403BfinalprojectUber

Yuhe Tian

3/7/2019

I. Introduction

dependent variable ???mean travel time range=upper-lower/average independent variable???weather / temperature/ holiday/ weekday

Data clean

```
## import data
rm(list=ls(all=TRUE))
setwd("/Users/hongyiwang/Desktop/winter/403B/final project")
ub=read.csv("uberdata.csv",header = T)
temp = read.csv("tempre.csv",header = T)
```

II. Data Exploratory

II.1 Data Quality

```
### find missing value
sum(is.na(ub))
```

```
## [1] 39
```

```
sum(is.na(temp$Precip))
```

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

There are 39 NAs in total in dataset "ub", 0 NAs in dataset "temp".

subsitute NAs, by: if NA, then NA value = average of the the the day before and the day after eg. day 10th is NA, X10 = (X9 + X11)/2 both ub[285,] ub[286,] are NA, so we define ub[285,] = ub[284,],ub[286,] = ub[287]

```
  ### define NAs in ub
index1 = which(is.na(ub$mean),arr.ind = T)
for (i in index1){
  ub$lower[i] = (ub$lower[i-1] + ub$lower[i+1])/2
  ub$upper[i] = (ub$upper[i-1] + ub$upper[i+1])/2
  ub$mean[i] = (ub$upper[i] + ub$lower[i] )/2
}
ub[285,2:4] = ub[284,2:4]
ub[286,2:4] = ub[287,2:4]

  ### define NAs in precip
precip = temp$Precip[1:365]
index2 = levels(precip)
for (i in 1:length(precip)){
  if (precip[i] == 'T') {
```

```
    precip[i] = 0
  } else if (precip[i] == ''){
    precip[i] = NA
  } else {
    for (j in index2[2:22]) {
      if (precip[i] == j) {
        precip[i] = as.numeric(j)
      }
    }
  }
}
index3 = which(is.na(precip))
 ## all NAs define as 0
precip[index3] = 0
```

Define dataset:

```
travel = ub$mean
travelrange = ub$upper - ub$lower
holiday = ub$holiady
  ## we will not use temp so I commented these two lines
#meantemp = (as.numeric(temp$High) + as.numeric(temp$low) )/2
#temprange = as.numeric(temp$High) - as.numeric(temp$low)
  # create dummy variable rainfall
rainfall = rep(1,365)
rainfall[precip == 0] = 0
```

II.2 Dummy Variables Creatation

```
weekdayloop = rep(1:7,53)
weekdayloop = weekdayloop[1:365]

Sun = rep(0,365)
Sun[weekdayloop == 1] = 1
Mon = rep(0,365)
Mon[weekdayloop == 2] = 1
Tue = rep(0,365)
Tue[weekdayloop == 3] = 1
Wed = rep(0,365)
Wed[weekdayloop == 4] = 1
Thur = rep(0,365)
Thur[weekdayloop == 5] = 1
Fri = rep(0,365)
Fri[weekdayloop == 6] = 1
Sat = rep(0,365)
Sat[weekdayloop == 7] = 1
```

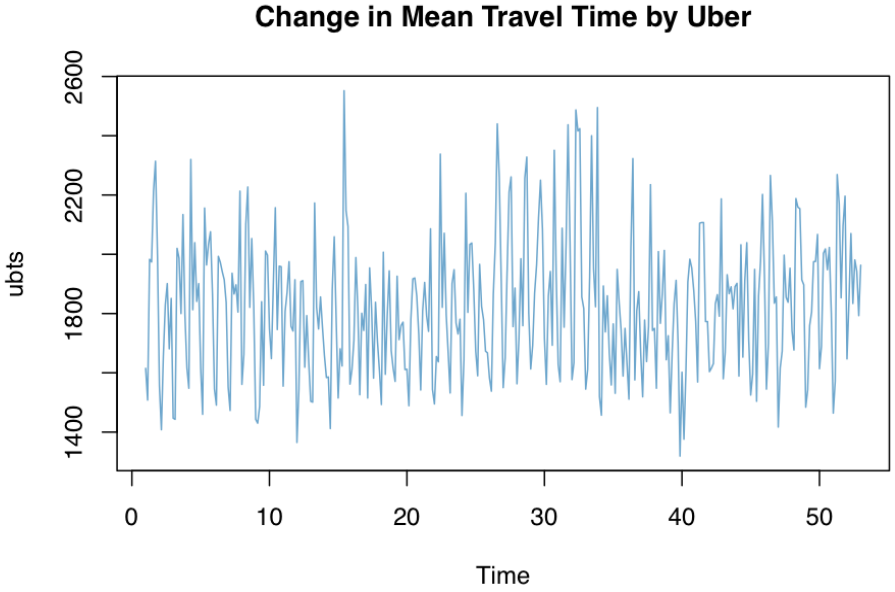
```
  #combine dataset
ubdata = data.frame(travel,travelrange,holiday,rainfall,Mon,Tue,Wed,Thur,Fri,Sat,Sun)
```

II.3 Time Series Classfication, Plot and Observation

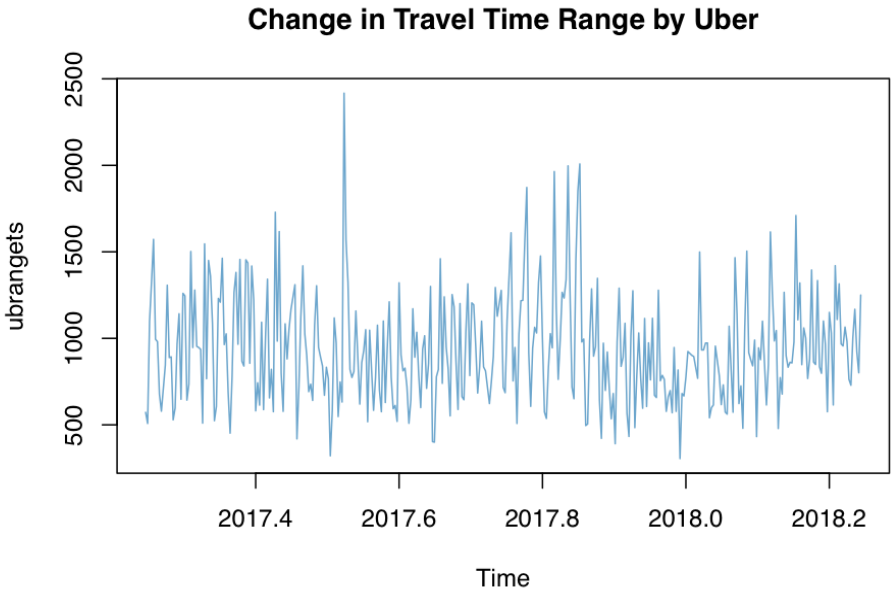
```
inds <- seq(as.Date("2017-04-01"), as.Date("2018-03-01"), by = "day")

#myts <- ts(rnorm(length(inds)),      # random data
#           start = c(2014, as.numeric(format(inds[1], "%j"))),
#           frequency = 365)
#ubts =ts(ubdata$travel,start=c(2017,as.numeric(format(inds[1],"%j"))),freq=365)
ubts =ts(ubdata$travel, start = c(1, 1), freq = 7)
ubrangets =ts(ubdata$travelrange,start=c(2017,as.numeric(format(inds[1],"%j"))),freq=365)
```

(a) Time Series Plot

```
plot(ubts,col='skyblue3',main="Change in Mean Travel Time by Uber")
```




```
plot(ubrangets,col='skyblue3',main="Change in Travel Time Range by Uber")
```



(b) Covariance Stationary

```
library(tseries)
adf.test(ubts)
```

```
## Warning in adf.test(ubts): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ubts
## Dickey-Fuller = -5.5863, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
adf.test(ubrangets)
```

```
## Warning in adf.test(ubrangets): p-value smaller than printed p-value
```

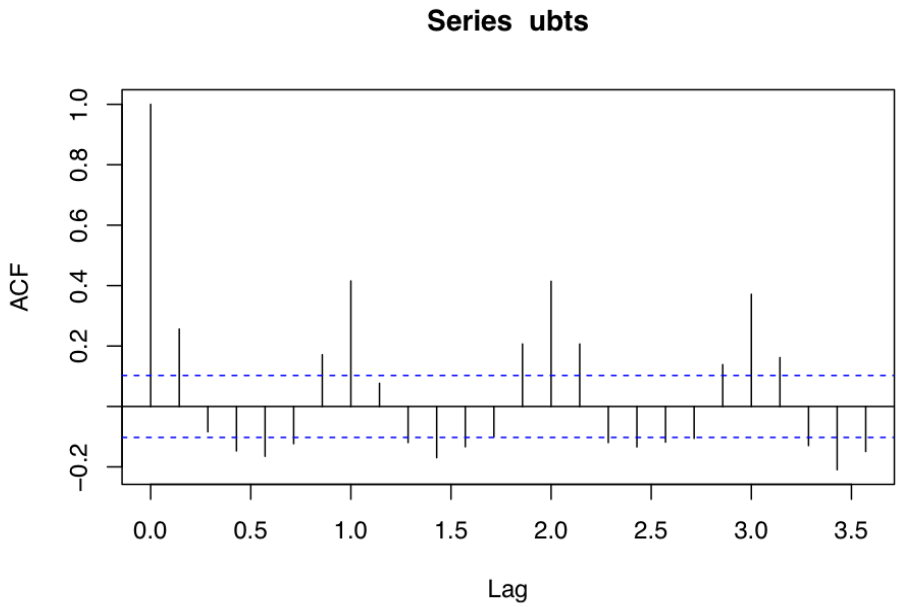
```
##
##  Augmented Dickey-Fuller Test
##
## data:  ubrangets
## Dickey-Fuller = -4.5237, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
## stationary
```

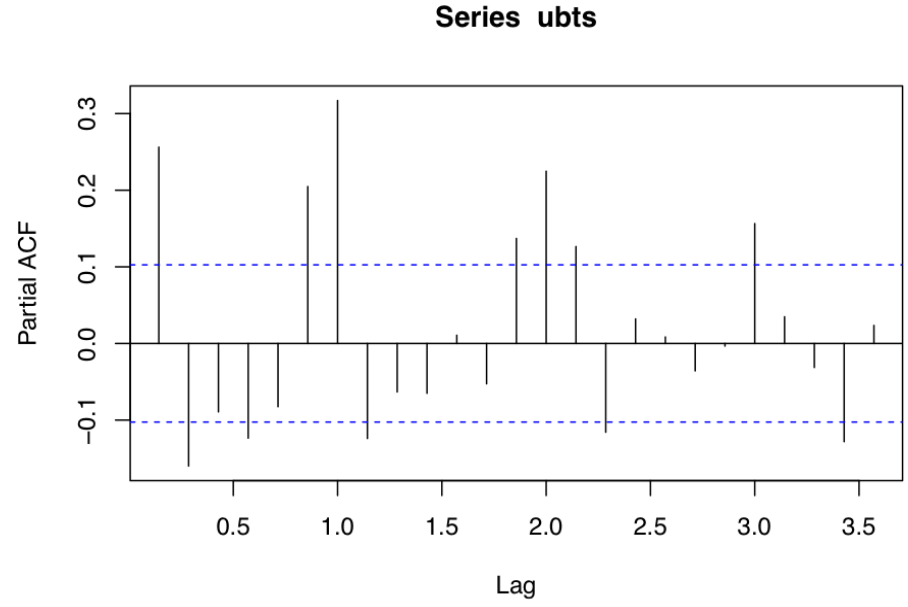
According to the results, since the p-value is smaller than 0.05, we could conclude that our time series is stationery. There is no need for transforming.

(c) Plot and Discuss the ACF and PACF

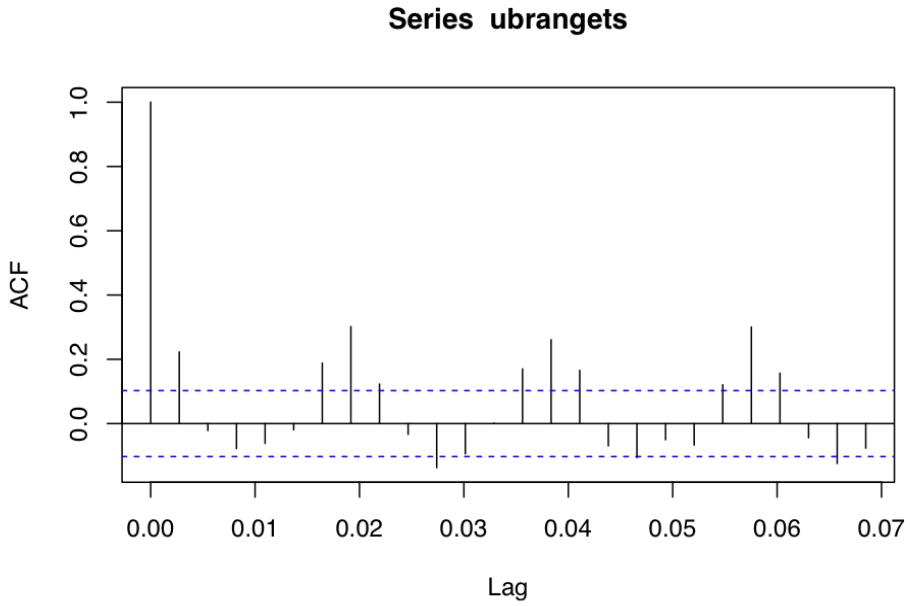
```
acf(ubts)
```



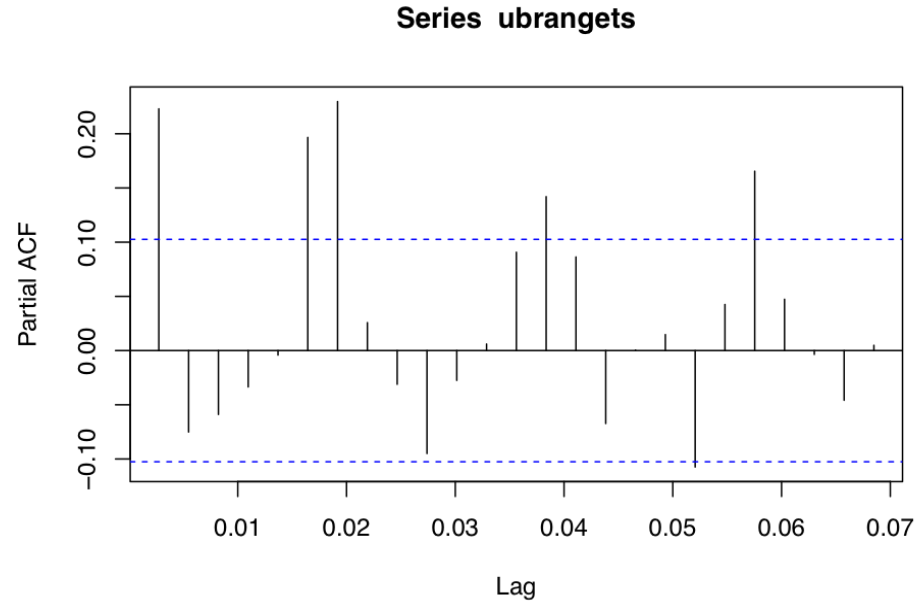

```
pacf(ubts)
```



```
acf(ubrangets)
```




```
pacf(ubrangets)
```



From the graph, we could notice that there exists a circle of seven days, which is the same as days in a week.

(d) 1st difference

```
ubts.1 <- diff(ubts, 1)
ubts.1.7 <- diff(ubts.1, 7)
summary(ur.df(ubts.1.7, type = "none"))
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1035.59  -173.39    -6.52   167.06  1141.80
##
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)
## z.lag.1    -1.81086    0.08570 -21.130  < 2e-16 ***
## z.diff.lag  0.29152    0.05127   5.686 2.73e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 277.7 on 353 degrees of freedom
## Multiple R-squared:  0.7237, Adjusted R-squared:  0.7221
## F-statistic: 462.2 on 2 and 353 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -21.1302
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau1 -2.58 -1.95 -1.62
```

III linear regression

(a) pure season dummy

```
m2 =lm(ubts ~ Mon+Tue+Sat+Sun+Thur+Wed)
m3 =lm(ubts ~ Mon+Tue+Sat+Sun+Thur+Fri)
m4 =lm(ubts ~ Mon+Tue+Sat+Fri+Thur+Wed)
m5 =lm(ubts ~ Sun+Tue+Sat+Fri+Thur+Wed)
m6 =lm(ubts ~ Mon+Sun+Sat+Fri+Thur+Wed)
## comparisons
AIC(m2,m3,m4,m5,m6)
```

```
##    df      AIC
```

```
## m2  8 4854.371
## m3  8 4854.371
## m4  8 4854.371
## m5  8 4854.371
## m6  8 4854.371
```

```
summary(m6)
```

```
##
## Call:
## lm(formula = ubts ~ Mon + Sun + Sat + Fri + Thur + Wed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -528.64 -110.76  -10.08   84.31  647.36
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1927.779     25.611  75.271   <2e-16 ***
## Mon         -349.087     36.220  -9.638   <2e-16 ***
## Sun         -313.694     36.048  -8.702   <2e-16 ***
## Sat          -80.139     36.220  -2.213   0.0276 *
## Fri           18.303     36.220   0.505   0.6136
## Thur         -19.558     36.220  -0.540   0.5896
## Wed            6.981     36.220   0.193   0.8473
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 184.7 on 358 degrees of freedom
## Multiple R-squared:  0.3908, Adjusted R-squared:  0.3806
## F-statistic: 38.28 on 6 and 358 DF,  p-value: < 2.2e-16
```

(b) add holiday and rainfall

```
m7=lm(ubts ~ Mon+Tue+Sat+Sun+Thur+Wed+Fri+holiday)
summary(m7)
```

```
##
## Call:
## lm(formula = ubts ~ Mon + Tue + Sat + Sun + Thur + Wed + Fri +
##     holiday)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -528.64 -111.24  -14.24   84.13  647.36
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1946.082     25.400  76.616  < 2e-16 ***
## Mon         -358.321     36.085  -9.930  < 2e-16 ***
## Tue           -6.212     36.212  -0.172  0.86390
## Sat          -98.442     35.921  -2.740  0.00644 **
## Sun         -329.031     35.769  -9.199  < 2e-16 ***
## Thur         -34.838     35.940  -0.969  0.33303
## Wed           -8.299     35.940  -0.231  0.81751
```

```
## Fri               NA         NA      NA       NA
## holiday     -157.184     59.546  -2.640  0.00866 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 183.2 on 357 degrees of freedom
## Multiple R-squared:  0.4025, Adjusted R-squared:  0.3908
## F-statistic: 34.35 on 7 and 357 DF,  p-value: < 2.2e-16
```

```
m8=lm(ubts~Mon+Tue+Sat+Sun+Thur+Wed+Fri+rainfall)
summary(m8)
```

```
##
## Call:
## lm(formula = ubts ~ Mon + Tue + Sat + Sun + Thur + Wed + Fri +
##     rainfall)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -526.86 -109.57   -9.75   86.09  649.14
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1943.110     25.869  75.112  < 2e-16 ***
## Mon         -366.201     36.263 -10.098  < 2e-16 ***
## Tue          -18.897     36.242  -0.521  0.60240
## Sat          -97.254     36.263  -2.682  0.00766 **
## Sun         -331.358     36.072  -9.186  < 2e-16 ***
## Thur         -37.266     36.242  -1.028  0.30452
## Wed           -9.539     36.298  -0.263  0.79285
## Fri               NA         NA      NA       NA
## rainfall      30.901     37.106   0.833  0.40552
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 184.8 on 357 degrees of freedom
## Multiple R-squared:  0.392,  Adjusted R-squared:  0.3801
## F-statistic: 32.88 on 7 and 357 DF,  p-value: < 2.2e-16
```

IV Model

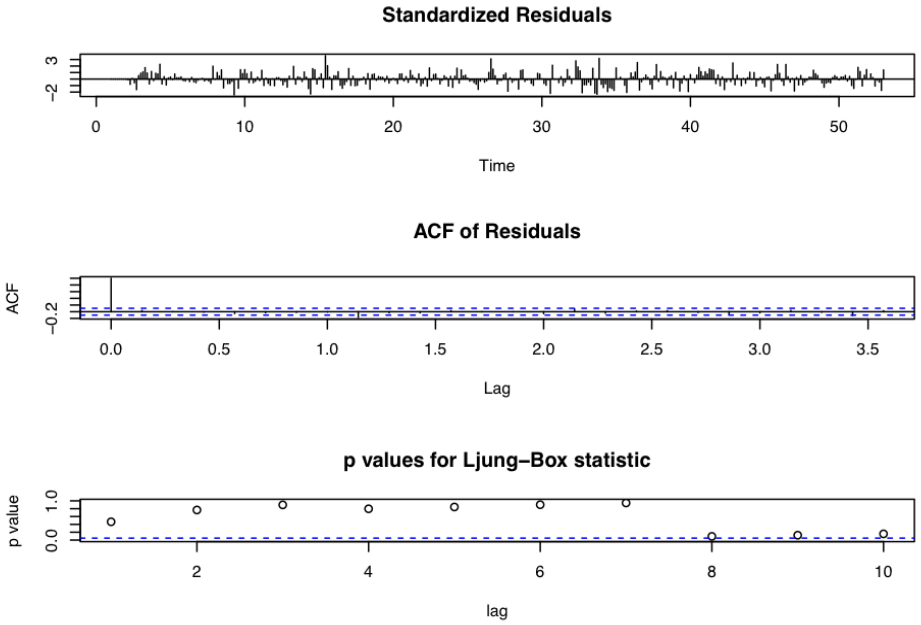
(a) arima

```
## arma(6,1)+seasonality
fit.a.1 <- arima(ubts, order = c(6, 1, 0), seasonal = list(order = c(7, 1, 0)))
confint(fit.a.1)
```

```
##           2.5 %      97.5 %
## ar1  -0.9448685 -0.73145401
## ar2  -0.8831545 -0.59873374
## ar3  -0.8422949 -0.51000909
## ar4  -0.7712368 -0.43776246
## ar5  -0.6795740 -0.33266517
## ar6  -0.6257342 -0.26331815
## sar1 -1.4274840 -1.04307564
```

```
## sar2 -1.4596211 -0.86244384
## sar3 -1.3179197 -0.67176613
## sar4 -1.1358802 -0.55443401
## sar5 -0.8762835 -0.38977152
## sar6 -0.6320074 -0.26717570
## sar7 -0.1998667  0.03526851
```

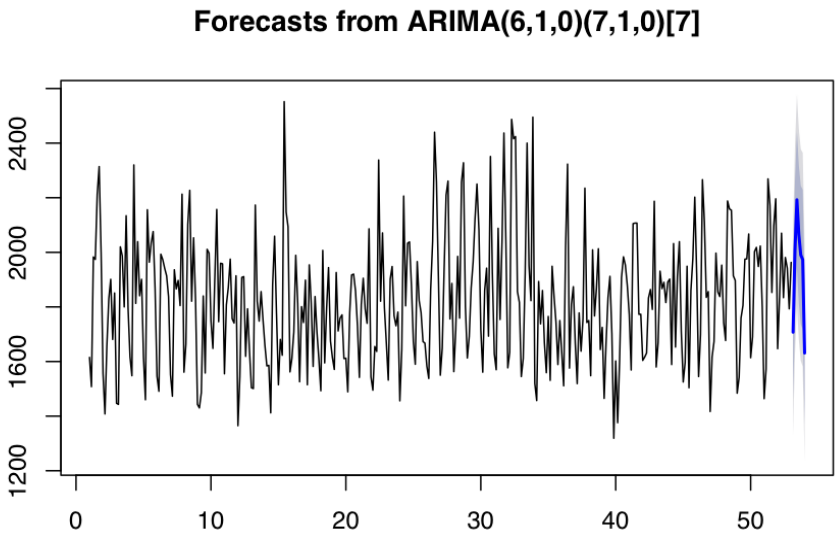
```
tsdiag(fit.a.1)
```




```
Box.test(fit.a.1$residuals, lag = 14, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  fit.a.1$residuals
## X-squared = 16.016, df = 14, p-value = 0.3124
```

```
plot(forecast(fit.a.1, h = 7))
```




```
summary(fit.a.1, h = 7)
```

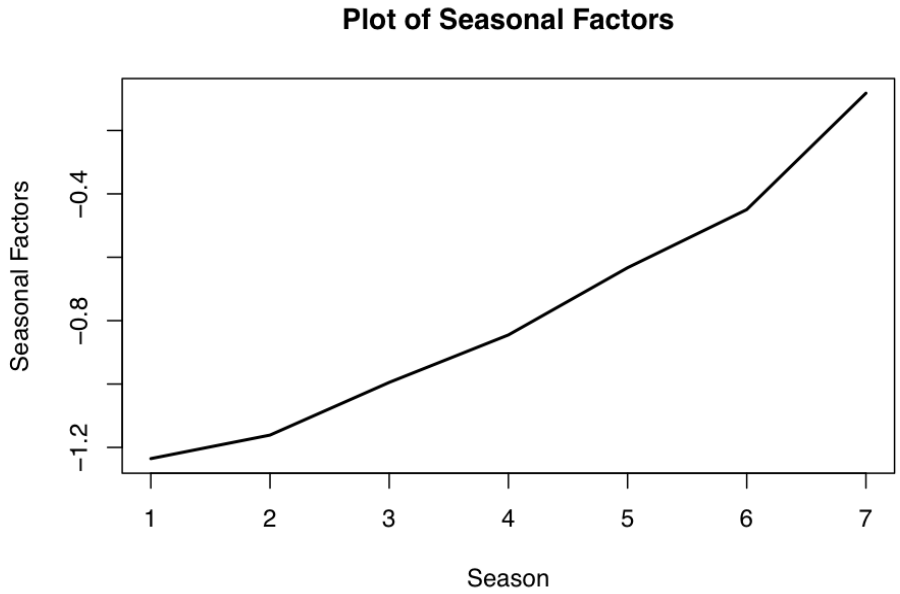
```
##
## Call:
## arima(x = ubts, order = c(6, 1, 0), seasonal = list(order = c(7, 1, 0)))
##
## Coefficients:
##           ar1      ar2      ar3      ar4      ar5      ar6     sar1
##       -0.8382  -0.7409  -0.6762  -0.6045  -0.5061  -0.4445  -1.2353
## s.e.   0.0544   0.0726   0.0848   0.0851   0.0885   0.0925   0.0981
##          sar2     sar3     sar4     sar5     sar6     sar7
##       -1.1610  -0.9948  -0.8452  -0.6330  -0.4496  -0.0823
## s.e.   0.1523   0.1648   0.1483   0.1241   0.0931   0.0600
##
## sigma^2 estimated as 36707:  log likelihood = -2389.09,  aic = 4806.18
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 4.498029 189.4795 144.6594 -0.5071734 7.883135 0.6466652
##                     ACF1
## Training set 0.03801147
```

```
# plot the estimated seasonal factors and interpret your plot

#install.packages("lmtest")
#coeftest(fit.a.1)

fit.a.1$coef
```

```
##         ar1         ar2         ar3         ar4         ar5         ar6
## -0.83816126 -0.74094412 -0.67615200 -0.60449963 -0.50611959 -0.44452618
##        sar1        sar2        sar3        sar4        sar5        sar6
## -1.23527984 -1.16103245 -0.99484291 -0.84515711 -0.63302750 -0.44959156
```

```
##        sar7
## -0.08229908
```

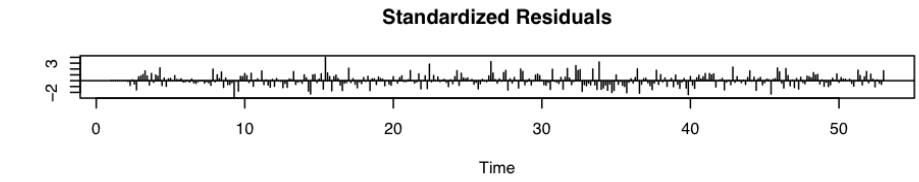
```
plot(fit.a.1$coef[7:13],type='l',ylab='Seasonal Factors',xlab="Season",lwd=2, main="Plot of Seasonal Fa
```

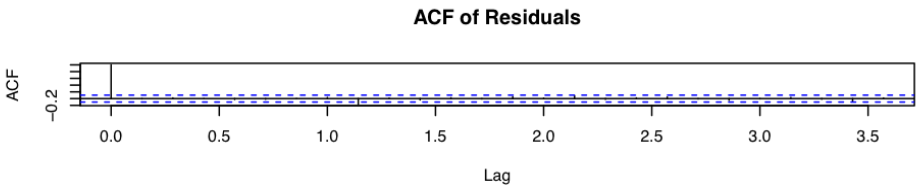


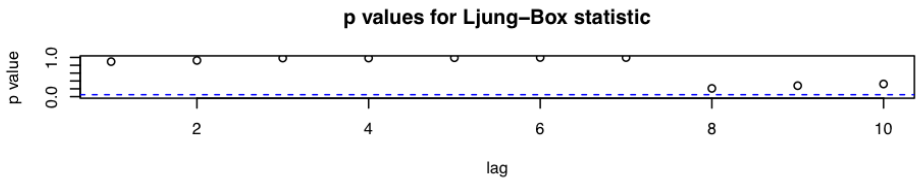

```
fit.b.1 <- arima(ubts, order = c(1, 1, 1), seasonal = list(order = c(0, 1, 1)))
confint(fit.b.1)
```

```
##              2.5 %     97.5 %
## ar1    0.009028253  0.2285816
## ma1   -0.984889362 -0.9080785
## sma1 -1.073345165 -0.9266372
```

```
tsdiag(fit.b.1)
```



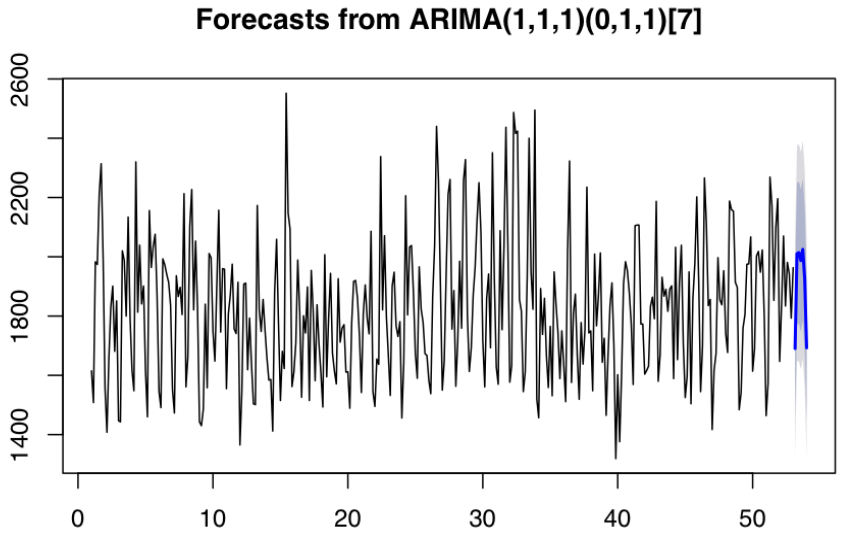





```
Box.test(fit.b.1$residuals, lag = 14, type = "Ljung-Box")
```

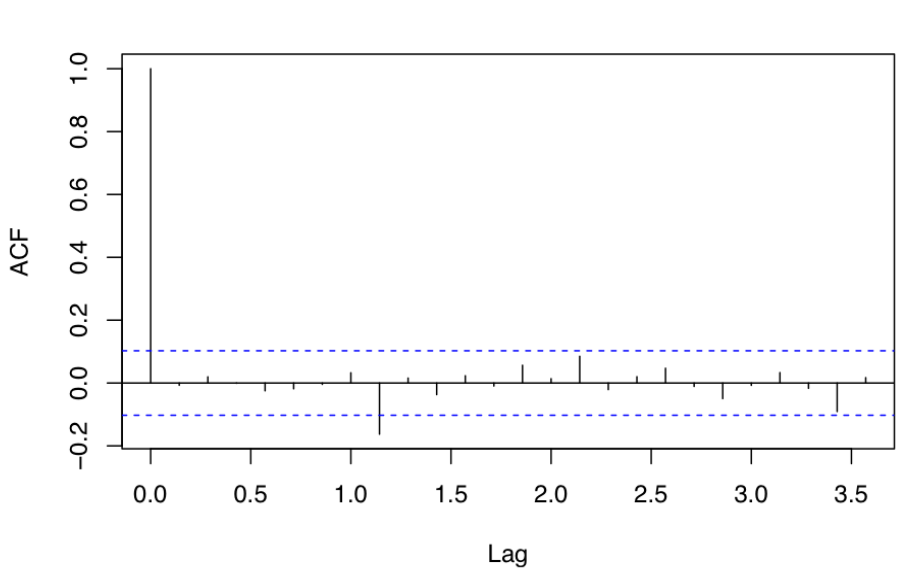
```
##
##  Box-Ljung test
##
## data:  fit.b.1$residuals
## X-squared = 12.973, df = 14, p-value = 0.5287
```

```
plot(forecast(fit.b.1, h = 7))
```

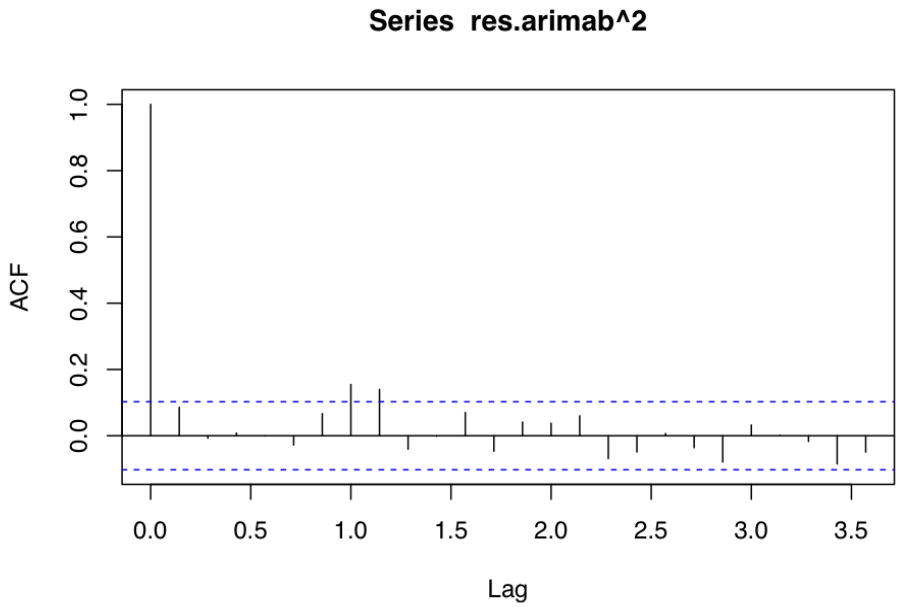


(b) GARCH

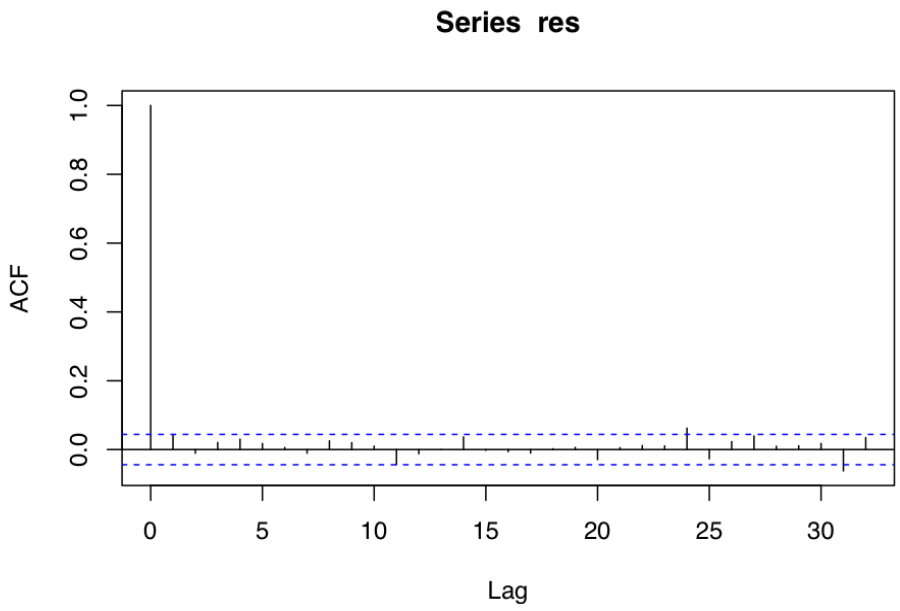
```
## ARCH(1)
res.arimab=fit.b.1$res
acf(res.arimab)
```



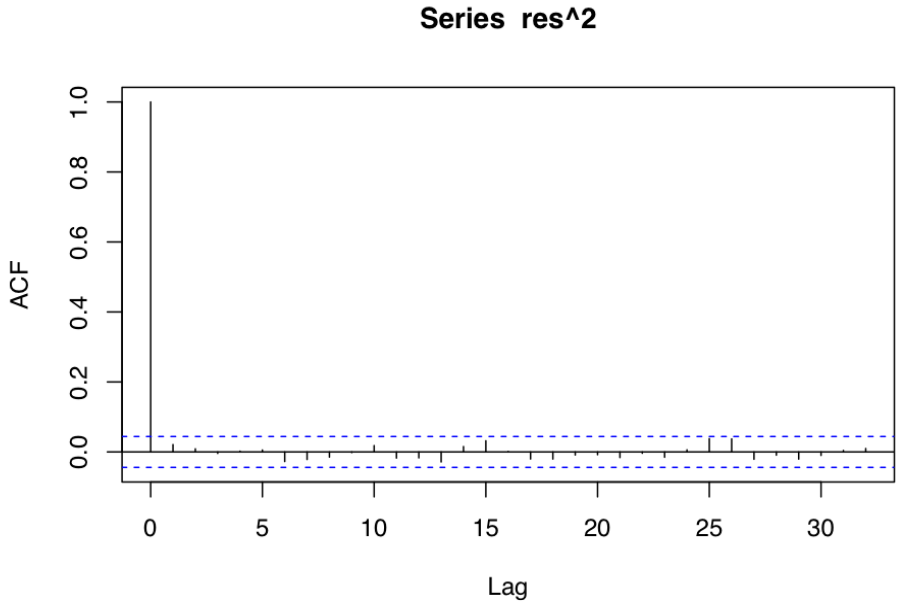
```
acf(res.arimab^2)
```




```
squared=res.arimab^2
m_g<-garchFit(res.arimab~garch(1,3),trace = FALSE)
res<-m_g@residuals/m_g@sigma.t
acf(res)
```




```
acf(res^2)
```



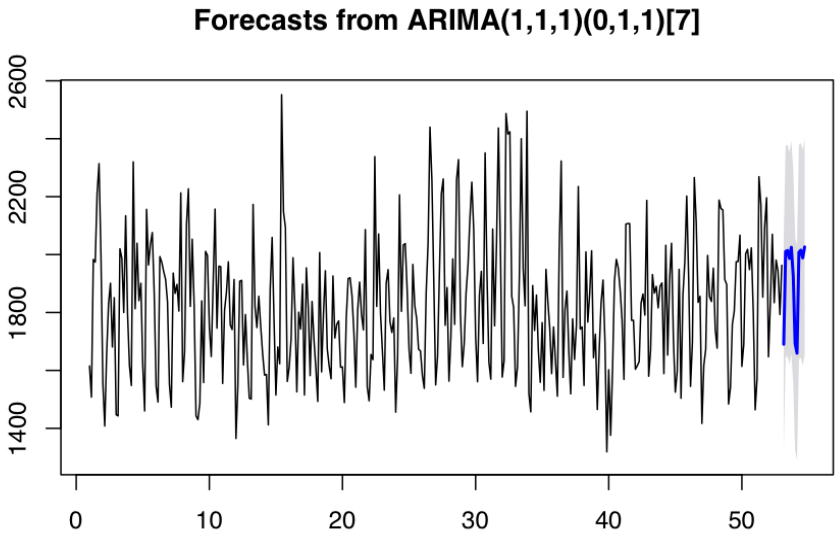

```
summary(m_g)
```

```
##
```

```
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = res.arimab ~ garch(1, 3), trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ garch(1, 3)
## <environment: 0x7fcc73305510>
##  [data = fGarch::dem2gbp]
##
## Conditional Distribution:
##  norm
##
## Coefficient(s):
##          mu        omega       alpha1        beta1        beta2
## -0.00376519   0.01127952   0.19055243   0.38822250   0.00000001
##       beta3
##  0.37653535
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##          Estimate  Std. Error  t value Pr(>|t|)
## mu     -3.765e-03   8.482e-03   -0.444 0.657123
## omega   1.128e-02   2.826e-03    3.991 6.58e-05 ***
## alpha1  1.906e-01   2.791e-02    6.828 8.63e-12 ***
## beta1   3.882e-01   1.038e-01    3.741 0.000184 ***
## beta2   1.000e-08   1.723e-01    0.000 1.000000
## beta3   3.765e-01   1.279e-01    2.944 0.003240 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  -1099.094    normalized:  -0.5567853
##
## Description:
##  Fri Mar 22 14:21:20 2019 by user:
##
##
## Standardised Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  999.1862  0
##  Shapiro-Wilk Test  R    W      0.9632659 0
##  Ljung-Box Test     R    Q(10)  9.320895  0.5019442
##  Ljung-Box Test     R    Q(15)  16.09236  0.37594
##  Ljung-Box Test     R    Q(20)  18.16239  0.5767117
##  Ljung-Box Test     R^2  Q(10)  4.600815  0.9162016
##  Ljung-Box Test     R^2  Q(15)  9.874221  0.8275762
##  Ljung-Box Test     R^2  Q(20)  11.76969  0.9237748
##  LM Arch Test       R    TR^2   5.615319  0.9342197
##
## Information Criterion Statistics:
```

```
##      AIC      BIC      SIC     HQIC
## 1.119650 1.136634 1.119631 1.125890
```

```
forecast212=forecast(fit.b.1,12,level=95)
plot(forecast212)
```




```
mgg<-garch(res.arimab,order=c(1,3),trace=FALSE)
```

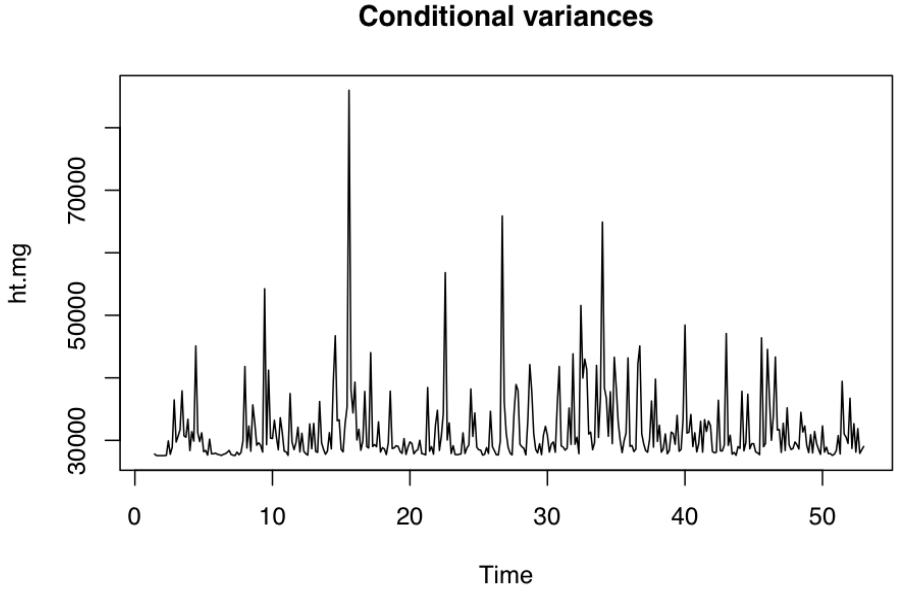
```
## Warning in garch(res.arimab, order = c(1, 3), trace = FALSE): singular
## information
```

```
summary(mgg)
```

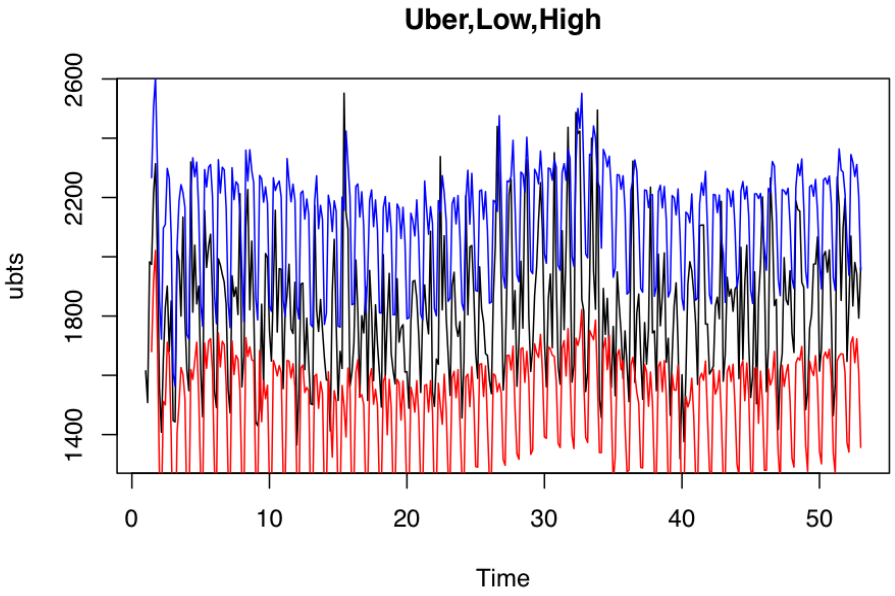
```
##
## Call:
## garch(x = res.arimab, order = c(1, 3), trace = FALSE)
##
## Model:
## GARCH(1,3)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -2.940830 -0.617130 -0.007312  0.610993  3.865123
##
## Coefficient(s):
##     Estimate  Std. Error  t value Pr(>|t|)
## a0 2.578e+04          NA       NA       NA
## a1 1.091e-01          NA       NA       NA
## a2 4.466e-10          NA       NA       NA
## a3 8.400e-03          NA       NA       NA
## b1 6.430e-02          NA       NA       NA
##
## Diagnostic Tests:
##  Jarque Bera Test
```

```
##
## data:  Residuals
## X-squared = 15, df = 2, p-value = 0.0005531
##
##
##  Box-Ljung test
##
## data:  Squared.Residuals
## X-squared = 0.0011018, df = 1, p-value = 0.9735
```

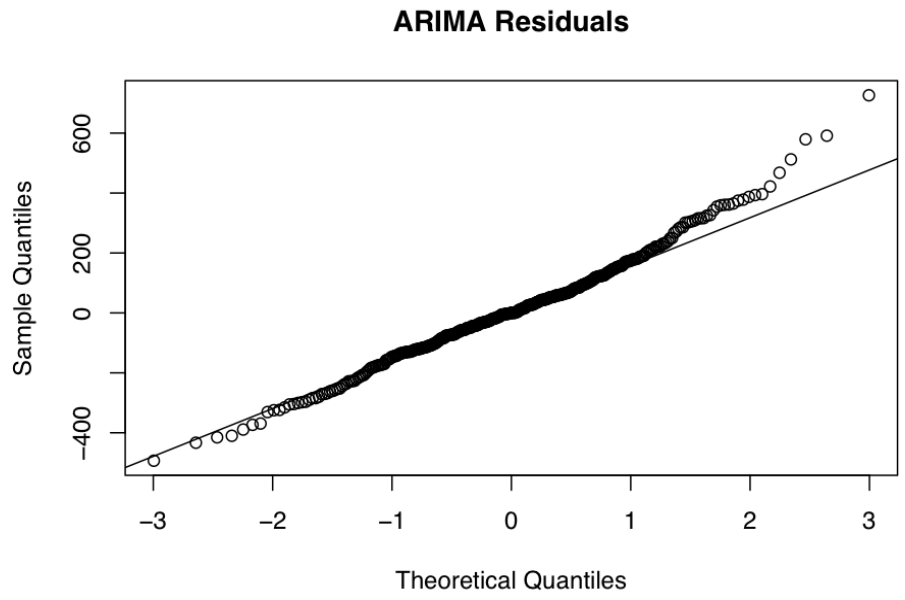
```
ht.mg =mgg$fit[,1]^2 #use 1st column of fit plot(ht.arch08,main='Conditional variances')
plot(ht.mg,main='Conditional variances')
```




```
res212<-mgg$residuals
fit212=fitted.values(fit.b.1)
low=fit212-1.76*sqrt(ht.mg)
high=fit212+1.76*sqrt(ht.mg)
plot(ubts,type='l',main='Uber,Low,High')
lines(low,col='red')
lines(high,col='blue')
```

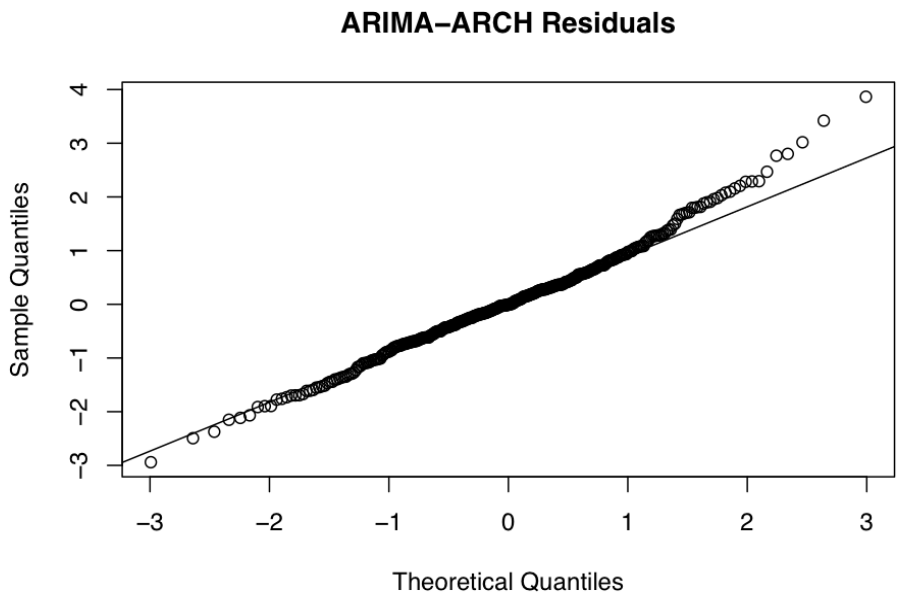



```
## normal ditribution
qqnorm(fit.b.1$residuals,main='ARIMA Residuals')
qqline(fit.b.1$residuals)
```



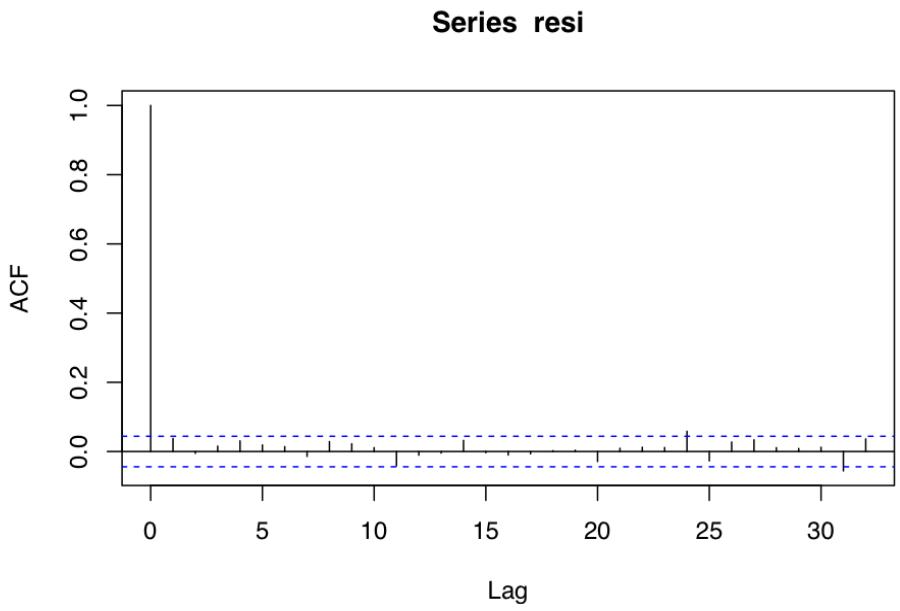

```
archres=res212
qqnorm(archres,main='ARIMA-ARCH Residuals')
```

```
qqline(archres)
```

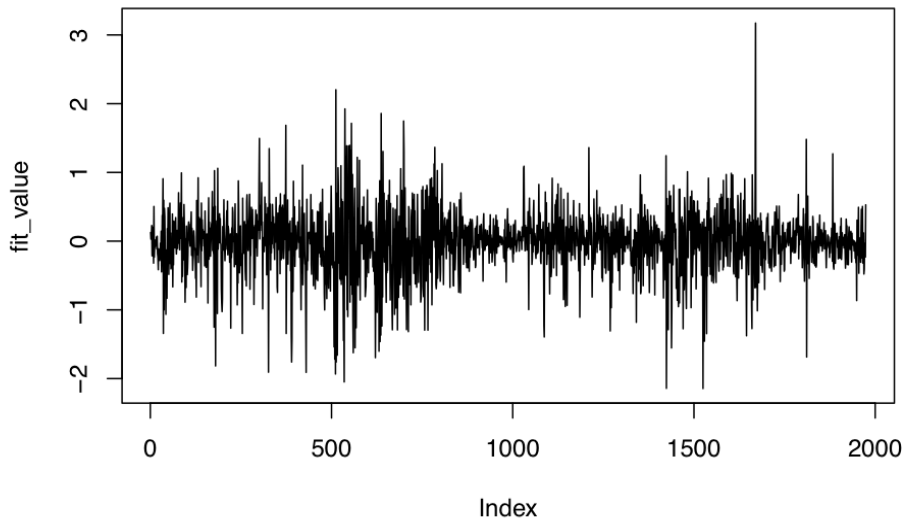


(c) Garch + arima

```
mag<-garchFit(formula =ubts.1~arma(1,1,1)+res.arimab~garch(1,3),cond.dist ="std",trace=FALSE)
resi=mag@residuals/mag@sigma.t   ###Standardised Residuals
acf(resi)
```




```
fit_value<-fitted(mag)
plot(fit_value,type="l")
```




```
AIC(mgg,fit.b.1)
```

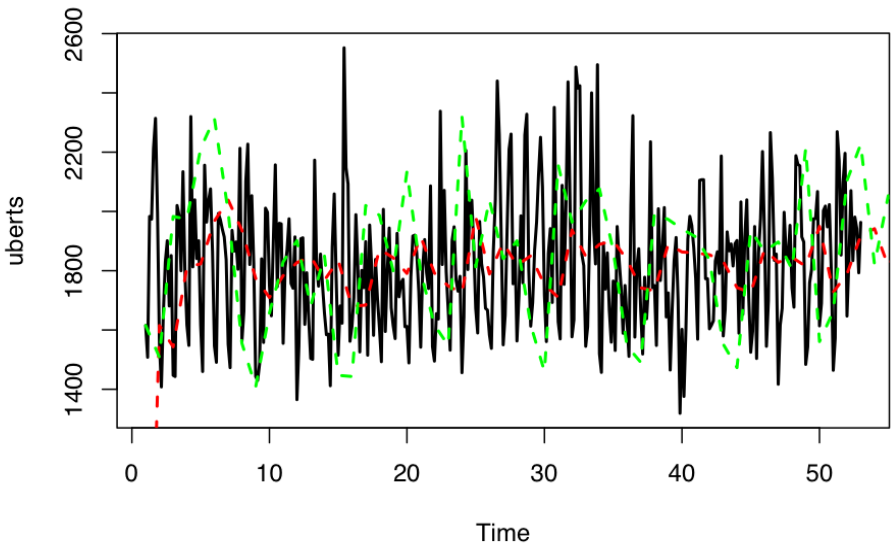
```
##          df      AIC
## mgg       5 4793.872
## fit.b.1   4 4766.726
```

```r
BIC(mgg,fit.b.1)
```

```
##         df      BIC
## mgg      5       NA
## fit.b.1  4 4782.236
```

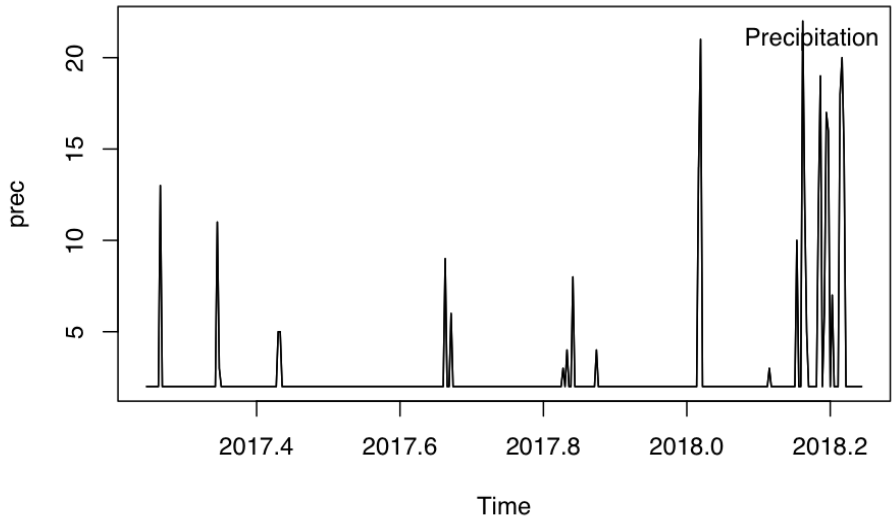
(d) Kalman kilter

```r
library(TSPred)
library(KFAS)


fitkf<-fittestArimaKF(ubts, timeseries.test=NULL, h=12, na.action=na.omit,
level=0.9, filtered = TRUE, initQ=NULL,rank.by="AIC")

pred <- fitkf$pred

#extracting Kalman filtered and smoothed time series from the best fitted model
fs <- KFAS::KFS(fitkf$model,filtering=c("state","mean"),smoothing=c("state","mean"))
f <- fitted(fs, filtered = TRUE) #Kalman filtered time  series
s <- fitted(fs) #Kalman smoothed time  series
#plotting the time series data
plot(ubts,type='l',lwd=2,xlab="Time",ylab="uberts")
#plotting the Kalman filtered time series
lines(f,col='red',lty=2,lwd=2)
#plotting the Kalman smoothed time series
lines(s,col='green',lty=2,lwd=2)
#plotting predicted values
lines(ts(pred$mean,start = 2017.4),lwd=2,col='blue')
#plotting prediction intervals
lines(ts(pred$upper,start = 2017.4),lwd=2,col='light blue')
lines(ts(pred$lower,start = 2017.4),lwd=2,col='light blue')
```

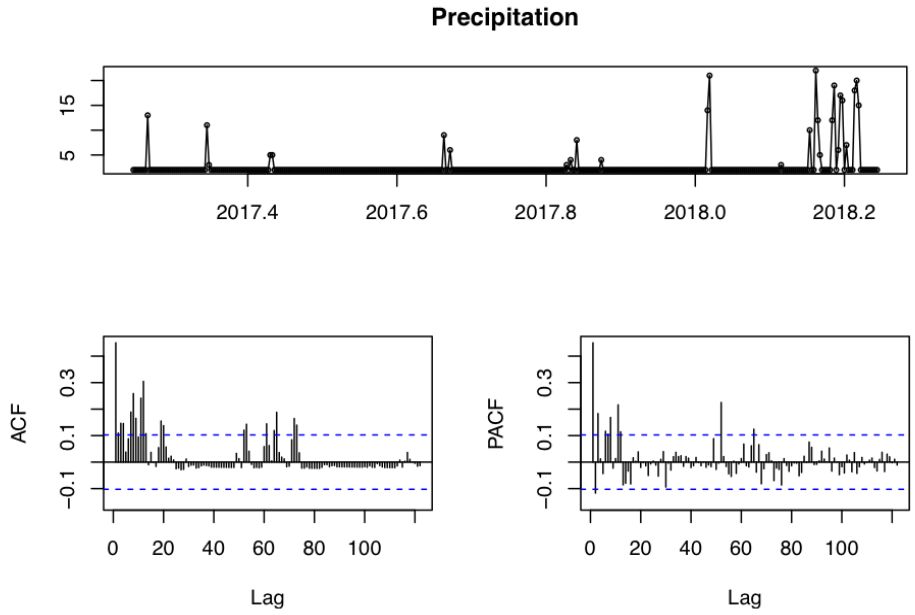


VAR

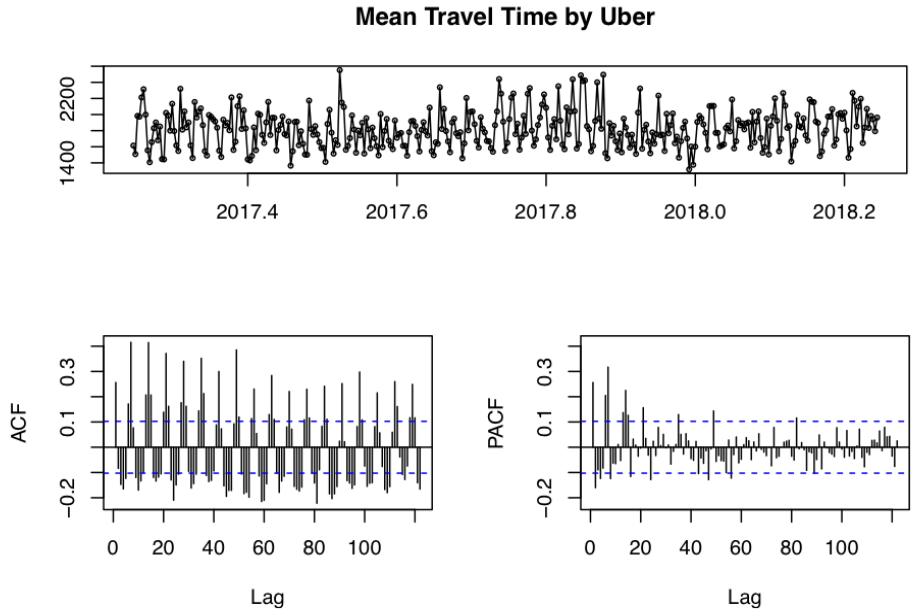
```
# Look at the data
prec =ts(precip,start=c(2017,as.numeric(format(inds[1],"%j"))),freq=365 )
ubts =ts(ubdata$travel,start=c(2017,as.numeric(format(inds[1],"%j"))),freq=365)

plot(prec)
nberShade()
lines(prec,ylab="Precipitation and Mean Travel Time by Uber")
legend("topright",legend=c("Precipitation"),text.col=c("black"),bty="n")
```



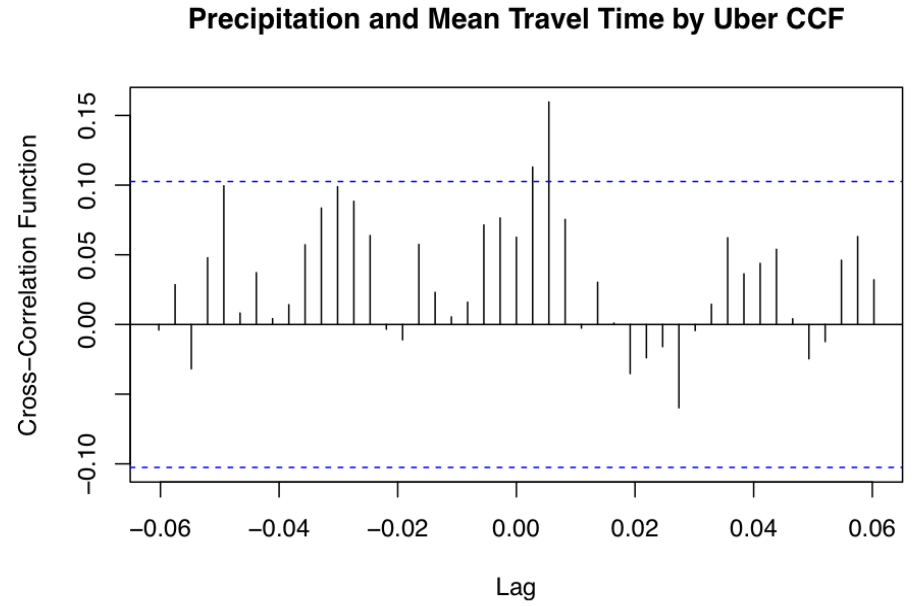

```
# Look at the ACF, PACF, and CCF (cros-correlation function)
tsdisplay(prec,main="Precipitation")
```




```
tsdisplay(ubts,main="Mean Travel Time by Uber")
```




```
ccf(prec,ubts,ylab="Cross-Correlation Function", main = "Precipitation and Mean Travel Time by Uber CCF
```




```
# Fit a VAR(p) model to the data
# Combine the variables into 1 data frame first:
```

```r
y=cbind(prec, ubts)
#y_ts=ts.union(starts, comps) # You can also use this function
y_tot=data.frame(y)
```

```r
# To fit a VAR(p) model, simply call 'VAR' and set p=value
y_tot<-y_tot[1:365,]
y_model=VAR(y_tot,p=4)
summary(y_model)
```

```
##
## VAR Estimation Results:
## ========================
## Endogenous variables: prec, ubts
## Deterministic variables: const
## Sample size: 361
## Log Likelihood: -3291.576
## Roots of the characteristic polynomial:
## 0.6914 0.6914 0.6534 0.5413 0.5413 0.5219 0.5219 0.05785
## Call:
## VAR(y = y_tot, p = 4)
##
##
## Estimation results for equation prec:
## =====================================
## prec = prec.l1 + ubts.l1 + prec.l2 + ubts.l2 + prec.l3 + ubts.l3 + prec.l4 + ubts.l4 + const
##
##           Estimate Std. Error t value Pr(>|t|)
## prec.l1  0.5110361  0.0533216   9.584  < 2e-16 ***
## ubts.l1  0.0007117  0.0005850   1.217 0.224569
## prec.l2 -0.2050054  0.0592538  -3.460 0.000607 ***
## ubts.l2  0.0010439  0.0006039   1.729 0.084745 .
## prec.l3  0.1682997  0.0589933   2.853 0.004589 **
## ubts.l3 -0.0001164  0.0006050  -0.192 0.847495
## prec.l4  0.0116481  0.0528936   0.220 0.825829
## ubts.l4 -0.0001154  0.0005877  -0.196 0.844420
## const   -1.4166064  1.8763355  -0.755 0.450762
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 2.462 on 352 degrees of freedom
## Multiple R-Squared: 0.2537,  Adjusted R-squared: 0.2368
## F-statistic: 14.96 on 8 and 352 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation ubts:
## =====================================
## ubts = prec.l1 + ubts.l1 + prec.l2 + ubts.l2 + prec.l3 + ubts.l3 + prec.l4 + ubts.l4 + const
##
##           Estimate Std. Error t value Pr(>|t|)
## prec.l1    5.73549    4.81925   1.190  0.23480
## ubts.l1    0.26154    0.05287   4.946 1.17e-06 ***
## prec.l2    5.50192    5.35541   1.027  0.30496
## ubts.l2   -0.15367    0.05458  -2.816  0.00514 **
```

```
## prec.l3   -1.26871    5.33186  -0.238  0.81206
## ubts.l3   -0.06884    0.05468  -1.259  0.20892
## prec.l4    1.32522    4.78056   0.277  0.78178
## ubts.l4   -0.13696    0.05312  -2.578  0.01033 *
## const   1970.82680  169.58479  11.621  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 222.5 on 352 degrees of freedom
## Multiple R-Squared: 0.1224,  Adjusted R-squared: 0.1024
## F-statistic: 6.134 on 8 and 352 DF,  p-value: 2.061e-07
##
##
##
## Covariance matrix of residuals:
##       prec     ubts
## prec  6.06    16.17
## ubts 16.17 49502.47
##
## Correlation matrix of residuals:
##         prec    ubts
## prec 1.00000 0.02952
## ubts 0.02952 1.00000
```

```
# We interpret the coefficients in the usual way,but now have a
# system of equations. For example, for VAR(1) we have:
# y1 = c11 y(1,t-1) + c12 y(2,t-1)
# y2 = c21 y(1,t-1) + c22 y(2,t-1)
# The ourtput from summary are cij, cov, and corr.
```

```
# Plot the fit and orginal data
quartz()
plot(y_model)
```

Diagram of fit and residuals for prec

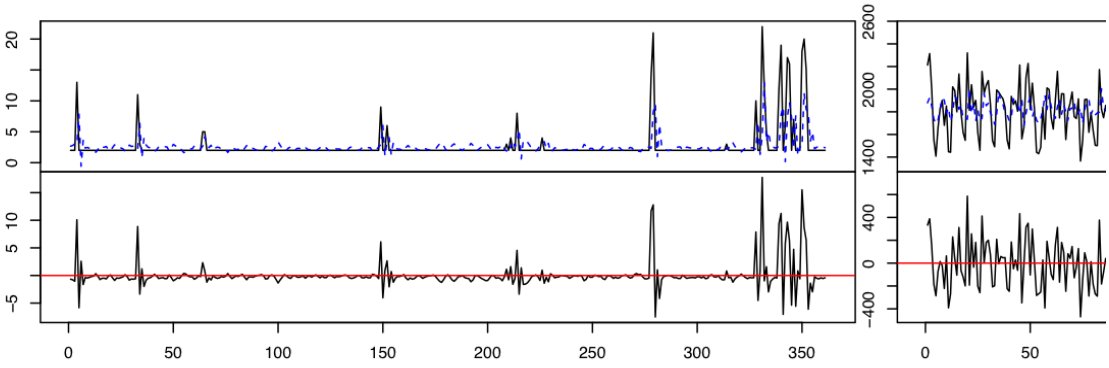
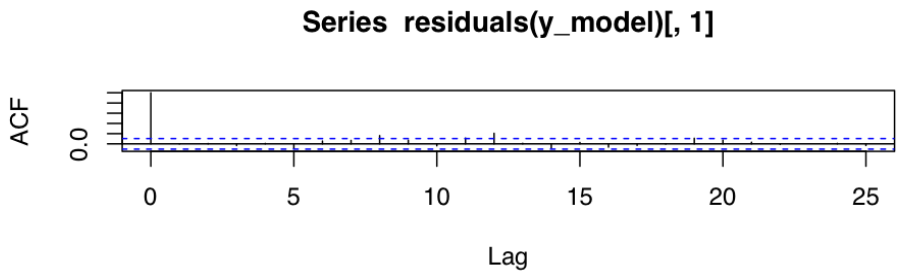
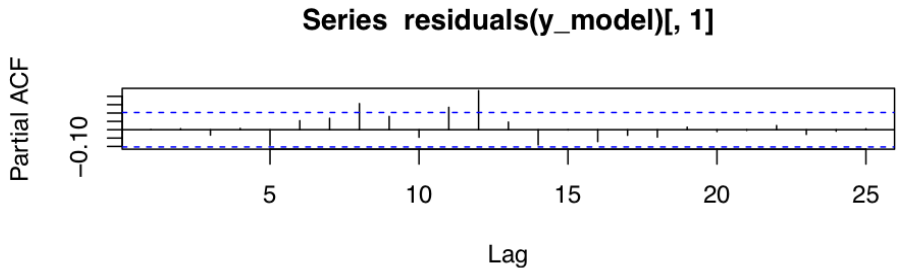


figure margins too large

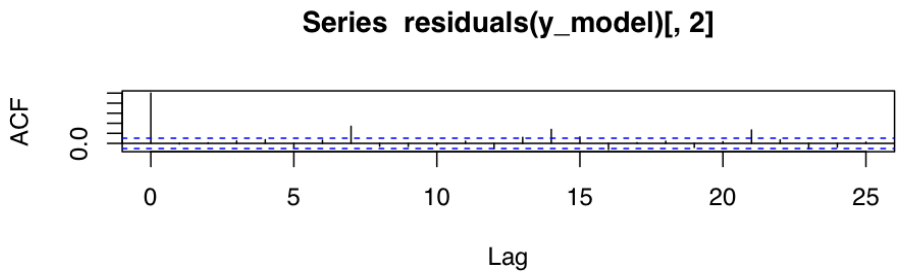
figure margins too

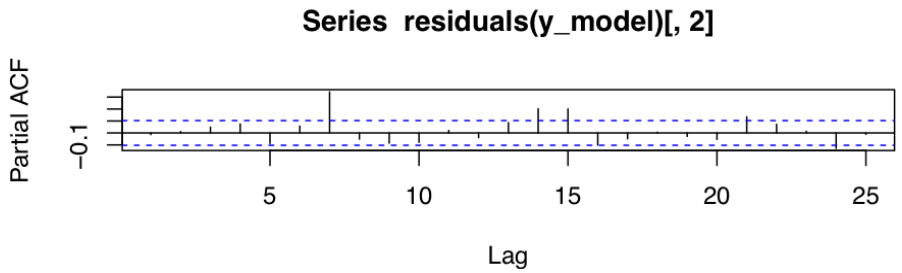
```r
# Look at ACF and PACf
par(mfrow=c(2,1))
acf(residuals(y_model)[,1])
pacf(residuals(y_model)[,1])
```



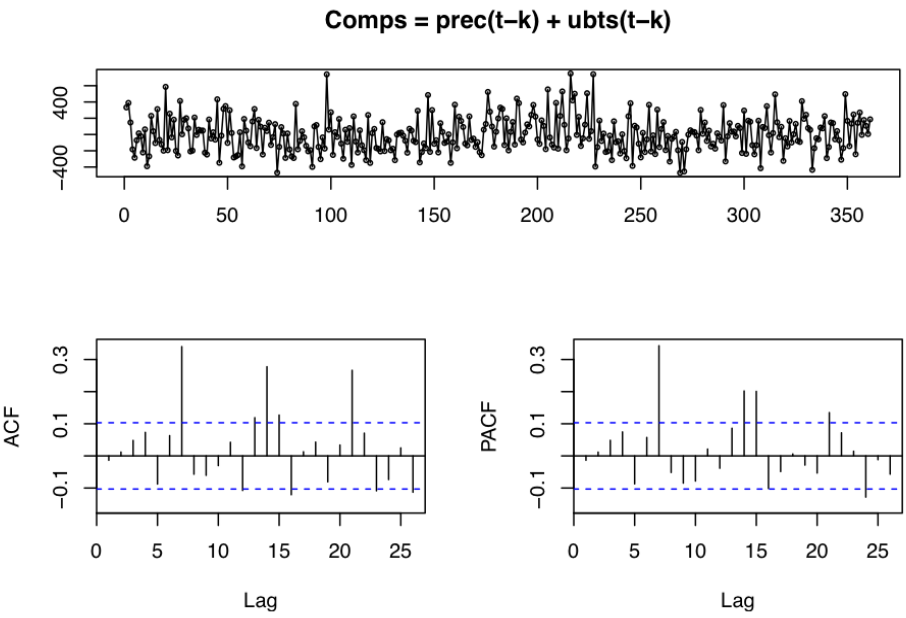



```r
par(mfrow=c(2,1))
acf(residuals(y_model)[,2])
pacf(residuals(y_model)[,2])
```






```
tsdisplay(residuals(y_model)[,2],main ="Comps = prec(t-k) + ubts(t-k)")
```

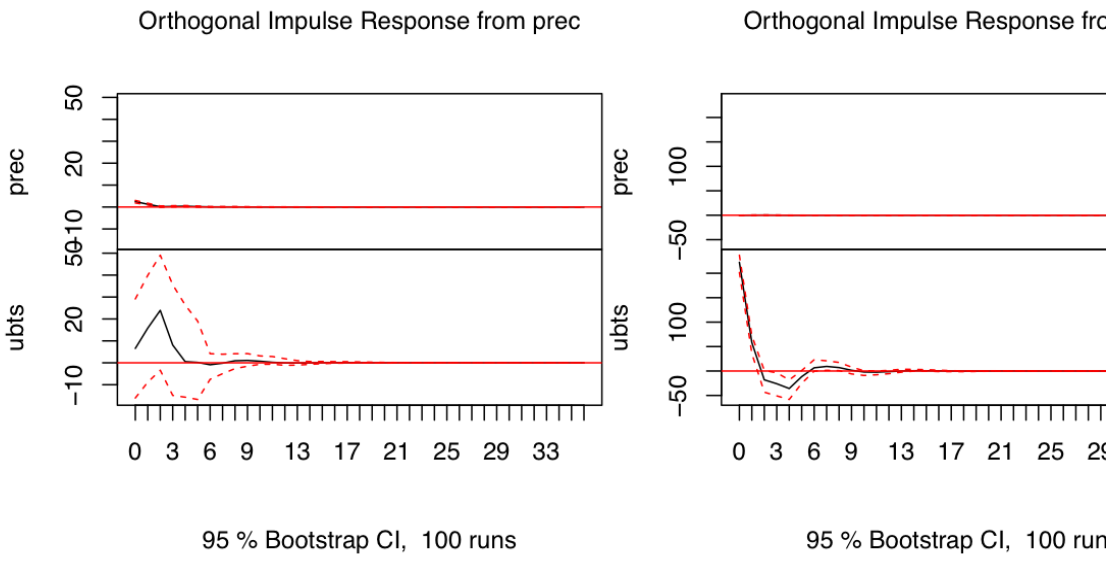



```
# Impulse Response Function
irf(y_model)
```

```
##
## Impulse response coefficients
## $prec
##              prec       ubts
##  [1,] 2.46171078  6.5668732
##  [2,] 1.26269694 15.8366040
##  [3,] 0.15874597 23.9190916
##  [4,] 0.26936090  8.1047065
##  [5,] 0.37442979  0.5332210
##  [6,] 0.18177838  0.1800728
##  [7,] 0.06029803 -0.8973371
##  [8,] 0.05825468 -0.1811793
##  [9,] 0.05121531  0.9365321
## [10,] 0.02705701  1.0885402
## [11,] 0.01571132  0.7191171
##
## $ubts
##              prec       ubts
##  [1,]  0.00000000 222.394573
##  [2,]  0.15828641  58.164895
##  [3,]  0.35443666 -18.054597
##  [4,]  0.17065039 -26.065064
##  [5,] -0.02865184 -35.777055
##  [6,] -0.04541401 -11.540400
##  [7,] -0.02492658   6.581525
##  [8,] -0.00645062   9.396940
##  [9,]  0.01286801   6.986139
## [10,]  0.01852126   1.520397
## [11,]  0.01197190  -2.071960
##
##
## Lower Band, CI= 0.95
## $prec
##               prec        ubts
##  [1,]  1.920963850 -18.1110234
##  [2,]  0.913228599  -2.1221102
##  [3,] -0.115319316  -2.9615759
##  [4,]  0.013124991 -14.6782229
##  [5,]  0.116912279 -18.4251474
##  [6,] -0.052850311 -17.9874519
##  [7,] -0.067376825  -7.4500466
##  [8,] -0.027850583  -5.4918182
##  [9,] -0.031578714  -2.4385456
## [10,] -0.023582371  -1.4055478
## [11,] -0.007781658  -0.7078437
##
## $ubts
##               prec        ubts
##  [1,]  0.000000000 203.4322519
##  [2,] -0.121632579  30.3515893
##  [3,]  0.063504229 -40.6580099
##  [4,] -0.115126526 -45.3770259
##  [5,] -0.319489233 -56.7162810
##  [6,] -0.250214270 -25.5315591
```

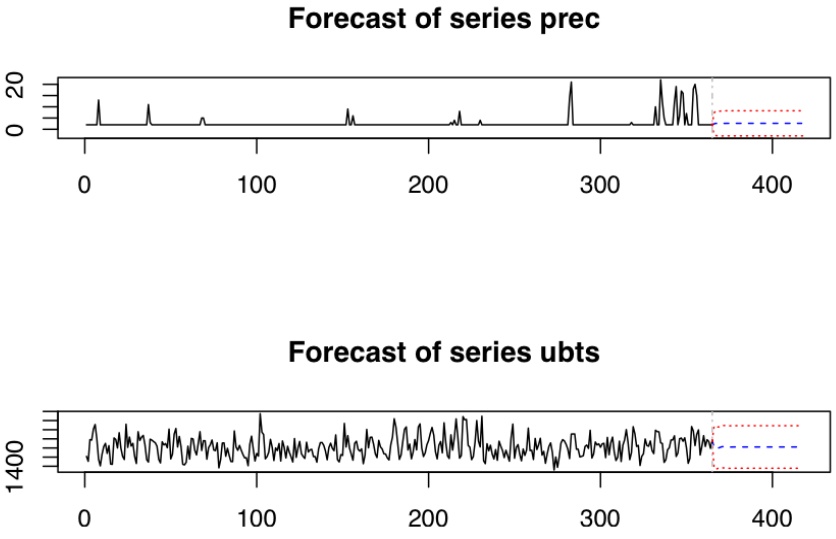
```
##  [7,] -0.122850286  -1.3912408
##  [8,] -0.060153115   1.1220402
##  [9,] -0.020640745   0.1057883
## [10,] -0.005583414  -2.8958085
## [11,] -0.007420173  -7.3874819
##
##
## Upper Band, CI= 0.95
## $prec
##              prec      ubts
##  [1,] 3.07683872 30.391871
##  [2,] 1.61989814 39.280888
##  [3,] 0.45049211 45.192755
##  [4,] 0.55397786 29.923379
##  [5,] 0.64808584 23.818441
##  [6,] 0.41663756 15.305859
##  [7,] 0.18300841  5.153154
##  [8,] 0.21091574  3.533929
##  [9,] 0.20056683  4.512264
## [10,] 0.13276400  3.755854
## [11,] 0.08553401  2.995258
##
## $ubts
##              prec        ubts
##  [1,] 0.00000000 241.9772098
##  [2,] 0.40130084  80.7478624
##  [3,] 0.63063225   2.9988325
##  [4,] 0.46749279  -5.3197409
##  [5,] 0.23618693 -12.0477771
##  [6,] 0.11684813   0.1174549
##  [7,] 0.03232911  16.8463193
##  [8,] 0.04890581  22.2931109
##  [9,] 0.05963831  15.4225757
## [10,] 0.05555062   7.2496209
## [11,] 0.03973105   0.7041196
```

```
#quartz()
#pdf("irf.pdf", width=8, height=8)
plot(irf(y_model, n.ahead=36))
```




```
#dev.off()
```

```
#Forecast
#holdout_matrix = hold out data
#var.predict = predict(object=y_model, n.ahead=52, dumvar=holdout_matrix);
var.predict = predict(object=y_model, n.ahead=52)
plot(var.predict)
```



```r
dev.print(device=postscript,"forecast.eps",width=7,height=7, horizontal=FALSE)
```

```
## pdf
##   2
```

```r
#dev.off()

#Granger Test
grangertest(ubts ~ prec, order = 8)
```

```
## Granger causality test
##
## Model 1: ubts ~ Lags(ubts, 1:8) + Lags(prec, 1:8)
## Model 2: ubts ~ Lags(ubts, 1:8)
##   Res.Df Df      F Pr(>F)
## 1    340
## 2    348 -8 0.9389 0.4843
```

```r
#Variance Decomposition (Forecast Error Variance Decomposition)
#plot(fevd(y_model, n.ahead = 5))

#CUSUM Plot
#plot(stability(y_model, type = "Rec-CUSUM"), plot.type="single")
```