Lab 3

Nishant Velagapudi, Bryan Moore, Morris Burkhardt

March 19, 2018

Question 1: E-Commerce Retail Sales as a Percentage of Total Sales

Our time series describes the percentage of sales attributed to e-commerce. Modelling this series will require three major steps. First, we will explore and visualize the series to understand potential transformations as well as AR and MA orders. Second, we will find the SARIMA specification with the optimal AIC value. Finally, we will compare various model specifications (informed either through our EDA or our AutoArima results) on the basis of model performance in a hold-out data set. Data up until 2015 will be used for training, with 2015 and 2016 data being used for model selection. We generate a forecast for 2017 data using the best-performing specification trained over all data available.

Exploration and Visualization

Although it has been dictated that we are to create a SARIMA model, we will first explore the dataset to check if this is suitable. First, we load the data, convert it to a time series, split the time series into training and testing data, display the first few rows, check for missing values and display the summary.

```
ecom = read.csv(file = "ECOMPCTNSA.csv")
ts_ecom = ts(ecom$ECOMPCTNSA, start = c(1999, 4), frequency = 4)
ts_ecom_train = ts(ts_ecom[time(ts_ecom) < 2015], start = c(1999, 4), freq = 4)
ts_ecom_test = ts(ts_ecom[time(ts_ecom) >= 2015], start = c(2015, 1), freq = 4)
head(ts_ecom, 13); sum(is.na(ts_ecom_train)); summary(ts_ecom_train)
```

```
##      Qtr1 Qtr2 Qtr3 Qtr4
## 1999                 0.7
## 2000  0.8  0.8  0.9  1.1
## 2001  1.1  1.0  1.0  1.3
## 2002  1.3  1.3  1.4  1.7
```

```
## [1] 0
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.700   1.700   3.300   3.331   4.700   7.700
```

We have no missing values in this series. The series ranges from 0.7 to 7.7 with a mean of 3.3 and a median of 3.3. Next, we plot the training time series.

```
par(mfrow=c(1,1), mai = c(.4, 0.4, 0.4, 0.4))
plot(ts_ecom_train, xlab = 'Date', ylab = 'E-Commerce Retail Sales', main = 'E-commerce Sales: % of Total')
lines(ma(ts_ecom_train, order = 4, centre = T), col = 'green')
```
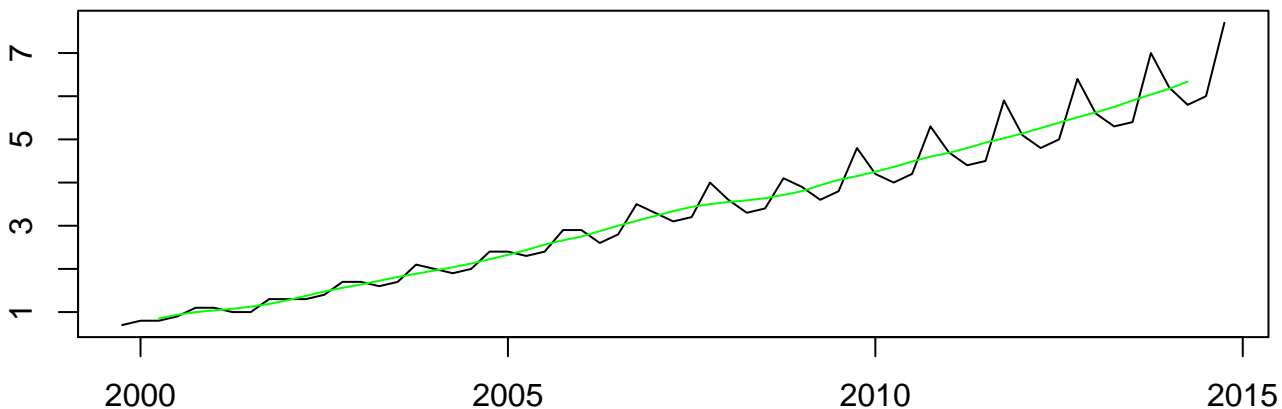

The plot shows both a deterministic trend as well as strong seasonality. A MA(4) model of this series smoothens out the variance: reinforcing the observation that we have annual seasonality in this series. We next look at autocorrelation and partial autocorrelation functions of the training series.

```
par(mfrow=c(1,2))
acf(ts_ecom_train)
pacf(ts_ecom_train)
```
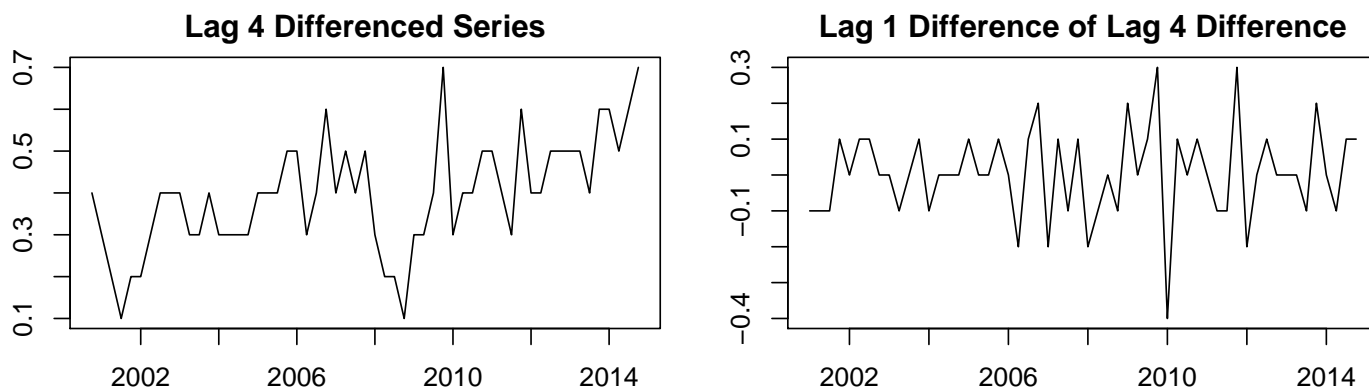
Series ts_ecom_train
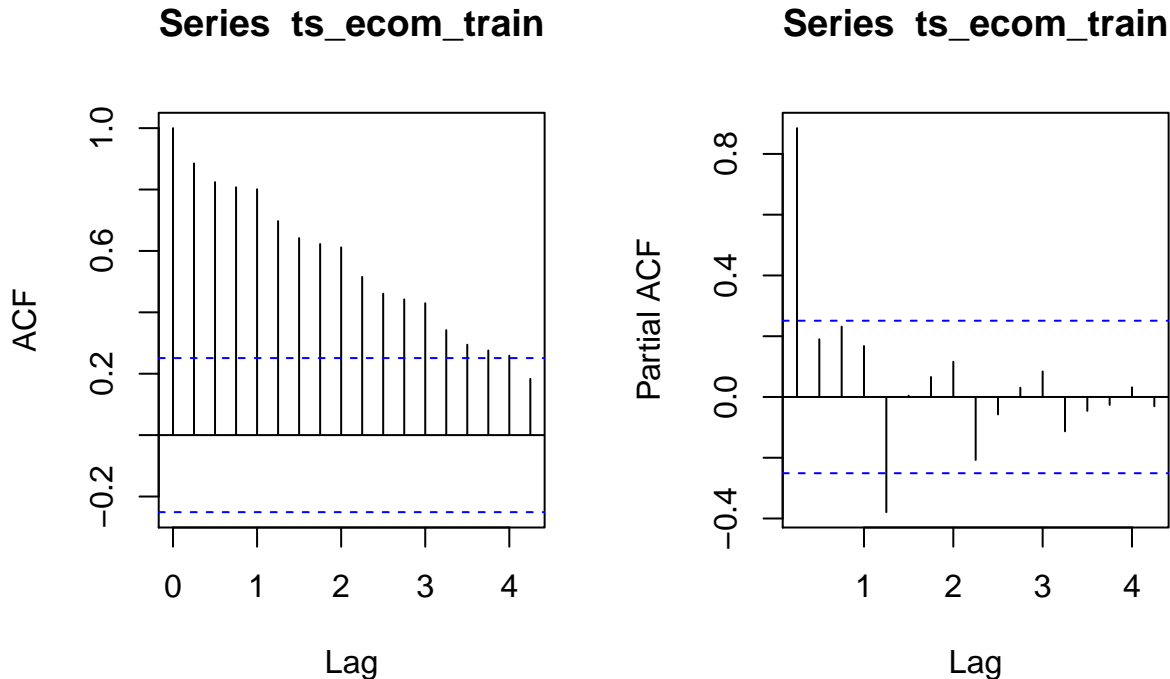
We see that the autocorrelations are significant for a large number of lags (16 quarters), which is further evidence of non-stationarity. The gradual decay in the acf without any spikes at seasonal intervals tells us that we will likely need a non-seasonal AR term p but may not need a seasonal AR term P.

Imposing stationarity will require differencing the series. We take first and fourth order differences as well as the first order difference of the fourth order differences (indicative of first order differencing after an annual seasonal differencing). For the time series that look stationary in the differenced plots, we conduct Augmented Dickey Fuller (ADF) tests. The associated null hypothesis is that the series contains unit roots (indicates non-stationary stochastic trend).

```
par(mfrow=c(1,2), mai = c(.4, 0.4, 0.4, 0.4))

plot(diff(ts_ecom_train, lag = 4),xlab=NULL,ylab=NULL,main = "Lag 4 Differenced Series")
plot(diff(diff(ts_ecom_train, lag = 4),lag=1),xlab=NULL,ylab=NULL,
     main = "Lag 1 Difference of Lag 4 Difference")
```



```
adf.test(diff(diff(ts_ecom_train, lag = 4),lag=1))$p.value
```

```
## [1] 0.01
```

We can see that our difference of differences approach leads to at least stochastic stationarity (corroborated by the ADF test result). As this series increases in magnitude, the associated seasonal variance also increases, and none of the differences taken above are homoskedastic (assessed visually).

We thus examine the effect of taking the log transform of this series prior to differencing.

```
par(mfrow=c(1,2), mai = c(.4, 0.4, 0.4, 0.4))

log_ts_ecom_train = log(ts_ecom_train); log_ts_ecom_test = log(ts_ecom_test)
plot(diff(log_ts_ecom_train, lag = 1),xlab=NULL,ylab=NULL,main = "Lag 1 Differenced Log Series")
plot(diff(log_ts_ecom_train, lag = 4),xlab=NULL,ylab=NULL,main = "Lag 4 Differenced Log Series")
```

Lag 1 Differenced Log Series   **Lag 4 Differenced Log Series**
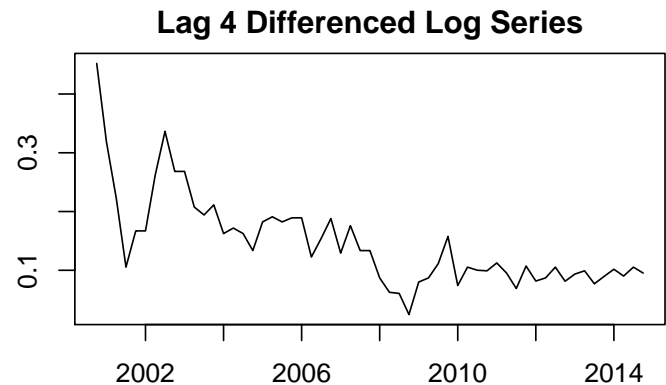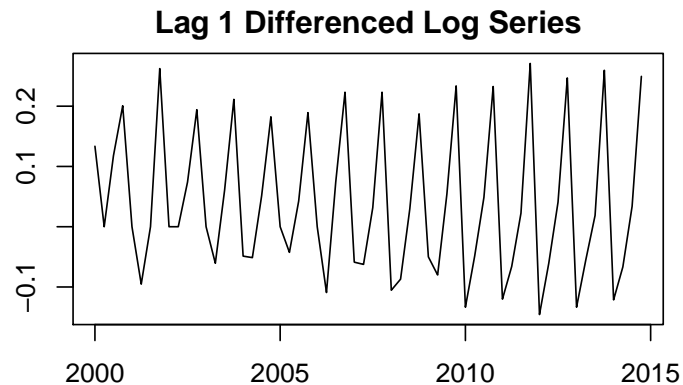

```
adf.test(diff(log_ts_ecom_train))$p.value
```

```
## [1] 0.01
```

We can see that the first order difference of the logarithm of the training series appears homoskedastic, and also achieves stationarity by the ADF test.

Modelling

First, we will specify two models purely informed by our exploratory data analysis. One model each for the log transformed and the raw (non-transformed) series.

The spike in the ACF at a lag of 1 quarter suggests a nonseasonal MA(1) (q=1) component and the spikes at intervals of 4 quarters of lag out to approximately lag 16 suggest a seasonal MA(4) (Q=4). Additionally, the spike at at a lag of 1 quarter in the PACF and the spikes at intervals of 4 quarters of lag tells us that a a nonseasonal AR(1) (p=1) component and a seasonal AR(1) (P=1) component are appropriate for our initial model. Therefore, we manually specify $ARIMA(1,1,1)(1,1,4)_4$ for this series.

In the log transformed case with no modelled seasonality, we observed a somewhat sinusoidal ACF with a PACF showing no significant spikes beyond lag 4 (not shown). We furthermore achieved steady variance through differencing with lag 1. Hence, we would manually specify an order of $ARIMA(0,1,4)$ for the log transformed series.

```
manual_model_raw <- Arima(ts_ecom_train, order=c(1,1,1), seasonal=c(1,1,4))
manual_model_log <- Arima(ts_ecom_train, order = c(0,1,4), seasonal=c(0,0,0), lambda=0)
```

Next, We will use the auto.arima function to identify the SARIMA parameters that minimize the corrected AIC. Again, we will explore modelling both, the log transformed time series (handled by the argument "lambda=0") and the non-transformed series.

```
auto_model_log <- auto.arima(ts_ecom_train, ic = 'aicc', lambda = 0)
auto_model_raw <- auto.arima(ts_ecom_train, ic = 'aicc')
```

For the log transformed series, the auto.arima algorithm returns a model of order (0,1,0)(2,1,0)[4] and for the non-transformed series it returns a model of order (0,1,1)(0,1,0)[4].

We will compare performance of each of these four specifications based on diagnostic tests, AIC, BIC, and in-sample RMSE before finalizing one model for our out-of-sample prediction.

Forecast for 2015 and 2016

We calculate and visualize forecasts for the years 2015 and 2016 (validation dataset) using each of the four specified models. Visuals of predictions versus actual values help us understand the forecasts.

```
forecast_log_auto = forecast(auto_model_log, h = 8, lambda=0)
forecast_auto = forecast(auto_model_raw, h = 8)
forecast_log_man = forecast(manual_model_raw, h = 8)
forecast_man = forecast(manual_model_log, h = 8, lambda=0)

par(mfrow=c(1,2), mai = c(.4, 0.4, 0.4, 0.4))
plot(forecast_log_auto, main="AutoArima Spec - Log Transformed"); lines(ts_ecom_test, col='green')
legend(2000, 11, legend=c('Predicted','Observed', 'historic'), col=c('blue','green', 'black'),
```

```
            lty=1, cex=0.8)
plot(forecast_auto, main="AutoArima Spec - Untransformed"); lines(ts_ecom_test, col='green')
```

AutoArima Spec – Log Transformed

AutoArima Spec – Untransformed
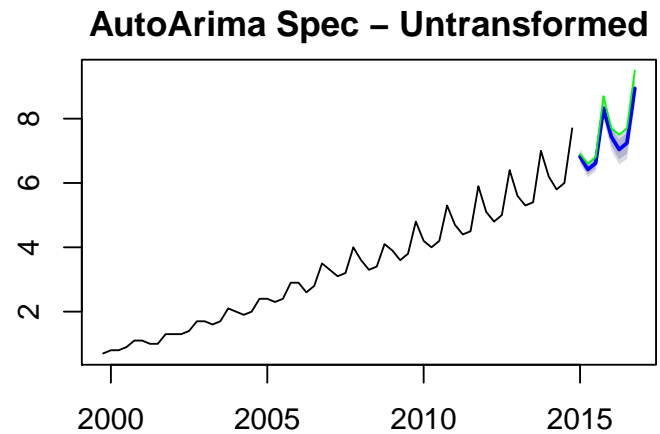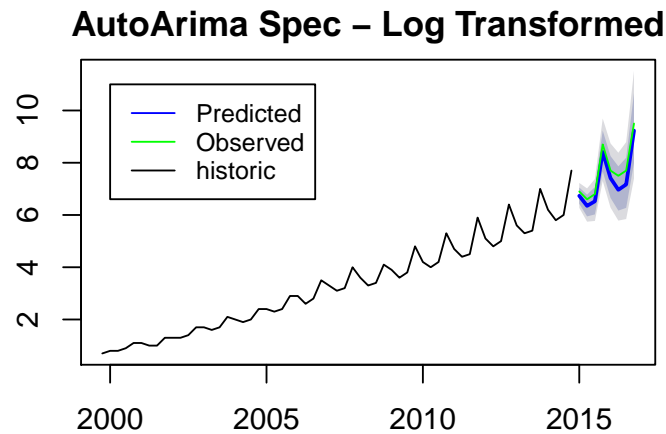

```
par(mfrow=c(1,2), mai = c(.4, 0.4, 0.4, 0.4))
plot(forecast_log_man, main="Manually Specified - Log Transformed"); lines(ts_ecom_test, col='green')
legend(2000, 9, legend=c('Predicted','Observed', 'historic'), col=c('blue','green', 'black'),
       lty=1, cex=0.8)
plot(forecast_man, main="Manually Specified - Untransformed"); lines(ts_ecom_test, col='green')
```

Manually Specified – Log Transformed

Manually Specified – Untransformed
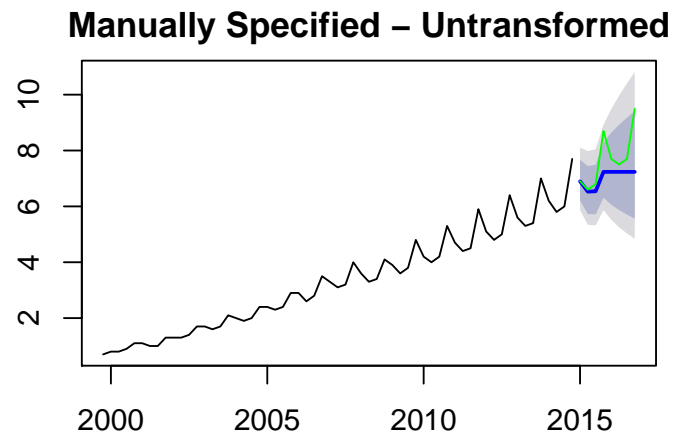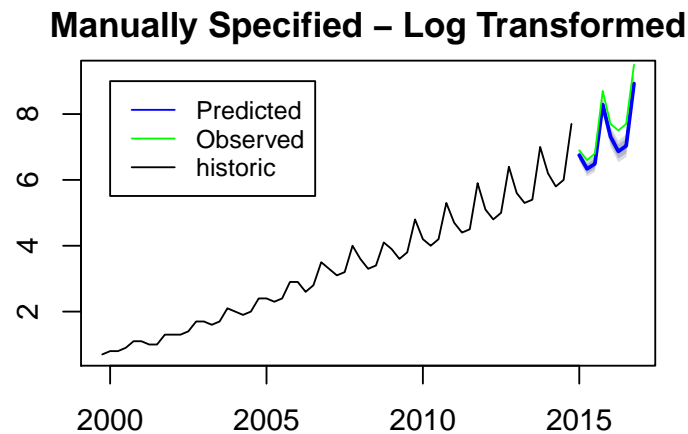
Residual diagnostics

We check each of these four models for validity: We use the Shapiro-Wilk and Box-Ljung tests. The former tests for normality of residuals within the training set, while the latter tests for autocorrelation within these same residuals. The null hypothesis of the Shapiro-Wilk test states that the residuals come from a normal distribution. The null hypothesis of the Box-Ljung test states that the autocorrelation is zero. We further compare these models on the basis of root mean square error (RMSE) of predictions in the validation holdout.

```
calculate_rmse = function(fcast, test){
  rmse = sqrt(mean((fcast - test)^2))
  return(rmse)
}
auto_model_log_sw <- shapiro.test(auto_model_log $resid)
auto_model_log_box <-Box.test(auto_model_log $resid, type = "Ljung-Box")
auto_model_raw_sw <-shapiro.test(auto_model_raw $resid)
auto_model_raw_box <-Box.test(auto_model_raw $resid, type = "Ljung-Box")
manual_model_raw_sw <-shapiro.test(manual_model_raw $resid)
manual_model_raw_box <-Box.test(manual_model_raw $resid, type = "Ljung-Box")
manual_model_log_sw <-shapiro.test(manual_model_log $resid)
manual_model_log_box <-Box.test(manual_model_log $resid, type = "Ljung-Box")
results <- data.frame(cbind(
  rbind(auto_model_log$aicc, auto_model_raw$aicc, manual_model_raw$aicc, manual_model_log$aicc),
  rbind(auto_model_log$bic, auto_model_raw$bic, manual_model_raw$bic, manual_model_log$bic),
  rbind(auto_model_log_sw$p.value,auto_model_raw_sw$p.value,
        manual_model_raw_sw$p.value, manual_model_log_sw$p.value),
  rbind(auto_model_log_box$p.value, auto_model_raw_box$p.value,
```

```
        manual_model_raw_box$p.value, manual_model_log_box$p.value),
  rbind(calculate_rmse(forecast_log_auto$mean,ts_ecom_test),
        calculate_rmse(forecast_auto$mean,ts_ecom_test),
        calculate_rmse(forecast_log_man$mean, ts_ecom_test),
        calculate_rmse(forecast_man$mean, ts_ecom_test))))

colnames(results) <- c('AICc', '       BIC', ' Shapiro-Wilk', ' Box-Ljung', '   RMSE')
rownames(results) <- c('Log, Auto','Raw, Auto','Log, Manual', 'Raw, Manual')
round(results,3)
```

```
##                 AICc       BIC  Shapiro-Wilk   Box-Ljung   RMSE
## Log, Auto   -206.038  -200.424         0.003       0.095  0.349
## Raw, Auto    -80.571   -76.747         0.170       0.609  0.356
## Log, Manual  -84.093   -70.954         0.079       0.269  0.459
## Raw, Manual -116.332  -106.972         0.104       0.434  0.990
```

Aside of the auto.arima specification over the transformed series, all models show no statistical significance for the Shapiro-Wilk or the Box-Ljung Test. We fail to prove that residuals are not normally distributed or that residuals are autocorrelated in each case except when using the AutoArima specification over the transformed series.

The auto.arima specification over the log transformed model ('Log,Auto') has the best AICc, BIC and RMSE in out of sample comparisons. However, this specification also leads to non-normal residuals and potential autocorrelation in the residuals (as identified by the Shapiro-Wilk and Box-Ljung test). We therefore decided to not consider this model any longer. Between the remaining models, we have to decide if we prefer to go with the model with the lowest AICc/BIC ('Raw,Manual') or the model with the lowest RMSE ('Raw,Auto'), or the model that exhibits a compromise between those two ('Log,Manual'). We chose to continue with the auto arima specification over the untransformed series because it has the best RMSE without failing tests for residual normality and autocorrelation.

Forecast for 2017

To produce our final forecast, we will extend our forecast from our best performing model to include 2017. We will not retrain the final model using 2015-2016 data.
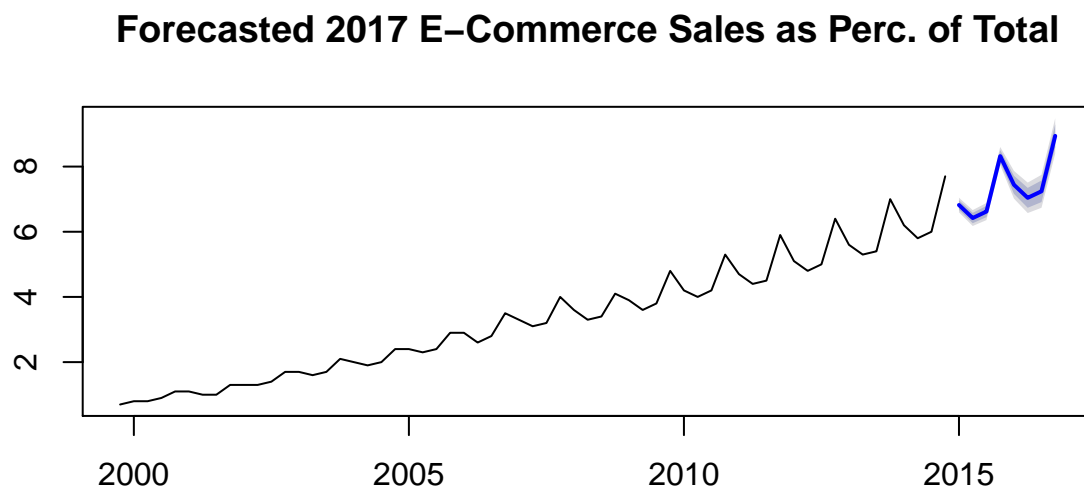
Our final model has the following form:

$$(1 - B) \cdot (1 - B^4) \cdot y_t = (1 - 0.5680103 \cdot B) \cdot e_t$$

We visualize the 2017 forecast in the below plot.

```
plot(forecast(auto_model_raw, h = 8), main='Forecasted 2017 E-Commerce Sales as Perc. of Total')
```


Question 2:

We will build a vector autoregressive (VAR) model to forecast four time series. Neither the names, nor the units of measures are given for the time series. We will therefore have to rely exclusively on statistical metrics, tests and graphical analysis to specify the order of

the VAR model and to measure the model fit. We will also discuss possible enhancments of the model that might provide a better fit.

Exploration and Visualization

First, we load the time series, split the data into test and train data, display the first few rows, check for missing values and print a summary of the train data.

```
mts = read.csv(file = "data_2018Spring_MTS.txt", sep=' ')
ts_mts = ts(mts[ , c("series1", "series2", "series3", "series4")], start = c(1947, 1), freq = 12)
ts_mts_train = ts(ts_mts[time(ts_mts) < 1993, ], start = c(1947, 1), freq = 12)
ts_mts_test = ts(ts_mts[time(ts_mts) >= 1993, ], start = c(1993, 1), end = c(1993,12), freq = 12)
head(ts_mts); sum(is.na(ts_mts)); summary(ts_mts_train)
```

```
##          series1 series2 series3 series4
## Jan 1947 11.1764 21.5500  5.8735 13.5172
## Feb 1947 11.5193 21.2241  5.9369 13.6239
## Mar 1947 11.7707 21.1833  5.9685 14.0771
## Apr 1947 11.8621 21.0204  6.0531 13.7305
## May 1947 11.7478 20.8575  6.0847 13.7838
## Jun 1947 11.8392 20.8575  6.1270 13.6505
```

```
## [1] 0
```

```
##     series1          series2          series3          series4
## Min.   :11.13    Min.   :20.86    Min.   : 5.007    Min.   :11.73
## 1st Qu.:17.51    1st Qu.:32.90    1st Qu.: 9.262    1st Qu.:21.94
## Median :32.95    Median :53.32    Median :16.938    Median :38.03
## Mean   :31.99    Mean   :51.62    Mean   :20.712    Mean   :36.38
## 3rd Qu.:43.79    3rd Qu.:68.19    3rd Qu.:31.952    3rd Qu.:49.90
## Max.   :59.54    Max.   :83.03    Max.   :47.987    Max.   :59.80
```

We have a total of four time series with no missing values. The time series all range between about 5 and 83. We plot our four time series with shared axes.

```
ts.plot(ts_mts_train, gpars=list(main="Plot of all four series", ylab="Unknown units",
                                 col=c('blue','red','green','black')))
legend("topleft",c("Series 1","Series 2","Series 3","Series 4"), lty=1,
       col = c('blue','red','green','black'), bty='n', cex=.75)
```

Plot of all four series
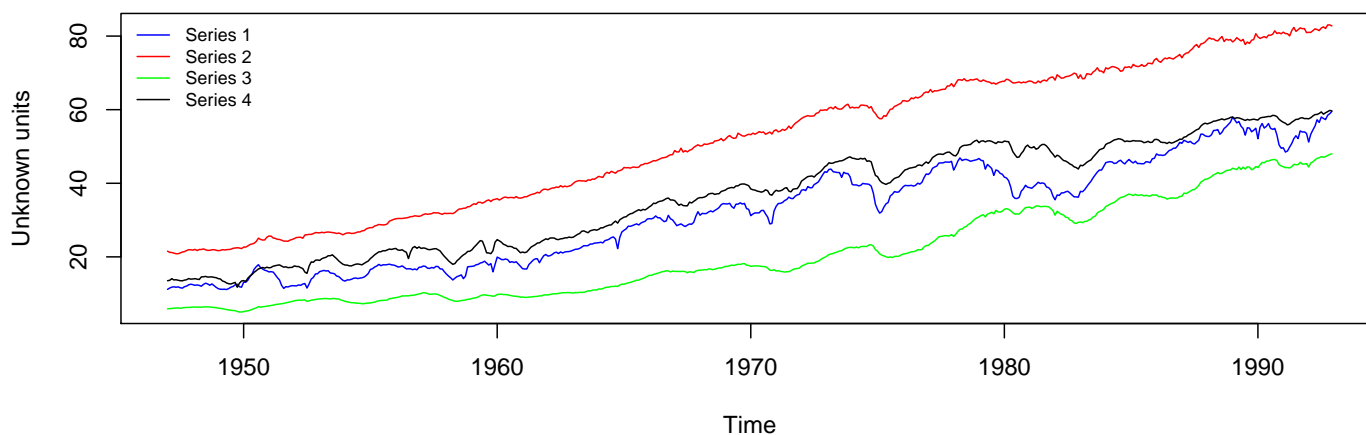
None of these series appear stationary: each actually appears to share a similar determinstic trend. The stochastic trend also seems to be similar between the four time series, though different in magnitude. Series1 has the strongest stochastic trend, followed by series4, series3 and series2. There does not appear to be an obvious seasonal trend within any of the four time series.

In the following plot we can see the distribution of the four time series, as well as pairwise scatterplots and correlation coefficients.

```
df_mts_train = data.frame(as.matrix(ts_mts_train), date=time(ts_mts_train))
ggpairs(df_mts_train[c('series1', 'series2', 'series3', 'series4')],
        title='Pairwise Scatterplot of series1, series2, series3 and series4')
```

Pairwise Scatterplot of series1, series2, series3 and series4
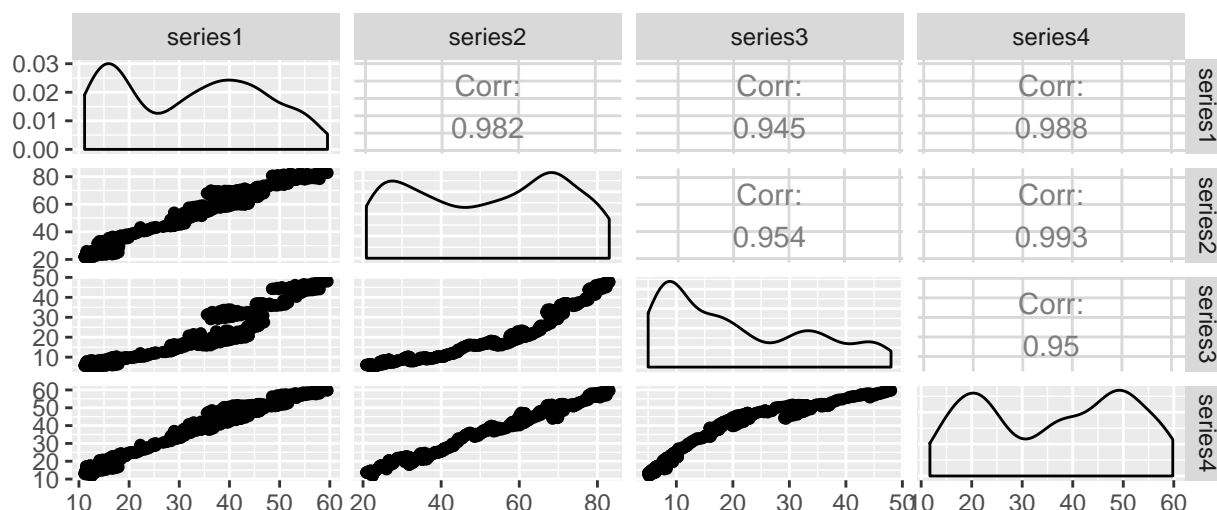
As expected, the series are highly correlated (> 0.95). High correlation however does not imply a causal relationship between the variables and in this case is uninformative due to the shared deterministic trend. Since we have no further information about the variables, we are unable to speculate on wheter or not this could be attributed to a common exogenous factor. We also see that the distributions for the series are fairly uniform except for series3, which is closer to following an exponential distribution.

Next, we conduct Dickey-Fuller Tests to test for unit roots. The null hypothesis states that the series contains unit roots (non-stationary stochastic trend), while the alternative hypothesis states that the series contains no unit roots (stationary stochastic trend).

```r
cbind(adf.test(df_mts_train$series1)$p.value, adf.test(df_mts_train$series2)$p.value,
adf.test(df_mts_train$series3)$p.value, adf.test(df_mts_train$series4)$p.value)
```

```
##           [,1]      [,2]      [,3]       [,4]
## [1,] 0.0701259 0.4872257 0.4880169 0.02509535
```

For series4, we reject the null hypothesis of non-stationarity. For all the other time series we fail to reject the null hypothesis. This means series 1 - 3 are likely to contain unit roots.

Next, we check if differencing alleviates this, by conducting Dickey-Fuller Tests on the the differenced series.

```r
cbind(adf.test(diff(df_mts_train$series1))$p.value, adf.test(diff(df_mts_train$series2))$p.value,
adf.test(diff(df_mts_train$series3))$p.value, adf.test(diff(df_mts_train$series4))$p.value)
```

```
##      [,1] [,2] [,3] [,4]
## [1,] 0.01 0.01 0.01 0.01
```

For the lag 1 differenced series, we reject the null hypothesis of non-stationarity for all four series.

The acf and pacf of the original time series (not shown here) showed that autocorrelations were significant for all lags, time series, and pairs of time series displayed in the plot. This is due to the fact that all series have a positive trend. We observe occasionally significant PACF values for higher lags in select pairs of times series - this suggests that the final VAR model may use a high order. We decided to difference all four series to achieve stationarity - the acf and pacf for the differenced time series are displayed below.

```r
acf(diff(ts_mts_train))
```
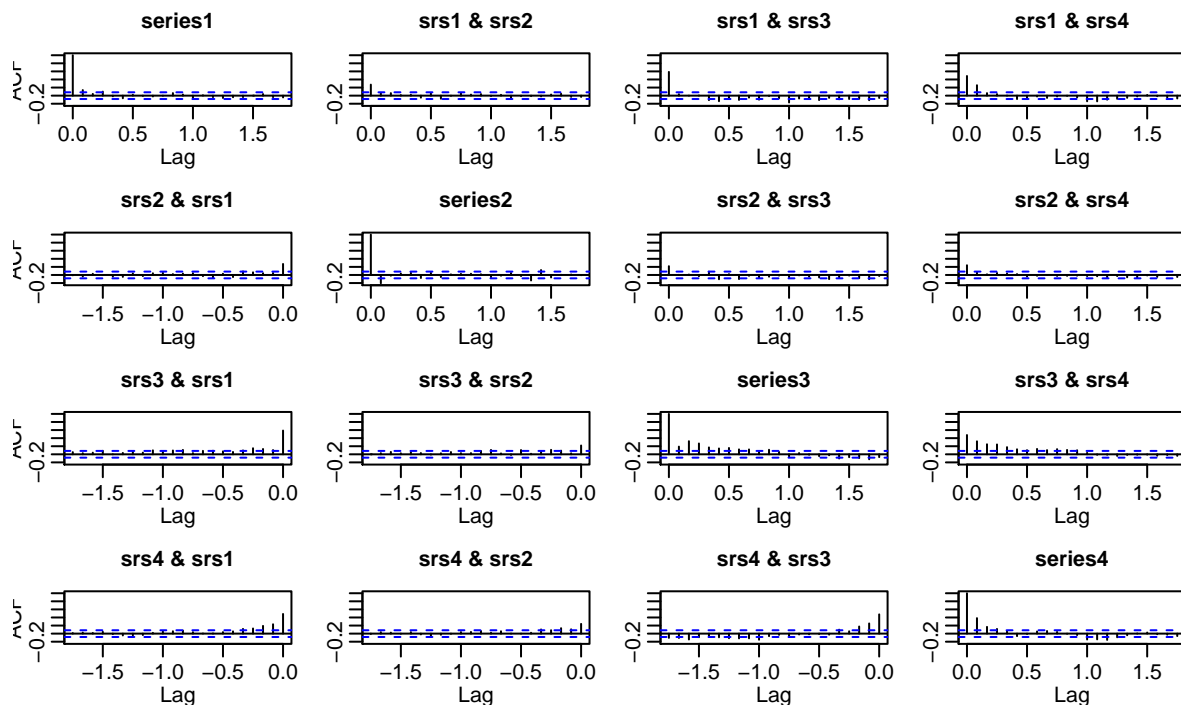
```r
pacf(diff(ts_mts_train))
```
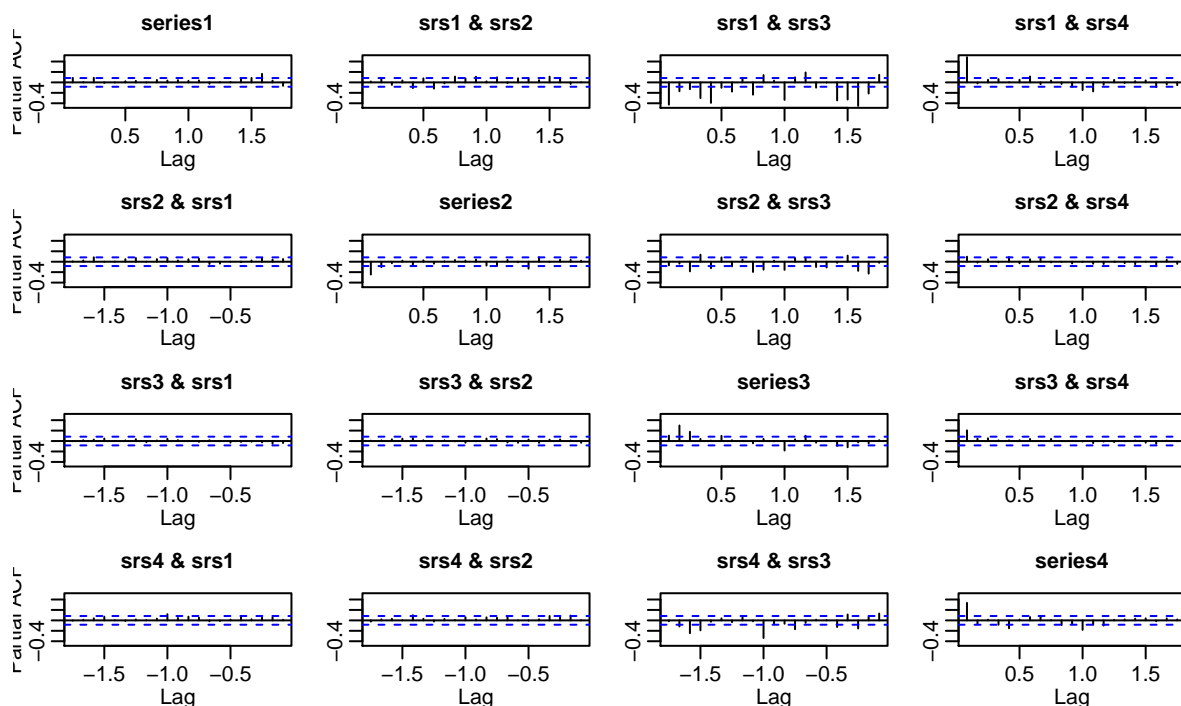

The differenced series' pacf and acf display statistically more convenient behavior. Differencing eliminates the positive drift from each of the series - which results in informative ACF plots. We furthermore do not observe any obvious seasonality as evidenced by PACF charts of differenced and non-differenced (not displayed) series. Further differencing did not provide any further insights from either acf or pacf (not shown).

We examine cointegration of our four series by conducting Phillips-Ouliaris tests, with the null hypothesis that two series are not cointegrated.

```r
cbind(po.test(df_mts_train[c('series1','series2')])$p.value, #column 1
po.test(df_mts_train[c('series1','series3')])$p.value, #column 2
po.test(df_mts_train[c('series1','series4')])$p.value, #column 3
```

```
po.test(df_mts_train[c('series2','series3')])$p.value, #column 4
po.test(df_mts_train[c('series2','series4')])$p.value, #column 5
po.test(df_mts_train[c('series3','series4')])$p.value) #column 6
```

```
##          [,1] [,2]       [,3] [,4] [,5] [,6]
## [1,] 0.024708 0.15 0.03785969 0.15 0.01 0.15
```

In a few cases we reject the null hypothesis: series1/series2, series1/series4 and series2/series4. In those cases it is likely that our time series are cointegrated. This means we would be able to fit a linear regression between both time series.

Since some of the series are cointegrated, we acknowledge that we should be using a VAR specification that includes an error correction term (vector error correction model) and alternative estimation methods to least squares. Since this was not covered in the course, we will have to model a simple VAR model despite detecting cointegration.

Modelling

First, we search for the order of the VAR model by using the VARselect function.

```
VARselect(diff(ts_mts_train), type = 'both')$selection
```

```
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      5      2      2      5
```

For VAR models we preferably use the BIC (SC). The minimum BIC is achieved for the order 2 VAR model. We specify our VAR 2 model and immediatly check if the residuals are well behaved. For this, we are conducting a Portmanteau Test. The null hypothesis of this test is that there is no serial correlation.

```
var = VAR(diff(ts_mts_train), p = 2, type = 'both')
serial.test(var, type = 'PT.asymptotic')$serial$p.value[1]
```

```
##  Chi-squared
## 4.027162e-05
```

We reject the null hypothesis of no serial correlation for the VAR(2) model. We also conducted this test for VAR(3), VAR(4) and VAR(5). For VAR(3) we received a p-value of around 0.015 and for VAR(4) we received a p-value of 0.08. The p-value for VAR(4) is the first p-value that does not indicate statistical significance. However, it is somewhat of a borderline case. To gain higher confidence we go one step further and chose a VAR(5) model - which also happens to be the one that was selected by the AIC. The p-value for the VAR(5) models is about 0.23 and thus far away from statistical significance.

```
var_model = VAR(diff(ts_mts_train), p = 5, type = 'both')
```

Forecasting for 1993 and 1994

Next, we calculate a two-year forecast based on our model. We plot our forecast for all four time series. To better visualize the forecasts, we only plot the time series from 1985 onwards. The training time series (1/1985 - 12/1992) is plotted in black, while the forecast (1/1993 - 12/1994) is plotted in blue and the test time series (1/1993 - 12/1993) is plotted in green. We also define a function to reconstruct our individual time series from differenced values.

```
calculate_int = function(model, forecast_steps, endvals){
  diff_forecast = forecast(model, h=forecast_steps)
  int_1 = endvals[1] + cumsum(diff_forecast$forecast$series1$mean)
  int_2 = endvals[2] + cumsum(diff_forecast$forecast$series2$mean)
  int_3 = endvals[3] + cumsum(diff_forecast$forecast$series3$mean)
  int_4 = endvals[4] + cumsum(diff_forecast$forecast$series4$mean)
  return(list(int_1, int_2, int_3, int_4))
}
int = calculate_int(var_model, 24, tail(ts_mts_train, n=1))
ts_forecast = ts(cbind(int[[1]], int[[2]], int[[3]], int[[4]]), start=c(1993, 1), freq=12)
ts_mts_train_window = window(ts_mts_train, start = c(1985,1))

par(mfrow=c(1,2), mai = c(.4, 0.4, 0.4, 0.4))
ts.plot(ts_mts_train_window[,1], ts_mts_test[,1], ts_forecast[,1], col=c('black', 'green', 'blue'),
        main = 'series1 with forecast')
legend(1985, 63, legend=c('Predicted','Observed', 'historic'), col=c('blue', 'green', 'black'),
       lty=1, cex=0.8)
ts.plot(ts_mts_train_window[,2], ts_mts_test[,2], ts_forecast[,2], col=c('black', 'green', 'blue'),
        main = 'series2 with forecast')
```

series1 with forecast

series2 with forecast
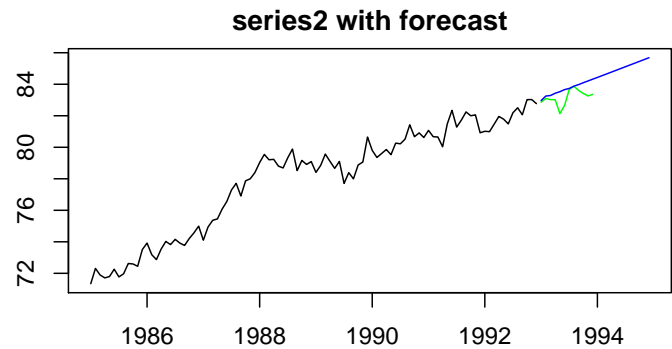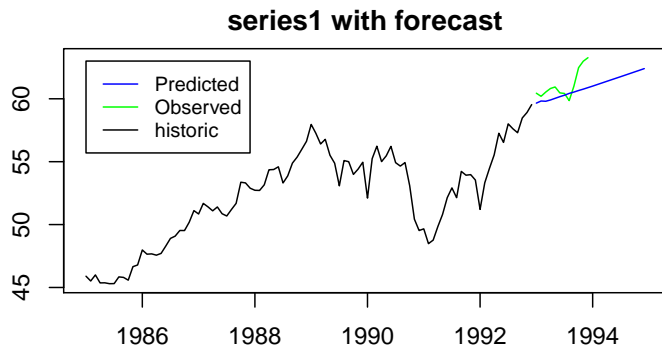

```
par(mfrow=c(1,2), mai = c(.4, 0.4, 0.4, 0.4))
ts.plot(ts_mts_train_window[,3], ts_mts_test[,3], ts_forecast[,3], col=c('black', 'green', 'blue'),
        main = 'series3 with forecast')
legend(1985, 50, legend=c('Predicted','Observed', 'historic'), col=c('blue', 'green', 'black'),
       lty=1, cex=0.8)
ts.plot(ts_mts_train_window[,4], ts_mts_test[,4], ts_forecast[,4], col=c('black', 'green', 'blue'),
        main = 'series4 with forecast')
```

series3 with forecast

series4 with forecast
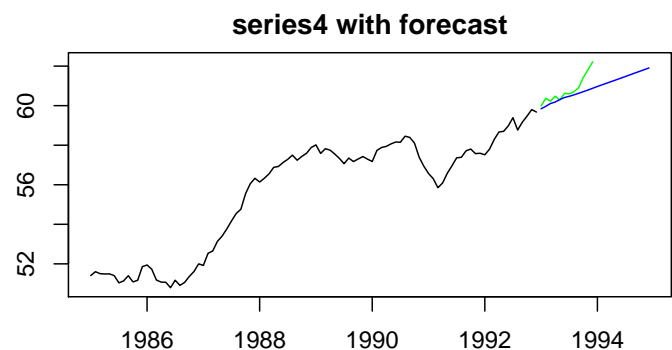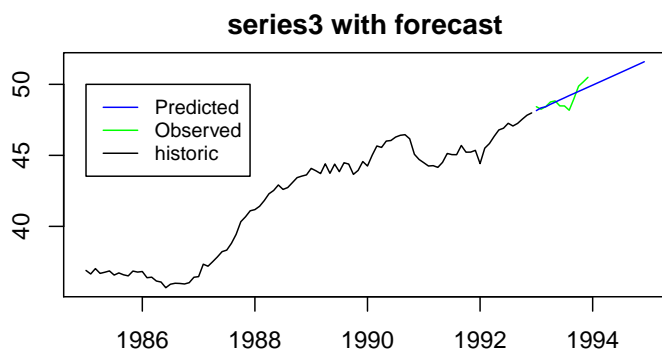
Our forecasts appear to be smoothed linear continuations of the last two years of the training time series.

Diagnostics

We conduct some in-depth residual diagnostics over the VARselect specification. While we plotted the residuals, we found that with only 12 points, it was difficult to tell if the series was truly white noise. Thus, residual plots are not included.

```
residuals = as.numeric(ts_mts_test) - window(ts_forecast, start = c(1993,1), end = c(1993,12))
#plot(residuals, main = 'Residuals of VAR(5) model', ylab = 'Date', type='b')
```

In our model selection process, we already examined the residuals using an asymptotic Portmanteau Test. We received a p-value of 0.2308 for the VAR(5) model. Next, we conduct a Breush-Godfrey LM test, with null hypothesis that there is no serial correlation

```
serial.test(var_model, type="BG")$serial$p.value
```

```
## Chi-squared
##   0.8203193
```

The Breush-Godfrey gives us a p-value of 0.9054, which means we have statistical significance and the residuals are likely not to be serially correlated.

Finally, we conduct a Jarque-Bera test for normality.

```
normality.test(var_model, multivariate.only = TRUE)$jb.mul$JB$p.value[1,1]
```

```
## [1] 0
```

We reject the null hypothesis with a highly statistically significant test result: proving that our residuals are not sampled from a normal distribution.

```
int = calculate_int(var_model, 12, tail(ts_mts_train, n=1))
df_RMSE = data.frame(rbind(calculate_rmse(int[[1]], ts_mts_test[,1]),
         calculate_rmse(int[[2]], ts_mts_test[,2]), calculate_rmse(
            int[[3]], ts_mts_test[,3]), calculate_rmse(int[[4]], ts_mts_test[,4])))

colnames(df_RMSE) = c('RMSE')
rownames(df_RMSE) = c('series1','series2','series3', 'series4')
df_RMSE
```

```
##              RMSE
## series1 1.2070008
## series2 0.6863241
## series3 0.4763056
## series4 0.5641822
```

Conclusion

Our final VAR model is not satisfactory in terms of fit and diagnostic test results. We now go on to explore how different orders perform in terms of RMSE, AIC, BIC, and diagnostic tests. We see that no single order provides an objectively "best" fit and we instead face a trade-off between different metrics. The table below shows an overview of different model specifications from VAR of order 1 to VAR of order 10 (indicated by the row label), with AIC, BIC, three different p-values for statistical tests and the sum of the root mean squared error over all four series.

```r
results = data.frame(); orderrange = 1:10
for(i in orderrange) {
  var_iter = VAR(diff(ts_mts_train), p = i, type="both")
  int = calculate_int(var_iter, 12, tail(ts_mts_train, n=1))

  PT = serial.test(var_iter, type="PT.asymptotic")$serial$p.value
  BG = serial.test(var_iter, type="BG")$serial$p.value
  norm = normality.test(var_iter, multivariate.only = TRUE)$jb.mul$JB$p.value
  rmse_sum = calculate_rmse(int[[1]], ts_mts_test[,1]) + calculate_rmse(int[[2]], ts_mts_test[,2]) +
             calculate_rmse(int[[3]], ts_mts_test[,3]) + calculate_rmse(int[[4]], ts_mts_test[,4])
  results = rbind(results, cbind(AIC(var_iter), BIC(var_iter), PT, BG, norm, round(rmse_sum,2)))
}
rownames(results) = orderrange
colnames(results) = c('AIC', '       BIC', ' Port. p-val',
                      ' BG p-val', 'JB p-val', 'sum(RMSE)')
round(results, 3)
```

```
##         AIC       BIC  Port. p-val  BG p-val  JB p-val  sum(RMSE)
## 1  2112.143  2215.581        0.000     0.000         0       3.06
## 2  2035.322  2207.646        0.000     0.000         0       3.11
## 3  2009.693  2250.845        0.010     0.009         0       2.97
## 4  2004.074  2313.995        0.055     0.121         0       3.03
## 5  1989.995  2368.626        0.166     0.820         0       2.93
## 6  2007.914  2455.196        0.122     0.803         0       2.84
## 7  2024.126  2540.000        0.131     0.076         0       2.85
## 8  2041.797  2626.204        0.072     0.020         0       2.89
## 9  2056.273  2709.154        0.055     0.046         0       2.91
## 10 2068.792  2790.086        0.026     0.052         0       2.44
```
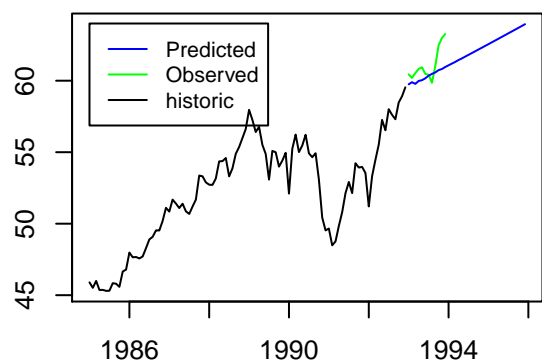
Given these data, we do believe that a VAR(7) model is actually more appropriate than a VAR(5) model. We will use this specification to make our final 12 month prediction (1995). The below plots include historic data, validation set performance (1993, 1994), and our 12 month post sample prediction (1995).

```r
var_model_final = VAR(diff(ts_mts_train), p = 7, type = 'both')

int = calculate_int(var_model_final, 36, tail(ts_mts_train, n=1))
ts_forecast_fin = ts(cbind(int[[1]], int[[2]], int[[3]], int[[4]]), start=c(1993, 1), freq=12)

par(mfrow=c(2,2), mai = c(.4, 0.4, 0.4, 0.4))
ts.plot(ts_mts_train_window[,1], ts_mts_test[,1], ts_forecast_fin[,1], col=c('black', 'green', 'blue'),
        main = 'series1 with forecast')
legend(1985, 64, legend=c('Predicted','Observed', 'historic'), col=c('blue', 'green', 'black'),
       lty=1, cex=0.8)
ts.plot(ts_mts_train_window[,2], ts_mts_test[,2], ts_forecast_fin[,2], col=c('black', 'green', 'blue'),
        main = 'series2 with forecast')
ts.plot(ts_mts_train_window[,3], ts_mts_test[,3], ts_forecast_fin[,3], col=c('black', 'green', 'blue'),
        main = 'series3 with forecast')
ts.plot(ts_mts_train_window[,4], ts_mts_test[,4], ts_forecast_fin[,4], col=c('black', 'green', 'blue'),
        main = 'series4 with forecast')
```
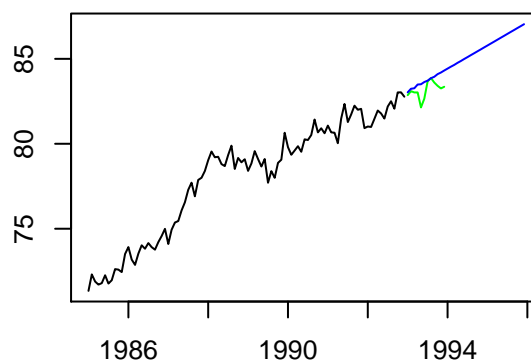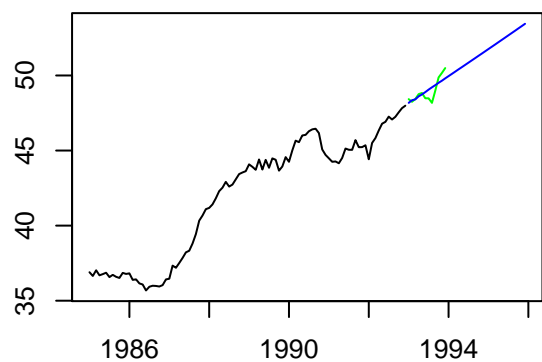
series1 with forecast

series2 with forecast

series3 with forecast

series4 with forecast