Imports ARIMA

Andrew Jowe

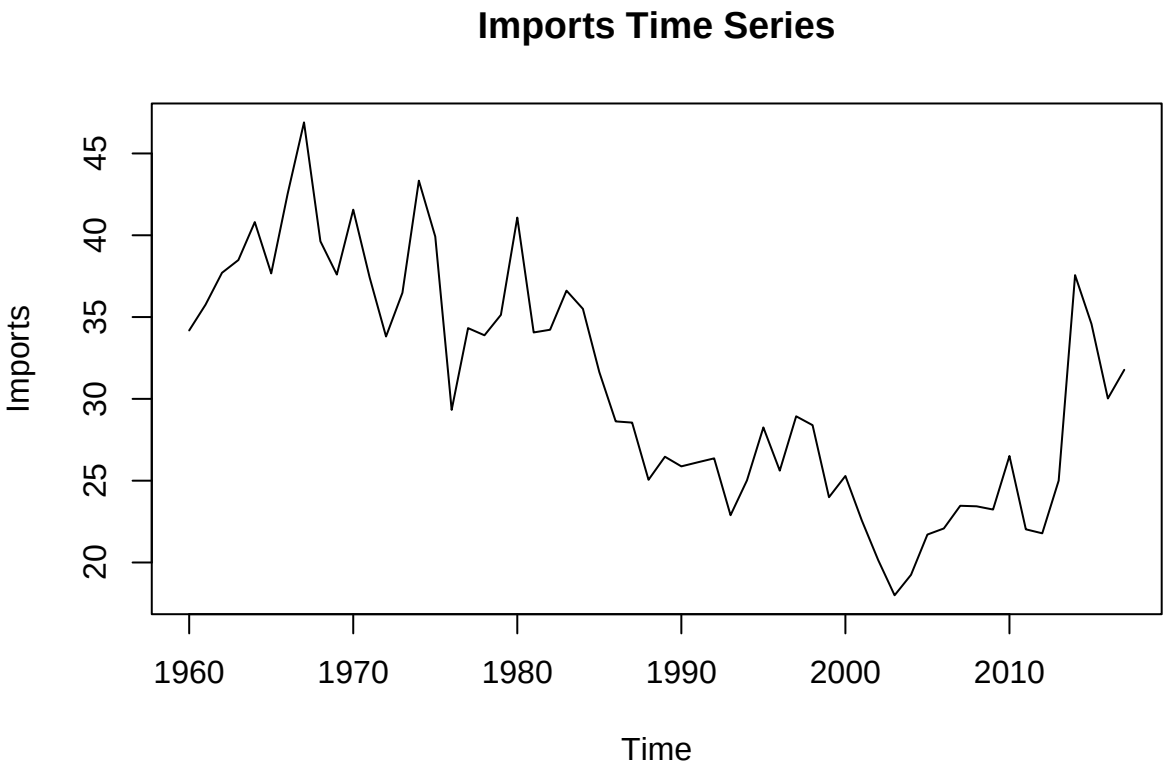
Col Removal

Keep Year, Imports, and GDP columns

```
finalPro_data <- finalPro_data[, c("Year", "Imports")]
```

Plot Time Series

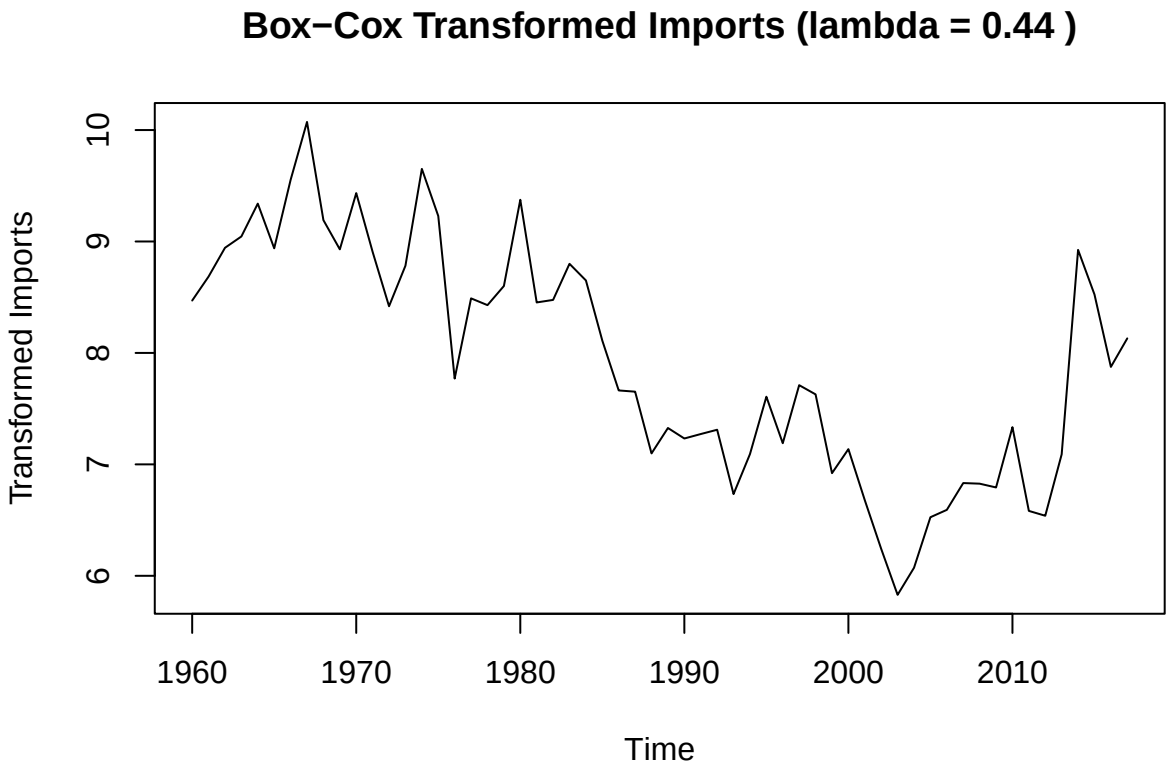
```
# Plot Imports
imports_ts <- ts(finalPro_data$Imports, start = 1960, frequency = 1)

ts.plot(imports_ts, main="Imports Time Series", ylab="Imports")
```



Summary: - Imports time series has upward trend, this shows this is non-stationary - It has peaks around every 10 year: 1980, 1990, 2010

Transform

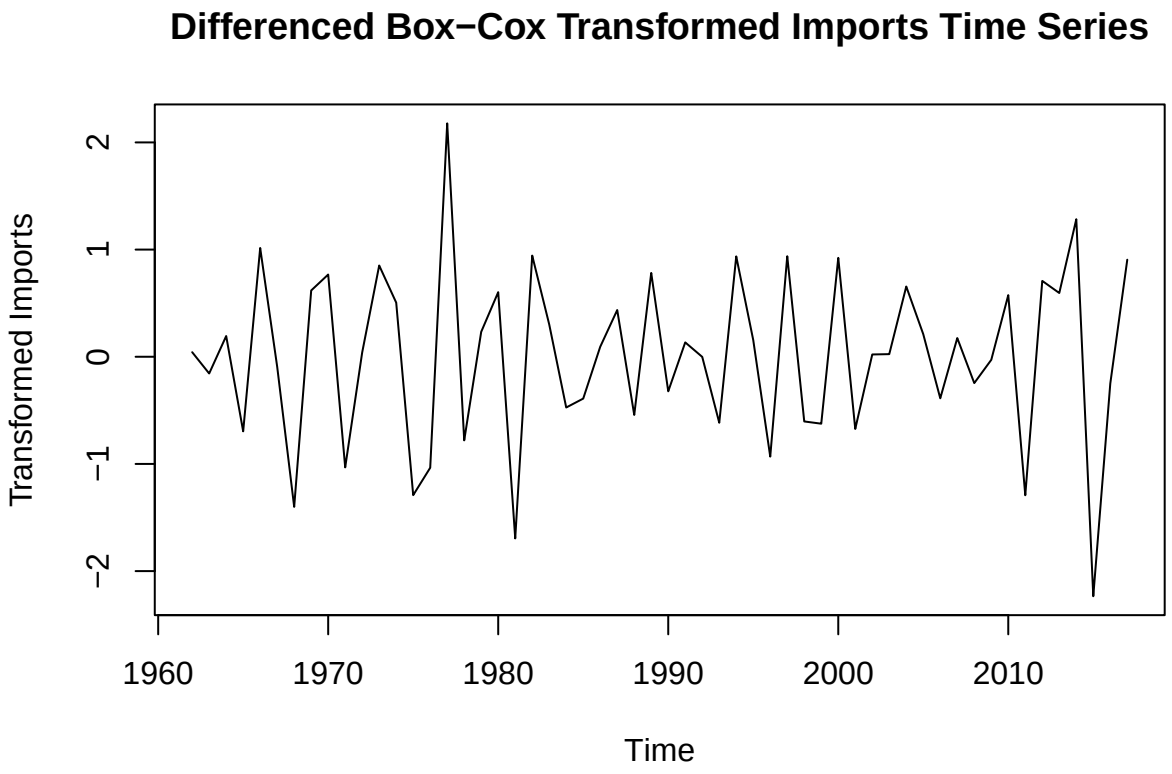
```
# Box-Cox transform Imports
lambda <- BoxCox.lambda(imports_ts)
boxcox_imports_ts <- BoxCox(imports_ts, lambda)
ts.plot(boxcox_imports_ts, main = paste("Box-Cox Transformed Imports (lambda =", round(lambda, 3), ")")
```



We tried log, but residuals not normal.

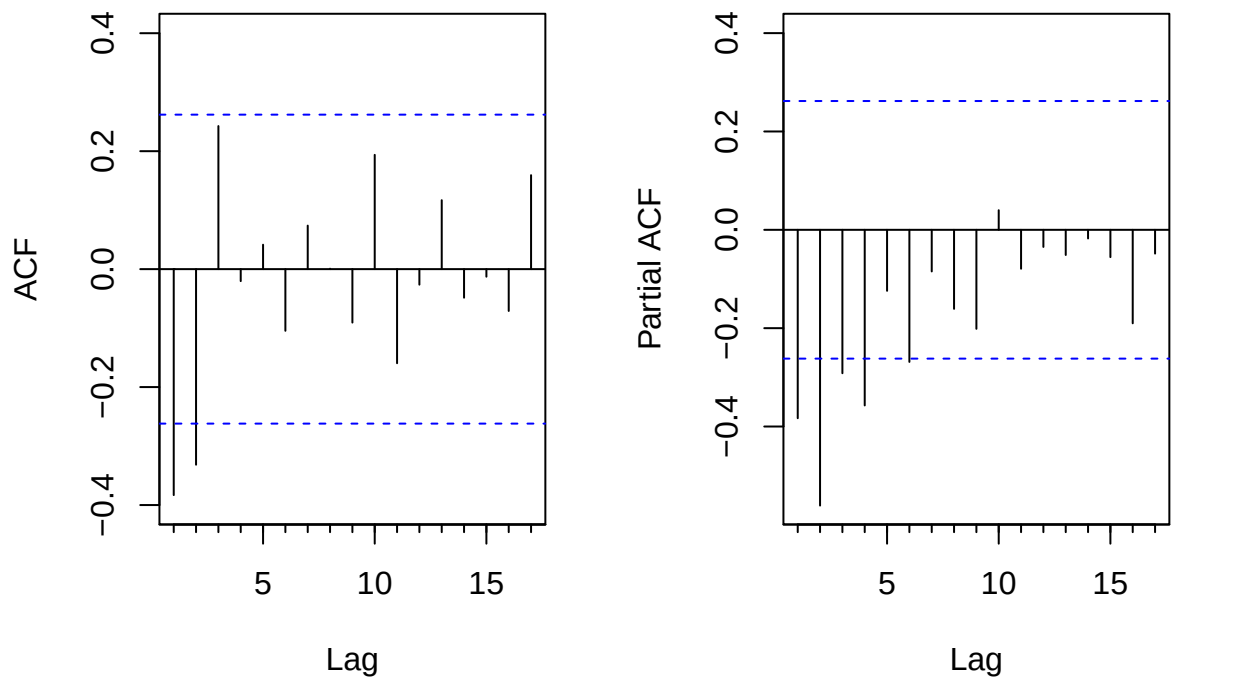
Differencing Imports

```
diff_imports_bc <- diff(boxcox_imports_ts, differences=2)

# Plot differenced Box-Cox Imports
ts.plot(diff_imports_bc, main="Differenced Box-Cox Transformed Imports Time Series", ylab="Transformed
```



ACF / PACF plots

```
# ACF and PACF of the transformed and differenced series
par(mfrow = c(1, 2))
Acf(diff_imports_bc, main = "ACF of Transformed + Differenced Series")
Pacf(diff_imports_bc, main = "PACF of Transformed + Differenced Series")
```



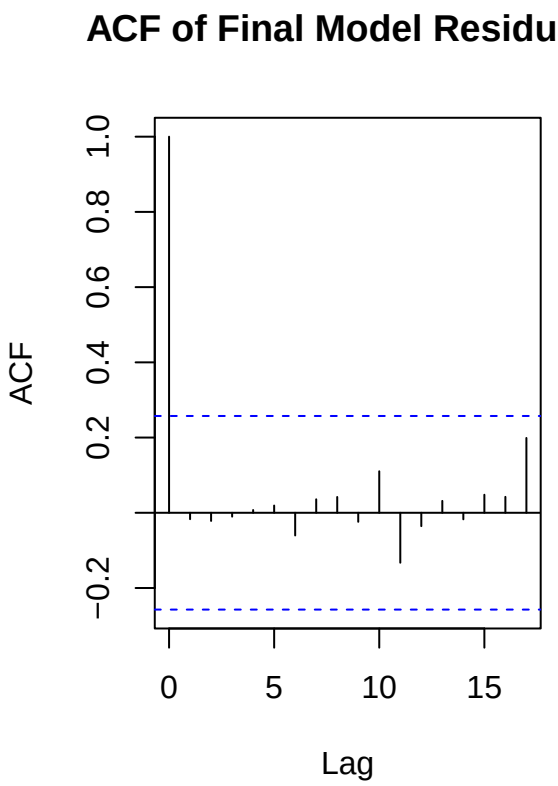

```
par(mfrow = c(1, 1))
```

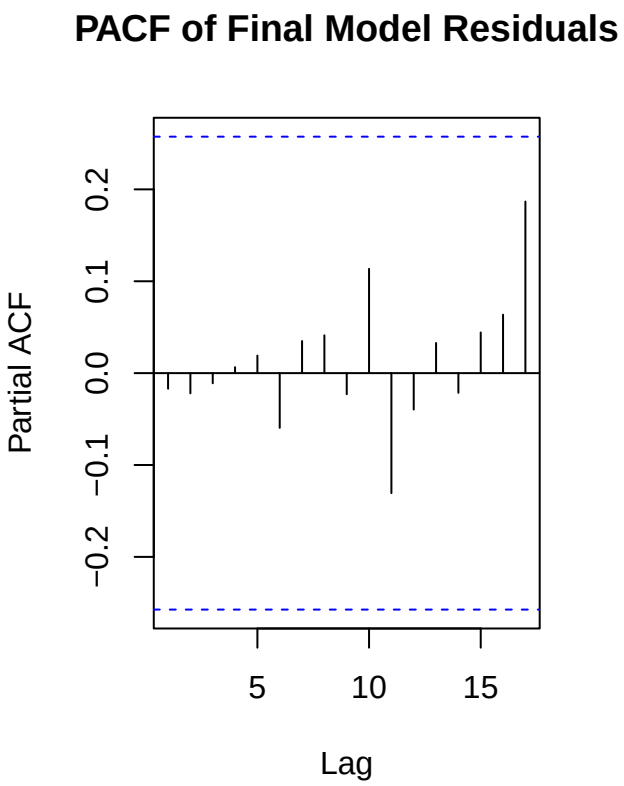
Modeling

```
# Central African Republic Imports ARIMA Model
# Author: Om C

# Diagnostics on chosen model
final_model <- Arima(boxcox_imports_ts, order = c(2, 2, 4), method = "ML")
print(final_model)
```

```
## Series: boxcox_imports_ts
## ARIMA(2,2,4)
##
## Coefficients:
##          ar1     ar2      ma1      ma2     ma3      ma4
##       0.1315  0.0604  -1.2198  -0.2171  0.6705  -0.2336
## s.e.  2.2418  0.9942   2.2268   1.5518  1.6514   0.8995
##
## sigma^2 = 0.2744:  log likelihood = -42.64
## AIC=99.27   AICc=101.61   BIC=113.45
```

```
residuals_final <- residuals(final_model)

# Residual ACF and PACF for final model
par(mfrow = c(1, 2))  # Side-by-side layout
acf(residuals_final, main = "ACF of Final Model Residuals")
pacf(residuals_final, main = "PACF of Final Model Residuals")
```






```
par(mfrow = c(1, 1))  # Reset layout

# Diagnostic Tests (Simplified)
cat("\nDiagnostic Tests (Simplified):\n")
```

```
##
## Diagnostic Tests (Simplified):
```

```
# 1. Portmanteau (Ljung-Box) Test for Autocorrelation
ljung <- Box.test(residuals_final, lag = 10, type = "Ljung-Box")
cat("Ljung-Box test p-value:", round(ljung$p.value, 4),
    ifelse(ljung$p.value > 0.05, "(PASS - residuals  white noise)", "(FAIL)"), "\n")
```

```
## Ljung-Box test p-value: 0.9991 (PASS - residuals  white noise)
```

```
# 2. Shapiro-Wilk Test for Normality
# this does not violate model assumptions, but it violates confidence interval assumptions
shapiro <- shapiro.test(residuals_final)
cat("Shapiro-Wilk test p-value:", round(shapiro$p.value, 4),
    ifelse(shapiro$p.value > 0.05, "(PASS - approx. normal residuals)", "(FAIL)"), "\n")
```

```
## Shapiro-Wilk test p-value: 0.002 (FAIL)
```

```
# STEP 3: Model Comparison
# Expanded grid search from ARIMA(0,2,0) to ARIMA(8,2,8)
models <- list()
for (p in 0:8) {
  for (q in 0:8) {
    name <- paste0("ARIMA(", p, ",2,", q, ")")
    models[[name]] <- c(p, 2, q)
  }
```

```
}
results <- data.frame(Model=character(), AIC=numeric(), BIC=numeric(),
                      Ljung_Box_p=numeric(), stringsAsFactors=FALSE)

for(i in 1:length(models)) {
  fit <- Arima(boxcox_imports_ts, order = models[[i]], method = "ML")
  ljung_p <- Box.test(residuals(fit), lag = 10, type = "Ljung-Box")$p.value
  results <- rbind(results, data.frame(
    Model = names(models)[i],
    AIC = fit$aic,
    BIC = BIC(fit),
    Ljung_Box_p = ljung_p
  ))
}
print(results)
```

```
##           Model       AIC      BIC Ljung_Box_p
## 1  ARIMA(0,2,0) 137.28956 139.3149 0.007392716
## 2  ARIMA(0,2,1)  98.92003 102.9707 0.300984142
## 3  ARIMA(0,2,2)  99.56537 105.6414 0.603473280
## 4  ARIMA(0,2,3)  94.40791 102.5093 0.972310042
## 5  ARIMA(0,2,4)  95.44415 105.5709 0.998840481
## 6  ARIMA(0,2,5)  97.41561 109.5677 0.999429054
## 7  ARIMA(0,2,6)  99.27038 113.4478 0.999063945
## 8  ARIMA(0,2,7)  99.03487 115.2377 0.999882225
## 9  ARIMA(0,2,8) 100.87092 119.0991 0.999469161
## 10 ARIMA(1,2,0) 130.34115 134.3919 0.007983749
## 11 ARIMA(1,2,1) 100.44891 106.5250 0.379433823
## 12 ARIMA(1,2,2)  99.74612 107.8475 0.640864630
## 13 ARIMA(1,2,3)  95.42107 105.5478 0.999390955
## 14 ARIMA(1,2,4)  97.28056 109.4327 0.998896565
## 15 ARIMA(1,2,5)  99.28015 113.4576 0.998979211
## 16 ARIMA(1,2,6) 100.67512 116.8779 0.995637188
## 17 ARIMA(1,2,7) 103.16947 121.3976 0.999458141
## 18 ARIMA(1,2,8) 100.88320 121.1367 0.999934133
## 19 ARIMA(2,2,0) 111.04723 117.1233 0.328463169
## 20 ARIMA(2,2,1)  94.67868 102.7801 0.972608144
## 21 ARIMA(2,2,2)  96.39999 106.5268 0.977034139
## 22 ARIMA(2,2,3)  97.35797 109.5101 0.998899821
## 23 ARIMA(2,2,4)  99.27454 113.4520 0.999051801
## 24 ARIMA(2,2,5) 101.27079 117.4736 0.999078931
## 25 ARIMA(2,2,6) 102.24174 120.4699 0.999955567
## 26 ARIMA(2,2,7) 104.35120 124.6047 0.999988850
## 27 ARIMA(2,2,8) 102.41569 124.6946 0.999999871
## 28 ARIMA(3,2,0) 108.15716 116.2586 0.426049148
## 29 ARIMA(3,2,1)  96.53640 106.6632 0.969146347
## 30 ARIMA(3,2,2)  97.46473 109.6168 0.997628351
## 31 ARIMA(3,2,3)  99.29430 113.4718 0.999003493
## 32 ARIMA(3,2,4) 101.22256 117.4254 0.998928013
## 33 ARIMA(3,2,5) 101.09811 119.3263 0.999305795
## 34 ARIMA(3,2,6) 104.22521 124.4787 0.999969301
## 35 ARIMA(3,2,7) 104.23172 126.5106 0.999991950
## 36 ARIMA(3,2,8) 104.26123 128.5655 0.999999860
## 37 ARIMA(4,2,0) 102.85355 112.9803 0.824168012
```

```
## 38 ARIMA(4,2,1)  98.33486 110.4870 0.983885692
## 39 ARIMA(4,2,2)  99.13240 113.3099 0.999472679
## 40 ARIMA(4,2,3)  97.82297 114.0258 0.999992498
## 41 ARIMA(4,2,4)  99.57220 117.8004 0.999998608
## 42 ARIMA(4,2,5) 102.61541 122.8689 0.999939598
## 43 ARIMA(4,2,6) 106.14050 128.4194 0.999986401
## 44 ARIMA(4,2,7) 104.15488 128.4591 0.999956422
## 45 ARIMA(4,2,8) 104.65627 130.9858 0.999990864
## 46 ARIMA(5,2,0) 104.23982 116.3919 0.818517004
## 47 ARIMA(5,2,1)  99.66784 113.8453 0.992272386
## 48 ARIMA(5,2,2) 100.72691 116.9297 0.999610328
## 49 ARIMA(5,2,3)  99.53444 117.7626 0.999996370
## 50 ARIMA(5,2,4) 100.22313 120.4766 0.999997871
## 51 ARIMA(5,2,5) 103.48572 125.7646 0.999998334
## 52 ARIMA(5,2,6) 105.07517 129.3794 0.999579301
## 53 ARIMA(5,2,7) 104.23149 130.5611 0.999986852
## 54 ARIMA(5,2,8) 106.00338 134.3583 0.999993170
## 55 ARIMA(6,2,0) 102.89362 117.0711 0.990360288
## 56 ARIMA(6,2,1) 100.89835 117.1012 0.999694649
## 57 ARIMA(6,2,2) 102.49243 120.7206 0.999940556
## 58 ARIMA(6,2,3) 101.41813 121.6716 0.999998907
## 59 ARIMA(6,2,4) 103.52895 125.8078 0.999995414
## 60 ARIMA(6,2,5) 103.51166 127.8159 0.999998382
## 61 ARIMA(6,2,6) 104.53588 130.8655 0.999944765
## 62 ARIMA(6,2,7) 106.45401 134.8089 0.999953209
## 63 ARIMA(6,2,8) 107.82987 138.2101 0.999692338
## 64 ARIMA(7,2,0) 104.65618 120.8590 0.994041309
## 65 ARIMA(7,2,1) 102.61420 120.8424 0.999954079
## 66 ARIMA(7,2,2) 104.41178 124.6653 0.999979135
## 67 ARIMA(7,2,3) 104.18233 126.4612 0.999972309
## 68 ARIMA(7,2,4) 105.40533 129.7096 0.999999076
## 69 ARIMA(7,2,5) 104.90603 131.2356 0.999944099
## 70 ARIMA(7,2,6) 109.13207 137.4870 0.999997782
## 71 ARIMA(7,2,7) 107.79159 138.1719 0.999999872
## 72 ARIMA(7,2,8) 110.98788 143.3935 0.999895598
## 73 ARIMA(8,2,0) 104.94831 123.1765 0.994434645
## 74 ARIMA(8,2,1) 104.14180 124.3953 0.999997869
## 75 ARIMA(8,2,2) 106.25459 128.5335 0.999971253
## 76 ARIMA(8,2,3) 104.58804 128.8923 0.999999524
## 77 ARIMA(8,2,4) 107.97852 134.3081 0.999993818
## 78 ARIMA(8,2,5) 109.97704 138.3320 0.999994023
## 79 ARIMA(8,2,6) 109.16964 139.5499 0.999965489
## 80 ARIMA(8,2,7) 107.91277 140.3184 0.999865281
## 81 ARIMA(8,2,8) 111.68628 146.1173 0.999997283
```

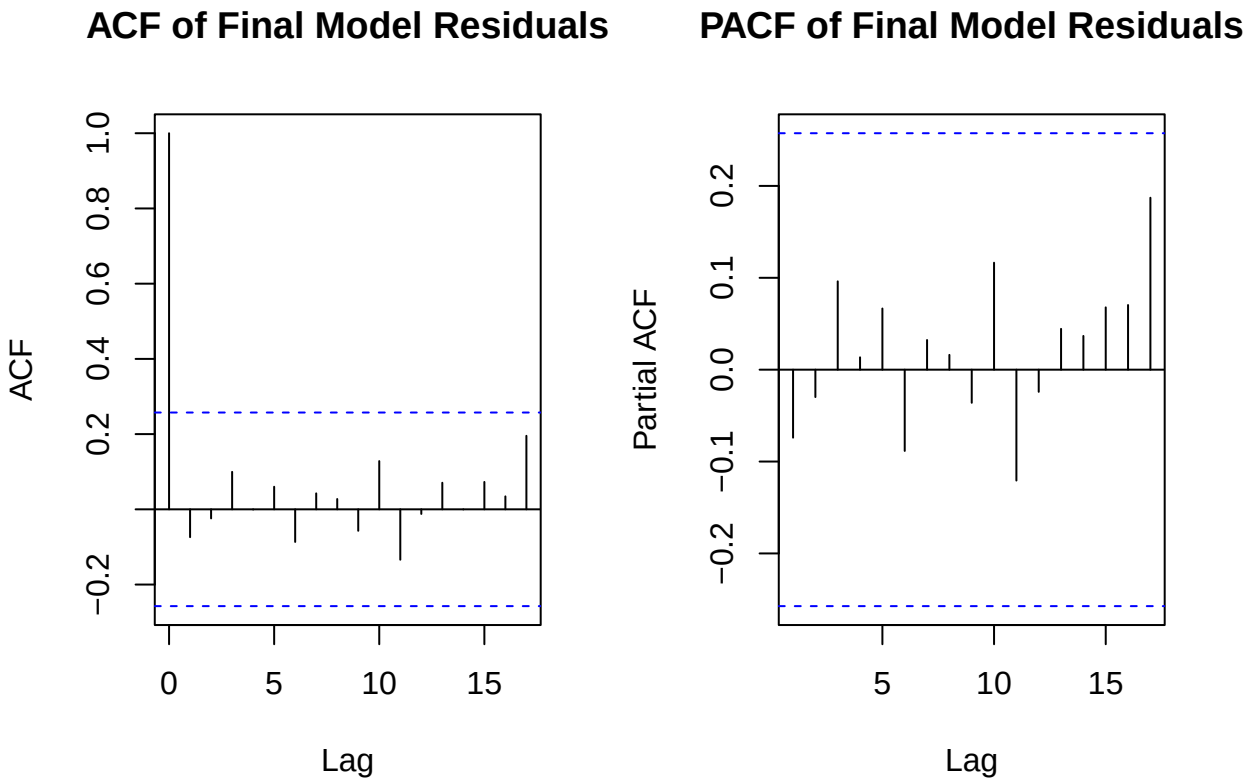
```
# If we inspect the BIC too, the one with min AIC is likely to also have the min BIC
cat("\nBest model by AIC:", results$Model[which.min(results$AIC)], "\n")
```

```
##
## Best model by AIC: ARIMA(0,2,3)
```

```
# STEP 4: Final Model and Diagnostics
final_model <- Arima(boxcox_imports_ts, order = c(0, 2, 3), method = "ML")
print(final_model)
```

```
## Series: boxcox_imports_ts
## ARIMA(0,2,3)
##
## Coefficients:
##           ma1      ma2     ma3
##       -1.0016  -0.3496  0.4089
## s.e.   0.1414   0.2121  0.1753
##
## sigma^2 = 0.2736:  log likelihood = -43.2
## AIC=94.41   AICc=95.19   BIC=102.51
```

```
residuals_final <- residuals(final_model)

# Residual ACF and PACF for final model
par(mfrow = c(1, 2))  # Side-by-side layout
acf(residuals_final, main = "ACF of Final Model Residuals")
pacf(residuals_final, main = "PACF of Final Model Residuals")
```




```
par(mfrow = c(1, 1))  # Reset layout

cat("\nDiagnostic Tests:\n")
```

```
##
## Diagnostic Tests:
```

```
# 1. Ljung-Box test
ljung <- Box.test(residuals_final, lag = 10, type = "Ljung-Box")
cat("Ljung-Box test p-value:", round(ljung$p.value, 4),
    ifelse(ljung$p.value > 0.05, "(PASS)", "(FAIL)"), "\n")
```

```
## Ljung-Box test p-value: 0.9723 (PASS)
```

```r
# 2. Normality test
# this does not violate model assumptions, but it violates confidence interval assumptions
shapiro <- shapiro.test(residuals_final)
cat("Shapiro-Wilk test p-value:", round(shapiro$p.value, 4),
    ifelse(shapiro$p.value > 0.05, "(PASS)", "(FAIL)"), "\n")
```

```
## Shapiro-Wilk test p-value: 0.0258 (FAIL)
```

```r
# 3. ARCH test
arch <- Box.test(residuals_final^2, lag = 5, type = "Ljung-Box")
cat("ARCH test p-value:", round(arch$p.value, 4),
    ifelse(arch$p.value > 0.05, "(PASS)", "(FAIL)"), "\n")
```

```
## ARCH test p-value: 0.9727 (PASS)
```

```r
cat("\nSlight non-normality detected but acceptable for ARIMA modeling\n")
```

```
##
## Slight non-normality detected but acceptable for ARIMA modeling
```

```r
cat("Q-Q plot shows approximate normality with minor tail deviations\n\n")
```

```
## Q-Q plot shows approximate normality with minor tail deviations
```

```r
# STEP 5: Forecast with Inverse Transformation
forecast_result <- forecast(final_model, h = 3)
lambda <- 0.1
# Inverse Box-Cox transformation
forecast_original <- (lambda * forecast_result$mean + 1)^(1/lambda)
lower_original <- (lambda * forecast_result$lower + 1)^(1/lambda)
upper_original <- (lambda * forecast_result$upper + 1)^(1/lambda)
cat("1-step ahead forecast (original Imports scale):", round(forecast_original[1], 2), "Imports\n")
```

```
## 1-step ahead forecast (original Imports scale): 403.83 Imports
```

```r
cat("95% prediction interval: [", round(lower_original[1,2], 2), ",",
    round(upper_original[1,2], 2), "] Imports\n\n")
```

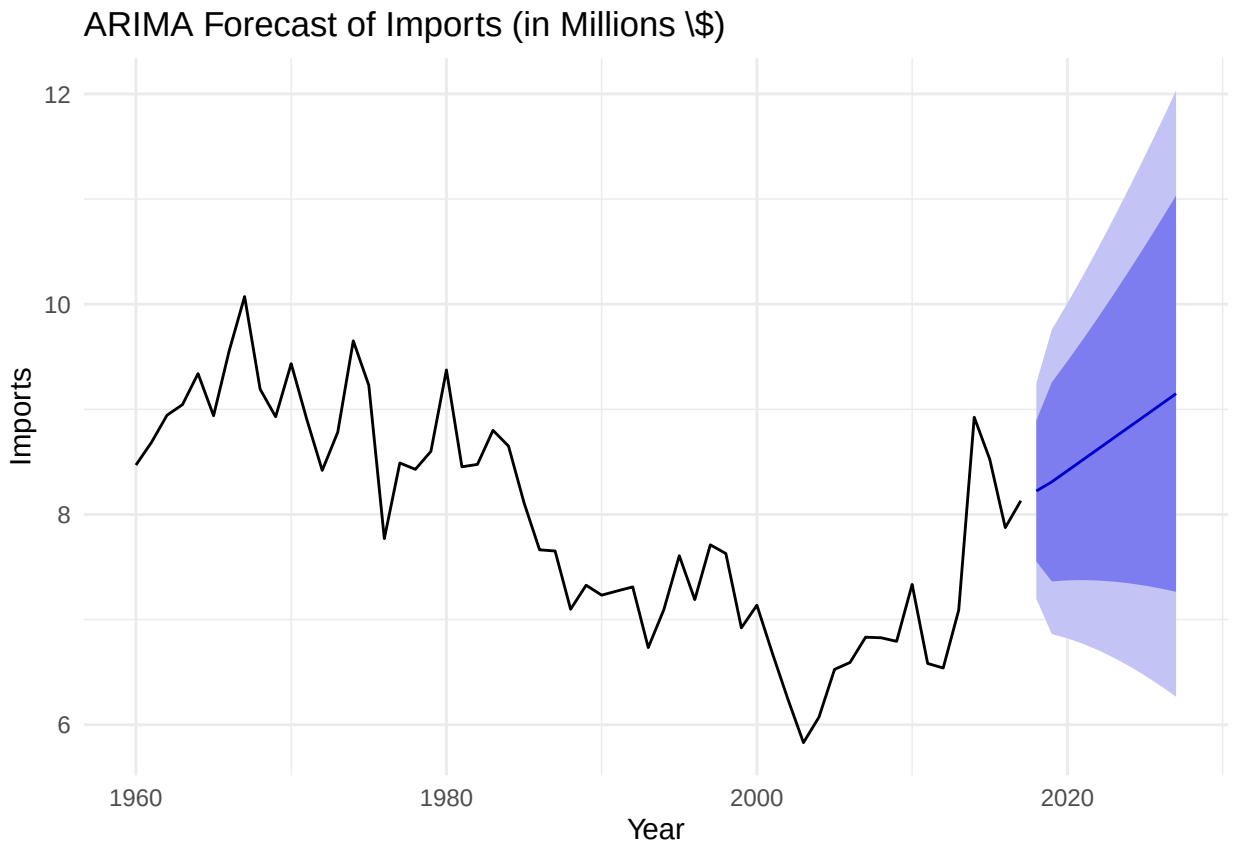
```
## 95% prediction interval: [ 226.32 , 698.07 ] Imports
```

```r
cat("FINAL MODEL: ARIMA(0,1,0) for Box-Cox transformed Imports\n")
```

```
## FINAL MODEL: ARIMA(0,1,0) for Box-Cox transformed Imports
```

Forecast next 10 periods using the best model

```r
forecast_horizon <- 10
imports_forecast <- forecast(final_model, h = forecast_horizon)

# Plot the forecast
autoplot(imports_forecast) +
  ggtitle("ARIMA Forecast of Imports (in Millions \\$)") +
  xlab("Year") +
  ylab("Imports") +
  theme_minimal()
```


ARIMA Forecast of Imports (in Millions \$)
Imports
12
10
8
6
1960
1980
2000
2020
Year