Statistical Methods for Discrete Response, Time Series, and Panel Data (W271): Lab 3

Kyle Chuang and Sullivan Swift

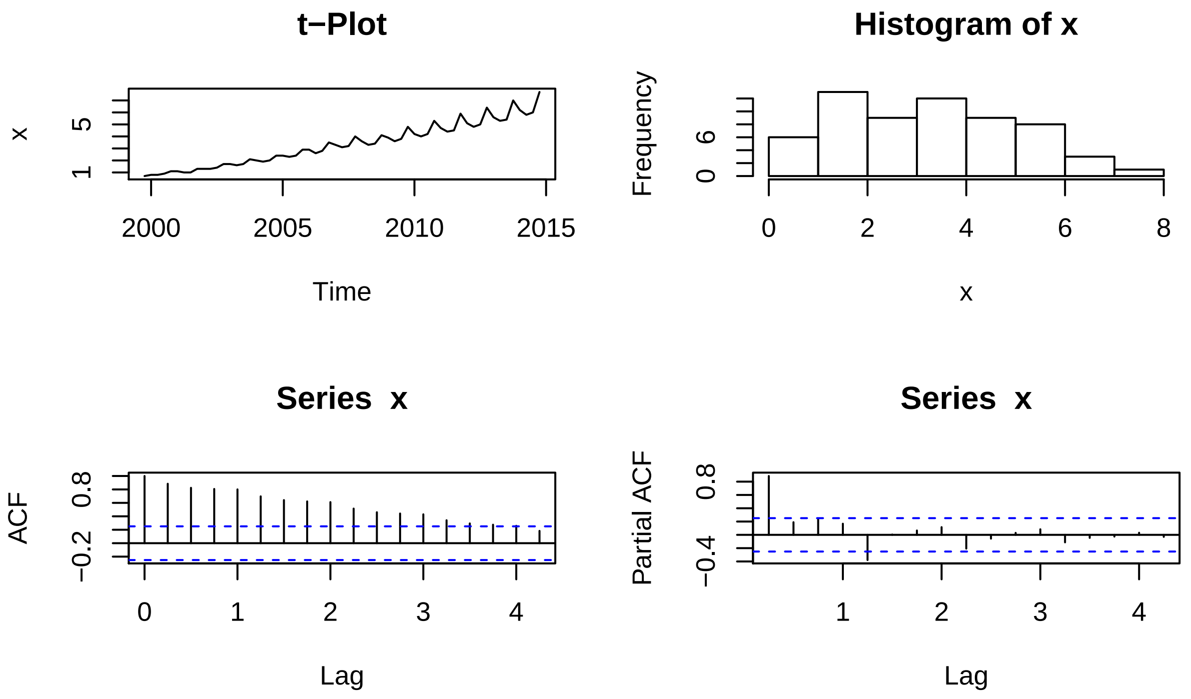
Question 1: Forecasting using a SARIMA model

Note: Custom function ts_plots() [plot time series], ts_resid() [plot residuals], forecast_exp() [exponentiate all values in forecasts objects], arimatable() [summarizes arima table] and run_arima_loop [loop through p and q variables on an arima model] are not included in the R pdf but is in the R-markdown file.

In the following report, we analyze and model quarterly data of E-Commerce Retail Sales as the Percent of Total Sales. Our goal is to use the data, ranging from Q4 1999 to Q4 2016, to forecast predictions for each quarter in 2017. First we explore the data and determine what models to further pursue. We selected two models to test in depth, including diagnostic tests and with in- and out-of-model sampling. We use our final model to make a 2017 prediction.

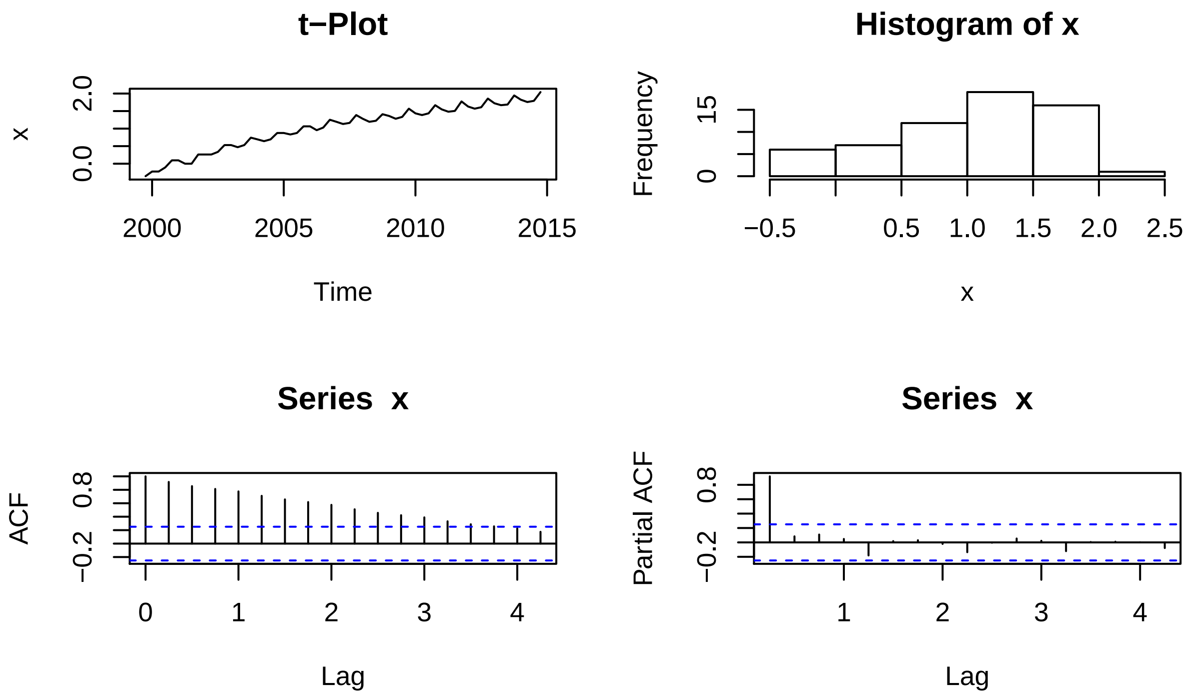
We begin by examining a time series plot, a histogram of the time series and the ACF and PACF plots.

```r
df = read.csv("ECOMPCTNSA.csv")
head(df)
# exclude 2015 & 2016
dfts = ts(df$ECOMPCTNSA, start = c(1999, 4), end = c(2014, 4), freq = 4)
ts_plots(dfts)
```



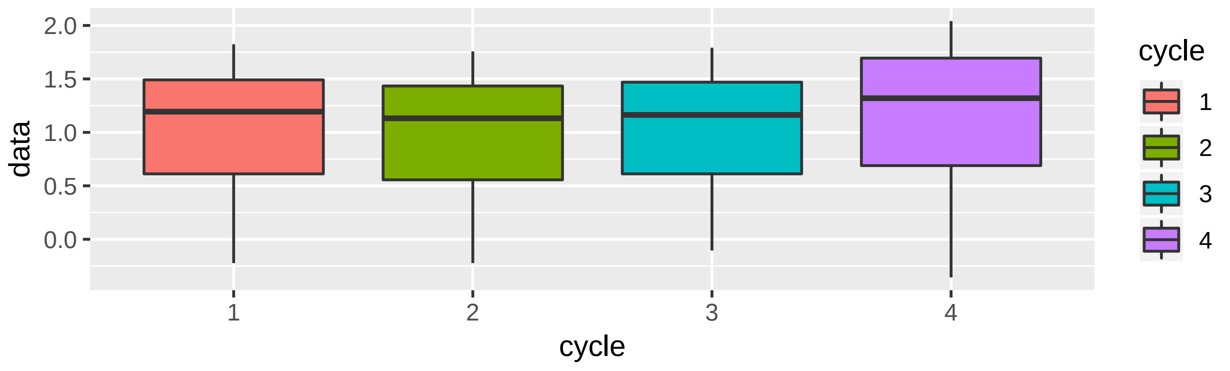
There is clearly a seaonsonal component based on the plot. Additionally, the plot seems to be potentially heteroskedastic. We will perform a `log` transformation on the time series to decrease the heteroskedasticity and re-analyze the time series from the starting point.

```r
dfts_log = log(dfts)
ts_plots(dfts_log)
```



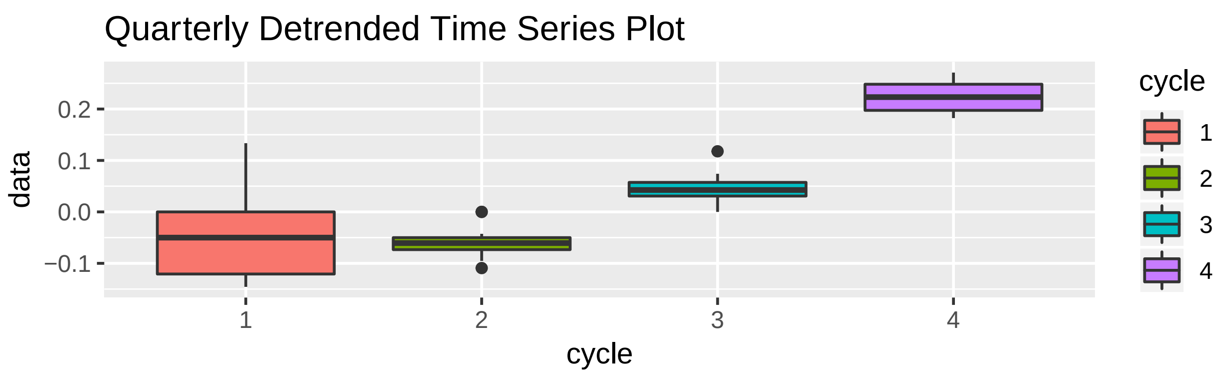
From the logged time series plots, there is a trend and seasonal component. The seasonality is likely to be quarterly from the t-plot. We will look at a box plot of the data by season.

```r
ggplot(data.frame(cycle = factor(cycle(dfts_log)), data = as.numeric(dfts_log)),
    aes(x = cycle, y = data, group = cycle, fill = cycle)) + geom_boxplot()
```



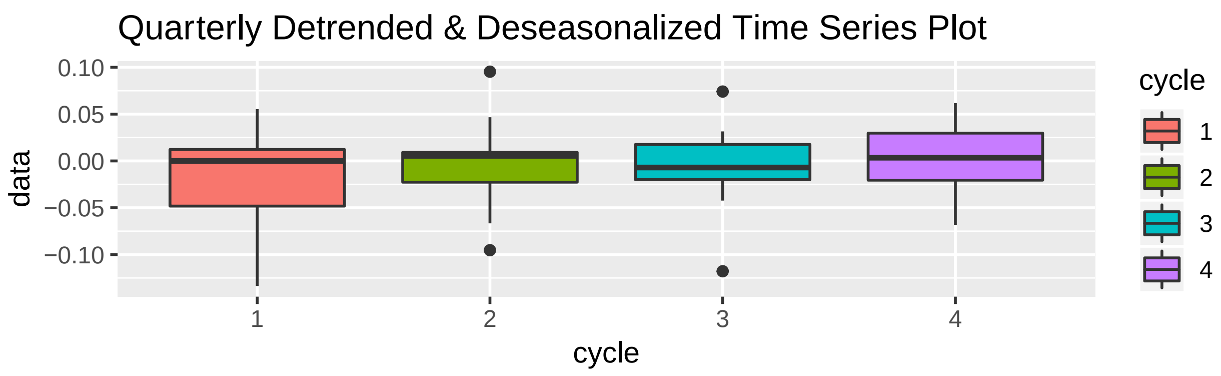
The seasonal boxplot does not show significant mean and variance differences in each of the quarters. However, this is possibly due to the trend component affecting the cycles. We will detrend the series by first differencing and re-examine the quarterly plots.

```r
tmp = diff(dfts_log, lag = 1)
ggplot(data.frame(cycle = factor(cycle(tmp)), data = as.numeric(tmp)), aes(x = cycle,
    y = data, group = cycle, fill = cycle)) + geom_boxplot() + ggtitle("Quarterly Detrended Ti
```



The detrended series shows a strong quarterly seasonality. The time series will be deseasonalized using a quarterly cycle.

```r
df_ds = diff(diff(dfts_log, lag = 1), lag = 4)
ggplot(data.frame(cycle = factor(cycle(df_ds)), data = as.numeric(df_ds)), aes(x = cycle,
    y = data, group = cycle, fill = cycle)) + geom_boxplot() + ggtitle("Quarterly Detrended & 
```



The plot of the detrended and deseasonalized data shows the quarterly mean and variance is now similar across the quarters, indicating a deseasonalized time series.

Augmented Dickey-Fuller and Phillips-Perron tests are performed on deseasonalized, detrended time series (the residuals). Both tests reject the non-stationary hypothesis. With a stationary time series, we can use an ARMA model to model the detrended, deseasonalized time series.

```r
adf.test(df_ds)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  df_ds
## Dickey-Fuller = -7.2107, Lag order = 3, p-value = 0.01
## alternative hypothesis: stationary
```

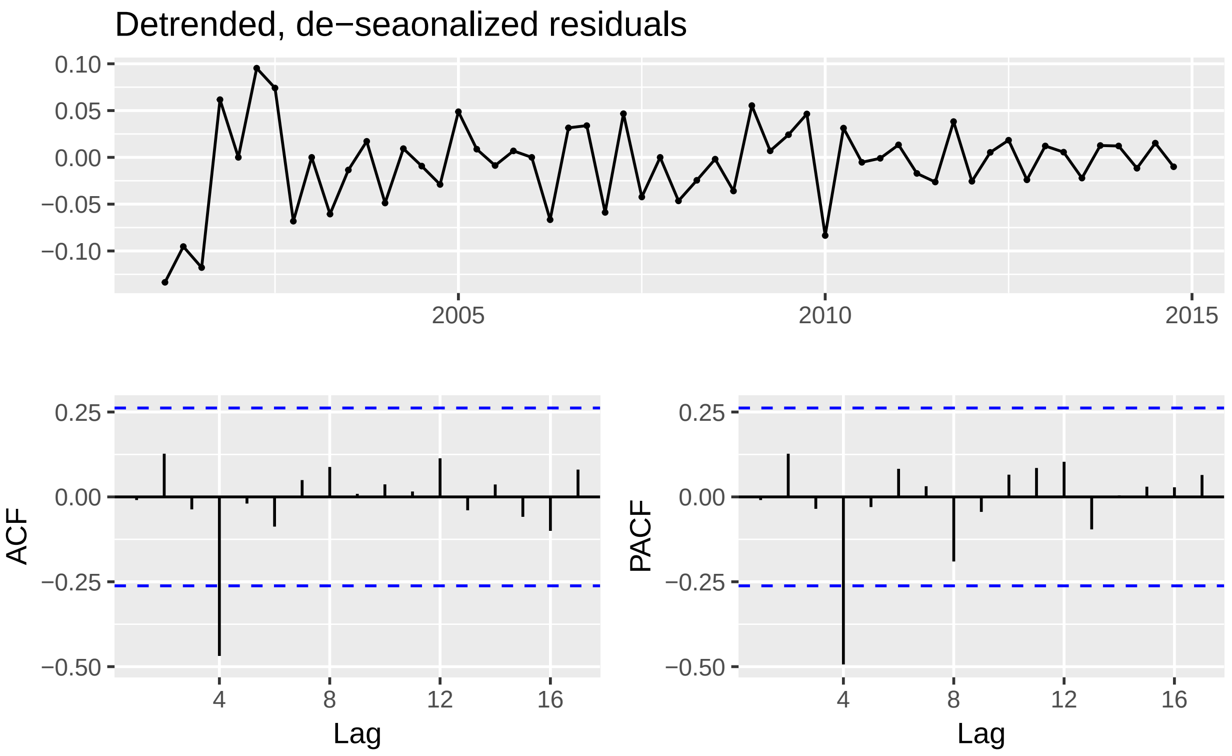
```r
pp.test(df_ds)
```

```
##
##  Phillips-Perron Unit Root Test
##
```

```
## data:  df_ds
## Dickey-Fuller Z(alpha) = -52.65, Truncation lag parameter = 3,
## p-value = 0.01
## alternative hypothesis: stationary
```

Since the there are $I(1)$ and $I(1)_4$ components in the time series, we will use the SARIMA model to model the original log-transformed time series rather than modeling the detrended and deseasonalized residuals, thus, combining the steps.

```
ggtsdisplay(df_ds, main = "Detrended, de-seaonalized residuals")
```



Reviewing the residuals of the detrended and deseasonalized data, we can confirm the removal of the trend and seasonality. On the PACF plot, there appears to strong serial quarterly correlation as it oscillates towards zero. The ACF plot has high serial correlation at the 4th lag. The strong ACF at lag 4 and cycling towards 0 in PACF suggest there is a seasonal MA(1) [SMA1] component.

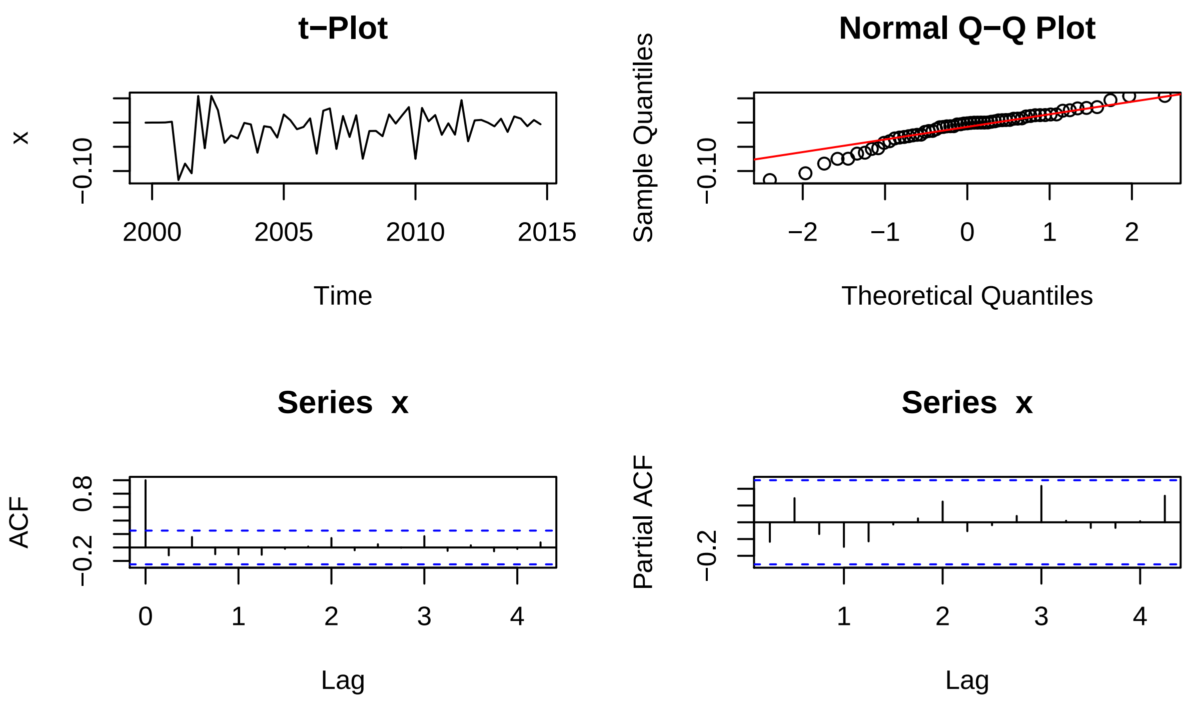
Based on our exploration, we know our model will have a seasonal period $s = 4$, that we will need differencing of $d = 1, D = 1$. Our intial model will be $SARIMA(0, 1, 0)(0, 1, 1)_4$.

```
m = arima(dfts_log, order = c(0, 1, 0), seasonal = list(order = c(0, 1, 1), period = 4))
xtable(arimatable(m, dfts_log))
```

	beta	SE	Sigma2	AIC	BIC	LogLik	ME	RMSE	MAE
sma1	-0.52	0.10	0.00	-201.54	-197.49	102.77	-0.01	0.04	0.03

The SMA1 β at -0.5167 with $SE = 0.0975$ suggest stationarity at the 95% confidence interval as it does not cross 1. It also does not cross 0 suggesting significance at lag 1.

```
res <- ts_resid(m$residuals)
```



```
res
```

[1] 0.003309518

The t-plot of the residuals appears to be white noise with heteroskedasticity. We should square the residuals and check the acf and pacf plots to possibly model the variance with GARCH/ARCH model. It is beyond the scope of the lab. The t-plot shows no trend or seasonality. The residuals appear to be a stationary process from the augmented Dickey Fuller and Phillips-Perron Test. The Shapiro-Wilks tests, $p < 0.05$, indicates non-normality of residuals. While this may cause inferences on the model to be invalid, we believe that the residual population distribution is normal based on Central Limit Theorem. Thus, the rejection of the H_0 in Shapiro-Wilks test may be potentially overlooked. Finally, the Ljung-box test with p-Value of 0.35 and visual inspection indicates uncorrelated residual, a property of white noise. From visual inspection and various test, we believe the residuals follow a Gaussian white noise process based on stationarity, normality and non-correlation.

We will examine other $SARIMA(p,1,q)(P,1,Q)_4$ up to $p = q = P = Q = 2$ to aid in choosing our final model.

```
results = run_arima_loop(dfts_log)
xtable(head(results[order(results$AIC, results$BIC), ], 5))
```

	p	d	q	P	D	Q	AIC	BIC	Log_Likelihood
3	0.00	1.00	0.00	0.00	1.00	2.00	-207.22	-201.14	106.61
7	0.00	1.00	0.00	2.00	1.00	0.00	-206.50	-200.42	106.25
4	0.00	1.00	0.00	1.00	1.00	0.00	-206.22	-202.17	105.11
30	1.00	1.00	0.00	0.00	1.00	2.00	-205.53	-197.42	106.76
5	0.00	1.00	0.00	1.00	1.00	1.00	-205.50	-199.42	105.75

Auto.arima() selected $SARIMA(0,1,0)(2,1,0)_4$.

```r
xtable(arimatable(auto.arima(dfts_log), dfts_log))
```

	beta	SE	Sigma2	AIC	BIC	LogLik	ME	RMSE	MAE
sar1	-0.80	0.15	0.00	-206.50	-200.42	106.25	-0.01	0.03	0.02
sar2	-0.25	0.16	0.00	-206.50	-200.42	106.25	-0.01	0.03	0.02

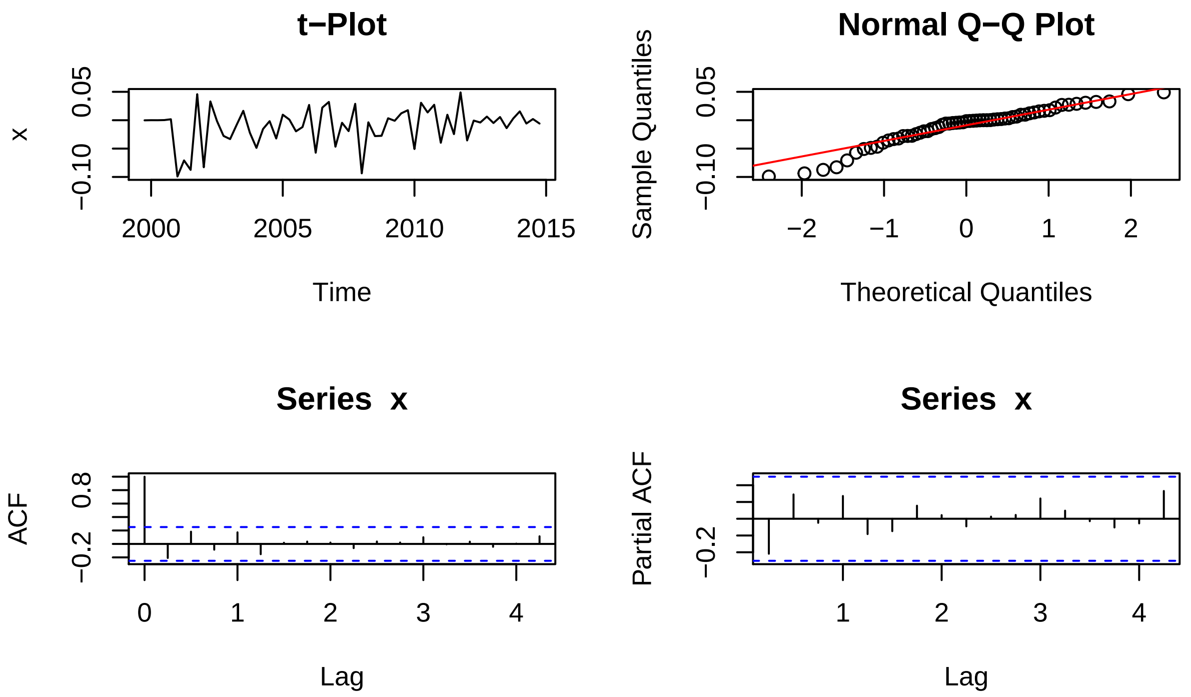
From the manual iterations and auto.arima(), $SARIMA(0,1,0)(0,1,2)_4$ and $SARIMA(0,1,0)(2,1,0)_4$ are chosen as the candidate models as they have the lowest AICs and BICs.

In the $SARIMA(0,1,0)(2,1,0)_4$ below, the βs do not include zero up to the 95% confidence interval and the residual appear to be stationary and white noise. The Ljung-Box p-Value is .13 and the heterskedasticity of the residuals is not evident. Though normality is rejected based on Shapiro-Wilks test, Central Limit Theorem is invoked. Based on visual inspections and statistical tests, we believe the residuals follow a Gaussian white noise process allowing for testing and inferences.

```r
m.010210 = arima(dfts_log, order = c(0, 1, 0), seasonal = list(order = c(2, 1, 0),
    period = 4))
xtable(arimatable(m.010210, dfts_log))
```

	beta	SE	Sigma2	AIC	BIC	LogLik	ME	RMSE	MAE
sar1	-0.80	0.15	0.00	-206.50	-200.42	106.25	-0.01	0.03	0.02
sar2	-0.25	0.16	0.00	-206.50	-200.42	106.25	-0.01	0.03	0.02

```r
ts_resid(m.010210$residuals)
```

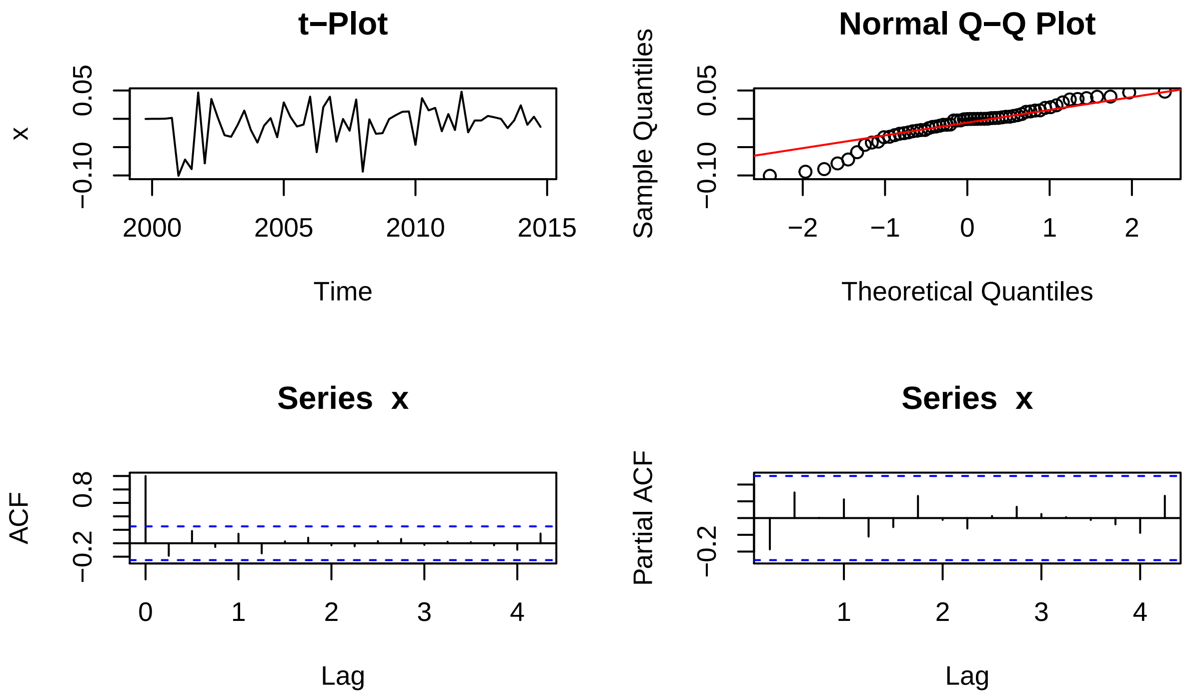


In the $SARIMA(0,1,0)(0,1,2)_4$ below, βs are statistically significant and the residuals do appear to be white noise with stationarity and no autocorrelation. The Ljung-Box p-Value is higher than the $SARIMA(0,1,0)(0,1,1)_4$ model. Note that the heteroskedasticity in residuals no longer appear. Here, the Shaprio-Wilks test also gives a significant p-value, $p < 0.05$, indicating the distribution is non-normal, but once again, Central Limit Theorem is invoked.

```r
m.010012 = arima(dfts_log, order = c(0, 1, 0), seasonal = list(order = c(0, 1, 2),
    period = 4))
xtable(arimatable(m.010012, dfts_log))
```

	beta	SE	Sigma2	AIC	BIC	LogLik	ME	RMSE	MAE
sma1	-0.77	0.16	0.00	-207.22	-201.14	106.61	-0.01	0.03	0.02
sma2	0.40	0.13	0.00	-207.22	-201.14	106.61	-0.01	0.03	0.02

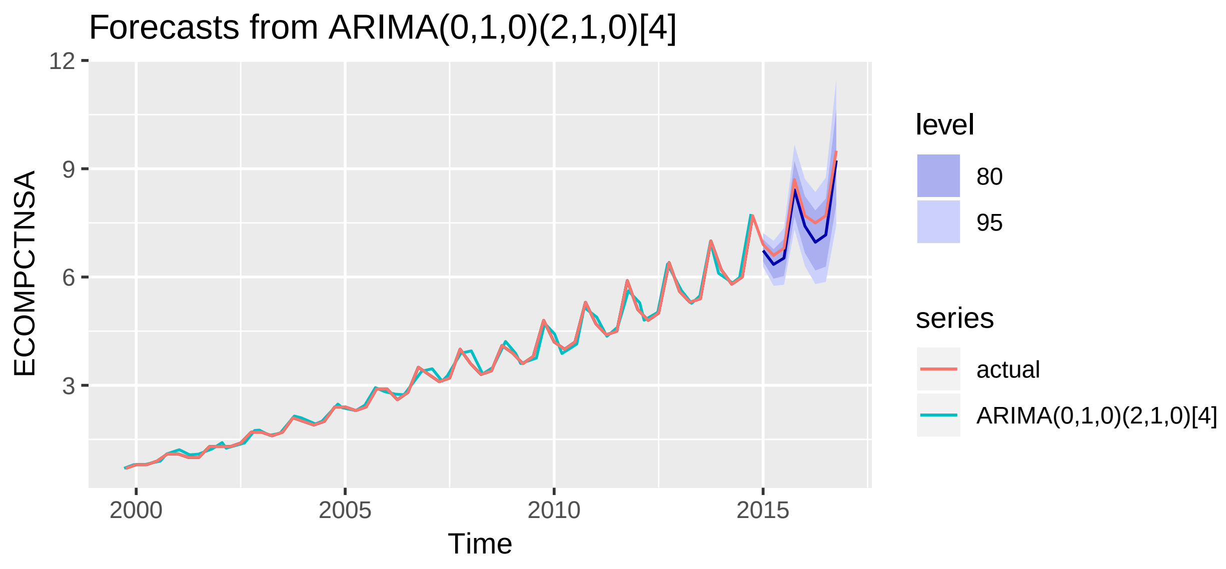
```r
ts_resid(m.010012$residuals)
```



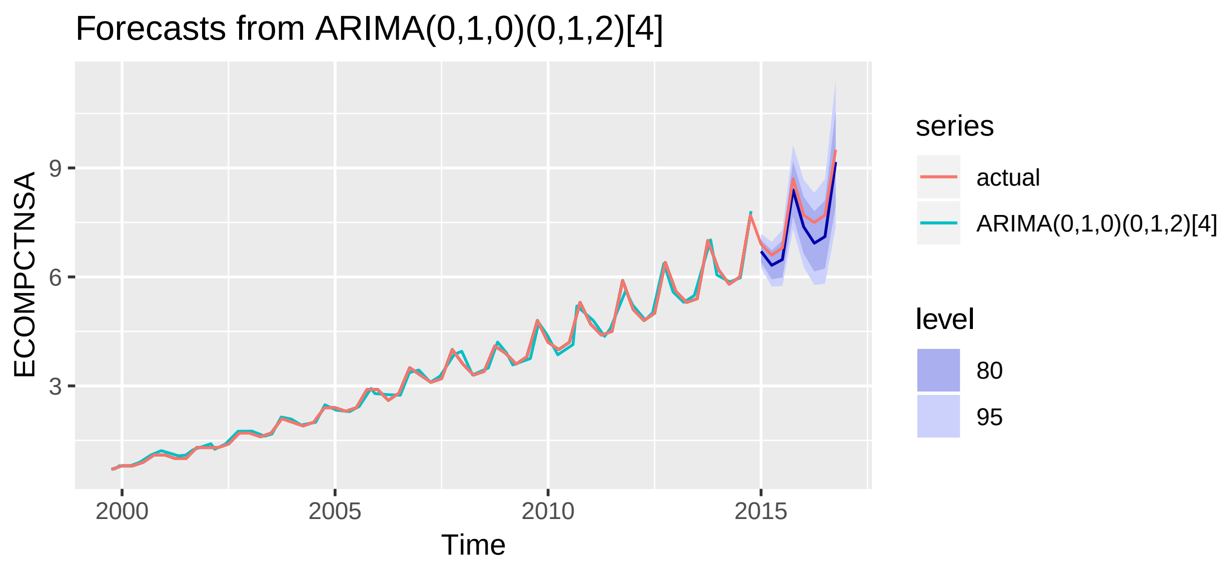
We first note that these model are very similar. We note the roots of the $SARIMA(0,1,0)(0,1,2)_4$ are 1.58, 1.58 and the roots of $SARIMA(0,1,0)(2,1,0)_4$ are 2, 2. The $SARIMA(0,1,0)(0,1,2)_4$ is invertible to $SARIMA(0,1,0)(2,1,0)_4$. Both models are quite similar.

The 2 models are chosen as potential candidates. We will examine both in-sample and out-of-sample fits to chose the final model.

```r
actual = ts(df$ECOMPCTNSA, start = c(1999, 4), freq = 4)
forecast_sar = forecast_exp_func(forecast(m.010210))
forecast_sma = forecast_exp_func(forecast(m.010012))
autoplot(forecast_sar) + autolayer(exp(fitted(m.010210)), series = "ARIMA(0,1,0)(2,1,0)[4]",
    position = position_jitter()) + ylab("ECOMPCTNSA") + autolayer(actual)
```



```
autoplot(forecast_sma) + autolayer(exp(fitted(m.010012)), series = "ARIMA(0,1,0)(0,1,2)[4]",
    position = position_jitter()) + autolayer(actual) + ylab("ECOMPCTNSA")
```



The in-sample fits for both models are extremely close to the historical fit. The predictions for both models are extremely similar. We will select the models based on accuracy of the time series. The time series is logged to avoid overweighting the larger values on the time series due to the trend and seasonality.

```
pred_test = window(log(actual), start = c(2015, 1))
xtable(accuracy(forecast(m.010210), pred_test), caption = "$ARIMA(0,1,0)(2,1,0)_4$")
```

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	-0.01	0.03	0.02	-Inf	Inf	0.16	-0.21	NA
Test set	0.04	0.05	0.04	2.17	2.17	0.30	0.18	0.37

```
xtable(accuracy(forecast(m.010012), pred_test), caption = "$ARIMA(0,1,0)(0,1,2)_4$")
```

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	-0.01	0.03	0.02	-Inf	Inf	0.16	-0.19	NA
Test set	0.05	0.05	0.05	2.44	2.44	0.34	0.20	0.41

Every accuracy measure tested showed a lower error with $SARIMA(0,1,0)(2,1,0)_4$ model. The final model chosen is

$$SARIMA(0,1,0)(2,1,0)_4$$

$$(1-0.80B-0.25B^2)_4(1-B)_4(1-B)x_t=\epsilon_t$$

$$(1-0.80B-0.25B^2)_4(x_t-x_{t-1}-x_{t-4}+x_{t-5})=\epsilon_t$$

$$\begin{aligned}x_t-x_{t-1}-x_{t-4}+x_{t-5}-0.80x_{t-4}+0.80x_{t-5}+0.80x_{t-8}-0.80x_{t-9}\\-0.25x_{t-8}+0.25x_{t-9}+0.25x_{t-12}-0.25x_{t-13}=\epsilon_t\end{aligned}$$

$$\begin{aligned}x_t=x_{t-1}+x_{t-4}-x_{t-5}+0.80x_{t-4}-0.80x_{t-5}-0.80x_{t-8}+0.80x_{t-9}\\+0.25x_{t-8}-0.25x_{t-9}-0.25x_{t-12}i+0.25x_{t-13}+\epsilon_t\end{aligned}$$

```
df_full_log = ts(log(df$ECOMPCTNSA), start = c(1999, 4), freq = 4)
m = m.010210
xtable(arimatable(m, df_full_log))
```

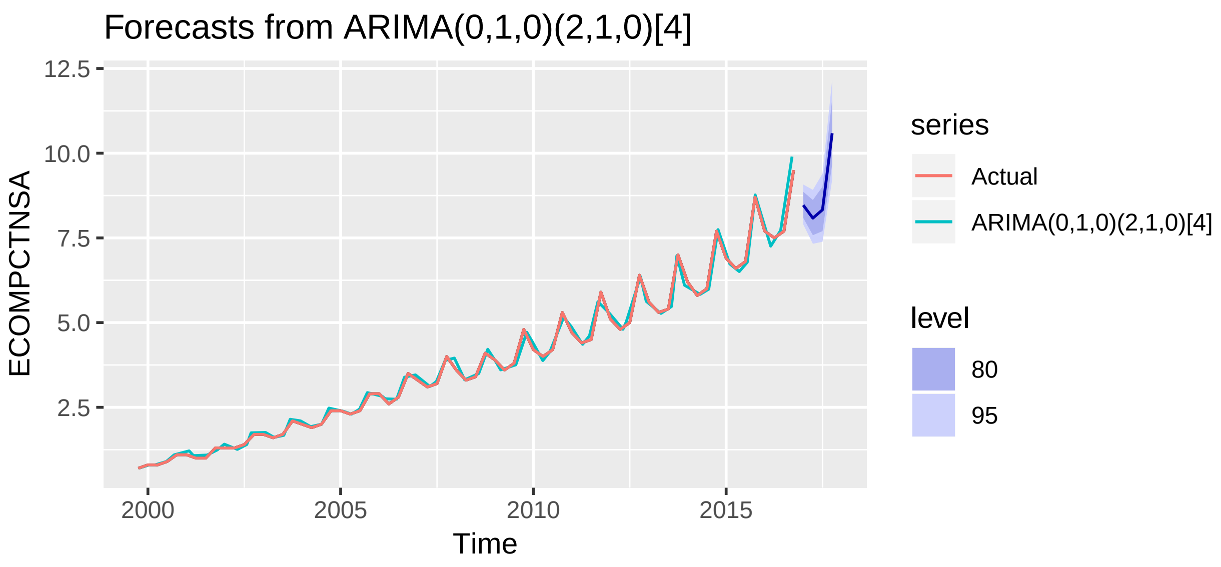
	beta	SE	Sigma2	AIC	BIC	LogLik	ME	RMSE	MAE
sar1	-0.80	0.15	0.00	-206.50	-200.42	106.25	-0.01	0.03	0.02
sar2	-0.25	0.16	0.00	-206.50	-200.42	106.25	-0.01	0.03	0.02

The forecast for 2017 using $SARIMA(0,1,0)(2,1,0)_4$ is

```
m = Arima(df_full_log, model = m)  # insert the new series into the m.010210 arima model
forecast_sar = forecast_exp_func(forecast(m, h = 4))
xtable(data.frame(forecast_sar))
```

	Point.Forecast	Lo.80	Hi.80	Lo.95	Hi.95
2017 Q1	8.47	8.09	8.86	7.90	9.08
2017 Q2	8.08	7.58	8.62	7.33	8.92
2017 Q3	8.33	7.70	9.01	7.39	9.40
2017 Q4	10.59	9.67	11.60	9.22	12.17

```
autoplot(forecast_sar, "Model") + autolayer(exp(fitted(m)), series = "ARIMA(0,1,0)(2,1,0)[4]",
    position = position_jitter()) + ylab("ECOMPCTNSA") + autolayer(exp(df_full_log),
    series = "Actual")
```



The forecast closely follows what we would expect of the trend going forward in 2017.

Question 2: Learning how to use the xts library

Only Task 5 is left for brevity.

Task 5:

1. Read AMAZ.csv and UMCSENT.csv into R as R DataFrames

```
library(xts)
amaz <- read.csv("AMAZ.csv")
umcsent <- read.csv("UMCSENT.csv")
xtable(head(amaz))
```

	Index	AMAZ.Open	AMAZ.High	AMAZ.Low	AMAZ.Close	AMAZ.Volume
1	2007-01-03	20.00	20.00	16.00	16.00	650
2	2007-01-04	20.00	20.00	20.00	20.00	67
3	2007-01-08	19.20	22.00	19.20	22.00	1801
4	2007-01-09	22.00	22.00	20.80	20.80	356
5	2007-01-10	20.80	20.80	20.80	20.80	438
6	2007-01-11	20.80	21.60	20.80	21.60	2318

```
xtable(tail(amaz))
```

	Index	AMAZ.Open	AMAZ.High	AMAZ.Low	AMAZ.Close	AMAZ.Volume
1174	2013-01-04	0.88	0.88	0.80	0.80	3850
1175	2013-01-07	0.80	1.00	0.80	1.00	2715
1176	2013-01-08	0.80	0.80	0.68	0.68	4668
1177	2013-01-09	0.88	0.88	0.80	0.80	2750
1178	2013-01-11	0.80	0.80	0.80	0.80	3000
1179	2013-01-15	0.68	0.68	0.68	0.68	1000

```
length(amaz$Index)
```

```
[1] 1179
```

```
xtable(head(umcsent))
```

	Index	UMCSENT
1	1978-01-01	83.70
2	1978-02-01	84.30
3	1978-03-01	78.80
4	1978-04-01	81.60
5	1978-05-01	82.90
6	1978-06-01	80.00

```
xtable(tail(umcsent))
```

	Index	UMCSENT
472	2017-04-01	97.00
473	2017-05-01	97.10
474	2017-06-01	95.10
475	2017-07-01	93.40
476	2017-08-01	96.80
477	2017-09-01	95.10

```
length(umcsent$Index)
```

```
[1] 477
```

2. Convert them to xts objects

```
amaz.xts <- as.xts(amaz[, 2:6], order.by = as.Date(amaz$Index, format = "%Y-%m-%d"))
xtable(head(amaz.xts))
```

	AMAZ.Open	AMAZ.High	AMAZ.Low	AMAZ.Close	AMAZ.Volume
1	20.00	20.00	16.00	16.00	650.00
2	20.00	20.00	20.00	20.00	67.00
3	19.20	22.00	19.20	22.00	1801.00
4	22.00	22.00	20.80	20.80	356.00
5	20.80	20.80	20.80	20.80	438.00
6	20.80	21.60	20.80	21.60	2318.00

```
xtable(tail(amaz.xts))
```

	AMAZ.Open	AMAZ.High	AMAZ.Low	AMAZ.Close	AMAZ.Volume
1	0.88	0.88	0.80	0.80	3850.00
2	0.80	1.00	0.80	1.00	2715.00
3	0.80	0.80	0.68	0.68	4668.00
4	0.88	0.88	0.80	0.80	2750.00
5	0.80	0.80	0.80	0.80	3000.00
6	0.68	0.68	0.68	0.68	1000.00

```
umcsent.xts <- as.xts(umcsent[, 2], order.by = as.Date(umcsent$Index, format = "%Y-%d-%m"))
colnames(umcsent.xts) = c("umcsent")
xtable(head(umcsent.xts))
```

	Value
1	83.70
2	84.30
3	78.80
4	81.60
5	82.90
6	80.00

```
xtable(tail(umcsent.xts))
```

	Value
1	97.00
2	97.10
3	95.10
4	93.40
5	96.80
6	95.10

It is important to note here that the `amaz.xts` has a shorter duration and span than the `umcscent.xts` series. However, the `amaz.xts` series has more observations than `umcsent.xts`.

3. Merge the two set of series together, perserving all of the observations in both set of series

```
merged <- merge(amaz.xts, umcsent.xts, join = "outer")
xtable(head(merged))
```

	AMAZ.Open	AMAZ.High	AMAZ.Low	AMAZ.Close	AMAZ.Volume	umcsent
1	NA	NA	NA	NA	NA	83.70
2	NA	NA	NA	NA	NA	84.30
3	NA	NA	NA	NA	NA	78.80
4	NA	NA	NA	NA	NA	81.60
5	NA	NA	NA	NA	NA	82.90
6	NA	NA	NA	NA	NA	80.00

```
xtable(tail(merged))
```

	AMAZ.Open	AMAZ.High	AMAZ.Low	AMAZ.Close	AMAZ.Volume	umcsent
1	NA	NA	NA	NA	NA	97.00
2	NA	NA	NA	NA	NA	97.10
3	NA	NA	NA	NA	NA	95.10
4	NA	NA	NA	NA	NA	93.40
5	NA	NA	NA	NA	NA	96.80
6	NA	NA	NA	NA	NA	95.10

```
dim(merged)
```

```
[1] 1610 6
```

`a. fill all of the missing values of the UMCSENT series with -9999`

```
umcsent01 = merged
xtable(head(umcsent01))
```

	AMAZ.Open	AMAZ.High	AMAZ.Low	AMAZ.Close	AMAZ.Volume	umcsent
1	NA	NA	NA	NA	NA	83.70
2	NA	NA	NA	NA	NA	84.30
3	NA	NA	NA	NA	NA	78.80
4	NA	NA	NA	NA	NA	81.60
5	NA	NA	NA	NA	NA	82.90
6	NA	NA	NA	NA	NA	80.00

```
umcsent01 = na.fill(umcsent01, -9999)
xtable(head(umcsent01))
```

	AMAZ.Open	AMAZ.High	AMAZ.Low	AMAZ.Close	AMAZ.Volume	umcsent
1	-9999.00	-9999.00	-9999.00	-9999.00	-9999.00	83.70
2	-9999.00	-9999.00	-9999.00	-9999.00	-9999.00	84.30
3	-9999.00	-9999.00	-9999.00	-9999.00	-9999.00	78.80
4	-9999.00	-9999.00	-9999.00	-9999.00	-9999.00	81.60
5	-9999.00	-9999.00	-9999.00	-9999.00	-9999.00	82.90
6	-9999.00	-9999.00	-9999.00	-9999.00	-9999.00	80.00

```
b. then create a new series, named UMCSENT02, from the original UMCSENT series replace all of
```

```
umcsent02 <- umcsent01
xtable(head(umcsent02))
```

	AMAZ.Open	AMAZ.High	AMAZ.Low	AMAZ.Close	AMAZ.Volume	umcsent
1	-9999.00	-9999.00	-9999.00	-9999.00	-9999.00	83.70
2	-9999.00	-9999.00	-9999.00	-9999.00	-9999.00	84.30
3	-9999.00	-9999.00	-9999.00	-9999.00	-9999.00	78.80
4	-9999.00	-9999.00	-9999.00	-9999.00	-9999.00	81.60
5	-9999.00	-9999.00	-9999.00	-9999.00	-9999.00	82.90
6	-9999.00	-9999.00	-9999.00	-9999.00	-9999.00	80.00

```
umcsent02[umcsent02 <= -9999] <- NA
xtable(head(umcsent02))
```

	AMAZ.Open	AMAZ.High	AMAZ.Low	AMAZ.Close	AMAZ.Volume	umcsent
1	NA	NA	NA	NA	NA	83.70
2	NA	NA	NA	NA	NA	84.30
3	NA	NA	NA	NA	NA	78.80
4	NA	NA	NA	NA	NA	81.60
5	NA	NA	NA	NA	NA	82.90
6	NA	NA	NA	NA	NA	80.00

```
c. then create a new series, named UMCSENT03, and replace the NAs with the last observation
```

```
umcsent03 = umcsent02
xtable(head(umcsent03))
```

	AMAZ.Open	AMAZ.High	AMAZ.Low	AMAZ.Close	AMAZ.Volume	umcsent
1	NA	NA	NA	NA	NA	83.70
2	NA	NA	NA	NA	NA	84.30
3	NA	NA	NA	NA	NA	78.80
4	NA	NA	NA	NA	NA	81.60
5	NA	NA	NA	NA	NA	82.90
6	NA	NA	NA	NA	NA	80.00

```
umcsent03 <- na.locf(umcsent02, na.rm = TRUE, fromLast = TRUE)
xtable(head(umcsent03))
```

	AMAZ.Open	AMAZ.High	AMAZ.Low	AMAZ.Close	AMAZ.Volume	umcsent
1	20.00	20.00	16.00	16.00	650.00	83.70
2	20.00	20.00	16.00	16.00	650.00	84.30
3	20.00	20.00	16.00	16.00	650.00	78.80
4	20.00	20.00	16.00	16.00	650.00	81.60
5	20.00	20.00	16.00	16.00	650.00	82.90
6	20.00	20.00	16.00	16.00	650.00	80.00

```
d. then create a new series, named UMCSENT04, and replace the NAs using linear interpolation.
```

```
umcsent04 = umcsent02
xtable(head(umcsent04))
```

	AMAZ.Open	AMAZ.High	AMAZ.Low	AMAZ.Close	AMAZ.Volume	umcsent
1	NA	NA	NA	NA	NA	83.70
2	NA	NA	NA	NA	NA	84.30
3	NA	NA	NA	NA	NA	78.80
4	NA	NA	NA	NA	NA	81.60
5	NA	NA	NA	NA	NA	82.90
6	NA	NA	NA	NA	NA	80.00

```
xtable(head(umcsent04["2007-01", ], 15))
```

	AMAZ.Open	AMAZ.High	AMAZ.Low	AMAZ.Close	AMAZ.Volume	umcsent
1	NA	NA	NA	NA	NA	96.90
2	NA	NA	NA	NA	NA	91.30
3	20.00	20.00	16.00	16.00	650.00	88.40
4	20.00	20.00	20.00	20.00	67.00	87.10
5	NA	NA	NA	NA	NA	88.30
6	NA	NA	NA	NA	NA	85.30
7	NA	NA	NA	NA	NA	90.40
8	19.20	22.00	19.20	22.00	1801.00	83.40
9	22.00	22.00	20.80	20.80	356.00	83.40
10	20.80	20.80	20.80	20.80	438.00	80.90
11	20.80	21.60	20.80	21.60	2318.00	76.10
12	22.00	22.00	22.00	22.00	306.00	75.50
13	21.60	21.60	21.20	21.20	925.00	NA
14	22.00	22.00	21.60	21.60	2138.00	NA
15	23.20	23.20	22.80	22.80	527.00	NA

```r
umcsent04 = na.approx(umcsent04, maxgap = 10000)
```

Note amazon has N/As in 1/1/17 and 1/2/17 because there is no data before 1/1/03 so there is nothing to interpolate.

```r
xtable(head(umcsent04["2007-01", ], 15))
```

	AMAZ.Open	AMAZ.High	AMAZ.Low	AMAZ.Close	AMAZ.Volume	umcsent
1	NA	NA	NA	NA	NA	96.90
2	NA	NA	NA	NA	NA	91.30
3	20.00	20.00	16.00	16.00	650.00	88.40
4	20.00	20.00	20.00	20.00	67.00	87.10
5	19.80	20.50	19.80	20.50	500.50	88.30
6	19.60	21.00	19.60	21.00	934.00	85.30
7	19.40	21.50	19.40	21.50	1367.50	90.40
8	19.20	22.00	19.20	22.00	1801.00	83.40
9	22.00	22.00	20.80	20.80	356.00	83.40
10	20.80	20.80	20.80	20.80	438.00	80.90
11	20.80	21.60	20.80	21.60	2318.00	76.10
12	22.00	22.00	22.00	22.00	306.00	75.50
13	21.60	21.60	21.20	21.20	925.00	75.53
14	22.00	22.00	21.60	21.60	2138.00	75.54
15	23.20	23.20	22.80	22.80	527.00	75.58

```
e. Print out some observations to ensure that your merge as well as the missing value imputatio
```

Observations to check the merge and imputation are printed in the above sections.

4. Calculate the daily return of the Amazon closing price (AMAZ.close), where daily return is defined as $(x(t) - x(t-1))/x(t-1)$. Plot the daily return series.

```r
amaz.xts.close = amaz.xts$AMAZ.Close
xtable(head(amaz.xts.close))
```

	Value
1	16.00
2	20.00
3	22.00
4	20.80
5	20.80
6	21.60

```r
xtable(head(diff(amaz.xts.close, lag = 1, differences = 1, log = FALSE, na.pad = FALSE)))
```

	Value
1	4.00
2	2.00
3	-1.20
4	0.00
5	0.80
6	0.40

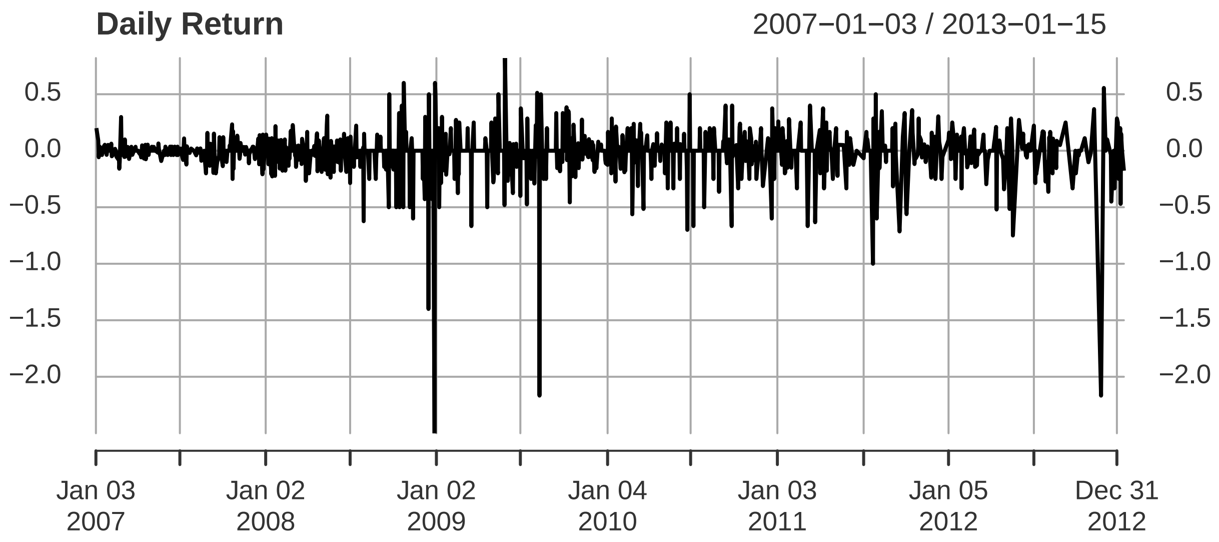
```
xtable(head(diff(amaz.xts.close, lag = 1, differences = 1, log = FALSE, na.pad = FALSE)/amaz.x
```

	Value
1	0.20
2	0.09
3	-0.06
4	0.00
5	0.04
6	0.02

```
tmp = cbind(amaz.xts.close, diff(amaz.xts.close, lag = 1, differences = 1, log = FALSE,
    na.pad = TRUE), diff(amaz.xts.close, lag = 1, differences = 1, log = FALSE, na.pad = TRUE)/
colnames(tmp) = c("Close", "Chg", "PctChg")
xtable(head(tmp))
```

	Close	Chg	PctChg
1	16.00	NA	NA
2	20.00	4.00	0.20
3	22.00	2.00	0.09
4	20.80	-1.20	-0.06
5	20.80	0.00	0.00
6	21.60	0.80	0.04

```
plot(tmp$PctChg, main = "Daily Return")
```



5. Create a 20-day and a 50-day rolling mean series from the AMAZ.close series.

The numbers below look odd but it is correct. AMAZ became a penny stock. Note, AMAZ is not Amazon.

```
xtable(tail(cbind(amaz.xts.close, rollapply(amaz.xts.close, 20, FUN = mean, na.rm = TRUE)),
    15))
```

	AMAZ.Close	AMAZ.Close.1
1	1.08	1.09
2	1.20	1.09
3	1.16	1.09
4	0.80	1.08
5	0.80	1.06
6	0.60	1.03
7	0.84	0.99
8	1.12	0.99
9	1.00	0.98
10	0.80	0.97
11	1.00	0.97
12	0.68	0.95
13	0.80	0.95
14	0.80	0.94
15	0.68	0.92

```
xtable(tail(cbind(amaz.xts.close, rollapply(amaz.xts.close, 50, FUN = mean, na.rm = TRUE)),
    15))
```

	AMAZ.Close	AMAZ.Close.1
1	1.08	1.21
2	1.20	1.21
3	1.16	1.21
4	0.80	1.21
5	0.80	1.20
6	0.60	1.19
7	0.84	1.18
8	1.12	1.17
9	1.00	1.16
10	0.80	1.15
11	1.00	1.14
12	0.68	1.13
13	0.80	1.12
14	0.80	1.11
15	0.68	1.09