Assignment 5: Water Quality in Lakes

Aislinn McLaughlin

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/fSe18vMhgzcjUKM39

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: 2022-02-22

Setup

1. Verify your working directory is set to the R project file. Load the tidyverse, lubridate, and LAGOSNE packages. Set your ggplot theme (can be theme_classic or something else)
2. Load the LAGOSdata database and the trophic state index csv file we created in class.

```
getwd()
```

```
## [1] "/Users/Aislinn/Documents/GitHub/Water_Data_Analytics_2022/Assignments"
```

```
setwd("/Users/Aislinn/Documents/GitHub/Water_Data_Analytics_2022/")

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(LAGOSNE)

mytheme <-
  theme_gray(base_size = 12) +
  theme(legend.background = element_rect(fill = "gray"), legend.position = "bottom",
        plot.title = element_text(face = "bold", size = 14, color = "black", hjust = 0.5),
        plot.subtitle = element_text(size = 10, color = "gray", hjust = 0.5))

theme_set(mytheme)

# still getting an error! using workaround
# lagosne_get(dest_folder = LAGOSNE:::lagos_path(), overwrite = TRUE)

load(file = "./Data/Raw/LAGOSdata.rda")
lagos.tsi <- read.csv("./Data/Processed/LAGOStrophic.csv")
```

Trophic State Index

3. Similar to the trophic.class column we created in class (determined from TSI.chl values), create two additional columns in the data frame that determine trophic class from TSI.secchi and TSI.tp (call these trophic.class.secchi and trophic.class.tp).

```
lagos.tsi <-
  lagos.tsi %>%
  mutate(trophic.class.secchi = case_when(TSI.secchi < 40 ~ "Oligotrophic",
                                  TSI.secchi >= 40 & TSI.secchi < 50 ~ "Mesotrophic",
                                  TSI.secchi >= 50 & TSI.secchi< 70 ~ "Eutrophic",
                                  TSI.secchi >= 70 ~ "Hypereutrophic"),
         trophic.class.tp = case_when(TSI.tp < 40 ~ "Oligotrophic",
                                  TSI.tp >= 40 & TSI.tp < 50 ~ "Mesotrophic",
                                  TSI.tp >= 50 & TSI.tp< 70 ~ "Eutrophic",
                                  TSI.tp >= 70 ~ "Hypereutrophic")
)

lagos.tsi$trophic.class <- factor(lagos.tsi$trophic.class,
                                        levels = c("Oligotrophic", "Mesotrophic", "Eutrophic", "Hypereu

lagos.tsi$trophic.class.secchi <- factor(lagos.tsi$trophic.class.secchi,
                                        levels = c("Oligotrophic", "Mesotrophic", "Eutrophic", "Hypereu

lagos.tsi$trophic.class.tp <- factor(lagos.tsi$trophic.class.tp,
                                        levels = c("Oligotrophic", "Mesotrophic", "Eutrophic", "Hypereu
```

4. How many observations fall into the four trophic state categories for the three metrics (trophic.class, trophic.class.secchi, trophic.class.tp)? Hint: `n()` function.

```
lagos.tsi %>%
  group_by(trophic.class) %>%
  summarise(count = n())
```

```
## # A tibble: 4 x 2
##   trophic.class  count
```

```
##   <fct>          <int>
## 1 Oligotrophic    2762
## 2 Mesotrophic    13964
## 3 Eutrophic      37457
## 4 Hypereutrophic 13234
```

```
lagos.tsi %>%
  group_by(trophic.class.secchi) %>%
  summarise(count = n())
```

```
## # A tibble: 4 x 2
##   trophic.class.secchi count
##   <fct>                <int>
## 1 Oligotrophic         14559
## 2 Mesotrophic          22344
## 3 Eutrophic            25793
## 4 Hypereutrophic        4721
```

```
lagos.tsi %>%
  group_by(trophic.class.tp) %>%
  summarise(count = n())
```

```
## # A tibble: 4 x 2
##   trophic.class.tp count
##   <fct>            <int>
## 1 Oligotrophic     17984
## 2 Mesotrophic      20607
## 3 Eutrophic        22419
## 4 Hypereutrophic    6407
```

5. What proportion of total observations are considered eutrophic or hypereutrophic according to the three different metrics (trophic.class, trophic.class.secchi, trophic.class.tp)?

```
lagos.tsi %>%
  group_by(trophic.class) %>%
  summarise(count = n()) %>%
  mutate(freq = count / sum(count))
```

```
## # A tibble: 4 x 3
##   trophic.class  count   freq
##   <fct>          <int>  <dbl>
## 1 Oligotrophic    2762 0.0410
## 2 Mesotrophic    13964 0.207
## 3 Eutrophic      37457 0.556
## 4 Hypereutrophic 13234 0.196
```

```
lagos.tsi %>%
  group_by(trophic.class.secchi) %>%
  summarise(count = n()) %>%
  mutate(freq = count / sum(count))
```

```
## # A tibble: 4 x 3
##   trophic.class.secchi count   freq
##   <fct>                <int>  <dbl>
## 1 Oligotrophic         14559 0.216
## 2 Mesotrophic          22344 0.331
## 3 Eutrophic            25793 0.383
## 4 Hypereutrophic        4721 0.0700
```

```
lagos.tsi %>%
  group_by(trophic.class.tp) %>%
  summarise(count = n()) %>%
  mutate(freq = count / sum(count))
```

```
## # A tibble: 4 x 3
##   trophic.class.tp count   freq
##   <fct>            <int>  <dbl>
## 1 Oligotrophic     17984 0.267
## 2 Mesotrophic      20607 0.306
## 3 Eutrophic        22419 0.333
## 4 Hypereutrophic    6407 0.0950
```

Which of these metrics is most conservative in its designation of eutrophic conditions? Why might this be?

> Eutrophic describes a lake that has high levels of productivity. Trophic.class.tp count is most conservative in its designation of eutrophic conditions because it does not measure primary productivity (phytoplankton) but rather the nutrient source for primary producers.

Nutrient Concentrations

6. Create a data frame that includes the columns lagoslakeid, sampledate, tn, tp, state, and state_name. Mutate this data frame to include sampleyear and samplemonth columns as well. Filter the data frame for May-September. Call this data frame LAGOSNandP.

```
LAGOSlocus <- LAGOSdata$locus
LAGOSstate <- LAGOSdata$state
LAGOSnutrient <- LAGOSdata$epi_nutr

LAGOSlocus$lagoslakeid <- as.factor(LAGOSlocus$lagoslakeid)
LAGOSnutrient$lagoslakeid <- as.factor(LAGOSnutrient$lagoslakeid)

LAGOSlocations <- left_join(LAGOSlocus, LAGOSstate, by = "state_zoneid")

LAGOSNandP <- LAGOSnutrient %>%
  left_join(., LAGOSlocations, by = "lagoslakeid") %>%
  select(lagoslakeid, sampledate, tn, tp, state, state_name) %>%
  mutate(sampleyear = year(sampledate),
         samplemonth = month(sampledate)) %>%
  filter(samplemonth >= 5 & samplemonth <=9) %>%
  drop_na(tn, tp, state)
```

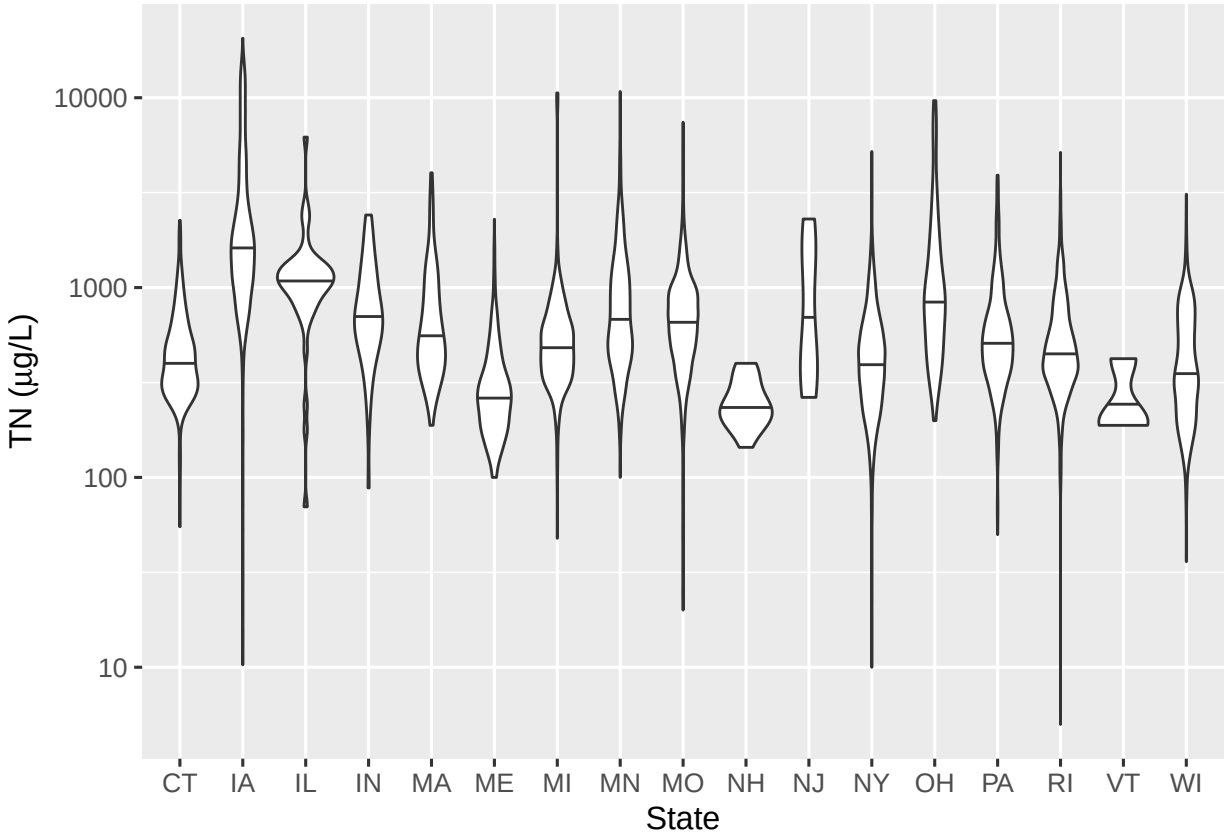
7. Create two violin plots comparing TN and TP concentrations across states. Include a 50th percentile line inside the violins. Create a logged y axis and relabel axes.

```
tn_violin <- ggplot(LAGOSNandP, aes(x = state, y = tn)) +
  geom_violin(draw_quantiles = 0.5) +
  scale_y_log10() +
  labs(x = "State", y = expression("TN ("*mu*"g/L)"))
tn_violin
```

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

```
## Warning: Removed 7 rows containing non-finite values (stat_ydensity).
```

```
## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values
```

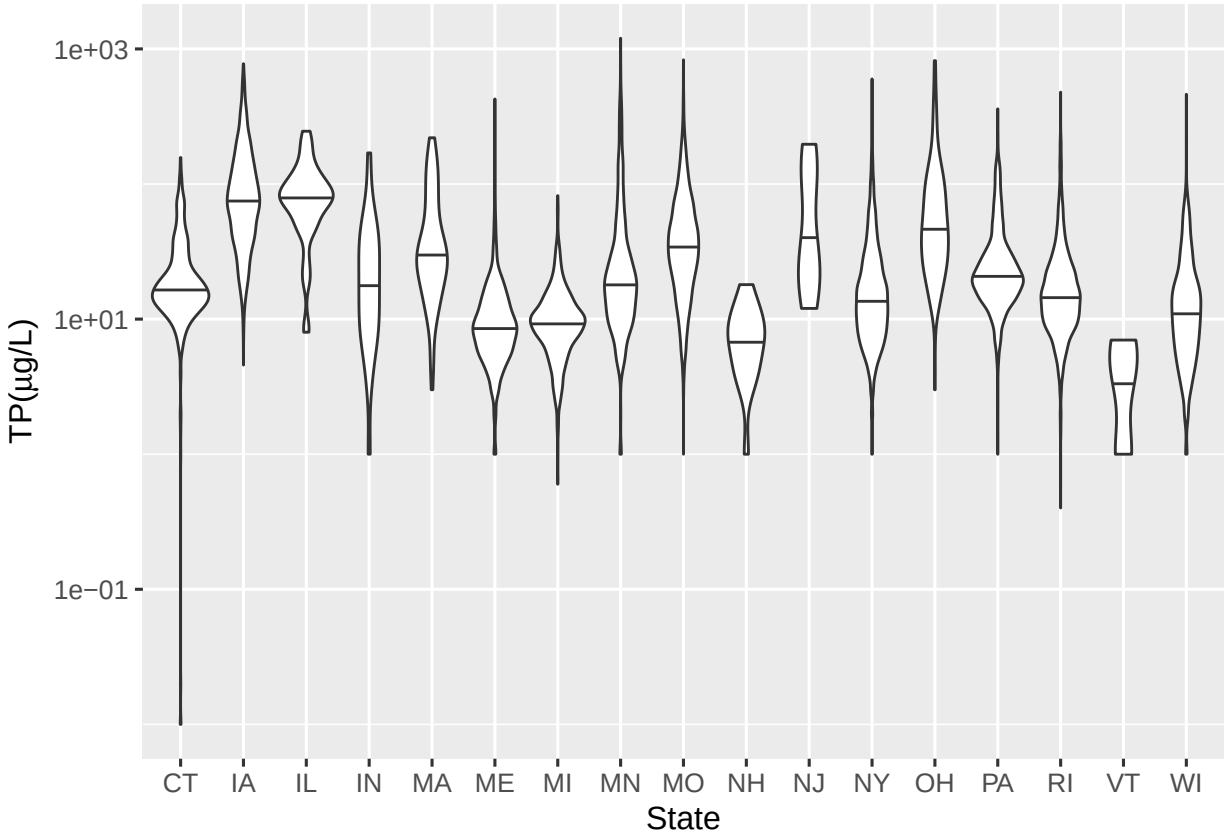



```
tp_violin <- ggplot(LAGOSNandP, aes(x = state, y = tp)) +
  geom_violin(draw_quantiles = 0.5) +
  scale_y_log10() +
  labs(x = "State", y = expression("TP("*mu*"g/L)"))
tp_violin
```

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

```
## Warning: Removed 65 rows containing non-finite values (stat_ydensity).
```

```
## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values
```



Which states have the highest and lowest median concentrations?

TN: Highest - IA and IN, lowest = VT/NH

TP: Highest - IL and IA, lowest - VT and NH

Which states have the largest and smallest concentration ranges?

TN: Largest - CT and RI, smallest - VT/NH

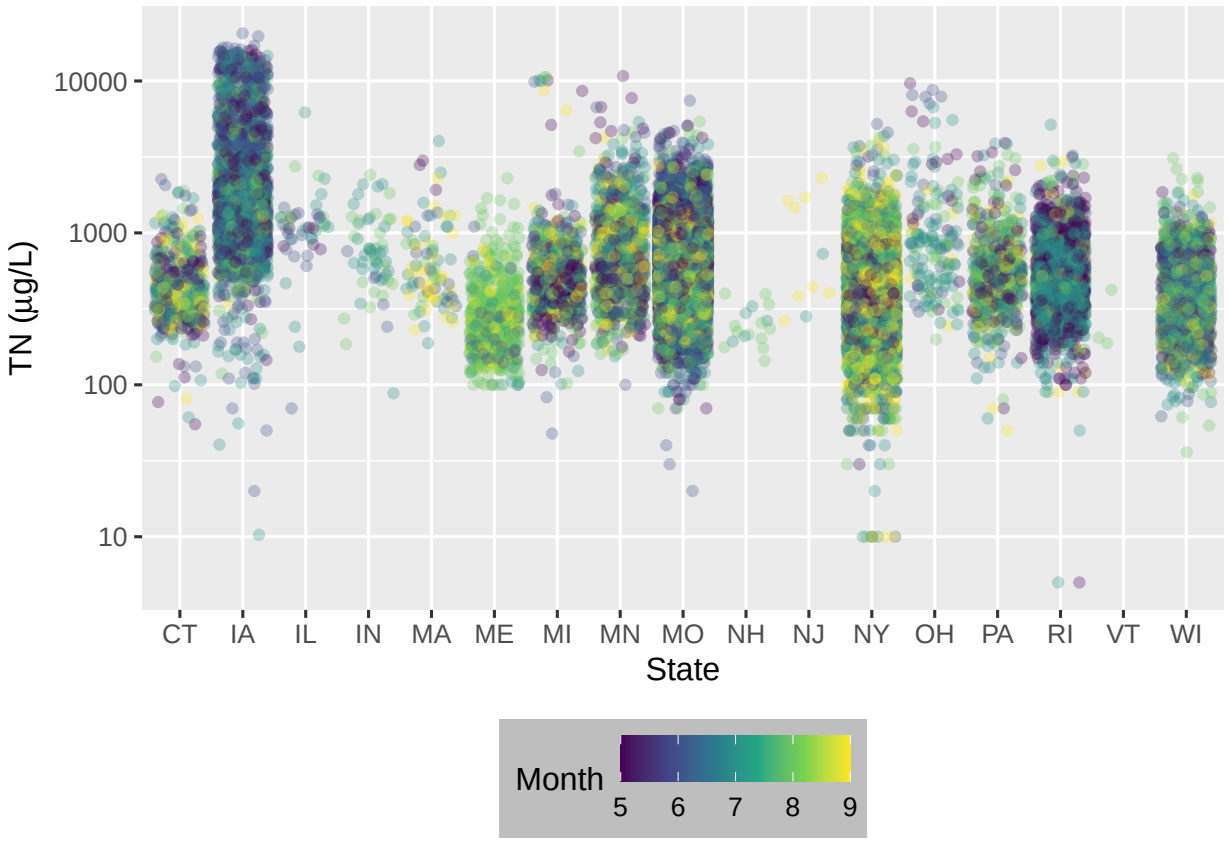
TP: Largest - CT and MN/MO, smallest - VT/NH

8. Create two jitter plots comparing TN and TP concentrations across states, with samplemonth as the color. Choose a color palette other than the ggplot default.

```
tn_jitter <-
  ggplot(LAGOSNandP, aes(x = state, y = tn, color = samplemonth)) +
  geom_jitter(alpha = 0.3) +
  scale_color_viridis_c() +
  labs(x = "State", y = expression("TN ("*mu*"g/L)"), color = "Month") +
  scale_y_log10()
tn_jitter
```

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

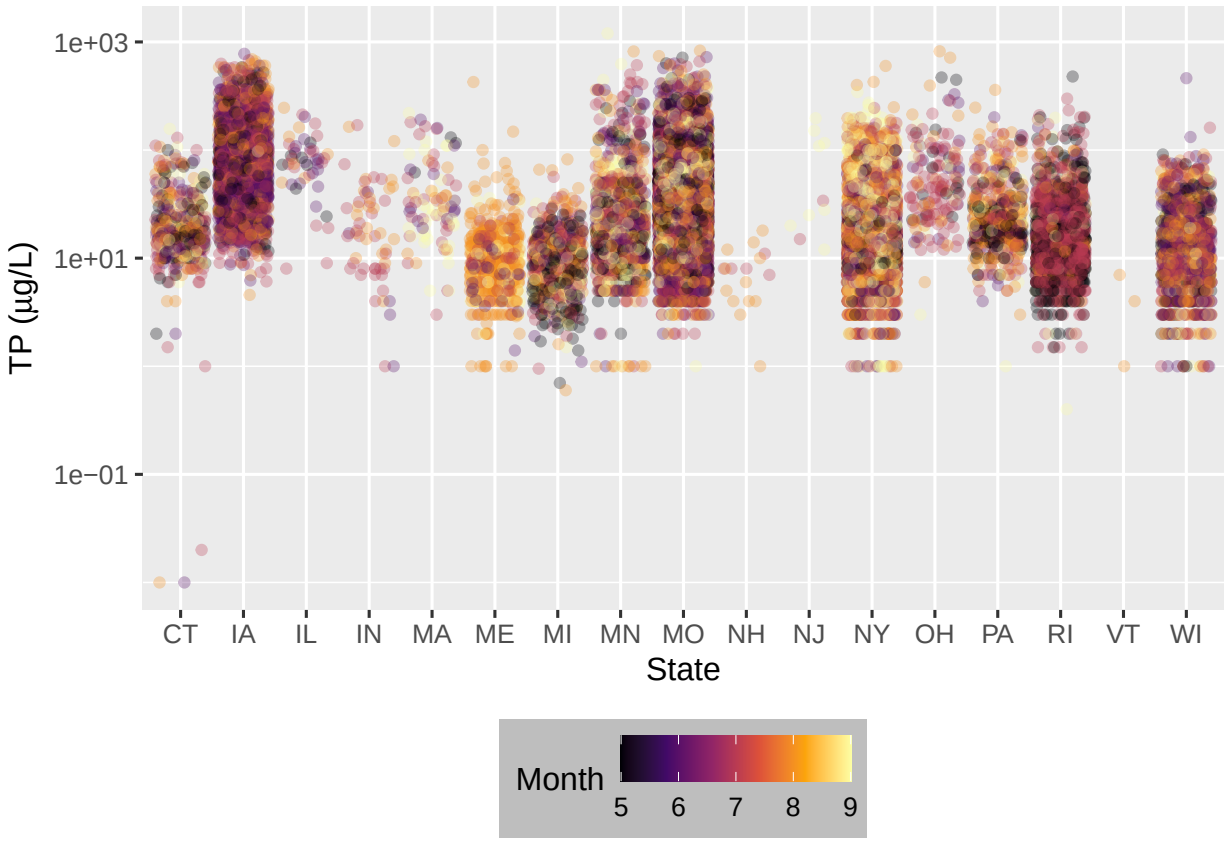
```
## Warning: Removed 7 rows containing missing values (geom_point).
```




```
tp_jitter <-
  ggplot(LAGOSNandP, aes(x = state, y = tp, color = samplemonth)) +
  geom_jitter(alpha = 0.3) +
  scale_color_viridis_c(option = "inferno") +
  labs(x = "State", y = expression("TP ("*mu*"g/L)"), color = "Month") +
  scale_y_log10()
tp_jitter
```

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

```
## Warning: Removed 65 rows containing missing values (geom_point).
```



Which states have the most samples? How might this have impacted total ranges from #7?

TN: IA/MO

TP: IA/MO

States with the most samples are likely to also be states with the largest total ranges because you have more opportunities to collect different data points.