Assignment 6: Water Quality in Lakes

Yikai Jing

OVERVIEW

This exercise accompanies the lessons in Water Data Analytics on water quality in lakes

Directions

1. Change "Student Name" on line 3 (above) with your name.
2. Work through the steps, **creating code and output** that fulfill each instruction.
3. Be sure to **answer the questions** in this assignment document.
4. When you have completed the assignment, **Knit** the text and code into a single PDF file.
5. After Knitting, check your PDF against the key and then submit your assignment completion survey at https://forms.gle/swoene3wmGGVUnnm7

Having trouble? See the assignment's answer key if you need a hint. Please try to complete the assignment without the key as much as possible - this is where the learning happens!

Target due date: 2022-03-01

Setup

1. Verify your working directory is set to the R project file. Load the tidyverse, lubridate, and dataRetrieval packages. Set your ggplot theme (can be theme_classic or something else)
2. Use the dataRetrieval package to fetch North Carolina lake TN, TP, and chlorophyll data from the Water Quality Portal. Since we didn't use this method in class, the code is provided for you. Be patient; this query will take some time.

General Data Import from Water Quality Portal Water Quality Portal Web Services Guide

```
setwd("/Users/me/Water_Data_Analytics_2022/Assignments")
getwd()
```

```
## [1] "/Users/me/Water_Data_Analytics_2022/Assignments"
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
##
```

```
## Attaching package: 'lubridate'
```

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

```
library(dataRetrieval)
library(viridis)
```

```
## Loading required package: viridisLite
```

```
theme_set(theme_classic())
options(scipen = 4)

?readWQPdata
NCdata <- readWQPdata(statecode = "North Carolina",
                      siteType = "Lake, Reservoir, Impoundment",
                      sampleMedia = c("Water", "water"),
                      CharacteristicName = c("Phosphorus", "Chlorophyll a"))

NCsites <- whatWQPsites(statecode = "North Carolina",
                      siteType = "Lake, Reservoir, Impoundment")
```

Water Quality Portal downloads have the same columns each time, but be aware that data are uploaded to the Water Quality Portal by individual organizations, which may or may not follow the same conventions. Data and metadata quality are not guaranteed! Make sure to carefully explore any data and make conservative quality assurance decisions where information is limited.

General data processing and quality assurance considerations:

- WQP data is acquired in long format. It may be useful to wrangle the dataset into wide format (we will do this today)
- `readWQPdata` does not inherently restrict the variables pulled from WQP. You may specify the desired variables by using, for instance: 'characteristicName = "pH" '
- **ResultMeasureValue** should be numeric, with details on detection limits, qualifiers, etc. provided in other columns. This is not always the case!
- **ResultSampleFractionText** specifies forms of constituents. In some cases, a single **CharacteristicName** will have both "Total" and "Dissolved" forms specified, which should not be combined.
- Some variables have different names but represent the same constituent (e.g., "Total Kjeldahl nitrogen (Organic N & NH3)" and "Kjeldahl nitrogen"). Always refer to the **ResultAnalyticalMethod** columns to verify methods are measuring the same constituent.
- **ActivityDepthHeightMeasure.MeasureValue** provides depth information. This is a crucial column for lake data but less often for river data.
- **ResultCommentText** often has details relating to additional QA.
- **MeasureQualifierCode** Contains information about data flags:
- *U* designates below detection limit (action: set value to 1/2 detection or quantitation limit from **DetectionQuantitationLimitMeasure.MeasureValue**)
- *J* designates above detection limit but below quantitation limit (action: retain value)
- Other codes may designate suspect data or other flags which may be described in detail in **ResultLaboratoryCommentText** or another column

3. Note the wrangling steps outlined here. Then, set ActivityStartDate as date.

4. Then, add a new pipe that does the following:

- Select the columns OrganizationIdentifier, OrganizationFormalName, ActivityStartDate, ActivityConductingOrganizationText, MonitoringLocationIdentifier, Depth, ResultMeasureValue, Variable
- Group by those same variables except ResultMeasureValue, then `summarise` the mean of ResultMeasureValue. This will generate a single row for each location-date-depth sample.
- Pivot your dataset into wide format. Hint: `pivot_wider` by Variable and ResultMeasureValue.
- Add in month and year as new columns. Filter the dataset for just months May-October.
- Filter the dataset for just depths 1 m or shallower.

5. Create a graph visualizing the chlorophyll-TP stressor-response relationship. Add a line of best fit for the linear regression, and adjust axis scales and labels as needed.

```
summary(as.factor(NCdata$ResultSampleFractionText[NCdata$CharacteristicName == "Chlorophyll a"]))
```

```
## Filtered, lab         Total          NA's
##            30          8782         15002
```

```
summary(as.factor(NCdata$ResultSampleFractionText[NCdata$CharacteristicName == "Phosphorus"]))
```

```
## Dissolved     Total      NA's
##      1104     20863        27
```

```
summary(as.factor(NCdata$ResultMeasure.MeasureUnitCode[NCdata$CharacteristicName == "Phosphorus"]))
```

```
##      mg/l mg/l as P      NA's
##     15852      4199      1943
```

```
summary(as.factor(NCdata$ResultSampleFractionText[NCdata$CharacteristicName == "Phosphorus"]))
```

```
## Dissolved     Total      NA's
##      1104     20863        27
```

```
summary(as.factor(NCdata$MeasureQualifierCode))
```

```
##     H     U  NA's
##     4   227 45577
```

```
NCdata$ResultMeasureValue <- as.numeric(NCdata$ResultMeasureValue)
```

```
## Warning: NAs introduced by coercion
```

```
NCdata_processed <- NCdata %>%
  mutate(Depth_m = case_when(ActivityDepthHeightMeasure.MeasureUnitCode %in% c("feet", "ft") ~
                               ActivityDepthHeightMeasure.MeasureValue * 0.3048,
                               ActivityDepthHeightMeasure.MeasureUnitCode %in% c("meters", "m") ~
                                 ActivityDepthHeightMeasure.MeasureValue),
         Variable = case_when(CharacteristicName == "Phosphorus" &
                                ResultSampleFractionText == "Total" ~ "TP_mgL",
                              CharacteristicName == "Phosphorus" &
                                ResultSampleFractionText == "Dissolved" ~ "TDP_mgL",
                              CharacteristicName == "Chlorophyll a" ~ "Chla_ugL"),
         ResultMeasureValue = case_when(MeasureQualifierCode == "U" ~
                                          DetectionQuantitationLimitMeasure.MeasureValue,
                                        TRUE ~ ResultMeasureValue)) %>%
  drop_na(Variable, ResultMeasureValue, Depth_m)

NCdata_processed$ActivityStartDate <- as.Date(NCdata_processed$ActivityStartDate, "%Y-%m-%d")

NCdata_processed <- NCdata_processed %>%
```

```
  select(OrganizationIdentifier, OrganizationFormalName, ActivityStartDate,
         ActivityConductingOrganizationText, MonitoringLocationIdentifier,
         Depth_m, ResultMeasureValue, Variable) %>%
  group_by(OrganizationIdentifier, OrganizationFormalName, ActivityStartDate,
           ActivityConductingOrganizationText, MonitoringLocationIdentifier,
           Depth_m, Variable) %>%
  summarise(ResultMeasureValue = mean(ResultMeasureValue, na.rm = TRUE)) %>%
  mutate(Month = month(ActivityStartDate),
         Year = year(ActivityStartDate)) %>%
  pivot_wider(names_from = "Variable", values_from = "ResultMeasureValue") %>%
  filter(Month %in% c(5:10)) %>%
  filter(Depth_m <= 1)
```

```
## `summarise()` has grouped output by 'OrganizationIdentifier', 'OrganizationFormalName', 'ActivitySta
```

```
ggplot(NCdata_processed, aes(x = TP_mgL, y = Chla_ugL)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm", se = FALSE, color = "black") +
  scale_x_log10() +
  scale_y_log10() +
  labs(x = "TP (mg/L)", y = expression("Chl a ("*mu*"g/L)"))
```

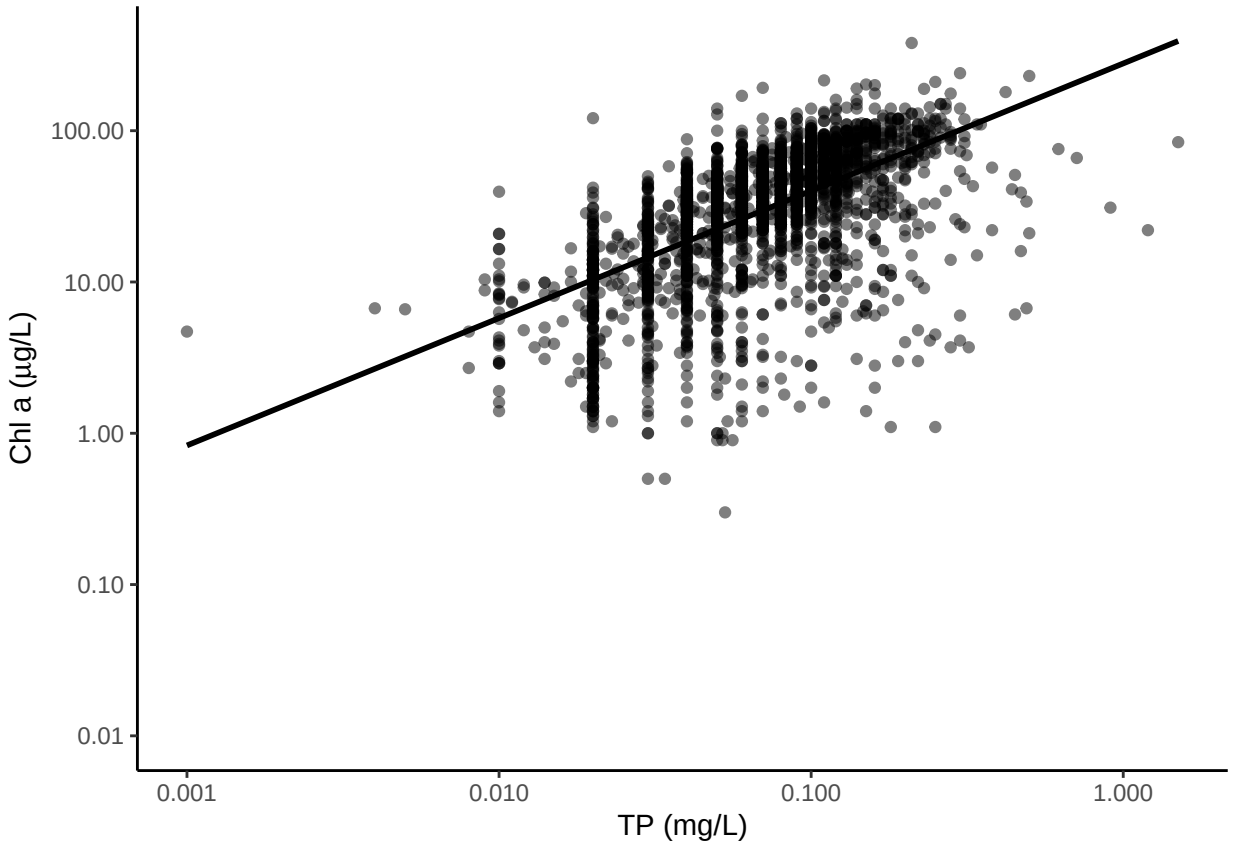
```
## Warning: Transformation introduced infinite values in continuous y-axis
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

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

```
## Warning: Removed 2418 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 2418 rows containing missing values (geom_point).
```



6. How might you expect this stressor-response relationship to change (or not) across the Mountains, Piedmont, and Coastal Plain regions of the state? How might you anticipate natural lakes might behave differently than reservoirs?