Predictive Analytics - Assignment 1

Elena Del Governatore (167318)
Antonia Strobl (167298)
Nicola Friedrich (167278)
Franka Johne (167334)

Contents

Import Packages

```
# Load libraries
library(AER)
```

```
## Loading required package: car
```

```
## Loading required package: carData
```

```
## Loading required package: lmtest
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sandwich
```

```
## Loading required package: survival
```

```
library(ggplot2)
library(tsibble) # the tsibble package, which is used for handling time series data
```

```
##
## Attaching package: 'tsibble'
```

```
## The following object is masked from 'package:zoo':
##
##     index
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, union
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:car':
##
##     recode
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(fpp3)
```

```
## -- Attaching packages --------------------------------------------- fpp3 0.5 --
```

```
## v tibble      3.2.1     v feasts      0.3.1
## v tidyr       1.3.0     v fable       0.3.3
## v lubridate   1.9.3     v fabletools  0.4.1
## v tsibbledata 0.4.1
```

```
## -- Conflicts ------------------------------------------------ fpp3_conflicts --
## x lubridate::date()     masks base::date()
## x dplyr::filter()       masks stats::filter()
## x tsibble::index()      masks zoo::index()
## x tsibble::intersect()  masks base::intersect()
## x lubridate::interval() masks tsibble::interval()
## x dplyr::lag()          masks stats::lag()
## x dplyr::recode()       masks car::recode()
## x tsibble::setdiff()    masks base::setdiff()
## x tsibble::union()      masks base::union()
```

```
library(forecast)
```

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

```
library(tidyverse,warn.conflicts=FALSE)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
```

```
## v forcats 1.0.0     v readr   2.1.4
## v purrr   1.0.2     v stringr 1.5.1
```

```
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter()       masks stats::filter()
## x lubridate::interval() masks tsibble::interval()
## x dplyr::lag()          masks stats::lag()
## x dplyr::recode()       masks car::recode()
## x purrr::some()         masks car::some()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to bec
```

Question 1

Create two plots, one for the GDP series and one for its autocorrelation. What are the relevant features of the data? Can you confirm them from the autocorrelation function? Would it make sense to transform the data? Why?

Load Data The data USMacroG, that will be used for Question 1 and Question 2, is part of the AER package. The dataset contains the following 12 time series that have quarterly frequency:

- Real gross domestic product (in billion USD),
- Real consumption expenditures,
- Real investment by private sector,
- Real government expenditures,
- Real disposable personal income,
- Consumer price index,
- Nominal money stock,
- Quarterly average of month end 90 day treasury bill rate,
- Unemployment rate, Population (in million),
- Inflation rate,
- Ex post real interest rate (computed as treasury bill rate - inflation).

```r
# The data used is part of the AER package
data('USMacroG',package = 'AER')

# Converts USMacroG into a tsibble object
USMacro <- USMacroG %>%
  as_tsibble(pivot_longer = FALSE)

# Convert to a data frame
USMacro_df <- as.data.frame(USMacroG)
```

First six rows of the dataset:

```r
head(USMacro_df)
```

```
##      gdp consumption invest government    dpi  cpi     m1 tbill unemp
## 1 1610.5      1058.9  198.1      361.0 1186.1 70.6 110.20  1.12   6.4
## 2 1658.8      1075.9  220.4      366.4 1178.1 71.4 111.75  1.17   5.6
## 3 1723.0      1131.0  239.7      359.6 1196.5 73.2 112.95  1.23   4.6
## 4 1753.9      1097.6  271.8      382.5 1210.0 74.9 113.93  1.35   4.2
## 5 1773.5      1122.8  242.9      421.9 1207.9 77.3 115.08  1.40   3.5
## 6 1803.7      1091.4  249.2      480.1 1225.8 77.6 116.19  1.53   3.1
##   population inflation interest
## 1    149.461        NA       NA
## 2    150.260    4.5071  -3.3404
## 3    151.064    9.9590  -8.7290
## 4    151.871    9.1834  -7.8301
## 5    152.393   12.6160 -11.2160
## 6    152.917    1.5494  -0.0161
```

Summary statistics of the variables in the dataset:

```r
summary(USMacro_df)
```

```
##       gdp        consumption       invest         government          dpi
##  Min.   :1610   Min.   :1059   Min.   : 197.7   Min.   : 359.6   Min.   :1178
##  1st Qu.:2602   1st Qu.:1640   1st Qu.: 309.3   1st Qu.: 740.6   1st Qu.:1822
##  Median :4142   Median :2715   Median : 568.5   Median : 952.0   Median :3133
##  Mean   :4563   Mean   :2999   Mean   : 652.3   Mean   : 997.0   Mean   :3341
##  3rd Qu.:6294   3rd Qu.:4235   3rd Qu.: 874.1   3rd Qu.:1300.8   3rd Qu.:4733
##  Max.   :9304   Max.   :6341   Max.   :1801.6   Max.   :1582.8   Max.   :6635
##
##       cpi               m1             tbill            unemp
##  Min.   : 70.60   Min.   : 110.2   Min.   : 0.810   Min.   : 2.600
##  1st Qu.: 91.15   1st Qu.: 147.5   1st Qu.: 3.087   1st Qu.: 4.400
##  Median :162.10   Median : 284.4   Median : 5.045   Median : 5.600
##  Mean   :225.82   Mean   : 453.9   Mean   : 5.229   Mean   : 5.675
##  3rd Qu.:350.12   3rd Qu.: 764.3   3rd Qu.: 6.645   3rd Qu.: 6.800
##  Max.   :521.10   Max.   :1152.1   Max.   :15.090   Max.   :10.700
##
##    population      inflation         interest
##  Min.   :149.5   Min.   :-2.530   Min.   :-11.2160
##  1st Qu.:185.6   1st Qu.: 1.762   1st Qu.: -0.1583
##  Median :214.7   Median : 3.138   Median :  1.5133
##  Mean   :214.0   Mean   : 3.939   Mean   :  1.3112
##  3rd Qu.:243.0   3rd Qu.: 5.591   3rd Qu.:  2.9155
##  Max.   :281.4   Max.   :16.864   Max.   : 10.6262
##                  NA's   :1        NA's   :1
```

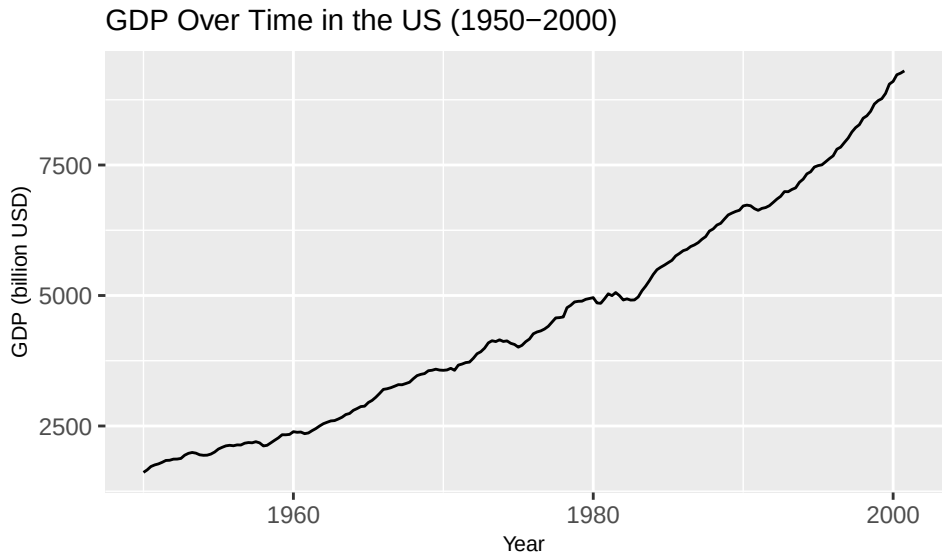
```r
# Set the first index to 1950 and the last to 2001
start_date <- as.Date("1950-01-01")
```

```r
end_date <- as.Date("2000-12-31")
dates <- seq(start_date, end_date, by = "quarter")

# Create a plot for the GDP series
gdp_series_plot <- ggplot(USMacro_df, aes(x = dates, y = gdp)) +
  geom_line() + #Add points for a scatterplot
  labs(title = "GDP Over Time in the US (1950-2000)",
       x = "Year",
       y = "GDP (billion USD)")+
  theme(plot.title = element_text(size = 11),  # Adjust the plot title font size
        axis.title = element_text(size = 8))

print(gdp_series_plot)
```

GDP Series and Autocorrelation

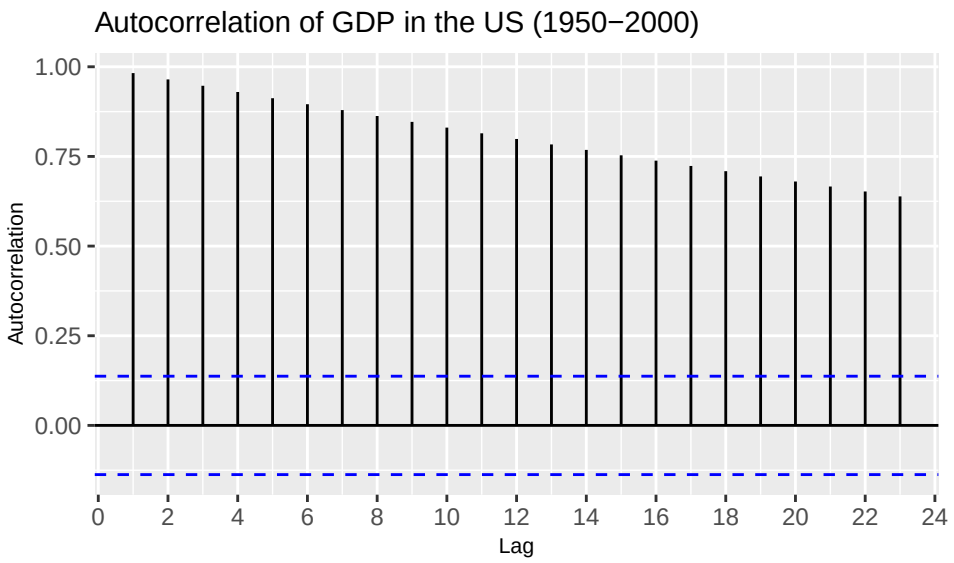


The time series plot of GDP over time shows a clear upward trend, indicating an increase in GDP values over the years. The trend is quite strong, suggesting exponential growth, which is typical for economic data over the long term.

There are no immediately obvious periodic fluctuations that would suggest seasonality. Moroever, it's difficult to infer cyclic behavior from this plot alone.

```r
# Plot the autocorrelation function
gdp_acf_plot <- USMacro %>%
  ACF(gdp) %>%
  autoplot() +
  labs(title = "Autocorrelation of GDP in the US (1950-2000)",
       x = "Lag",
       y = "Autocorrelation") +
  theme(plot.title = element_text(size = 11),  # Adjust the plot title font size
        axis.title = element_text(size = 8))
```

```r
print(gdp_acf_plot)
```



The ACF plot indicates significant positive autocorrelation at many lag intervals. All spikes surpass the blue dotted line, with the autocorrelation peaking at 1, signifying substantial autocorrelation. This high autocorrelation underscores the considerable influence of previous years on subsequent data points.

The slow decay of the autocorrelation as lag increases suggests a non-stationary time series. There is no clear pattern of spikes at regular intervals in the ACF plot that would suggest seasonality. Similarly, the absence of cycles in the ACF plot means we cannot confirm cyclic behavior from this function alone.

However having data with very high correlation is not optimal as the trend is so dominant that it is hard to see anything else. This means that some of the explanatory variables and valuable information for the analysis would be obscure. Therefore, removing the trend is highly recommended to explore signfiicant features. This could be done by transforming data. Common transformations for trend reduction include differencing, logarithmic transformation, or Box-Cox Transformation.

Question 2

Transform the series to create a series for the growth rate of GDP in US quarter on quarter. Plot the growth series and its autocorrelation function and comment it. Are the data still trending? Do you see any sign of seasonality or cycle? Perform a statistical test to verify whether data are autocorrelated or not.

```r
# Create tsibble from dates and gdp columns
USMacro_ts <- tibble::tibble(
  dates = dates,
  gdp = USMacro_df$gdp
) %>%
  as_tsibble(index = dates)
```

```r
# Calculate quarter-on-quarter growth rate
USMacro_ts <- USMacro_ts %>%
  mutate(gdp_growth = (gdp - lag(gdp)) / lag(gdp))
```

Data Transformation In order to normalise most of the variation and detect relevant features and patterns, data is transformed to create a growth series. Indeed, transforming the data may help in stabilizing the variance and achieving stationarity.

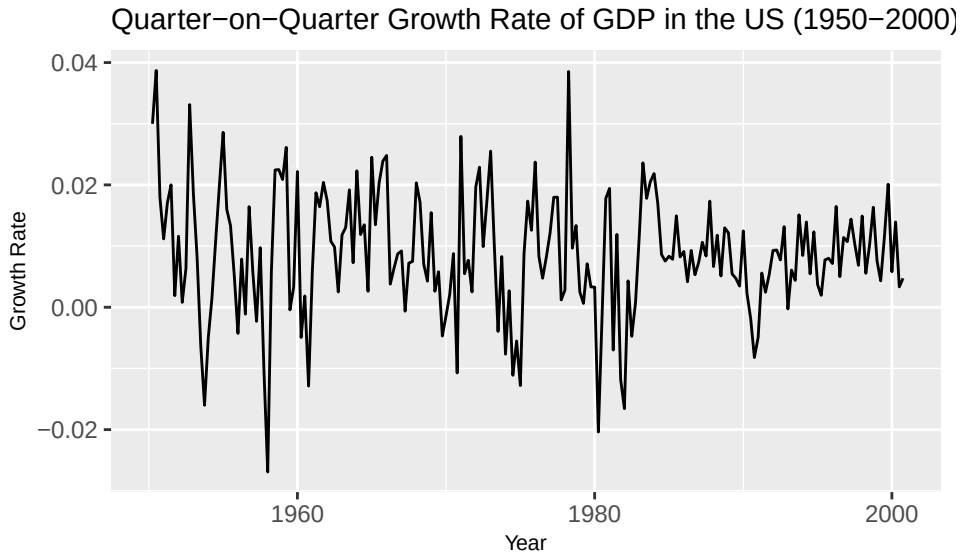
Stationarity in a time series is characterized by consistent statistical properties such as a constant mean, variance, and autocorrelation throughout its temporal evolution. Many statistical forecasting methods rely on the assumption of stationarity. When dealing with non-stationary data, challenges in model estimation arise, leading to potentially invalid inferences. Non-stationary time series often exhibit trends, seasonality, and cyclic patterns, which can compromise the stability of time series models. Trends can be addressed through differencing, a process involving the subtraction of the current value from the previous one. Seasonal effects, on the other hand, can be mitigated by employing seasonal differencing, which involves taking the difference with a lag corresponding to the seasonality period. A hallmark of a stationary time series is an autocorrelation function (ACF) plot that swiftly drops to zero, indicating the absence of discernible patterns extending into the future.

```r
# Plot growth series
gdp_growth_plot <- USMacro_ts %>%
  ggplot(aes(x = dates, y = gdp_growth)) +
  geom_line() +
  labs(title = "Quarter-on-Quarter Growth Rate of GDP in the US (1950-2000)",
       x = "Year",
       y = "Growth Rate")+
  theme(plot.title = element_text(size = 11),  # Adjust the plot title font size
        axis.title = element_text(size = 8))

print(gdp_growth_plot)
```

GDP Growth Series and Autocorrelation

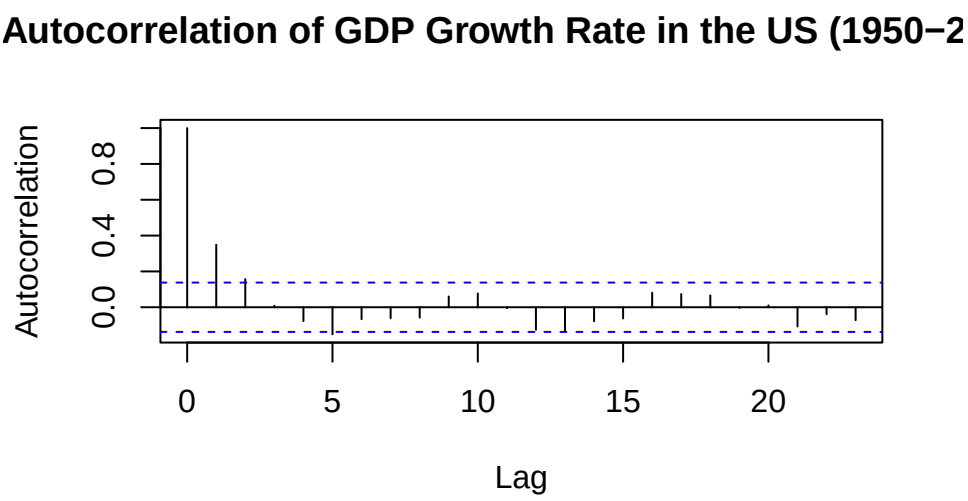
```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_line()`).
```




```r
#Remove null values in the gdp growth variable
gdp_growth <- na.omit(USMacro_ts$gdp_growth)

# Calculate the autocorrelation of the "gap" variable
autocorrelation <- acf(gdp_growth, plot = FALSE)

# Plot the autocorrelation
plot(autocorrelation,
     main = "Autocorrelation of GDP Growth Rate in the US (1950-2000)",
     xlab = "Lag",
     ylab = "Autocorrelation")
```



In the provided plot, discerning a clear trend is challenging, and while seasonality or cyclic behavior could be present, the visual pattern is suggestive of random spikes. The ACF values for higher lags remain within certain bounds, and the absence of a repeating pattern of spikes further obscures any evident signs of seasonality or cyclic behavior in the GDP growth rate series, as indicated by this ACF plot. It is crucial to recognize that the lack of a discernible pattern in the ACF does not conclusively negate the existence of seasonality or cycles. The absence could result from method

limitations or may necessitate adjustments or transformations to make seasonality more apparent in the data.

```
# Perform the Ljung-Box test
ljung_box_test <- Box.test(USMacro_ts$gdp_growth, lag = 20, type = "Ljung-Box")

# Print the test result
print(ljung_box_test)
```

Statistical Test

```
##
##  Box-Ljung test
##
## data:  USMacro_ts$gdp_growth
## X-squared = 54.221, df = 20, p-value = 5.364e-05
```

The Box-Ljung test serves as a valuable tool for evaluating autocorrelation within the residuals of a time series model. In the context of this analysis, the time series data under consideration is the gdp_growth variable. The test provides essential outputs, notably the test statistic denoted as 'X-squared,' the degrees of freedom represented by 'df,' and the p-value, which serves as an indicator of statistical significance. The null hypothesis (H0) posits the absence of autocorrelation in the residuals. In this specific instance, the calculated p-value is less than the conventional significance level of 0.05. Consequently, the rejection of the null hypothesis is warranted. This implies that the residuals likely exhibit autocorrelation, underscoring the potential need for refining the model to better capture underlying patterns or account for unaddressed features in the time series data.

Question 3

Load data Data set olympic_running in the fpp3 package contains the winning times (in seconds) in each Olympic Games sprint, middle-distance and long-distance track events from 1896 to 2016. (Hint: typing olympic_running into R will automatically call the dataset. This is the same method, how we usually call the fpp3 datasets in the workshops.)

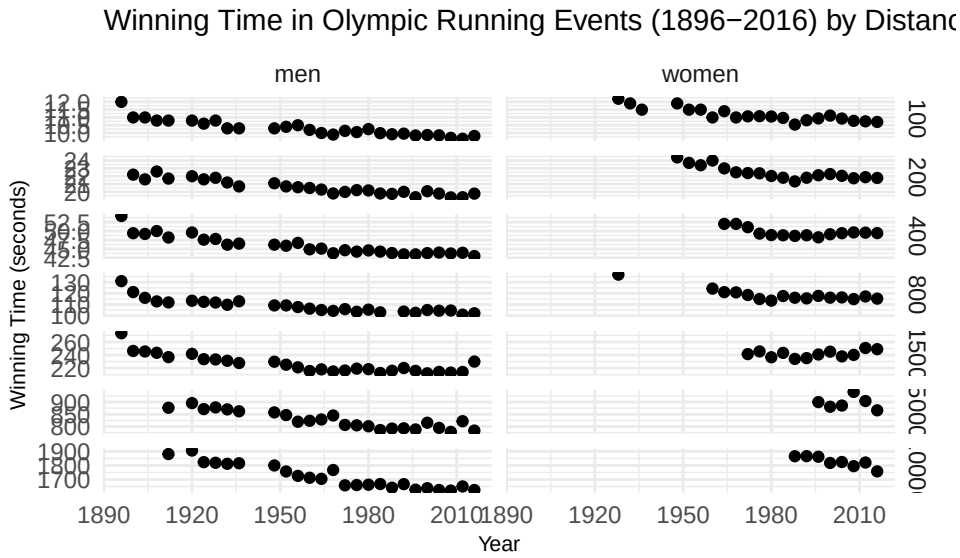
```
# Load data
data("olympic_running")
head(olympic_running)
```

```
## # A tsibble: 6 x 4 [4Y]
## # Key:       Length, Sex [1]
##    Year Length Sex    Time
##   <int>  <int> <chr> <dbl>
## 1  1896    100 men    12
## 2  1900    100 men    11
## 3  1904    100 men    11
## 4  1908    100 men    10.8
## 5  1912    100 men    10.8
## 6  1916    100 men    NA
```

```r
# Plot winning time against the year for each event
olympic_running %>%
  ggplot(aes(x = Year, y = Time)) +
  #Add points for a scatterplot
  geom_point() +
  # Facet by distance and gender
  facet_grid(Length ~ Sex, scales = "free_y") +
  labs(title = "Winning Time in Olympic Running Events (1896-2016) by Distance and Gender",
       x = "Year",
       y = "Winning Time (seconds)") +
  theme_minimal() +
  theme(plot.title = element_text(size = 11),  # Adjust the plot title font size
        axis.title = element_text(size = 8))
```

a) Plot the winning time against the year for each event. Describe the main features of the plot.

```
## Warning: Removed 31 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



The observed trends and periodicity in the data present noteworthy insights. Across both men and women, a discernible trend is evident – the winning times for most events have consistently decreased over the years. This decline in winning times serves as a compelling indication of improved athletic performance across various Olympic events. The trend is particularly pronounced in events characterized by longer winning times, reflecting advancements and achievements in athletic capabilities.

Additionally, a periodic pattern emerges in the dataset, marked by distinct absences of data points. This periodicity is likely associated with years when the Olympic Games did not take place. Indeed, the Olympic Game were cancelled in 1916, and from 1940 until 1944, due to the World Wars. The periodic gaps in the data align with historical events that disrupted the regular occurrence of the Olympic Games, highlighting the impact of external factors on the continuity of athletic

competitions. These observations contribute to a comprehensive understanding of the temporal dynamics and historical influences within the dataset.

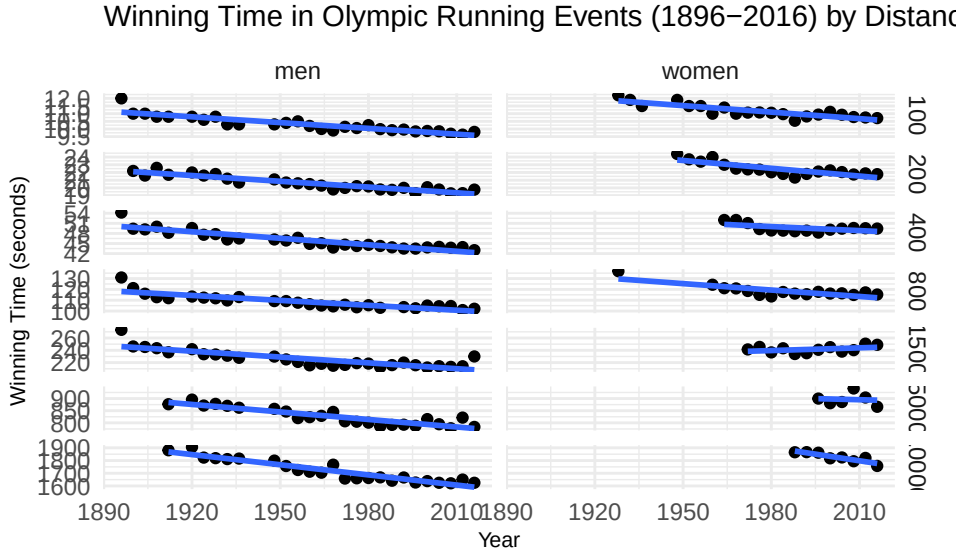
```
olympic_running %>%
  ggplot(aes(x = Year, y = Time)) +
  #Add points for a scatterplot
  geom_point() +
  # Add a smoothed line
  geom_smooth(method = "lm", se = FALSE) +
  # Facet by distance and gender
  facet_grid(Length ~ Sex, scales = "free_y") +
  labs(title = "Winning Time in Olympic Running Events (1896-2016) by Distance and Gender",
       x = "Year",
       y = "Winning Time (seconds)") +
  theme_minimal() +
  theme(plot.title = element_text(size = 11),  # Adjust the plot title font size
        axis.title = element_text(size = 8))
```

b) Fit a regression line to the data for each event. Obviously the winning times have been decreasing, but at what average rate per year?

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 31 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 31 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



The plot clearly shows the decrease of winning times for most of the lengths over the years. Only for the women race with the length of 1500m the regression line shows a slight increase for unexplained reasons. The highest absolute decrease is for men in the 10000m with almost 300 seconds decrease.

```r
# Fit linear regression models for each event
regression_models <- olympic_running %>%
  group_by(Length, Sex) %>%
  do(model = lm(Time ~ Year, data = .))

# Extract coefficients (slope) from each model and calculate the mean slope
coefficients <- regression_models %>%
  summarise(Length = unique(Length),
            Sex = unique(Sex),
            Slope = coef(model)[2])  # Extract the coefficient for the 'Year' variable

# Calculate the mean of slopes
average_slope <- mean(coefficients$Slope)

# View the average slope
print(average_slope)
```

```
## [1] -0.5859262
```

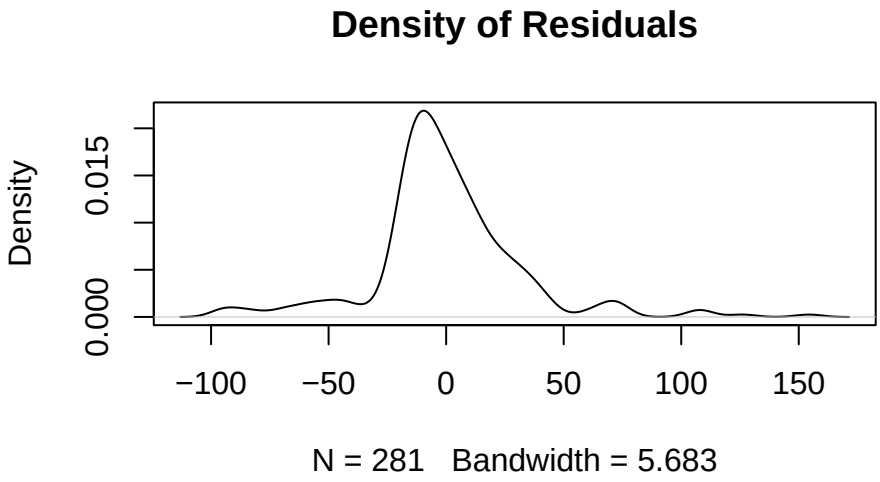
The resulting average rate decrease for each event is approximately -0.59. This represents the average rate of decrease in winning times per year for each event. More precisely, the coefficient should be in terms of seconds per year, concluding that the winning time has decreased by 0.59 seconds on average each year.

```r
#fit a regression model
model <- lm(Time~Year+Length + Sex, data=olympic_running)

#get list of residuals
res <- resid(model)

plot(density(res),
     main = "Density of Residuals",
     ylab = "Density")
```

c) Plot the residuals against the year. What does this indicate about the suitability of the fitted lines?



The shape of the density plot is asymmetric and it looks like there might be some outliers or extreme values, as suggested by the long tails, particularly towards the right. There is a peak away from zero, which indicates that the residuals are biased in one direction for a number of observations. Since the residuals do not appear to be normally distributed, this suggests that the assumptions of the linear regression may not have been fully met.

```r
# Fit linear regression models for women
regression_models_women <- list()

# Loop over unique lengths for women
for (length in unique(olympic_running$Length)) {
  # Subset data for women
  subset_data <- subset(olympic_running, Length == length & Sex == "women")
  # Fit linear regression model for women
  lm_model <- lm(Time ~ Year, data = subset_data)
  # Store the model in the regression_models_women list with appropriate names
  model_name <- paste("Length", length, "Sex_women", sep = "")
  regression_models_women[[model_name]] <- lm_model
}

# Fit linear regression models for men
regression_models_men <- list()

# Loop over unique lengths for men
for (length in unique(olympic_running$Length)) {
  # Subset data for men
  subset_data <- subset(olympic_running, Length == length & Sex == "men")
  # Fit linear regression model for men
  lm_model <- lm(Time ~ Year, data = subset_data)
  # Store the model in the regression_models_men list with appropriate names
  model_name <- paste("Length", length, "Sex_men", sep = "")
  regression_models_men[[model_name]] <- lm_model
}
```

```r
# Create a dataframe with all possible combinations of Length, Sex, and Year 2020 for women
prediction_data_women <- expand.grid(Year = 2020, Length = unique(olympic_running$Length), Sex

# Predict winning time and calculate prediction intervals for women
# Initialize an empty dataframe to store predictions for women
predictions_women <- data.frame()

# Loop over unique lengths for women
for (length in unique(olympic_running$Length)) {
  # Get the name of the model for women
  model_name <- paste("Length", length, "Sex_women", sep = "")
  # Check if the model exists for women
  if (model_name %in% names(regression_models_women)) {
    # Access the regression model for women
    lm_model <- regression_models_women[[model_name]]
    # Subset prediction data for the specific combination of length and sex for women
    subset_data <- prediction_data_women[prediction_data_women$Length == length, ]
    # Predict winning time and calculate prediction interval for women
    pred_interval <- predict(lm_model, newdata = subset_data, interval = "prediction")
    # Append predictions to the dataframe for women
    predictions_women <- rbind(predictions_women, pred_interval)
  } else {
    # Issue a warning if the regression model does not exist for the combination of length and
    warning(paste("No regression model found for Length:", length, "and Sex: women"))
  }
}

# Combine prediction data with the original dataframe for women
prediction_data_women <- cbind(prediction_data_women, predictions_women)

# Add title to the dataframe for women
names(prediction_data_women) <- c("Year", "Length", "Sex", "Fit", "Lower", "Upper")

# View predictions for women
print("Predictions for Women:")
```

d) Predict the winning time for each race in the 2020 Olympics. Give a prediction interval for your forecasts. What assumptions have you made in these calculations?

```
## [1] "Predictions for Women:"
```

```r
print(prediction_data_women)
```

```
##   Year Length   Sex        Fit      Lower      Upper
## 1 2020    100 women   10.53441   10.02458   11.04424
## 2 2020    200 women   21.19974   20.01734   22.38214
## 3 2020    400 women   48.43626   45.86503   51.00750
## 4 2020    800 women  111.48343  103.40936  119.55751
```

```
## 5 2020   1500 women  245.45652  232.18997  258.72306
## 6 2020   5000 women  892.11467  782.40065 1001.82868
## 7 2020  10000 women 1763.01536 1706.68532 1819.34539
```

```r
# Create a dataframe with all possible combinations of Length, Sex, and Year 2020 for men
prediction_data_men <- expand.grid(Year = 2020, Length = unique(olympic_running$Length), Sex =

# Predict winning time and calculate prediction intervals for men
# Initialize an empty dataframe to store predictions for men
predictions_men <- data.frame()

# Loop over unique lengths for men
for (length in unique(olympic_running$Length)) {
  # Get the name of the model for men
  model_name <- paste("Length", length, "Sex_men", sep = "")
  # Check if the model exists for men
  if (model_name %in% names(regression_models_men)) {
    # Access the regression model for men
    lm_model <- regression_models_men[[model_name]]
    # Subset prediction data for the specific combination of length and sex for men
    subset_data <- prediction_data_men[prediction_data_men$Length == length, ]
    # Predict winning time and calculate prediction interval for men
    pred_interval <- predict(lm_model, newdata = subset_data, interval = "prediction")
    # Append predictions to the dataframe for men
    predictions_men <- rbind(predictions_men, pred_interval)
  } else {
    # Issue a warning if the regression model does not exist for the combination of length and
    warning(paste("No regression model found for Length:", length, "and Sex: men"))
  }
}

# Combine prediction data with the original dataframe for men
prediction_data_men <- cbind(prediction_data_men, predictions_men)

# Add title to the dataframe for men
names(prediction_data_men) <- c("Year", "Length", "Sex", "Fit", "Lower", "Upper")

# View predictions for men
print("Predictions for Men:")
```

```
## [1] "Predictions for Men:"
```

```r
print(prediction_data_men)
```

```
##   Year Length Sex         Fit       Lower      Upper
## 1 2020    100 men    9.534507    9.025795   10.04322
## 2 2020    200 men   19.117439   18.387811   19.84707
## 3 2020    400 men   42.042310   39.552856   44.53176
## 4 2020    800 men   99.237699   91.776651  106.69875
```

```
## 5 2020   1500 men  206.998282  189.267607  224.72896
## 6 2020   5000 men  772.824562  737.812419  807.83670
## 7 2020  10000 men 1580.351220 1515.752337 1644.95010
```

The following assumptions were made in the calculations:

1. Linear Relationship: Assumes the relationship between year and winning time is linear.
2. Constant Variance: Assumes the variance of residuals is constant across all levels of the independent variables.
3. Independence: Assumes observations (winning times) are independent of each other.
4. Normality of Residuals: Assumes residuals follow a normal distribution.
5. Stationarity: Assumes the data, including winning times and other variables, are stationary.
6. Validity of Regression Model: Assumes the fitted regression model accurately captures the relationship between variables without issues like omitted variables or multicollinearity.