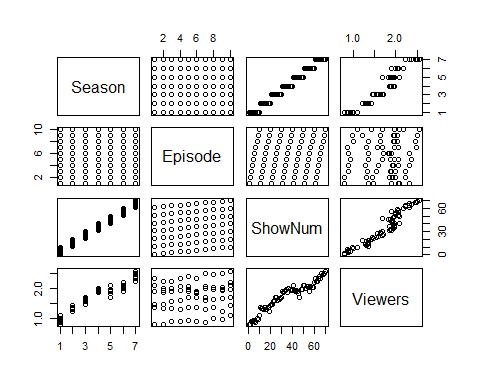
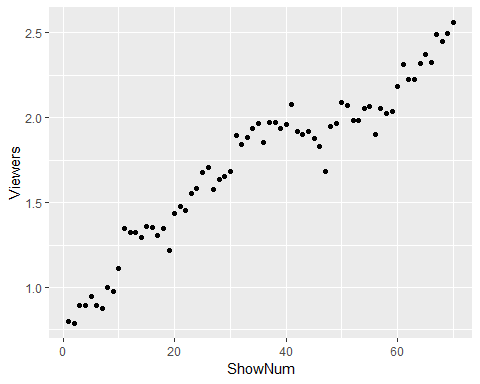
Homework 3 Viewership

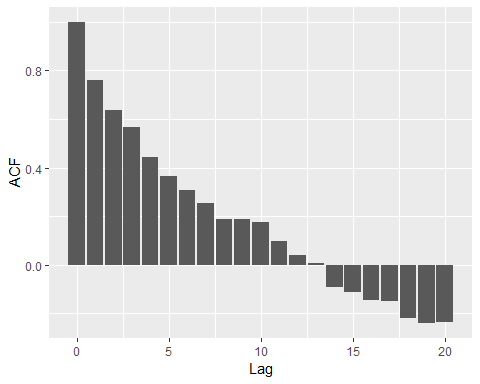
Jonah Meherg

#1



There appears to be a linear relationship between log(viewers) and ShowNum, but there appears to be a clear change in the basic scatter plot around ShowNum = 40, this necessitates a look at a time series.

#2



The acf of the residuals from the linear model shows that there is temporal correlation in the residuals and thus this should be accounted for in the model. This temporal correlation means that past viewership affects viewership in the future, for example people may be more likely to watch the next episode of the series based on the previous episode or season.

#3

Running a sarima model for each possible combination of values for p, q, P, and Q, keeping d = 0 and D = 1, the appropriate values are p = 2, q = 0, P = 0, and Q = 1. As this gives the best AIC and BIC values. SOt he best model is SARIMA(2,0,0,0,1,1) where s = 10.

#4

y = XB + ε

Y is the response variable of viewers

X is the explanatory variable matrix with a column of ones followed by columns for each explanatory variable.

ε is the residuals that are distributed by SARIMA with s = 10.

ε ~ SARIMA(2,0,0,0,1,1)10

εt - εt-10 = dt

dt = Φ1dt-1 + Φ2dt-2 + Θdt-10 + wt

where wt ~ N(0, σ2)

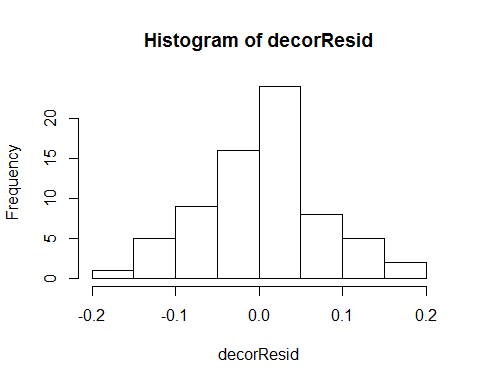
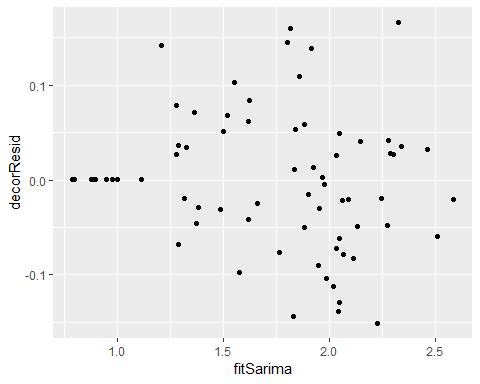
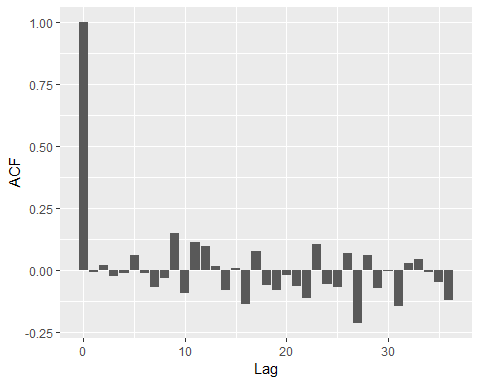
and Φ1 is the affect of one show previous,

Φ2 is the affect of two shows ago, and

Θ is the affect of the corresponding show of the previous season.

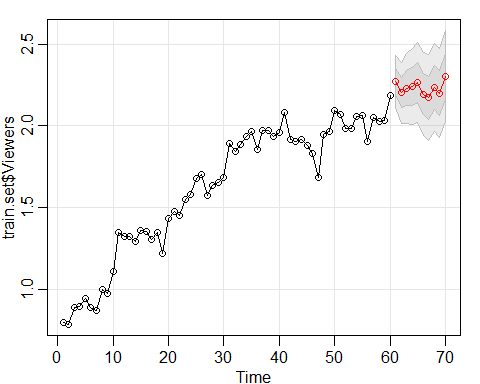
Statistical inference can be made according to the maximum specified during the creation of the splines, and no prediction can be considered accurate past that point, in this case being one season in the future, all results for viewers are expressed in percent.

#5



Linearity is validated because the fitted values against the residuals plot appears to be linear, there appears to be independence as the ACF appears to no longer be temporally correlated, normality is seen through the normality of the residuals. And there is assumed to be equal variance based on our model having equal variance.

#6

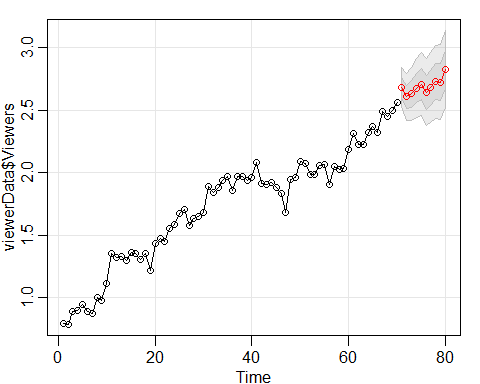


This model results in an RPMSE of 0.1835391, with increases in viewership of a couple of percent between each show. The predictions for each show have a 95% confidence lower estimate of “2.114273 2.017491 2.017702 2.011621 2.020206 1.939456 1.908454 1.967351 1.926863 2.024952” and an upper estimate of “2.429220 2.391132 2.442838 2.472545 2.508832 2.449351 2.434875 2.506565 2.475839 2.581355”, with the predictions being “2.271746 2.204312 2.230270 2.242083 2.264519 2.194403 2.171665 2.236958 2.201351 2.303154” and the actual values being “2.313525 2.226783 2.224624 2.319442 2.372111 2.326302 2.490723 2.447551 2.494857 2.562639”

#7

Looking at the estimated coefficients of the autoregressive and moving average pieces, the only one that seems to be decreasing is the moving average coefficient of the model, and as such, the viewership may have moments with a decrease from one episode to another, but will over see an increase in viewership throughout each season.

#8



Looking at the predictions, there appears to still be a steady increase in viewership across the season showing that season 8 ends successfully and so a season 9 would be expected to have similar results.

Appendix

#1

viewerData <- read.table("Viewership.txt", header = TRUE)  
viewerData$Viewers <- log(viewerData$Viewers)  
ggplot(data = viewerData, mapping=aes(x=ShowNum, y=Viewers)) + geom\_point()

pairs(viewerData)

#2

viewer.lm <- lm(Viewers~ShowNum, data = viewerData)  
viewer.ACF <- acf(resid(viewer.lm), lag.max=20)

ACF.dframe <- data.frame(Lag=viewer.ACF$lag, ACF=viewer.ACF$acf)  
ggplot(data=ACF.dframe, aes(x=Lag, y=ACF)) + geom\_col()

#3

key.sarima <- data.frame(c(0,0,0,0,1,0),  
 c(1,0,0,0,1,0),  
 c(2,0,0,0,1,0),  
 c(0,0,1,0,1,0),  
 c(0,0,2,0,1,0),  
 c(1,0,2,0,1,0),  
 c(2,0,1,0,1,0),  
 c(2,0,2,0,1,0),  
 c(1,0,1,0,1,0),  
 c(0,0,0,1,1,0),  
 c(1,0,0,1,1,0),  
 c(2,0,0,1,1,0),  
 c(0,0,1,1,1,0),  
 c(0,0,2,1,1,0),  
 c(1,0,2,1,1,0),  
 c(2,0,1,1,1,0),  
 c(2,0,2,1,1,0),  
 c(1,0,1,1,1,0),  
 c(0,0,0,0,1,1),  
 c(1,0,0,0,1,1),  
 c(2,0,0,0,1,1),  
 c(0,0,1,0,1,1),  
 c(0,0,2,0,1,1),  
 c(1,0,2,0,1,1),  
 c(2,0,1,0,1,1),  
 c(2,0,2,0,1,1),  
 c(1,0,1,0,1,1),  
 c(0,0,0,1,1,1),  
 c(1,0,0,1,1,1),  
 c(2,0,0,1,1,1),  
 c(0,0,1,1,1,1),  
 c(0,0,2,1,1,1),  
 c(1,0,2,1,1,1),  
 c(2,0,1,1,1,1),  
 c(2,0,2,1,1,1),  
 c(1,0,1,1,1,1))  
X <- bs(x=viewerData$ShowNum, knots=45, degree=1, Boundary.knots= c(min(viewerData$ShowNum), max(viewerData$ShowNum)+10))  
s <- 10  
aics <- rep(x=NA, times = ncol(key.sarima))  
bics <- rep(x=NA, times = ncol(key.sarima))  
for(i in 1:ncol(key.sarima)) {  
 my.ts.model <- sarima(viewerData$Viewers, p=key.sarima[1,i], d=key.sarima[2,i], q=key.sarima[3,i], P=key.sarima[4,i],   
 D=key.sarima[5,i], Q=key.sarima[6,i], S=s, xreg=X, details=FALSE)  
 aics[i] <- my.ts.model$AIC  
 bics[i] <- my.ts.model$BIC  
   
}   
key.sarima[,which(aics == min(aics))]

## [1] 2 0 0 0 1 1

key.sarima[,which(bics == min(bics))]

## [1] 2 0 0 0 1 1

#5

viewers.model <- sarima(viewerData$Viewers, p = 2, d = 0, q = 0, P = 0, D = 1, Q = 1, S = s, xreg = X, details = FALSE)  
  
decorResid <- resid(viewers.model$fit)  
decorResid.ACF <- acf(decorResid, lag.max=36)

decorResid.ACF.dframe <- data.frame(Lag=decorResid.ACF$lag, ACF=decorResid.ACF$acf)  
  
ggplot(data=decorResid.ACF.dframe, aes(x=Lag, y=ACF)) + geom\_col()

fitSarima <- viewerData$Viewers - decorResid  
ggplot(mapping=aes(x=fitSarima, y=decorResid)) + geom\_point()

## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.  
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.

hist(decorResid)

#6

newX <- bs(viewerData$ShowNum, knots = 45, degree = 1)  
train.set <- viewerData[1:(nrow(viewerData)-10),]  
test.set <- viewerData[-(1:(nrow(viewerData)-10)),]  
predicts <- max(viewerData$ShowNum) + seq(1, 10, by=1)  
x.train <- X[1:(nrow(viewerData)-10),]  
x.test <- X[-(1:(nrow(viewerData)-10)),]  
  
  
viewer.for <- sarima.for(train.set$Viewers, p=2, d=0, q=0, P=0, D=1, Q=1, S=10, xreg = x.train, n.ahead=10, newxreg=x.test)

n <- 5  
  
  
(viewer.for$pred - test.set$Viewers)^2 %>% mean() %>% sqrt()

## [1] 0.1835391

low <- viewer.for$pred - qt(0.975, df=nrow(train.set)-n)\*viewer.for$se  
upr <- viewer.for$pred + qt(0.975, df=nrow(train.set)-n)\*viewer.for$se  
viewer.for$pred

## Time Series:  
## Start = 61   
## End = 70   
## Frequency = 1   
## [1] 2.271746 2.204312 2.230270 2.242083 2.264519 2.194403 2.171665  
## [8] 2.236958 2.201351 2.303154

low

## Time Series:  
## Start = 61   
## End = 70   
## Frequency = 1   
## [1] 2.114273 2.017491 2.017702 2.011621 2.020206 1.939456 1.908454  
## [8] 1.967351 1.926863 2.024952

upr

## Time Series:  
## Start = 61   
## End = 70   
## Frequency = 1   
## [1] 2.429220 2.391132 2.442838 2.472545 2.508832 2.449351 2.434875  
## [8] 2.506565 2.475839 2.581355

viewerData$Viewers[61:70]

## [1] 2.313525 2.226783 2.224624 2.319442 2.372111 2.326302 2.490723  
## [8] 2.447551 2.494857 2.562639

#7

viewers.model$ttable

## Estimate SE t.value p.value  
## ar1 0.6388 0.1239 5.1555 0.000  
## ar2 0.2817 0.1273 2.2135 0.031  
## sma1 -0.7840 0.2248 -3.4870 0.001  
## 1 1.1482 0.2387 4.8095 0.000  
## 2 1.9818 0.3583 5.5316 0.000

#8

pred.Season <- max(viewerData$ShowNum) + seq(1, 10, by=1)  
Xpred <- predict(X, newx=pred.Season)  
my.for <- sarima.for(viewerData$Viewers, p=2, d=0, q=0, P=0, D=1, Q=1, S=10, n.ahead=10, newxreg=Xpred)

my.for$pred

## Time Series:  
## Start = 71   
## End = 80   
## Frequency = 1   
## [1] 2.684463 2.607017 2.631389 2.676295 2.709451 2.643435 2.682187  
## [8] 2.728309 2.725808 2.828278