Hydrology Assignment 4: Water Supply Intake Structure at Clayton, NC

Emma Kaufman

2024-11-25

```r
# load packages
library(readxl)
library(dplyr)
```

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

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

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

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v readr     2.1.5
## v ggplot2   3.5.1     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(zoo)
```

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

```r
library(ggplot2)
library(knitr)
library(tidyr)


# read in raw data
Falls_raw <- read_csv("Data/neuse_river_at_falls_02087183_daily_flow_updated.csv",
    col_types = cols(site_no = col_factor(levels = c("2087183")),
        Date = col_date(format = "%m/%d/%Y")),
    na = "NA")
```

```
## Warning: One or more parsing issues, call ‘problems()‘ on your data frame for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)
```

```r
Clayton_raw <- read_csv("Data/neuse_river_at_clayton_02087500_daily_flow_updated.csv",
    col_types = cols(site_no = col_factor(levels = c("2087500")),
        Date = col_date(format = "%m/%d/%Y")),
    na = "NA")
```

```
## Warning: One or more parsing issues, call ‘problems()‘ on your data frame for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)
```

Begin by examining the data at Clayton, only column with NA's is daily max discharge. Data for this column is available starting 2004-10-01. Also look to see if the average daily discharge is ever 0 and it isn't

```r
summary(Clayton_raw)
```

```
##   agency_cd            site_no           Date
##  Length:35546       2087500:35546   Min.   :1927-08-01
##  Class :character                   1st Qu.:1951-11-29
##  Mode  :character                   Median :1976-03-28
##                                     Mean   :1976-03-28
##                                     3rd Qu.:2000-07-26
##                                     Max.   :2024-11-24
##
##  daily_max_discharge_cfs daily_mean_discharge_cfs USGS_QA_code
##  Min.   :  166           Min.   :   45            Length:35546
##  1st Qu.:  329           1st Qu.:  309            Class :character
##  Median :  569           Median :  540            Mode  :character
##  Mean   : 1185           Mean   : 1133
##  3rd Qu.: 1490           3rd Qu.: 1240
##  Max.   :20200           Max.   :22500
##  NA's   :30101
##       year           month              day
##  Min.   :1927   Min.   : 1.000   Min.   : 1.00
##  1st Qu.:1951   1st Qu.: 4.000   1st Qu.: 8.00
##  Median :1976   Median : 7.000   Median :16.00
##  Mean   :1976   Mean   : 6.533   Mean   :15.73
##  3rd Qu.:2000   3rd Qu.:10.000   3rd Qu.:23.00  
##  Max.   :2024   Max.   :12.000   Max.   :31.00  
##
```

```
sum(Clayton_raw$daily_mean_discharge_cfs == 0)
```

```
## [1] 0
```

```
tail(Clayton_raw)
```

```
## # A tibble: 6 x 9
##   agency_cd site_no Date       daily_max_discharge_cfs daily_mean_discharge_cfs
##   <chr>     <fct>   <date>                       <dbl>                    <dbl>
## 1 USGS      2087500 2024-11-19                      NA                      419
## 2 USGS      2087500 2024-11-20                      NA                      403
## 3 USGS      2087500 2024-11-21                      NA                      439
## 4 USGS      2087500 2024-11-22                      NA                      387
## 5 USGS      2087500 2024-11-23                      NA                      359
## 6 USGS      2087500 2024-11-24                      NA                      345
## # i 4 more variables: USGS_QA_code <chr>, year <dbl>, month <dbl>, day <dbl>
```

Only work with mean daily discharge data from 1981-2023 (don't have the entire year of data for 2024, so go through 2023). Take the weekly rolling average of the daily mean discharge. Find the maximum of this weekly average for each year, then rank them within the two timeframes and find recurrence intervals.

```
# create column of weekly average
Clayton_1981_2023 <- Clayton_raw %>%
  filter(year >= 1981 & year <= 2023) %>%
  mutate(weekly_avg_Q_cfs = rollmean(daily_mean_discharge_cfs,
                               k= 7, fill= NA, align = "right")) %>%
   mutate(Timeframe = as.factor(ifelse(year>=2001,"2001-2023","1981-2000")))

# write to excel
write.csv(Clayton_1981_2023, "Data/Clayton_7_avg_flow_1981_2023.csv", row.names = FALSE)


# find yearly maximum
Clayton_yearly_max <- Clayton_1981_2023 %>%
  slice(-1, -2, -3, -4, -5, -6) %>% #skip first three rows bc NA
  group_by(year, Timeframe) %>%
  summarize(yearly_max_Q_cfs = max(weekly_avg_Q_cfs))
```

```
## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```

```
# Rank yearly maximums
# split up rankings by time period, before and after intake design
Clayton_max_recurrence2 <- Clayton_yearly_max %>%
  filter(Timeframe == '1981-2000') %>%
  arrange(desc(yearly_max_Q_cfs))
```

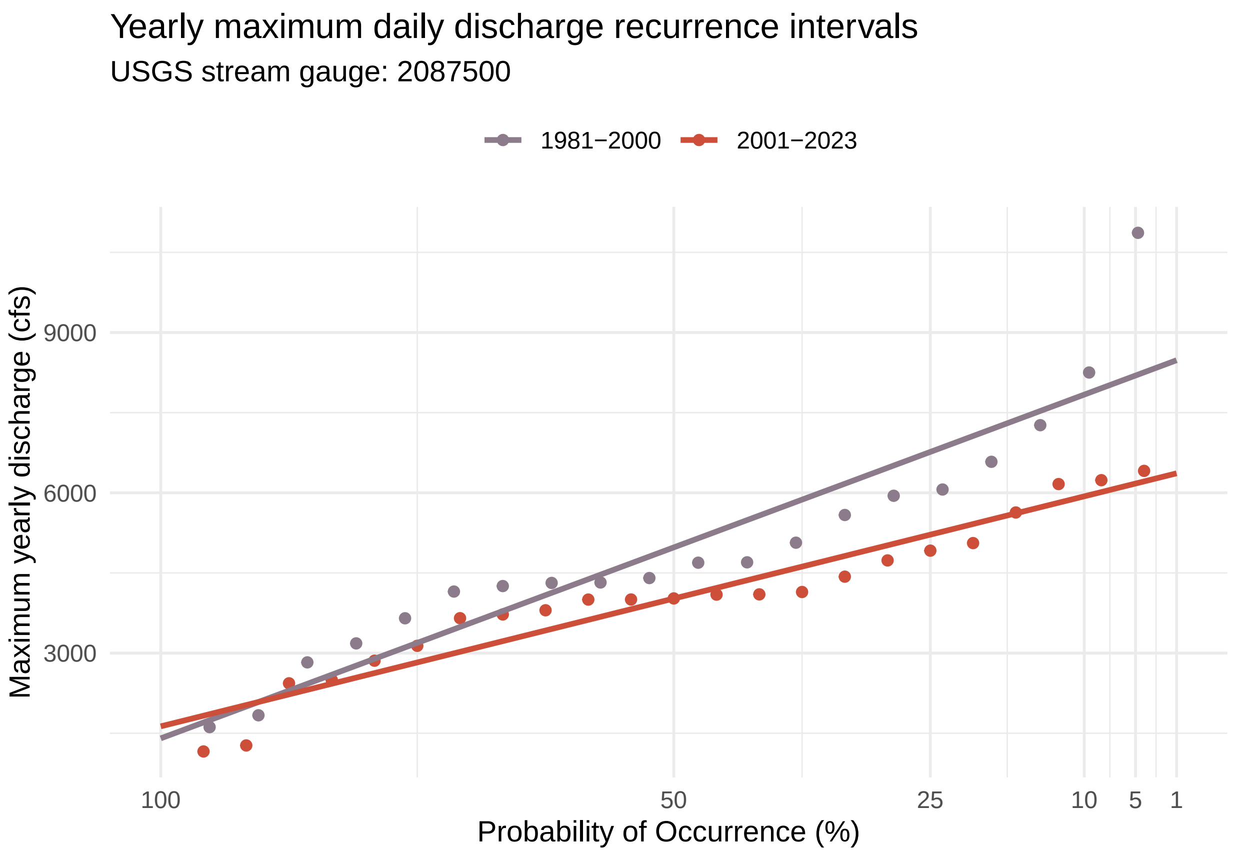
```
Clayton_max_recurrence_81_2000 <- Clayton_max_recurrence2 %>%
  ungroup() %>%
  mutate(
    Rank = rank(-yearly_max_Q_cfs),
    weibull_return = ((n()+1)/Rank),
    #weibull_return_24_year = weibull_return_24hr/12,
    weibull_percent_likelihood=(1/weibull_return)*100,
    #weibull_percent_likelihood_yr=(1/weibull_return_24_year)*100,
    log_prob_occurence_weibull = log10(weibull_percent_likelihood)
  )

Clayton_max_recurrence3 <- Clayton_yearly_max %>%
  filter(Timeframe == '2001-2023') %>%
  arrange(desc(yearly_max_Q_cfs))

Clayton_max_recurrence_2001_2023 <- Clayton_max_recurrence3 %>%
  ungroup() %>%
  mutate(
    Rank = rank(-yearly_max_Q_cfs),
    weibull_return = ((n()+1)/Rank),
    #weibull_return_24_year = weibull_return_24hr/12,
    weibull_percent_likelihood=(1/weibull_return)*100,
    #weibull_percent_likelihood_yr=(1/weibull_return_24_year)*100,
    log_prob_occurence_weibull = log10(weibull_percent_likelihood)
  )

# Combine the two dataframes
Clayton_max_recurrence_combined <- bind_rows(
  Clayton_max_recurrence_81_2000,
  Clayton_max_recurrence_2001_2023
)

# find yearly minimums
Clayton_yearly_min <- Clayton_1981_2023 %>%
  slice(-1, -2, -3, -4, -5, -6) %>% #skip first three rows bc NA
  group_by(year, Timeframe) %>%
  summarize(yearly_min_Q_cfs = min(weekly_avg_Q_cfs))
```

```
## ‘summarise()‘ has grouped output by ’year’. You can override using the
## ‘.groups‘ argument.
```

```
# Rank yearly minimums, lowest to highest
# split up rankings by time period, before and after intake design
Clayton_min_recurrence4 <- Clayton_yearly_min %>%
  filter(Timeframe == '1981-2000') %>%
  arrange(yearly_min_Q_cfs)

Clayton_min_recurrence_81_2000 <- Clayton_min_recurrence4 %>%
  ungroup() %>%
  mutate(
    Rank = rank(yearly_min_Q_cfs),
    weibull_return = ((n()+1)/Rank),
    #weibull_return_24_year = weibull_return_24hr/12,
    weibull_percent_likelihood=(1/weibull_return)*100,
    #weibull_percent_likelihood_yr=(1/weibull_return_24_year)*100,
    log_prob_occurence_weibull = log10(weibull_percent_likelihood)
  )

Clayton_min_recurrence5 <- Clayton_yearly_min %>%
  filter(Timeframe == '2001-2023') %>%
  arrange(yearly_min_Q_cfs)

Clayton_min_recurrence_2001_2023 <- Clayton_min_recurrence5 %>%
  ungroup() %>%
  mutate(
    Rank = rank(yearly_min_Q_cfs),
    weibull_return = ((n()+1)/Rank),
    #weibull_return_24_year = weibull_return_24hr/12,
    weibull_percent_likelihood=(1/weibull_return)*100,
    #weibull_percent_likelihood_yr=(1/weibull_return_24_year)*100,
    log_prob_occurence_weibull = log10(weibull_percent_likelihood)
  )

# Combine the two dataframes
Clayton_min_recurrence_combined <- bind_rows(
  Clayton_min_recurrence_81_2000,
  Clayton_min_recurrence_2001_2023
)
```

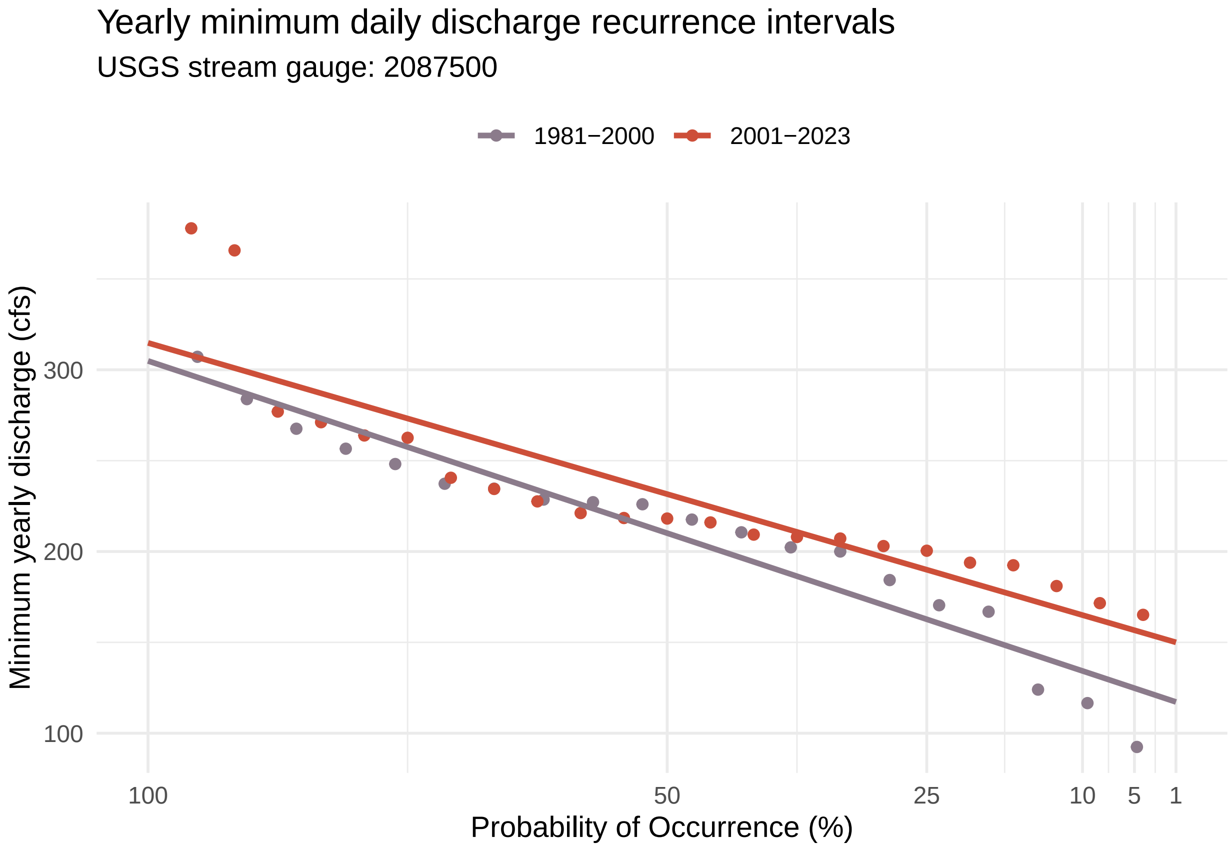
```
# Plotting maximum recurrence intervals in the two time periods

Maximum_recurrence <- Clayton_max_recurrence_combined %>%
  ggplot(aes(x = weibull_percent_likelihood,
             y = yearly_max_Q_cfs,
             color = Timeframe)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, fullrange = TRUE) +
  labs(x = "Probability of Occurrence (%)",
       y = "Maximum yearly discharge (cfs)",
       title = "Yearly maximum daily discharge recurrence intervals",
       subtitle = "USGS stream gauge: 2087500",
       color = NULL) +
  theme_minimal() +
  scale_color_manual(values = c("thistle4", "tomato3"))+
  scale_x_log10(
    trans = "reverse",
    limits = c(100, 1),
    breaks = c(100, 50, 25, 10, 5, 1),
    labels = c(100, 50, 25, 10, 5, 1)
  )+
  theme(legend.position = "top")

Maximum_recurrence
```

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



```
# Plotting minimum recurrence intervals in the two time periods
Minimum_recurrence <- Clayton_min_recurrence_combined %>%
  ggplot(aes(x = weibull_percent_likelihood,
             y = yearly_min_Q_cfs,
             color = Timeframe)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, fullrange = TRUE) +
  labs(x = "Probability of Occurrence (%)",
       y = "Minimum yearly discharge (cfs)",
       title = "Yearly minimum daily discharge recurrence intervals",
       subtitle = "USGS stream gauge: 2087500",
       color = NULL) +
  theme_minimal() +
  scale_color_manual(values = c("thistle4", "tomato3"))+
  scale_x_log10(
    trans = "reverse",
    limits = c(100, 1),
    breaks = c(100, 50, 25, 10, 5, 1),
    labels = c(100, 50, 25, 10, 5, 1)
  )+
  theme(legend.position = "top")

Minimum_recurrence
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```



The lines of best fit on each of the above graphs represent the probability distributions of maximum yearly discharge (range on y-axis) in 1981-2000 and 2001-2023. We used these probability distributions to extract the max and min discharge for 10, 25, 50, and 100 year events for the two time periods, as seen in the tables below:

```r
# Load necessary libraries
library(dplyr)

# Define the specific probabilities of occurrence you want to predict
probabilities <- c(1, 2, 5, 10)

# Create an empty list to store results
results <- list()

# Loop through each timeframe and fit models
for (tf in unique(Clayton_max_recurrence_combined$Timeframe)) {

  # Filter data for the current timeframe
  data_subset <- Clayton_max_recurrence_combined %>%
    filter(Timeframe == tf)

  # Fit the linear model for the current timeframe
  model <- lm(yearly_max_Q_cfs ~ weibull_percent_likelihood, data = data_subset)

  # Create a new data frame for predictions
  new_data <- data.frame(weibull_percent_likelihood = probabilities)
  # Make predictions using the model
  predicted_values <- predict(model, newdata = new_data)

  # Combine the predictions with the corresponding probability and timeframe
  predicted_results <- data.frame(
    Probability = probabilities,
    Predicted_max_discharge = predicted_values,
    Timeframe = tf
  )

  # Store the results in the list
  results[[tf]] <- predicted_results
}

# Combine all results into a single data frame
final_results_max <- bind_rows(results)

# Display the predicted discharge for each timeframe
print(final_results_max)
```

```
##       Probability Predicted_max_discharge Timeframe
## 1...1           1                8481.049 1981-2000
## 2...2           2                8409.559 1981-2000
## 3...3           5                8195.091 1981-2000
## 4...4          10                7837.644 1981-2000
## 1...5           1                6364.408 2001-2023
## 2...6           2                6316.574 2001-2023
## 3...7           5                6173.072 2001-2023
## 4...8          10                5933.901 2001-2023
```

```r
# Load necessary libraries
library(dplyr)

# Define the specific probabilities of occurrence you want to predict
probabilities <- c(1, 2, 5, 10)

# Create an empty list to store results
results <- list()

# Loop through each timeframe and fit models
for (tf in unique(Clayton_min_recurrence_combined$Timeframe)) {

  # Filter data for the current timeframe
  data_subset <- Clayton_min_recurrence_combined %>%
    filter(Timeframe == tf)

  # Fit the linear model for the current timeframe
  model <- lm(yearly_min_Q_cfs ~ weibull_percent_likelihood, data = data_subset)

  # Create a new data frame for predictions
  new_data <- data.frame(weibull_percent_likelihood = probabilities)

  # Make predictions using the model
  predicted_values <- predict(model, newdata = new_data)

  # Combine the predictions with the corresponding probability and timeframe
  predicted_results <- data.frame(
    Probability = probabilities,
    Predicted_min_discharge = predicted_values,
    Timeframe = tf
  )

  # Store the results in the list
  results[[tf]] <- predicted_results
}

# Combine all results into a single data frame
final_results_min <- bind_rows(results)

# Display the predicted discharge for each timeframe
print(final_results_min)
```

```
##       Probability Predicted_min_discharge Timeframe
## 1...1           1                117.1988 1981-2000
## 2...2           2                119.0946 1981-2000
## 3...3           5                124.7820 1981-2000
## 4...4          10                134.2610 1981-2000
## 1...5           1                150.0047 2001-2023
## 2...6           2                151.6694 2001-2023
## 3...7           5                156.6637 2001-2023
## 4...8          10                164.9875 2001-2023
```