DEBY_Raw_Envr_Data

2024-04-03

```
setwd(dirname(rstudioapi::getActiveDocumentContext()$path))
```

Load required packages.

```
library("dplyr") #Used for working with data frames
```

```
##
## 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
```

```
library("lubridate") #Used for time-date conversions
```

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

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

```
library("readr") #Used to read the CSV file
library("ggplot2")
```

Note the date of data download and source. All available data should be used for each site regardless of year. Note from the CSV file how often the site was sampled, and if there are replicates in the data. Also describe if the sampling occurred at only low tide, only high tide, or continuously.

```
#Data was downloaded on 04/02/2024
#Source - http://vecos.vims.edu/StationDetail.aspx?param=YRK005.40&program=CMON
#The site was sampled continuously every 15 min from 2003
```

```
#Create text strings with metadata information that we want to include in the final data frame.
download_date <- ("04-02-2024")
source_description <- ("Virginia Estuarine and Coastal Observing System, VIMS")
site_name <- ("DEBY")
collection_type <- ("continuous")
```

Use the file path name in your working directory or desktop, see example below. Or, import data set through the "Files" window in R studio. Store the file in a variable with the "raw_ID_Site" format. If salinity and temperature data are in separate files, read in both and store them with "_sal" or "_temp" in the variable names.

```
#The files we will be working with are from Gloucester Point, VA, which is for the DEBY selection line.

#Environmental data could only be downloaded by year, so first we need to merge the yearly data sets.

#set working directory to location of files
setwd("../../data/envr_of_origin/raw_envr_data/DEBY_York_Raw_Yearly")

#merge files into one
raw_DEBY_env <- list.files(path=".") %>%
  lapply(read.csv) %>%
  bind_rows

#set working directory back to rmd file location
setwd("../../../../src/envr_data")

#The metadata for these data (located at the bottom of this site: http://vecos.vims.edu/Content.aspx?id

subset_DEBY_env_error <- subset(raw_DEBY_env, select = c(SAMPLE_DATETIME, WTEMP, WTEMP_A, SALINITY, SAL

subset1_DEBY_env <- subset_DEBY_env_error[(subset_DEBY_env_error$WTEMP_A %in% c("", NA)), ] #keep only

subset2_DEBY_env <- subset1_DEBY_env[(subset1_DEBY_env$SALINITY_A %in% c("", NA)),] #keep only rows whe

View(subset2_DEBY_env)
#sorting the data frame by ascending and descending values for WTEMP_A and SALINITY_A shows only blank

subset_DEBY_env <- subset(subset2_DEBY_env, select = c(SAMPLE_DATETIME, WTEMP, SALINITY)) #remove WTEMP

# View how the data are stored. Note the variable names and the format and units that the data are stor
summary(subset_DEBY_env)
```

```
##  SAMPLE_DATETIME        WTEMP           SALINITY
##  Length:667345      Min.   : 0.020   Min.   : 8.06
##  Class :character   1st Qu.: 9.639   1st Qu.:17.69
##  Mode  :character   Median :17.300   Median :19.68
##                     Mean   :17.122   Mean   :19.34
##                     3rd Qu.:25.010   3rd Qu.:21.31
##                     Max.   :32.290   Max.   :26.09
```

```r
#rename columns. "datetime" = date and time of data collection, "temp" = water temperature in degrees C

colnames(subset_DEBY_env) <- c("datetime", "temp", "salinity")
```

Start with the date and time of collection. We will use the lubridate package to standardize all values into the date-time format called POSIXct. This format stores the date and time in number of seconds since a past point (1/1/1970). This makes comparisons easy and helps to standardizes values.

```r
#Convert to POSIXct format. Tell R what the current date/time format is so it knows how to convert it.
subset_DEBY_env$datetime <- as.POSIXct(subset_DEBY_env$datetime, "%m/%d/%Y %H:%M:%S %p", tz = "")


#Print the new data frame and examine to make sure the new datetime column is in the correct format.
head(subset_DEBY_env)
```

```
##              datetime  temp salinity
## 1 2003-05-28 04:15:00 19.07    15.57
## 2 2003-05-28 04:30:00 19.15    15.45
## 3 2003-05-28 04:45:00 19.38    15.49
## 4 2003-05-28 05:00:00 19.81    15.58
## 5 2003-05-28 05:15:00 19.80    15.60
## 6 2003-05-28 05:30:00 19.89    15.56
```

#Standardize column and variable names. We will use "lat" for latitude in degrees, and "lon" for longitude in degrees.

```r
#Store variables that we will include in the final data frame
lat <- 37.247284
lon <- -76.499369
firstyear <- 2003
finalyear <- 2024
```

Filter any of the variables that have data points outside of normal range. We will use 0-40 as the accepted range for salinity (ppt) and temperature (C) values. Note, in the summer, salinity values can sometimes exceed 40. Check to see if there are values above 40. In this case, adjust the range or notify someone that the site has particularly high salinity values.

```r
#Filter the data between the values of 0 and 40 for both salinity and temperature.
filtered_DEBY_sal <- subset_DEBY_env %>%
    filter(between(salinity, 0, 40))

summary(subset_DEBY_env)
```

```
##     datetime                         temp           salinity
##  Min.   :2003-05-28 04:15:00.00   Min.   : 0.020   Min.   : 8.06
##  1st Qu.:2008-12-20 09:30:00.00   1st Qu.: 9.639   1st Qu.:17.69
```

```
##  Median :2014-01-13 02:30:00.00   Median :17.300   Median :19.68
##  Mean   :2013-12-24 11:40:09.11   Mean   :17.122   Mean   :19.34
##  3rd Qu.:2019-01-08 04:00:00.00   3rd Qu.:25.010   3rd Qu.:21.31
##  Max.   :2024-01-11 12:45:00.00   Max.   :32.290   Max.   :26.09
##  NA's   :152
```

```
filtered_DEBY_env <- filtered_DEBY_sal %>%
    filter(between(temp, 0, 40))

# Sanity check - print the ranges to ensure values are filtered properly. We can see that the ranges fo
print(summary(filtered_DEBY_env$salinity))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    8.06   17.69   19.68   19.34   21.31   26.09
```

```
print(summary(filtered_DEBY_env$temp))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.020   9.639  17.300  17.122  25.010  32.290
```

```
#Store our data into a variable name with just the site name.
DEBY_env <- filtered_DEBY_env

# we have NAs in the our data frame in the datetime column - need to remove these
count.nas_env <- is.na(DEBY_env$datetime) # store our NAs in a variable
summary(count.nas_env) # we have 152 NAs that are stored as "TRUE" in our count.nas
```

```
##    Mode   FALSE    TRUE
## logical  667193     152
```

```
nrow(DEBY_env) # figure out how many rows we have in the original df: 667345
```

```
## [1] 667345
```

```
which(count.nas_env == TRUE) # find the number of NA rows that we need to remove: 152
```

```
##   [1]  19905  19906  19907  19908  19953  19954  19955  19956  46203  46204
##  [11]  46205  46206  46251  46252  46253  46254  79678  79679  79680  79681
##  [21]  79726  79727  79728  79729 108089 108090 108091 108092 108137 108138
##  [31] 108139 108140 141594 141595 141596 141597 141642 141643 141644 141645
##  [41] 174170 174171 174172 174173 174218 174219 174220 174221 206883 206884
##  [51] 206885 206886 206931 206932 206933 206934 237216 237217 237218 237219
##  [61] 237264 237265 237266 237267 271307 271308 271309 271310 271355 271356
##  [71] 271357 271358 305058 305059 305060 305061 305106 305107 305108 305109
##  [81] 338850 338851 338852 338853 338898 338899 338900 338901 372369 372370
##  [91] 372371 372372 372417 372418 372419 372420 405987 405988 405989 405990
## [101] 406035 406036 406037 406038 439794 439795 439796 439797 439842 439843
## [111] 439844 439845 472348 472349 472350 472351 472396 472397 472398 472399
## [121] 506313 506314 506315 506316 506358 506359 506360 506361 539401 539402
## [131] 539403 539404 539449 539450 539451 539452 605533 605534 605535 605536
## [141] 605581 605582 605583 605584 640317 640318 640319 640320 640365 640366
## [151] 640367 640368
```

```r
DEBY_env <- na.omit(DEBY_env) #remove NAs using na.omit
nrow(DEBY_env) #there are 667193 rows in the new data frame
```

```
## [1] 667193
```

```r
check_env <- 667345 - 667193#the value of the check should be 152
check_env #we removed 152 NA rows!
```

```
## [1] 152
```

```r
# check for NAs in our temperature column
count.nas_temp <- is.na(DEBY_env$temp) # store our NAs in temp in a variable
summary(count.nas_temp) # we have 0 NAs
```

```
##    Mode   FALSE
## logical  667193
```

```r
#check for NAs in our temperature column
count.nas_sal <- is.na(DEBY_env$salinity) #store our NAs in salinity in a variable
summary(count.nas_sal) #we have no NAs
```

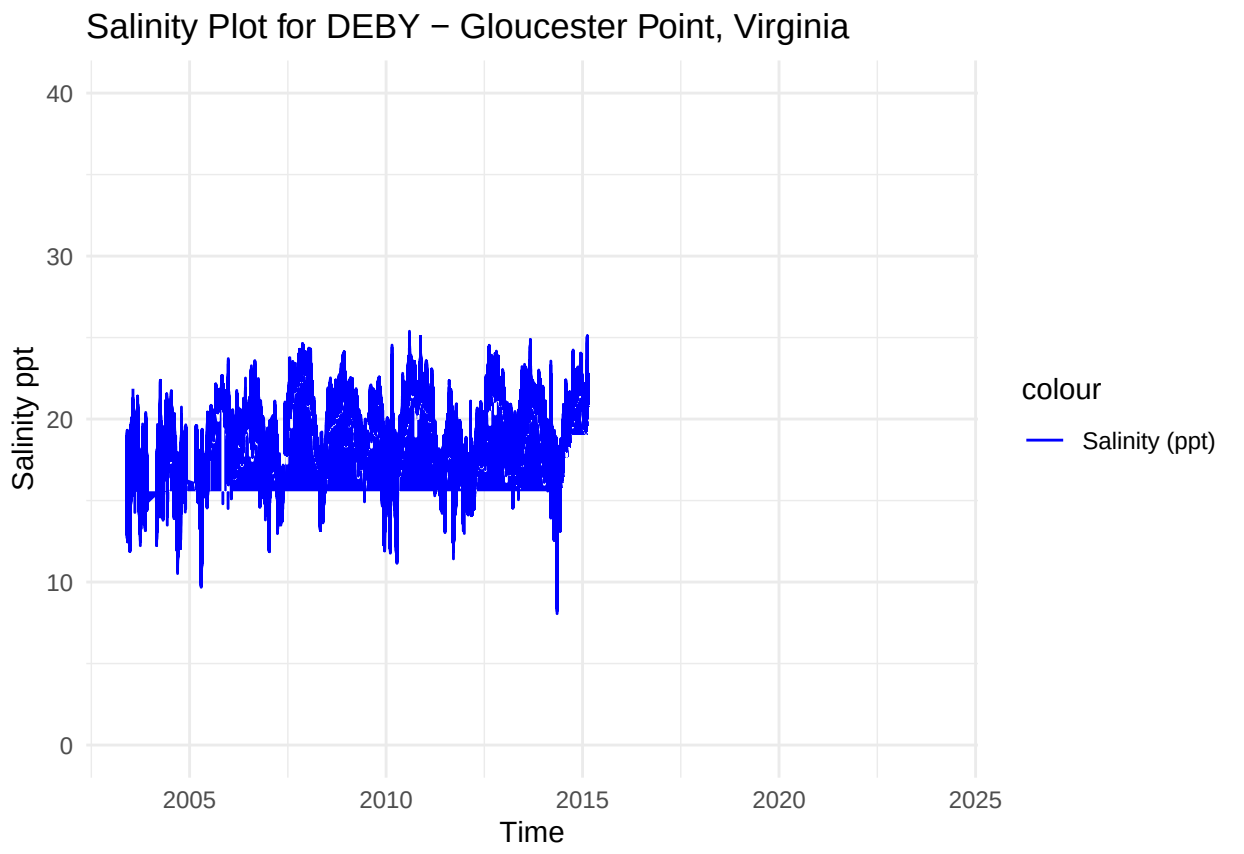
```
##    Mode   FALSE
## logical  667193
```

#Data sets for violin plots

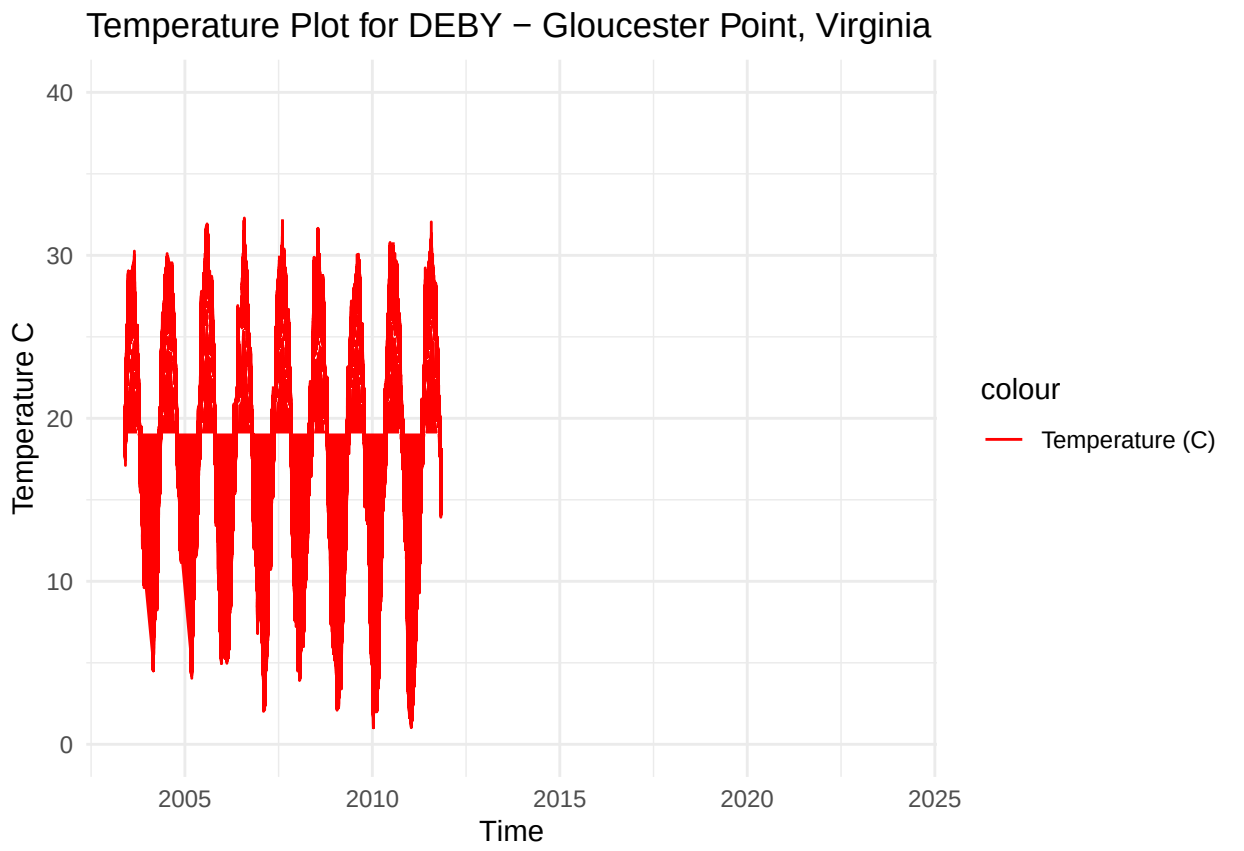
```r
#add site name and create new data frame with full envr data set
DEBY_env_full <- DEBY_env %>%
  mutate(site_name, site_name = "DEBY")

#reorder columns with site_name first
DEBY_env_full <- DEBY_env_full[, c(4, 1, 2, 3)]

DEBY_temp_full <- DEBY_env_full[, c(1,2,3)]

DEBY_sal_full <- DEBY_env_full[, c(1, 2, 4)]

#save DEBY_temp_full and DEBY_sal_full as csv files for future analyses
write.csv(DEBY_temp_full, "../../data/envr_of_origin/full_temp/DEBY_temp_full.csv", row.names = FALSE)

write.csv(DEBY_sal_full, "../../data/envr_of_origin/full_sal/DEBY_sal_full.csv", row.names = FALSE)
```

Visualize the salinity, temperature, and date ranges over time. This can help us see if there are any anomalies or gaps in the data and make sure the filtering was done correctly. Sanity check - do the temperature and salinity ranges look appropriate for the geography of the site (ex. near full ocean salinity for coastal sites, lower salinity for estuaries or near rivers)?

```
salplot <- ggplot(DEBY_env, aes(x = datetime)) +
    geom_line(aes(y = salinity, color = "Salinity (ppt)")) +
    ylim(0,40) +
    labs(x = "Time", y = "Salinity ppt", title = "Salinity Plot for DEBY - Gloucester Point, Virginia") +
    scale_color_manual(values = c("Salinity (ppt)" = "blue")) +
    theme_minimal()


salplot
```




```
tempplot <- ggplot(DEBY_env, aes(x = datetime)) +
    geom_line(aes(y = temp, color = "Temperature (C)")) +
    ylim(0, 40) +
    labs(x = "Time", y = "Temperature C", title = "Temperature Plot for DEBY - Gloucester Point, Virgini
    scale_color_manual(values = c( "Temperature (C)" = "red")) +
    theme_minimal()

tempplot
```



We need to calculate the mean, maximum, and minimum values for salinity and temperature per month and year. First make two data frames to contain each of the annual and monthly averages.

```
#Calculate the mean, maximum, and minimum values for salinity and temperature for each month.
DEBY_envrmonth_sal <- DEBY_env %>%
    mutate(year = year(datetime), month = month(datetime)) %>%
    group_by(year, month) %>%
    summarise(
      min_salinity = min(salinity),
      max_salinity = max(salinity),
      mean_salinity = mean(salinity),
      length_salinity = length(salinity))
```

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

```
DEBY_envrmonth_temp <- DEBY_env %>%
    mutate(year = year(datetime), month = month(datetime)) %>%
    group_by(year, month) %>%
    summarise(
      min_temp = min(temp),
      max_temp = max(temp),
```

```
      mean_temp = mean(temp),
      length_temp = length(temp))
```

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

```
print(DEBY_envrmonth_sal)
```

```
## # A tibble: 246 x 6
## # Groups:   year [22]
##     year month min_salinity max_salinity mean_salinity length_salinity
##    <dbl> <dbl>        <dbl>        <dbl>         <dbl>           <int>
##  1  2003     5         12.7         15.7          14.4             319
##  2  2003     6         11.9         19.3          14.7            2880
##  3  2003     7         14.3         21.9          18.0            2976
##  4  2003     8         14.3         20.6          18.1            2976
##  5  2003     9         13.0         21.4          17.4            2880
##  6  2003    10         12.2         19.7          16.4            2975
##  7  2003    11         13.2         20.4          16.8            2880
##  8  2003    12         14.4         16.2          15.5              33
##  9  2004     2         12.2         15.5          14.1             241
## 10  2004     3         13.0         19.7          16.4            1448
## # i 236 more rows
```

```
print(DEBY_envrmonth_temp)
```

```
## # A tibble: 246 x 6
## # Groups:   year [22]
##     year month min_temp max_temp mean_temp length_temp
##    <dbl> <dbl>    <dbl>    <dbl>     <dbl>       <int>
##  1  2003     5    18.6     20.9      19.6          319
##  2  2003     6    17.1     29.0      22.6         2880
##  3  2003     7    23.7     29.1      26.5         2976
##  4  2003     8    25.0     30.3      27.0         2976
##  5  2003     9    21.7     27.8      24.4         2880
##  6  2003    10    15.4     23.2      19.2         2975
##  7  2003    11     9.61    19.5      14.9         2880
##  8  2003    12    10.4     11.9      11.2           33
##  9  2004     2     4.5      7.64      5.59         241
## 10  2004     3     5.27    12.7       8.63        1448
## # i 236 more rows
```

```
#Calculate the mean, maximum, and minimum values for salinity and temperature for each year.
DEBY_envryear_sal <- DEBY_env %>%
    mutate(year = year(datetime)) %>%
    group_by(year) %>%
    summarise(
      min_salinity = min(salinity),
      max_salinity = max(salinity),
      mean_salinity = mean(salinity))
```

```
DEBY_envryear_temp <- DEBY_env %>%
    mutate(year = year(datetime)) %>%
    group_by(year) %>%
    summarise(
      min_temp = min(temp),
      max_temp = max(temp),
      mean_temp = mean(temp))

print(DEBY_envryear_sal)
```

```
## # A tibble: 22 x 4
##     year min_salinity max_salinity mean_salinity
##    <dbl>        <dbl>        <dbl>         <dbl>
##  1  2003        11.9          21.9          16.9
##  2  2004        10.5          22.4          17.2
##  3  2005         9.68         23.7          18.6
##  4  2006        13.8          23.6          19.1
##  5  2007        11.9          24.6          20.1
##  6  2008        13.2          24.4          20.1
##  7  2009        11.9          22.6          19.8
##  8  2010        11.2          25.4          20.1
##  9  2011        11.4          23.6          18.1
## 10  2012        14.1          24.5          19.7
## # i 12 more rows
```

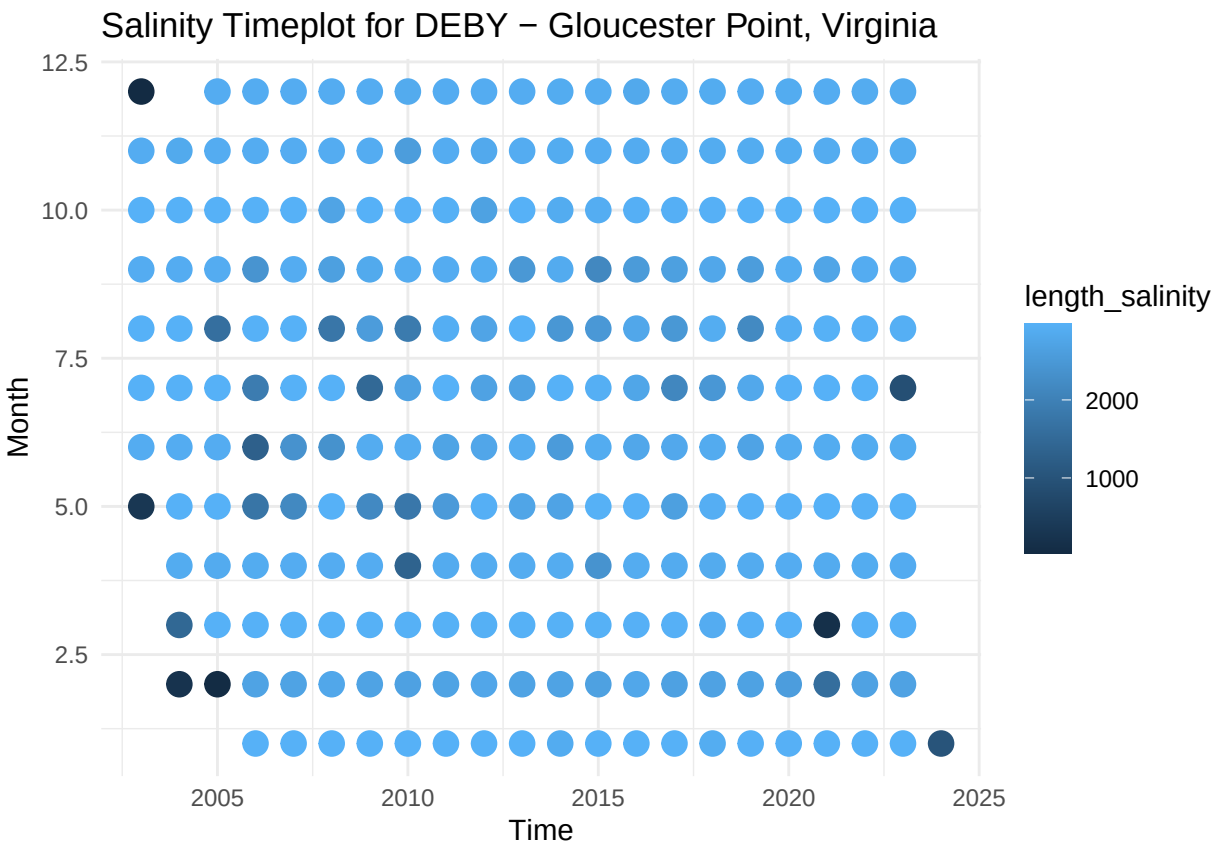
```
print(DEBY_envryear_temp)
```

```
## # A tibble: 22 x 4
##     year min_temp max_temp mean_temp
##    <dbl>    <dbl>    <dbl>     <dbl>
##  1  2003     9.61     30.3      22.4
##  2  2004     4.5      30.1      20.6
##  3  2005     4.05     31.9      18.4
##  4  2006     4.97     32.3      16.0
##  5  2007     2.02     32.2      17.1
##  6  2008     3.92     31.7      16.4
##  7  2009     2.09     30.1      15.9
##  8  2010     1.01     30.8      16.0
##  9  2011     1.02     32.0      16.9
## 10  2012     5.86     30.8      17.5
## # i 12 more rows
```

Plot the months and years of data collection to check if there are any collection gaps in the data.

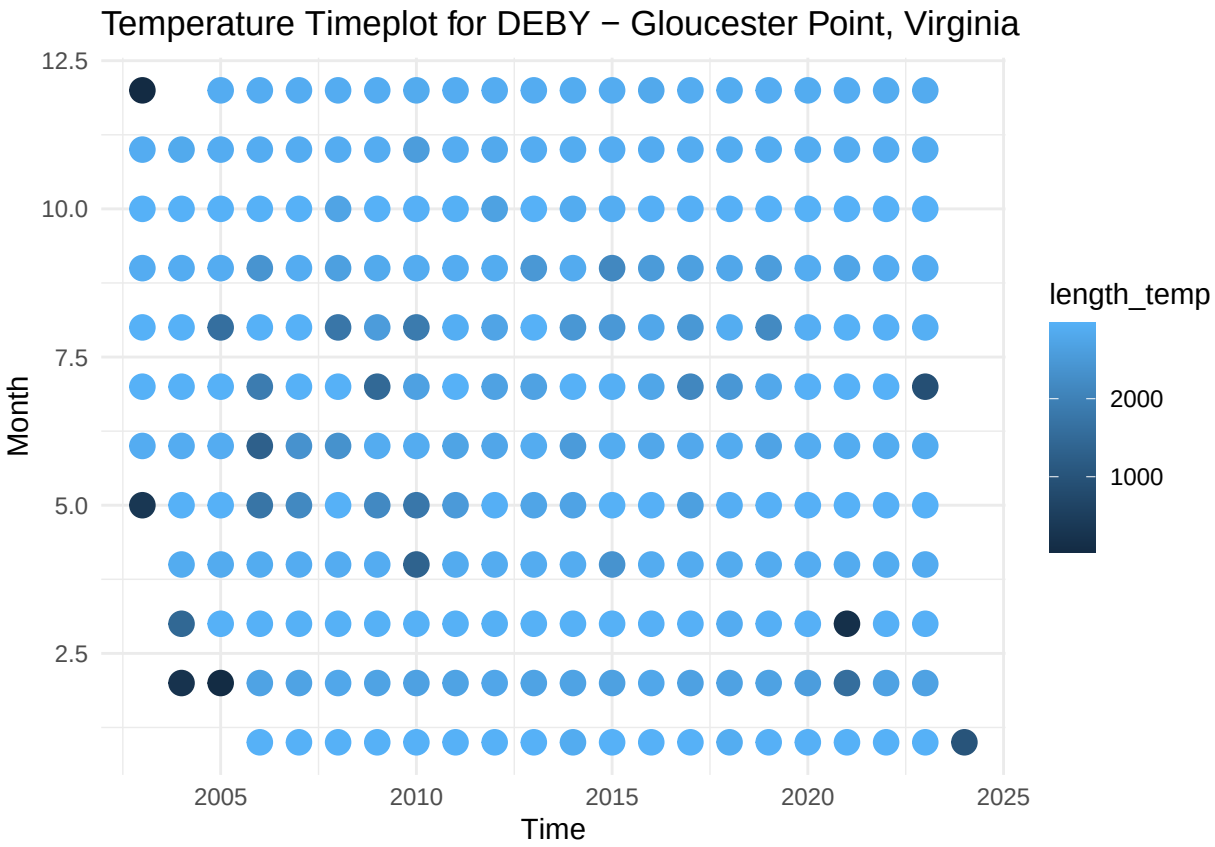
```
timeplot <- ggplot(DEBY_envrmonth_sal, aes(x = year)) +
    geom_point(aes(y = month, color = length_salinity), size = 4) +
    labs(x = "Time", y = "Month", title = "Salinity Timeplot for DEBY - Gloucester Point, Virginia") +
    ylim(1,12) +
    theme_minimal()
```

```
timeplot
```



Plot the months and years of data collection to check if there are any collection gaps in the data.

```
timeplot_temp <- ggplot(DEBY_envrmonth_temp, aes(x = year)) +
    geom_point(aes(y = month, color = length_temp), size = 4) +
    labs(x = "Time", y = "Month", title = "Temperature Timeplot for DEBY - Gloucester Point, Virginia") +
    ylim(1,12) +
    theme_minimal()

timeplot_temp
```



We can now calculate a list of variables that we will have collected for all sites. This will allow us to compare sites easily. We will calculate the number of observations from each site, the mean annual, maximum annual, and minimum annual value for all variables.

Our list of variables includes:

- Mean_Annual_Temperature_C: average of all available data
- Mean_max_temperature_C: average of maximums for each year
- Mean_min_temperature_C: average of minimums for each year
- Temperature_st_dev: standard deviation of all available data
- Temperature_n: total number of data points
- Temperature_years: number of years in data set
- Mean_Annual_Salinity_ppt: average of all available data
- Mean_min_Salinity_ppt: average of minimums for each year
- Mean_max_Salinity_ppt: average of maximums for each year
- Salinity_st_dev: standard deviation of all available data
- Salinity_n: total number of data points
- Salinity_years: number of years in data set

```r
#Calculate temperature variables.
#Calculate temperature variables.
Mean_Annual_Temperature_C <- mean(DEBY_env$temp)
Mean_max_temperature_C <- mean(DEBY_envryear_temp$max_temp)
Mean_min_temperature_C <- mean(DEBY_envryear_temp$min_temp)
Temperature_st_dev <- sd(DEBY_env$temp)
Temperature_n <- nrow(DEBY_env)
Temperature_years <- nrow(DEBY_envryear_temp)

#Create a data frame to store the temperature results
DEBY_temp <- cbind(site_name, download_date, source_description, lat, lon, firstyear, finalyear, Mean_A
print(DEBY_temp)
```

```
##      site_name download_date
## [1,] "DEBY"    "04-02-2024"
##      source_description                                    lat
## [1,] "Virginia Estuarine and Coastal Observing System, VIMS" "37.247284"
##      lon          firstyear finalyear Mean_Annual_Temperature_C
## [1,] "-76.499369" "2003"    "2024"    "17.1242424156129"
##      Mean_max_temperature_C Mean_min_temperature_C Temperature_st_dev
## [1,] "30.1437727272727"     "3.54609090909091"     "8.07505645766884"
##      Temperature_n Temperature_years collection_type
## [1,] "667193"      "22"              "continuous"
```

```r
# Write to a unique new CSV file
write.csv(DEBY_temp, "../../data/envr_of_origin/env_summarized/Temperature/DEBY_temp_summarized.csv", r
```

```r
#Calculate the salinity variables
Mean_Annual_Salinity_ppt <- mean(DEBY_env$salinity)
Mean_max_Salinity_ppt <- mean(DEBY_envryear_sal$max_salinity)
Mean_min_Salinity_ppt <- mean(DEBY_envryear_sal$min_salinity)
Salinity_st_dev <- sd(DEBY_env$salinity)
Salinity_n <- nrow(DEBY_env)
Salinity_years <- nrow(DEBY_envryear_sal)


#Create a data frame to store the temperature results
DEBY_salinity <- cbind(site_name, download_date, source_description, lat, lon, firstyear, finalyear, Mea
print(DEBY_salinity)
```

```
##      site_name download_date
## [1,] "DEBY"    "04-02-2024"
##      source_description                                    lat
## [1,] "Virginia Estuarine and Coastal Observing System, VIMS" "37.247284"
##      lon          firstyear finalyear Mean_Annual_Salinity_ppt
## [1,] "-76.499369" "2003"    "2024"    "19.3391307162995"
##      Mean_max_Salinity_ppt Mean_min_Salinity_ppt Salinity_st_dev    Salinity_n
## [1,] "24.1890909090909"    "12.9063636363636"    "2.56014876611224" "667193"
##      Salinity_years collection_type
## [1,] "22"           "continuous"
```

```
# Write to the combined file with all sites
# Write to a unique new CSV file
write.csv(DEBY_salinity, "../../data/envr_of_origin/env_summarized/Salinity/DEBY_salinity_summarized.csv
```