JR_corrected_raw_env

2024-10-22

```
setwd("/Users/nicolemongillo/Desktop/GitHub/MVP_Chesapeake_VIMS_hatchery/src/NM")
```

```
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 10/01/2024
#Source - https://www.ndbc.noaa.gov/historical_data.shtml#ocean
#The site was sampled continuously every hour from 2008-2019. When compared to data from VIMS water qua

#Create text strings with metadata information that we want to include in the final data frame.
download_date <- ("10-01-2024")
source_description <- ("NOAA National Buoy Data Center (NBDC), Chesapeake Bay Interpretive Buoy System
site_name <- ("JR")
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 the James River, Jamestown, VA. The ID_Site for this site J

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

#set working directory to files location
setwd("/Users/nicolemongillo/Desktop/GitHub/MVP_Chesapeake_VIMS_hatchery/data/envr_raw_data")

#read in file
raw_JR_env <- read.csv("/Users/nicolemongillo/Desktop/GitHub/MVP_Chesapeake_VIMS_hatchery/data/envr_raw

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

```
##        X             datetime           temp_NBDC     salinity_NBDC
##  Min.   :     1   Length:113785      Min.   : 0.00   Min.   : 0.000
##  1st Qu.: 28447   Class :character   1st Qu.:11.80   1st Qu.: 0.400
##  Median : 56893   Mode  :character   Median :21.50   Median : 2.600
##  Mean   : 56893                      Mean   :19.97   Mean   : 3.268
##  3rd Qu.: 85339                      3rd Qu.:27.60   3rd Qu.: 5.300
##  Max.   :113785                      Max.   :99.00   Max.   :99.000
##                                      NA's   :349     NA's   :349
##  temp_corrected  salinity_corrected
##  Min.   :-1.40   Min.   : 12.10
##  1st Qu.:10.40   1st Qu.: 12.50
##  Median :20.10   Median : 14.70
##  Mean   :18.57   Mean   : 15.37
##  3rd Qu.:26.20   3rd Qu.: 17.40
##  Max.   :97.60   Max.   :111.10
##  NA's   :349     NA's   :349
```

```
#remove extra column X
raw_JR_env <- subset(raw_JR_env, select = -c(X))

#remove row with no time in the datetime column
raw_JR_env <- raw_JR_env[-c(1), ]

raw_JR_env$datetime <- as.POSIXct(raw_JR_env$datetime, "%Y-%m-%d %H:%M:%S", tz = "")

colnames(raw_JR_env) <- c("datetime", "temp", "salinity", "corrected_temp", "corrected_salinity")
```

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

```
#Store variables that we will include in the final data frame. Lat and lon data from this site: https:/
lat <- 37.211
lon <- -76.787
firstyear <- 2008
finalyear <- 2019
```

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.

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

filtered_JR_env <- filtered_JR_sal %>%
    filter(between(corrected_temp, 0, 40))

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

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   12.10   12.50   14.70   15.19   17.40   24.10
```

```
print(summary(filtered_JR_env$corrected_temp))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   10.50   20.20   18.48   26.20   33.60
```

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

# check for NAs
count.nas_env <- is.na(JR_env) # store our NAs in a variable
summary(count.nas_env) # we have 3131 NAs in datetime
```

```
##   datetime          temp          salinity       corrected_temp
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:109712    FALSE:112843    FALSE:112843    FALSE:112843
##  TRUE :3131
##  corrected_salinity
##  Mode :logical
##  FALSE:112843
##
```

```
JR_env <- na.omit(JR_env)

#re-check for NAs
count.nas_env <- is.na(JR_env) # store our NAs in a variable
summary(count.nas_env) # we have no NAs in datetime
```

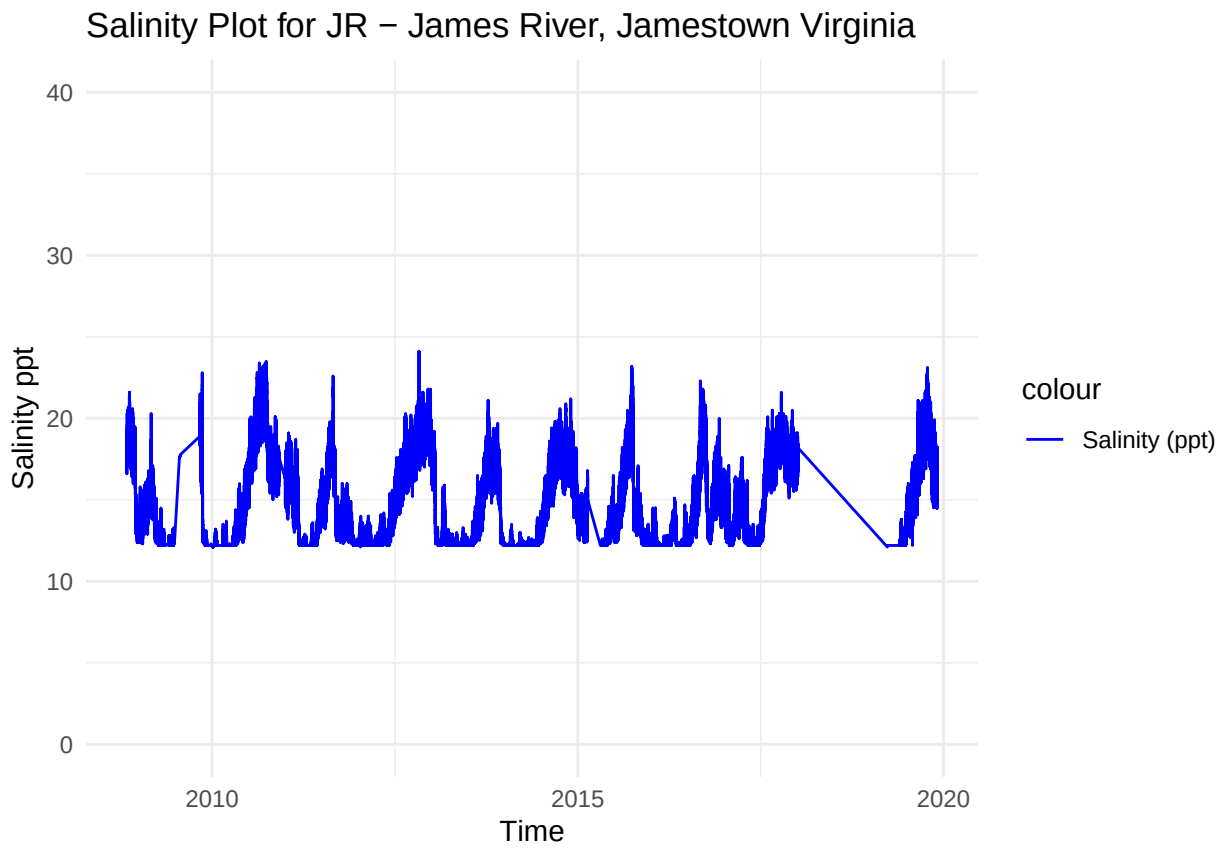
```
##   datetime          temp          salinity       corrected_temp
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:109712    FALSE:109712    FALSE:109712    FALSE:109712
##  corrected_salinity
##  Mode :logical
##  FALSE:109712
```

#Data sets for violin plots

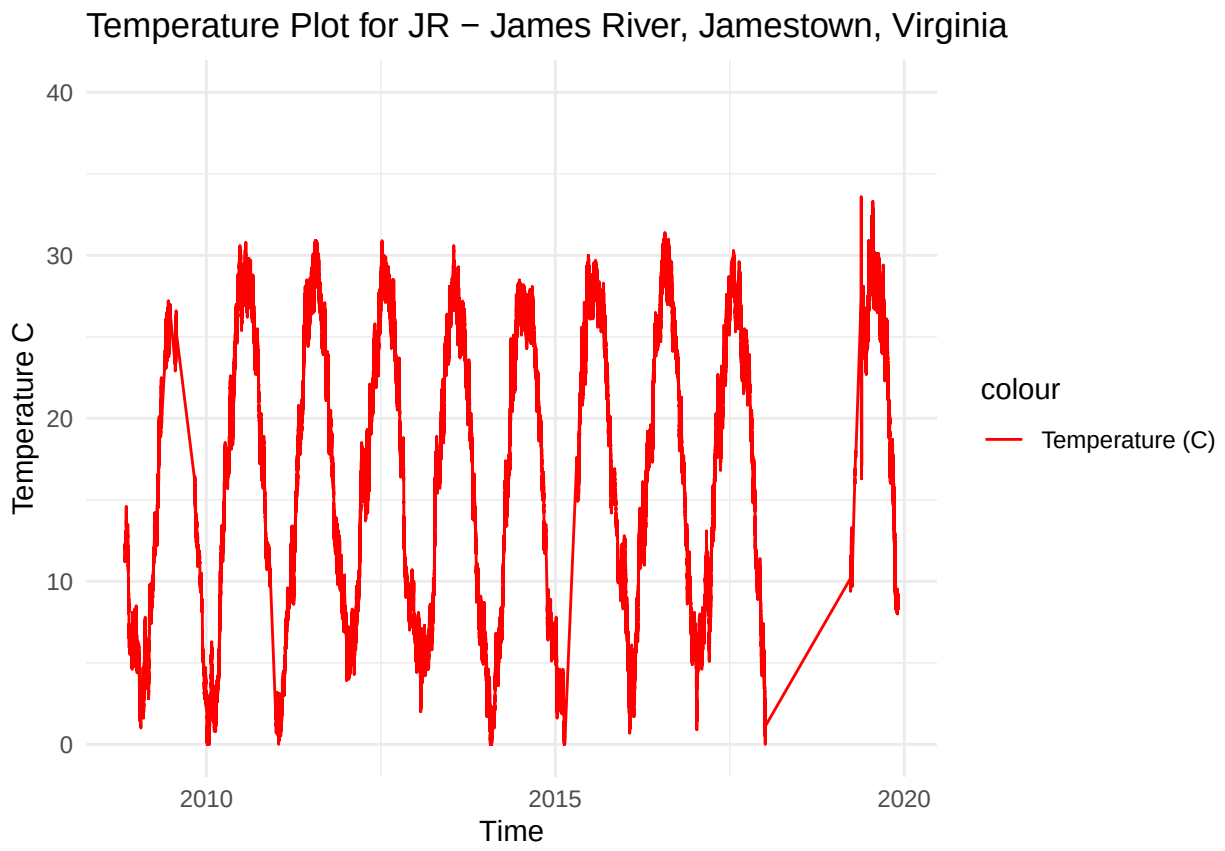
```
#add site name and create new data frame with full envr data set
JR_env_full <- JR_env %>%
  mutate(site_name, site_name = "JR")

#reorder columns with site_name first
JR_env_full <- JR_env_full[, c(6, 1, 2, 3, 4, 5)]

JR_temp_full <-JR_env_full[, c(1,2,3,5)]

JR_sal_full <- JR_env_full[, c(1,2,4,6)]

#save JR_env_full as csv for future analyses
write.csv(JR_temp_full, "/Users/nicolemongillo/Desktop/GitHub/MVP_Chesapeake_VIMS_hatchery/data/envr_ra

write.csv(JR_sal_full, "/Users/nicolemongillo/Desktop/GitHub/MVP_Chesapeake_VIMS_hatchery/data/envr_raw
```

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(JR_env, aes(x = datetime)) +
    geom_line(aes( y = corrected_salinity, color = "Salinity (ppt)")) +
    ylim(0,40) +
    labs(x = "Time", y = "Salinity ppt", title = "Salinity Plot for JR - James River, Jamestown Virginia
    scale_color_manual(values = c("Salinity (ppt)" = "blue")) +
    theme_minimal()

salplot
```




```
tempplot <- ggplot(JR_env, aes(x = datetime)) +
    geom_line(aes(y = corrected_temp, color = "Temperature (C)")) +
    ylim(0, 40) +
    labs(x = "Time", y = "Temperature C", title = "Temperature Plot for JR - James River, Jamestown, Vi
    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.
JR_envrmonth_sal <- JR_env %>%
    mutate(year = year(datetime), month = month(datetime)) %>%
    group_by(year, month) %>%
    summarise(
      min_salinity = min(corrected_salinity),
      max_salinity = max(corrected_salinity),
      mean_salinity = mean(corrected_salinity),
      length_salinity = length(corrected_salinity))
```

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

```
JR_envrmonth_temp <- JR_env %>%
    mutate(year = year(datetime), month = month(datetime)) %>%
    group_by(year, month) %>%
    summarise(
      min_temp = min(corrected_temp),
      max_temp = max(corrected_temp),
```

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

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

```
head(JR_envrmonth_sal)
```

```
## # A tibble: 6 x 6
## # Groups:   year [2]
##    year month min_salinity max_salinity mean_salinity length_salinity
##   <dbl> <dbl>        <dbl>        <dbl>         <dbl>           <int>
## 1  2008    11         16.4         21.6          18.8             596
## 2  2008    12         12.4         20.4          15.1             687
## 3  2009     1         12.3         15.9          13.3             571
## 4  2009     2         12.9         17.1          14.8             557
## 5  2009     3         12.3         20.3          14.6             648
## 6  2009     4         12.2         14.5          12.5             566
```

```
head(JR_envrmonth_temp)
```

```
## # A tibble: 6 x 6
## # Groups:   year [2]
##    year month min_temp max_temp mean_temp length_temp
##   <dbl> <dbl>    <dbl>    <dbl>     <dbl>       <int>
## 1  2008    11      5.5     14.6     10.1           596
## 2  2008    12      4.6      8.5      6.58          687
## 3  2009     1      1        6.9      4.01          571
## 4  2009     2      1.6      7.8      4.90          557
## 5  2009     3      2.8     12.1      7.81          648
## 6  2009     4     11.1     20.1     14.4           566
```

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

JR_envryear_temp <- JR_env %>%
    mutate(year = year(datetime)) %>%
    group_by(year) %>%
    summarise(
      min_temp = min(corrected_temp),
      max_temp = max(corrected_temp),
      mean_temp = mean(corrected_temp))

head(JR_envryear_sal)
```

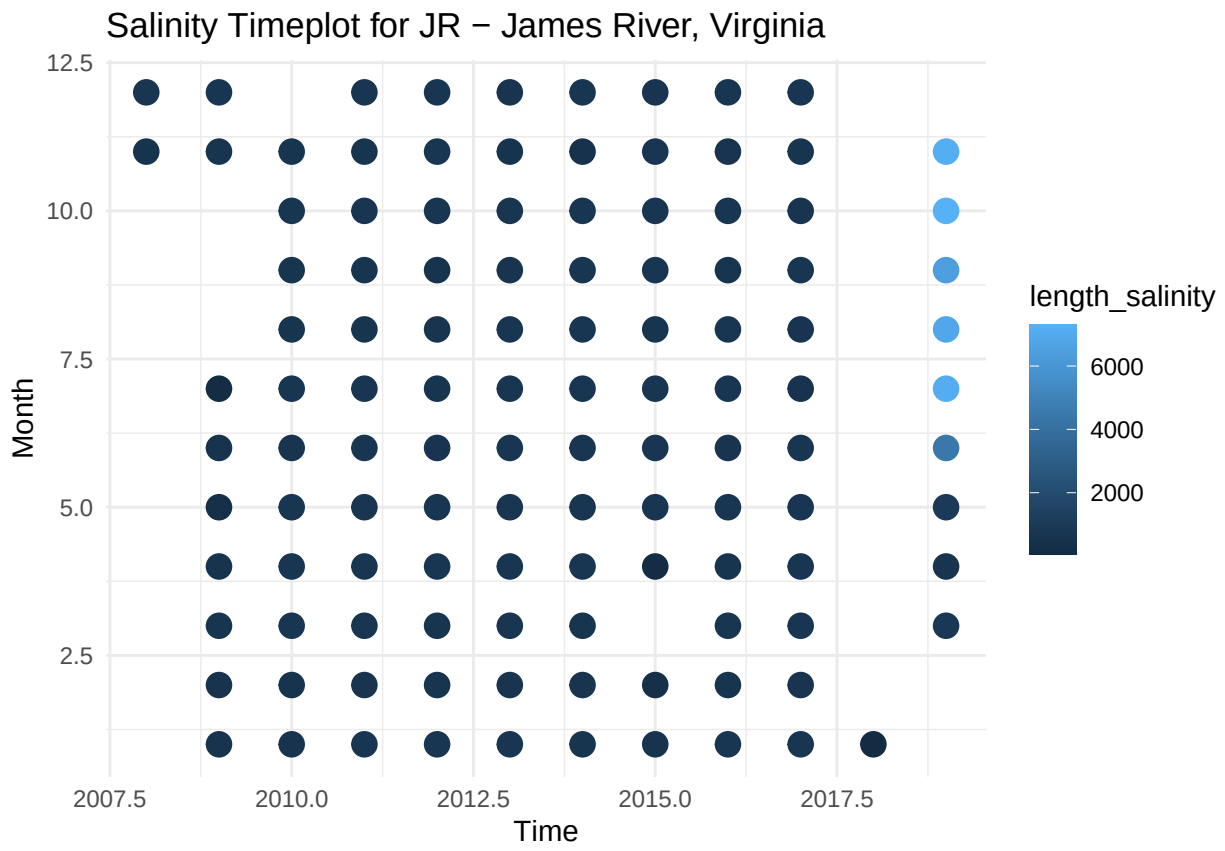
```
## # A tibble: 6 x 4
##    year min_salinity max_salinity mean_salinity
##   <dbl>        <dbl>        <dbl>         <dbl>
## 1  2008         12.4         21.6          16.8
## 2  2009         12.2         22.8          13.6
## 3  2010         12.1         23.5          15.6
## 4  2011         12.2         22.6          14.1
## 5  2012         12.1         24.1          15.3
## 6  2013         12.2         21.1          13.8
```

```
head(JR_envryear_temp)
```

```
## # A tibble: 6 x 4
##    year min_temp max_temp mean_temp
##   <dbl>    <dbl>    <dbl>     <dbl>
## 1  2008      4.6     14.6      8.24
## 2  2009      1       27.2     11.5
## 3  2010      0       30.8     17.5
## 4  2011      0       30.9     16.4
## 5  2012      3.9     30.9     16.7
## 6  2013      2       30.6     15.9
```

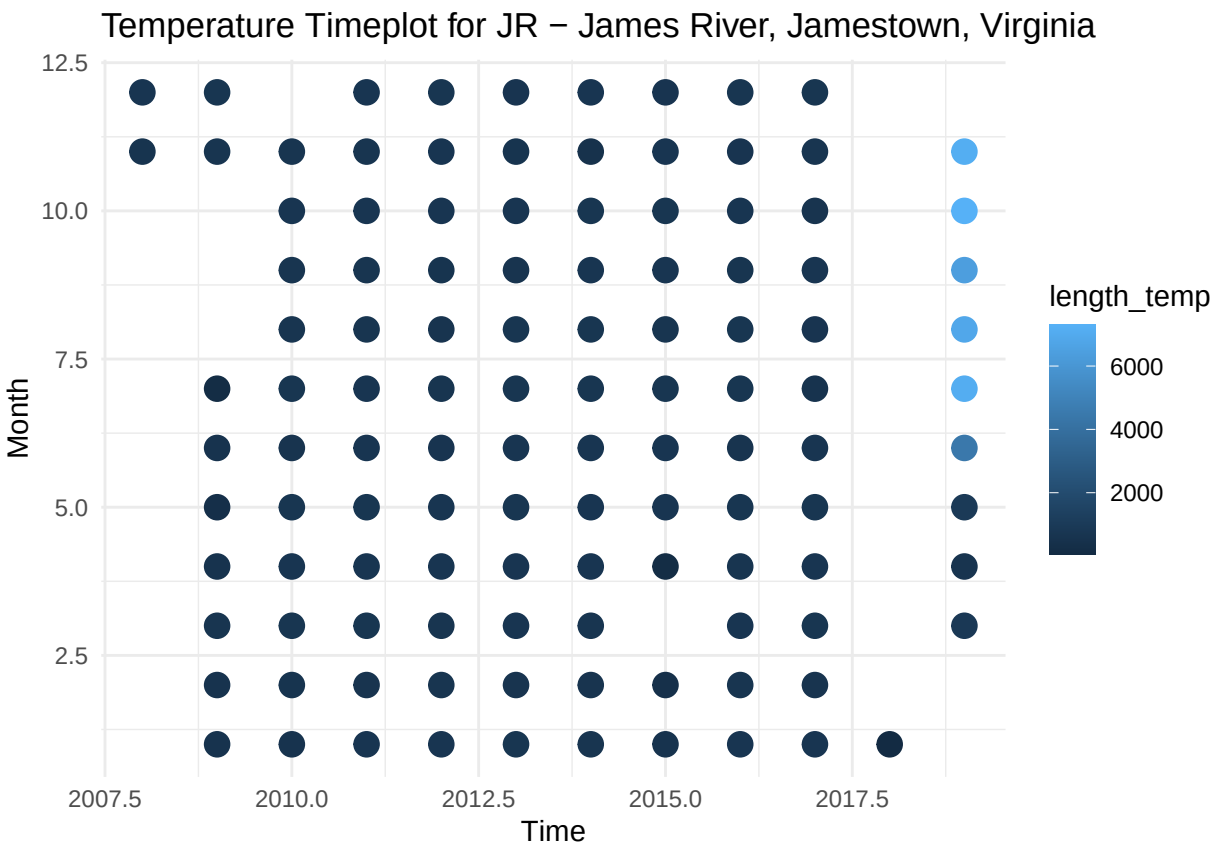
Plot the months and years of data collection to check if there are any collection gaps in the data.

```
timeplot <- ggplot(JR_envrmonth_sal, aes(x = year)) +
    geom_point(aes(y = month, color = length_salinity), size = 4) +
    labs(x = "Time", y = "Month", title = "Salinity Timeplot for JR - James River, 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(JR_envrmonth_temp, aes(x = year)) +
    geom_point(aes(y = month, color = length_temp), size = 4) +
    labs(x = "Time", y = "Month", title = "Temperature Timeplot for JR - James River, Jamestown, Virgini
    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(JR_env$corrected_temp)
Mean_max_temperature_C <- mean(JR_envryear_temp$max_temp)
Mean_min_temperature_C <- mean(JR_envryear_temp$min_temp)
Temperature_st_dev <- sd(JR_env$corrected_temp)
Temperature_n <- nrow(JR_env)
Temperature_years <- nrow(JR_envryear_temp)

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

```
##      site_name download_date
## [1,] "JR"      "10-01-2024"
##      source_description
## [1,] "NOAA National Buoy Data Center (NBDC), Chesapeake Bay Interpretive Buoy System with values cor
##      lat      lon       firstyear finalyear Mean_Annual_Temperature_C
## [1,] "37.211" "-76.787" "2008"    "2019"    "18.5346352267756"
##      Mean_max_temperature_C Mean_min_temperature_C Temperature_st_dev
## [1,] "26.8416666666667"     "1.75833333333333"     "8.48879806824356"
##      Temperature_n Temperature_years collection_type
## [1,] "109712"      "12"              "continuous"
```

```r
# Write to a unique new CSV file
write.csv(JR_temp, "/Users/nicolemongillo/Desktop/GitHub/MVP_Chesapeake_VIMS_hatchery/data/envr_raw_data
```

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


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

```
##      site_name download_date
## [1,] "JR"      "10-01-2024"
##      source_description
## [1,] "NOAA National Buoy Data Center (NBDC), Chesapeake Bay Interpretive Buoy System with values cor
##      lat      lon       firstyear finalyear Mean_Annual_Salinity_ppt
## [1,] "37.211" "-76.787" "2008"    "2019"    "15.2082260828351"
##      Mean_max_Salinity_ppt Mean_min_Salinity_ppt Salinity_st_dev    Salinity_n
## [1,] "22.1833333333333"    "12.5416666666667"    "2.70141647539953" "109712"
##      Salinity_years collection_type
## [1,] "12"           "continuous"
```

```r
# Write to the combined file with all sites
# Write to a unique new CSV file
write.csv(JR_salinity, "/Users/nicolemongillo/Desktop/GitHub/MVP_Chesapeake_VIMS_hatchery/data/envr_raw
```