

Final Project

Hannah Wilder and Chathura Gunasekara

April 9, 2016

Change working directory

Load data (assumes file is in working directory)

```
#load the data
beerData<-read.csv("monthly-beer-production-in-austr.csv")
```

```
#cut off the last row which is NA
beerData<-beerData[-nrow(beerData),]
colnames(beerData)<-c("Month", "Production")
```

```
#Get a five number summary of the data
summary(beerData$Production)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      64.8   112.9   139.2   136.4   158.8   217.8
```

```
#turn into time series also hold back the last two years of data for forecasting
beerTS<-ts(beerData[1:(nrow(beerData)-24),2], frequency=12, start=c(1956,1))
beer_forecast<-ts(beerData[(nrow(beerData)-23):nrow(beerData), 2], start=c(1993,9), frequency=12)

adf.test(beerTS)
```

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

```
##
## Augmented Dickey-Fuller Test
##
## data: beerTS
## Dickey-Fuller = -4.206, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

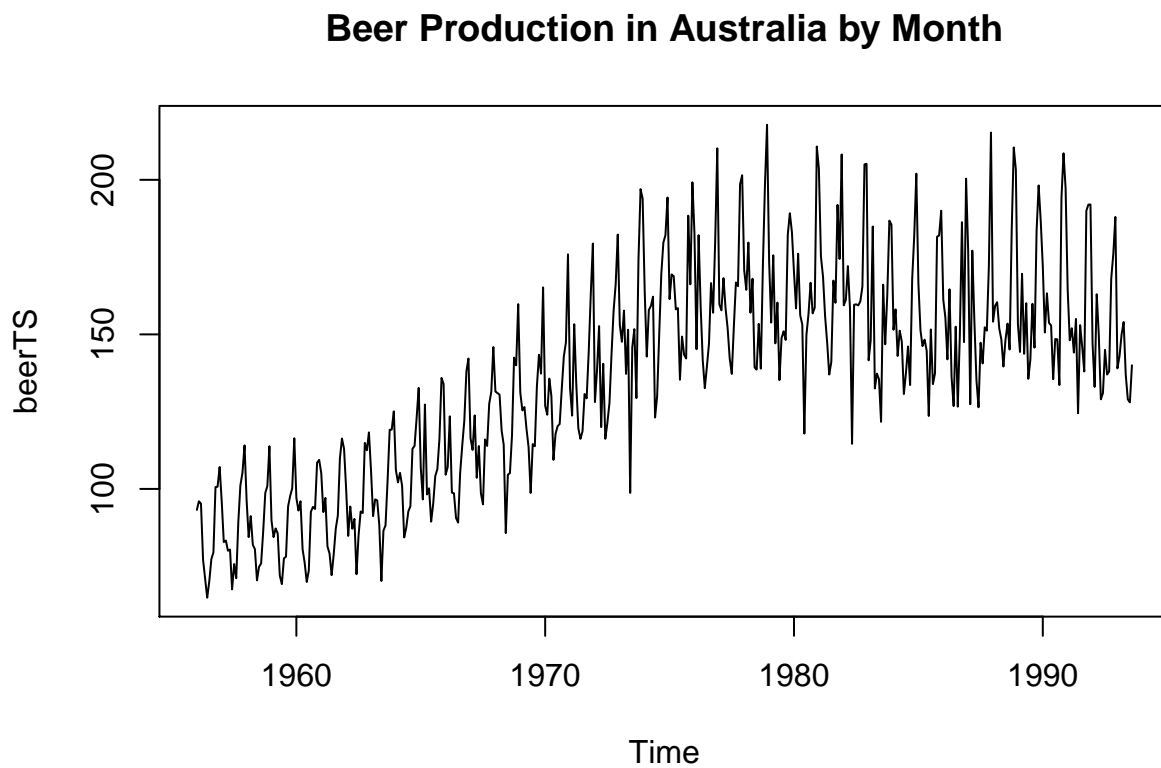
```
pp.test(beerTS)
```

```
## Warning in pp.test(beerTS): p-value smaller than printed p-value
```

```
##
## Phillips-Perron Unit Root Test
##
## data: beerTS
## Dickey-Fuller Z(alpha) = -167.6, Truncation lag parameter = 5,
## p-value = 0.01
## alternative hypothesis: stationary
```

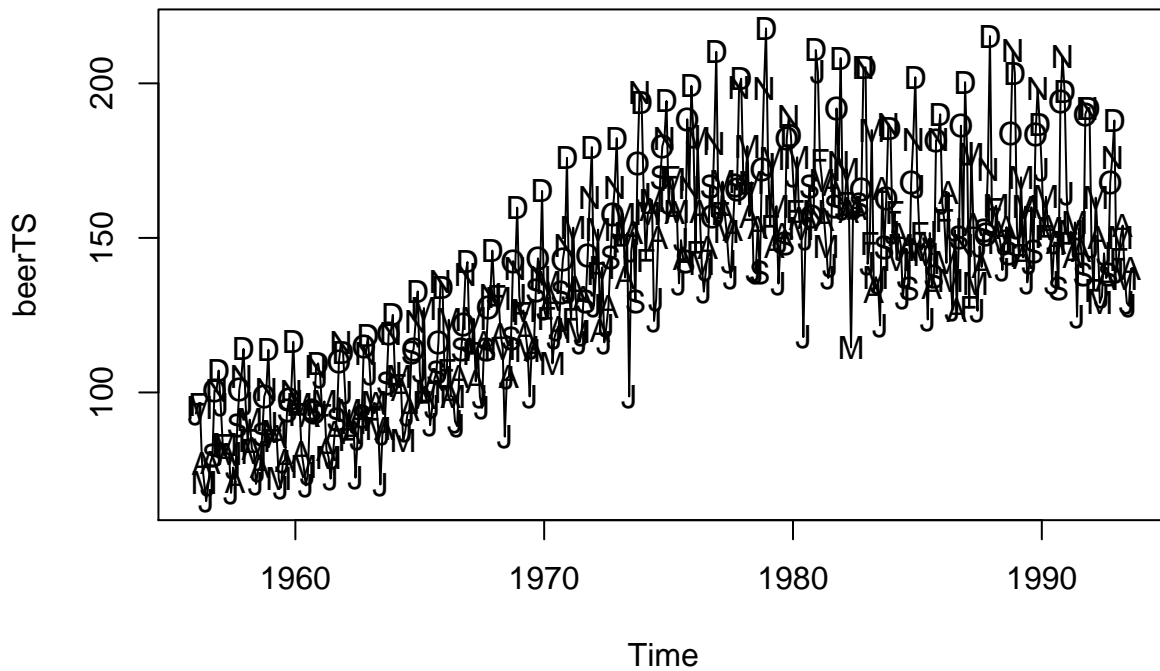
Plot population data

```
library(TSA)
par(mfrow=c(1,1))
plot(beerTS, main="Beer Production in Australia by Month")
```



```
plot(beerTS, main="Beer Production in Australia by Month (seasons marked)", type="l")
points(y=beerTS, x=time(beerTS), pch=as.vector(season(beerTS)))
```

Beer Production in Australia by Month (seasons marked)



```
require(fpp)
```

```
## Loading required package: fpp
```

```
## Loading required package: forecast
```

```
## Loading required package: zoo
```

```
##
```

```
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
```

```
##
```

```
##      as.Date, as.Date.numeric
```

```
## Loading required package: timeDate
```

```
##
```

```
## Attaching package: 'timeDate'
```

```
## The following objects are masked from 'package:TSA':
```

```
##
```

```
##      kurtosis, skewness
```

```
## This is forecast 7.0
```

```
##
```

```
## Attaching package: 'forecast'
```

```
## The following objects are masked from 'package:TSA':
##
## fitted.Arima, plot.Arima

## The following object is masked from 'package:nlme':
##
## getResponse
```

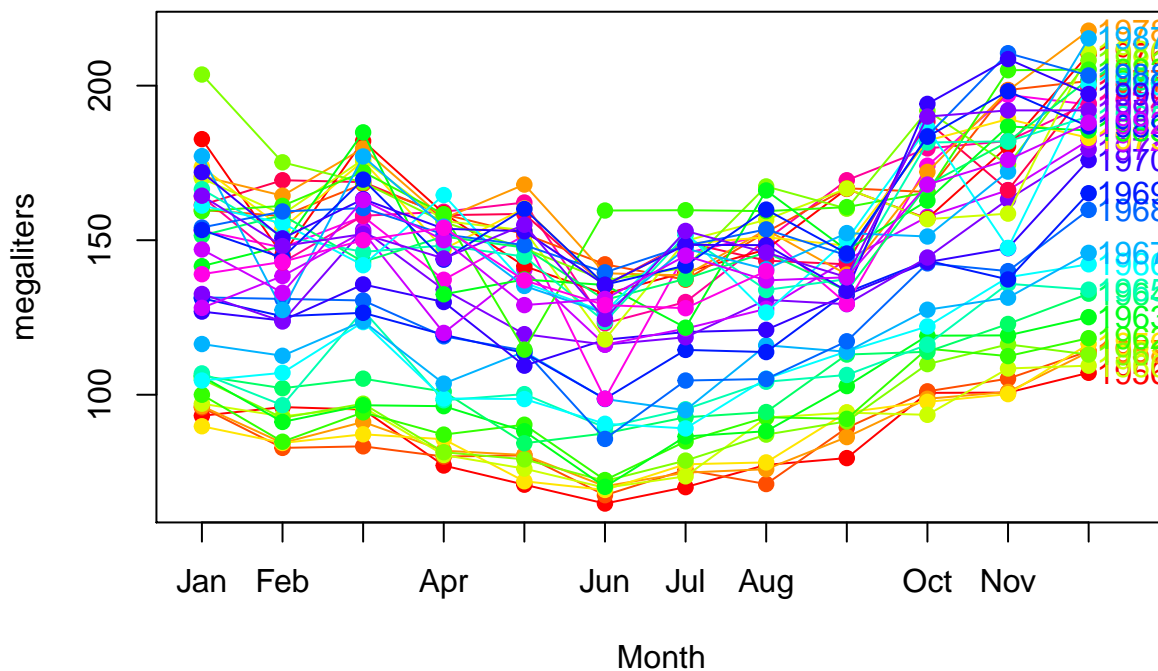
```
## Loading required package: fma
```

```
## Loading required package: expsmooth
```

```
## Loading required package: lmtest
```

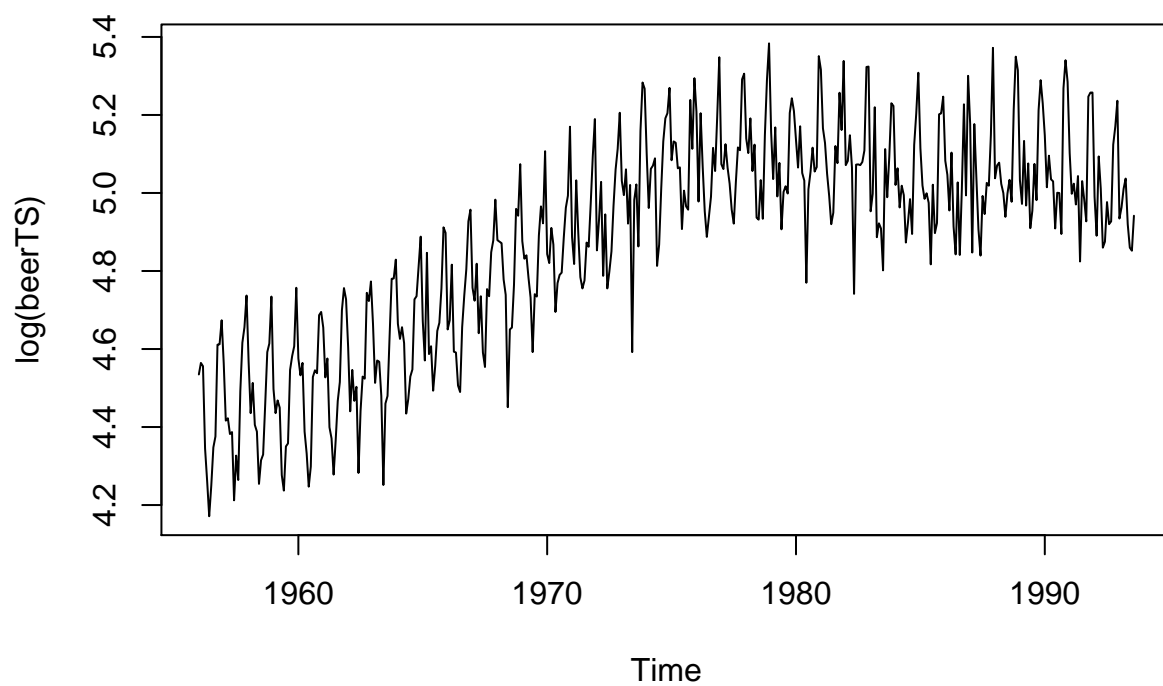
```
seasonplot(beerTS,year.labels=TRUE,ylab="megaliters",main="Seasonal plot: quarterly beer production", col=rainbow(12))
```

Seasonal plot: quarterly beer production



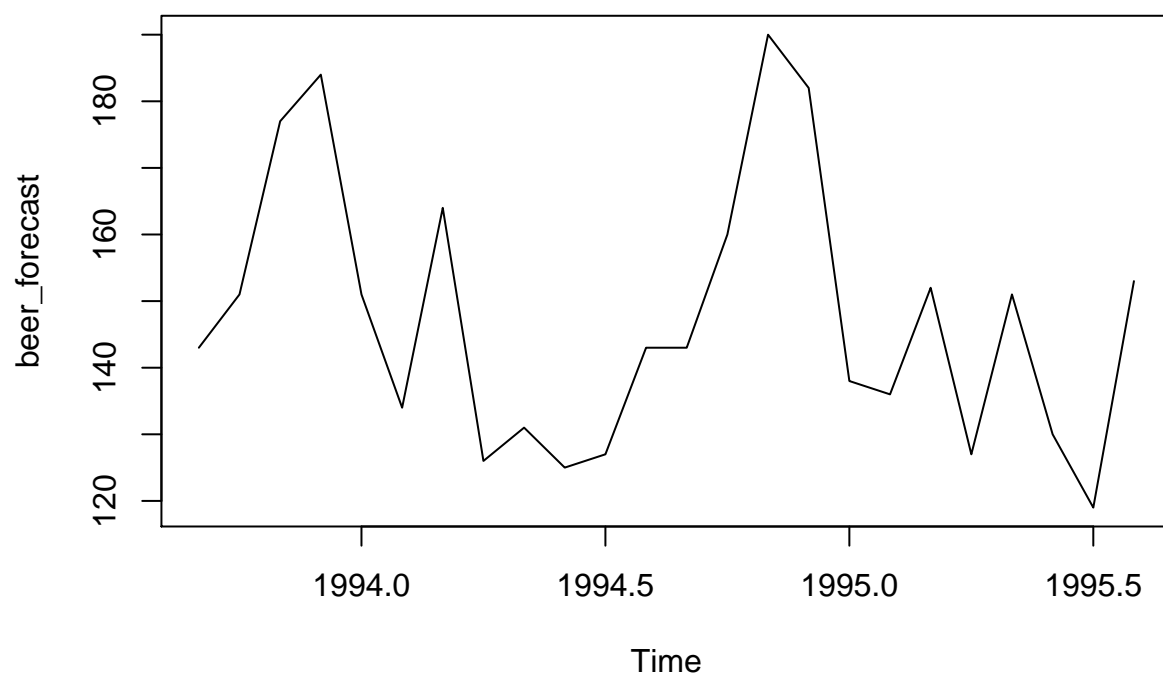
```
plot(log(beerTS), main="Logged Beer Production in Australia by Month")
```

Logged Beer Production in Australia by Month



```
plot(beer_forecast, main="Beer production values to be forecasted")
```

Beer production values to be forecasted



```
logBeer<-log(beerTS)
```

Investigate possible relationship with population data

```
#load population data
library(reshape)
```

```
## Warning: package 'reshape' was built under R version 3.2.5
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.2.5
```

```
#Clean up population data
```

```
pop_totalData<-t(read.csv("Pop_total.csv", row.names=1))
dropCols<-colnames(pop_totalData) %in% c("Unspecified","Period not indicated")
rownames(pop_totalData)<-c(1921:2011)
pop_totalDataLong<-pop_totalData[,!dropCols]
pop_totalData<-pop_totalData[paste(1956:1995),!dropCols]
```

```
#Aggregate beer data
```

```
beerYear<-seq(from=1956, to=1996, by=1)
beerYear<-rep(beerYear, each=12)
beerYear<-beerYear[1:nrow(beerData)]
beerAg<-aggregate(beerData[,2], FUN=mean, by=list(year=beerYear))
```

```
#Attach to beer data
```

```
beerPop<-data.frame(cbind(beer=beerAg[,2],pop_totalData))
beerPopScale<-scale(beerPop)
```

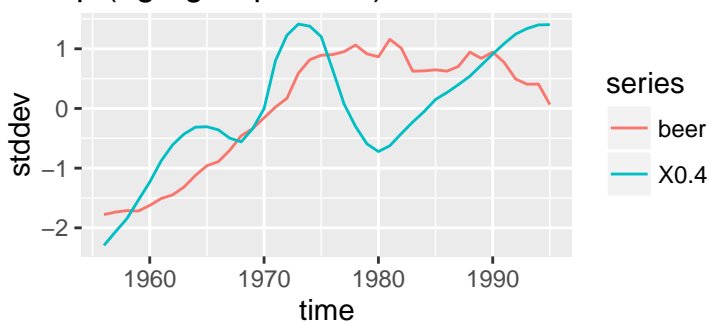
```
beerPopRes<-melt(beerPopScale, variable.name="series")
colnames(beerPopRes)<-c("time", "series", "stddev")
```

```
allNames<-colnames(beerPop)[2:length(colnames(beerPop))]
```

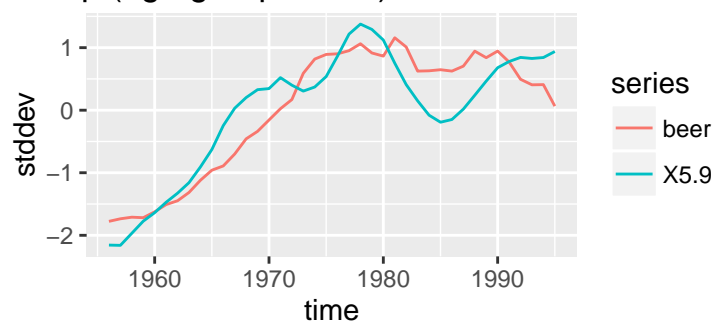
```
#Plot data for each age group and beer data on same plot
```

```
par(mfrow=c(2,2))
for (name in allNames) {
  subset_data<-subset(beerPopRes, beerPopRes$series%in%c("beer", name))
  newPlot<-ggplot(subset_data, aes(time,stddev)) + geom_line(aes(colour = series)) +ggtitle(paste("Pop (age gr", name)
  print(newPlot)
}
```

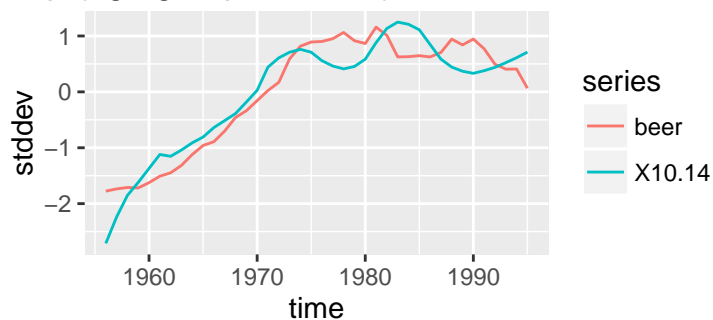
Pop (age group X0.4) and Beer Prod



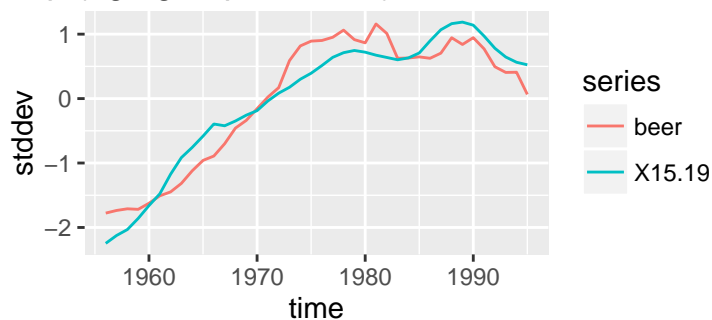
Pop (age group X5.9) and Beer Prod



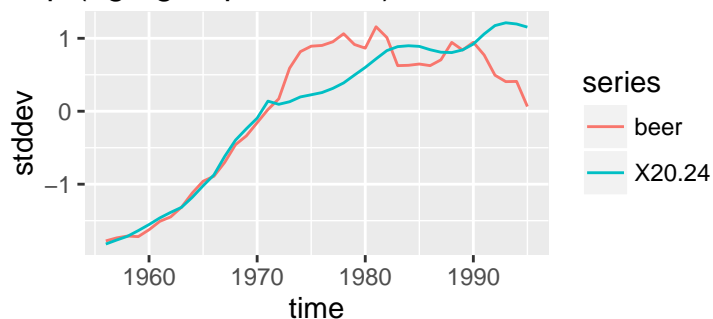
Pop (age group X10.14) and Beer Prod



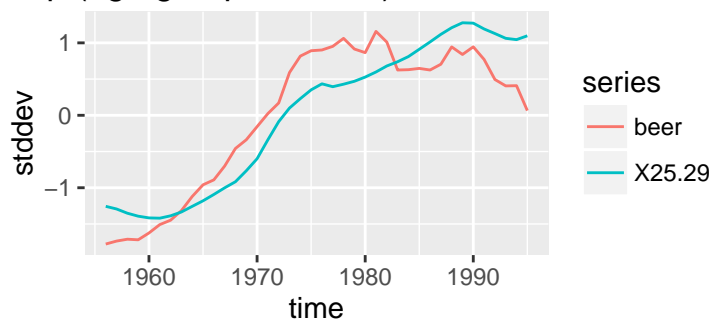
Pop (age group X15.19) and Beer Prod



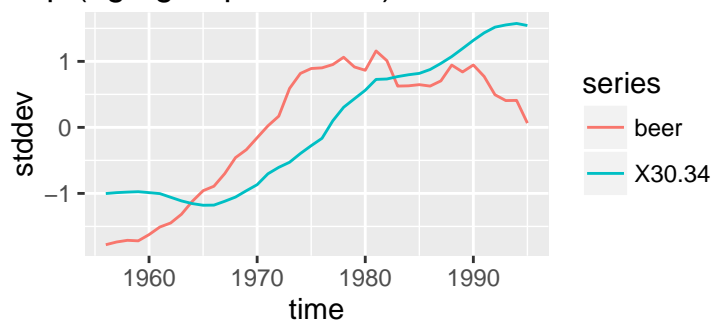
Pop (age group X20.24) and Beer Prod



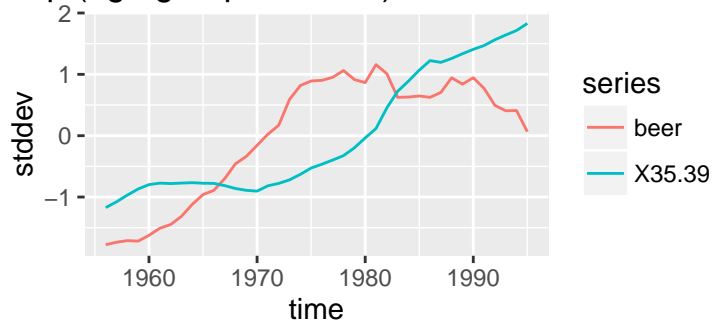
Pop (age group X25.29) and Beer Prod



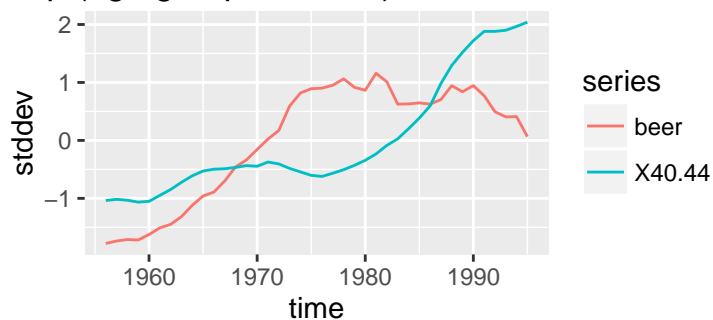
Pop (age group X30.34) and Beer Prod



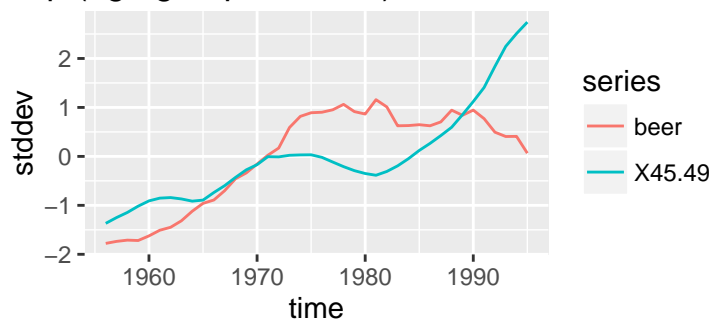
Pop (age group X35.39) and Beer Prod



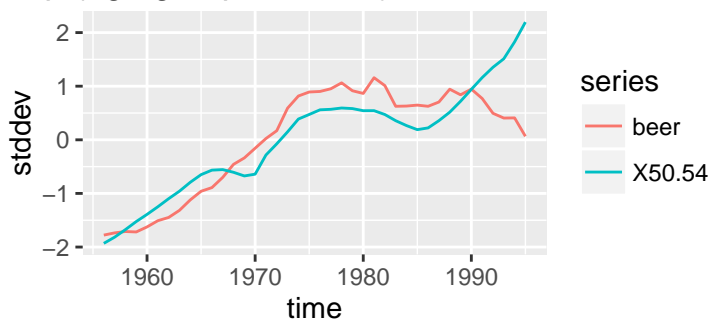
Pop (age group X40.44) and Beer Prod



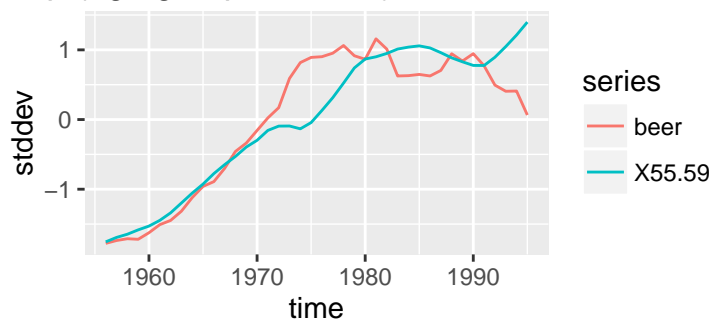
Pop (age group X45.49) and Beer Prod



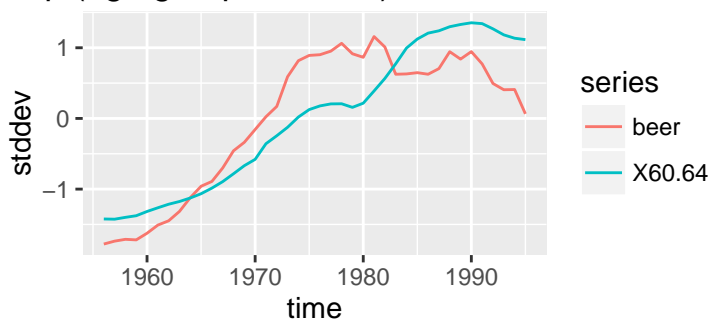
Pop (age group X50.54) and Beer Prod



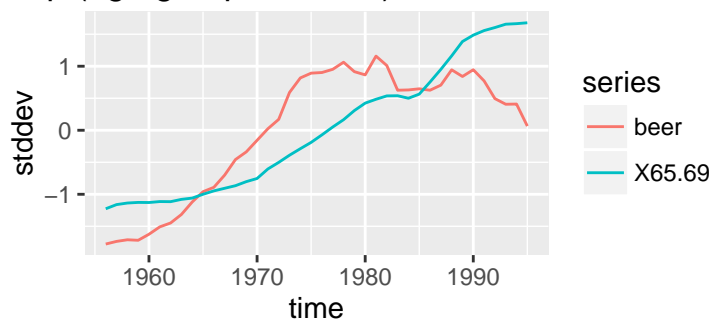
Pop (age group X55.59) and Beer Prod



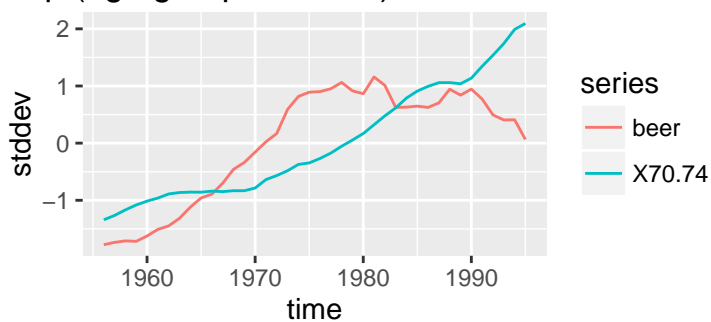
Pop (age group X60.64) and Beer Prod



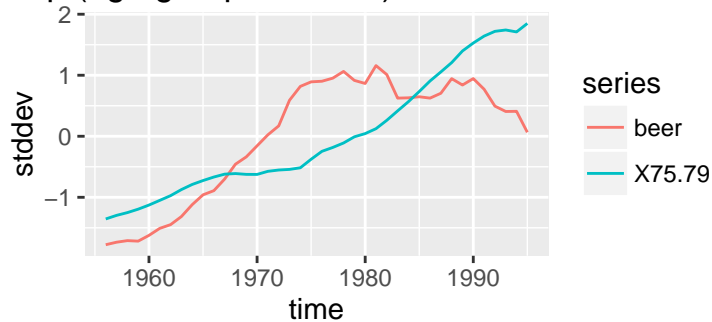
Pop (age group X65.69) and Beer Prod



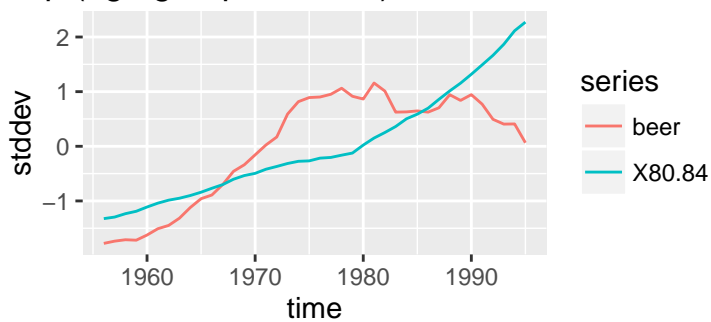
Pop (age group X70.74) and Beer Prod



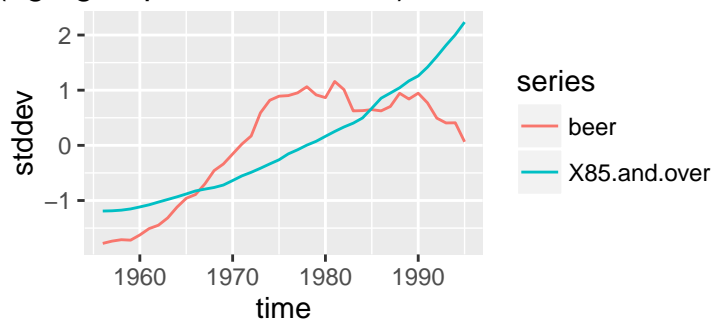
Pop (age group X75.79) and Beer Prod



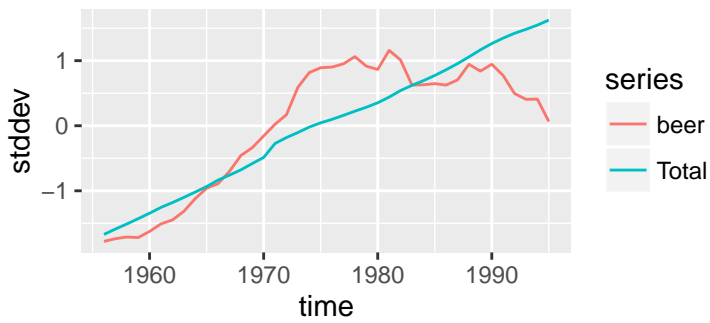
Pop (age group X80.84) and Beer Prod



(age group X85.and.over) and Beer Prod



Pop (age group Total) and Beer Prod



```
par(mfrow=c(1,1))
```

```
#Make a model based on the 15-19 age group
yearModel1<-lm(beer ~ X15.19, data=beerPop)
summary(yearModel1)
```

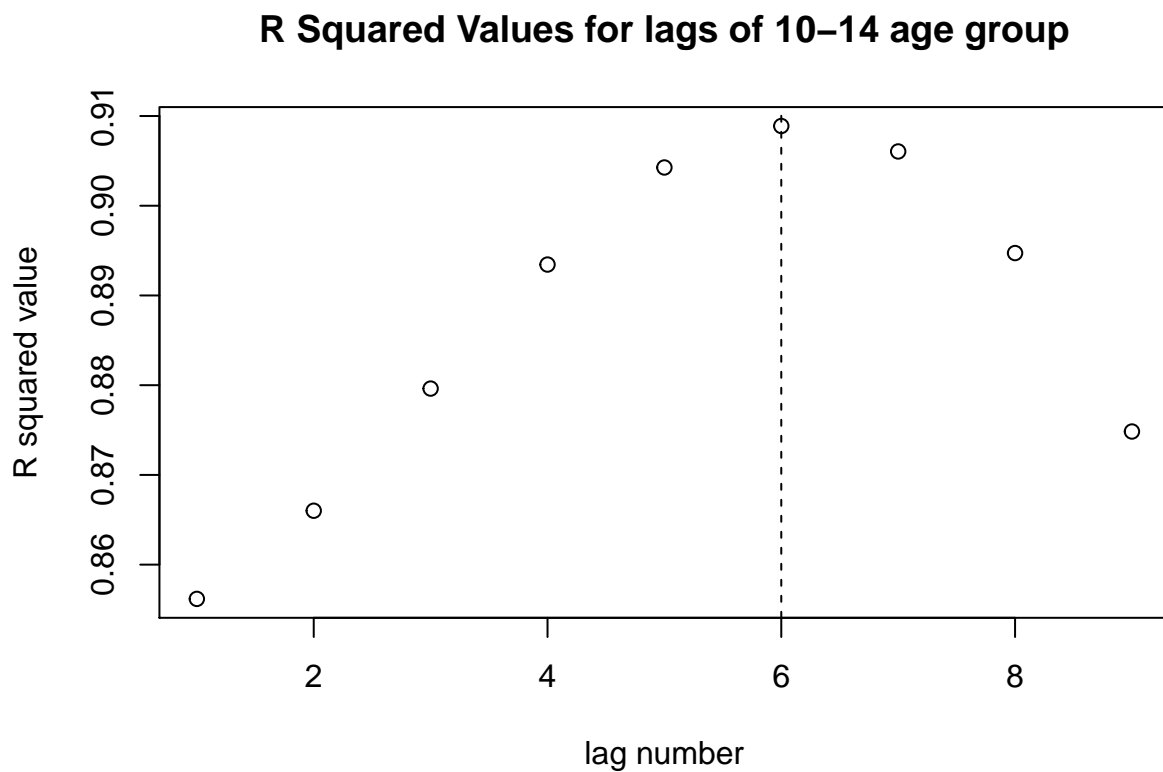
```
##
## Call:
## lm(formula = beer ~ X15.19, data = beerPop)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.6061  -6.6167  -0.9743   6.8537  15.0442
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.713e-01  7.188e+00  -0.038    0.97
## X15.19       1.194e-04  6.163e-06  19.374 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.706 on 38 degrees of freedom
## Multiple R-squared:  0.9081, Adjusted R-squared:  0.9057
## F-statistic: 375.4 on 1 and 38 DF,  p-value: < 2.2e-16
```

```
#plot(ts(residuals(yearModel1), frequency=1, start=c(1956)), main="Residuals from modeling beer production with
```

Explore lagged x10.14 data

```
laggedData<-data.frame(beer=beerAg[,2])
models<-list()
modelRsqs<-c()
for (lag in 0:8) {
  newColNames<-c(colnames(laggedData), paste("lag", lag, sep=""))
  newLag<-pop_totalDataLong[paste(1956:1995-lag), "10-14"]
  laggedData<-data.frame(laggedData, newLag)
  newModel<-lm(beer ~ newLag, data=laggedData)
  models[[paste("lag", lag, sep=")]]<-newModel
  modelRsqs<-c(modelRsqs, summary(newModel)$r.squared)
  colnames(laggedData)<-newColNames
}
```

```
plot(modelRsqr, main="R Squared Values for lags of 10-14 age group", xlab="lag number", ylab="R squared value")
abline(v=6, lty=2)
```



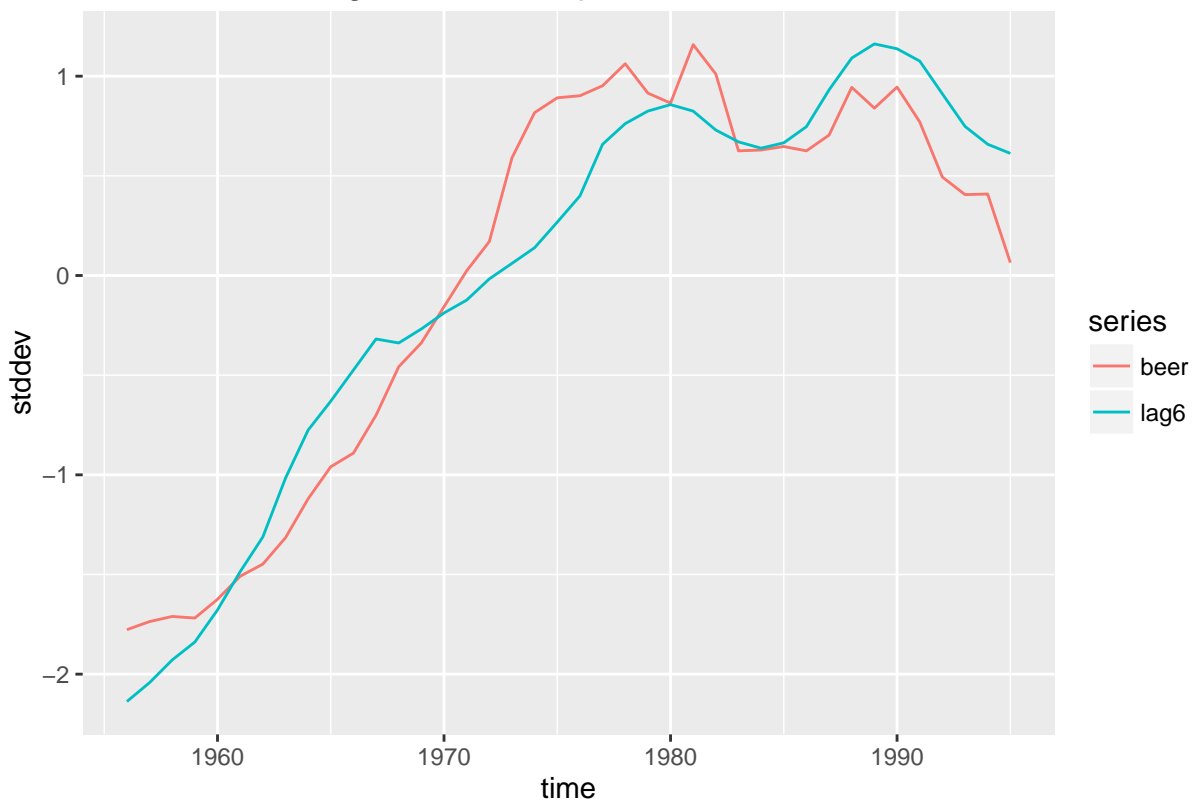
```
max(modelRsqr)
```

```
## [1] 0.9088865
```

```
lagDataScale<-scale(laggedData)[,c(1,8)]
lagDataMelt<-melt(lagDataScale, variable.name="series")
colnames(lagDataMelt)<-c("time", "series", "stddev")

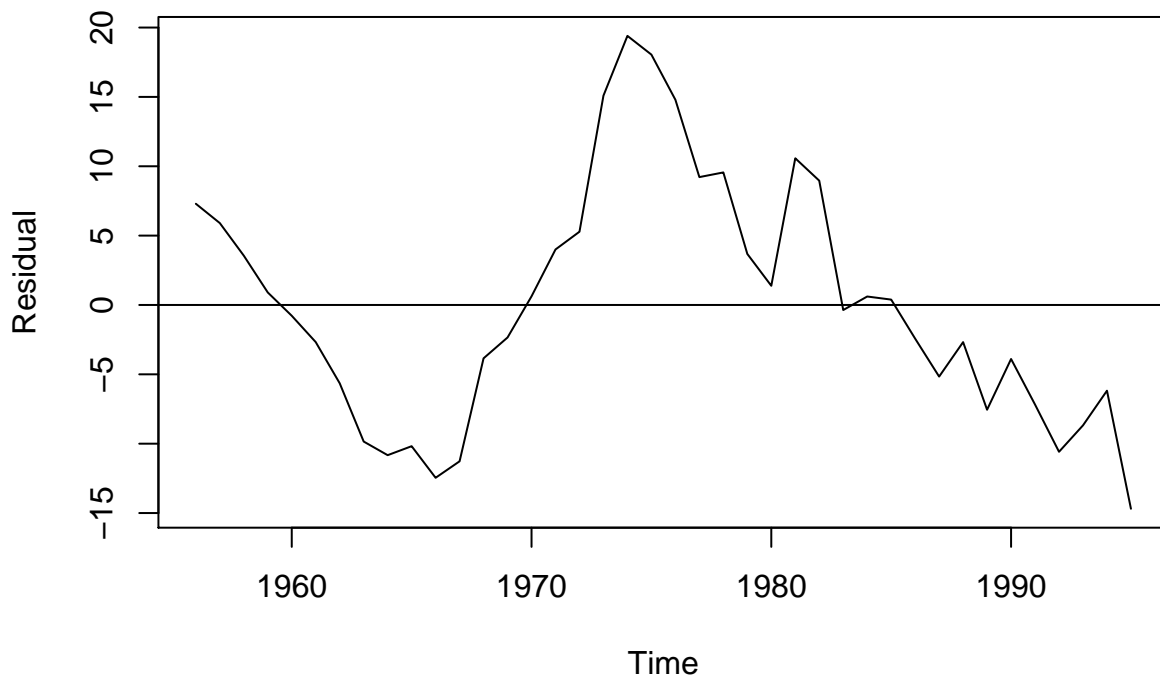
newPlot<-ggplot(lagDataMelt, aes(time,stddev)) + geom_line(aes(colour = series)) +ggtitle(paste("Lags of 10-14"))
print(newPlot)
```

Lags of 10–14 Pop and Beer Prod



```
plot(ts(residuals(models[["lag6"]]), frequency=1, start=c(1956)), main="Residuals from modeling beer production",
      abline(h=0))
```

Residuals from modeling beer production with 10–14 lag 6



Interpolate Monthly Numbers

```
library(zoo)

#Create a vector with missing values for zoo to interpolate
withNA<-c()
for (year in 1:nrow(laggedData)) {
  withNA<-c(withNA, laggedData$lag6[year], rep(NA, 11))
}

#Create a vector with missing values for zoo to interpolate
withNALong<-c()
for (year in 1:nrow(pop_totalDataLong)) {
  withNALong<-c(withNALong, pop_totalDataLong[year, "10-14"], rep(NA, 11))
}

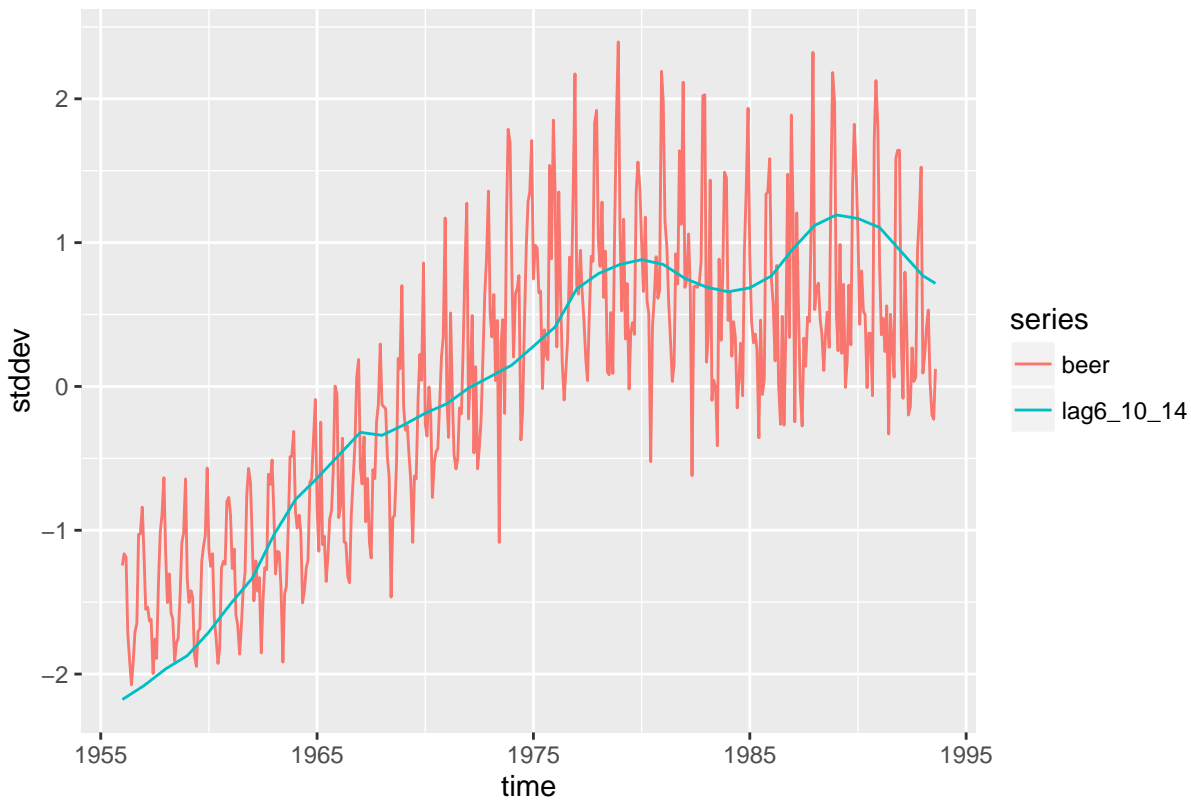
#Interpolate values using zoo library
zooSeries<-zoo(withNA, frequency=12)
wAppx<-na.approx(zooSeries, na.rm=FALSE)
monthlyLag6<-wAppx

#Interpolate long series for forecasting
zooSeriesLong<-zoo(withNALong, frequency=12)
wAppxLong<-data.frame(na.approx(zooSeriesLong, na.rm=FALSE))
rownames(wAppxLong)<-round(seq(from=(1921+6), length.out=nrow(wAppxLong), by=1/12),2)
monthlyLag6Long<-wAppxLong

#Reattach to beer numbers for plotting
beerPopMonth<-data.frame(beer=beerTS, lag6_10_14=monthlyLag6[1:length(beerTS)])
rownames(beerPopMonth)<-round(seq(from=1956, length.out=length(beerTS), by=1/12),2)
scaleMonth<-scale(beerPopMonth)
scaleMonthMelt<-melt(scaleMonth, variable.name="series")
colnames(scaleMonthMelt)<-c("time", "series", "stddev")

#Make a pretty plot
newPlot<-ggplot(scaleMonthMelt, aes(time,stddev)) + geom_line(aes(colour = series)) +ggtitle(paste("Lag 6 of 1
print(newPlot)
```

Lag 6 of 10–14 Pop and Beer Prod



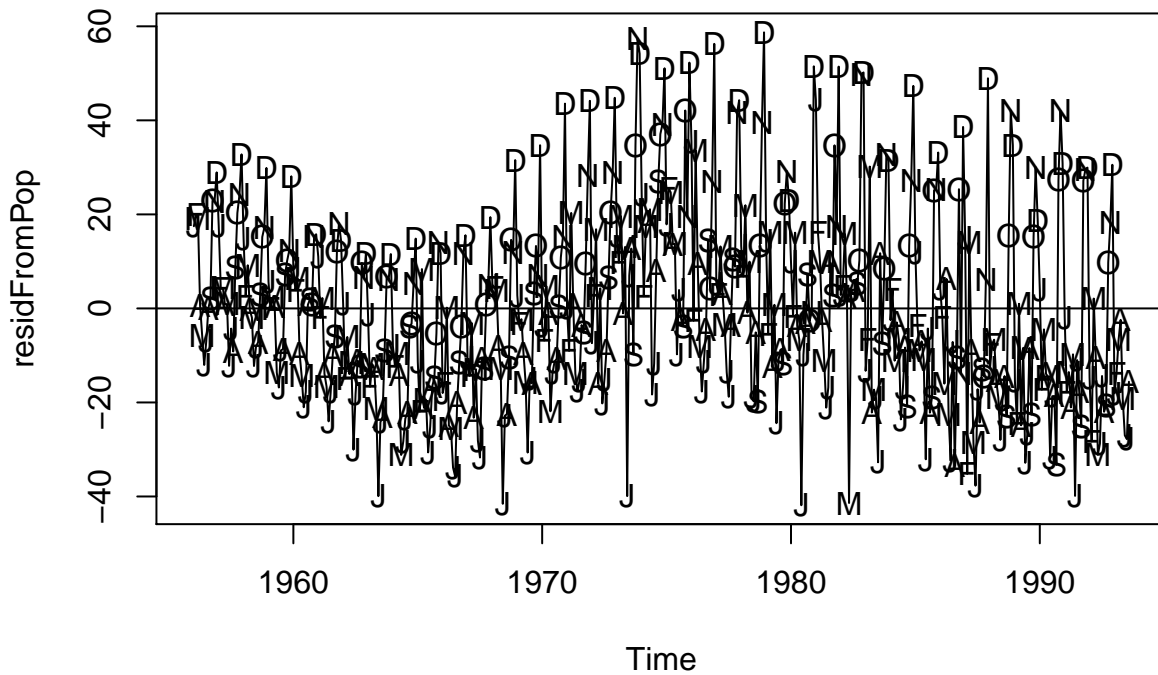
#Try a model

```
monthlyPopModel<-lm(beer ~ lag6_10_14, data=beerPopMonth)
summary(monthlyPopModel)
```

```
##
## Call:
## lm(formula = beer ~ lag6_10_14, data = beerPopMonth)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -41.862 -14.399  -2.561  13.099  58.717
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.919e+00  4.554e+00   1.519   0.129
## lag6_10_14   1.180e-04  4.076e-06  28.946 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.25 on 450 degrees of freedom
## Multiple R-squared:  0.6506, Adjusted R-squared:  0.6498
## F-statistic: 837.9 on 1 and 450 DF, p-value: < 2.2e-16
```

```
residFromPop<-ts(residuals(monthlyPopModel), frequency=12, start=c(1956,1))
plot(residFromPop, type="l", main="Plot of residuals from Pop model")
points(y=residFromPop, x=time(residFromPop), pch=as.vector(season(residFromPop)))
abline(h=0)
```

Plot of residuals from Pop model



Define helper functions

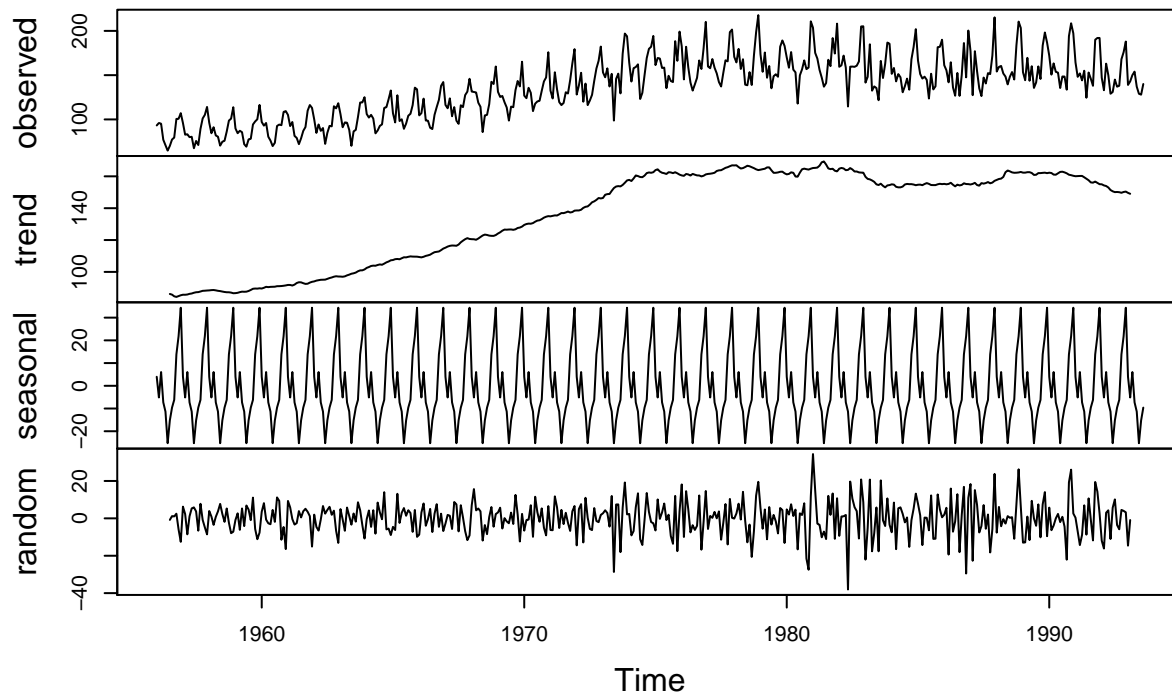
```
getModelStateString<-function(model) {
  modSpec<-model$arma
  modelString<-paste("SARIMA(", modSpec[1], ",", modSpec[6], ",", modSpec[2], ")(", modSpec[3], ",", modSpec[7],
  return(modelString)
}

plotResid<-function(model) {
  residuals<-ts(residuals(model), frequency=12, start=c(1956,1))
  modelString<-getModelStateString(model)
  par(mfrow=c(1,1))
  acf(residuals, main=paste("ACF of", modelString), lag.max=40, cex=.5)
  plot.new()
  pacf(residuals, main=paste("PACF of", modelString), lag.max=40, cex=.5)
  par(mfrow=c(1,1))
}
```

Decomposing the time series to see trends and patterns

```
decompbeer = decompose (beerTS, type="additive")
plot (decompbeer)
```

Decomposition of additive time series



```
#monthplot(beerTS, main="Decomposition of Series by Month")
```

Try to figure out deterministic trend

```
t<-1:length(beerTS)
t2<-t^2
t3<-t^3
t4<-t^4
t5<-t^5

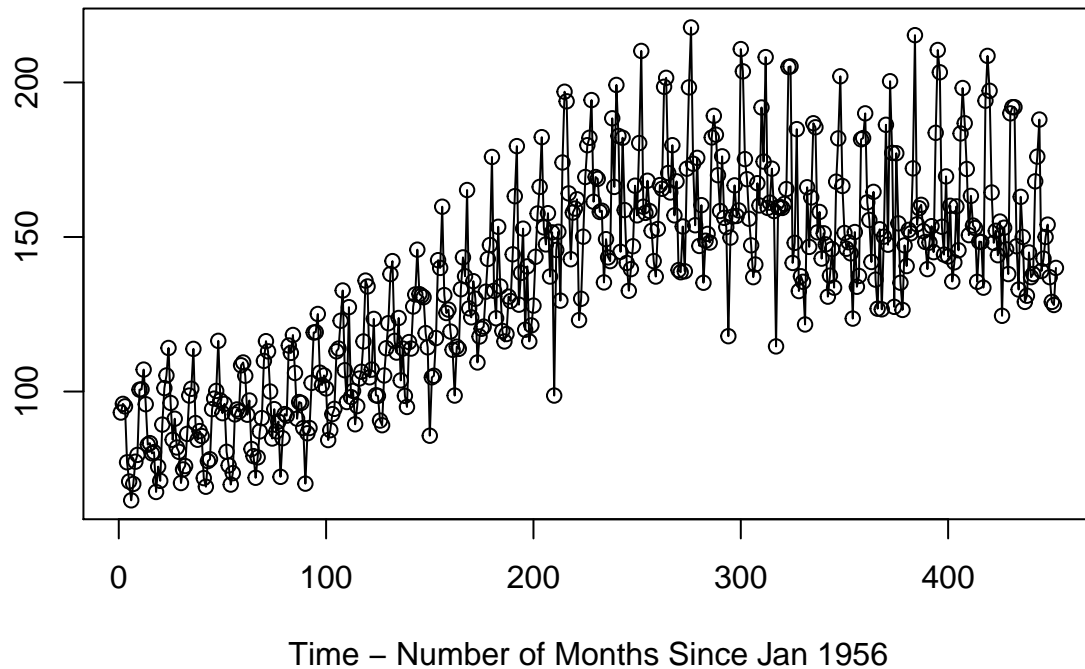
quadFit<-lm(beerTS~t+t2)
summary(quadFit)
```

```
##
## Call:
## lm(formula = beerTS ~ t + t2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -46.861 -14.133  -1.991  11.937  61.174
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.560e+01  2.828e+00  23.20  <2e-16 ***
## t            5.429e-01  2.883e-02  18.83  <2e-16 ***
## t2          -7.721e-04  6.163e-05 -12.53  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 19.95 on 449 degrees of freedom
## Multiple R-squared:  0.6616, Adjusted R-squared:  0.6601
## F-statistic: 439 on 2 and 449 DF,  p-value: < 2.2e-16

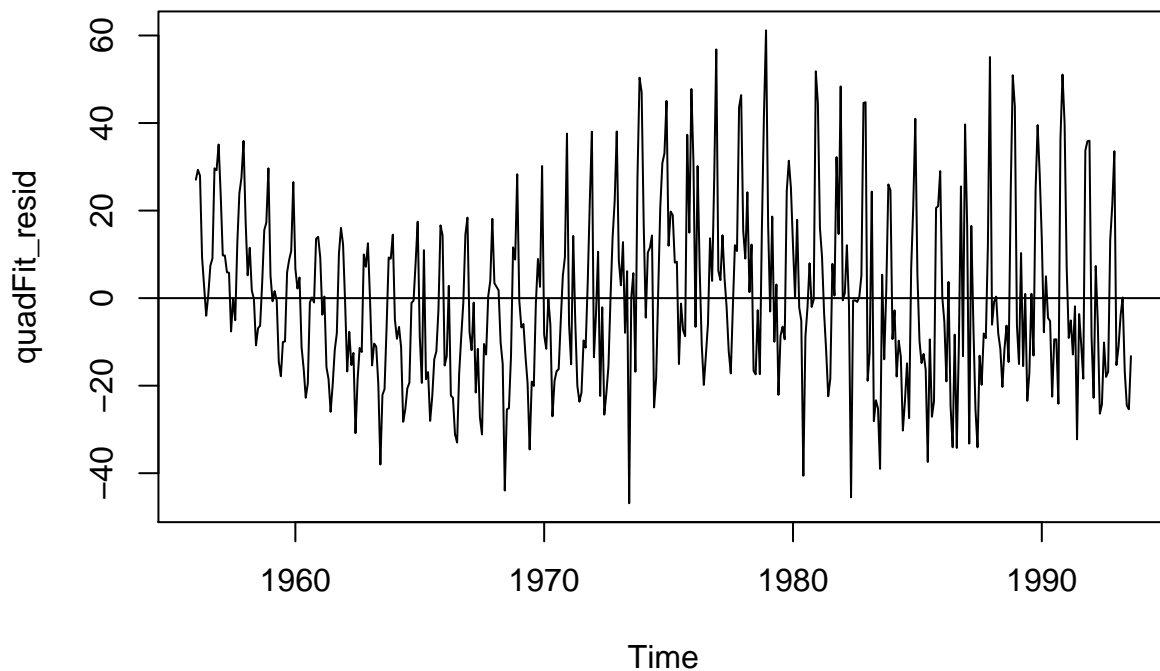
#### plot the data and the fitted quadratic trend function
plot(x=1:length(beerTS),y=beerTS,type='o',ylab="",xlab="Time - Number of Months Since Jan 1956",main="Quadratic Fit on Beer Production Data",
curve(expr = coef(quadFit)[1]+coef(quadFit)[2]*x+coef(quadFit)[3]*x^2+coef(quadFit)[4]*x^3,lty=1,add = TRUE,c
```

Quadratic Fit on Beer Production Data



```
quadFit_resid<-ts(residuals(quadFit),frequency=12, start=c(1956,1))
plot(quadFit_resid, main="Residuals from a Quadratic Trend Fit")
abline(h=0)
```


Residuals from a Quadratic Trend Fit



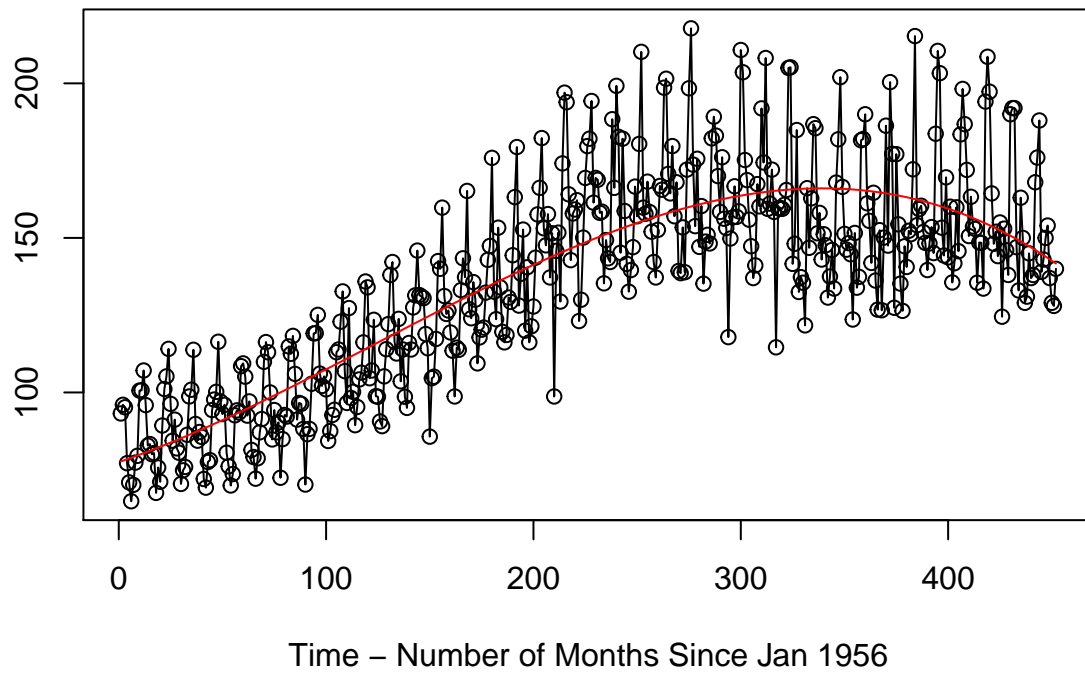
```
cubicFit<-lm(beerTS~t+t2+t3)
summary(cubicFit)
```

```
##
## Call:
## lm(formula = beerTS ~ t + t2 + t3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -50.660 -13.783  -2.601  12.434  57.639
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.745e+01  3.695e+00  20.963  < 2e-16 ***
## t            2.307e-01  7.056e-02   3.270  0.00116 **
## t2           9.490e-04  3.617e-04   2.624  0.00900 **
## t3          -2.533e-06  5.249e-07  -4.826  1.92e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.47 on 448 degrees of freedom
## Multiple R-squared:  0.6784, Adjusted R-squared:  0.6762
## F-statistic: 315 on 3 and 448 DF, p-value: < 2.2e-16
```

```
#### plot the data and the fitted quadratic trend function
```

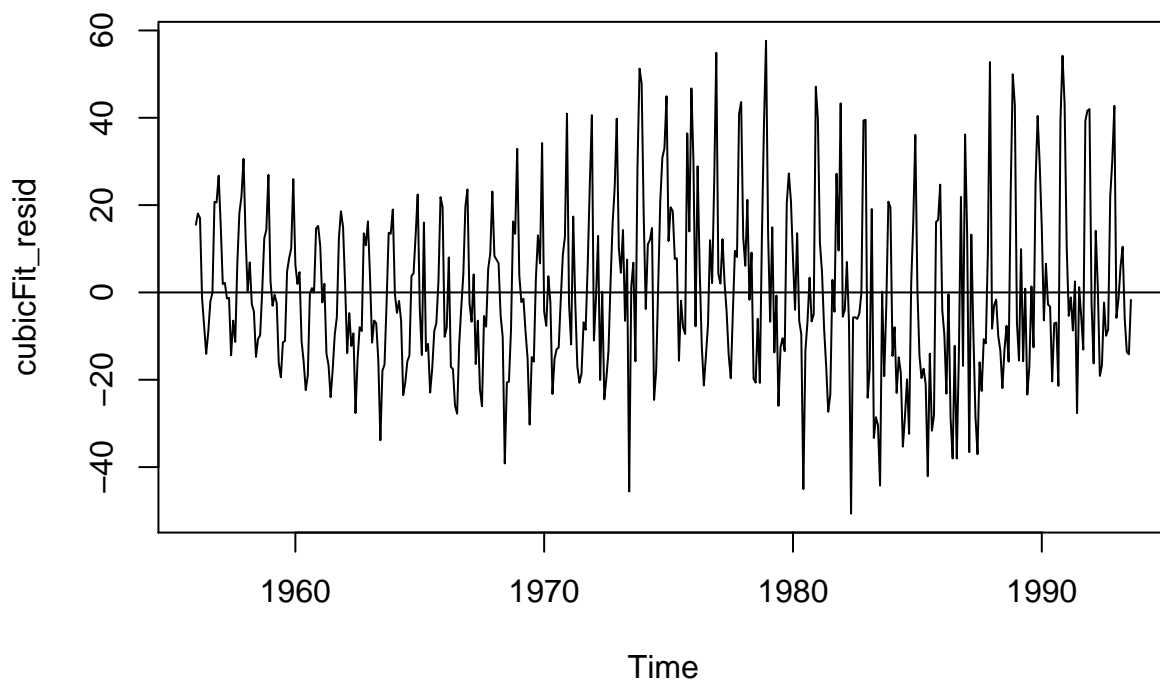
```
plot(x=1:length(beerTS),y=beerTS,type='o',ylab="",xlab="Time - Number of Months Since Jan 1956",main="Cubic Fit",
curve(expr = coef(cubicFit)[1]+coef(cubicFit)[2]*x+coef(cubicFit)[3]*x^2+coef(cubicFit)[4]*x^3,lty=1,add = TRUE))
```

Cubic Fit on Beer Production Data



```
cubicFit_resid<-ts(residuals(cubicFit),frequency=12, start=c(1956,1))  
plot(cubicFit_resid, main="Residuals from a Cubic Trend Fit")  
abline(h=0)
```

Residuals from a Cubic Trend Fit



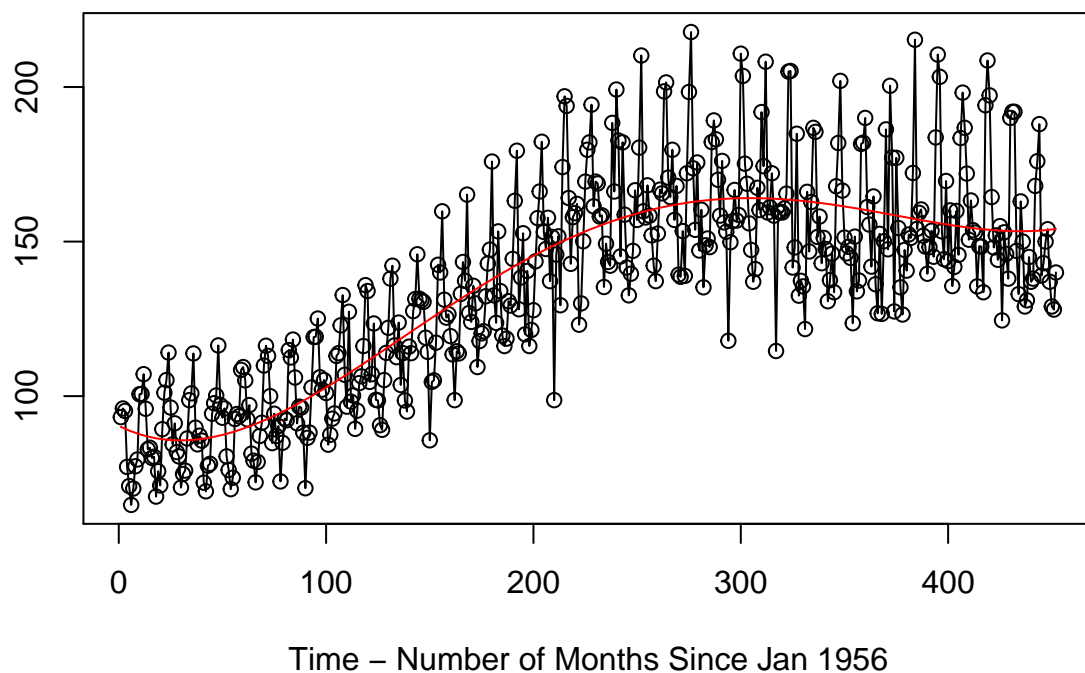
```
order4polyFit<-lm(beerTS~t+t2+t3+t4)
summary(order4polyFit)
```

```
##
## Call:
## lm(formula = beerTS ~ t + t2 + t3 + t4)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -50.079 -12.721  -3.199  10.135  57.983
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.037e+01  4.536e+00  19.924  < 2e-16 ***
## t           -3.341e-01  1.384e-01  -2.414   0.0162 *
## t2           6.545e-03  1.241e-03   5.276  2.07e-07 ***
## t3          -2.173e-05  4.113e-06  -5.285  1.97e-07 ***
## t4           2.119e-08  4.504e-09   4.706  3.38e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.03 on 447 degrees of freedom
## Multiple R-squared:  0.6935, Adjusted R-squared:  0.6908
## F-statistic: 252.9 on 4 and 447 DF,  p-value: < 2.2e-16
```

```
#### plot the data and the fitted 4th order polynomial trend function
```

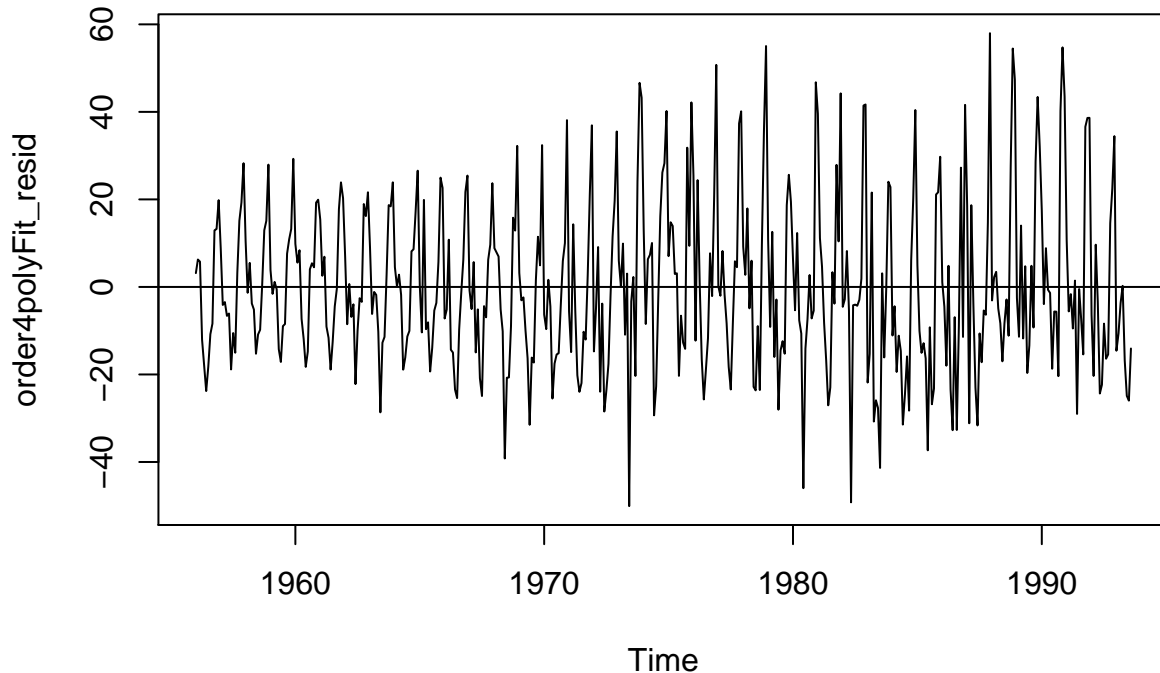
```
plot(x=1:length(beerTS),y=beerTS,type='o',ylab="",xlab="Time - Number of Months Since Jan 1956",main="order4poly
curve(expr = coef(order4polyFit)[1]+coef(order4polyFit)[2]*x+coef(order4polyFit)[3]*x^2+coef(order4polyFit)[4]
```

order4poly Fit on Beer Production Data



```
order4polyFit_resid<-ts(residuals(order4polyFit),frequency=12, start=c(1956,1))
plot(order4polyFit_resid, main="Residuals from a order4poly Trend Fit")
abline(h=0)
```

Residuals from a order4poly Trend Fit



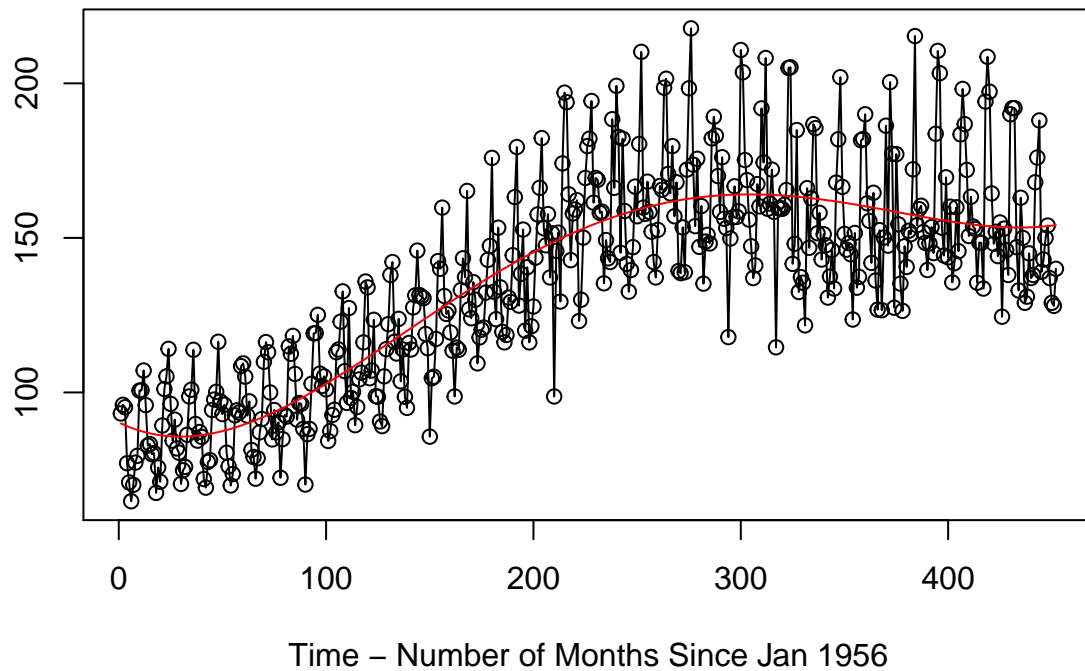
```
order5polyFit<-lm(beerTS~t+t2+t3+t4+t5)
summary(order5polyFit)
```

```
##
## Call:
## lm(formula = beerTS ~ t + t2 + t3 + t4 + t5)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -50.069 -12.729  -3.179  10.132  58.012
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.029e+01  5.483e+00  16.469  <2e-16 ***
## t            -3.288e-01  2.436e-01  -1.350   0.1778
## t2             6.463e-03  3.323e-03   1.945   0.0524 .
## t3            -2.126e-05  1.858e-05  -1.144   0.2532
## t4             2.000e-08  4.519e-08   0.443   0.6582
## t5             1.049e-12  3.970e-11   0.026   0.9789
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.05 on 446 degrees of freedom
## Multiple R-squared:  0.6935, Adjusted R-squared:  0.6901
## F-statistic: 201.9 on 5 and 446 DF, p-value: < 2.2e-16
```

```
#### plot the data and the fitted 5th order polynomial trend function
```

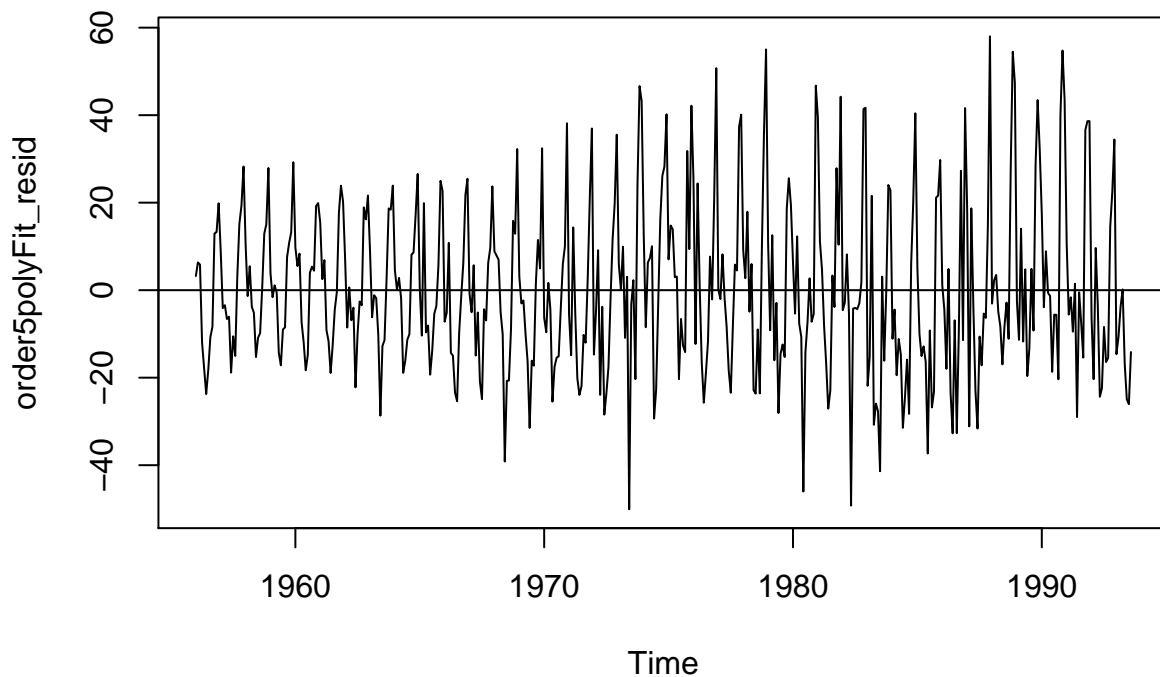
```
plot(x=1:length(beerTS),y=beerTS,type='o',ylab="",xlab="Time - Number of Months Since Jan 1956",main="order5poly  
curve(expr = coef(order5polyFit)[1]+coef(order5polyFit)[2]*x+coef(order5polyFit)[3]*x^2+coef(order5polyFit)[4]
```

order5poly Fit on Beer Production Data



```
order5polyFit_resid<-ts(residuals(order5polyFit),frequency=12, start=c(1956,1))  
plot(order5polyFit_resid, main="Residuals from a order5poly Trend Fit")  
abline(h=0)
```

Residuals from a order5poly Trend Fit



Fit deterministic cosine trend model

```
startDate<-round(start(beer_forecast)[1]+start(beer_forecast)[2]/12,2)
endDate<-round(end(beer_forecast)[1]+end(beer_forecast)[2]/12,2)
numToFor<-length(beer_forecast)

allBeerData<-ts(c(beerTS, beer_forecast), start=c(1956, 1), frequency=12)

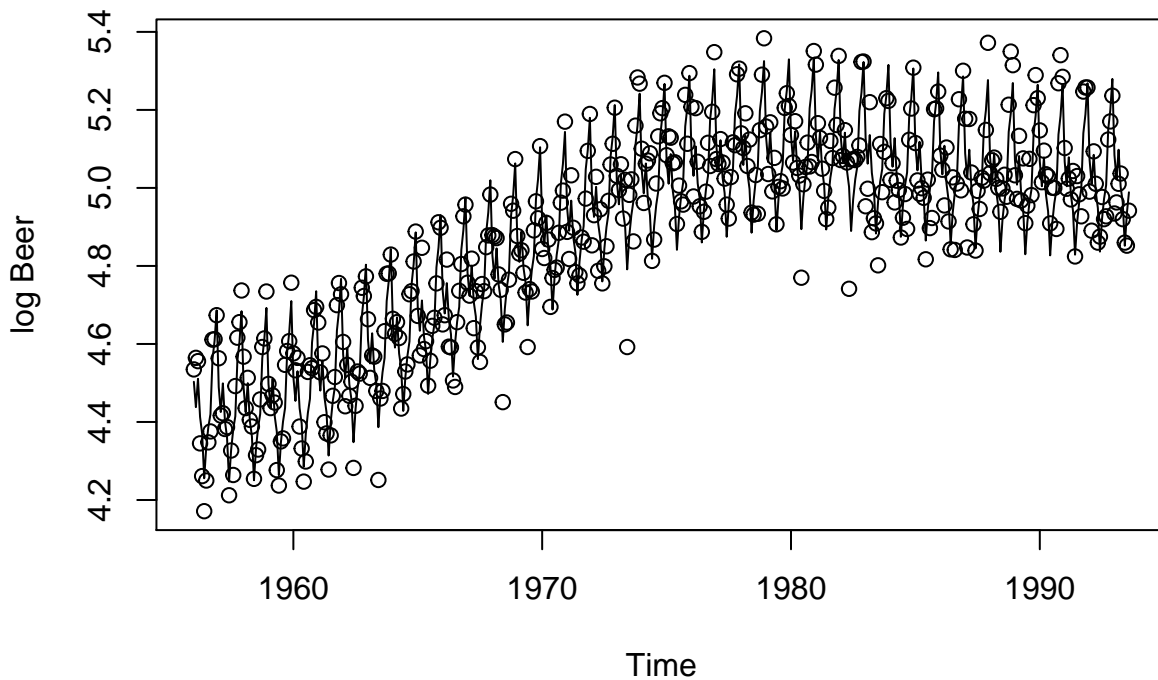
tnew<-1:length(allBeerData)
t2new<-tnew^2
t3new<-tnew^3
t4new<-tnew^4

har.=harmonic(logBeer,5)
modelHR=lm(logBeer~har.+t+t2+t3+t4)
summary(modelHR)
```

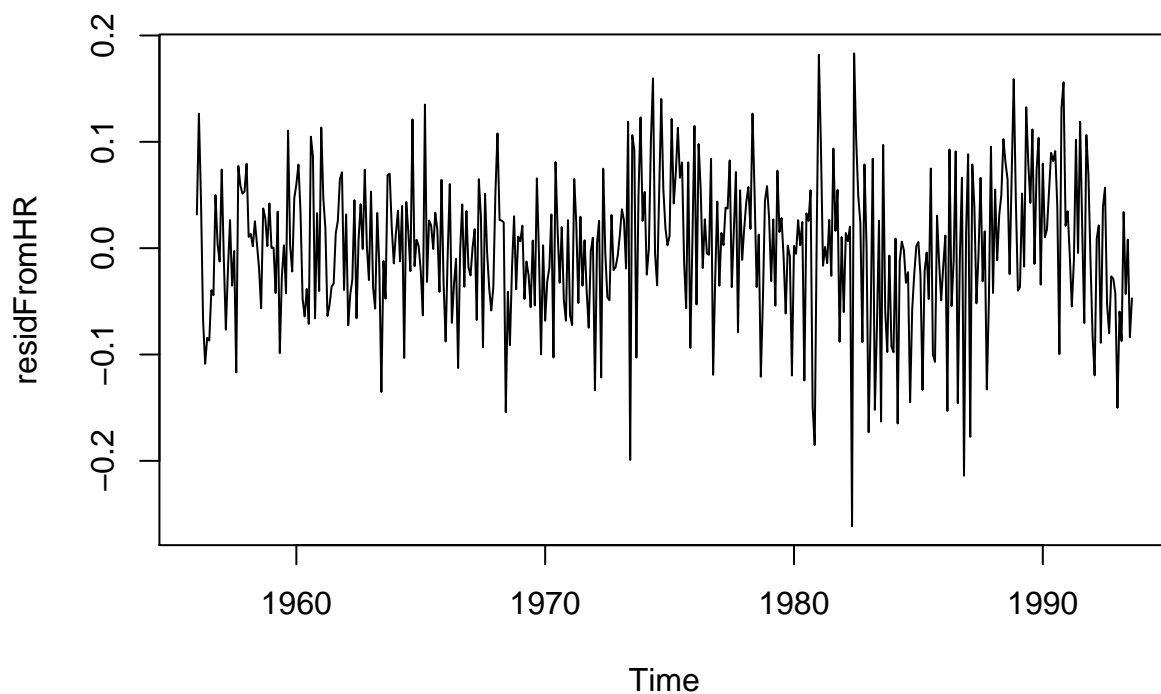
```
##
## Call:
## lm(formula = logBeer ~ har. + t + t2 + t3 + t4)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.261463 -0.039931  0.002551  0.043469  0.183185
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.467e+00  1.642e-02 272.071  < 2e-16 ***
## har.cos(2*pi*t) 1.228e-01  4.572e-03  26.854  < 2e-16 ***
```

```
## har.cos(4*pi*t) -3.543e-02 4.577e-03 -7.742 6.87e-14 ***
## har.cos(6*pi*t) -2.830e-02 4.579e-03 -6.180 1.47e-09 ***
## har.cos(8*pi*t) -5.185e-03 4.576e-03 -1.133 0.257829
## har.cos(10*pi*t) -1.557e-02 4.571e-03 -3.406 0.000720 ***
## har.sin(2*pi*t) -8.192e-02 4.589e-03 -17.850 < 2e-16 ***
## har.sin(4*pi*t) -2.568e-02 4.581e-03 -5.605 3.70e-08 ***
## har.sin(6*pi*t) -3.967e-02 4.579e-03 -8.663 < 2e-16 ***
## har.sin(8*pi*t) -1.067e-02 4.581e-03 -2.328 0.020366 *
## har.sin(10*pi*t) -3.546e-02 4.586e-03 -7.733 7.30e-14 ***
## t -1.797e-03 5.009e-04 -3.587 0.000372 ***
## t2 4.927e-05 4.490e-06 10.973 < 2e-16 ***
## t3 -1.745e-07 1.488e-08 -11.724 < 2e-16 ***
## t4 1.785e-10 1.630e-11 10.952 < 2e-16 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06882 on 437 degrees of freedom
## Multiple R-squared: 0.9365, Adjusted R-squared: 0.9345
## F-statistic: 460.5 on 14 and 437 DF, p-value: < 2.2e-16
```

```
plot(ts(fitted(modelHR),freq=12,start=c(1956,1)),ylab='log Beer',type='l',ylim=range(c(fitted(modelHR),logBeer
points(logBeer)
```



```
residFromHR<-ts(residuals(modelHR), frequency=12, start=c(1956,1))
plot(residFromHR)
```



```
chosenMod<-modelHR
library(TSA)

har.=harmonic(log(beer_forecast),5)
newRegData<-data.frame(t=tnew, t2=t2new, t3=t3new, t4=t4new)
newRegData<-newRegData[(nrow(newRegData)-numToFor+1):nrow(newRegData),]
newRegData<-data.frame(har., newRegData)
colnames(newRegData)<-c(colnames(har.), "t", "t2", "t3", "t4")
```

Diagnostics on Cosine Trend

```
adf.test(residFromHR)
```

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

```
##
## Augmented Dickey-Fuller Test
##
## data: residFromHR
## Dickey-Fuller = -5.3975, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
pp.test(residFromHR)
```

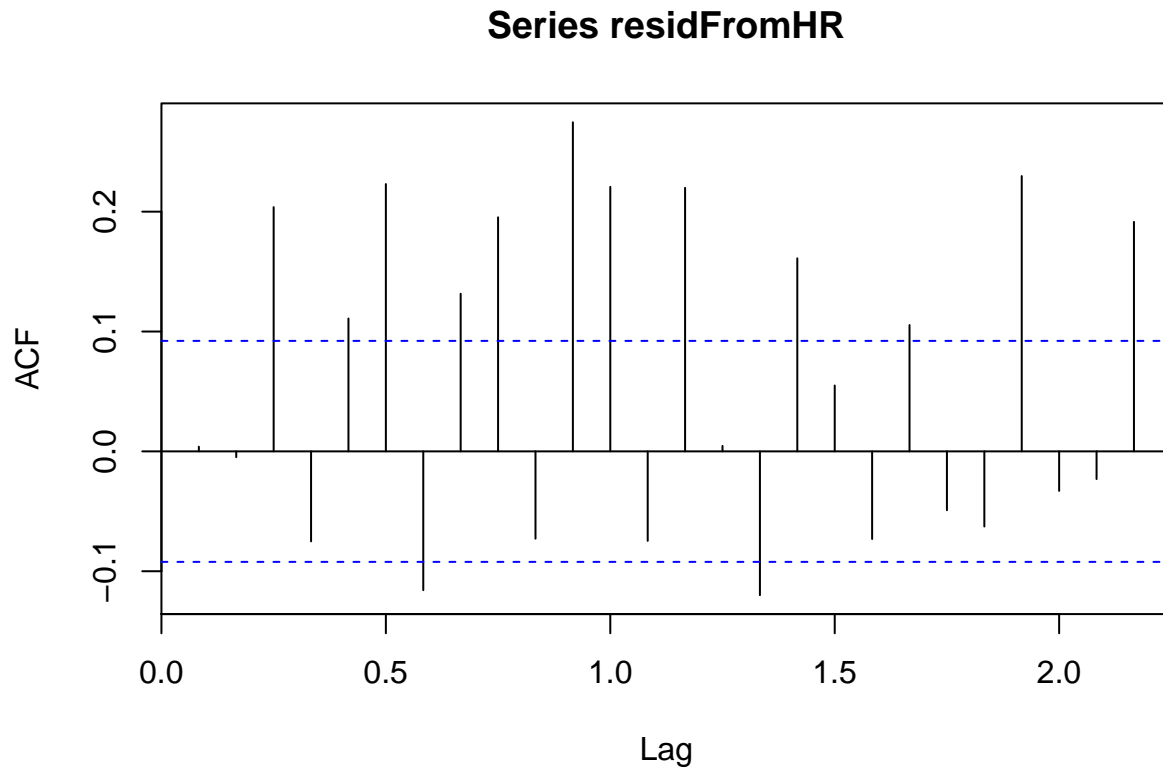
```
## Warning in pp.test(residFromHR): p-value smaller than printed p-value
```

```
##
```



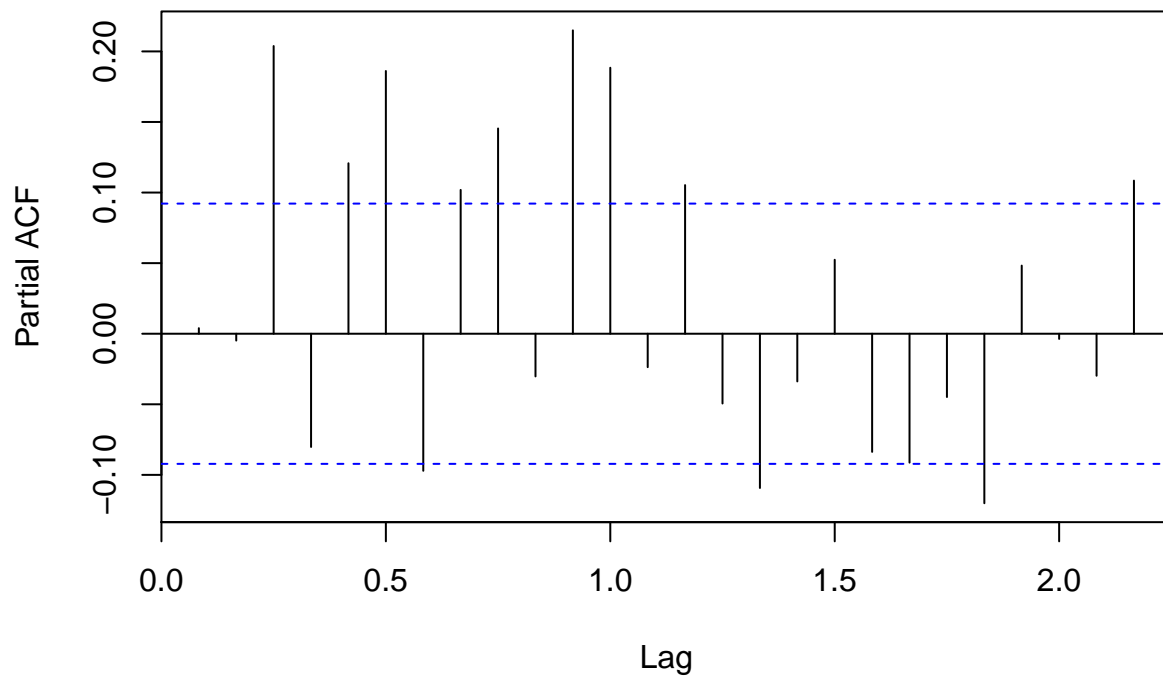
```
## Phillips-Perron Unit Root Test
##
## data: residFromHR
## Dickey-Fuller Z(alpha) = -490.03, Truncation lag parameter = 5,
## p-value = 0.01
## alternative hypothesis: stationary
```

```
acf(residFromHR)
```



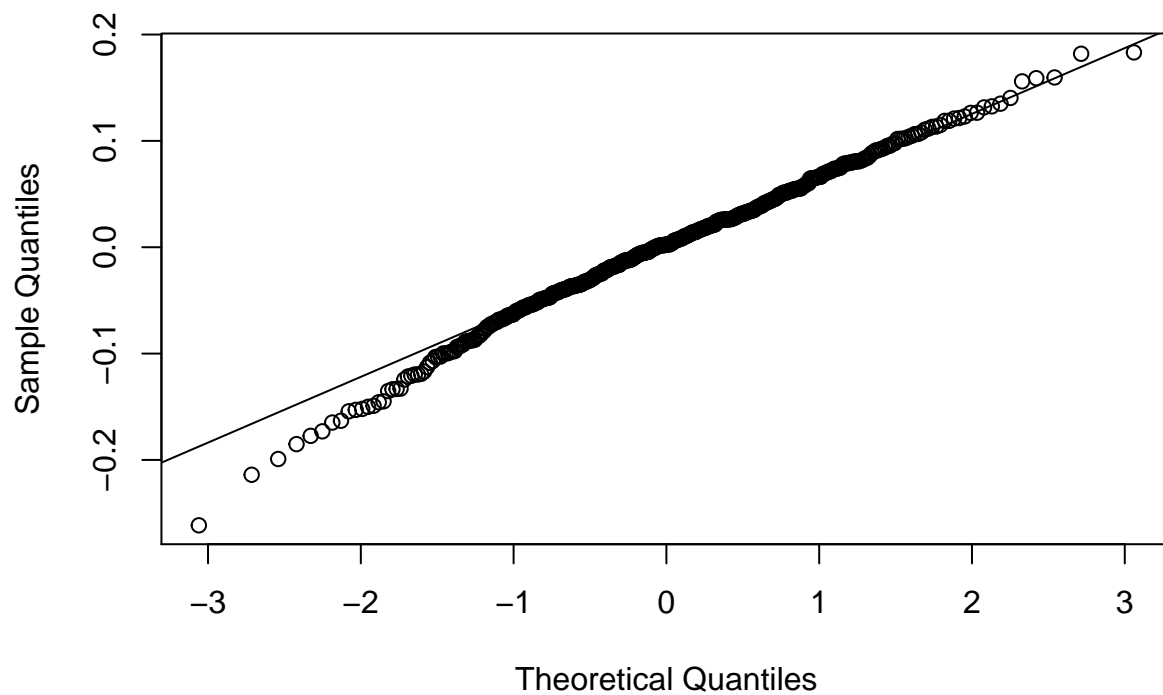
```
pacf(residFromHR)
```

Series residFromHR



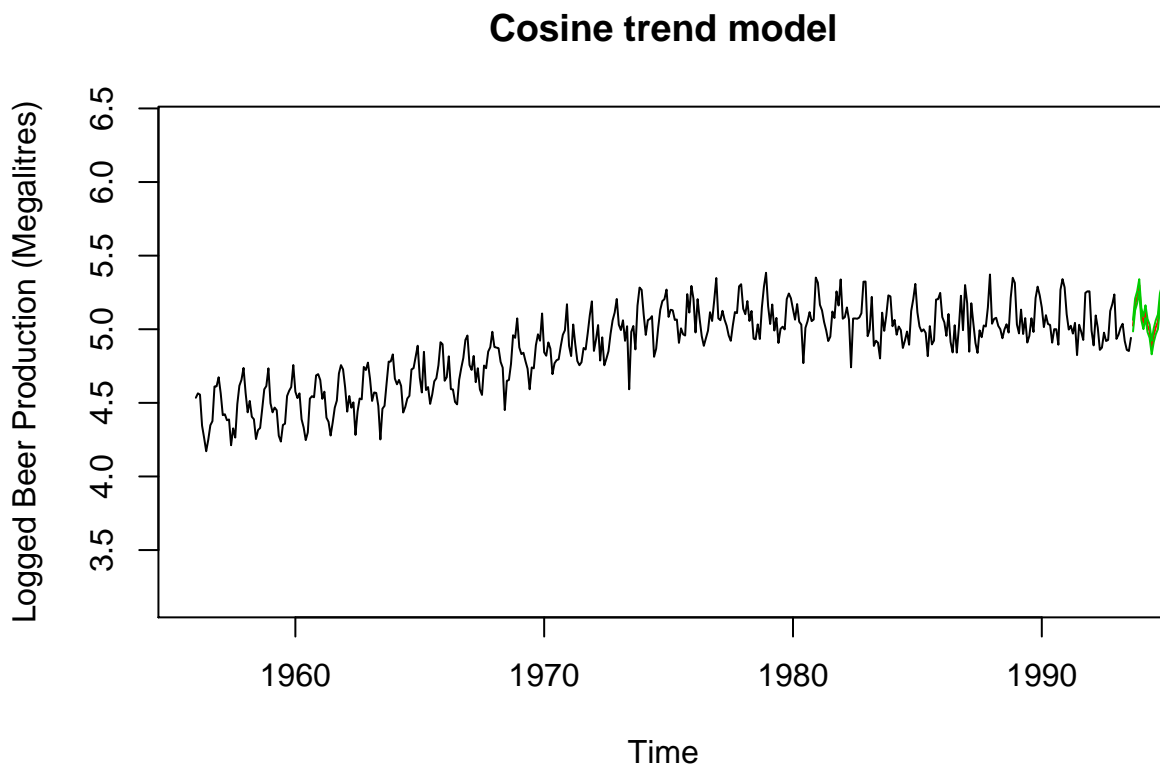
```
qqnorm(residFromHR)  
qqline(residFromHR)
```

Normal Q-Q Plot



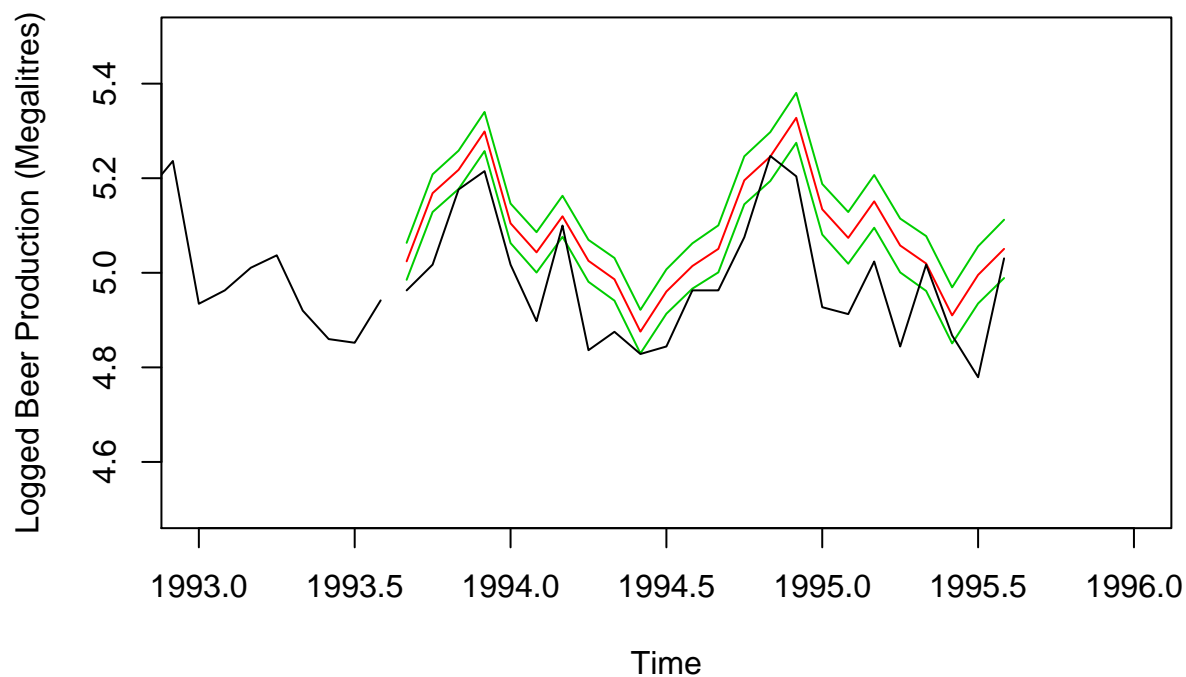
Forecast based on cosine trend model

```
modelString<-"Cosine trend model"
predictions<-predict(chosenMod,newdata =newRegData,se.fit = T)
pred<-ts(predictions$fit,start = c(1993,9),frequency = 12)
uci<-ts(pred+2*predictions$se.fit,start = c(1993,9),frequency = 12)
lci<-ts(pred-2*predictions$se.fit,start = c(1993,9),frequency = 12)
ymin=min(c(as.vector(lci),logBeer))-1
ymax=max(c(as.vector(uci),logBeer))+1
plot(logBeer,ylim=c(ymin,ymax),main=modelString, ylab='Logged Beer Production (Megalitres)')
lines(pred,col=2)
lines(uci,col=3)
lines(lci,col=3)
```



```
ymin=min(c(as.vector(lci),logBeer))-1
ymax=max(c(as.vector(uci),logBeer))+1
plot(logBeer,xlim=c(1993, 1996), ylim=c(4.5,5.5),main=modelString, ylab='Logged Beer Production (Megalitres)')
lines(pred,col=2)
lines(uci,col=3)
lines(lci,col=3)
lines(log(beer_forecast), col="black")
```

Cosine trend model



Seasonal means model with ARIMA

```
logBeer<-log(beerTS)
t<-1:length(logBeer)
t2<-t^2
t3<-t^3
t4<-t^4
month<-season(logBeer)

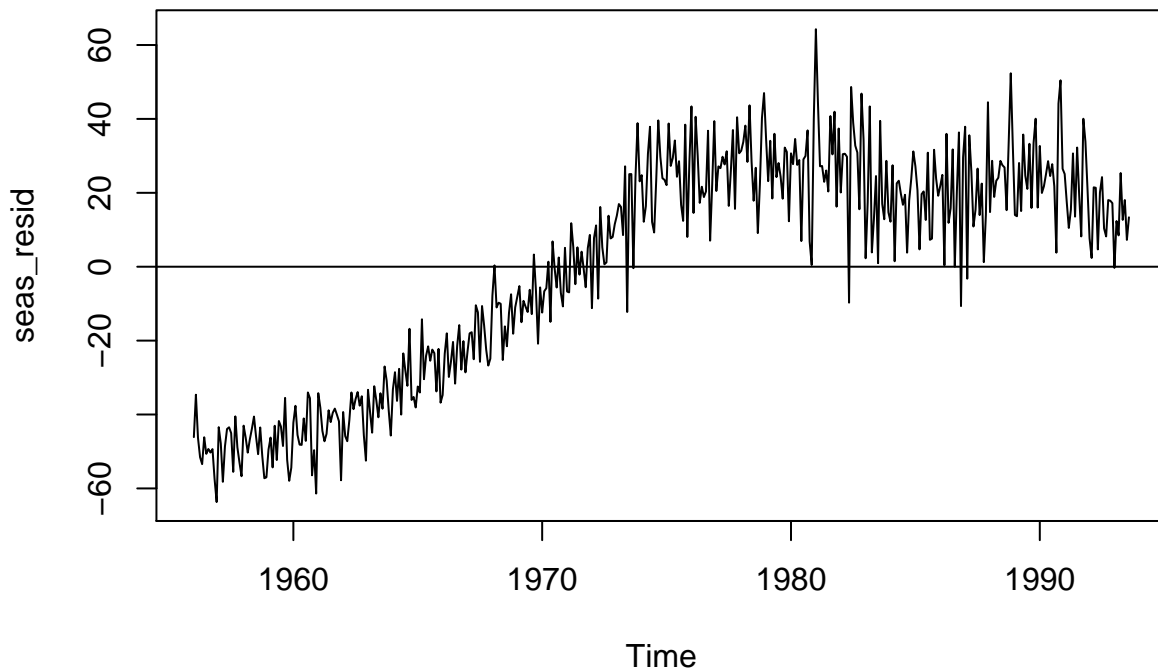
seasMod<-lm(beerTS~month)
summary(seasMod)
```

```
##
## Call:
## lm(formula = beerTS ~ month)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -63.70 -27.90   8.20  24.91  64.28
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   139.316     4.957   28.106 < 2e-16 ***
## monthFebruary    -8.674     7.010   -1.237  0.21662
## monthMarch        2.203     7.010    0.314  0.75350
## monthApril     -10.674     7.010   -1.523  0.12856
## monthMay       -14.989     7.010   -2.138  0.03304 *
## monthJune     -28.361     7.010   -4.046 6.16e-05 ***
## monthJuly      -18.571     7.010   -2.649  0.00836 **
```

```
## monthAugust      -12.687      7.010   -1.810   0.07100 .
## monthSeptember   -9.516      7.057   -1.348   0.17822
## monthOctober     10.641      7.057    1.508   0.13231
## monthNovember    18.830      7.057    2.668   0.00791 **
## monthDecember    31.479      7.057    4.461  1.04e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 30.56 on 440 degrees of freedom
## Multiple R-squared:  0.2224, Adjusted R-squared:  0.2029
## F-statistic: 11.44 on 11 and 440 DF,  p-value: < 2.2e-16
```

```
seas_resid<-ts(residuals(seasMod),frequency=12, start=c(1956,1))
plot(seas_resid, main="Residuals from Seasonal Means Model")
abline(h=0)
```

Residuals from Seasonal Means Model



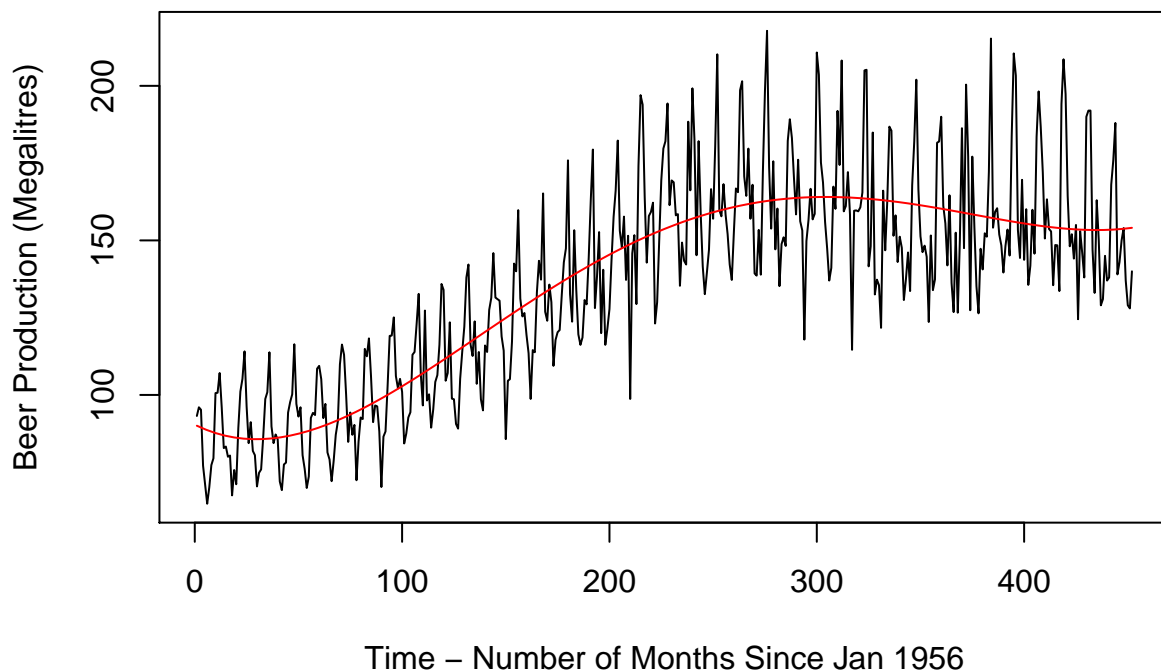
```
order4polyFit<-lm(beerTS~t+t2+t3+t4)
summary(order4polyFit)
```

```
##
## Call:
## lm(formula = beerTS ~ t + t2 + t3 + t4)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -50.079 -12.721  -3.199  10.135  57.983
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.037e+01  4.536e+00  19.924 < 2e-16 ***
## t            -3.341e-01  1.384e-01  -2.414  0.0162 *
```

```
## t2          6.545e-03  1.241e-03   5.276 2.07e-07 ***
## t3         -2.173e-05  4.113e-06  -5.285 1.97e-07 ***
## t4          2.119e-08  4.504e-09   4.706 3.38e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.03 on 447 degrees of freedom
## Multiple R-squared:  0.6935, Adjusted R-squared:  0.6908
## F-statistic: 252.9 on 4 and 447 DF,  p-value: < 2.2e-16
```

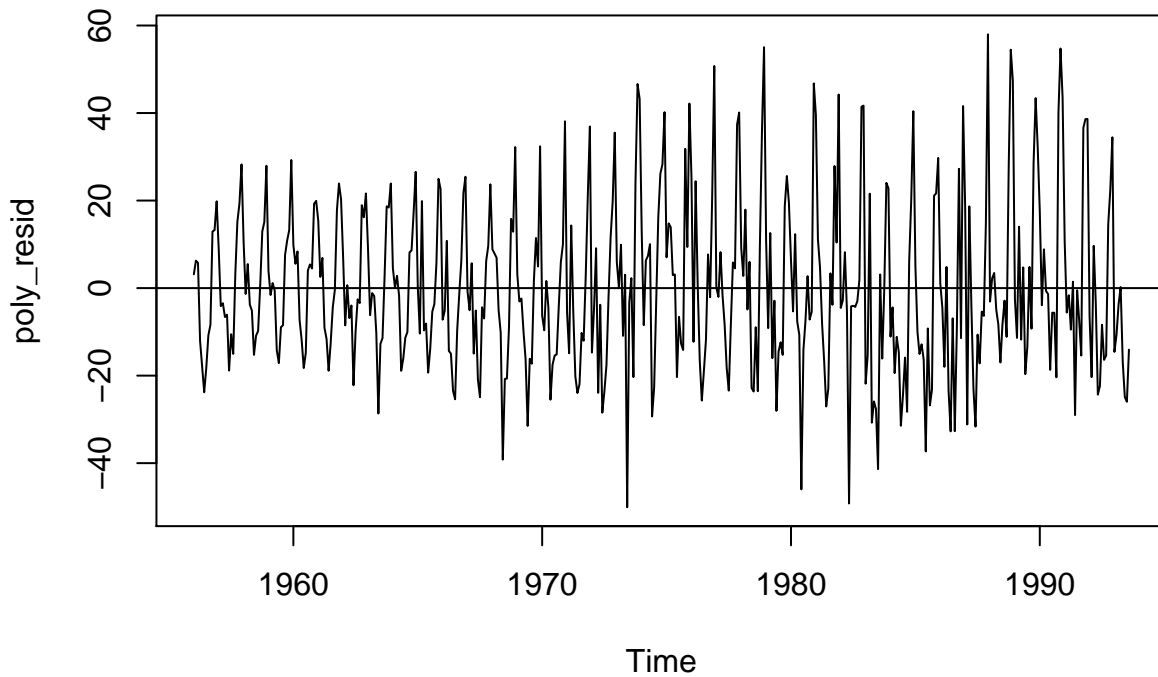
```
#### plot the data and the fitted 4th order polynomial trend function
plot(x=1:length(beerTS),y=beerTS,type='l',ylab="Beer Production (Megalitres)",xlab="Time - Number of Months Since Jan 1956",
curve(expr = coef(order4polyFit)[1]+coef(order4polyFit)[2]*x+coef(order4polyFit)[3]*x^2+coef(order4polyFit)[4]*x^3+coef(order4polyFit)[5]*x^4))
```

order4poly Fit on Beer Production Data



```
poly_resid<-ts(residuals(order4polyFit),frequency=12, start=c(1956,1))
plot(poly_resid, main="Residuals from 4th order polynomial Model")
abline(h=0)
```

Residuals from 4th order polynomial Model



```
logSeasPoly<-lm(logBeer~t+t2+t3+t4+month)
summary(logSeasPoly)
```

```
##
## Call:
## lm(formula = logBeer ~ t + t2 + t3 + t4 + month)
##
## Residuals:
```

	Min	1Q	Median	3Q	Max
	-0.262750	-0.039816	0.003297	0.043475	0.184483

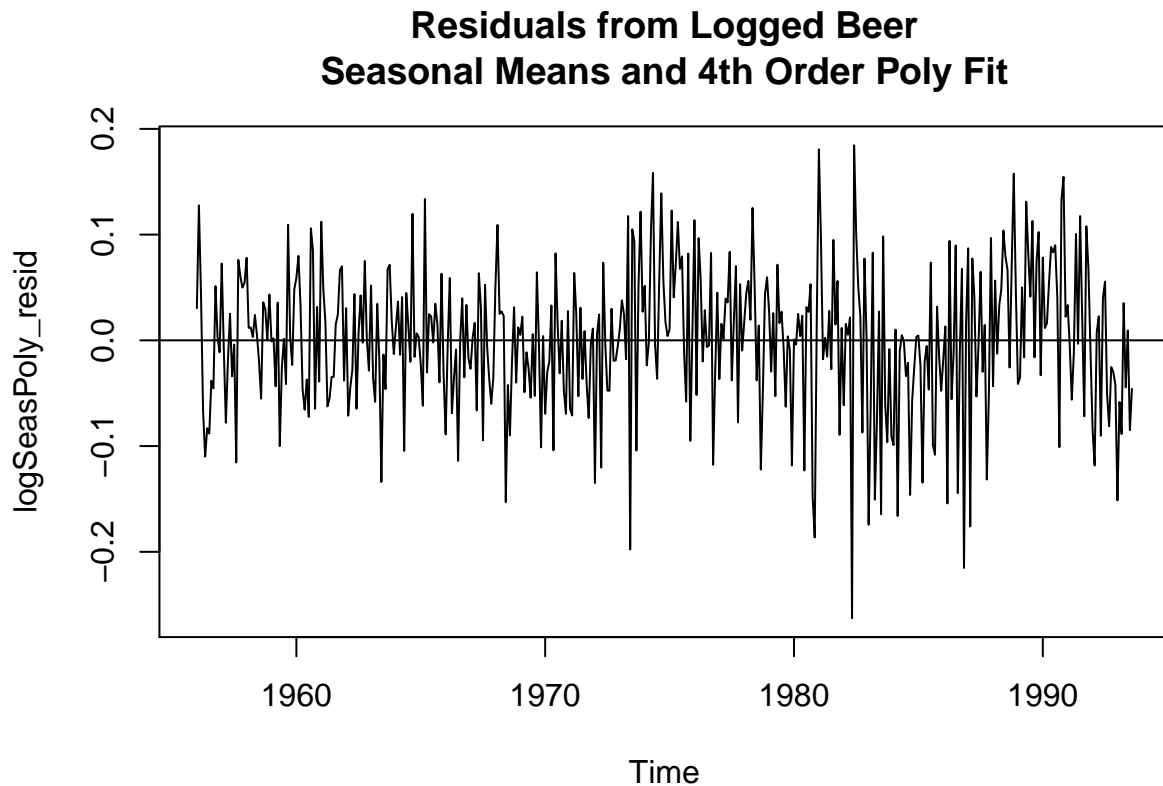
```
##
## Coefficients:
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	4.506e+00	1.945e-02	231.675	< 2e-16 ***
t	-1.796e-03	5.014e-04	-3.583	0.000378 ***
t2	4.926e-05	4.494e-06	10.962	< 2e-16 ***
t3	-1.745e-07	1.490e-08	-11.713	< 2e-16 ***
t4	1.785e-10	1.631e-11	10.941	< 2e-16 ***
monthFebruary	-6.602e-02	1.580e-02	-4.178	3.56e-05 ***
monthMarch	1.069e-02	1.580e-02	0.676	0.499268
monthApril	-8.834e-02	1.581e-02	-5.590	4.02e-08 ***
monthMay	-1.271e-01	1.581e-02	-8.042	8.39e-15 ***
monthJune	-2.427e-01	1.581e-02	-15.354	< 2e-16 ***
monthJuly	-1.578e-01	1.581e-02	-9.984	< 2e-16 ***
monthAugust	-1.089e-01	1.581e-02	-6.890	1.96e-11 ***
monthSeptember	-7.261e-02	1.591e-02	-4.563	6.57e-06 ***
monthOctober	6.706e-02	1.591e-02	4.213	3.06e-05 ***
monthNovember	1.172e-01	1.592e-02	7.363	9.03e-13 ***
monthDecember	1.936e-01	1.592e-02	12.164	< 2e-16 ***

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

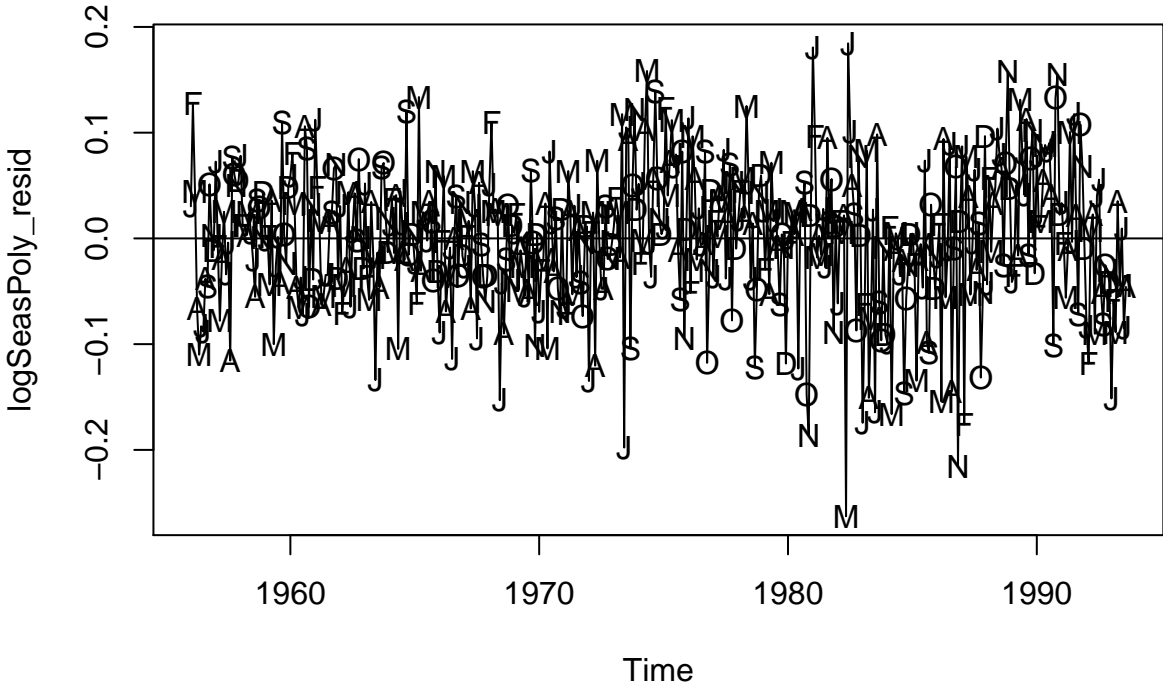
```
##
## Residual standard error: 0.06889 on 436 degrees of freedom
## Multiple R-squared:  0.9365, Adjusted R-squared:  0.9344
## F-statistic: 429 on 15 and 436 DF,  p-value: < 2.2e-16
```

```
logSeasPoly_resid<-ts(residuals(logSeasPoly),frequency=12, start=c(1956,1))
plot(logSeasPoly_resid, main="Residuals from Logged Beer\nSeasonal Means and 4th Order Poly Fit")
abline(h=0)
```



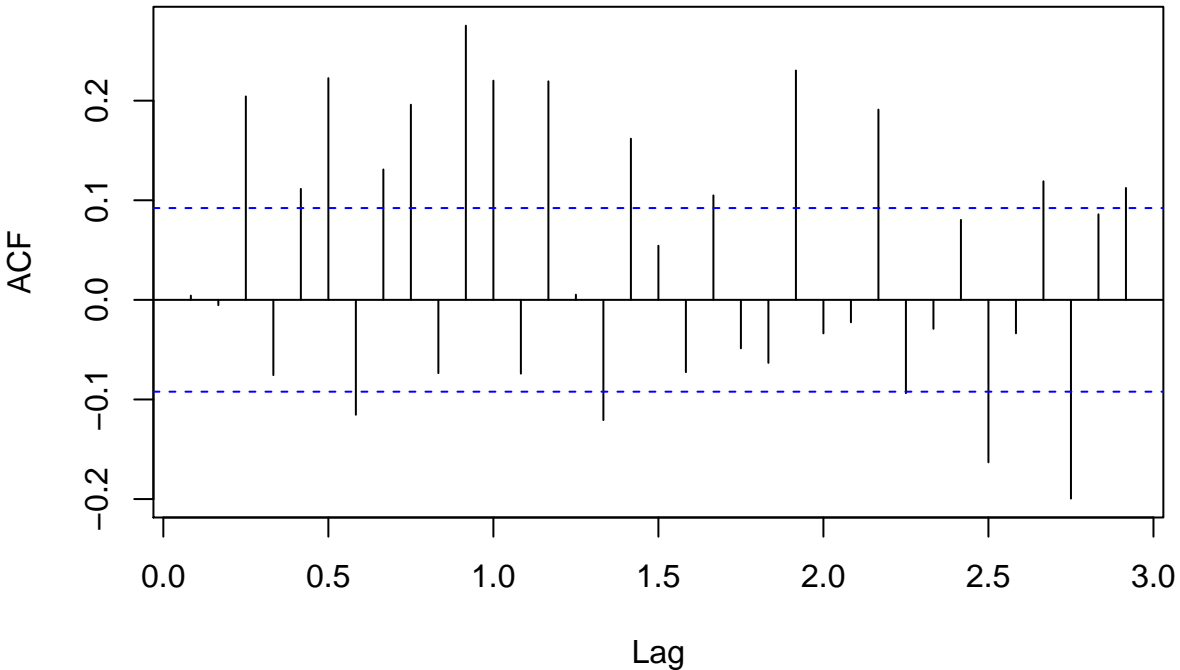
```
plot(logSeasPoly_resid, main="Residuals from Logged Beer\nSeasonal Means and 4th Order Poly Fit", type="l")
points(y=logSeasPoly_resid, x=time(logSeasPoly_resid), pch=as.vector(season(logSeasPoly_resid)))
abline(h=0)
```


Residuals from Logged Beer Seasonal Means and 4th Order Poly Fit



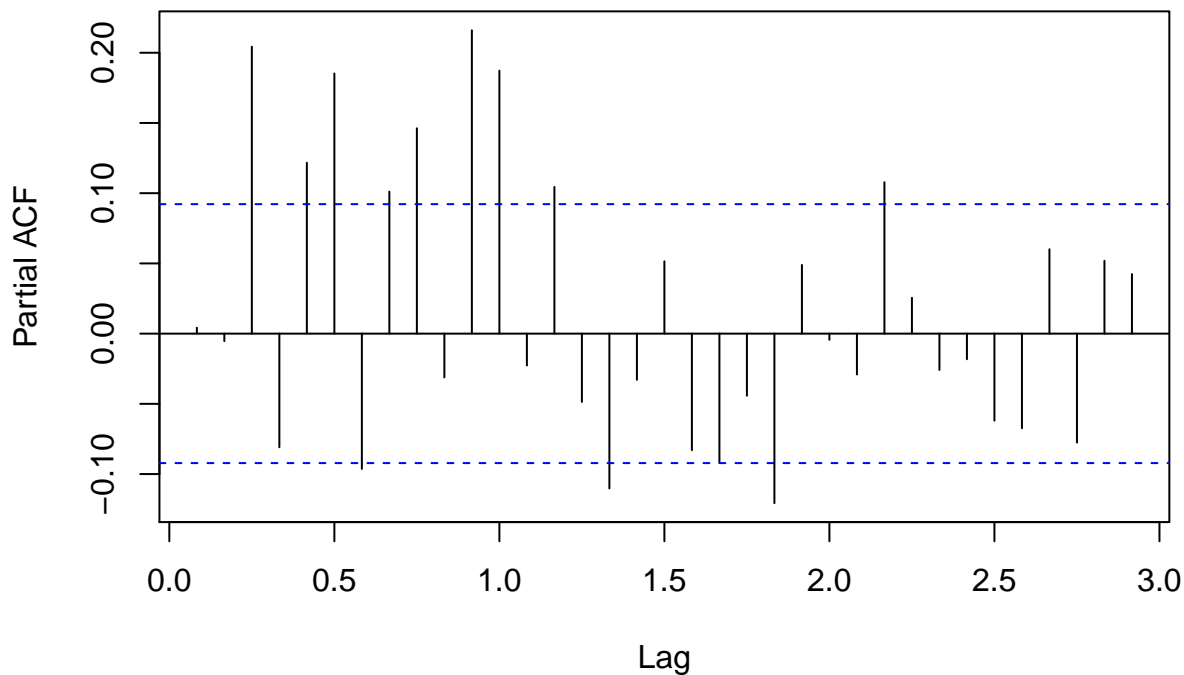
```
acf(logSeasPoly_resid, main="ACF of Residuals from Logged Beer- Seas Means and Poly Fit", lag=35)
```

ACF of Residuals from Logged Beer– Seas Means and Poly Fit



```
pacf(logSeasPoly_resid,main="PACF of Residuals from Logged Beer- Seas Means and Poly Fit", lag=35)
```

PACF of Residuals from Logged Beer- Seas Means and Poly Fit



```
adf.test(logSeasPoly_resid)
```

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

```
##  
## Augmented Dickey-Fuller Test  
##  
## data: logSeasPoly_resid  
## Dickey-Fuller = -5.3999, Lag order = 7, p-value = 0.01  
## alternative hypothesis: stationary
```

```
pp.test(logSeasPoly_resid)
```

```
## Warning in pp.test(logSeasPoly_resid): p-value smaller than printed p-value
```

```
##  
## Phillips-Perron Unit Root Test  
##  
## data: logSeasPoly_resid  
## Dickey-Fuller Z(alpha) = -489.81, Truncation lag parameter = 5,  
## p-value = 0.01  
## alternative hypothesis: stationary
```

Try an AR 12 model with the seasonal means and polynomial

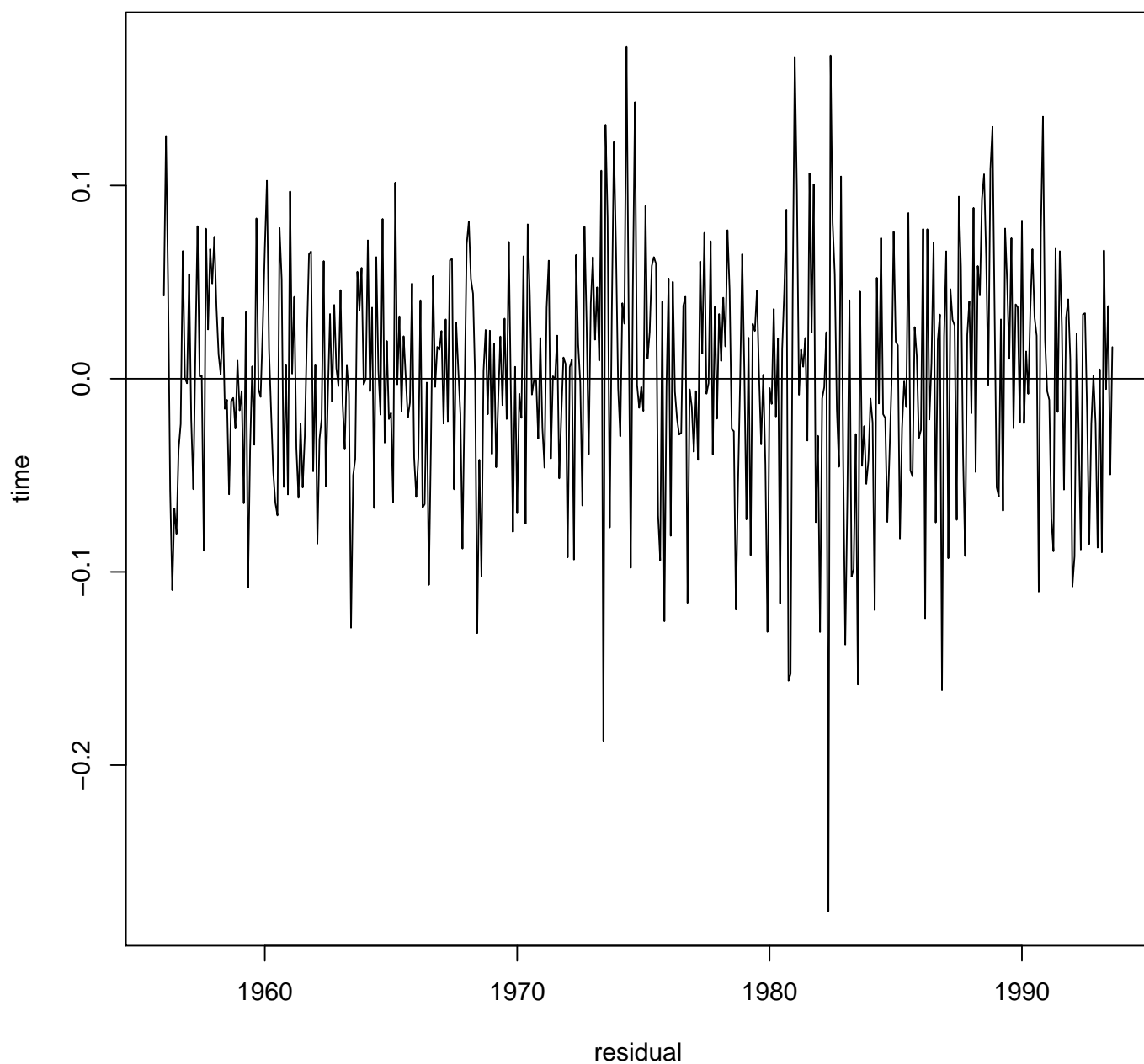
```
#Set up external regressors and dummy vars
library(forecast)
monthDummies<-seasonaldummy(logBeer)
externReg<-data.frame(t, t2, t3, t4, monthDummies)
```

```
ar12_poly<-arima(logBeer, order=c(12,0,0), xreg=externReg)
ar12_poly
```

```
##
## Call:
## arima(x = logBeer, order = c(12, 0, 0), xreg = externReg)
##
## Coefficients:
##          ar1          ar2          ar3          ar4          ar5          ar6          ar7          ar8
##      -0.0185  -0.0373   0.0592  -0.0600   0.0852   0.1086  -0.0873   0.0867
## s.e.   0.0460   0.0449   0.0449   0.0441   0.0443   0.0442   0.0443   0.0438
##          ar9          ar10          ar11          ar12 intercept          t          t2          t3          t4
##      0.1421  -0.0077   0.2301   0.2099          4.6792  -0.0009   0e+00          0          0
## s.e.  0.0440   0.0440   0.0440   0.0453          0.0781   0.0014   2e-04          0          0
##          Jan          Feb          Mar          Apr          May          Jun          Jul
##      -0.1923  -0.2566  -0.1788  -0.2787  -0.3181  -0.4344  -0.3506
## s.e.   0.0154   0.0170   0.0156   0.0165   0.0167   0.0156   0.0167
##          Aug          Sep          Oct          Nov
##      -0.3012  -0.2658  -0.1259  -0.0756
## s.e.   0.0165   0.0157   0.0171   0.0155
##
## sigma^2 estimated as 0.003593:  log likelihood = 629.73,  aic = -1203.47
```

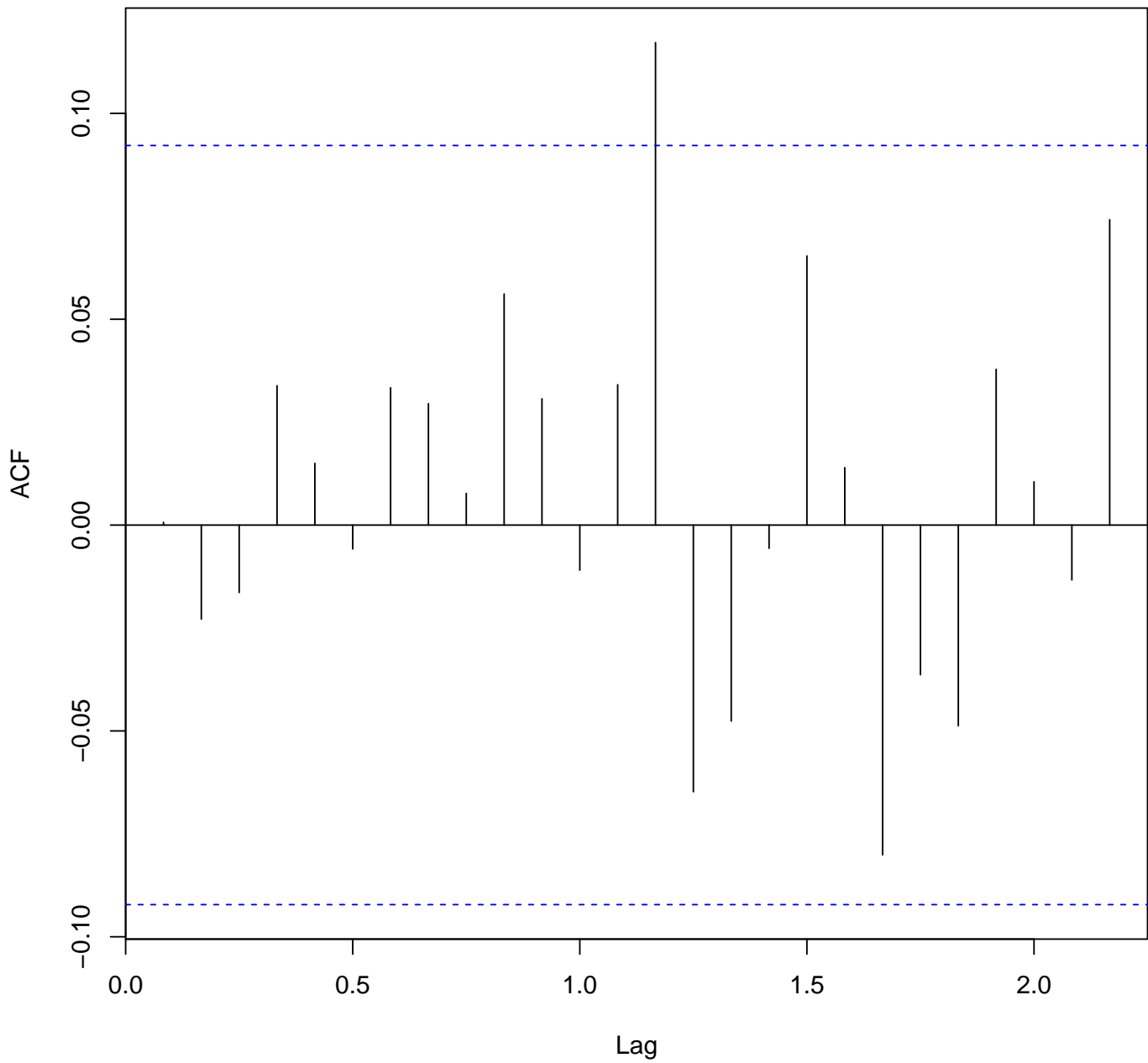
```
ar12_poly_resid<-ts(resid(ar12_poly), start=c(1956, 1), frequency=12)
plot(ar12_poly_resid, xlab="residual", ylab="time",main="Residuals from ARIMA(12,0,0) with Seas/Poly Determini
abline(h=0)
```

Residuals from ARIMA(12,0,0) with Seas/Poly Deterministic Trend



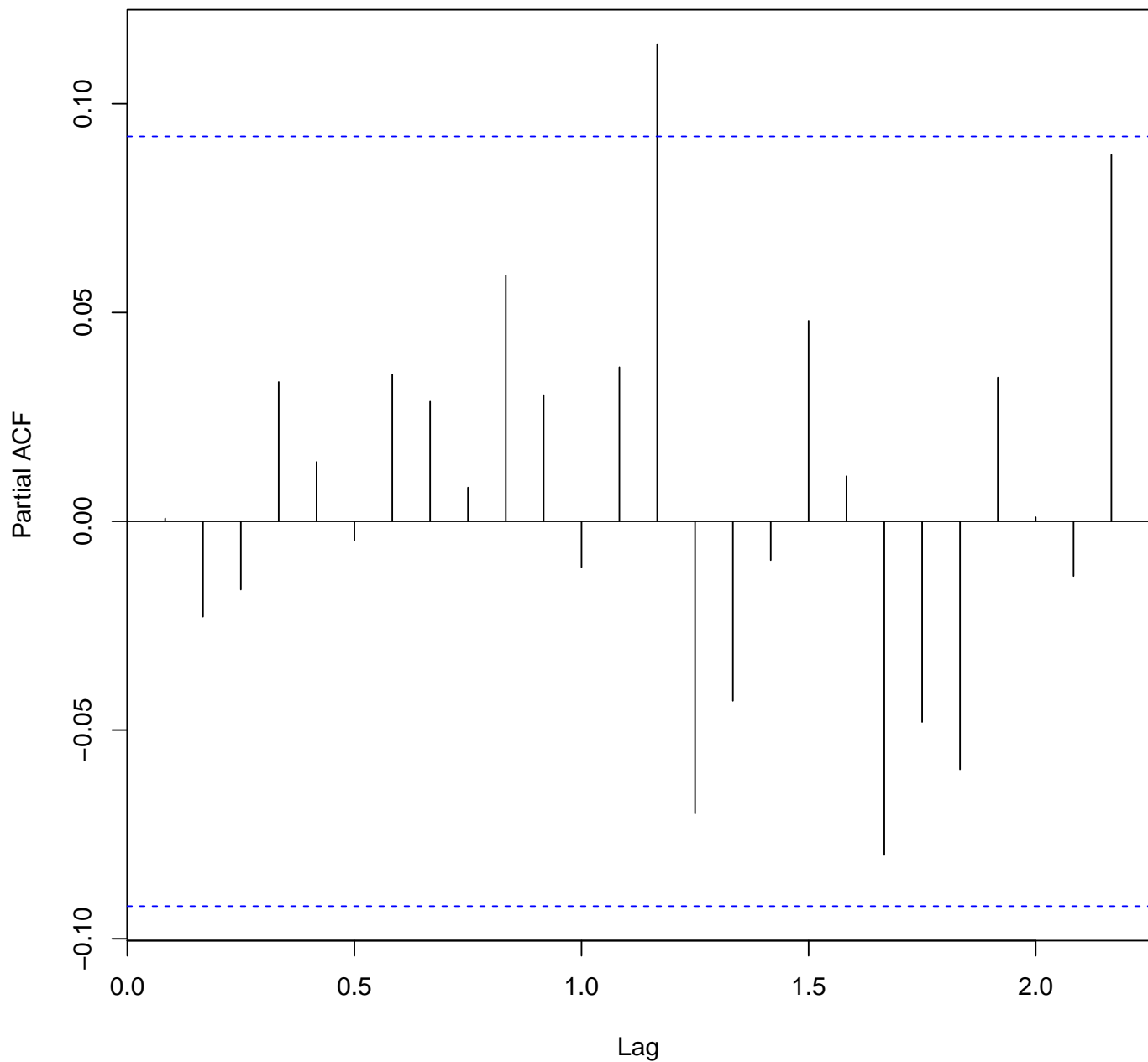
```
acf(ar12_poly_resid,main="ACF of Residuals from ARIMA(12,0,0)\nwith Seas/Poly Deterministic Trend")
```

**ACF of Residuals from ARIMA(12,0,0)
with Seas/Poly Deterministic Trend**



```
pacf(ar12_poly_resid,main="PACF of Residuals from ARIMA(12,0,0)\nwith Seas/Poly Deterministic Trend",mar=c(4,2
```

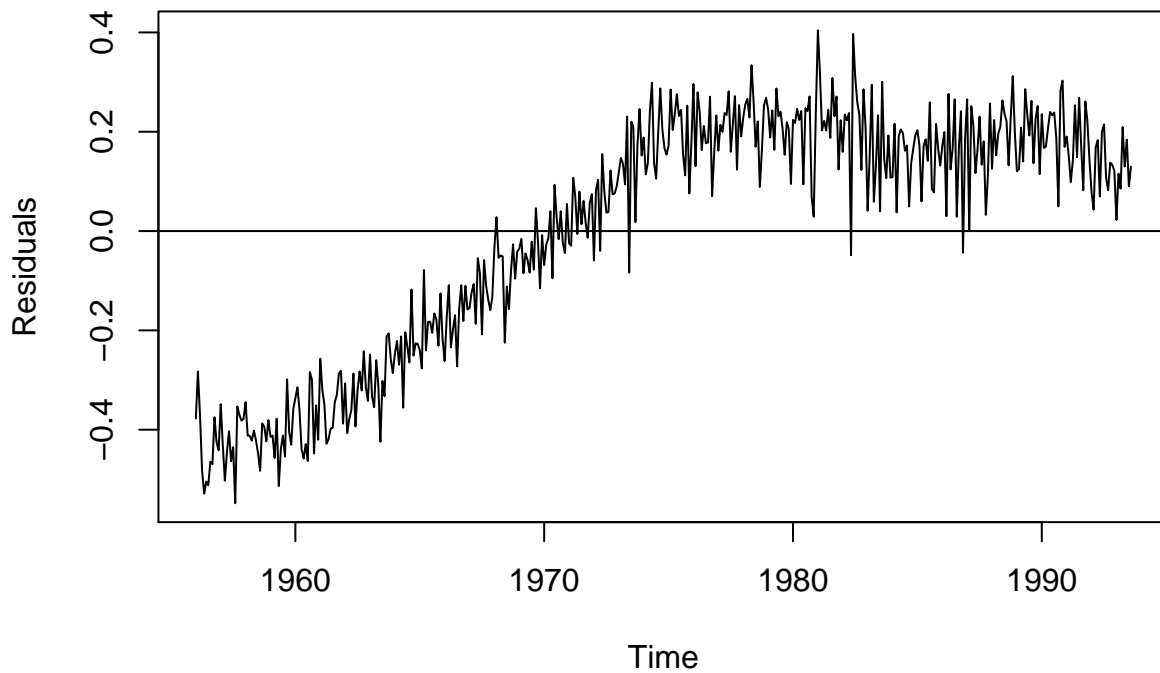
PACF of Residuals from ARIMA(12,0,0) with Seas/Poly Deterministic Trend



Build models based on the seasonal means only

```
seasMeansMod<-lm(logBeer~season(logBeer))  
plot(ts(residuals(seasMeansMod), start=c(1956,1),frequency=12), ylab="Residuals", main="Residual from Seasonal  
abline(h=0)
```

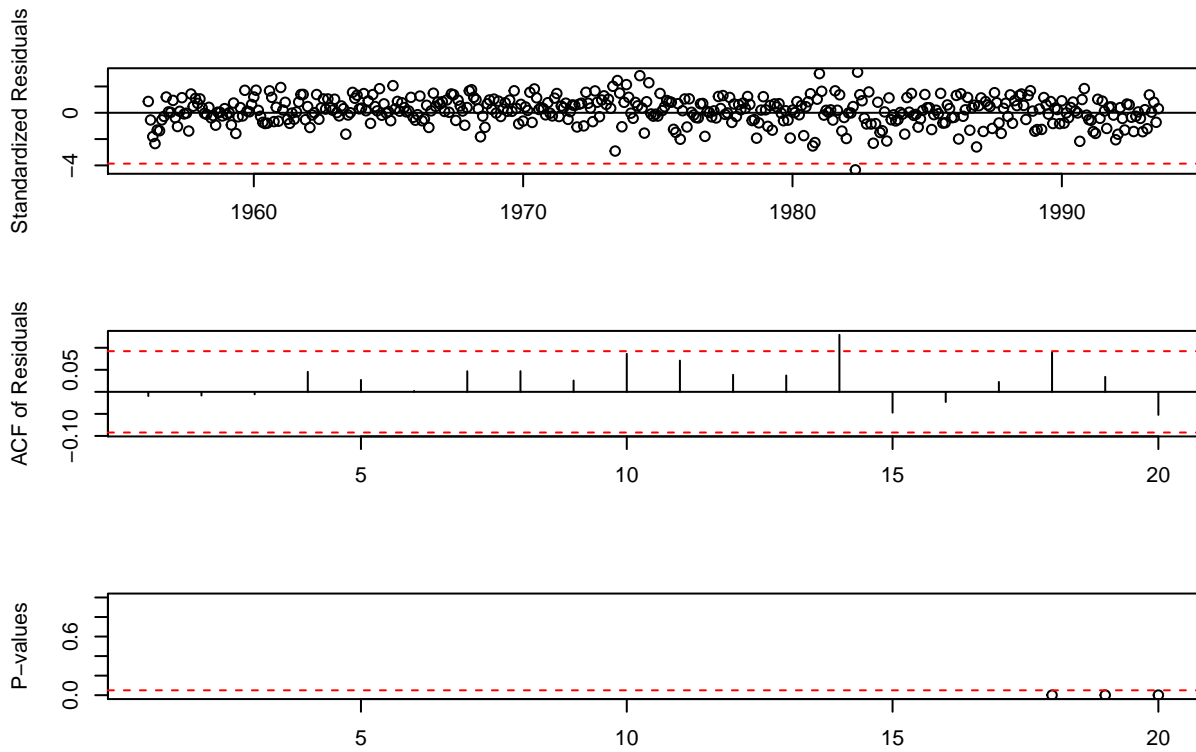
Residual from Seasonal Means Fit



```
ar12<-arima(logBeer, order=c(12,0,0), xreg=monthDummies)
ar12
```

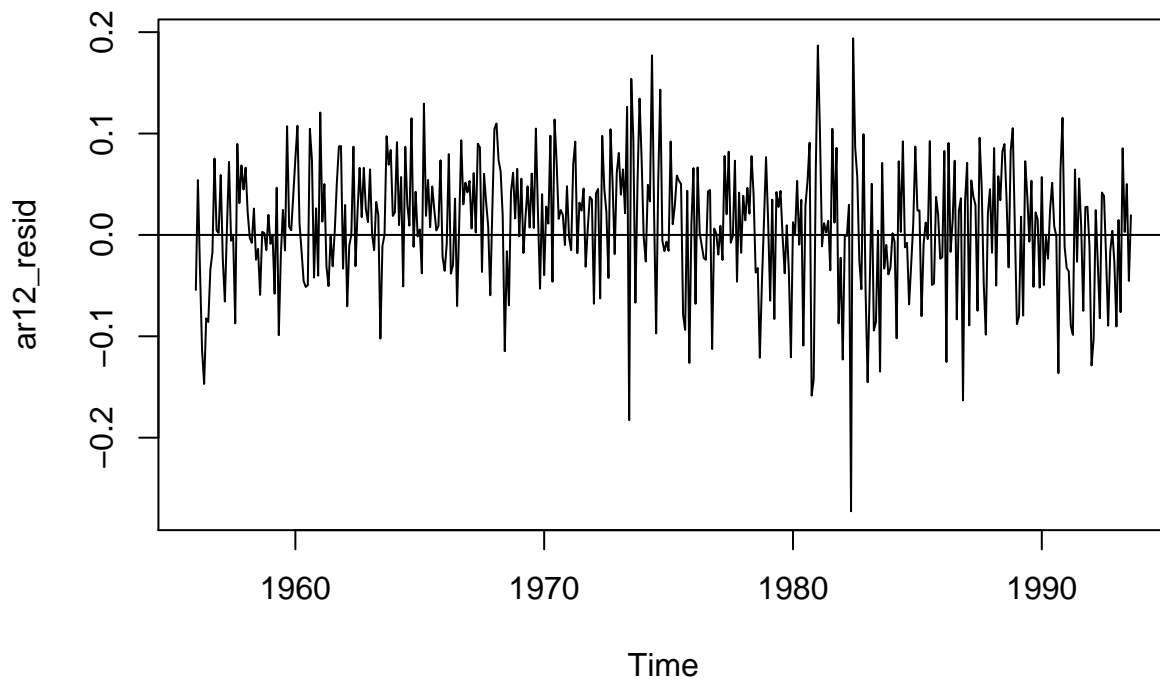
```
##
## Call:
## arima(x = logBeer, order = c(12, 0, 0), xreg = monthDummies)
##
## Coefficients:
##          ar1      ar2      ar3      ar4      ar5      ar6      ar7      ar8
##      0.0631  0.0283  0.1255 -0.0127  0.1286  0.1447 -0.0733  0.0983
## s.e.  0.0464  0.0458  0.0458  0.0457  0.0458  0.0460  0.0461  0.0459
##          ar9      ar10     ar11     ar12  intercept      Jan      Feb
##      0.1446 -0.0278  0.2102  0.1670      4.9720 -0.1925 -0.2574
## s.e.  0.0461  0.0461  0.0461  0.0471      0.2644  0.0152  0.0169
##          Mar      Apr      May      Jun      Jul      Aug      Sep
##      -0.1798 -0.2797 -0.3188 -0.4349 -0.3507 -0.3012 -0.2657
## s.e.  0.0154  0.0165  0.0167  0.0153  0.0167  0.0165  0.0155
##          Oct      Nov
##      -0.126 -0.0757
## s.e.  0.017  0.0153
##
## sigma^2 estimated as 0.003946:  log likelihood = 606.68,  aic = -1165.36
```

```
tsdiag(ar12, gof.lag=20)
```

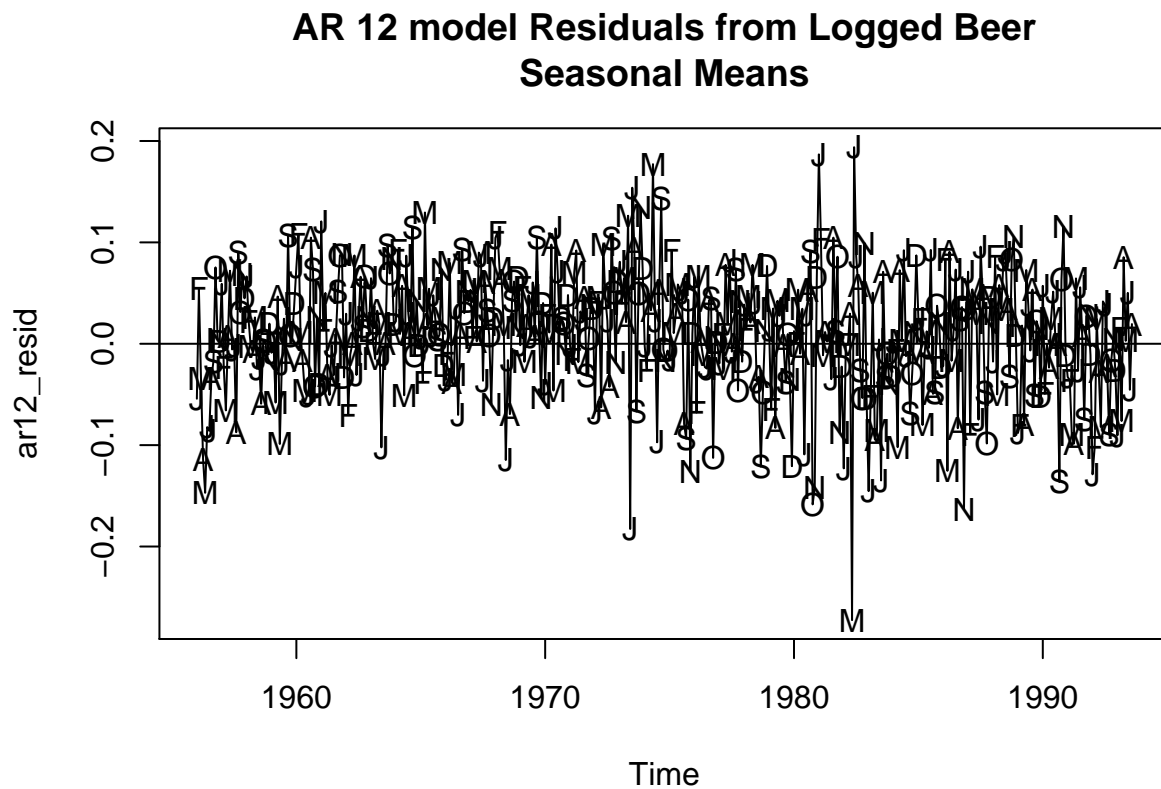


```
#residuals
ar12_resid<-ts(residuals(ar12), frequency=12, start=c(1956,1))
plot(ar12_resid, main="AR 12 model Residuals from Logged Beer\nSeasonal Means")
abline(h=0)
```

AR 12 model Residuals from Logged Beer Seasonal Means

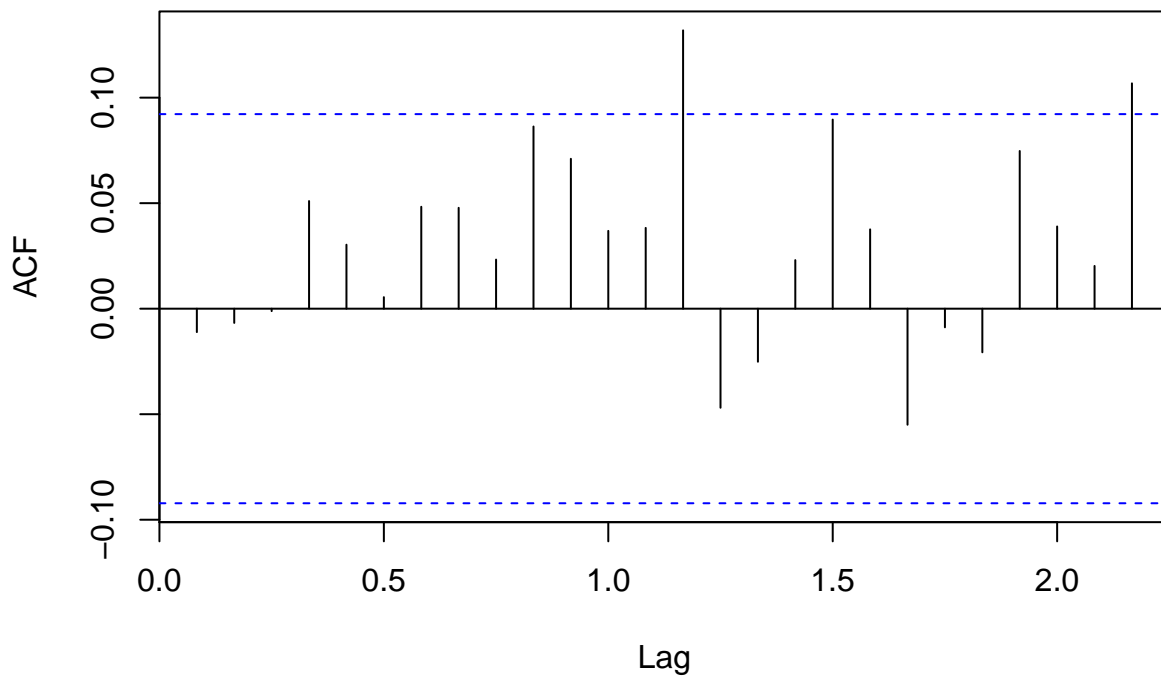



```
plot(ar12_resid, main="AR 12 model Residuals from Logged Beer\nSeasonal Means", type="l")
points(y=ar12_resid, x=time(ar12_resid), pch=as.vector(season(ar12_resid)))
abline(h=0)
```



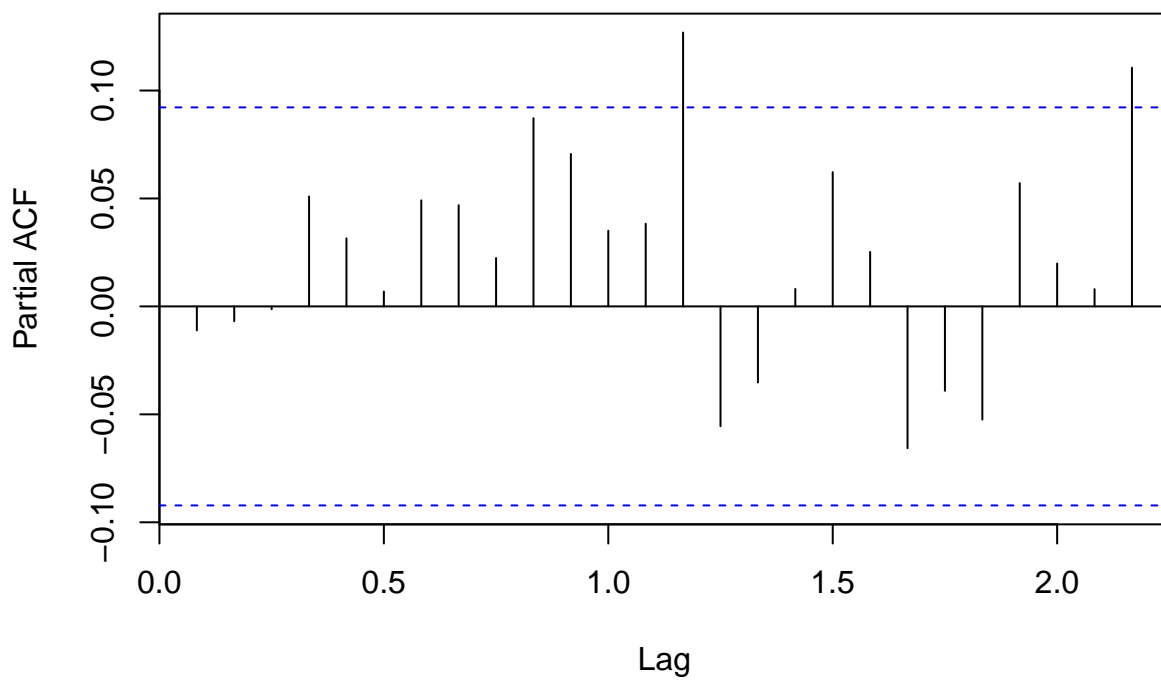
```
acf(ar12_resid, main="ACF Plot of Residuals from ARIMA(12,0,0)")
```

ACF Plot of Residuals from ARIMA(12,0,0)



```
pacf(ar12_resid, main="PACF Plot of Residuals from ARIMA(12,0,0)")
```

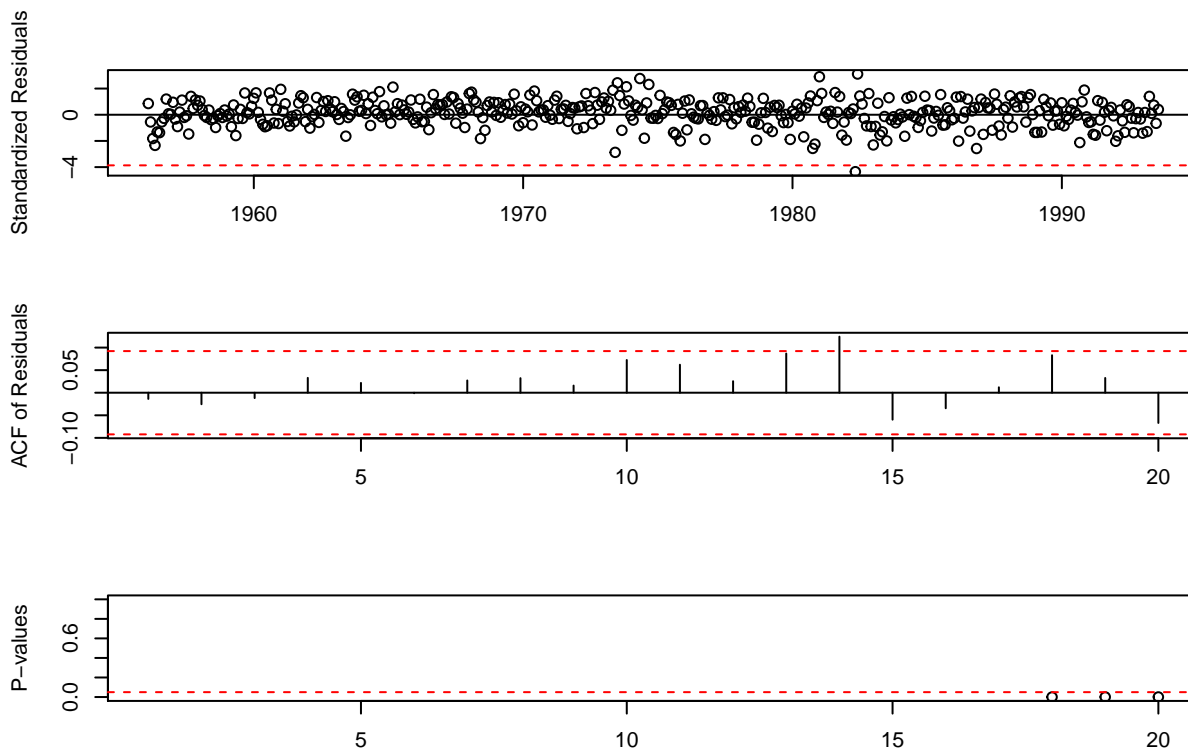
PACF Plot of Residuals from ARIMA(12,0,0)



```
ar13<-arima(logBeer, order=c(13,0,0), xreg=monthDummies)
ar13
```

```
##
## Call:
## arima(x = logBeer, order = c(13, 0, 0), xreg = monthDummies)
##
## Coefficients:
##      ar1      ar2      ar3      ar4      ar5      ar6      ar7      ar8
##  0.0731  0.0406  0.1232 -0.0038  0.1344  0.1406 -0.0638  0.1074
## s.e.  0.0470  0.0467  0.0458  0.0461  0.0459  0.0460  0.0466  0.0464
##      ar9      ar10      ar11      ar12      ar13  intercept      Jan
##  0.1442 -0.0190  0.2112  0.1696 -0.0614      4.9746 -0.1925
## s.e.  0.0460  0.0465  0.0460  0.0470  0.0477      0.2625  0.0152
##      Feb      Mar      Apr      May      Jun      Jul      Aug
## -0.2571 -0.1795 -0.2793 -0.3185 -0.4344 -0.3504 -0.3009
## s.e.  0.0164  0.0152  0.0159  0.0162  0.0152  0.0162  0.0159
##      Sep      Oct      Nov
## -0.2656 -0.1258 -0.0755
## s.e.  0.0153  0.0165  0.0153
##
## sigma^2 estimated as 0.003931:  log likelihood = 607.51,  aic = -1165.02
```

```
tsdiag(ar13, gof.lag=20)
```

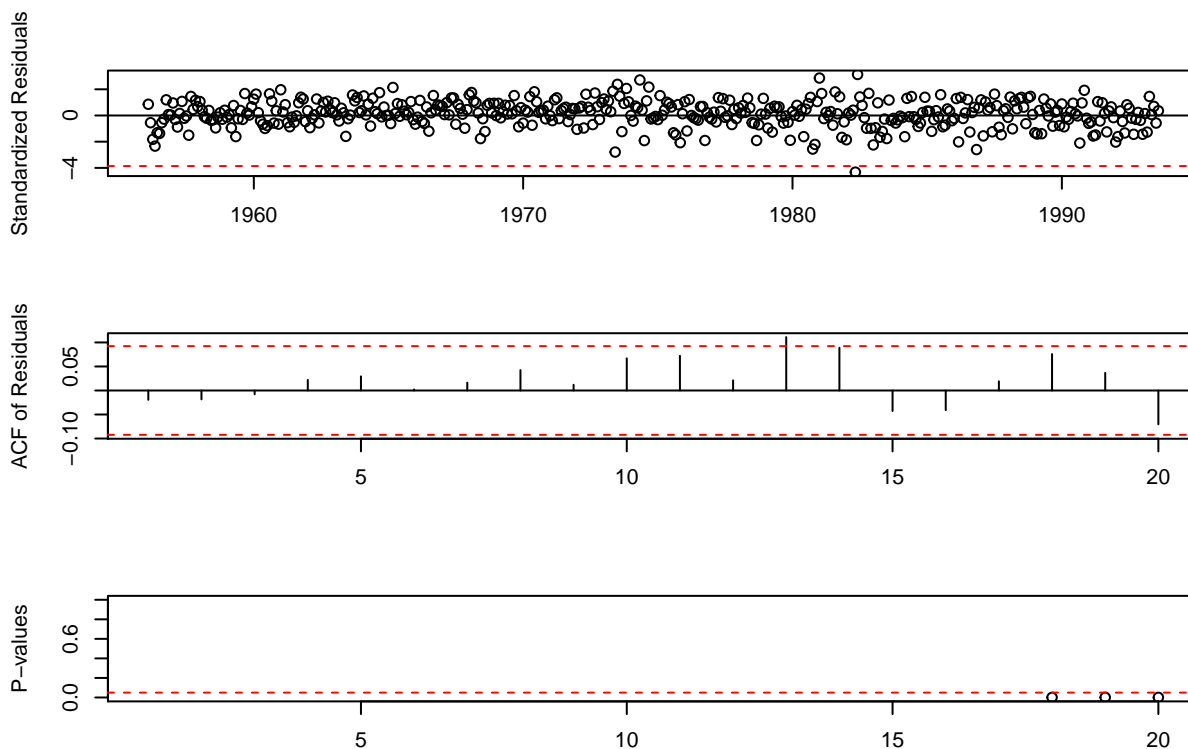


```
ar12ma1<-arima(logBeer, order=c(12,0,1), xreg=monthDummies)
ar12ma1
```

```
##
## Call:
```

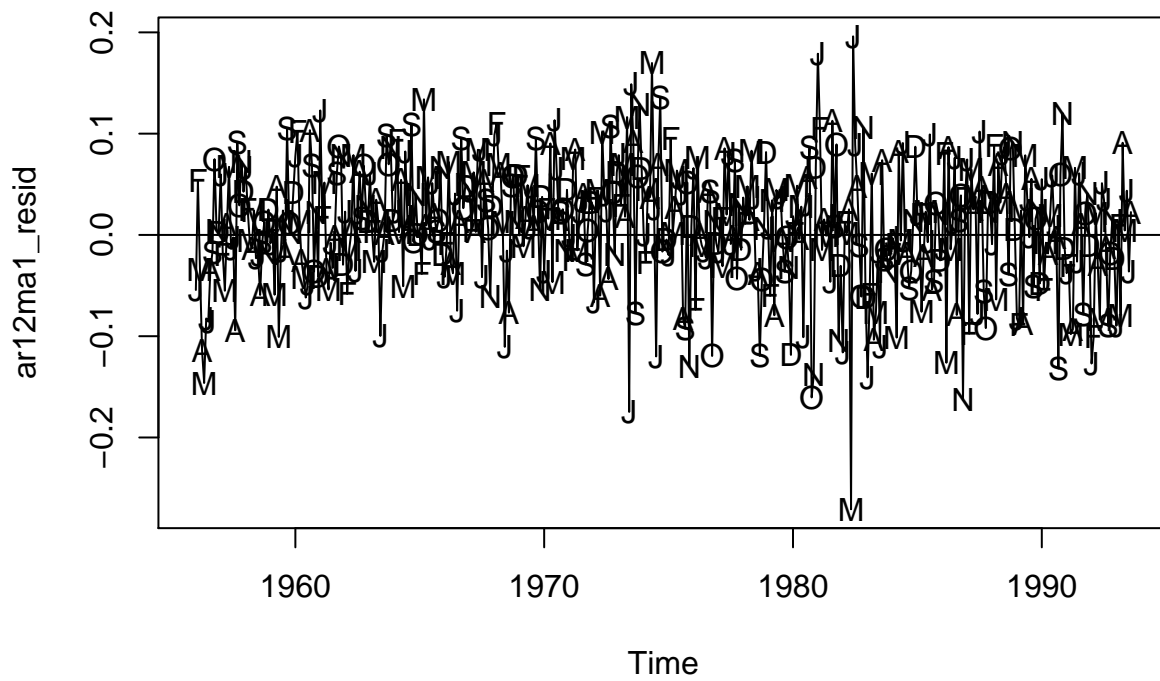
```
## arima(x = logBeer, order = c(12, 0, 1), xreg = monthDummies)
##
## Coefficients:
##      ar1      ar2      ar3      ar4      ar5      ar6      ar7      ar8
##    -0.4282  0.0770  0.1375  0.0634  0.1301  0.205  0.0136  0.0737
## s.e.   0.1845  0.0522  0.0501  0.0576  0.0494  0.056  0.0609  0.0504
##      ar9      ar10      ar11      ar12      ma1 intercept      Jan      Feb
##    0.1947  0.0575  0.1979  0.2726  0.5101      4.9738 -0.1927 -0.2573
## s.e.   0.0542  0.0593  0.0499  0.0503  0.1902      0.2632  0.0151  0.0160
##      Mar      Apr      May      Jun      Jul      Aug      Sep
##    -0.1799 -0.2794 -0.3189 -0.4344 -0.3507 -0.3009 -0.2659
## s.e.   0.0152  0.0157  0.0160  0.0150  0.0160  0.0157  0.0153
##      Oct      Nov
##    -0.1260 -0.0757
## s.e.   0.0161  0.0152
##
## sigma^2 estimated as 0.003916:  log likelihood = 608.36,  aic = -1166.72
```

```
tsdiag(ar12ma1, gof.lag=20)
```



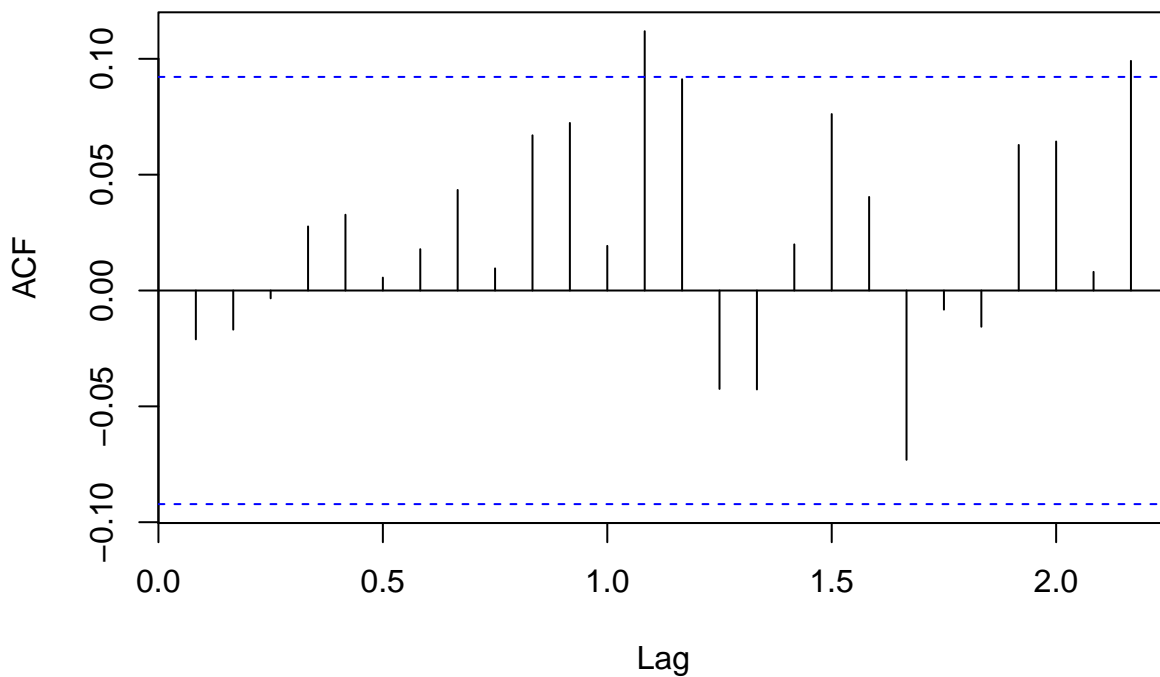
```
ar12ma1_resid<-ts(residuals(ar12ma1), frequency=12, start=c(1956,1))
plot(ar12ma1_resid, main="AR 12 MA 1 model Residuals from Logged Beer\nseasonal Means", type="l")
points(y=ar12ma1_resid, x=time(ar12ma1_resid), pch=as.vector(season(ar12ma1_resid)))
abline(h=0)
```

AR 12 MA 1 model Residuals from Logged Beer seasonal Means



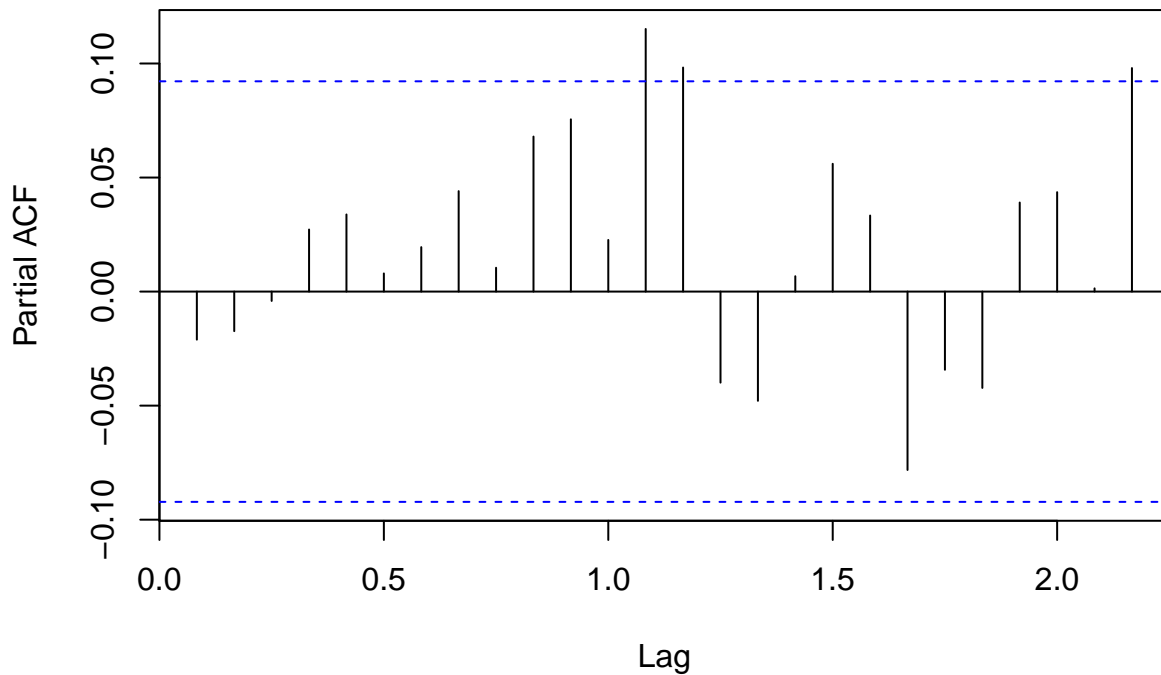
```
acf(ar12ma1_resid,main="ACF of Residuals from ARIMA(12,0,1)\nwith Seas Trend")
```

ACF of Residuals from ARIMA(12,0,1) with Seas Trend



```
pacf(ar12ma1_resid,main="PACF of Residuals from ARIMA(12,0,1)\nwith Seas Trend",mar=c(5,2,2,0.8), oma=c(1,1,1,
```

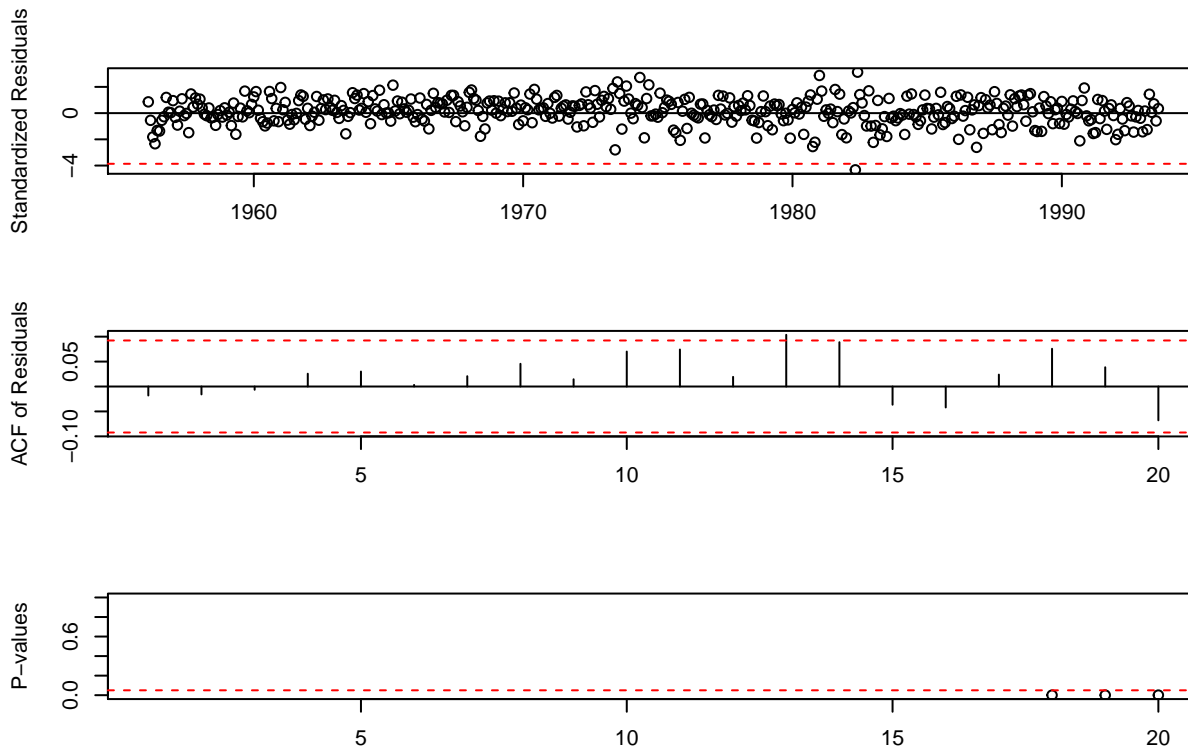
PACF of Residuals from ARIMA(12,0,1) with Seas Trend



```
ar13ma1<-arima(logBeer, order=c(13,0,1), xreg=monthDummies)
ar13ma1
```

```
##
## Call:
## arima(x = logBeer, order = c(13, 0, 1), xreg = monthDummies)
##
## Coefficients:
##      ar1      ar2      ar3      ar4      ar5      ar6      ar7      ar8
##    -0.5046  0.0793  0.1393  0.0704  0.1281  0.2144  0.0217  0.0664
## s.e.   0.2870  0.0532  0.0517  0.0608  0.0512  0.0633  0.0647  0.0565
##      ar9      ar10     ar11     ar12     ar13      ma1  intercept      Jan
##    0.2012  0.0663  0.1953  0.2919  0.0248  0.5841      4.9734  -0.1927
## s.e.   0.0581  0.0642  0.0520  0.0780  0.0804  0.2826      0.2635  0.0152
##      Feb      Mar      Apr      May      Jun      Jul      Aug
##    -0.2573 -0.1799 -0.2794 -0.3190 -0.4345 -0.3507 -0.3009
## s.e.   0.0161  0.0153  0.0158  0.0162  0.0151  0.0162  0.0158
##      Sep      Oct      Nov
##    -0.2660 -0.1260 -0.0757
## s.e.   0.0155  0.0162  0.0153
##
## sigma^2 estimated as 0.003915:  log likelihood = 608.41,  aic = -1164.82
```

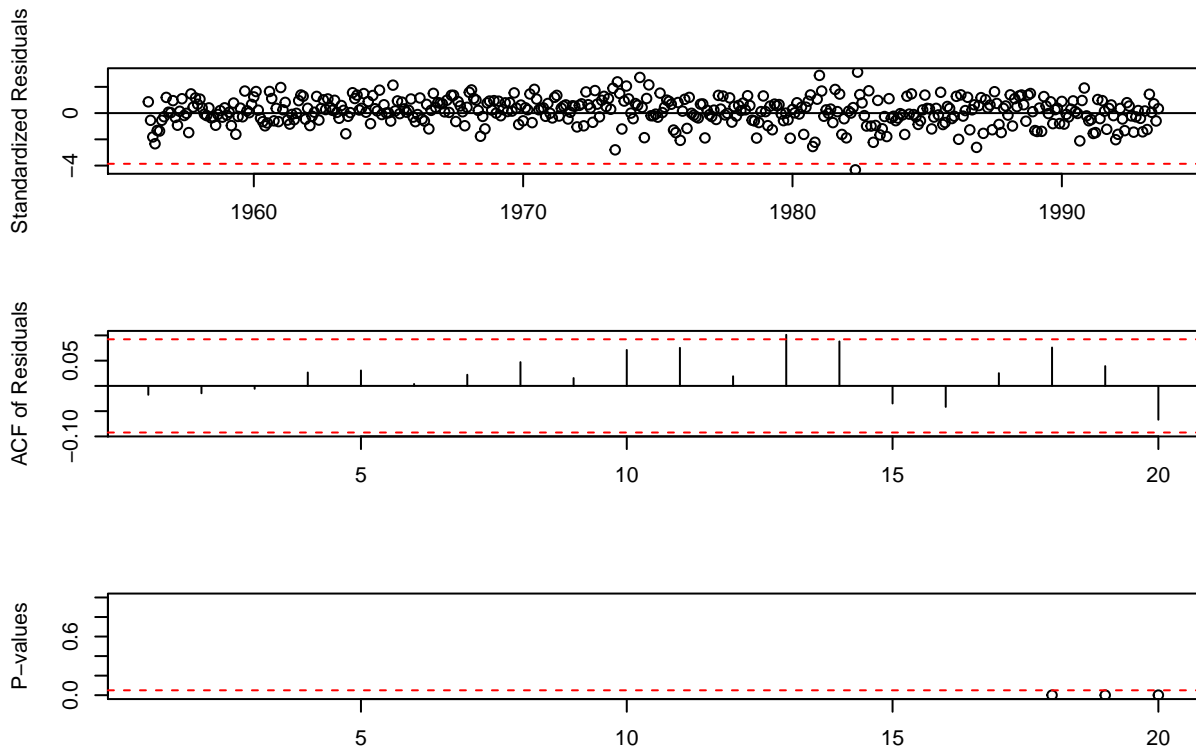
```
tsdiag(ar13ma1, gof.lag=20)
```



```
ar12ma2<-arima(logBeer, order=c(12,0,2), xreg=monthDummies)
ar12ma2
```

```
##
## Call:
## arima(x = logBeer, order = c(12, 0, 2), xreg = monthDummies)
##
## Coefficients:
##          ar1      ar2      ar3      ar4      ar5      ar6      ar7      ar8
##      -0.4094  0.1303  0.1308  0.0563  0.1201  0.2020  0.000  0.0627
## s.e.   0.1777  0.1517  0.0528  0.0592  0.0559  0.0553  0.069  0.0581
##          ar9      ar10      ar11      ar12      ma1      ma2 intercept      Jan
##      0.1944  0.0460  0.1877  0.2742  0.4882 -0.0600      4.9731 -0.1928
## s.e.  0.0534  0.0652  0.0571  0.0496  0.1843  0.1602      0.2638  0.0152
##          Feb      Mar      Apr      May      Jun      Jul      Aug
##      -0.2573 -0.1800 -0.2795 -0.3190 -0.4345 -0.3508 -0.3009
## s.e.   0.0162  0.0154  0.0159  0.0162  0.0151  0.0162  0.0159
##          Sep      Oct      Nov
##      -0.2660 -0.1260 -0.0757
## s.e.   0.0155  0.0163  0.0153
##
## sigma^2 estimated as 0.003915:  log likelihood = 608.43,  aic = -1164.85
```

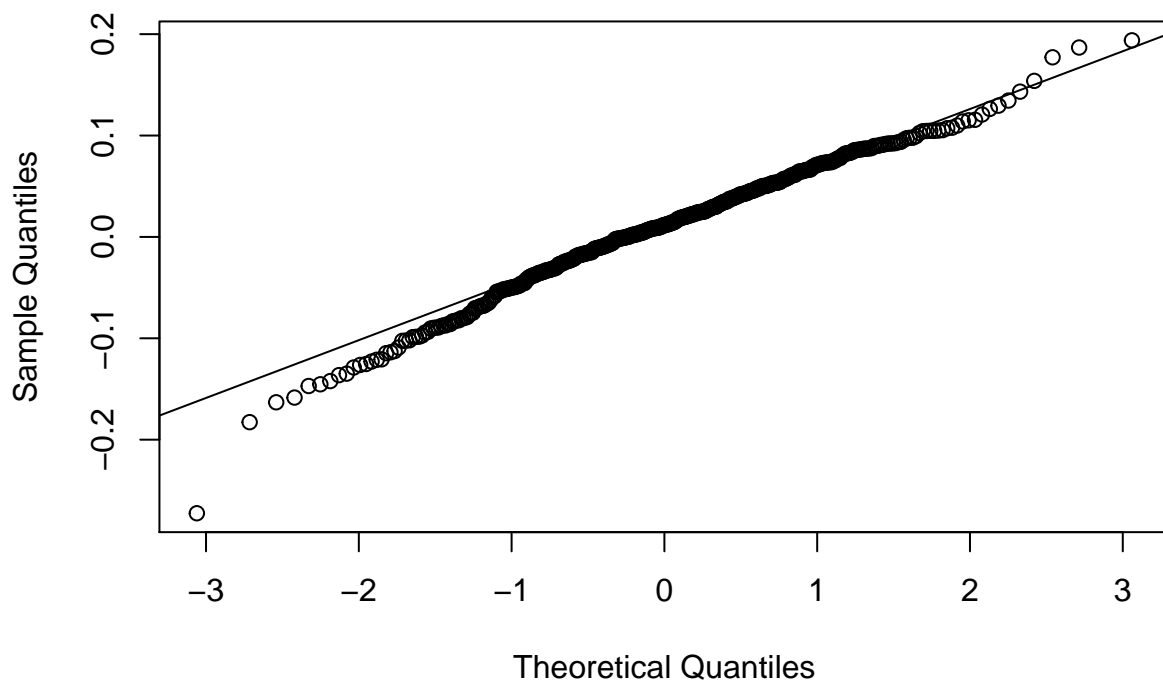
```
tsdiag(ar12ma2, gof.lag=20)
```



```
par(mfrow=c(1,1))

qqnorm(residuals(ar12), main="Normal QQ Plot of Residuals from ARIMA(12,0,0)")
qqline(ar12_resid)
```

Normal QQ Plot of Residuals from ARIMA(12,0,0)




```
shapiro.test(ar12_resid)
```

```
##  
## Shapiro-Wilk normality test  
##  
## data: ar12_resid  
## W = 0.98657, p-value = 0.000348
```

```
LB.test(ar12, lag=35)
```

```
##  
## Box-Ljung test  
##  
## data: residuals from ar12  
## X-squared = 49.126, df = 23, p-value = 0.001198
```

```
pacf_acf<-data.frame(acfVal=acf(ar12_resid, plot=FALSE)$acf, pacfVal=pacf(ar12_resid, plot=FALSE)$acf)  
#print(pacf_acf)
```

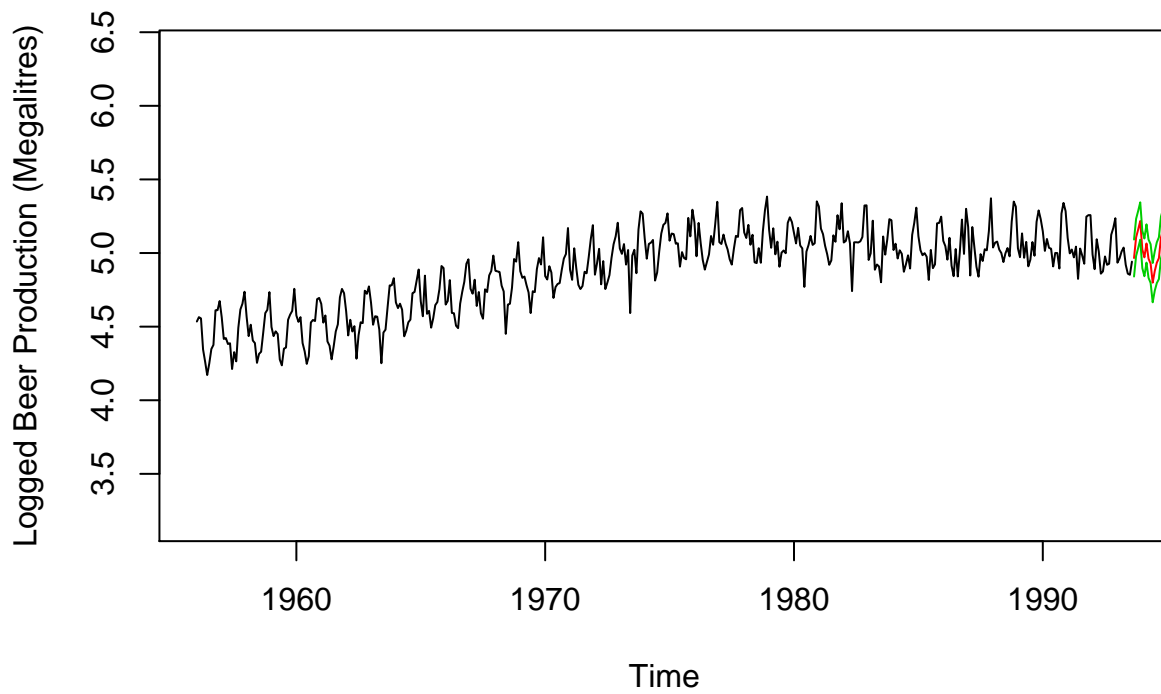
Make the forecasts- set up external regressor data frame

```
newMonthDummy<-seasonaldummy(beer_forecast)
```

Plot the model forecasts

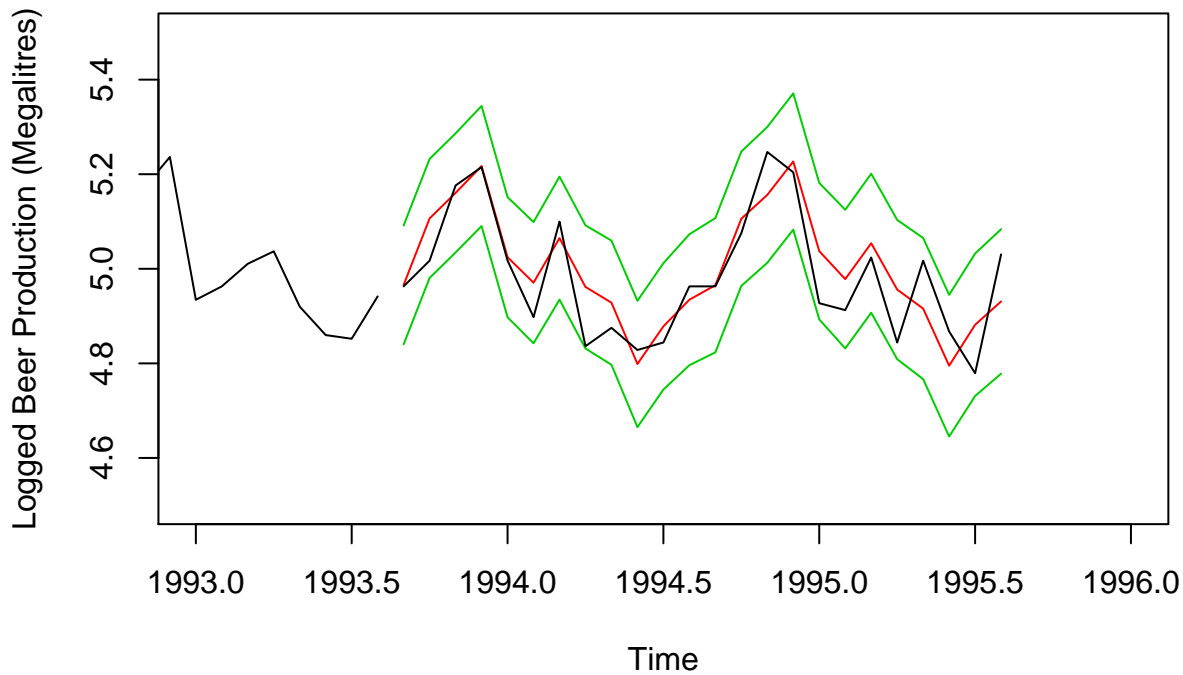
```
modelString<-"Seasonal Means - ARIMA(12,0,0)"  
chosenMod<-ar12  
predictions<-predict(chosenMod, n.ahead=24,newxreg=newMonthDummy)  
pred<-predictions$pred  
uci<-pred+2*predictions$se  
lci<-pred-2*predictions$se  
  
sumSqErrSeasMean<-sum((log(beer_forecast)-pred)^2)  
  
ymin=min(c(as.vector(lci),logBeer))-1  
ymax=max(c(as.vector(uci),logBeer))+1  
plot(logBeer,ylim=c(ymin,ymax),main=modelString, ylab='Logged Beer Production (Megalitres)')  
lines(pred,col=2)  
lines(uci,col=3)  
lines(lci,col=3)
```

Seasonal Means – ARIMA(12,0,0)



```
ymin=min(c(as.vector(lci),logBeer))-1
ymax=max(c(as.vector(uci),logBeer))+1
plot(logBeer,xlim=c(1993, 1996), ylim=c(4.5,5.5),main=modelString, ylab='Logged Beer Production (Megalitres)')
lines(pred,col=2)
lines(uci,col=3)
lines(lci,col=3)
lines(log(beer_forecast), col="black")
```

Seasonal Means – ARIMA(12,0,0)



SARIMA model with population data

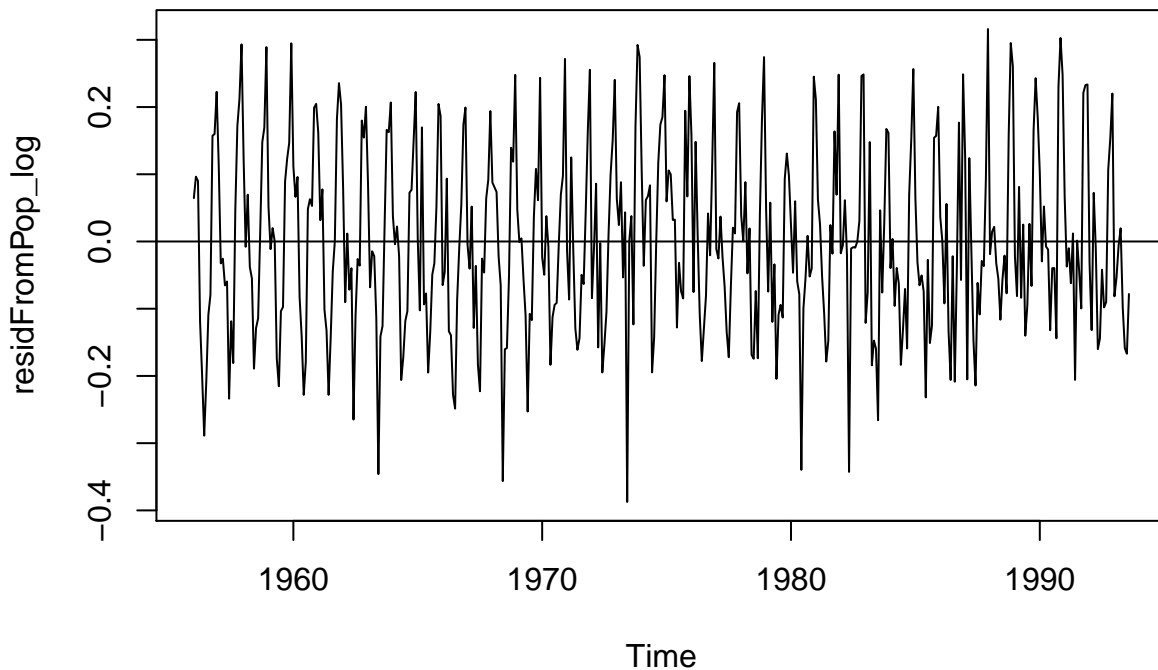
```
logBeer<-log(beerPopMonth$beer)
lag6_10_14<-beerPopMonth$lag6_10_14
t<-1:length(beerTS)
t2<-t^2
t3<-t^3
t4<-t^4
externalReg<-data.frame(lag6_10_14, t, t2, t3, t4)
monthlyPopModel_log<-lm(logBeer ~ lag6_10_14+t+t2+t3+t4)
monthlyPopOnly_log<-lm(logBeer ~ lag6_10_14)
summary(monthlyPopModel_log)
```

```
##
## Call:
## lm(formula = logBeer ~ lag6_10_14 + t + t2 + t3 + t4)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.38742 -0.09499 -0.01169  0.08863  0.31604
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.233e+00  1.167e-01  36.286 < 2e-16 ***
## lag6_10_14    4.122e-07  2.091e-07   1.971  0.0493 *
## t            -3.114e-03  1.322e-03  -2.356  0.0189 *
## t2             4.855e-05  8.989e-06   5.401 1.08e-07 ***
## t3            -1.647e-07  2.938e-08  -5.604 3.67e-08 ***
## t4             1.668e-10  3.217e-11   5.185 3.28e-07 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1357 on 446 degrees of freedom
## Multiple R-squared:  0.748, Adjusted R-squared:  0.7452
## F-statistic: 264.8 on 5 and 446 DF,  p-value: < 2.2e-16
```

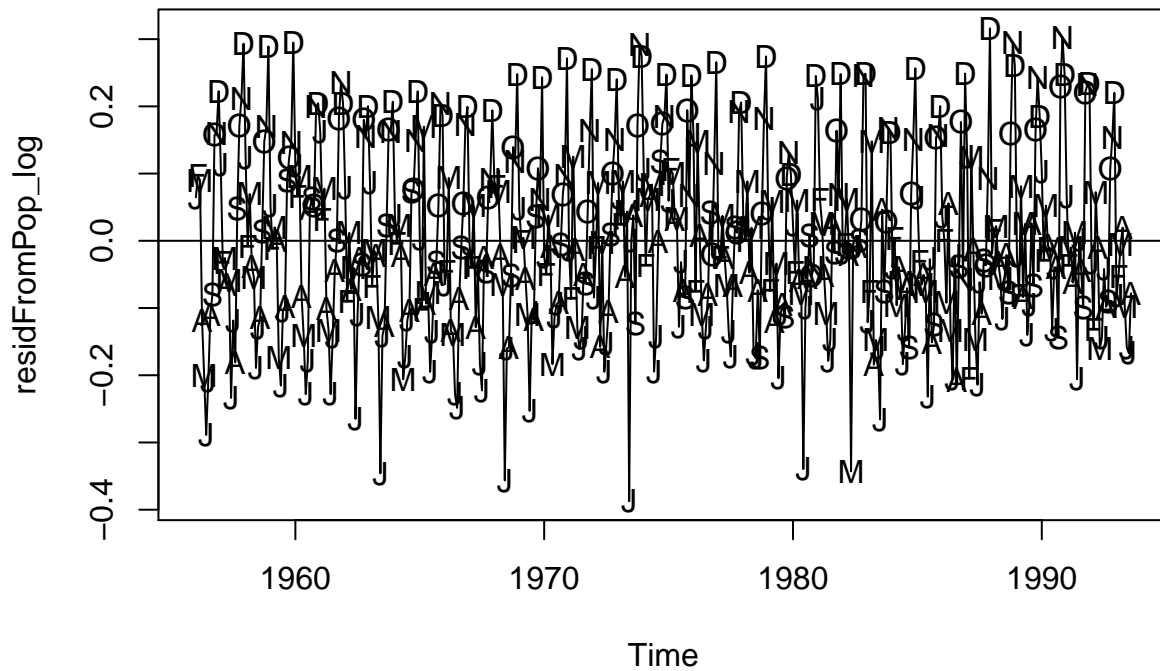
```
residFromPop_log<-ts(residuals(monthlyPopModel_log), frequency=12, start=c(1956,1))
plot(residFromPop_log, main="Plot of residuals from Pop/Poly Trend logged model")
abline(h=0)
```

Plot of residuals from Pop/Poly Trend logged model



```
plot(residFromPop_log, type="l", main="Plot of residuals from Pop/Poly Trend logged model")
points(y=residFromPop_log, x=time(residFromPop_log), pch=as.vector(season(residFromPop_log)))
abline(h=0)
```

Plot of residuals from Pop/Poly Trend logged model



```
adf.test(residFromPop_log)
```

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

```
##
## Augmented Dickey-Fuller Test
##
## data: residFromPop_log
## Dickey-Fuller = -14.166, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
pp.test(residFromPop_log)
```

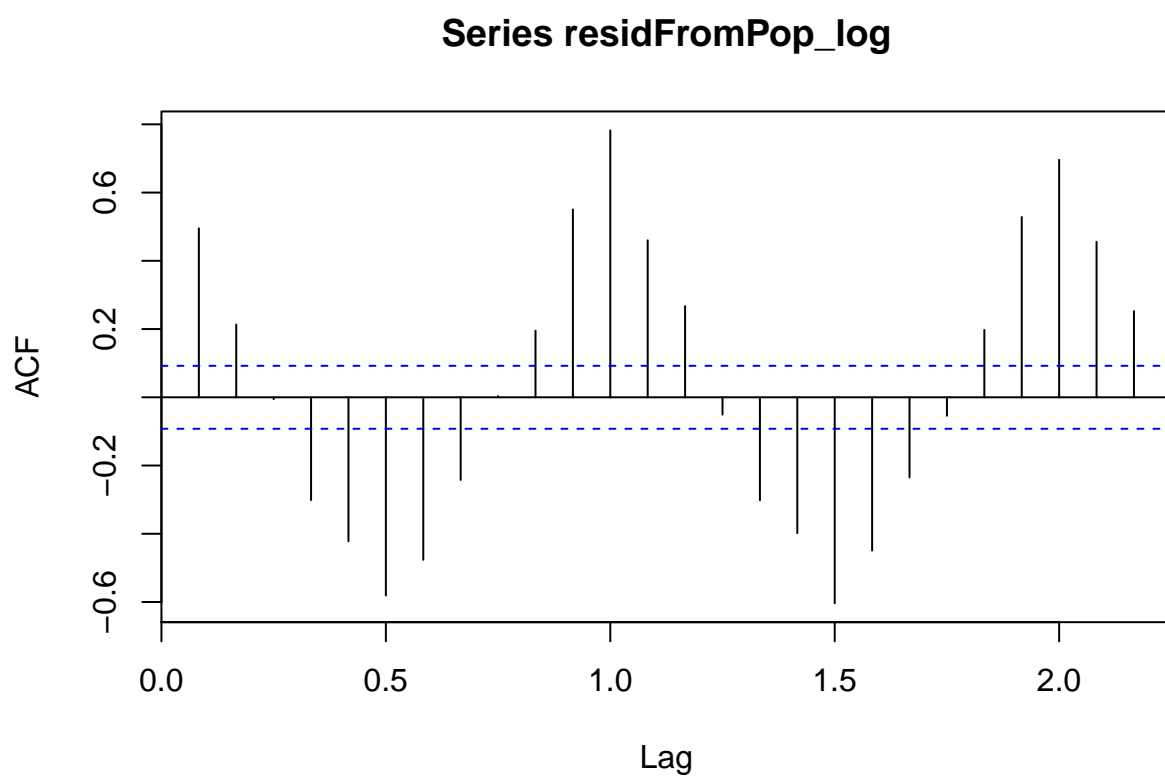
```
## Warning in pp.test(residFromPop_log): p-value smaller than printed p-value
```

```
##
## Phillips-Perron Unit Root Test
##
## data: residFromPop_log
## Dickey-Fuller Z(alpha) = -217.82, Truncation lag parameter = 5,
## p-value = 0.01
## alternative hypothesis: stationary
```

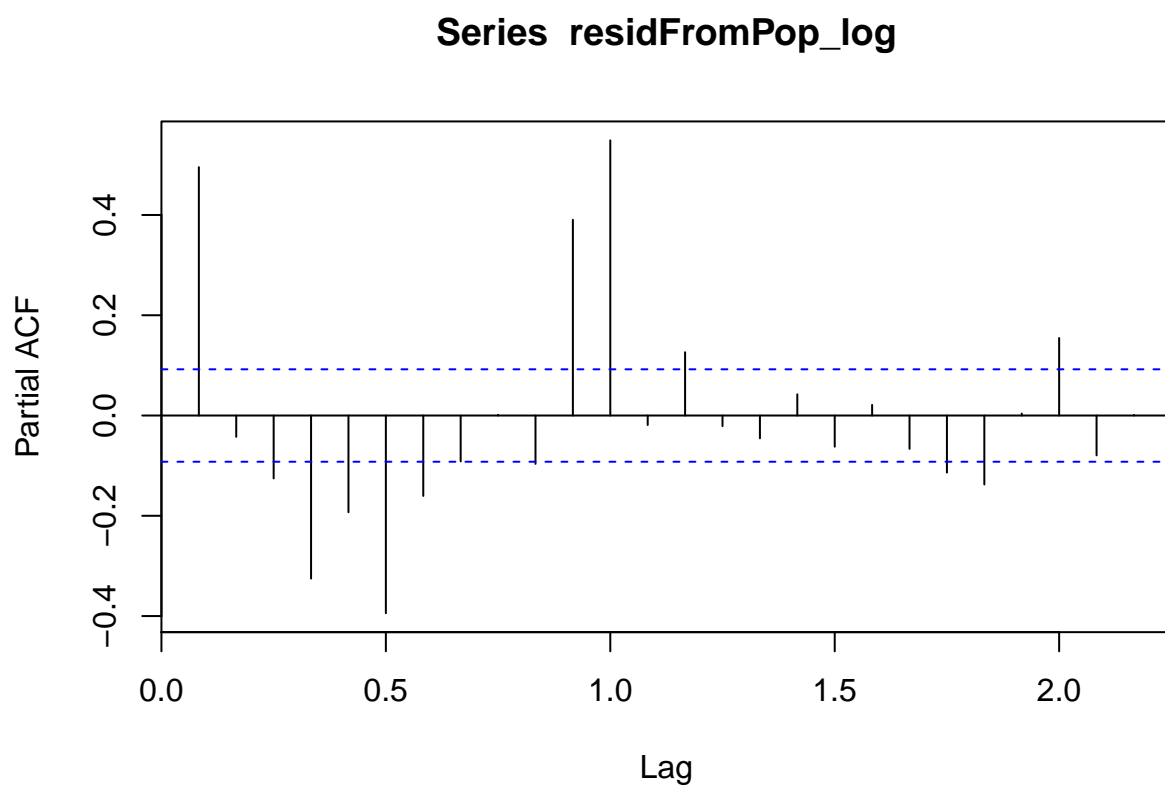
```
nsdiffs(residFromPop_log, m=frequency(residFromPop_log), test=c("ocsb","ch"), max.D=1)
```

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

```
acf(residFromPop_log)
```



```
pacf(residFromPop_log)
```

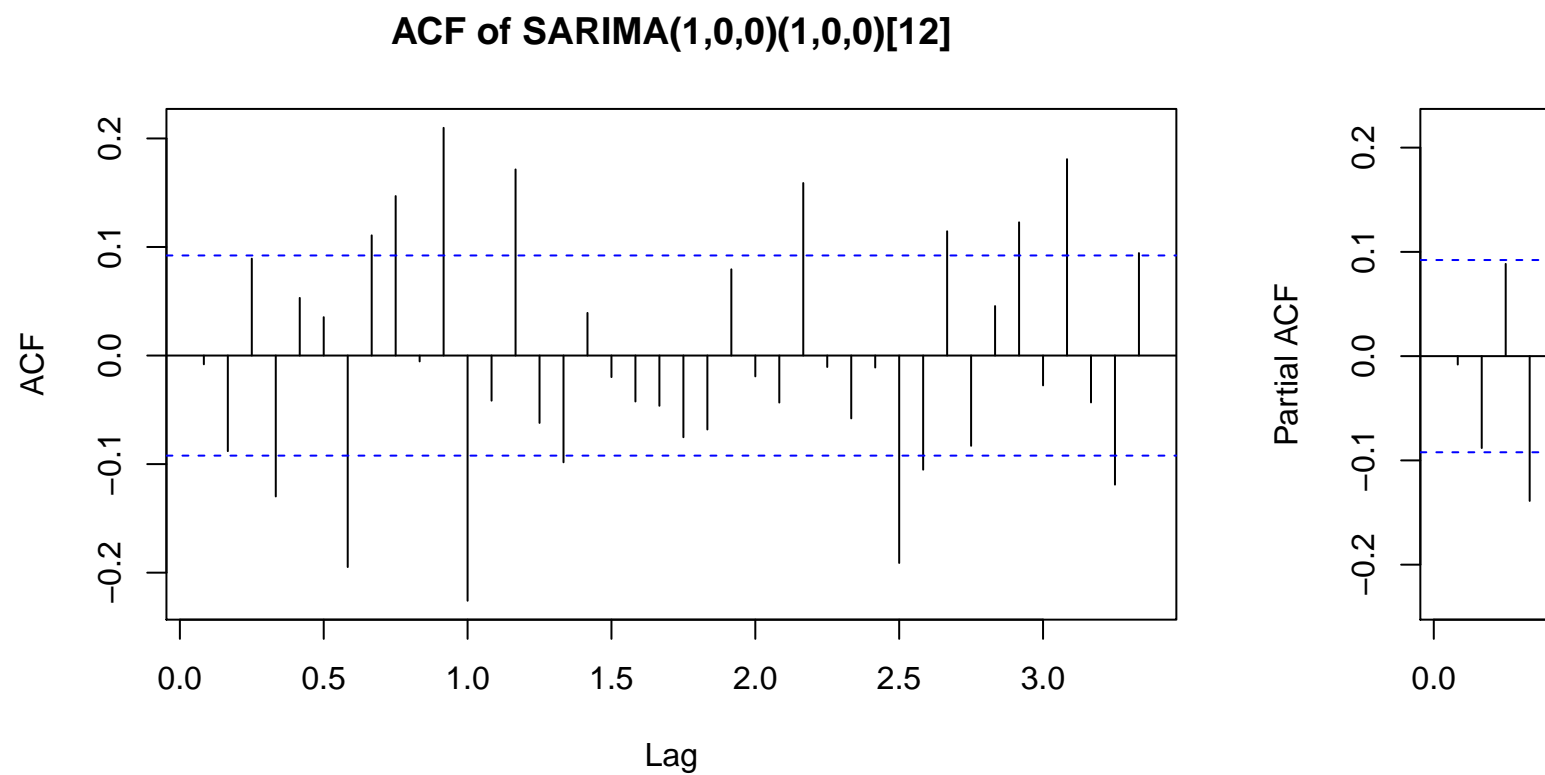


```
eacf(residFromPop_log)
```

```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x x o x x x x x o x x x x x
## 1 o x o x o x x x o o x x o x
## 2 x o x x o x o o o o o x o x
## 3 x o x o x x o o o x o x x x
## 4 x x x o x x o o o o o x x x
## 5 x x x x x o o o o o o x x x
## 6 x o x x x o o o o x o x x o
## 7 x x o o x x o o o o o x x o
```

Build initial model

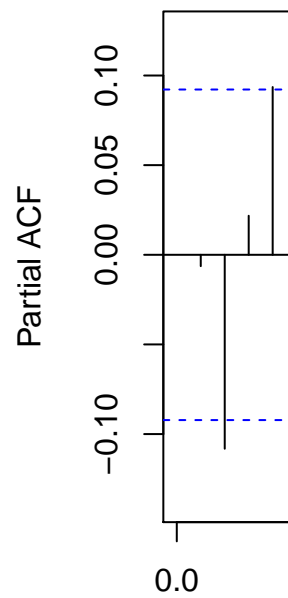
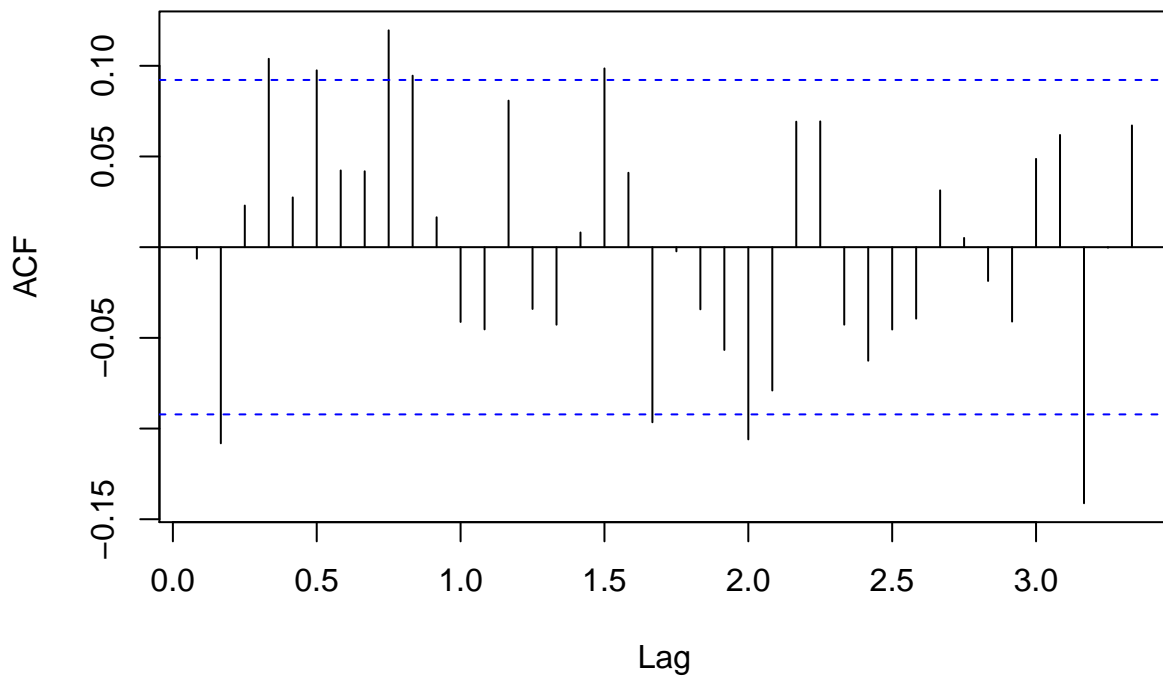
```
popModel1<-arima(residFromPop_log, order=c(1,0,0), seasonal=list(order=c(1,0,0), period=12))
plotResid(popModel1)
```



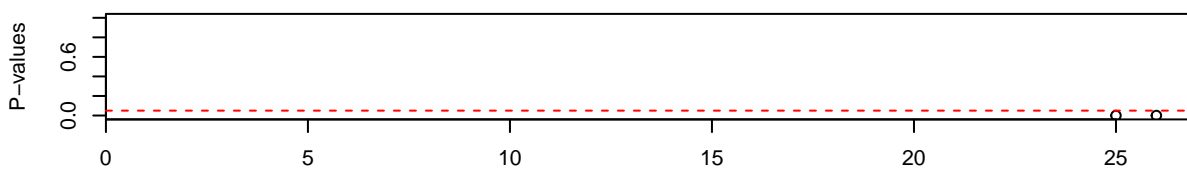
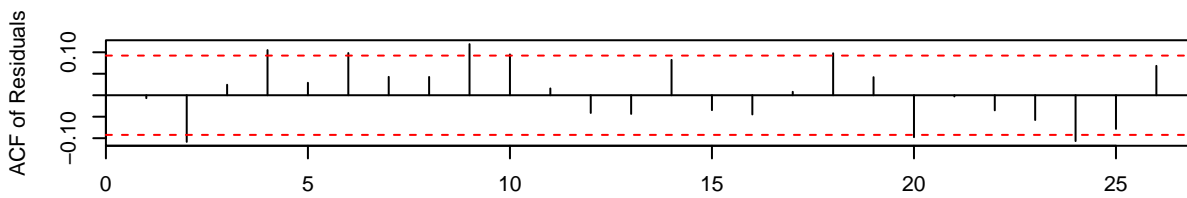
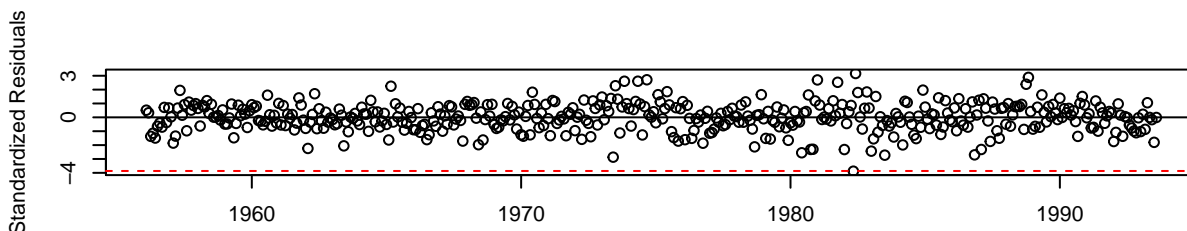
Try p=12 and q=12

```
popModel2<-arima(residFromPop_log, order=c(12,0,0), seasonal=list(order=c(1,0,0), period=12))
plotResid(popModel2)
```

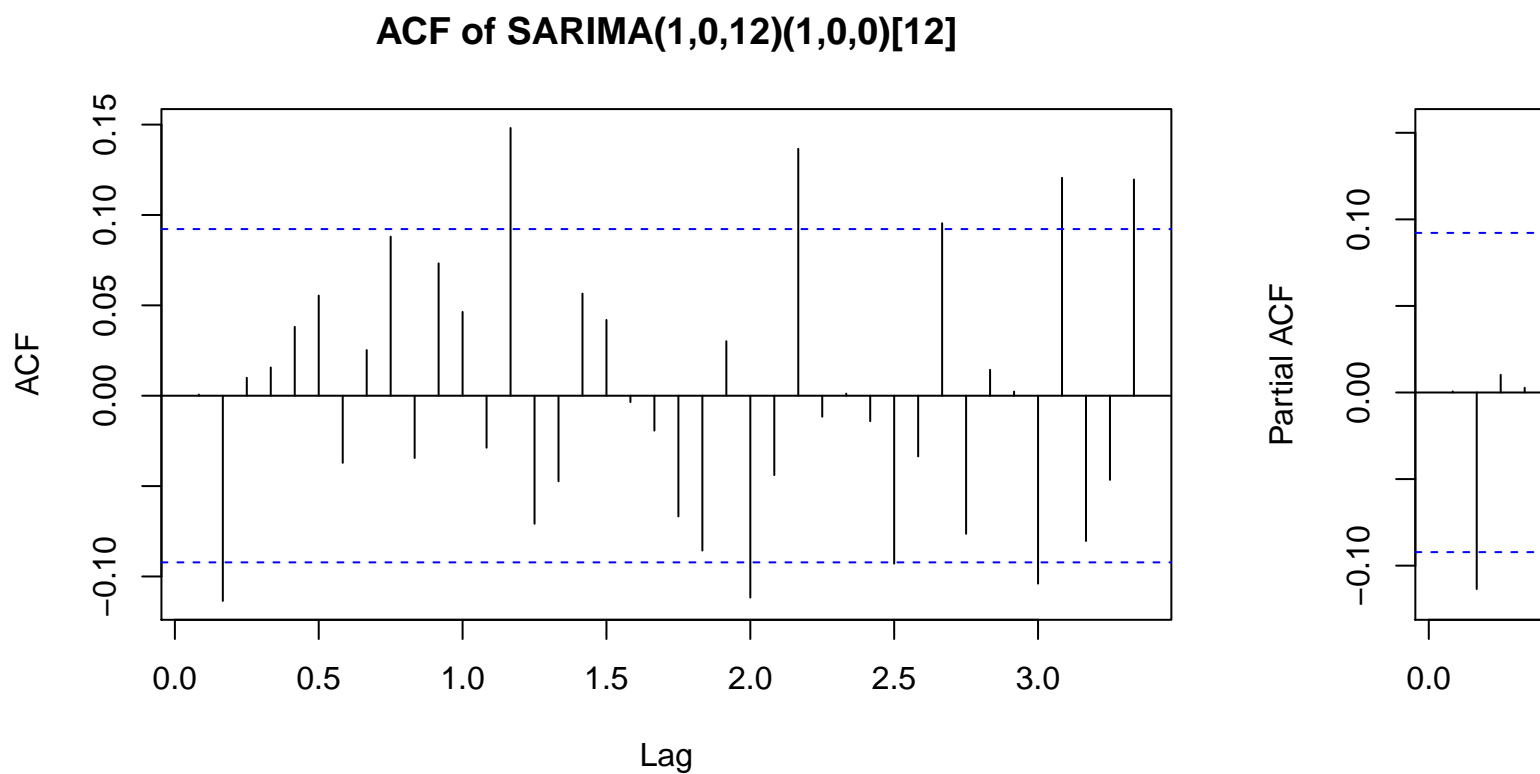
ACF of SARIMA(12,0,0)(1,0,0)[12]



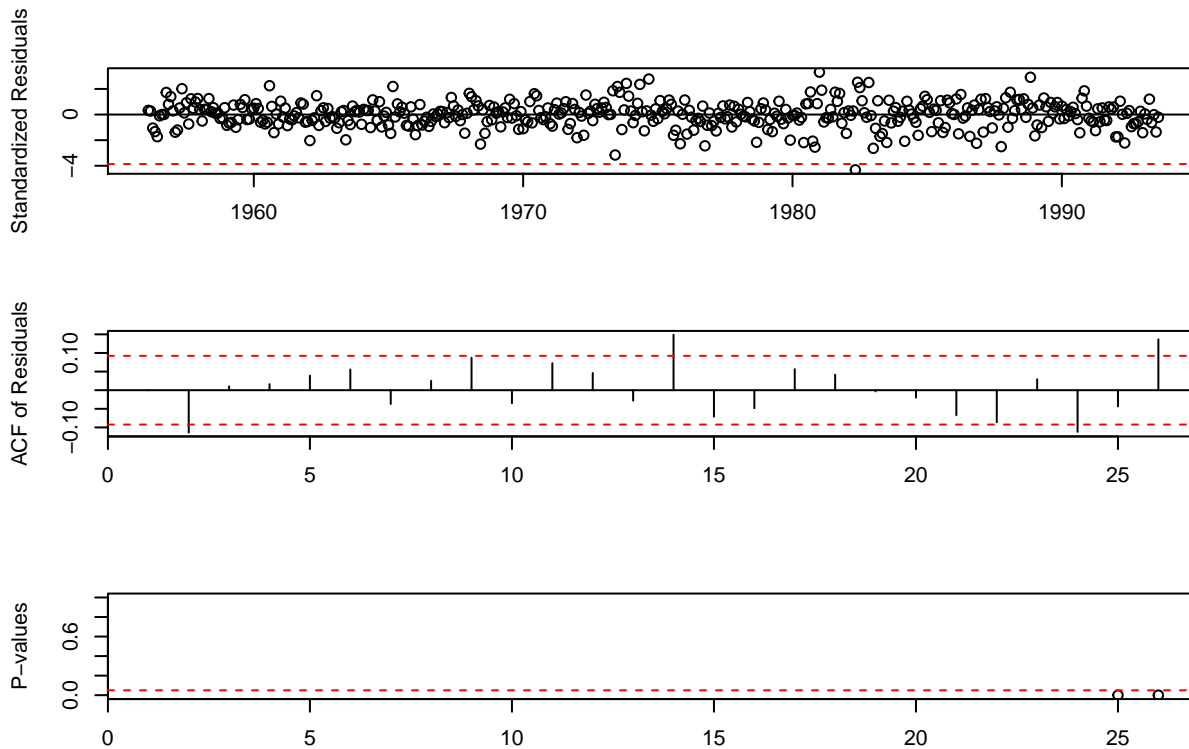
```
tsdiag(popModel12)
```




```
popModel3<-arima(residFromPop_log, order=c(1,0,12), seasonal=list(order=c(1,0,0), period=12))  
plotResid(popModel3)
```



```
tsdiag(popModel3)
```



Try overfitting the SARIMA(12,0,0)(1,0,0)[12]

```
#popModel4<-arima(residFromPop_log, order=c(13,0,0), seasonal=list(order=c(1,0,0), period=12))
#Produces error
```

```
popModel5<-arima(residFromPop_log, order=c(12,0,1), seasonal=list(order=c(1,0,0), period=12))
popModel5
```

```
##
## Call:
## arima(x = residFromPop_log, order = c(12, 0, 1), seasonal = list(order = c(1,
##      0, 0), period = 12))
##
## Coefficients:
##      ar1      ar2      ar3      ar4      ar5      ar6      ar7      ar8
##      0.0083  0.0269  0.0119 -0.1320  0.1293 -0.1074 -0.0709  0.0316
## s.e.  0.0493  0.0327  0.0295  0.0291  0.0311  0.0307  0.0336  0.0299
##      ar9      ar10      ar11      ar12      ma1      sar1      intercept
##      0.0174 -0.1302  0.2063  0.7417 -0.0301 -0.3180 -0.0008
## s.e.  0.0296  0.0292  0.0318  0.0405  0.0825  0.0528  0.0082
##
## sigma^2 estimated as 0.00456:  log likelihood = 571,  aic = -1112
```

```
popModel6<-arima(residFromPop_log, order=c(12,0,0), seasonal=list(order=c(2,0,0), period=12))
popModel6
```

```
##
## Call:
```

```
## arima(x = residFromPop_log, order = c(12, 0, 0), seasonal = list(order = c(2,
##      0, 0), period = 12))
##
## Coefficients:
##      ar1      ar2      ar3      ar4      ar5      ar6      ar7      ar8
##    -0.0548 -0.1008  0.0923 -0.0492  0.0658  0.0729 -0.0987  0.0549
## s.e.   0.0500  0.0483  0.0465  0.0460  0.0473  0.0515  0.0506  0.0481
##      ar9      ar10     ar11     ar12     sar1     sar2 intercept
##      0.2046 -0.0134  0.2072 -0.0683  0.5945  0.3155   -0.0092
## s.e.   0.0470  0.0490  0.0491  0.0707  0.0723  0.0650    0.0401
##
## sigma^2 estimated as 0.004827:  log likelihood = 554.21,  aic = -1078.41

popModel7<-arima(residFromPop_log, order=c(12,0,0), seasonal=list(order=c(1,0,1), period=12))
popModel7
```

```
##
## Call:
## arima(x = residFromPop_log, order = c(12, 0, 0), seasonal = list(order = c(1,
##      0, 1), period = 12))
##
## Coefficients:
##      ar1      ar2      ar3      ar4      ar5      ar6      ar7      ar8
##    -0.0127  0.0180 -0.0031 -0.0785  0.0743 -0.0670 -0.0219  0.0234
## s.e.   0.0197  0.0219  0.0183  0.0284  0.0339  0.0244  0.0232  0.0226
##      ar9      ar10     ar11     ar12     sar1     sma1 intercept
##      -0.0127 -0.0726  0.0907  0.9079  0.1563 -0.7465    0.0008
## s.e.   0.0183  0.0273  0.0418  0.0390  0.0779  0.1105    0.0056
##
## sigma^2 estimated as 0.004344:  log likelihood = 580.28,  aic = -1130.57
```

Overfit model 7 (SARIMA(12,0,0)(1,0,1)[12])

```
popModel8<-arima(residFromPop_log, order=c(13,0,0), seasonal=list(order=c(1,0,1), period=12))
popModel8

##
## Call:
## arima(x = residFromPop_log, order = c(13, 0, 0), seasonal = list(order = c(1,
##      0, 1), period = 12))
##
## Coefficients:

## Warning in sqrt(diag(x$var.coef)): NaNs produced

##      ar1      ar2      ar3      ar4      ar5      ar6      ar7      ar8
##    -0.0600  0.016  0.0068 -0.0947  0.0988 -0.0754 -0.0276  0.0322
## s.e.   0.0056   NaN  0.0037   NaN  0.0047   NaN  0.0033   NaN
##      ar9      ar10     ar11     ar12     ar13     sar1     sma1 intercept
##      -0.0027 -0.0964  0.1196  0.8842  0.0480  0.0909 -0.6295   -0.0015
## s.e.   0.0040  0.0013  0.0071   NaN  0.0038   NaN  0.0345    0.0076
##
## sigma^2 estimated as 0.004419:  log likelihood = 577.49,  aic = -1122.98
```

```
popModel9<-arima(residFromPop_log, order=c(12,0,1), seasonal=list(order=c(1,0,1), period=12))
popModel9
```

```
##
## Call:
## arima(x = residFromPop_log, order = c(12, 0, 1), seasonal = list(order = c(1,
## 0, 1), period = 12))
##
## Coefficients:
##          ar1      ar2      ar3      ar4      ar5      ar6      ar7      ar8
##      -0.0005  0.0087  0.0001 -0.0702  0.0667 -0.0649 -0.0088  0.0146
## s.e.   0.0208  0.0219  0.0179  0.0292  0.0380  0.0268  0.0226  0.0234
##          ar9      ar10     ar11     ar12      ma1      sar1      sma1
##      -0.0089 -0.0649  0.0790  0.9158 -0.0730  0.1851 -0.7909
## s.e.   0.0180  0.0279  0.0451  0.0412  0.0561  0.0748  0.1079
##      intercept
##           0.0009
## s.e.      0.0051
##
## sigma^2 estimated as 0.004323:  log likelihood = 581.08,  aic = -1130.15
```

```
popModel10<-arima(residFromPop_log, order=c(12,0,0), seasonal=list(order=c(2,0,1), period=12))
popModel10
```

```
##
## Call:
## arima(x = residFromPop_log, order = c(12, 0, 0), seasonal = list(order = c(2,
## 0, 1), period = 12))
##
## Coefficients:
##          ar1      ar2      ar3      ar4      ar5      ar6      ar7      ar8
##      -0.0246 -0.0221  0.0663 -0.0715  0.0796  0.0831 -0.1084  0.0583
## s.e.   0.0482  0.0505  0.0453  0.0447  0.0451  0.0482  0.0472  0.0457
##          ar9      ar10     ar11     ar12      sar1      sar2      sma1 intercept
##      0.1370 -0.0217  0.2249  0.2416  0.9178  0.0803 -0.9069 -0.0014
## s.e.   0.0492  0.0478  0.0462  0.1065  0.1056  0.1054  0.0335  0.0711
##
## sigma^2 estimated as 0.003779:  log likelihood = 602.36,  aic = -1172.72
```

```
popModel11<-arima(residFromPop_log, order=c(12,0,0), seasonal=list(order=c(1,0,2), period=12))
popModel11
```

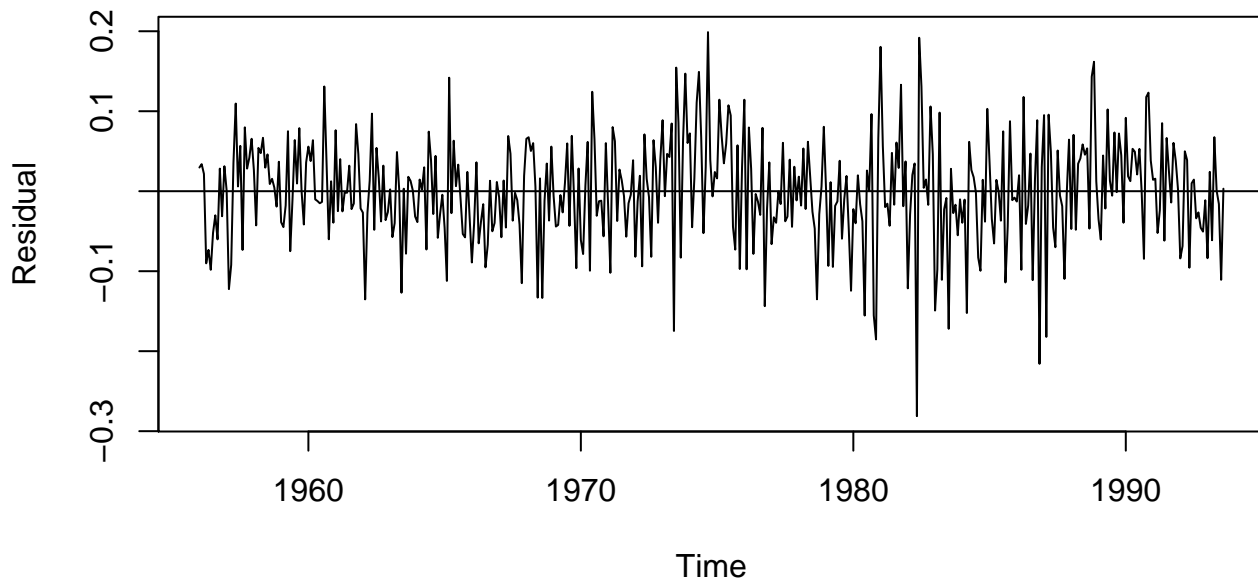
```
##
## Call:
## arima(x = residFromPop_log, order = c(12, 0, 0), seasonal = list(order = c(1,
## 0, 2), period = 12))
##
## Coefficients:
##          ar1      ar2      ar3      ar4      ar5      ar6      ar7      ar8
##      -0.0124  0.0179 -0.0020 -0.0773  0.0729 -0.0658 -0.0217  0.0232
## s.e.   0.0196  0.0227  0.0183  0.0296  0.0363  0.0253  0.0236  0.0236
##          ar9      ar10     ar11     ar12      sar1      sma1      sma2
##      -0.0118 -0.0710  0.0892  0.9091 -0.0100 -0.5788 -0.1180
## s.e.   0.0183  0.0283  0.0448  0.0413  0.3515  0.3665  0.2292
##      intercept
##           0.0010
## s.e.      0.0057
##
## sigma^2 estimated as 0.004342:  log likelihood = 580.41,  aic = -1128.83
```

Run diagnostics on model 7

```
chosenMod<-popModel7
modelString<-paste(getModelString(chosenMod), "with Population Data")

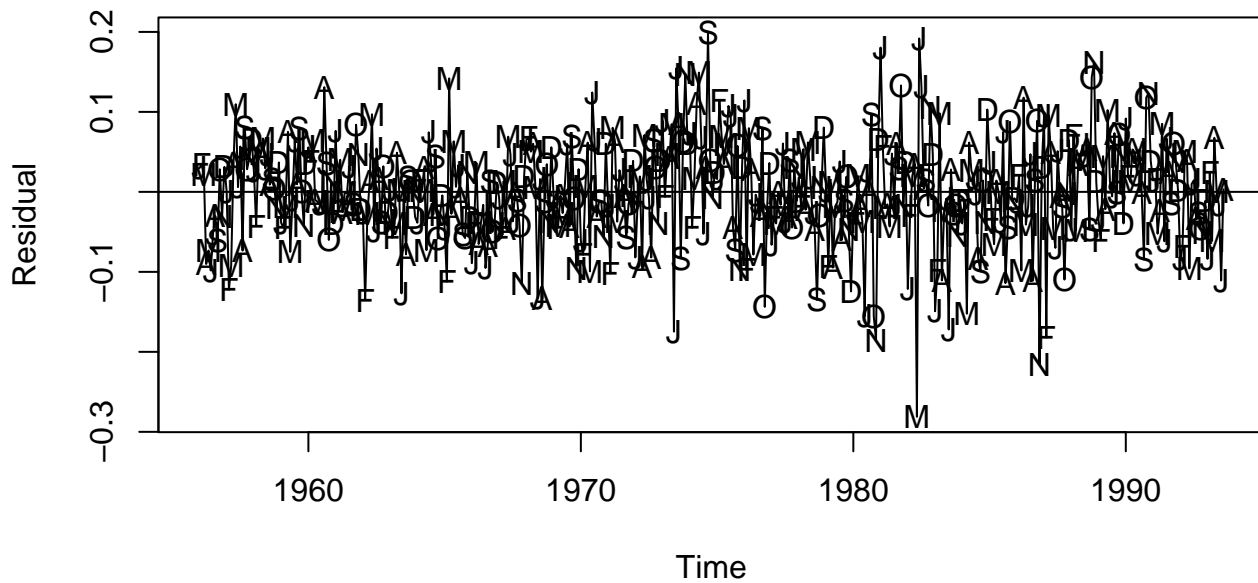
par(mfrow=c(1,1))
plot(residuals(chosenMod), main=paste("Residuals of Model", modelString), ylab="Residual")
abline(h=0)
```

Residuals of Model SARIMA(12,0,0)(1,0,1)[12] with Population Data



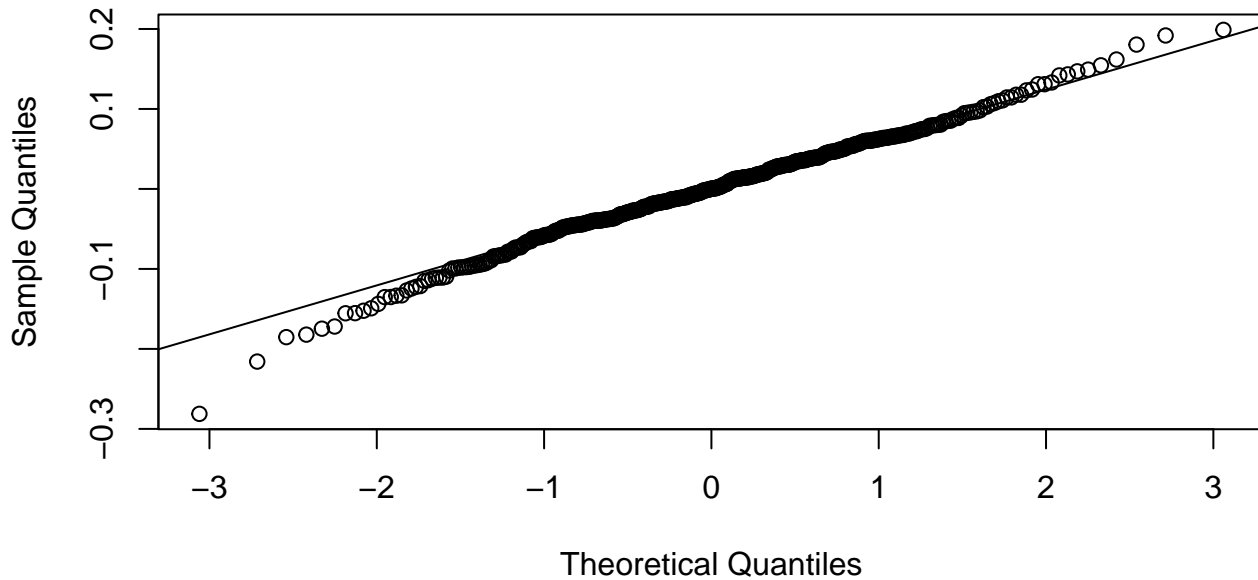
```
par(mfrow=c(1,1))
plot(residuals(chosenMod), main=paste("Residuals of Model", modelString), ylab="Residual", type="l")
points(y=residuals(chosenMod), x=time(residuals(chosenMod)), pch=as.vector(season(residuals(chosenMod))))
abline(h=0)
```

Residuals of Model SARIMA(12,0,0)(1,0,1)[12] with Population Data



```
par(mfrow=c(1,1))
qqnorm(residuals(chosenMod), main=paste("Normal QQ Plot of Residuals from", modelString))
qqline(residuals(chosenMod))
```

Normal QQ Plot of Residuals from SARIMA(12,0,0)(1,0,1)[12] with Population



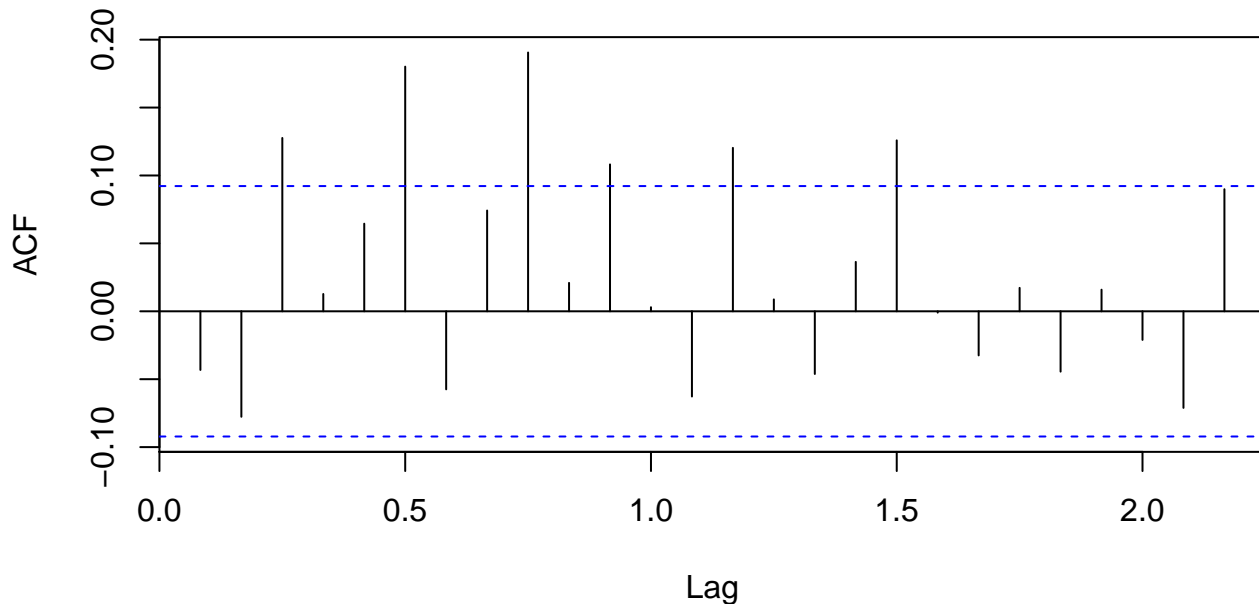
```
shapiro.test(residuals(chosenMod))
```

```
##
## Shapiro-Wilk normality test
##
```

```
## data: residuals(chosenMod)
## W = 0.9911, p-value = 0.008028
```

```
par(mfrow=c(1,1))
acf(residuals(chosenMod), main=paste("ACF of Residuals from", modelString))
```

ACF of Residuals from SARIMA(12,0,0)(1,0,1)[12] with Population Data



```
LB.test(chosenMod, lag=35)
```

```
##
## Box-Ljung test
##
## data: residuals from chosenMod
## X-squared = 87.347, df = 21, p-value = 4.611e-10
```

```
runs(residuals(chosenMod))
```

```
## $pvalue
## [1] 0.963
##
## $observed.runs
## [1] 227
##
## $expected.runs
## [1] 226.9956
##
## $n1
## [1] 225
##
## $n2
## [1] 227
##
## $k
## [1] 0
```

Set up for forecast of population model

```
startDate<-round(start(beer_forecast)[1]+start(beer_forecast)[2]/12,2)
endDate<-round(end(beer_forecast)[1]+end(beer_forecast)[2]/12,2)
numToFor<-length(beer_forecast)

allBeerData<-ts(c(beerTS, beer_forecast), start=c(1956, 1), frequency=12)

#Pull data for population
lag6_10_14new<-subset(monthlyLag6Long, as.numeric(rownames(monthlyLag6Long))>=startDate)
lag6_10_14new<-subset(lag6_10_14new, as.numeric(rownames(lag6_10_14new))<=endDate)

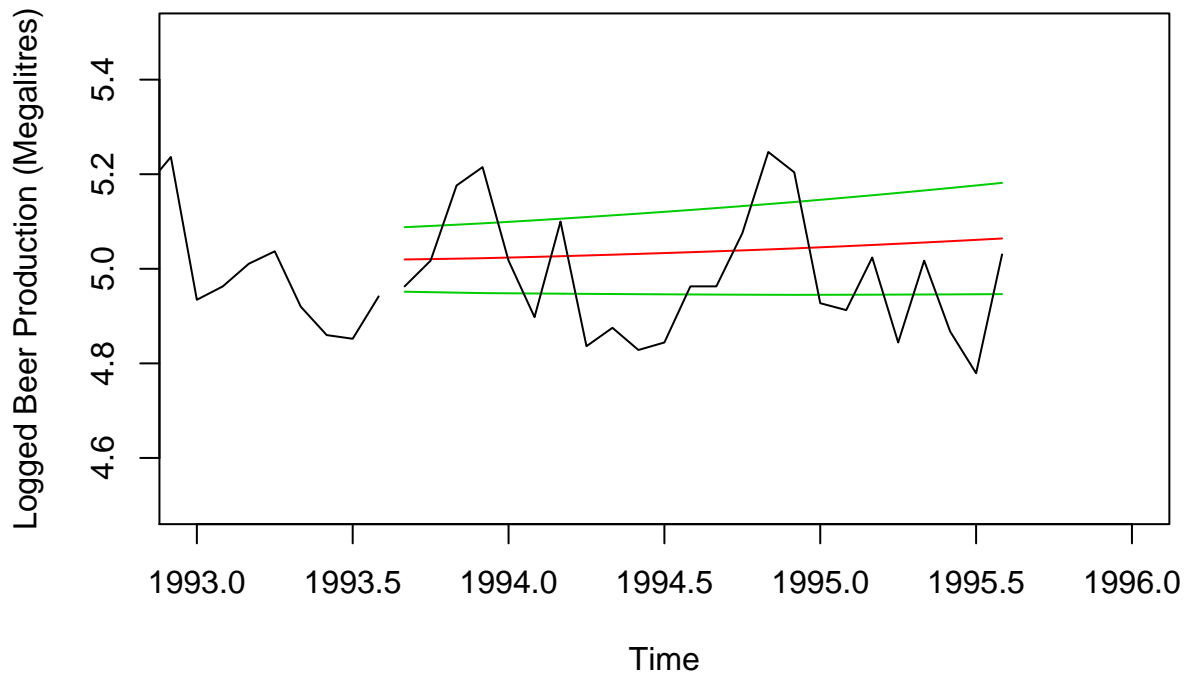
tnew<-1:length(allBeerData)
t2new<-tnew^2
t3new<-tnew^3
t4new<-tnew^4

newRegData<-data.frame(t=tnew, t2=t2new, t3=t3new, t4=t4new)
newRegData<-newRegData[(nrow(newRegData)-numToFor+1):nrow(newRegData),]
newRegData<-data.frame(lag6_10_14=lag6_10_14new, newRegData)
colnames(newRegData)<-c("lag6_10_14", colnames(newRegData)[2:5])

predFromPop<-predict(monthlyPopModel_log, newdata=newRegData, se.fit=TRUE)
uci_lm<-ts(predFromPop$fit+2*predFromPop$se.fit, start=c(1993, 9), frequency = 12)
lci_lm<-ts(predFromPop$fit-2*predFromPop$se.fit, start=c(1993, 9), frequency = 12)

ymin=min(c(as.vector(lci),logBeer))-1
ymax=max(c(as.vector(uci),logBeer))+1
plot(logBeer,xlim=c(1993, 1996), ylim=c(4.5,5.5),main=modelString, ylab='Logged Beer Production (Megalitres)')
lines(ts(predFromPop$fit, start=c(1993, 9), frequency = 12),col=2)
lines(uci_lm,col=3)
lines(lci_lm,col=3)
lines(log(beer_forecast), col="black")
```


SARIMA(12,0,0)(1,0,1)[12] with Population Data



Plot the model forecasts

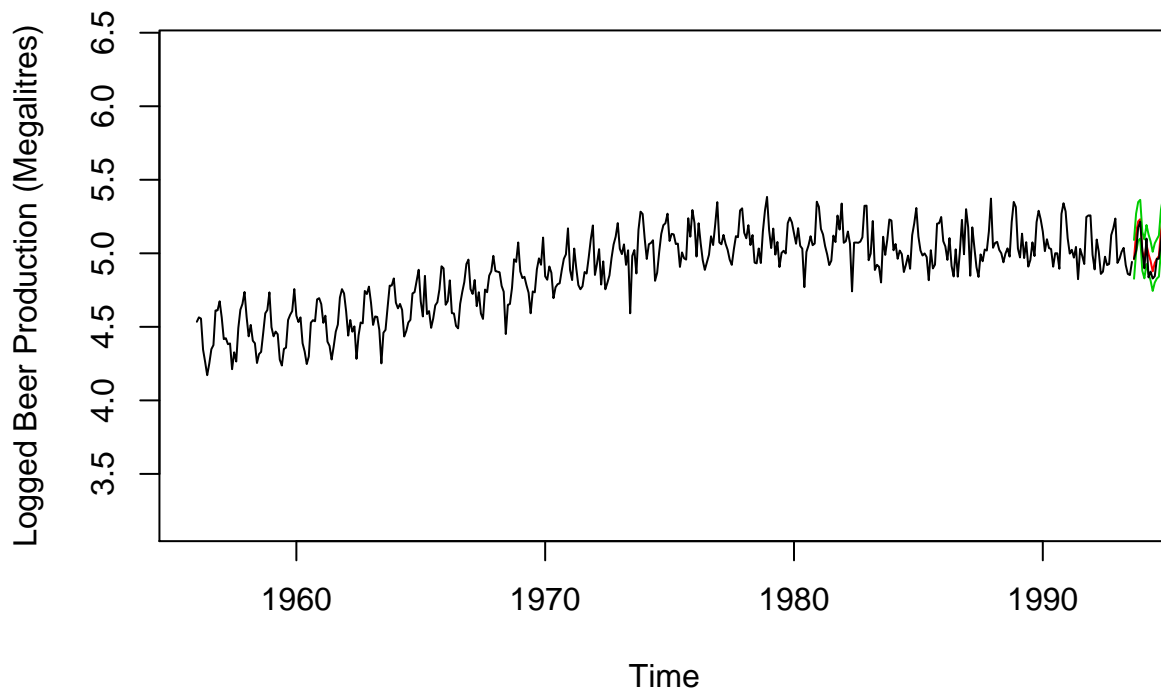
```
predictions<-predict(chosenMod, n.ahead=24)

pred_raw<-(predictions$pred+predFromPop$fit)
pred<-predictions$pred+predFromPop$fit
uci<-pred_raw+2*predictions$se
lci<-pred_raw-2*predictions$se
uci_lm<-pred_raw+2*(predictions$se+predFromPop$se.fit)
lci_lm<-pred_raw-2*(predictions$se+predFromPop$se.fit)

sumSqErrSARIMA<-sum((log(beer_forecast)-pred)^2)

ymin=min(c(as.vector(lci),logBeer))-1
ymax=max(c(as.vector(uci),logBeer))+1
plot(logBeer,ylim=c(ymin,ymax),main=modelString, ylab='Logged Beer Production (Megalitres)')
lines(pred,col=2)
lines(uci,col=3)
lines(lci,col=3)
lines(log(beer_forecast), col="black")
```

SARIMA(12,0,0)(1,0,1)[12] with Population Data



```
ymin=min(c(as.vector(lci),log(beer_forecast)))-0.1
ymax=max(c(as.vector(uci),log(beer_forecast)))+0.1
plot(logBeer,xlim=c(1993, 1996), ylim=c(ymin,ymax),main=modelString, ylab='Logged Beer Production (Megalitres)')
lines(pred,col=2)
lines(uci,col=3)
lines(lci,col=3)
lines(uci_lm,col="blue")
lines(lci_lm,col="blue")
lines(log(beer_forecast), col="black")
```

SARIMA(12,0,0)(1,0,1)[12] with Population Data

