DATA 624 Fall 2017: Project 1

Dan Smilowitz

October 24, 2017

library(tidyverse)  
library(scales)  
theme\_set(theme\_light())

# Part A: ATM Forecast

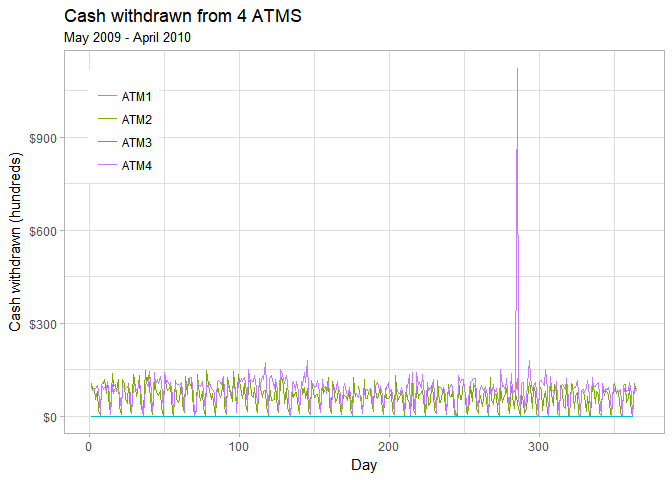
All ATM data is read into R, then converted to a ts object with one series for each ATM:

# read in data  
library(readxl)  
ATM\_df <- read\_excel("data/ATM624Data.xlsx")  
# drop missing columns, convert each ATM to column, fix date column  
ATM\_df <- ATM\_df %>%  
 drop\_na() %>%  
 spread(ATM, Cash) %>%   
 mutate(DATE = as.Date(DATE, origin = "1899-12-30")) # in Excel, 1 == 1/1/1900  
# convert to timeseries  
ATM\_ts <- ts(ATM\_df %>% select(-DATE))

## Data Exploration

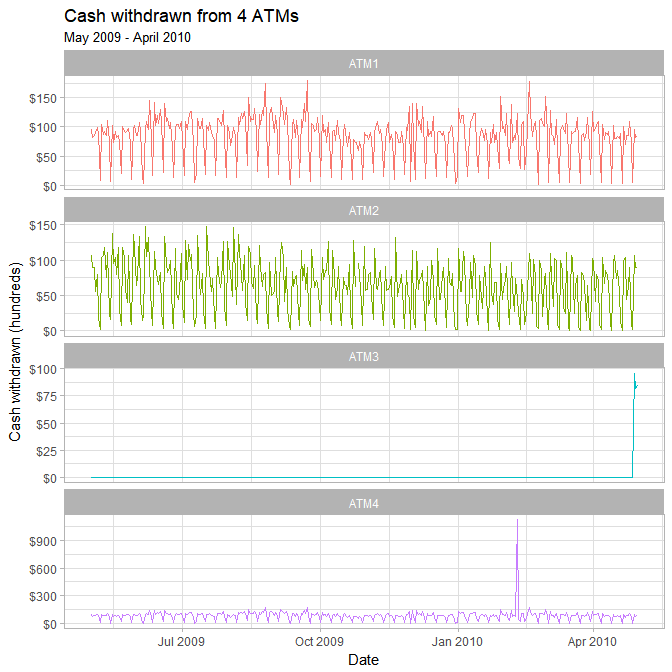
The timeseries can be plotted with a separate line for each ATM:

library(forecast) # to plot ts objects  
autoplot(ATM\_ts) +  
 labs(title = "Cash withdrawn from 4 ATMS",  
 subtitle = "May 2009 - April 2010",  
 x = "Day") +  
 scale\_y\_continuous("Cash withdrawn (hundreds)", labels = dollar) +  
 scale\_color\_discrete(NULL) +  
 theme(legend.position = c(0.1, 0.8))



The plot is a bit too busy to interpret, so the original data frame object is used instead to try to gain some additional clarity:

# convert df back to tidy format for faceted plotting  
ATM\_df %>% gather(ATM, Cash, -DATE) %>%   
 # plot  
 ggplot(aes(x = DATE, y = Cash, col = ATM)) +  
 geom\_line(show.legend = FALSE) +  
 facet\_wrap(~ ATM, ncol = 1, scales = "free\_y") +  
 labs(title = "Cash withdrawn from 4 ATMs",  
 subtitle = "May 2009 - April 2010",  
 x = "Date") +  
 scale\_y\_continuous("Cash withdrawn (hundreds)", labels = dollar)



From this chart, it is clear that ATM1 & ATM2 see a fair amount of variation between $0-$15,000, with only a few observations exceeding these values. ATM3 shows zero withdrawals for most of the year until the final 3 days, with observations in the area of $10,000. ATM4 shows a similar pattern as ATM1 & ATM2, with the exception of one day showing withdrawals over $100,000.

As explained above, each of the ATMs behaves differently; as such, each will be forecasted separately, using the following approach based on the previously-stated observations:

* ATM1 & ATM2 each exhibit similar patterns through the time window and will use the entire timeseries.
* It is possible that ATM3 was inactive and was only put in service in the last three days of the supplied window; these three days will be used to forecast future periods.
* It is reasonable to assume that the one-day spike observed at ATM4 was an aberration and will not be considered in forecasting.

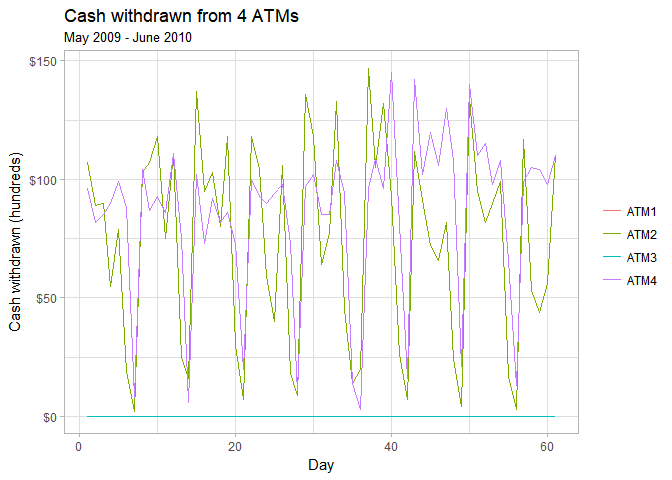
Separate timeseries objects are created to perform these forecasts:

# create ts objects for ATM1 & ATM2  
ATM1 <- ATM\_ts[, "ATM1"]  
ATM2 <- ATM\_ts[, "ATM2"]  
# get last 3 observations of ATM3 & convert to ts  
ATM3 <- ATM\_ts[(nrow(ATM\_ts) - 2):nrow(ATM\_ts), "ATM3"]  
ATM3 <- ts(ATM3, start = 363)  
# create ts object for ATM3 & replace observations of 0 with NA  
ATM3 <- ATM\_ts[, "ATM3"]  
ATM3[which(ATM3 == 0)] <- NA  
# create ts object for ATM4 & impute spike with median  
ATM4 <- ATM\_ts[, "ATM4"]  
ATM4[which.max(ATM4)] <- median(ATM4, na.rm = TRUE)

## Fitting

Viewing the plots of ATM1, ATM2, & ATM4, it appears that there may be some form of seasonality in the withdrawals from the ATMs. This can be further investigated by viewing the first two months of the data:

autoplot(ts(ATM\_ts[1:61, ])) +  
 labs(title = "Cash withdrawn from 4 ATMs",  
 subtitle = "May 2009 - June 2010",  
 x = "Day") +  
 scale\_y\_continuous("Cash withdrawn (hundreds)", labels = dollar) +  
 scale\_color\_discrete(NULL)



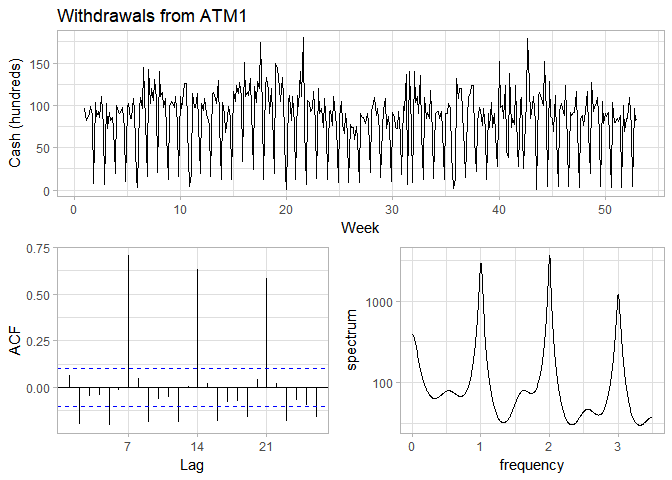
From this plot, it appears that there is a weekly seasonality present in the data -- each of the ATMs with data in this period show 8 distinct dips over the two-month period. Daily timeseries (i.e. ts objects with frequency = 1) can not be decomposed to capture the seasonality identified above. Per Rob J. Hyndman's [website](https://robjhyndman.com/hyndsight/dailydata/), the easiest way to capture weekly seasonal behavior is to set frequency = 7. This is performed for ATMs 1, 2, & 4:

ATM1 <- ts(ATM1, frequency = 7)  
ATM2 <- ts(ATM2, frequency = 7)  
# impute NA value -- decomposition can not be performed on series with NA  
ATM2[which(is.na(ATM2))] <- median(ATM2, na.rm = TRUE)  
ATM4 <- ts(ATM4, frequency = 7)

### ATM 1

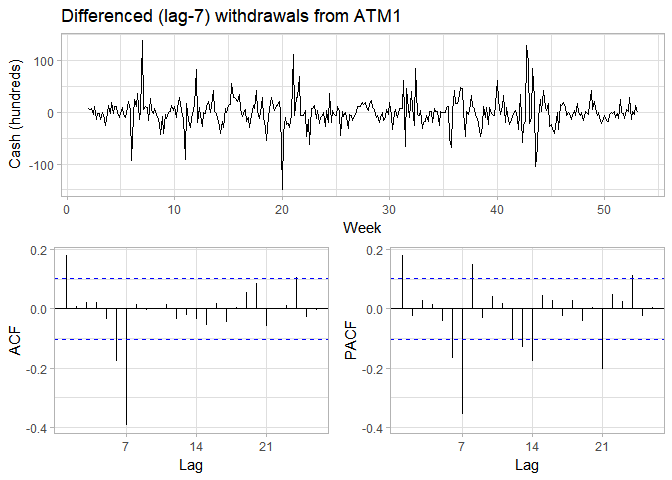
The ATM1 timeseries is displayed below with its ACF & spectrum plots:

ggtsdisplay(ATM1, points = FALSE, plot.type = "spectrum",  
 main = "Withdrawals from ATM1", xlab = "Week", ylab = "Cash (hundreds)")



The ACF & spectrum plots show a very clear weekly seasonality -- there are large spikes in the ACF lags 7, 14, and 21 as well as large spikes in the spectrum plot at frequencies 1, 2, and 3. Both of these suggest a seasonal ARIMA model. To account for the above-identified autocorrelation, the time series is differenced with a lag of 7:

ggtsdisplay(diff(ATM1, 7), points = FALSE,  
 main = "Differenced (lag-7) withdrawals from ATM1",  
 xlab = "Week", ylab = "Cash (hundreds)")



The timeseries appears stationary, so no non-seasonal differencing is suggested by the data. The significant spikes in the ACF and PACF at suggest non-seasonal AR(1) and/or MA(1) components of the model. The spikes in the ACF and PACF at followed by decreasing spikes at and suggest seasonal AR(1) and/or seasonal MA(1) components. This suggests fifteen possible models: ARIMA(p, 0, q)(P, 1, Q) for excluding the case where

The models are calculated and their AIC values returned:

# get optimal lambda for Box-cox transformation  
ATM1\_lambda <- BoxCox.lambda(ATM1)  
# define function to create models & return AIC values for timeseries  
ATM\_aic <- function(p, d, q, P, D, Q) {  
 # create model with Box-Cox and specified ARIMA parameters; extract AIC  
 AIC(Arima(ATM1, order = c(p, d, q), seasonal = c(P, D, Q), lambda = ATM1\_lambda))  
}  
# create possible combinations of p, q, P, Q except all zero  
expand.grid(p = 0:1, q = 0:1, P = 0:1, Q = 0:1) %>%  
 filter(p > 0 | q > 0 | P > 0 | Q > 0) %>%   
 # calc AIC for models  
 mutate(aic = pmap\_dbl(list(p, 0, q, P, 1, Q), ATM\_aic)) %>%   
 # return best AIC  
 slice(which.min(aic))

# A tibble: 1 x 5  
 p q P Q aic  
 <int> <int> <int> <int> <dbl>  
1 1 1 0 1 1221.26

The minimum AIC value is for non-seasonal AR(1) & MA(1) and seasonal AR(0) & MA(1) -- the model used is ARIMA(1,0,1)(0,1,1):

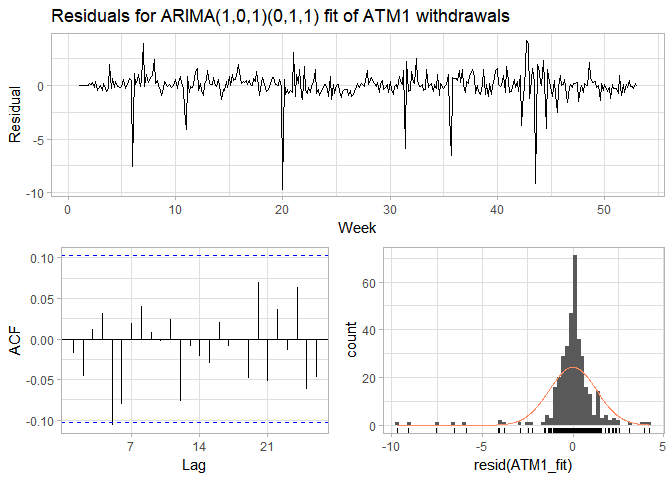
ATM1\_fit <- Arima(ATM1, order = c(1, 0, 1), seasonal = c(0, 1, 1), lambda = ATM1\_lambda)

The residuals are investigated using a Ljung-Box test and diagnostic plotting:

Box.test(resid(ATM1\_fit), type = "L", fitdf = 3, lag = 7)

Box-Ljung test  
  
data: resid(ATM1\_fit)  
X-squared = 8.0497, df = 4, p-value = 0.08977

ggtsdisplay(resid(ATM1\_fit), points = FALSE, plot.type = "histogram",  
 main = "Residuals for ARIMA(1,0,1)(0,1,1) fit of ATM1 withdrawals",  
 xlab = "Week", ylab = "Residual")

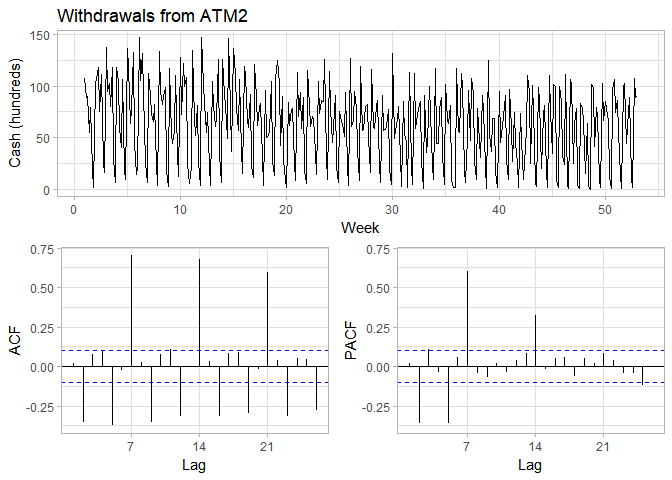


The Ljung-Box test returns a p-value > 0.05, suggesting that the residuals may be white noise. The residuals appear to be approximately normally distributed with a mean around zero. They do not appear to be autocorrelated, but there is an almost-significant spike at . This model is acceptable and will be used for forecasting.

### ATM 2

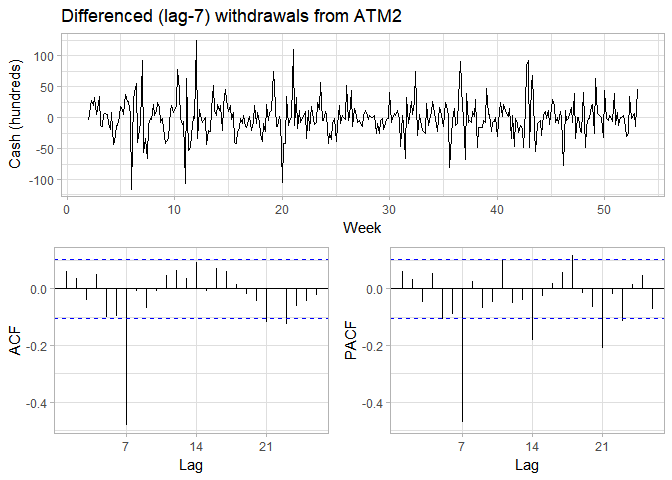
The same procedure is repeated for ATM2:

ggtsdisplay(ATM2, points = FALSE,  
 main = "Withdrawals from ATM2", xlab = "Week", ylab = "Cash (hundreds)")



The same weekly seasonality is seen as for ATM1; it is also differenced with lag = 7:

ggtsdisplay(diff(ATM2, 7), points = FALSE,  
 main = "Differenced (lag-7) withdrawals from ATM2",  
 xlab = "Week", ylab = "Cash (hundreds)")



As above, the large spike at suggests , while the stationary nature of the timeseries suggests . The spikes in ACF & PACF in the non-differenced series at & suggest .The lack of significant spikes in the ACF & PACF of the differenced series do not strongly suggest any need for seasonal AR or MA elements, but since the values at are followed by decreasing values, are also investigated. Each of the above mentioned models are investigated using the function created above:

# get optimal lambda for Box-cox transformation  
ATM2\_lambda <- BoxCox.lambda(ATM2)  
# repurpose above function for ATM2  
ATM\_aic <- function(p, d, q, P, D, Q) {  
 # create model with Box-Cox and specified ARIMA parameters; extract AIC  
 AIC(Arima(ATM2, order = c(p, d, q), seasonal = c(P, D, Q), lambda = ATM2\_lambda))  
}  
# create possible combinations of p, q, P, Q except all zero  
expand.grid(p = c(0, 2, 5), q = c(0, 2, 5), P = 0:1, Q = 0:1) %>%  
 filter(p > 0 | q > 0 | P > 0 | Q > 0) %>%   
 # calc AIC for models  
 mutate(aic = pmap\_dbl(list(p, 0, q, P, 1, Q), ATM\_aic)) %>%   
 # return best AIC  
 slice(which.min(aic))

# A tibble: 1 x 5  
 p q P Q aic  
 <dbl> <dbl> <int> <int> <dbl>  
1 5 5 0 1 2338.334

The minimum AIC value is for non-seasonal AR(5) & MA(5) and seasonal AR(0) & MA(1) -- the model used is ARIMA(5,0,5)(0,1,1):

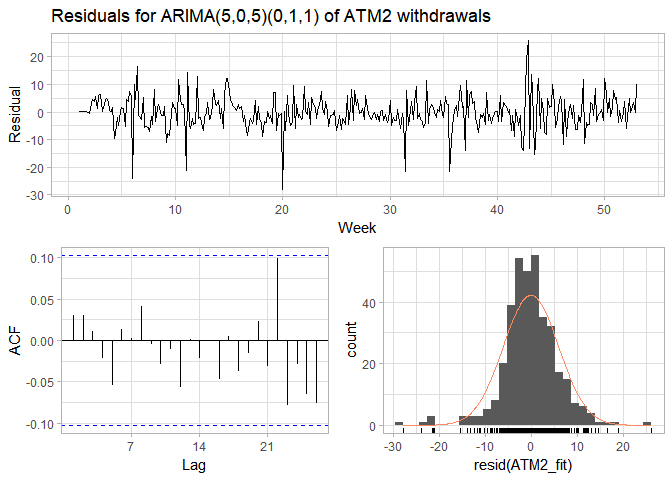
ATM2\_fit <- Arima(ATM2, order = c(5, 0, 5), seasonal = c(0, 1, 1), lambda = ATM2\_lambda)

The residuals are investigated using a Ljung-Box test and diagnostic plotting:

Box.test(resid(ATM2\_fit), type = "L", fitdf = 11, lag = 14)

Box-Ljung test  
  
data: resid(ATM2\_fit)  
X-squared = 4.4847, df = 3, p-value = 0.2137

ggtsdisplay(resid(ATM2\_fit), points = FALSE, plot.type = "histogram",  
 main = "Residuals for ARIMA(5,0,5)(0,1,1) of ATM2 withdrawals",  
 xlab = "Week", ylab = "Residual")



The Ljung-Box test (using lag = 14 due to the high number of parameters in the fit) returns a p-value >> 0.05, suggesting that the residuals may be white noise. The residuals appear to be approximately normally distributed with a mean around zero. This model is acceptable and will be used for forecasting.

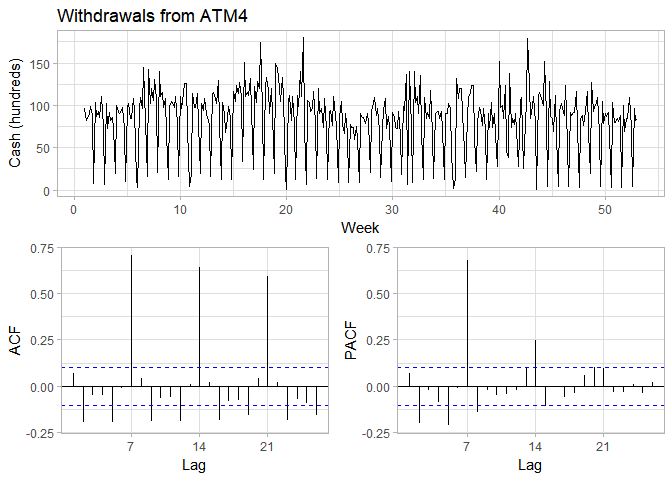
### ATM 3

As mentioned above, there are only three observations at ATM3, and only these observations are used for the forecast. A simple mean forecast will be used for this ATM.

### ATM 4

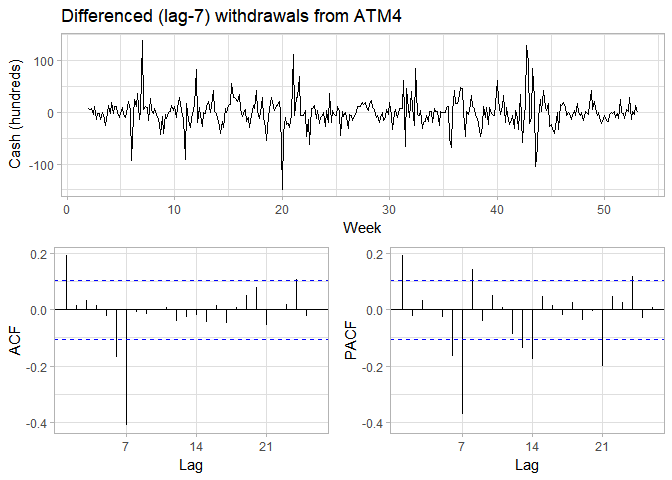
Finally, the procedure used for ATM1 & ATM2 is repeated for ATM4.

ggtsdisplay(ATM4, points = FALSE,  
 main = "Withdrawals from ATM4", xlab = "Week", ylab = "Cash (hundreds)")



The same weekly seasonality is seen as for ATM1 & ATM2; it is also differenced with lag = 7:

ggtsdisplay(diff(ATM4, 7), points = FALSE,  
 main = "Differenced (lag-7) withdrawals from ATM4",  
 xlab = "Week", ylab = "Cash (hundreds)")



Again, the stationary time series with a large spike at suggests and . Similar spikes in the ACF & PACF of both the original and differenced timeseries as ATM2 suggest and (though the evidence for seasonal AR and/or MA components are stronger in this case). The code from above is reused to investigate the same possible models for ATM4:

# get optimal lambda for Box-cox transformation  
ATM4\_lambda <- BoxCox.lambda(ATM4)  
# repurpose above function for ATM4  
ATM\_aic <- function(p, d, q, P, D, Q) {  
 # create model with Box-Cox and specified ARIMA parameters; extract AIC  
 AIC(Arima(ATM4, order = c(p, d, q), seasonal = c(P, D, Q), lambda = ATM4\_lambda))  
}  
# create possible combinations of p, q, P, Q except all zero  
expand.grid(p = c(0, 2, 5), q = c(0, 2, 5), P = 0:1, Q = 0:1) %>%  
 filter(p > 0 | q > 0 | P > 0 | Q > 0) %>%   
 # calc AIC for models  
 mutate(aic = pmap\_dbl(list(p, 0, q, P, 1, Q), ATM\_aic)) %>%   
 # return best AIC  
 slice(which.min(aic))

# A tibble: 1 x 5  
 p q P Q aic  
 <dbl> <dbl> <int> <int> <dbl>  
1 0 2 0 1 1161.917

The minimum AIC value is for non-seasonal AR(0) & MA(2) and seasonal AR(0) & MA(1) -- the model used is ARIMA(0,0,2)(0,1,1):

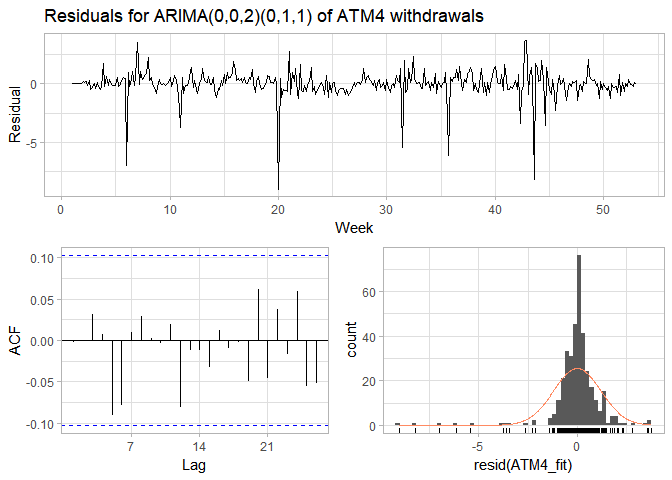
ATM4\_fit <- Arima(ATM4, order = c(0, 0, 2), seasonal = c(0, 1, 1), lambda = ATM4\_lambda)

The residuals are investigated using a Ljung-Box test and diagnostic plotting:

Box.test(resid(ATM4\_fit), type = "L", fitdf = 3, lag = 7)

Box-Ljung test  
  
data: resid(ATM4\_fit)  
X-squared = 5.7899, df = 4, p-value = 0.2154

ggtsdisplay(resid(ATM4\_fit), points = FALSE, plot.type = "histogram",  
 main = "Residuals for ARIMA(0,0,2)(0,1,1) of ATM4 withdrawals",  
 xlab = "Week", ylab = "Residual")



The Ljung-Box test again returns a p-value >> 0.05, with residuals approximately normally distributed with a mean around zero. This model is acceptable and will be used for forecasting.

## 

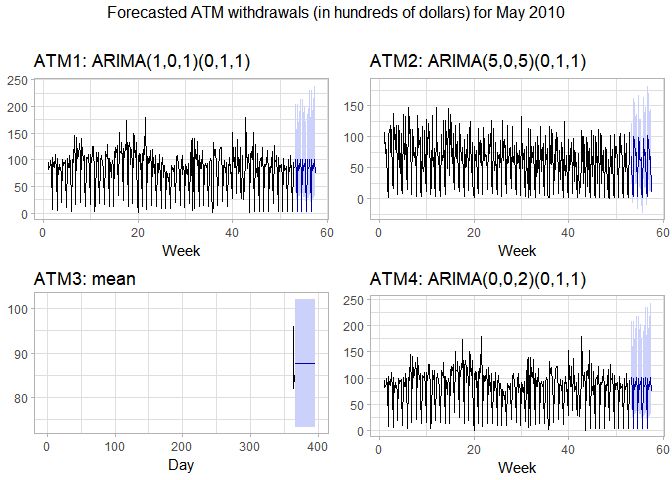
## Forecasting

The four forecasts identified above are performed for May 2010 (31 days):

ATM1\_forecast <- forecast(ATM1\_fit, 31, level = 95)  
ATM2\_forecast <- forecast(ATM2\_fit, 31, level = 95)  
ATM3\_forecast <- meanf(ATM3, 31, level = 95)  
ATM4\_forecast <- forecast(ATM4\_fit, 31, level = 95)

Each of the forecasts are plotted below:

gridExtra::grid.arrange(  
 autoplot(ATM1\_forecast) +   
 labs(title = "ATM1: ARIMA(1,0,1)(0,1,1)", x = "Week", y = NULL) +  
 theme(legend.position = "none"),  
 autoplot(ATM2\_forecast) +   
 labs(title = "ATM2: ARIMA(5,0,5)(0,1,1)", x = "Week", y = NULL) +  
 theme(legend.position = "none"),  
 autoplot(ATM3\_forecast) +   
 labs(title = "ATM3: mean", x = "Day", y = NULL) +  
 theme(legend.position = "none"),  
 autoplot(ATM4\_forecast) +   
 labs(title = "ATM4: ARIMA(0,0,2)(0,1,1)", x = "Week", y = NULL) +  
 theme(legend.position = "none"),  
 top = grid::textGrob("Forecasted ATM withdrawals (in hundreds of dollars) for May 2010\n")  
)



As expected, these values show seasonality for ATMs 1, 2, and 4, with a single value forecast for ATM3. The forecast values are gathered and output to a .csv, which is manually tranferred to Excel for submission:

data\_frame(DATE = rep(max(ATM\_df$DATE) + 1:31, 4),  
 ATM = rep(names(ATM\_df)[-1], each = 31),  
 Cash = c(ATM1\_forecast$mean, ATM2\_forecast$mean,  
 ATM3\_forecast$mean, ATM4\_forecast$mean)) %>%   
 write\_csv("project1\_ATM.csv")

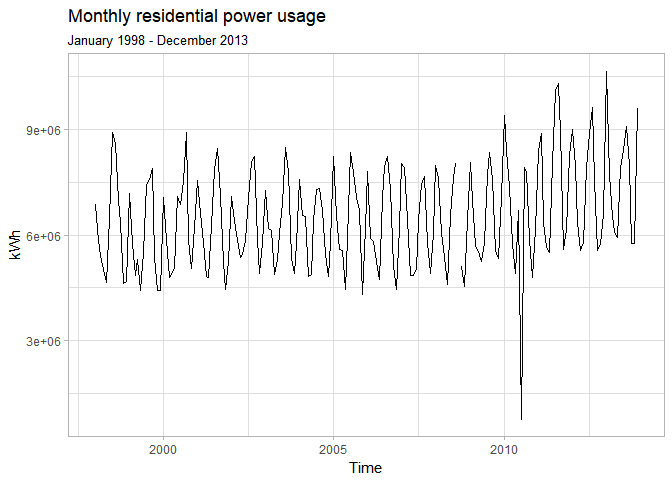
# Part B: Forecasting Power

kWh <- read\_excel("data/ResidentialCustomerForecastLoad-624.xlsx")  
kWh <- ts(kWh[, "KWH"], start = c(1998, 1), frequency = 12)

## Data Exploration

The timeseries is plotted to inspect its features:

autoplot(kWh) +  
 labs(title = "Monthly residential power usage",  
 subtitle = "January 1998 - December 2013")



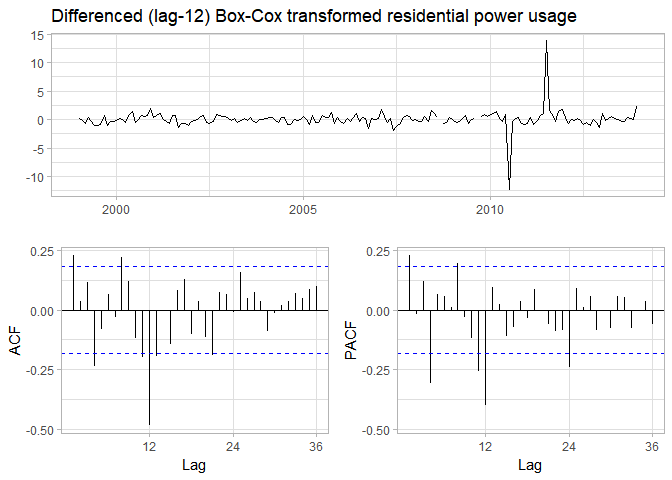
There is a clear seasonality in this data -- it appears to be semi-annual, with a peak every six months, but it may be annual, as the peaks seem to alternate in their height. This is consistent with problem 8.8 from *Forecasting: Principles and Practice*, completed in homework 1. There is a very noticeable dip in value in July 2010. It seems that the variance of the series may increase with its level; therefore a Box-cox transformation is also investigated.

# get Box-cox paramter  
kWh\_lambda <- BoxCox.lambda(kWh)  
kWh\_trans <- BoxCox(kWh, kWh\_lambda)

## Fitting

The data, transformed using , are plotted below with lag-12 differencing:

ggtsdisplay(diff(kWh\_trans, 12), points = FALSE,  
 main = "Differenced (lag-12) Box-Cox transformed residential power usage")



The series appears stationary, so no non-seasonal differencing appears necessary. The decaying seasonal spikes in the PACF suggests a seasonal AR(1) component, while the very quickly-decaying seasonal spikes in the ACF suggest the possibility of a seasonal MA(1) component. Spikes in the PACF and ACF at and suggest non-seasonal AR(1) or AR(4) components, and non-seasonal MA(1) or MA(4) components. The function used to select the model with lowest AIC in Part A is redefined for use on the kWh timeseries with and :

# redefine function  
kWh\_aic <- function(p, q, P, Q) {  
 # create model with Box-Cox and specified ARIMA parameters; extract AIC  
 AIC(Arima(kWh, order = c(p, 0, q), seasonal = c(P, 1, Q), lambda = kWh\_lambda))  
}  
# create possible combinations except all zero & p = q = 4; P = Q = 1 (returns error)  
expand.grid(p = c(0, 1, 4), q = c(0, 1, 4), P = 0:1, Q = 0:1) %>%  
 filter(p > 0 | q > 0 | P > 0 | Q > 0, p < 4 | q < 4 | P < 1 | Q < 1) %>%  
 # calc AIC for models  
 mutate(aic = pmap\_dbl(list(p, q, P, Q), kWh\_aic)) %>%   
 # return best AIC  
 slice(which.min(aic))

# A tibble: 1 x 5  
 p q P Q aic  
 <dbl> <dbl> <int> <int> <dbl>  
1 1 0 0 1 575.2659

The minimum AIC value returned is for the ARIMA(1,0,0)(0,1,1) model; this is used:

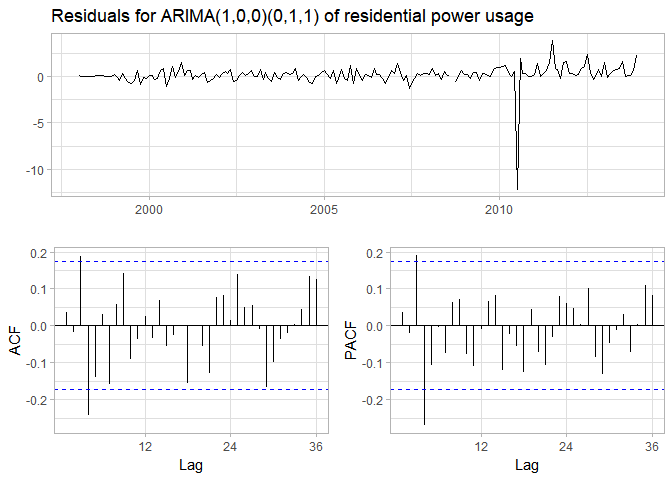
kWh\_fit <- Arima(kWh, order = c(1, 0, 0), seasonal = c(0, 1, 1), lambda = kWh\_lambda)

The residuals of this fit are investigated with a Ljung-Box test and diagnostic plotting:

Box.test(resid(kWh\_fit), type = "L", fitdf = 3, lag = 12)

Box-Ljung test  
  
data: resid(kWh\_fit)  
X-squared = 6.4543, df = 9, p-value = 0.6937

ggtsdisplay(resid(kWh\_fit), points = FALSE,  
 main = "Residuals for ARIMA(1,0,0)(0,1,1) of residential power usage")



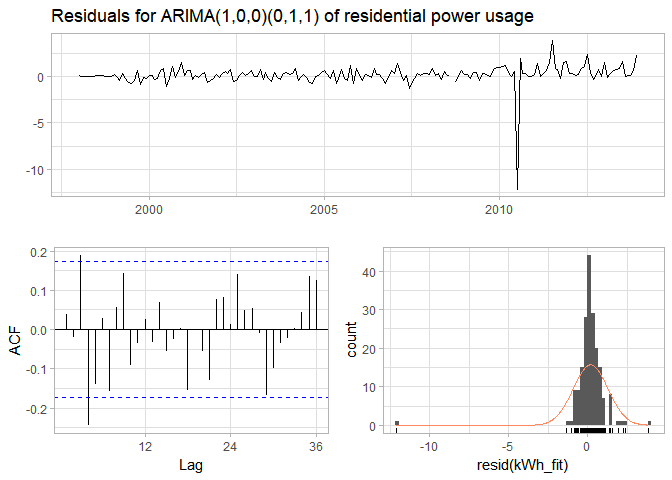
The Ljung-Box test returns a p-value >> 0.05, but the spikes in ACF & PACF at was addressed above). Investigation of these does not yield any AIC values lower than that of the above-identified model:

expand.grid(p = c(1, 3), q = c(1, 3)) %>%  
 mutate(aic = pmap\_dbl(list(p, q, 0, 1), kWh\_aic))

p q aic  
1 1 1 575.7090  
2 3 1 579.7385  
3 1 3 579.9391  
4 3 3 581.4511

Viewing the residuals of the fit model again with a histogram, the model is acceptable. The residuals appear to be roughly normally distributed around zero (with the exception of the significant dip in July 2010) without any significant autocorrelation:

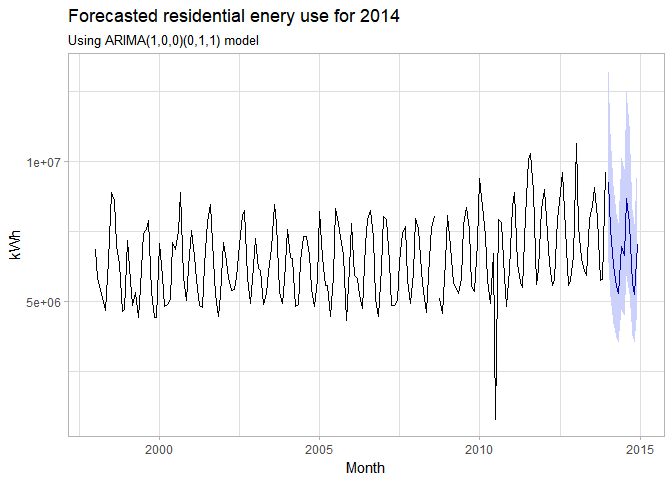
ggtsdisplay(resid(kWh\_fit), points = FALSE, plot.type = "histogram",  
 main = "Residuals for ARIMA(1,0,0)(0,1,1) of residential power usage")



## Forecasting

Using the ARIMA(1,0,0)(0,1,1) model, the next year (12 months) is forecast, and this forecast is plotted:

kWh\_forecast <- forecast(kWh\_fit, 12, level = 95)  
autoplot(kWh\_forecast) +   
 labs(title = "Forecasted residential enery use for 2014",  
 subtitle = "Using ARIMA(1,0,0)(0,1,1) model", x = "Month", y = "kWh") +  
 theme(legend.position = "none")



As expected, the forecast shows annual seasonality while showing some drift due to the non-seasonal autocorrelation. The forecast values are output to a .csv for inclusion in the required Excel submission:

data\_frame(`YYYY-MMM` = paste0(2014, "-", month.abb),  
 KWH = kWh\_forecast$mean) %>%   
 write\_csv("project1\_kWh.csv")

# Part C: Waterflow

## Data Munging & Exploration

Prior to loading in the provided data for the two pipelines, the following changes are made in both files to make the data more easily readable by R:

* Cell A1 is renamed from "Date Time" to "DateTime"
* The format of column A is changed to yyyy-mm-dd hh:mm
* The format of column A is changed to a number with 13 decimal places

water1 <- read\_excel("data/Waterflow\_Pipe1.xlsx")  
water2 <- read\_excel("data/Waterflow\_Pipe2.xlsx")

Both sets of readings have the same number of observations and start on the same date (10/23/2015), but end on different dates and have different timestamps -- pipeline 2 has readings at the end of every hour through 12/3/2105, while pipeline 1 has readings in the middle of hours, sometimes more than once per hour through 11/1/2015. In order to use the two series together, the readings for pipeline 1 must be converted to hourly:

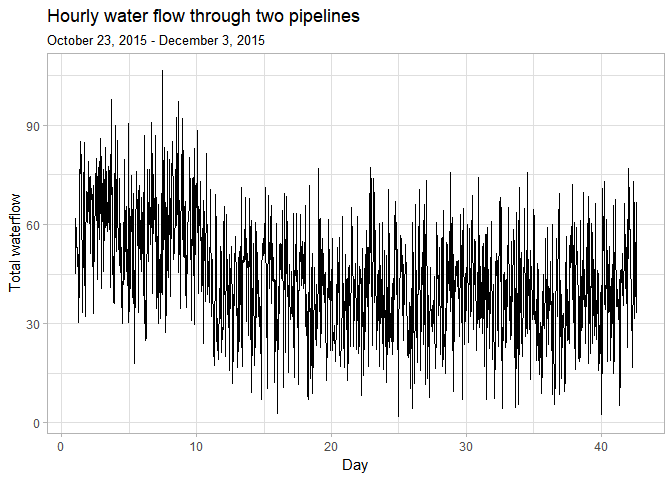
library(lubridate)  
water1 <- water1 %>%   
 # separate date & hour components of readings  
 mutate(Date = date(DateTime),  
 # convert hour to hour-ending to match pipeline 2  
 Hour = hour(DateTime) + 1) %>%   
 # get average reading for each date & hour  
 group\_by(Date, Hour) %>%   
 summarize(WaterFlow = mean(WaterFlow)) %>%   
 # convert back to DateTime and drop separate date/hour columns  
 ungroup() %>%  
 mutate(DateTime = ymd\_h(paste(Date, Hour))) %>%   
 select(DateTime, WaterFlow)

Now it can be seen that there are only observations for pipeline 1 in 236 of the 1000 hours with observations for pipeline 2. The two datasets are joined and a total waterflow is created, then converted to a timeseries:

# create df with both observations for each hour  
water\_df <- full\_join(water1, water2, by = "DateTime", suffix = c("\_1", "\_2")) %>%   
 # convert missing pipeline 1 readings to zero  
 mutate(WaterFlow\_1 = ifelse(is.na(WaterFlow\_1), 0, WaterFlow\_1)) %>%   
 # get total waterflow by hour  
 mutate(WaterFlow = WaterFlow\_1 + WaterFlow\_2) %>%   
 # drop individual numbers  
 select(DateTime, WaterFlow)  
# create hourly timeseries object  
water\_ts <- ts(water\_df$WaterFlow, frequency = 24)

The timeseries is plotted to inspect its features:

autoplot(water\_ts) +  
 labs(title = "Hourly water flow through two pipelines",  
 subtitle = "October 23, 2015 - December 3, 2015",  
 x = "Day", y = "Total waterflow")

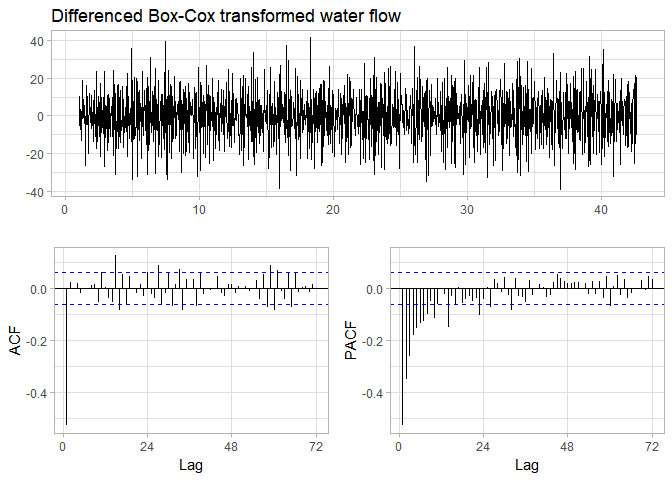


This plot shows a decent amount of variability across the whole range, with an initial downward trend before day 10 followed by a roughly flat period through the end of the time window.

## Fitting

The variance seems roughly constant, but a Box-Cox transformation is performed nonetheless. Due to the apparent non-stationarity, a lag-1 difference is taken:

# get Box-cox paramter & transform  
water\_lambda <- BoxCox.lambda(water\_ts)  
water\_trans <- BoxCox(water\_ts, water\_lambda)  
# plot differenced transformed series  
ggtsdisplay(diff(water\_trans), points = FALSE,  
 main = "Differenced Box-Cox transformed water flow")



This timeseries appears stationary, but shows significant spikes in the ACF and PACF at , strongly suggesting non-seasonal AR(1) and MA(1) components. There is no apparent seasonal behavior. Thus, an ARIMA(1,1,1) model is used:

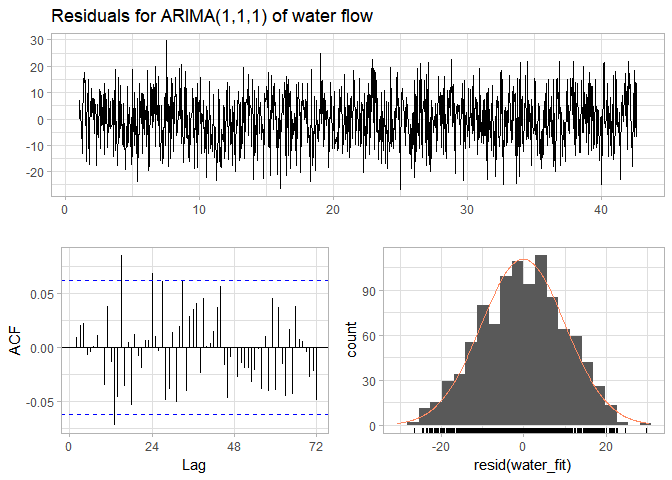
water\_fit <- Arima(water\_ts, order = c(1, 1, 1), lambda = water\_lambda)

The residuals of this fit are investigated:

Box.test(resid(water\_fit), type = "L")

Box-Ljung test  
  
data: resid(water\_fit)  
X-squared = 2.0096e-05, df = 1, p-value = 0.9964

ggtsdisplay(resid(water\_fit), points = FALSE, plot.type = "histogram",  
 main = "Residuals for ARIMA(1,1,1) of water flow")

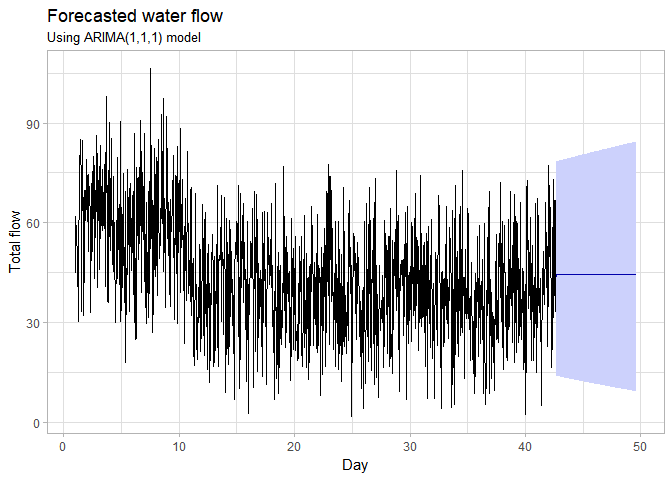


The Ljung-Box test returns a value of almost 1; the residuals appear to be roughly normally distributed around 0 without significant autocorrelation. The model is acceptable and will be used for forecasting.

## Forecasting

Using the ARIMA(1,1,1) model, one week (168 hours) is forecast, and the forecast plotted:

water\_forecast <- forecast(water\_fit, 168, level = 95)  
autoplot(water\_forecast) +   
 labs(title = "Forecasted water flow",  
 subtitle = "Using ARIMA(1,1,1) model", x = "Day", y = "Total flow") +  
 theme(legend.position = "none")



Due to the near-constant mean of the end of the data and lack of seasonality, a single value is forecast for the entire window. The forecast values are output to a .csv file:

data\_frame(DateTime = max(water\_df$DateTime) + hours(1:168),  
 WaterFlow = water\_forecast$mean) %>%   
 write\_csv("project1\_water.csv")