Time Series Project 2

Congyu Hang

2023-11-13

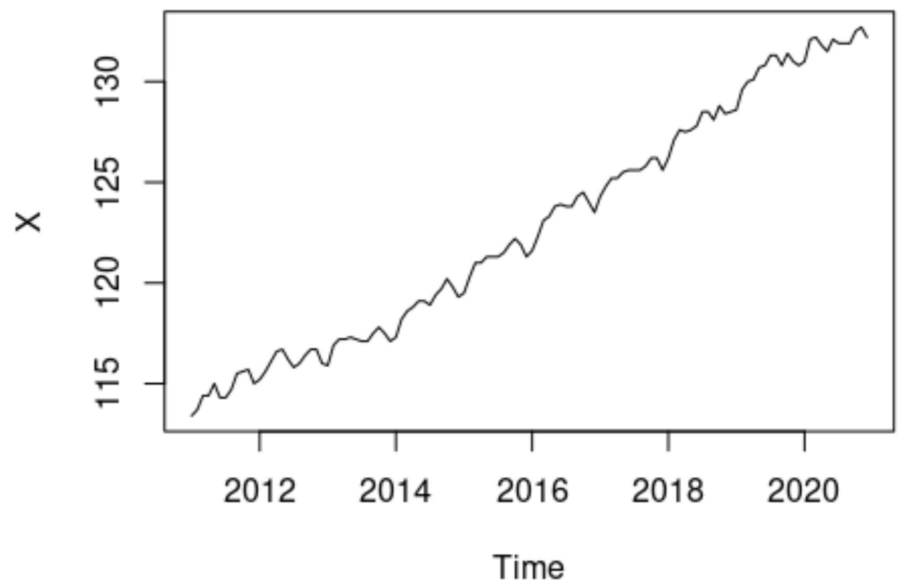
Practice

1. For this question you will work with unadjusted monthly Consumer Price Index (CPI) data from Statistics Canada. This index is used to measure inflation, an important economic measure that tracks the overall change in the prices of a representative basket of goods and services. You will work with the CPI of a particular price category from Jan 2011 to Dec 2020. Use your student ID# as a seed to pick a random number from 1-17, and download the corresponding series from the table in the Appendix; below is starter code for downloading the CPI of All items. **Make sure to use the right data, otherwise you will lose marks.**

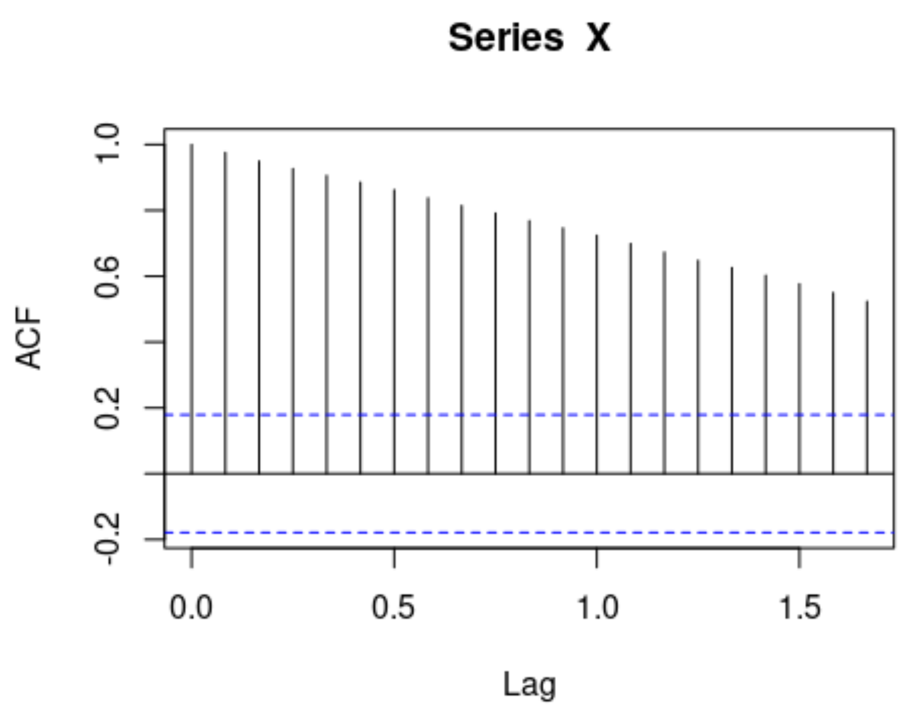
```
set.seed(1006740369) # set random see to your student ID
sample(1:17, size =1) # pick a random number from 1-60

## [1] 11

library(cansim)
library(tidyverse)

# unadjusted series for All-items
X = get_cansim_vector( "v41691233", start_time = "2011-01-01", end_time =
"2020-12-01") %>%
    pull(VALUE) %>% ts( start = c(2011,1), frequency = 12)

## Accessing CANSIM NDM vectors from Statistics Canada
```

a. [4 marks] Plot your raw series, its ACF and PACF, and comment on its characteristics (trend, seasonality, stationarity). Perform appropriate pre-processing (e.g. detrending, differencing, transformation) that will make the series stationary, and plot the resulting ACF and PACF.

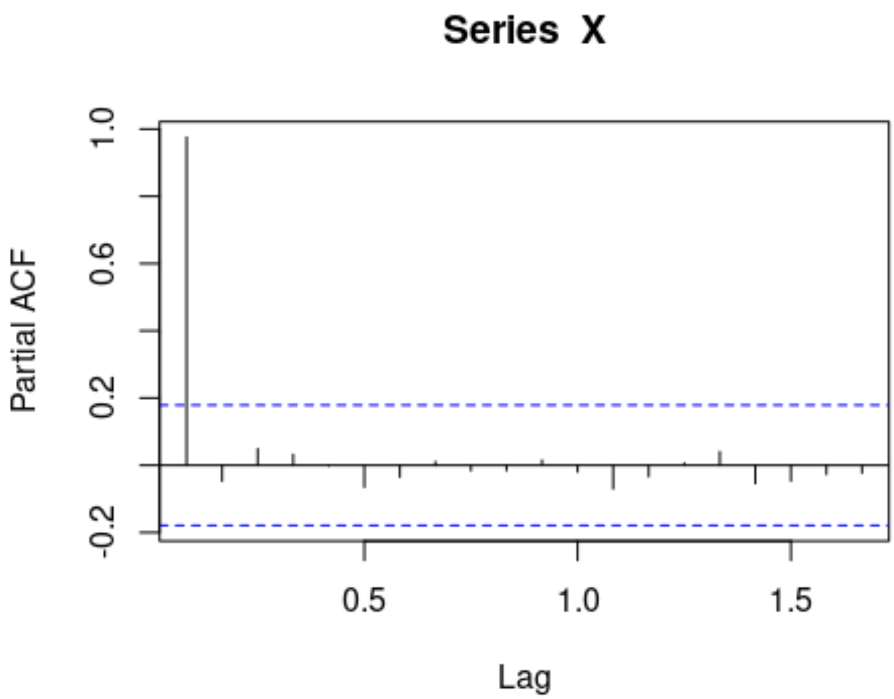
```
plot(X)
```




```r
acf(X)
```

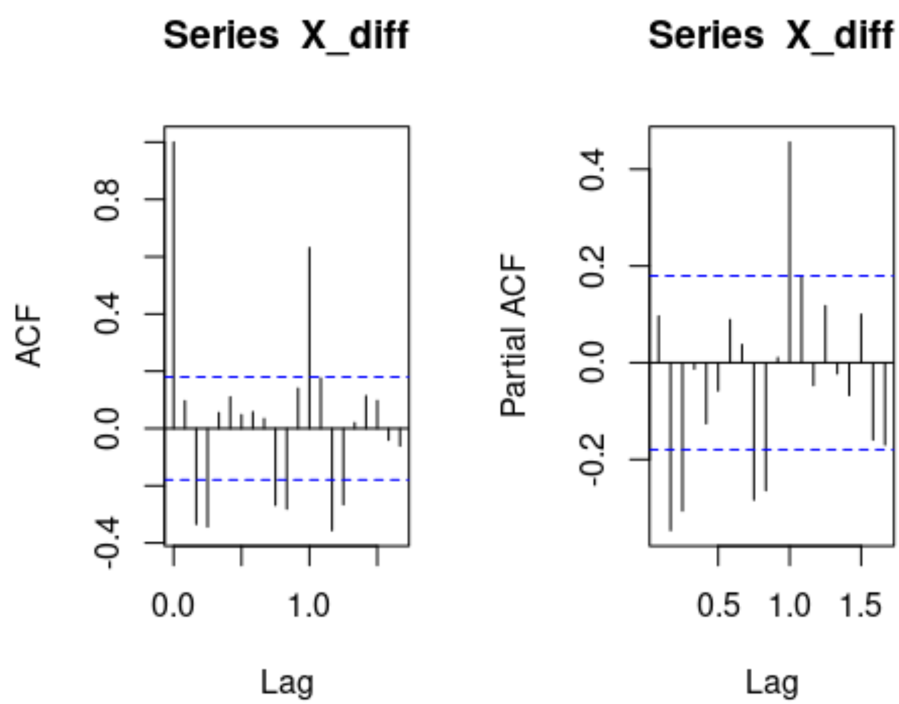



```r
pacf(X)
```



Random Walk Behavior, so take difference of the series

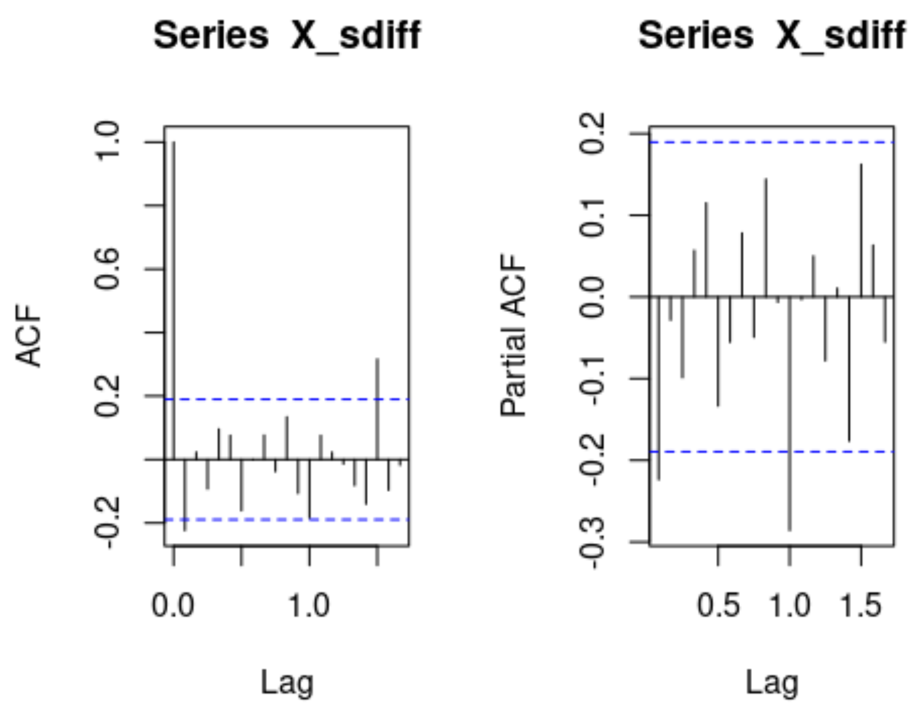
```
X_diff = diff(X)
par(mfrow = c(1,2))
acf(X_diff)
pacf(X_diff)
```



The difference series has significant acf when lag=12 in both acf and pacf, this indicate seasonality, both acf and pacf are significant when lag=2,3

Difference by lag=12 and conclude X_diff should apply seasonal ARMA model

```
X_sdiff = diff(X_diff,lag = 12)
par(mfrow = c(1,2))
acf(X_sdiff)
pacf(X_sdiff)
```



The pacf tails off and at lag=1 there is a significant acf when lag = 1, conclude that this processed data follow MA(1) model.

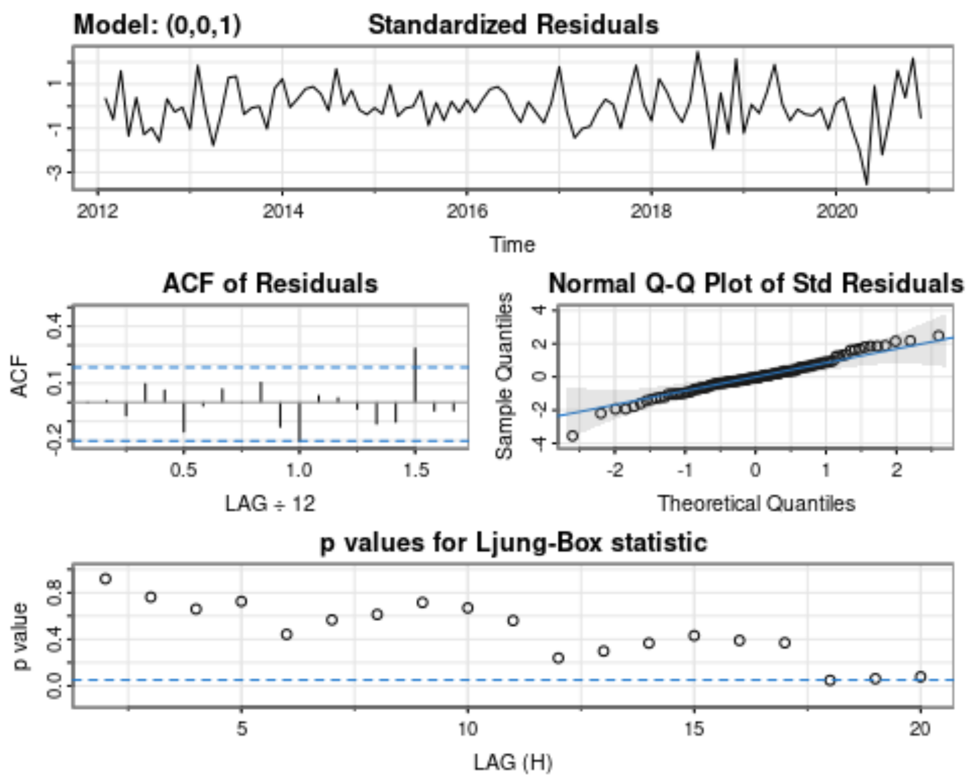
b. [4 marks] Identify an appropriate (seasonal) ARMA model for your pre-processed series, based on your analysis in the previous part. Fit the model and report its parameters. Present diagnostics for your model, and comment on its fit and limitations.

```
library(astsa)
ma_fit = arima(X_sdiff,order = c(0,0,1))
ma_fit$coef

##          ma1    intercept
## -0.232809105 -0.003570381
```

The coefficients of the MA(1) model is theta = -0.232809105

```
AIC(ma_fit)

## [1] 44.17978

sarima(X_sdiff,0,0,1)

## initial  value -1.214016
## iter   2 value -1.240705
## iter   3 value -1.240770
## iter   4 value -1.240771
## iter   5 value -1.240771
```

```
## iter   5 value -1.240771
## iter   5 value -1.240771
## final  value -1.240771
## converged
## initial  value -1.240526
## iter   2 value -1.240528
## iter   2 value -1.240528
## iter   2 value -1.240528
## final  value -1.240528
## converged
```




```
## $fit
##
## Call:
## arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D, Q),
period = S),
##     xreg = xmean, include.mean = FALSE, transform.pars = trans, fixed =
fixed,
##     optim.control = list(trace = trc, REPORT = 1, reltol = tol))
##
## Coefficients:
##           ma1    xmean
##       -0.2328  -0.0036
## s.e.   0.0941   0.0215
##
## sigma^2 estimated as 0.08361:  log likelihood = -19.09,  aic = 44.18
##
```

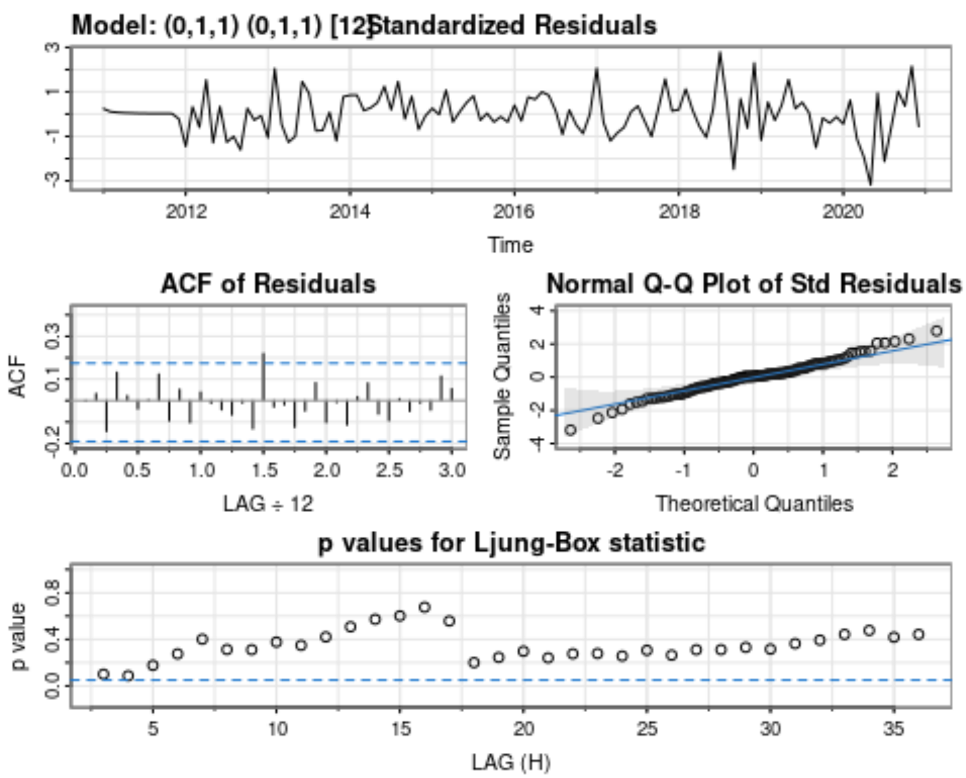
```
## $degrees_of_freedom
## [1] 105
##
## $ttable
##       Estimate     SE t.value p.value
## ma1    -0.2328 0.0941  -2.473  0.0150
## xmean  -0.0036 0.0215  -0.166  0.8685
##
## $AIC
## [1] 0.4128952
##
## $AICc
## [1] 0.4139735
##
## $BIC
## [1] 0.4878343
```

The residuals look normal and p-values of Ljung-Box are high. The model can be applied without limitations. The model fits the data well.

c. [3 marks] Select a SARIMA model for your *original series* (no pre-processing) based on Akaike's Information Criterion, using `forecast::auto.arima()`. Report the model parameters, comment on its fit (diagnostics), and compare its AIC to the one from the previous part (if they are different).

```
library(forecast)

auto.fit = auto.arima(X)
summary(auto.fit)

## Series: X
## ARIMA(0,1,1)(0,1,1)[12]
##
## Coefficients:
##           ma1     sma1
##       -0.2599  -0.4396
## s.e.   0.0946   0.1081
##
## sigma^2 = 0.07651:  log likelihood = -13.52
## AIC=33.04   AICc=33.27   BIC=41.06
##
## Training set error measures:
##                       ME      RMSE       MAE         MPE      MAPE
MASE
## Training set 0.001106759 0.2587462 0.1909556 0.001004373 0.1542373
0.09924547
##                      ACF1
## Training set -0.002696453

sarima(X,0,1,1,0,1,1,12)
```

```
## initial  value -1.213936
## iter   2 value -1.272776
## iter   3 value -1.290231
## iter   4 value -1.292165
## iter   5 value -1.292361
## iter   6 value -1.292361
## iter   6 value -1.292361
## final  value -1.292361
## converged
## initial  value -1.292474
## iter   2 value -1.292595
## iter   3 value -1.292600
## iter   3 value -1.292600
## iter   3 value -1.292600
## final  value -1.292600
## converged
```



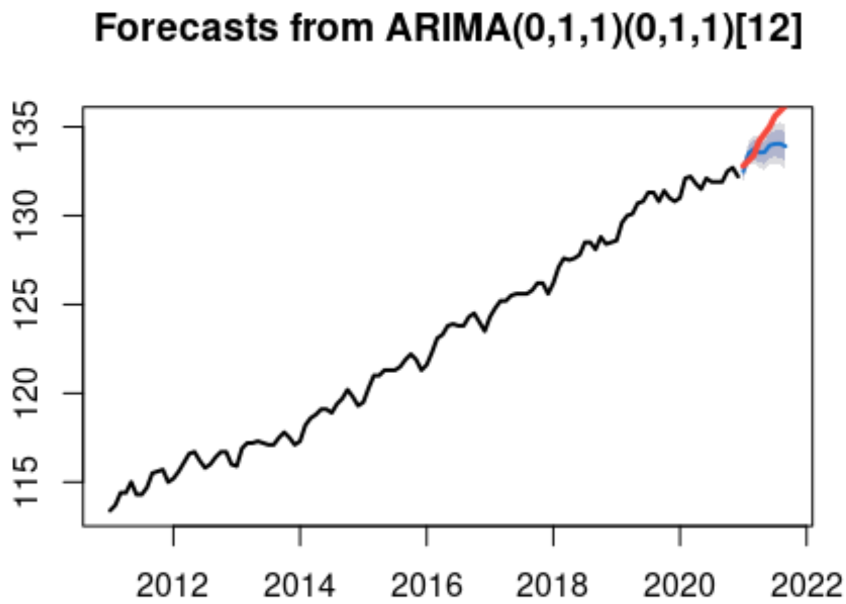

```
## $fit
##
## Call:
## arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D, Q),
period = S),
##     include.mean = !no.constant, transform.pars = trans, fixed = fixed,
optim.control = list(trace = trc,
##         REPORT = 1, reltol = tol))
##
## Coefficients:
```

```
##           ma1      sma1
##       -0.2599   -0.4396
## s.e.   0.0946   0.1081
##
## sigma^2 estimated as 0.07354:  log likelihood = -13.52,  aic = 33.04
##
## $degrees_of_freedom
## [1] 105
##
## $ttable
##      Estimate     SE t.value p.value
## ma1   -0.2599 0.0946 -2.7468  0.0071
## sma1  -0.4396 0.1081 -4.0654  0.0001
##
## $AIC
## [1] 0.3087526
##
## $AICc
## [1] 0.3098309
##
## $BIC
## [1] 0.3836917
```

The AIC of auto.fit model is 33.04, which is relatively lower than AIC of ma_fit, 44.179, since auto.fit includes a seasonal MA term and ma_fit does not. However, AIC of 2 models are calculated based on different data, one is the processed version and the other is the raw data, therefore, AIC is not comparable.

d. [3 marks] Create 1- to 9-step-ahead forecasts for the *original series*, using your model from `auto.arima`. Plot the series and your forecasts, and overlay the actual CPI values for Jan to Sep 2021 (you need to download them). Comment on the quality of your predictions.

```
Xnew = get_cansim_vector( "v41691233", start_time = "2021-01-01", end_time =
"2021-09-01") %>%
    pull(VALUE) %>% ts( start = c(2021,1), frequency = 12)

## Accessing CANSIM NDM vectors from Statistics Canada

forecast(auto.fit,h=9) %>% plot(lwd=2)
lines(Xnew,col=2,lwd=3)
```

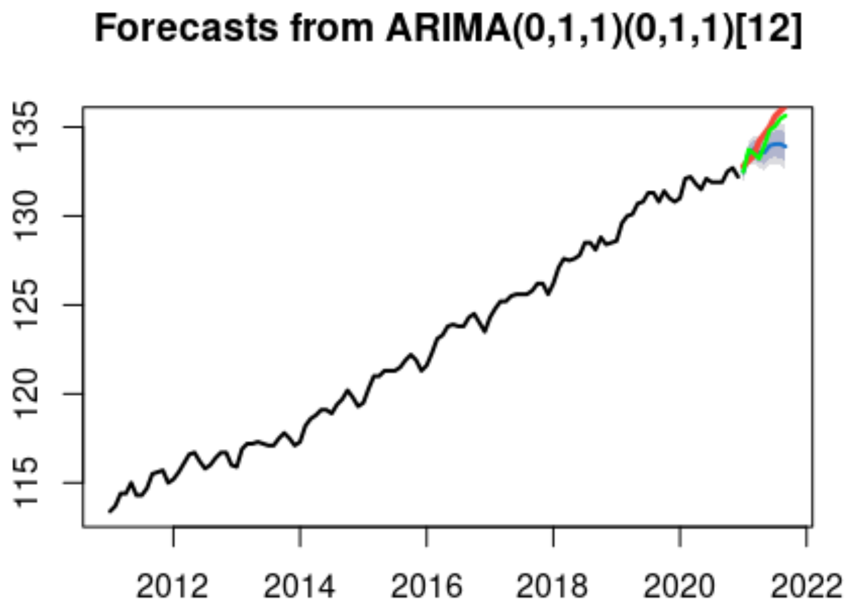


The prediction quality is good for the first a few predictions but the values for predictions with larger steps ahead, the values look of from the true values.

e. [4 marks] Repeat the previous step using *iterative 1-step-ahead predictions* for Jan to Sep 2021. Apply the fitted model from the previous part to the expanded data set (till Sep 2021), and calculate the fitted values; see this post, for details on how to achieve this using `forecast::Arima()`. Compare the iterative 1-step-ahead predictions to those of the previous part; which ones are more accurate and why?

```
new.fit = Arima(X,order = c(0, 1, 1),
  seasonal = c(0, 1, 1))
X_expanded = get_cansim_vector( "v41691233", start_time = "2011-01-01",
end_time = "2021-09-01") %>%
    pull(VALUE) %>% ts( start = c(2011,1), frequency = 12)

## Accessing CANSIM NDM vectors from Statistics Canada

fit.value = Arima(X_expanded,model = new.fit)
fit.values = fit.value$fitted

Xnew = get_cansim_vector( "v41691233", start_time = "2021-01-01", end_time =
"2021-09-01") %>%
    pull(VALUE) %>% ts( start = c(2021,1), frequency = 12)

## Reading CANSIM NDM vectors from temporary cache
```

```r
forecast(auto.fit,h=9) %>% plot(lwd=2)
lines(Xnew,col=2,lwd=3)
lines(tail(fit.values,9),col = "green",lwd=2)
```



The 1-step-ahead predictions (green) are very close to true values and has better prediction quality than n-step ahead predictions (blue) because iterative 1-step-ahead predictions are made with more information available.

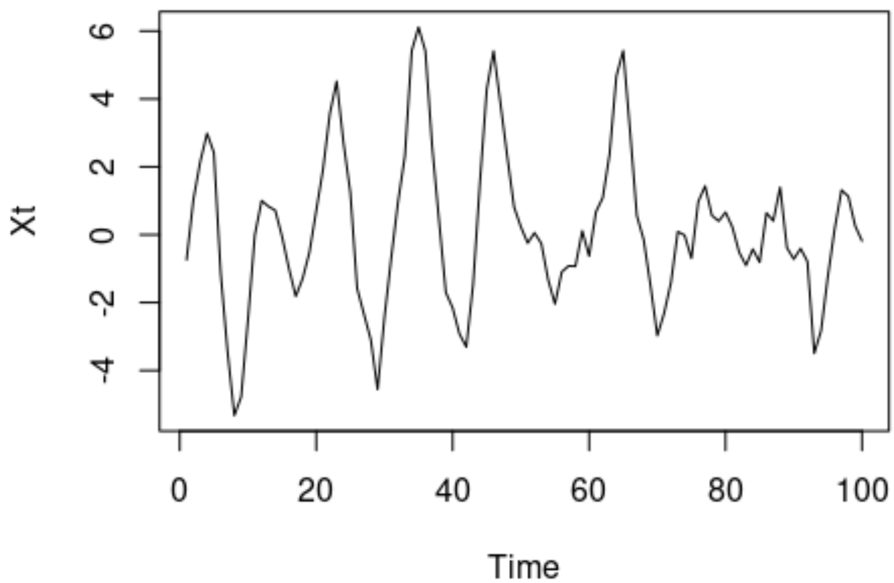
2. For this part you will verify the Yule-Walker (i.e., Method of Moments) estimates from R.

b. [2 marks] Using your student ID# as a seed, simulate 100 values from the AR(3) model

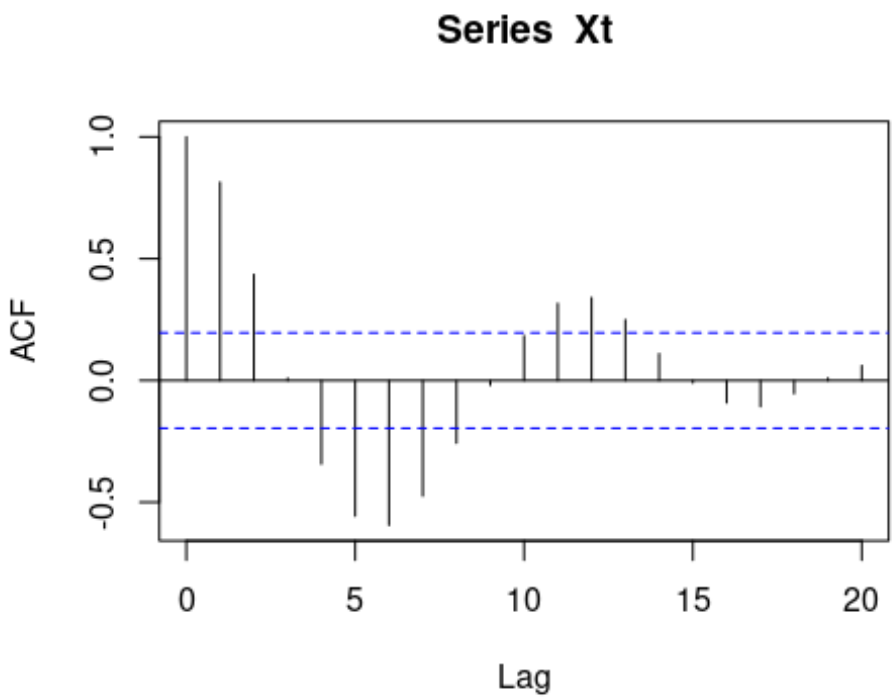
$$X_t = 1.3X_{t-1} - .4X_{t-2} - .3X_{t-3} + W_t, \quad W_t \sim^{iid} N(0,1)$$

Plot the series and their sample ACF and PACF. (Hint: You can use function `arima.sim()` to simulate the series.)

```
Xt = arima.sim(n = 100, list(ar = c(1.3, -0.4, -0.3)))
plot(Xt)
```

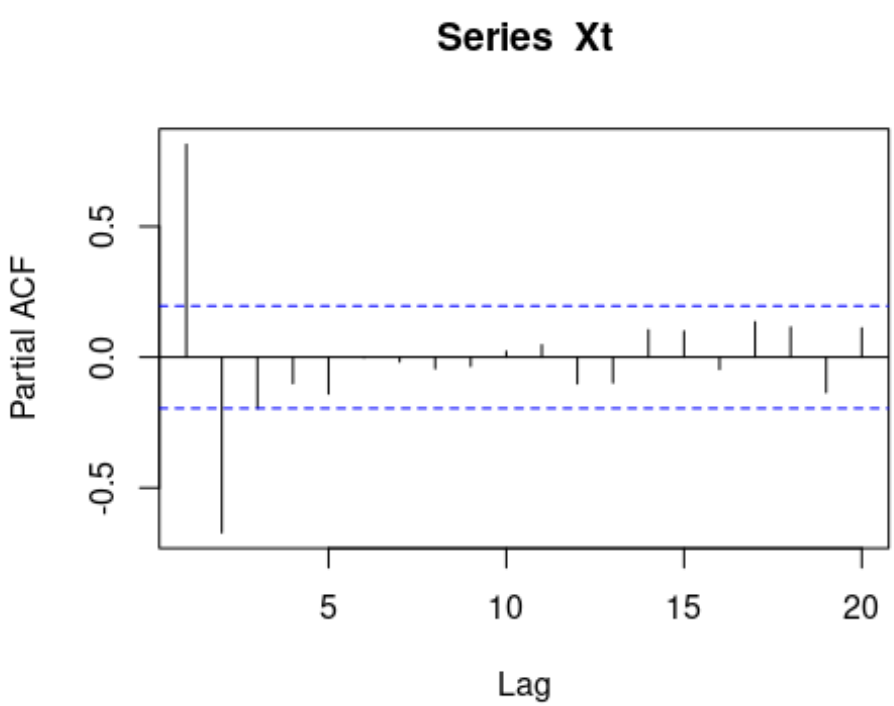



```
acf(Xt)
```



Series Xt
ACF
1.0
0.5
0.0
-0.5
0
5
10
15
20
Lag


```
pacf(Xt)
```



Series Xt
Partial ACF
0.5
0.0
-0.5
5
10
15
20
Lag

c. [2 marks] Run Yule-Walker estimation to fit an AR(3) model to your simulated data. Report the AR coefficients and their asymptotic variance-covariance matrix.

```
out.yw = ar.yw( Xt,aic = F,order.max = 3)
out.yw$ar

## [1]  1.2286755 -0.4051420 -0.1959475
```

The ar coeficients are 1.3272424 -0.4477390 -0.2766742

asymptotic variance-covariance matrix

```
out.yw$asy.var.coef

##              [,1]        [,2]         [,3]
## [1,]  0.010016714 -0.01362565  0.006728104
## [2,] -0.013625646  0.02403237 -0.013625646
## [3,]  0.006728104 -0.01362565  0.010016714
```

d. [2 marks] Calculate the *sample* ACVF of the data, and use it to build the $\underline{\hat{\Gamma}_3}$ matrix and the $\underline{\hat{\gamma}_3}$ vector of sample correlations. (Hint: You can use function `acf( , type = "covariance")` to get the sample covariance, and function `toeplitz()` to create the $\underline{\hat{\Gamma}_3}$ matrix.)

```
sample_cov = acf(Xt, type = "covariance",plot = F)
gamma_3 = as.vector(sample_cov$acf)[2:4]
GAMMA_3 = toeplitz(as.vector(sample_cov$acf)[1:3])
```

e. [2 marks] Solve the system of equations $\underline{\hat{\Gamma}_3}\underline{\hat{\phi}_3} = \underline{\hat{\gamma}_3}$ to find the estimated AR(3) coefficients $\underline{\hat{\phi}_3}$, and verify that they are the same as the ones from `ar.yw` (Hint: use function `solve()` to solve the system)

```
(phi = solve(GAMMA_3)%*%gamma_3)

##            [,1]
## [1,]  1.2286755
## [2,] -0.4051420
## [3,] -0.1959475
```

f. [2 marks] Calculate the asymptotic variance-covariance matrix of the estimated coefficients using the formula $\sigma_w^2\underline{\hat{\Gamma}_p}^{-1}/n$, and verify that it is the same as the one from `ar.yw`. (Hint: You can get $\hat{\sigma}_w^2$ from the output of `ar.yw`)

```
sigma2 = out.yw$var.pred
(est.matrix = sigma2*solve(GAMMA_3)/100)

##              [,1]        [,2]         [,3]
## [1,]  0.010016714 -0.01362565  0.006728104
## [2,] -0.013625646  0.02403237 -0.013625646
## [3,]  0.006728104 -0.01362565  0.010016714

out.yw$asy.var.coef
```

```
##                [,1]        [,2]         [,3]
## [1,]  0.010016714 -0.01362565  0.006728104
## [2,] -0.013625646  0.02403237 -0.013625646
## [3,]  0.006728104 -0.01362565  0.010016714
```

It is verified that the 2 matrices have values the same.

Appendix

#	CPI Category	Vector
1	All-items	v41690973
2	Food 5	v41690974
3	Shelter 6	v41691050
4	Household operations, furnishings and equipment	v41691067
5	Clothing and footwear	v41691108
6	Transportation	v41691128
7	Gasoline	v41691136
8	Health and personal care	v41691153
9	Recreation, education and reading	v41691170
10	Alcoholic beverages, tobacco products and recreational cannabis	v41691206
11	All-items excluding food and energy 7	v41691233
12	All-items excluding mortgage interest cost	v41691240
13	All-items excluding shelter	v41691234
14	All-items excluding energy 7	v41691238
15	Energy 7	v41691239
16	Goods 8	v41691222
17	Services 9	v41691230