Master 2 - Time-Series Take-home Exam

Zhihan ZHANG

1. (Multiple Choice, 4 points) Consider the following two sample paths.

One is generated from a trend stationary process, whereas the other one is generated from a unit root with drift.

a. Which one is generated from a trend stationary process?

(1) Left
(2) Right
(3) Both

Answer: (1)

b. Consider the time series on the left panel. You want to test the null hypothesis of unit root using ADF test. Which test equation is a proper one?

(1) $X_t = \alpha X_{t-1} + u_t$
(2) $X_t = c_0 + \alpha X_{t-1} + u_t$
(3) $X_t = c_0 + c_1 t + \alpha X_{t-1} + u_t$

Answer: (3)

2. (Multiple Choice, 4 points) Let $X_t, Y_t \sim I(1)$, and consider the following nonstationary regression model:

$Y_t = \beta X_t + u_t$
For the regression error process u_t, we consider the following two conditions
Condition 1: $u_t = 0.5u_{t-1} + 0.5u_{t-2} + \varepsilon_t$ (AR(2) Stationary, $I(0)$)
Condition 2: $u_t = 0.5u_{t-1} - 0.5u_{t-2} + \varepsilon_t$ (non-stationary)
where $\varepsilon_t \sim iid(0,\ \sigma^2)$. Let $\hat{\beta}$ be the OLS estimator for β. Find ALL correct statements (choose up to 4 Answers).

(1) Under Condition 1, Y_t and X_t are cointegrated.
(2) Under Condition 2, Y_t and X_t are cointegrated.
(3) Under Condition 1, the regression is spurious.
(4) Under Condition 2, the regression is spurious.
(5) Under Condition 1, $\hat{\beta}$ is super consistent, i.e., $\hat{\beta} - \beta = O_p(1/T)$
(6) Under Condition 2, $\hat{\beta}$ is super consistent, i.e., $\hat{\beta} - \beta = O_p(1/T)$
(7) Under Condition 1, $\hat{\beta}$ is \sqrt T-consistent, i.e., $\hat{\beta} - \beta = O_p(1/\sqrt{T})$
(8) Under Condition 2, $\hat{\beta}$ is \sqrt T-consistent, i.e., $\hat{\beta} - \beta = O_p(1/\sqrt{T})$
(9) Under Condition 1, $\hat{\beta}$ is inconsistent, i.e., $\hat{\beta} \nrightarrow_p \beta$.
(10) Under Condition 2, $\hat{\beta}$ is inconsistent, i.e., $\hat{\beta} \nrightarrow_p \beta$.

Answer: (1) (4) (5) (10)

3. (Short Answer; Multiple Choice, 6 points) Consider bivariate SF-VAR (1) models

Model 1:

$$\begin{bmatrix} 1 & 0 \\ 0.2 & 1 \end{bmatrix} \begin{bmatrix} y_{1,t} \\ y_{2,t} \end{bmatrix} = \begin{bmatrix} 0.5 & 0 \\ 0.1 & 0.5 \end{bmatrix} \begin{bmatrix} y_{1,t-1} \\ y_{2,t-1} \end{bmatrix} + \begin{bmatrix} \varepsilon_{1,t} \\ \varepsilon_{2,t} \end{bmatrix}, \Sigma_\varepsilon = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Model 2:

$$\begin{bmatrix} 1 & 0 \\ -0.2 & 1 \end{bmatrix} \begin{bmatrix} y_{1,t} \\ y_{2,t} \end{bmatrix} = \begin{bmatrix} 0.5 & 0 \\ 0.1 & 0.5 \end{bmatrix} \begin{bmatrix} y_{1,t-1} \\ y_{2,t-1} \end{bmatrix} + \begin{bmatrix} \varepsilon_{1,t} \\ \varepsilon_{2,t} \end{bmatrix}, \ \Sigma_\varepsilon = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

For the above VARs, (y_i) does not Granger-cause (y_j) for $i,\ j\ =\ 1,\ 2$ and $i \neq j$ if $E[y_{j,t+1}|y_{1,t}, y_{2,t}] = E[y_{j,t+1}|y_{j,t}]$.

a. Let $e_t = (e_{1,t},\ e_{2,t})\prime$ be reduced form errors of the RF-VAR(1) corresponding to the SFVAR(1) above. Find $Cov(e_{1,t},\ e_{2,t})$ for each model.

Answer:

Model 1:

$$B_0 = \begin{bmatrix} 1 & 0 \\ 0.2 & 1 \end{bmatrix}, B_1 = \begin{bmatrix} 0.5 & 0 \\ 0.1 & 0.5 \end{bmatrix}, \varepsilon_t \sim \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right)$$

SF-VAR (1): $B_0 y_t = B_1 y_{t-1} + \varepsilon_t,\ \varepsilon_t \sim (0,\ \Sigma_\varepsilon)$

RF-VAR (1): $y_t = A_1 y_{t-1} + e_t,\ A_1 = B_0{}^{-1} B_1,\ e_t \sim (0,\ \Sigma_e)$

$$\Sigma_e = B_0{}^{-1} \Sigma_\varepsilon B_0{}^{-1\prime} = \frac{1}{0.8} \begin{bmatrix} 1 & 0 \\ 0.2 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} * \left[\frac{1}{0.8} \begin{bmatrix} 1 & 0 \\ 0.2 & 1 \end{bmatrix} \right]' = \begin{bmatrix} 1 & 0 \\ 0.2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0.2 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0.2 \\ 0.2 & 1.04 \end{bmatrix}$$

$Cov(e_{1,t}, e_{2,t}) = \mathbf{0.2}$

Model 2:

$$B_0 = \begin{bmatrix} 1 & 0 \\ -0.2 & 1 \end{bmatrix}, B_1 = \begin{bmatrix} 0.5 & 0 \\ 0.1 & 0.5 \end{bmatrix}, \varepsilon_t \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right)$$

SF-VAR (1): $B_0 y_t = B_1 y_{t-1} + \varepsilon_t,\ \varepsilon_t \sim (0,\ \Sigma_\varepsilon)$

RF-VAR (1): $y_t = A_1 y_{t-1} + e_t,\ A_1 = B_0{}^{-1} B_1,\ e_t \sim (0,\ \Sigma_e)$

$$\Sigma_e = B_0{}^{-1} \Sigma_\varepsilon B_0{}^{-1\prime} = \frac{1}{0.8} \begin{bmatrix} 1 & 0 \\ -0.2 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} * \left[\frac{1}{0.8} \begin{bmatrix} 1 & 0 \\ -0.2 & 1 \end{bmatrix} \right]' = \begin{bmatrix} 1 & 0 \\ -0.2 & 1 \end{bmatrix} \begin{bmatrix} 1 & -0.2 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & -0.2 \\ -0.2 & 1.04 \end{bmatrix}$$

$Cov(e_{1,t},\ e_{2,t}) = -\mathbf{0.2}$

b. Find ALL correct statements about Model 1 (choose up to 2 Answers).

(1) (y_1) does not Granger-cause (y_2).
(2) (y_2) does not Granger-cause (y_1).
(3) (y_1) Granger-causes (y_2).
(4) (y_2) Granger-causes (y_1).

Answer: (1) (2)

c. Find ALL correct statements about Model 2 (choose up to 2 Answers).

(1) (y_1) does not Granger-cause (y_2).
(2) (y_2) does not Granger-cause (y_1).
(3) (y_1) Granger-causes (y_2).
(4) (y_2) Granger-causes (y_1).

Answer: (2) (3)

4. (Computer Exercise, 10 points)

Obtain quarterly Real Personal Consumption Expenditures for the 1955Q1-2018Q4 sample, available on FRED under code PCECC96. Construct the log changes in the Real Personal Consumption Expenditures $\Delta \log c_t = log c_t - log c_{t-1}$, where c_t is the original quarterly Real Personal Consumption Expenditures.

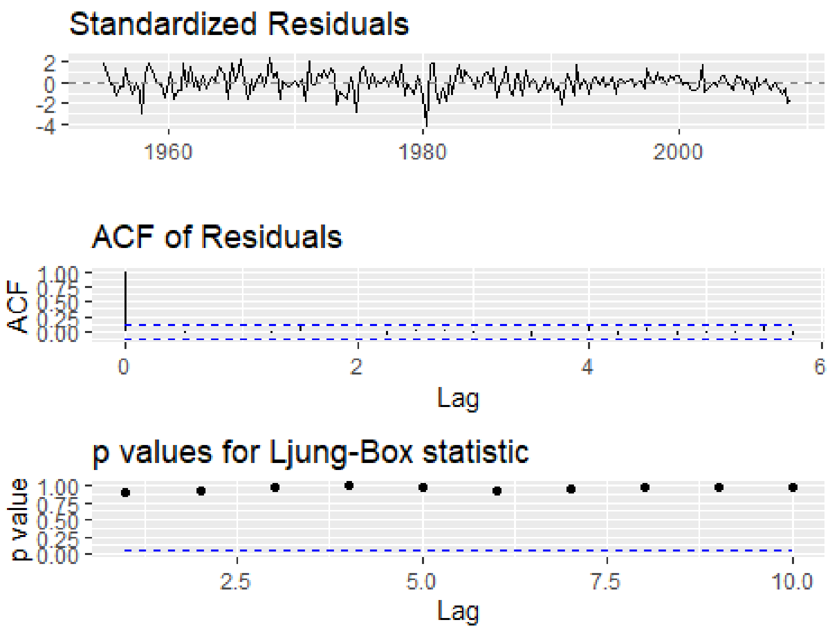
a. Split the sample into two parts: first one up to 2008Q4, second one from 2009Q1 onward. Use `auto.arima` with `ic = aic and stationary = TRUE, stepwise = FALSE, approximation = FALSE` to find the best model. Check the estimated model for adequacy, diagnose residuals using `ggtsdiag`.

Answer:

Here I use `auto.arima` from `library(forecast)` to choose the best model and use `ggtsdiag` from `library(ggfortify)` to diagnose residuals. This code is also used for similar problems later.

```
library(forecast)
estimated4a <-auto.arima(data4_est,ic="aic", stationary = TRUE, stepwise = FALSE,
                         approximation = FALSE)
##best model ARIMA(0,0,3)(2,0,0)[4] MA(3) for nonseasonal part, AR(2)[4] for seasonal part
library(ggfortify)
ggtsdiag(estimated4a)
##white noise
```

The best model is **ARIMA (0,0,3) (2,0,0) [4]**. MA (3) for non-seasonal part, and AR (2) [12] for seasonal part.



The residuals are statistically uncorrelated, and statistically indistinguishable from zero. The p-value is high. The ACF results have no statistically significant coefficients. The null of white noise is not rejected. The model seems to fit well.

b. Use the estimated model with `forecast` to generate 1 to 36 steps ahead forecast for the prediction subsample, 2009Q1-2018Q4.

Answer:

Actually I generate 1 to 40 steps ahead forecast since (2018-2009+1)*4 = 40.

```
forecast4a <- forecast(estimated4a,h=40)
```

This code is also used for similar problems later.

	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
2009 Q1	0.0015024	-0.0069181	0.0099229	-0.0113756	0.0143804
2009 Q2	0.0027495	-0.0058909	0.0113899	-0.0104649	0.0159638
2009 Q3	0.0048598	-0.0040459	0.0137655	-0.0087602	0.0184798
2009 Q4	0.0082069	-0.0009695	0.0173833	-0.0058272	0.0222410
2010 Q1	0.0096238	0.0004339	0.0188137	-0.0044310	0.0236785
2010 Q2	0.0090803	-0.0001103	0.0182710	-0.0049755	0.0231362
2010 Q3	0.0107746	0.0015831	0.0199661	-0.0032825	0.0248318
2010 Q4	0.0111674	0.0019750	0.0203598	-0.0028912	0.0252260
2011 Q1	0.0096735	0.0003991	0.0189480	-0.0045105	0.0238576
2011 Q2	0.0094548	0.0001760	0.0187336	-0.0047358	0.0236454
2011 Q3	0.0092391	-0.0000450	0.0185232	-0.0049597	0.0234380
2011 Q4	0.0087615	-0.0005283	0.0180512	-0.0054460	0.0229689
2012 Q1	0.0084612	-0.0008297	0.0177521	-0.0057480	0.0226704
2012 Q2	0.0085296	-0.0007614	0.0178205	-0.0056797	0.0227389
2012 Q3	0.0082633	-0.0010278	0.0175543	-0.0059461	0.0224727
2012 Q4	0.0081763	-0.0011148	0.0174674	-0.0060333	0.0223858
2013 Q1	0.0083821	-0.0009107	0.0176749	-0.0058300	0.0225942
2013 Q2	0.0084189	-0.0008740	0.0177117	-0.0057934	0.0226311
2013 Q3	0.0084354	-0.0008576	0.0177284	-0.0057770	0.0226478
2013 Q4	0.0085017	-0.0007914	0.0177948	-0.0057108	0.0227143
2014 Q1	0.0085588	-0.0007343	0.0178520	-0.0056538	0.0227715
2014 Q2	0.0085508	-0.0007424	0.0178439	-0.0056619	0.0227634
2014 Q3	0.0085916	-0.0007016	0.0178848	-0.0056211	0.0228043
2014 Q4	0.0086085	-0.0006846	0.0179017	-0.0056041	0.0228212
2015 Q1	0.0085811	-0.0007121	0.0178743	-0.0056316	0.0227938
2015 Q2	0.0085751	-0.0007181	0.0178683	-0.0056376	0.0227879
2015 Q3	0.0085751	-0.0007181	0.0178683	-0.0056376	0.0227878
2015 Q4	0.0085661	-0.0007271	0.0178593	-0.0056466	0.0227789
2016 Q1	0.0085560	-0.0007372	0.0178492	-0.0056567	0.0227687
2016 Q2	0.0085568	-0.0007364	0.0178500	-0.0056559	0.0227696
2016 Q3	0.0085507	-0.0007425	0.0178439	-0.0056620	0.0227634
2016 Q4	0.0085477	-0.0007455	0.0178409	-0.0056651	0.0227604
2017 Q1	0.0085512	-0.0007420	0.0178444	-0.0056616	0.0227639
2017 Q2	0.0085521	-0.0007411	0.0178453	-0.0056606	0.0227648

	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
2017 Q3	0.0085518	-0.0007414	0.0178450	-0.0056610	0.0227645
2017 Q4	0.0085529	-0.0007403	0.0178461	-0.0056598	0.0227656
2018 Q1	0.0085546	-0.0007386	0.0178478	-0.0056581	0.0227674
2018 Q2	0.0085546	-0.0007386	0.0178478	-0.0056582	0.0227673
2018 Q3	0.0085555	-0.0007377	0.0178487	-0.0056573	0.0227682
2018 Q4	0.0085560	-0.0007372	0.0178492	-0.0056567	0.0227687

c. Use `slide` from the `tsibble` package to generate a rolling scheme forecast, in particular a sequence of 1 period ahead forecasts for the prediction subsample, 2009Q1-2018Q4.

Answer:

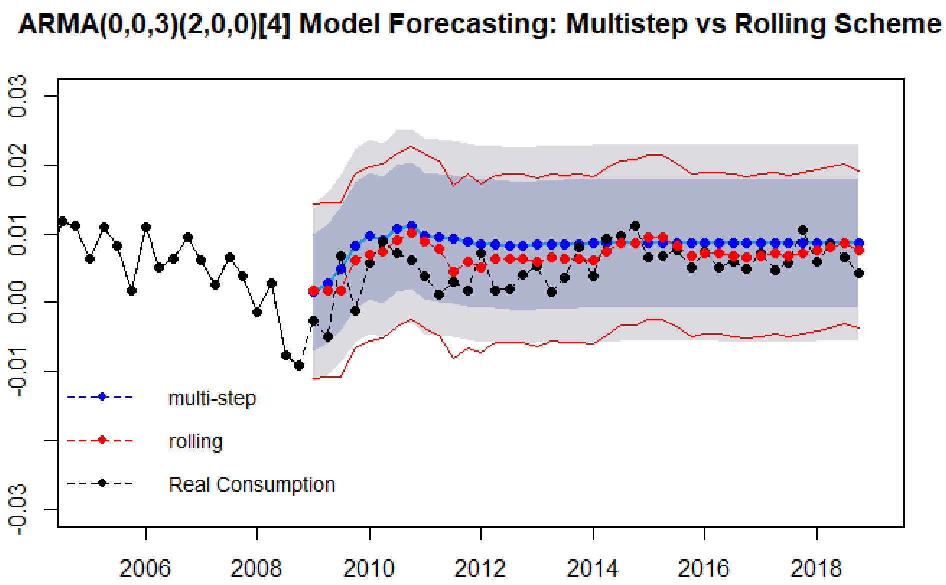
Since I cannot use `slide` from the `tsibble` package, I chose to use looping to generate a rolling scheme forecast. This code is also used for similar problems later.

```
forecast4arolling <- zoo()
firstQ <- 1955.25
lastQ <- 2008.75
ci1=0
ci2=0
for(i in 1:length(data4_fore)) {
  temp <- window(data4dlog, start = firstQ + (i-1)/4, end = lastQ + (i-1) / 4)
  data4_est.update <- arima(temp, order = c(0,0,3), seasonal = list(order =
c(2,0,0),period=4))
  forecast4arolling <- c(forecast4arolling, forecast(data4_est.update, h=1)$mean)
  ci1=c(ci1,forecast(data4_est.update,1,level=95)$upper)
  ci2=c(ci2,forecast(data4_est.update,1,level=95)$lower)
}

forecast4arolling <- as.ts(forecast4arolling)
ci1=ci1[2:81]
ci2=ci2[2:81]
ci1=ts(ci1, start=c(2009,1), end=c(2018,4), frequenc=4)
ci2=ts(ci2, start=c(2009,1), end=c(2018,4), frequenc=4)
```

	Qtr1	Qtr2	Qtr3	Qtr4
2009	0.001681852	0.001832660	0.001841681	0.006099423
2010	0.007037150	0.007471731	0.008998746	0.010102977
2011	0.008952416	0.007881130	0.004420621	0.006004313
2012	0.005040168	0.006326406	0.006380442	0.006371575
2013	0.005867143	0.006577653	0.006366830	0.006441542
2014	0.006130787	0.007368500	0.008630470	0.008731162
2015	0.009410460	0.009517493	0.008304843	0.006798723
2016	0.007153572	0.007162185	0.006828657	0.006598302
2017	0.006833607	0.007182021	0.006804777	0.007217253
2018	0.007579134	0.008019896	0.008582629	0.007696807

d. Plot the multistep forecast and the 1 step ahead rolling forecasts, with their confidence intervals.



e. Use accuracy to evaluate the out of sample accuracy of the two sets of forecasts.

Answer:

```
accuracy(forecast4a$mean, x =  data4_fore)
accuracy(forecast4arolling, x =  data4_fore)
```

Multi-step	ME	RMSE	MAE	MPE	MAPE
Test set	-0.0032002	0.0043014	0.0036079	-69.93423	132.3069

Rolling	ME	RMSE	MAE	MPE	MAPE
Test set	-0.0017187	0.0032471	0.0027052	-36.41866	96.05964

Compared with the multi-step forecasting method, the rolling forecasting error is smaller. Therefore, in this case, the accuracy of the rolling scheme prediction is greater than that of the multi-step prediction method.

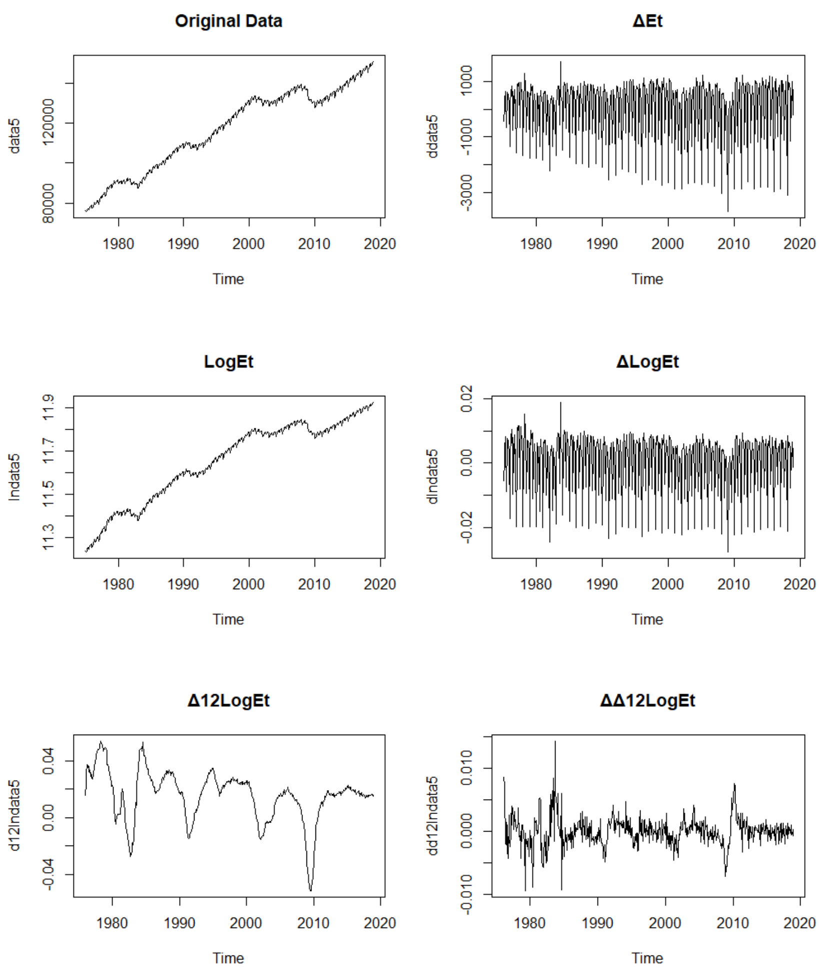
5. (Computer Exercise, 20 points)

Obtain monthly data for Total Nonfarm Payroll Employment for the 1975M1-2018M12 sample (Not Seasonally Adjusted) available on FRED under code PAYNSA.

a. Construct the following transformed time series

• change in Total Nonfarm Payroll Employment $\Delta E_t = E_t - E_{t-1}$
• log of Total Nonfarm Payroll Employment $logE_t$
• log change in Total Nonfarm Payroll Employment $\Delta \log E_t = logE_t - logE_{t-1}$
• 12 months log change in Total Nonfarm Payroll Employment $\Delta 12 logE_t = logE_t - logE_{t-12}$
• twice differenced Total Nonfarm Payroll Employment $\Delta\Delta 12 logE_t = \Delta 12 logE_t - \Delta 12 logE_{t-1}$
Plot the original and the transformed time series. Comment on their trends, volatility, seasonal patterns.

Answer:



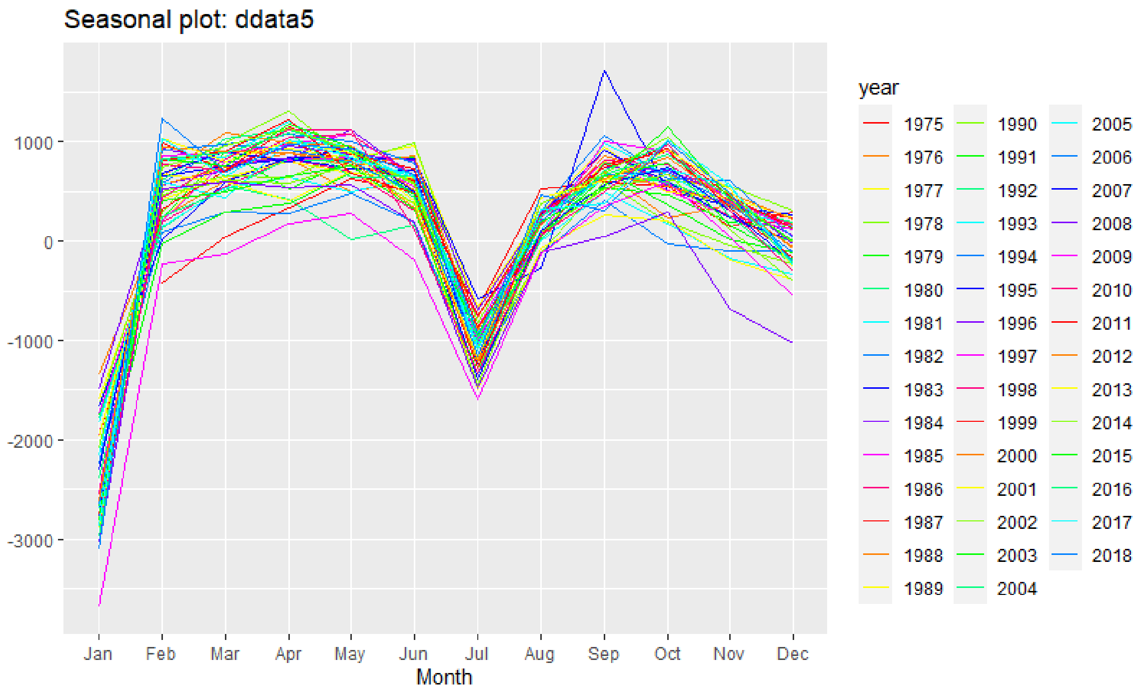
The image shapes of Original Data and $logE_t$ are similar, and both show a very obvious upward trend. If we zoom in, we can see the seasonality in the plots of Original Data and $logE_t$, that is, the rough volatility. Although the two plots show an overall upward trend, we can still see troughs around 1983, 1992, 2003 and 2010. We can see very obvious fluctuations in the ΔE_t and $\Delta logE_t$ images. The volatility in ΔE_t gradually increases with time, and the amplitude of $\Delta log E_t$ is more balanced. Both the $\Delta 12 log E_t$ and $\Delta\Delta 12 log E_t$ images show significant volatility at 1983 and 2010, especially 2010 in $\Delta 12 log E_t$ shows the lowest point.

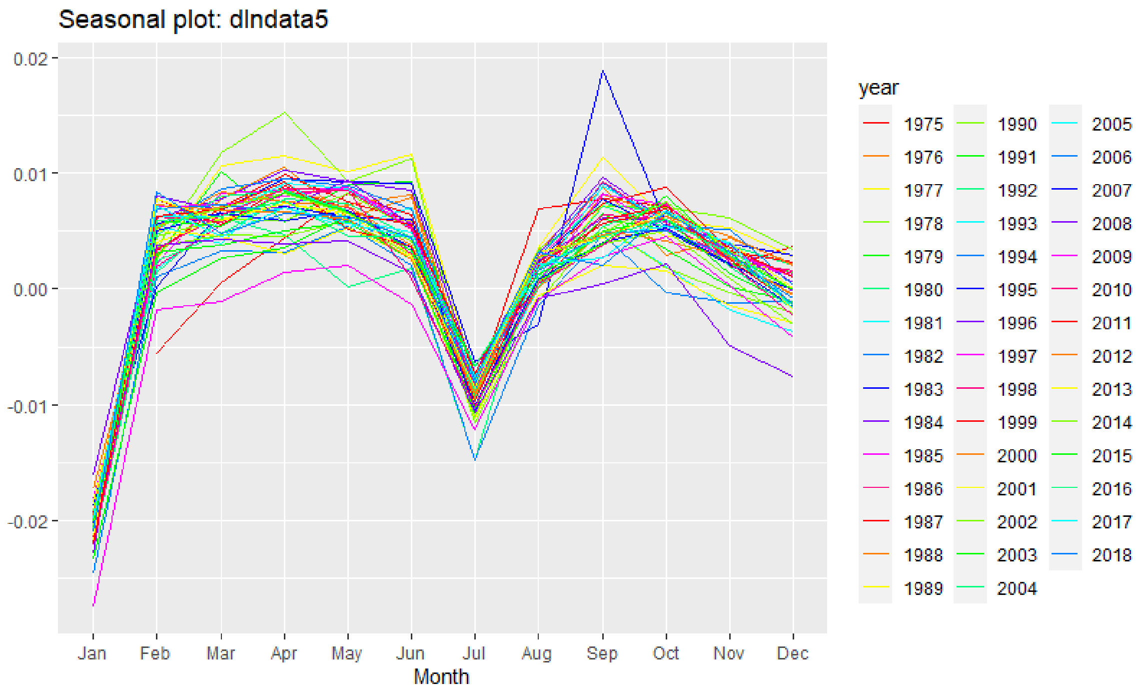
b. Use `ggseasonplot` to create seasonal plots for ΔE_t and $\Delta log E_t$. Comment on the seasonal patterns.

Answer:

I use `ggseasonplot` here

```
ggseasonplot(ddata5,col=rainbow(12))
ggseasonplot(dlndata5,col=rainbow(12))
```

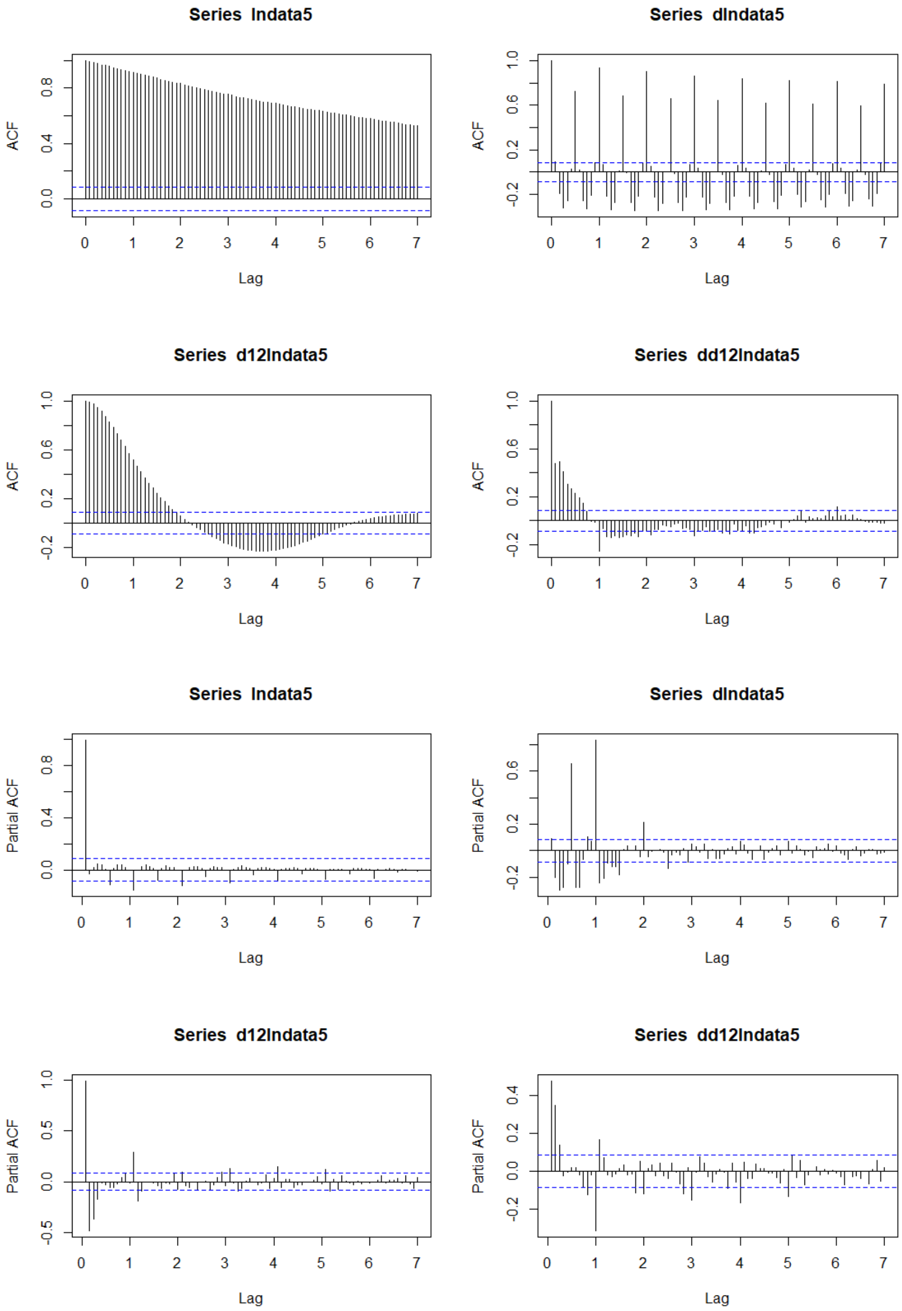




Both images show that spring is often the time when employment grows the fastest, that is, from February to June. And almost every July, employment growth is suddenly the lowest and negative, which is the lowest period of the whole year. However this situation will improve from August every year, however, after October, employment growth has gradually declined again.

c. Plot ACF and PACF for $logE_t$, $\Delta logE_t$, $\Delta 12logE_t$, $\Delta\Delta 12logE_t$ comment on their shape.

Answer:



The ACF of $logE_t$ is very significant in the first 7 years. However, the slow decay of its ACF indicates that there is a unit root and that the data is non-stationary. The ACF of $\Delta logE_t$ seems to deal with the unit root problem and has significant correlation in every half year and every year which shows a seasonality, that of $\Delta 12logE_t$ is significant in the first 5 years and that of $\Delta\Delta 12logE_t$ is obviously significant at year 0 and year 1.

The PACF of $logE_t$ has significant correlation in every year. The PACF of $\Delta logE_t$ has significant correlation in first 6 half years, that of $\Delta 12logE_t$ is significant in the first year and that of $\Delta\Delta 12logE_t$ is obviously significant at 0, 1, 2, 3, 4 and 5.

In summary, only $logE_t$ shows there exists a unit root. But $\Delta logE_t$, $\Delta 12logE_t$, $\Delta\Delta 12logE_t$ all show stationarity.

d. Perform the ADF and KPSS tests on $logE_t$, $\Delta 12 logE_t$, $\Delta\Delta 12 logE_t$. Summarize the results.

Answer:

In this question, We can use the method recommended by the professor in the class, using package `urca`, and then for ADF test, use code `ur.df(Date, type ="trend", selectlags = "AIC")`. For KPSS test, use code `ur.kpss(Data, type ="tau", lags = "long")`. I choose to use `adf.test` in `library(tseries)` for ADF test, and then use `ur.kpss` in `library(urca)` for KPSS test. This code is also used for similar problems later.

```
> adf.test(lndata5)

	Augmented Dickey-Fuller Test

data:  lndata5
Dickey-Fuller = -1.8277, Lag order = 8, p-value = 0.6512
alternative hypothesis: stationary

> adf.test(d12lndata5)

	Augmented Dickey-Fuller Test

data:  d12lndata5
Dickey-Fuller = -4.9778, Lag order = 8, p-value = 0.01
alternative hypothesis: stationary

> adf.test(dd12lndata5)

	Augmented Dickey-Fuller Test

data:  dd12lndata5
Dickey-Fuller = -5.6885, Lag order = 8, p-value = 0.01
alternative hypothesis: stationary
> summary(kpsslndata5) #difference stationary

#######################
# KPSS Unit Root Test #
#######################

Test is of type: tau with 18 lags.

Value of test-statistic is: 0.5803

Critical value for a significance level of:
                10pct  5pct 2.5pct  1pct
critical values 0.119 0.146  0.176 0.216

> summary(kpssd12lndata5) #stationary

#######################
# KPSS Unit Root Test #
#######################

Test is of type: tau with 18 lags.

Value of test-statistic is: 0.0685

Critical value for a significance level of:
                10pct  5pct 2.5pct  1pct
critical values 0.119 0.146  0.176 0.216

> summary(kpssdd12lndata5) #stationary

#######################
# KPSS Unit Root Test #
```

```
########################

Test is of type: tau with 18 lags.

Value of test-statistic is: 0.0233

Critical value for a significance level of:
                10pct  5pct 2.5pct  1pct
critical values 0.119 0.146  0.176 0.216
```

Test	Result
adf - $logE_t$	$\alpha = 1$, there is a unit root. The basic alternate is that the time series is not stationary.
adf - $\Delta 12 logE_t$	$\|\alpha\| < 1$, there is not a unit root. The basic alternate is that the time series is stationary.
adf -$\Delta\Delta 12 logE_t$	$\|\alpha\| < 1$, there is not a unit root. The basic alternate is that the time series is stationary.
kpss - $logE_t$	Difference stationary
kpss - $\Delta 12 logE_t$	Stationary. An observable time series is stationary around a deterministic trend.
kpss - $\Delta\Delta 12 logE_t$	Stationary. An observable time series is stationary around a deterministic trend.

e. Split the sample into two parts: estimation sample from 1975M1 to 2014M12, and prediction sample from 2015M1 to 2018M12. Use ACF and PACF from (c) to identify and estimate a suitable model for $\Delta\Delta 12 logE_t$ using `arima`. Check the estimated model for adequacy - diagnose residuals using `ggtsdiag`.

Answer:

According to the ACF plot of $\Delta\Delta 12 logE_t$ in c The ACF does not appear to be dying out, which suggests that an additional amount of differencing is required. Again, there is a choice between differencing at s = 1 and at s = 12. Because there is no strong seasonal pattern, we will try differencing at s = 1.

I solve this problem in 2 ways. I first use function with `arima` to choose the lowest AIC.

```
PQmax=6
AICs<-matrix(nrow=PQmax+1,ncol=PQmax+1)
rownames(AICs)<-c('AR0','AR1','AR2','AR3','AR4','AR5','AR6')
colnames(AICs)<-c('MA0','MA1','MA2','MA3','MA4','MA5','MA6')

for(ar in 0:PQmax){
  for(ma in 0:PQmax){
    fit <- arima(dd12lndata5a, c(ar,0,ma))
    AICs[ar+1,ma+1] <-AIC(fit)
  }}

AICs
minimizer = which(AICs==min(AICs), arr.ind =TRUE)
minimizer
```

Then I also use `auto.arima` to find a best model.

When using function with `arima`, if we set `PQmax=6`, the result should be:

	MA0	MA1	MA2	MA3	MA4	MA5	MA6
AR0	-4264.095	-4334.258	-4391.761	-4431.998	-4444.194	-4452.398	-4451.007
AR1	-4391.149	-4451.064	-4464.240	-4462.241	-4461.696	-4459.916	-4460.744
AR2	-4456.597	-4461.641	-4462.241	-4475.424	-4474.114	-4493.099	-4495.574
AR3	-4464.155	-4462.783	-4461.686	-4473.978	-4478.816	-4482.534	-4504.227
AR4	-4463.081	-4461.303	-4459.347	-4477.093	-4467.183	-4476.058	-4500.420
AR5	-4461.461	-4459.484	-4472.930	-4489.661	-4492.660	-4484.826	**-4515.609**
AR6	-4459.516	-4472.998	-4484.510	-4494.592	-4492.468	-4489.787	-4500.127

The best model in this way is ARMA(5, 6)

When using function with `arima`, if we set `PQmax=10`, the result should be:

	MA0	MA1	MA2	MA3	MA4	MA5	MA6	MA7	MA8	MA9	MA10
AR0	-4264.095	-4334.258	-4391.761	-4431.998	-4444.194	-4452.398	-4451.007	-4449.330	-4449.026	-4478.022	-4510.242
AR1	-4391.149	-4451.064	-4464.240	-4462.241	-4461.696	-4459.916	-4460.744	-4473.851	-4483.386	-4497.873	-4518.556
AR2	-4456.597	-4461.641	-4462.241	-4475.424	-4474.114	-4493.099	-4495.574	-4503.999	-4515.701	-4528.188	-4514.948
AR3	-4464.155	-4462.783	-4461.686	-4473.978	-4478.816	-4482.534	-4504.227	-4508.355	-4513.263	-4541.637	-4545.230
AR4	-4463.081	-4461.303	-4459.347	-4477.093	-4467.183	-4476.058	-4500.420	-4519.338	-4525.050	-4523.294	-4559.860
AR5	-4461.461	-4459.484	-4472.930	-4489.661	-4492.660	-4484.826	-4515.609	-4523.666	-4527.951	-4529.807	-4565.651
AR6	-4459.516	-4472.998	-4484.510	-4494.592	-4492.468	-4489.787	-4500.127	-4504.649	-4525.364	-4557.424	-4572.610
AR7	-4457.637	-4472.803	-4484.173	-4496.303	-4496.001	-4522.258	-4515.718	-4522.864	-4540.220	-4564.354	**-4575.392**
AR8	-4455.787	-4471.236	-4486.616	-4494.305	-4506.462	-4499.492	-4522.897	-4525.320	-4560.320	-4564.739	-4574.296
AR9	-4458.816	-4467.396	-4483.296	-4499.752	-4528.784	-4527.235	-4520.483	-4525.144	-4567.904	-4560.095	-4572.421
AR10	-4464.772	-4482.759	-4486.040	-4500.784	-4532.269	-4538.386	-4557.285	-4535.306	-4518.004	-4560.530	-4571.097

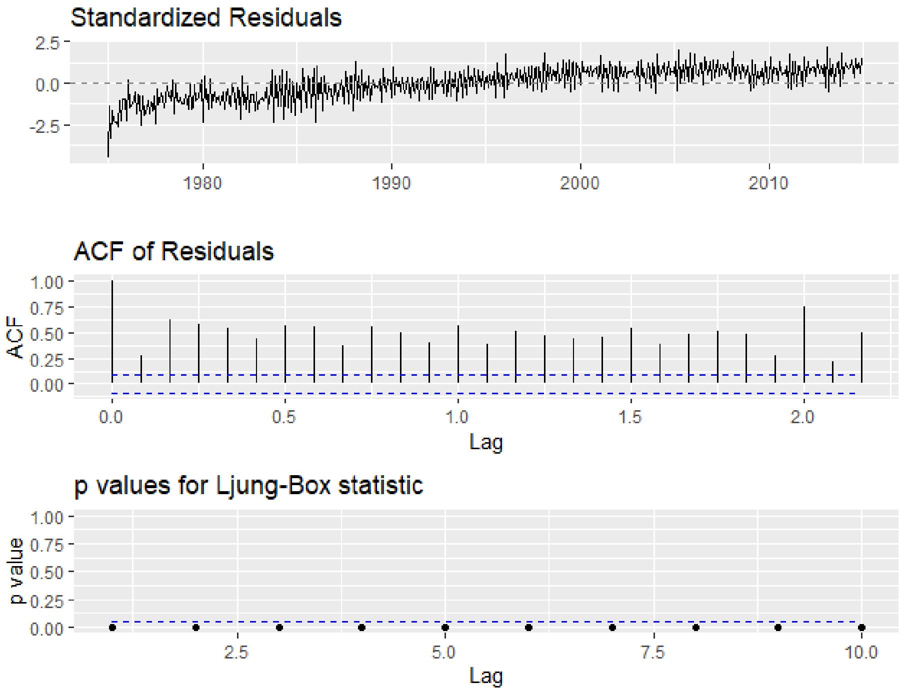
The best model in this way is ARMA(7, 10)

But judging from the original data, as well as the plots of ACF and PACF, we know that its plot also has seasonal characteristics, so this naive function judgment is not accurate. We use `auto.arima` to find the best model of $\Delta\Delta 12 logE_t$: ***ARIMA(3,0,0)(1,0,1)[12] with zero mean***.

f. Use `auto.arima` to find the best model for $logE_t$. Check the estimated model for adequacy - diagnose residuals using `ggtsdiag`.

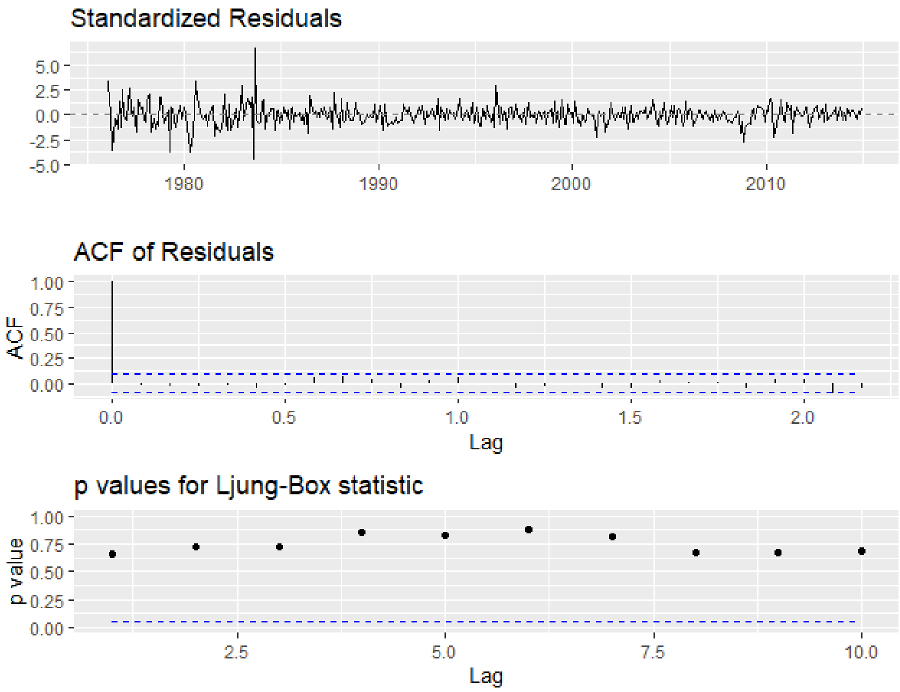
Answer:

Series: $logE_t$
ARIMA(0,0,4)(0,0,1)[12] with non-zero mean.



However, through the Ljung-Box test, we find the null of white noise is rejected, so the best model provided by `auto.arima` does not fit well.

Then we can try $\Delta\Delta 12 logE_t$, which we have already the best model: ***ARIMA(3,0,0)(1,0,1)[12] with zero mean***



The residuals are statistically uncorrelated, and statistically indistinguishable from zero. The p-value is high. The ACF results have no statistically significant coefficients. The null of white noise is not rejected. The model seems to fit well.

g. Use slide from `tsibble` package to create a rolling scheme sequence of 1 period ahead forecasts for the prediction subsample 2015M1-2018M12 using the same model specification as in (f).

Answer:

Rolling scheme prediction for Series: $logE_t$ ***ARIMA(0,0,4)(0,0,1)[12] with non-zero mean***.

	Jan	Feb	Mar	Apr	May	Jun
2015	11.83486	11.81790	11.85354	11.83592	11.85713	11.85131
2016	11.84830	11.84152	11.85620	11.85768	11.86787	11.86481
2017	11.86424	11.85553	11.88514	11.86857	11.87623	11.88211
2018	11.87140	11.87662	11.88636	11.89209	11.89596	11.89849
	Jul	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**
2015	11.85173	11.84961	11.84580	11.87329	11.86036	11.87063
2016	11.87143	11.85662	11.86945	11.87016	11.88152	11.87881
2017	11.89119	11.87991	11.87886	11.89963	11.89169	11.90023
2018	11.90123	11.89042	11.90135	11.89930	11.91373	11.90550

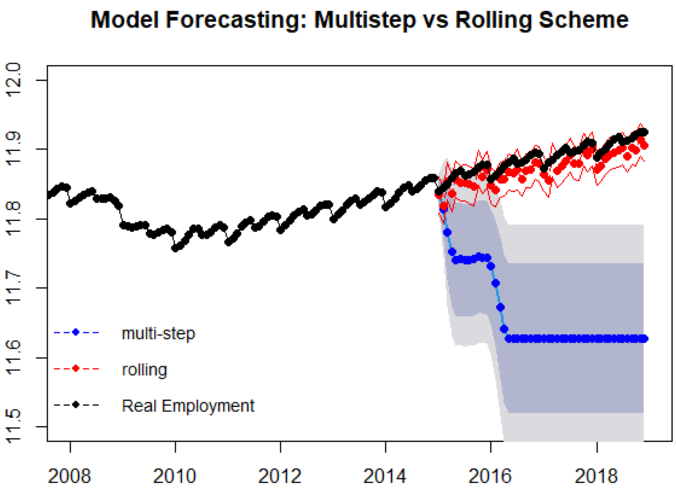
Rolling scheme prediction for Series: $\Delta\Delta 12 logE_t$: ***ARIMA(3,0,0)(1,0,1)[12] with zero mean***..

	Jan	Feb	Mar	Apr	May	Jun
2015	3.777170e-04	1.080105e-03	8.820155e-06	-8.224175e-04	-1.720849e-04	-7.196533e-04
2016	-1.942129e-05	9.165181e-05	7.482691e-04	-7.289134e-04	-1.409644e-04	-6.161682e-04
2017	-3.360577e-04	8.079881e-07	3.666489e-04	1.271999e-04	9.099831e-04	-1.541523e-03
	Jul	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**
2015	-8.175453e-04	-2.727912e-05	2.331028e-04	-1.005195e-03	-6.718464e-04	-8.065821e-04
2016	-1.970031e-03	7.243955e-04	1.229317e-03	-7.838769e-04	-5.225342e-04	-1.178350e-03
2017	-1.236026e-03	3.473993e-04	2.739574e-04	1.250245e-04	-7.360958e-04	

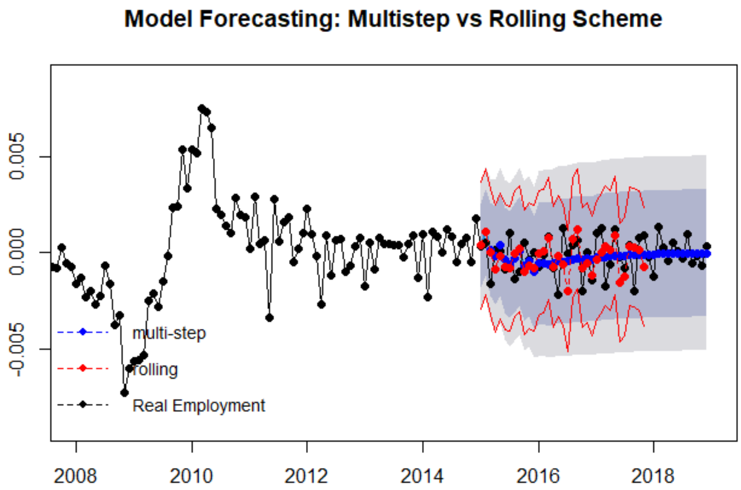
h. Plot the forecast for E_t from (g) together with its confidence intervals and the actual data for the period 2008M1-2018M12.

Answer:

Here we consider E_t as $logE_t$. ***ARIMA(0,0,4)(0,0,1)[12] with non-zero mean***.



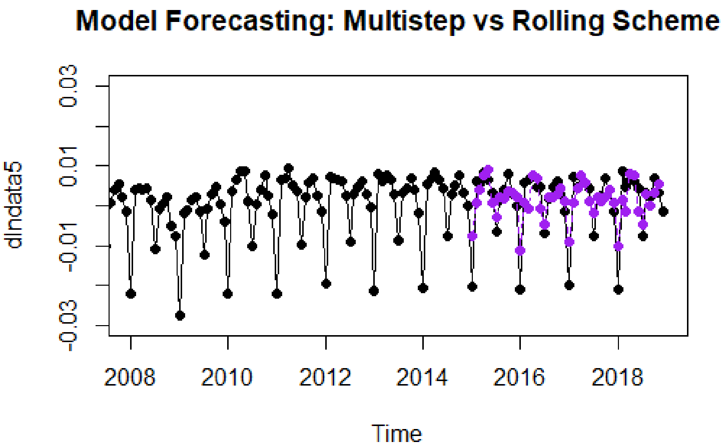
But since the result of the best model of $logE_t$ is not good enough, I also want to make a plot for $\Delta\Delta 12 logE_t$. ***ARIMA(3,0,0)(1,0,1)[12] with zero mean***.



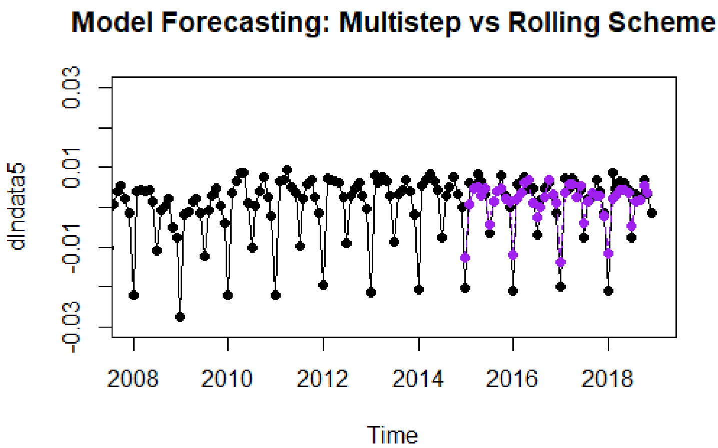
i. Use the forecast for E_t from (g) to construct the forecast for ΔE_t , plot it together with the actual data.

Answer:

Here we consider ΔE_t as $\Delta log E_t$. If we use the forecast ***ARIMA(0,0,4)(0,0,1)[12] with non-zero mean***, the plot is:



However if we first use `auto.arima` to find the best model for $\Delta log E_t$, we will have a model of dlndata5a **ARIMA(1,0,0)(0,0,2)[12] with non-zero mean** . Then the plot will be:

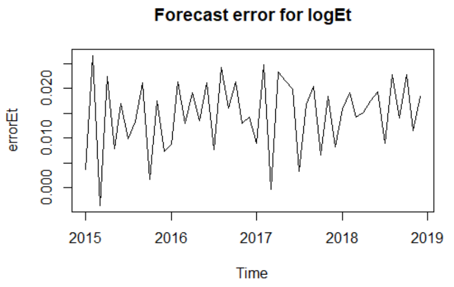


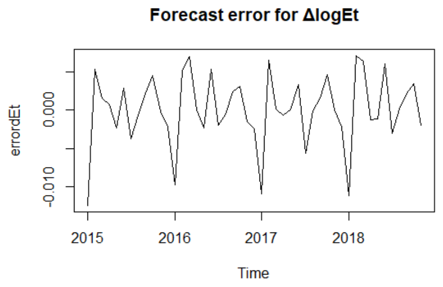
j. Construct and plot the forecast errors for E_t and for ΔE_t .

Answer:

Here we consider E_t as $logE_t$ and ΔE_t as $\Delta logE_t$.

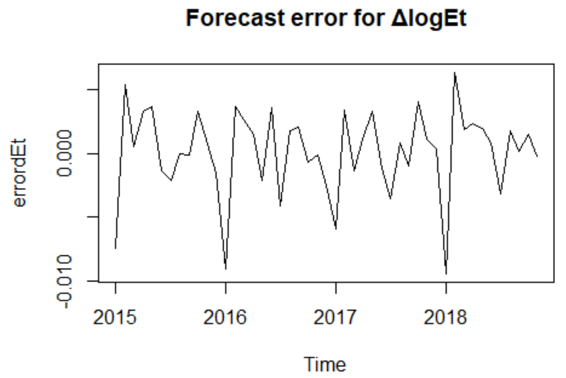
The model of $logE_t$ and $\Delta logE_t$ ***ARIMA(0,0,4)(0,0,1)[12] with non-zero mean***.





The forecast error for $\Delta logE_t$ seems to show a seasonality in every beginning of the year.

However if use model of $\Delta logE_t$ ***ARIMA(1,0,0)(0,0,2)[12] with non-zero mean***. The plot is:



6. (Computer Exercise, 6 points)

The response of hours worked to different shocks has been studied extensively since Gali (1999), who argued that hours worked show a decline in response to a positive technology shock. In this problem, you will replicate some of his results. Obtain the following two quarterly time series for the period 1947Q1-2017Q4 from FRED: labor productivity, measured as Nonfarm Business Sector: Real Output Per Hour of All Persons OPHNFB and for total hours worked, measured as Nonfarm Business Sector: Hours of All Persons HOANBS.

a. Test the log of real output per hour $y_{1,t} = logOPHNFB_t$ and the log of hours $y_{2,t} = logHOANBS_t$ for the presence of unit root using ADF test. Afterwards apply the ADF unit root test also to the first differences, $\Delta y_{1,t}$ and $\Delta y_{1,t}$. Comment on results.

Answer:

```
> adf.test(lnlabor)

    Augmented Dickey-Fuller Test

data:  lnlabor
Dickey-Fuller = -2.3352, Lag order = 6, p-value = 0.4349
alternative hypothesis: stationary

> adf.test(lnhours)

    Augmented Dickey-Fuller Test
```

```
data:  lnhours
Dickey-Fuller = -1.974, Lag order = 6, p-value = 0.5871
alternative hypothesis: stationary

> adf.test(dlnlabor)

    Augmented Dickey-Fuller Test

data:  dlnlabor
Dickey-Fuller = -6.8656, Lag order = 6, p-value = 0.01
alternative hypothesis: stationary

> adf.test(dlnhours)

    Augmented Dickey-Fuller Test

data:  dlnhours
Dickey-Fuller = -6.0531, Lag order = 6, p-value = 0.01
alternative hypothesis: stationary
```

Test	Result
adf - $logOPHNFB_t$	$\alpha = 1$, there is a unit root, The basic alternate is that the time series is not stationary.
adf - $logHOANBS_t$	$\alpha = 1$, there is a unit root, The basic alternate is that the time series is not stationary.
adf - $\Delta logOPHNFB_t$	$\|\alpha\| < 1$, there is not a unit root, The basic alternate is that the time series is stationary.
adf - $\Delta logHOANBS_t$	$\|\alpha\| < 1$, there is not a unit root, The basic alternate is that the time series is stationary.

b. Estimate a bivariate reduced form VAR for $y_t = (\Delta y_{1,t}, \Delta y_{2,t})'$, using AIC information criteria to select number of lags.

Answer:

Use `VARselect` and `VAR` function.

```
y<-cbind(dlnlabor,dlnhours)
y<-na.trim(y)
y<-sweep(y,2,apply(y,2,mean))
VARselect(y,lag.max=8)
#AIC  lag=3
var1<-VAR(y,p=3, type="const")
summary(var1)
```

AIC(n)	HQ(n)	SC(n)	FPE(n)
3	2	2	3

We select lag=3 through AIC information criteria.

VAR Estimation Result for $\Delta y_{1,t}$:

	Estimate	Std. Error	t value	Pr(>|t|)
dlnlabor.l1	-8.490e-02	5.865e-02	-1.447	0.14893
dlnhours.l1	6.439e-02	7.035e-02	0.915	0.36082
dlnlabor.l2	6.167e-02	5.547e-02	1.112	0.26715
dlnhours.l2	-1.907e-01	8.301e-02	-2.297	***0.02238***
dlnlabor.l3	-3.087e-03	5.467e-02	-0.056	0.95502
dlnhours.l3	-1.985e-01	7.227e-02	-2.746	***0.00643***
const	-5.431e-05	4.730e-04	-0.115	0.90868

VAR Estimation Result for $\Delta y_{2,t}$:

	Estimate	Std. Error	t value	Pr(>|t|)
dlnlabor.l1	1.105e-01	5.053e-02	2.186	***0.0296***
dlnhours.l1	6.273e-01	6.060e-02	10.352	***<2e-16***
dlnlabor.l2	1.095e-01	4.778e-02	2.291	***0.0227***
dlnhours.l2	-2.642e-02	7.151e-02	-0.369	0.7121
dlnlabor.l3	7.608e-02	4.709e-02	1.615	0.1074
dlnhours.l3	-3.228e-02	6.226e-02	-0.518	0.6046
const	-1.283e-05	4.075e-04	-0.031	0.9749

The results of $dlnhours.l2$, $dlnhours.l3$, $dlnlabor.l1$, $dlnhours.l1$ and $dlnlabor.l2$ are significant. This output is not good enough.

Code:

```
######4
library(Quandl)
Quandl.api_key("KmxULt3z1Vz1neVxGioB")
dataq4 <- Quandl("FRED/PCECC96", type="zoo")
dataq4dlog <- diff(log(dataq4), lag=1, differences = 1)
data4dlog <-window(dataq4dlog, start="1955 Q1", end="2018 Q4")
data4_est<-window(data4dlog,start="1955 Q1", end="2008 Q4")
data4_fore<-window(data4dlog,start="2009 Q1", end="2018 Q4")

#(a)
library(forecast)
estimated4a<-auto.arima(data4_est,ic="aic", stationary = TRUE, stepwise = FALSE,
                        approximation = FALSE)
estimated4a
##best model ARIMA(0,0,3)(2,0,0)[4] MA(3)for non seasonal part and AR(2)[4] for seasonal
part
library(ggfortify)
ggtsdiag(estimated4a)
##white noise

#(b)
forecast4a <- forecast(estimated4a,h=40)
forecast4a
```

```
#(c)
forecast4arolling <- zoo()
firstQ <- 1955.25
lastQ <- 2008.75
ci1=0
ci2=0
for(i in 1:length(data4_fore)) {
  temp <- window(data4dlog, start = firstQ + (i-1)/4, end = lastQ + (i-1) / 4)
  data4_est.update <- arima(temp, order = c(0,0,3), seasonal = list(order =
c(2,0,0),period=4))
  forecast4arolling <- c(forecast4arolling, forecast(data4_est.update, h=1)$mean)
  ci1=c(ci1,forecast(data4_est.update,1,level=95)$upper)
  ci2=c(ci2,forecast(data4_est.update,1,level=95)$lower)
}

forecast4arolling <- as.ts(forecast4arolling)
ci1=ci1[2:81]
ci2=ci2[2:81]
ci1=ts(ci1, start=c(2009,1), end=c(2018,4), frequenc=4)
ci2=ts(ci2, start=c(2009,1), end=c(2018,4), frequenc=4)

#(d)
plot(forecast4a, type="o", pch=16, xlim=c(2005,2019), ylim=c(-0.03,0.03),
     main="ARMA(0,0,3)(2,0,0)[4] Model Forecasting: Multistep vs Rolling Scheme")
lines(forecast4a$mean, type="p", pch=16, lty="dashed", col="blue")
lines(data4dlog, type="o", pch=16, lty="dashed")
lines(forecast4arolling, type="o", pch=16, lty="dashed",col="red")
lines(ci1, col="red")
lines(ci2, col="red")
legend("bottomleft", c("multi-step", "rolling","Real Consumption"), pch =
c(16,16),col=c("blue","red","black"),lty="dashed", cex = 0.9, bty="n")

#(e)
accuracy(forecast4a$mean, x =  data4_fore)
accuracy(forecast4arolling, x =  data4_fore)
##Rolling scheme forecasting method has less error than the multi-step forecating method

######5
Quandl.api_key("KmxULt3z1Vz1neVxGioB")
dataq5 <- Quandl("FRED/PAYNSA", type="zoo")
data5 <- window(dataq5,start="1月 1975", end="12月 2018")
data5 <-as.ts(data5)

#(a)
ddata5=diff(data5,lag=1,difference=1)
lndata5=log(data5)
dlndata5=diff(lndata5,lag=1,difference=1)
d12lndata5=diff(lndata5,lag=12,difference=1)
dd12lndata5=diff(d12lndata5,lag=1,difference=1)
plot.ts(data5, main="Original Data")
plot.ts(ddata5, main="ΔEt")
plot.ts(lndata5, main="LogEt")
plot.ts(dlndata5, main="ΔLogEt")
plot.ts(d12lndata5, main="Δ12LogEt")
plot.ts(dd12lndata5, main="ΔΔ12LogEt")
##We can clearly see that log lp and log twh are not stationary.
##However, the first differences of both variables do look somewhat stationary.

#(b)
ggseasonplot(ddata5,col=rainbow(12))
ggseasonplot(dlndata5,col=rainbow(12))

#(c)
acf(lndata5, lag=84)
```

```r
acf(dlndata5, lag=84)
acf(d12lndata5, lag=84)
acf(dd12lndata5, lag=84)
pacf(lndata5, lag=84)
pacf(dlndata5, lag=84)
pacf(d12lndata5, lag=84)
pacf(dd12lndata5, lag=84)

#(d)
library(tseries)
library(urca)
#adflndata5<-ur.df(lndata5, type ="trend", selectlags = "AIC")
#adfd12lndata5<-ur.df(d12lndata5, type ="trend", selectlags = "AIC")
#adfdd12lndata5<-ur.df(dd12lndata5, type ="trend", selectlags = "AIC")

adf.test(lndata5)
adf.test(d12lndata5)
adf.test(dd12lndata5)

kpsslndata5<-ur.kpss(lndata5, type ="tau", lags = "long")
kpssd12lndata5<-ur.kpss(d12lndata5, type ="tau", lags = "long")
kpssdd12lndata5<-ur.kpss(dd12lndata5, type ="tau", lags = "long")

#summary(adflndata5)
#summary(adfd12lndata5)
#summary(adfdd12lndata5)

summary(kpsslndata5) #difference stationary
summary(kpssd12lndata5) #stationary
summary(kpssdd12lndata5) #stationary

#(e)
dataq5a <- window(dataq5,start="1月 1975", end="12月 2014")
data5a <- as.ts(dataq5a)
ddata5a=diff(data5a,lag=1,difference=1)
lndata5a=log(data5a)
dlndata5a=diff(lndata5a,lag=1,difference=1)
d12lndata5a=diff(lndata5a,lag=12,difference=1)
dd12lndata5a=diff(d12lndata5a,lag=1,difference=1)

dataq5b <- window(dataq5,start="1月 2015", end="12月 2018")
data5b <- as.ts(dataq5b)
ddata5b=diff(data5b,lag=1,difference=1)
lndata5b=log(data5b)
dlndata5b=diff(lndata5b,lag=1,difference=1)
d12lndata5b=diff(lndata5b,lag=12,difference=1)
dd12lndata5b=diff(d12lndata5b,lag=1,difference=1)

PQmax=6

AICs<-matrix(nrow=PQmax+1,ncol=PQmax+1)
rownames(AICs)<-c('AR0','AR1','AR2','AR3','AR4','AR5','AR6')
colnames(AICs)<-c('MA0','MA1','MA2','MA3','MA4','MA5','MA6')

for(ar in 0:PQmax){
  for(ma in 0:PQmax){
    fit <- arima(dd12lndata5a, c(ar,0,ma))
    AICs[ar+1,ma+1] <-AIC(fit)
  }}

AICs

minimizer = which(AICs==min(AICs), arr.ind =TRUE)
minimizer
```

```
PQmax=10

AICs<-matrix(nrow=PQmax+1,ncol=PQmax+1)
rownames(AICs)<-c('AR0','AR1','AR2','AR3','AR4','AR5','AR6','AR7','AR8','AR9','AR10')
colnames(AICs)<-c('MA0','MA1','MA2','MA3','MA4','MA5','MA6','MA7','MA8','MA9','MA10')

for(ar in 0:PQmax){
  for(ma in 0:PQmax){
    fit <- arima(dd12lndata5a, c(ar,0,ma))
    AICs[ar+1,ma+1] <-AIC(fit)
  }}

AICs

#7, 10
minimizer = which(AICs==min(AICs), arr.ind =TRUE)
minimizer

auto.arima(dd12lndata5a,ic="aic", stationary = TRUE, stepwise = FALSE,
           approximation = FALSE)

#(f)

estimated5a <- auto.arima(lndata5a,ic="aic", stationary = TRUE, stepwise = FALSE,
                          approximation = FALSE)
estimated5a1 <- auto.arima(dd12lndata5a,ic="aic", stationary = TRUE, stepwise = FALSE,
                           approximation = FALSE)
ggtsdiag(estimated5a)


ggtsdiag(estimated5a1)

#(g)
forecast5arolling <- zoo()
ci1=0
ci2=0
firstQ <- 1975+1/12
lastQ <- 2014+11/12
for(i in 1:length(lndata5b)) {
  temp <- window(lndata5, start = firstQ + (i-1)/12, end = lastQ + (i-1)/12)
  data5_est.update <- arima(temp, order = c(0,0,4), seasonal = list(order =
c(0,0,1),period=12))
  forecast5arolling <- c(forecast5arolling, forecast(data5_est.update, h=1)$mean)
  ci1=c(ci1,forecast(data5_est.update,1,level=95)$upper)
  ci2=c(ci2,forecast(data5_est.update,1,level=95)$lower)
}

forecast5arolling <- as.ts(forecast5arolling)
ci1=ci1[2:81]
ci2=ci2[2:81]
ci1=ts(ci1, start=c(2015,1), end=c(2018,12), frequenc=12)
ci2=ts(ci2, start=c(2015,1), end=c(2018,12), frequenc=12)
forecast5a<-forecast(estimated5a,h=48)

forecast5a1rolling <- zoo()
ci1=0
ci2=0
firstQ <- 1975+1/12
lastQ <- 2014+11/12
for(i in 1:length(dd12lndata5b)) {
  temp <- window(dd12lndata5, start = firstQ + (i-1)/12, end = lastQ + (i-1)/12)
  data5_est.update <- arima(temp, order = c(3,0,0), seasonal = list(order =
c(1,0,1),period=12))
```

```
  forecast5a1rolling <- c(forecast5a1rolling, forecast(data5_est.update, h=1)$mean)
  ci1=c(ci1,forecast(data5_est.update,1,level=95)$upper)
  ci2=c(ci2,forecast(data5_est.update,1,level=95)$lower)
}

forecast5a1rolling <- as.ts(forecast5a1rolling)
ci1=ci1[2:81]
ci2=ci2[2:81]
ci1=ts(ci1, start=c(2015,1), end=c(2018,12), frequenc=12)
ci2=ts(ci2, start=c(2015,1), end=c(2018,12), frequenc=12)
forecast5a1<-forecast(estimated5a1,h=48)

#(h)

plot(forecast5a, type="o", pch=16, xlim=c(2008,2019), ylim=c(11.5,12),
     main="Model Forecasting: Multistep vs Rolling Scheme")
lines(lndata5, type="o", pch=16)
lines(forecast5a$mean, type="p", pch=16, lty="dashed", col="blue")
lines(forecast5arolling, type="o", PI=TRUE,pch=16, lty="dashed",col="red")
lines(ci1, col="red")
lines(ci2, col="red")
legend("bottomleft", c("multi-step", "rolling","Real Employment"), pch =
c(16,16),col=c("blue","red","black"),lty="dashed", cex = 0.9, bty="n")

plot(forecast5a1, type="o", pch=16, xlim=c(2008,2019), ylim=c(-0.009,0.009),
     main="Model Forecasting: Multistep vs Rolling Scheme")
lines(dd12lndata5, type="o", pch=16)
lines(forecast5a1$mean, type="p", pch=16, lty="dashed", col="blue")
lines(forecast5a1rolling, type="o", PI=TRUE,pch=16, lty="dashed",col="red")
lines(ci1, col="red")
lines(ci2, col="red")
legend("bottomleft", c("multi-step", "rolling","Real Employment"), pch =
c(16,16),col=c("blue","red","black"),lty="dashed", cex = 0.9, bty="n")


#(i)
forecast5arollingdln <- zoo()
firstQ <- 1975+1/12
lastQ <- 2014+11/12
for(i in 1:length(dlndata5b)) {
  temp <- window(dlndata5, start = firstQ + (i-1)/12, end = lastQ + (i-1)/12)
  data5_estdln.update <- arima(temp, order = c(0,0,4), seasonal = list(order =
c(0,0,1),period=12))
  forecast5arollingdln <- c(forecast5arollingdln, forecast(data5_estdln.update, h=1)$mean)
}

forecast5arollingdln <- as.ts(forecast5arollingdln)
plot(dlndata5, type="o", pch=16, xlim=c(2008,2019), ylim=c(-0.03,0.03),
     main="Model Forecasting: Multistep vs Rolling Scheme")
lines(dlndata5, type="o",pch=16)
lines(forecast5arollingdln, type="o", pch=16, lty="dashed",col="purple")

estimateddlndata <- auto.arima(dlndata5a,ic="aic", stationary = TRUE, stepwise = FALSE,
                           approximation = FALSE)

forecast5arollingdlni2 <- zoo()
firstQ <- 1975+1/12
lastQ <- 2014+11/12
for(i in 1:length(dlndata5b)) {
  temp <- window(dlndata5, start = firstQ + (i-1)/12, end = lastQ + (i-1)/12)
  data5_estdln.update <- arima(temp, order = c(1,0,0), seasonal = list(order =
c(0,0,2),period=12))
  forecast5arollingdlni2 <- c(forecast5arollingdlni2, forecast(data5_estdln.update,
h=1)$mean)
```

```
}

forecast5arollingdlni2 <- as.ts(forecast5arollingdln)
plot(dlndata5, type="o", pch=16, xlim=c(2008,2019), ylim=c(-0.03,0.03),
     main="Model Forecasting: Multistep vs Rolling Scheme")
lines(dlndata5, type="o",pch=16)
lines(forecast5arollingdlni2, type="o", pch=16, lty="dashed",col="purple")

#(j)
errorEt<-lndata5-forecast5arolling
plot(errorEt, main="Forecast error for logEt")
errordEt<-dlndata5-forecast5arollingdln
plot(errordEt, main="Forecast error for ΔlogEt")
errordEt<-dlndata5-forecast5arollingdlni2
plot(errordEt, main="Forecast error for ΔlogEt")

######4
#(a)
Quandl.api_key("KmxULt3z1Vz1neVxGioB")
library(xts)
library(vars)
library(gdata)
library(tseries)
laboro<-Quandl("FRED/OPHNFB", type="zoo")
hourso<-Quandl("FRED/HOANBS", type="zoo")
labor<-window(laboro,start="1947 Q1", end="2017 Q4")
hours<-window(hourso,start="1947 Q1", end="2017 Q4")
lnlabor<-log(labor)
lnhours<-log(hours)
dlnlabor<-diff(lnlabor, lag=1, differences = 1)
dlnhours<-diff(lnhours, lag=1, differences = 1)
par(mfrow=c(2,2))
#adflnlabor<-ur.df(lnlabor, type ="trend", selectlags = "AIC")
#adflnhours<-ur.df(lnhours, type ="trend", selectlags = "AIC")
#adfdlnlabor<-ur.df(dlnlabor, type ="trend", selectlags = "AIC")
#adfdlnhours<-ur.df(dlnhours, type ="trend", selectlags = "AIC")
adf.test(lnlabor)
adf.test(lnhours)
adf.test(dlnlabor)
adf.test(dlnhours)

#(2)
y<-cbind(dlnlabor,dlnhours)
y<-na.trim(y)
y<-sweep(y,2,apply(y,2,mean))
VARselect(y,lag.max=8)
#AIC  lag=3
var1<-VAR(y,p=3, type="const")
summary(var1)
```