EC405 Paper

Reetom Gangopadhyay

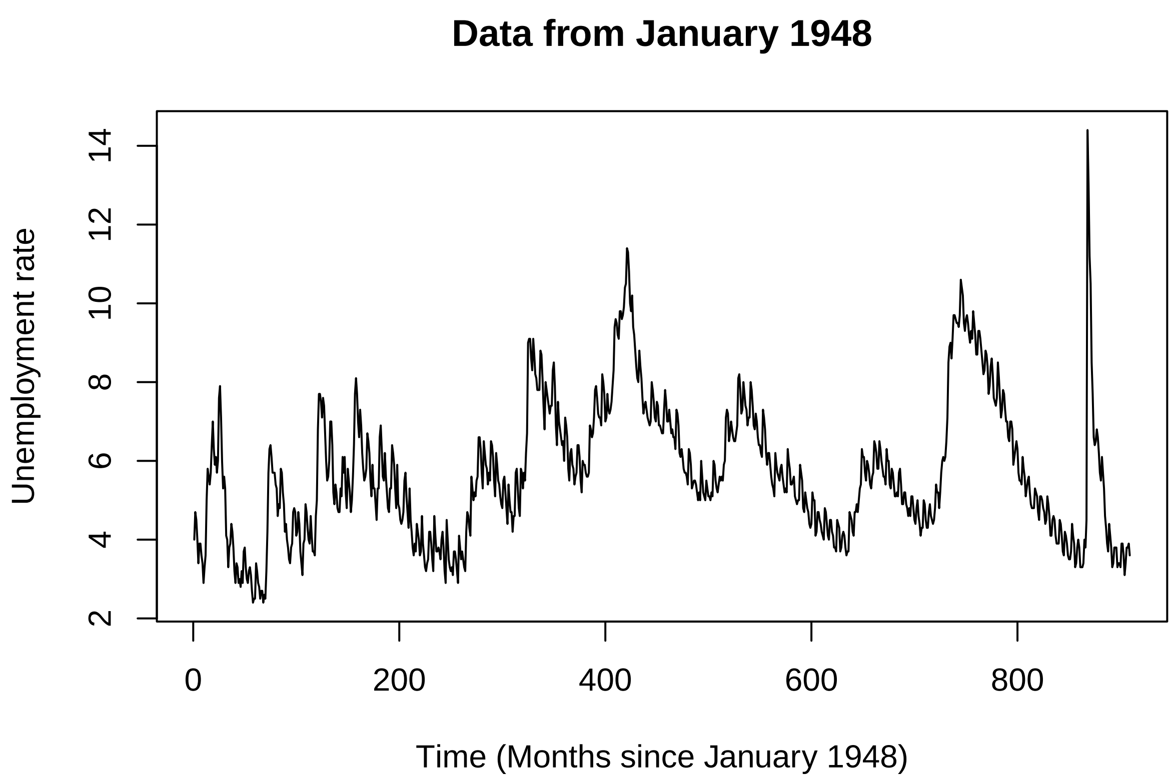
2023-10-24

```r
data <- read.csv('unemployment.csv')

data$DATE <- as.Date(data$DATE)

# Subset the data to include only the period from January 2010 onwards
data_2000 <- data[data$DATE >= as.Date("2000-01-01"), ]

# data_2000
```

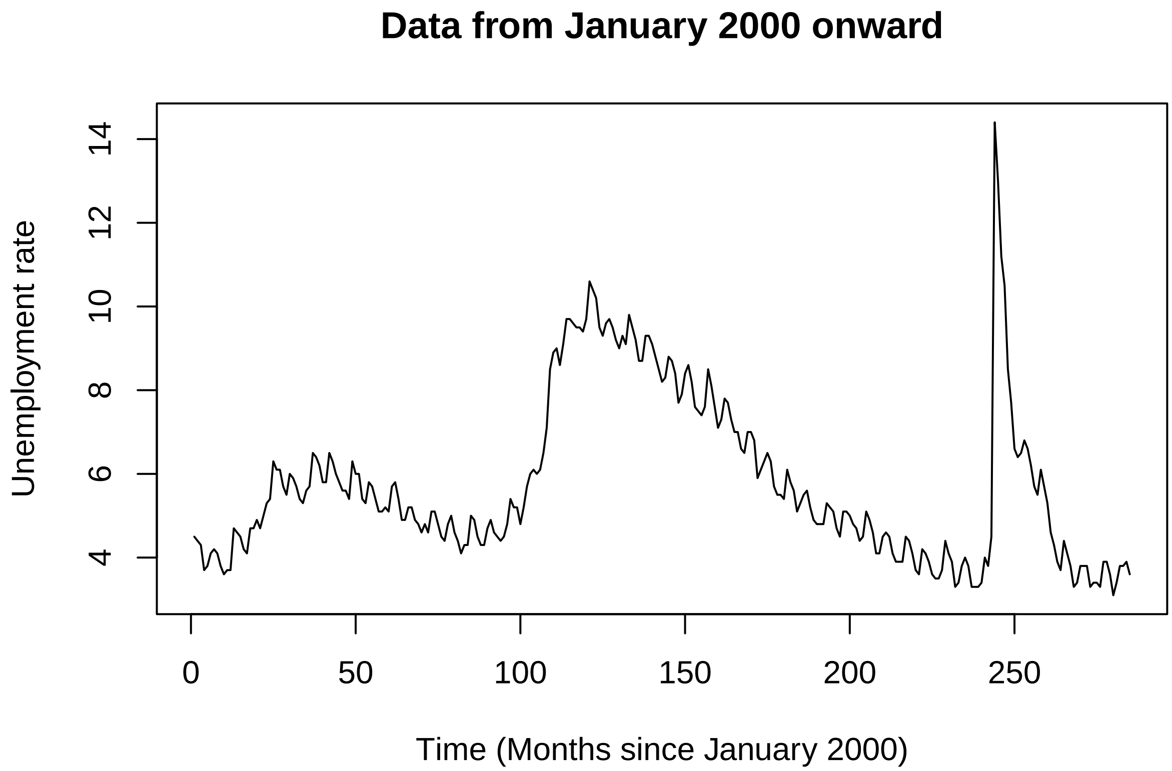
```r
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```r
plot.ts(data$UNRATENSA, main = "Data from January 1948", ylab = "Unemployment rate",xlab="Time (Months s
```

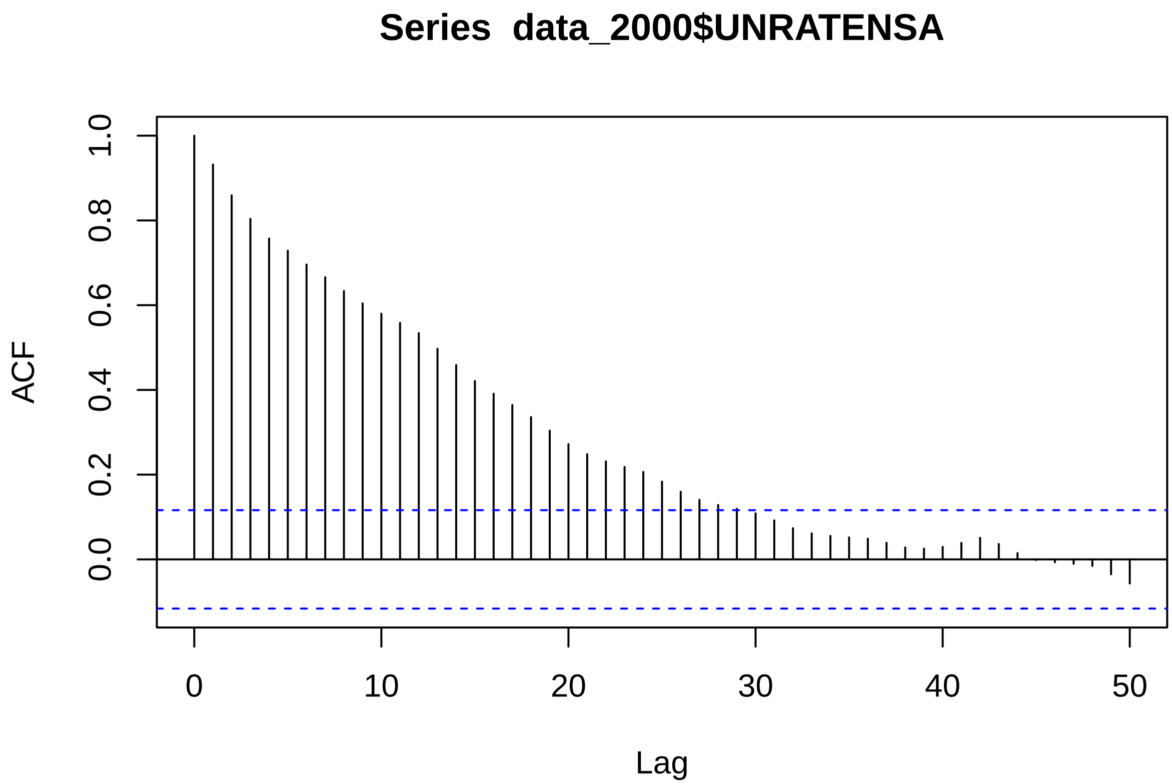



```r
plot.ts(data_2000$UNRATENSA, main = "Data from January 2000 onward",ylab = "Unemployment rate",xlab="Ti
```

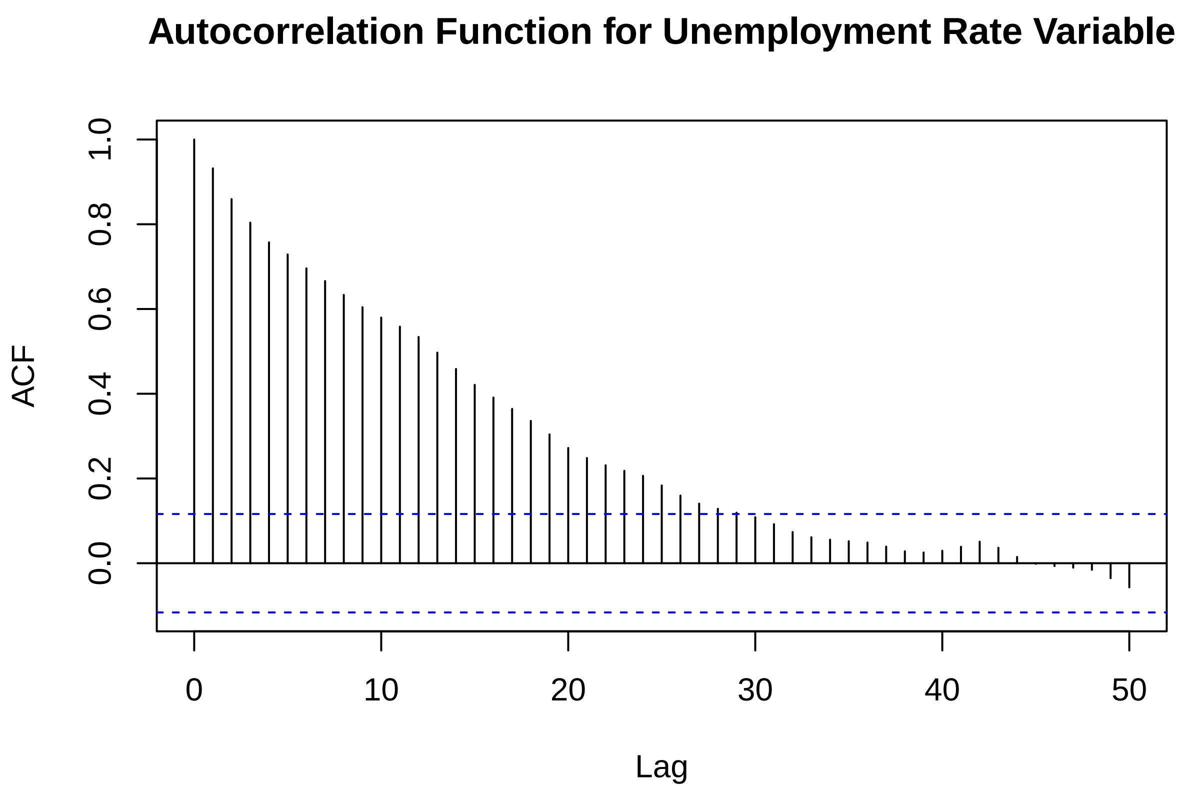



```r
# mod <- auto.arima(data$UNRATENSA)

# set.seed(123)
# model <- arima.sim(n = 100, list(order = c(5,1,1), ar = c(0.2, -0.3, 0.4, -0.1, 0.2), ma = 0.5))

# Plot the simulated ARIMA(5,1,1) model
# lines(model, col = "red")
```

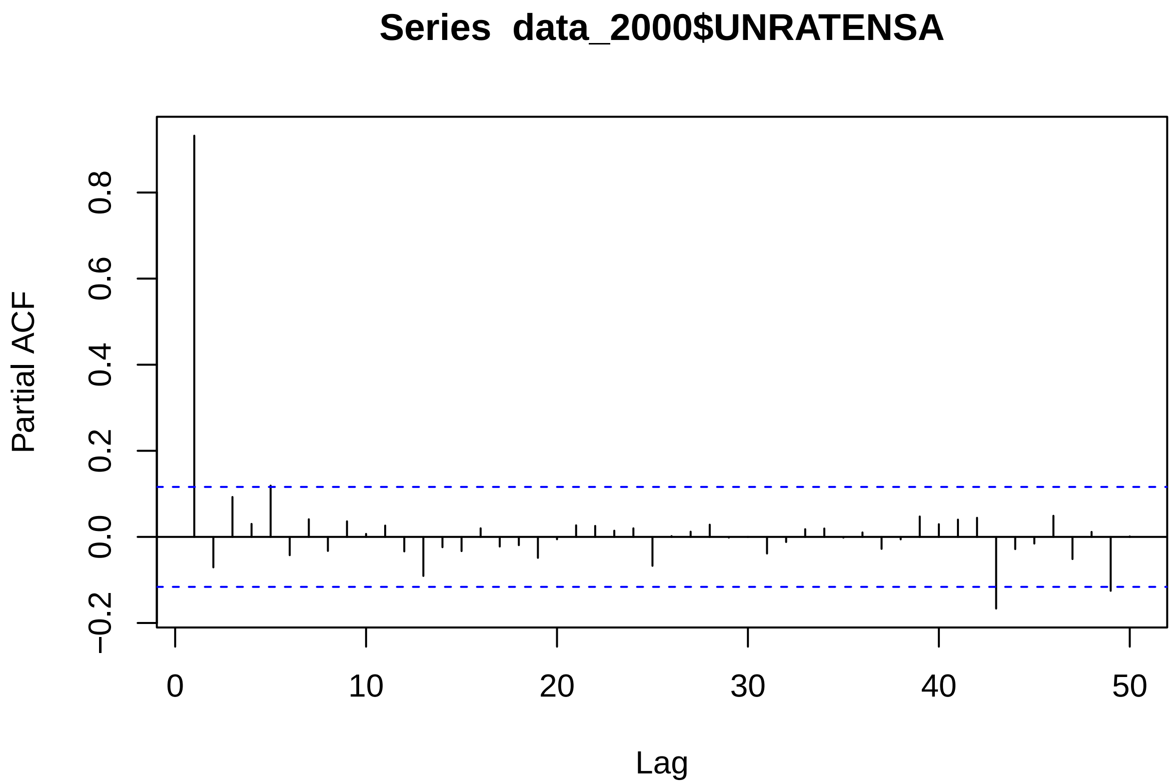
```r
acf_result <- acf(data_2000$UNRATENSA, lag.max = 50)
```



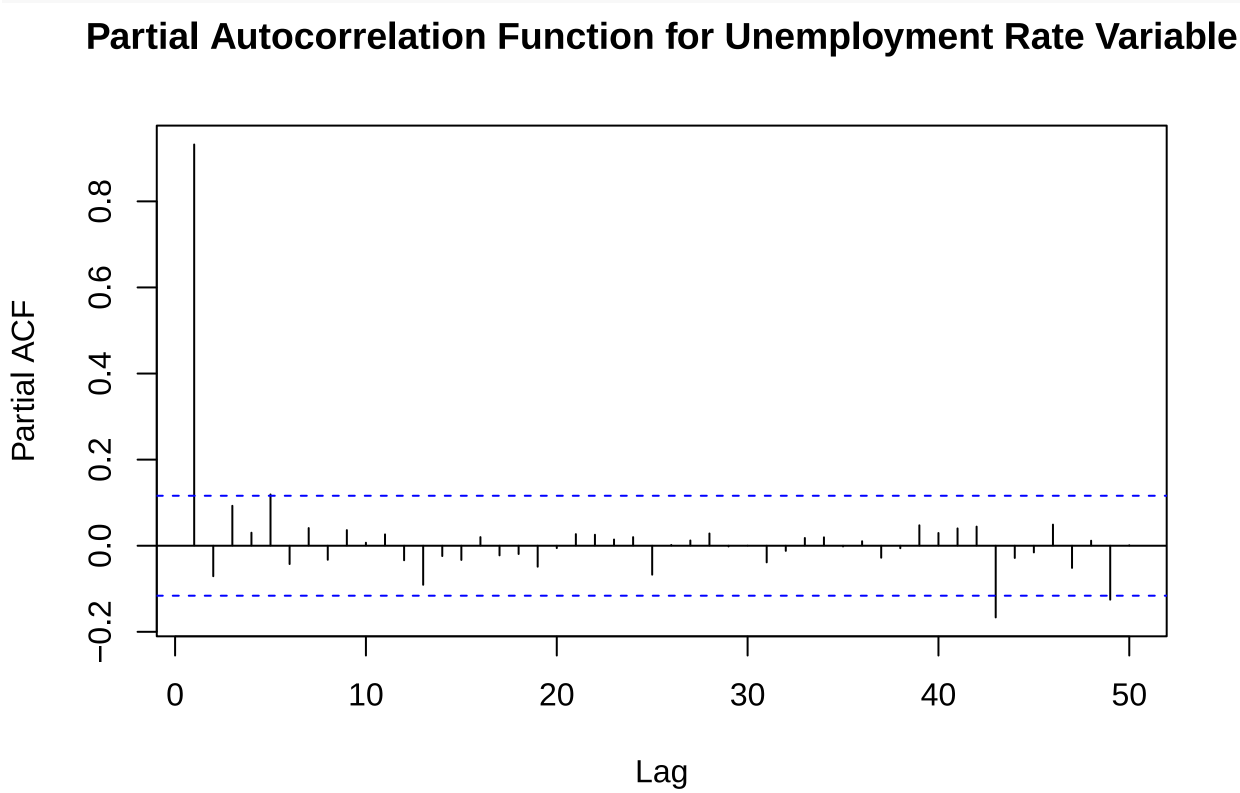

```r
plot(acf_result, main = "Autocorrelation Function for Unemployment Rate Variable")
```



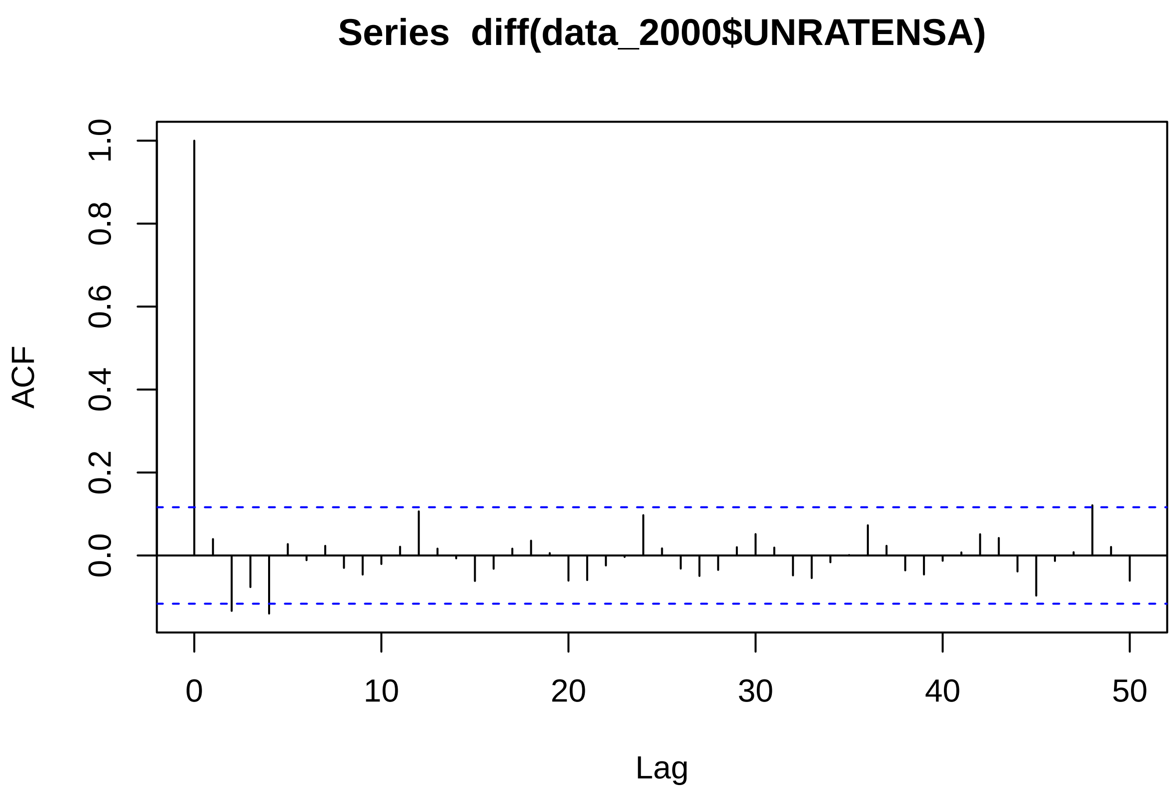

```r
pacf_result <- pacf(data_2000$UNRATENSA, lag.max = 50)
```



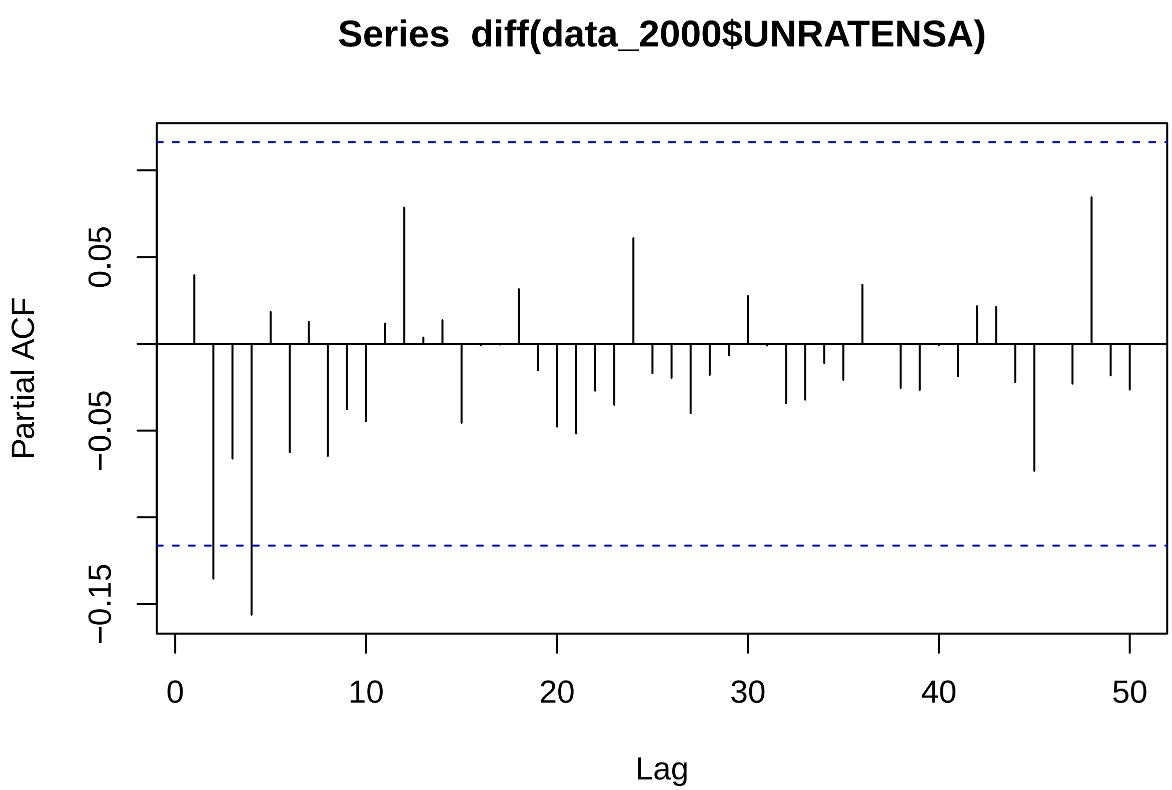

```
plot(pacf_result,main = "Partial Autocorrelation Function for Unemployment Rate Variable")
```



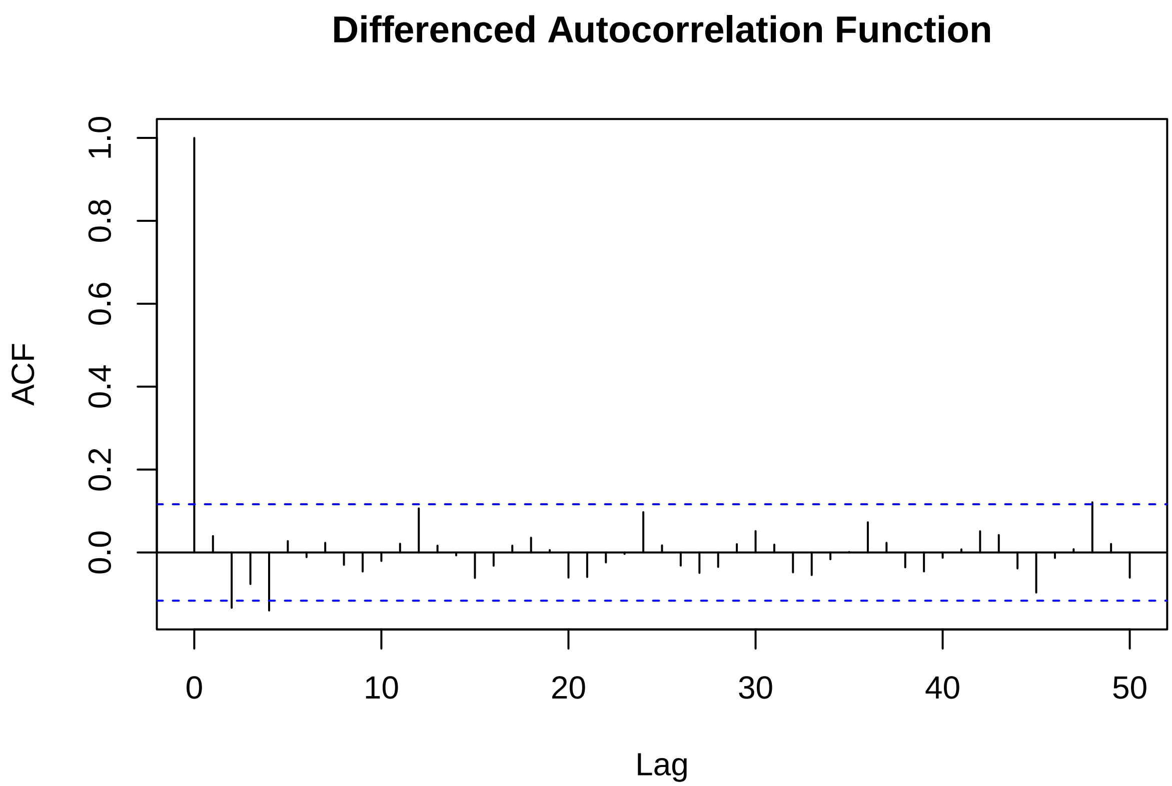

```
diff_acf <- acf(diff(data_2000$UNRATENSA), lag.max = 50)
```



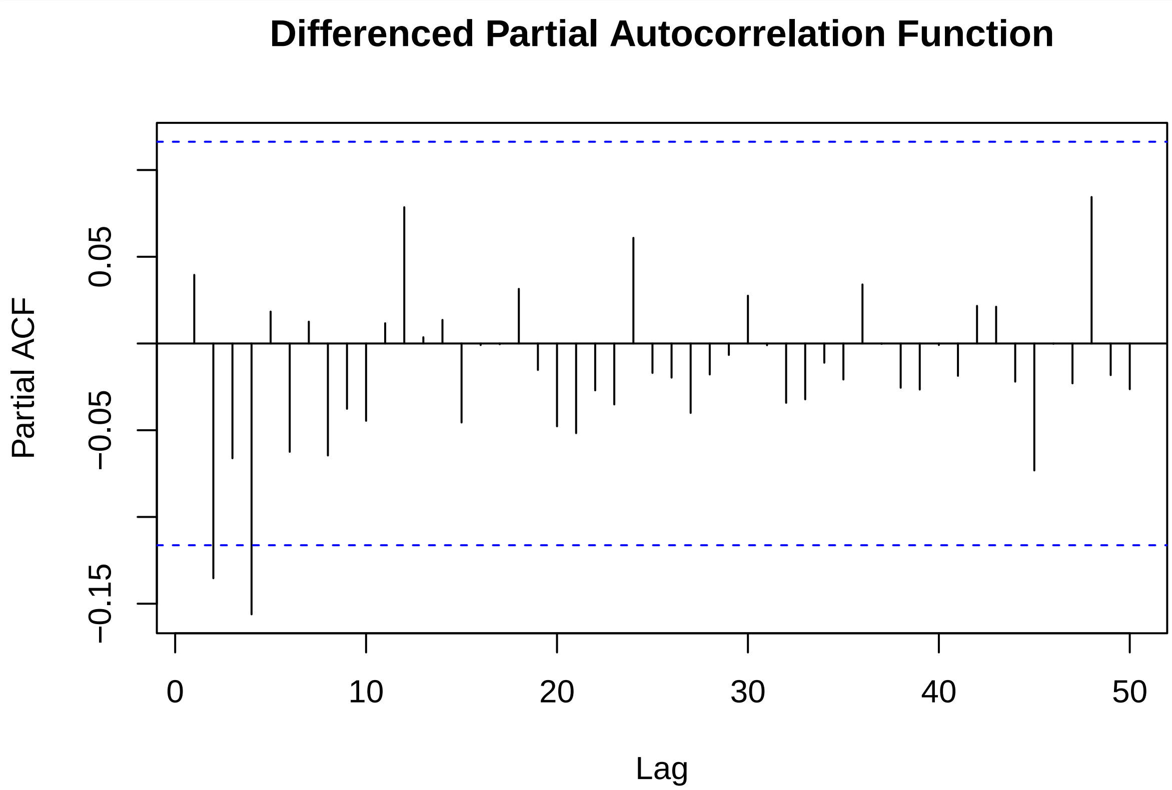

```
diff_pacf <- pacf(diff(data_2000$UNRATENSA), lag.max = 50)
```




```
plot(diff_acf, main = "Differenced Autocorrelation Function")
```




```r
plot(diff_pacf,main = "Differenced Partial Autocorrelation Function")
```

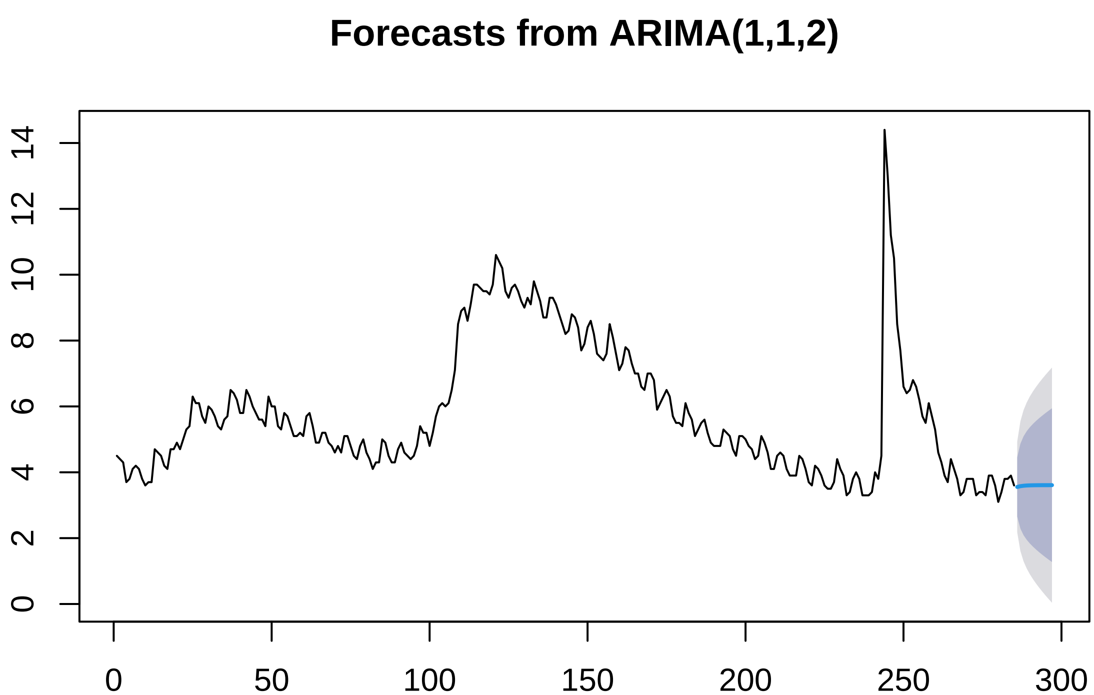



```r
fit <- auto.arima(data_2000$UNRATENSA)
fullfit <- auto.arima(data$UNRATENSA)
```

```r
summary(fit)
```

```
## Series: data_2000$UNRATENSA
## ARIMA(1,1,2)
##
## Coefficients:
##          ar1      ma1      ma2
##       0.5773  -0.5578  -0.1778
## s.e.  0.1818   0.1856   0.0682
##
## sigma^2 = 0.4953:  log likelihood = -301.76
## AIC=611.53   AICc=611.67   BIC=626.13
##
## Training set error measures:
##                        ME      RMSE       MAE        MPE     MAPE      MASE
## Training set -0.00476489 0.6988179 0.3217766 -0.7039407 5.455424 0.9629563
##                        ACF1
## Training set -0.0009124731
```

```r
tsdiag(fit)
```




```r
cat("\nar1:",0.5773/0.1818)
```

```
##
## ar1: 3.175468
```

```r
cat("\nma1:",0.5578/0.1856)
```

```
##
## ma1: 3.005388
```

```r
cat("\nma2",0.1778/0.0682)
```

```
##
## ma2 2.607038
```

```r
Box.test(residuals(fit), lag = 20, type = "Ljung")
```

```
##
##  Box-Ljung test
##
## data:  residuals(fit)
## X-squared = 10.047, df = 20, p-value = 0.9673
```

```r
library(tseries)
adf.test(data_2000$UNRATENSA)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  data_2000$UNRATENSA
## Dickey-Fuller = -2.3087, Lag order = 6, p-value = 0.446
## alternative hypothesis: stationary
```

Dataset contains 2 things: date and % unemployed from FRED.

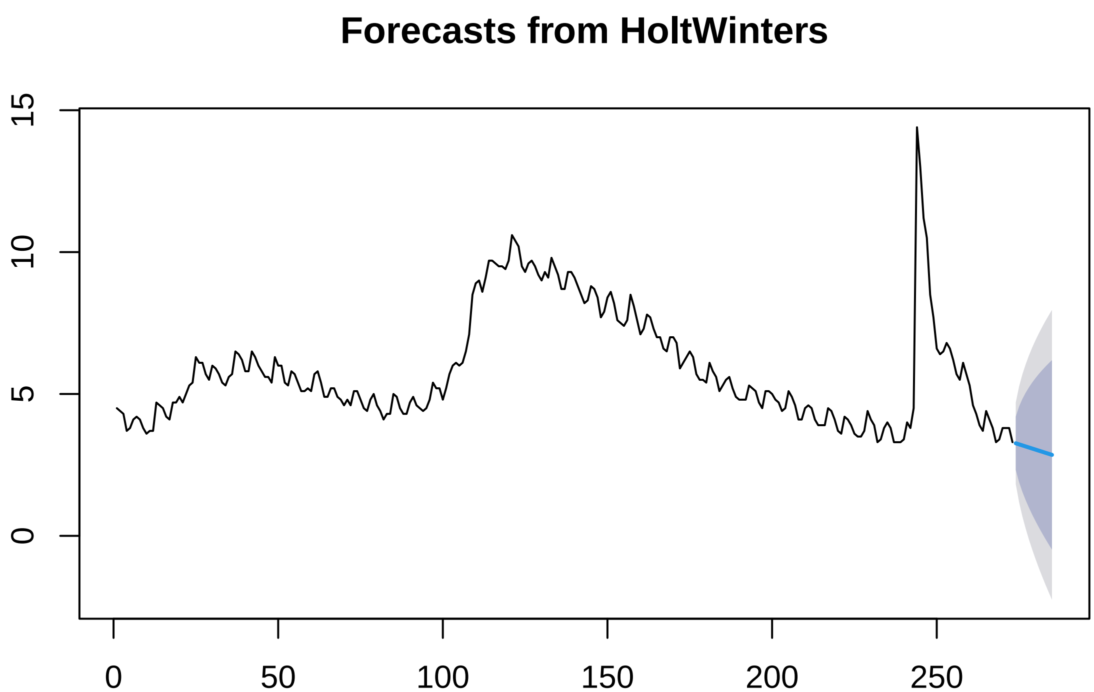
```r
kpss_result <- kpss.test(data_2000$UNRATENSA)


kpss_result
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  data_2000$UNRATENSA
## KPSS Level = 0.59276, Truncation lag parameter = 5, p-value = 0.02329
```

```r
pp_result <- pp.test(data_2000$UNRATENSA)

pp_result
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  data_2000$UNRATENSA
## Dickey-Fuller Z(alpha) = -16.325, Truncation lag parameter = 5, p-value
## = 0.1947
## alternative hypothesis: stationary
```

```r
train <- data_2000[1:273, "UNRATENSA"]
test <- data_2000[274:285,"UNRATENSA"]

library(forecast)

holt_2000 <- HoltWinters(data_2000$UNRATENSA,gamma = F)
forecast_2000 <- forecast(holt_2000,h=12)
#plot(forecast_2000)



train_holt <- HoltWinters(train,gamma=F)
forecast_train <- forecast(train_holt,h=12)
plot(forecast_train)
```




```r
HWerr = test-forecast_train$mean
HWrmse = sqrt(mean(HWerr^2))
HWmae = mean(abs(HWerr))
HWmape = mean(abs((HWerr*100)/test))


cat("Errors: ",HWerr)
```

```
## Errors:  0.1371416 0.1742831 0.1114247 0.7485663 0.7857078 0.5228494 0.05999097 0.3971325 0.8342741
```

```r
cat("\nMAE: ",HWmae)
```

```
##
## MAE:  0.5330869
```

```r
cat("\nMAPE: ",HWmape)
```

```
##
## MAPE:  14.2897
```

```r
cat("\nRMSE: ",HWrmse)
```

```
##
## RMSE:  0.6260061
```

```r
forecast_train
```

```
##     Point Forecast       Lo 80    Hi 80      Lo 95    Hi 95
## 274       3.262858  2.32382280 4.201894  1.8267274 4.698989
## 275       3.225717  1.89448679 4.556947  1.1897762 5.261658
## 276       3.188575  1.55419514 4.822955  0.6890066 5.688144
## 277       3.151434  1.25963444 5.043233  0.2581764 6.044691
## 278       3.114292  0.99407202 5.234512 -0.1283047 6.356889
## 279       3.077151  0.74894809 5.405353 -0.4835279 6.637829
## 280       3.040009  0.51919040 5.560828 -0.8152504 6.895268
## 281       3.002867  0.30150222 5.704233 -1.1285142 7.134249
```

```
## 282       2.965726  0.09359926 5.837853 -1.4268127 7.358265
## 283       2.928584 -0.10617762 5.963346 -1.7126836 7.569852
## 284       2.891443 -0.29907786 6.081963 -1.9880374 7.770923
## 285       2.854301 -0.48606979 6.194672 -2.2543554 7.962958
```

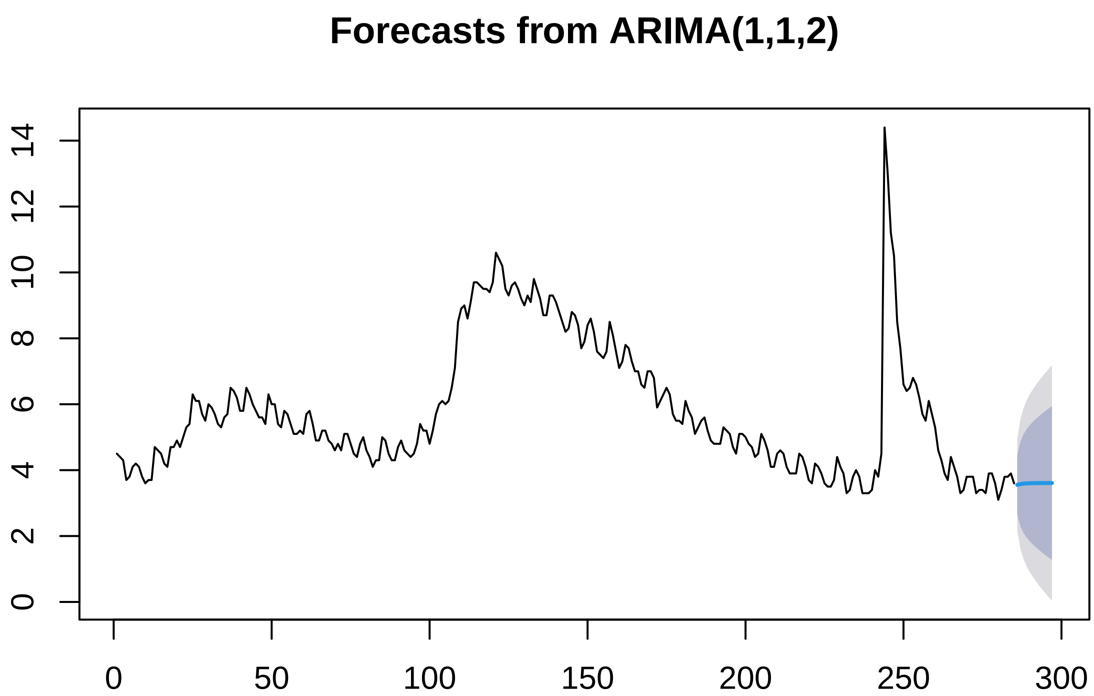
```r
auto.arima(train)
```

```
## Series: train
## ARIMA(1,1,2)
##
## Coefficients:
##          ar1      ma1      ma2
##       0.5811  -0.5606  -0.1760
## s.e.  0.1866   0.1907   0.0696
##
## sigma^2 = 0.514:  log likelihood = -294
## AIC=596   AICc=596.15   BIC=610.43
```

```r
train_fit <- arima(data_2000$UNRATENSA,order = c(1,1,2))
train_cast <- forecast(fit,h=12)

train_cast
```

```
##     Point Forecast    Lo 80    Hi 80      Lo 95    Hi 95
## 286       3.554715 2.652792 4.456638 2.17534296 4.934087
## 287       3.577746 2.289726 4.865766 1.60788917 5.547602
## 288       3.591042 2.090731 5.091352 1.29651487 5.885568
## 289       3.598717 1.950386 5.247048 1.07781243 6.119622
## 290       3.603149 1.837579 5.368718 0.90294298 6.303354
## 291       3.605707 1.739760 5.471653 0.75198797 6.459426
## 292       3.607184 1.651061 5.563306 0.61555253 6.598815
## 293       3.608036 1.568462 5.647610 0.48877734 6.727295
## 294       3.608528 1.490294 5.726763 0.36896794 6.848089
## 295       3.608813 1.415569 5.802056 0.25453659 6.963089
## 296       3.608977 1.343671 5.874282 0.14449075 7.073463
## 297       3.609071 1.274186 5.943957 0.03817171 7.179971
```

```r
plot(train_cast)
```




```
err = test-train_cast$mean
rmse = sqrt(mean(err^2))
mae = mean(abs(err))
mape = mean(abs((err*100)/test))


cat("\nErrors: ",err)
```

```
##
## Errors:  -0.1547148 -0.1777457 -0.2910415 0.3012827 0.2968514 -0.005706771 -0.5071836 -0.2080362 0.1
```

```
cat("\nMAE: ",mae)
```

```
##
## MAE:  0.2187764
```

```
cat("\nMAPE: ",mape)
```

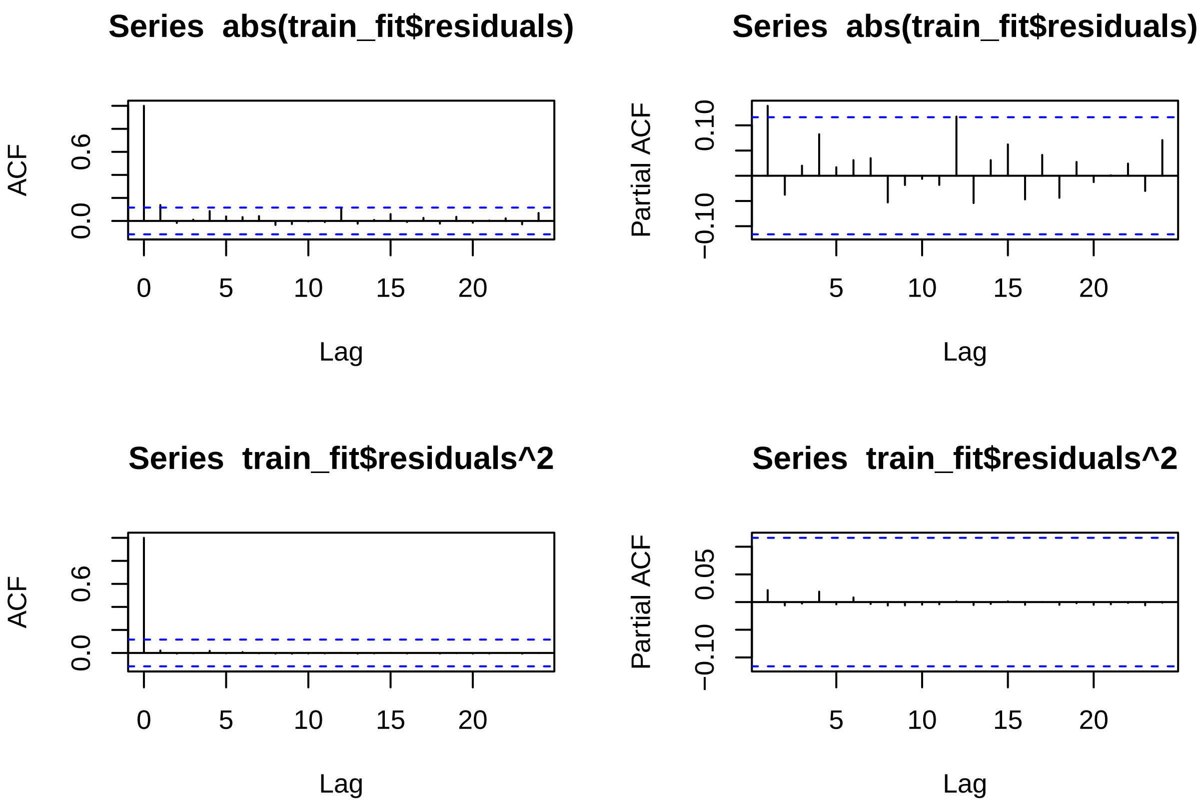
```
##
## MAPE:  6.196379
```

```
cat("\nRMSE: ",rmse)
```

```
##
## RMSE:  0.2545691
```

ARIMA is better forecast

```
par(mfrow = c(2,2))
acf(abs(train_fit$residuals))
pacf(abs(train_fit$residuals))
acf(train_fit$residuals^2)
pacf(train_fit$residuals^2)
```




```r
library(fGarch)
```

```
## NOTE: Packages 'fBasics', 'timeDate', and 'timeSeries' are no longer
## attached to the search() path when 'fGarch' is attached.
##
## If needed attach them yourself in your R script by e.g.,
##         require("timeSeries")
```

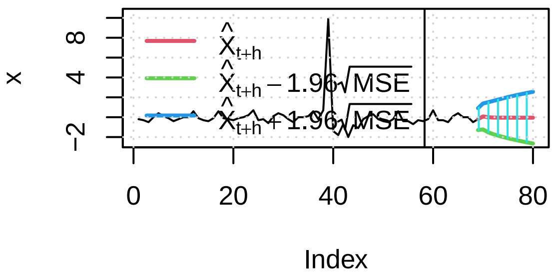
```r
fitgr = garchFit(formula ~ arma(1,2) + garch(1,1),data = diff(train), trace = F)

# plot(fitgr)

pred_garch <- predict(fitgr,n.ahead=12,plot=T)

print(pred_garch)
```

```
##    meanForecast meanError standardDeviation lowerInterval upperInterval
## 1   -0.19481520 0.5618006         0.5618006     -1.295924     0.9062937
## 2    0.07759353 0.6665016         0.6540633     -1.228726     1.3839127
## 3   -0.00723056 0.7698103         0.7348317     -1.516031     1.5015699
## 4   -0.03479320 0.8512485         0.8075621     -1.703210     1.6336233
## 5   -0.04374937 0.9246146         0.8742628     -1.855961     1.7684619
## 6   -0.04665958 0.9924806         0.9362234     -1.991886     1.8985666
## 7   -0.04760522 1.0559848         0.9943305     -2.117297     2.0220871
## 8   -0.04791249 1.1158800         1.0492245     -2.234997     2.1391722
## 9   -0.04801234 1.1727201         1.1013859     -2.346501     2.2504768
## 10  -0.04804478 1.2269297         1.1511863     -2.452783     2.3566932
## 11  -0.04805532 1.2788434         1.1989198     -2.554542     2.4584317
## 12  -0.04805875 1.3287304         1.2448243     -2.652322     2.5562050
```

```r
fitgr
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = formula ~ arma(1, 2) + garch(1, 1), data = diff(train),
##     trace = F)
##
## Mean and Variance Equation:
##  data ~ arma(1, 2) + garch(1, 1)
## <environment: 0x151518d58>
##  [data = diff(train)]
##
## Conditional Distribution:
##  norm
##
## Coefficient(s):
##          mu          ar1          ma1          ma2        omega       alpha1
## -0.03244371   0.32493880  -0.09681060  -0.38430867   0.11217889   0.99999999
##       beta1
##  0.00000001
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##          Estimate  Std. Error  t value Pr(>|t|)
## mu     -3.244e-02   1.385e-02   -2.342   0.0192 *
## ar1     3.249e-01   1.434e-01    2.266   0.0234 *
## ma1    -9.681e-02   8.985e-02   -1.077   0.2813
## ma2    -3.843e-01   6.409e-02   -5.997 2.02e-09 ***
## omega   1.122e-01   2.837e-02    3.955 7.67e-05 ***
## alpha1  1.000e+00   1.461e-01    6.847 7.57e-12 ***
## beta1   1.000e-08   1.123e-01    0.000   1.0000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  -189.7581    normalized:  -0.6976402
##
## Description:
##  Mon Dec 18 19:32:39 2023 by user:
```

Prediction with confidence intervals



Lack of significance on lags so don't fit GARCH

```r
library(rugarch)
```

```
## Loading required package: parallel

##
## Attaching package: 'rugarch'

## The following object is masked from 'package:stats':
##
##     sigma
```

```r
library(forecast)

combined_model_spec <- ugarchspec(mean.model = list(armaOrder = c(1, 1, 2)), variance.model = list(garc

combined_model <- ugarchfit(spec = combined_model_spec, data = train)

garchcast <- ugarchforecast(combined_model, n.ahead = 12,data=train)

forecasted_std_dev <- sigma(garchcast)

# Calculate the error between the actual and forecasted values
err <- test - forecasted_std_dev

# Calculate RMSE
rmse <- sqrt(mean(err^2))

# Calculate MAE
mae <- mean(abs(err))

# Calculate MAPE
mape <- mean(abs((err * 100) / test))

print(err)
```

```
##      1970-09-30 20:00:00
## T+1             2.965606
## T+2             2.963090
## T+3             2.861621
## T+4             3.460760
## T+5             3.460256
## T+6             3.159961
## T+7             2.659788
```

```
## T+8            2.959686
## T+9            3.359626
## T+10           3.359591
## T+11           3.459571
## T+12           3.159559
```

```
cat("\nmae:",mae)
```

```
##
## mae: 3.152426
```

```
cat("\nmape:",mape)
```

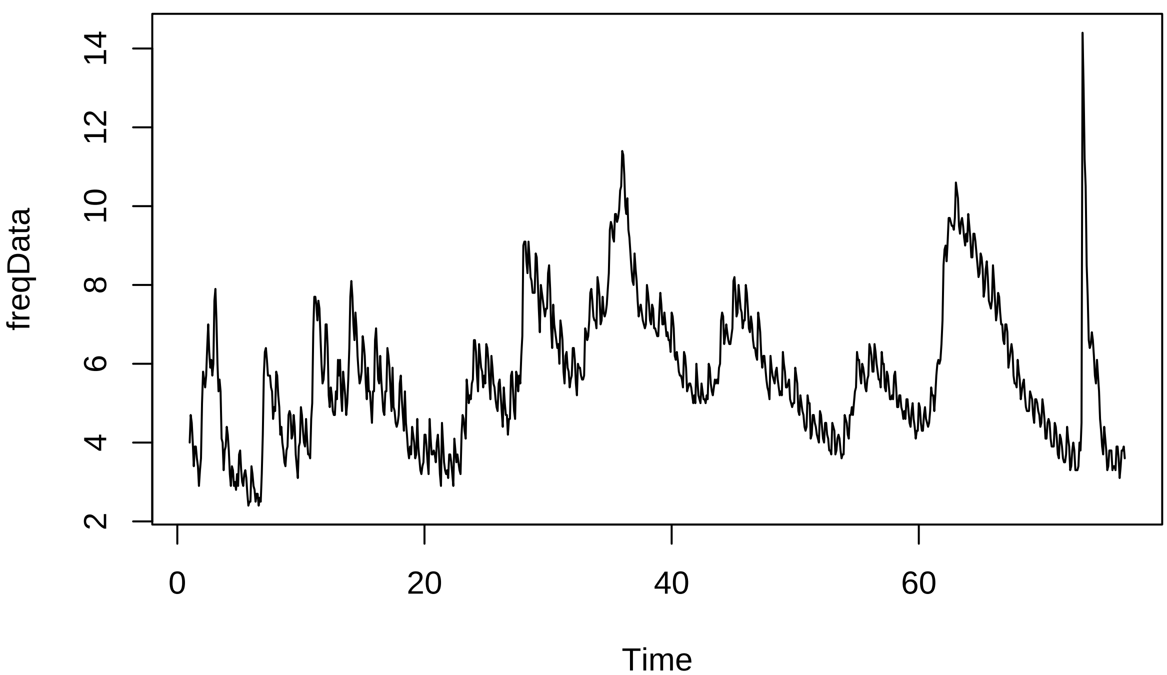
```
##
## mape: 87.70592
```

```
cat("\nrmse:",rmse)
```

```
##
## rmse: 3.163052
```

```
# plot(garchcast)
```

```
freqData <- ts(data$UNRATENSA,frequency = 12)

season_fit <- auto.arima(freqData)

plot.ts(freqData)
```




```
print(season_fit)
```

```
## Series: freqData
## ARIMA(1,1,1)(2,0,0)[12]
##
## Coefficients:
##          ar1     ma1    sar1    sar2
```

```
##        -0.8428  0.9140  0.3243  0.3047
## s.e.    0.0522  0.0392  0.0321  0.0317
##
## sigma^2 = 0.2349:  log likelihood = -631.39
## AIC=1272.79   AICc=1272.86   BIC=1296.84
```

```r
cat("\nar1:",0.8428/0.0522)
```

```
##
## ar1: 16.14559
```

```r
cat("\nma1:",0.9140/0.0392)
```

```
##
## ma1: 23.31633
```

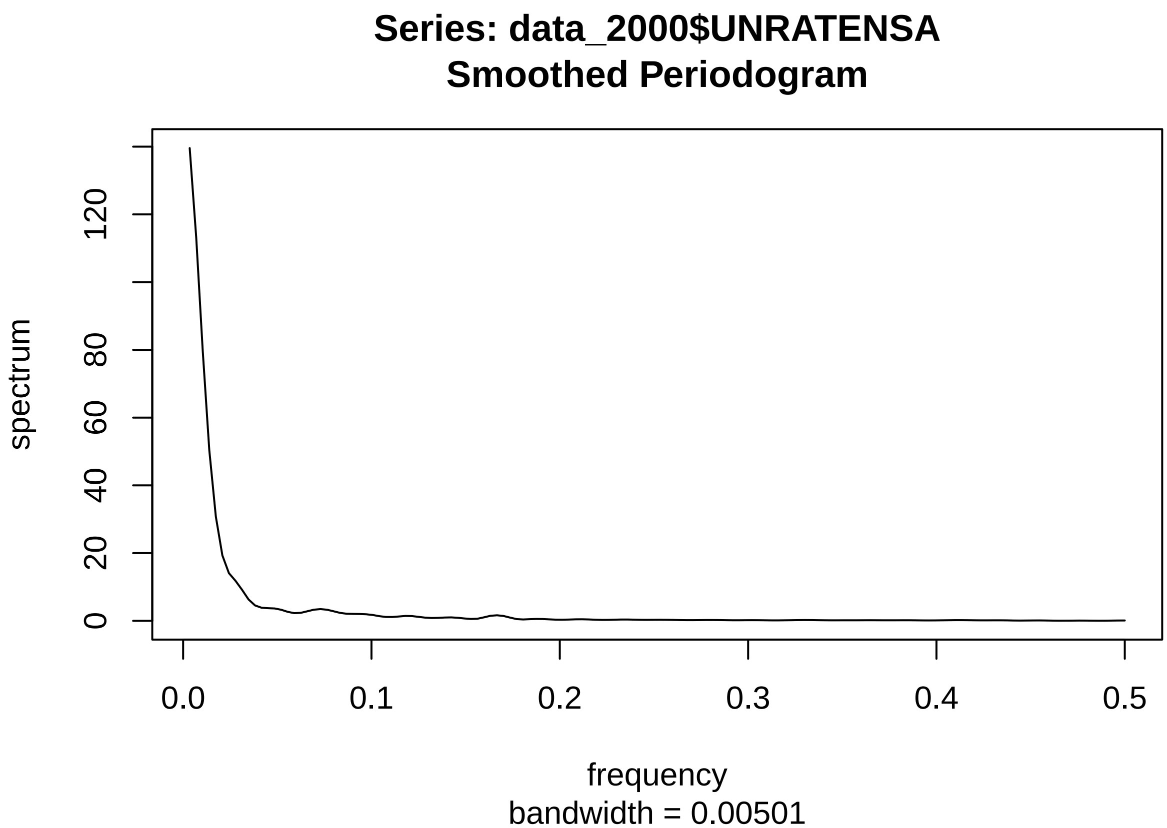
```r
cat("\nsar1",0.3243/0.0321)
```

```
##
## sar1 10.1028
```

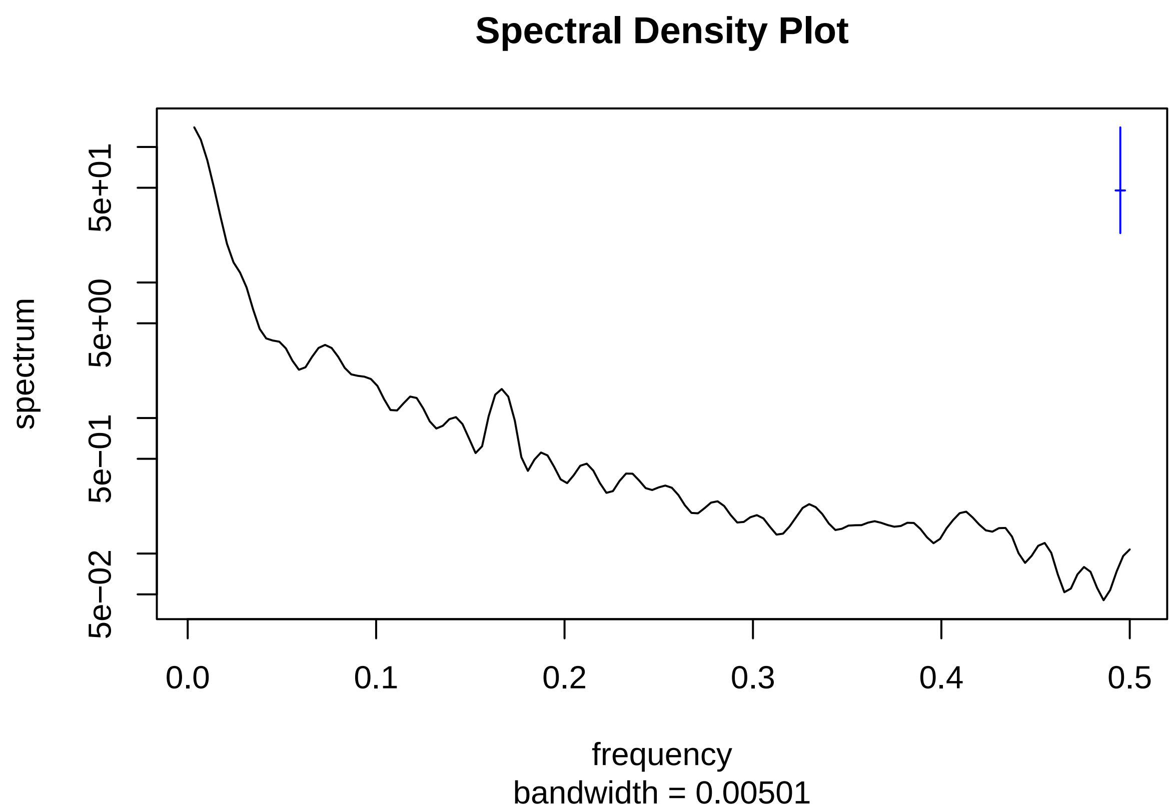
```r
cat("\nsar2",0.3047/0.0317)
```

```
##
## sar2 9.611987
```

```r
spec_result <- spec.pgram(data_2000$UNRATENSA, taper = 0, log = "no", span =c(3,5))
```




```r
plot(spec_result, main = "Spectral Density Plot")
```

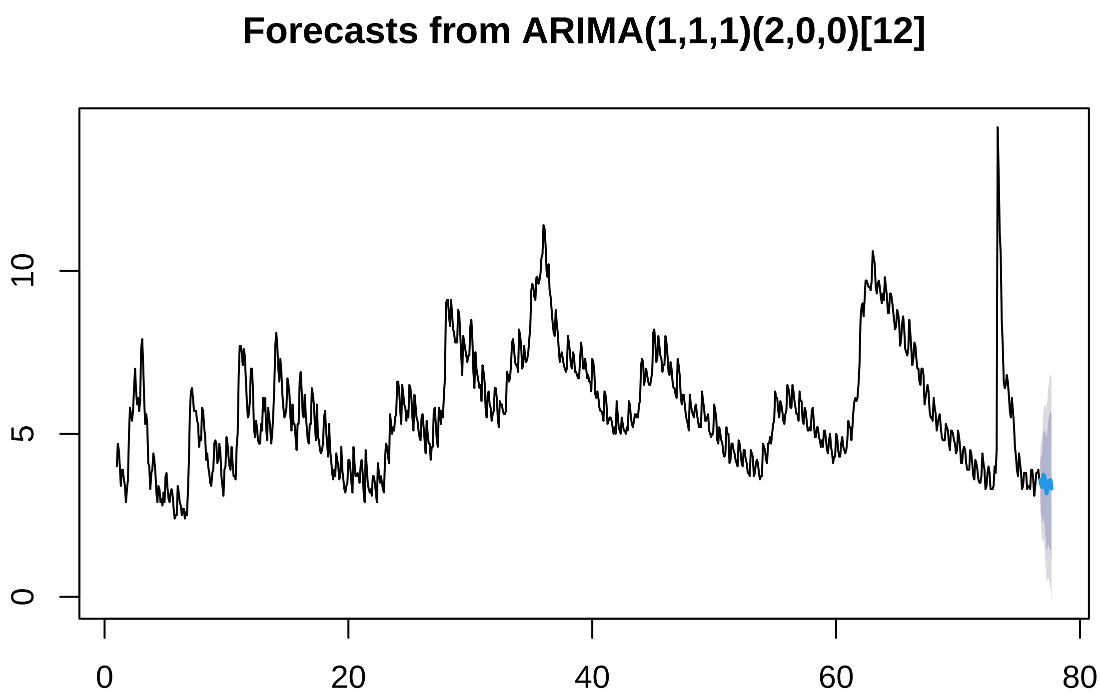


FORECAST SEASON

```
train_fit_s <- arima(freqData,order = c(1,1,1),seasonal = list(order = c(2,0,0), frequency = 12))
train_cast_s <- forecast(season_fit,h=12)

train_cast_s
```

```
##          Point Forecast    Lo 80    Hi 80       Lo 95    Hi 95
## Oct 76       3.585183 2.964004 4.206363  2.63517098 4.535195
## Nov 76       3.426111 2.515793 4.336429  2.03389980 4.818323
## Dec 76       3.364106 2.258105 4.470106  1.67262407 5.055588
## Jan 77       3.745532 2.457789 5.033275  1.77609961 5.714964
## Feb 77       3.676408 2.241461 5.111355  1.48184594 5.870970
## Mar 77       3.468941 1.891399 5.046484  1.05629895 5.881584
## Apr 77       3.170272 1.469128 4.871415  0.56859739 5.771946
## May 77       3.284700 1.462728 5.106673  0.49823350 6.071167
## Jun 77       3.547531 1.616724 5.478337  0.59461618 6.500445
## Jul 77       3.538060 1.500666 5.575454  0.42213472 6.653985
## Aug 77       3.578474 1.442667 5.714281  0.31203899 6.844909
## Sep 77       3.322120 1.089926 5.554314 -0.09172576 6.735967
```

```
plot(train_cast_s)
```




```
errs = test-train_cast_s$mean
rmses = sqrt(mean(errs^2))
maes = mean(abs(errs))
mapes = mean(abs((errs*100)/test))


cat("\nErrors: ",errs)
```

```
##
## Errors:  -0.1851832 -0.02611131 -0.06410582 0.154468 0.223592 0.1310586 -0.07027154 0.1152996 0.2524
```

```
cat("\nMAE: ",maes)
```

```
##
## MAE:  0.1736588
```

```
cat("\nMAPE: ",mapes)
```

```
##
## MAPE:  4.720811
```

```
cat("\nRMSE: ",rmses)
```

```
##
## RMSE:  0.1960646
```

///break

```
library(urca)

za <- ur.za(data_2000$UNRATENSA, model = c("intercept", "trend", "both"), lag=NULL)

summary(za)
```

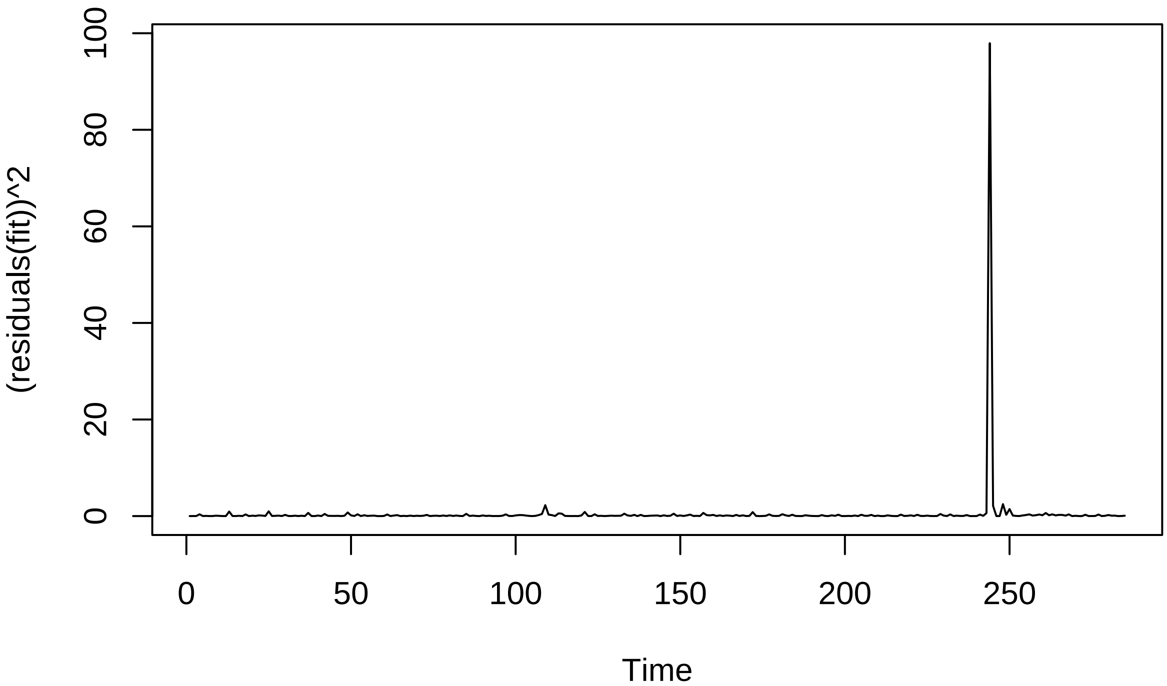
```
##
## ################################
```

```
## # Zivot-Andrews Unit Root Test #
## ################################
##
##
## Call:
## lm(formula = testmat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.2797 -0.2629 -0.0756  0.1454  9.8853
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.791909   0.188398   4.203 3.54e-05 ***
## y.l1         0.879893   0.026895  32.716  < 2e-16 ***
## trend       -0.003581   0.001104  -3.245  0.00132 **
## du           0.637057   0.196760   3.238  0.00135 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6947 on 280 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.8786, Adjusted R-squared:  0.8773
## F-statistic: 675.3 on 3 and 280 DF,  p-value: < 2.2e-16
##
##
## Teststatistic: -4.4658
## Critical values: 0.01= -5.34 0.05= -4.8 0.1= -4.58
##
## Potential break point at position: 100
```

```r
Box.test((residuals(fit))^2, lag = 20, type = "Ljung")
```

```
##
##  Box-Ljung test
##
## data:  (residuals(fit))^2
## X-squared = 0.34636, df = 20, p-value = 1
```

```r
plot((residuals(fit))^2)
```




```r
library(TSA)
```

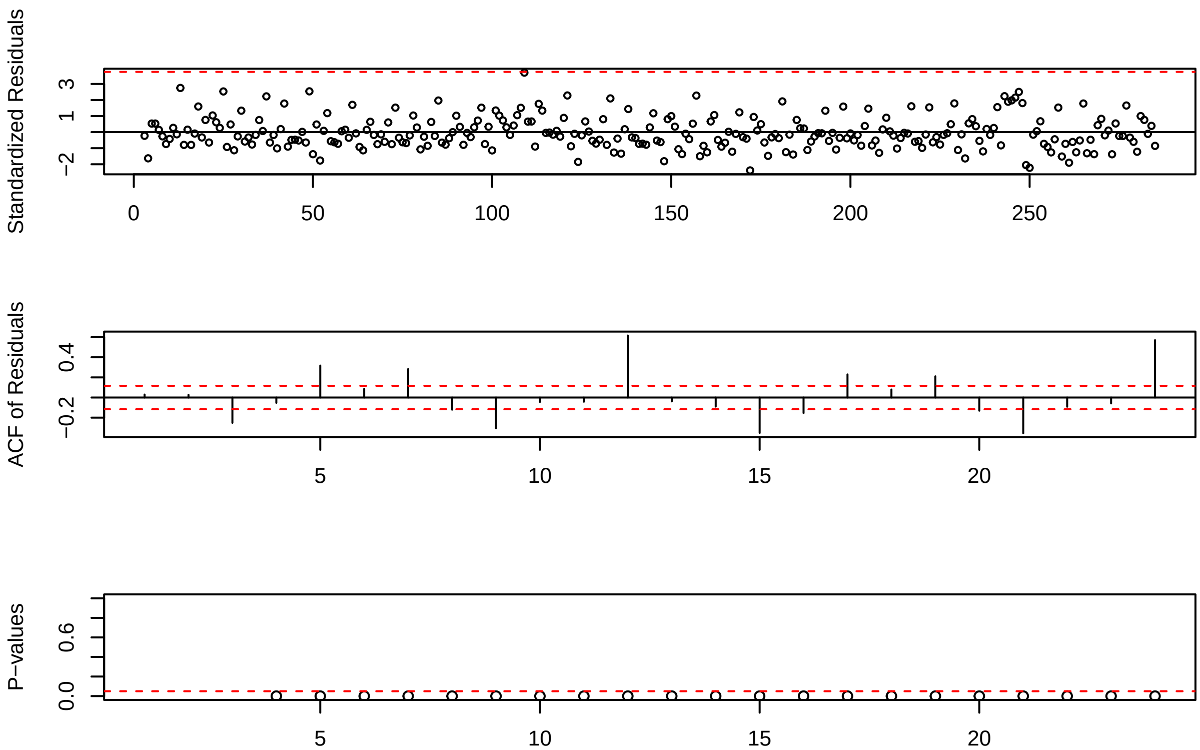
```
## Registered S3 methods overwritten by 'TSA':
##   method       from
##   fitted.Arima forecast
##   plot.Arima   forecast

##
## Attaching package: 'TSA'

## The following objects are masked from 'package:stats':
##
##     acf, arima

## The following object is masked from 'package:utils':
##
##     tar
```

```r
break.model=arimax(data_2000$UNRATENSA,order=c(1,1,2),
                   xtransf = data.frame(br=1*(seq(data_2000$UNRATENSA) == 244),
                                        br=1*(seq(data_2000$UNRATENSA) == 245),
                                        br=1*(seq(data_2000$UNRATENSA) == 246),
                                        br=1*(seq(data_2000$UNRATENSA) == 247)),
                    transfer=list(c(0,0),c(0,0),c(0,0),c(0,0)))

break.model
```

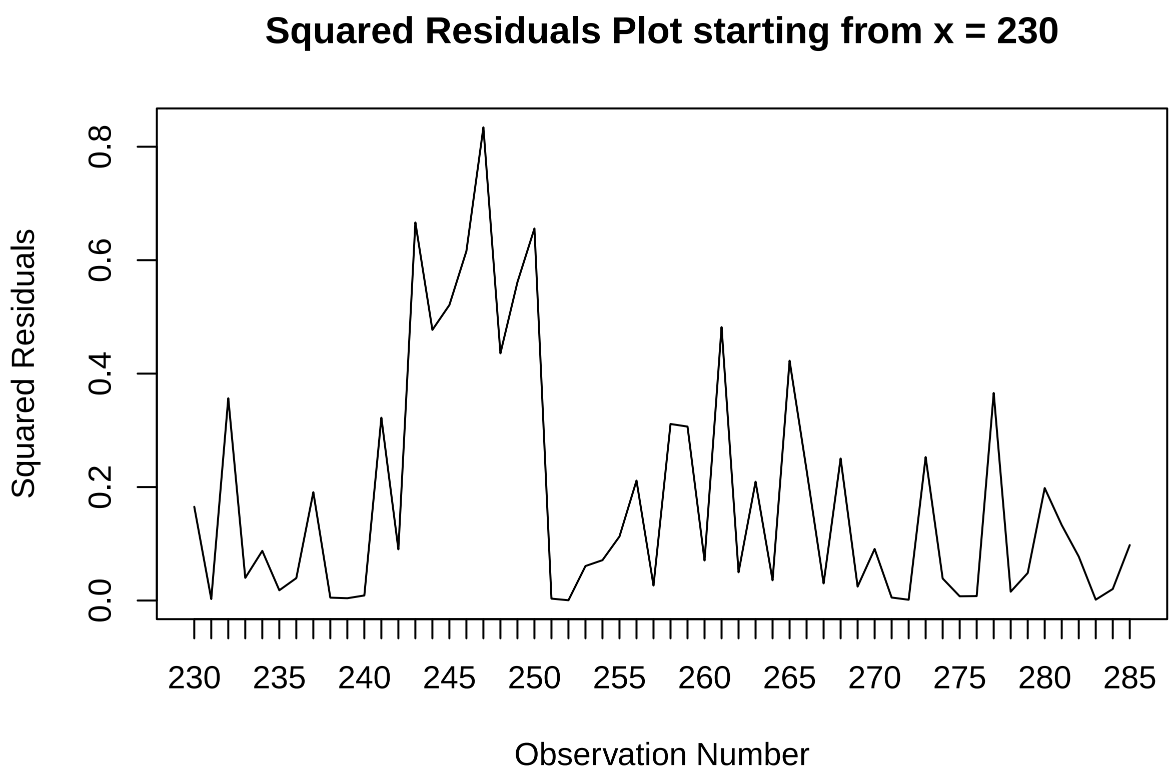
```
##
## Call:
## arimax(x = data_2000$UNRATENSA, order = c(1, 1, 2), xtransf = data.frame(br = 1 *
##     (seq(data_2000$UNRATENSA) == 244), br = 1 * (seq(data_2000$UNRATENSA) ==
##     245), br = 1 * (seq(data_2000$UNRATENSA) == 246), br = 1 * (seq(data_2000$UNRATENSA) ==
##     247)), transfer = list(c(0, 0), c(0, 0), c(0, 0), c(0, 0)))
##
## Coefficients:
```

```
##          ar1      ma1      ma2  br-MA0  br.1-MA0  br.2-MA0  br.3-MA0
##       0.4908  -0.3092  -0.1957  9.0594    6.8984    4.2985    2.6766
## s.e.  0.1750   0.1674   0.0515  0.3322    0.4570    0.4647    0.3410
##
## sigma^2 estimated as 0.1333:  log likelihood = -116.84,  aic = 247.69
```

```r
tsdiag(break.model)
```




```r
# Calculate residuals
residuals_squared <- residuals(break.model)^2

# Create a vector for x values
x_values <- seq_along(residuals_squared)

# Plot squared residuals starting from x = 230 with x-axis spacing by 1
plot(x_values[230:length(x_values)], residuals_squared[230:length(x_values)],
     type = 'l',  # 'l' for line plot
     xlab = 'Observation Number',
     ylab = 'Squared Residuals',
     main = 'Squared Residuals Plot starting from x = 230',
     xaxt = 'n')  # 'n' to suppress x-axis

# Add custom x-axis with spacing by 1
axis(1, at = seq(230, length(x_values), by = 1), labels = seq(230, length(x_values), by = 1))
```




```r
cat("\nar1:",0.4908/0.1750)
```

```
##
## ar1: 2.804571
```

```r
cat("\nma1:",0.3092/0.1674)
```

```
##
## ma1: 1.847073
```

```r
cat("\nma2:",0.1957/0.0515)
```

```
##
## ma2: 3.8
```

```r
cat("\nbr-MA0",9.054/0.3322)
```

```
##
## br-MA0 27.25467
```

```r
cat("\nbr.1-MA0",6.8984/0.4570)
```

```
##
## br.1-MA0 15.09497
```

```r
cat("\nbr.2-MA0",4.2985/0.4647)
```

```
##
## br.2-MA0 9.250054
```

```r
cat("\nbr.3-MA0",2.6777/0.3410)
```

```
##
## br.3-MA0 7.852493
```

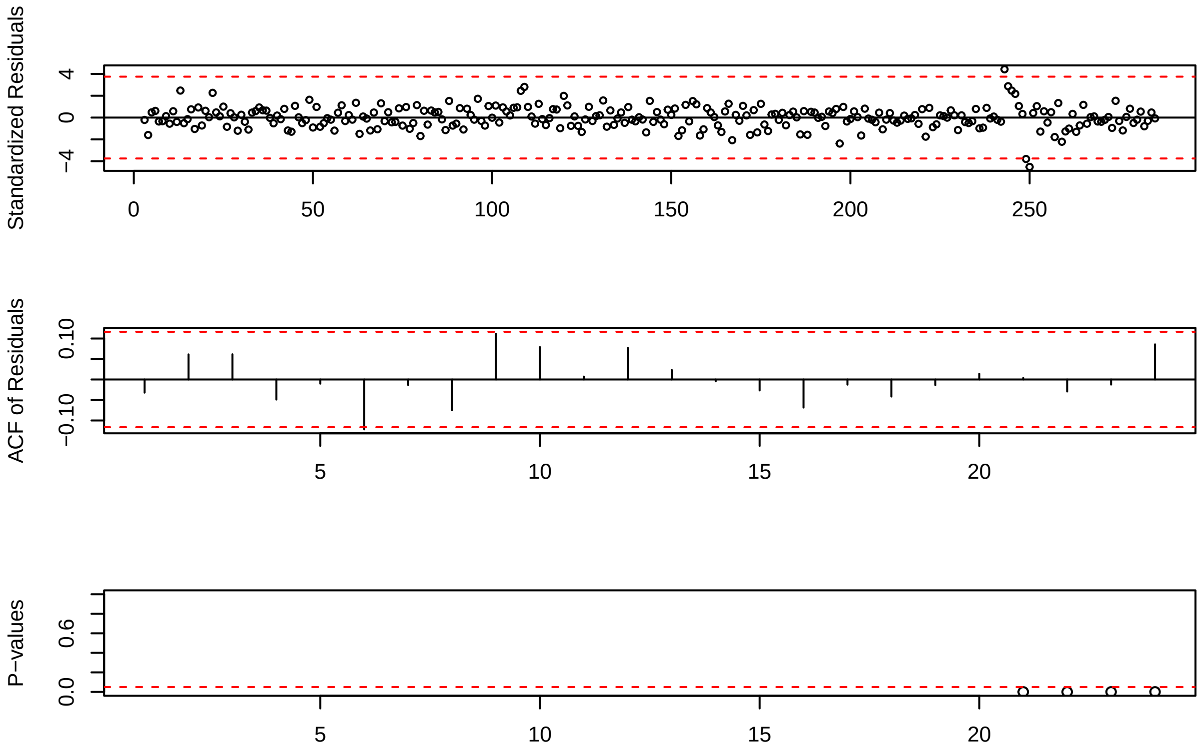
```r
library(TSA)

sbreak.model=arimax(data_2000$UNRATENSA,order=c(1,1,1), seasonal = list(order = c(9,0,9), frequency = 1
                    xtransf = data.frame(br=1*(seq(data_2000$UNRATENSA) == 244),
                                         br=1*(seq(data_2000$UNRATENSA) == 245),
                                         br=1*(seq(data_2000$UNRATENSA) == 246),
                                         br=1*(seq(data_2000$UNRATENSA) == 247)),
                     transfer=list(c(0,0),c(0,0),c(0,0),c(0,0)))
```

```
## Warning in arimax(data_2000$UNRATENSA, order = c(1, 1, 1), seasonal =
## list(order = c(9, : possible convergence problem: optim gave code=1
```

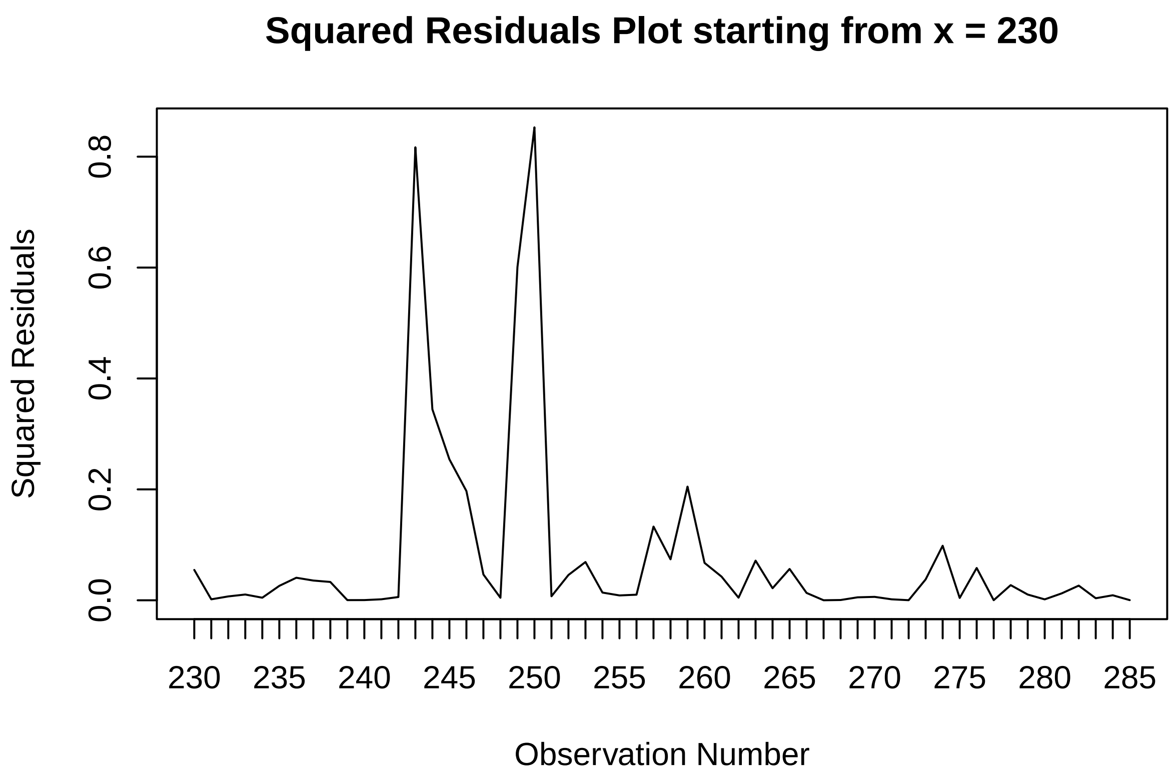
```r
sbreak.model
```

```
##
## Call:
## arimax(x = data_2000$UNRATENSA, order = c(1, 1, 1), seasonal = list(order = c(9,
##     0, 9), frequency = 12), xtransf = data.frame(br = 1 * (seq(data_2000$UNRATENSA) ==
##     244), br = 1 * (seq(data_2000$UNRATENSA) == 245), br = 1 * (seq(data_2000$UNRATENSA) ==
##     246), br = 1 * (seq(data_2000$UNRATENSA) == 247)), transfer = list(c(0,
##     0), c(0, 0), c(0, 0), c(0, 0)))
##
## Coefficients:
##          ar1      ma1    sar1    sar2     sar3    sar4    sar5     sar6    sar7
##       0.7627  -0.4603  0.2194  0.4311  -0.4743  0.6969  0.7292  -0.4304  0.4761
## s.e.  0.1037   0.1147  0.0479  0.0354   0.0147  0.0363  0.0046   0.0346  0.0110
##         sar8     sar9     sma1     sma2    sma3     sma4     sma5    sma6
##       0.2941  -0.9546  -0.2173  -0.4497  0.4473  -0.7146  -0.7464  0.4304
## s.e.  0.0343   0.0464   0.0684   0.0406  0.0429   0.0539   0.0654  0.0521
##          sma7     sma8    sma9  br-MA0  br.1-MA0  br.2-MA0  br.3-MA0
##       -0.4807  -0.2250  0.9562  9.5445    7.0245    3.8486    2.4218
## s.e.   0.0442   0.0548  0.0644  0.1778    0.2370    0.2405    0.1821
##
## sigma^2 estimated as 0.04168:  log likelihood = 33.28,  aic = -18.55
```

```r
tsdiag(sbreak.model)
```




```r
# Calculate residuals
sresiduals_squared <- residuals(sbreak.model)^2

# Create a vector for x values
x_valuess <- seq_along(sresiduals_squared)

# Plot squared residuals starting from x = 230 with x-axis spacing by 1
plot(x_valuess[230:length(x_valuess)], sresiduals_squared[230:length(x_valuess)],
     type = 'l',  # 'l' for line plot
     xlab = 'Observation Number',
     ylab = 'Squared Residuals',
     main = 'Squared Residuals Plot starting from x = 230',
     xaxt = 'n')  # 'n' to suppress x-axis

# Add custom x-axis with spacing by 1
axis(1, at = seq(230, length(x_valuess), by = 1), labels = seq(230, length(x_valuess), by = 1))
```




```r
cat("\nar1:",0.7627/0.1037)
```

```
##
## ar1: 7.35487
```

```r
cat("\nma1:",0.4603/0.1147)
```

```
##
## ma1: 4.013078
```

```r
cat("\nsar1:",0.2194/0.0479)
```

```
##
## sar1: 4.580376
```

```r
cat("\nsar2:",0.4311/0.0354)
```

```
##
## sar2: 12.17797
```

```r
cat("\nsar3:",0.4743/0.0147)
```

```
##
## sar3: 32.26531
```

```r
cat("\nsar4:",0.6969/0.0363)
```

```
##
## sar4: 19.19835
```

```r
cat("\nsar5:",0.7292/0.0046)
```

```
##
## sar5: 158.5217
```

```r
cat("\nsar6:",0.4304/0.0346)
```

```
##
## sar6: 12.43931
```

```r
cat("\nsar7:",0.4761/0.0110)
```

```
##
## sar7: 43.28182
```

```r
cat("\nsar8:",0.6969/0.0363)
```

```
##
## sar8: 19.19835
```

```r
cat("\nsar9:",0.9546/0.0464)
```

```
##
## sar9: 20.57328
```

```r
cat("\nsma1:",0.2173/0.0684)
```

```
##
## sma1: 3.176901
```

```r
cat("\nsma2:",0.4497/0.0406)
```

```
##
## sma2: 11.07635
```

```r
cat("\nsma3:",0.4473/0.0429)
```

```
##
## sma3: 10.42657
```

```r
cat("\nsma4:",0.7146/0.0539)
```

```
##
## sma4: 13.25788
```

```r
cat("\nsma5:",0.7464/0.00654)
```

```
##
## sma5: 114.1284
```

```r
cat("\nsma6:",0.4304/0.0521)
```

```
##
## sma6: 8.261036
```

```r
cat("\nsma7:",0.4807/0.0442)
```

```
##
## sma7: 10.87557
```

```r
cat("\nsma8:",0.2250/0.0548)
```

```
##
## sma8: 4.105839
```

```r
cat("\nsma9:",0.9562/0.0644)
```

```
##
```

```
## sma9: 14.84783
```

```r
cat("\nbr-MA0",9.5445/0.1778)
```

```
##
## br-MA0 53.6811
```

```r
cat("\nbr.1-MA0",7.0245/0.237)
```

```
##
## br.1-MA0 29.63924
```

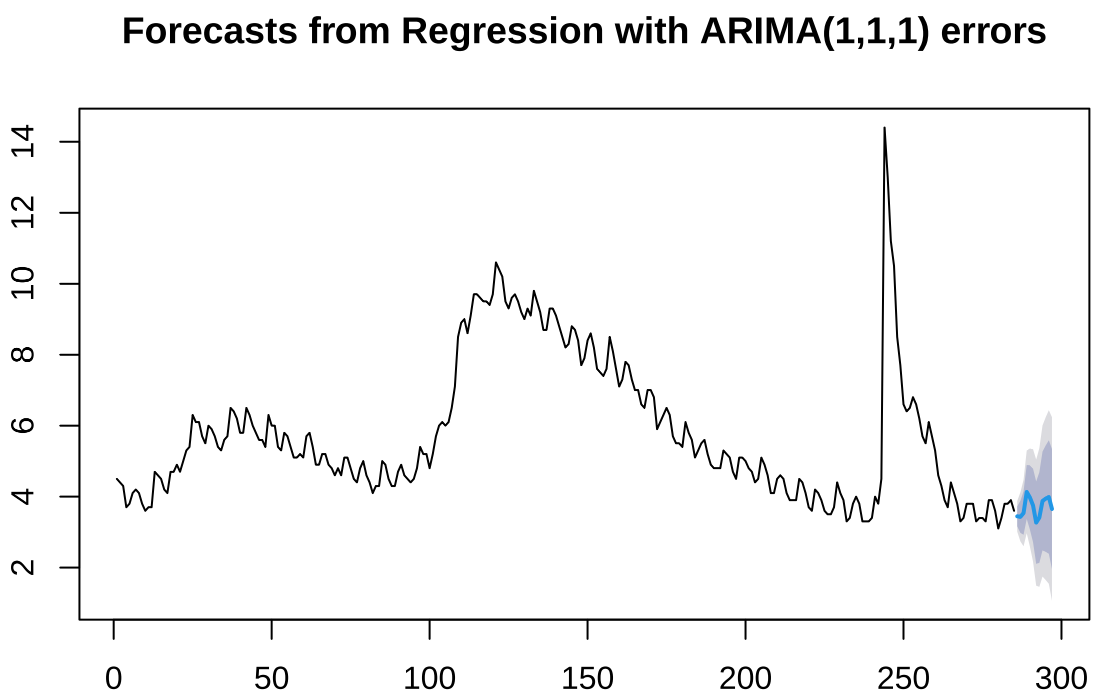
```r
cat("\nbr.2-MA0",3.8486/2.4218)
```

```
##
## br.2-MA0 1.589149
```

```r
cat("\nbr.3-MA0",2.4218/0.1821)
```

```
##
## br.3-MA0 13.29929
```

```r
### STBREAK AND SARIMA
transA=c(rep(0,243),9.0594,6.8984,4.2985,2.6766,rep(0,50))

library(forecast)

sbrarima=Arima(data_2000$UNRATENSA,order=c(1,1,1), seasonal = list(order = c(9,0,9), frequency = 12),xre

sbrfc <- forecast(sbrarima, h = 12, xreg=transA[286:297])

plot(sbrfc)
```

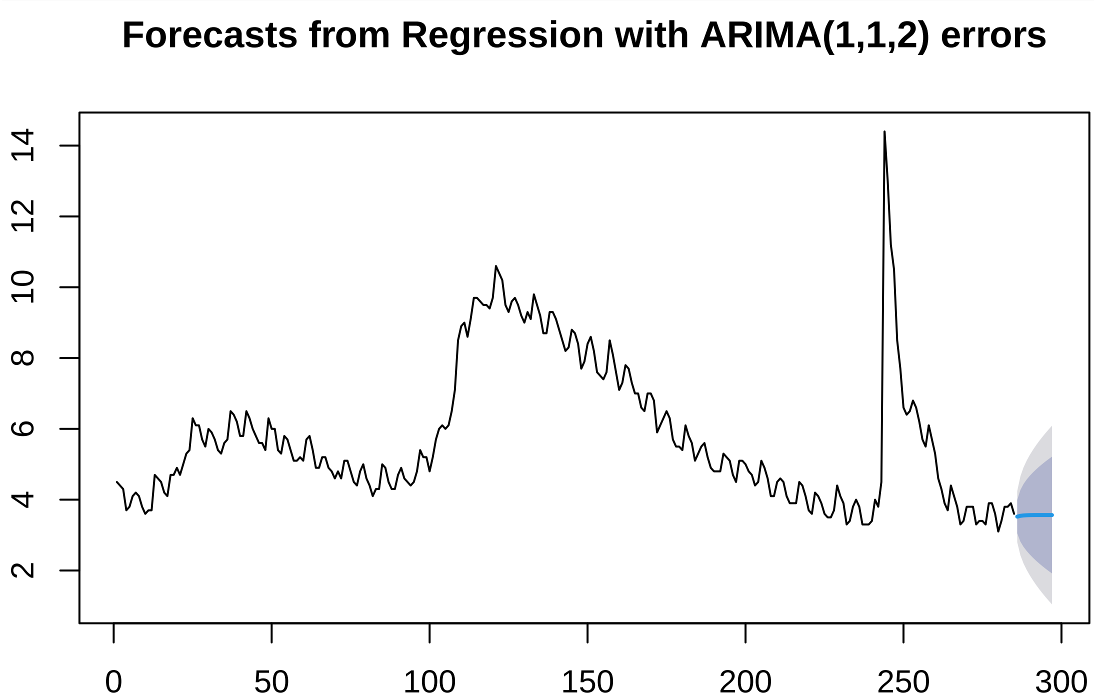



```r
sbrfc
```

```
##     Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 286       3.444510 3.162373 3.726647 3.013018 3.876001
```

```
## 287        3.430300 2.977625 3.882974 2.737993 4.122606
## 288        3.537111 2.924636 4.149587 2.600411 4.473812
## 289        4.132150 3.370706 4.893593 2.967622 5.296677
## 290        3.976395 3.074291 4.878499 2.596746 5.356044
## 291        3.750805 2.714936 4.786674 2.166580 5.335030
## 292        3.266686 2.105578 4.427794 1.490924 5.042448
## 293        3.409814 2.132851 4.686776 1.456868 5.362759
## 294        3.875720 2.484382 5.267058 1.747853 6.003587
## 295        3.941698 2.444057 5.439339 1.651254 6.232142
## 296        3.988446 2.389724 5.587169 1.543411 6.433482
## 297        3.651613 1.961140 5.342085 1.066258 6.236967
```

```r
trans=c(rep(0,243),9.0594,6.8984,4.2985,2.6766,rep(0,50))

library(forecast)

brarima=Arima(data_2000$UNRATENSA,order=c(1,1,2),xreg=trans[1:285])

brfc <- forecast(brarima, h = 12, xreg=trans[286:297])

plot(brfc)
```




```r
brfc
```

```
##      Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 286        3.521557 3.050360 3.992754 2.800924 4.242191
## 287        3.544250 2.814875 4.273624 2.428768 4.659731
## 288        3.555387 2.667397 4.443376 2.197324 4.913450
## 289        3.560853 2.550559 4.571146 2.015742 5.105963
## 290        3.563536 2.449380 4.677692 1.859581 5.267490
## 291        3.564852 2.358018 4.771687 1.719158 5.410546
## 292        3.565498 2.273655 4.857342 1.589795 5.541202
## 293        3.565816 2.194702 4.936929 1.468878 5.662753
```

```
## 294       3.565971 2.120150 5.011793 1.354778 5.777165
## 295       3.566048 2.049297 5.082798 1.246378 5.885718
## 296       3.566085 1.981626 5.150544 1.142864 5.989306
## 297       3.566104 1.916737 5.215470 1.043615 6.088592
```