Data Science Project

AUTHOR
Jin Sook Song, Daniel S Tapp

DATA 613

Group Project

Jin Sook Song, Daniel S Tapp

USGS Water Data Analysis

Using the U.S. Geological Survey (USGS) data (https://water.usgs.gov/owq/data.html), our group analyzed the **Seasonal Effects on Discharge and Water Temperature in the Potomac River during 2015-2024.**

Our research questions include:

- **Is there a significant difference in river discharge across seasons?**
- **Has river discharge changed over 10 years (2015-2024)?**
- **Is summer discharge lower than other seasons?**
- **Does water temperature correlate with discharge?**

1. API Querying

```
# install.packages(c("dataRetrieval", "tidyverse", "lubridate", "broom"))
```

```
# Load packages
library(dataRetrieval)
library(tidyverse)
```

```
── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
✔ dplyr     1.1.4     ✔ readr     2.1.5
✔ forcats   1.0.0     ✔ stringr   1.5.1
✔ ggplot2   3.5.1     ✔ tibble    3.2.1
✔ lubridate 1.9.4     ✔ tidyr     1.3.1
✔ purrr     1.0.4
── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(lubridate)
library(broom)
```

```
# Get USGS data
siteNumber <- "01646500"              # site: Potomac River at Point of Rocks, MD
parameterCd <- "00060"                # River discharge
startDate <- "2015-01-01"
endDate <- "2024-12-31"

# Use USGS API to fetch daily mean discharge data
raw_data <- readNWISdv(siteNumber, parameterCd, startDate, endDate)
```

```
GET:https://waterservices.usgs.gov/nwis/dv/?
site=01646500&format=waterml%2C1.1&ParameterCd=00060&StatCd=00003&startDT=2015-01-01&endDT=2024-12-31
```

```r
        # Preview available columns
        names(raw_data)
```

```
[1] "agency_cd"        "site_no"          "Date"             "X_00060_00003"
[5] "X_00060_00003_cd"
```

2. Preprocess the Data

```r
        # Clean and prep data
        flow_data <- raw_data %>%
          rename(
            date = Date,
            discharge_cfs = X_00060_00003  # mean daily discharge
          ) %>%
          mutate(
            year = year(date),
            month = month(date),
            season = case_when(
              month %in% c(12, 1, 2) ~ "Winter",
              month %in% c(3, 4, 5) ~ "Spring",
              month %in% c(6, 7, 8) ~ "Summer",
              month %in% c(9, 10, 11) ~ "Fall"
            )
          ) %>%
          filter(!is.na(discharge_cfs))  # remove missing data
```

Note: In the context of rivers, cfs stands for cubic feet per second, a unit used to measure the volume of water flowing past a specific point in a river within one second. It essentially quantifies the river's discharge or flow rate.

3. Analysis

1. ANOVA analysis: **Is there a significant difference in river discharge across seasons?**

```r
        # 1. ANOVA: River discharge across seasons
        anova_result <- aov(discharge_cfs ~ season, data = flow_data)
        summary(anova_result)
```

```
              Df    Sum Sq   Mean Sq F value Pr(>F)
season         3 6.777e+10 2.259e+10   136.4 <2e-16 ***
Residuals   3649 6.043e+11 1.656e+08
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Hypothesis:

- H0: All seasons have the same mean river discharge.
- H1: There is a significant difference in mean river discharge across seasons.

Since p-value < 2e-16 which is smaller than 0.05, we reject the null hypothesis at the 95% confidence level. There is a statistically significant difference in mean river discharge across seasons.

2. T-test: Is summer discharge lower than other seasons?

```r
# 2. T-test: Is summer discharge lower than other seasons?

# Group into Summer vs Other Seasons
flow_data <- flow_data %>%
  mutate(season_group = if_else(season == "Summer", "Summer", "Other"))

# Perform t-test
t_test_result <- t.test(discharge_cfs ~ season_group, data = flow_data)
t_test_result
```

```
    Welch Two Sample t-test

data:  discharge_cfs by season_group
t = 13.04, df = 2230, p-value < 2.2e-16
alternative hypothesis: true difference in means between group Other and group Summer is not equal to 0
95 percent confidence interval:
 4771.756 6460.958
sample estimates:
 mean in group Other mean in group Summer
           12761.464             7145.108
```

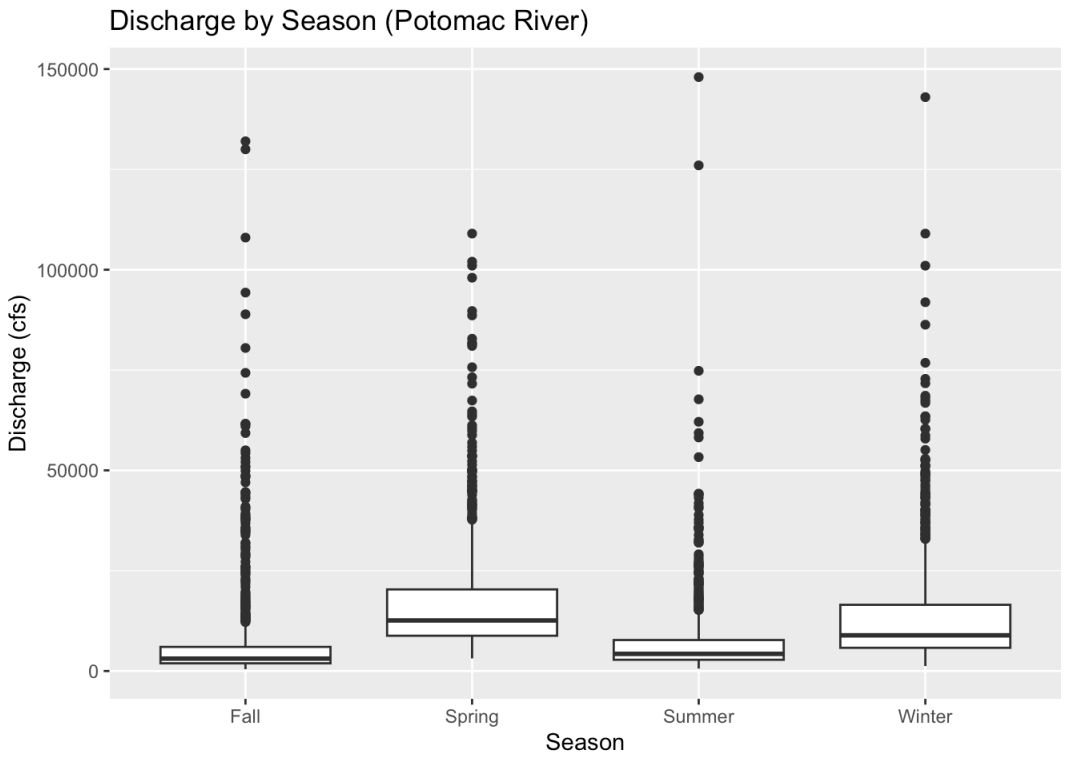
We grouped the data in to Summer and other seasons (Spring, Fall, Winter), then used a Welch t-test to compare mean river discharge between the two groups.

Hypothesis:

- H0: There is no significant difference in mean river discharges between Summer and other seasons.
- H1: There is a significant difference in mean river discharges between Summer and other seasons.

The p-value is < 2.2e-16, so we reject the null hypothesis at the 95% confidence level. There is a statistically significant difference in mean river discharges between Summer and other seasons.

```r
# Boxplot of Seasonal Discharge
ggplot(flow_data, aes(x = season, y = discharge_cfs)) +
  geom_boxplot() +
  labs(title = "Discharge by Season (Potomac River)",
       y = "Discharge (cfs)", x = "Season")
```



The above plot shows the average discharge of Potomac river by season. It compares daily discharge distributions across Spring, Summer, Fall and Winter.

According to the plot, the river discharge varies by season. Spring has the highest flow overall. Summer has the lowest, consistent with your t-test results showing a significant drop in flow during summer months.

3. Regression Analysis

3.1. Simple Regression: Has river discharge changed over 10 years (2015-2024)?

```
# 3.1. Regression: Has river discharge changed over 10 years?

# Aggregate by year
yearly_avg <- flow_data %>%
  group_by(year) %>%
  summarize(mean_discharge = mean(discharge_cfs, na.rm = TRUE))

# Linear model: discharge ~ year
reg_model <- lm(mean_discharge ~ year, data = yearly_avg)
summary(reg_model)
```

```
Call:
lm(formula = mean_discharge ~ year, data = yearly_avg)

Residuals:
    Min      1Q  Median      3Q     Max
-4136.2 -3389.1 -1285.4   530.7 13790.5

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  932348.9  1266295.9   0.736    0.483
year           -456.1      627.0  -0.727    0.488

Residual standard error: 5695 on 8 degrees of freedom
```

```
Multiple R-squared:  0.06202,   Adjusted R-squared:  -0.05522
F-statistic: 0.529 on 1 and 8 DF,  p-value: 0.4878
```

We've tested whether the average yearly discharge has changed over time, and whether there is a linear trend from 2015 to 2024.

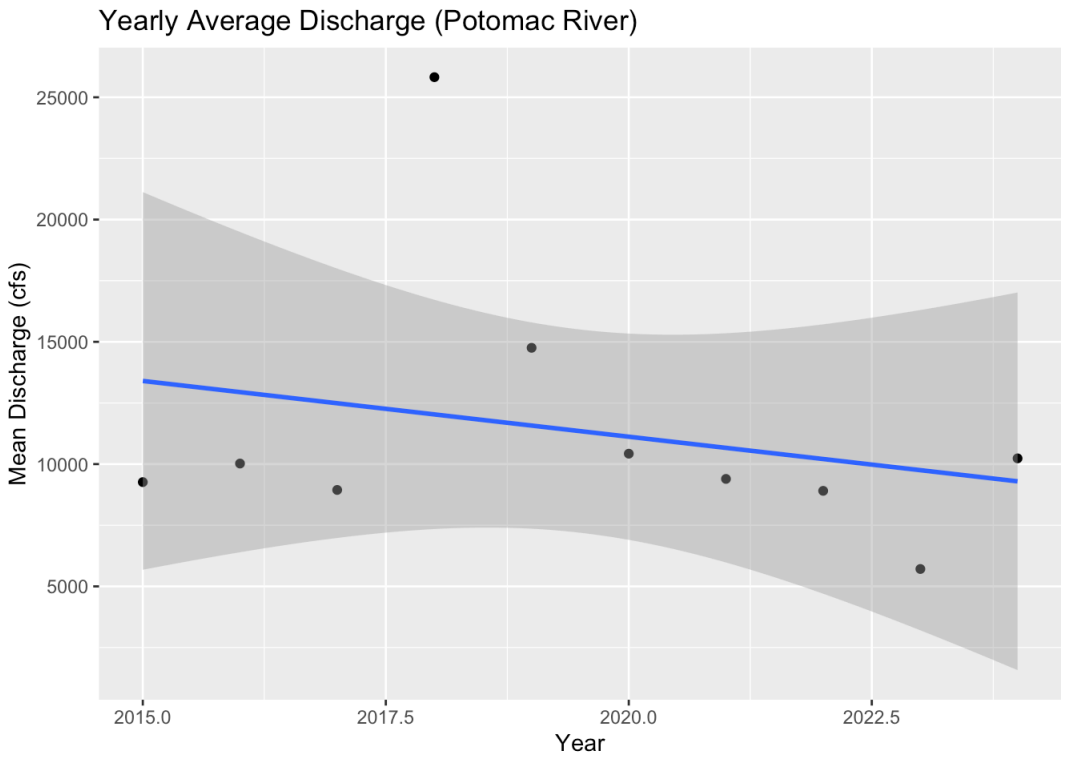
Hypothesis:

- H0: There is no significant trend in river discharge over time (2015-2024).
- H1: There is a significant trend in river discharge over time (2015-2024).

The p-value for year is 0.488 > 0.05, which is **not statistically significant** at the 95% confidence level. Therefore we fail to reject the null hypothesis, and there is no statistically significant trend in discharge over time (2015-2024).

```
        # Plot
        ggplot(yearly_avg, aes(x = year, y = mean_discharge)) +
          geom_point() +
          geom_smooth(method = "lm", se = TRUE) +
          labs(title = "Yearly Average Discharge (Potomac River)",
               y = "Mean Discharge (cfs)", x = "Year")
```

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



The above regression plot shows the yearly average discharge of Potomac river.

There appears to be a slight downward trend in yearly average discharge over the last 10 years, but it's not statistically significant. Discharge varies a lot year-to-year.

3.2. Simple Regression: Has river discharge affected by water temperature (2015-2024)?

```
        # Load libraries
        library(tidyverse)
        library(lubridate)
        library(dataRetrieval)
        library(broom)

        # === 1. GET DATA ===
```

```
siteNumber <- "01646500"
startDate <- "2015-01-01"
endDate <- "2024-12-31"

# Discharge: parameter code 00060
discharge_data <- readNWISdv(siteNumber, "00060", startDate, endDate)
```

GET:https://waterservices.usgs.gov/nwis/dv/?
site=01646500&format=waterml%2C1.1&ParameterCd=00060&StatCd=00003&startDT=2015-01-01&endDT=2024-12-31

```
# Temperature: parameter code 00010
temperature_data <- readNWISdv(siteNumber, "00010", startDate, endDate)
```

GET:https://waterservices.usgs.gov/nwis/dv/?
site=01646500&format=waterml%2C1.1&ParameterCd=00010&StatCd=00003&startDT=2015-01-01&endDT=2024-12-31

```
# === 2. CLEAN AND PREP ===

# Discharge
flow_data <- discharge_data %>%
  rename(date = Date, discharge_cfs = X_00060_00003) %>%
  mutate(
    year = year(date),
    month = month(date),
    season = case_when(
      month %in% c(12, 1, 2) ~ "Winter",
      month %in% c(3, 4, 5) ~ "Spring",
      month %in% c(6, 7, 8) ~ "Summer",
      month %in% c(9, 10, 11) ~ "Fall"
    )
  ) %>%
  filter(!is.na(discharge_cfs))

# Temperature
temperature_data_cleaned <- temperature_data |>
  rename(
    "4_1_ft_C" = "X_4.1.ft.from.riverbed..middle....Discontinued._00010_00003",
    "1_ft_C" = "X_1.0.ft.from.riverbed..bottom....Discontinued._00010_00003",
    "7_1_ft_C" = "X_7.1.ft.from.riverbed..top....Discontinued._00010_00003",
    "old_multiparameter_C" = "X_From.multiparameter.sonde...Discontinued._00010_00003",
    "current_multiparameter_C" = "X_From.multiparameter.sonde_00010_00003",
    "date" = "Date"
  ) |>
  select("date", "4_1_ft_C", "1_ft_C", "7_1_ft_C",
         "old_multiparameter_C", "current_multiparameter_C")

avg_temp <- transform(temperature_data_cleaned,
                      avg_temp_C = rowMeans(temperature_data_cleaned[,-1], na.rm = TRUE))

temperature_data_final <- avg_temp |> select("date", "avg_temp_C")

# === 3. COMBINE ===
combined_data <- left_join(temperature_data_final, flow_data, by = "date") %>%
  filter(!is.na(avg_temp_C))
```

```
# === 4. SIMPLE LINEAR REGRESSION : Discharge ~ Temperature
temp_only_model <- lm(discharge_cfs ~ avg_temp_C, data = combined_data)
summary(temp_only_model)
```

Call:

```
lm(formula = discharge_cfs ~ avg_temp_C, data = combined_data)

Residuals:
   Min     1Q Median     3Q    Max
-15636  -6975  -3278   1830 137472

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 16979.52     438.96   38.68   <2e-16 ***
avg_temp_C   -350.62      23.74  -14.77   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 13260 on 3584 degrees of freedom
Multiple R-squared:  0.05735,   Adjusted R-squared:  0.05709
F-statistic:   218 on 1 and 3584 DF,  p-value: < 2.2e-16
```

We've tested whether the water temperature has effect on river discharge from 2015 to 2024.

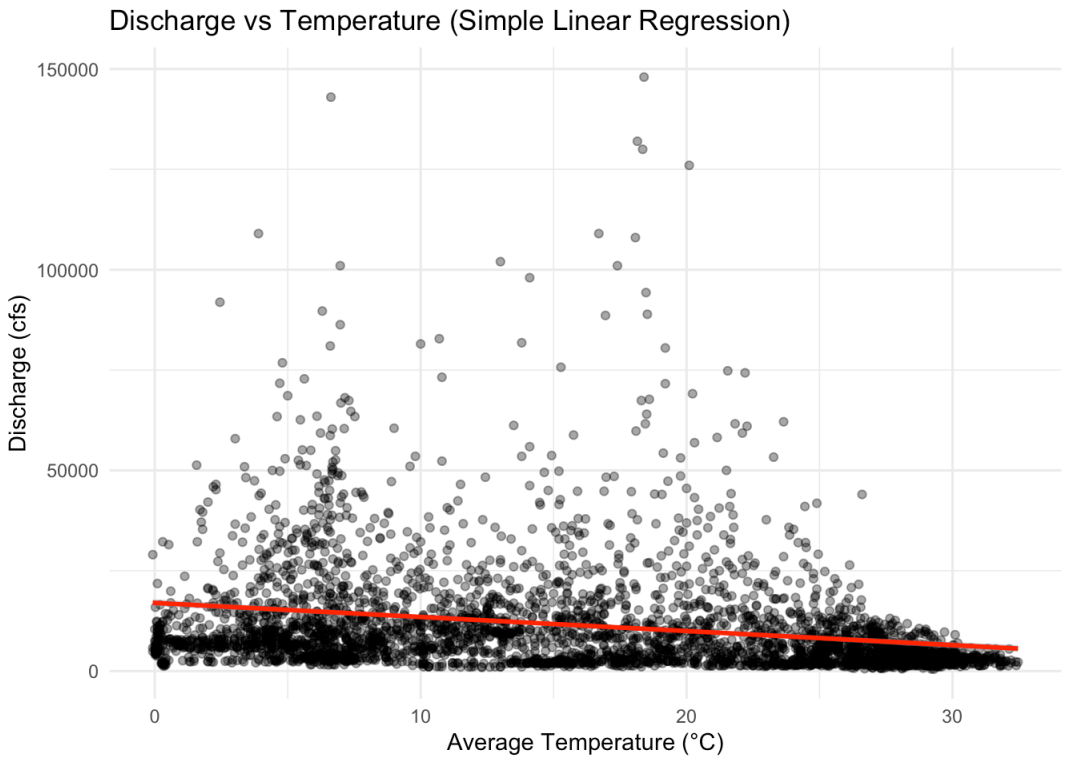
Hypothesis:

- H0: There is no linear relationship between water temperature and river discharge.
- H1: There is a significant linear relationship between water temperature and discharge.

The p-value < 2e-16 < 0.05, which is statistically significant at the 95% confidence level. Therefore we reject the null hypothesis, and there is a statistically significant relationship between temperature and river discharge (2015-2024). As water temperature increases, discharge tends to decrease.

```r
# Plot
ggplot(combined_data, aes(x = avg_temp_C, y = discharge_cfs)) +
  geom_point(alpha = 0.4) +
  geom_smooth(method = "lm", se = TRUE, color = "red") +
  labs(
    title = "Discharge vs Temperature (Simple Linear Regression)",
    x = "Average Temperature (°C)",
    y = "Discharge (cfs)"
  ) +
  theme_minimal()
```

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



3.3. Multiple Regression: How do both average water temperature and season affect the river discharge (2015-2024)?

```r
# === 4. MULTIPLE REGRESSION === Discharge ~ Temperature, season
multi_model <- lm(discharge_cfs ~ avg_temp_C + season, data = combined_data)
summary(multi_model)
```

```
Call:
lm(formula = discharge_cfs ~ avg_temp_C + season, data = combined_data)

Residuals:
   Min     1Q Median     3Q    Max
-17596  -6306  -3174    789 137463

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  13800.99     931.78  14.811  < 2e-16 ***
avg_temp_C    -372.81      47.47  -7.854 5.29e-15 ***
seasonSpring  8733.91     623.43  14.009  < 2e-16 ***
seasonSummer  3595.85     770.35   4.668 3.16e-06 ***
seasonWinter  1779.61     861.90   2.065    0.039 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 12850 on 3581 degrees of freedom
Multiple R-squared:  0.1143,    Adjusted R-squared:  0.1133
F-statistic: 115.5 on 4 and 3581 DF,  p-value: < 2.2e-16
```

We've tested whether the river temperatures and seasons have effect on river discharge from 2015 to 2024.

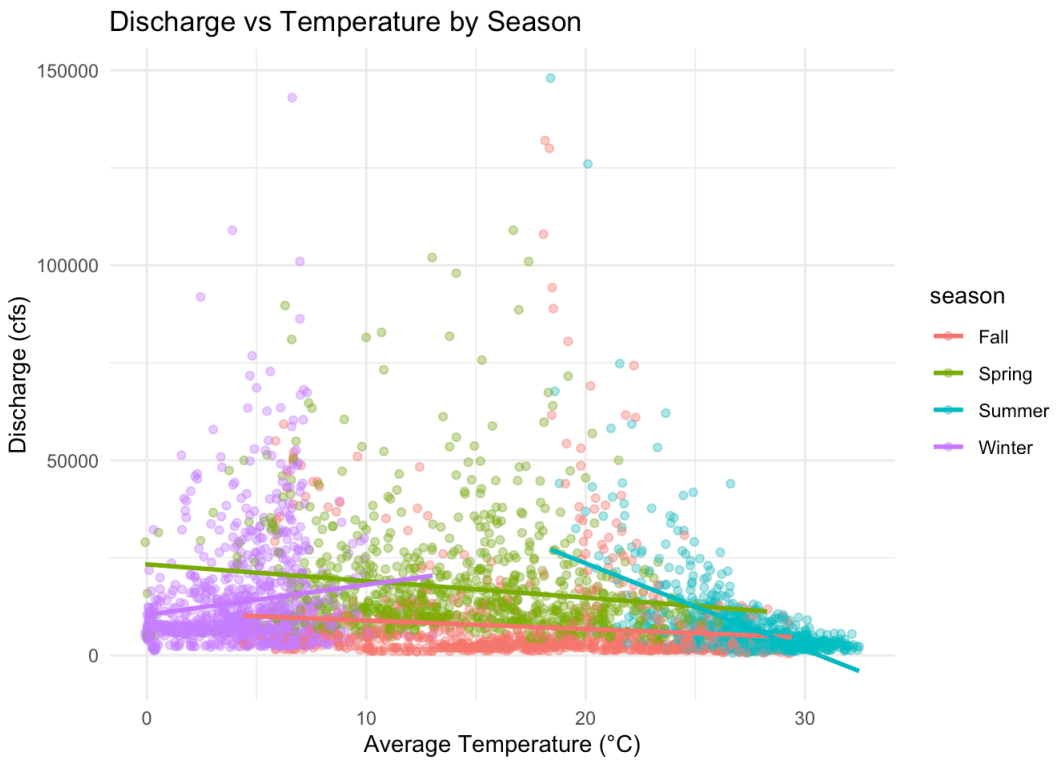
Hypothesis:

- H0: All regression coefficients = 0 (temperature and seasons have no effect on discharge)
- H1: At least one regression coefficient ≠ 0 (temperature and/or seasons affect discharge).

The p-value < 2.2e-16 < 0.05, which is statistically significant at the 95% confidence level. Therefore we reject the null hypothesis, and This model shows a **statistically significant** relationship between river discharge and both temperature and season. Notably, as temperature increases, discharge decreases (2015-2024).

```
# Plot
library(ggplot2)
ggplot(combined_data, aes(x = avg_temp_C, y = discharge_cfs, color = season)) +
  geom_point(alpha = 0.4) +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "Discharge vs Temperature by Season",
       x = "Average Temperature (°C)", y = "Discharge (cfs)") +
  theme_minimal()
```

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



4. Shiny App

```
# install.packages(c("shiny","bslib", "plotly", "DT"))
```

```
# Load packages
library(shiny)
library(bslib)
```

```
Attaching package: 'bslib'

The following object is masked from 'package:broom':

    bootstrap

The following object is masked from 'package:utils':

    page
```

```
        library(plotly)
```

```
Attaching package: 'plotly'

The following object is masked from 'package:ggplot2':

    last_plot

The following object is masked from 'package:stats':

    filter

The following object is masked from 'package:graphics':

    layout
```

```
        library(DT)
```

```
Attaching package: 'DT'

The following objects are masked from 'package:shiny':

    dataTableOutput, renderDataTable
```

```
        library(tidyverse)
        library(lubridate)
        library(dataRetrieval)
        library(broom)
```

```
        # Get the USGS data
        siteNumber <- "01646500"               # site: Potomac River at Point of Rocks, MD
        parameterCd <- "00060"                 # River discharge
        temp_pCode <- "00010"                  # Temperature
        startDate <- "2015-01-01"
        endDate <- "2024-12-31"

        # Use USGS API to fetch daily mean discharge data
        raw_data <- readNWISdv(siteNumber, parameterCd, startDate, endDate)
```

```
GET:https://waterservices.usgs.gov/nwis/dv/?
site=01646500&format=waterml%2C1.1&ParameterCd=00060&StatCd=00003&startDT=2015-01-01&endDT=2024-12-31
```

```
        # Use USGS API to fetch daily temperature data
        temperature_data <- readNWISdv(siteNumber,temp_pCode,startDate,endDate)
```

```
GET:https://waterservices.usgs.gov/nwis/dv/?
site=01646500&format=waterml%2C1.1&ParameterCd=00010&StatCd=00003&startDT=2015-01-01&endDT=2024-12-31
```

```
        # Clean and prep discharge data
        flow_data <- raw_data %>%
          rename(
            date = Date,
            discharge_cfs = X_00060_00003  # mean daily discharge
          ) %>%
          mutate(
            year = year(date),
            month = month(date),
            season = case_when(
```

```
      month %in% c(12, 1, 2) ~ "Winter",
      month %in% c(3, 4, 5) ~ "Spring",
      month %in% c(6, 7, 8) ~ "Summer",
      month %in% c(9, 10, 11) ~ "Fall"
    )
  ) %>%
  filter(!is.na(discharge_cfs))  # remove missing data
```

```
# Clean and prep temperature data
temperature_data_cleaned <- temperature_data |>
  rename(
    "4_1_ft_C" = "X_4.1.ft.from.riverbed..middle....Discontinued._00010_00003",
    "1_ft_C" = "X_1.0.ft.from.riverbed..bottom....Discontinued._00010_00003",
    "7_1_ft_C" = "X_7.1.ft.from.riverbed..top....Discontinued._00010_00003",
    "old_multiparameter_C" = "X_From.multiparameter.sonde...Discontinued._00010_00003",
    "current_multiparameter_C" = "X_From.multiparameter.sonde_00010_00003",
    "date" = "Date") |>
  select(
    "date","4_1_ft_C","1_ft_C","7_1_ft_C",
    "old_multiparameter_C","current_multiparameter_C"
  )

avg_temp <- transform(temperature_data_cleaned,avg_temp_C = rowMeans(temperature_data_cleaned[,-1],na.rm

temperature_data_final <- avg_temp |>
  select("date","avg_temp_C"
  )
```

```
# Combine temperature and discharge data
temp_and_discharge <- left_join(temperature_data_final, flow_data, by = "date")
```

```
# UI for the reactive graphs and tables
# === UI ===
ui <- fluidPage(
  theme = bs_theme(version = 5, bootswatch = "cosmo"),
  titlePanel("🌊 Potomac River Discharge – USGS Data Explorer"),

  sidebarLayout(
    sidebarPanel(
      selectInput("xvar", "X-axis Variable:", choices = c("Year" = "year", "Average Temperature (°C)" = "
      selectInput("yvar", "Y-axis Variable:", choices = c("Discharge (cfs)" = "discharge_cfs")),
      selectInput("seasonFilter", "Filter by Season:", choices = c("All", unique(flow_data$season)), sele
      dateRangeInput("dateRange", "Select Date Range:", start = min(flow_data$date), end = max(flow_data$
      checkboxInput("logY", "Log-transform Y-axis", value = FALSE),
      checkboxInput("runAnova", "Show ANOVA Summary", value = TRUE),
      checkboxInput("runRegression", "Show Custom Regression", value = TRUE),
      checkboxInput("runTTest", "Show T-Test (Summer vs Other)", value = TRUE)
    ),

    mainPanel(
      tabsetPanel(
        tabPanel("Scatterplot (Custom X vs Y)", plotlyOutput("scatterPlot")),
        tabPanel("Discharge by Season (Boxplot)", plotlyOutput("boxPlot")),
        tabPanel("Yearly Trend", plotlyOutput("trendPlot")),
        tabPanel("Simple Regression: Temp vs Discharge", plotlyOutput("tempOnlyPlot")),
        tabPanel("Multiple Regression: Temp + Season", plotlyOutput("multiRegPlot")),
        tabPanel("Data Table", DTOutput("dataTable")),
        tabPanel("Model Summaries",
                 h4("Custom X vs Y Regression"),
                 verbatimTextOutput("modelSummary"),
```

```
                h4("ANOVA: Discharge ~ Season"),
                verbatimTextOutput("anovaSummary"),
                h4("T-Test: Summer vs Other Seasons"),
                verbatimTextOutput("ttestSummary"),
                h4("Simple Regression: Discharge ~ Temperature"),
                verbatimTextOutput("tempOnlyModelSummary"),
                h4("Multiple Regression: Discharge ~ Temperature + Season"),
                verbatimTextOutput("multiModelSummary")
        )
      )
    )
  )
)
```

```
# Server
server <- function(input, output, session) {

  # === Filtered Data ===
  filteredData <- reactive({
    req(input$dateRange)
    df <- temp_and_discharge %>%
      filter(date >= input$dateRange[1] & date <= input$dateRange[2])
    if (input$seasonFilter != "All") {
      df <- df %>% filter(season == input$seasonFilter)
    }
    df
  })

  # === Scatterplot with Custom X/Y ===
  output$scatterPlot <- renderPlotly({
    df <- filteredData()
    p <- ggplot(df, aes_string(x = input$xvar, y = input$yvar)) +
      geom_point(alpha = 0.5) +
      geom_smooth(method = "lm", se = TRUE, color = "blue") +
      scale_y_continuous(trans = ifelse(input$logY, "log10", "identity")) +
      labs(title = "Scatterplot with Regression Line")
    ggplotly(p)
  })

  # === Boxplot: Discharge by Season ===
  output$boxPlot <- renderPlotly({
    df <- filteredData()
    p <- ggplot(df, aes(x = season, y = discharge_cfs)) +
      geom_boxplot(fill = "steelblue") +
      labs(title = "Discharge by Season", y = "Discharge (cfs)", x = "Season")
    ggplotly(p)
  })

  # === Trend Plot: Yearly Average Discharge ===
  output$trendPlot <- renderPlotly({
    df <- filteredData() %>%
      group_by(year) %>%
      summarize(mean_discharge = mean(discharge_cfs, na.rm = TRUE))
    p <- ggplot(df, aes(x = year, y = mean_discharge)) +
      geom_point() +
      geom_smooth(method = "lm", se = TRUE) +
      labs(title = "Yearly Average Discharge", x = "Year", y = "Mean Discharge (cfs)")
    ggplotly(p)
  })

  # === Simple Regression: Discharge ~ Temperature ===
```

```r
  output$tempOnlyPlot <- renderPlotly({
    df <- filteredData()
    p <- ggplot(df, aes(x = avg_temp_C, y = discharge_cfs)) +
      geom_point(alpha = 0.5) +
      geom_smooth(method = "lm", se = TRUE, color = "blue") +
      labs(title = "Discharge vs Temperature (Simple Regression)", x = "Average Temperature (°C)", y = "D
      theme_minimal()
    ggplotly(p)
  })

  output$tempOnlyModelSummary <- renderPrint({
    df <- filteredData()
    model <- lm(discharge_cfs ~ avg_temp_C, data = df)
    summary(model)
  })

  # === Multiple Regression: Discharge ~ Temp + Season ===
  output$multiRegPlot <- renderPlotly({
    df <- filteredData()
    p <- ggplot(df, aes(x = avg_temp_C, y = discharge_cfs, color = season)) +
      geom_point(alpha = 0.5) +
      geom_smooth(method = "lm", se = FALSE) +
      labs(title = "Discharge vs Temperature by Season (Multiple Regression)", x = "Average Temperature (
      theme_minimal()
    ggplotly(p)
  })

  output$multiModelSummary <- renderPrint({
    df <- filteredData()
    model <- lm(discharge_cfs ~ avg_temp_C + season, data = df)
    summary(model)
  })

  # === ANOVA: Discharge by Season ===
  output$anovaSummary <- renderPrint({
    req(input$runAnova)
    df <- filteredData()
    model <- aov(discharge_cfs ~ season, data = df)
    summary(model)
  })

  # === T-Test: Summer vs Other ===
  output$ttestSummary <- renderPrint({
    req(input$runTTest)
    df <- filteredData() %>%
      mutate(season_group = if_else(season == "Summer", "Summer", "Other"))
    t.test(discharge_cfs ~ season_group, data = df)
  })

  # === Custom Regression (X vs Y from dropdown) ===
  output$modelSummary <- renderPrint({
    req(input$runRegression)
    df <- filteredData()
    model <- lm(as.formula(paste(input$yvar, "~", input$xvar)), data = df)
    summary(model)
  })

  # === Data Table ===
  output$dataTable <- renderDT({
    df <- filteredData()
    datatable(df, options = list(pageLength = 10), filter = "top")
  })
}
```

```
# Run the app
shinyApp(ui = ui, server = server)
```

Shiny applications not supported in static R Markdown documents