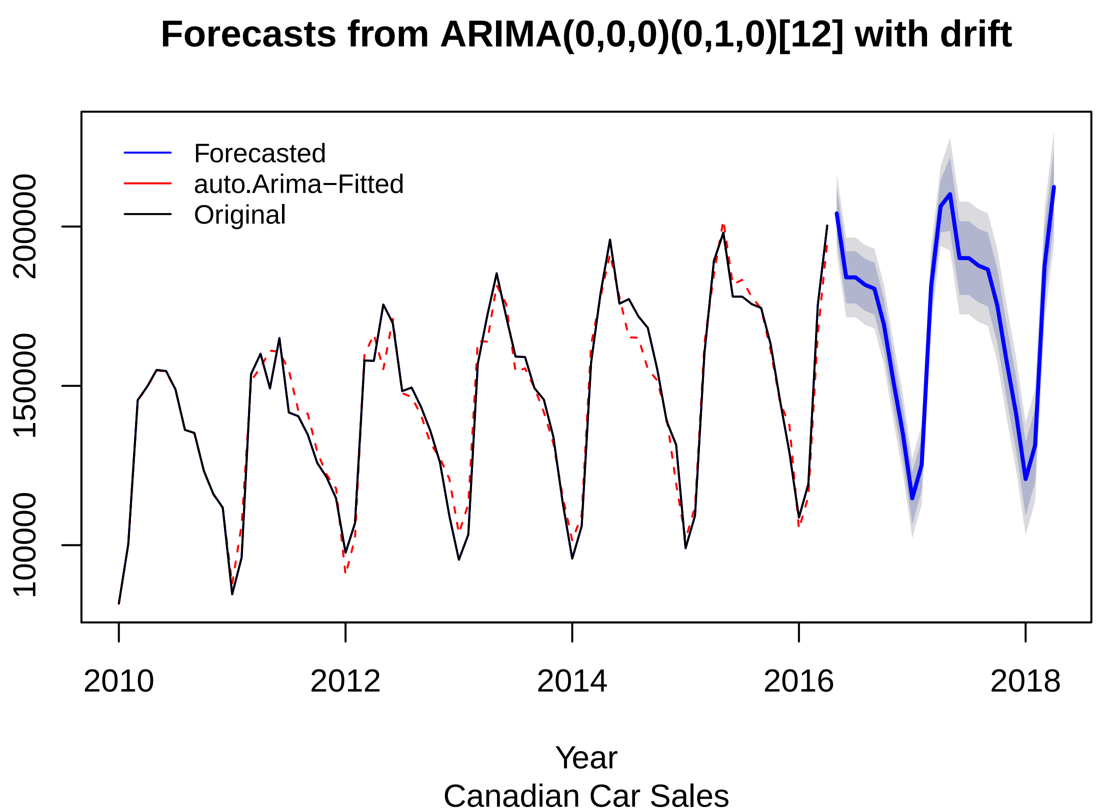


Arima forecast of the ts timeseries

```r
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 3.2.5

## Loading required package: zoo

##
## Attaching package: 'zoo'

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

## Loading required package: timeDate

## This is forecast 7.1
```

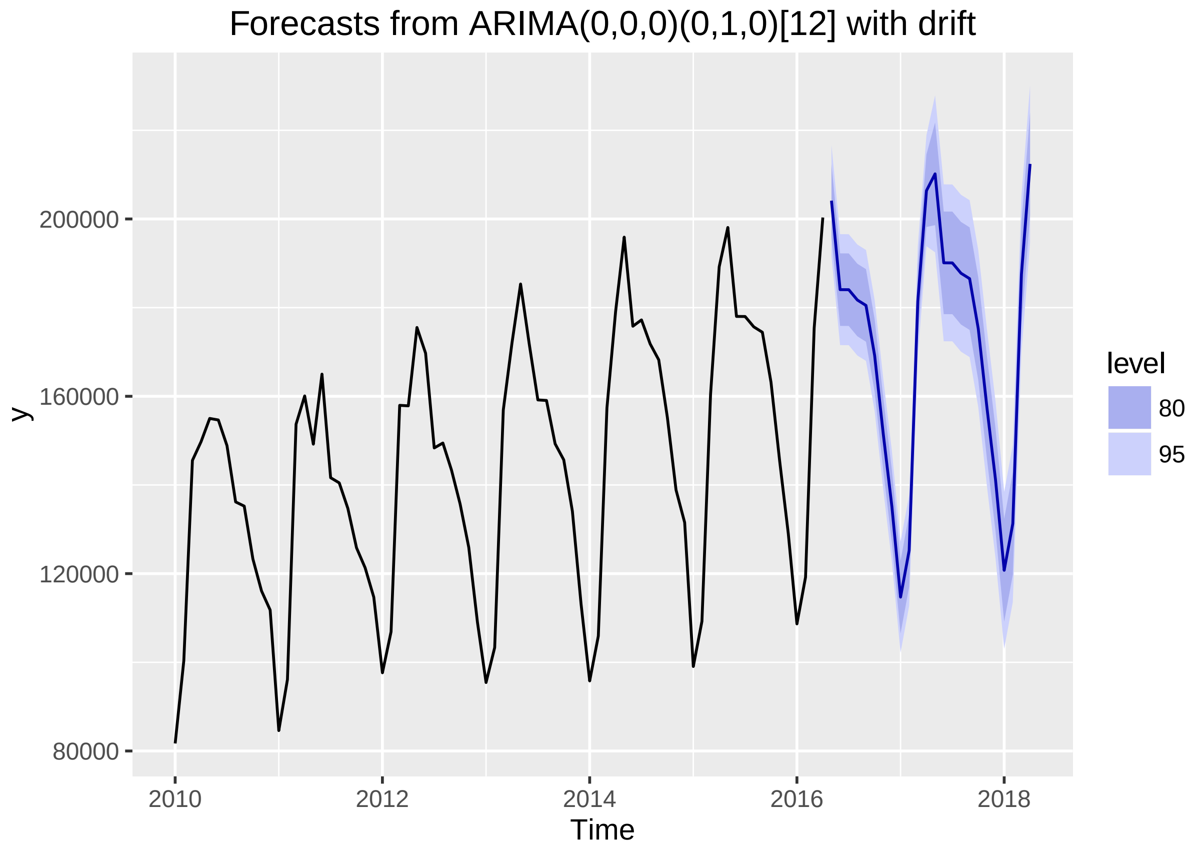
```
carsales.forecast = forecast.Arima(auto.arima(carsales.ts))

# Compare actual, auto.arima and forecast
plot.forecast(carsales.forecast, col="blue", xlab="Year", sub="Canadian Car Sales")
lines(carsales.forecast$fitted, col="red", lty=2)
lines(carsales.ts, col="black")
legend(
  'topleft', inset=.02,
  legend=c("Forecasted", "auto.Arima-Fitted", "Original"),
  col=c("blue", "red", "black"),
  lty=1, box.lty=0, cex=0.8
)
```




```
# another way to plot the forevast using ggplot2's autoplot()
library(ggplot2)
```

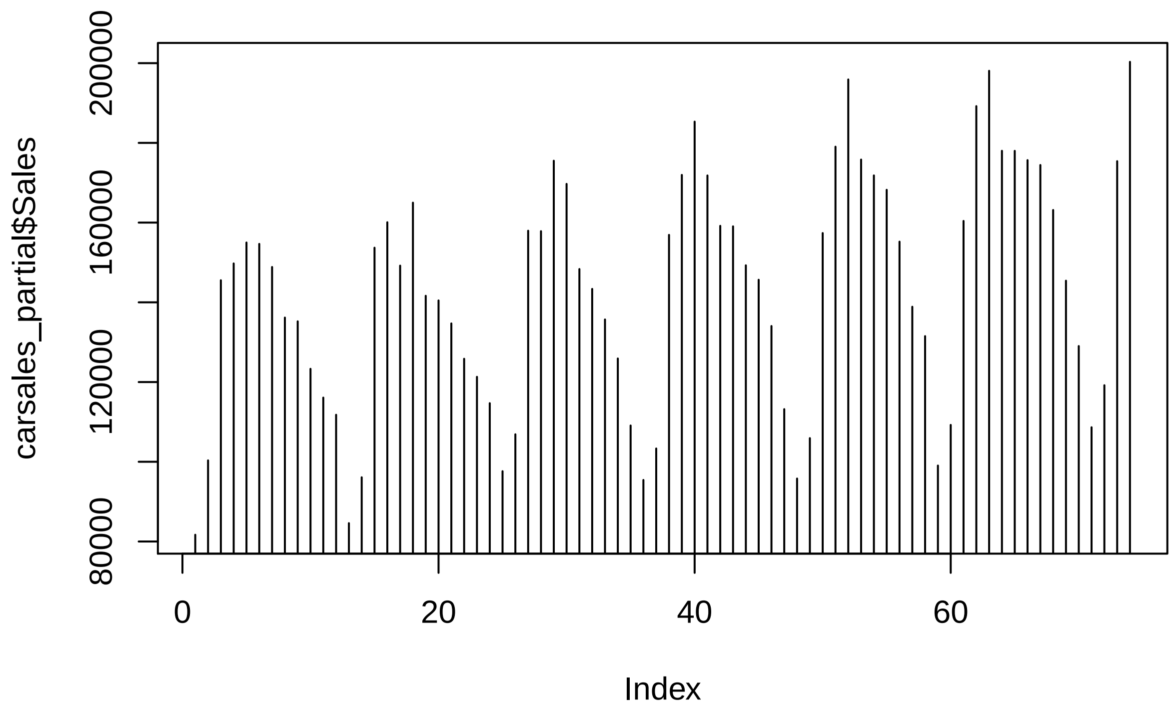
```
## Warning: package 'ggplot2' was built under R version 3.2.5
```

```
autoplot(carsales.forecast)
```



Create some chaos, delete sample # 32 and 55

```
carsales_partial = carsales[-c(32, 55),]
# We won't be able to tell the missing samples from a non-timeseries plot
plot(carsales_partial$Sales, type='h')
```



Create a timeseries using zoo() as ts() cannot be used to create a timeseries with missing samples

```
library(zoo)
# create a date column to be used as time index
carsales_partial$date = as.Date(
  paste0(carsales_partial$Year,'-', trimws(carsales_partial$Month), '-01'),
  format="%Y-%B-%d"
)
carsales_partial.z = zoo(
  carsales_partial$Sales,
  order.by=carsales_partial$date,
  frequency=12
)
str(carsales_partial.z)
```

```
## 'zooreg' series from 2010-01-01 to 2016-04-01
##   Data: num [1:74] 81693 100352 145539 149757 155008 ...
##   Index:  Date[1:74], format: "2010-01-01" "2010-02-01" "2010-03-01" "2010-04-01" ...
##   Frequency: 12
```

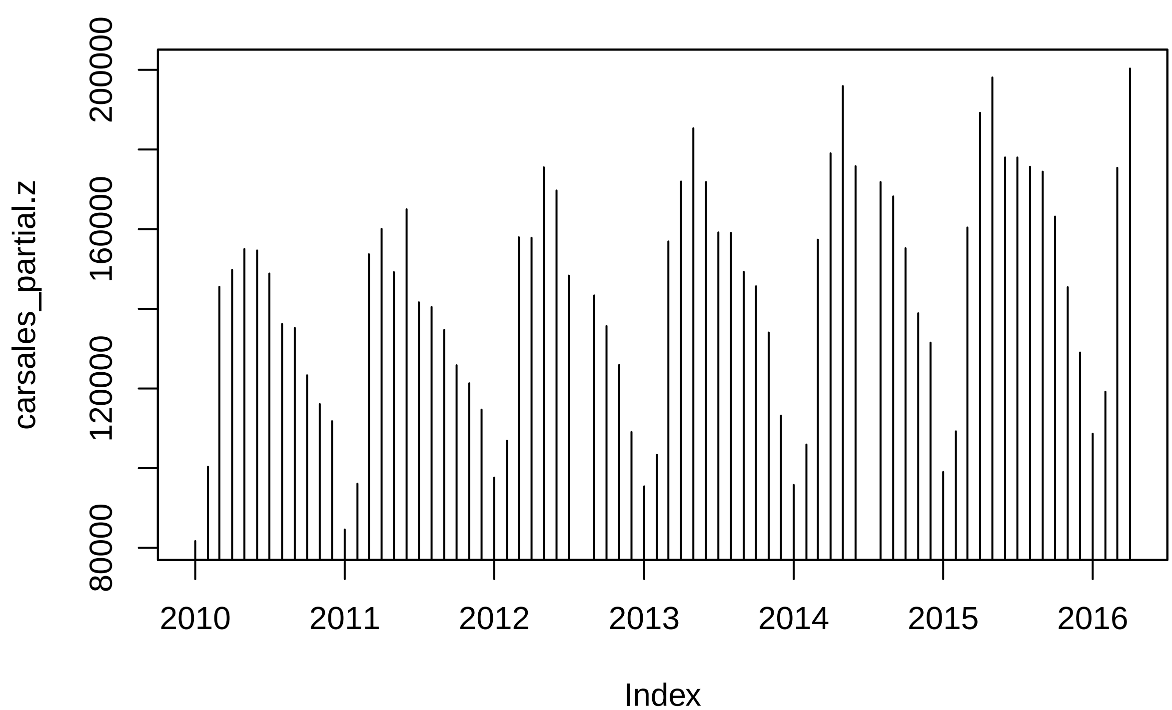
```r
# view the index portion
head(index(carsales_partial.z))
```

```
## [1] "2010-01-01" "2010-02-01" "2010-03-01" "2010-04-01" "2010-05-01"
## [6] "2010-06-01"
```

```r
# view the data value portion
head(coredata(carsales_partial.z))
```

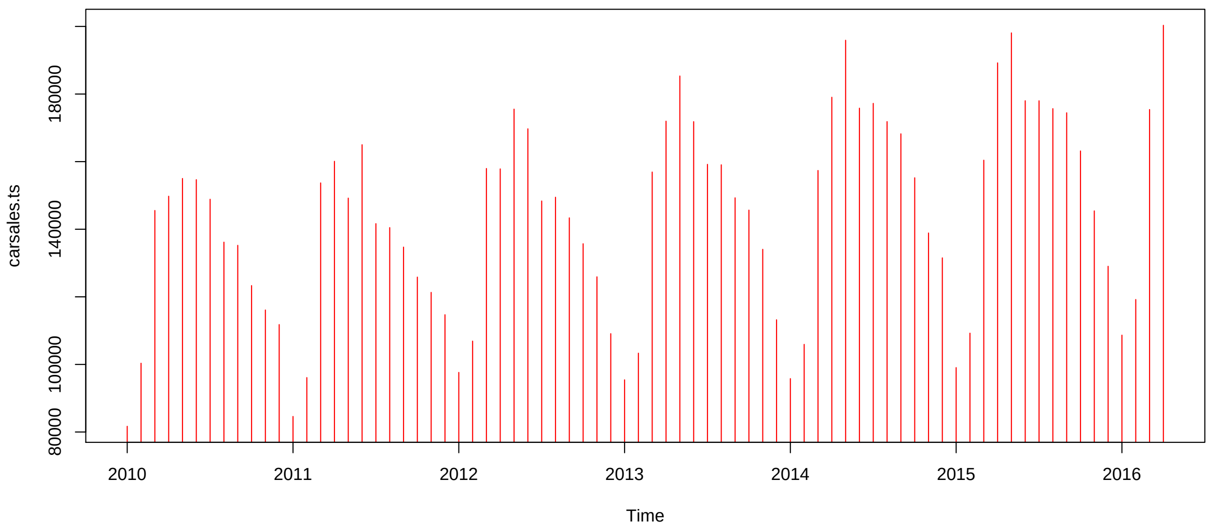
```
## [1]  81693 100352 145539 149757 155008 154656
```

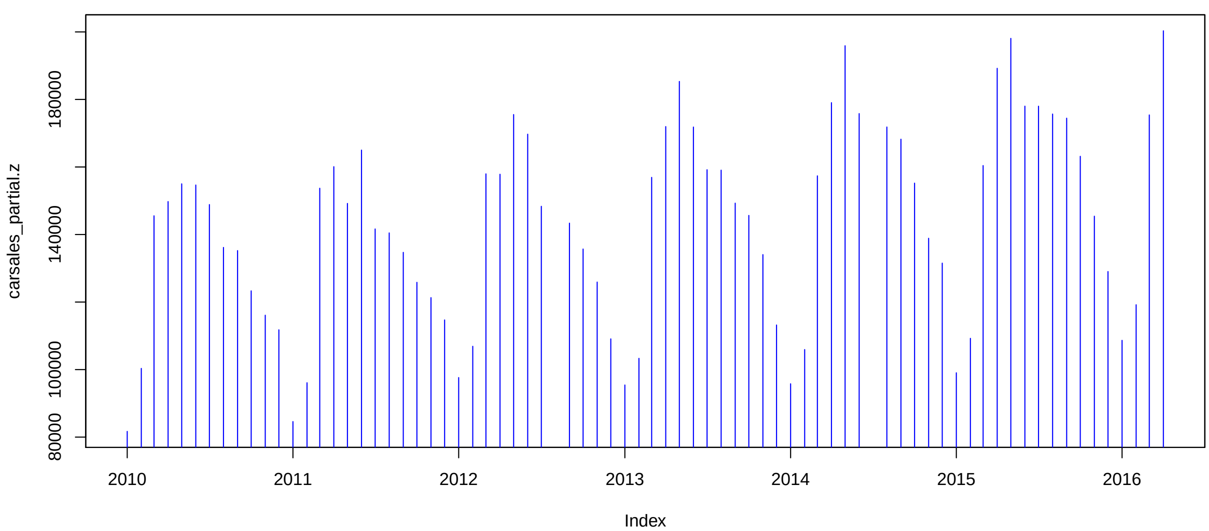
```r
plot(carsales_partial.z, type="h")
```




```r
# get date range from the timeseries
sub_text = range(index(carsales_partial.z))
```

```r
# compare with good timeseries
par(mfrow=c(2,1))
plot(carsales.ts, col="red", type='h')
plot(carsales_partial.z, col="blue", type='h')
```






```r
par(mfrow=c(1,1))
```

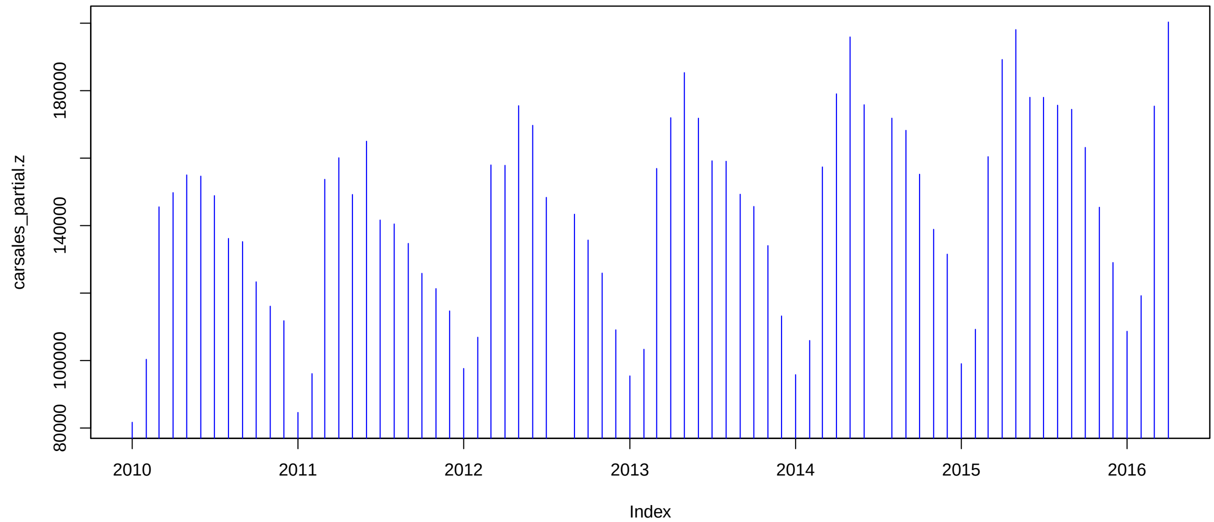
Fix the missing samples by imputing

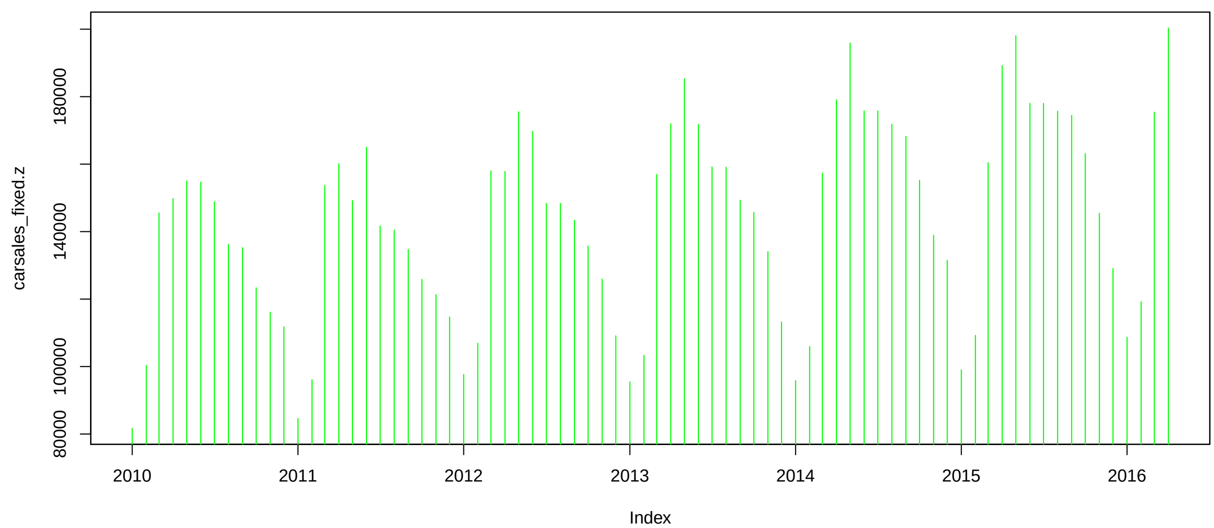
Replace the missing values with the na.locf() function (last observation carried forward)

Fixing is required for time series decomposition and Arima forecast.

```r
# generate a sequence of monthly intervals
g = seq(start(carsales_partial.z), end(carsales_partial.z), by="month")
carsales_fixed.z = na.locf(carsales_partial.z, xout=g)
```

```r
# compare the missing sample series with the fixed series
par(mfrow=c(2,1))
plot(carsales_partial.z, col="blue", type='h')
plot(carsales_fixed.z, col="green", type='h')
```






```r
par(mfrow=c(1,1))
carsales_fixed.z
```

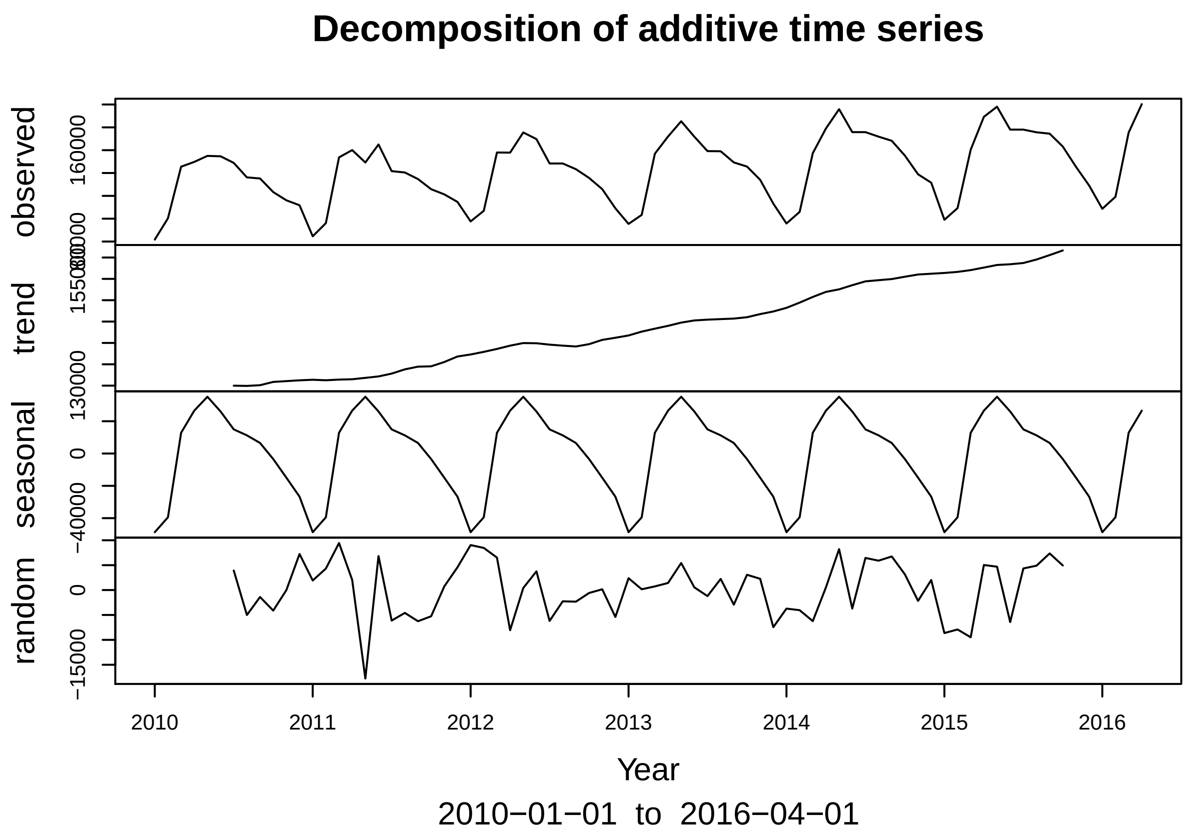
```
## 2010-01-01 2010-02-01 2010-03-01 2010-04-01 2010-05-01 2010-06-01
##      81693     100352     145539     149757     155008     154656
## 2010-07-01 2010-08-01 2010-09-01 2010-10-01 2010-11-01 2010-12-01
##     148865     136173     135232     123317     116100     111787
## 2011-01-01 2011-02-01 2011-03-01 2011-04-01 2011-05-01 2011-06-01
```

```
##       84599      96105     153700     160089     149203     165001
## 2011-07-01 2011-08-01 2011-09-01 2011-10-01 2011-11-01 2011-12-01
##     141641     140474     134708     125835     121306     114696
## 2012-01-01 2012-02-01 2012-03-01 2012-04-01 2012-05-01 2012-06-01
##      97635     106894     157962     157847     175525     169708
## 2012-07-01 2012-08-01 2012-09-01 2012-10-01 2012-11-01 2012-12-01
##     148350     148350     143363     135696     125912     109096
## 2013-01-01 2013-02-01 2013-03-01 2013-04-01 2013-05-01 2013-06-01
##      95434     103330     156918     171965     185332     171825
## 2013-07-01 2013-08-01 2013-09-01 2013-10-01 2013-11-01 2013-12-01
##     159186     159059     149287     145657     134052     113201
## 2014-01-01 2014-02-01 2014-03-01 2014-04-01 2014-05-01 2014-06-01
##      95796     105927     157373     179044     195905     175809
## 2014-07-01 2014-08-01 2014-09-01 2014-10-01 2014-11-01 2014-12-01
##     175809     171837     168224     155216     138886     131520
## 2015-01-01 2015-02-01 2015-03-01 2015-04-01 2015-05-01 2015-06-01
##      99051     109248     160416     189216     198084     178013
## 2015-07-01 2015-08-01 2015-09-01 2015-10-01 2015-11-01 2015-12-01
##     177999     175670     174447     163157     145426     129031
## 2016-01-01 2016-02-01 2016-03-01 2016-04-01
##     108660     119201     175407     200327
```

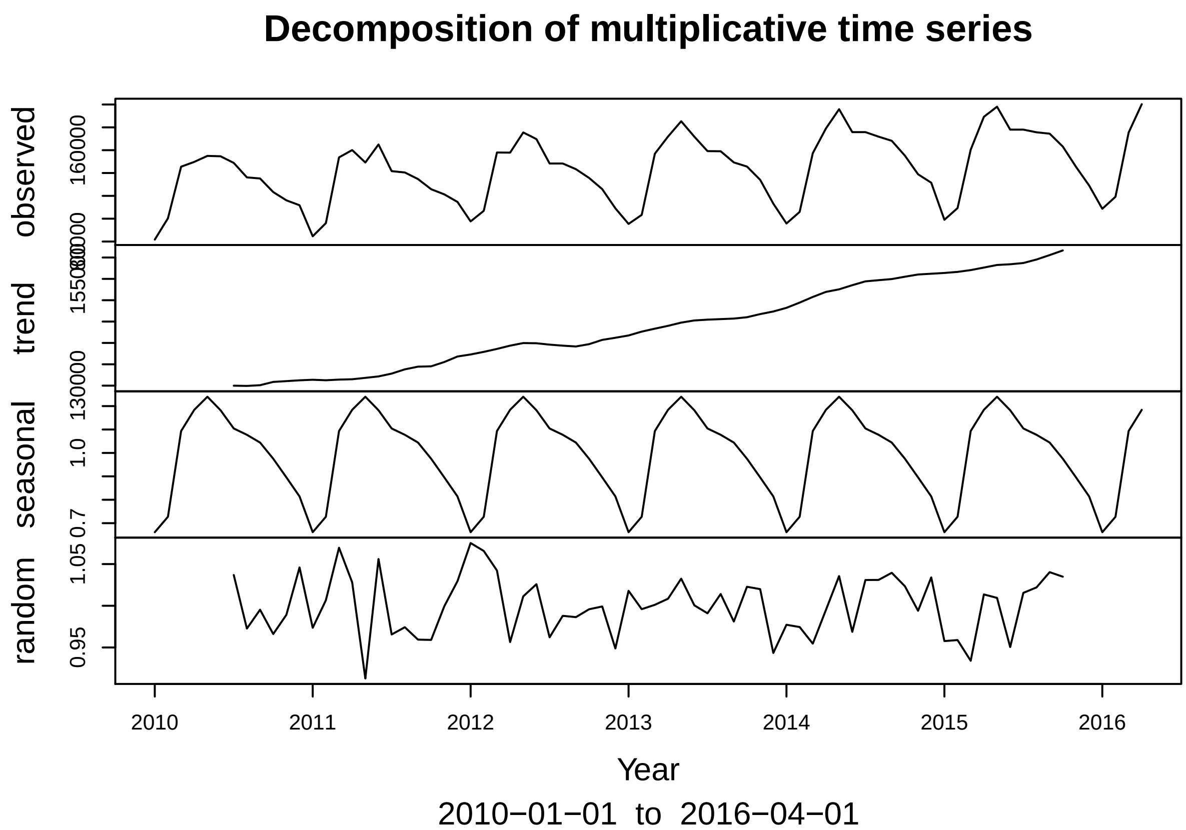
Timeseries decomposition with decompose() of the fixed zoo object

We will try additive and then multiplicative decompose

```
carsales_fixed.decompose = decompose(
  ts(
    carsales_fixed.z,
    frequency = 12,
    start=c(2010,1) # we cheat by hard coding the value here
  ),
  "additive"
)
#
plot(carsales_fixed.decompose, xlab="Year")
title(sub=paste(sub_text,collapse="  to  "))
```

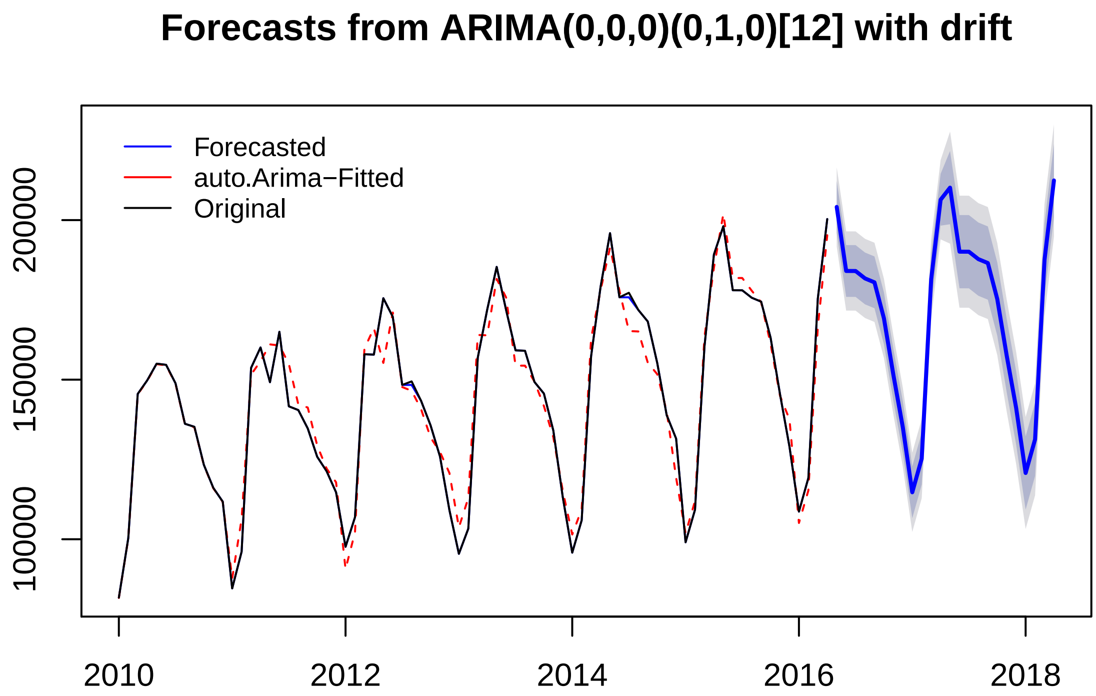



```
carsales_fixed.decompose = decompose(
  ts(
    carsales_fixed.z,
    frequency = 12,
    start=c(2010,1) # we cheat by hard coding the value here
  ),
  "multiplicative"
)
#
plot(carsales_fixed.decompose, xlab="Year")
title(sub=paste(sub_text,collapse="  to  "))
```

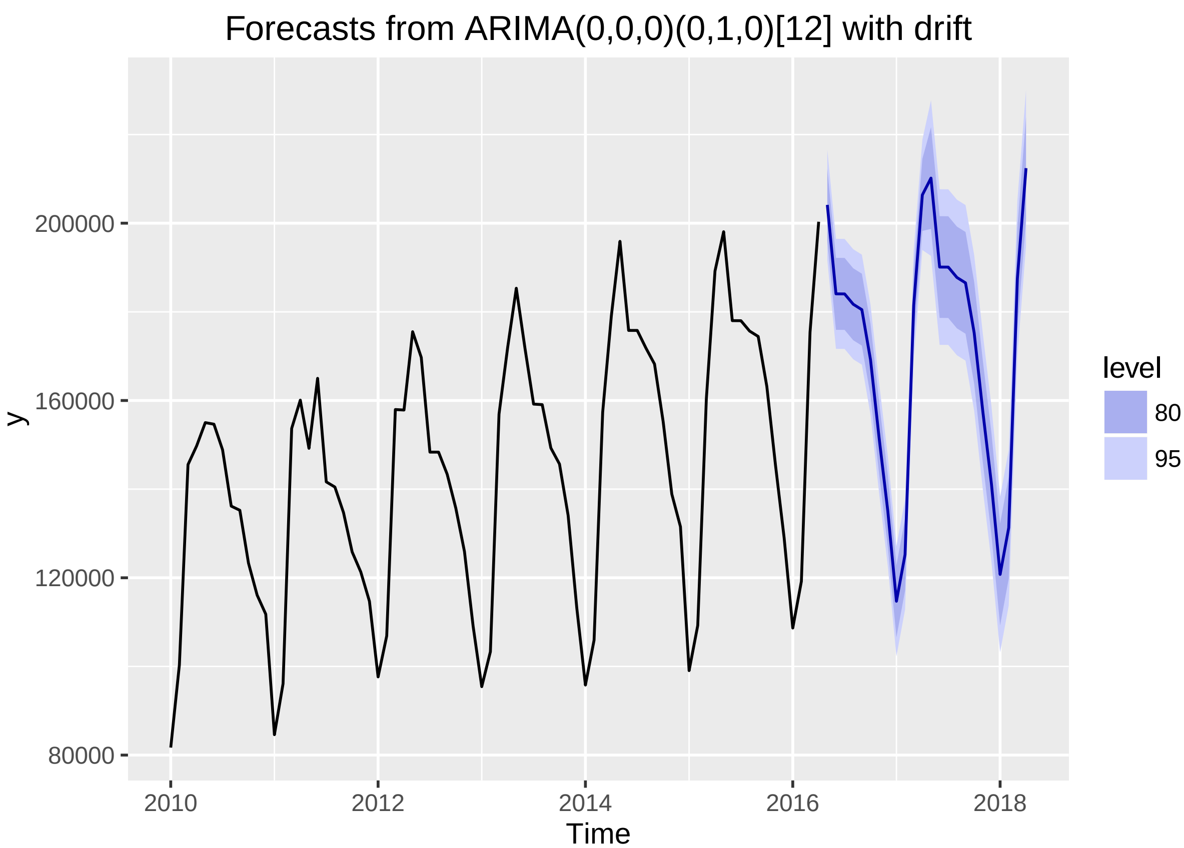


Arima forecast

```r
carsales_fixed.z.forecast = forecast.Arima(
  auto.arima(
    ts(
      carsales_fixed.z,
      frequency = 12,
      start=c(2010,1) # we cheat by hard coding the value here
    )
  )
)
# Compare actual, auto.arima and forecast
plot.forecast(carsales_fixed.z.forecast, col="blue")
lines(carsales_fixed.z.forecast$fitted, col="red", lty=2)
lines(carsales.ts, col="black")
legend(
  'topleft', inset=.02,
  legend=c("Forecasted", "auto.Arima-Fitted", "Original"),
  col=c("blue", "red", "black"),
  lty=1, box.lty=0, cex=0.8
)
```

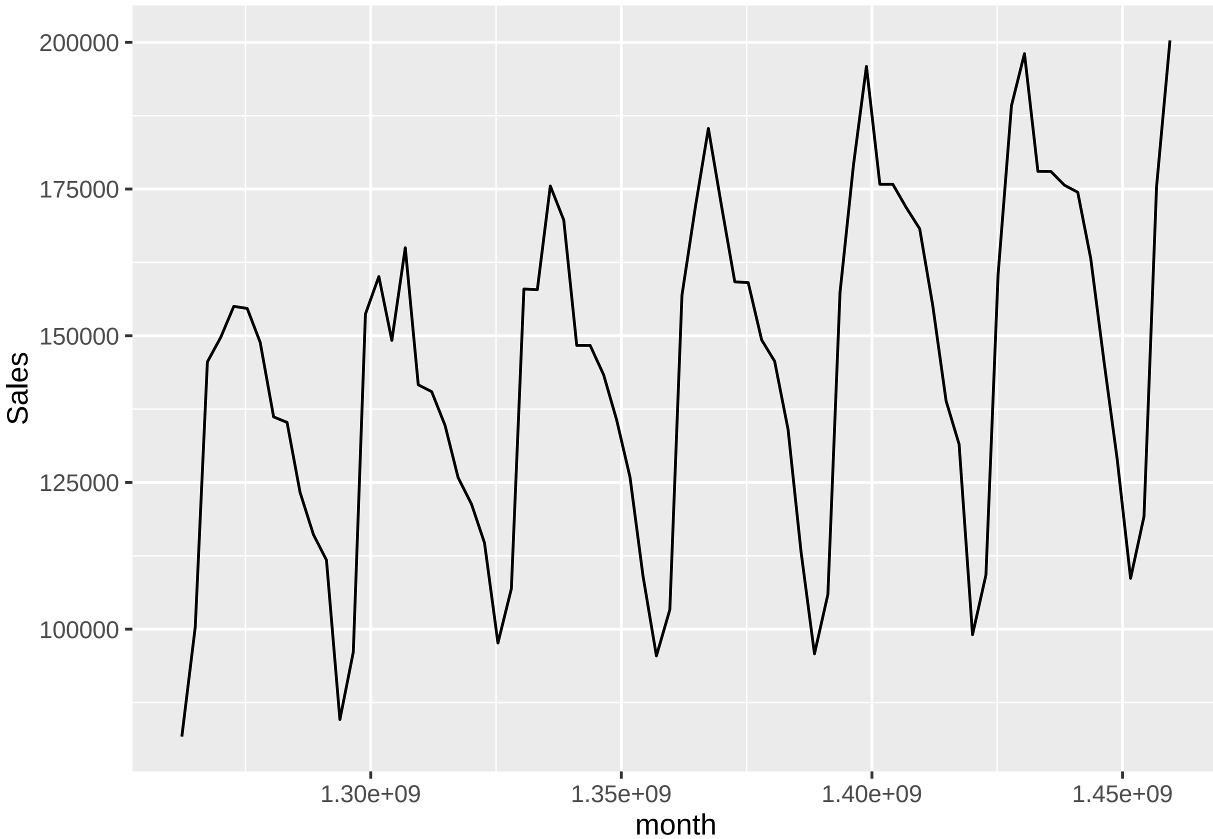



```
# Another way to plot arima forecast with autoplot()
library(ggplot2)
autoplot(carsales_fixed.z.forecast)
```



Anomaly dection (None detected)

```
library(AnomalyDetection)

myts = as.data.frame(
  cbind(
    as.POSIXct(index(carsales_fixed.z)),
    coredata(carsales_fixed.z)
  )
)
colnames(myts) = c("month", "Sales")
attr(myts$month, "tzone") = "UTC"

ggplot(myts,
       aes(x=month, y=Sales)
) +
  geom_line()
```




```
data_anomaly = AnomalyDetectionTs(myts, max_anoms=0.01, direction="pos", plot=F, e_value = T, na.rm = T)
```

```r
# No anomaly detected as NULL result returned
data_anomaly
```

```
## $anoms
## data frame with 0 columns and 0 rows
##
## $plot
## NULL
```

```r
data_anomaly$plot
```

```
## NULL
```

Conditions for timeseries

http://www.statosphere.com.au/check-time-series-stationary-r/

```r
# Compute the Box-Pierce or Ljung-Box test statistic for examining the null hypothesis of independence
Box.test(carsales.ts)
```

```
##
##  Box-Pierce test
##
## data:  carsales.ts
## X-squared = 36.895, df = 1, p-value = 1.247e-09
```

```r
Box.test(carsales.ts, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  carsales.ts
## X-squared = 38.371, df = 1, p-value = 5.85e-10
```

```r
# The Augmented Dickey-Fuller (ADF) t-statistic test: small p-values suggest the data is stationary and
library(tseries)
adf.test(carsales.ts)
```

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

```
##
##  Augmented Dickey-Fuller Test
##
## data:  carsales.ts
## Dickey-Fuller = -5.2851, Lag order = 4, p-value = 0.01
## alternative hypothesis: stationary
```

```r
# The Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test; here accepting the null hypothesis means that the
kpss.test(carsales.ts)
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  carsales.ts
## KPSS Level = 0.49462, Truncation lag parameter = 2, p-value =
## 0.04288
```