STAT 443 - Project

Monica Iyer

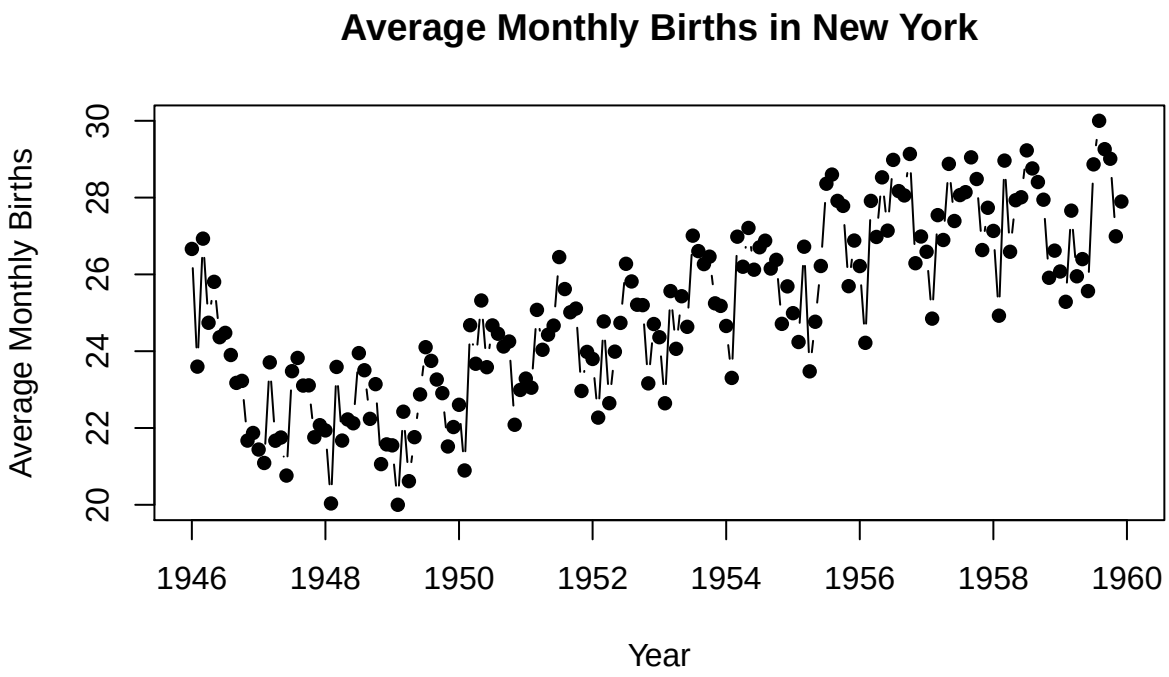
24/04/2020

Summary

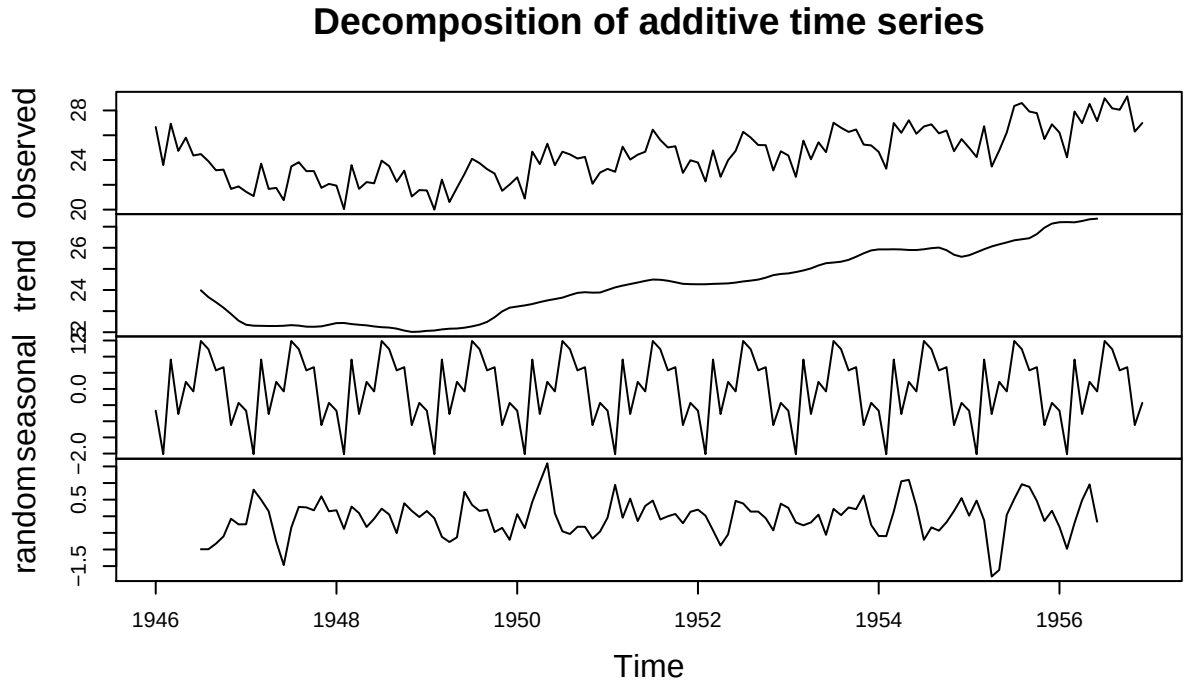
Through this report, I assess the different models that can be used to forecast the growth of the City of New York based upon monthly average birth data provided in *NYBirth.csv*. I have used Polynomial Regression, Regularized Regression, Holt Winters Modeling Techniques and Box-Jenkins Models. In order to compare the predictive power and significance of each of the models, I have compared using the Residuals Plot Daignostics and Average Prediction Error *APE* . The final model can be used to forecast the average number of births in New York City in 1960-1961.

Data Pre-processing

```
nybirth <- read.csv("NYBirth.csv") #eliminate the first column from NYBirth
                                   #and save it as a csv
nybirth_ts <- ts(nybirth, frequency=12, start=1946, end=1959+11/12)

#divide into training and test sets
nybirth_train <- window(nybirth_ts, start= 1946, end=1956+11/12)
nybirth_test<- window(nybirth_ts, start=1957, end=1959+11/12)
log_train <- log(nybirth_train)
log_test <- log(nybirth_test)
#plots
color <- c(rep("black", 144), rep("blue",24))
plot(nybirth_ts, col=color, type='b', pch=16,xlim=c(1946, 1960),xlab="Year",
     ylab="Average Monthly Births",
     main="Average Monthly Births in New York")
```

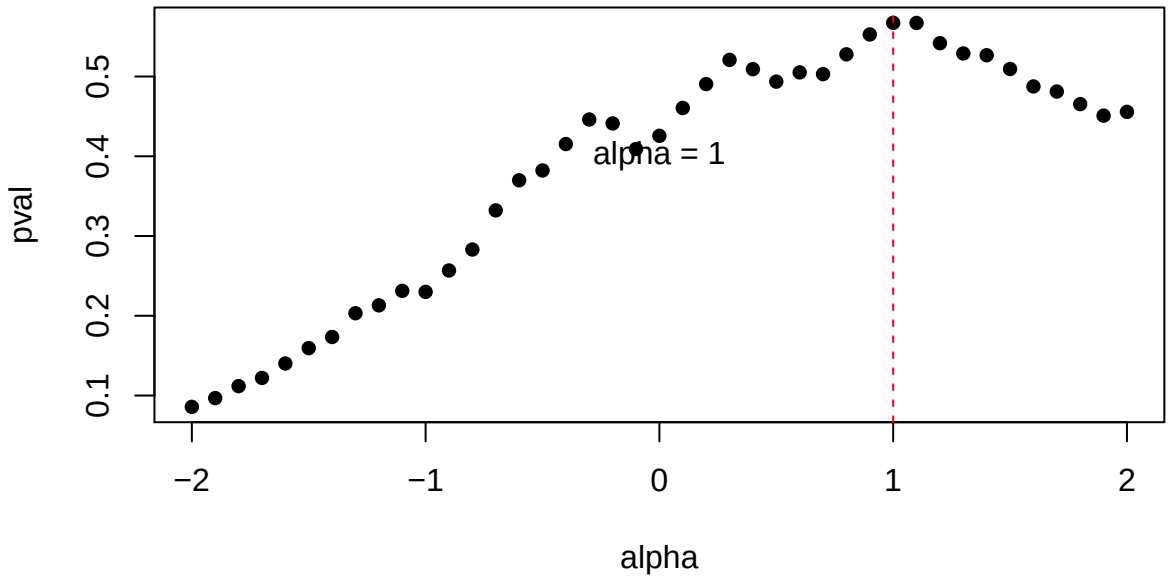



```
nybirthtrain_decompose <- decompose(nybirth_train, type="additive")
nybirthtest_decompose <- decompose(nybirth_test, type="additive")
plot(nybirthtrain_decompose)
```



Insights:

- There is an obvious growing trend
- The data is seasonal as well (frequency =12)
- Seems like the (trend + seasonal + random) component is equal to the observed data, so $X_t = m_t + S_t + R_t$
- There doesnt seem to be much of a difference between additive and multiplicative in terms of decomposition - so I stick with additive decomposition
- The non-stationarity due to increasing trend is okay for regression, but since there is non-constant variance, it has to be handled.



When observing the Fligner test above, we see that the p-value is the highest for alpha =1 which is $(nybirth)^1$ hence no transformation is required in this case since the data has constant variance.

Model Selection and Evaluation

Traditional polynomial regression models

- The regression model will have to account for seasonality in the data, hence I have used *nyb_seasonal* and for the trend which is represented by *t1* .

```
nyb_seasonal <- as.vector(cycle(nybirth_train))
nyb_seasonal_test <- as.vector(cycle(nybirth_test))

t1<- as.vector(time(nybirth_train))
t2 <- t1^2
t3 <- t1^3
t4 <- t1^4

t_test<- as.vector(time(nybirth_test))
d2 <- data.frame(t1=t_test,t2=t_test^2, nyb_seasonal= nyb_seasonal_test)
d3 <- data.frame(t1=t_test, t2=t_test^2, t3=t_test^3, nyb_seasonal= nyb_seasonal_test)
d4 <- data.frame(t1=t_test,t2=t_test^2,t3=t_test^3,t4=t_test^4, nyb_seasonal= nyb_seasonal_test)


poly1 <- lm(nybirth_train~t1+t2+nyb_seasonal)
poly2 <- lm(nybirth_train~t1+t2+t3+nyb_seasonal)
poly3 <- lm(nybirth_train~t1+t2+t3+t4+nyb_seasonal)

pred1 <- predict(poly1, d2)
pred2 <- predict(poly2, d3)
pred3 <- predict(poly3, d4)
APE_C=c()
APE_C[1] = sum((pred1- nybirth_test)^2)/length(nybirth_test)
APE_C[2] = sum((pred2- nybirth_test)^2)/length(nybirth_test)
APE_C[3] = sum((pred3- nybirth_test)^2)/length(nybirth_test)
APE_C
```

```
## [1] 10.966 10.966 10.966
```

The best polynomial regression model of the three seems to be the one with $X_t = t + t^2 + season + \epsilon_t$ with the regular training and testing data. However I have tested different variations of the model based on the best APE value obtained - with the seasonal component *nyb_seasonal* - without it - with the log transformed data - different polynomial degrees, etc.

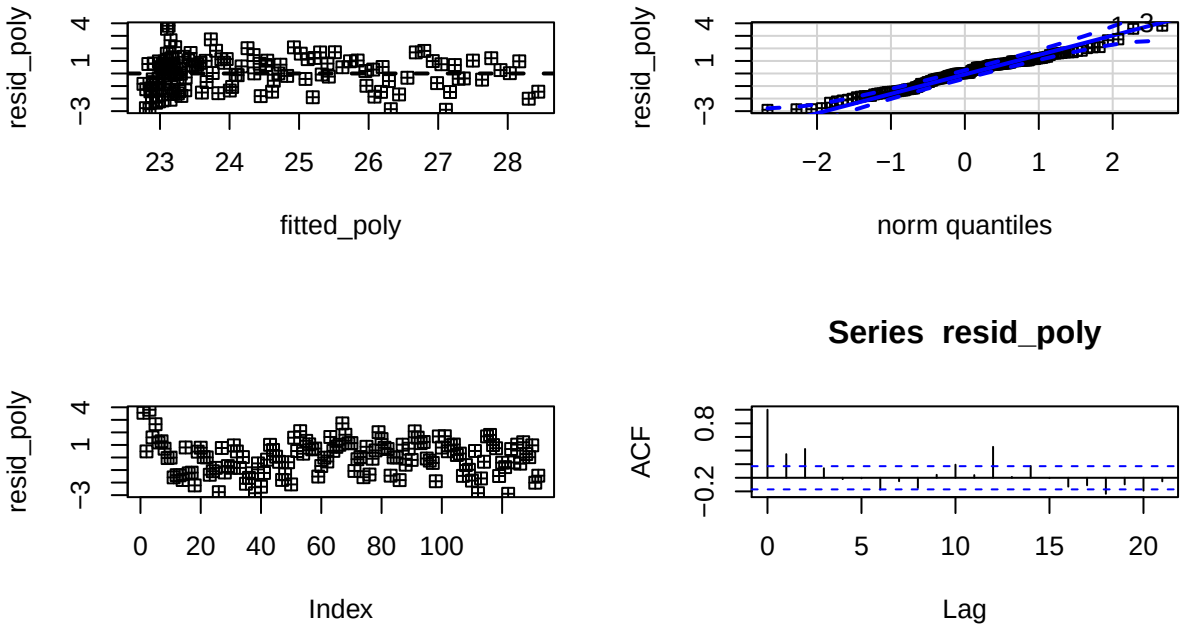
I perform residual diagnostics on $X_t = t + t^2 + season + \epsilon_t$ to see if we can make statistically significant assumptions using the model

```
resid_poly = poly1$residuals
fitted_poly = poly1$fitted.values
par(mfrow=c(2,2))
# residuals vs. fitted values
plot(resid_poly~fitted_poly , pch=12)
abline(h=0,lwd=2,lty=2)
#QQplot
#qqnorm(resid_model)
#qqline(resid_model)
```

```
library(car)
qqPlot(resid_poly , pch=12)
```

```
## [1] 3 1
```

```
#Residuals plot
plot(resid_poly , pch=12)
#ACF plot
acf(resid_poly , pch=12)
```




```
#test constant variance
resid_seg <- rep(1:11, each=12)
fligner.test(resid_poly, resid_seg)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  resid_poly and resid_seg
## Fligner-Killeen:med chi-squared = 7.1757, df = 10, p-value = 0.7088
```

```
#test Normality
shapiro.test(resid_poly)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid_poly
## W = 0.98563, p-value = 0.1808
```

```r
#randomness test
library(lawstat)
runs.test(resid_poly)
```

```
##
##  Runs Test - Two sided
##
## data:  resid_poly
## Standardized Runs Statistic = -4.0192, p-value = 5.841e-05
```

```r
library(randtests)
difference.sign.test(resid_poly)
```

```
##
##  Difference Sign Test
##
## data:  resid_poly
## statistic = -0.75094, n = 132, p-value = 0.4527
## alternative hypothesis: nonrandomness
```

Insights from the residuals plot:

- The residuals vs the fitted values indicates some pattern and so does the residuals time series plot. This suggests that the mean is not constant 0
- The p-value in the runs test (5.841e-05) is very small, which confirms evidence against randomness of residuals. In the ACF plot, more than 5% of the spikes cross the confidence interval, indicating that the residuals are in fact dependent.
- With the Fligner test we see that the p-value (0.7088) is fairly large, hence there is no violation of the constant variance assumption.
- From the QQ plot, we see that the upper right and lower left tails of the distribution deviates sligthly from the normal distribution, however the Shapiro test's p-value (0.1808) indicates that the residuals are also normal.

From the Residuals diagnostics, we see that the constant mean 0 and independence rules have been violated. However, the residuals have constant variance and follow the rules of a normal distribution. Hence the model $X_t = t + t^2 + season + \epsilon_t$, cannot be used to make statistically significant assumptions. This model will not be considered as a valid candidate in the model selection process.

Regularized regression model (LASSO, Ridge, Elastic Net)

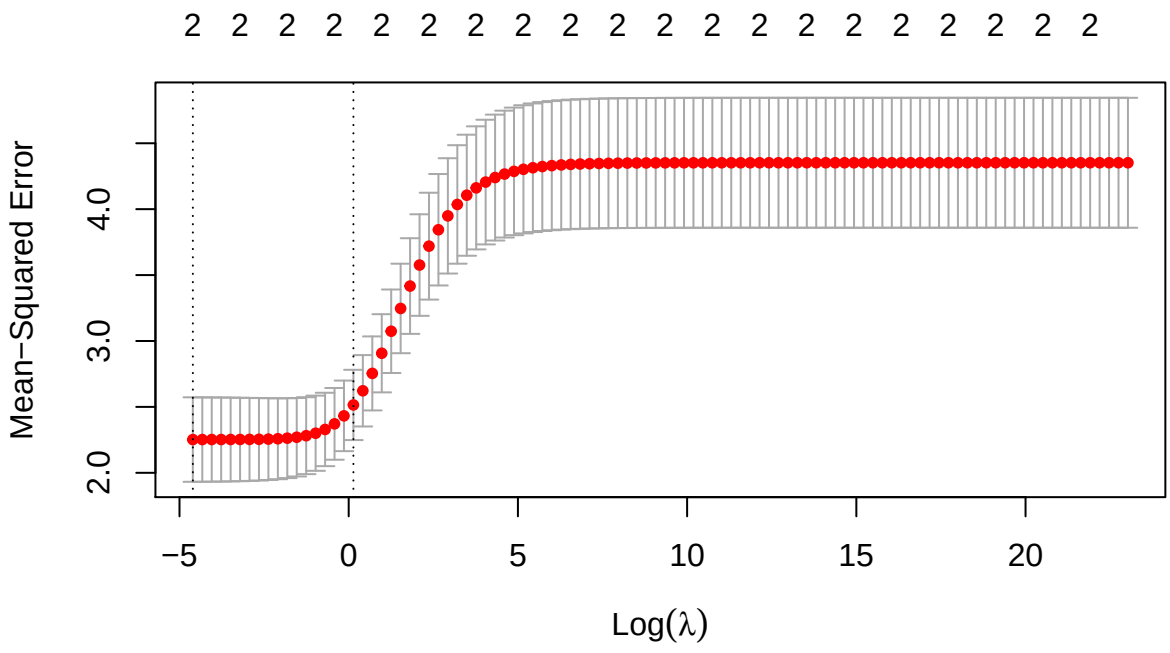
Ridge Regression

For this form of regularized regression, I choose an optimal λ value by using *cv.glmnet* and a range of possible lambda values. I then used the λ and $\alpha = 0$ to determine *fit_ridge* . For this regression model, I also tried variations of the time factor t as t^2 and t^3, however *t_bind* worked best when it consisted of time t and seaonal effect *month* since this produced the lowest APE value.

```
set.seed(1)
library(glmnet)
t <- as.vector(time(nybirth_train))
month <- as.factor(cycle(nybirth_train))
#t_2 <- t^2
#t_3 <- t^3
t_bind <- as.matrix(cbind(t, month))
target <- as.vector(nybirth_train)

fit_ridge <- glmnet(x=t_bind, y=target, alpha=0, standardize = FALSE,
                    intercept = TRUE, family = 'gaussian')
lambdas <- 10^seq(10,-2,length=100)
cv_ridge <- cv.glmnet(x=t_bind, y=target,alpha=0, lambda=lambdas)
optimal_lambda_ridge <- cv_ridge$lambda.min
optimal_lambda_ridge
```

```
## [1] 0.01
```

```
plot(cv_ridge)
```



```
dtest = as.matrix(cbind(time(nybirth_test),cycle(nybirth_test)))
colnames(dtest) <- c("t","month")
pred_ridge <- predict(fit_ridge, s=optimal_lambda_ridge, newx=dtest)
ape_ridge <- sum((pred_ridge- nybirth_test)^2)/length(nybirth_test)
ape_ridge
```

```
## [1] 1.695028
```

We find that the optimal lambda value in this case is 0.01. We use this optimal lambda value to build the Ridge regression model.

Lasso Regression

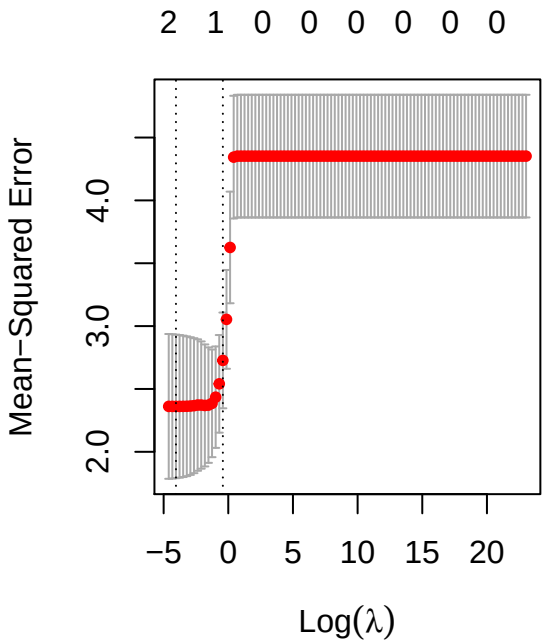
We use the same *t_bind* and *target* factors defined for the Ridge Regression. In this case $\alpha = 1$ and we obtain a new *optimal_lambda_lasso* value that is used to fit the Lasso Regression model.

```
cv_lasso <- cv.glmnet(x=t_bind, y=target,alpha=1, lambda=lambdas)
optimal_lambda_lasso <- cv_lasso$lambda.min
optimal_lambda_lasso
```

```
## [1] 0.01747528
```

```
par(mfrow=c(1,2))
plot(cv_lasso)
fit_lasso <- glmnet(x=t_bind, y=target, alpha=1, standardize = FALSE,
                    intercept = TRUE, family = 'gaussian')
pred_lasso <- predict(fit_lasso, s=optimal_lambda_lasso, newx=dtest)
ape_lasso <- sum((pred_lasso - nybirth_test)^2)/length(nybirth_test)
ape_lasso
```

```
## [1] 1.705692
```



We find that the optimal lambda value in this case is 0.0174753. We use this optimal lambda value to build the Lasso regression model.

Elastic Net Regression

I choose a different strategy than in Lasso and Ridge. I will have to tune the training model over different ranges of λ and α. For this I can use the function *train* and *trainControl* from the *caret* package. The reason I am using these functions is because they allow me flexibility to train the model with optimal tuning parameters of *lambda* and *alpha*. Most tuning parameters for the function are chosen using the associated *R Documentation* .

For *trainControl* , I tuned the function in the following way:

- method: *repeatedcv* (for repeated training/test splits)
- number: *10* - *15* number of folds or resampling iterations
- repeats: *5* - *10* The number of complete sets of folds to compute - for repeated cv
- search: *random*, a random search procedure is used which describes howthe tuning parameter grid is determined (Bergstra and Bengio, 2012)
- verboseIter: *TRUE*, in order to print the training log

For *train* I tuned the function in the following way:

- preProcess: preprocessing that has to be done on any test set (determined by looking at the training set). *center* subtracts the mean of the predictor's data (again from the data in x) from the predictor values while *scale* divides by the standard deviation.
- tuneLength: We choose *10* , its the granularity of the tuning parameter grid.
- trControl: *trainControl* with the model training tuning parameters defined as above.

```
library(tidyverse)
library(ggplot2)
library(caret)

tc <- trainControl(method="repeatedcv", number=10, repeats=5, search="random",
                   verboseIter=FALSE)
#use training control in the elastic fit
dtrain <- data.frame(cbind(target,t,month))
fit_elastic <- train(target~.,data=dtrain,method="glmnet",
                     preProcess=c("center","scale"),tuneLength=10, trControl=tc)
fit_elastic$bestTune
```

```
##       alpha      lambda
## 2 0.2321611 0.003941781
```

Now that we have the values of α and λ as 0.232, 0.004 we use those values to fit the Elastic Net Regression model.

```
pred_elastic <- predict(fit_elastic, dtest)
ape_elastic <- sum((pred_elastic - nybirth_test)^2)/length(nybirth_test)
ape_elastic
```

```
## [1] 1.702868
```

So our final prediciton power with Ridge, Lasso and Elastic Net Regression is determined with their respective *validation APE*.

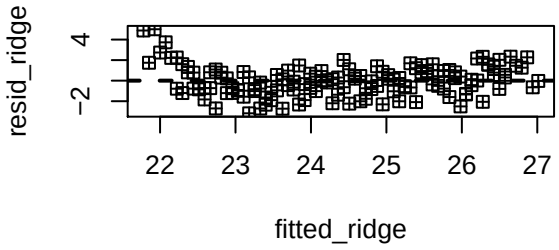
- Ridge: 1.6950281
- Lasso: 1.7056917
- Elastic Net: 1.7028677

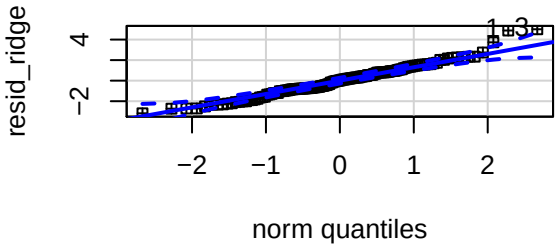
Of the three, the one with the best prediction power is *Ridge Regression* . In order to determine if the model can be used to make statistically significant inferences, I perform residual diagnostics for the model.

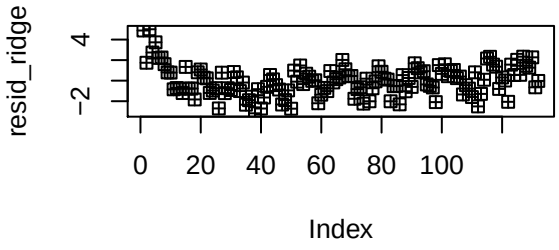
```
fit_ridge_train<- predict(fit_ridge, s=optimal_lambda_ridge, newx=t_bind)
resid_ridge <- as.vector(nybirth_train) - as.vector(fit_ridge_train)
fitted_ridge <- as.vector(fit_ridge_train)
par(mfrow=c(2,2))
# residuals vs. fitted values
plot(resid_ridge~fitted_ridge , pch=12)
abline(h=0,lwd=2,lty=2)
#QQplot
#qqnorm(resid_model)
#qqline(resid_model)
library(car)
qqPlot(resid_ridge , pch=12)
```

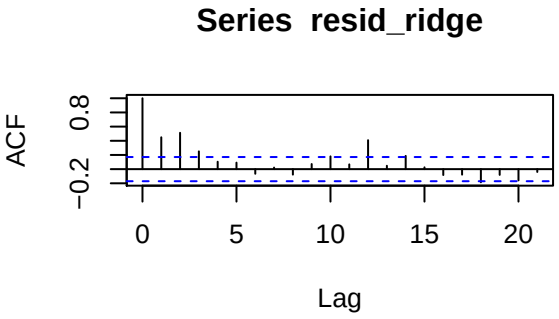
```
## [1] 3 1
```

```
#Residuals plot
plot(resid_ridge , pch=12)
#ACF plot
acf(resid_ridge , pch=12)
```










```
#test constant variance
resid_seg <- rep(1:11, each=12)
fligner.test(resid_ridge, resid_seg)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  resid_ridge and resid_seg
## Fligner-Killeen:med chi-squared = 9.6143, df = 10, p-value = 0.475
```

```
#test Normality
shapiro.test(resid_ridge)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid_ridge
## W = 0.97831, p-value = 0.03296
```

```
#randomness test
library(lawstat)
runs.test(resid_ridge)
```

```
##
##  Runs Test
##
## data:  resid_ridge
## statistic = -4.0192, runs = 44, n1 = 66, n2 = 66, n = 132, p-value =
## 5.841e-05
## alternative hypothesis: nonrandomness
```

```
library(randtests)
difference.sign.test(resid_ridge)
```

```
##
##  Difference Sign Test
##
## data:  resid_ridge
## statistic = -1.0513, n = 132, p-value = 0.2931
## alternative hypothesis: nonrandomness
```

Insights from the residuals plot:

- The residuals vs the fitted values indicates a bowl-shaped pattern and the residuals time series plot indicates a bowl-shaped pattern as well . This suggests that the mean is not constant 0
- The p-value in the runs test (5.841e-05) is very small, which confirms evidence against randomness of residuals. In the ACF plot, more than 5% of the spikes cross the confidence interval, indicating that the residuals are in fact dependant.
- With the Fligner test we see that the p-value (0.475) is fairly large, hence there is no violation of the constant variance assumption.
- From the QQ plot, we see that the upper right tail has significant outliers and the lower left tail of the distribution deviates slightly from the normal distribution, however the Shapiro test's p-value (0.03296) indicates that the residuals are not normal.

From the Residuals diagnostics, we see that the constant mean 0,independence and normality rules have been violated. However, the residuals have constant variance. Hence the *Ridge Regression* model , cannot be used to make statistically significant assumptions. This model will not be considered as a valid candidate in the model selection process.

Holt-Winters Model

I use the Exponential,Additive and Multiplicative Holt Winters Models to fit the data.

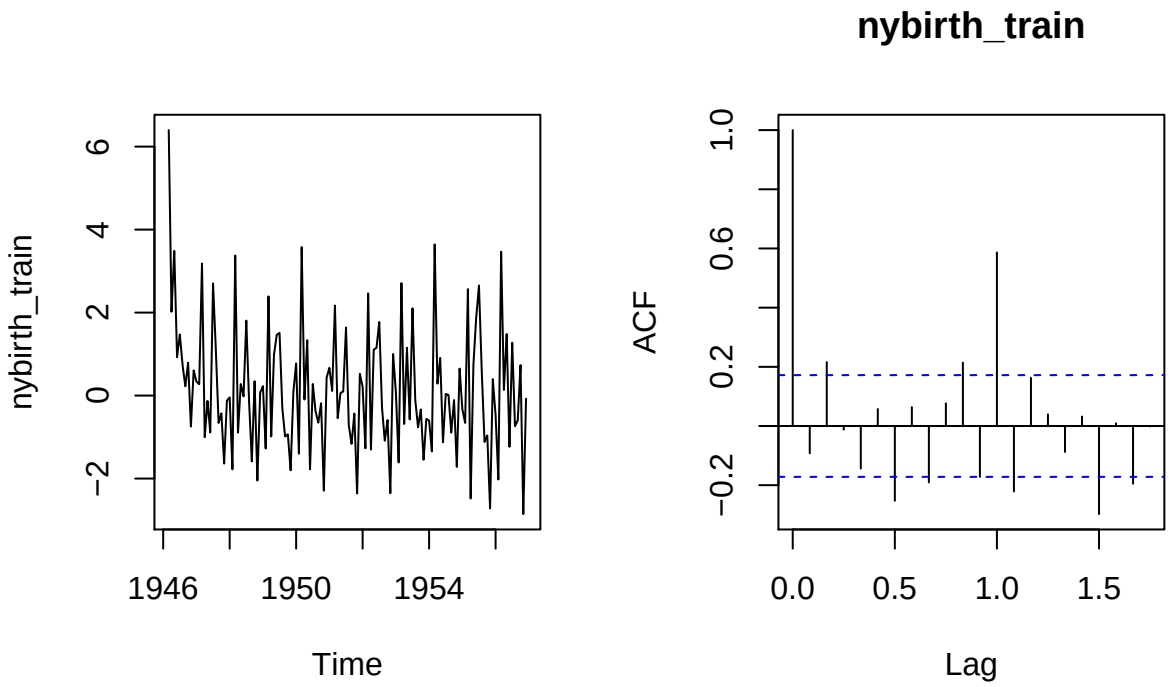
```
library(stats)

#exponential data
hwfit_exp <- HoltWinters(nybirth_train, gamma=FALSE)
random_exp <- nybirth_train - hwfit_exp$fitted[,1]
sse_train_exp <- hwfit_exp$SSE
pred <- predict(hwfit_exp, n.ahead=36)
ape_validation_exp<-  sum((pred - nybirth_test)^2)/length(nybirth_test)
sse_train_exp
```

```
## [1] 304.2382
```

```
ape_validation_exp
```

```
## [1] 24.40066
```

```
par(mfrow=c(1,2))
plot(random_exp)
acf(random_exp)
```




```
#additive model
hwfit_additive <- HoltWinters(nybirth_train, seasonal='additive')
random_additive <- nybirth_train - hwfit_additive$fitted[,1]
```

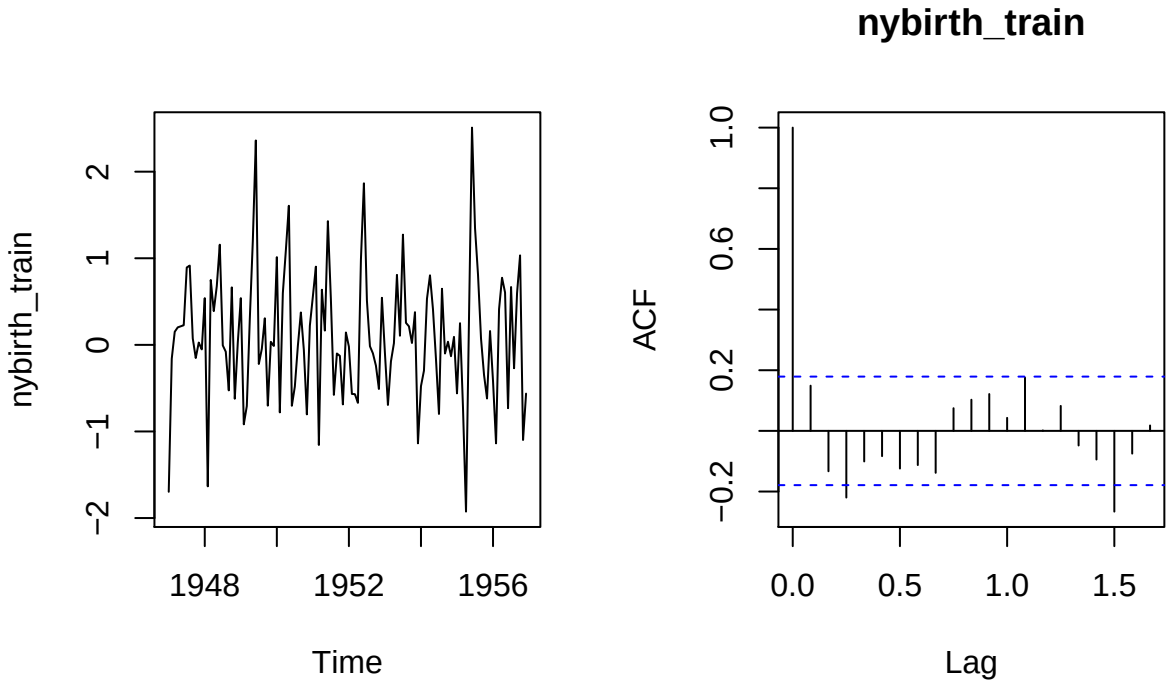
```
sse_train_add <- hwfit_additive$SSE
pred1 <- predict(hwfit_additive, n.ahead = 36)
ape_validation_add <- sum((pred1 - nybirth_test)^2)/length(nybirth_test)
sse_train_add
```

```
## [1] 68.04345
```

```
ape_validation_add
```

```
## [1] 1.169169
```

```
par(mfrow=c(1,2))
plot(random_additive)
acf(random_additive)
```



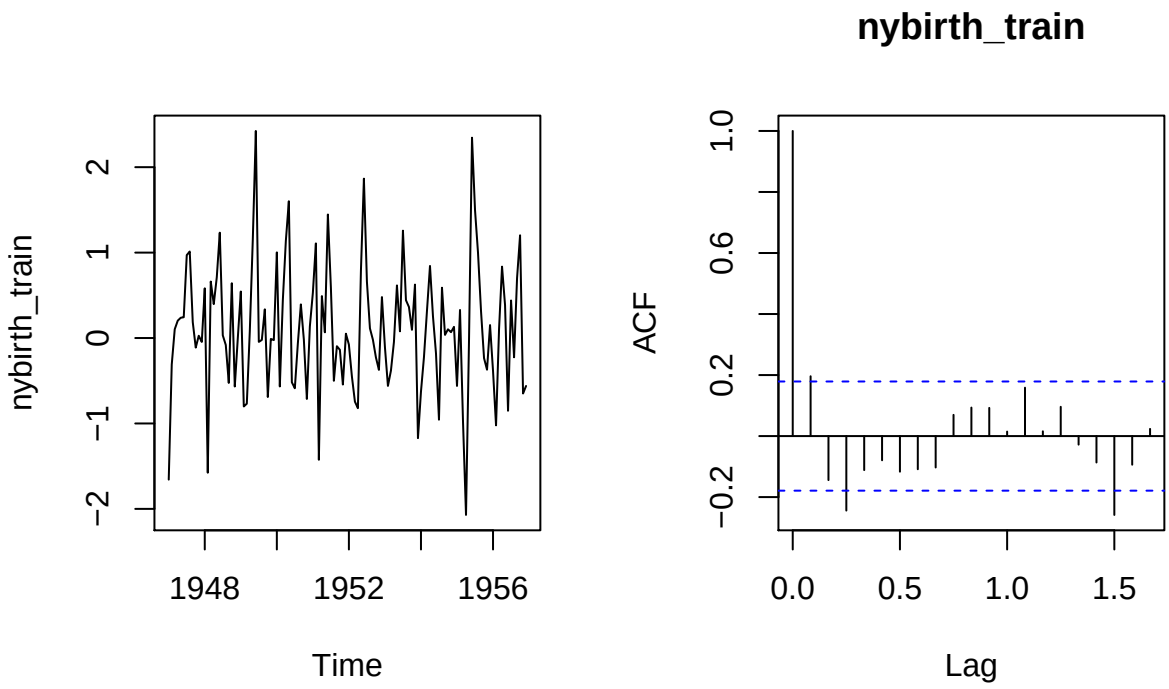

```
#multiplicative model
hwfit_mult <- HoltWinters(nybirth_train, seasonal="multiplicative")
random_mult <- nybirth_train - hwfit_mult$fitted[,1]
sse_train_mult <- hwfit_mult$SSE
pred2 <- predict(hwfit_mult, n.ahead = 36)
ape_validation_mult <- sum((pred2 - nybirth_test)^2)/length(nybirth_test)
sse_train_mult
```

```
## [1] 68.34582
```

```
ape_validation_mult
```

```
## [1] 1.923149
```

```
par(mfrow=c(1,2))
plot(random_mult)
acf(random_mult)
```



We see that the Exponetial Model performs very poorly (significant trend in the residuals and dependancy as seen in the ACF) hence we eliminate it immediately and only discuss the other two.

Insights from the Additive and Multiplicative Holt Winters models:

- There is no trend/seasonality from the plots (additive and multiplicative) hence the random components are stationary.
- Since the importance of the model will be based on its ability to forecast the growth of the city, we test both the fit and prediction power of the model, but primarily focus on the prediction power.
- The fit with the multiplicative HW Model 68.3458175 is slightly better than with the additive HW model 68.0434467 , however the predictive power of the additive model 1.1691691 is much better that the multiplicative model's predictive power 1.9231493.
- Hence of the two, I would choose the *Holt Winters Additive model* for its predictive power.

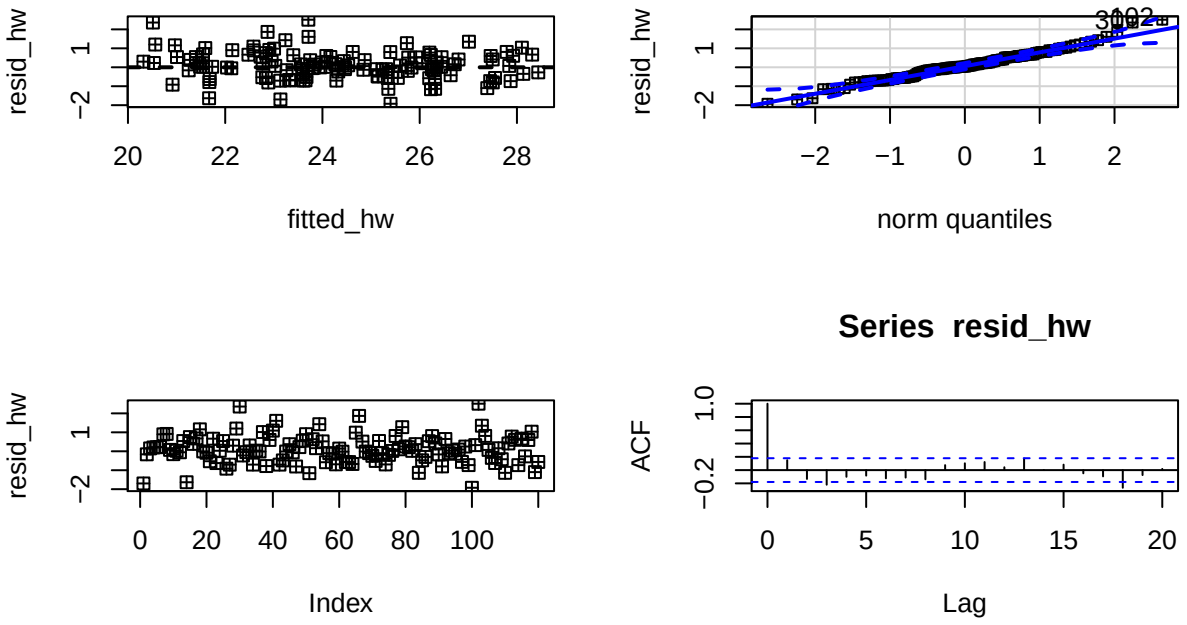
In order to determine if the model can be used to make statistically significant inferences, I perform residual diagnostics for the model.

```
fit_hw_train<- hwfit_additive$fitted[,1]
nytrain <- window(nybirth_ts, start= 1947, end=1956+11/12)
resid_hw <- as.vector(nytrain) - as.vector(fit_hw_train)
```

```r
fitted_hw <- as.vector(fit_hw_train)
par(mfrow=c(2,2))
# residuals vs. fitted values
plot(resid_hw~fitted_hw , pch=12)
abline(h=0,lwd=2,lty=2)
#QQplot
#qqnorm(resid_model)
#qqline(resid_model)
library(car)
qqPlot(resid_hw , pch=12)
```

```
## [1] 102  30
```

```r
#Residuals plot
plot(resid_hw , pch=12)
#ACF plot
acf(resid_hw , pch=12)
```




```r
#test constant variance
resid_seg <- rep(1:10, each=12)
fligner.test(resid_hw, resid_seg)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  resid_hw and resid_seg
## Fligner-Killeen:med chi-squared = 9.1344, df = 9, p-value = 0.425
```

```r
#test Normality
shapiro.test(resid_hw)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid_hw
## W = 0.98384, p-value = 0.1612
```

```r
#randomness test
library(lawstat)
runs.test(resid_hw)
```

```
##
##  Runs Test
##
## data:  resid_hw
## statistic = -1.1001, runs = 55, n1 = 60, n2 = 60, n = 120, p-value =
## 0.2713
## alternative hypothesis: nonrandomness
```

```r
library(randtests)
difference.sign.test(resid_hw)
```

```
##
##  Difference Sign Test
##
## data:  resid_hw
## statistic = 0.47238, n = 120, p-value = 0.6367
## alternative hypothesis: nonrandomness
```

Insights from the residuals plot:

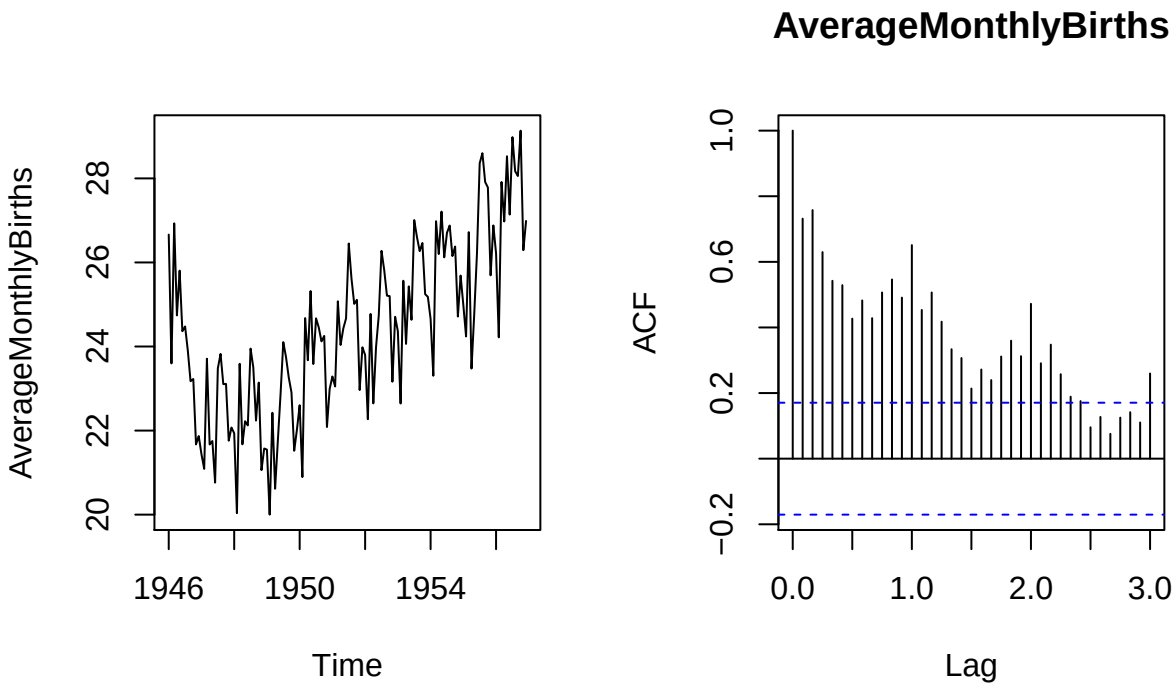
- The residuals vs the fitted values is well scatters about the mean 0 and the residuals time series plot is well scattered too with no significant pattern . This suggests that the mean is a constant about 0.
- The p-value in the runs test (0.2713) is high, which confirms the hypothesis that the residuals are random. In the ACF plot, less than 5% of the spikes cross the confidence interval, indicating that the residuals are in fact independent.
- With the Fligner test we see that the p-value (0.425) is fairly large, hence the constant variance assumption of residuals holds true.
- From the QQ plot, we see that the upper right tail has significant outliers and the lower left tail of the distribution deviates sligthly from the normal distribution, however the Shapiro test's p-value (0.1612) indicates that the residuals are normal.

From the Residuals diagnostics, we see that the constant mean 0,independence, constant variance, normality and randomness test have been fulfilled! Hence the *Holt Winters Additive* model can be used to make statistically significant assumptions. This model will be considered as a valid candidate in the model selection process.

Box Jenkins Model (ARIMA and/or SARIMA)

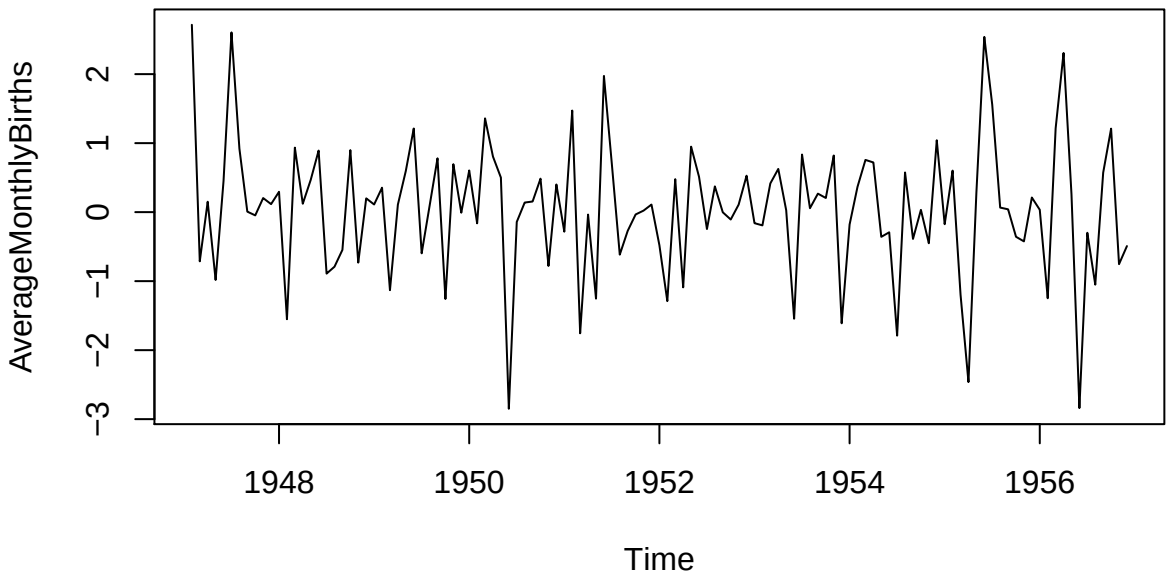
- ARIMA is represented by (p, d, q)
- SARIMA is represented by $(p, d, q)(P, D, Q)s$

```
par(mfrow=c(1,2))
plot(nybirth_train)
acf(nybirth_train, lag.max=36)
```

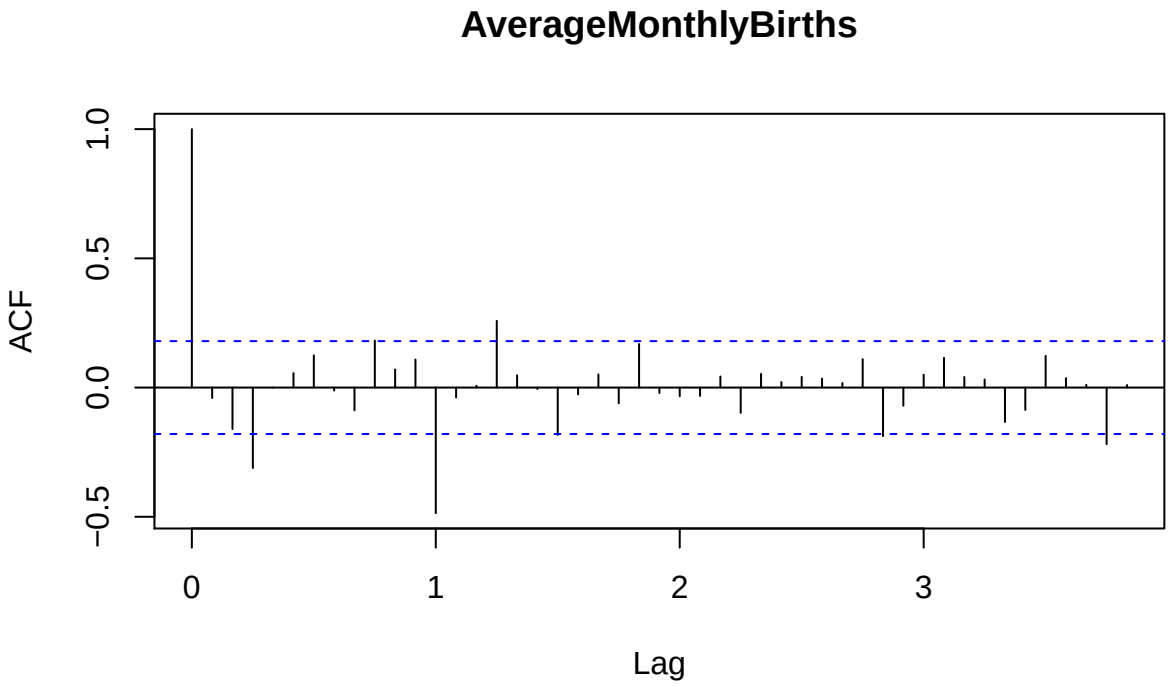


We observe that the data has both seasonality and a decreasing trend. The ACF plot also has a clear decreasing trend and periodic with a seasonal lag every 12 months. Hence seasonal and regular differencing is required.

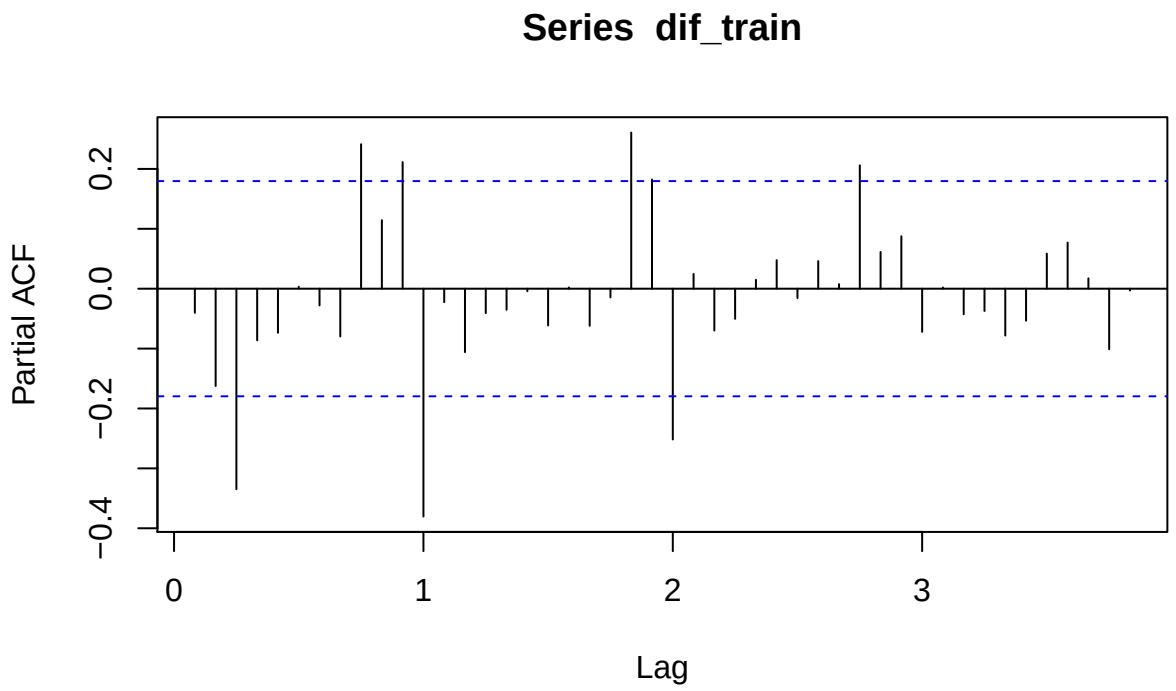
```
dif_train <- diff(nybirth_train)#differencing to remove linear trend
dif_train <- diff(dif_train, lag=12)#differencing to remove seasonality

plot(dif_train)
```




```
acf(dif_train, lag.max=46)
```



```
pacf(dif_train, lag.max=46)
```




```
seg <- c(rep(1:11, each=10), rep(12, 9))
flig1 <- fligner.test(dif_train,seg)
```

We also use the Fligner Test of homogeniety of variances and the differenced data has constant variance since the p value 0.8333696 is the greater than 0.05. So the differenced data is stationary with $d = D = 1$ *and* $s = 12$. The SARIMA models that I propose are -

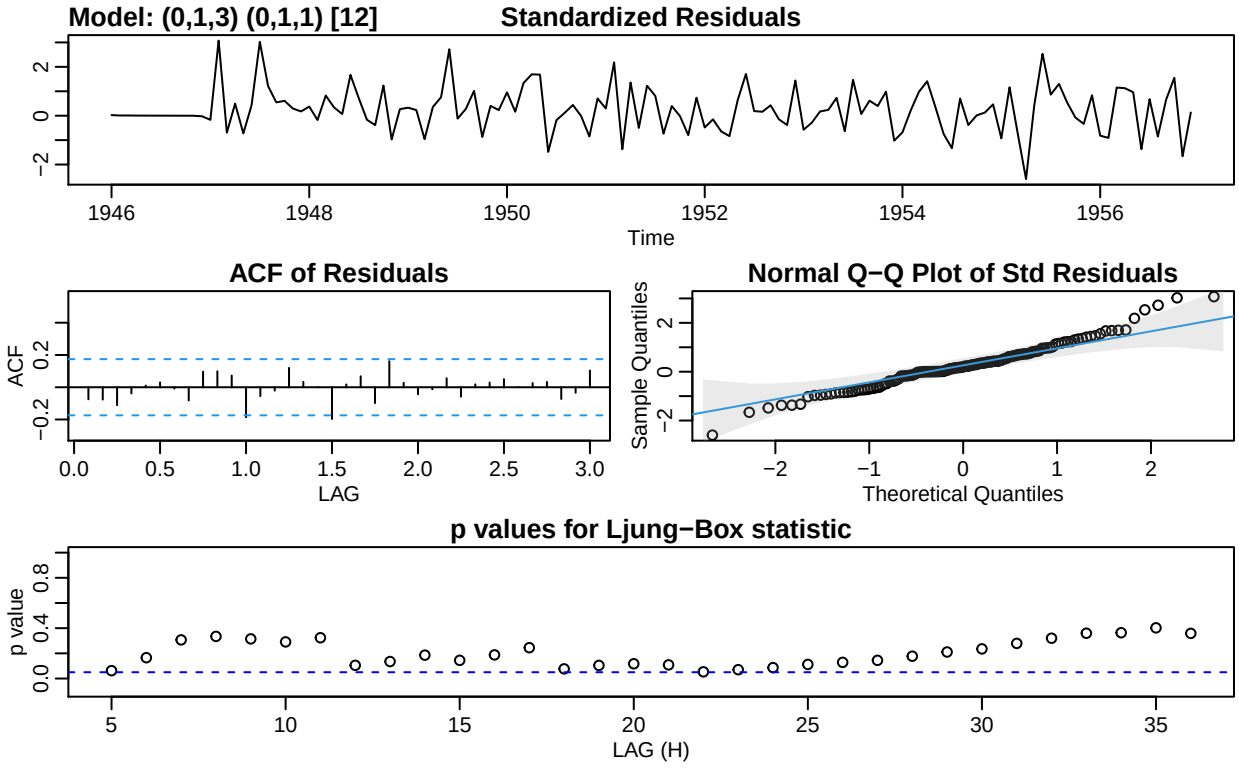
- Model 1: We first ignore the seasonal lags. The pacf undergoes exponential decay, the acf cuts off after lag 3, so $q = 3$ and $p = 0$. Focussing on the seasonal lags now, the acf cuts off after lag 1 , so $Q = 1$ and $P = 0$. So one model can be $SARIMA(0, 1, 3)(0, 1, 1)_{12}$
- Model 2: We first ignore the seasonal lags. The acf undergoes exponential decay, and the pacf cuts off after lag 3, so $q = 0$ and $p = 3$. Focussing on the seasonal lags now, the pacf cuts off after lag 2, so $Q = 0$ and $P = 2$. So one model can be $SARIMA(3, 1, 0)(2, 1, 0)_{12}$
- Model 3: $SARIMA(0, 1, 3)(2, 1, 0)_{12}$
- Model 4: $SARIMA(3, 1, 0)(0, 1, 1)_{12}$

The other two combinations of models that originate from Model 1 and 2 are Model 3 and 4. Lets use these four models to evaluate their prediction power.

```
library(astsa)
fit_sar1 <- sarima(nybirth_train,p=0,d=1,q=3,P=0,D=1,Q=1,S=12,Model = TRUE)
```

```
## initial  value -0.036382
## iter   2 value -0.251489
```

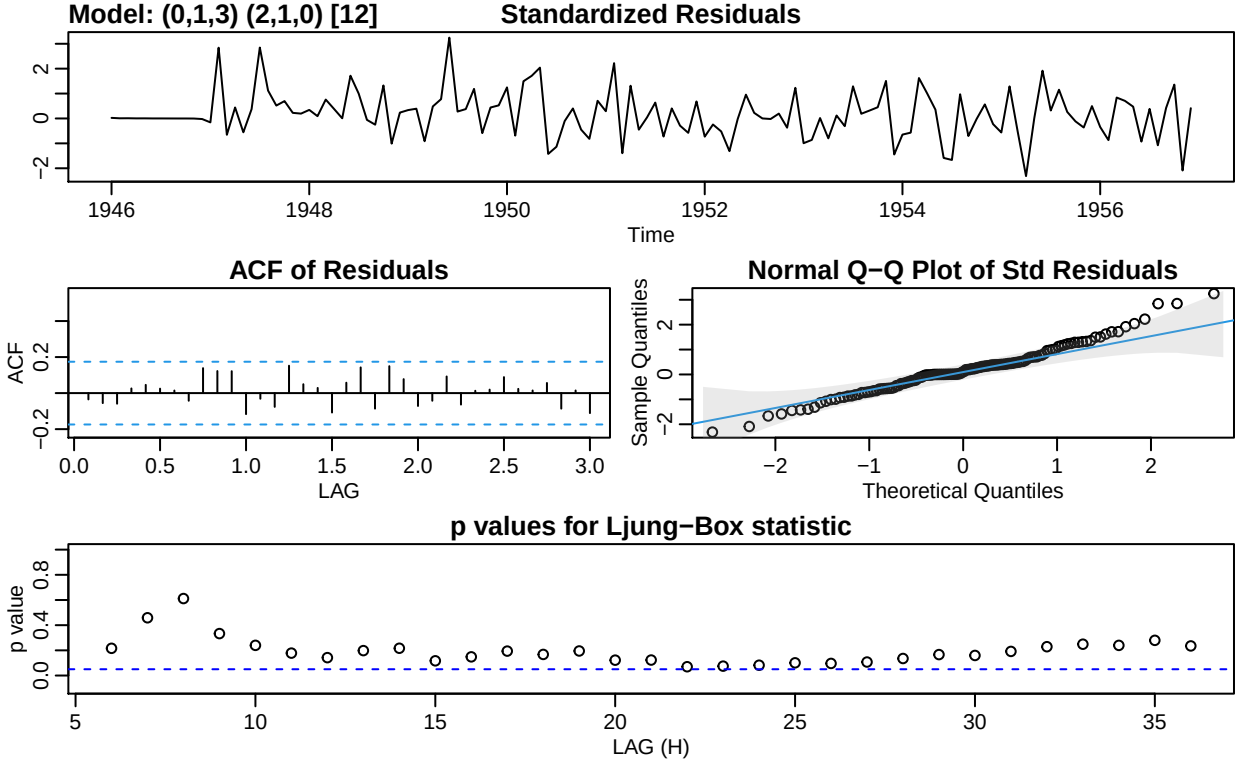
```
## iter   3 value -0.285679
## iter   4 value -0.292031
## iter   5 value -0.295261
## iter   6 value -0.295451
## iter   7 value -0.295455
## iter   8 value -0.295455
## iter   8 value -0.295455
## iter   8 value -0.295455
## final  value -0.295455
## converged
## initial  value -0.333606
## iter   2 value -0.368029
## iter   3 value -0.371304
## iter   4 value -0.371603
## iter   5 value -0.371737
## iter   6 value -0.371757
## iter   7 value -0.371759
## iter   8 value -0.371759
## iter   8 value -0.371759
## iter   8 value -0.371759
## final  value -0.371759
## converged
```




```
fit_sar2 <- sarima(nybirth_train,p=0,d=1,q=3,P=2,D=1,Q=0,S=12,Model = TRUE)
```

```
## initial  value -0.047611
## iter   2 value -0.330885
## iter   3 value -0.380386
```

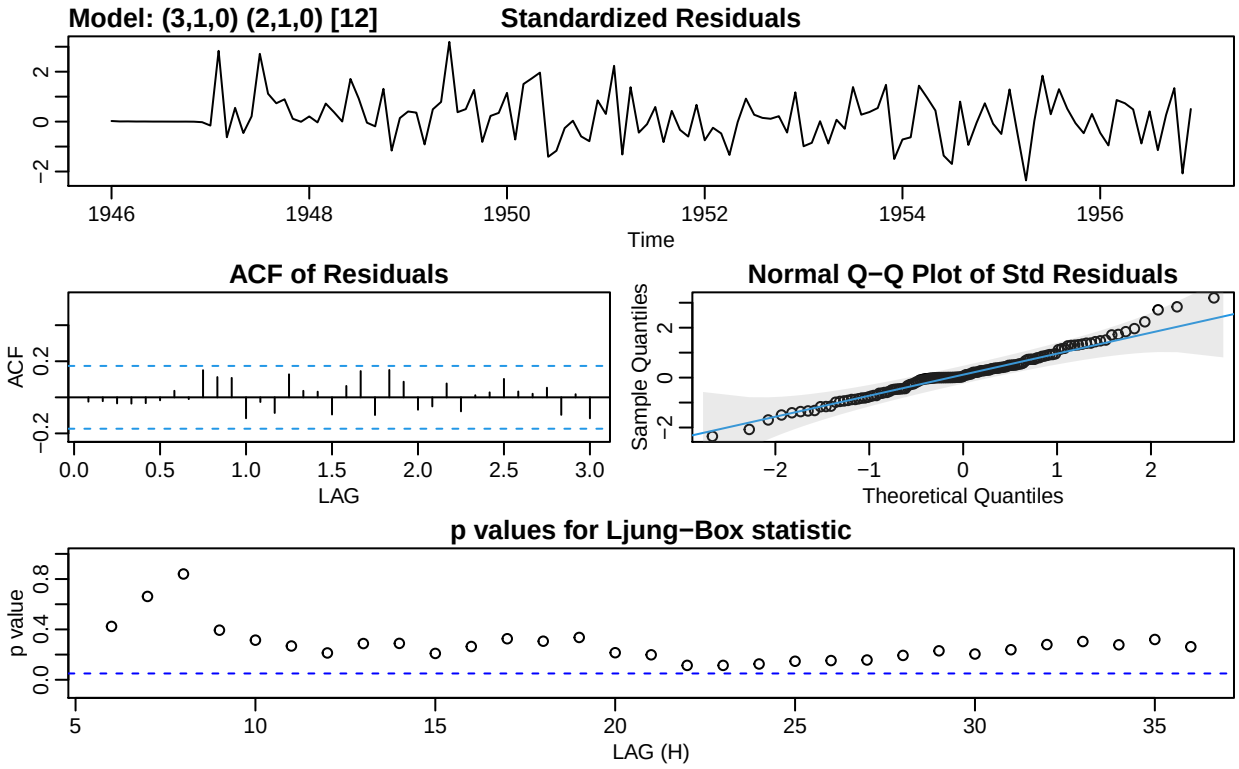
```
## iter   4 value -0.428206
## iter   5 value -0.431909
## iter   6 value -0.433031
## iter   7 value -0.433052
## iter   8 value -0.433059
## iter   9 value -0.433059
## iter  10 value -0.433059
## iter  10 value -0.433059
## iter  10 value -0.433059
## final  value -0.433059
## converged
## initial  value -0.359316
## iter   2 value -0.361836
## iter   3 value -0.363931
## iter   4 value -0.364090
## iter   5 value -0.364106
## iter   6 value -0.364109
## iter   7 value -0.364109
## iter   7 value -0.364110
## iter   7 value -0.364110
## final  value -0.364110
## converged
```




```
fit_sar3 <- sarima(nybirth_train,p=3,d=1,q=0,P=2,D=1,Q=0,S=12,Model = TRUE)
```

```
## initial  value -0.039834
## iter   2 value -0.318101
## iter   3 value -0.386916
```

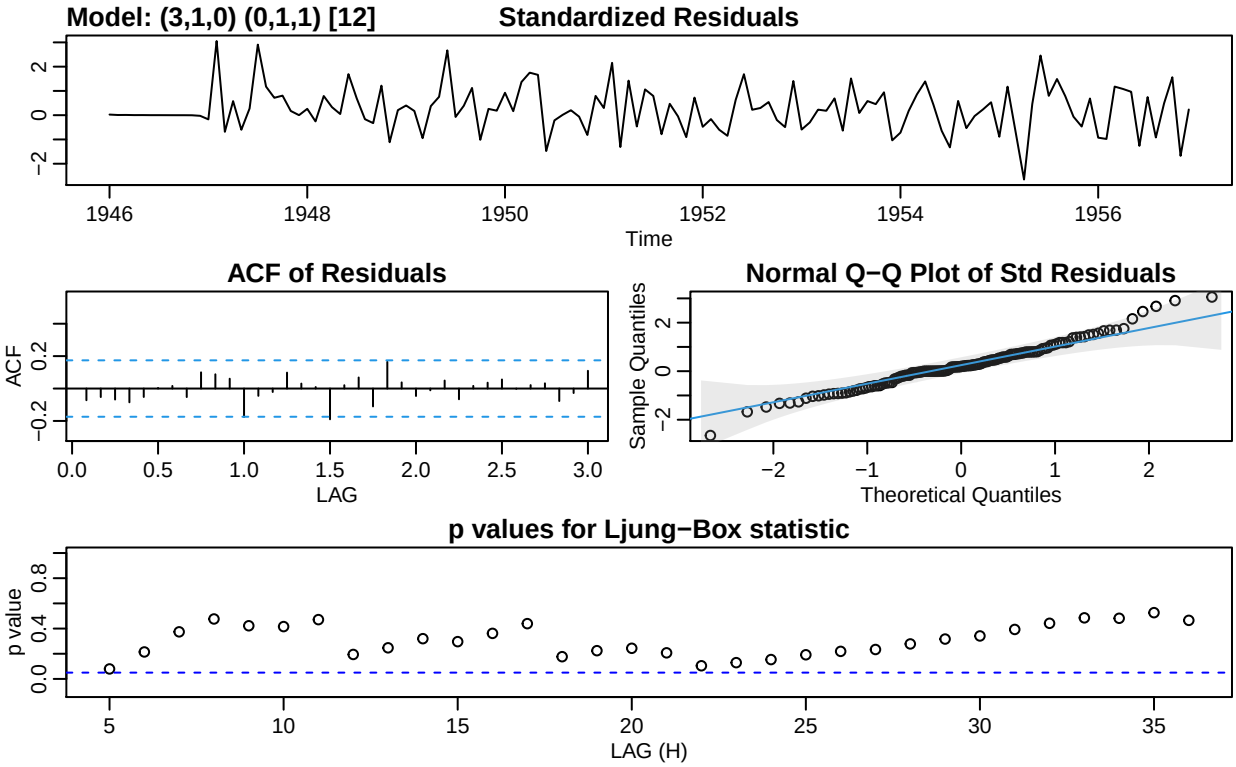
```
## iter   4 value -0.421909
## iter   5 value -0.424350
## iter   6 value -0.424475
## iter   7 value -0.424522
## iter   8 value -0.424522
## iter   8 value -0.424522
## iter   8 value -0.424522
## final  value -0.424522
## converged
## initial  value -0.368042
## iter   2 value -0.369355
## iter   3 value -0.369570
## iter   4 value -0.369607
## iter   5 value -0.369609
## iter   6 value -0.369609
## iter   7 value -0.369609
## iter   7 value -0.369609
## iter   7 value -0.369609
## final  value -0.369609
## converged
```




```
fit_sar4 <- sarima(nybirth_train,p=3,d=1,q=0,P=0,D=1,Q=1,S=12,Model = TRUE)
```

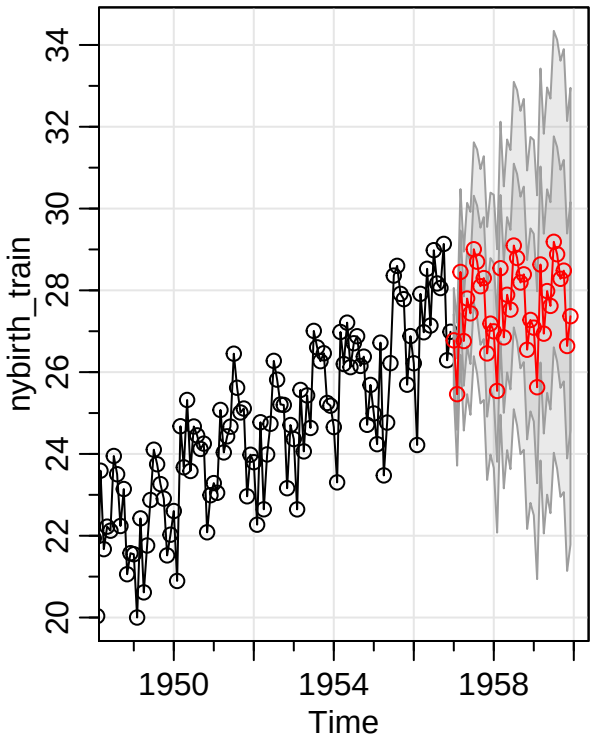
```
## initial  value -0.060664
## iter   2 value -0.308051
## iter   3 value -0.328937
## iter   4 value -0.336352
## iter   5 value -0.344623
```

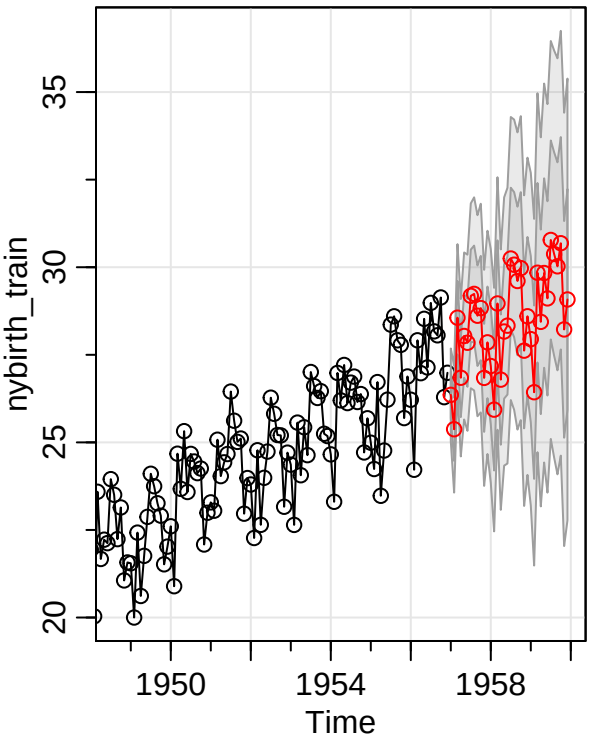
```
## iter   6 value -0.345752
## iter   7 value -0.345852
## iter   8 value -0.345854
## iter   8 value -0.345854
## iter   8 value -0.345854
## final  value -0.345854
## converged
## initial  value -0.349950
## iter   2 value -0.373580
## iter   3 value -0.376002
## iter   4 value -0.376652
## iter   5 value -0.376758
## iter   6 value -0.376815
## iter   7 value -0.376826
## iter   8 value -0.376826
## iter   8 value -0.376826
## iter   8 value -0.376826
## final  value -0.376826
## converged
```



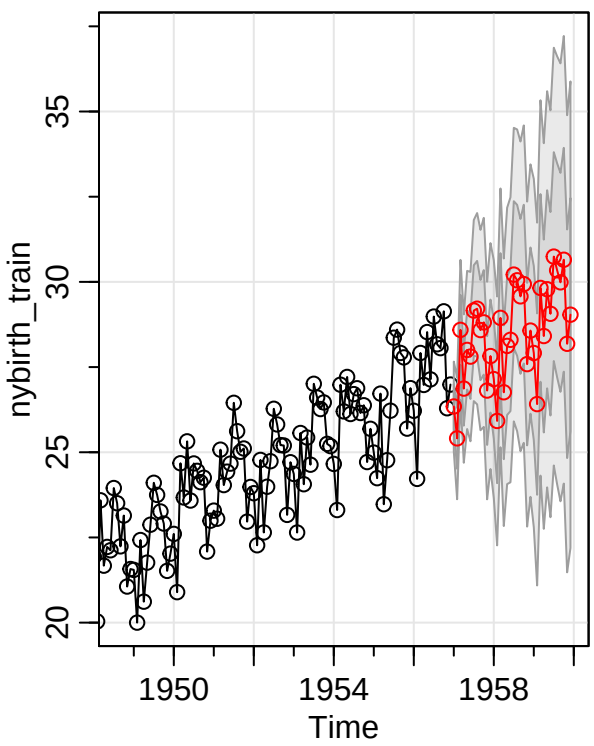
To see which of the models has the best prediction power, we look at the *PRESS* statistic for models 1 - 4.

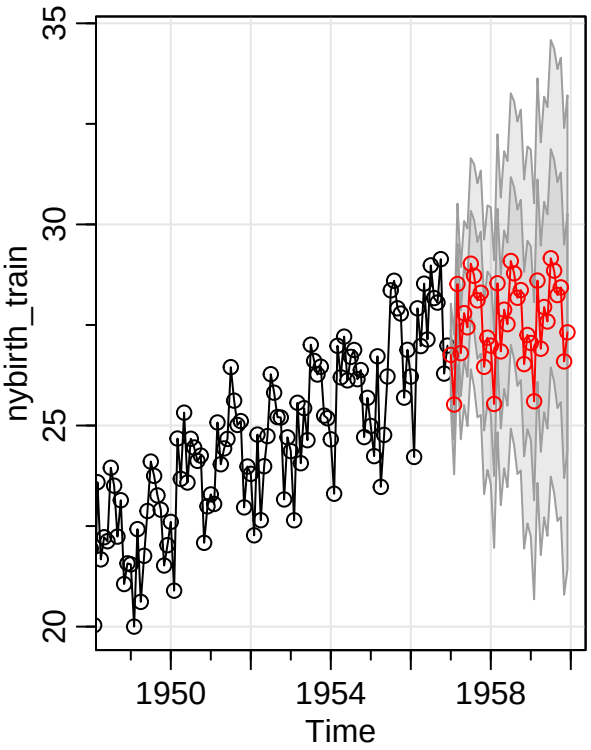
```
par(mfrow=c(1,2))
pred_sar1 <- sarima.for(nybirth_train,n.ahead=36,p=0,d=1,q=3,P=0,D=1,Q=1,S=12)
pred_sar2 <- sarima.for(nybirth_train,n.ahead=36,p=0,d=1,q=3,P=2,D=1,Q=0,S=12)
```






```
par(mfrow=c(1,2))
pred_sar3 <- sarima.for(nybirth_train,n.ahead=36,p=3,d=1,q=0,P=2,D=1,Q=0,S=12)
pred_sar4 <- sarima.for(nybirth_train,n.ahead=36,p=3,d=1,q=0,P=0,D=1,Q=1,S=12)
```





```
APE_SAR =c()
APE_SAR[1]= sum((pred_sar1$pred - nybirth_test)^2)/length(nybirth_test)
APE_SAR[2]= sum((pred_sar2$pred - nybirth_test)^2)/length(nybirth_test)
APE_SAR[3]= sum((pred_sar3$pred - nybirth_test)^2)/length(nybirth_test)
APE_SAR[4]= sum((pred_sar4$pred - nybirth_test)^2)/length(nybirth_test)
APE_SAR
```

```
## [1] 0.5458567 1.9926862 1.9294886 0.5459265
```

Based on the PRESS statistic, the best model is a choice between $SARIMA(0,1,3)(0,1,1)_{12}$ and $SARIMA(3,1,0)(0,1,1)_{12}$. We go ahead and assess the Residual Plot Diagnostic Tests for the two models that have been shown above when training the models. However, after observing the residuals plots for the two models, we confirm that the best one is $SARIMA(3,1,0)(0,1,1)_{12}$. The reason for this choice has been explained below.

- Model 1: $SARIMA(0,1,3)(0,1,1)_{12}$
  - There does not seem to be any pattern or variance instability in the Standardized Residuals Plot.
  - There is no correlation seen in the ACF of the residuals, and we require the residuals to be *white noise*
  - Most points (except the right upper tail and left lower tail) follow the *Gaussian Distribution*
  - For the Ljung Box Statistic we would require all the points to lie above the dotted line, in this case quite a few touch the dotted line.
- Model 2: $SARIMA(3,1,0)(0,1,1)_{12}$
  - There does not seem to be any pattern or variance instability in the Standardized Residuals Plot.
  - There is no correlation seen in the ACF of the residuals, and we require the residuals to be *white noise*
  - Most points (except the right upper tail and left lower tail) follow the *Gaussian Distribution*
  - For the Ljung Box Statistic we would require all the points to lie above the dotted line, which is the case for this model.

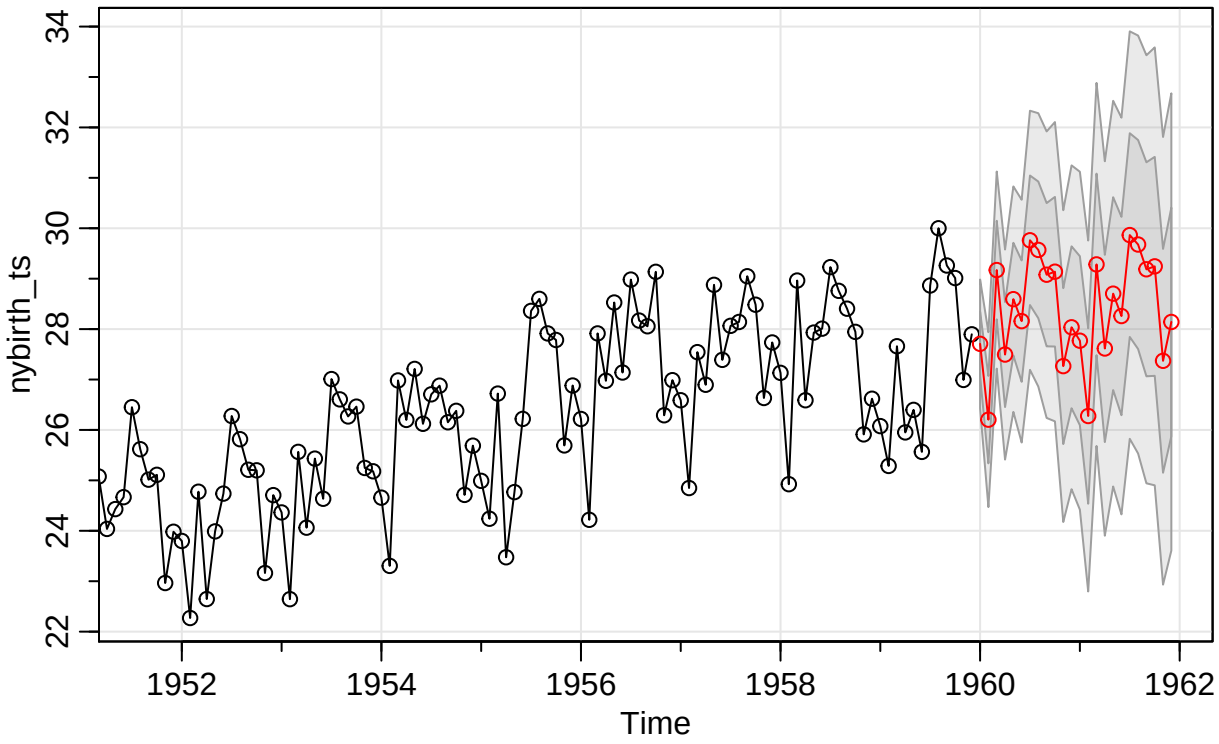
Final Model

Generally models can be chosen based on their best fit, with values like *AIC, BIC and AICc*. To test the predictive power of the models in order to forecast the average births in New York, its best to choose the model with the lowest *APE* that can be used to make statistically significant inferences. The two final candidates are the *Holt Winters Additive Model* and $SARIMA(3,1,0)(0,1,1)_{12}$ since their residual diagnostic plots have been evaluated and fulfill the rules of residuals. When comparing the *APE* we see that the predictive power of the $SARIMA(3,1,0)(0,1,1)_{12}$ (0.5459265) is better than the *Holt Winters* model (1.1691691). Hence the final model is the $SARIMA(3,1,0)(0,1,1)_{12}$.

Prediction

I use the $SARIMA(3,1,0)(0,1,1)_{12}$ to predict the average monthly births in NY for the years 1960 and 1961.

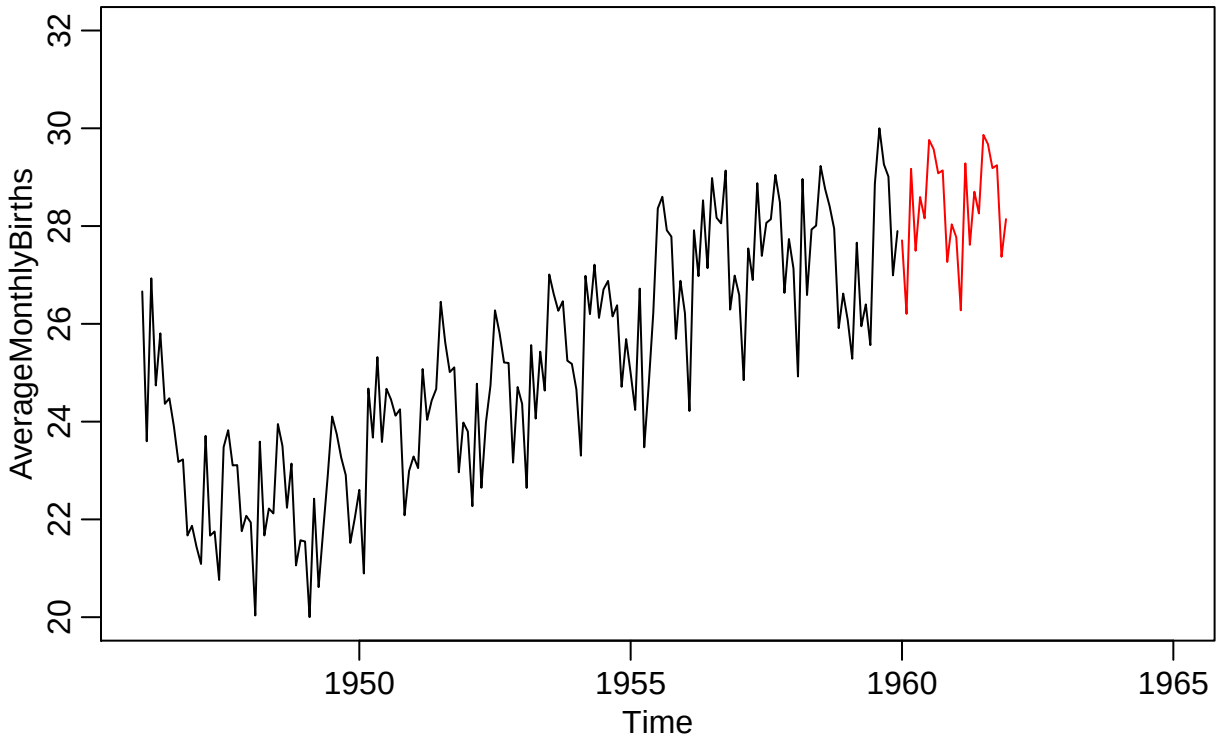
```
final_pred <- sarima.for(nybirth_ts,n.ahead=24,p=3,d=1,q=0,P=0,D=1,Q=1,S=12, plot.all=FALSE)
```




```
final_pred$pred
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 1960 27.70722 26.20384 29.17087 27.49510 28.59369 28.15794 29.76159 29.57086
## 1961 27.76998 26.27639 29.28311 27.61544 28.70292 28.25929 29.86458 29.67740
##           Sep      Oct      Nov      Dec
## 1960 29.07890 29.13834 27.26567 28.03585
## 1961 29.18619 29.24451 27.37113 28.14149
```

```r
plot(nybirth_ts, xlim=c(1946,1965), ylim=c(20,32))
lines(final_pred$pred, col="red")
```



Conclusion

The model-building process in this report highlights the trial and error and insights that develop with building simpler to advanced models for Time series data. It is imperative to note that both the Residual Diagnostics and the predictive power of a model determined by APE, play a significant role in choosing the best model to forecast the average births. The *SARIMA* was the best model to use for forecasting with the NYBirth time series data but other models may have been chosen where fit is also concerned.