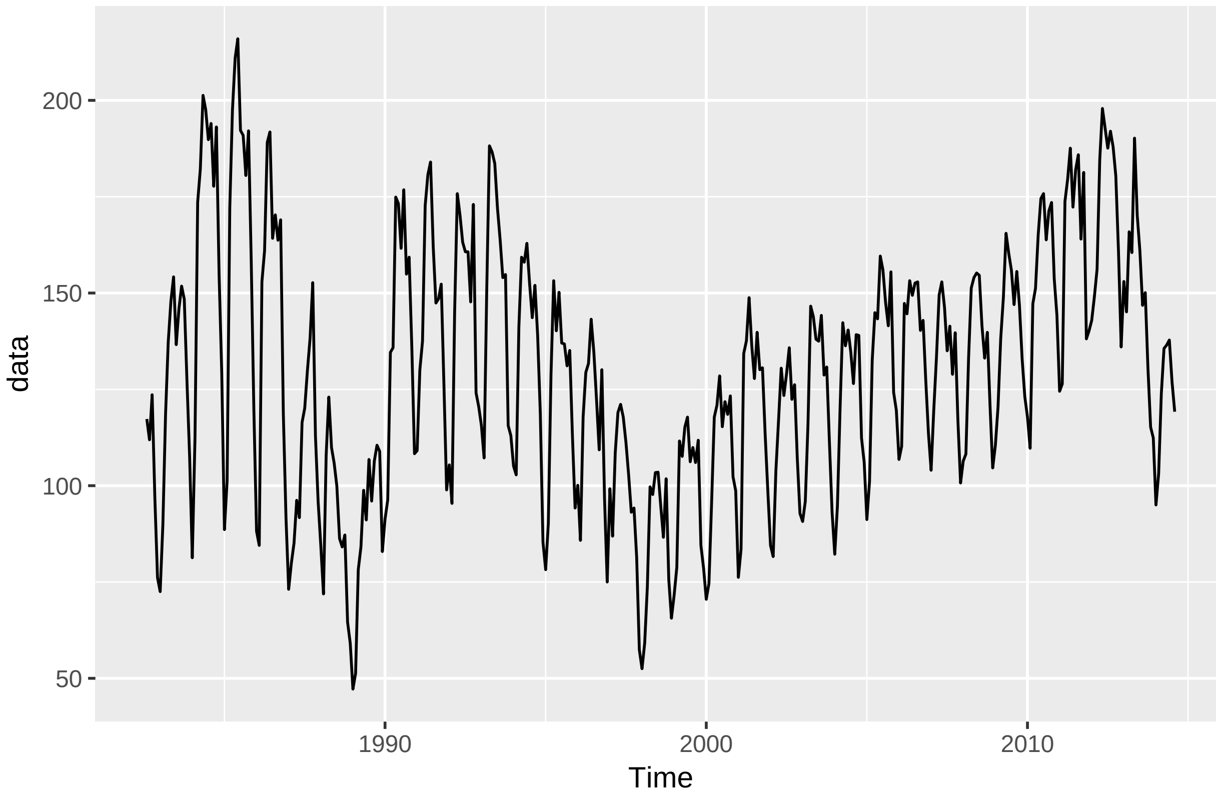
2 AutoRegression

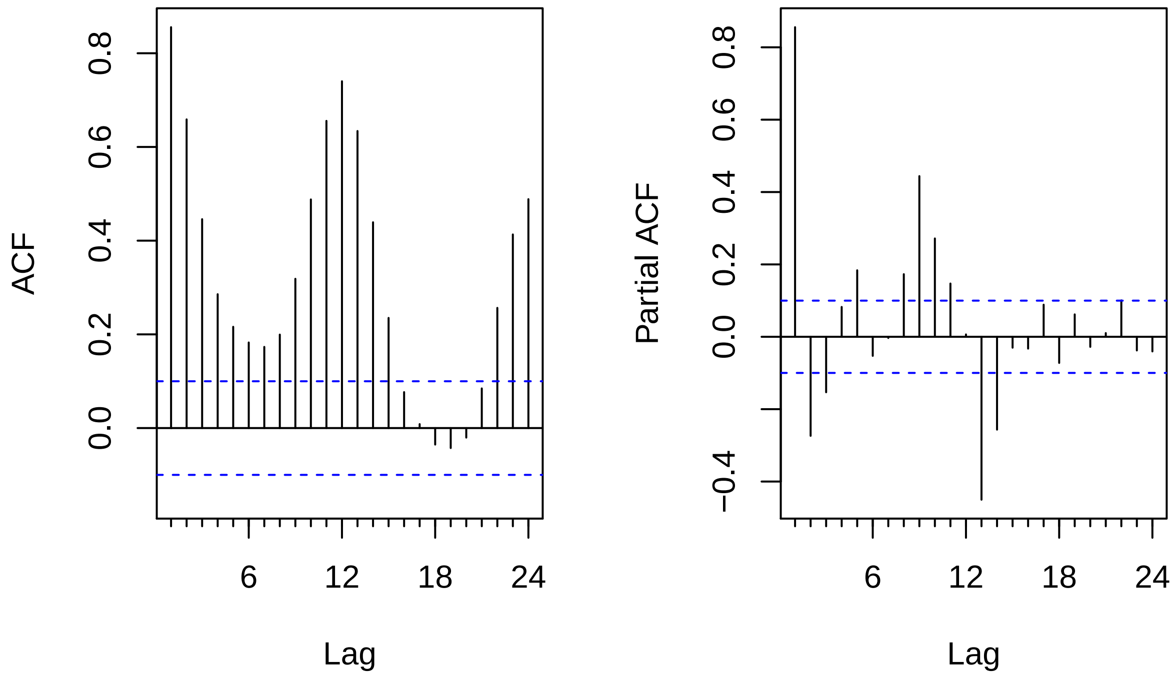
```r
library(forecast)
library(ggplot2)
```

```r
df = read.csv("data.csv")
data  = ts(df$Private.Housing.Starts,start = c(1982,8),frequency = 12)
autoplot(data)
```



1. Autocorrelations at different lags

```r
par(mfrow=c(1,2))
Acf(data, lag.max = 24, main = "")
Pacf(data, lag.max = 24, main = "")
```



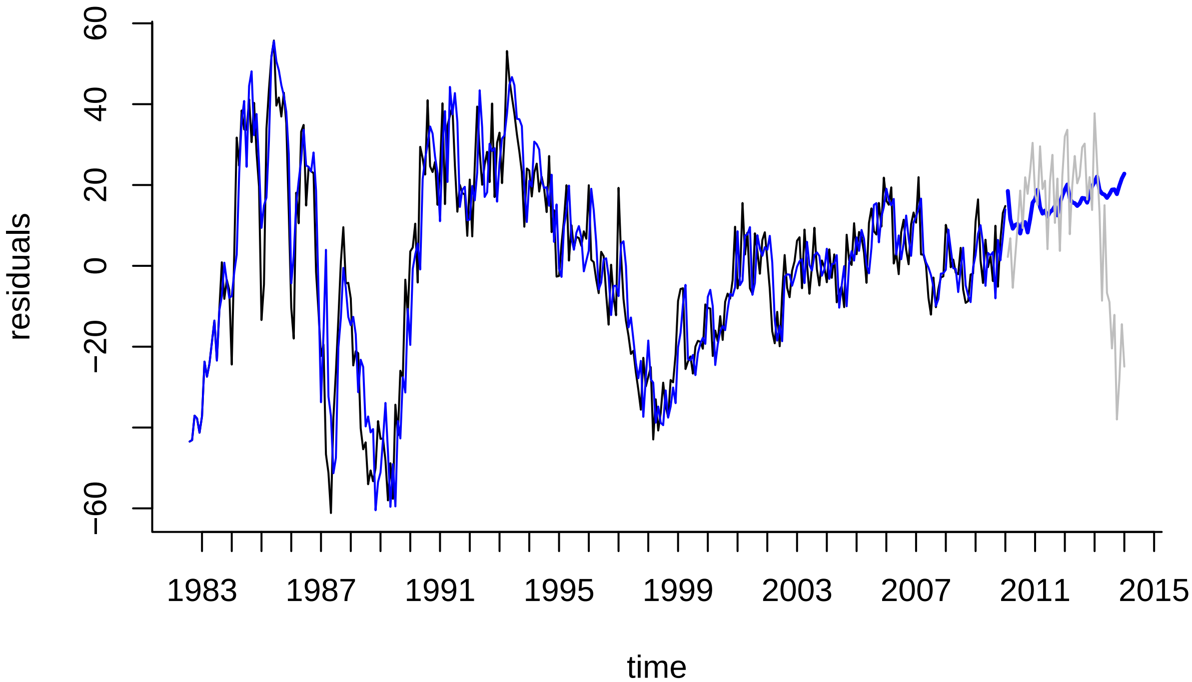
1.1 Autoregressive (AR) Models - Second-layer Model:

Use residuals to build the AR model and then add it to the series forecast

```
nvalid = 48
ntrain = length(data) - nvalid
train.ts = window(data, start = c(1982,08), end = c(1982, ntrain))
valid.ts = window(data, start = c(1982, ntrain+1), end = c(1982, ntrain+nvalid))
```

Use trend and season to build the predict model and then use arima(1,0,0) to predict the residual.

```
train.lm.trend.season = tslm(train.ts ~ trend +I(trend^2) + season)
train.lm.trend.season.pred = forecast(train.lm.trend.season, h = nvalid, level = 0)
train.res.arima = Arima(train.lm.trend.season$residuals, order = c(10,1,1), seasonal = c(1,1,1))
# seasonal = c(1,0,0), P =1, consider the lag-12
train.res.arima.pred = forecast(train.res.arima, h = nvalid, level = 0 )
residuals_valid = valid.ts-train.lm.trend.season.pred$mean

plot(train.res.arima.pred, ylab = "residuals", xlab = "time", bty = "l", xaxt = "n", main = "", col = 'l
axis(1,at = seq(1983, 2015,1), labels = format(seq(1983, 2015,1)))
lines(train.res.arima$fitted, lwd = 1, col = 'blue')
lines(residuals_valid, col = 'grey')
```

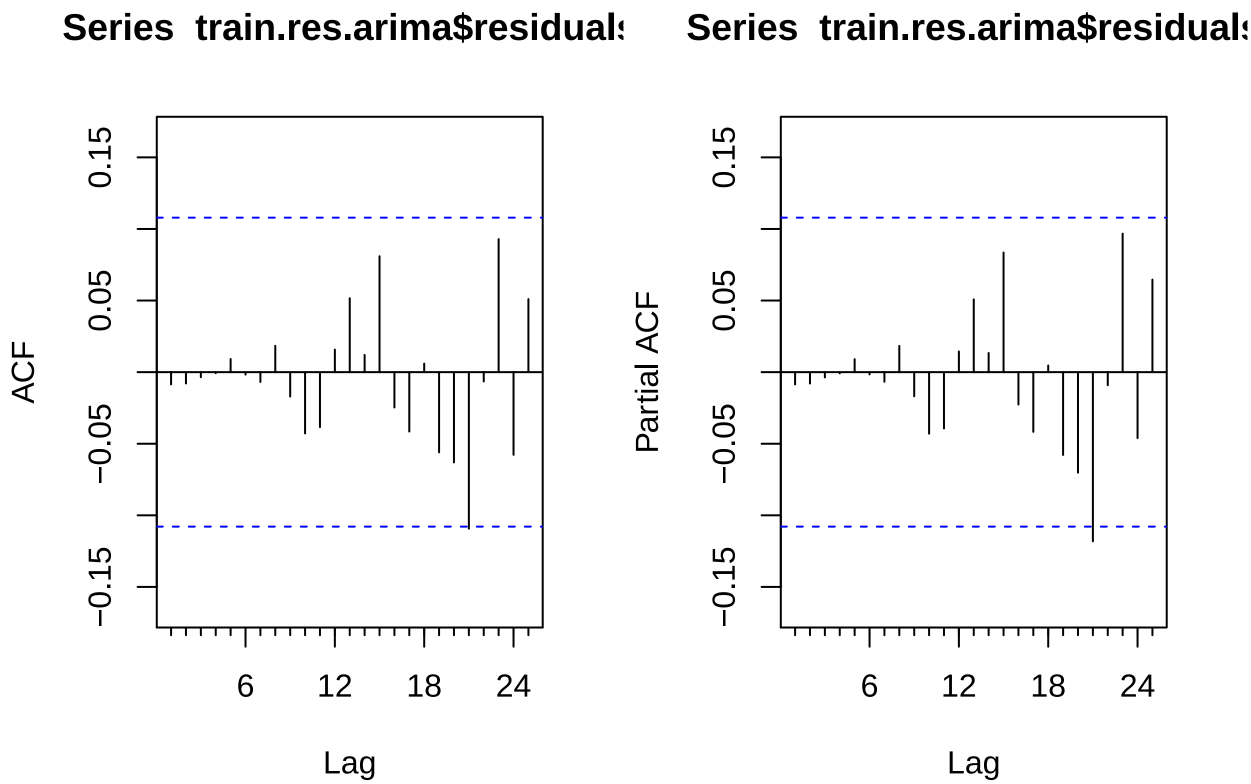



```
summary(train.res.arima)
```

```
## Series: train.lm.trend.season$residuals
## ARIMA(10,1,1)(1,1,1)[12]
##
## Coefficients:
##           ar1      ar2      ar3      ar4      ar5      ar6      ar7
##       -1.0071  -0.2422  -0.0154  -0.0973  -0.1609  -0.0378  -0.0209
## s.e.   0.1614   0.0895   0.0803   0.0814   0.0822   0.0812   0.0810
##           ar8      ar9     ar10     ma1     sar1     sma1
##       -0.1242  -0.0954  -0.1153  0.7342  0.2745  -0.9305
## s.e.   0.0819   0.0812   0.0627  0.1557  0.0724   0.0504
##
## sigma^2 estimated as 109.8:  log likelihood=-1197.46
## AIC=2422.92   AICc=2424.31   BIC=2475.55
##
## Training set error measures:
##                        ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -0.5853517 10.05657 7.628267 14.74761 143.9852 0.4549521
##                      ACF1
## Training set -0.008561735
```

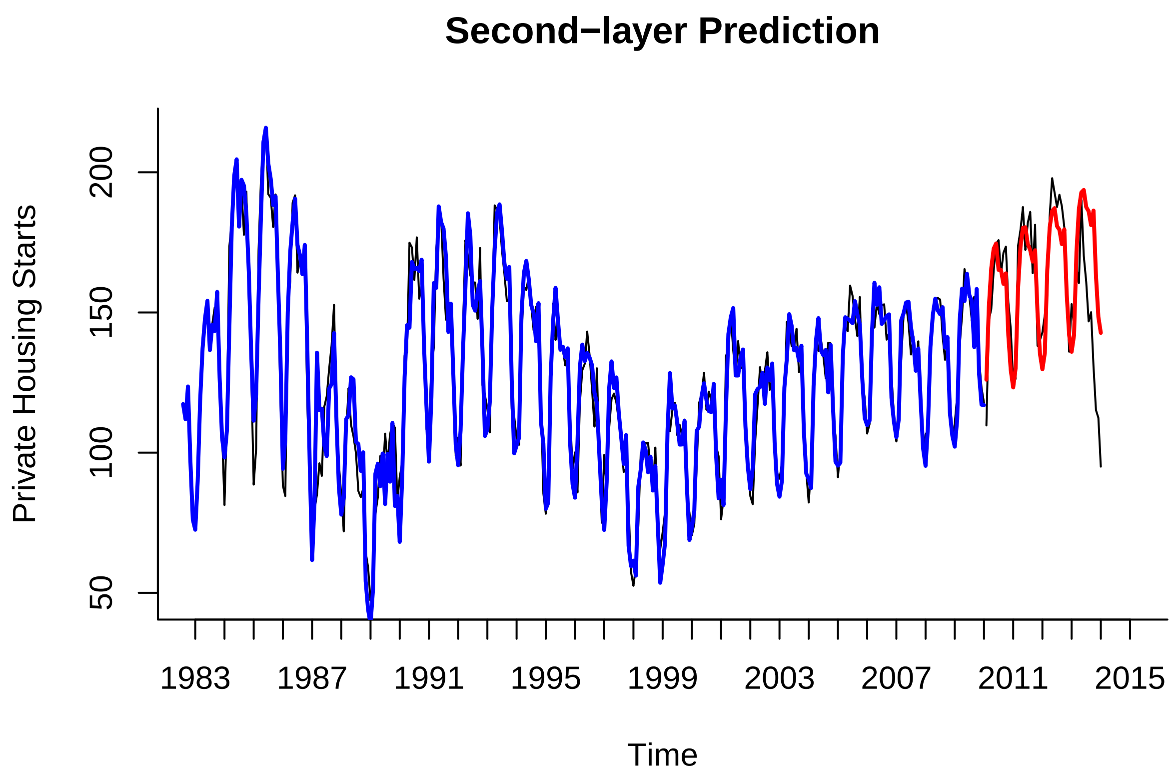
Have a look of the residuals of residuals, to make sure that there is no pattern left

```
par(mfrow = c(1,2))
Acf(train.res.arima$residuals)
Pacf(train.res.arima$residuals)
```



Combine residuals prediction with forecast model together.

```
fitted_train = train.lm.trend.season.pred$fitted + train.res.arima.pred$fitted
fitted_valid = train.lm.trend.season.pred$mean + train.res.arima.pred$mean

plot(train.ts, bty = "l", xaxt = "n", main = "Second-layer Prediction", xlim = c(1983, 2015), ylab = "P
axis(1,at = seq(1983, 2015,1), labels = format(seq(1983, 2015,1)))
lines(valid.ts)
lines(fitted_train,lwd = 2, col = "blue")
lines(fitted_valid,lwd = 2, col = "red")
```




```r
accuracy(train.lm.trend.season.pred, valid.ts)
```

```
##                        ME     RMSE      MAE       MPE     MAPE     MASE
## Training set 1.055385e-15 22.92443 17.86421 -3.726880 15.62442 1.057231
## Test set     1.155323e+01 20.76060 18.54536  6.379762 11.92569 1.097542
##                   ACF1 Theil's U
## Training set 0.8847026        NA
## Test set     0.6743047  1.172525
```

Testing RMSE for the second-layer model, improve a lot

```r
(sum((fitted_valid - valid.ts)**2)/nvalid)**0.5
```

```
## [1] 18.83108
```

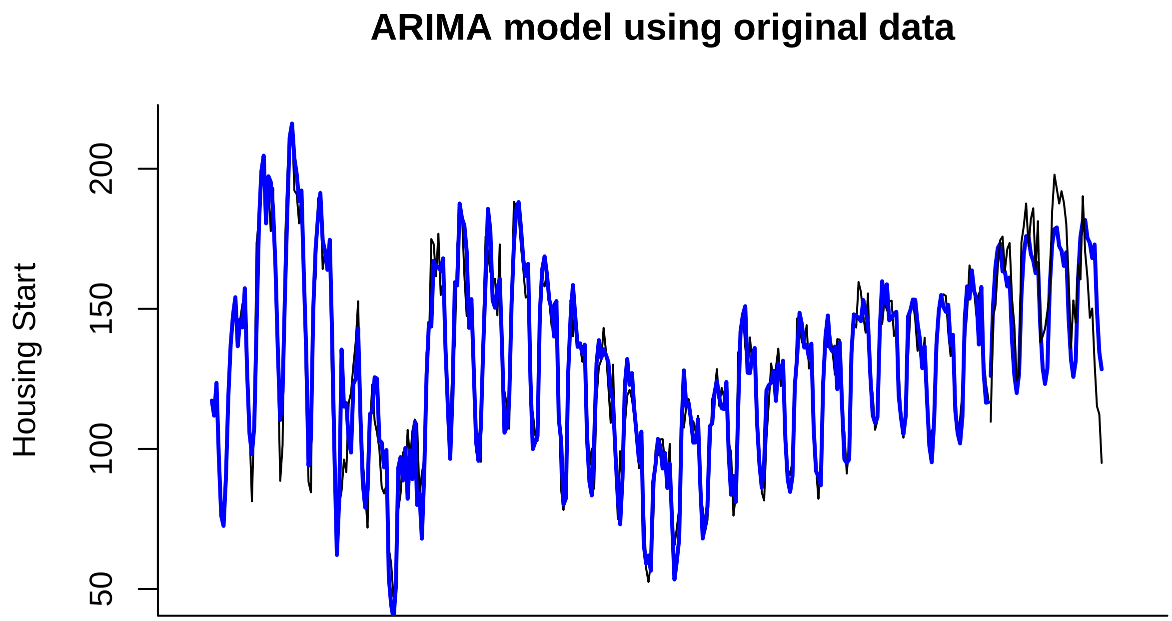
1.2 ARIMA Models

```r
train.arima = Arima(train.ts, order = c(10,1,2), seasonal = c(1,1,1))
summary(train.arima)
```

```
## Series: train.ts
## ARIMA(10,1,2)(1,1,1)[12]
##
## Coefficients:
##           ar1     ar2     ar3      ar4      ar5     ar6      ar7      ar8
##       -0.6744  0.1395  0.0742  -0.0931  -0.1306  0.0146  -0.0097  -0.1215
## s.e.   0.2850  0.3218  0.1043   0.0691   0.0750  0.0816   0.0692   0.0704
##           ar9    ar10     ma1      ma2    sar1     sma1
##       -0.0642 -0.1156  0.3960  -0.2974  0.2700  -0.9292
## s.e.   0.0821  0.0642  0.2858   0.2544  0.0729   0.0497
```

```
##
## sigma^2 estimated as 109.8:  log likelihood=-1196.93
## AIC=2423.86   AICc=2425.46   BIC=2480.25
##
## Training set error measures:
##                      ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -0.4278077 10.04128 7.594941 -0.5288352 6.628711 0.4494801
##                     ACF1
## Training set -0.00402262
```

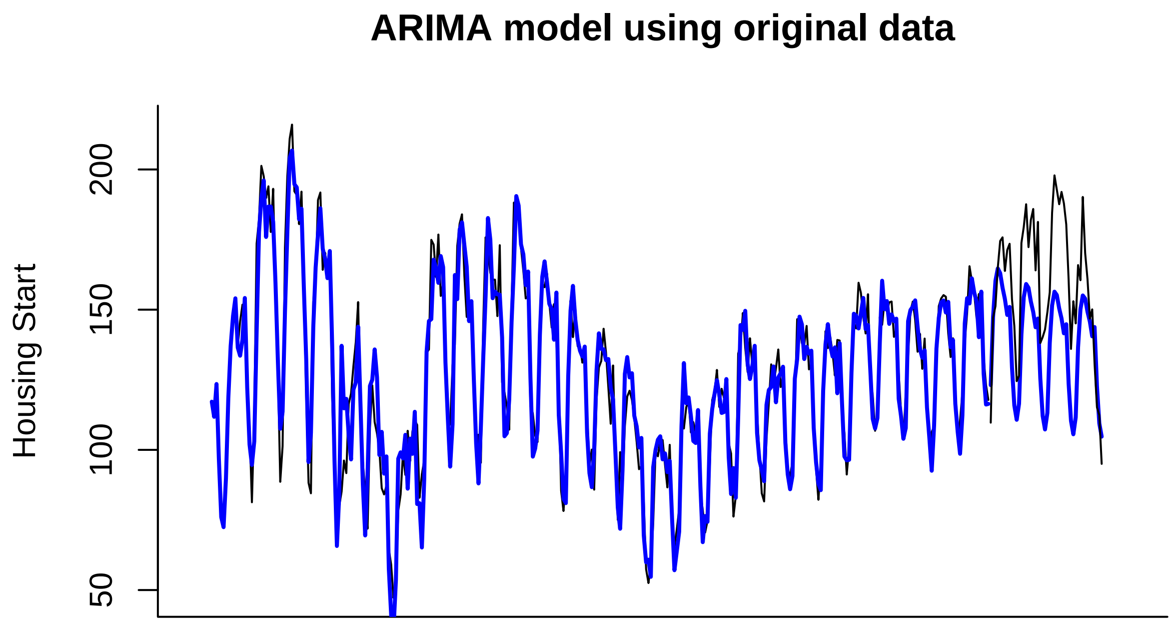
```
train.arima.pred = forecast(train.arima, h = nvalid, level = 0)
plot(train.arima.pred, main = "ARIMA model using original data", bty = "l", xlim = c(1982,2015), ylab =
lines(valid.ts)
lines(train.arima.pred$fitted, col = "blue", lwd = 2)
```




```
train.auto = auto.arima(train.ts)
summary(train.auto)
```

```
## Series: train.ts
## ARIMA(2,0,0)(0,1,1)[12]
##
## Coefficients:
##          ar1     ar2     sma1
##       0.7058  0.2234  -0.8252
## s.e.  0.0551  0.0553   0.0446
##
## sigma^2 estimated as 115.3:  log likelihood=-1211.86
## AIC=2431.72   AICc=2431.85   BIC=2446.77
##
## Training set error measures:
##                     ME     RMSE   MAE        MPE     MAPE      MASE
## Training set 0.2086102 10.49191 7.944 -0.2789156 6.914446 0.4701379
##                      ACF1
## Training set -0.006179956
```

```
train.auto.pred = forecast(train.auto, h = nvalid, level = 0)
plot(train.auto.pred, main = "ARIMA model using original data", bty = "l", xlim = c(1982,2015), ylab =
lines(valid.ts)
lines(train.auto.pred$fitted, col = "blue", lwd = 2)
```



2 Use external data (Weather, temperature, etc.)

2.1 linear regression

```
library(lubridate)
```

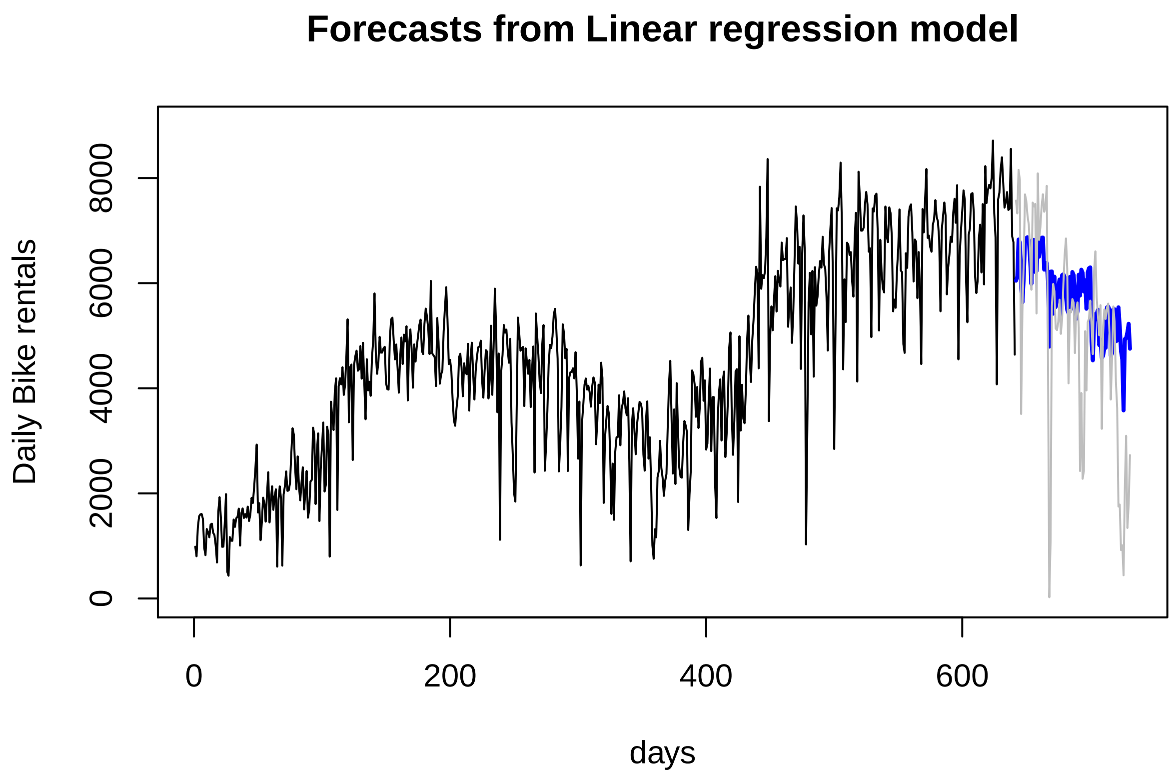
```
##
## Attaching package: 'lubridate'

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

```
bike.df = read.csv("BikeSharingDaily.csv")
bike.df$Date =  as.Date(bike.df$dteday, format = "%Y-%m-%d")
bike.df$Month = month(bike.df$Date, label = T)
bike.df$DOW = wday(bike.df$Date, label = T)
bike.df$WorkingDay = factor(bike.df$workingday, levels = c(0,1), labels = c("not_working","working"))
bike.df$Weather = factor(bike.df$weathersit, levels = c(1,2,3), labels = c("clear","mist","rain_snow"))
```

```
month.dummies = model.matrix(~0+Month, data = bike.df)
dow.dummies = model.matrix(~0 + DOW, data = bike.df)
workingday_weather.dummies = model.matrix(~0 +WorkingDay:Weather, data = bike.df)
colnames(month.dummies) = gsub("Month", "", colnames(month.dummies))
colnames(dow.dummies) = gsub("DOW", "", colnames(dow.dummies))
colnames(workingday_weather.dummies) = gsub("WorkingDay","", colnames(workingday_weather.dummies))
colnames(workingday_weather.dummies) = gsub("Weather","", colnames(workingday_weather.dummies))
colnames(workingday_weather.dummies) = gsub(":","", colnames(workingday_weather.dummies))
```

```r
x = as.data.frame(cbind(month.dummies[,-12],dow.dummies[,-7], workingday_weather.dummies[,-6]))
y = bike.df$cnt
ntotal = length(y)
nvalid = 90
ntrain = ntotal-nvalid
xtrain = x[1:ntrain,]
ytrain = y[1:ntrain]
xvalid = x[(ntrain+1):ntotal,]
yvalid = y[(ntrain+1):ntotal]
```

```r
ytrain.ts = ts(ytrain)
formula = as.formula(paste("ytrain.ts", paste(c("trend",colnames(xtrain)), collapse = "+"), sep = "~"))
bike.tslm = tslm(formula, data = xtrain, lambda = 1)
bike.tslm.pred = forecast(bike.tslm, newdata = xvalid, level = 0)
plot(bike.tslm.pred, ylim = c(0,9000), xlab = "days", ylab = "Daily Bike rentals")
lines(ts(yvalid, start = ntrain+1), col = "grey")
```




```r
summary(bike.tslm)
```

```
##
## Call:
## tslm(formula = formula, data = xtrain, lambda = 1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3196.7  -390.4    46.7   450.3  3742.9
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)             -483.4300   330.2092  -1.464 0.143700
```

```
## trend                      6.2764     0.1801  34.842  < 2e-16 ***
## Jan                      296.6630   172.9395   1.715 0.086771 .
## Feb                      542.6007   174.9410   3.102 0.002012 **
## Mar                     1423.1126   171.9153   8.278 7.77e-16 ***
## Apr                     2075.0033   172.8791  12.003  < 2e-16 ***
## May                     2682.1939   171.0316  15.682  < 2e-16 ***
## Jun                     2780.7568   171.9511  16.172  < 2e-16 ***
## Jul                     2406.3115   171.5136  14.030  < 2e-16 ***
## Aug                     2322.2856   172.1182  13.492  < 2e-16 ***
## Sep                     2418.4101   172.1933  14.045  < 2e-16 ***
## Oct                     1715.8447   194.0877   8.841  < 2e-16 ***
## Nov                      833.7700   198.4463   4.201 3.04e-05 ***
## Sun                     -364.1915   114.8201  -3.172 0.001590 **
## Mon                     -632.8539   207.4100  -3.051 0.002377 **
## Tue                     -492.6959   231.5251  -2.128 0.033729 *
## Wed                     -495.5486   229.8076  -2.156 0.031441 *
## Thu                     -422.8810   229.5692  -1.842 0.065946 .
## Fri                     -413.2995   227.7813  -1.814 0.070093 .
## not_workingclear        1492.1283   293.0865   5.091 4.73e-07 ***
## workingclear            1969.2146   219.6669   8.965  < 2e-16 ***
## not_workingmist          971.7656   306.8881   3.167 0.001619 **
## workingmist             1279.9223   222.8292   5.744 1.45e-08 ***
## not_workingrain_snow -1728.3534   461.5525  -3.745 0.000198 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 773.2 on 617 degrees of freedom
## Multiple R-squared:  0.8437, Adjusted R-squared:  0.8379
## F-statistic: 144.8 on 23 and 617 DF,  p-value: < 2.2e-16
```

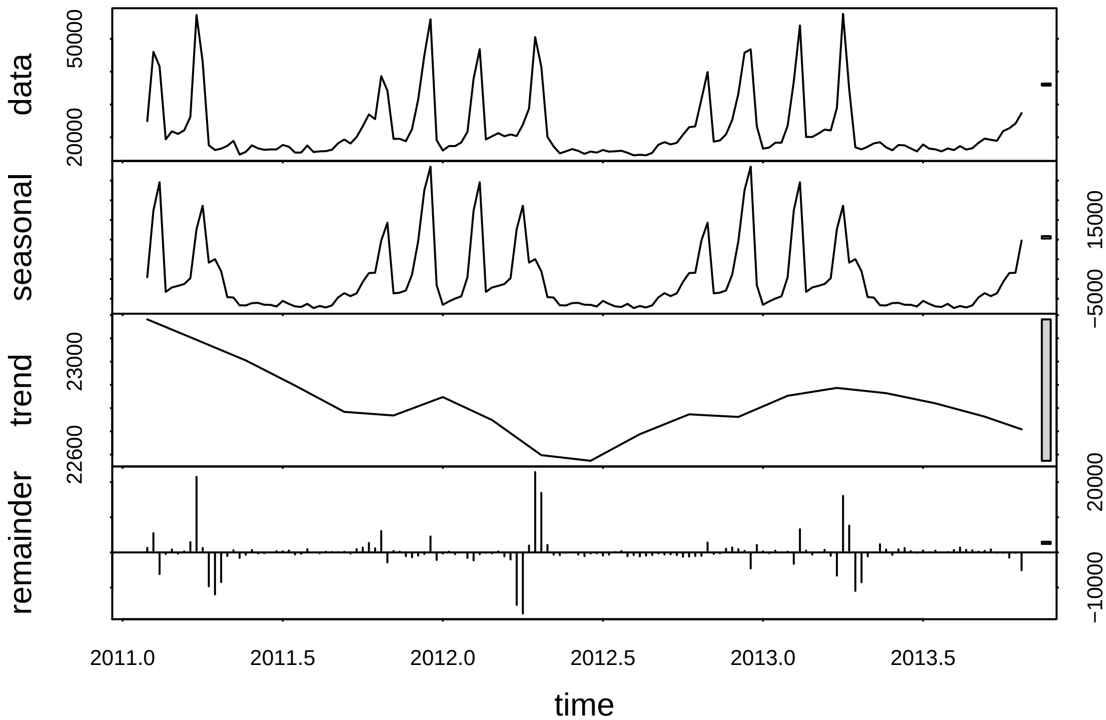
2.2 tslm regression using ARIMA

2.2.1 use stl function to decomposition manually

```r
one = read.csv("walmart_train.csv")
ntrain = 143
ntest = 39
ytrain.ts = ts(one$Weekly_Sales[1:ntrain], frequency = 52, start = c(2011,5))
ytest.ts = ts(one$Weekly_Sales[ntrain+1:ntest], frequency =52, start = c(2011, 5+ntrain))

one$IsHoliday = as.character(one$IsHoliday)
one$holiday_factor = ifelse(one$IsHoliday == "TRUE",1,0)
xtrain = one$holiday_factor[1:ntrain]
xtest = one$holiday_factor[(ntrain+1):(ntrain+ntest)]
```

First use stl to decompose the time series

```r
stl.run = stl(ytrain.ts, s.window = "periodic") # periodic means that seasonal will be identical over t
plot(stl.run)
```



Second user arima to train a model, and add it back

```
seasonal.comp = stl.run$time.series[,1]
deseasonalized.ts = ytrain.ts - seasonal.comp

arima.fit.deas = auto.arima(deseasonalized.ts, xreg = xtrain)
arima.fit.deas.pred = forecast(arima.fit.deas, xreg = xtest, h = ntest, level = 0)

seasonal.comp.pred = snaive(seasonal.comp, h = ntest)
alt.forecast = arima.fit.deas.pred$mean + seasonal.comp.pred$mean
```

use tslm automatically forecast

```
stlm.reg.fit = stlm(ytrain.ts, s.window = "periodic", xreg = xtrain, method = "arima")
stlm.reg.fit$model
```

```
## Series: x
## Regression with ARIMA(0,0,1) errors
##
## Coefficients:
##          ma1   intercept      xreg
##       0.5345  22782.8402   -10.283
## s.e.  0.0749    529.1278  1114.445
##
## sigma^2 estimated as 17096004:  log likelihood=-1392.35
## AIC=2792.69   AICc=2792.98   BIC=2804.55
```

```r
stlm.reg.pred = forecast(stlm.reg.fit, newxreg = xtest, h = ntest, level = 0)
stlm.reg.pred$mean - alt.forecast  ### these two method are the same!
```

```
## Time Series:
## Start = c(2013, 44)
## End = c(2014, 30)
## Frequency = 52
##  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [36] 0 0 0 0
```

```r
plot(stlm.reg.pred, xlab = "year", ylab = "weekly sales")
lines(ytest.ts, col = 'grey')
```