Time Series Analysis of Rainfall in the Rio Grande Valley, Texas, 1941-present

John Knight

Final Project – STAT 6380

07/09/2024

Introduction

The Earth is getting hotter,[1] but how does this affect rainfall? An increase in temperature could make an area more desert-like,[2] or it could increase severe weather events such as thunderstorms and hurricanes.[3] In this study, a time series of precipitation levels over the past century in the Rio Grande Valley, Texas, is analyzed.

The first objective of this study is to assess whether there is a discernable trend in annual precipitation levels across time. The second objective is to show seasonal patterns in precipitation in the Rio Grande Valley. The third objective is to investigate autocorrelation in precipitation totals between consecutive months and years, and the final objective is to use spectral analysis to reveal the underlying frequencies of rainfall patterns.

Methods

Daily precipitation amounts, in inches, were downloaded from the free archive of the National Centers for Environmental Information[4] (NCEI) for four weather stations: McAllen, McAllen Airport, Weslaco, and Harlingen (see Table 1). The start date of 1941-06-01 was chosen because this was the earliest available data for McAllen (both Harlingen and Weslaco data go back further). Missing values were imputed by taking the adjusted mean of the other stations on each date. Note that McAllen Airport data was only used for imputation purposes, and was omitted from further analysis to avoid overweighting the McAllen area relative to the other two stations. The daily mean precipitation was then calculated for each date, and a monthly mean was calculated as the average of the daily means for each month (to avoid bias toward months with more days).

Table 1. Coverage and mean daily precipitation for the four weather stations.

Station	Start Date	Total Rows	NA Rows	Coverage (%)	Mean Prcp.
Harlingen	1941-06-01	30331	1809	94.0	0.0728
McAllen	1941-06-01	30331	2459	91.9	0.0571
McAllen Airport	1961-01-14	23164	597	97.4	0.0594
Weslaco	1941-06-01	30331	3484	88.5	0.0667

All statistical analysis was performed using R software. The time series was decomposed into its trend, seasonal, and remainder components using STL decomposition. A range of trend window parameters were tested to find an appropriate smoothing balance. Boxplots were created to visualize the seasonal distribution by month. The remainder component was then assessed for autocorrelation by inspection of the correlogram. The strengths of the trend and seasonality were measured using the following formulae:

$F_t = \max\left(0, 1 - \frac{VAR(Z_t)}{VAR(M_t+Z_t)}\right)$ with range [0,1] (no trend to very strong trend).

$F_s = \max\left(0, 1 - \frac{VAR(Z_t)}{VAR(S_t+Z_t)}\right)$ with range [0,1] (no seasonality to very strong seasonality).

The autocorrelation function (ACF) and partial ACF of the series were inspected, and an ARIMA model was fitted to the data with the auto.arima function using parameters approximation=FALSE and stepwise=FALSE, thereby instructing a more thorough search for appropriate models. The residuals of the resulting model underwent a visual inspection as well as

a Ljung-Box test for autocorrelation of residuals. This model was then used to create a forecast of precipitation for the next 5 years (60 months).

Finally, spectral analysis was performed on the time series. A smoothed periodogram was created using the Daniell kernel, with various sized windows tested to find an appropriate level of smoothing. Confidence intervals were calculated for frequencies of prominent peaks in the resulting periodogram to add context to their significance.

Results

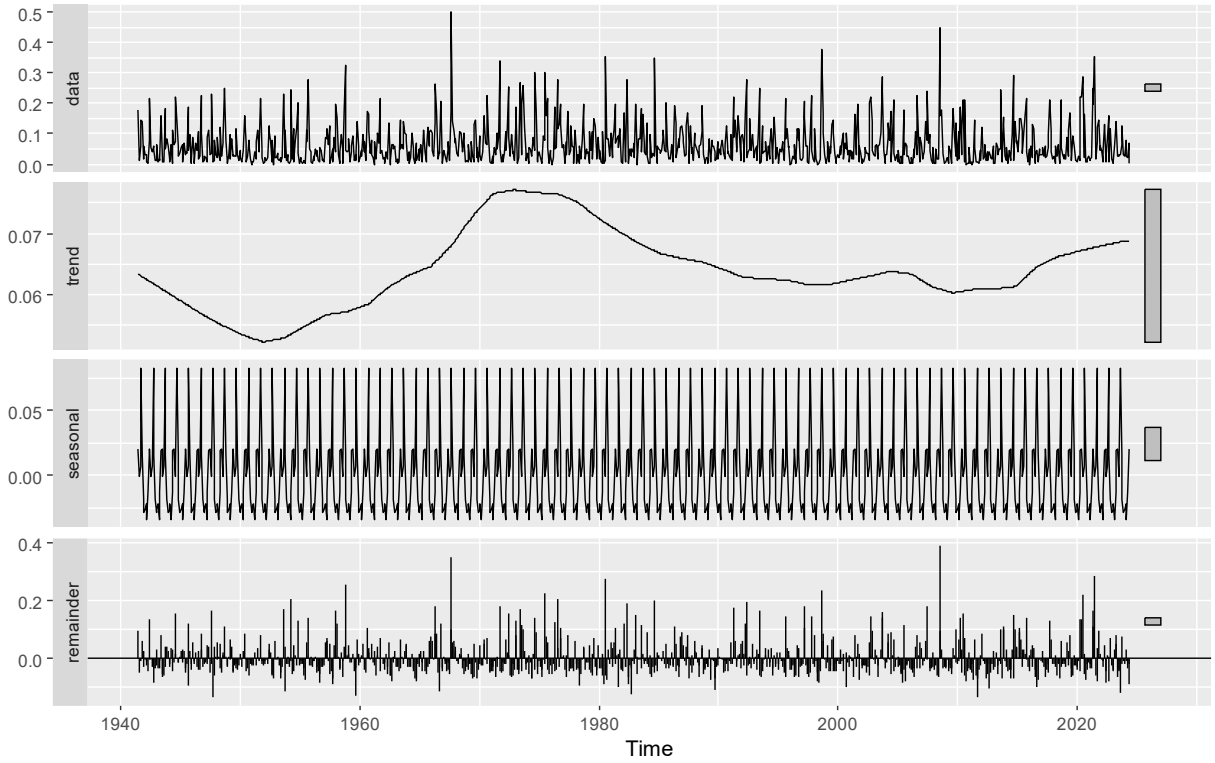

Figure 1. STL decomposition of monthly mean precipitation for Rio Grande Valley, 1941-2024.

The time series is summarized in Figure 1, which shows the raw series, plus the trend, seasonal, and remainder components. A trend window of 201 was used to achieve the smooth trend seen above – smaller values resulted in more fluctuation. The trend can be seen to peak in the 1970s, decreasing thereafter before rising again in the past 10 years. The strength of the trend was found to be very weak, 0.012, and the strength of the seasonality was 0.215. In the correlogram of the remainders, there was significant autocorrelation at lag 1, while in the correlogram of the raw series, there was clear evidence of seasonal autocorrelation at lags 12 and 24 (see Figures 4 and 5 in Appendix).

The seasonal boxplots in Figure 2 reveal that September is the month with the highest precipitation on average. May and June are also months with high precipitation. The lowest rainfall occurs from November through to April.

The auto.arima function found the optimal model to be ARIMA(1,0,0)(2,0,0)[12]. This aligns with earlier inspection of the correlograms which showed autocorrelation at lags 1, 12, and 24. The coefficients of the ARIMA model were AR1 = 0.127, SAR1 = 0.223, SAR2 = 0.141, with a

mean of 0.064. The Ljung-Box test for autocorrelation returned a p-value of 0.267, indicating that there is no significant autocorrelation in the residuals. The Akaike Information Criterion (AIC) was -2656.61. This model was used to forecast the next 5 years with a relatively flat trend and prediction intervals shown in Figure 6 in the Appendix.

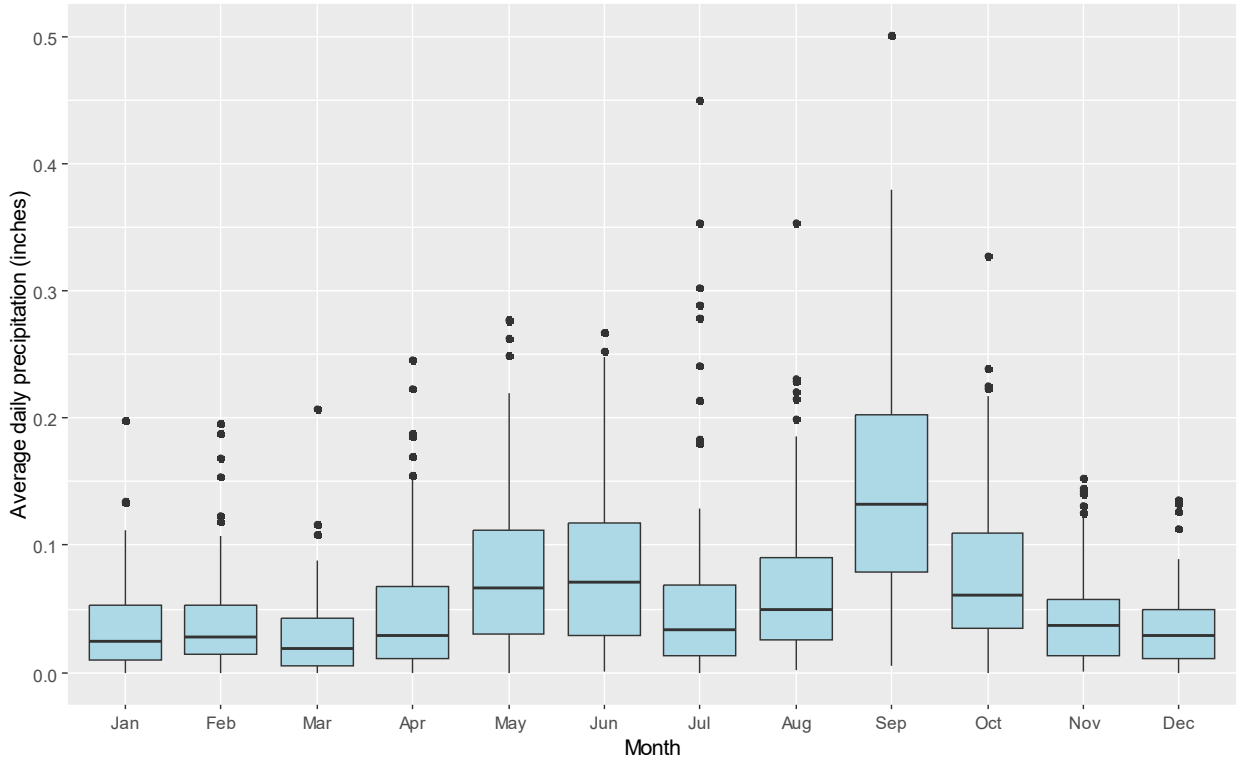

Figure 2. Boxplot displaying monthly distribution of precipitation.

The periodogram for the spectral analysis can be seen in Figure 3. A Daniell kernel with window size (13, 13) and tapering of 0.1 was found to be suitable for a smooth and clear spectrum. The most prominent peak, as expected, was at frequency 1 (annual) but there was also a prominent peak at frequency 3. Smaller peaks were observed at frequencies 2, 4, and 5. Confidence intervals were calculated for frequencies 1, 2, 3, 4, and 5 and they can be seen along with the estimated spectral densities in Table 2.

Table 2. Spectral density estimates and 95% confident intervals for peak values in periodogram.

Frequency	Spectral Density	95% CI
1	0.000396	(0.000211, 0.000639)
2	0.000441	(0.000235, 0.000711)
3	0.000362	(0.000193, 0.000584)
4	0.000249	(0.000132, 0.000401)
5	0.000253	(0.000135, 0.000408)

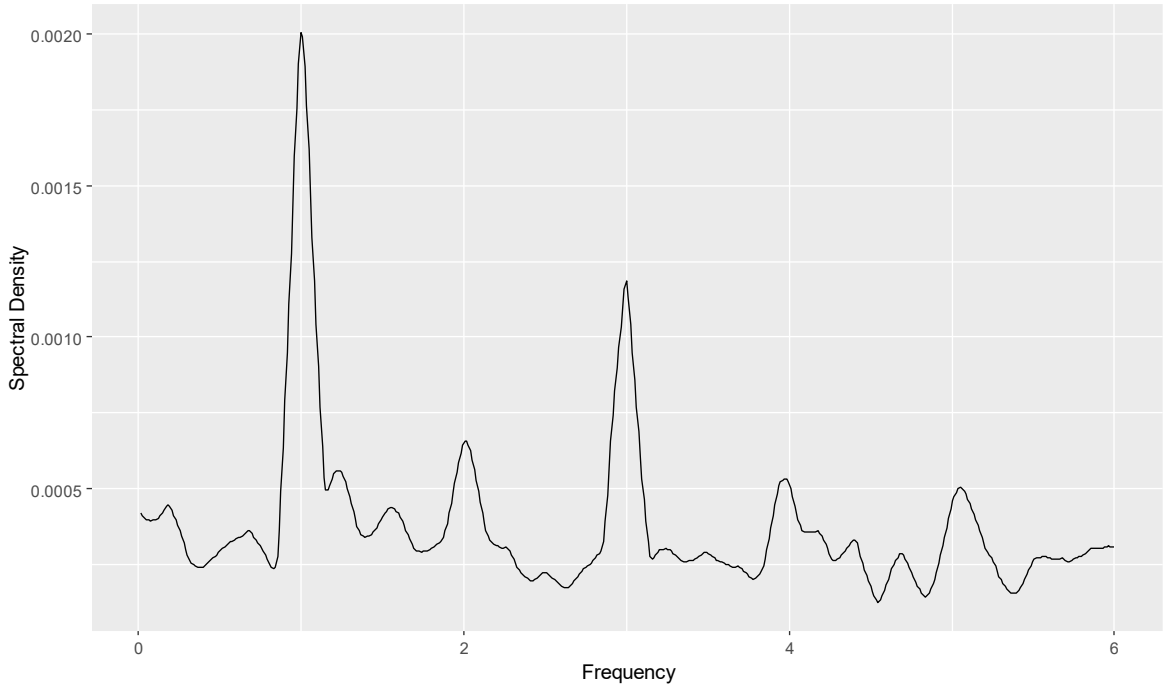

Figure 3. Periodogram of monthly precipitation time series smoothed using Daniell kernel.

Discussion

The study found no significant evidence of a long-term change in the trend of precipitation in the Rio Grande Valley. However, the autocorrelation in the ARIMA model suggested the presence of cycles in rainfall patterns. These may be related to various weather phenomena such as El Niño and La Niña.[5] The highest seasonal rainfall in September evidently coincides with the peak of the Atlantic hurricane season.[6] The prominent peak at frequency 1 in the spectral analysis is not surprising, since the Earth's climate is known to experience annual seasons. However, further peaks at frequencies 2, 3, 4, and 5 suggest more complex underlying systems that interact throughout the year. This helps to explain why the seasonal totals from January to December to not resemble a simple curve, as the equivalent measures for temperature would do.

The strength of this study is that it leverages raw data, direct from source, to analyze the climate in a specific region of the United States whose climate is significantly hotter than other areas. This method can uncover more accurate insights than simply by looking at national or global climate summaries. The use of ARIMA provides a robust model to detect autocorrelation among observations in a time series, helping to explain monthly and yearly changes in weather patterns.

A possible limitation is the unknown reliability of the data. An equivalent dataset for temperature was explored, and it contained many outliers which were obvious errors (for example, a temperature of 0°F or a temperature vastly different from a neighboring city). However, these errors are harder to detect in precipitation, because a value of 0 is always valid, and it is plausible for two cities a short distance apart to experience drastically different rainfall on a given day. Some possible avenues for further exploration of this topic would be to look at the number of major rain events, rather than the total precipitation, or a multivariate analysis comparing temperature and precipitation.

References

1. NASA Earth Observatory. (2023, January 13). *2022 tied for fifth warmest year on record*. NASA. Retrieved from https://earthobservatory.nasa.gov/images/150828/2022-tied-for-fifth-warmest-year-on-record

2. Doyle, A. (2016, October 27). Global warming to turn parts of Europe into desert by end of the century, report warns. The Independent. Retrieved from https://www.independent.co.uk/climate-change/news/global-warming-desert-europe-climate-change-a7375276.html

3. NASA Earth Observatory. (2023, January 13). *How climate change may be impacting storms over Earth's tropical oceans*. NASA. Retrieved from https://science.nasa.gov/earth-science/oceanography/living-ocean/hurricanes-and-climate-change

4. National Centers for Environmental Information (NCEI). (n.d.). Data access. NOAA. Retrieved July 2, 2024, from https://www.ncei.noaa.gov/access/search/

5. National Oceanic and Atmospheric Administration (NOAA). (2024, June 16). *What are El Niño and La Niña?* NOAA. Retrieved from https://oceanservice.noaa.gov/facts/ninonina.html

6. National Oceanic and Atmospheric Administration (NOAA). (2016, August 22). *The peak of the hurricane season – why now?* NOAA. Retrieved from https://www.noaa.gov/stories/peak-of-hurricane-season-why-now

Appendix

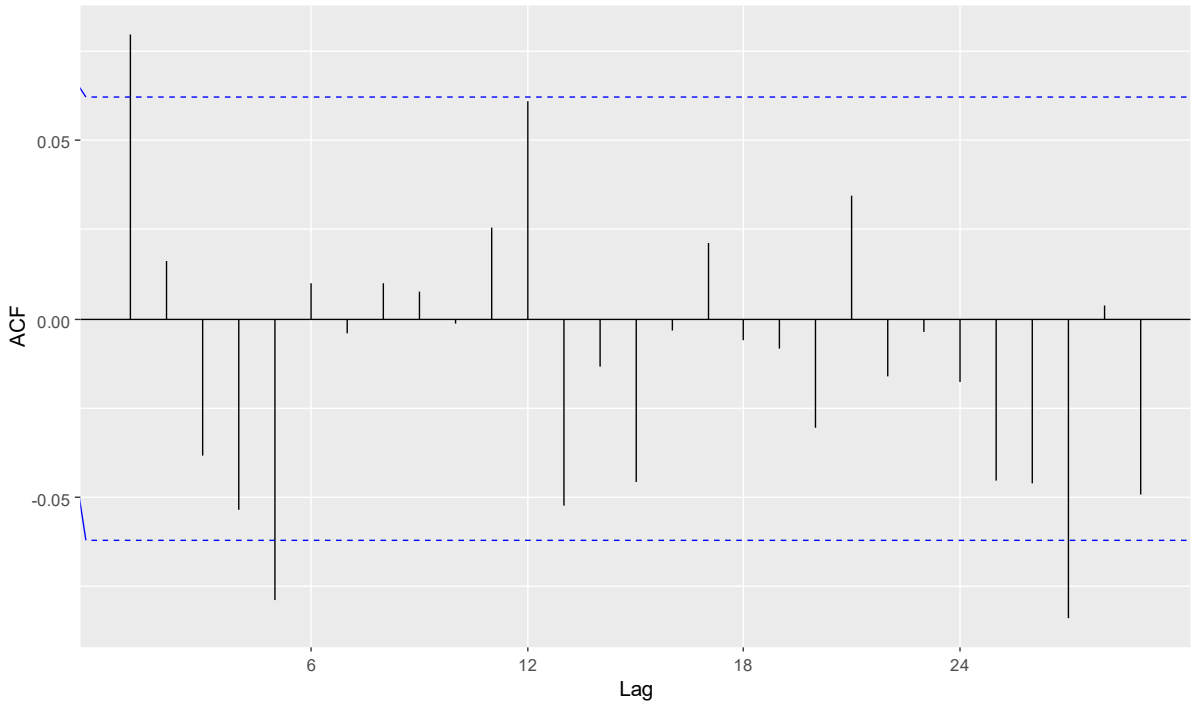

Figure 4. Correlogram showing ACF or remainder component after trend and seasonality have been removed.

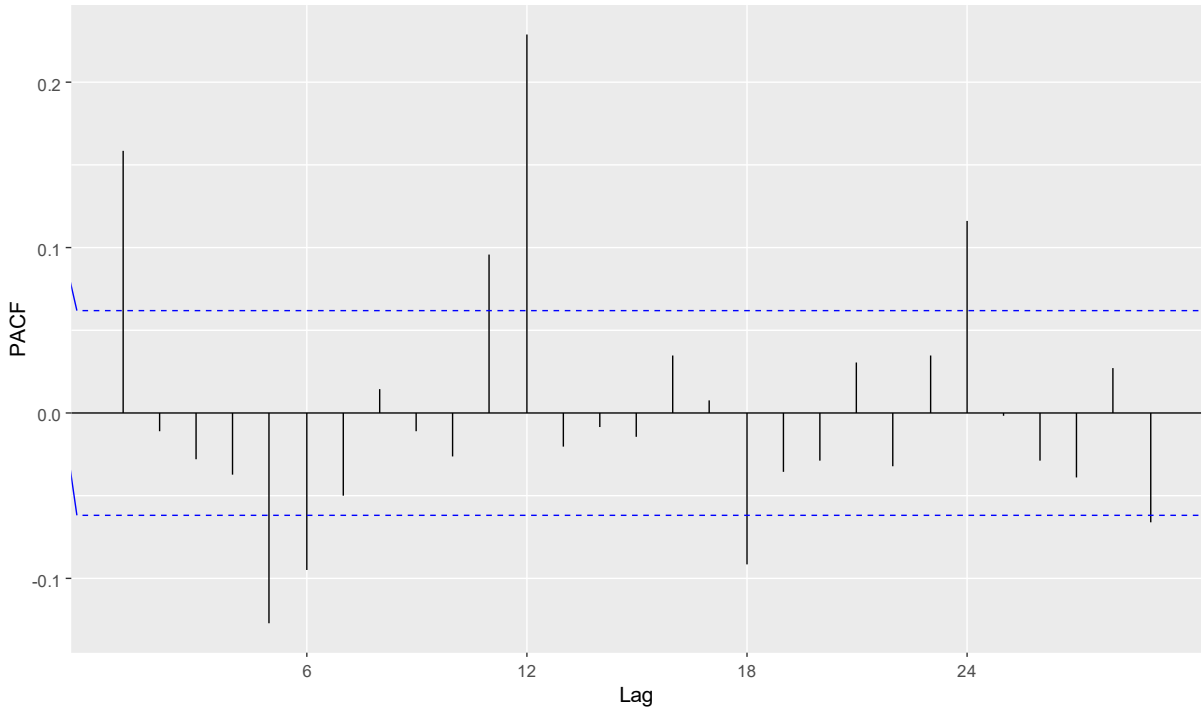

Figure 5. Partial autocorrelation function of monthly precipitation.

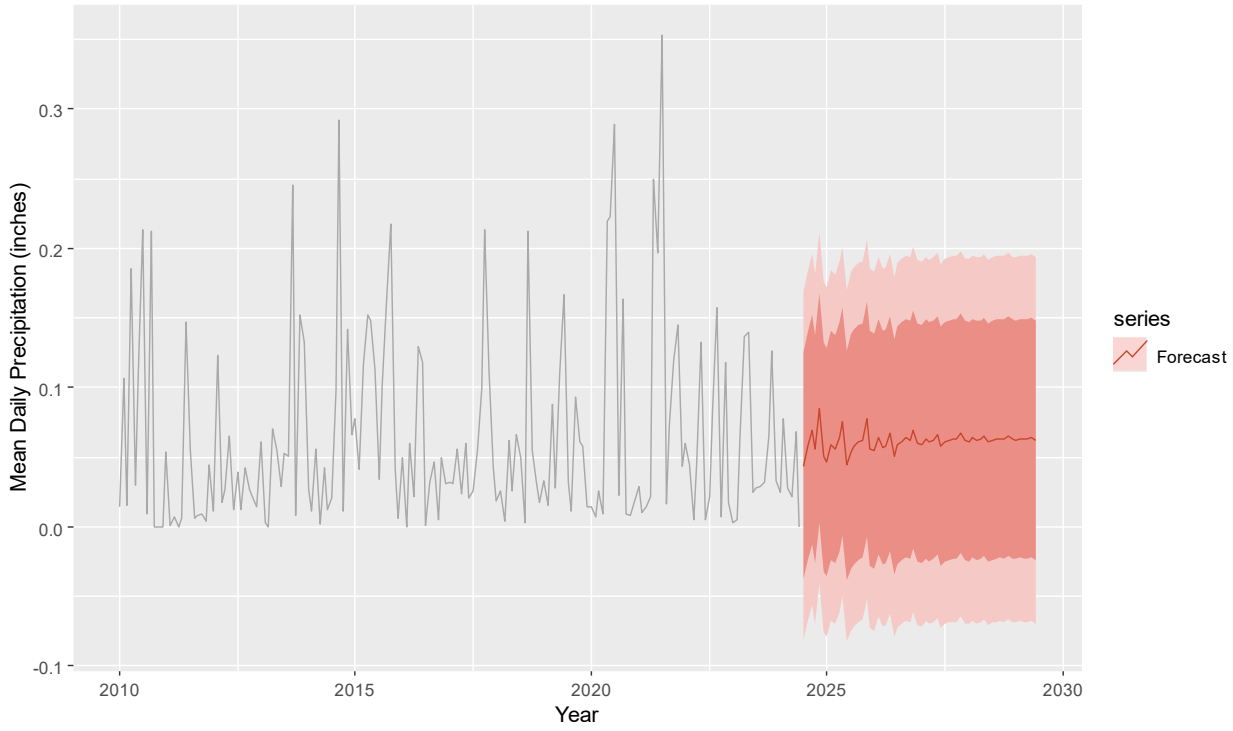

Figure 6. Forecast of precipitation for next 5 years (60 months).

R Code

```
### John Knight - STAT 6380 ###

### Final Project - RGV Climate ###

# Load the necessary libraries
library(dplyr)
library(tidyr)
library(lubridate)
library(forecast)
library(ggplot2)
theme_set(theme_gray() +
          theme(plot.caption = element_text(hjust = 0))) # Set the default theme with left-justified
captions
library(zoo)
library(tidyverse)
library(tibble)

options(scipen = 999) # To avoid scientific notation in plots

# Read the CSV file
weather_data <- read.csv("weather_data.csv")
```

```
# Remove the STATION column
weather_data <- weather_data %>% select(-STATION)

# Convert DATE column to Date type
weather_data$DATE <- mdy(weather_data$DATE)

# Filter out rows where DATE is before 1941-06-01
weather_data <- weather_data %>% filter(DATE >= as.Date("1941-06-01"))

# Pivot the data so that the stations are columns
pivoted_data <- weather_data %>%
  pivot_wider(names_from = STATION_NAME, values_from = c(PRCP, TMAX)) %>%
  rename_with(~gsub(" ", "_", .))

############################################################

# Data cleaning and imputation

# 1. Detect outliers in TMAX and set them to NA.

# Detect outliers in TMAX. For each station, get diff with mean of other three stations.
pivoted_data$McAllen_Airport_Diff <- pivoted_data$TMAX_McAllen_Airport -
rowMeans(pivoted_data[, c("TMAX_Weslaco", "TMAX_Harlingen", "TMAX_McAllen")],
na.rm = TRUE)
pivoted_data$McAllen_Diff <- pivoted_data$TMAX_McAllen - rowMeans(pivoted_data[,
c("TMAX_Weslaco", "TMAX_Harlingen", "TMAX_McAllen_Airport")], na.rm = TRUE)
pivoted_data$Weslaco_Diff <- pivoted_data$TMAX_Weslaco - rowMeans(pivoted_data[,
c("TMAX_McAllen", "TMAX_Harlingen", "TMAX_McAllen_Airport")], na.rm = TRUE)
pivoted_data$Harlingen_Diff <- pivoted_data$TMAX_Harlingen - rowMeans(pivoted_data[,
c("TMAX_Weslaco", "TMAX_McAllen", "TMAX_McAllen_Airport")], na.rm = TRUE)

# Iterate through rows removing outliers
# Criteria: see if row has an absolute difference >= 20
# If yes, see how many diffs are +/-
# If 2v2 or 1v1, ignore
# If 3v1 or 2v1, turn the 1 into NA
n <- dim(pivoted_data)[1]
for (i in 1:n) {
  diffs <- pivoted_data[i, c("Weslaco_Diff", "Harlingen_Diff", "McAllen_Airport_Diff",
"McAllen_Diff")]

  # Check if any absolute value >= 20
  if (any(abs(diffs) >= 20, na.rm = TRUE)) {
    # Count negative and nonnegative values
    neg_count <- sum(diffs < 0, na.rm = TRUE)
```

```
    pos_count <- sum(diffs >= 0, na.rm = TRUE)

    if (neg_count != pos_count) {
      if (pos_count > neg_count) {
        # More positive than negative, turn the negative value's corresponding TMAX to NA
        if (!is.na(diffs["Weslaco_Diff"]) && diffs["Weslaco_Diff"] < 0) pivoted_data[i,
"TMAX_Weslaco"] <- NA
        if (!is.na(diffs["Harlingen_Diff"]) && diffs["Harlingen_Diff"] < 0) pivoted_data[i,
"TMAX_Harlingen"] <- NA
        if (!is.na(diffs["McAllen_Airport_Diff"]) && diffs["McAllen_Airport_Diff"] < 0)
pivoted_data[i, "TMAX_McAllen_Airport"] <- NA
        if (!is.na(diffs["McAllen_Diff"]) && diffs["McAllen_Diff"] < 0) pivoted_data[i,
"TMAX_McAllen"] <- NA
      } else {
        # More negative than positive, turn the positive value's corresponding TMAX to NA
        if (!is.na(diffs["Weslaco_Diff"]) && diffs["Weslaco_Diff"] >= 0) pivoted_data[i,
"TMAX_Weslaco"] <- NA
        if (!is.na(diffs["Harlingen_Diff"]) && diffs["Harlingen_Diff"] >= 0) pivoted_data[i,
"TMAX_Harlingen"] <- NA
        if (!is.na(diffs["McAllen_Airport_Diff"]) && diffs["McAllen_Airport_Diff"] >= 0)
pivoted_data[i, "TMAX_McAllen_Airport"] <- NA
        if (!is.na(diffs["McAllen_Diff"]) && diffs["McAllen_Diff"] >= 0) pivoted_data[i,
"TMAX_McAllen"] <- NA
      }
    }
  }
}

# 2. Get the coverage % for each of the 4 stations for PRCP and TMAX.

# Function to get column statistics for a single column
get_column_stats <- function(data, column_name) {
  # Get total rows in the original data
  total_rows <- nrow(data)

  # Find the earliest non-NA date
  non_na_rows <- data[!is.na(data[[column_name]]), ]
  start_date <- min(non_na_rows$DATE, na.rm = TRUE)

  # Filter the data from the start date onwards
  filtered_data <- data[data$DATE >= start_date, ]
  filtered_total_rows <- nrow(filtered_data)
  na_count <- sum(is.na(filtered_data[[column_name]]))
  non_na_percentage <- (1 - na_count / filtered_total_rows) * 100

  data.frame(
```

```
    Column = column_name,
    Start_Date = start_date,
    Total_Rows = filtered_total_rows,
    NA_Count = na_count,
    Non_NA_Percentage = round(non_na_percentage, 2)
  )
}

# Specify columns to analyze
columns_to_analyze <- c("TMAX_Harlingen", "TMAX_McAllen", "TMAX_McAllen_Airport",
"TMAX_Weslaco",
                        "PRCP_Harlingen", "PRCP_McAllen", "PRCP_McAllen_Airport",
"PRCP_Weslaco")

coverage <- data.frame()

# Loop through columns and get stats
for (col in columns_to_analyze) {
  col_stats <- get_column_stats(pivoted_data, col)
  coverage <- rbind(coverage, col_stats)
}

print(coverage)


# 3. Find rows with no PRCP or no TMAX for any station.

library(dplyr)

pivoted_data <- pivoted_data %>%
  mutate(
    TMAX_all_na = ifelse(
      rowSums(is.na(select(., TMAX_Harlingen, TMAX_McAllen, TMAX_McAllen_Airport,
TMAX_Weslaco))) == 4,
      1,
      0
    ),
    PRCP_all_na = ifelse(
      rowSums(is.na(select(., PRCP_Harlingen, PRCP_McAllen, PRCP_McAllen_Airport,
PRCP_Weslaco))) == 4,
      1,
      0
    )
  )
```

```
# 4. Get the mean PRCP and TMAX for each station, only using rows where all 4 have values.

mean_prcp <- pivoted_data %>%
  filter(!is.na(PRCP_Harlingen) &
           !is.na(PRCP_McAllen) &
           !is.na(PRCP_McAllen_Airport) &
           !is.na(PRCP_Weslaco)) %>%
  summarise(
    PRCP_Harlingen = mean(PRCP_Harlingen, na.rm = TRUE),
    PRCP_McAllen = mean(PRCP_McAllen, na.rm = TRUE),
    PRCP_McAllen_Airport = mean(PRCP_McAllen_Airport, na.rm = TRUE),
    PRCP_Weslaco = mean(PRCP_Weslaco, na.rm = TRUE)
  )

mean_prcp_df <- as.data.frame(mean_prcp)
mean_prcp_df <- data.frame(lapply(mean_prcp_df, function(x) sprintf("%.4f", x)))

print(mean_prcp_df)

mean_temps <- pivoted_data %>%
  filter(!is.na(TMAX_Harlingen) &
           !is.na(TMAX_McAllen) &
           !is.na(TMAX_McAllen_Airport) &
           !is.na(TMAX_Weslaco)) %>%
  summarise(
    TMAX_Harlingen = mean(TMAX_Harlingen, na.rm = TRUE),
    TMAX_McAllen = mean(TMAX_McAllen, na.rm = TRUE),
    TMAX_McAllen_Airport = mean(TMAX_McAllen_Airport, na.rm = TRUE),
    TMAX_Weslaco = mean(TMAX_Weslaco, na.rm = TRUE)
  )

mean_temps_df <- as.data.frame(mean_temps)
mean_temps_df <- data.frame(lapply(mean_temps_df, function(x) sprintf("%.3f", x)))

print(mean_temps_df)

# 5. Impute missing values of PRCP and TMAX based on available values, with an adjustment
based on means.

# Calculate the overall mean for precipitation and temperature
overall_mean_prcp <- mean(as.numeric(mean_prcp_df), na.rm = TRUE)
overall_mean_temp <- mean(as.numeric(mean_temps_df), na.rm = TRUE)

# Calculate the ratio adjustments for PRCP
prcp_adjustments <- as.numeric(mean_prcp_df) / overall_mean_prcp
temp_adjustments <- as.numeric(mean_temps_df) - overall_mean_temp
```

```
# Rename adjustments for easier reference
names(prcp_adjustments) <- names(mean_prcp_df)
names(temp_adjustments) <- names(mean_temps_df)

# Function to adjust PRCP values by ratio
adjust_prcp_values <- function(value, adjustment) {
  if (is.na(value)) {
    return(NA)
  } else {
    return(value / adjustment)
  }
}

# Function to adjust TMAX values by difference
adjust_tmax_values <- function(value, adjustment) {
  if (is.na(value)) {
    return(NA)
  } else {
    return(value - adjustment)
  }
}

# Apply the adjustments to each row
pivoted_data <- pivoted_data %>%
  rowwise() %>%
  mutate(
    Adjusted_PRCP_McAllen_Airport = adjust_prcp_values(PRCP_McAllen_Airport,
prcp_adjustments["PRCP_McAllen_Airport"]),
    Adjusted_PRCP_Harlingen = adjust_prcp_values(PRCP_Harlingen,
prcp_adjustments["PRCP_Harlingen"]),
    Adjusted_PRCP_McAllen = adjust_prcp_values(PRCP_McAllen,
prcp_adjustments["PRCP_McAllen"]),
    Adjusted_PRCP_Weslaco = adjust_prcp_values(PRCP_Weslaco,
prcp_adjustments["PRCP_Weslaco"]),
    Adjusted_TMAX_McAllen_Airport = adjust_tmax_values(TMAX_McAllen_Airport,
temp_adjustments["TMAX_McAllen_Airport"]),
    Adjusted_TMAX_Harlingen = adjust_tmax_values(TMAX_Harlingen,
temp_adjustments["TMAX_Harlingen"]),
    Adjusted_TMAX_McAllen = adjust_tmax_values(TMAX_McAllen,
temp_adjustments["TMAX_McAllen"]),
    Adjusted_TMAX_Weslaco = adjust_tmax_values(TMAX_Weslaco,
temp_adjustments["TMAX_Weslaco"])
  ) %>%
  ungroup()
```

```
# Calculate the mean of the non-NA values for the adjusted PRCP and TMAX columns
mean_adjusted_prcp <- pivoted_data %>%
  select(Adjusted_PRCP_Harlingen, Adjusted_PRCP_McAllen,
Adjusted_PRCP_McAllen_Airport, Adjusted_PRCP_Weslaco) %>%
  summarise(across(everything(), ~mean(., na.rm = TRUE)))

mean_adjusted_tmax <- pivoted_data %>%
  select(Adjusted_TMAX_Harlingen, Adjusted_TMAX_McAllen,
Adjusted_TMAX_McAllen_Airport, Adjusted_TMAX_Weslaco) %>%
  summarise(across(everything(), ~mean(., na.rm = TRUE)))

# Create the imputed columns based on row means of non-NA adjusted values
pivoted_data <- pivoted_data %>%
  rowwise() %>%
  mutate(
    Imputed_PRCP_Harlingen = ifelse(is.na(Adjusted_PRCP_Harlingen),
                                    ifelse(all(is.na(c(Adjusted_PRCP_Harlingen, Adjusted_PRCP_McAllen,
Adjusted_PRCP_McAllen_Airport, Adjusted_PRCP_Weslaco))), NA,
                                           mean(c(Adjusted_PRCP_McAllen,
Adjusted_PRCP_McAllen_Airport, Adjusted_PRCP_Weslaco), na.rm = TRUE)),
                                    Adjusted_PRCP_Harlingen),
    Imputed_PRCP_McAllen = ifelse(is.na(Adjusted_PRCP_McAllen),
                                  ifelse(all(is.na(c(Adjusted_PRCP_Harlingen, Adjusted_PRCP_McAllen,
Adjusted_PRCP_McAllen_Airport, Adjusted_PRCP_Weslaco))), NA,
                                         mean(c(Adjusted_PRCP_Harlingen,
Adjusted_PRCP_McAllen_Airport, Adjusted_PRCP_Weslaco), na.rm = TRUE)),
                                  Adjusted_PRCP_McAllen),
    Imputed_PRCP_McAllen_Airport = ifelse(is.na(Adjusted_PRCP_McAllen_Airport),
                                          ifelse(all(is.na(c(Adjusted_PRCP_Harlingen,
Adjusted_PRCP_McAllen, Adjusted_PRCP_McAllen_Airport, Adjusted_PRCP_Weslaco))),
NA,
                                                 mean(c(Adjusted_PRCP_Harlingen, Adjusted_PRCP_McAllen,
Adjusted_PRCP_Weslaco), na.rm = TRUE)),
                                          Adjusted_PRCP_McAllen_Airport),
    Imputed_PRCP_Weslaco = ifelse(is.na(Adjusted_PRCP_Weslaco),
                                  ifelse(all(is.na(c(Adjusted_PRCP_Harlingen, Adjusted_PRCP_McAllen,
Adjusted_PRCP_McAllen_Airport, Adjusted_PRCP_Weslaco))), NA,
                                         mean(c(Adjusted_PRCP_Harlingen, Adjusted_PRCP_McAllen,
Adjusted_PRCP_McAllen_Airport), na.rm = TRUE)),
                                  Adjusted_PRCP_Weslaco),
    Imputed_TMAX_Harlingen = ifelse(is.na(Adjusted_TMAX_Harlingen),
                                    ifelse(all(is.na(c(Adjusted_TMAX_Harlingen,
Adjusted_TMAX_McAllen, Adjusted_TMAX_McAllen_Airport, Adjusted_TMAX_Weslaco))),
NA,
                                           mean(c(Adjusted_TMAX_McAllen,
Adjusted_TMAX_McAllen_Airport, Adjusted_TMAX_Weslaco), na.rm = TRUE)),
```

```
                          Adjusted_TMAX_Harlingen),
    Imputed_TMAX_McAllen = ifelse(is.na(Adjusted_TMAX_McAllen),
                          ifelse(all(is.na(c(Adjusted_TMAX_Harlingen, Adjusted_TMAX_McAllen,
Adjusted_TMAX_McAllen_Airport, Adjusted_TMAX_Weslaco))), NA,
                                 mean(c(Adjusted_TMAX_Harlingen,
Adjusted_TMAX_McAllen_Airport, Adjusted_TMAX_Weslaco), na.rm = TRUE)),
                          Adjusted_TMAX_McAllen),
    Imputed_TMAX_McAllen_Airport = ifelse(is.na(Adjusted_TMAX_McAllen_Airport),
                                  ifelse(all(is.na(c(Adjusted_TMAX_Harlingen,
Adjusted_TMAX_McAllen, Adjusted_TMAX_McAllen_Airport, Adjusted_TMAX_Weslaco))),
NA,
                                         mean(c(Adjusted_TMAX_Harlingen, Adjusted_TMAX_McAllen,
Adjusted_TMAX_Weslaco), na.rm = TRUE)),
                                  Adjusted_TMAX_McAllen_Airport),
    Imputed_TMAX_Weslaco = ifelse(is.na(Adjusted_TMAX_Weslaco),
                          ifelse(all(is.na(c(Adjusted_TMAX_Harlingen, Adjusted_TMAX_McAllen,
Adjusted_TMAX_McAllen_Airport, Adjusted_TMAX_Weslaco))), NA,
                                 mean(c(Adjusted_TMAX_Harlingen, Adjusted_TMAX_McAllen,
Adjusted_TMAX_McAllen_Airport), na.rm = TRUE)),
                          Adjusted_TMAX_Weslaco)
  ) %>%
  ungroup()


############################################################

### TIME SERIES ANALYSIS ###

# First, get the mean for each day, then the mean for each month and make time series.
# Ignore rows where all values are NA

# Step 1: Calculate row-wise means excluding McAllen Airport
pivoted_data <- pivoted_data %>%
  rowwise() %>%
  mutate(
    Mean_PRCP = ifelse(
      all(is.na(c(Imputed_PRCP_Harlingen, Imputed_PRCP_McAllen,
Imputed_PRCP_Weslaco))),
      NA,
      mean(c(Imputed_PRCP_Harlingen, Imputed_PRCP_McAllen, Imputed_PRCP_Weslaco),
na.rm = TRUE)
    ),
    Mean_TMAX = ifelse(
      all(is.na(c(Imputed_TMAX_Harlingen, Imputed_TMAX_McAllen,
Imputed_TMAX_Weslaco))),
      NA,
```

```
    mean(c(Imputed_TMAX_Harlingen, Imputed_TMAX_McAllen,
Imputed_TMAX_Weslaco), na.rm = TRUE)
   )
 ) %>%
 ungroup()

# Step 2: Create a time series for monthly mean PRCP and TMAX
# Convert DATE to yearmon format for aggregation by month
pivoted_data$YearMonth <- as.yearmon(pivoted_data$DATE)

# Aggregate to calculate the mean PRCP and TMAX for each month
monthly_data <- pivoted_data %>%
 group_by(YearMonth) %>%
 summarise(
   Monthly_Mean_PRCP = mean(Mean_PRCP, na.rm = TRUE),
   Monthly_Mean_TMAX = mean(Mean_TMAX, na.rm = TRUE)
 ) %>%
 ungroup()

# Create time series objects
monthly_prcp_ts <- ts(monthly_data$Monthly_Mean_PRCP, start =
c(year(min(pivoted_data$DATE)), month(min(pivoted_data$DATE))), frequency = 12)
monthly_tmax_ts <- ts(monthly_data$Monthly_Mean_TMAX, start =
c(year(min(pivoted_data$DATE)), month(min(pivoted_data$DATE))), frequency = 12)

# Plot the time series
autoplot(monthly_prcp_ts) +
 labs(caption="Caption", x="Year", y="Mean daily precipitation (inches)")

# Create boxplots to view seasons
monthly_prcp_df <- data.frame(
 Date = as.Date(time(monthly_prcp_ts)),
 Precipitation = as.numeric(monthly_prcp_ts)
)
monthly_prcp_df$Month <- factor(format(monthly_prcp_df$Date, "%b"), levels = month.abb)
ggplot(monthly_prcp_df, aes(x = Month, y = Precipitation)) +
 geom_boxplot(fill='lightblue') +
 labs(x = "Month",
      y = "Average daily precipitation (inches)",
      caption="Figure 2. Boxplot displaying monthly distribution of precipitation.")

# Decompose the monthly PRCP time series using STL
stl_prcp <- stl(monthly_prcp_ts, s.window = "periodic", t.window=201) # Parameters need to be
odd
autoplot(stl_prcp) +
```

```
  labs(caption="Figure 1. STL decomposition of monthly mean precipitation for Rio Grande
Valley, 1941-2024.")

# Assess autocorrelation among the random component after removing trend and seasonality.
remainder_component <- stl_prcp$time.series[, "remainder"] # Extract the remainder component
ggAcf(remainder_component) +
  labs(title="",
       caption="Figure 4. Correlogram showing ACF or remainder component after trend and
seasonality have been removed.")

# Measure strength of trend and seasonality (final slide of lecture 2)

trend_component <- stl_prcp$time.series[, "trend"] # Extract components
seasonal_component <- stl_prcp$time.series[, "seasonal"]
remainder_component <- stl_prcp$time.series[, "remainder"]

var_remainder <- var(remainder_component, na.rm = TRUE) # Calculate variances
var_trend_plus_remainder <- var(trend_component + remainder_component, na.rm = TRUE)
var_seasonal_plus_remainder <- var(seasonal_component + remainder_component, na.rm =
TRUE)

Ft <- max(0, 1 - (var_remainder / var_trend_plus_remainder)) # Calculate strength of trend

Fs <- max(0, 1 - (var_remainder / var_seasonal_plus_remainder)) # Calculate strength of
seasonality

cat("Strength of Trend (Ft):", Ft, "\n")
cat("Strength of Seasonality (Fs):", Fs, "\n")

# ACF and partial ACF
ggAcf(monthly_prcp_ts) +
  labs(title="")
ggPacf(monthly_prcp_ts) +
  labs(title = "", caption = "Figure 5. Partial autocorrelation function of monthly precipitation.")

### ARIMA ###

# ndiffs and nsdiffs to determine the number of differences
ndiffs(monthly_prcp_ts) # 0
nsdiffs(monthly_prcp_ts) # 0

my_arima <- auto.arima(monthly_prcp_ts, approximation = FALSE, stepwise = FALSE) #
Parameters cause slower, more thorough search.

# Check residuals of ARIMA model by plotting residuals and Portmanteau test (Ljung-Box)
residuals <- residuals(my_arima)
```

```
ggAcf(residuals) +
  labs(title = "ACF of Residuals")
ggPacf(residuals) +
  labs(title = "PACF of Residuals")
autoplot(residuals) +
  labs(title = "Residuals of ARIMA Model", x = "Time", y = "Residuals")
Box.test(residuals, lag = 20, type = "Ljung-Box")

# Forecast next 5 years
forecast_horizon <- 60
forecast <- forecast(my_arima, h = forecast_horizon)

# Filter the time series to include only data from 2010 onwards for plotting
start_year <- 2010
filtered_ts <- window(monthly_prcp_ts, start = c(start_year, 1))

# Plot the filtered time series and the forecast
autoplot(filtered_ts, series = "Observed", color='darkgrey') +
  autolayer(forecast, series = "Forecast") +
  labs(title = "",
       x = "Year",
       y = "Mean Daily Precipitation (inches)",
       caption = "Figure 6. Forecast of precipitation for next 5 years (60 months).")

## Spectral Analysis

spec <- spec.pgram(monthly_prcp_ts, plot = FALSE) # Periodogram
plot(spec, main = "Periodogram of Monthly Precipitation", xlab = "Frequency", ylab = "Spectral
Density")

# Smoothed periodogram with Daniell kernel
spec_smooth <- spec.pgram(monthly_prcp_ts, spans = c(13, 13), taper = 0.1, log = "no")

# Extract the frequency and spectral density
freq <- spec_smooth$freq
spec_density <- spec_smooth$spec

# Create a data frame for ggplot2
spec_data <- tibble(Frequency = freq, Spectral_Density = spec_density)

ggplot(spec_data, aes(x = Frequency, y = Spectral_Density)) +
  geom_line() +
  labs(title = "",
       x = "Frequency",
       y = "Spectral Density",
```

```
      caption = "Figure 3. Periodogram of monthly precipitation time series smoothed using
Daniell kernel.")

# Get confidence intervals at frequencies 1, 2, 3, 4, 5.
freqs_of_interest <- c(1, 2, 3, 4, 5) / 12  # Convert to frequency in cycles per month
indices <- sapply(freqs_of_interest, function(f) which.min(abs(freq - f)))
spec_values <- spec_density[indices]

dof <- 2 * 13  # 2 times the smoothing span, spans = c(13, 13)

ci_multiplier <- qchisq(c(0.025, 0.975), df = dof) / dof
ci_lower <- spec_values * ci_multiplier[1]
ci_upper <- spec_values * ci_multiplier[2]

result <- tibble(Frequency = freqs_of_interest * 12,  # Convert back to original scale
                 Spectral_Density = spec_values,
                 CI_Lower = ci_lower,
                 CI_Upper = ci_upper)
print(result)
```