HW5

Nikhil Gopal

11/19/2021

```r
rm(list = ls())
#setwd("/Users/nikhil/My Drive/STAT 3105/HW5")
library(caret)
library(tidyverse)
dat <- read.csv("NYbirths.csv")
births <- read.csv("NYbirths.csv", header = F)$V1
```

Question 1

1a

1a:

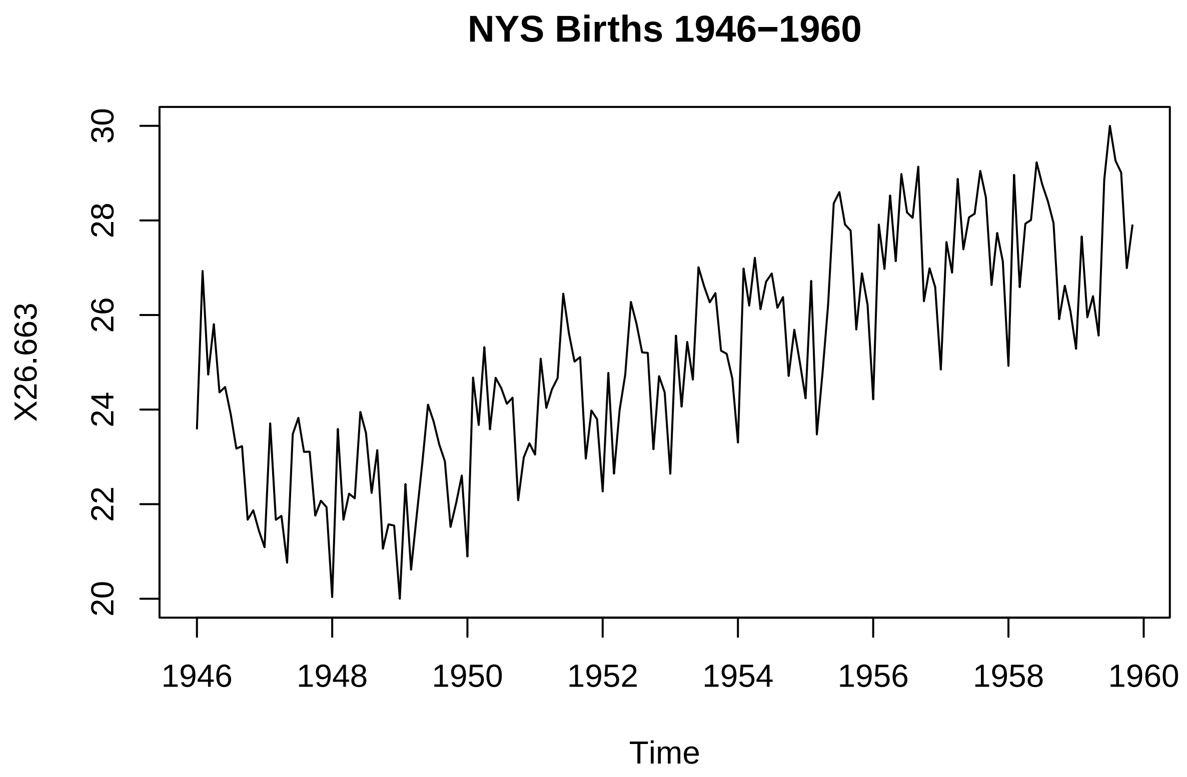
Create a time series objective by using ts(*, frequency=12, start=c(1946,1))

```r
ts <- ts(dat, frequency = 12, start = c(1946, 1))
```

1b

1b: Make a plot of the time series data. From this plot, do you observe some trend of the data? More specifically, do you see a seasonal trend and what is the overall trend?

```r
plot(ts, main = "NYS Births 1946-1960")
```

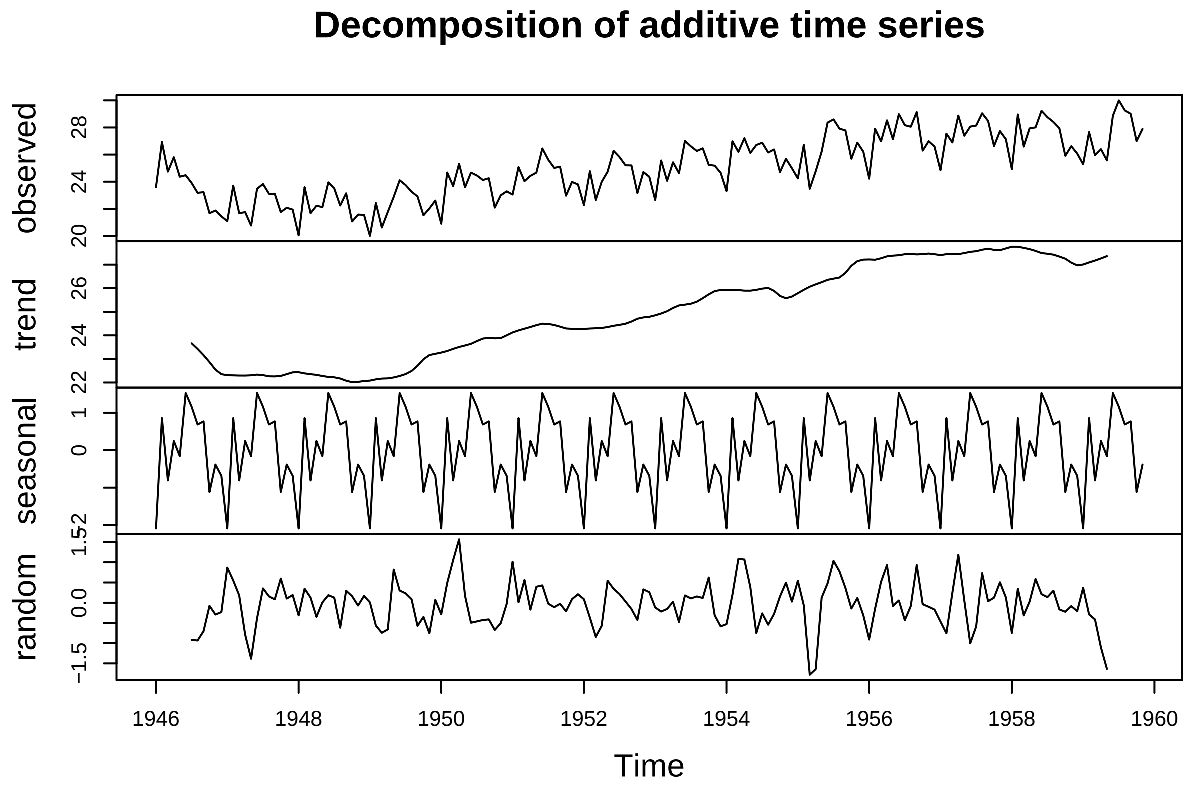


We see an overall trend of the number of births increasing over time. We also notice a sharp drop in births during the years of WW2, presumably because many more men were away serving in the military than usual, but this picks up again in 1947. Seasonally we see the birth rate peak in the middle of the year, then drop then rise again.

1c

1c: A seasonal time series consists of a trend component, a seasonal component and an irregular component. Decomposing the time series means separating the time series into these three components: that is, estimating these three components. Use decompose() in R and plot the decomposed time series. What can you tell from the results?

```
decomp_ts <- decompose(ts)
plot(decomp_ts)
```



The trend and the seasonal graphs confirm my thesis from the above question. Overall, births increase overtime except during the war years as I mentioned above. Additionally, birth numbers are seasonal, increasing and decreasing in the same proportion at the same time every year, as demonstrated by the trends graph.

*Trend:

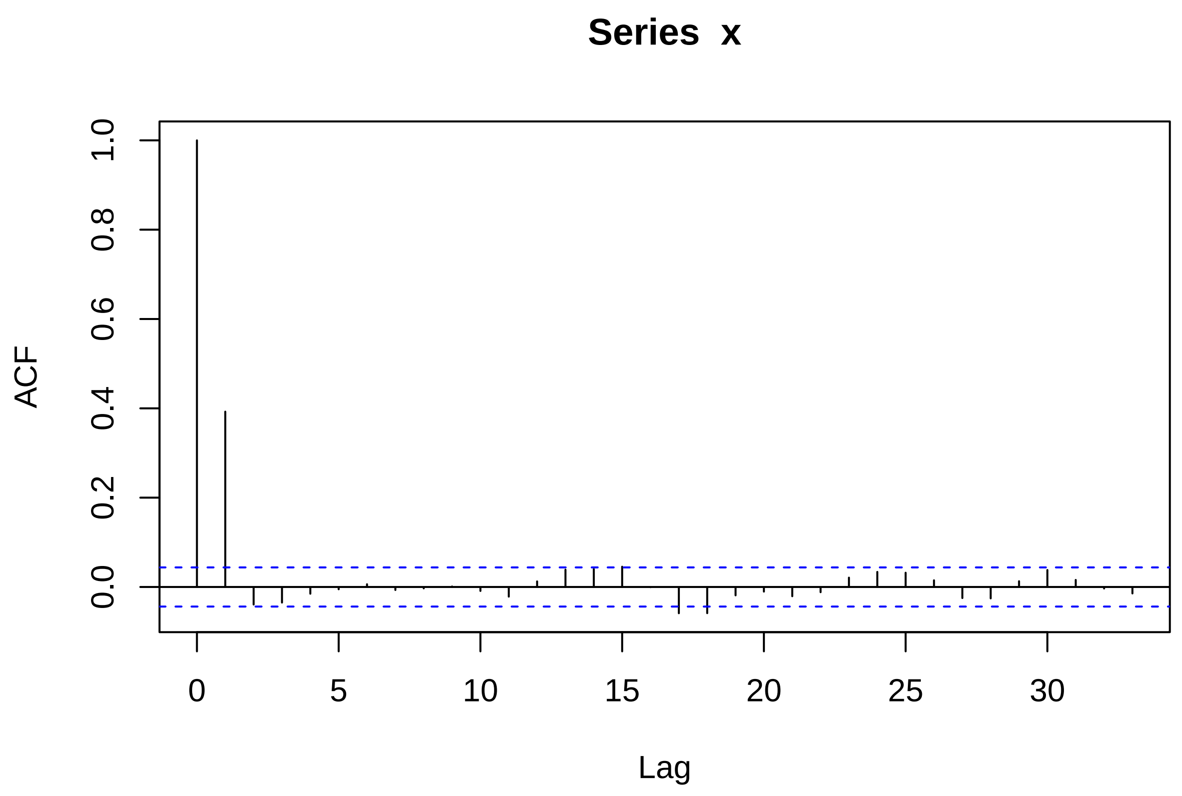
+The trend for the first and last 6 elements are missing. +There are symmetric coefficients on both the right and left +Frequency sums to 12 *Seasonality + Use the data after subtracting the trend. +For the seasonality in each month, take the average of the detrended data for all years in this month.* Irregular + That which is left after subtracting the trend and the seasonality

1d

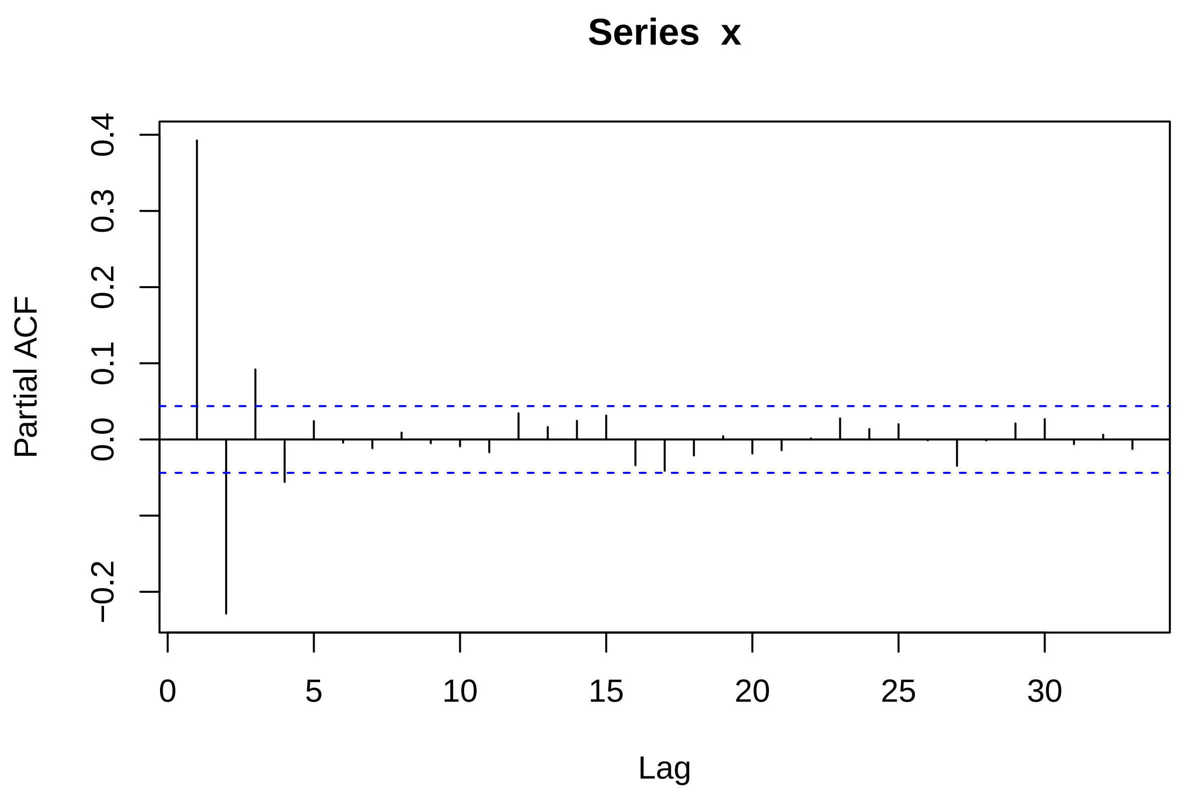
1d: Our time series analysis mainly focused on the irregular component. ARIMA models are defined for stationary time series. The time series of the irregular component appears to be stationary in mean and variance, and so an ARMA(p, q) model is appropriate. The next step is to select the appropriate ARMA model, which means finding the values of most appropriate values of p and q for an ARMA(p, q) model.

- To do this, you usually need to examine the correlogram and partial correlogram of the stationary time series. Plot the correlogram and partial correlogram of the irregular component using acf() and pacf() functions in R, respectively. Based on your plots, which lags exceed the significance bounds?

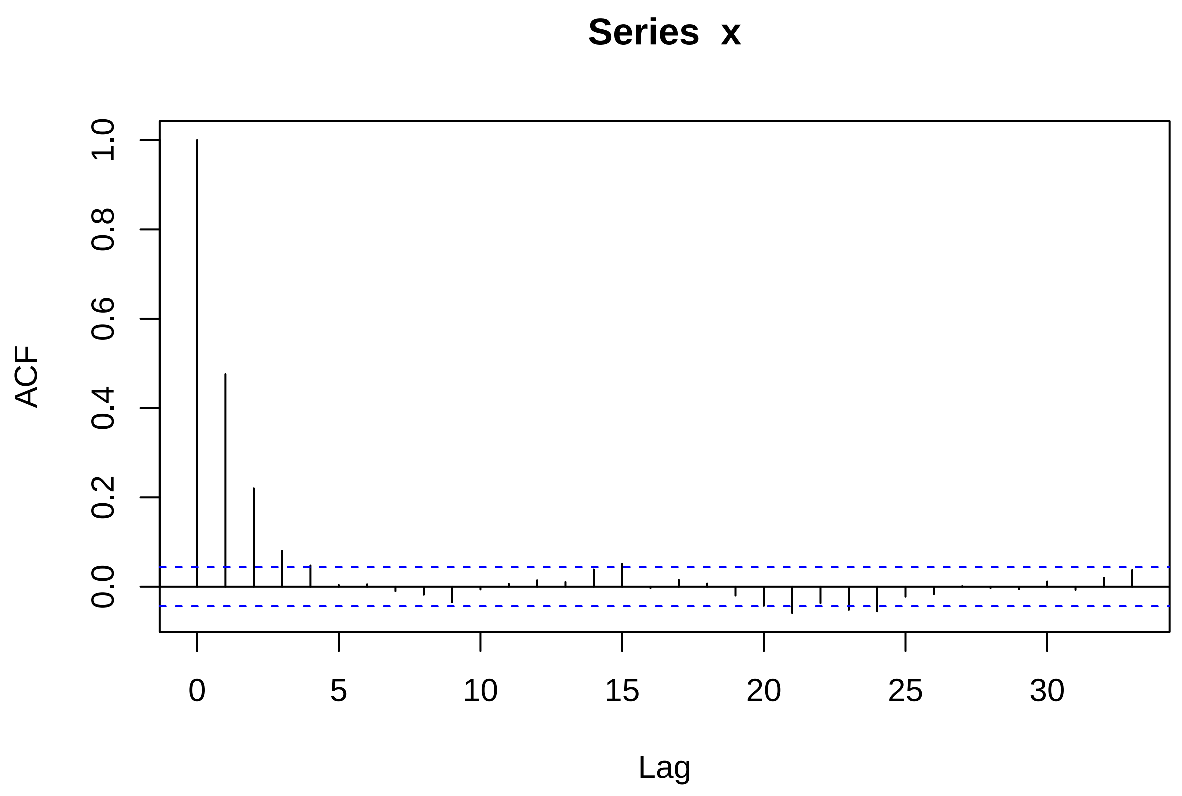
```
## MA(1)
x <- arima.sim(list(ar = c(), ma = c(2)), n = 2000)
acf(x)
```



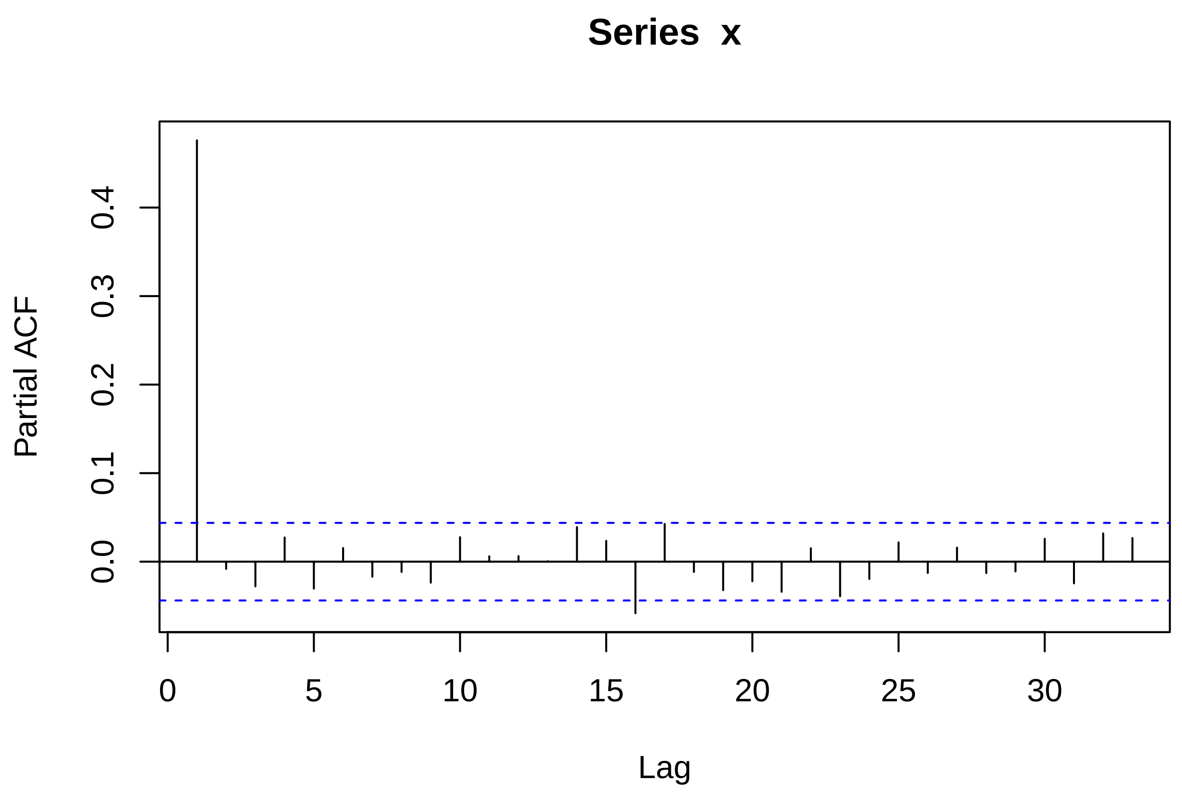
```
pacf(x)
```



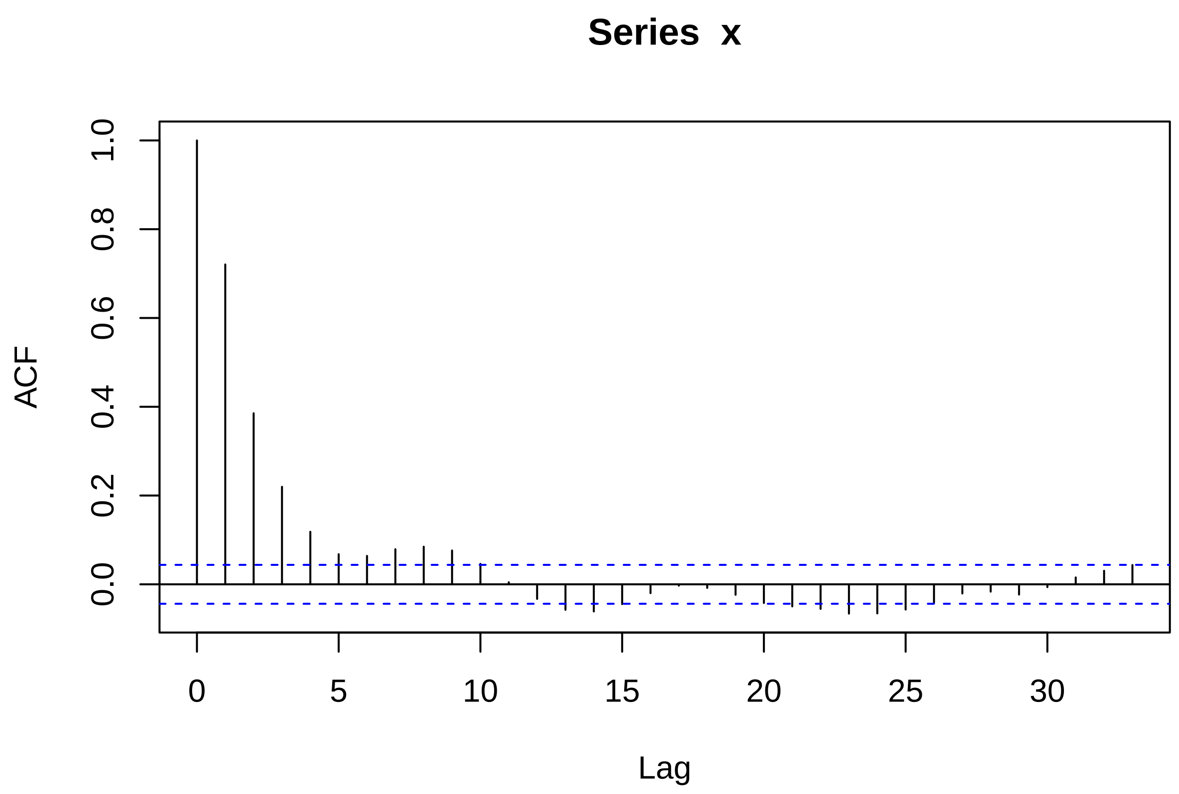
```r
## AR(1)
x <- arima.sim(list(ar = c(0.5), ma = c()), n = 2000)
acf(x)
```



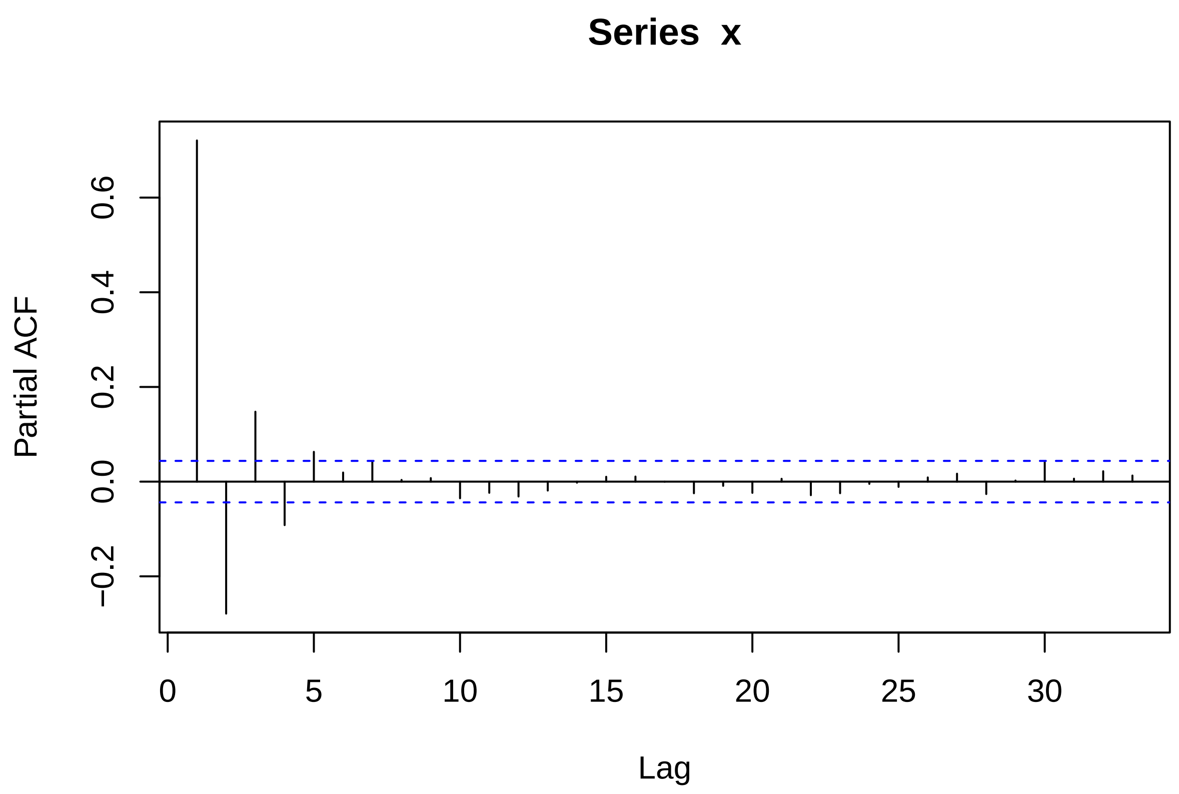
```
pacf(x)
```



```
## ARMA(1, 1)
x <- arima.sim(list(ar = c(0.5), ma = c(2)), n = 2000)
acf(x)
```

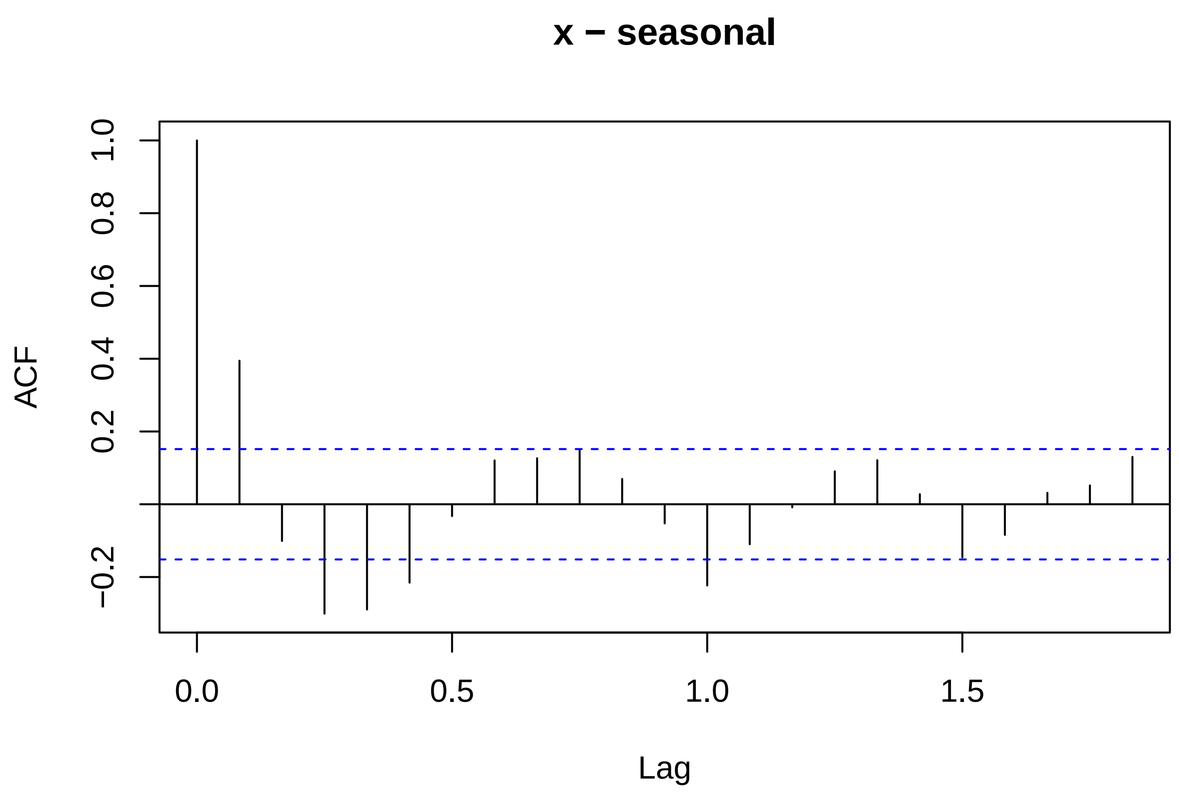


```
pacf(x)
```

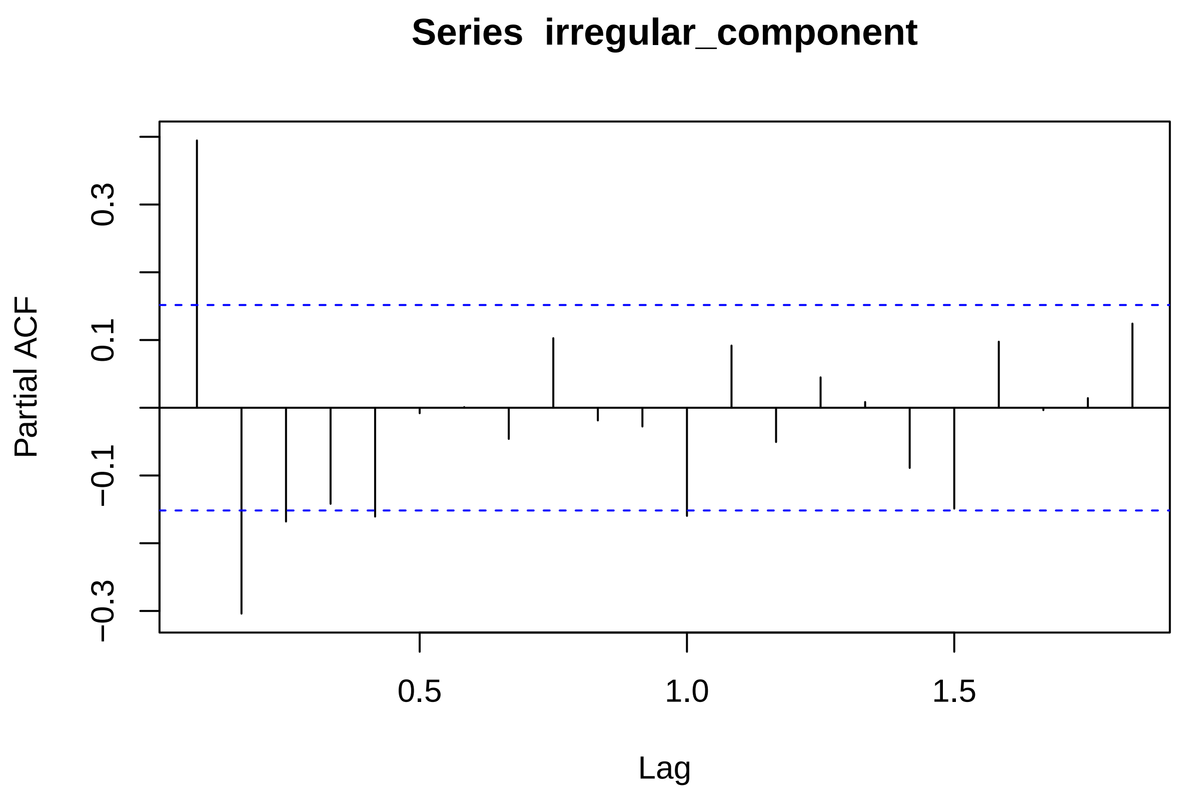


Examine ACF and PACF

```
irregular_component <- decomp_ts$random
acf(irregular_component, na.action = na.pass)
```



```
pacf(irregular_component, na.action = na.pass)
```



Fit the model:

#https://rpubs.com/JSHAH/481706

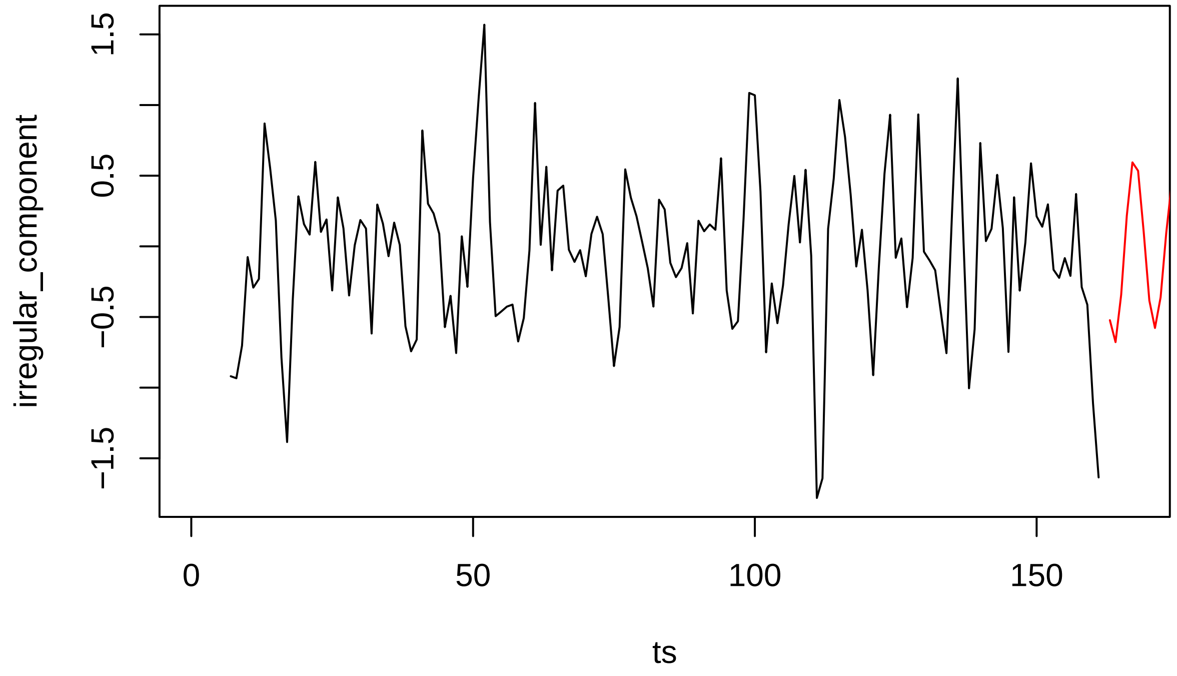
- Let p equals the largest lag that is still significant in the correlogram , and q for the largest significant lag in the partial correlogram. Then (p, q) is the best candidate for an ARMA model. Please fit the ARMA(p, q) model.

```
p <- 3
q <- 5
births_arma <- arima(irregular_component, order = c(p, 0, q))
births_arma
```

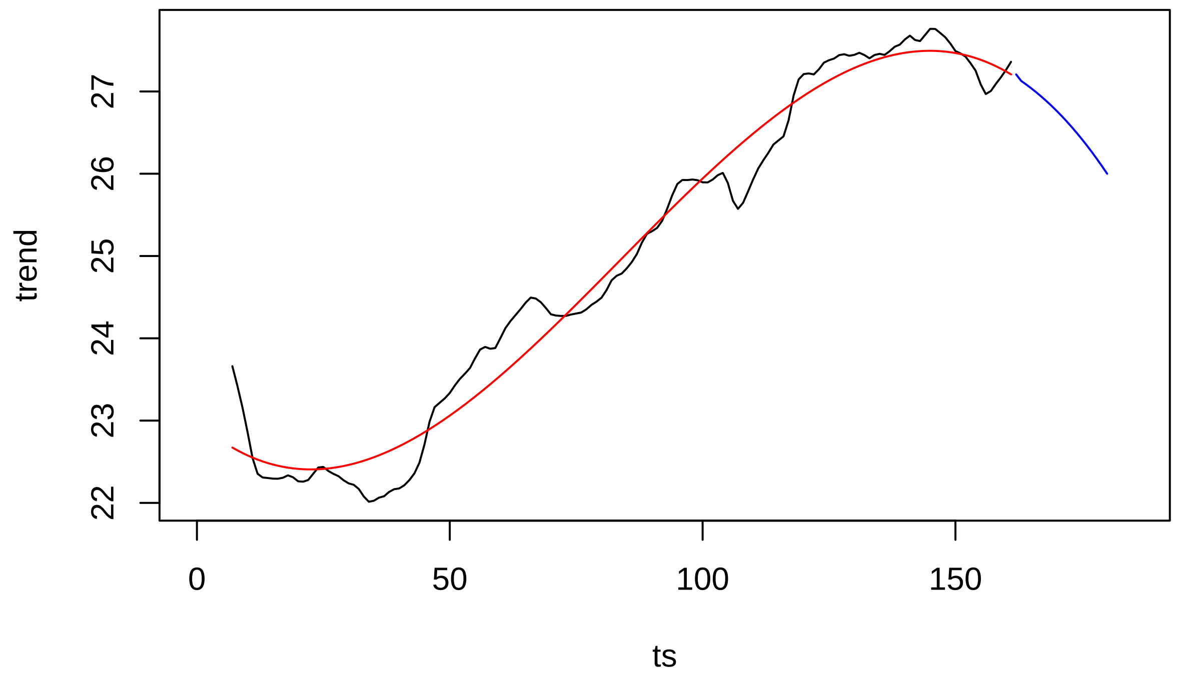
```
##
## Call:
## arima(x = irregular_component, order = c(p, 0, q))
##
## Coefficients:
##          ar1      ar2     ar3      ma1     ma2     ma3      ma4      ma5
##       1.6418  -1.4398  0.3731  -1.3666  0.7384  0.0458  -0.0806  -0.3370
## s.e.  0.1601   0.2111  0.1575   0.1606  0.2315  0.1567   0.1591   0.1271
##       intercept
##          -0.004
## s.e.      0.003
##
## sigma^2 estimated as 0.1742:  log likelihood = -89.14,  aic = 198.28
```

Forecast:

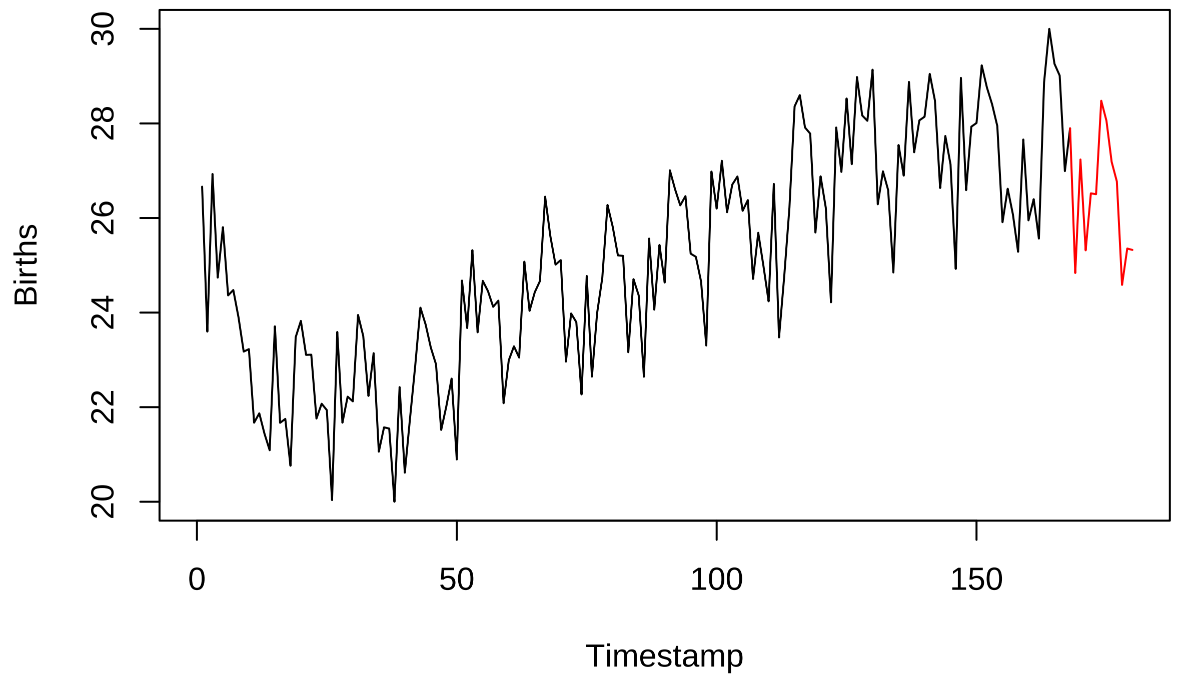
```
## Forecast the irregular component
forecasted_irregular_component <- predict(births_arma, n.ahead = 18)
ts <- 1:length(irregular_component)
plot(ts, irregular_component, type = "l")
lines(length(births) + -6:12,
      c(irregular_component[length(births) - 6],
forecasted_irregular_component$pred[1:18]),
col = "red")
```



```
## Fit and Forecast the trend
ts <- 1:length(births)
trend <- decomp_ts$trend[ts]
trend_df <- data.frame(trend = trend, ts = ts)
trend_reg <- lm(trend ~ polym(ts, degree = 3, raw = T),
data = trend_df)
plot(ts, trend, type="l", xlim = c(0, 185))
lines(ts[!is.na(trend)], trend_reg$fitted.values, col="red")
trend_forecast_df <- data.frame(ts = length(births) + -5:12)
forecasted_trend <- predict(trend_reg, trend_forecast_df)
lines(c(length(births) - 6, trend_forecast_df$ts),
c(trend_reg$fitted.values[length(trend_reg$fitted.values)],
forecasted_trend), col = "blue")
```



```r
## Add everything together
forecasted_births <- forecasted_irregular_component$pred[7:18] +
forecasted_trend[7:18] + decomp_ts$seasonal[1:12]
plot(1:(length(births)), births, type = "l", xlim = c(0, 180),
xlab = "Timestamp", ylab = "Births")
lines(length(births) + 0:12, c(births[length(births)],
forecasted_births), col = "red")
```



- Using your fitted model, predict the number of births per month in 1960.

```
predict(births_arma, n.ahead = 12)
```

```
## $pred
##              Jan         Feb         Mar         Apr         May         Jun
## 1959
## 1960 -0.67791291 -0.34297896  0.21627675  0.59424913  0.53456460  0.10105050
##              Jul         Aug         Sep         Oct         Nov         Dec
## 1959                                                                -0.52254175
## 1960 -0.38372224 -0.57771873 -0.36000752  0.09585869  0.45845100
##
## $se
##            Jan       Feb       Mar       Apr       May       Jun       Jul
## 1959
## 1960 0.5179648 0.5200423 0.5270620 0.5305755 0.5305494 0.5345084 0.5399037
##            Aug       Sep       Oct       Nov       Dec
## 1959                                         0.5170289
## 1960 0.5408967 0.5417249 0.5462052 0.5492819
```

#Beat the kaggle baseline

Code not run to save run time when knitting:

```r
flights <- read.csv("pnwflights14.csv")

# Remove NAs
flights <- na.omit(flights)

# Remove nonsense entries
flights <- flights[-which(flights$dep_time < 0),]
flights <- flights[-which(flights$air_time < 0),]
flights <- flights[-which(flights$distance < 0),]




#convert the timing into a date time object
flights$date <- paste(flights$year, "-", flights$month, "-", flights$day, " ", flights$hour, ":", fligh

flights$dep_date <- strptime(flights$date, format="%Y-%m-%d %H:%M")

flights2 <- flights


flights2$arr_time2 <- substr(as.POSIXct(sprintf("%04.0f", flights$arr_time), format='%H%M'), 12, 16)

flights2$arr_date <- paste(flights$year, "-", flights$month, "-", flights$day, " ", flights2$arr_time2,

flights2$arr_date <- strptime(flights2$arr_date, format="%Y-%m-%d %H:%M")

flights$arr_date <- flights2$arr_date

#flights2 <- select(flights, -c(month, day, year, date))

flights2 <- flights

flights2$dep_date <- as.Date(flights2$dep_date)
flights2$arr_date <- as.Date(flights2$arr_date)
```

Test/Train:

```r
#Test/Train Split
trainIndex <- createDataPartition(flights2$arr_delay, p = .8,
                                  list = FALSE,
                                  times = 1)
train <- flights2[trainIndex,]
test <- flights2[-trainIndex,]
```

Simple Regression Model

```r
# Define training control
set.seed(123)
train.control <- trainControl(method = "cv", number = 10)

# Train the model
model <- train(arr_delay ~ air_time+carrier+origin+distance+dest+dep_date,data = flights2, method = "lm
#RMSE = 29.88867
print(model)
```

Random Forest:

```
library(randomForest)
require(caTools)


rf <- randomForest(
  arr_delay ~ air_time+carrier+origin+distance+dest+dep_delay+month,
  data = train
)
```

Add Kaggle Data

```
test_kaggle <- read.csv("G:/My Drive/STAT 3105/HW4/test_student.csv")


#add back dep_date col
test_kaggle$date <- paste(test_kaggle$year, "-", test_kaggle$month, "-", test_kaggle$day, " ", test_kagg
test_kaggle$dep_date <- strptime(test_kaggle$date, format="%Y-%m-%d %H:%M")
test_kaggle$dep_date <- as.Date(test_kaggle$dep_date)
```

Generate final answer

```
submission <- predict(rf, test_kaggle)




submission_df <- data.frame(test_kaggle$Id)
submission_df$arr_delay <- submission
names(submission_df) <- c("Id", "arr_delay")

library(readr)
write.csv(submission_df, "G:/My Drive/STAT 3105/HW5/submission.csv", row.names = FALSE)
```