P3\_G8\_Time\_Series

g8

13/07/2020

## Load Library

library(tidyverse)

## -- Attaching packages ---------------------------------------------------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2 v purrr 0.3.4  
## v tibble 3.0.3 v dplyr 1.0.0  
## v tidyr 1.1.0 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.5.0

## -- Conflicts ------------------------------------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(ggplot2)  
  
library(ggfortify)  
  
  
#install.packages("autoplotly") # For autoplot  
library(autoplotly)  
#install.packages("forecast") # For ggseasonplot  
library(forecast)

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

## Registered S3 methods overwritten by 'forecast':  
## method from   
## autoplot.Arima ggfortify  
## autoplot.acf ggfortify  
## autoplot.ar ggfortify  
## autoplot.bats ggfortify  
## autoplot.decomposed.ts ggfortify  
## autoplot.ets ggfortify  
## autoplot.forecast ggfortify  
## autoplot.stl ggfortify  
## autoplot.ts ggfortify  
## fitted.ar ggfortify  
## fortify.ts ggfortify  
## residuals.ar ggfortify

library(lubridate) # parse\_date\_time

##   
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':  
##   
## date, intersect, setdiff, union

#install.packages("changepoint")  
library(changepoint) # For Cahnge point

## Loading required package: zoo

##   
## Attaching package: 'zoo'

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

## Successfully loaded changepoint package version 2.2.2  
## NOTE: Predefined penalty values changed in version 2.2. Previous penalty values with a postfix 1 i.e. SIC1 are now without i.e. SIC and previous penalties without a postfix i.e. SIC are now with a postfix 0 i.e. SIC0. See NEWS and help files for further details.

library(stats)  
  
library(dygraphs) # For dygraph  
library(dplyr)  
library(tseries)  
library(vars)

## Loading required package: MASS

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':  
##   
## select

## Loading required package: strucchange

## Loading required package: sandwich

##   
## Attaching package: 'strucchange'

## The following object is masked from 'package:stringr':  
##   
## boundary

## Loading required package: urca

## Loading required package: lmtest

## Load Data

We get data from : <https://www150.statcan.gc.ca/t1/tbl1/en/tv.action?pid=1810020501> # Two DataSet first with Stats , Second with Cities in the Canada Stats

housing\_canada<- read.csv("CSV/house1980\_2020.csv")  
housing\_canada\_city<- read.csv("CSV/house1980\_2020\_with\_city.csv")

## Exploration Data

str (housing\_canada)

## 'data.frame': 1892 obs. of 15 variables:  
## $ ï..REF\_DATE : Factor w/ 473 levels "1981-01","1981-02",..: 1 2 3 4 5 6 7 8 9 10 ...  
## $ GEO : Factor w/ 4 levels "British Columbia",..: 2 2 2 2 2 2 2 2 2 2 ...  
## $ DGUID : Factor w/ 4 levels "2016A000011124",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ New.housing.price.indexes: Factor w/ 1 level "House only": 1 1 1 1 1 1 1 1 1 1 ...  
## $ UOM : Factor w/ 1 level "Index, 201612=100": 1 1 1 1 1 1 1 1 1 1 ...  
## $ UOM\_ID : int 347 347 347 347 347 347 347 347 347 347 ...  
## $ SCALAR\_FACTOR : Factor w/ 1 level "units": 1 1 1 1 1 1 1 1 1 1 ...  
## $ SCALAR\_ID : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ VECTOR : Factor w/ 4 levels "v111955443","v111955473",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ COORDINATE : num 1.2 1.2 1.2 1.2 1.2 1.2 1.2 1.2 1.2 1.2 ...  
## $ VALUE : num 36.1 36.5 37.3 38.1 38.9 39.1 39.2 39 39 38.8 ...  
## $ STATUS : Factor w/ 2 levels "","..": 1 1 1 1 1 1 1 1 1 1 ...  
## $ SYMBOL : logi NA NA NA NA NA NA ...  
## $ TERMINATED : logi NA NA NA NA NA NA ...  
## $ DECIMALS : int 1 1 1 1 1 1 1 1 1 1 ...

head(housing\_canada)

## ï..REF\_DATE GEO DGUID New.housing.price.indexes UOM  
## 1 1981-01 Canada 2016A000011124 House only Index, 201612=100  
## 2 1981-02 Canada 2016A000011124 House only Index, 201612=100  
## 3 1981-03 Canada 2016A000011124 House only Index, 201612=100  
## 4 1981-04 Canada 2016A000011124 House only Index, 201612=100  
## 5 1981-05 Canada 2016A000011124 House only Index, 201612=100  
## 6 1981-06 Canada 2016A000011124 House only Index, 201612=100  
## UOM\_ID SCALAR\_FACTOR SCALAR\_ID VECTOR COORDINATE VALUE STATUS SYMBOL  
## 1 347 units 0 v111955443 1.2 36.1 NA  
## 2 347 units 0 v111955443 1.2 36.5 NA  
## 3 347 units 0 v111955443 1.2 37.3 NA  
## 4 347 units 0 v111955443 1.2 38.1 NA  
## 5 347 units 0 v111955443 1.2 38.9 NA  
## 6 347 units 0 v111955443 1.2 39.1 NA  
## TERMINATED DECIMALS  
## 1 NA 1  
## 2 NA 1  
## 3 NA 1  
## 4 NA 1  
## 5 NA 1  
## 6 NA 1

summary(housing\_canada)

## ï..REF\_DATE GEO DGUID   
## 1981-01: 4 British Columbia:473 2016A000011124:473   
## 1981-02: 4 Canada :473 2016A000224 :473   
## 1981-03: 4 Ontario :473 2016A000235 :473   
## 1981-04: 4 Quebec :473 2016A000259 :473   
## 1981-05: 4   
## 1981-06: 4   
## (Other):1868   
## New.housing.price.indexes UOM UOM\_ID SCALAR\_FACTOR  
## House only:1892 Index, 201612=100:1892 Min. :347 units:1892   
## 1st Qu.:347   
## Median :347   
## Mean :347   
## 3rd Qu.:347   
## Max. :347   
##   
## SCALAR\_ID VECTOR COORDINATE VALUE STATUS   
## Min. :0 v111955443:473 Min. : 1.2 Min. : 33.00 :1712   
## 1st Qu.:0 v111955473:473 1st Qu.: 8.7 1st Qu.: 53.98 ..: 180   
## Median :0 v111955491:473 Median :14.2 Median : 74.85   
## Mean :0 v111955551:473 Mean :16.7 Mean : 73.94   
## 3rd Qu.:0 3rd Qu.:22.2 3rd Qu.: 94.40   
## Max. :0 Max. :37.2 Max. :112.30   
## NA's :180   
## SYMBOL TERMINATED DECIMALS  
## Mode:logical Mode:logical Min. :1   
## NA's:1892 NA's:1892 1st Qu.:1   
## Median :1   
## Mean :1   
## 3rd Qu.:1   
## Max. :1   
##

## Preparation Data

Filtering and Cleaning the data

names(housing\_canada)

## [1] "ï..REF\_DATE" "GEO"   
## [3] "DGUID" "New.housing.price.indexes"  
## [5] "UOM" "UOM\_ID"   
## [7] "SCALAR\_FACTOR" "SCALAR\_ID"   
## [9] "VECTOR" "COORDINATE"   
## [11] "VALUE" "STATUS"   
## [13] "SYMBOL" "TERMINATED"   
## [15] "DECIMALS"

names(housing\_canada\_city)

## [1] "ï..REF\_DATE" "GEO"   
## [3] "DGUID" "New.housing.price.indexes"  
## [5] "UOM" "UOM\_ID"   
## [7] "SCALAR\_FACTOR" "SCALAR\_ID"   
## [9] "VECTOR" "COORDINATE"   
## [11] "VALUE" "STATUS"   
## [13] "SYMBOL" "TERMINATED"   
## [15] "DECIMALS"

#ReName column 1   
colnames(housing\_canada)[1] <- "ref\_date" # assign new name  
colnames(housing\_canada\_city)[1] <- "ref\_date" # assign new name  
  
# Convert ref\_date from Vactor to Date Format   
head(housing\_canada$ref\_date,5) # check rows

## [1] 1981-01 1981-02 1981-03 1981-04 1981-05  
## 473 Levels: 1981-01 1981-02 1981-03 1981-04 1981-05 1981-06 1981-07 ... 2020-05

housing\_canada$full\_date = parse\_date\_time(housing\_canada$ref\_date, '%Y-%m')  
housing\_canada$full\_date <- as.Date(housing\_canada$full\_date, format = "%Y.%m.%d")  
head(housing\_canada$full\_date,5)

## [1] "1981-01-01" "1981-02-01" "1981-03-01" "1981-04-01" "1981-05-01"

housing\_canada$yr <-year(housing\_canada$full\_date)  
head(housing\_canada, 5)

## ref\_date GEO DGUID New.housing.price.indexes UOM  
## 1 1981-01 Canada 2016A000011124 House only Index, 201612=100  
## 2 1981-02 Canada 2016A000011124 House only Index, 201612=100  
## 3 1981-03 Canada 2016A000011124 House only Index, 201612=100  
## 4 1981-04 Canada 2016A000011124 House only Index, 201612=100  
## 5 1981-05 Canada 2016A000011124 House only Index, 201612=100  
## UOM\_ID SCALAR\_FACTOR SCALAR\_ID VECTOR COORDINATE VALUE STATUS SYMBOL  
## 1 347 units 0 v111955443 1.2 36.1 NA  
## 2 347 units 0 v111955443 1.2 36.5 NA  
## 3 347 units 0 v111955443 1.2 37.3 NA  
## 4 347 units 0 v111955443 1.2 38.1 NA  
## 5 347 units 0 v111955443 1.2 38.9 NA  
## TERMINATED DECIMALS full\_date yr  
## 1 NA 1 1981-01-01 1981  
## 2 NA 1 1981-02-01 1981  
## 3 NA 1 1981-03-01 1981  
## 4 NA 1 1981-04-01 1981  
## 5 NA 1 1981-05-01 1981

# For 2nd Data  
head(housing\_canada\_city$ref\_date,5) # check rows

## [1] 1981-01 1981-02 1981-03 1981-04 1981-05  
## 473 Levels: 1981-01 1981-02 1981-03 1981-04 1981-05 1981-06 1981-07 ... 2020-05

housing\_canada\_city$full\_date = parse\_date\_time(housing\_canada\_city$ref\_date, '%Y-%m')  
housing\_canada\_city$full\_date <- as.Date(housing\_canada\_city$full\_date, format = "%Y.%m.%d")  
head(housing\_canada\_city$full\_date,5)

## [1] "1981-01-01" "1981-02-01" "1981-03-01" "1981-04-01" "1981-05-01"

housing\_canada\_city$yr <-year(housing\_canada\_city$full\_date)  
head(housing\_canada\_city, 5)

## ref\_date GEO DGUID New.housing.price.indexes UOM  
## 1 1981-01 Canada 2016A000011124 House only Index, 201612=100  
## 2 1981-02 Canada 2016A000011124 House only Index, 201612=100  
## 3 1981-03 Canada 2016A000011124 House only Index, 201612=100  
## 4 1981-04 Canada 2016A000011124 House only Index, 201612=100  
## 5 1981-05 Canada 2016A000011124 House only Index, 201612=100  
## UOM\_ID SCALAR\_FACTOR SCALAR\_ID VECTOR COORDINATE VALUE STATUS SYMBOL  
## 1 347 units 0 v111955443 1.2 36.1 NA  
## 2 347 units 0 v111955443 1.2 36.5 NA  
## 3 347 units 0 v111955443 1.2 37.3 NA  
## 4 347 units 0 v111955443 1.2 38.1 NA  
## 5 347 units 0 v111955443 1.2 38.9 NA  
## TERMINATED DECIMALS full\_date yr  
## 1 NA 1 1981-01-01 1981  
## 2 NA 1 1981-02-01 1981  
## 3 NA 1 1981-03-01 1981  
## 4 NA 1 1981-04-01 1981  
## 5 NA 1 1981-05-01 1981

colSums(is.na(housing\_canada)) # checking NA in all columns

## ref\_date GEO DGUID   
## 0 0 0   
## New.housing.price.indexes UOM UOM\_ID   
## 0 0 0   
## SCALAR\_FACTOR SCALAR\_ID VECTOR   
## 0 0 0   
## COORDINATE VALUE STATUS   
## 0 180 0   
## SYMBOL TERMINATED DECIMALS   
## 1892 1892 0   
## full\_date yr   
## 0 0

#Remove Al NA in VALUE column   
# For 1St Data  
housing\_canada <- housing\_canada %>% filter(!VALUE=="NA")  
# Remove Total (house and land) and Land only  
  
housing\_canada %>% filter(New.housing.price.indexes=="Land only" & New.housing.price.indexes=="Total (house and land)") # check for items, unfortunately not items is there

## [1] ref\_date GEO   
## [3] DGUID New.housing.price.indexes  
## [5] UOM UOM\_ID   
## [7] SCALAR\_FACTOR SCALAR\_ID   
## [9] VECTOR COORDINATE   
## [11] VALUE STATUS   
## [13] SYMBOL TERMINATED   
## [15] DECIMALS full\_date   
## [17] yr   
## <0 rows> (or 0-length row.names)

housing\_canada <- housing\_canada %>% filter(!New.housing.price.indexes=="Land only" & !New.housing.price.indexes=="Total (house and land)") # removing above items if available  
  
  
# For 2nd Data  
colSums(is.na(housing\_canada\_city)) # checking NA in all columns

## ref\_date GEO DGUID   
## 0 0 0   
## New.housing.price.indexes UOM UOM\_ID   
## 0 0 0   
## SCALAR\_FACTOR SCALAR\_ID VECTOR   
## 0 0 0   
## COORDINATE VALUE STATUS   
## 0 2766 0   
## SYMBOL TERMINATED DECIMALS   
## 10406 10406 0   
## full\_date yr   
## 0 0

housing\_canada\_city <- housing\_canada\_city %>% filter(!VALUE=="NA")  
# Remove Total (house and land) and Land only  
unique(housing\_canada\_city$New.housing.price.indexes)

## [1] House only  
## Levels: House only

housing\_canada\_city %>% filter(New.housing.price.indexes=="Land only" & New.housing.price.indexes=="Total (house and land)") # checking items

## [1] ref\_date GEO   
## [3] DGUID New.housing.price.indexes  
## [5] UOM UOM\_ID   
## [7] SCALAR\_FACTOR SCALAR\_ID   
## [9] VECTOR COORDINATE   
## [11] VALUE STATUS   
## [13] SYMBOL TERMINATED   
## [15] DECIMALS full\_date   
## [17] yr   
## <0 rows> (or 0-length row.names)

housing\_canada\_city <- housing\_canada\_city %>% filter(!New.housing.price.indexes=="Land only" & !New.housing.price.indexes=="Total (house and land)") # removing above item if available

## Select the specific Stat or City

With shiny we have to make List of all states take from 1st Data and List of all cities take it from 2nd Data

# List Of States  
unique(housing\_canada$GEO)

## [1] Canada Quebec Ontario British Columbia  
## Levels: British Columbia Canada Ontario Quebec

# List Of Cities  
unique(housing\_canada\_city$GEO)

## [1] Canada   
## [2] Quebec   
## [3] QuÃ©bec, Quebec   
## [4] Sherbrooke, Quebec   
## [5] Trois-RiviÃ¨res, Quebec   
## [6] MontrÃ©al, Quebec   
## [7] Ottawa-Gatineau, Quebec part, Ontario/Quebec   
## [8] Ontario   
## [9] Ottawa-Gatineau, Ontario part, Ontario/Quebec  
## [10] Oshawa, Ontario   
## [11] Toronto, Ontario   
## [12] Hamilton, Ontario   
## [13] St. Catharines-Niagara, Ontario   
## [14] Kitchener-Cambridge-Waterloo, Ontario   
## [15] Guelph, Ontario   
## [16] London, Ontario   
## [17] Windsor, Ontario   
## [18] Greater Sudbury, Ontario   
## [19] British Columbia   
## [20] Kelowna, British Columbia   
## [21] Vancouver, British Columbia   
## [22] Victoria, British Columbia   
## 22 Levels: British Columbia Canada Greater Sudbury, Ontario ... Windsor, Ontario

#Here the have be for nest loop to get the option and subset Data from the Data UNLIMITED  
# Supose we select Stats Canada & Quebec  
# And Cities Toronto, Ontario & MontrÃ©al, Quebec  
  
write.csv(housing\_canada, "housing\_canada.csv")  
write.csv(housing\_canada\_city, "housing\_canada\_city.csv")  
  
cases <- c('Canada', 'Quebec', 'Toronto, Ontario' ,'Windsor, Ontario')  
  
  
housing\_sp1 <- housing\_canada %>% filter( GEO == cases[1]) # province level 1  
housing\_sp2 <- housing\_canada %>% filter( GEO == cases[2]) # province level 2  
  
housing\_sp3 <- housing\_canada\_city %>% filter( GEO == cases[3]) # City level 1  
housing\_sp4 <- housing\_canada\_city %>% filter( GEO == cases[4]) # city level 2  
  
  
  
# Cut Data By The Start Date and End Date which be inputs   
# I make Code To Select Date as interactive face   
start\_date <- parse\_date\_time("1980-01-01", "%Y-%m-%d")  
end\_date <- parse\_date\_time("2020-12-01", "%Y-%m-%d")  
  
  
# Get Start year from each sub-Data set  
start\_year1 <- min(housing\_sp1$yr) # min start date of province level 1  
start\_year2 <- min(housing\_sp2$yr) # min start date of province level 2  
start\_year3 <- min(housing\_sp3$yr) # min start date of city level 1  
start\_year4 <- min(housing\_sp4$yr) # min start date of city level 2

## Creating a Time Series Object

To create a times series object, we start with the ts function The most basic option is a singular index, so by year or some measure of time. This increments up by one for each observation but we will make it in terms of months. This may seem a bit redundant order is frequency = 12.

time\_series\_m1 <- ts(housing\_sp1$VALUE, start = c(start\_year1, 1), frequency = 12) # ts for province level 1  
time\_series\_m2 <- ts(housing\_sp2$VALUE, start = c(start\_year2, 2), frequency = 12) # ts for province level 2  
  
time\_series\_m3 <- ts(housing\_sp3$VALUE, start = c(start\_year3, 3), frequency = 12) # ts for city level 1  
time\_series\_m4 <- ts(housing\_sp4$VALUE, start = c(start\_year4, 4), frequency = 12) # ts for city level 2

## Time plots

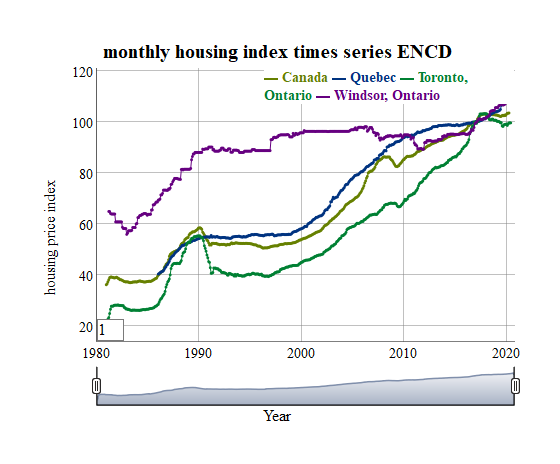
ggplot2 does not normally support the ts class, so we will need the package ggfortify to allow us to make the plots. To do this we use autoplot

Here we notice a steady cyclical climb until about 2008 when the housing bubble popped. Then steady increase after about 2012. This makes sense given hsitorical context, so our plot has been sucessfully made.

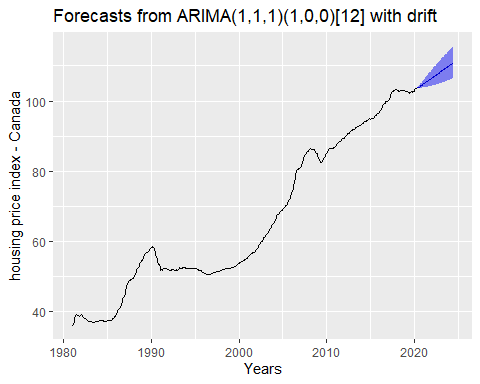
webshot::install\_phantomjs( force = TRUE)

## phantomjs has been installed to C:\Users\asamnani\AppData\Roaming\PhantomJS

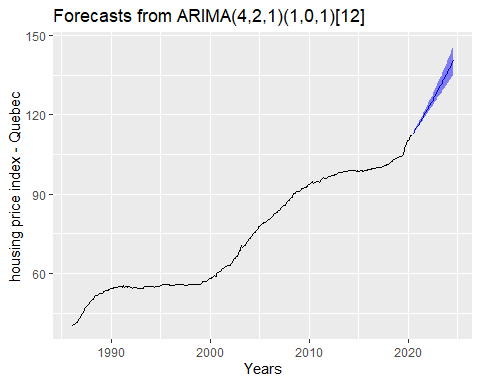
All <- cbind (time\_series\_m1,time\_series\_m2,time\_series\_m3,time\_series\_m4) # combine province 1 & 2 and city 1 & 2  
  
dygraph (All , main = "monthly housing index times series ENCD" ,xlab="Year" ) %>%  
 dySeries("time\_series\_m1", label= cases[1], drawPoints = TRUE) %>%  
 dySeries("time\_series\_m2", label= cases[2], drawPoints = TRUE) %>%  
 dySeries("time\_series\_m3", label= cases[3], drawPoints = TRUE) %>%  
 dySeries("time\_series\_m4", label= cases[4], drawPoints = TRUE) %>%  
 dyAxis("y",label = "housing price index") %>%  
 dyAxis("y2", label = "Years") %>%  
 dyLegend(show = "always", hideOnMouseOut = FALSE ) %>%  
 dyRangeSelector(dateWindow = c(start\_date, end\_date)) %>%  
 dyRoller()



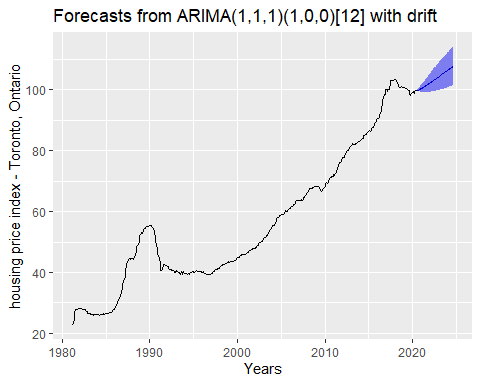
# Plotting with forecast package for Each series ARIMA (Auto-Regressive Integrated Moving Average)  
d.arima1 <- auto.arima(time\_series\_m1)  
d.forecast1 <- forecast(d.arima1, level = c(50), h = 50)  
autoplot(d.forecast1 , title = "sss" , xlab = "Years", ylab = "housing price index - Canada" ) # for province level 1



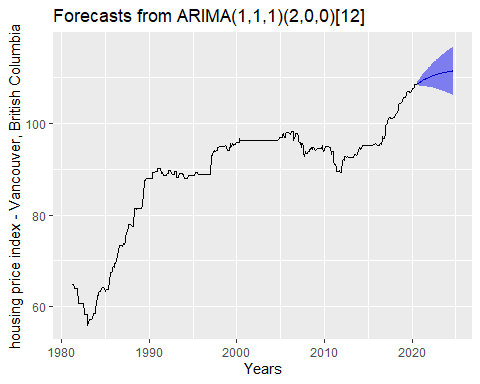
d.arima2 <- auto.arima(time\_series\_m2)  
d.forecast2 <- forecast(d.arima2, level = c(50), h = 50)  
autoplot(d.forecast2 , title = "sss" , xlab = "Years", ylab = "housing price index - Quebec" ) # for province level 2



d.arima3 <- auto.arima(time\_series\_m3)  
d.forecast3 <- forecast(d.arima3, level = c(50), h = 50)  
autoplot(d.forecast3 , title = "sss" , xlab = "Years", ylab = "housing price index - Toronto, Ontario" ) # for city level 1

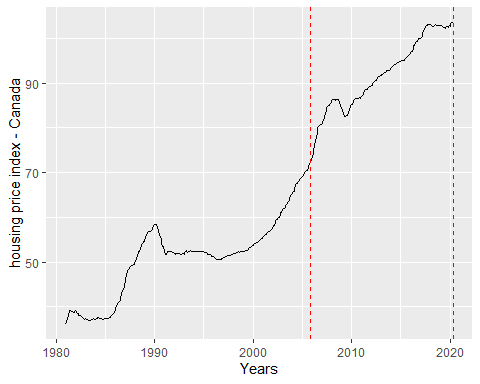


d.arima4 <- auto.arima(time\_series\_m4)  
d.forecast4 <- forecast(d.arima4, level = c(50), h = 50)  
autoplot(d.forecast4 , title = "sss" , xlab = "Years", ylab = "housing price index - Vancouver, British Columbia" ) # for city level 2

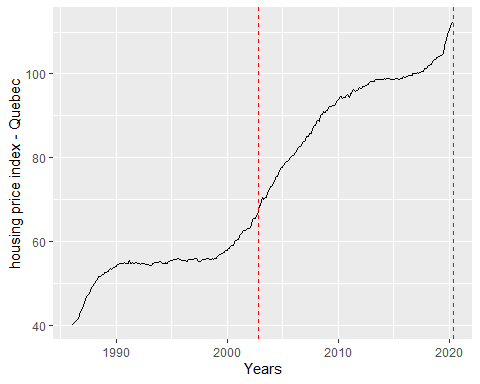


#What are Changepoints?  
#Changepoint analysis for time series is an increasingly important aspect of statistics. Simply put, a changepoint is an instance in time where the statistical properties before and after this time point differ. With potential changes naturally occurring in data and many statistical methods assuming a “no change” setup, changepoint analysis is important in both applied and theoretical statistics  
#Implements various mainstream and specialised changepoint methods for finding single and multiple changepoints within data. Many popular non-parametric and frequentist methods are included.  
#Users should start by looking at the documentation for cpt.mean(), cpt.var() and cpt.meanvar()  
  
#Plotting with changepoint package #OK  
autoplot(cpt.meanvar(time\_series\_m1) , title = "sss \n dd" , xlab = "Years", ylab = "housing price index - Canada" ) # for province level 1

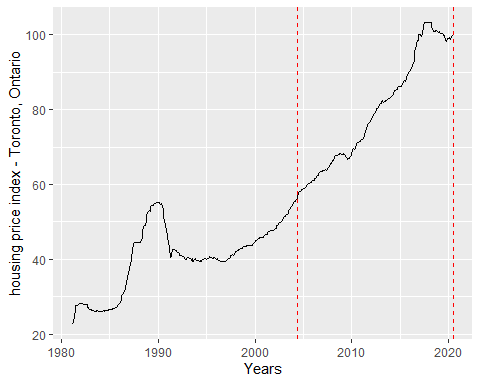
## Warning: `filter\_()` is deprecated as of dplyr 0.7.0.  
## Please use `filter()` instead.  
## See vignette('programming') for more help  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_warnings()` to see where this warning was generated.



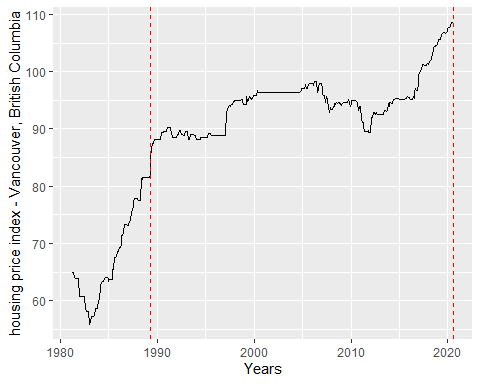
autoplot(cpt.meanvar(time\_series\_m2) , title = "sss \n dd" , xlab = "Years", ylab = "housing price index - Quebec" ) # for province level 2



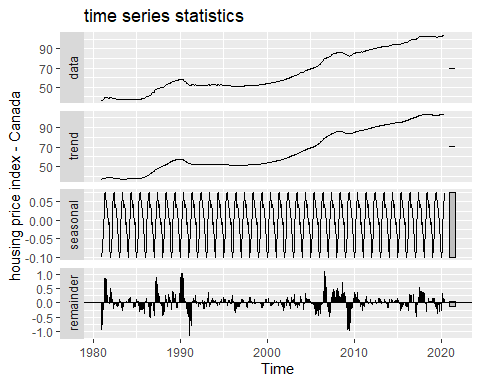
autoplot(cpt.meanvar(time\_series\_m3) , title = "sss \n dd" , xlab = "Years", ylab = "housing price index - Toronto, Ontario" ) # for city level 1



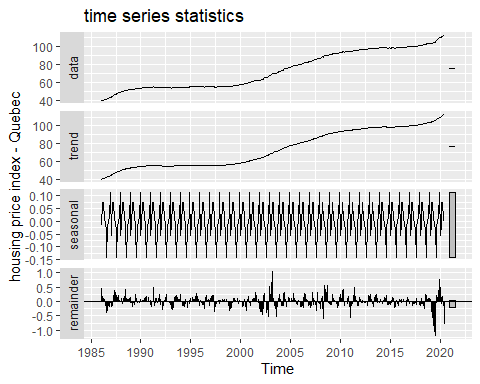
autoplot(cpt.meanvar(time\_series\_m4) , title = "sss \n dd" , xlab = "Years", ylab = "housing price index - Vancouver, British Columbia" ) # for city level 2



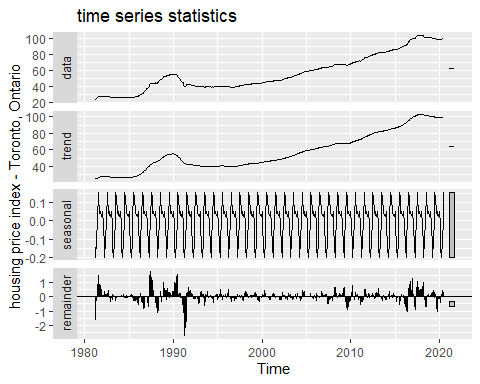
# Plotting time series statistics OK  
autoplot(stl(time\_series\_m1, s.window = 'periodic'), ts.colour = 'blue', ylab = "housing price index - Canada", main ="time series statistics") # for province level 1



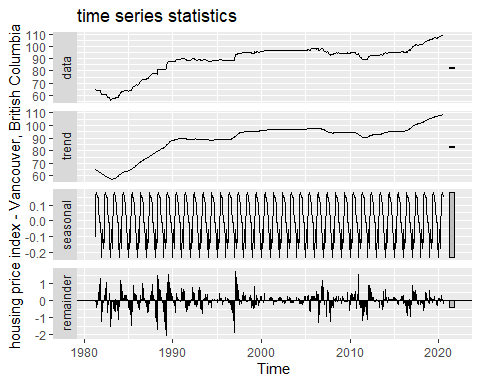
autoplot(stl(time\_series\_m2, s.window = 'periodic'), ts.colour = 'blue', ylab = "housing price index - Quebec", main ="time series statistics") # for province level 2



autoplot(stl(time\_series\_m3, s.window = 'periodic'), ts.colour = 'blue', ylab = "housing price index - Toronto, Ontario", main ="time series statistics") # for city level 1

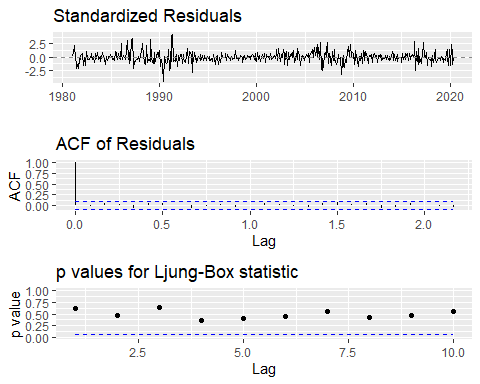


autoplot(stl(time\_series\_m4, s.window = 'periodic'), ts.colour = 'blue', ylab = "housing price index - Vancouver, British Columbia", main ="time series statistics") # for city level 2

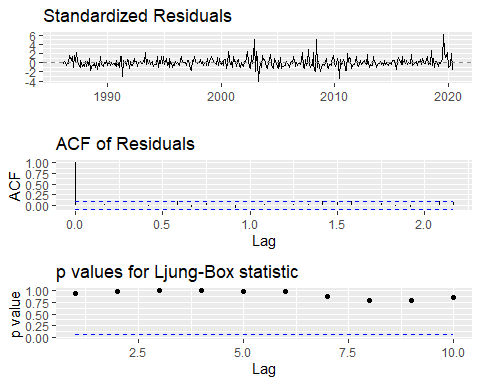


# Plots time-series diagnostics OK   
ggtsdiag(auto.arima(time\_series\_m1) , title = "sss \n dd") # # for province level 1

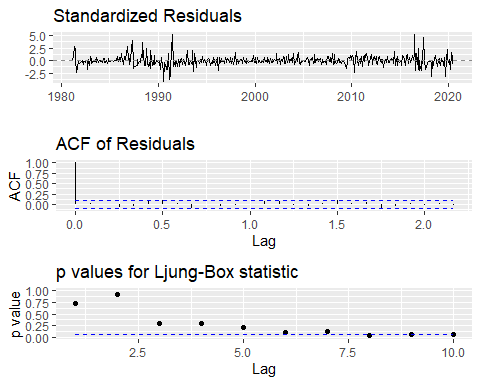
## Warning: `mutate\_()` is deprecated as of dplyr 0.7.0.  
## Please use `mutate()` instead.  
## See vignette('programming') for more help  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_warnings()` to see where this warning was generated.



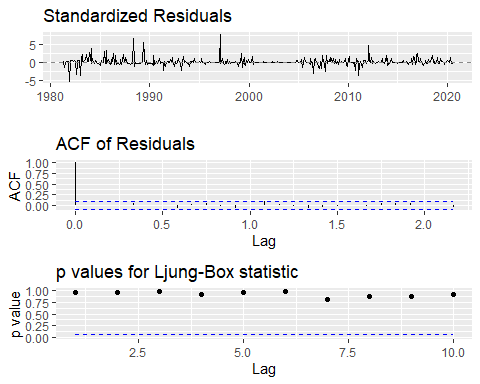
ggtsdiag(auto.arima(time\_series\_m2) , title = "sss \n dd" ) # # for province level 2



ggtsdiag(auto.arima(time\_series\_m3) , title = "sss \n dd" ) # # for city level 1



ggtsdiag(auto.arima(time\_series\_m4) , title = "sss \n dd" ) # # for city level 2

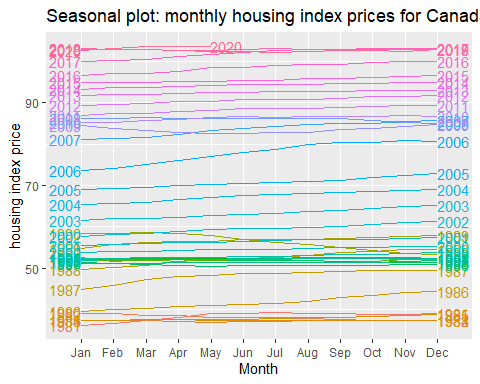


#--------------------------

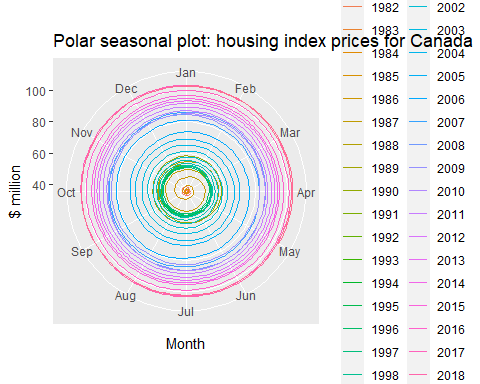
## Seasonal Plots

Seasonality refers to periodic fluctuations Now we will look at seasonal time plots, which will allows us to examine trends throughout the year, as opposed to over the years. We will only look at the monthly

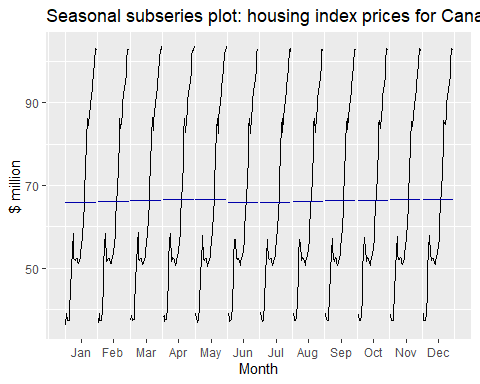
ggseasonplot(time\_series\_m1, year.labels=TRUE, year.labels.left=TRUE) +  
 ylab("housing index price") +  
 ggtitle("Seasonal plot: monthly housing index prices for Canada")



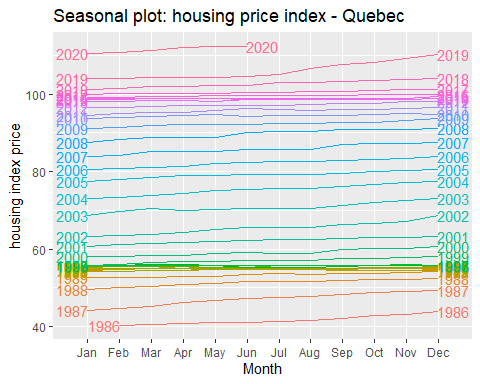
# The 2008 decline will make the price circles overlap alot around the outer part of the circle. This can be a bit hard to interpret, but in the right scenario it can be very useful.  
ggseasonplot(time\_series\_m1, polar=TRUE) +  
 ylab("$ million") +  
 ggtitle("Polar seasonal plot: housing index prices for Canada")



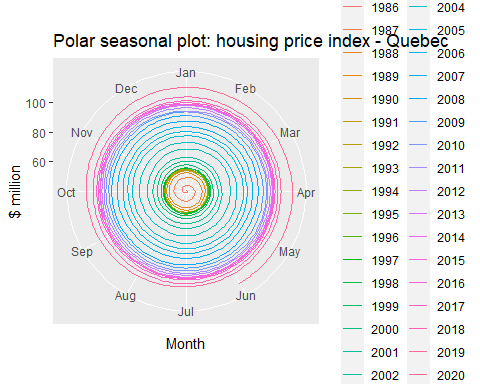
#A subseriesplot shows us the trends pertaining to each month  
# The trend we observe fits with the simple time series plots, as all of the months show steady increase, then decline around what I presume is 2008.  
#there is a clear Monthly seasonality  
ggsubseriesplot(time\_series\_m1) +  
 ylab("$ million") +  
 ggtitle("Seasonal subseries plot: housing index prices for Canada")



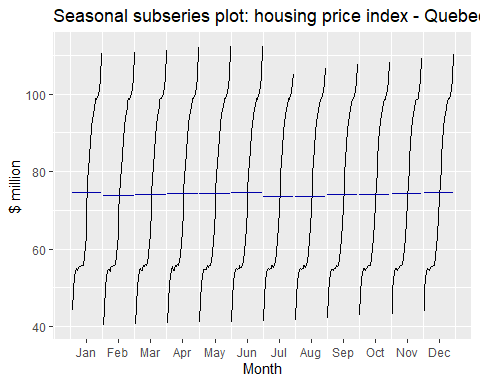
# for all quebec  
ggseasonplot(time\_series\_m2, year.labels=TRUE, year.labels.left=TRUE) +  
 ylab("housing index price") +  
 ggtitle("Seasonal plot: housing price index - Quebec")



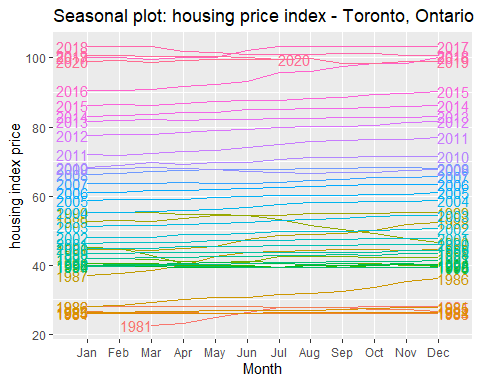
ggseasonplot(time\_series\_m2, polar=TRUE) +  
 ylab("$ million") +  
 ggtitle("Polar seasonal plot: housing price index - Quebec")



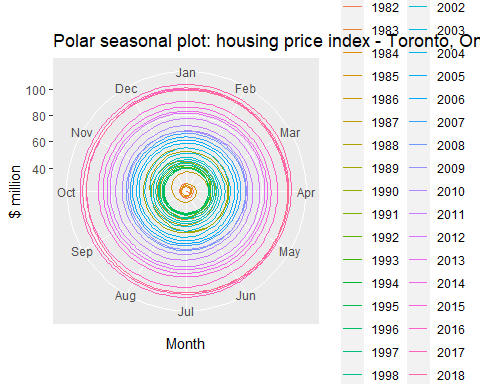
ggsubseriesplot(time\_series\_m2) +  
 ylab("$ million") +  
 ggtitle("Seasonal subseries plot: housing price index - Quebec")



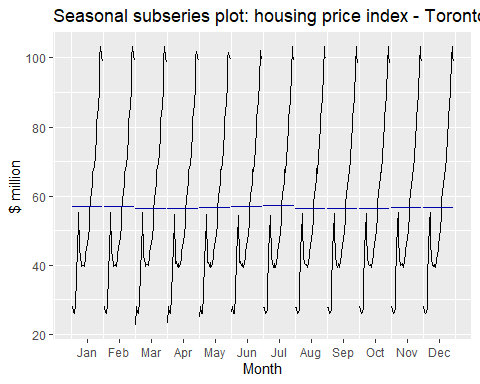
# for all toronto  
ggseasonplot(time\_series\_m3, year.labels=TRUE, year.labels.left=TRUE) +  
 ylab("housing index price") +  
 ggtitle("Seasonal plot: housing price index - Toronto, Ontario")



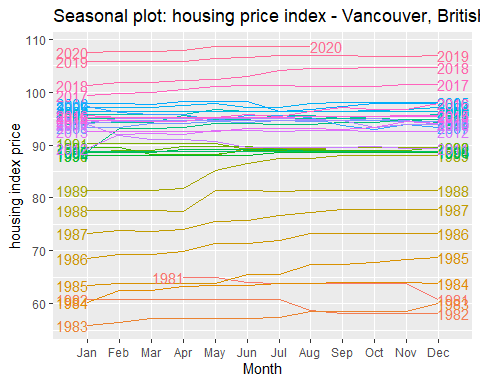
ggseasonplot(time\_series\_m3, polar=TRUE) +  
 ylab("$ million") +  
 ggtitle("Polar seasonal plot: housing price index - Toronto, Ontario")



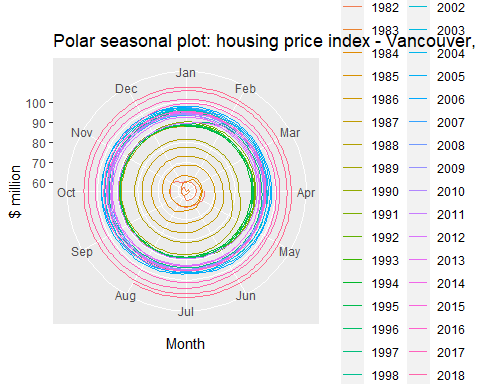
ggsubseriesplot(time\_series\_m3) +  
 ylab("$ million") +  
 ggtitle("Seasonal subseries plot: housing price index - Toronto, Ontario")



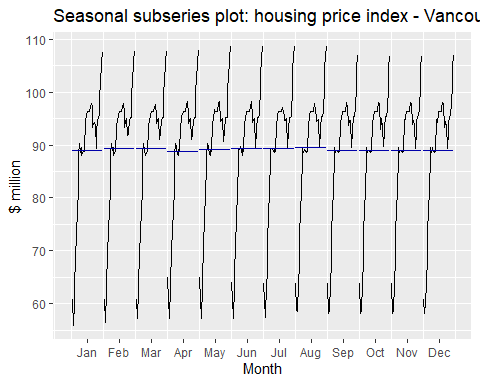
# for all toronto  
ggseasonplot(time\_series\_m4, year.labels=TRUE, year.labels.left=TRUE) +  
 ylab("housing index price") +  
 ggtitle("Seasonal plot: housing price index - Vancouver, British Columbia")



ggseasonplot(time\_series\_m4, polar=TRUE) +  
 ylab("$ million") +  
 ggtitle("Polar seasonal plot: housing price index - Vancouver, British Columbia")



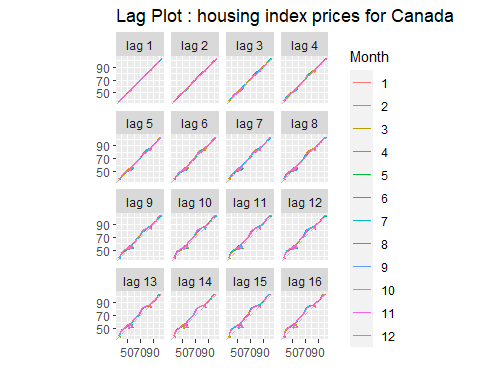
ggsubseriesplot(time\_series\_m4) +  
 ylab("$ million") +  
 ggtitle("Seasonal subseries plot: housing price index - Vancouver, British Columbia")



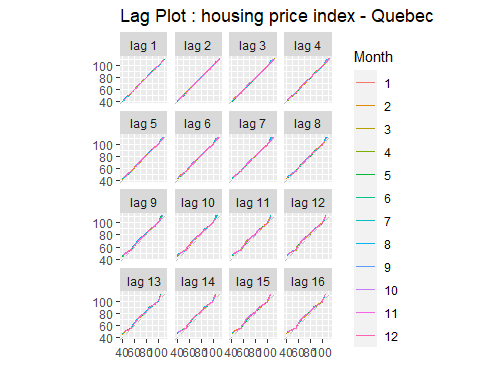
## Lag plots

A lag plot is used to help evaluate whether the values in a dataset or time series are random. If the data are random, the lag plot will exhibit no identifiable pattern. … The type of pattern can aid the user in identifying the non-random structure in the data. Lag plots can also help to identify outliers.

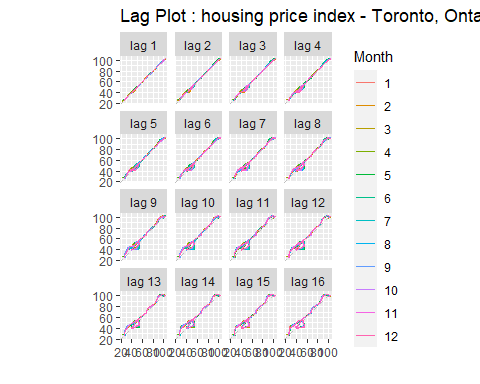
gglagplot(time\_series\_m1, main= "Lag Plot : housing index prices for Canada") # for province level 1



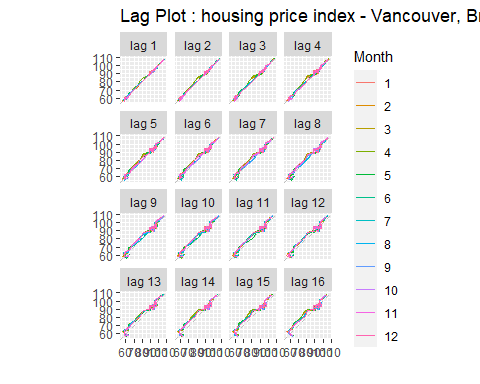
gglagplot(time\_series\_m2, main= "Lag Plot : housing price index - Quebec") # for province level 2



gglagplot(time\_series\_m3, main= "Lag Plot : housing price index - Toronto, Ontario") # for city level 1



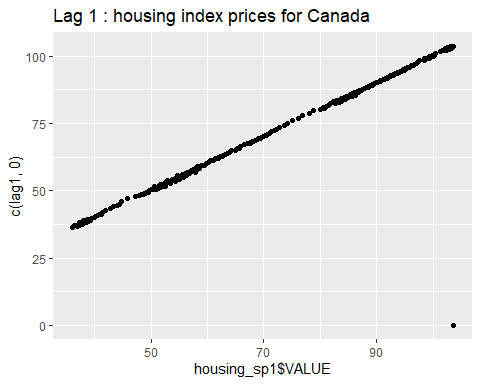
gglagplot(time\_series\_m4, main= "Lag Plot : housing price index - Vancouver, British Columbia") # for city level 2



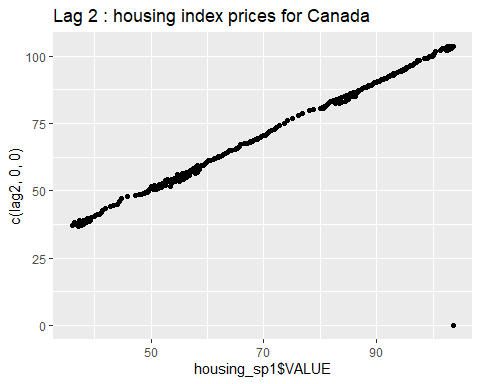
## Autocorrelation

Autocorrelation measures the linear relationship of a variable and its lagged value. Lag order one would be between yt and yt−1, lag order two would be between yt and yt−2, and on…

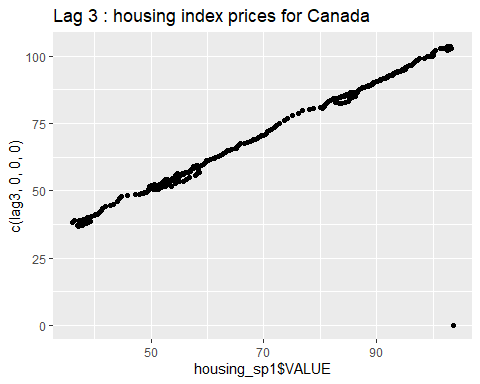
# To start we create a lag variable of order one. We use the base R tail function which will give us a vector with all but the last observation, which is yt−1. We then plot this against the original variable with qplot. Under y = we need to append 0 to the end of our lag so that the vectors will have the same length.  
  
lag1 <- tail(housing\_sp1$VALUE,-1) # last part of dataset of province level 1  
qplot(x=housing\_sp1$VALUE,y=c(lag1,0), main = "Lag 1 : housing index prices for Canada" )



# Here we do the same thing, but for lag order two, for this we just change -1 to -2 in the tail function and append an additional 0 when plotting.  
  
lag2 <- tail(housing\_sp1$VALUE,-2)  
qplot(x=housing\_sp1$VALUE,y=c(lag2,0,0), main = "Lag 2 : housing index prices for Canada" )

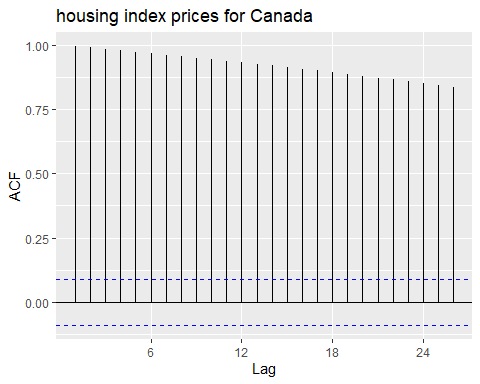


lag3 <- tail(housing\_sp1$VALUE,-3)  
qplot(x=housing\_sp1$VALUE,y=c(lag3,0,0,0), main = "Lag 3 : housing index prices for Canada" )

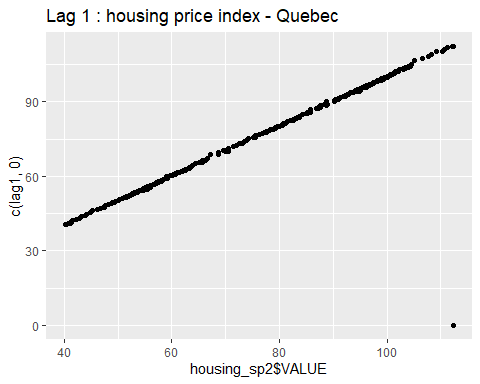


# We observe that both of the above plots have auto correlation values of approximately 1 due to the strong positive linear relationship  
  
# We can see these auto correlation values for many lag variable with the ggAcf function. All we need is to input our ts object  
ggAcf(time\_series\_m1, main = "housing index prices for Canada" )

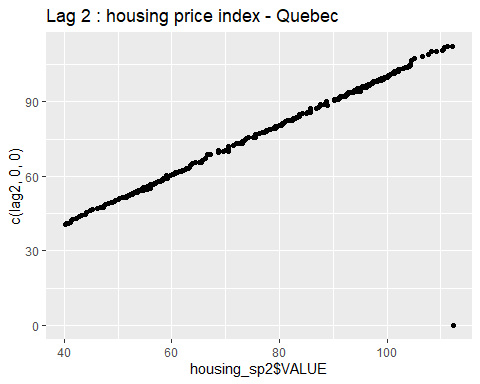
## Warning: Ignoring unknown parameters: main



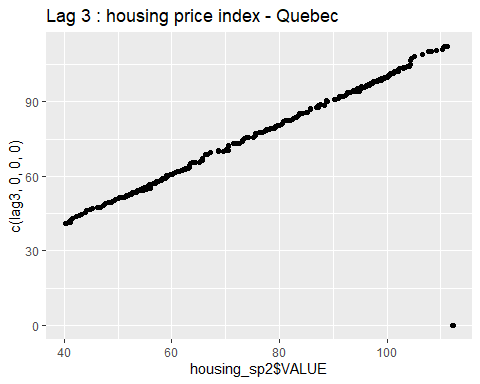
#It is important to note that in the case of no auto correlation, 95 percent of the vertical lines will be within the dotted blues lines.  
  
  
lag1 <- tail(housing\_sp2$VALUE,-1) # last part of dataset of province level 2  
qplot(x=housing\_sp2$VALUE,y=c(lag1,0), main = "Lag 1 : housing price index - Quebec" )



lag2 <- tail(housing\_sp2$VALUE,-2)  
qplot(x=housing\_sp2$VALUE,y=c(lag2,0,0), main = "Lag 2 : housing price index - Quebec" )

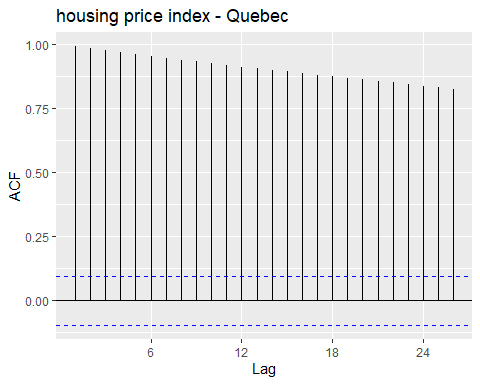


lag3 <- tail(housing\_sp2$VALUE,-3)  
qplot(x=housing\_sp2$VALUE,y=c(lag3,0,0,0), main = "Lag 3 : housing price index - Quebec" )

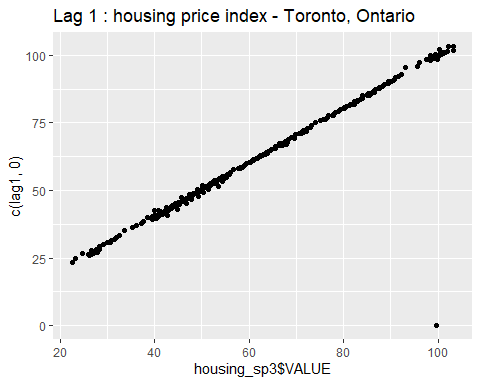


ggAcf(time\_series\_m2, main = "housing price index - Quebec" )

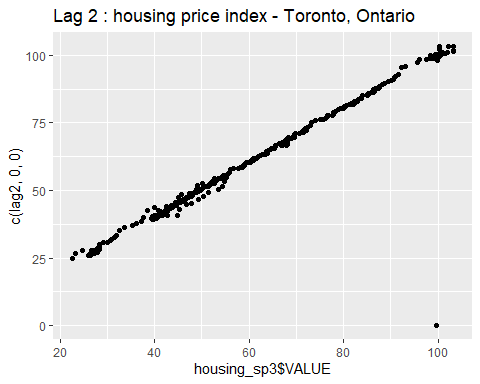
## Warning: Ignoring unknown parameters: main



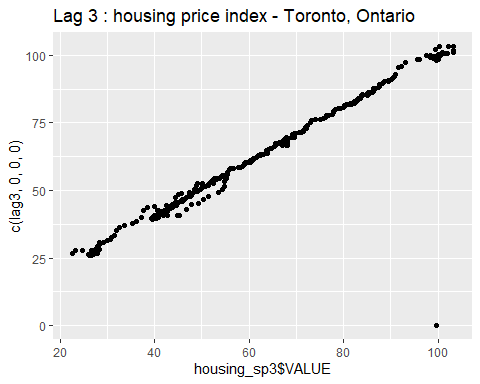
lag1 <- tail(housing\_sp3$VALUE,-1) # last part of dataset of city level 1  
qplot(x=housing\_sp3$VALUE,y=c(lag1,0), main = "Lag 1 : housing price index - Toronto, Ontario" )



lag2 <- tail(housing\_sp3$VALUE,-2)  
qplot(x=housing\_sp3$VALUE,y=c(lag2,0,0), main = "Lag 2 : housing price index - Toronto, Ontario" )

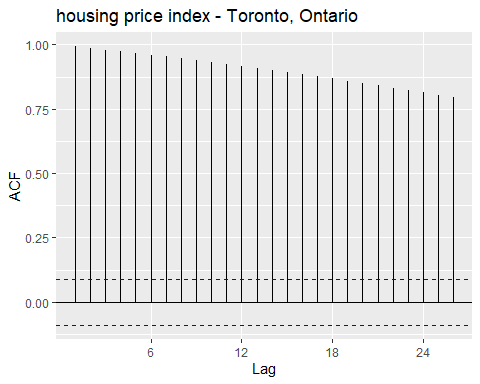


lag3 <- tail(housing\_sp3$VALUE,-3)  
qplot(x=housing\_sp3$VALUE,y=c(lag3,0,0,0), main = "Lag 3 : housing price index - Toronto, Ontario" )

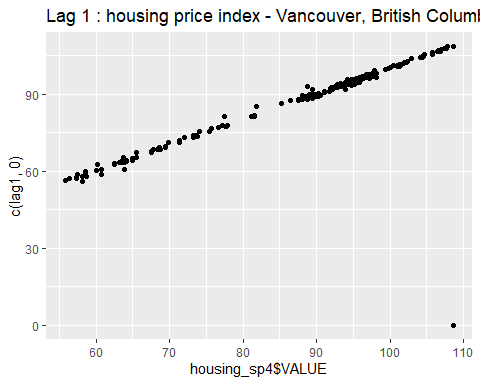


ggAcf(time\_series\_m3, main = "housing price index - Toronto, Ontario" )

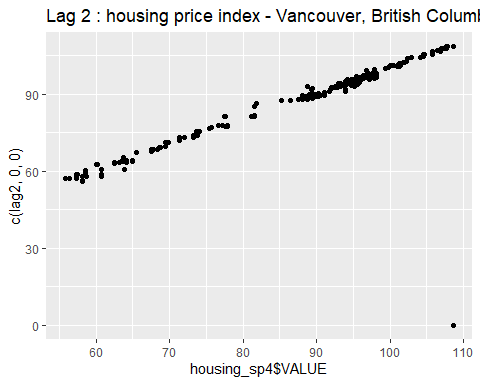
## Warning: Ignoring unknown parameters: main



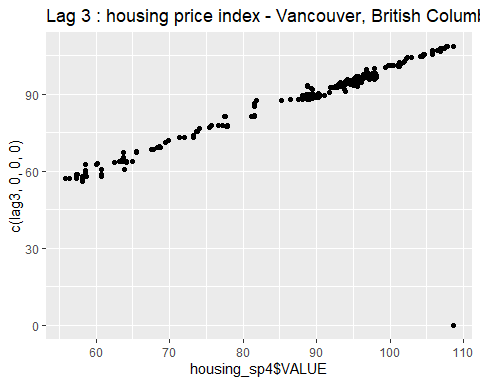
lag1 <- tail(housing\_sp4$VALUE,-1) # last part of dataset of city level 2  
qplot(x=housing\_sp4$VALUE,y=c(lag1,0), main = "Lag 1 : housing price index - Vancouver, British Columbia" )



lag2 <- tail(housing\_sp4$VALUE,-2)  
qplot(x=housing\_sp4$VALUE,y=c(lag2,0,0), main = "Lag 2 : housing price index - Vancouver, British Columbia" )

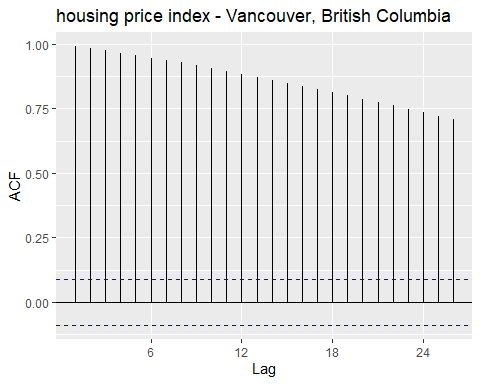


lag3 <- tail(housing\_sp4$VALUE,-3)  
qplot(x=housing\_sp4$VALUE,y=c(lag3,0,0,0), main = "Lag 3 : housing price index - Vancouver, British Columbia" )



ggAcf(time\_series\_m4, main = "housing price index - Vancouver, British Columbia" )

## Warning: Ignoring unknown parameters: main

 ## Predection with ARIMA For all time serieses we selectd

linear regression models in non-stationary data

head (All, 5) # top 5 rows

## time\_series\_m1 time\_series\_m2 time\_series\_m3 time\_series\_m4  
## Jan 1981 36.1 NA NA NA  
## Feb 1981 36.5 NA NA NA  
## Mar 1981 37.3 NA 22.6 NA  
## Apr 1981 38.1 NA 23.2 64.9  
## May 1981 38.9 NA 24.8 64.9

colnames(All) <- paste0(cases) # assign name of column  
head (All, 5) # top 5 rows

## Canada Quebec Toronto, Ontario Windsor, Ontario  
## Jan 1981 36.1 NA NA NA  
## Feb 1981 36.5 NA NA NA  
## Mar 1981 37.3 NA 22.6 NA  
## Apr 1981 38.1 NA 23.2 64.9  
## May 1981 38.9 NA 24.8 64.9

All <- na.remove(All) # remove na  
head (All, 5) # top 5 rows

## Canada Quebec Toronto, Ontario Windsor, Ontario  
## Feb 1986 39.9 40.3 28.4 69.4  
## Mar 1986 40.2 40.5 29.2 69.4  
## Apr 1986 40.6 40.8 30.0 69.8  
## May 1986 40.9 41.1 30.6 71.4  
## Jun 1986 41.2 41.2 30.8 71.4  
## attr(,"na.removed")  
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19  
## [20] 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38  
## [39] 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57  
## [58] 58 59 60 61 474 475 476

# VARselect() enables the to determine an optimal lag length  
VARselect(All, lag.max = 5, type = "const")

## $selection  
## AIC(n) HQ(n) SC(n) FPE(n)   
## 5 4 4 5   
##   
## $criteria  
## 1 2 3 4 5  
## AIC(n) -7.9495203157 -8.6968552821 -9.513964e+00 -9.712222e+00 -9.725549e+00  
## HQ(n) -7.8715618630 -8.5565300673 -9.311273e+00 -9.447164e+00 -9.398123e+00  
## SC(n) -7.7525270388 -8.3422673836 -9.001782e+00 -9.042445e+00 -8.898177e+00  
## FPE(n) 0.0003528331 0.0001671153 7.382025e-05 6.055079e-05 5.975953e-05

var.2c <- VAR(All, p = 2, type = "const") # generate estimation results  
var.2c # show estimation results

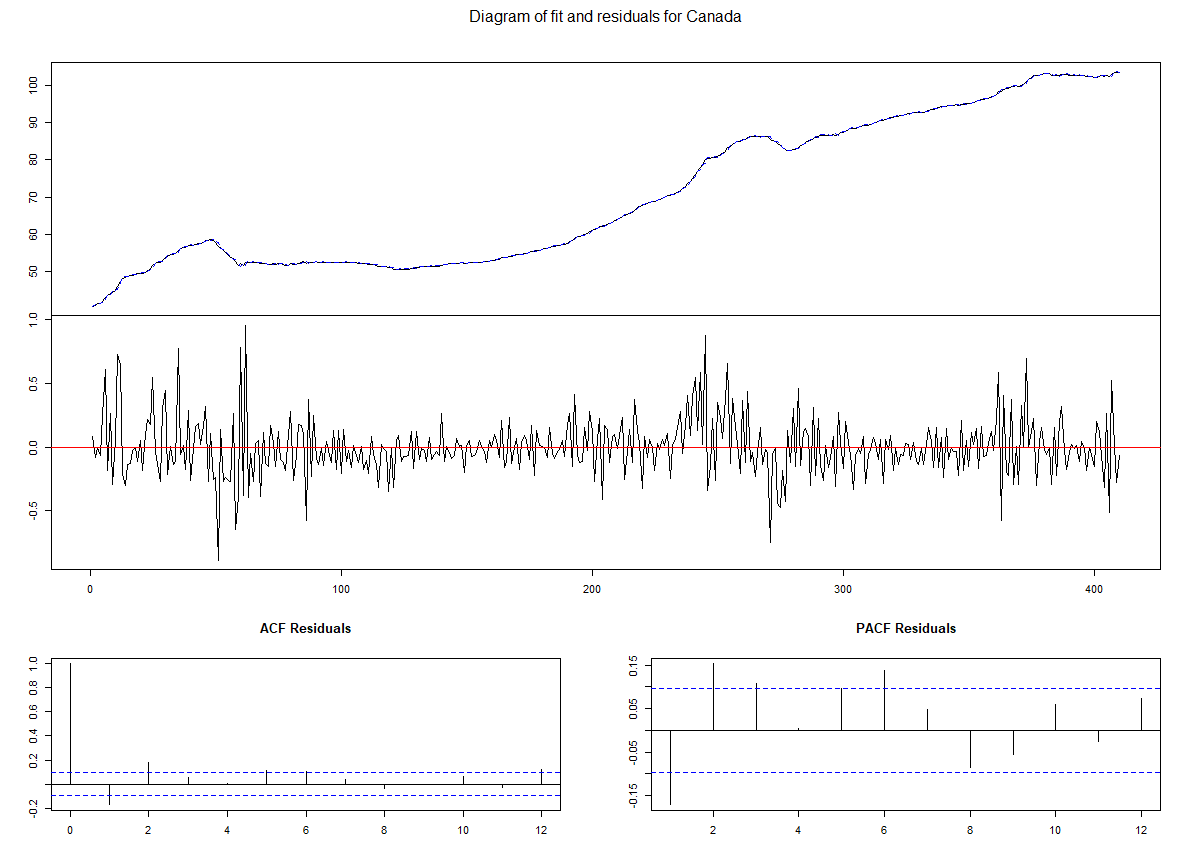
##   
## VAR Estimation Results:  
## =======================   
##   
## Estimated coefficients for equation Canada:   
## ===========================================   
## Call:  
## Canada = Canada.l1 + Quebec.l1 + Toronto..Ontario.l1 + Windsor..Ontario.l1 + Canada.l2 + Quebec.l2 + Toronto..Ontario.l2 + Windsor..Ontario.l2 + const   
##   
## Canada.l1 Quebec.l1 Toronto..Ontario.l1 Windsor..Ontario.l1   
## 1.548238856 0.057726683 0.074385937 -0.002892160   
## Canada.l2 Quebec.l2 Toronto..Ontario.l2 Windsor..Ontario.l2   
## -0.565606202 -0.044395590 -0.069982379 -0.001284579   
## const   
## 0.402326816   
##   
##   
## Estimated coefficients for equation Quebec:   
## ===========================================   
## Call:  
## Quebec = Canada.l1 + Quebec.l1 + Toronto..Ontario.l1 + Windsor..Ontario.l1 + Canada.l2 + Quebec.l2 + Toronto..Ontario.l2 + Windsor..Ontario.l2 + const   
##   
## Canada.l1 Quebec.l1 Toronto..Ontario.l1 Windsor..Ontario.l1   
## 0.29420537 1.20023641 -0.01331229 0.03223788   
## Canada.l2 Quebec.l2 Toronto..Ontario.l2 Windsor..Ontario.l2   
## -0.29477535 -0.20121076 0.01441108 -0.02786271   
## const   
## -0.26647729   
##   
##   
## Estimated coefficients for equation Toronto..Ontario:   
## =====================================================   
## Call:  
## Toronto..Ontario = Canada.l1 + Quebec.l1 + Toronto..Ontario.l1 + Windsor..Ontario.l1 + Canada.l2 + Quebec.l2 + Toronto..Ontario.l2 + Windsor..Ontario.l2 + const   
##   
## Canada.l1 Quebec.l1 Toronto..Ontario.l1 Windsor..Ontario.l1   
## 0.40504347 0.15278974 1.32493830 0.04673003   
## Canada.l2 Quebec.l2 Toronto..Ontario.l2 Windsor..Ontario.l2   
## -0.40291708 -0.14584691 -0.32974091 -0.06262533   
## const   
## 1.12395049   
##   
##   
## Estimated coefficients for equation Windsor..Ontario:   
## =====================================================   
## Call:  
## Windsor..Ontario = Canada.l1 + Quebec.l1 + Toronto..Ontario.l1 + Windsor..Ontario.l1 + Canada.l2 + Quebec.l2 + Toronto..Ontario.l2 + Windsor..Ontario.l2 + const   
##   
## Canada.l1 Quebec.l1 Toronto..Ontario.l1 Windsor..Ontario.l1   
## -0.036038402 0.146737939 0.084376920 1.018699901   
## Canada.l2 Quebec.l2 Toronto..Ontario.l2 Windsor..Ontario.l2   
## 0.005692318 -0.141995993 -0.056893171 -0.034858763   
## const   
## 1.673878736

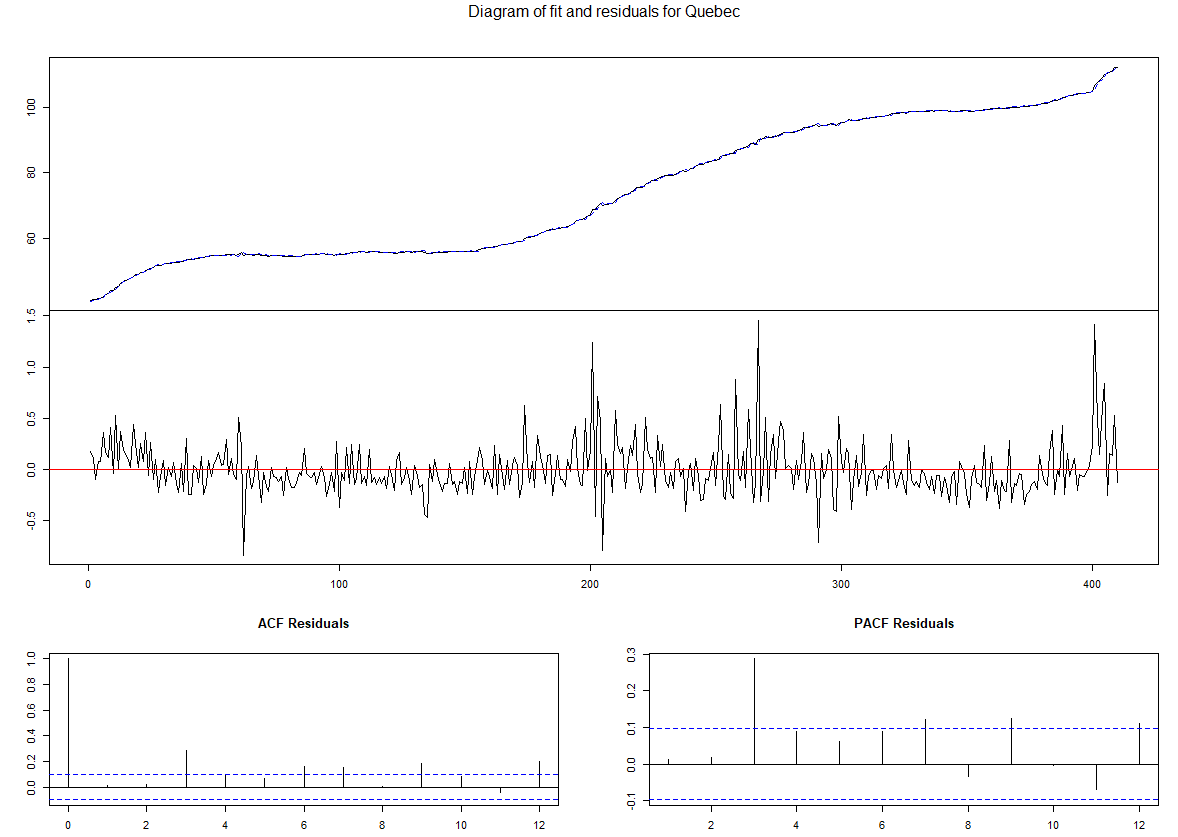
names(var.2c) # show the funcation name

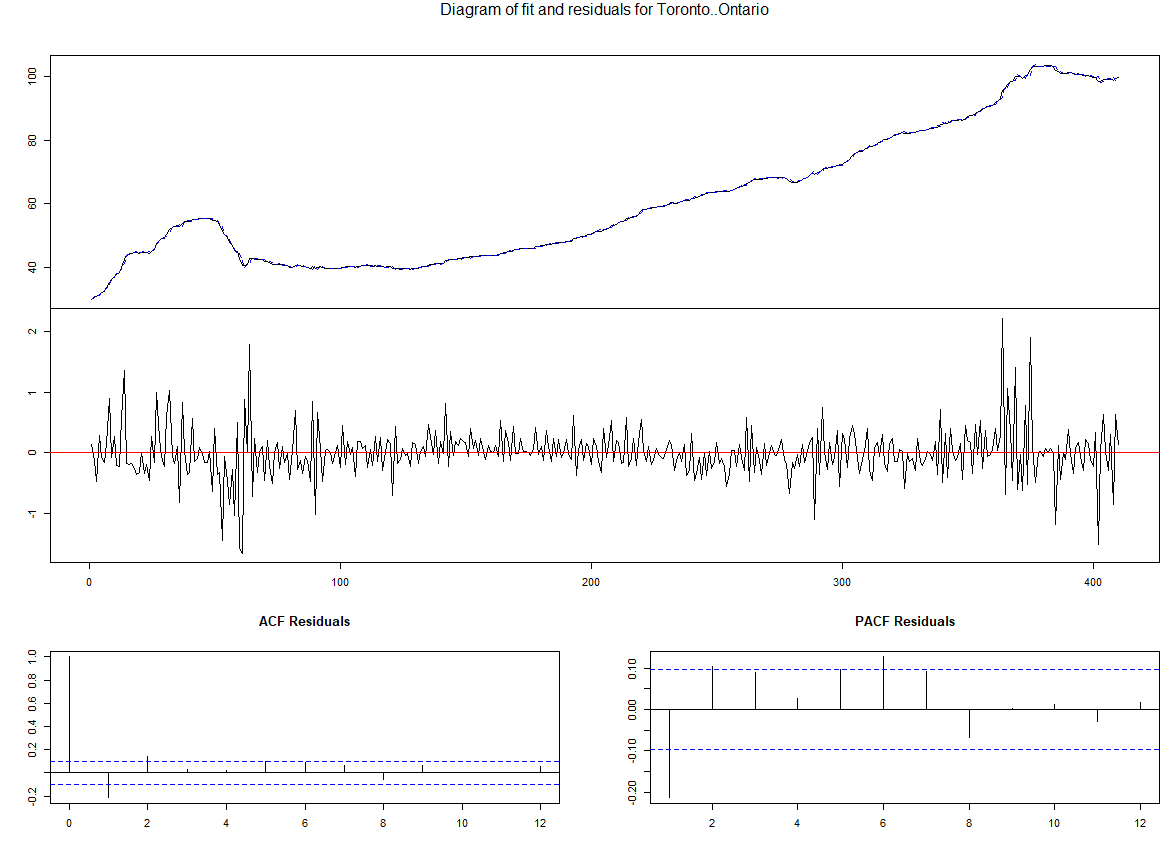
## [1] "varresult" "datamat" "y" "type" "p"   
## [6] "K" "obs" "totobs" "restrictions" "call"

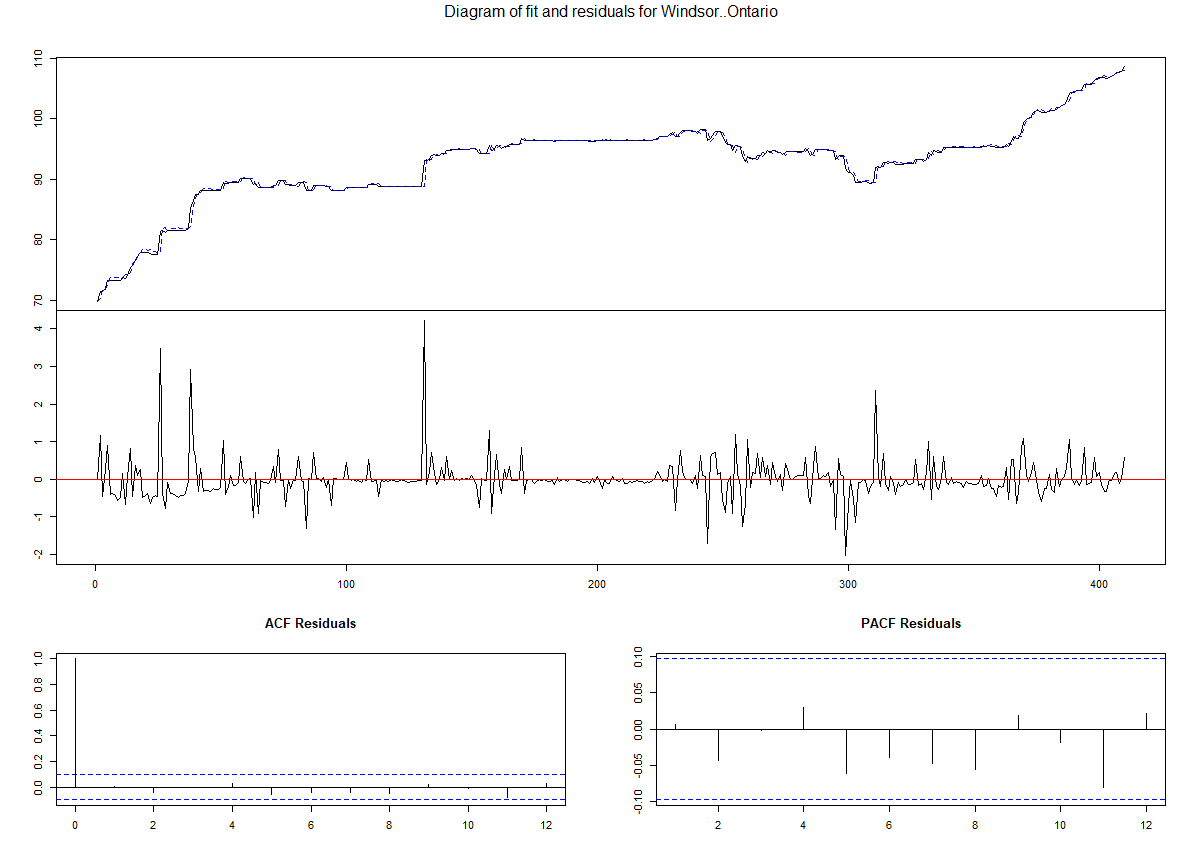
# The function returns a list object with the optimal lag-order according to each of the criteria  
summary(var.2c)

##   
## VAR Estimation Results:  
## =========================   
## Endogenous variables: Canada, Quebec, Toronto..Ontario, Windsor..Ontario   
## Deterministic variables: const   
## Sample size: 410   
## Log Likelihood: -506.506   
## Roots of the characteristic polynomial:  
## 1.001 0.99 0.99 0.9693 0.6978 0.2338 0.2338 0.0006977  
## Call:  
## VAR(y = All, p = 2, type = "const")  
##   
##   
## Estimation results for equation Canada:   
## =======================================   
## Canada = Canada.l1 + Quebec.l1 + Toronto..Ontario.l1 + Windsor..Ontario.l1 + Canada.l2 + Quebec.l2 + Toronto..Ontario.l2 + Windsor..Ontario.l2 + const   
##   
## Estimate Std. Error t value Pr(>|t|)   
## Canada.l1 1.548239 0.042651 36.300 < 2e-16 \*\*\*  
## Quebec.l1 0.057727 0.042538 1.357 0.17552   
## Toronto..Ontario.l1 0.074386 0.025855 2.877 0.00423 \*\*   
## Windsor..Ontario.l1 -0.002892 0.022411 -0.129 0.89738   
## Canada.l2 -0.565606 0.042761 -13.227 < 2e-16 \*\*\*  
## Quebec.l2 -0.044396 0.042731 -1.039 0.29945   
## Toronto..Ontario.l2 -0.069982 0.025944 -2.697 0.00728 \*\*   
## Windsor..Ontario.l2 -0.001285 0.022181 -0.058 0.95385   
## const 0.402327 0.217670 1.848 0.06529 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
##   
## Residual standard error: 0.2313 on 401 degrees of freedom  
## Multiple R-Squared: 0.9999, Adjusted R-squared: 0.9999   
## F-statistic: 3.69e+05 on 8 and 401 DF, p-value: < 2.2e-16   
##   
##   
## Estimation results for equation Quebec:   
## =======================================   
## Quebec = Canada.l1 + Quebec.l1 + Toronto..Ontario.l1 + Windsor..Ontario.l1 + Canada.l2 + Quebec.l2 + Toronto..Ontario.l2 + Windsor..Ontario.l2 + const   
##   
## Estimate Std. Error t value Pr(>|t|)   
## Canada.l1 0.29421 0.04718 6.235 1.14e-09 \*\*\*  
## Quebec.l1 1.20024 0.04706 25.505 < 2e-16 \*\*\*  
## Toronto..Ontario.l1 -0.01331 0.02860 -0.465 0.642   
## Windsor..Ontario.l1 0.03224 0.02479 1.300 0.194   
## Canada.l2 -0.29478 0.04731 -6.231 1.17e-09 \*\*\*  
## Quebec.l2 -0.20121 0.04727 -4.256 2.59e-05 \*\*\*  
## Toronto..Ontario.l2 0.01441 0.02870 0.502 0.616   
## Windsor..Ontario.l2 -0.02786 0.02454 -1.135 0.257   
## const -0.26648 0.24080 -1.107 0.269   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
##   
## Residual standard error: 0.2558 on 401 degrees of freedom  
## Multiple R-Squared: 0.9998, Adjusted R-squared: 0.9998   
## F-statistic: 3.263e+05 on 8 and 401 DF, p-value: < 2.2e-16   
##   
##   
## Estimation results for equation Toronto..Ontario:   
## =================================================   
## Toronto..Ontario = Canada.l1 + Quebec.l1 + Toronto..Ontario.l1 + Windsor..Ontario.l1 + Canada.l2 + Quebec.l2 + Toronto..Ontario.l2 + Windsor..Ontario.l2 + const   
##   
## Estimate Std. Error t value Pr(>|t|)   
## Canada.l1 0.40504 0.07582 5.342 1.54e-07 \*\*\*  
## Quebec.l1 0.15279 0.07562 2.020 0.04400 \*   
## Toronto..Ontario.l1 1.32494 0.04596 28.826 < 2e-16 \*\*\*  
## Windsor..Ontario.l1 0.04673 0.03984 1.173 0.24153   
## Canada.l2 -0.40292 0.07602 -5.300 1.91e-07 \*\*\*  
## Quebec.l2 -0.14585 0.07596 -1.920 0.05558 .   
## Toronto..Ontario.l2 -0.32974 0.04612 -7.149 4.15e-12 \*\*\*  
## Windsor..Ontario.l2 -0.06263 0.03943 -1.588 0.11303   
## const 1.12395 0.38696 2.905 0.00388 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
##   
## Residual standard error: 0.4111 on 401 degrees of freedom  
## Multiple R-Squared: 0.9996, Adjusted R-squared: 0.9996   
## F-statistic: 1.291e+05 on 8 and 401 DF, p-value: < 2.2e-16   
##   
##   
## Estimation results for equation Windsor..Ontario:   
## =================================================   
## Windsor..Ontario = Canada.l1 + Quebec.l1 + Toronto..Ontario.l1 + Windsor..Ontario.l1 + Canada.l2 + Quebec.l2 + Toronto..Ontario.l2 + Windsor..Ontario.l2 + const   
##   
## Estimate Std. Error t value Pr(>|t|)   
## Canada.l1 -0.036038 0.094934 -0.380 0.704431   
## Quebec.l1 0.146738 0.094681 1.550 0.121975   
## Toronto..Ontario.l1 0.084377 0.057548 1.466 0.143378   
## Windsor..Ontario.l1 1.018700 0.049883 20.422 < 2e-16 \*\*\*  
## Canada.l2 0.005692 0.095178 0.060 0.952339   
## Quebec.l2 -0.141996 0.095111 -1.493 0.136238   
## Toronto..Ontario.l2 -0.056893 0.057746 -0.985 0.325103   
## Windsor..Ontario.l2 -0.034859 0.049370 -0.706 0.480555   
## const 1.673879 0.484494 3.455 0.000609 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
##   
## Residual standard error: 0.5147 on 401 degrees of freedom  
## Multiple R-Squared: 0.9943, Adjusted R-squared: 0.9942   
## F-statistic: 8786 on 8 and 401 DF, p-value: < 2.2e-16   
##   
##   
##   
## Covariance matrix of residuals:  
## Canada Quebec Toronto..Ontario Windsor..Ontario  
## Canada 0.053478 0.001497 0.0103121 -0.0105014  
## Quebec 0.001497 0.065450 -0.0078376 -0.0092195  
## Toronto..Ontario 0.010312 -0.007838 0.1690116 -0.0005901  
## Windsor..Ontario -0.010501 -0.009219 -0.0005901 0.2649462  
##   
## Correlation matrix of residuals:  
## Canada Quebec Toronto..Ontario Windsor..Ontario  
## Canada 1.00000 0.02530 0.108468 -0.088222  
## Quebec 0.02530 1.00000 -0.074520 -0.070012  
## Toronto..Ontario 0.10847 -0.07452 1.000000 -0.002789  
## Windsor..Ontario -0.08822 -0.07001 -0.002789 1.000000

plot(var.2c)  


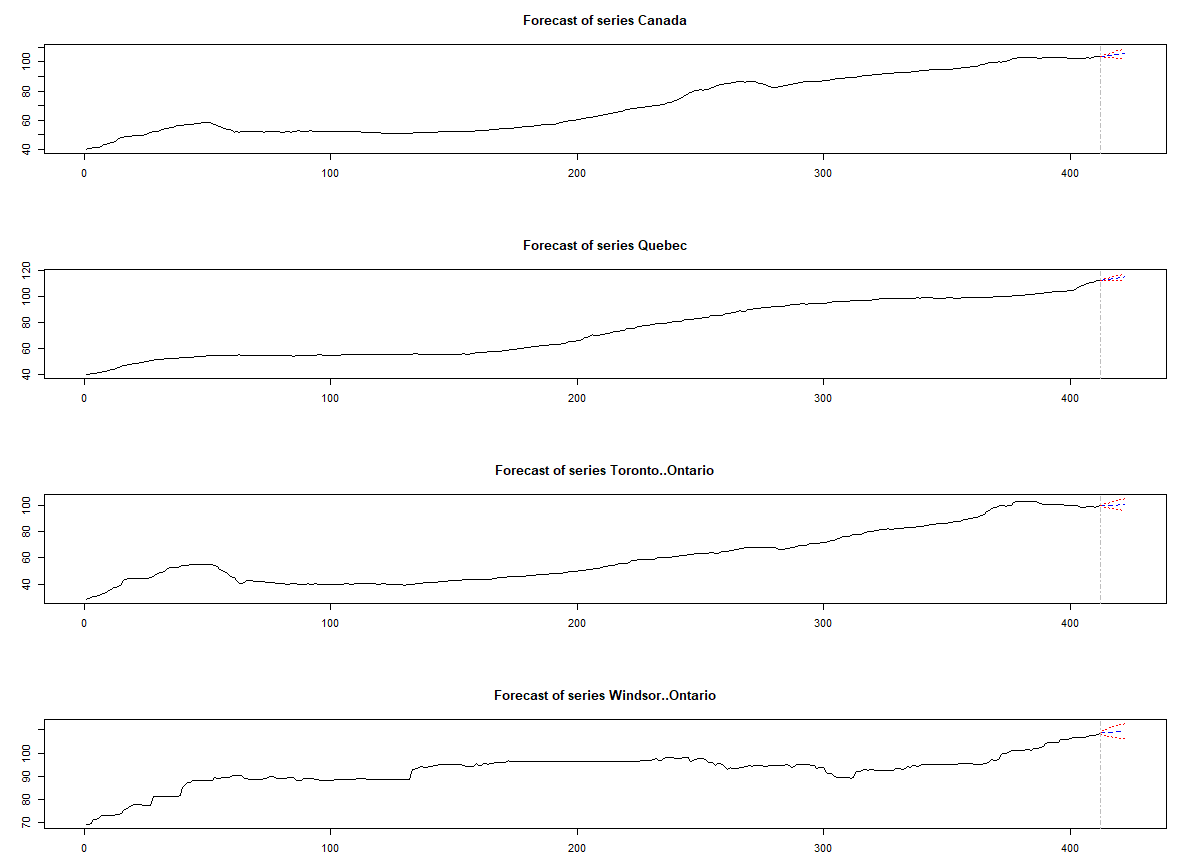




  
# forecasting 10 steps ahead  
#estimated with the function VAR() and as deterministic regresses a constant is included  
var.f10 <- predict(var.2c, n.ahead = 10, ci = 0.95)  
  
names(var.f10) # show object name

## [1] "fcst" "endog" "model" "exo.fcst"

plot(var.f10)



fanchart(var.f10)

