PS 4

Xiang (Cyllia) Li

2024-02-20

source("create\_monthly\_price.R")  
source("model\_selection\_function.R") # function for model selection  
source("ols\_function.R")  
source("t\_test\_function.R")   
source("generate\_data\_functions.R")  
source("expanding\_window\_forecast\_function.R") # function for forecasting using expanding windows  
source("rolling\_window\_forecast\_function.R") # function for forecasting using rolling windows  
library("quantmod")

# Fetch Data

getSymbols(Symbols ="GNP",src = "FRED",warnings = FALSE)

## [1] "GNP"

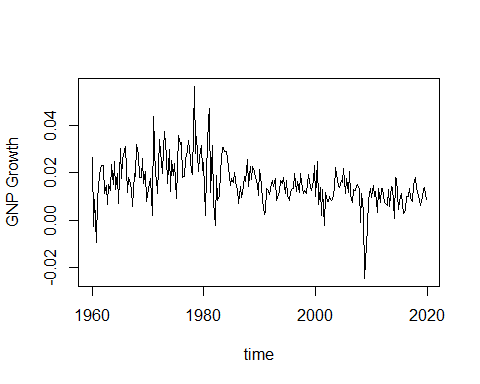
Y <- as.matrix(GNP[,1])  
DY <- diff(Y)/Y[1:(dim(Y)[1]-1),]  
head(DY)

## GNP  
## 1947-04-01 0.01196435  
## 1947-07-01 0.01478570  
## 1947-10-01 0.04094274  
## 1948-01-01 0.02357260  
## 1948-04-01 0.02587849  
## 1948-07-01 0.02420397

tail(DY)

## GNP  
## 2022-04-01 0.022580326  
## 2022-07-01 0.017074777  
## 2022-10-01 0.015584918  
## 2023-01-01 0.014233663  
## 2023-04-01 0.009771869  
## 2023-07-01 0.019756463

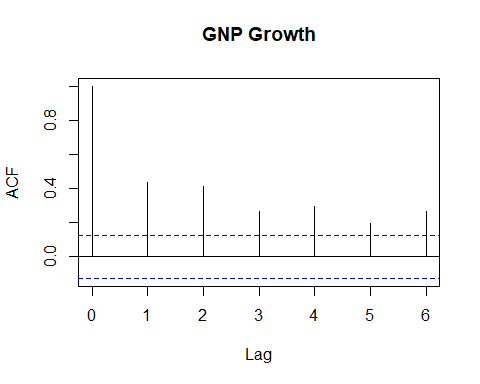
keep\_data <- seq(from = as.Date("1960-01-01"), to = as.Date("2019-10-1"), by = "quarter")  
Y\_new = as.matrix(Y[as.Date(rownames(Y)) %in% keep\_data,]) # precovid data  
DY\_new = as.matrix(DY[as.Date(rownames(DY)) %in% keep\_data,])  
colnames(DY\_new) = "GNP Growth"  
n\_obs = dim(DY\_new)[1]  
DY\_new\_date = as.Date(row.names(DY\_new))  
  
plot(x = DY\_new\_date, y = DY\_new,xlab='time',ylab='GNP Growth',type='l',col="black")



basicStats(DY\_new)

## GNP.Growth  
## nobs 240.000000  
## NAs 0.000000  
## Minimum -0.024656  
## Maximum 0.056421  
## 1. Quartile 0.010223  
## 3. Quartile 0.020117  
## Mean 0.015709  
## Median 0.014341  
## Sum 3.770248  
## SE Mean 0.000617  
## LCL Mean 0.014494  
## UCL Mean 0.016925  
## Variance 0.000091  
## Stdev 0.009560  
## Skewness 0.368624  
## Kurtosis 2.576997

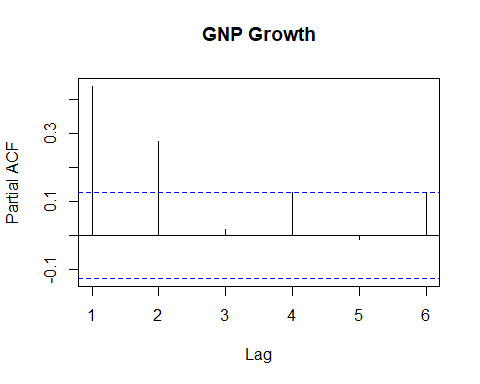
acf(DY\_new,lag=round(n\_obs^(1/3))) # command to obtain sample ACF of the data



Box.test(DY\_new, lag = round(n\_obs^(1/3)), type = "Ljung-Box") # applying Ljung and Box (1978) joint test of auto correlations

##   
## Box-Ljung test  
##   
## data: DY\_new  
## X-squared = 154.78, df = 6, p-value < 2.2e-16

pacf(DY\_new,lag=round(n\_obs^(1/3)),main="GNP Growth") # command to obtain sample PACF of the data



# select the number of lags and model checking

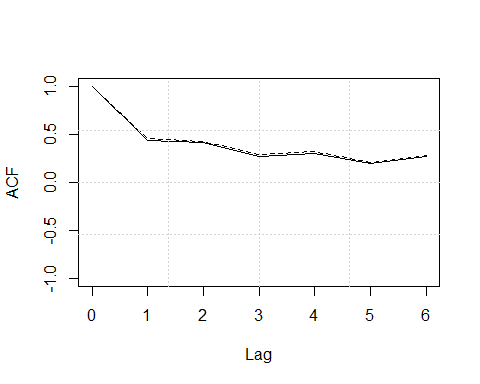
results <- model\_selection(round(n\_obs^(1/3)),DY\_new)  
aic\_values = results$AIC  
bic\_values = results$BIC  
num\_lags\_aic = results$op\_lag\_AIC   
num\_lags\_bic = results$op\_lag\_BIC  
num\_lags\_aic

## [1] 6

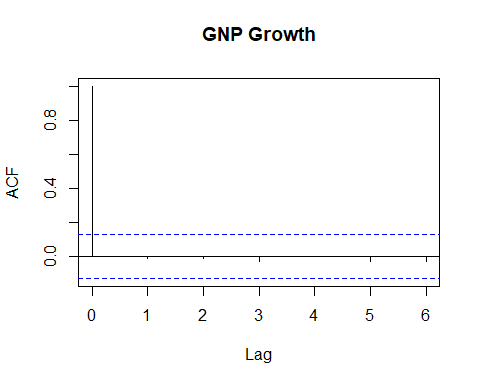
num\_lags\_bic

## [1] 2

num\_lags = num\_lags\_aic  
lags\_DY\_new = matrix(NA,nrow = n\_obs, ncol = num\_lags)  
for (i in 1:num\_lags) {  
 lags\_DY\_new[(i+1):n\_obs,i] = as.matrix(DY\_new[1:(n\_obs-i),1])  
}  
intercept = matrix(1,n\_obs)  
X = cbind(intercept,lags\_DY\_new)  
y = DY\_new  
reg\_result = ols(X[(num\_lags+1):n\_obs,],as.matrix(y[(num\_lags+1):n\_obs,1]))  
beta\_hat = reg\_result$beta\_hat  
  
ar\_coeff <- as.numeric(beta\_hat[2:(num\_lags+1)])  
ma\_coeff <- 0  
ACF = acf(DY\_new,lag=round(n\_obs^(1/3)),plot = FALSE) # command to obtain sample ACF of the data  
TACF <- ARMAacf(ar\_coeff, ma\_coeff, lag.max = round(n\_obs^(1/3))) # command to obtain theorical ACF  
plot(c(0:round(n\_obs^(1/3))),ACF$acf,type='l',xlab='Lag',ylab='ACF',ylim=c(-1,1))  
lines(0:round(n\_obs^(1/3)),TACF,lty=2)  
grid(nx = 4, ny = 4)



residuals = reg\_result$u\_hat # get the AR model residuals  
acf(residuals,lag=round(n\_obs^(1/3)),main = "GNP Growth") # command to obtain sample ACF of the data

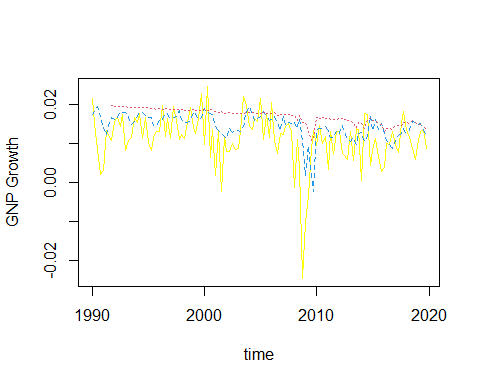


Box.test(residuals, lag = round(n\_obs^(1/3)), type = "Ljung-Box") # applying Ljung and Box (1978) joint test of auto correlations

##   
## Box-Ljung test  
##   
## data: residuals  
## X-squared = 0.6574, df = 6, p-value = 0.9954

#1. Expanding windows

## Forecasting GNP growth using expanding windows  
## Using AIC  
lag\_choice = NA  
init\_win\_len = 120 # the first 30 years  
num\_step\_ahead = 8 # 1 to 8 steps ahead forecastes   
prediction\_results = expanding\_window(y = DY\_new, init\_win\_len = init\_win\_len, pre\_sel\_num\_lags = lag\_choice, num\_step\_ahead = num\_step\_ahead, sel\_method = 'aic')  
yhat\_f\_aic <- prediction\_results$forecast  
  
y\_f\_aic <- prediction\_results$actual\_value  
  
## Plot  
plot(x = DY\_new\_date[121:n\_obs], y = y\_f\_aic, xlab='time',ylab='GNP Growth',type='l',col="yellow")  
lines(x = DY\_new\_date[121:n\_obs],y = yhat\_f\_aic[,1],lty=2, col = 4)  
lines(x = DY\_new\_date[121:n\_obs],y = yhat\_f\_aic[,8],lty=3, col = 2)

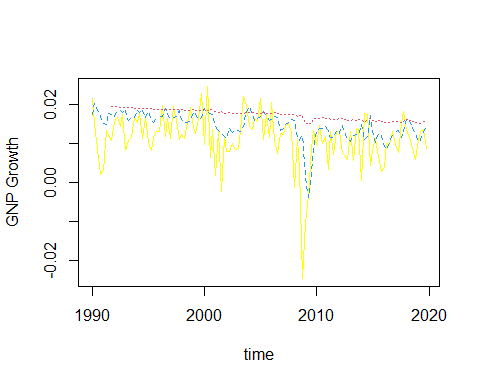


## Root mean square forecast errors  
forecast\_error = kronecker(matrix(1,ncol = num\_step\_ahead),y\_f\_aic) - yhat\_f\_aic  
rmsfe\_ar\_aic = sqrt(colMeans(forecast\_error^2, na.rm = TRUE, dims = 1))  
rmsfe\_ar\_aic

## [1] 0.007000978 0.007180999 0.007735637 0.007892120 0.008108322 0.008156490  
## [7] 0.008342269 0.008432837

##2. Using BIC

## Forecasting GNP growth using expanding windows  
## Using BIC  
lag\_choice = NA  
init\_win\_len = 120 # the first 8 years  
num\_step\_ahead = 8 # 1 to 8 steps ahead forecastes   
prediction\_results = expanding\_window(y = DY\_new, init\_win\_len = init\_win\_len, pre\_sel\_num\_lags = lag\_choice, num\_step\_ahead = num\_step\_ahead, sel\_method = 'bic')  
yhat\_f\_bic <- prediction\_results$forecast  
  
y\_f\_bic <- prediction\_results$actual\_value  
  
## Plot  
plot(x = DY\_new\_date[121:n\_obs], y = y\_f\_bic, xlab='time',ylab='GNP Growth',type='l',col="yellow")  
lines(x = DY\_new\_date[121:n\_obs],y = yhat\_f\_bic[,1],lty=2, col = 4)  
lines(x = DY\_new\_date[121:n\_obs],y = yhat\_f\_bic[,8],lty=3, col = 2)



## Root mean square forecast errors  
forecast\_error = kronecker(matrix(1,ncol = num\_step\_ahead),y\_f\_bic) - yhat\_f\_bic  
rmsfe\_ar\_bic = sqrt(colMeans(forecast\_error^2, na.rm = TRUE, dims = 1))  
rmsfe\_ar\_bic

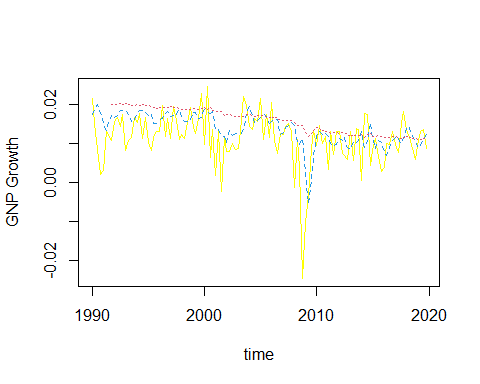
## [1] 0.006768129 0.007327876 0.007891828 0.008170631 0.008322748 0.008401066  
## [7] 0.008560302 0.008668237

## Aerage  
yhat\_f\_ave = (yhat\_f\_aic + yhat\_f\_bic)/2  
forecast\_error = kronecker(matrix(1,ncol = num\_step\_ahead),y\_f\_bic) - yhat\_f\_ave  
rmsfe\_ave = sqrt(colMeans(forecast\_error^2, na.rm = TRUE, dims = 1))  
rmsfe\_ave

## [1] 0.006800907 0.007213739 0.007799146 0.008014225 0.008197299 0.008265888  
## [7] 0.008442060 0.008540591

#3. Rolling windows

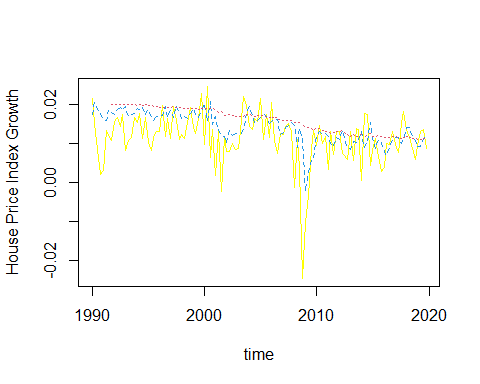
## AIC  
lag\_choice = NA  
init\_win\_len = 120 # the first 30 years  
num\_step\_ahead = 8 # 1 to 8 steps ahead forecastes   
prediction\_results = rolling\_window(y = DY\_new, init\_win\_len = init\_win\_len, pre\_sel\_num\_lags = lag\_choice, num\_step\_ahead = num\_step\_ahead, sel\_method = 'aic')  
yhat\_f\_aic <- prediction\_results$forecast  
  
y\_f\_aic <- prediction\_results$actual\_value  
  
## Plot  
plot(x = DY\_new\_date[121:n\_obs], y = y\_f\_aic,xlab='time',ylab='GNP Growth',type='l',col="yellow")  
lines(x = DY\_new\_date[121:n\_obs],y = yhat\_f\_aic[,1],lty=2, col = 4)  
lines(x = DY\_new\_date[121:n\_obs],y = yhat\_f\_aic[,8],lty=3, col = 2)



## Root mean square forecast errors  
forecast\_error = kronecker(matrix(1,ncol = num\_step\_ahead),y\_f\_aic) - yhat\_f\_aic  
rmsfe\_ar\_aic = sqrt(colMeans(forecast\_error^2, na.rm = TRUE, dims = 1))  
rmsfe\_ar\_aic

## [1] 0.006532133 0.006869404 0.007297877 0.007496990 0.007556114 0.007561955  
## [7] 0.007672865 0.007741224

## BIC  
lag\_choice = NA  
init\_win\_len = 120 # the first 30 years  
num\_step\_ahead = 8 # 1 to 24 steps ahead forecastes   
prediction\_results = rolling\_window(y = DY\_new, init\_win\_len = init\_win\_len, pre\_sel\_num\_lags = lag\_choice, num\_step\_ahead = num\_step\_ahead, sel\_method = 'bic')  
yhat\_f\_bic <- prediction\_results$forecast  
  
y\_f\_bic <- prediction\_results$actual\_value  
  
## Plot  
plot(x = DY\_new\_date[121:n\_obs], y = y\_f\_bic,xlab='time',ylab='House Price Index Growth',type='l',col="yellow")  
lines(x = DY\_new\_date[121:n\_obs],y = yhat\_f\_bic[,1],lty=2, col = 4)  
lines(x = DY\_new\_date[121:n\_obs],y = yhat\_f\_bic[,8],lty=3, col = 2)



forecast\_error = kronecker(matrix(1,ncol = num\_step\_ahead),y\_f\_bic) - yhat\_f\_bic  
rmsfe\_ar\_bic = sqrt(colMeans(forecast\_error^2, na.rm = TRUE, dims = 1))  
rmsfe\_ar\_bic

## [1] 0.006767772 0.007373192 0.007748851 0.007883883 0.007847288 0.007784938  
## [7] 0.007838395 0.007868308

yhat\_f\_ave = (yhat\_f\_aic + yhat\_f\_bic)/2  
forecast\_error = kronecker(matrix(1,ncol = num\_step\_ahead),y\_f\_bic) - yhat\_f\_ave  
rmsfe\_ave = sqrt(colMeans(forecast\_error^2, na.rm = TRUE, dims = 1))  
rmsfe\_ave

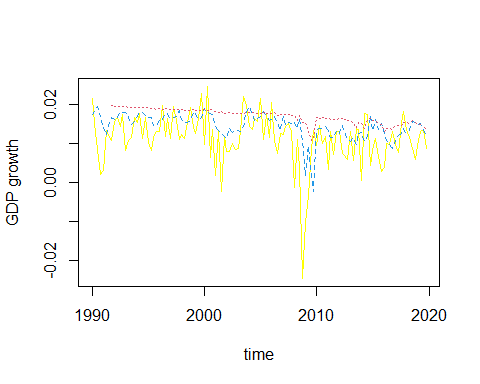
## [1] 0.006612314 0.007095963 0.007506544 0.007679328 0.007695086 0.007669349  
## [7] 0.007753135 0.007803207

rmsfe\_all\_rolling = rbind(rmsfe\_ar\_aic,rmsfe\_ar\_bic,rmsfe\_ave)  
rmsfe\_all\_rolling

## [,1] [,2] [,3] [,4] [,5]  
## rmsfe\_ar\_aic 0.006532133 0.006869404 0.007297877 0.007496990 0.007556114  
## rmsfe\_ar\_bic 0.006767772 0.007373192 0.007748851 0.007883883 0.007847288  
## rmsfe\_ave 0.006612314 0.007095963 0.007506544 0.007679328 0.007695086  
## [,6] [,7] [,8]  
## rmsfe\_ar\_aic 0.007561955 0.007672865 0.007741224  
## rmsfe\_ar\_bic 0.007784938 0.007838395 0.007868308  
## rmsfe\_ave 0.007669349 0.007753135 0.007803207

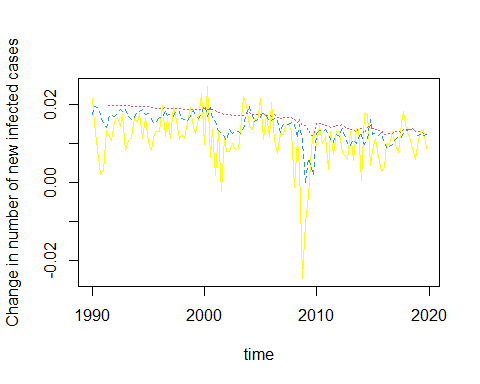
#4.

## Forecasting using AR Model with aic selected number of lags  
lag\_choice = NA  
init\_win\_len = 120 # the first 30 years  
num\_step\_ahead = 8 # 1 to 8 steps ahead forecastes   
prediction\_results = expanding\_window(y = DY\_new, init\_win\_len = init\_win\_len, pre\_sel\_num\_lags = lag\_choice, num\_step\_ahead = num\_step\_ahead, sel\_method = 'aic')  
y\_f\_aic <- prediction\_results$actual\_value  
yhat\_f\_aic <- prediction\_results$forecast  
selected\_num\_lags <- prediction\_results$sel\_num\_lags  
  
plot(x = DY\_new\_date[121:n\_obs], y = y\_f\_aic,xlab='time',ylab='GDP growth',type='l',col="yellow")  
lines(x = DY\_new\_date[121:n\_obs],y = yhat\_f\_aic[,1],lty=2, col = 4)  
lines(x = DY\_new\_date[121:n\_obs],y = yhat\_f\_aic[,8],lty=3, col = 2)



forecast\_error = kronecker(matrix(1,ncol = num\_step\_ahead),y\_f\_aic) - yhat\_f\_aic  
rmsfe\_ar\_aic = sqrt(colMeans(forecast\_error^2, na.rm = TRUE, dims = 1))

# Forecasting by Model averaging and RMSFE comparison  
yhat\_f\_ave = (yhat\_f\_aic + yhat\_f\_bic)/2  
forecast\_error = kronecker(matrix(1,ncol = num\_step\_ahead),y\_f\_bic) - yhat\_f\_ave  
rmsfe\_ave = sqrt(colMeans(forecast\_error^2, na.rm = TRUE, dims = 1))  
  
plot(x = DY\_new\_date[121:n\_obs], y = y\_f\_bic,xlab='time',ylab='Change in number of new infected cases',type='l',col="yellow")  
lines(x = DY\_new\_date[121:n\_obs],y = yhat\_f\_ave[,1],lty=2, col = 4)  
lines(x = DY\_new\_date[121:n\_obs],y = yhat\_f\_ave[,7],lty=3, col = 2)



rmsfe\_all = rbind(rmsfe\_ar\_aic,rmsfe\_ar\_bic,rmsfe\_ave)  
rmsfe\_all

## [,1] [,2] [,3] [,4] [,5]  
## rmsfe\_ar\_aic 0.007000978 0.007180999 0.007735637 0.007892120 0.008108322  
## rmsfe\_ar\_bic 0.006767772 0.007373192 0.007748851 0.007883883 0.007847288  
## rmsfe\_ave 0.006786340 0.007192845 0.007680859 0.007826290 0.007914883  
## [,6] [,7] [,8]  
## rmsfe\_ar\_aic 0.008156490 0.008342269 0.008432837  
## rmsfe\_ar\_bic 0.007784938 0.007838395 0.007868308  
## rmsfe\_ave 0.007910045 0.008028727 0.008088655