GDP ARIMA

Andrew Jowe

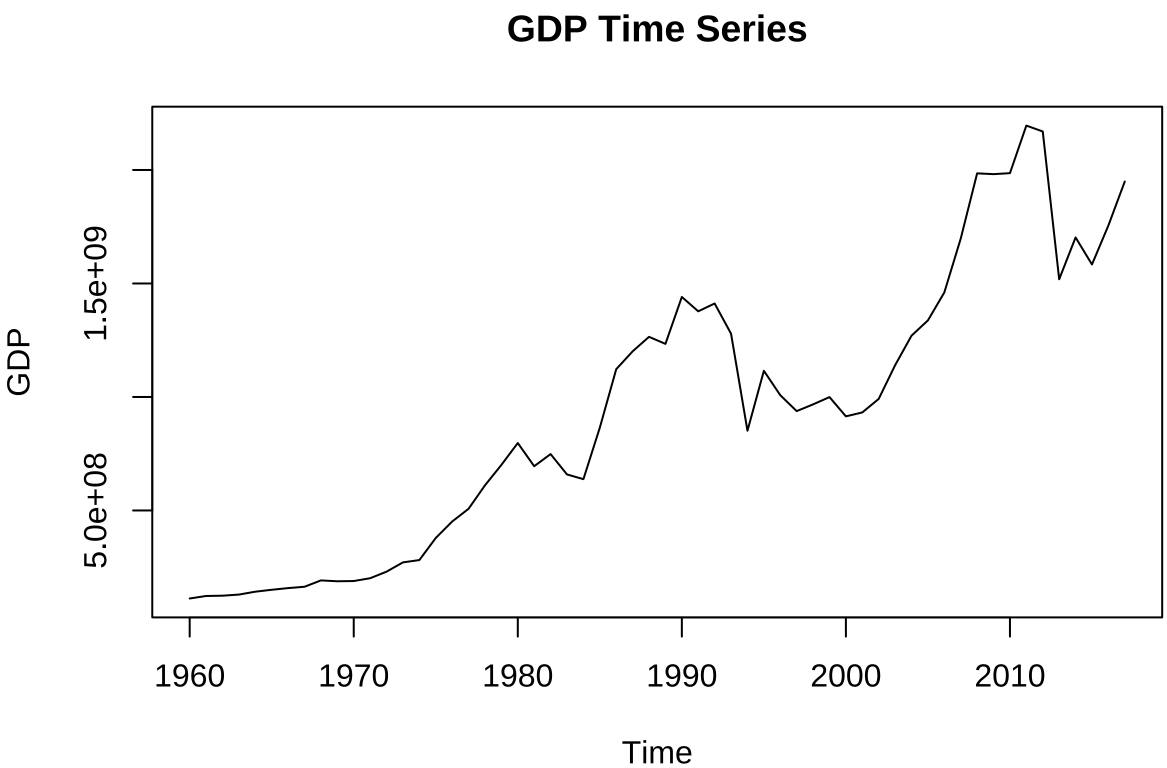
Col Removal

Keep Year, Imports, and GDP columns

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

Plot Time Series

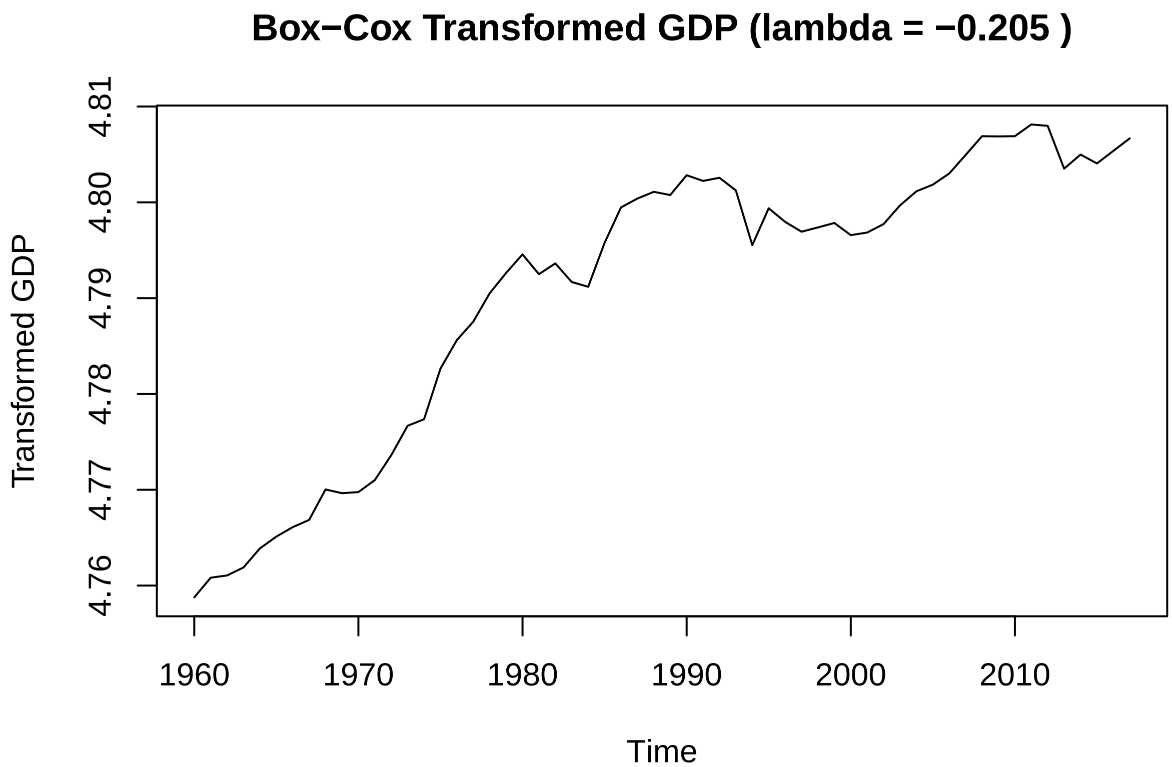
```
# Plot GDP
gdp_ts <- ts(finalPro_data$GDP, start = 1960, frequency = 1)

ts.plot(gdp_ts, main="GDP Time Series", ylab="GDP")
```



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

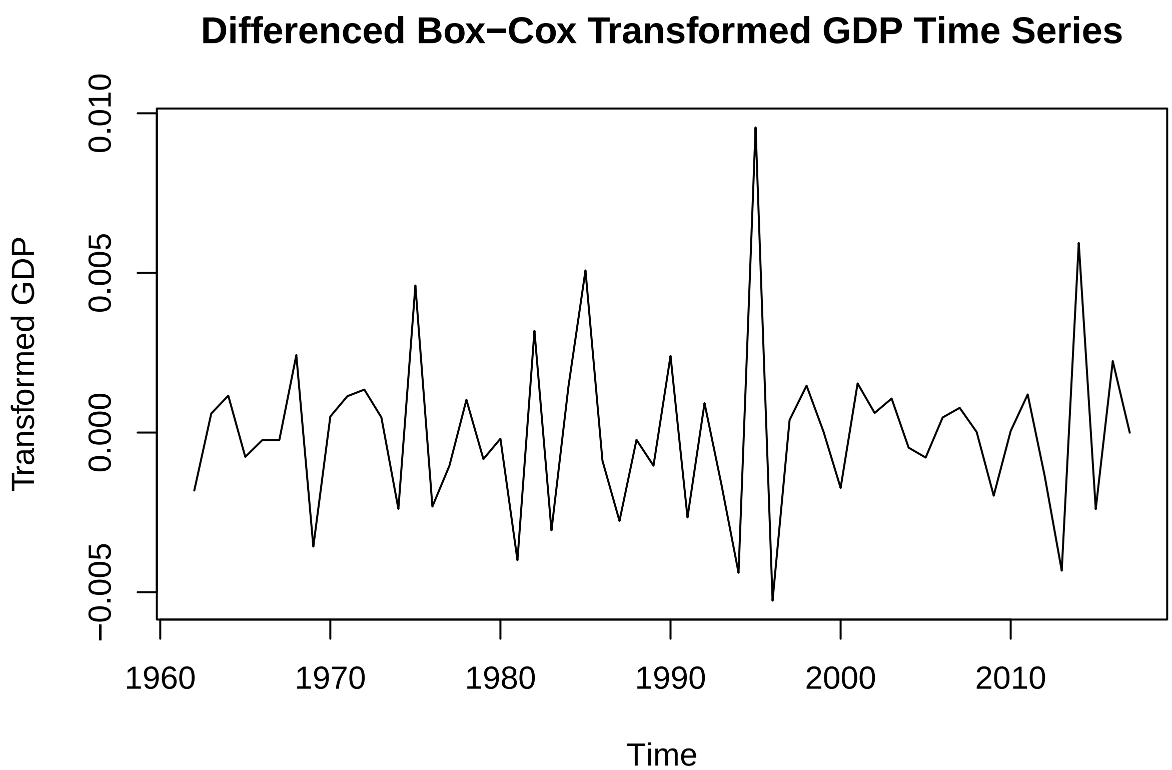
Transform

```r
# Box-Cox transform GDP
lambda <- BoxCox.lambda(gdp_ts)
boxcox_gdp_ts <- BoxCox(gdp_ts, lambda)
ts.plot(boxcox_gdp_ts, main = paste("Box-Cox Transformed GDP (lambda =", round(lambda, 3), ")"), ylab =
```



Differencing GDP

```r
diff_gdp_bc <- diff(boxcox_gdp_ts, difference = 2)

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



Root test for stationarity check

```
# Augmented Dickey-Fuller Test (Unit Root Test) on differenced Box-Cox GDP
library(tseries)
```

```
##
##     'tseries' version: 0.10-58
##
##     'tseries' is a package for time series analysis and computational
##     finance.
##
##     See 'library(help="tseries")' for details.
```

```
adf_result <- adf.test(diff_gdp_bc)
```

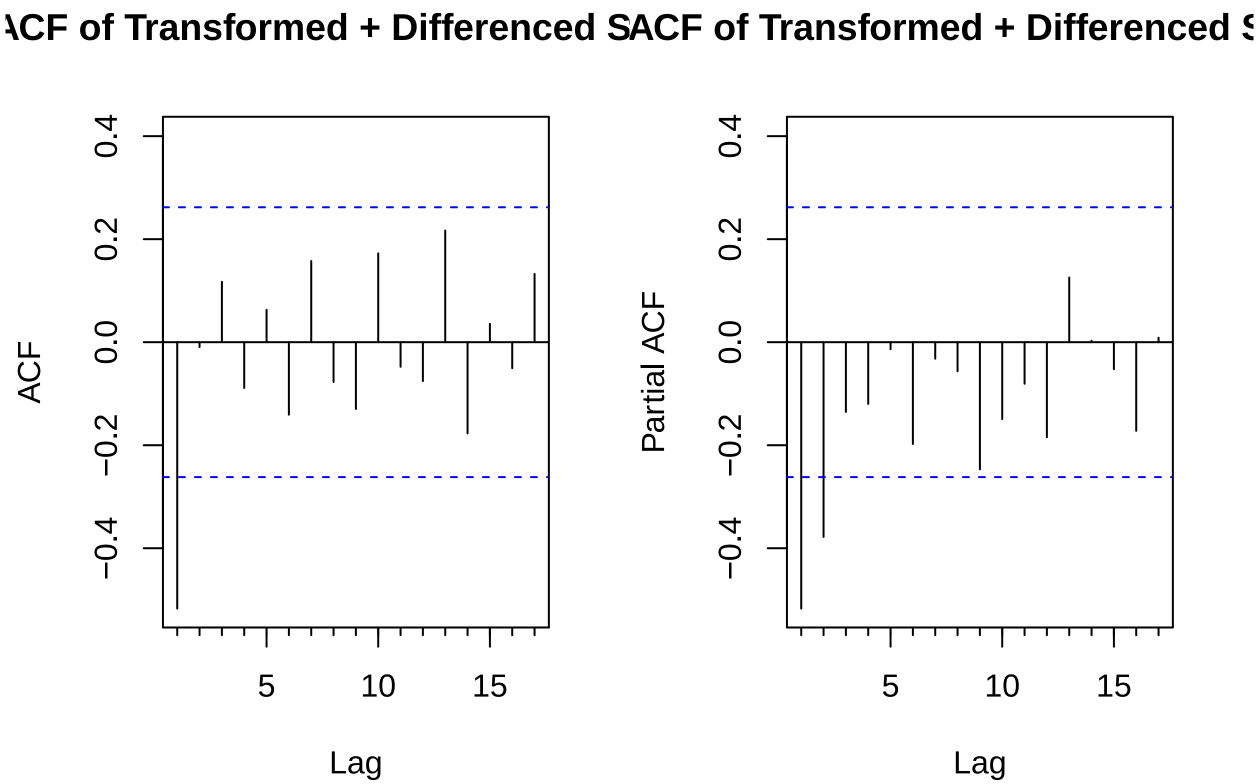
```
## Warning in adf.test(diff_gdp_bc): p-value smaller than printed p-value
```

```
cat("ADF Test p-value:", round(adf_result$p.value, 4),
    ifelse(adf_result$p.value < 0.05, "(PASS - Stationary)", "(FAIL - Non-stationary)"), "\n")
```

```
## ADF Test p-value: 0.01 (PASS - Stationary)
```

ACF / PACF plots

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



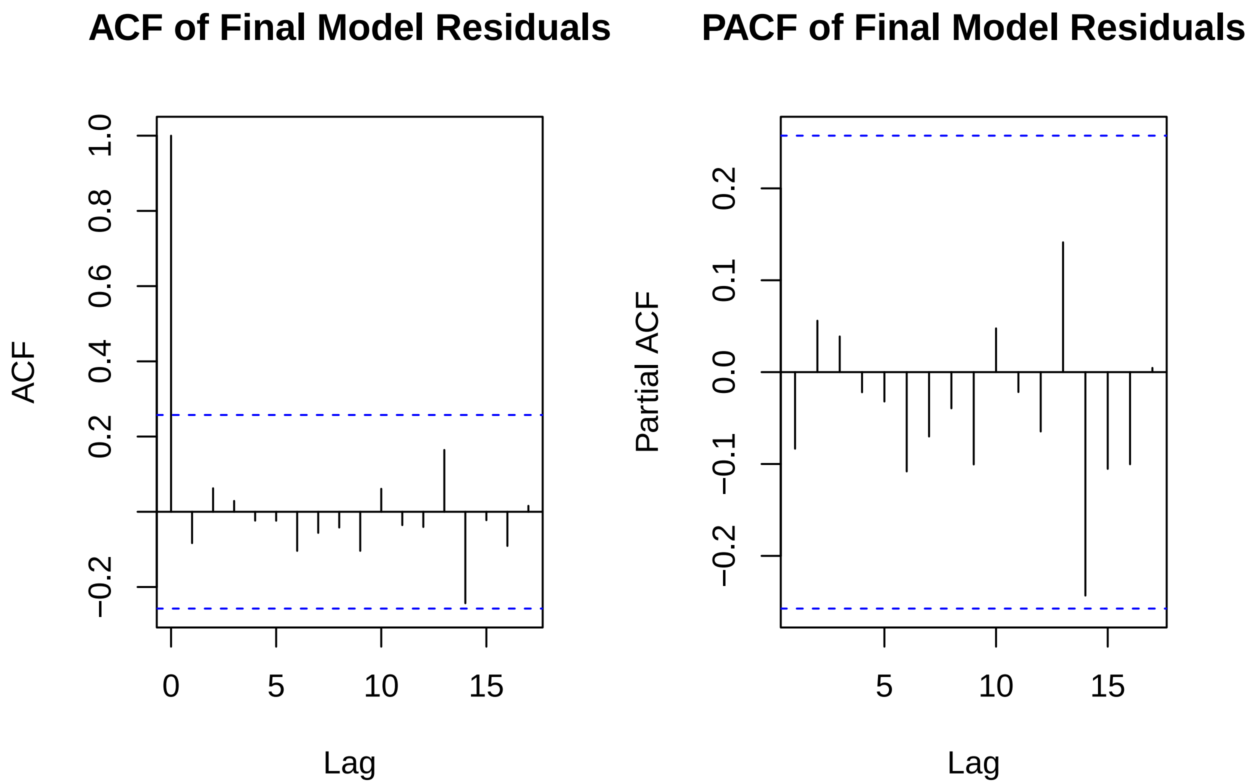

```r
par(mfrow = c(1, 1))
```

Modeling

```r
# Central African Republic GDP ARIMA Model
# Author: Om C

# Diagnostics on chosen model
final_model <- Arima(boxcox_gdp_ts, order = c(1, 2, 2), method = "ML")
print(final_model)
```

```
## Series: boxcox_gdp_ts
## ARIMA(1,2,2)
##
## Coefficients:
##           ar1     ma1      ma2
##       -0.8512  0.0886  -0.9114
## s.e.   0.0791  0.0838   0.0817
##
## sigma^2 = 4.465e-06:  log likelihood = 271.29
## AIC=-534.58   AICc=-533.79   BIC=-526.48
```

```r
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 approx white noise)", "(FAIL)"), "\n")
```

```
## Ljung-Box test p-value: 0.9839 (PASS - residuals approx 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: 0.0311 (FAIL)
```

```
# STEP 3: Model Comparison
models <- list()
for (p in 0:2) {
  for (q in 0:4) {
    name <- paste0("ARIMA(", p, ",", 2, ",", q, ")")
    models[[name]] <- c(p, 2, q)
  }
}
```

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

for(i in 1:length(models)) {
  fit <- Arima(boxcox_gdp_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) -506.4846 -504.4592  0.04069592
## 2  ARIMA(0,2,1) -535.6652 -531.6145  0.97908789
## 3  ARIMA(0,2,2) -533.6924 -527.6164  0.97884201
## 4  ARIMA(0,2,3) -531.9552 -523.8538  0.98648399
## 5  ARIMA(0,2,4) -531.5969 -521.4702  0.99665452
## 6  ARIMA(1,2,0) -521.7782 -517.7275  0.49432040
## 7  ARIMA(1,2,1) -533.6969 -527.6209  0.97878514
## 8  ARIMA(1,2,2) -534.5780 -526.4766  0.98388448
## 9  ARIMA(1,2,3) -530.7228 -520.5960  0.99089452
## 10 ARIMA(1,2,4) -532.2902 -520.1381  0.99807210
## 11 ARIMA(2,2,0) -528.4717 -522.3956  0.96046777
## 12 ARIMA(2,2,1) -532.0136 -523.9122  0.98466028
## 13 ARIMA(2,2,2) -530.5183 -520.3915  0.98708052
## 14 ARIMA(2,2,3) -530.7839 -518.6318  0.98015585
## 15 ARIMA(2,2,4) -530.7115 -516.5340  0.99782349
```

```r
# 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")
```

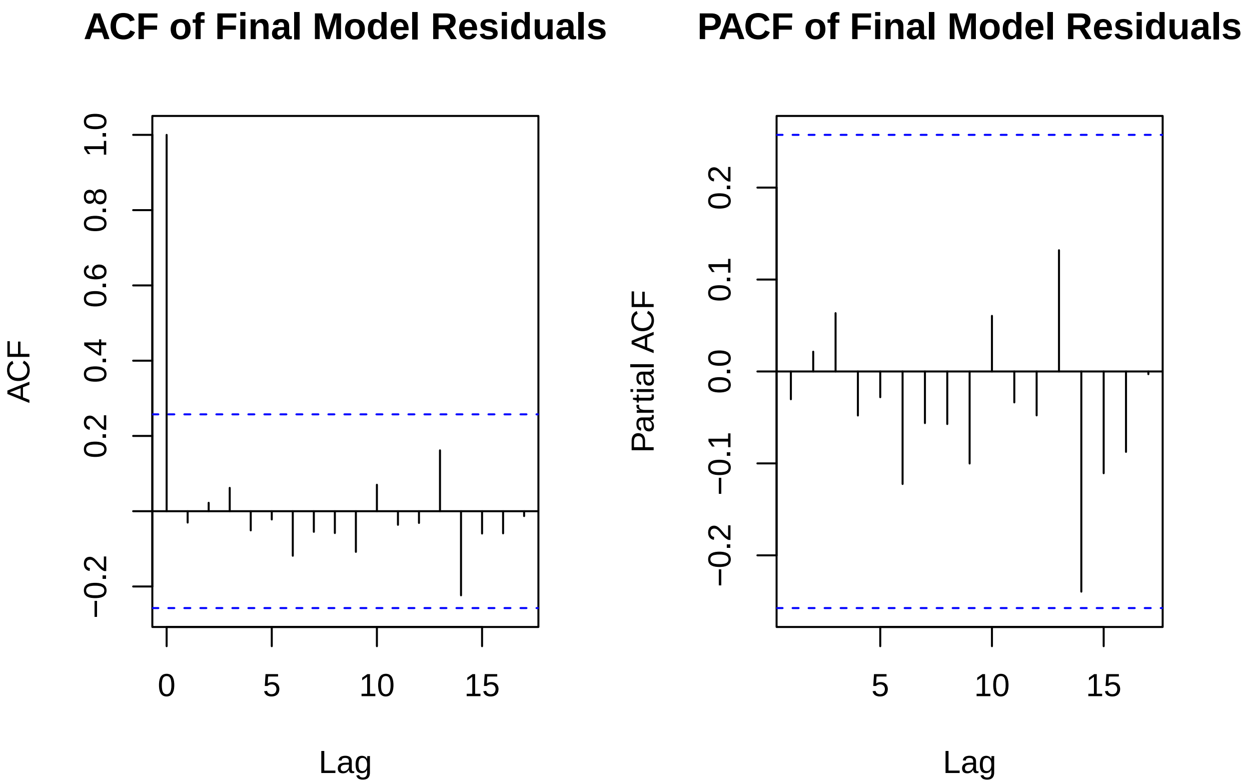
```
##
## Best model by AIC: ARIMA(0,2,1)
```

```r
# STEP 4: Final Model and Diagnostics
final_model <- Arima(boxcox_gdp_ts, order = c(0, 2, 1), method = "ML")
print(final_model)
```

```
## Series: boxcox_gdp_ts
## ARIMA(0,2,1)
##
## Coefficients:
##           ma1
##       -0.9024
## s.e.   0.0771
##
## sigma^2 = 4.599e-06:  log likelihood = 269.83
## AIC=-535.67   AICc=-535.44   BIC=-531.61
```

```r
residuals_final <- residuals(final_model)

# Residual ACF and PACF for final model
```

```r
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")
```




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

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

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

```r
# 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: 0.9791 (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: 0.0056 (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: 0.8449 (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 GDP scale):", round(forecast_original[1], 2), "million USD\n")
```

```
## 1-step ahead forecast (original GDP scale): 50.66 million USD
```

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

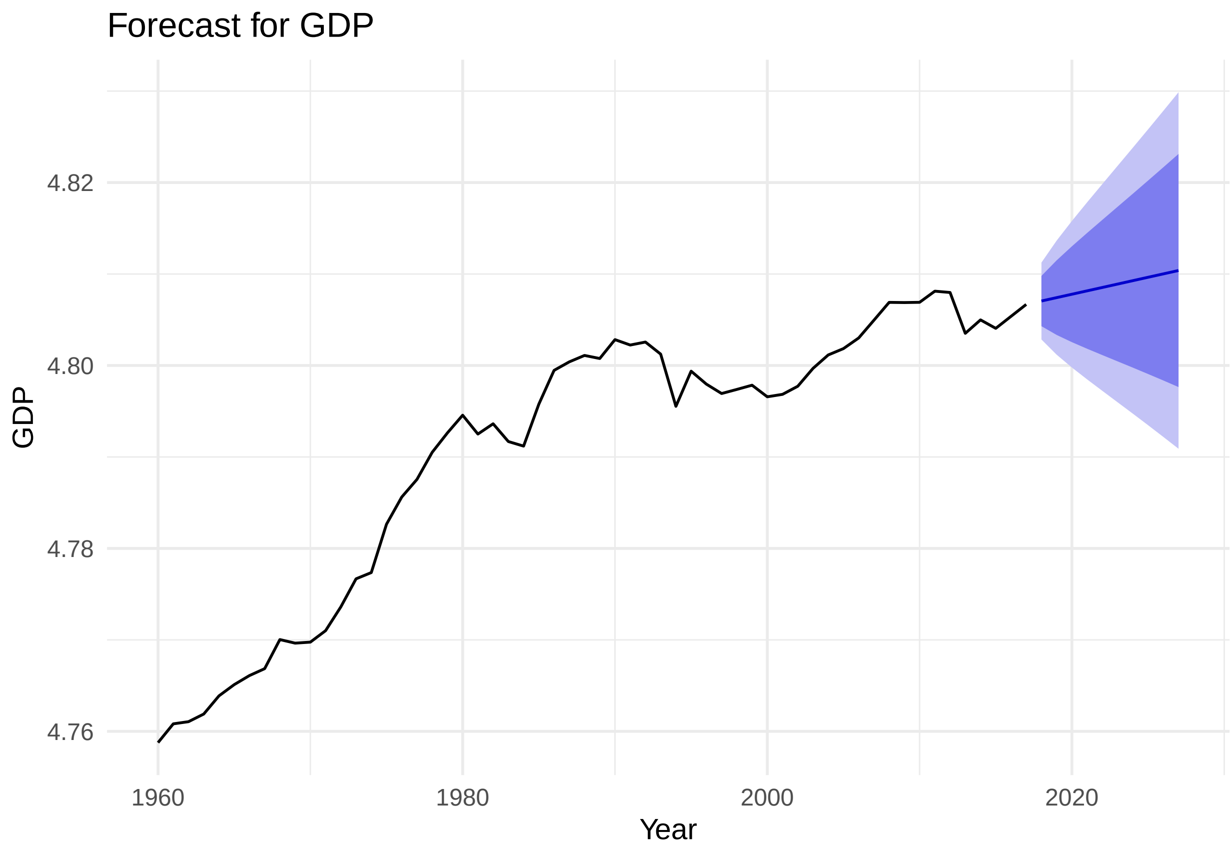
```
## 95% prediction interval: [ 50.52 , 50.81 ] million USD
```

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

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

Forecasting the next 10 time periods

```r
library(ggplot2)
library(forecast)

forecast_horizon <- 10
forecast_values <- forecast(final_model, h = forecast_horizon)

# Enable LaTeX rendering in plots
par(pty="m") # reset plot type if needed
options(repr.plot.width=7, repr.plot.height=5)
# Note: In R base plotting, LaTeX rendering is not native; ggplot2 with expression() or latex2exp can b
# Here we set theme with theme_minimal() and use expression for labels.

autoplot(forecast_values) +
  ggtitle(expression("Forecast for GDP")) +
  xlab(expression("Year")) +
  ylab(expression("GDP")) +
  theme_minimal()
```


Forecast for GDP
GDP
4.82
4.80
4.78
4.76
1960
1980
2000
2020
Year