# w271: Homework 7 (Due: Week 8)

Grace Lin

# Due: Before the Live Session of Week 8

# Instructions (Please Read it Carefully!):

- **Page limit of the pdf report: None, but please be reasonable**
- Page setup:
  - Use the following font size, margin, and linespace:
    - fontsize=11pt
    - margin=1in
    - line_spacing=single
- Submission:
  - Homework needs to be completed individually; this is not a group project.
  - Each student submits his/her homework to the course github repo by the deadline; submission and revision made after the deadline will not be graded
  - Submit 2 files:
    1. A pdf file that details your answers. Include all the R codes used to produce the answers. *Please do not suppress the codes in your pdf file.*
    2. R markdown file used to produce the pdf file
  - Use the following file-naming convensation; fail to do so will receive $10\%$ reduction in the grade:
    - StudentFirstNameLastName_HWNumber.fileExtension
    - For example, if the student's name is Kyle Cartman for homework 1, name your files as
      - KyleCartman_HW1.Rmd
      - KyleCartman_HW1.pdf
  - Although it sounds obvious, please write your name on page 1 of your pdf and Rmd files.
  - For statistical methods that we cover in this course, use only the R libraries and functions that are covered in this course. If you use libraries and functions for statistical modeling that we have not covered, you have to (1) provide an explanation of why such libraries and functions are used instead and (2) reference to the library documentation. **Lacking the explanation and reference to the documentation will result in a score of zero for the corresponding question.** For data wrangling and data visualization, you are free to use other libraries, such as dplyr, ggplot2, etc.
  - For mathematical formulae, type them in your R markdown file. **Do not write them on a piece of paper, snap a photo, and either insert the image file or sumbit the image file separately. Doing so will receive a $0$ for that whole question.**
  - Students are expected to act with regards to UC Berkeley Academic Integrity.

In this homework, you are asked to use `quantmod` to get a time sereis `HOUST` from the Federal website, conduct Time Series EDA, examine seasonality, develop a model that can capture both trend and seasonality in the series, and plot the observed vs fitted value, and use the model to make a 12-step ahead forecast.

```
#install.packages("quantmod") #if you have not installed quantmod
library(quantmod)
```

```
## Loading required package: xts
```

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

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

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

```
## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo
```

```
## Loading required package: TTR
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
## Version 0.4-0 included new data defaults. See ?getSymbols.
```

```
# Use Quantmod to load data
HOUST = getSymbols('HOUST', src='FRED', auto.assign = F)
```

```
## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be checked for
## alternate defaults.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.
```

```
str(HOUST)
```

```
## An 'xts' object on 1959-01-01/2019-05-01 containing:
##   Data: int [1:725, 1] 1657 1667 1620 1590 1498 1503 1547 1430 1540 1355 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr "HOUST"
##   Indexed by objects of class: [Date] TZ: UTC
##   xts Attributes:
## List of 2
##  $ src    : chr "FRED"
##  $ updated: POSIXct[1:1], format: "2019-07-01 10:55:33"
```

```
head(HOUST)
```

```
##            HOUST
## 1959-01-01  1657
## 1959-02-01  1667
## 1959-03-01  1620
## 1959-04-01  1590
## 1959-05-01  1498
## 1959-06-01  1503
```

```
tail(HOUST)
```

```
##            HOUST
## 2018-12-01  1142
## 2019-01-01  1291
## 2019-02-01  1149
## 2019-03-01  1199
## 2019-04-01  1281
## 2019-05-01  1269
```

```
house.starts = ts(HOUST, frequency = 12, start = c(1959,1))
# subset your time series to one starting in Januagy 2010
house.starts = window(house.starts, start = c(2010,1))

# Examine your data
# YOUR CODE TO BE HERE
summary(house.starts)
```

```
##      HOUST
##  Min.   : 517.0
##  1st Qu.: 740.0
##  Median :1023.0
##  Mean   : 973.7
##  3rd Qu.:1184.0
##  Max.   :1335.0
```

```
# Examine seasonality
# YOUR CODE TO BE HERE
library(ggplot2)
```

```
## Registered S3 methods overwritten by 'ggplot2':
##   method         from
##   [.quosures     rlang
##   c.quosures     rlang
##   print.quosures rlang
```

```
library(fpp2)
```

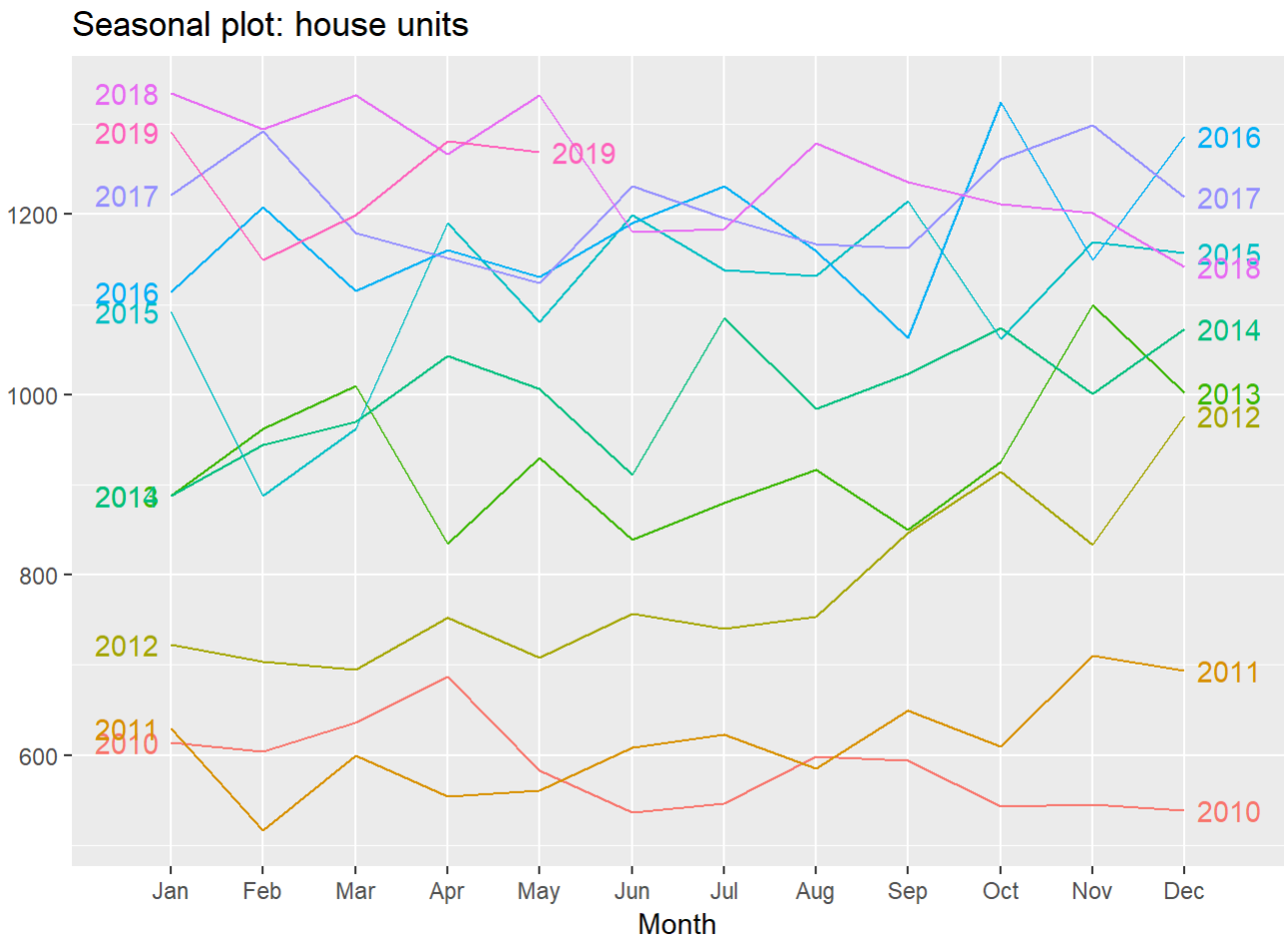
```
## Loading required package: forecast
```

```
## Registered S3 methods overwritten by 'forecast':
##   method             from
##   fitted.fracdiff    fracdiff
##   residuals.fracdiff fracdiff
```

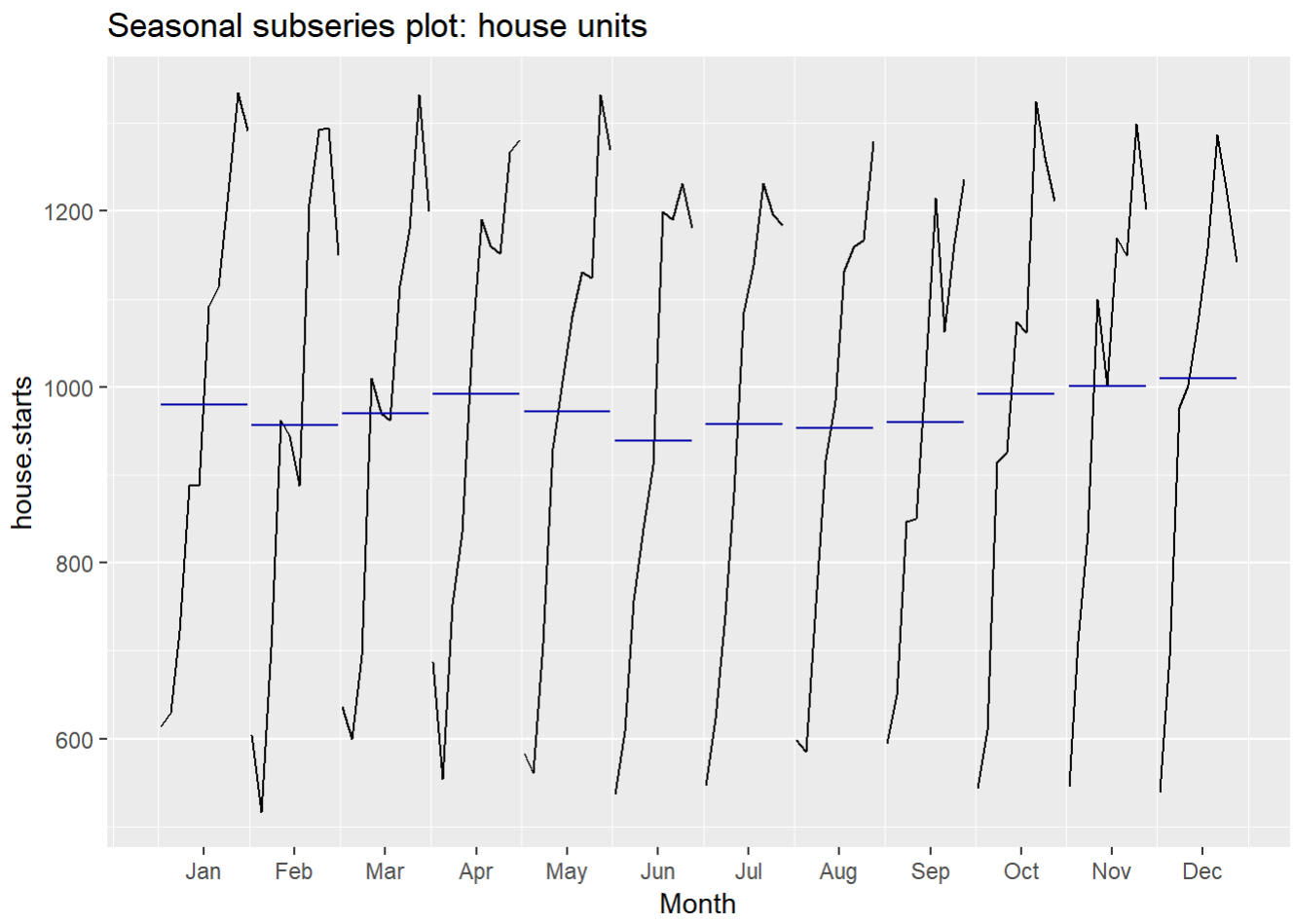
```
## Loading required package: fma
```

```
## Loading required package: expsmooth
```

```
ggseasonplot(house.starts, year.labels=TRUE, year.labels.left=TRUE) +
  ggtitle("Seasonal plot: house units")
```




```
ggsubseriesplot(house.starts) +
  ggtitle("Seasonal subseries plot: house units")
```




```
  str(house.starts)
```

```
##  Time-Series [1:113, 1] from 2010 to 2019: 614 604 636 687 583 536 546 599 594 543 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr "HOUST"
```

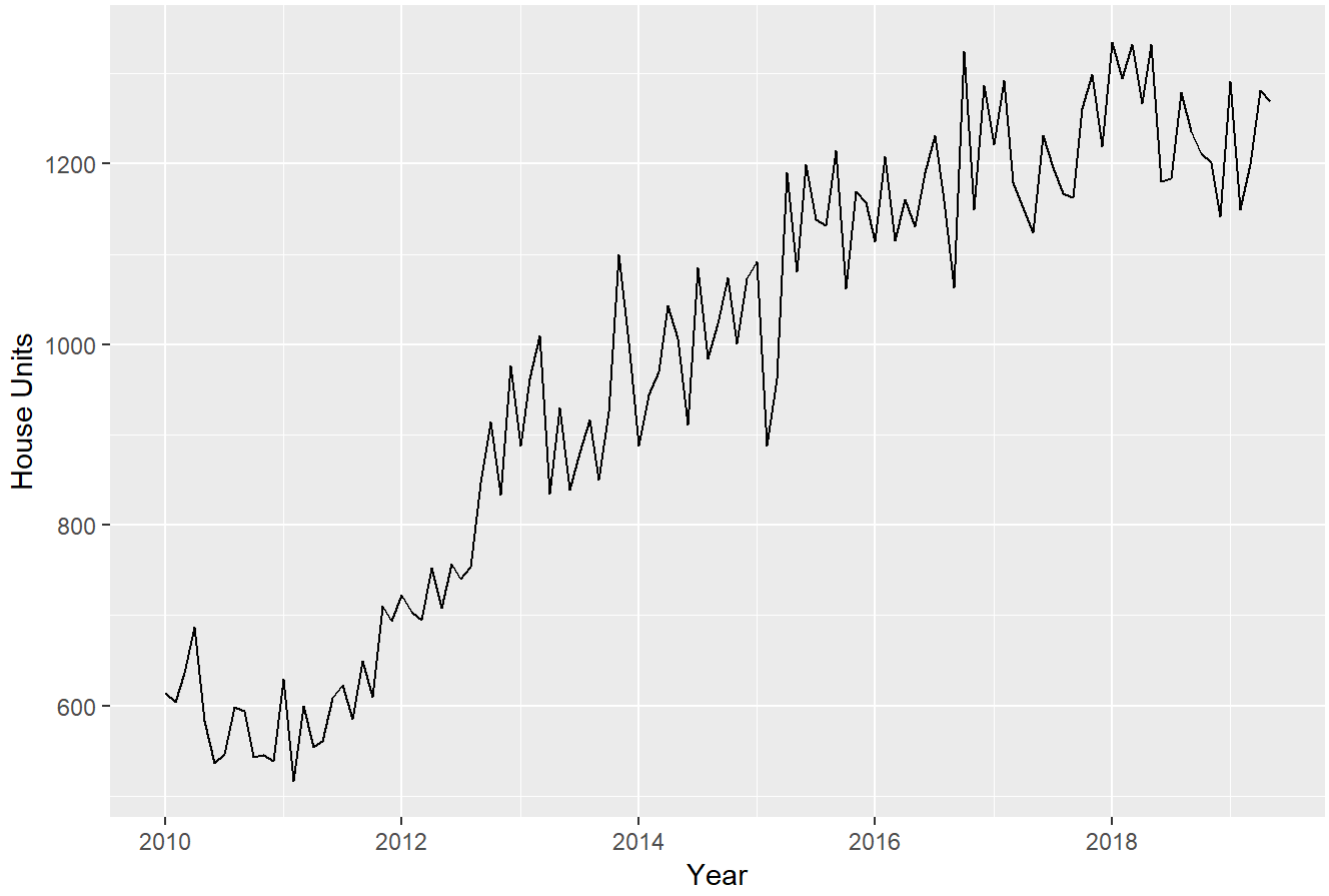
```
  head(house.starts)
```

```
##      Jan Feb Mar Apr May Jun
## 2010 614 604 636 687 583 536
```

```
  tail(house.starts)
```

```
##       Jan  Feb  Mar  Apr  May Jun Jul Aug Sep Oct Nov  Dec
## 2018                                                1142
## 2019 1291 1149 1199 1281 1269
```

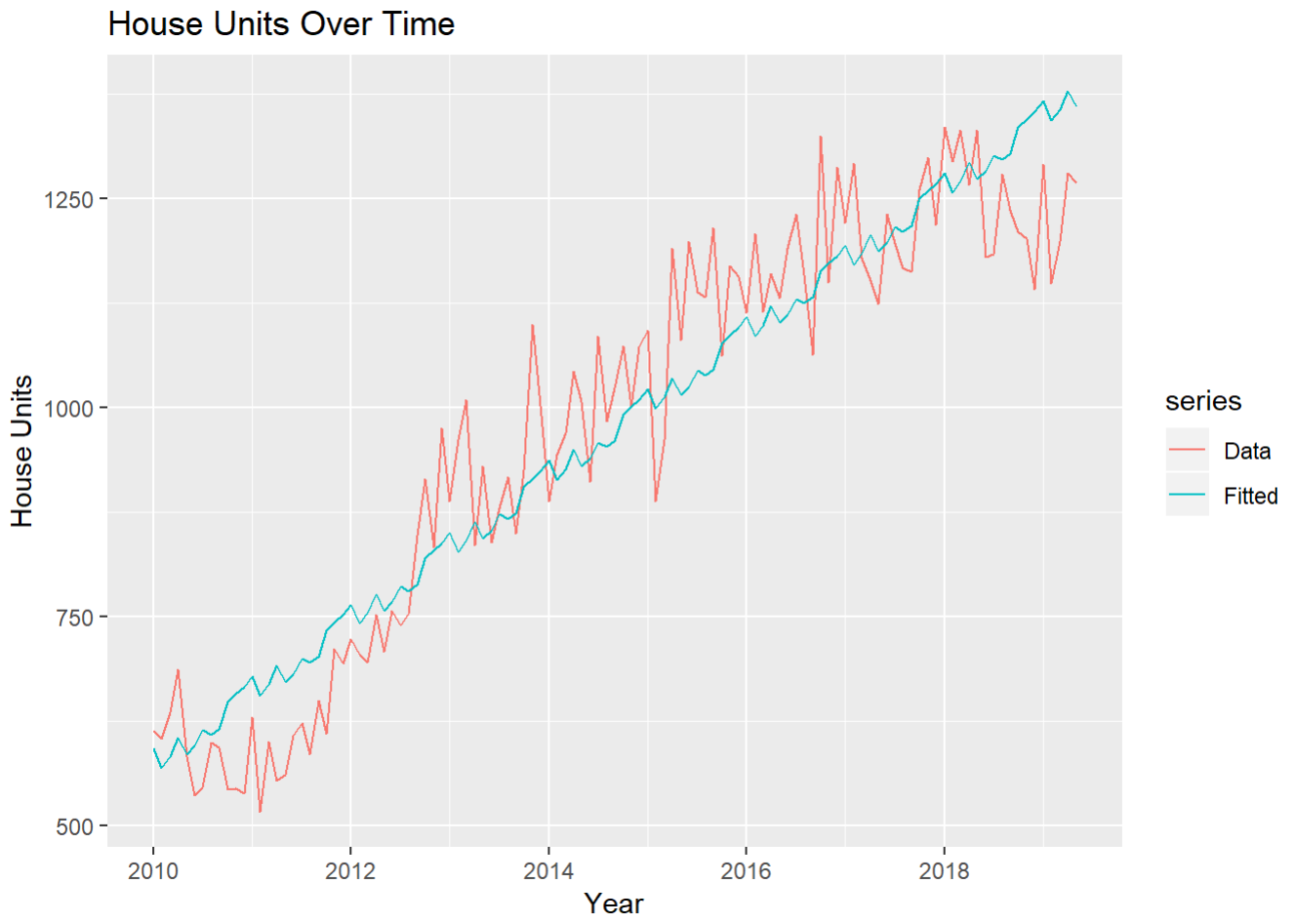
```
autoplot(house.starts) + xlab("Year") + ylab("House Units")
```




```
# Estimate a model with trend, seasonlity, or both
# YOUR CODE TO BE HERE
fit.house <- tslm(house.starts ~ trend + season)
summary(fit.house)
```

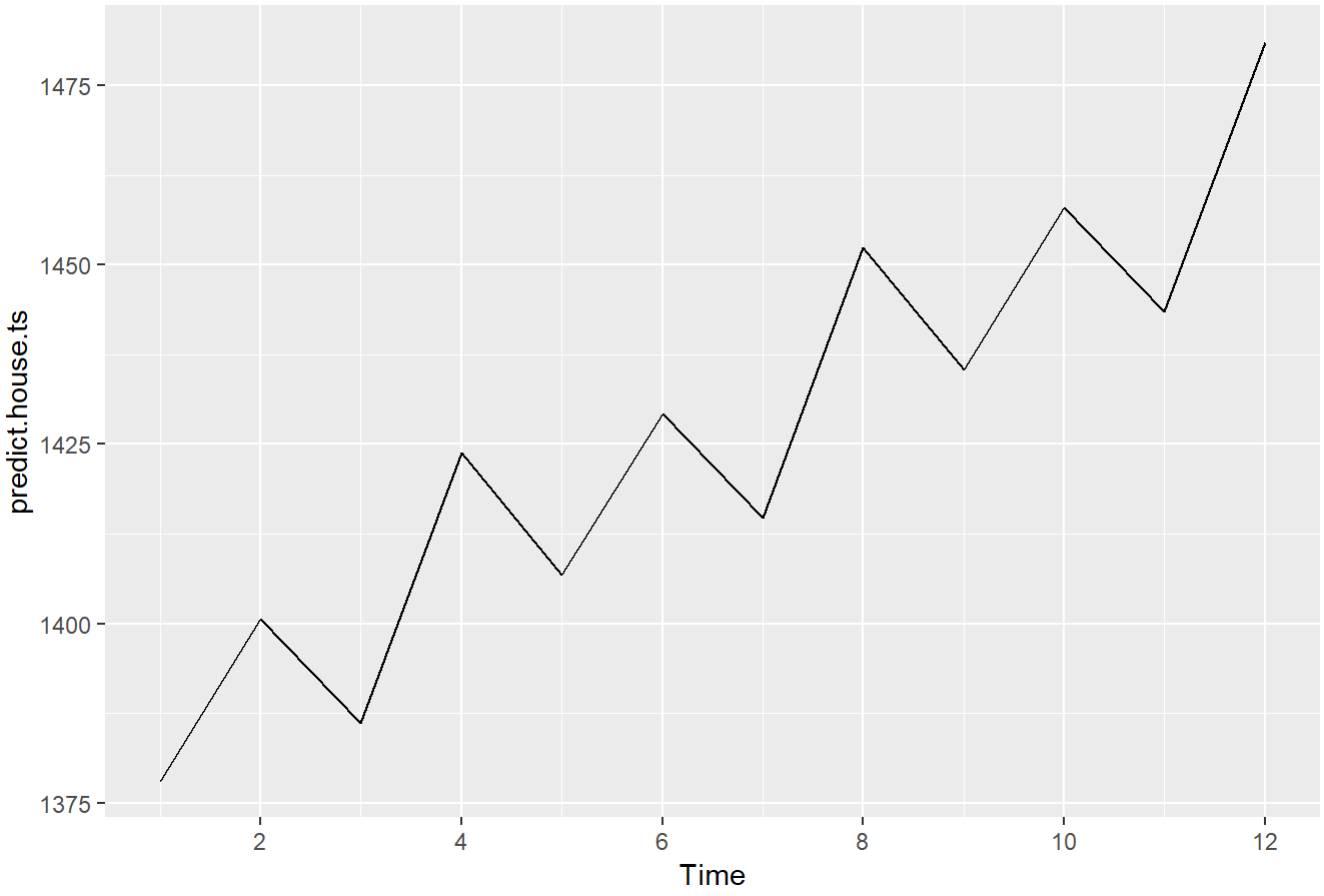
```
##
## Call:
## tslm(formula = house.starts ~ trend + season)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -211.730  -58.008   -5.701   61.339  184.849
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 585.6150    31.9512  18.328   <2e-16 ***
## trend         7.1634     0.2608  27.470   <2e-16 ***
## season2     -30.4634    40.3785  -0.754    0.452
## season3     -24.1267    40.3811  -0.597    0.552
## season4      -8.7901    40.3853  -0.218    0.828
## season5     -35.6535    40.3912  -0.883    0.380
## season6     -33.2144    41.4850  -0.801    0.425
## season7     -21.3778    41.4841  -0.515    0.607
## season8     -33.8745    41.4850  -0.817    0.416
## season9     -33.8156    41.4874  -0.815    0.417
## season10     -9.3123    41.4915  -0.224    0.823
## season11     -7.1423    41.4973  -0.172    0.864
## season12     -5.5279    41.5046  -0.133    0.894
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 90.29 on 100 degrees of freedom
## Multiple R-squared:  0.8838, Adjusted R-squared:  0.8698
## F-statistic: 63.36 on 12 and 100 DF,  p-value: < 2.2e-16
```

```
# Plot the observed and fitted values
autoplot(house.starts, series="Data") +
  autolayer(fitted(fit.house), series="Fitted") +
  xlab("Year") + ylab("House Units") +
  ggtitle("House Units Over Time")
```




```
# Make a 12-step ahead (out-of-sample) forecast
# That is, forecast from 2018-10-01 to 2019-09-01
predict.time = data.frame(intercept=c(1,1,1,1,1,1,1,1,1,1,1,1),
  trend= seq(max(fit.house$model['trend'])+1, max(fit.house$model['trend'])+12,1),
season2 = c(0,0,1,0,0,0,1,0,0,0,1,0),
season3 = c(1,0,0,0,1,0,0,0,1,0,0,0),
season4 = c(0,1,0,0,0,1,0,0,0,1,0,0))

predict.house = fit.house$coefficients[1]*predict.time[1] +
  fit.house$coefficients[2]*predict.time[2] +
  fit.house$coefficients[3]*predict.time[3] +
fit.house$coefficients[4]*predict.time[4] +
  fit.house$coefficients[5]*predict.time[5]
predict.house.ts = ts(predict.house)
autoplot(predict.house.ts)
```


predict.house.ts
1475
1450
1425
1400
1375
2
4
6
8
10
12
Time