SEACAR Continuous Water Quality Analysis: Water Temperature Northwest Region

Last compiled on 18 April, 2022

Contents

Purpose

The purpose of this script is to analyze the continuous water temperature data that is created from the SEACAR database, apply filtering criteria, create summary plots, and perform seasonal Kendall Tau analysis for each program location and summary statistics for values measured at the desired depth.

All scripts and outputs can be found on the SEACAR GitHub repository:

https://github.com/FloridaSEACAR/SEACAR_Panzik

Note: The top 2% of data is excluded when computing mean and standard deviations in plotting sections solely for the purpose of getting y-axis scales. The exclusion of the top 2% is not used in any statistics that are exported.

Adjustable Inputs

This is placed early so that is is easier to edit parameters that users may want to adjust.

The first variable is whether you want to create the summary plots in the appendices. If you want to see all appendix plots, set `APP_Plots` to `TRUE`. If you would like to only perform the analysis and export the data files with minimal plots, set `APP_Plots` to `FALSE`. This option is available because generating the plots in the appendices increases the processing time significantly.

Since the file names all have similar structure with only the parameter name being varied, the code below sets variables to include standard string information that is the same across all data files.

This includes: the raw data directory (`in_dir`), output file directory (`out_dit`), file prefix (`file_pref`), date the files were created from the database (`file_date`), the name of the parameter of interest (`param_name`), and region location (`region`). The complete file name is created by pasting all of the strings together with the specific parameter name without spaces (`paste0` command).

```
APP_Plots <- TRUE
in_dir <- "data/"
out_dir <- "output/"
file_pref <- "Combined_WQ_WC_NUT_cont_"
file_date <- "2022-Apr-12"
param_name <- "Water_Temperature"
region <- "NW"
```

Libraries

Loads libraries used in the script. The inclusion of `scipen` option limits how frequently R defaults to scientific notation.

```
library(knitr)
library(data.table)
library(dplyr)
library(lubridate)
library(ggplot2)
library(ggpubr)
library(scales)
library(EnvStats)
library(tidyr)
options(scipen = 999)
```

File Import

Creates file name from inputs above and read in the file from txt format with pipe delimiters.

The code creates output directories for the output files if they don't exist in the directory.

The command `fread` is used because of its improved speed while handling large data files. Only columns that are used by the script are imported from the file, and are designated in the `select` input.

```
if(!file.exists("output")){
   dir.create("output")}

file_in <- paste0(in_dir, file_pref, param_name, "_", region, "-", file_date, ".txt")
data <-fread(file_in, sep = "|", header = TRUE, stringsAsFactors = FALSE,
             select = c("ManagedAreaName", "ProgramID", "ProgramName",
                        "ProgramLocationID", "SampleDate", "Year", "Month",
                        "RelativeDepth", "ResultValue", "ParameterUnits",
                        "ValueQualifier", "SEACAR_QAQCFlagCode", "Include"),
             na.strings = "")
```

Data Filtering

Most data filtering is performed on export from the database, and is indicated by the `Include` variable. `Include` values of 1 indicate the data should be used for analysis, values of 0 indicate the data should not be used for analysis. Documentation on the database filtering is provided here: SEACAR Documentation-Analysis Filters and Calculations.docx

The filtering that is performed by the script at this point removes rows that are missing values for `ResultValue` and `RelativeDepth`, and removes any activity type that has "Blank" in the description. Data passes the filtering the process if it is has an `Include` value of 1.

The script then gets the units of the parameter, sets the `SampleDate` as a date object, and creates various scales of the date to be used by plotting functions.

Because the continuous data is extensive and most measurements are taken every 15 minutes, a daily average is determined and used based on grouping `ManagedAreaName`, `ProgramID`, `ProgramName`, `ProgramLocationID`, and `SampleDate`. The new `ResultValue` is the mean of all values on that date from that specific monitoring location.

Creates a variable for each `MonitoringID` which is defined as a unique combination of `ManagedAreaName`, `ProgramID`, `ProgramAreaName`, and `ProgramLocationID`.

After the initial filtering, a second filter variable is created to determine whether enough time is represented in the managed area, which is that each managed area has 5 year or more of unique year entries for observation that pass the initial filter. If data passes the first set of filtering criteria and the time criteria, they are used in the analysis.

```
data$Include <- as.logical(data$Include)
data <- data[data$Include==TRUE,]
data <- data[!is.na(data$ResultValue),]
data <- data[!is.na(data$RelativeDepth),]
data <- data[!grep("Blank", data$ActivityType),]

if(param_name == "Water_Temperature"){
   data <- data[data$ResultValue>=-5,]
} else{
   data <- data[data$ResultValue>=0,]
```

```
}

unit <- unique(data$ParameterUnits)

data <- data %>%
   group_by(ManagedAreaName, ProgramID, ProgramName, ProgramLocationID,
            SampleDate) %>%
   summarise(Year = unique(Year), Month = unique(Month),
             RelativeDepth = unique(RelativeDepth),
             ResultValue = mean(ResultValue), Include = unique(Include))
```

```
## `summarise()` has grouped output by 'ManagedAreaName', 'ProgramID', 'ProgramName', 'ProgramLocationI
```

```
data$SampleDate <- as.Date(data$SampleDate)
data$YearMonth <- paste0(data$Month, "-", data$Year)
data$YearMonthDec <- data$Year + ((data$Month-0.5) / 12)
data$DecDate <- decimal_date(data$SampleDate)
data <- as.data.table(data[order(data$ManagedAreaName, data$ProgramID, data$ProgramName,
                                 data$ProgramLocationID), ])

data <- data %>%
   mutate(MonitoringID = group_indices(., ManagedAreaName, ProgramID,
                                       ProgramName, ProgramLocationID))
```

```
## Warning: The `...` argument of `group_keys()` is deprecated as of dplyr 1.0.0.
## Please `group_by()` first
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```

```
Mon_Years <- data[data$Include == TRUE, ] %>%
   group_by(MonitoringID) %>%
   summarize(ManagedAreaName = unique(ManagedAreaName),
             ProgramID = unique(ProgramID), ProgramName = unique(ProgramName),
             ProgramLocationID = unique(ProgramLocationID),
             Y = length(unique(Year)), RelativeDepth = unique(RelativeDepth))
Mon_Years <- as.data.table(Mon_Years[
   order(Mon_Years$MonitoringID), ])
Mon_Years$Enough_Time <- ifelse(Mon_Years$Y < 5, FALSE, TRUE)
data$Exclude_MonitoringID <- is.element(data$MonitoringID,
                                        Mon_Years$MonitoringID[
                                           Mon_Years$Enough_Time == FALSE])
data$Use_In_Analysis <- ifelse(data$Include == TRUE &
                                  data$Exclude_MonitoringID == FALSE,
                               TRUE, FALSE)
Mon_IDs <- unique(data$MonitoringID[data$Use_In_Analysis == TRUE])
Mon_IDs <- Mon_IDs[order(Mon_IDs)]
n <- length(Mon_IDs)
```

Monitoring Location Statistics

Gets summary statistics for each monitoring location. Excluded monitoring locations are not included into whether the data should be used or not. Uses piping from dplyr package to feed into subsequent steps. The following steps are performed:

1. Take the `data` variable and only include rows that have a `Use_In_Analysis` value of TRUE
2. Group data that have the same `ManagedAreaName`, `ProgramID`, `ProgramName`, `ProgramLocationID`, `Year`, and `Month`.
   - Second summary statistics consider the monitoring location grouping and `Year`.
   - Third summary statistics consider the monitoring location grouping and `Month`.
3. For each group, provide the following information: Earliest Sample Date (EarliestSampleDate), Latest Sample Date (LastSampleDate), Number of Entries (N), Lowest Value (Min), Largest Value (Max), Median, Mean, Standard Deviation, and a list of all Program IDs included in these measurements.
4. Sort the data in ascending (A to Z and 0 to 9) order based on `ManagedAreaName`, `ProgramID`, `ProgramName`, `ProgramLocationID`, `Year`, and `Month` in that order.
5. Write summary stats to a pipe-delimited .txt file in the output directory

```
Mon_YM_Stats <- data[data$Use_In_Analysis == TRUE, ] %>%
   group_by(ManagedAreaName, ProgramID, ProgramName, ProgramLocationID,
            Year, Month) %>%
   summarize(RelativeDepth = unique(RelativeDepth),
             EarliestSampleDate = min(SampleDate),
             LastSampleDate = max(SampleDate), N = length(ResultValue),
             Min = min(ResultValue), Max = max(ResultValue),
             Median = median(ResultValue), Mean = mean(ResultValue),
             StandardDeviation = sd(ResultValue))
Mon_YM_Stats <- as.data.table(Mon_YM_Stats[order(Mon_YM_Stats$ManagedAreaName,
                                                 Mon_YM_Stats$ProgramID,
                                                 Mon_YM_Stats$ProgramName,
                                                 Mon_YM_Stats$ProgramLocationID,
                                                 Mon_YM_Stats$Year,
                                                 Mon_YM_Stats$Month), ])
fwrite(Mon_YM_Stats, paste0(out_dir,"/", param_name, "_", file_date, "_", region,
                            "_MonitoringLoc_YearMonth_Stats.txt"), sep = "|")

Mon_Y_Stats <- data[data$Use_In_Analysis == TRUE, ] %>%
   group_by(ManagedAreaName, ProgramID, ProgramName, ProgramLocationID,
            Year) %>%
   summarize(RelativeDepth = unique(RelativeDepth),
             EarliestSampleDate = min(SampleDate),
             LastSampleDate = max(SampleDate), N = length(ResultValue),
             Min = min(ResultValue), Max = max(ResultValue),
             Median = median(ResultValue), Mean = mean(ResultValue),
             StandardDeviation = sd(ResultValue))
Mon_Y_Stats <- as.data.table(Mon_Y_Stats[order(Mon_Y_Stats$ManagedAreaName,
                                               Mon_Y_Stats$ProgramID,
                                               Mon_Y_Stats$ProgramName,
                                               Mon_Y_Stats$ProgramLocationID,
                                               Mon_Y_Stats$Year), ])
fwrite(Mon_Y_Stats, paste0(out_dir,"/", param_name, "_", file_date, "_", region,
                           "_MonitoringLoc_Year_Stats.txt"), sep = "|")
```

```
Mon_M_Stats <- data[data$Use_In_Analysis == TRUE, ] %>%
   group_by(ManagedAreaName, ProgramID, ProgramName, ProgramLocationID,
            Month) %>%
   summarize(RelativeDepth = unique(RelativeDepth),
             EarliestSampleDate = min(SampleDate),
             LastSampleDate = max(SampleDate), N = length(ResultValue),
             Min = min(ResultValue), Max = max(ResultValue),
             Median = median(ResultValue), Mean = mean(ResultValue),
             StandardDeviation = sd(ResultValue))
Mon_M_Stats <- as.data.table(Mon_M_Stats[order(Mon_M_Stats$ManagedAreaName,
                                               Mon_M_Stats$ProgramID,
                                               Mon_M_Stats$ProgramName,
                                               Mon_M_Stats$ProgramLocationID,
                                               Mon_M_Stats$Month), ])

fwrite(Mon_M_Stats, paste0(out_dir,"/", param_name, "_", file_date, "_", region,
                           "_MonitoringLoc_Month_Stats.txt"), sep = "|")
```

Seasonal Kendall Tau Analysis

Gets seasonal Kendall Tau statistics using the `kendallSeasonalTrendTest` from the `EnvStats` package. The `Trend` parameter is determined from a user-defined function based on the median, Senn slope, and p values from the data. Analysis modified from that performed at The Water Atlas: https://sarasota.wateratlas.usf.edu/water-quality-trends/#analysis-overview

The following steps are performed:

1. Define the trend function.
2. Take the `data` variable and only include rows that have a `Use_In_Analysis` value of `TRUE`
3. Group data that have the same `ManagedAreaName`, `ProgramID`, `ProgramName`, and `ProgramLocationID`.
4. For each group, provides the following information: Earliest Sample Date (EarliestSampleDate), Latest Sample Date (LastSampleDate), Number of Entries (N), Lowest Value (Min), Largest Value (Max), Median, Mean, Standard Deviation,
5. For each group, a temporary variable is created to run the `kendallSeasonalTrendTest` function using the `Year` values for year, and `Month` as the seasonal qualifier, and Trend.
   - An independent.obs value of `TRUE` indicates that the data should be treated as not being serially auto-correlated. An independent.obs value of `FALSE` indicates that it is treated as being serially auto-correlated, but also requires one observation per season per year for the full time of observation.
   - tau, Senn Slope (SennSlope), Senn Intercept (SennIntercept), and p are extracted from the model results.
6. The two stats tables are merged based on similar groups, and then Trend is determined from the user-defined function.
7. Write summary stats to a pipe-delimited .txt file in the output directory

- Click this text to open Git directory with output files

8. Add the Monitoring IDS to `KT.Stats` for easier use while plotting.

```
tauSeasonal <- function(data, independent, stats.median, stats.minYear,
                        stats.maxYear) {
   tau <- NULL
   tryCatch({
      ken <-
         kendallSeasonalTrendTest(
            y = data$ResultValue,
            season = data$Month,
            year = data$Year,
            independent.obs = independent
         )
      tau <- ken$estimate[1]
      p <- ken$p.value[2]
      slope <- ken$estimate[2]
      intercept <- ken$estimate[3]
      trend <- trend_calculator(slope, stats.median, p)
   }, warning = function(w) {
      print(w)
   }, error = function(e) {
      print(e)
   }, finally = {
      if (!exists("tau")) {
         tau <- NULL
      }
      if (!exists("p")) {
         p <- NULL
      }
      if (!exists("slope")) {
         slope <- NULL
      }
      if (!exists("intercept")) {
         intercept <- NULL
      }
      if (!exists("trend")) {
         trend <- NULL
      }
   })
   KT <-c(unique(data$MonitoringID),
          independent,
          stats.median,
          nrow(data),
          stats.minYear,
          stats.maxYear,
          tau,
          p,
          slope,
          intercept,
          trend)
   return(KT)
}
```

```r
runStats <- function(data) {
   data$Index <- as.Date(data$SampleDate) # , "%Y-%m-%d")
   data$ResultValue <- as.numeric(data$ResultValue)
   # Calculate basic stats
   stats.median <- median(data$ResultValue, na.rm = TRUE)
   stats.minYear <- min(data$Year, na.rm = TRUE)
   stats.maxYear <- max(data$Year, na.rm = TRUE)
   # Calculate Kendall Tau and Slope stats, then update appropriate columns and table
   KT <- tauSeasonal(data, TRUE, stats.median,
                     stats.minYear, stats.maxYear)
   if (is.null(KT[11])) {
      KT <- tauSeasonal(data, FALSE, stats.median,
                        stats.minYear, stats.maxYear)
   }
   if (is.null(KT.Stats) == TRUE) {
      KT.Stats <- KT
   } else{
      KT.Stats <- rbind(KT.Stats, KT)
   }
   return(KT.Stats)
}
trend_calculator <- function(slope, median_value, p) {
   trend <-
      if (p < .05 & abs(slope) > abs(median_value) / 10.) {
         if (slope > 0) {
            2
         }
         else {
            -2
         }
      }
   else if (p < .05 & abs(slope) < abs(median_value) / 10.) {
      if (slope > 0) {
         1
      }
      else {
         -1
      }
   }
   else
      0
   return(trend)
}
KT.Stats <- NULL
# Loop that goes through each managed area. List of managed areas stored in MA_Years$ManagedAreaName
c_names <- c("MonitoringID", "Independent", "Median", "N", "EarliestYear",
             "LatestYear", "tau", "p", "SennSlope", "SennIntercept", "Trend")
if(n==0){
   c_names <- c("ManagedAreaName", "ProgramID", "ProgramName",
                "ProgramLocationID", "Independent", "Median", "N",
                "EarliestYear", "LatestYear", "tau", "p", "SennSlope",
                "SennIntercept", "Trend")
   KT.Stats <- data.frame(matrix(ncol=14, nrow=0))
```

```
    colnames(KT.Stats) <- c_names
    fwrite(KT.Stats, paste0(out_dir,"/", param_name, "_", file_date, "_", region,
                            "_KendallTau_Stats.txt"), sep = "|")
} else{
    for (i in 1:n) {
        values <- data[data$Use_In_Analysis == TRUE &
                        data$MonitoringID == Mon_IDs[i], ]
        if (nrow(values) > 0) {
            KT.Stats <- runStats(values)
        }
    }
    KT.Stats <- as.data.frame(KT.Stats)
    if(dim(KT.Stats)[2]==1){
        KT.Stats <- as.data.frame(t(KT.Stats))
    }

    c_names <- c("MonitoringID", "Independent", "Median", "N", "EarliestYear",
                 "LatestYear", "tau", "p", "SennSlope", "SennIntercept", "Trend")
    colnames(KT.Stats) <- c_names
    rownames(KT.Stats) <- seq(1:nrow(KT.Stats))
    KT.Stats$Independent <- as.logical(KT.Stats$Independent)
    KT.Stats$Median <- as.numeric(KT.Stats$Median)
    KT.Stats$N <- as.integer(KT.Stats$N)
    KT.Stats$EarliestYear <- as.integer(KT.Stats$EarliestYear)
    KT.Stats$LatestYear <- as.integer(KT.Stats$LatestYear)
    KT.Stats$tau <- round(as.numeric(KT.Stats$tau), digits=4)
    KT.Stats$p <- round(as.numeric(KT.Stats$p), digits=4)
    KT.Stats$SennSlope <- as.numeric(KT.Stats$SennSlope)
    KT.Stats$SennIntercept <- as.numeric(KT.Stats$SennIntercept)
    KT.Stats$Trend <- as.integer(KT.Stats$Trend)
    KT.Stats <- merge.data.frame(Mon_Years[,-c("Y", "Enough_Time")], KT.Stats, by = "MonitoringID")
    KT.Stats$MonitoringID <- NULL
    fwrite(KT.Stats, paste0(out_dir,"/", param_name, "_", file_date, "_", region,
                            "_KendallTau_Stats.txt"), sep = "|")
    KT.Stats$MonitoringID <- Mon_IDs
}
```

Appendix I: Dataset Summary Box Plots

Box plots are created by using the entire data set and excludes any data that has been previously filtered out. The scripts that create plots follow this format

1. Use the data set that only has `Use_In_Analysis` of `TRUE`
2. Set what values are to be used for the x-axis, y-axis, and the variable that should determine groups for the box plots
3. Set the plot type as a box plot with the size of the outlier points
4. Create the title, x-axis, y-axis, and color fill labels
5. Set the y and x limits
6. Make the axis labels bold

7. Plot the arrangement as a set of panels

This set of box plots are grouped by year.

```
min_RV <- min(data$ResultValue[data$Include == TRUE])
mn_RV <- mean(data$ResultValue[data$Include == TRUE &
                                  data$ResultValue <
                                  quantile(data$ResultValue, 0.98)])
sd_RV <- sd(data$ResultValue[data$Include == TRUE &
                                data$ResultValue <
                                quantile(data$ResultValue, 0.98)])
y_scale <- mn_RV + 4 * sd_RV

p1 <- ggplot(data = data[data$Include == TRUE, ],
             aes(x = Year, y = ResultValue, group = Year)) +
   geom_boxplot(outlier.size = 0.5) +
   labs(subtitle = "Autoscale", x = "Year",
        y = paste0("Values (", unit, ")")) +
   theme(axis.text.x = element_text(face = "bold"),
         axis.text.y = element_text(face = "bold"))

p2 <- ggplot(data = data[data$Include == TRUE, ],
             aes(x = Year, y = ResultValue, group = Year)) +
   geom_boxplot(outlier.size = 0.5) +
   labs(subtitle = "Scaled to 4x Standard Deviation", x = "Year",
        y = paste0("Values (", unit, ")")) +
   ylim(0, y_scale) + theme(axis.text.x = element_text(face = "bold"),
                            axis.text.y = element_text(face = "bold"))

p3 <- ggplot(data = data[data$Include == TRUE, ],
             aes(x = as.integer(Year), y = ResultValue, group = Year)) +
   geom_boxplot(outlier.size = 0.5) +
   labs(subtitle = "Scaled to 4x Standard Deviation, Last 10 Years",
        x = "Year", y = paste0("Values (", unit, ")")) +
   ylim(0, y_scale) +
   scale_x_continuous(limits = c(max(data$Year) - 10.5, max(data$Year)+0.5),
                      breaks = seq(max(data$Year) - 10, max(data$Year), 2)) +
   theme(axis.text.x = element_text(face = "bold"),
         axis.text.y = element_text(face = "bold"))

set <- ggarrange(p1, p2, p3, ncol = 1)

p0 <- ggplot() + labs(title = "Summary Box Plots for Entire Data",
                      subtitle = "By Year") + theme_bw() +
   theme(plot.title = element_text(face="bold"),
         panel.border = element_blank(), panel.grid.major = element_blank(),
         panel.grid.minor = element_blank(), axis.line = element_blank())

Yset <- ggarrange(p0, set, ncol=1, heights = c(0.07, 1))
```

This set of box plots are grouped by year and month with the color being related to the month.

```r
p1 <- ggplot(data = data[data$Include == TRUE, ],
             aes(x = YearMonthDec, y = ResultValue,
                 group = YearMonth, color = as.factor(Month))) +
  geom_boxplot(outlier.size = 0.5) +
  labs(subtitle = "Autoscale", x = "Year",
       y = paste0("Values (", unit, ")"), color="Month") +
  theme(legend.position = "top", legend.box = "horizontal",
        axis.text.x = element_text(face = "bold"),
        axis.text.y = element_text(face = "bold")) +
  guides(color = guide_legend(nrow = 1))

p2 <- ggplot(data = data[data$Include == TRUE, ],
             aes(x = YearMonthDec, y = ResultValue,
                 group = YearMonth, color = as.factor(Month))) +
  geom_boxplot(outlier.size = 0.5) +
  labs(subtitle = "Scaled to 5x Standard Deviation",
       x = "Year", y = paste0("Values (", unit, ")")) +
  ylim(0, y_scale) +
  theme(legend.position = "none", axis.text.x = element_text(face = "bold"),
        axis.text.y = element_text(face = "bold"))

p3 <- ggplot(data = data[data$Include == TRUE, ],
             aes(x = YearMonthDec, y = ResultValue,
                 group = YearMonth, color = as.factor(Month))) +
  geom_boxplot(outlier.size = 0.5) +
  labs(subtitle = "Scaled to 5x Standard Deviation, Last 10 Years",
       x = "Year", y = paste0("Values (", unit, ")")) +
  ylim(0, y_scale) +
  scale_x_continuous(limits = c(max(data$Year) - 10.5, max(data$Year)+0.5),
                     breaks = seq(max(data$Year) - 10, max(data$Year), 2)) +
  theme(legend.position = "none", axis.text.x = element_text(face = "bold"),
        axis.text.y = element_text(face = "bold"))
leg <- get_legend(p1)
set <- ggarrange(leg, p1 + theme(legend.position = "none"), p2, p3, ncol = 1,
                 heights = c(0.1, 1, 1, 1))

p0 <- ggplot() + labs(title = "Summary Box Plots for Entire Data",
                      subtitle = "By Year & Month") + theme_bw() +
  theme(plot.title = element_text(face="bold"),
        panel.border = element_blank(), panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(), axis.line = element_blank())

YMset <- ggarrange(p0, set, ncol=1, heights = c(0.07, 1))
```

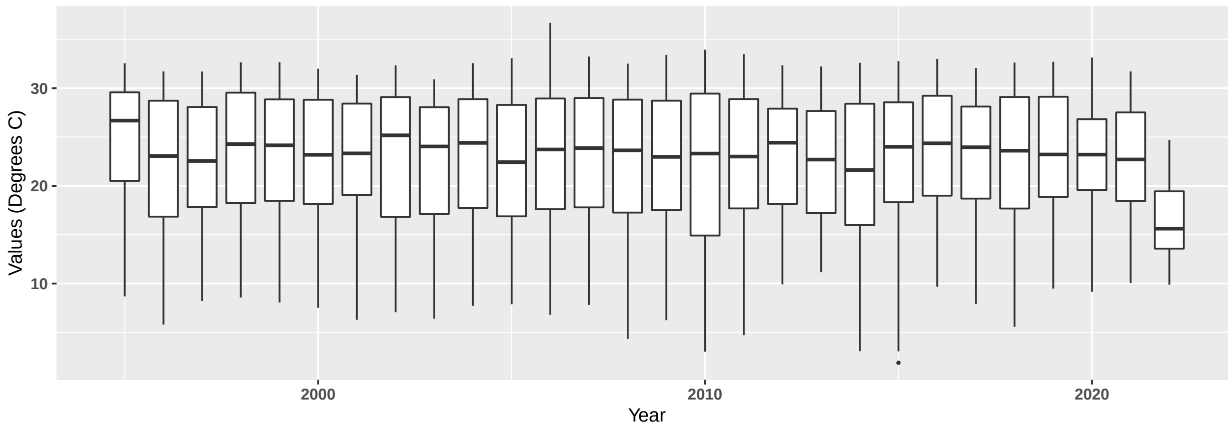
The following box plots are grouped by month with fill color being related to the month. This is designed to view potential seasonal trends.

```r
p1 <- ggplot(data = data[data$Include == TRUE, ],
             aes(x = Month, y = ResultValue,
                 group = Month, fill = as.factor(Month))) +
  geom_boxplot(outlier.size = 0.5) +
  labs(subtitle = "Autoscale", x = "Month",
       y = paste0("Values (", unit, ")"), fill="Month") +
```

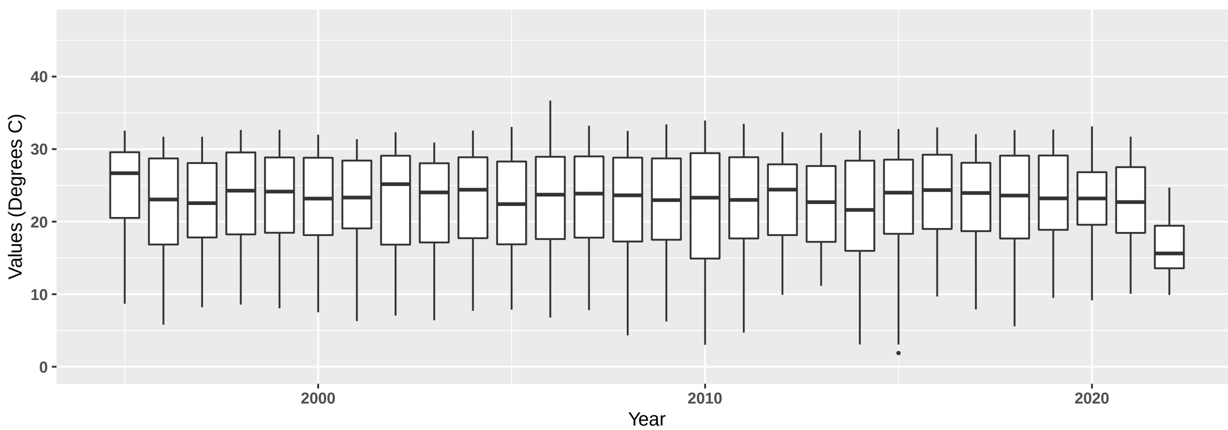
```
  scale_x_continuous(limits = c(0, 13), breaks = seq(3, 12, 3)) +
  theme(legend.position = "top", legend.box = "horizontal",
        axis.text.x = element_text(face = "bold"),
        axis.text.y = element_text(face = "bold")) +
  guides(fill = guide_legend(nrow = 1))

p2 <- ggplot(data = data[data$Include == TRUE, ],
             aes(x = Month, y = ResultValue,
                 group = Month, fill = as.factor(Month))) +
  geom_boxplot(outlier.size = 0.5) +
  labs(subtitle = "Scaled to 5x Standard Deviation",
       x = "Month", y = paste0("Values (", unit, ")")) +
  ylim(0, y_scale) +
  scale_x_continuous(limits = c(0, 13), breaks = seq(3, 12, 3)) +
  theme(legend.position = "none", axis.text.x = element_text(face = "bold"),
        axis.text.y = element_text(face = "bold"))

p3 <- ggplot(data = data[data$Include == TRUE &
                            data$Year >= max(data$Year) - 10, ],
             aes(x = Month, y = ResultValue,
                 group = Month, fill = as.factor(Month))) +
  geom_boxplot(outlier.size = 0.5) +
  labs(subtitle = "Scaled to 5x Standard Deviation, Last 10 Years",
       x = "Month", y = paste0("Values (", unit, ")")) +
  ylim(0, y_scale) +
  scale_x_continuous(limits = c(0, 13), breaks = seq(3, 12, 3)) +
  theme(legend.position = "none", axis.text.x = element_text(face = "bold"),
        axis.text.y = element_text(face = "bold"))
leg <- get_legend(p1)
set <- ggarrange(leg, p1 + theme(legend.position = "none"), p2, p3, ncol = 1,
                 heights = c(0.1, 1, 1, 1))

p0 <- ggplot() + labs(title = "Summary Box Plots for Entire Data",
                      subtitle = "By Month") + theme_bw() +
  theme(plot.title = element_text(face="bold"),
        panel.border = element_blank(), panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(), axis.line = element_blank())

Mset <- ggarrange(p0, set, ncol=1, heights = c(0.07, 1))
```

Summary Box Plots for Entire Data
By Year

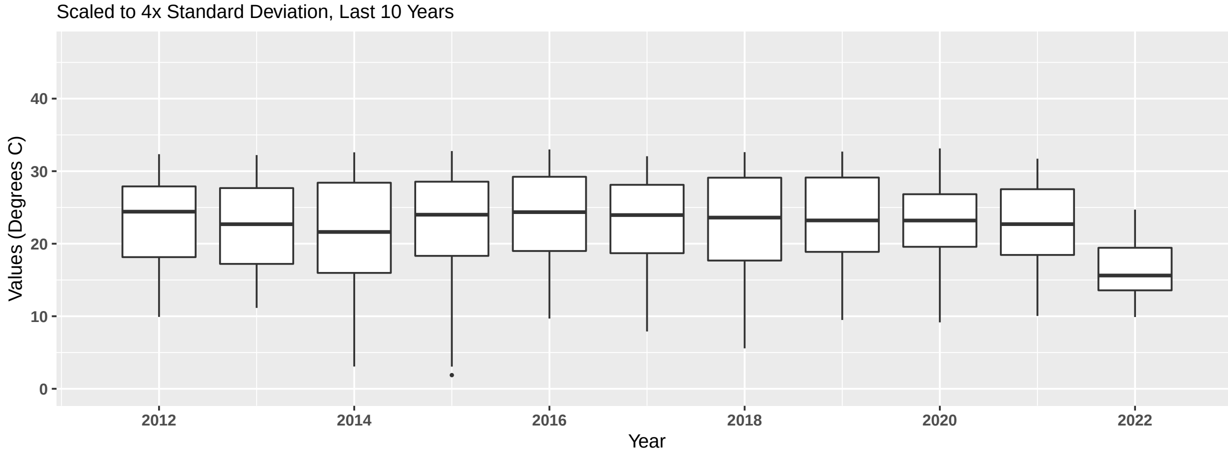
Autoscale



Scaled to 4x Standard Deviation

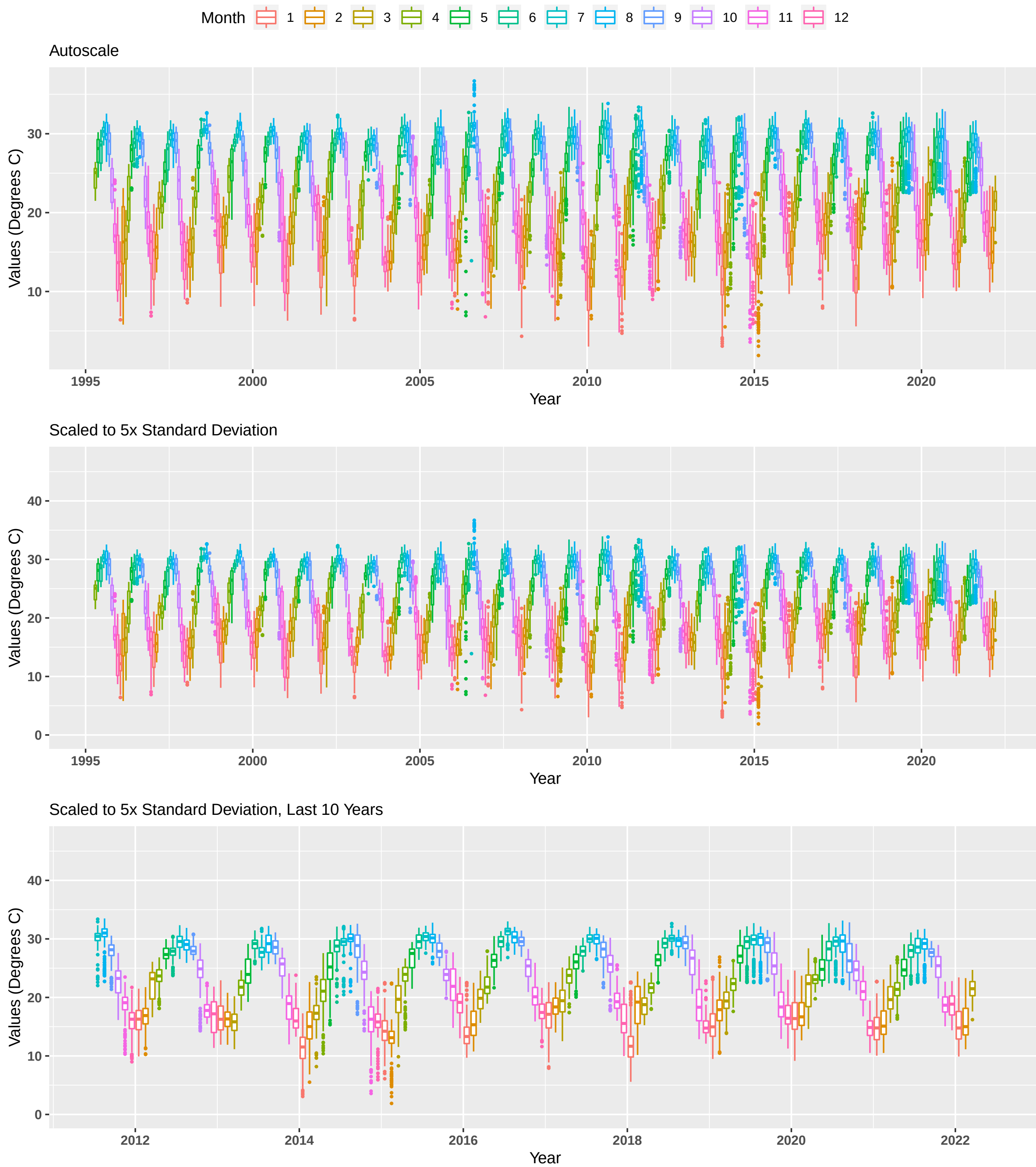


Scaled to 4x Standard Deviation, Last 10 Years

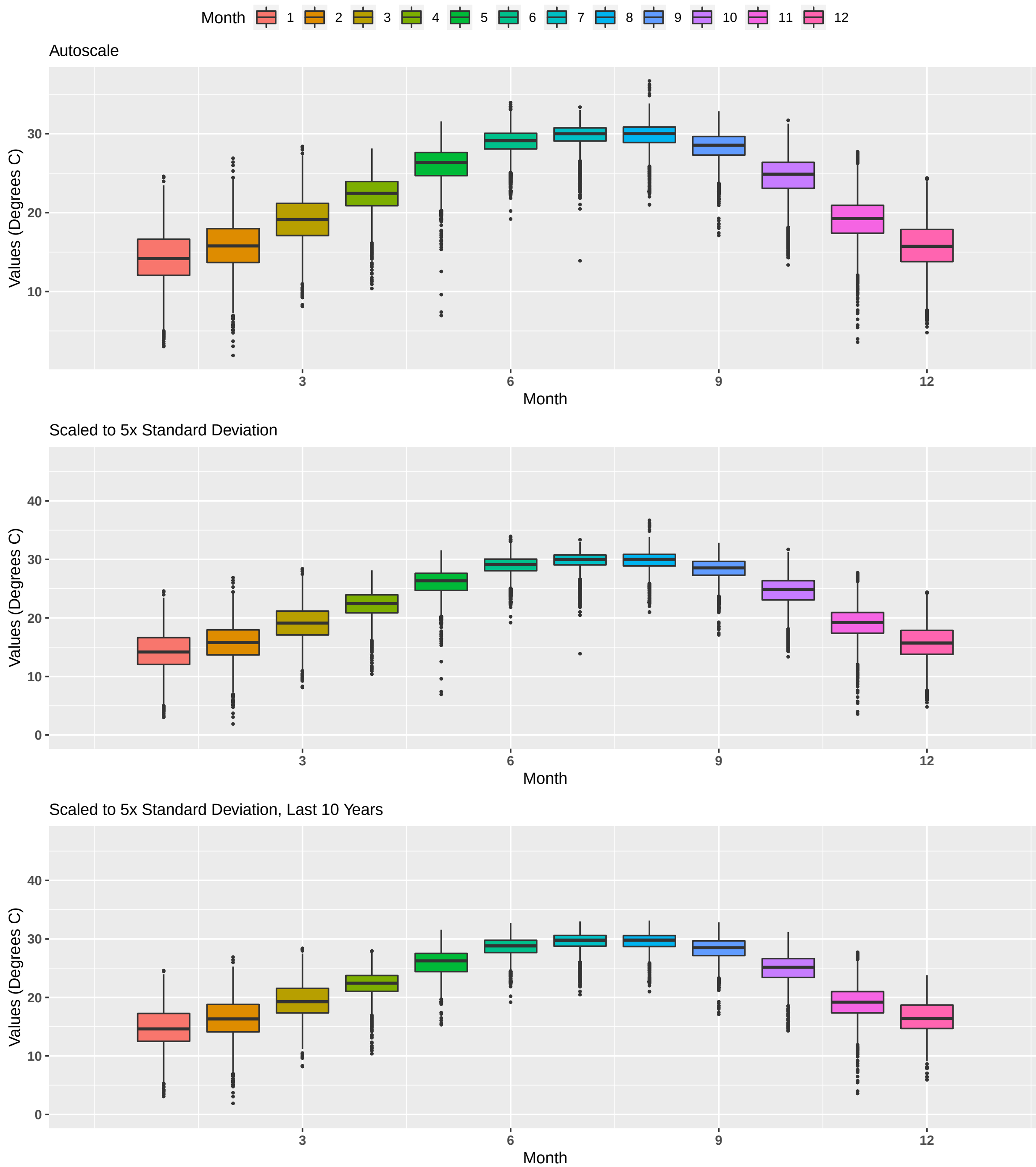


Summary Box Plots for Entire Data

By Year & Month



Summary Box Plots for Entire Data
By Month

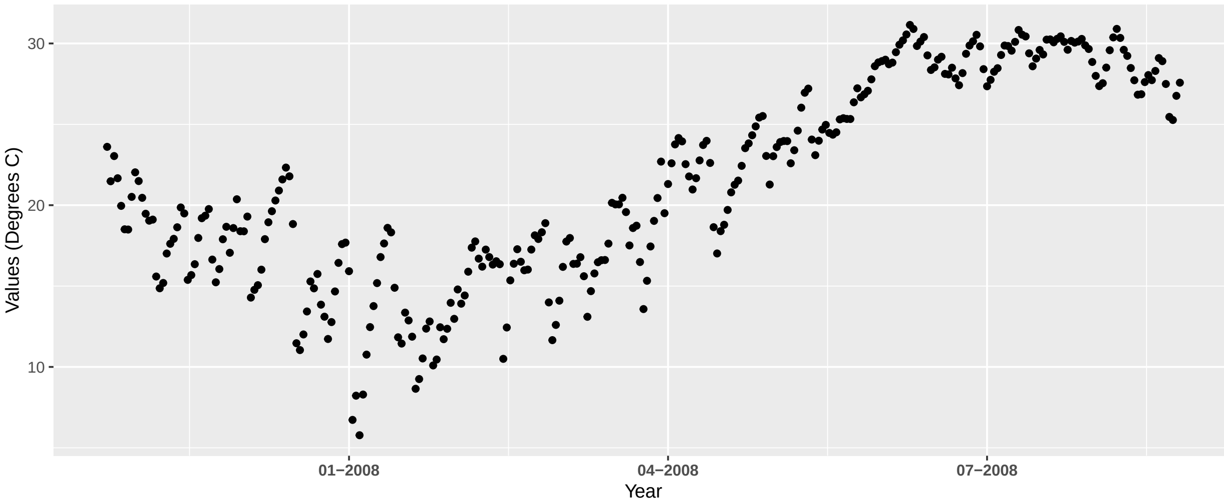


Appendix II: Excluded Monitoring Locations

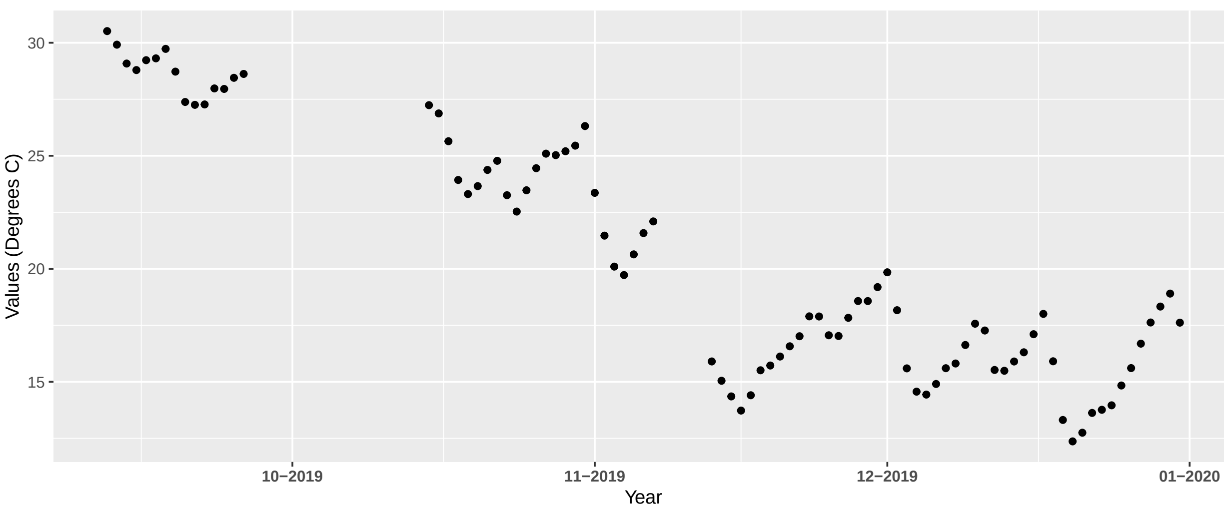
Scatter plots of data values are created for monitoring locationss that have fewer than 5 separate years of data entries.

```
Mon_Exclude <- Mon_Years[Mon_Years$Enough_Time==FALSE,]
Mon_Exclude <- Mon_Exclude[order(Mon_Exclude$MonitoringID),]
z=length(Mon_Exclude$MonitoringID)

if(z==0){
   print("There are no monitoring locations that qualify.")
} else {
   for(i in 1:z){
      MA_name <- unique(data$ManagedAreaName[
         data$MonitoringID==Mon_Exclude$MonitoringID[i]])
      Mon_name <- paste(unique(data$ProgramID[
         data$MonitoringID==Mon_Exclude$MonitoringID[i]]),
         unique(data$ProgramName[
            data$MonitoringID==Mon_Exclude$MonitoringID[i]]),
         unique(data$ProgramLocationID[
            data$MonitoringID==Mon_Exclude$MonitoringID[i]]),
         sep = " | ")

      p1<-ggplot(data=data[data$MonitoringID==Mon_Exclude$MonitoringID[i]&
                              data$Include == TRUE, ],
                 aes(x = SampleDate, y = ResultValue)) +
         geom_point() +
         labs(title=
                 paste0("Scatter Plot of Excluded Monitoring Location ",
                        MA_name, "\n", Mon_name, "\n(", Mon_Exclude$Y[i],
                        " Unique Years)"),
              subtitle="Autoscale", x = "Year",
              y = paste0("Values (", unit, ")")) +
         theme(axis.text.x = element_text(face = "bold")) +
         scale_x_date(labels = date_format("%m-%Y"))
      print(p1)
   }
}
```

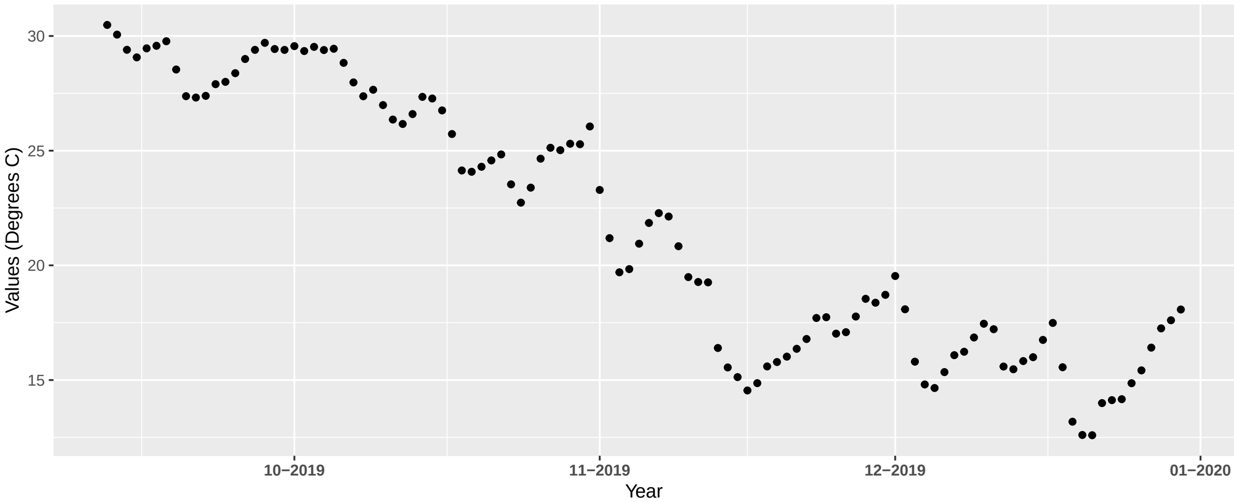
Scatter Plot of Excluded Monitoring Location Alligator Harbor Aquatic Preserve
468 | Central Panhandle Aquatic Preserves Continuous Water Quality Monitoring | CPAH
(2 Unique Years)
Autoscale



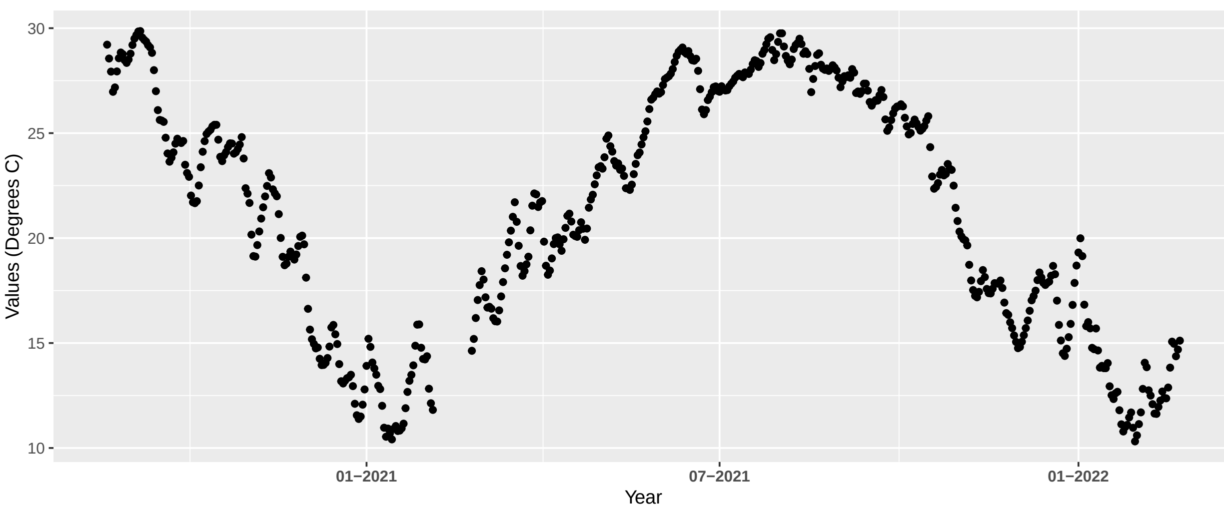
Scatter Plot of Excluded Monitoring Location Alligator Harbor Aquatic Preserve
468 | Central Panhandle Aquatic Preserves Continuous Water Quality Monitoring | CPAH2
(1 Unique Years)
Autoscale

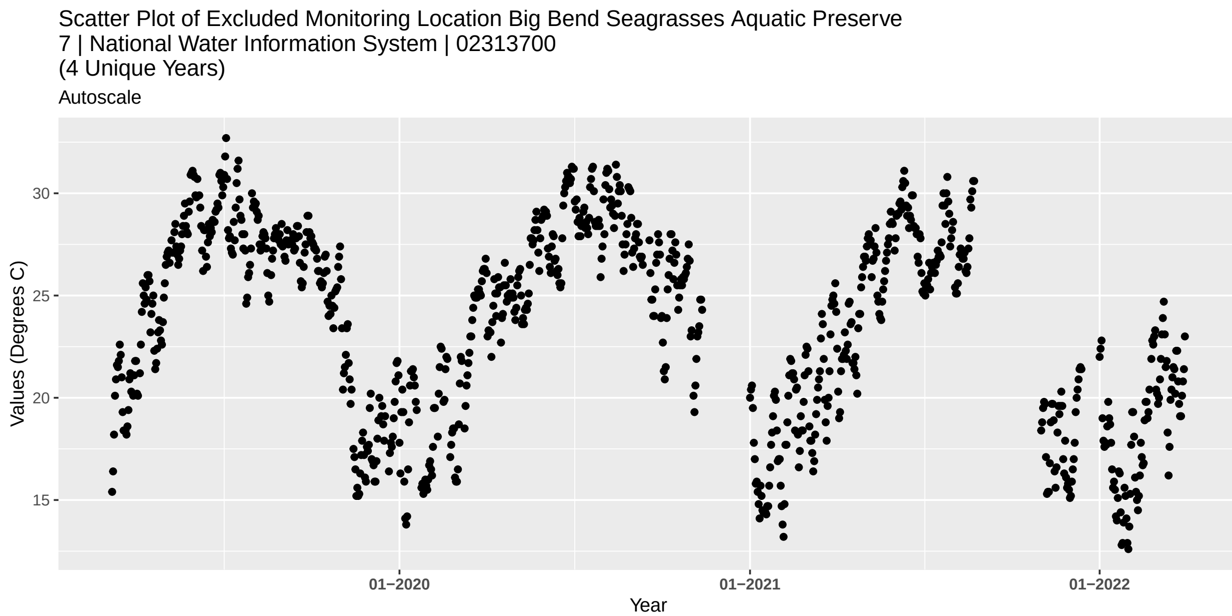


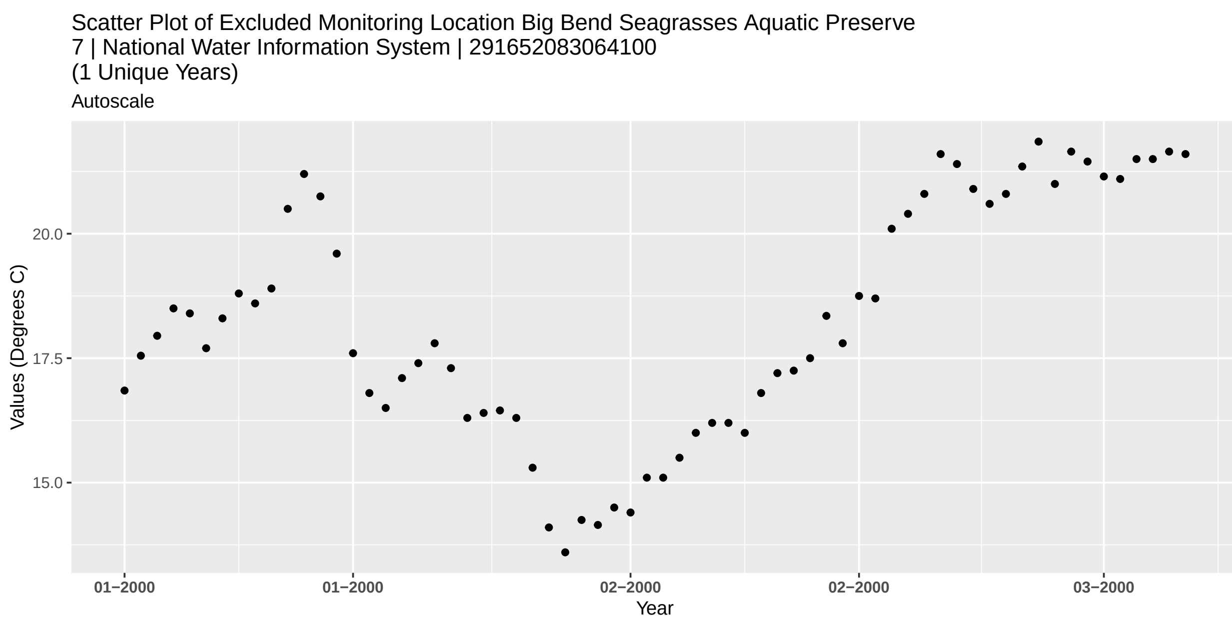
Scatter Plot of Excluded Monitoring Location Alligator Harbor Aquatic Preserve
468 | Central Panhandle Aquatic Preserves Continuous Water Quality Monitoring | CPFS
(1 Unique Years)
Autoscale



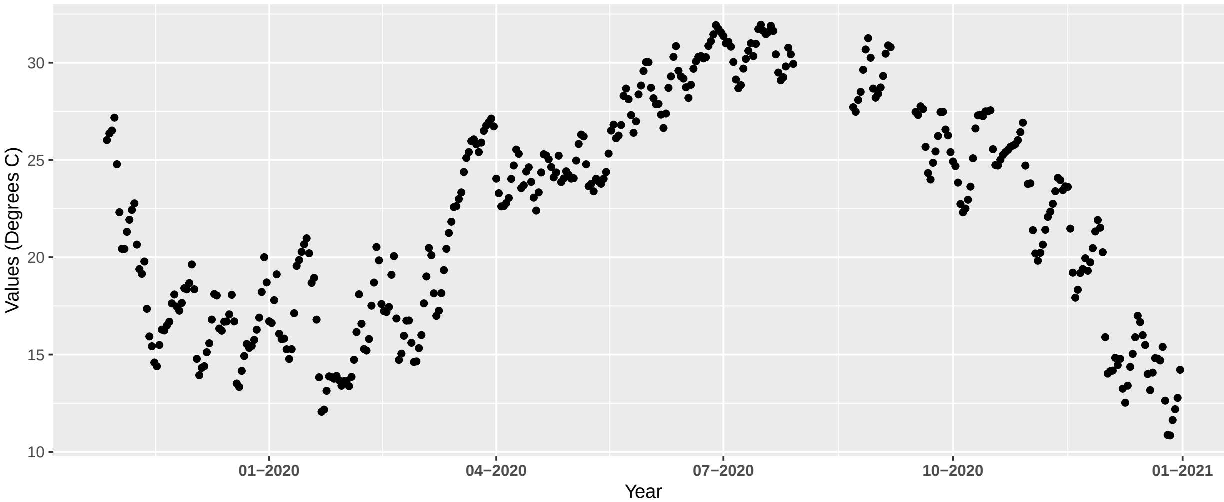
Scatter Plot of Excluded Monitoring Location Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apabpwq
(3 Unique Years)
Autoscale



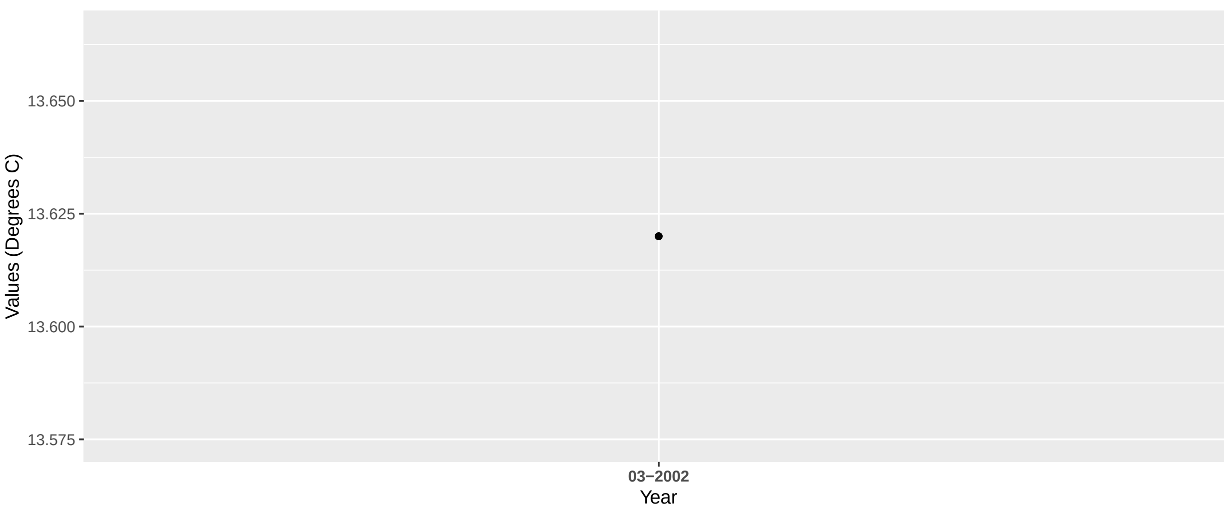

Scatter Plot of Excluded Monitoring Location Big Bend Seagrasses Aquatic Preserve
7 | National Water Information System | 02313700
(4 Unique Years)
Autoscale
Values (Degrees C)
30
25
20
15
01−2020
01−2021
01−2022
Year

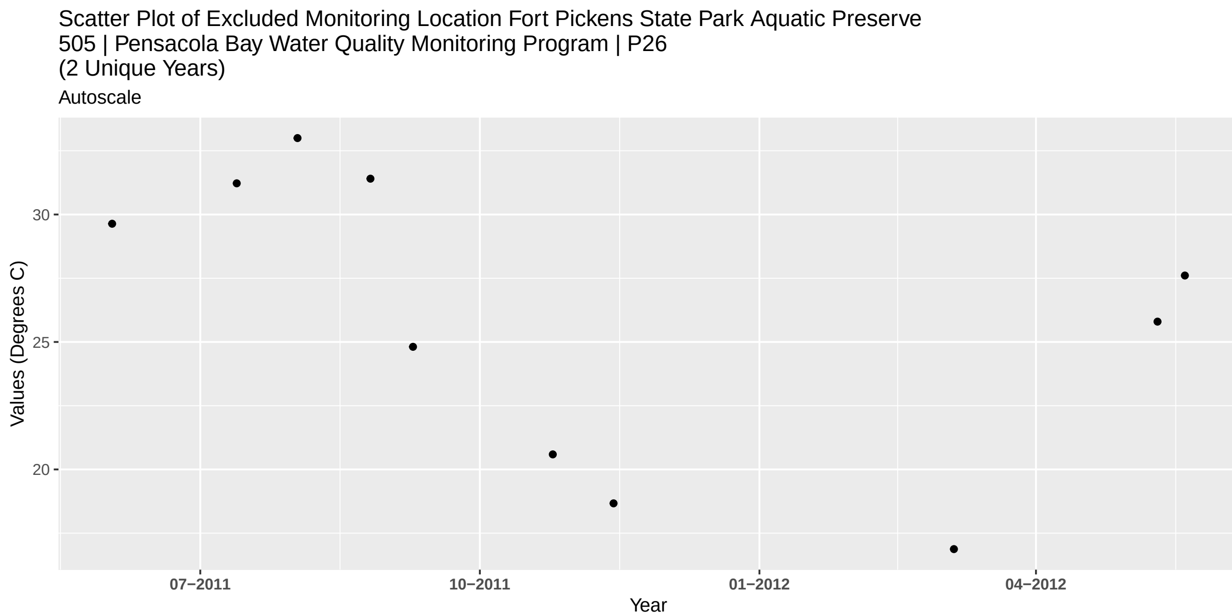

Scatter Plot of Excluded Monitoring Location Big Bend Seagrasses Aquatic Preserve
7 | National Water Information System | 291652083064100
(1 Unique Years)
Autoscale
Values (Degrees C)
20.0
17.5
15.0
01−2000
01−2000
02−2000
02−2000
03−2000
Year

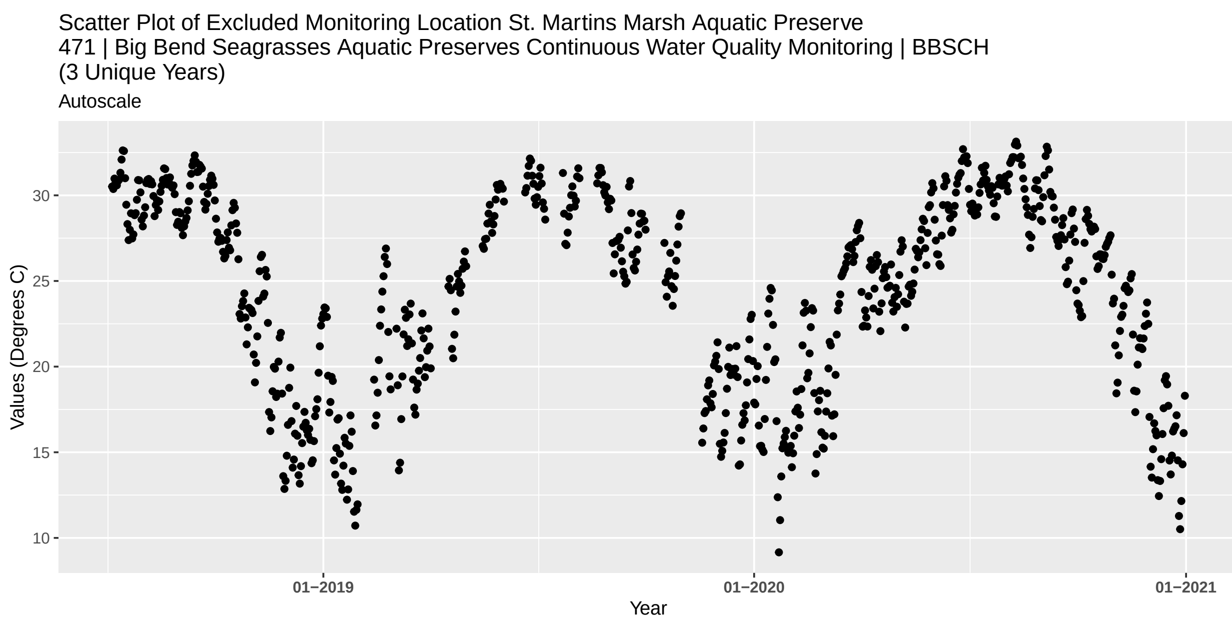
Scatter Plot of Excluded Monitoring Location Big Bend Seagrasses Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring | BBSST
(2 Unique Years)
Autoscale



Scatter Plot of Excluded Monitoring Location Fort Pickens State Park Aquatic Preserve
505 | Pensacola Bay Water Quality Monitoring Program | EX4
(1 Unique Years)
Autoscale







Appendix III: Monitoring Location Trendlines

The plots created in this section are designed to show the general trend of the data. Data is taken and grouped by `MonitoringID`. The trendlines on the plots are created using the Senn slope and intercept from the seasonal Kendall Tau analysis. The scripts that create plots follow this format

1. Use the data set that only has `Use_In_Analysis` of `TRUE` for the desired monitoring location
2. Determine the earliest and latest year of the data to create x-axis scale and intervals
3. Determine the minimum, mean, and standard deviation for the data to be used for y-axis scales

- Excludes the top 2% of values to reduce the impact of extreme outliers on the y-axis scale

4. Set what values are to be used for the x-axis, y-axis, and the variable that should determine groups for the plots
5. Set the plot type as a point plot with the size of the points
6. Add the linear trend
7. Create the title, x-axis, y-axis, and color fill labels
8. Set the y and x limits
9. Make the axis labels bold
10. Plot the arrangement as a set of panels

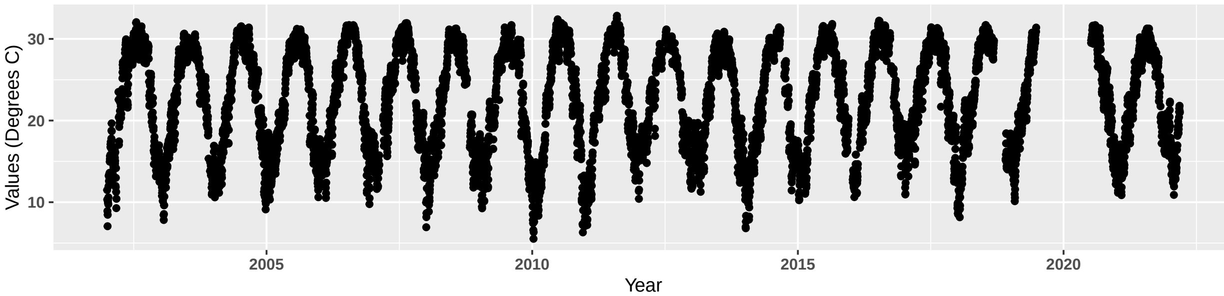
```
if(n==0){
   print("There are no monitoring locations that qualify.")
} else {
   for (i in 1:n) {
      year_lower <- min(data$Year[data$Use_In_Analysis == TRUE &
                                  data$MonitoringID == Mon_IDs[i]])
      year_upper <- max(data$Year[data$Use_In_Analysis == TRUE &
                                  data$MonitoringID == Mon_IDs[i]])
      min_RV <- min(data$ResultValue[data$Use_In_Analysis == TRUE &
                                     data$MonitoringID == Mon_IDs[i]])
      mn_RV <- mean(data$ResultValue[data$Use_In_Analysis == TRUE &
                                     data$MonitoringID == Mon_IDs[i] &
                                     data$ResultValue <
                                     quantile(data$ResultValue, 0.98)])
      sd_RV <- sd(data$ResultValue[data$Use_In_Analysis == TRUE &
                                   data$MonitoringID == Mon_IDs[i] &
                                   data$ResultValue <
                                   quantile(data$ResultValue, 0.98)])
      x_scale <- ifelse(year_upper - year_lower > 30, 10, 5)
      y_scale <- mn_RV + 4 * sd_RV

      tau <- KT.Stats$tau[KT.Stats$MonitoringID == Mon_IDs[i]]
      s_slope <- KT.Stats$SennSlope[KT.Stats$MonitoringID == Mon_IDs[i]]
      s_int <- KT.Stats$SennIntercept[KT.Stats$MonitoringID == Mon_IDs[i]]
      trend <- KT.Stats$Trend[KT.Stats$MonitoringID == Mon_IDs[i]]
      p <- KT.Stats$p[KT.Stats$MonitoringID == Mon_IDs[i]]

      model <- lm(ResultValue ~ DecDate,
                  data = data[data$Use_In_Analysis == TRUE &
                              data$MonitoringID == Mon_IDs[i]])
      m_int <- coef(model)[[1]]
      m_slope <- coef(model)[[2]]
      MA_name <- KT.Stats$ManagedAreaName[KT.Stats$MonitoringID == Mon_IDs[i]]
      Mon_name <- paste(KT.Stats$ProgramID[KT.Stats$MonitoringID == Mon_IDs[i]],
         KT.Stats$ProgramName[KT.Stats$MonitoringID == Mon_IDs[i]],
         KT.Stats$ProgramLocationID[KT.Stats$MonitoringID == Mon_IDs[i]],
         sep = " | ")

      p1 <- ggplot(data = data[data$Use_In_Analysis == TRUE &
                               data$MonitoringID == Mon_IDs[i], ],
                   aes(x = DecDate, y = ResultValue)) +
         geom_point(size = 1.5) +
         geom_abline(aes(slope=s_slope, intercept=s_int),
```

```
                    color="red", size=1.5) +
         labs(subtitle = "Autoscale",
              x = "Year", y = paste0("Values (", unit, ")")) +
         theme(axis.text.x = element_text(face = "bold"),
               axis.text.y = element_text(face="bold"))

      p2 <- ggplot(data = data[data$Use_In_Analysis == TRUE &
                                  data$MonitoringID == Mon_IDs[i], ],
                   aes(x = DecDate, y = ResultValue)) +
         geom_point(size = 1.5) +
         geom_abline(aes(slope=s_slope, intercept=s_int),
                    color="red", size=1.5) +
         ylim(min_RV-0.1*y_scale, y_scale) +
         labs(subtitle = "Scaled to 4x Standard Deviation",
              x = "Year", y = paste0("Values (", unit, ")")) +
         theme(axis.text.x = element_text(face = "bold"),
               axis.text.y = element_text(face="bold"))
      KTset <- ggarrange(p1, p2, ncol = 1, heights = c(1, 1))

      p0 <- ggplot() + labs(title = paste0("Data Points with Trendlines for ",
                                           MA_name, "\n", Mon_name),
                           subtitle =paste0("Senn Slope = ", s_slope,
                                             ",     Senn Intercept = ", s_int,
                                             "\nTrend = ", trend,
                                             ",     tau = ", tau,
                                             ",     p = ", p,
                                             "\nLinear Trendline: ",
                                             "y = ", m_slope,"x + ",m_int)) +
         theme_bw() + theme(plot.title = element_text(face="bold"),
                           panel.border = element_blank(),
                           panel.grid.major = element_blank(),
                           panel.grid.minor = element_blank(),
                           axis.line = element_blank())

      print(ggarrange(p0, KTset, ncol = 1, heights = c(0.20, 1)))
   }
}
```

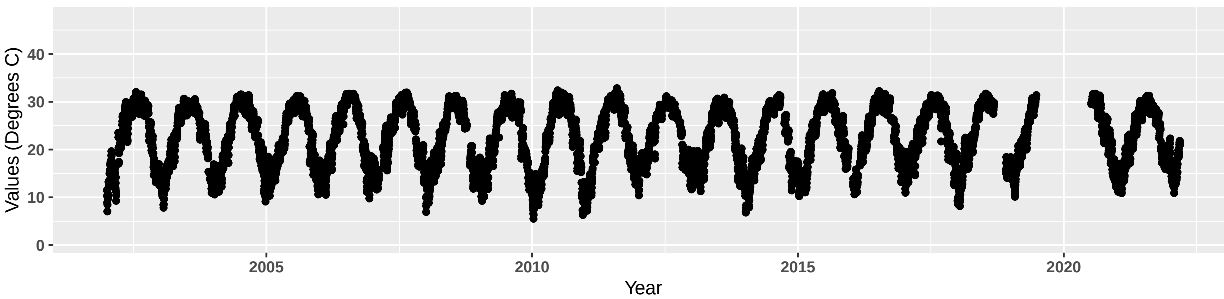
Data Points with Trendlines for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apadbwq

Senn Slope = 0.025416666666667, Senn Intercept = 8.65750000000006
Trend = 1, tau = 0.0499, p = 0
Linear Trendline: y = 0.0179683531660155x + −13.5867250925648

Autoscale



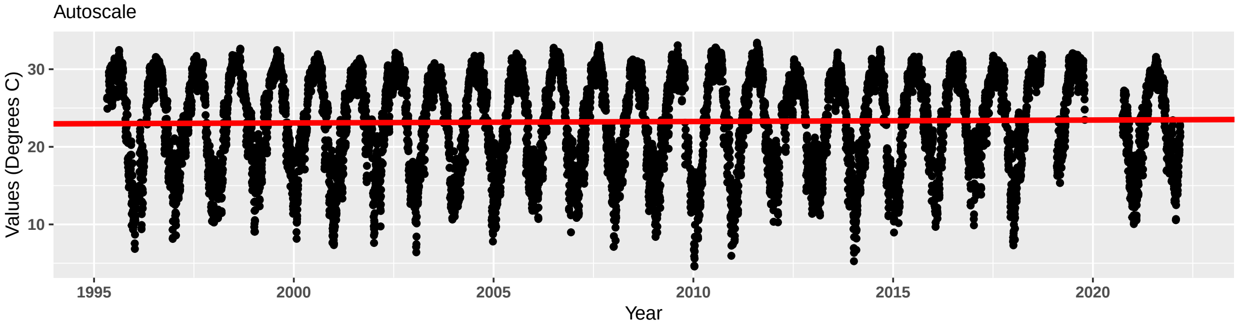
Scaled to 4x Standard Deviation



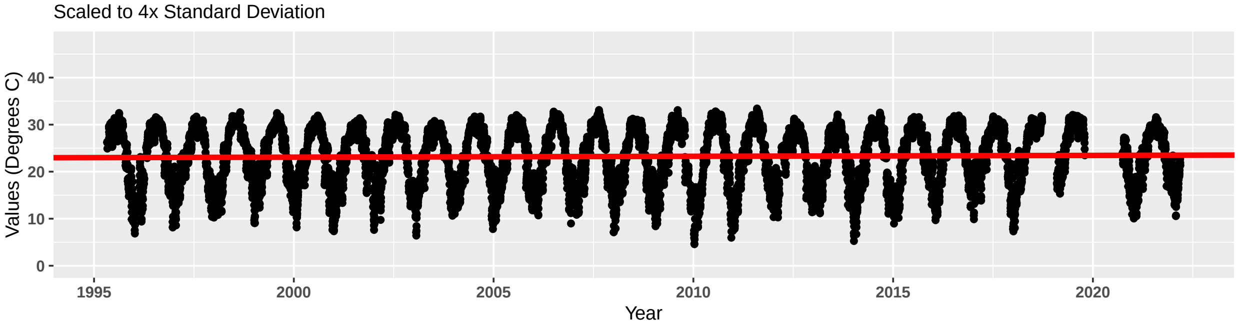
Data Points with Trendlines for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaebwq

Senn Slope = 0.0186848958333334, Senn Intercept = −14.2931249999998
Trend = 1, tau = 0.0453, p = 0
Linear Trendline: y = 0.00674647537173045x + 9.58344923925973

Autoscale



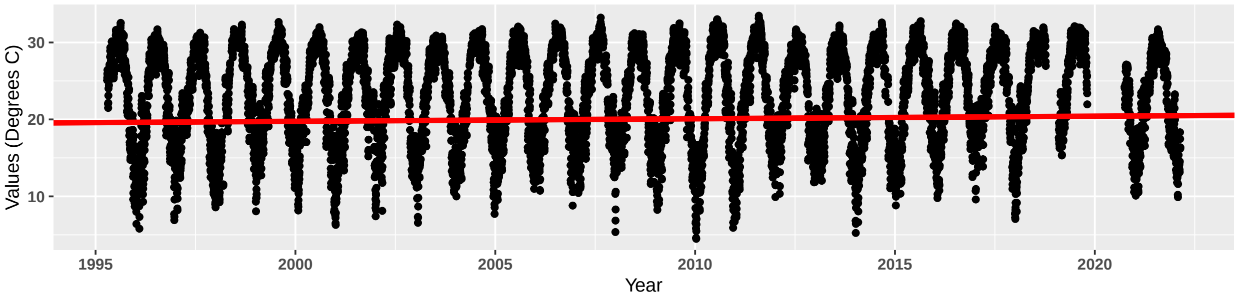
Scaled to 4x Standard Deviation



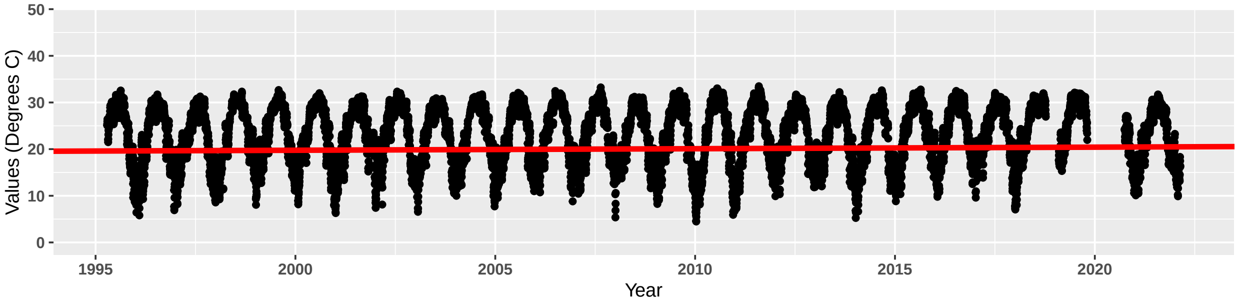
Data Points with Trendlines for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaeswq

Senn Slope = 0.0331845238095238,   Senn Intercept = −46.6137862723215
Trend = 1,   tau = 0.0799,   p = 0
Linear Trendline: y = 0.0432146770704752x + −63.697272622716

Autoscale



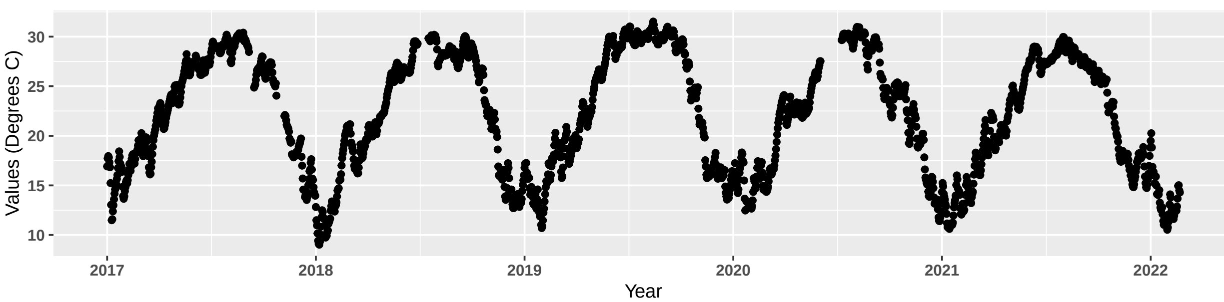
Scaled to 4x Standard Deviation



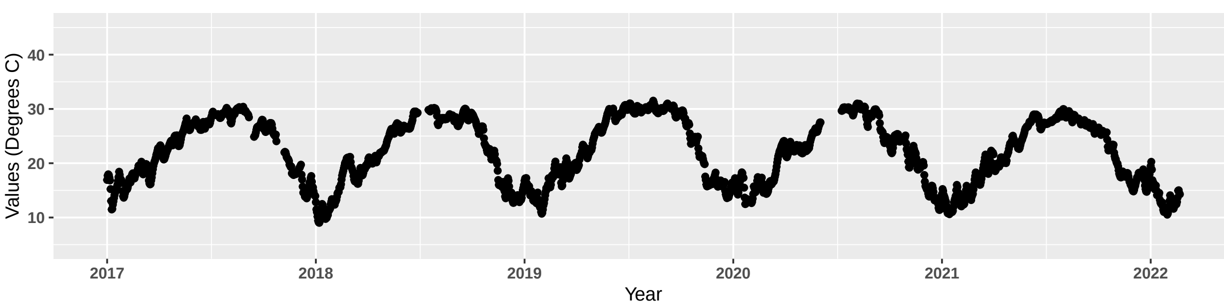
Data Points with Trendlines for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apalmwq

Senn Slope = −0.202387152777778,   Senn Intercept = 356.921679687501
Trend = −1,   tau = −0.1009,   p = 0
Linear Trendline: y = −0.23316250676901x + 492.910208978837

Autoscale



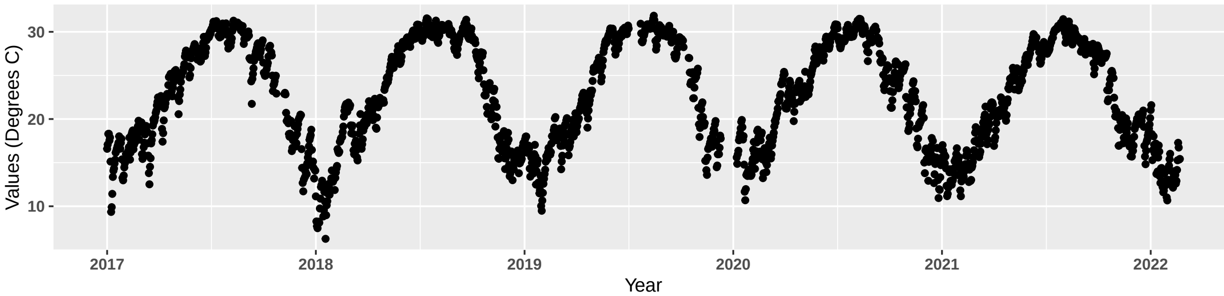
Scaled to 4x Standard Deviation



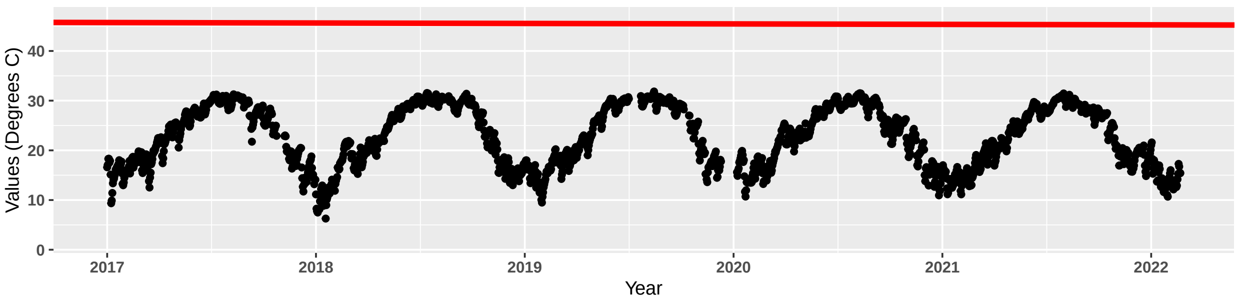
Data Points with Trendlines for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apapcwq

Senn Slope = −0.0924045138888886, Senn Intercept = 232.101692708334
Trend = −1, tau = −0.0488, p = 0.0014
Linear Trendline: y = −0.056062712888517x + 135.918125570908

Autoscale



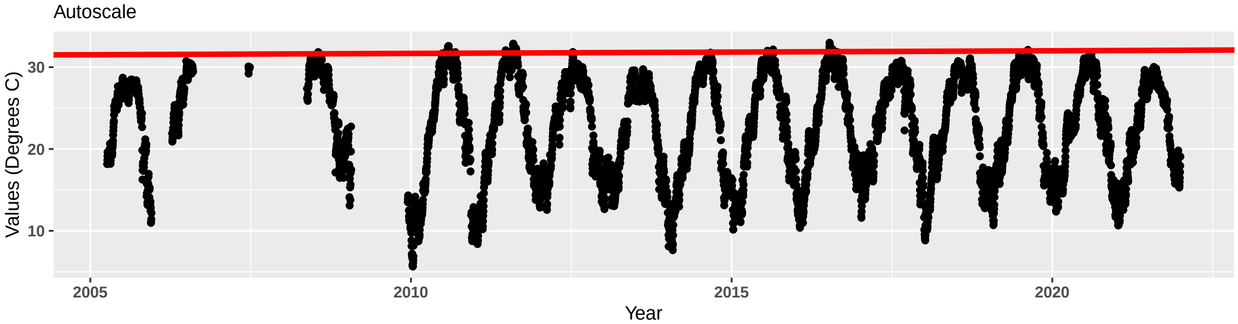
Scaled to 4x Standard Deviation



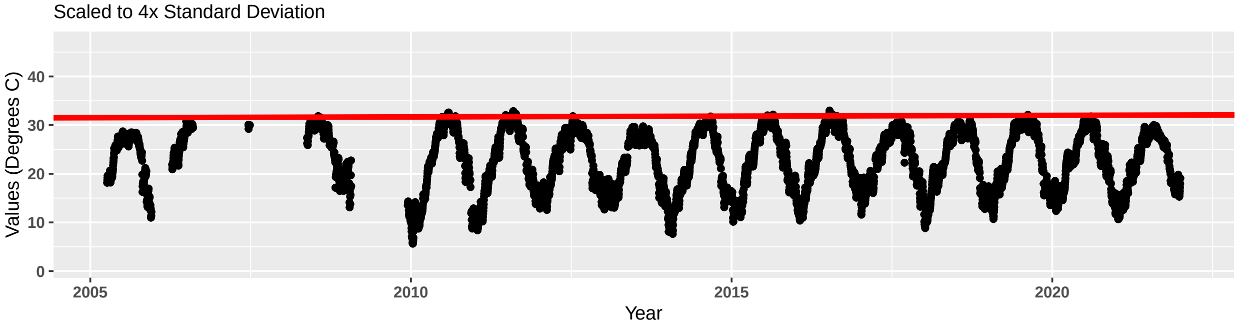
Data Points with Trendlines for Apalachicola National Estuarine Research Reserve
5 | National Data Buoy Center | APCF1

Senn Slope = 0.0321129943502825, Senn Intercept = −32.8663397579983
Trend = 1, tau = 0.0502, p = 0
Linear Trendline: y = −0.0389950067023035x + 101.361518676645

Autoscale



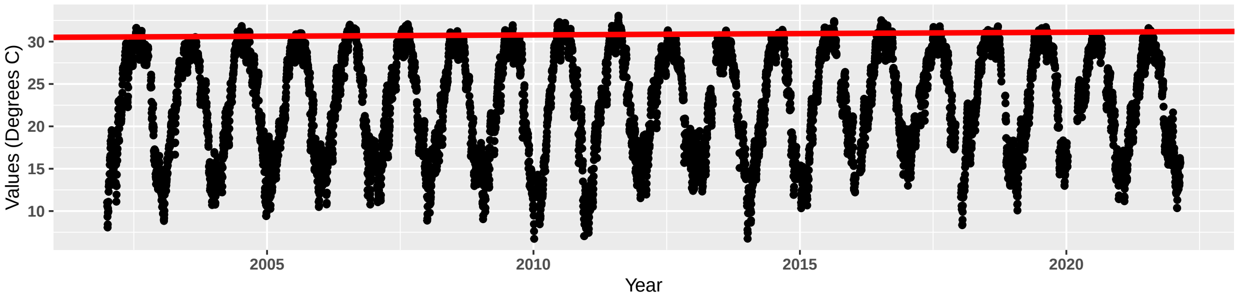
Scaled to 4x Standard Deviation



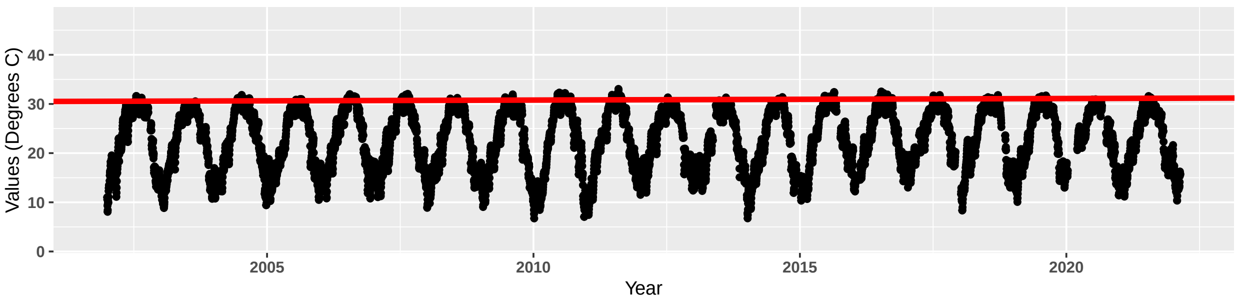
Data Points with Trendlines for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apacpwq

Senn Slope = 0.03125,   Senn Intercept = −32.0222489316239
Trend = 1,   tau = 0.0666,   p = 0
Linear Trendline: y = 0.0608493731078776x + −99.6331207967384

Autoscale



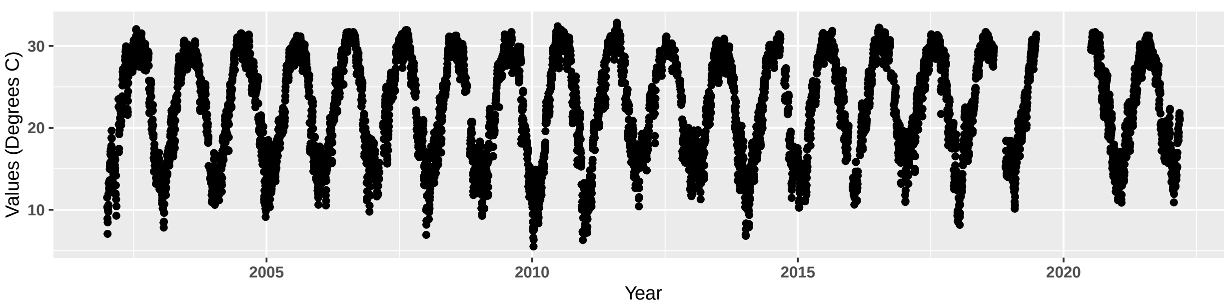
Scaled to 4x Standard Deviation



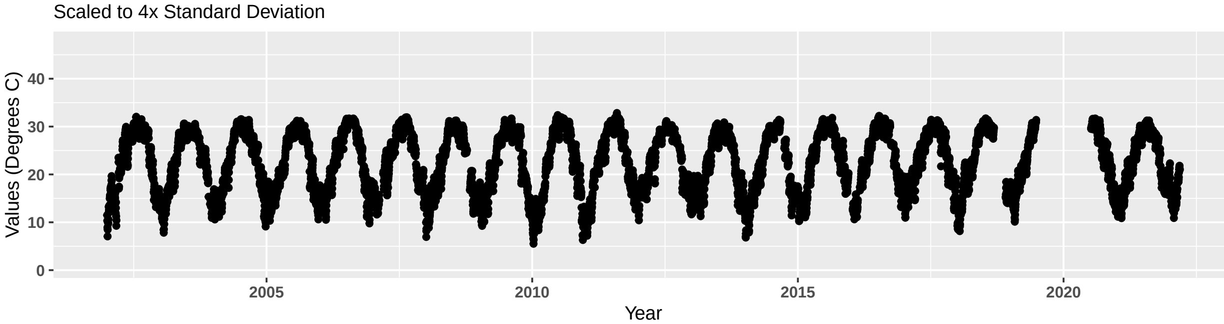
Data Points with Trendlines for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apadbwq

Senn Slope = 0.025416666666667,   Senn Intercept = 8.65750000000006
Trend = 1,   tau = 0.0499,   p = 0
Linear Trendline: y = 0.0179683531660155x + −13.5867250925648

Autoscale



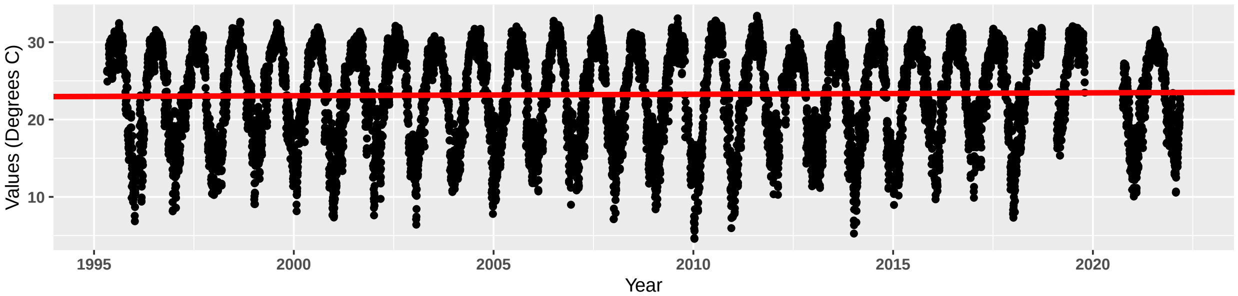
Scaled to 4x Standard Deviation



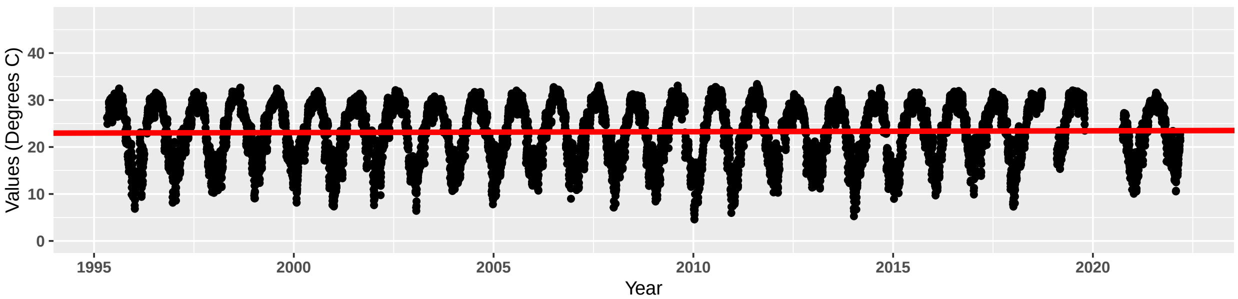
Data Points with Trendlines for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaebwq

Senn Slope = 0.0186848958333334, Senn Intercept = −14.2931249999998
Trend = 1, tau = 0.0453, p = 0
Linear Trendline: y = 0.00674647537173045x + 9.58344923925973

Autoscale



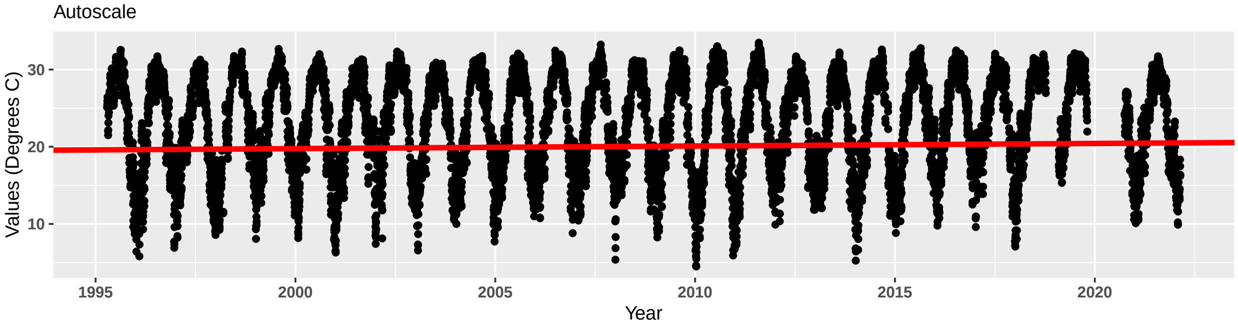
Scaled to 4x Standard Deviation



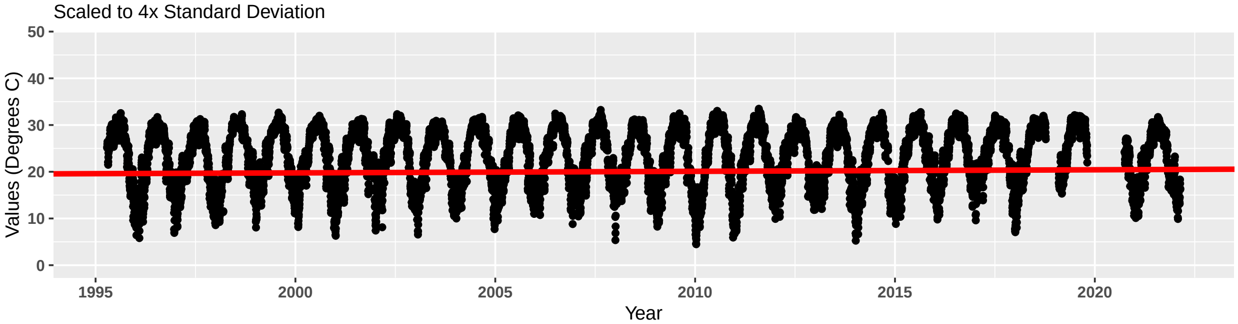
Data Points with Trendlines for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaeswq

Senn Slope = 0.0331845238095238, Senn Intercept = −46.6137862723215
Trend = 1, tau = 0.0799, p = 0
Linear Trendline: y = 0.0432146770704752x + −63.697272622716

Autoscale

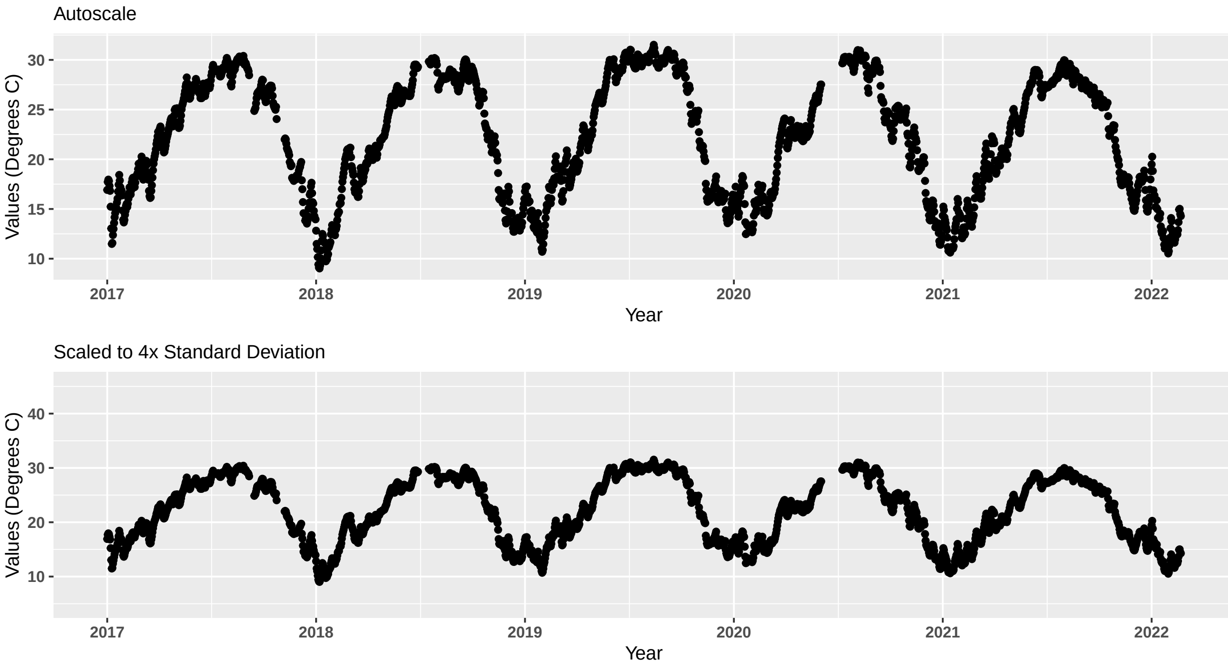


Scaled to 4x Standard Deviation



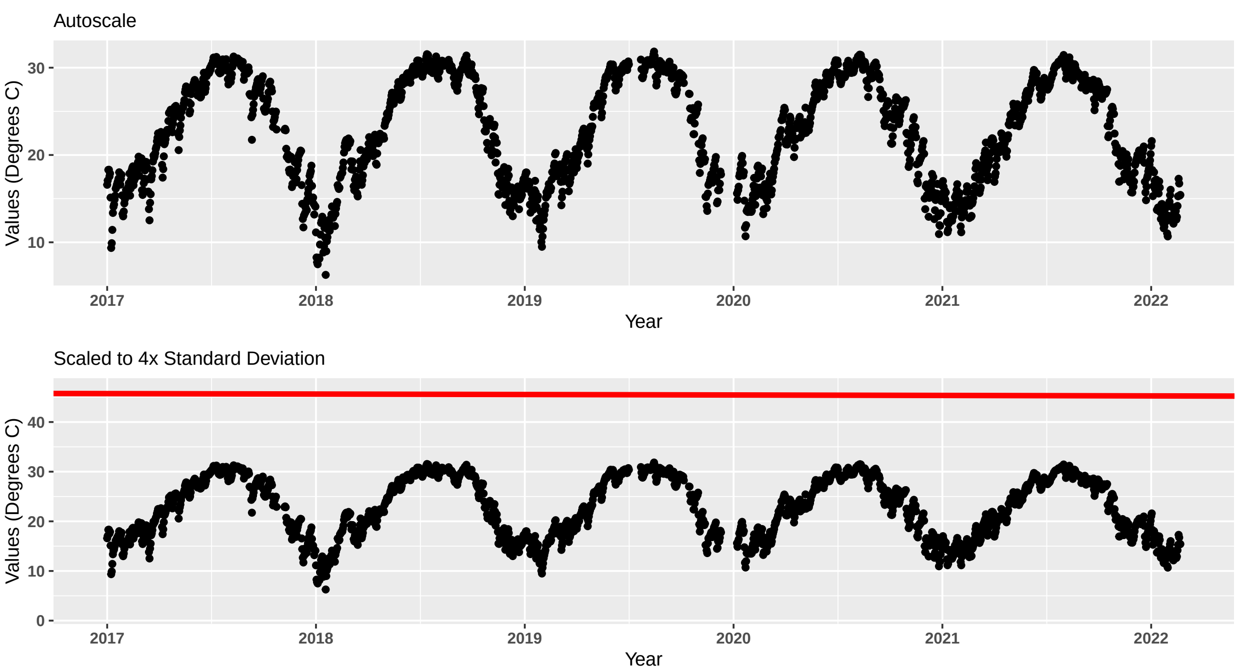
Data Points with Trendlines for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apalmwq

Senn Slope = −0.202387152777778,   Senn Intercept = 356.921679687501
Trend = −1,   tau = −0.1009,   p = 0
Linear Trendline: y = −0.23316250676901x + 492.910208978837



Data Points with Trendlines for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apapcwq

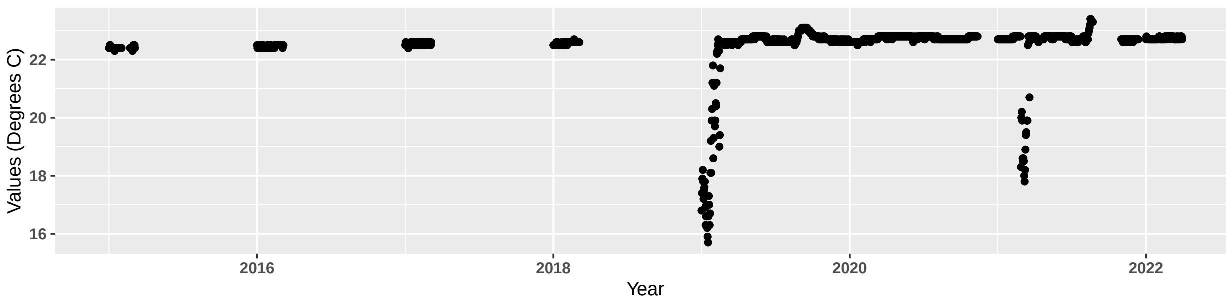
Senn Slope = −0.0924045138888886,   Senn Intercept = 232.101692708334
Trend = −1,   tau = −0.0488,   p = 0.0014
Linear Trendline: y = −0.056062712888517x + 135.918125570908



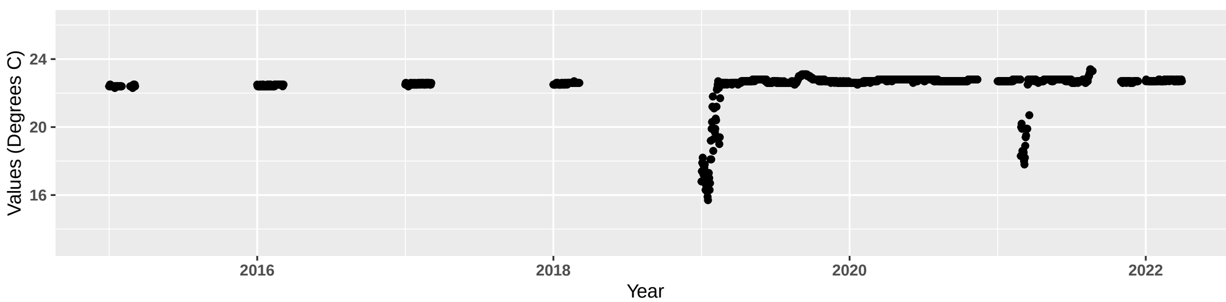
Data Points with Trendlines for Big Bend Seagrasses Aquatic Preserve 7 | National Water Information System | 02323566

Senn Slope = 0.0499999999999989, Senn Intercept = −17.6999999999999
Trend = 1, tau = 0.2685, p = 0
Linear Trendline: y = 0.0637767707911917x + −106.321648057791

Autoscale



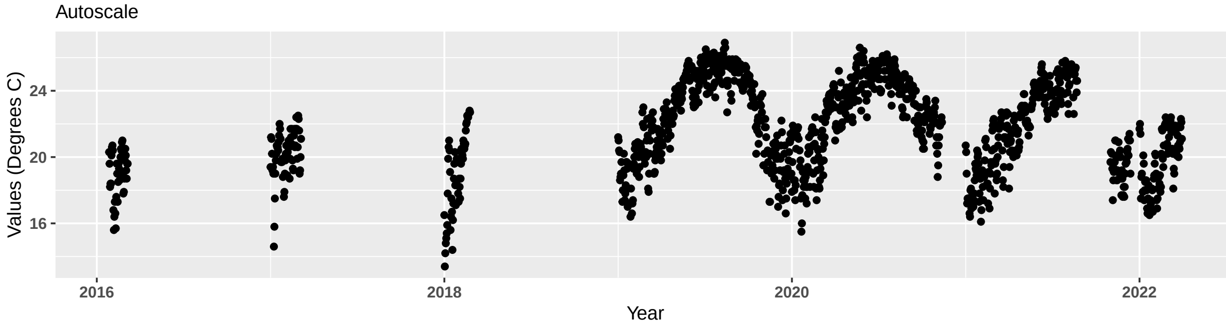
Scaled to 4x Standard Deviation



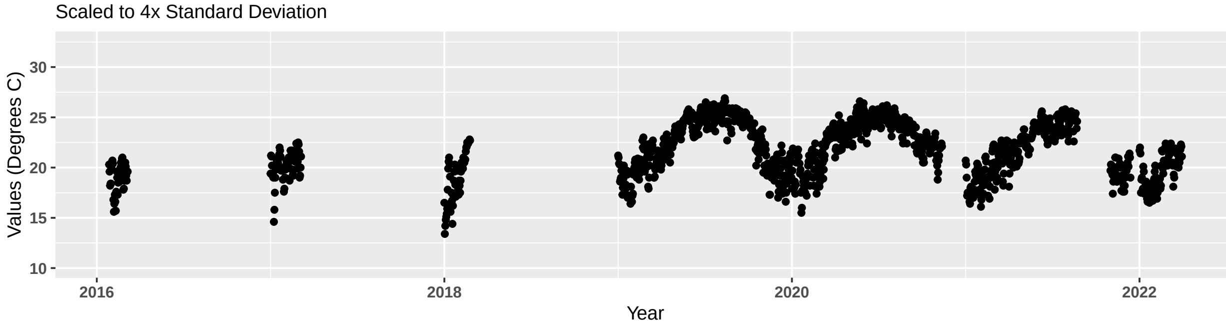
Data Points with Trendlines for Big Bend Seagrasses Aquatic Preserve 7 | National Water Information System | 02326526

Senn Slope = −0.199999999999999, Senn Intercept = 829.049999999999
Trend = −1, tau = −0.143, p = 0
Linear Trendline: y = 0.297425612717862x + −579.096916184273

Autoscale

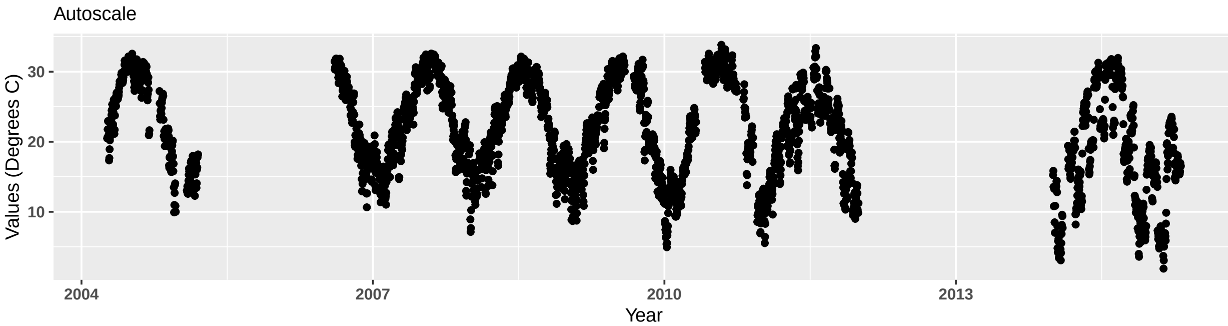


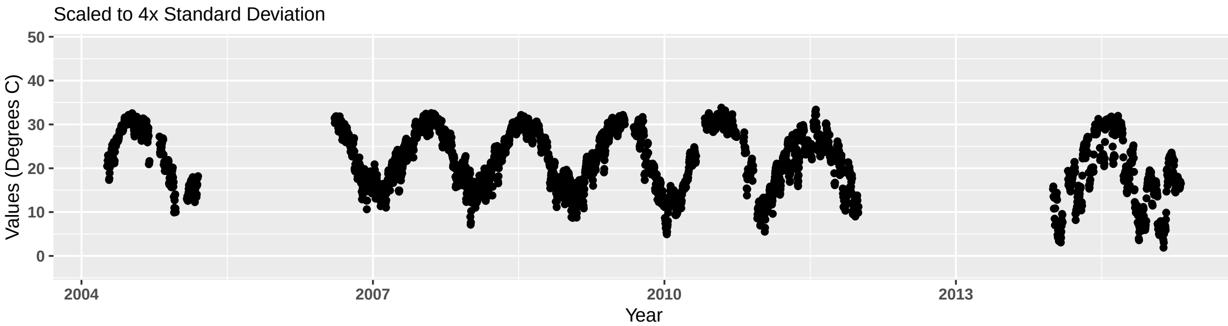
Scaled to 4x Standard Deviation



Data Points with Trendlines for Big Bend Seagrasses Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring | BBSSK

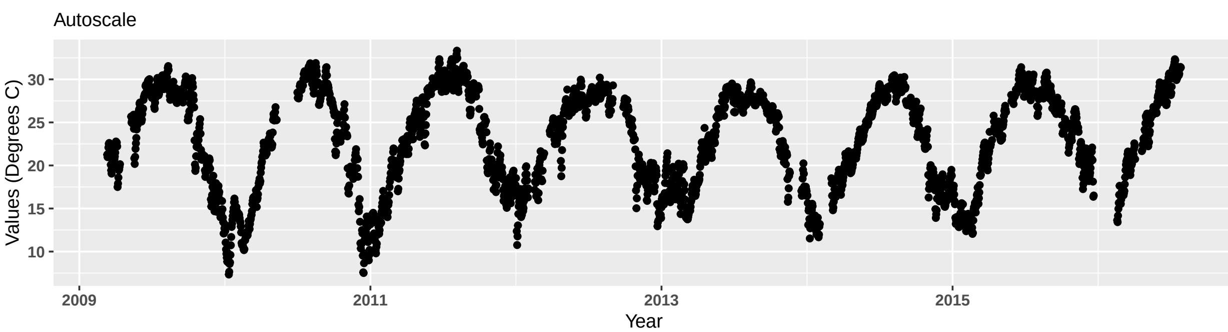
Senn Slope = −0.361458333333334, Senn Intercept = 804.180539772727
Trend = −1, tau = −0.2136, p = 0
Linear Trendline: y = −0.61851879210426x + 1264.62392556368

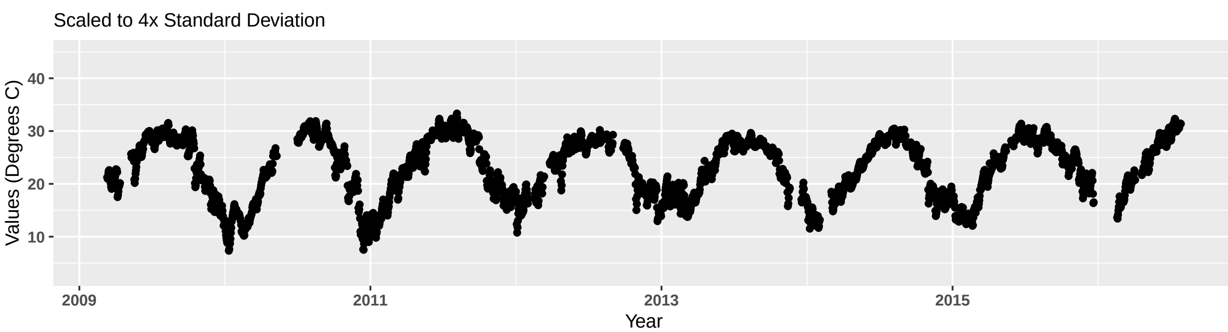




Data Points with Trendlines for Big Bend Seagrasses Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring | BBSSW

Senn Slope = 0.00625, Senn Intercept = −20.0907446946172
Trend = 0, tau = 0.0058, p = 0.7669
Linear Trendline: y = 0.167195620569622x + −313.600135589666

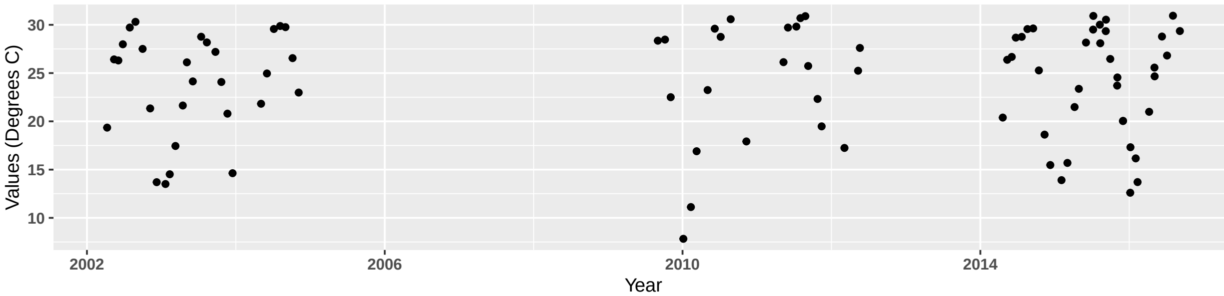




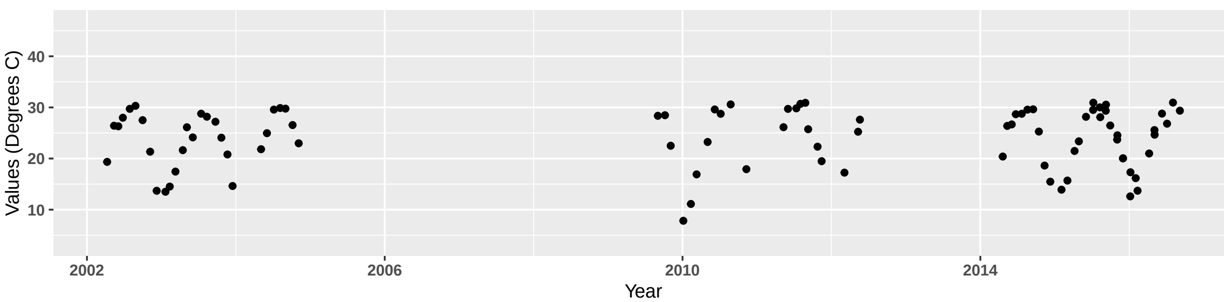
Data Points with Trendlines for Fort Pickens State Park Aquatic Preserve
505 | Pensacola Bay Water Quality Monitoring Program | P09

Senn Slope = 0.0587299999999999, Senn Intercept = −241.330818322173
Trend = 0, tau = 0.1071, p = 0.2259
Linear Trendline: y = 0.0341713918080144x + −44.6616142811487

Autoscale



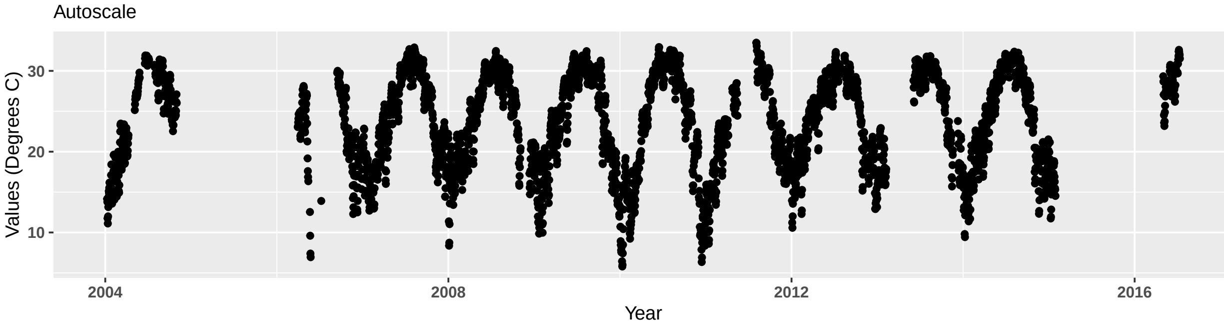
Scaled to 4x Standard Deviation



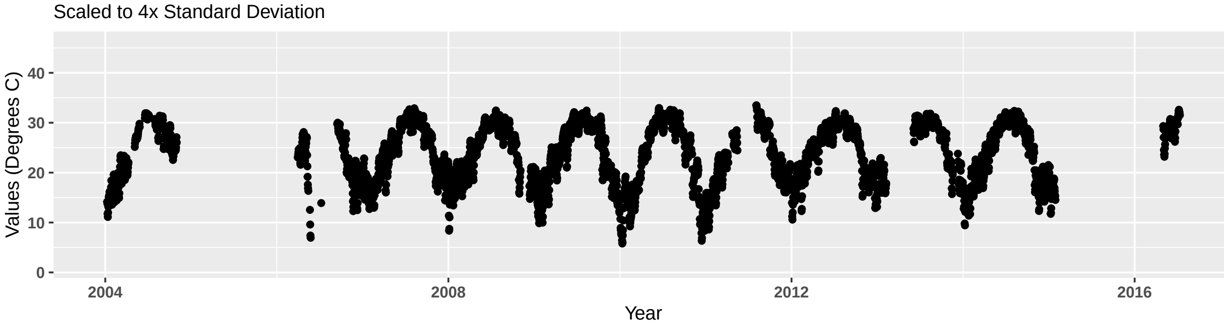
Data Points with Trendlines for Nature Coast Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring | BBSHS

Senn Slope = 0.0250000000000004, Senn Intercept = −62.4815848214288
Trend = 0, tau = 0.0215, p = 0.0521
Linear Trendline: y = 0.0999913455187762x + −177.272582637183

Autoscale



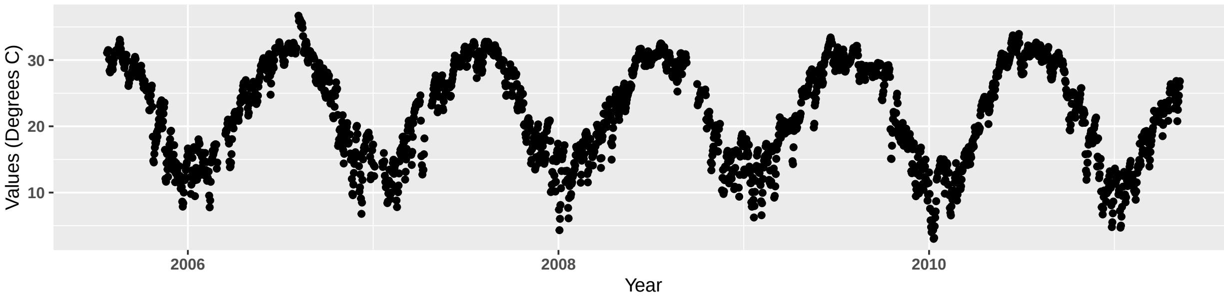
Scaled to 4x Standard Deviation



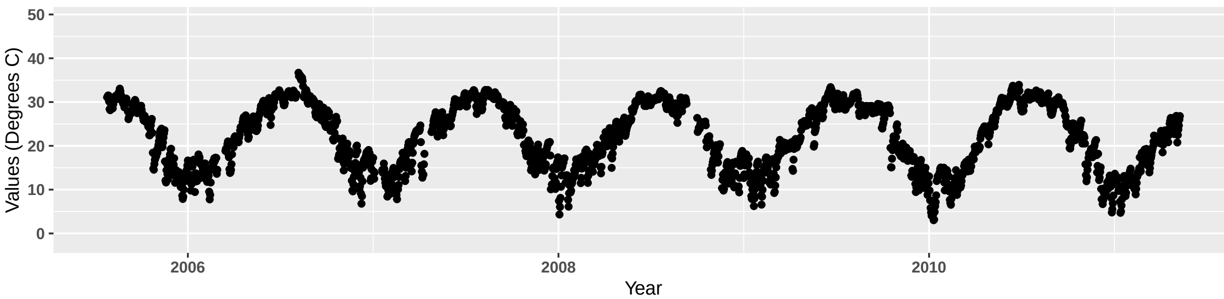
Data Points with Trendlines for St. Joseph Bay Aquatic Preserve
468 | Central Panhandle Aquatic Preserves Continuous Water Quality Monitoring | CPRH

Senn Slope = −0.129166666666666, Senn Intercept = 314.349305555556
Trend = −1, tau = −0.0544, p = 0.0001
Linear Trendline: y = −0.444919305778536x + 915.61791396811

Autoscale



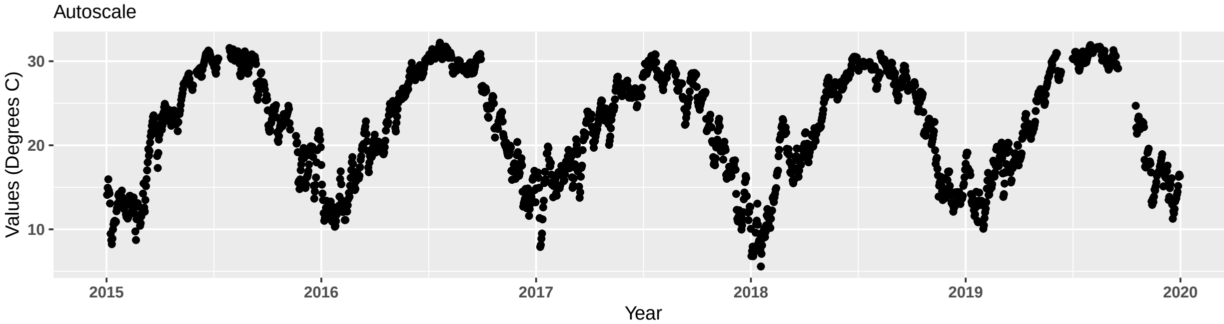
Scaled to 4x Standard Deviation



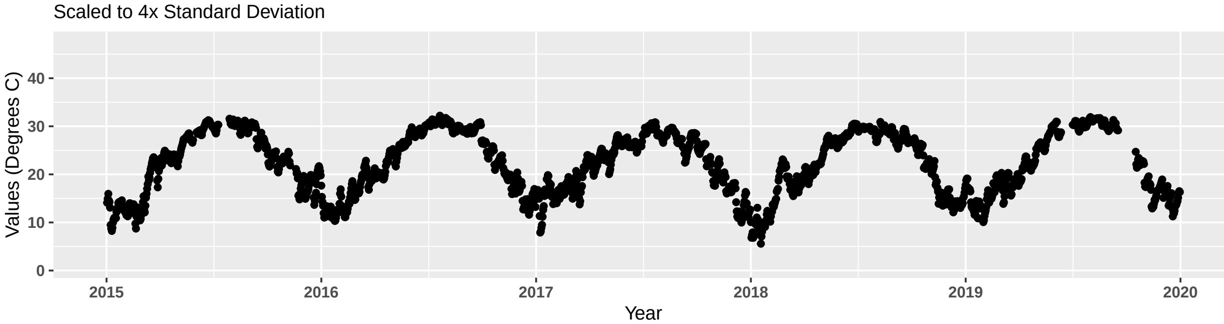
Data Points with Trendlines for Yellow River Marsh Aquatic Preserve
467 | Yellow River Marsh Aquatic Preserve Continuous Water Quality Monitoring | YRMAP1

Senn Slope = −0.111458333333332, Senn Intercept = 166.513401988636
Trend = −1, tau = −0.0467, p = 0.0033
Linear Trendline: y = 0.0946448513359908x + −168.77526923302

Autoscale

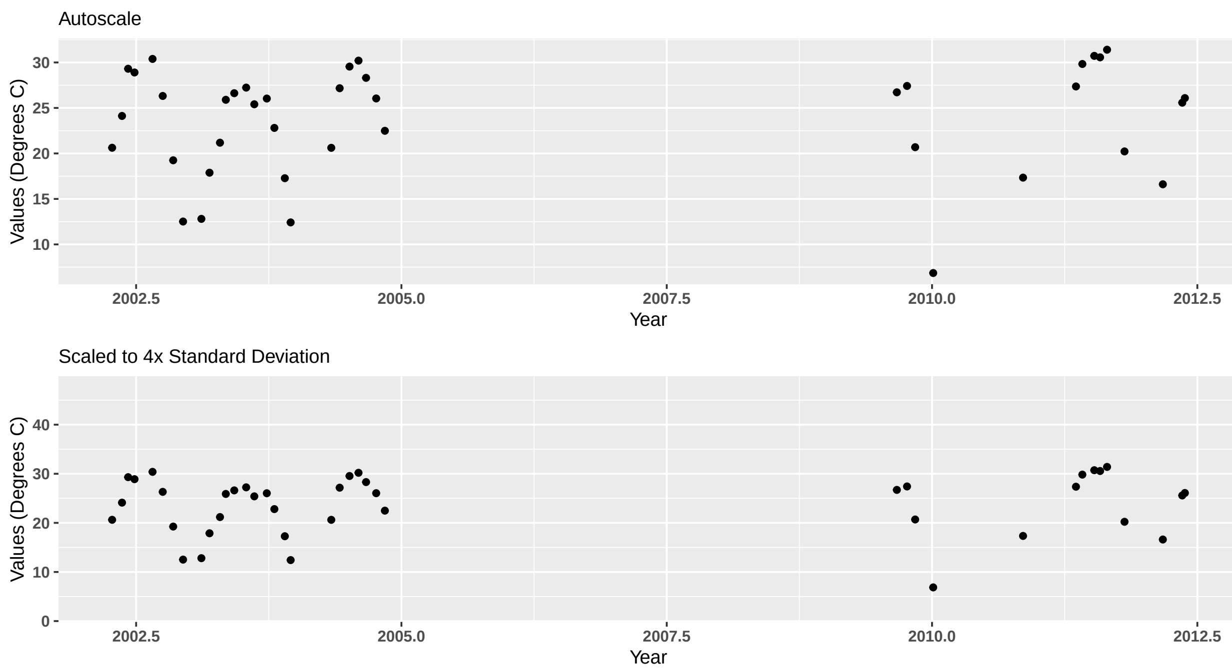


Scaled to 4x Standard Deviation



Data Points with Trendlines for Yellow River Marsh Aquatic Preserve
505 | Pensacola Bay Water Quality Monitoring Program | P11

Senn Slope = 0.103553333333333, Senn Intercept = −141.0433025
Trend = 0, tau = 0.1474, p = 0.3229
Linear Trendline: y = 0.136296290900375x + −249.719956821996



Appendix IV: Monitoring Location Summary Box Plots

Data is taken and grouped by `MonitoringID`. The scripts that create plots follow this format

1. Use the data set that only has `Use_In_Analysis` of `TRUE` for the desired monitoring location
2. Determine the earliest and latest year of the data to create x-axis scale and intervals
3. Determine the minimum, mean, and standard deviation for the data to be used for y-axis scales
    - Excludes the top 2% of values to reduce the impact of extreme outliers on the y-axis scale
4. Set what values are to be used for the x-axis, y-axis, and the variable that should determine groups for the box plots
5. Set the plot type as a box plot with the size of the outlier points
6. Create the title, x-axis, y-axis, and color fill labels
7. Set the y and x limits
8. Make the axis labels bold
9. Plot the arrangement as a set of panels

The following plots are arranged by `MonitoringID` with data grouped by `Year`, then `Year` and `Month`, then finally `Month` only. Each program area will have 3 sets of plots, each with 3 panels in them. Each panel goes as follows:

1. Y-axis autoscaled
2. Y-axis set to be mean + 4 times the standard deviation

3. Y-axis set to be mean + 4 times the standard deviation for most recent 10 years of data

```
if(n==0){
   print("There are no monitoring locations that qualify.")
} else {
   for (i in 1:n) {
      year_lower <- min(data$Year[data$Use_In_Analysis == TRUE &
                                 data$MonitoringID == Mon_IDs[i]])
      year_upper <- max(data$Year[data$Use_In_Analysis == TRUE &
                                 data$MonitoringID == Mon_IDs[i]])
      min_RV <- min(data$ResultValue[data$Use_In_Analysis == TRUE &
                                       data$MonitoringID == Mon_IDs[i]])
      mn_RV <- mean(data$ResultValue[data$Use_In_Analysis == TRUE &
                                       data$MonitoringID == Mon_IDs[i] &
                                       data$ResultValue <
                                       quantile(data$ResultValue, 0.98)])
      sd_RV <- sd(data$ResultValue[data$Use_In_Analysis == TRUE &
                                     data$MonitoringID == Mon_IDs[i] &
                                     data$ResultValue <
                                     quantile(data$ResultValue, 0.98)])
      x_scale <- ifelse(year_upper - year_lower > 30, 10, 5)
      y_scale <- mn_RV + 4 * sd_RV
      MA_name <- KT.Stats$ManagedAreaName[KT.Stats$MonitoringID == Mon_IDs[i]]
      Mon_name <- paste(KT.Stats$ProgramID[KT.Stats$MonitoringID == Mon_IDs[i]],
         KT.Stats$ProgramName[KT.Stats$MonitoringID == Mon_IDs[i]],
         KT.Stats$ProgramLocationID[KT.Stats$MonitoringID == Mon_IDs[i]],
         sep = " | ")

      ##Year plots
      p1 <- ggplot(data = data[data$Use_In_Analysis == TRUE &
                                  data$MonitoringID == Mon_IDs[i], ],
                   aes(x = Year, y = ResultValue, group = Year)) +
         geom_boxplot(outlier.size = 0.5) +
         labs(subtitle = "Autoscale",
              x = "Year", y = paste0("Values (", unit, ")")) +
         scale_x_continuous(limits = c(year_lower - 1, year_upper + 1),
                            breaks = rev(seq(year_upper,
                                             year_lower, -x_scale))) +
         theme(axis.text.x = element_text(face = "bold"),
               axis.text.y = element_text(face = "bold"))

      p2 <- ggplot(data = data[data$Use_In_Analysis == TRUE &
                                  data$MonitoringID == Mon_IDs[i], ],
                   aes(x = Year, y = ResultValue, group = Year)) +
         geom_boxplot(outlier.size = 0.5) +
         labs(subtitle = "Scaled to 4x Standard Deviation",
              x = "Year", y = paste0("Values (", unit, ")")) +
         ylim(min_RV, y_scale) +
         scale_x_continuous(limits = c(year_lower - 1, year_upper + 1),
                            breaks = rev(seq(year_upper,
                                             year_lower, -x_scale))) +
         theme(axis.text.x = element_text(face = "bold"),
               axis.text.y = element_text(face = "bold"))
```

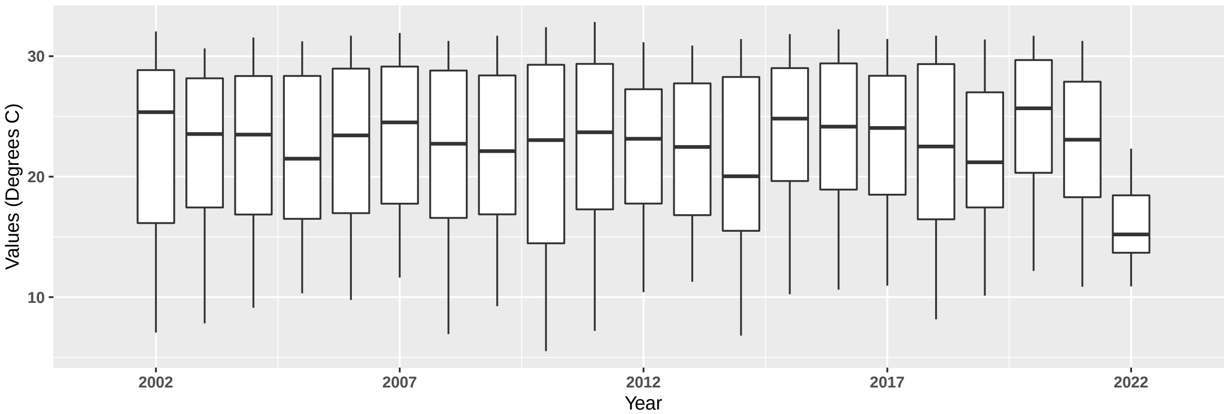
```
p3 <- ggplot(data = data[data$Use_In_Analysis == TRUE &
                            data$MonitoringID == Mon_IDs[i] &
                            data$Year>=year_upper-10, ],
             aes(x = Year, y = ResultValue, group = Year)) +
   geom_boxplot(outlier.size = 0.5) +
   labs(subtitle = "Scaled to 4x Standard Deviation, Last 10 Years",
        x = "Year", y = paste0("Values (", unit, ")")) +
   ylim(min_RV, y_scale) +
   scale_x_continuous(limits = c(year_upper - 10.5, year_upper + 1),
                      breaks = rev(seq(year_upper, year_upper - 10,-2))) +
   theme(axis.text.x = element_text(face = "bold"),
         axis.text.y = element_text(face = "bold"))

Yset <- ggarrange(p1, p2, p3, ncol = 1)

p0 <- ggplot() + labs(title = paste0("Summary Box Plots for ",
                                     MA_name, "\n", Mon_name),
                      subtitle = "By Year") +
   theme_bw() + theme(plot.title = element_text(face="bold"),
         panel.border = element_blank(),
         panel.grid.major = element_blank(),
         panel.grid.minor = element_blank(), axis.line = element_blank())


## Year & Month Plots
p4 <- ggplot(data = data[data$Use_In_Analysis == TRUE &
                            data$MonitoringID == Mon_IDs[i], ],
             aes(x = YearMonthDec, y = ResultValue,
                 group = YearMonth, color = as.factor(Month))) +
   geom_boxplot(outlier.size = 0.5) +
   labs(subtitle = "Autoscale",
        x = "Year", y = paste0("Values (", unit, ")"), color = "Month") +
   scale_x_continuous(limits = c(year_lower - 1, year_upper + 1),
                      breaks = rev(seq(year_upper,
                                       year_lower, -x_scale))) +
   theme(legend.position = "none",
         axis.text.x = element_text(face = "bold"),
         axis.text.y = element_text(face = "bold"))

p5 <- ggplot(data = data[data$Use_In_Analysis == TRUE &
                            data$MonitoringID == Mon_IDs[i], ],
             aes(x = YearMonthDec, y = ResultValue,
                 group = YearMonth, color = as.factor(Month))) +
   geom_boxplot(outlier.size = 0.5) +
   labs(subtitle = "Scaled to 4x Standard Deviation",
        x = "Year", y = paste0("Values (", unit, ")"), color = "Month") +
   ylim(min_RV, y_scale) +
   scale_x_continuous(limits = c(year_lower - 1, year_upper + 1),
                      breaks = rev(seq(year_upper,
                                       year_lower, -x_scale))) +
   theme(legend.position = "top", legend.box = "horizontal",
         axis.text.x = element_text(face = "bold"),
         axis.text.y = element_text(face = "bold")) +
```

```
    guides(color = guide_legend(nrow = 1))

  p6 <- ggplot(data = data[data$Use_In_Analysis == TRUE &
                             data$MonitoringID == Mon_IDs[i], ],
               aes(x = YearMonthDec, y = ResultValue,
                   group = YearMonth, color = as.factor(Month)
               )) +
    geom_boxplot(outlier.size = 0.5) +
    labs(subtitle = "Scaled to 4x Standard Deviation, Last 10 Years",
         x = "Year", y = paste0("Values (", unit, ")"), color = "Month") +
    ylim(min_RV, y_scale) +
    scale_x_continuous(limits = c(year_upper - 10.5, year_upper + 1),
                       breaks = rev(seq(year_upper, year_upper - 10,-2))) +
    theme(legend.position = "none",
          axis.text.x = element_text(face = "bold"),
          axis.text.y = element_text(face = "bold"))

  leg1 <- get_legend(p5)
  YMset <- ggarrange(leg1, p4, p5 + theme(legend.position = "none"), p6,
                     ncol = 1, heights = c(0.1, 1, 1, 1))

  p00 <- ggplot() + labs(title = paste0("Summary Box Plots for ",
                                       MA_name, "\n", Mon_name),
                       subtitle = "By Year & Month") + theme_bw() +
    theme(plot.title = element_text(face="bold"),
          panel.border = element_blank(),
          panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(), axis.line = element_blank())

  ## Month Plots
  p7 <- ggplot(data = data[data$Use_In_Analysis == TRUE &
                             data$MonitoringID == Mon_IDs[i], ],
               aes(x = Month, y = ResultValue,
                   group = Month, fill = as.factor(Month))) +
    geom_boxplot(outlier.size = 0.5) +
    labs(subtitle = "Autoscale",
         x = "Month", y = paste0("Values (", unit, ")"), fill = "Month") +
    scale_x_continuous(limits = c(0, 13), breaks = seq(3, 12, 3)) +
    theme(legend.position = "none",
          axis.text.x = element_text(face = "bold"),
          axis.text.y = element_text(face = "bold"))

  p8 <- ggplot(data = data[data$Use_In_Analysis == TRUE &
                             data$MonitoringID == Mon_IDs[i], ],
               aes(x = Month, y = ResultValue,
                   group = Month, fill = as.factor(Month))) +
    geom_boxplot(outlier.size = 0.5) +
    labs(subtitle = "Scaled to 4x Standard Deviation",
         x = "Month", y = paste0("Values (", unit, ")"), fill = "Month") +
    ylim(min_RV, y_scale) +
    scale_x_continuous(limits = c(0, 13), breaks = seq(3, 12, 3)) +
    theme(legend.position = "top", legend.box = "horizontal",
          axis.text.x = element_text(face = "bold"),
```

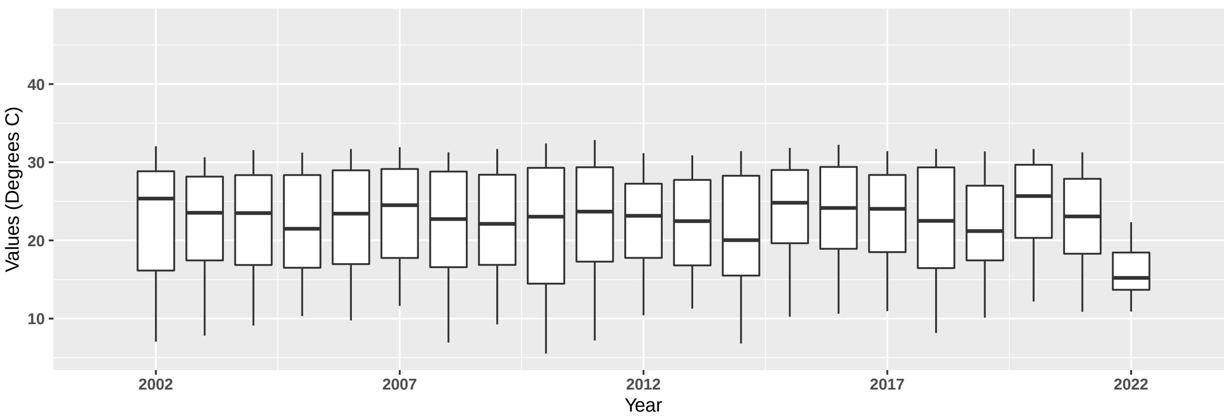
```
            axis.text.y = element_text(face = "bold")) +
      guides(fill = guide_legend(nrow = 1))

    p9 <- ggplot(data = data[data$Use_In_Analysis == TRUE &
                              data$MonitoringID == Mon_IDs[i] &
                              data$Year >= year_upper - 10, ],
                 aes(x = Month, y = ResultValue,
                     group = Month, fill = as.factor(Month))) +
      geom_boxplot(outlier.size = 0.5) +
      labs(subtitle = "Scaled to 4x Standard Deviation, Last 10 Years",
           x = "Month", y = paste0("Values (", unit, ")"), fill = "Month") +
      ylim(min_RV, y_scale) +
      scale_x_continuous(limits = c(0, 13), breaks = seq(3, 12, 3)) +
      theme(legend.position = "none",
            axis.text.x = element_text(face = "bold"),
            axis.text.y = element_text(face = "bold"))

    leg2 <- get_legend(p8)
    Mset <- ggarrange(leg2, p7, p8 + theme(legend.position = "none"), p9,
                      ncol = 1, heights = c(0.1, 1, 1, 1))

    p000 <- ggplot() + labs(title = paste0("Summary Box Plots for ",
                                            MA_name, "\n", Mon_name),
                            subtitle = "By Month") + theme_bw() +
      theme(plot.title = element_text(face="bold"),
            panel.border = element_blank(),
            panel.grid.major = element_blank(),
            panel.grid.minor = element_blank(), axis.line = element_blank())

    print(ggarrange(p0, Yset, ncol = 1, heights = c(0.1, 1)))
    print(ggarrange(p00, YMset, ncol = 1, heights = c(0.1, 1)))
    print(ggarrange(p000, Mset, ncol = 1, heights = c(0.1, 1)))
  }
}
```

Summary Box Plots for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apadbwq
By Year

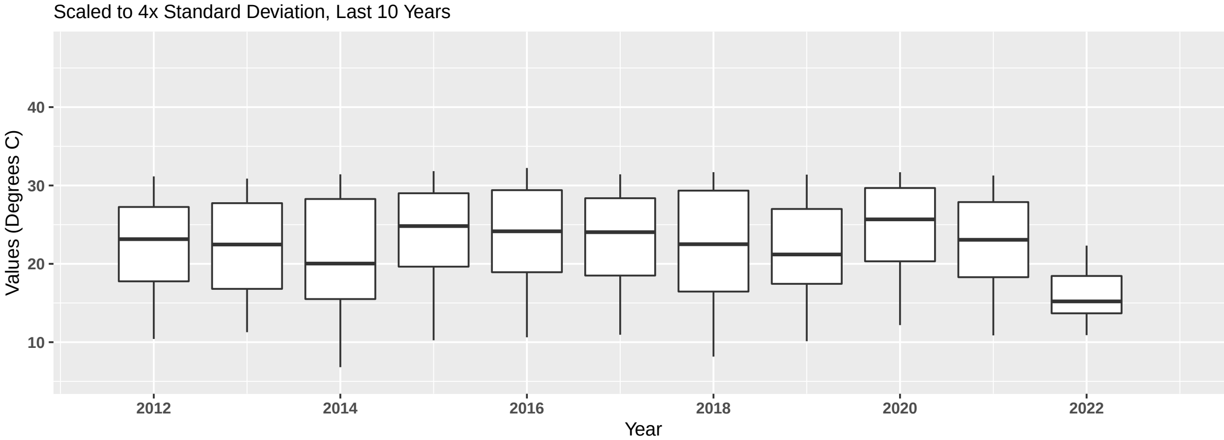
Autoscale



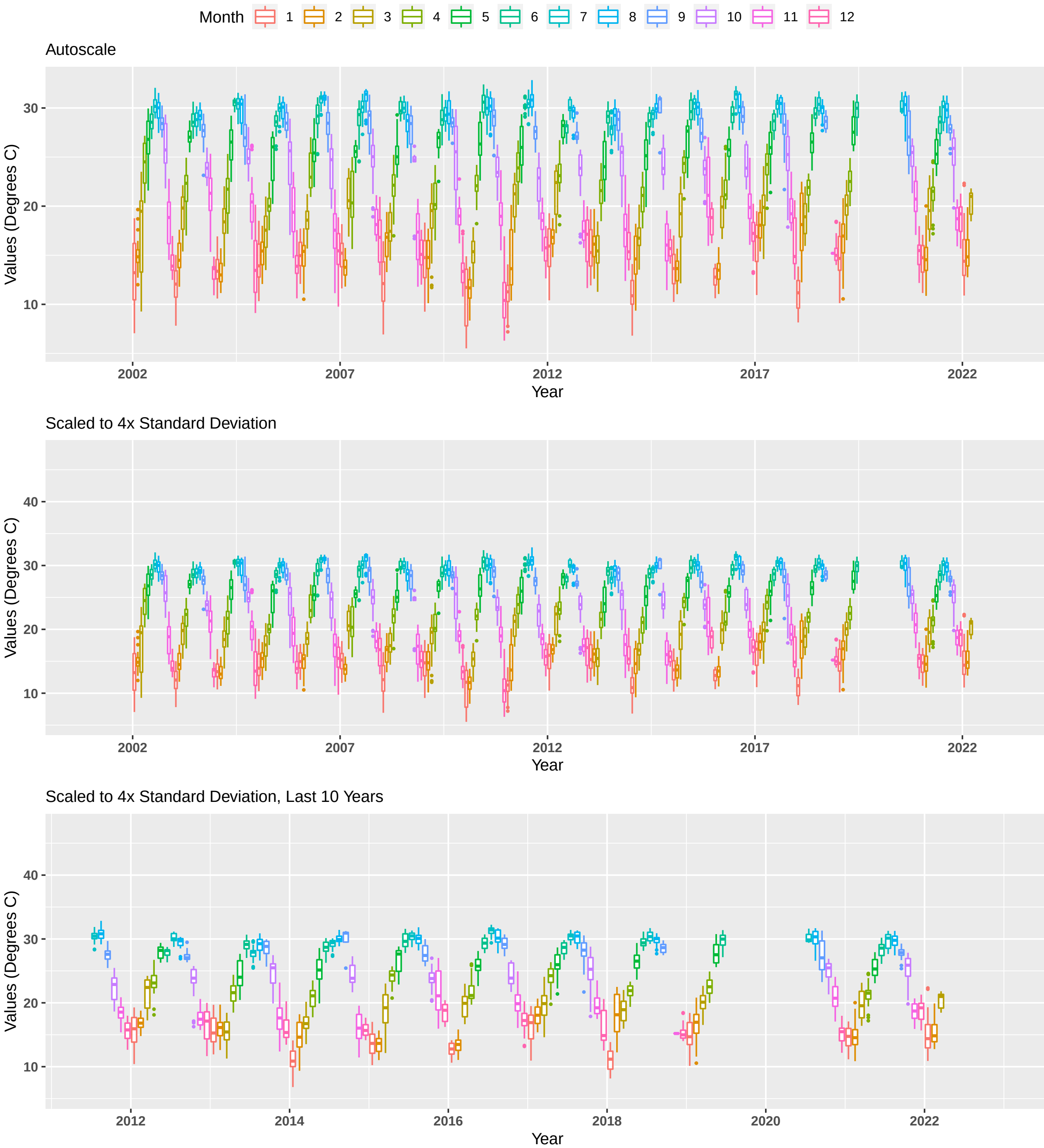
Scaled to 4x Standard Deviation



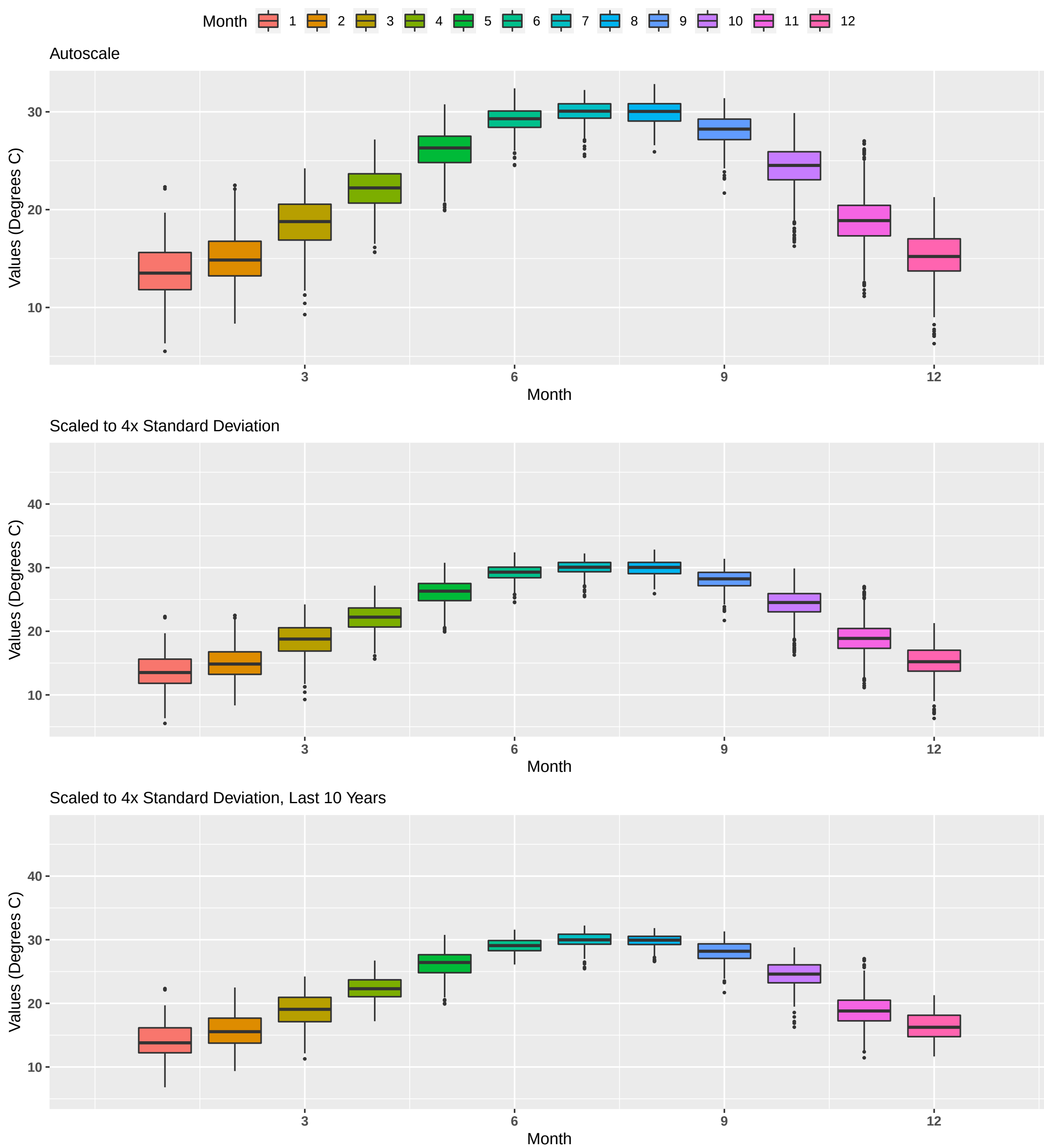
Scaled to 4x Standard Deviation, Last 10 Years



Summary Box Plots for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apadbwq
By Year & Month

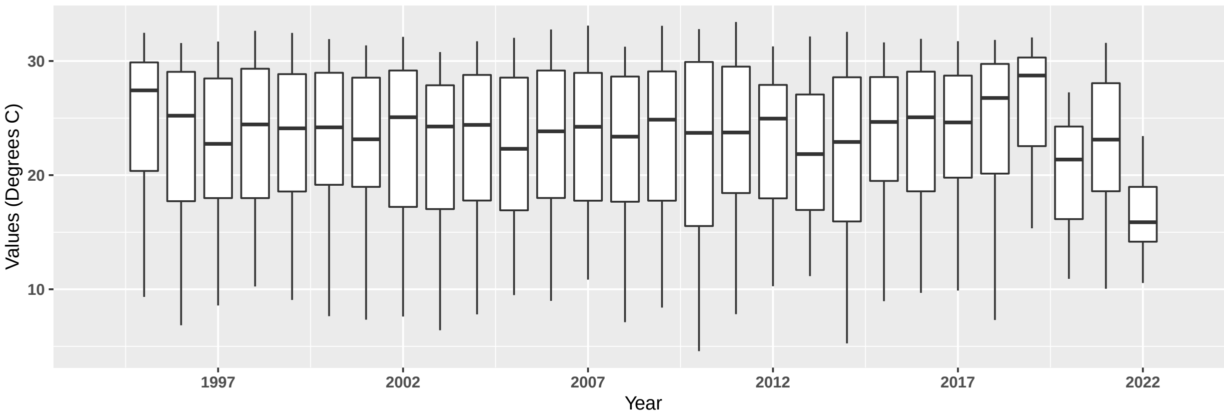


Summary Box Plots for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apadbwq
By Month

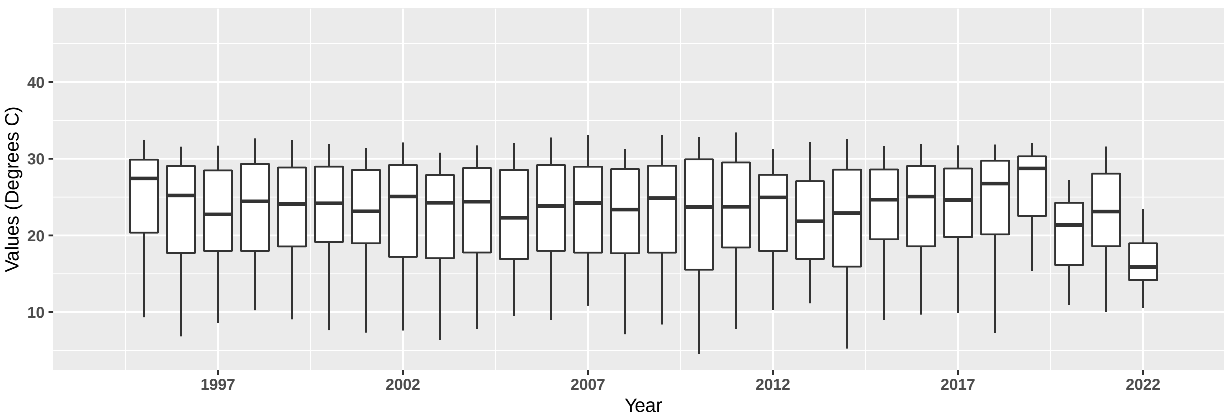


Summary Box Plots for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaebwq
By Year

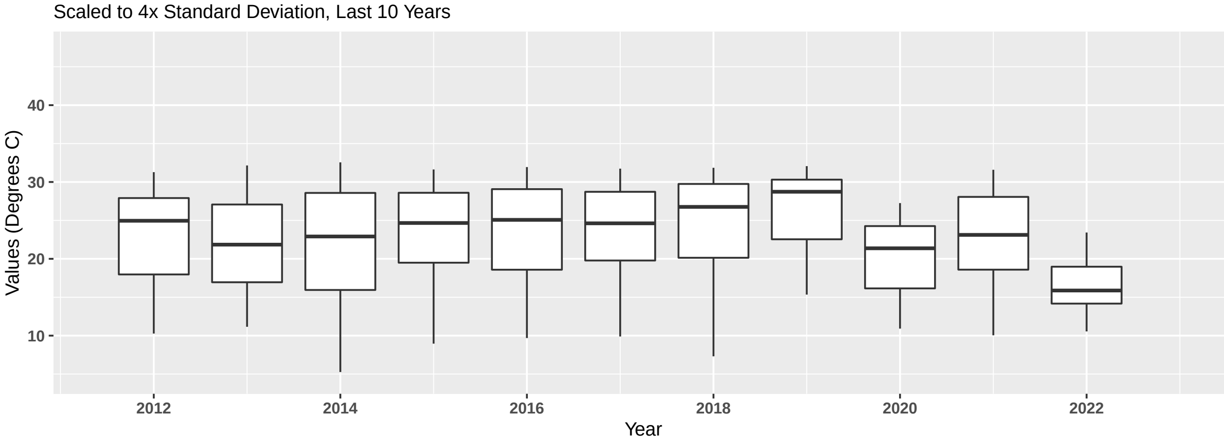
Autoscale



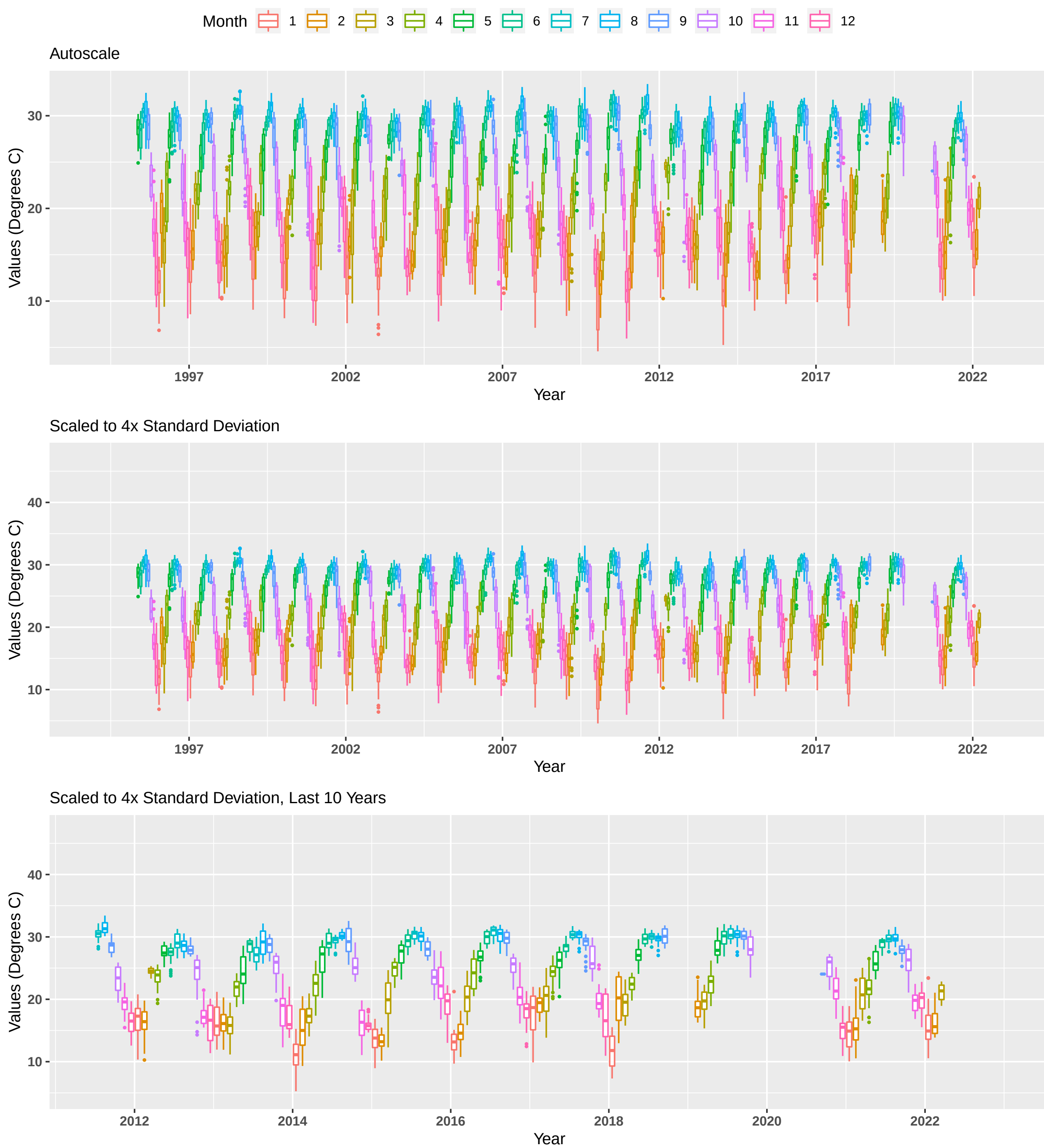
Scaled to 4x Standard Deviation



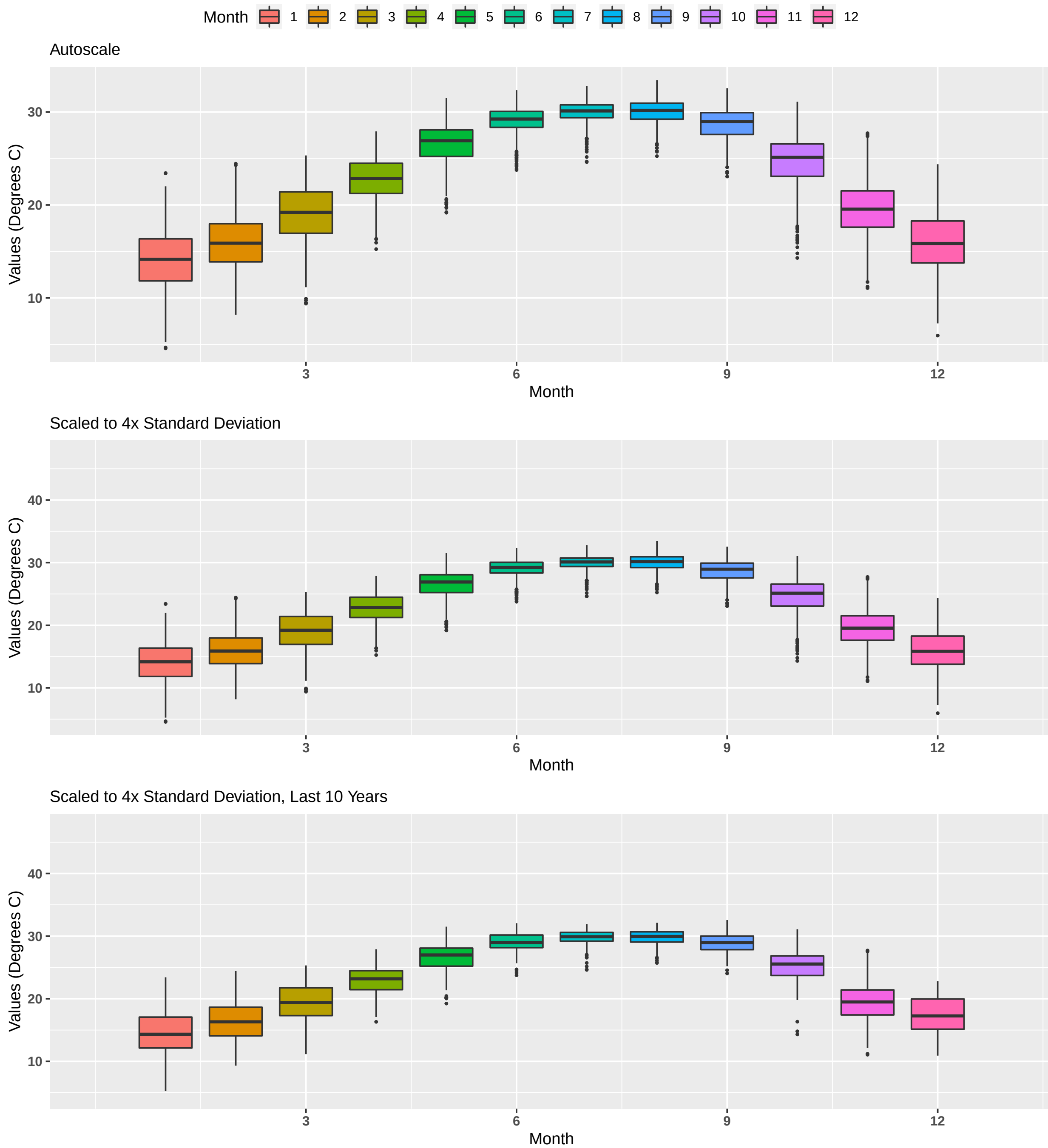
Scaled to 4x Standard Deviation, Last 10 Years



Summary Box Plots for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaebwq
By Year & Month

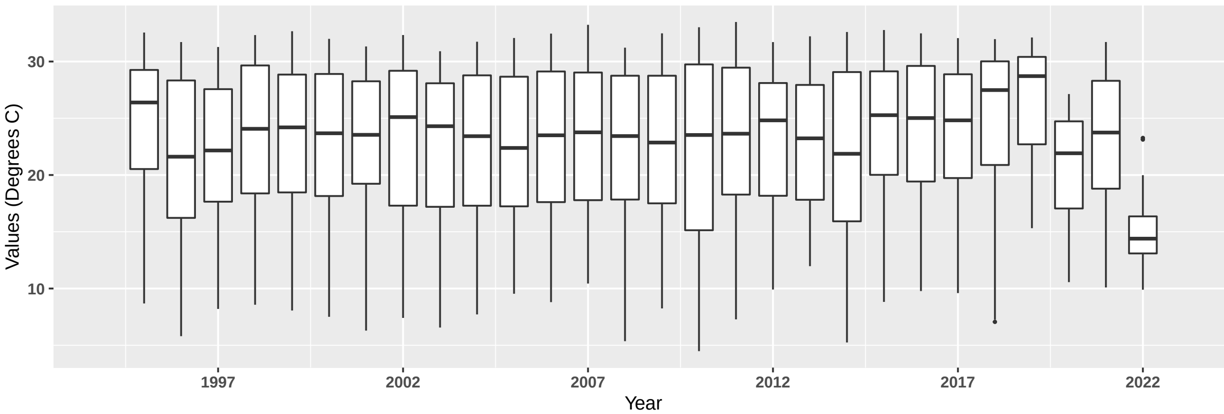


Summary Box Plots for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaebwq
By Month

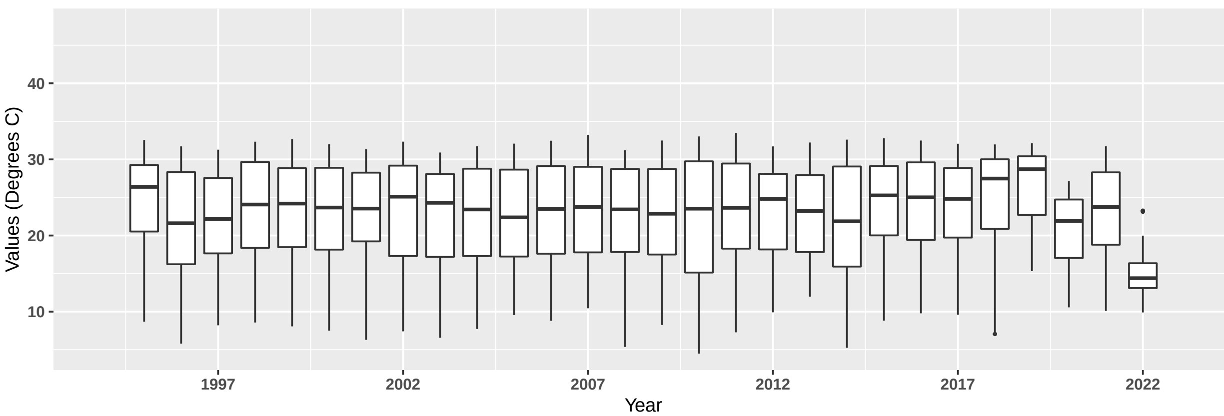


Summary Box Plots for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaeswq
By Year

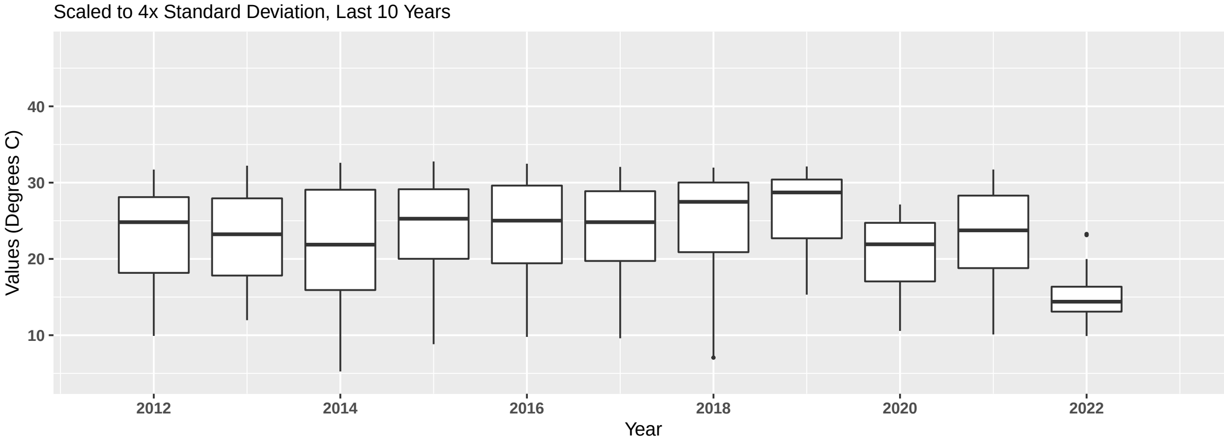
Autoscale



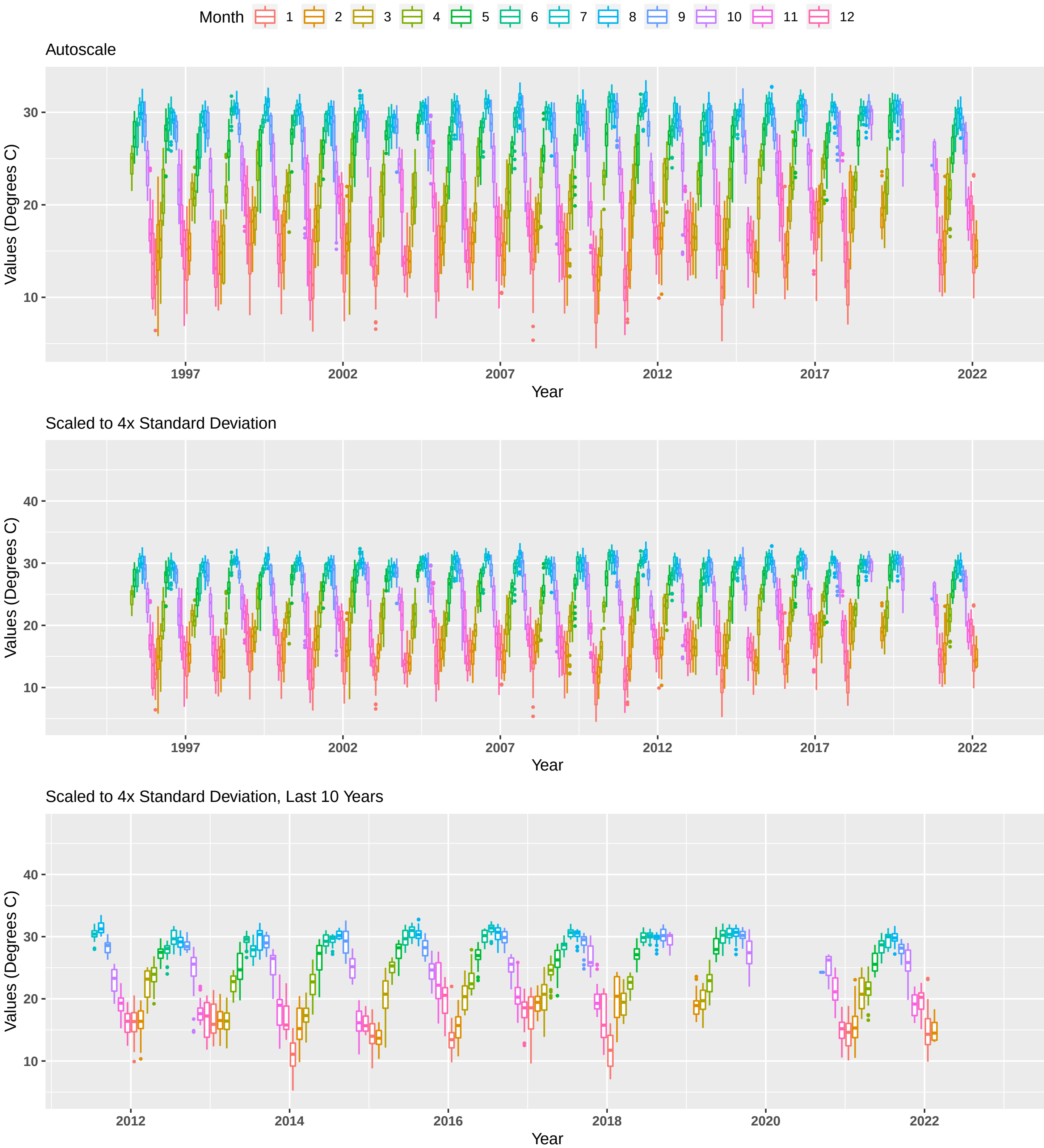
Scaled to 4x Standard Deviation



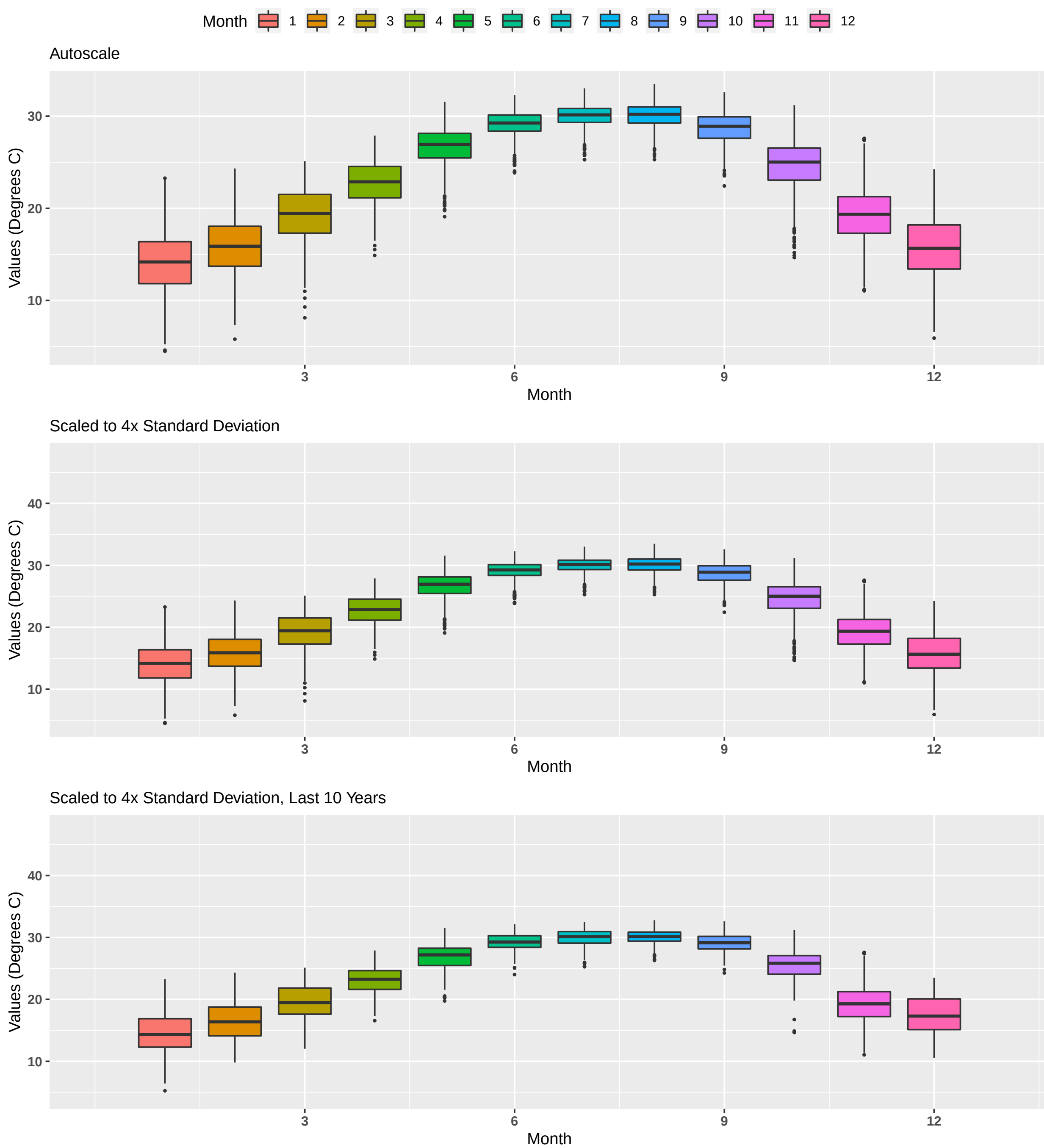
Scaled to 4x Standard Deviation, Last 10 Years



Summary Box Plots for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaeswq
By Year & Month

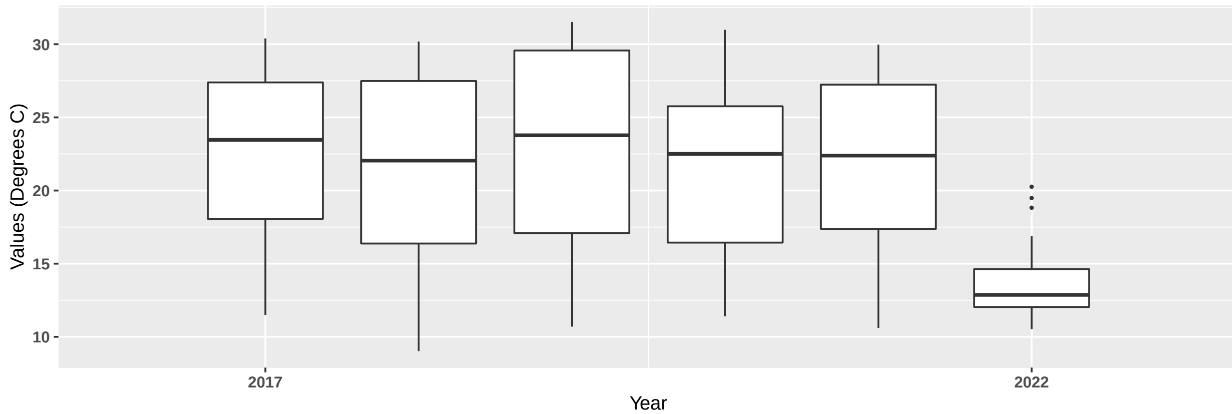


Summary Box Plots for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaeswq
By Month

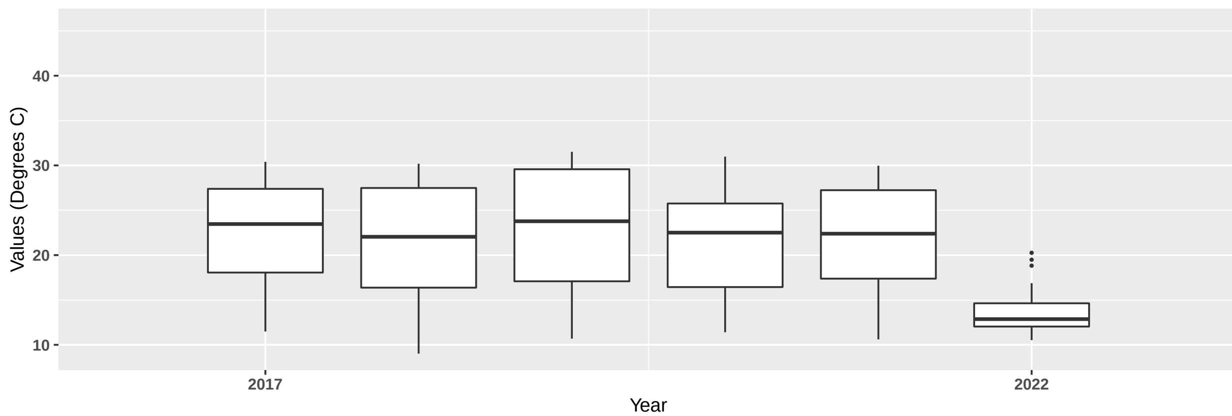


Summary Box Plots for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apalmwq
By Year

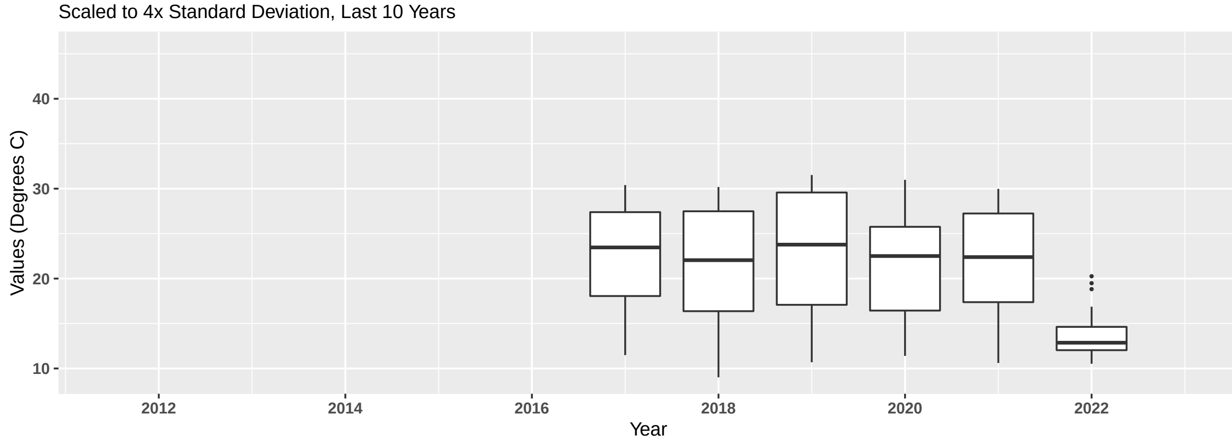
Autoscale



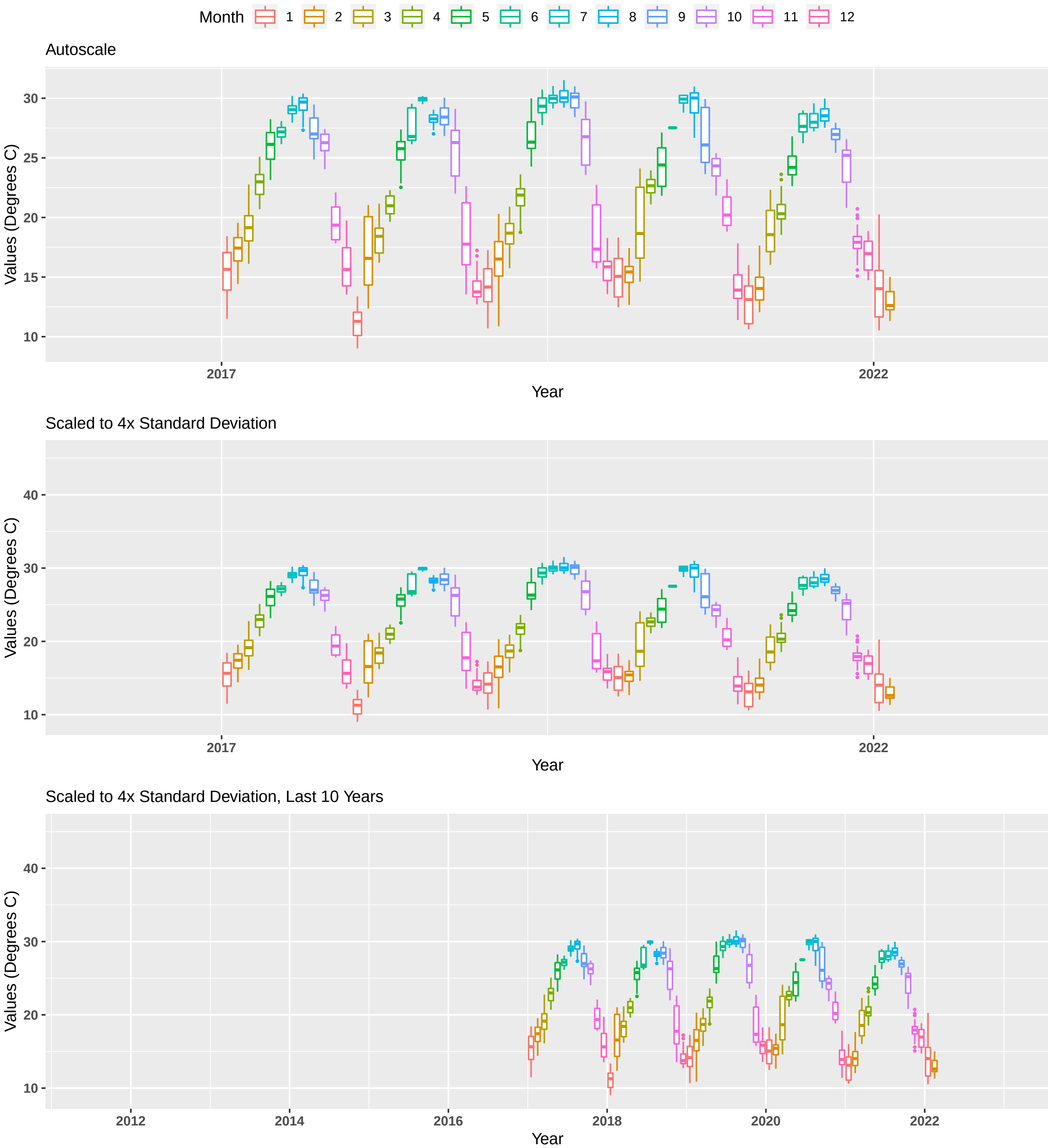
Scaled to 4x Standard Deviation



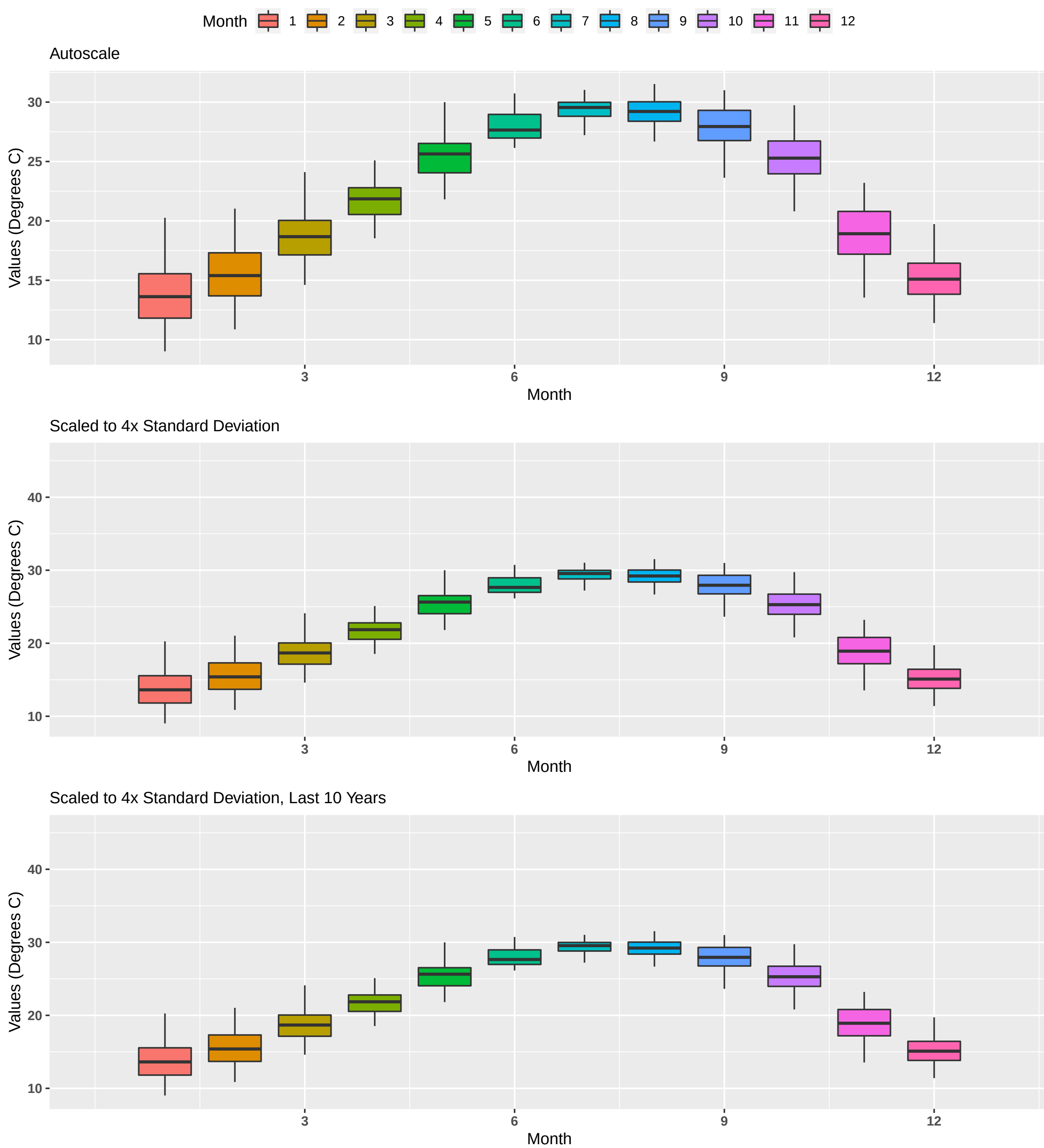
Scaled to 4x Standard Deviation, Last 10 Years



Summary Box Plots for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apalmwq
By Year & Month

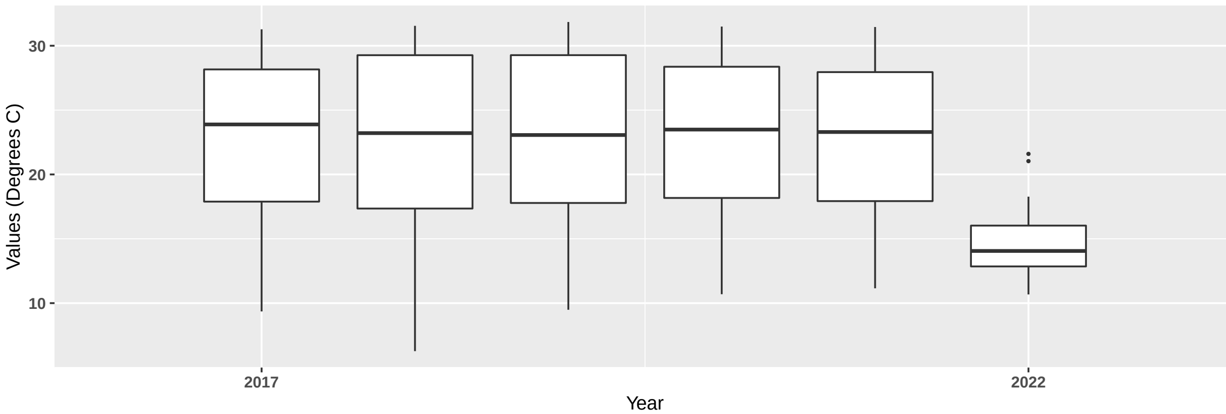


Summary Box Plots for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apalmwq
By Month

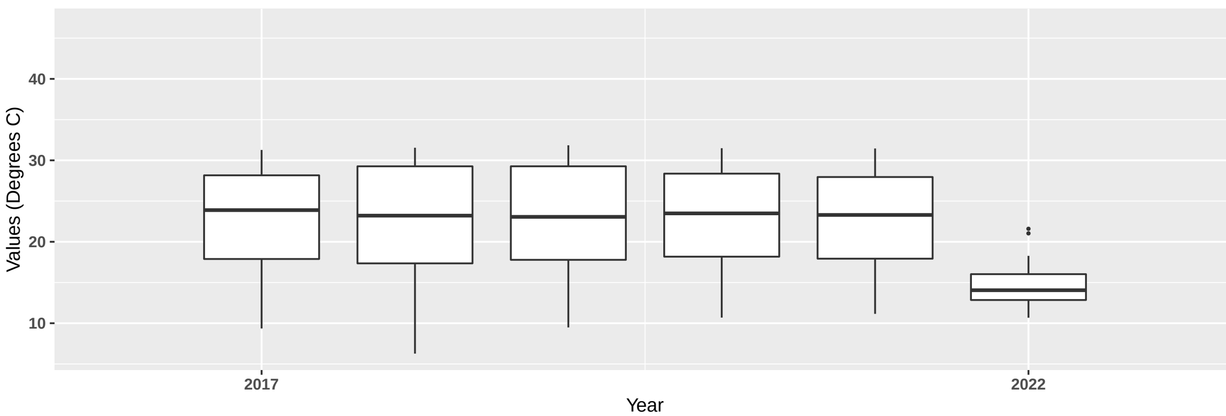


Summary Box Plots for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apapcwq
By Year

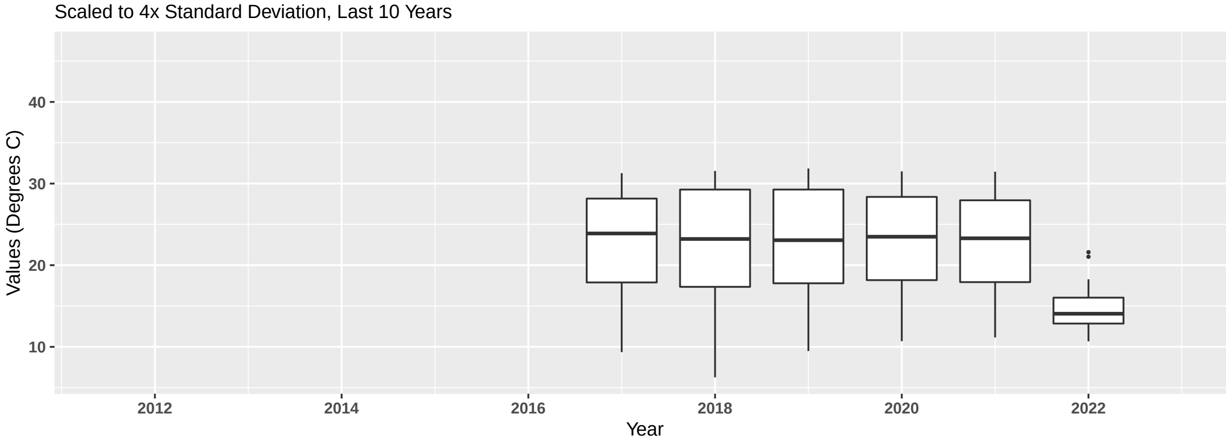
Autoscale



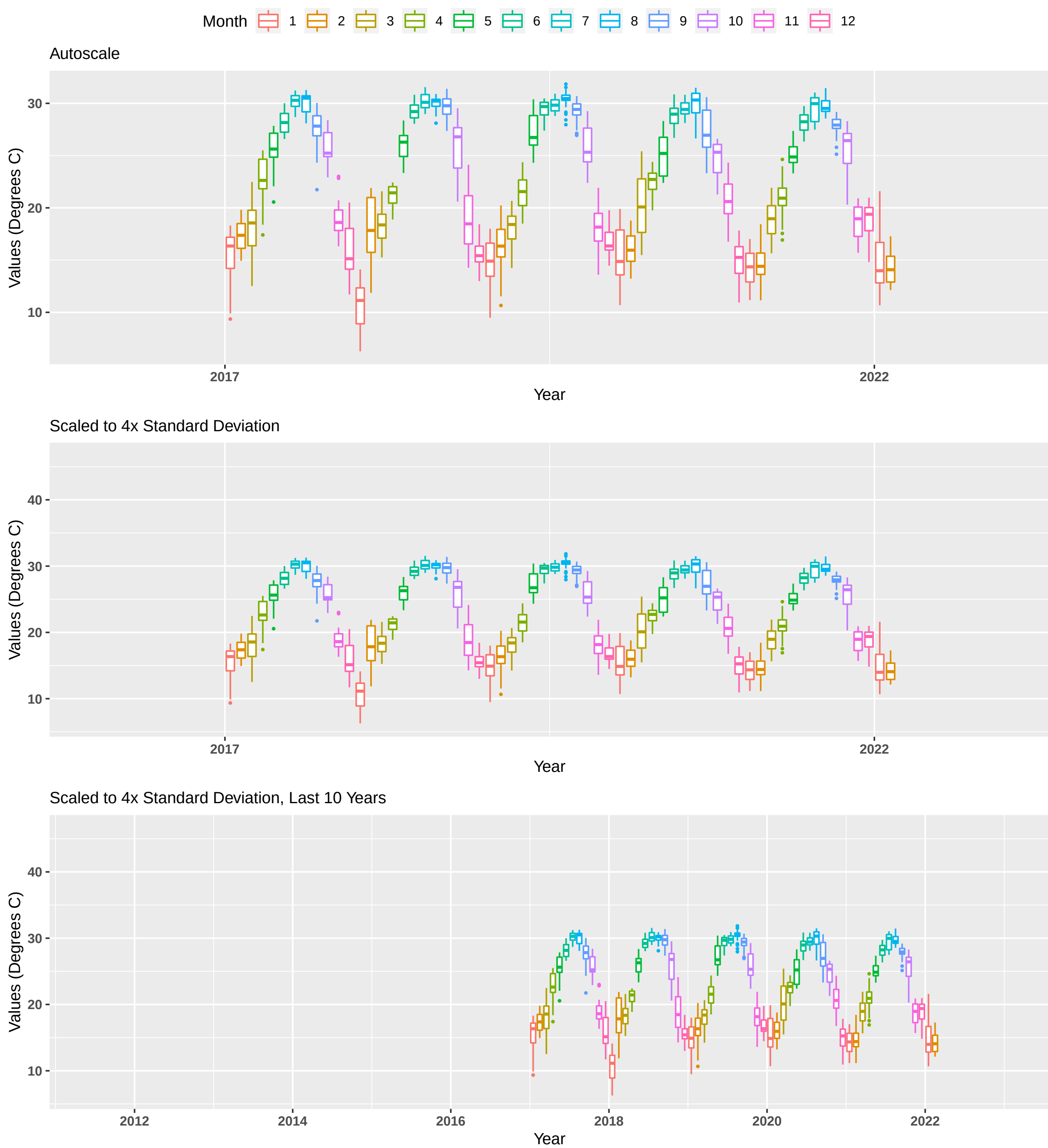
Scaled to 4x Standard Deviation



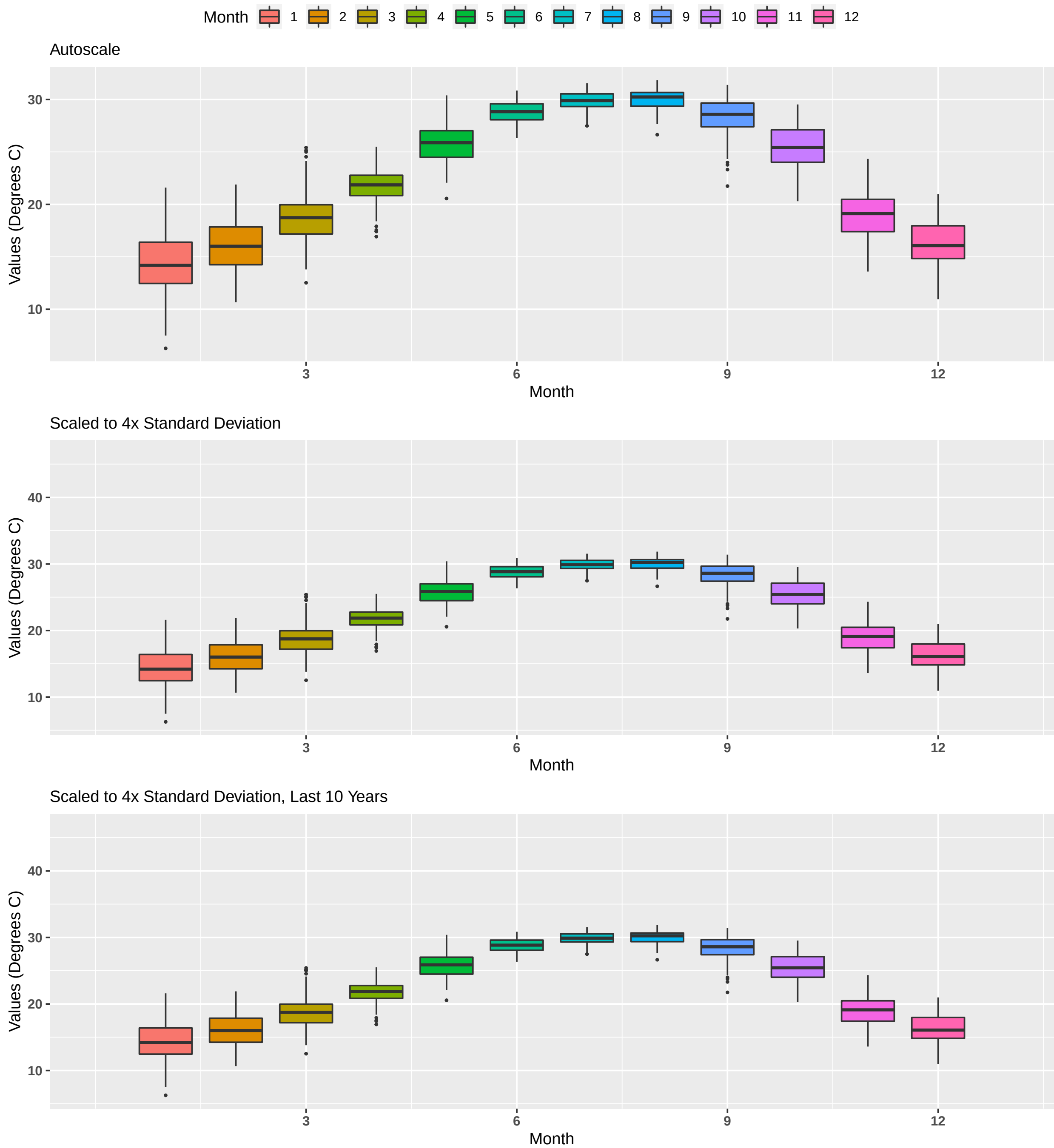
Scaled to 4x Standard Deviation, Last 10 Years



Summary Box Plots for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apapcwq
By Year & Month

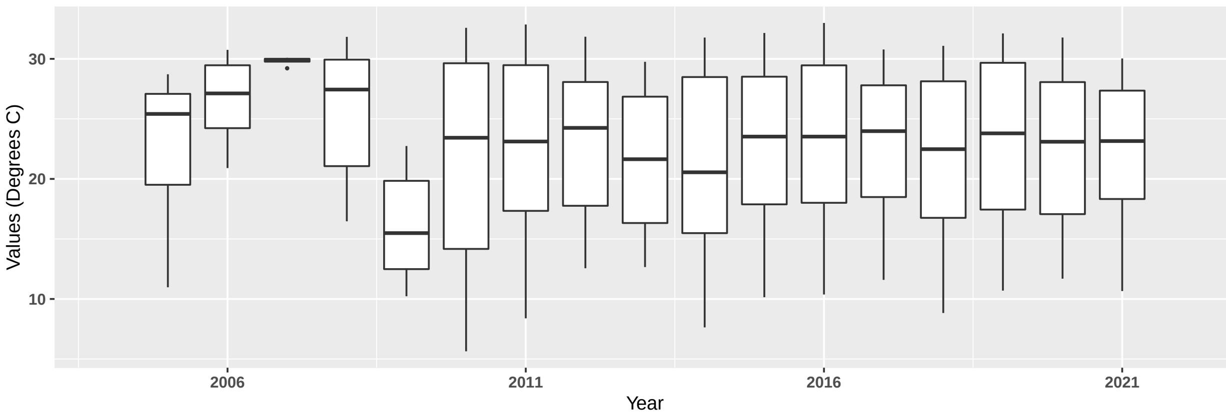


Summary Box Plots for Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apapcwq
By Month

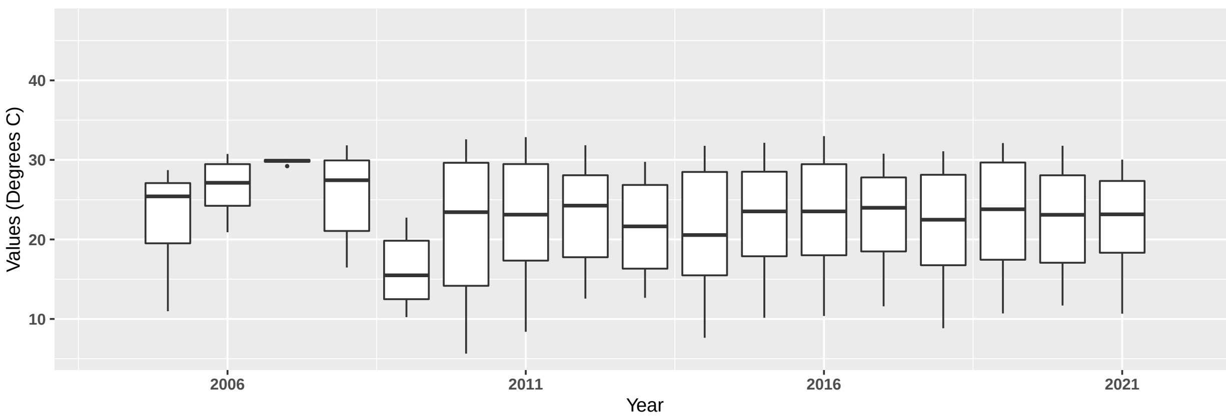


Summary Box Plots for Apalachicola National Estuarine Research Reserve
5 | National Data Buoy Center | APCF1
By Year

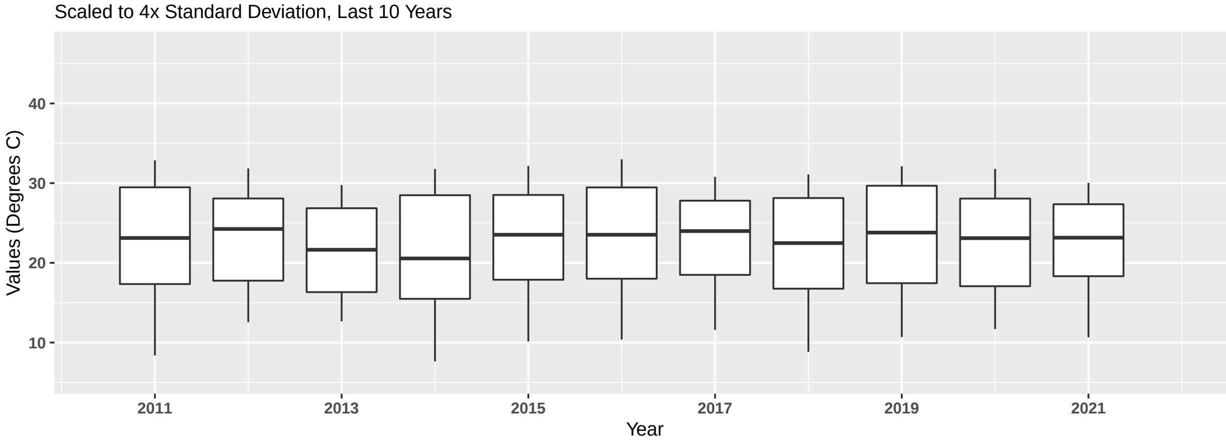
Autoscale



Scaled to 4x Standard Deviation

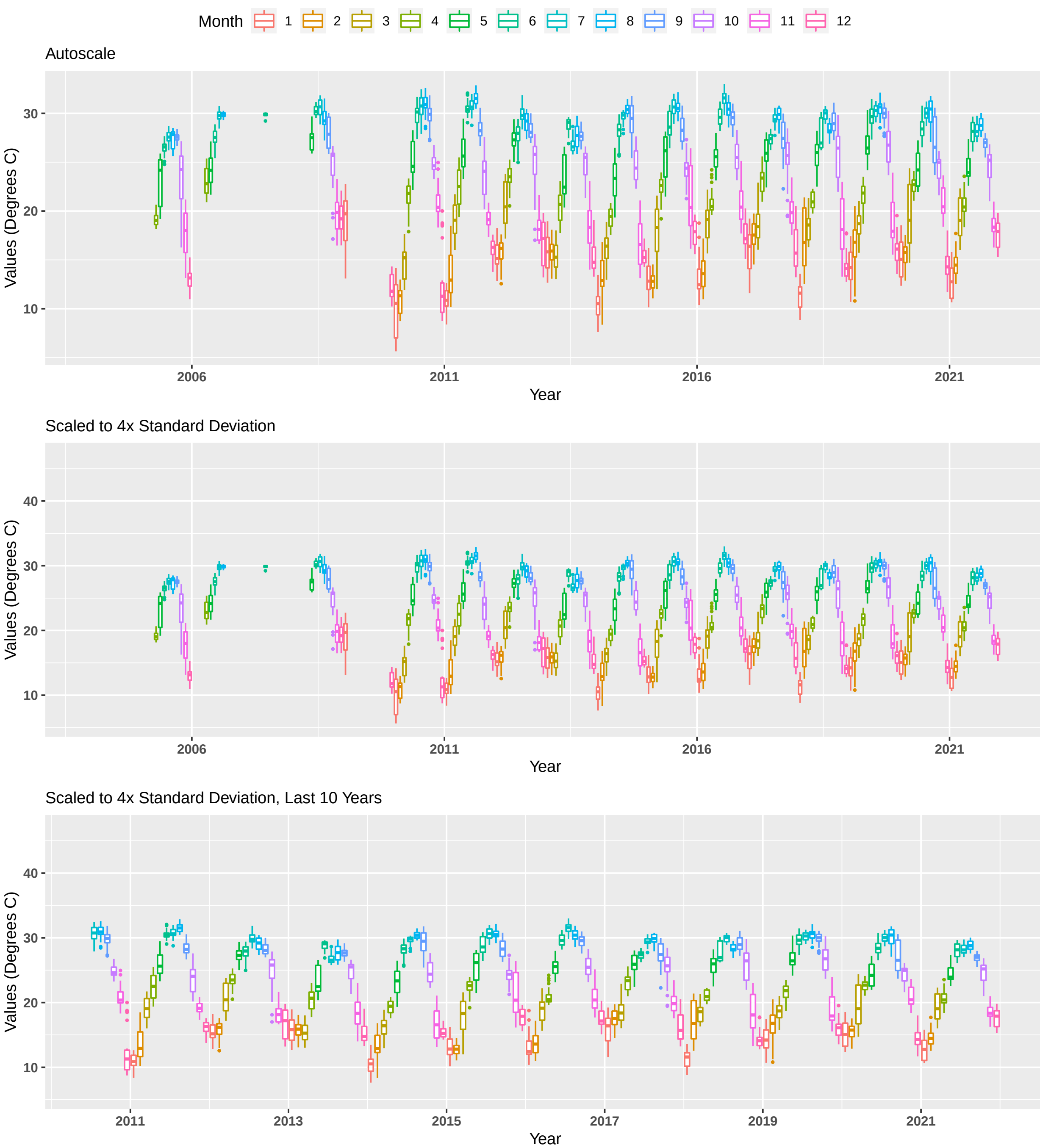


Scaled to 4x Standard Deviation, Last 10 Years

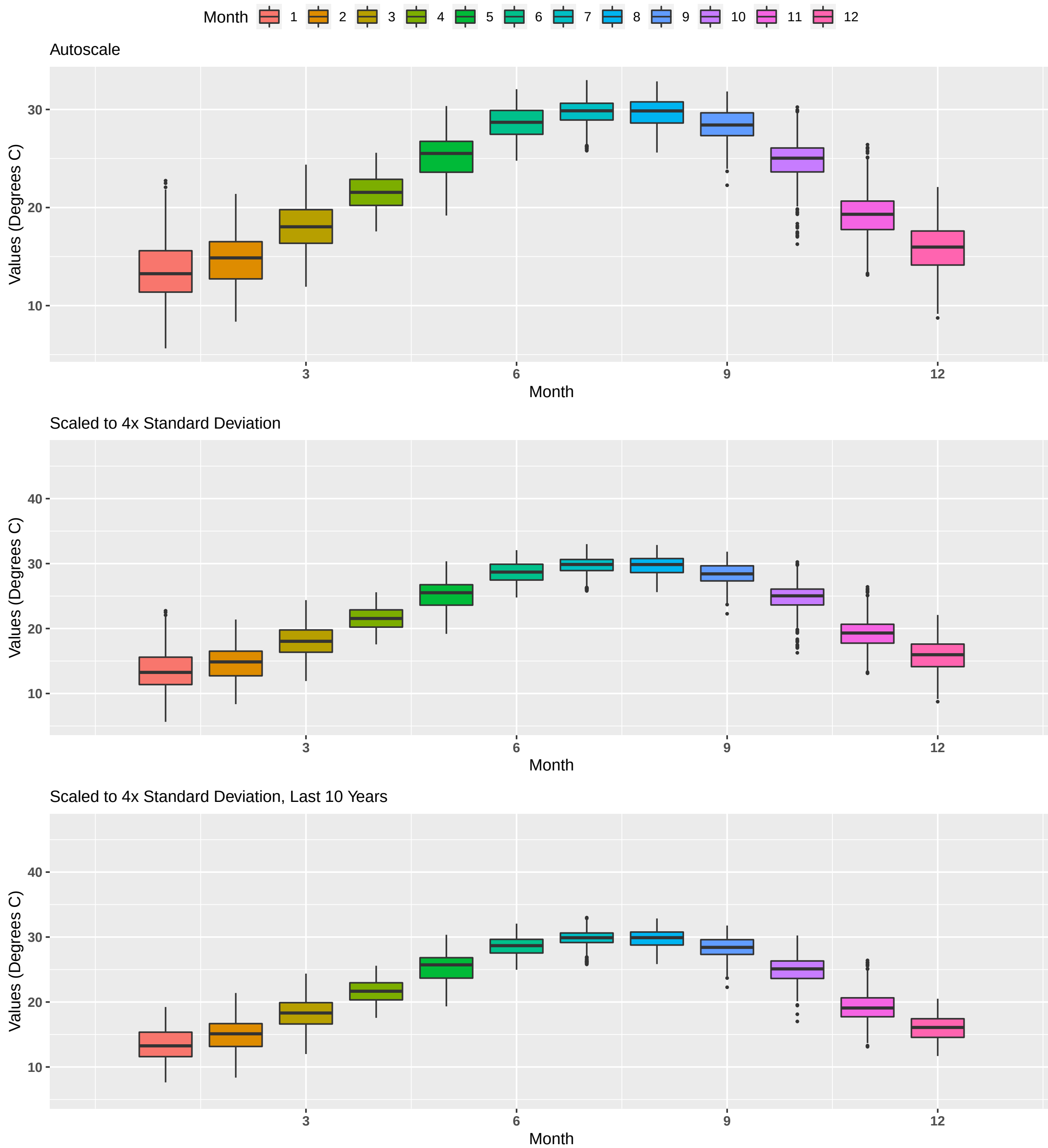


Summary Box Plots for Apalachicola National Estuarine Research Reserve
5 | National Data Buoy Center | APCF1

By Year & Month

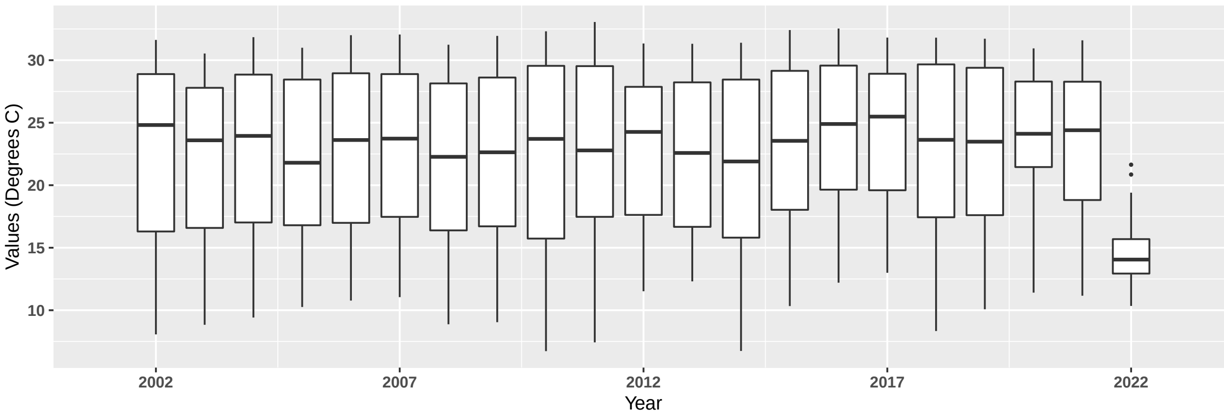


Summary Box Plots for Apalachicola National Estuarine Research Reserve
5 | National Data Buoy Center | APCF1
By Month

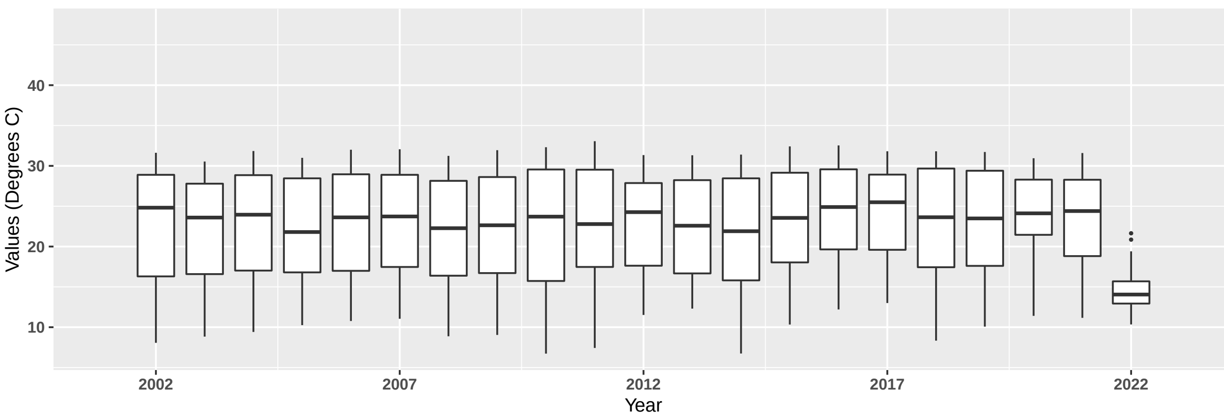


Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apacpwq
By Year

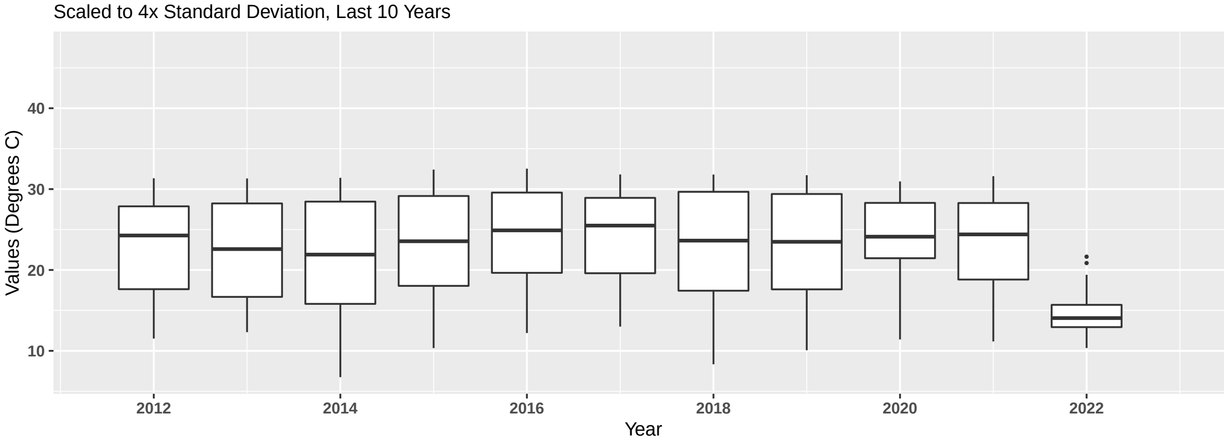
Autoscale



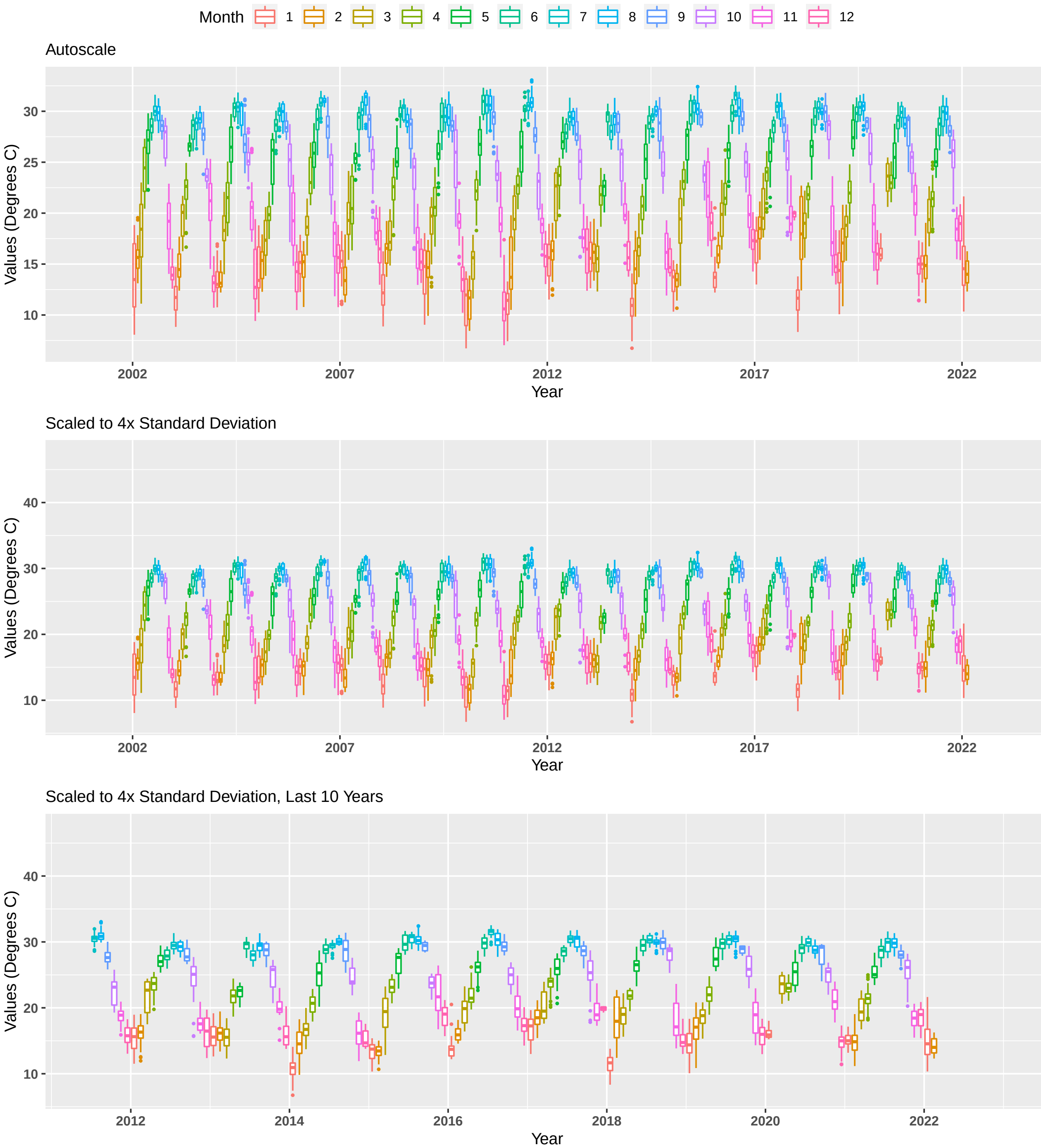
Scaled to 4x Standard Deviation



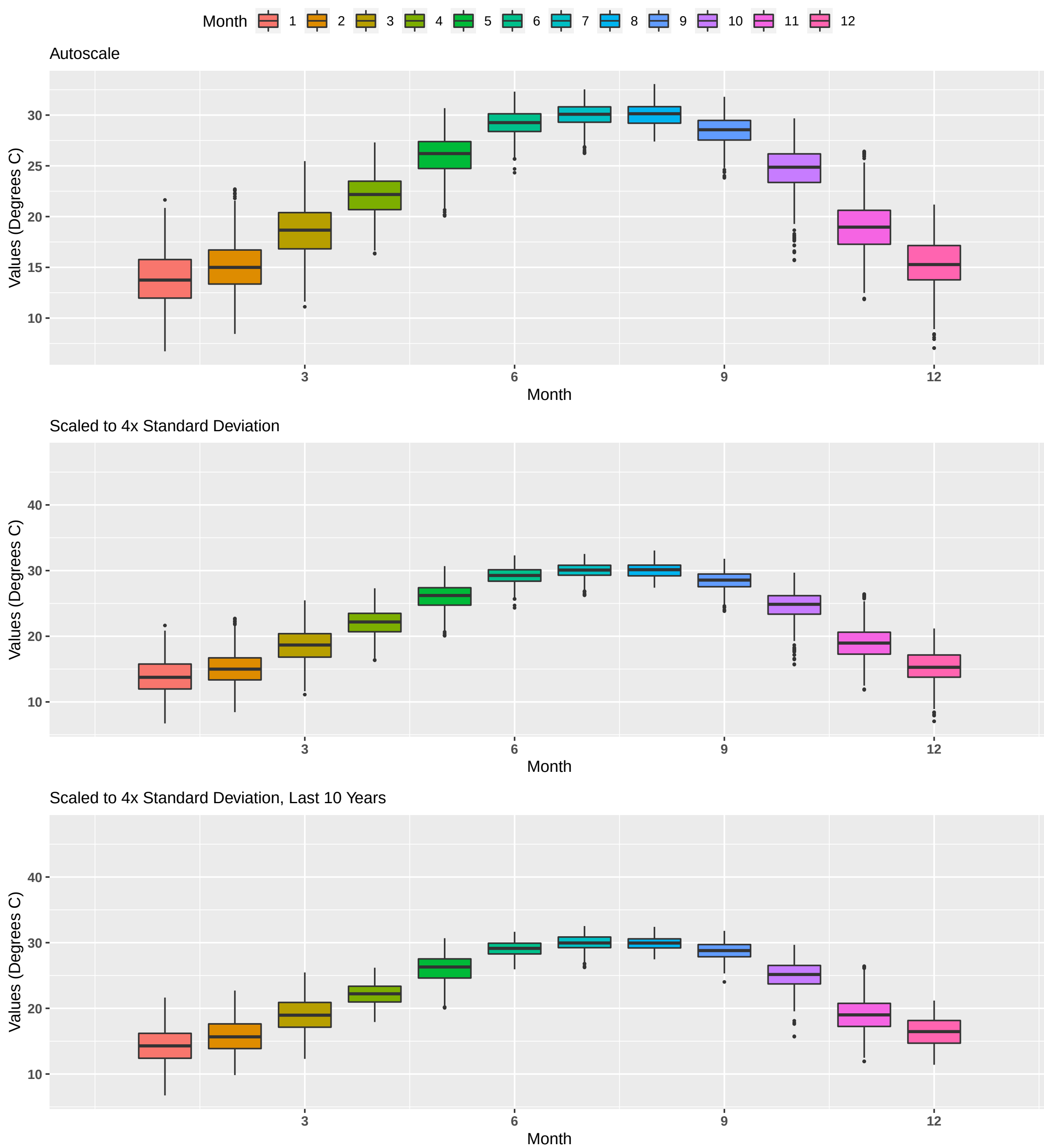
Scaled to 4x Standard Deviation, Last 10 Years



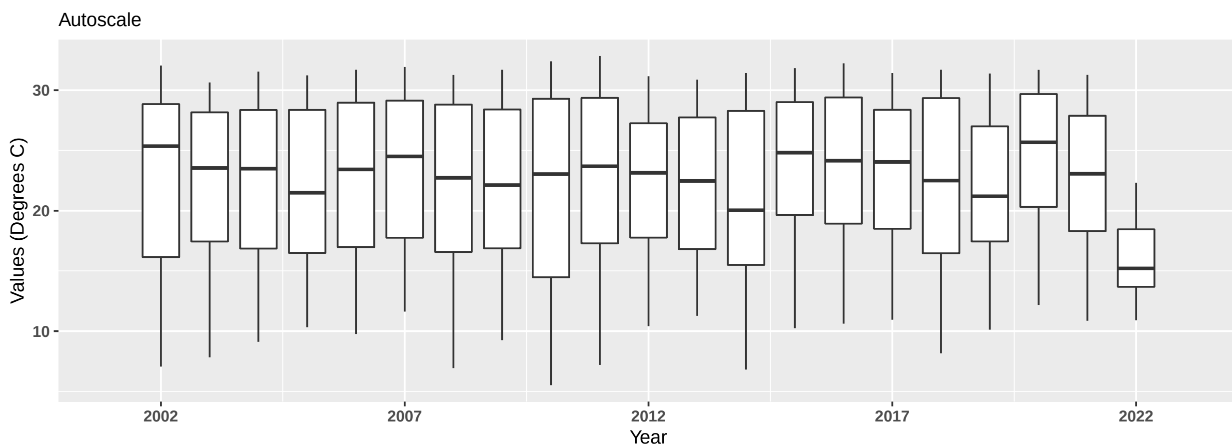
Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apacpwq
By Year & Month

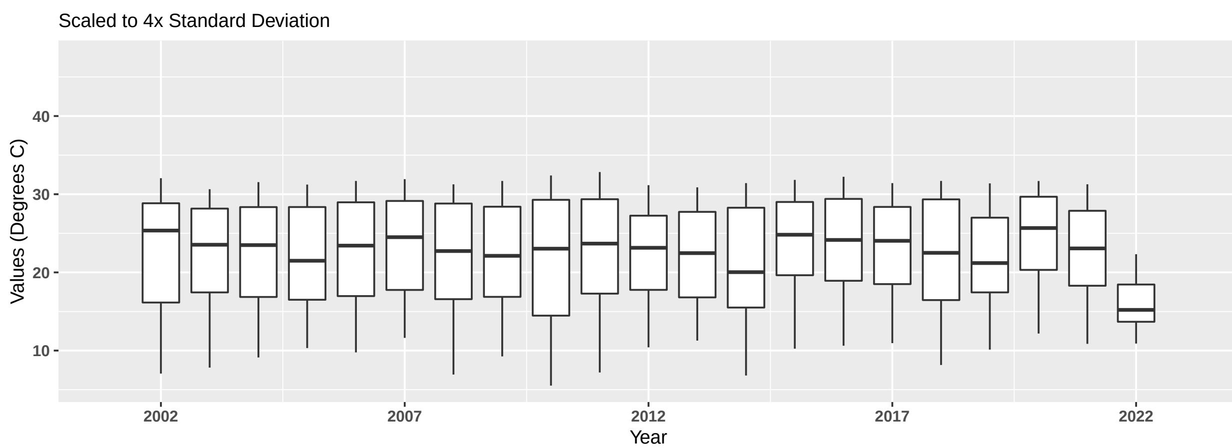


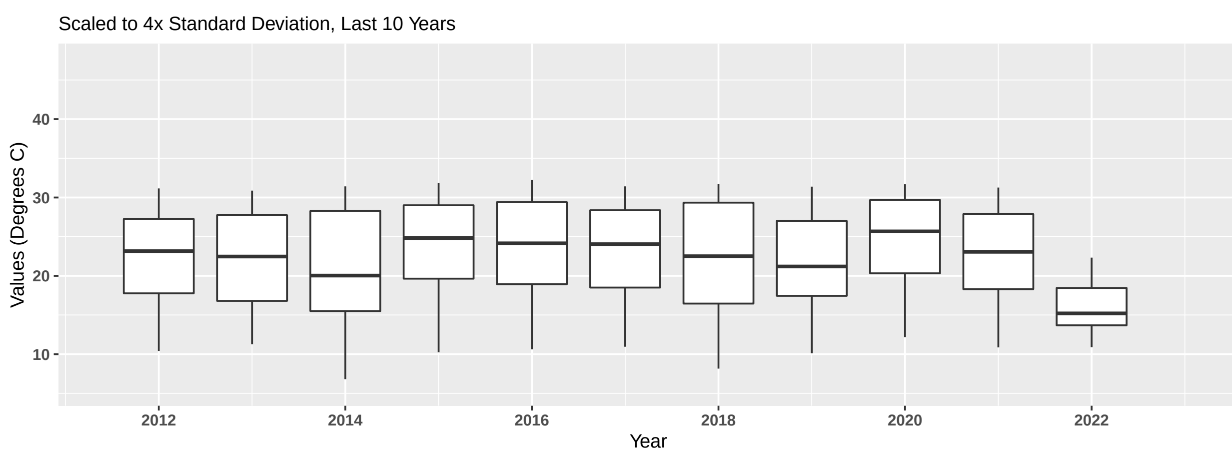
Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apacpwq
By Month



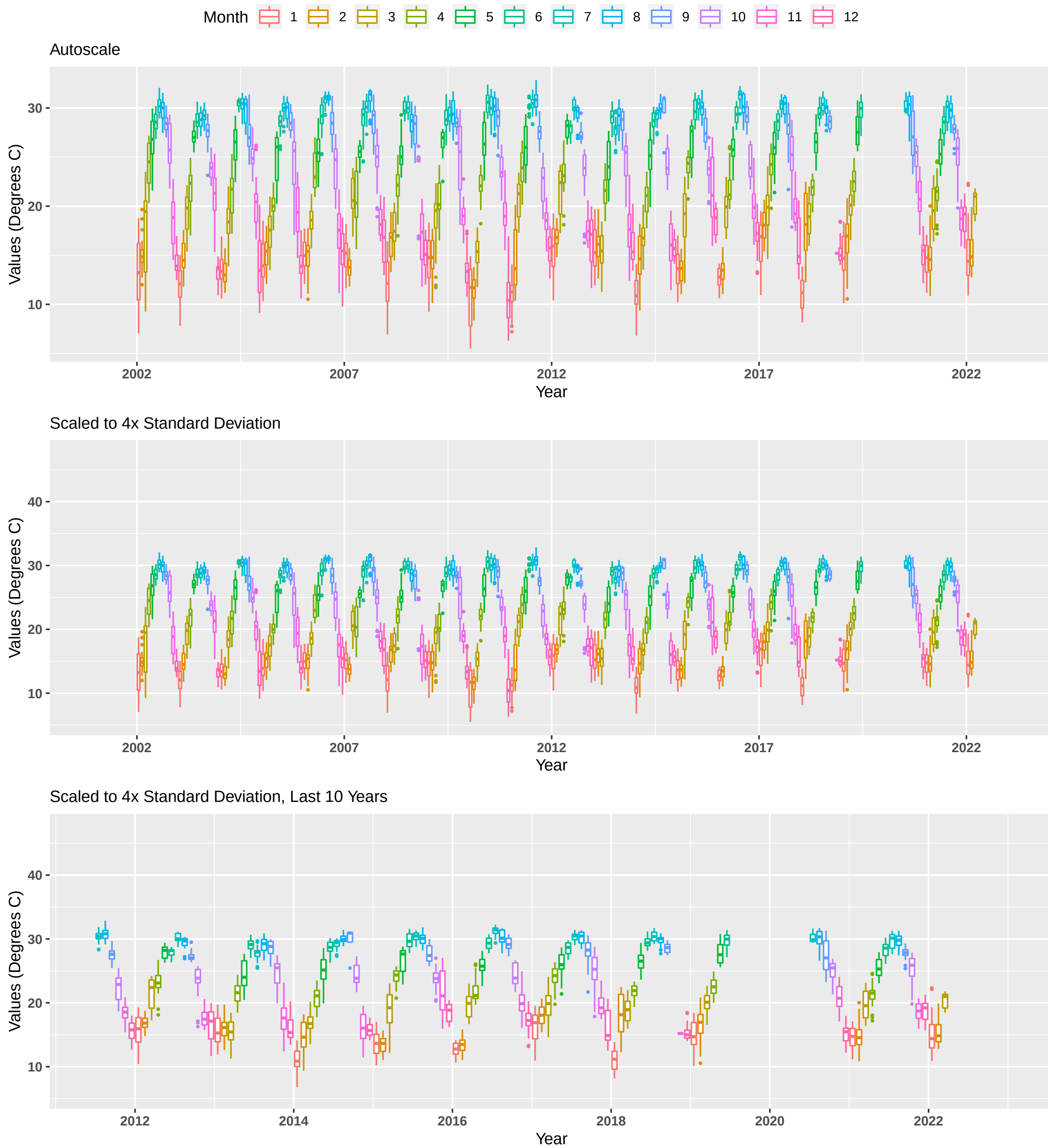
Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apadbwq
By Year



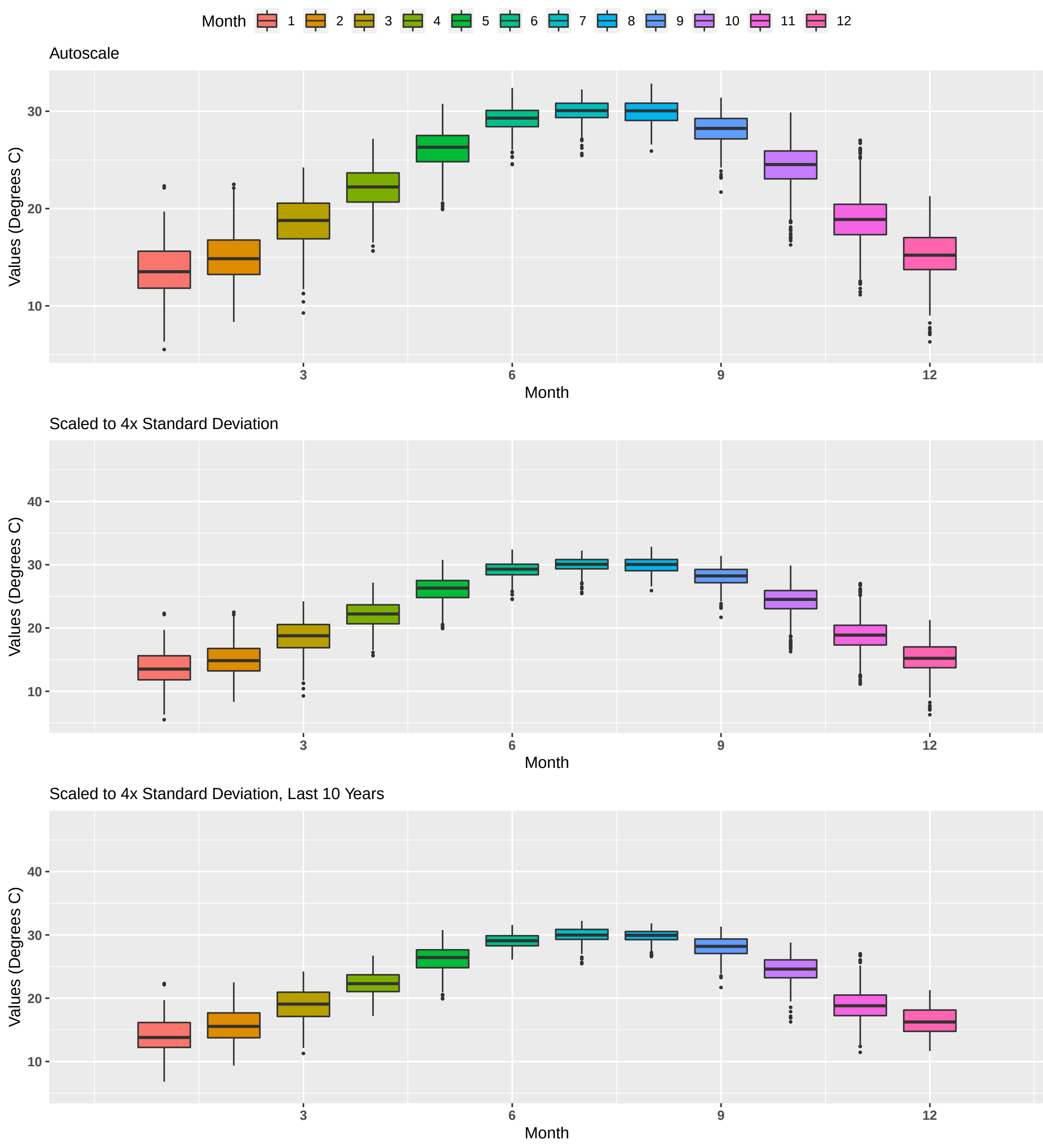




Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apadbwq
By Year & Month

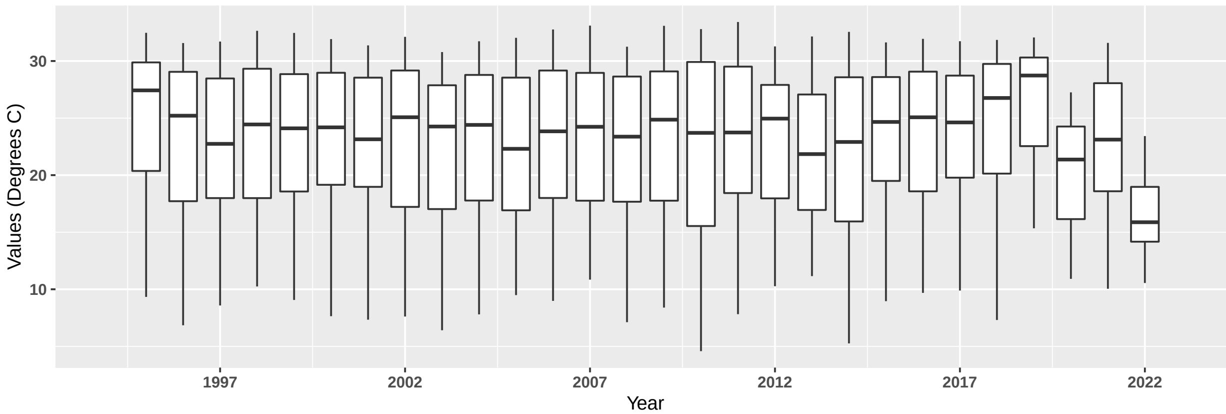


Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apadbwq
By Month

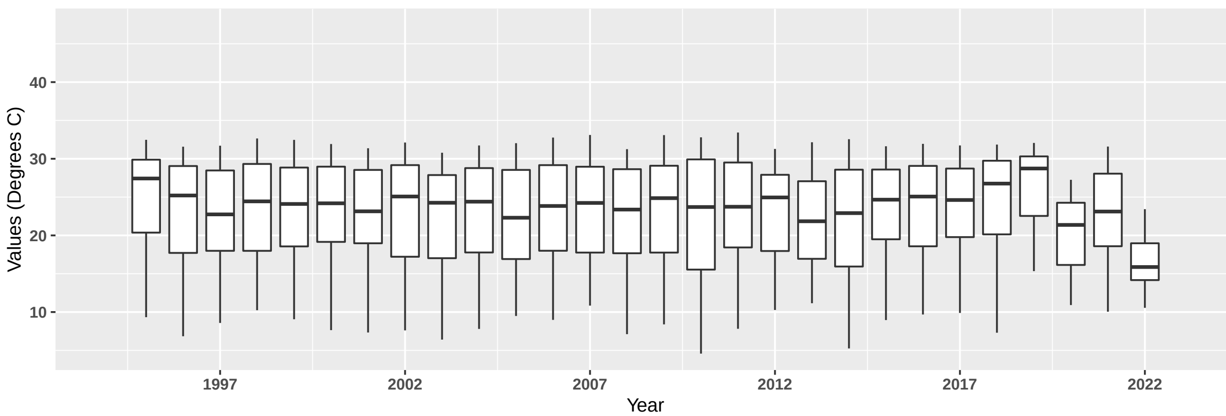


Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaebwq
By Year

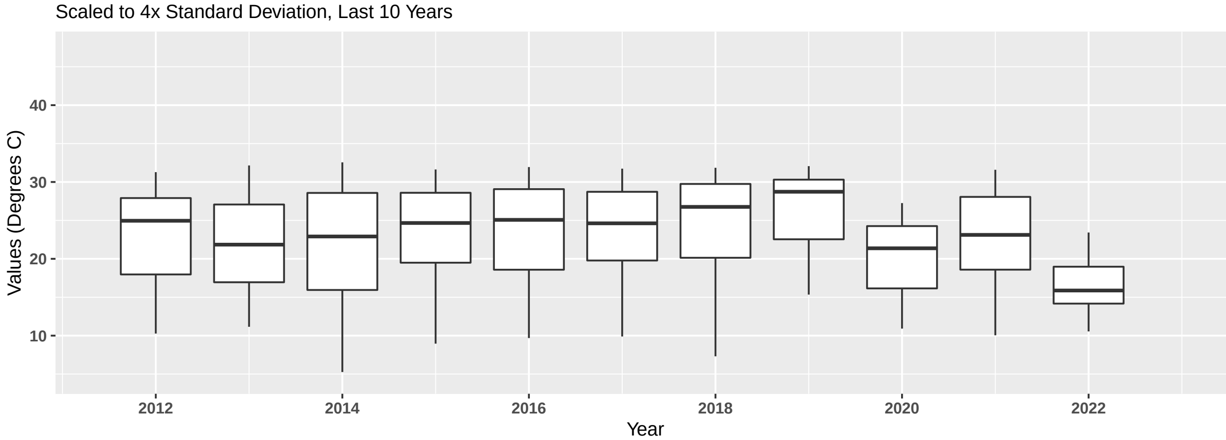
Autoscale



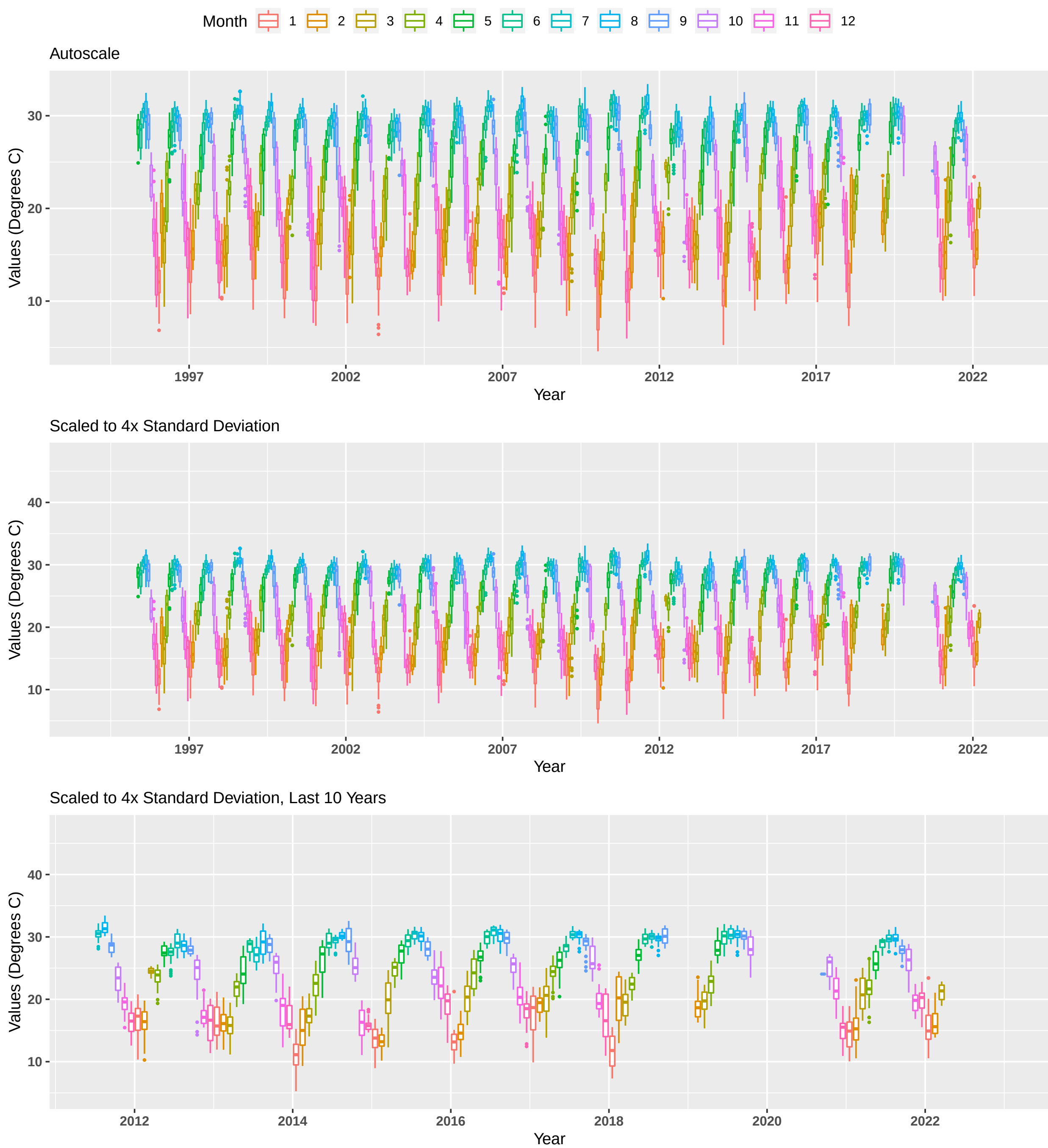
Scaled to 4x Standard Deviation



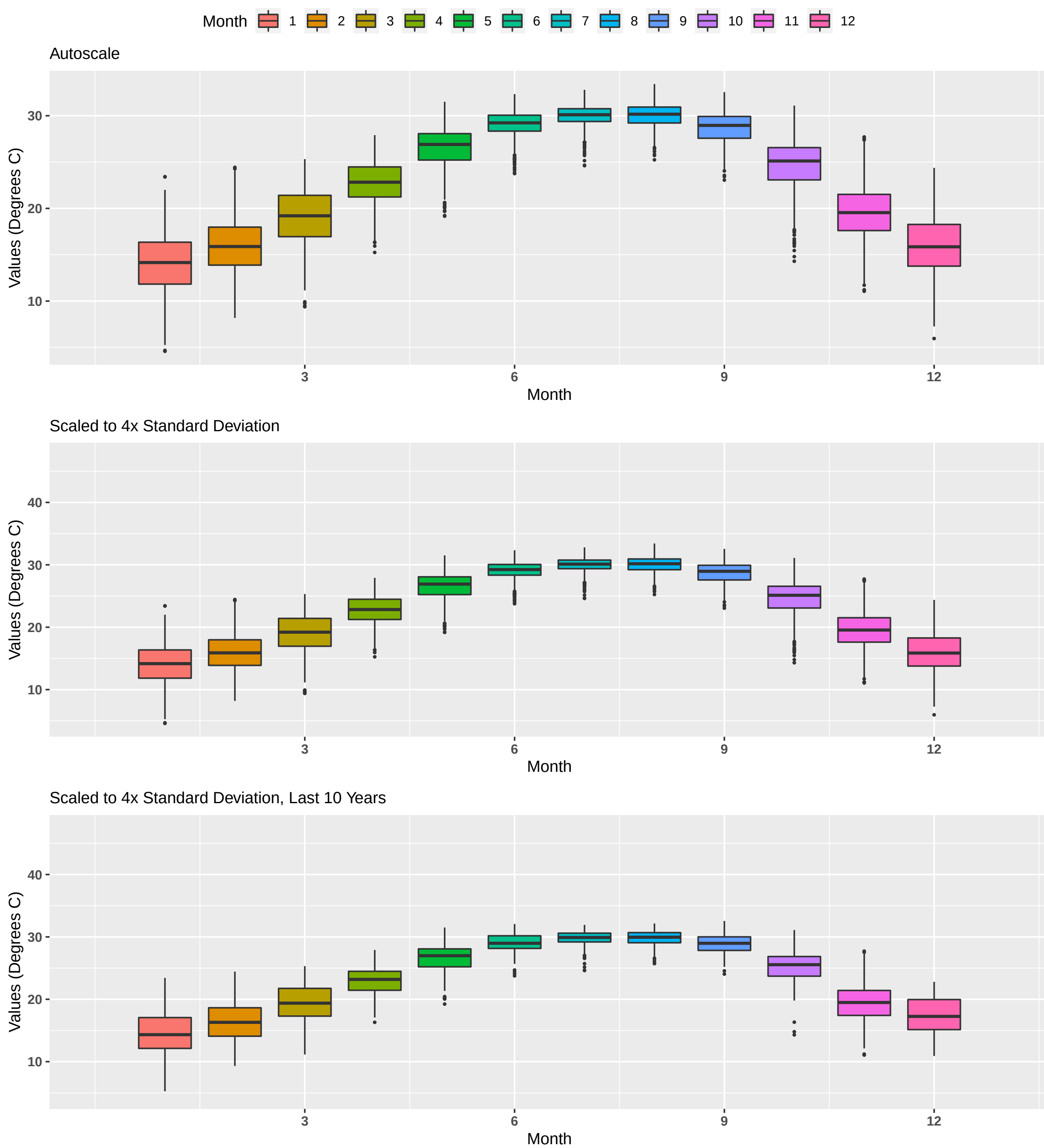
Scaled to 4x Standard Deviation, Last 10 Years



Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaebwq
By Year & Month

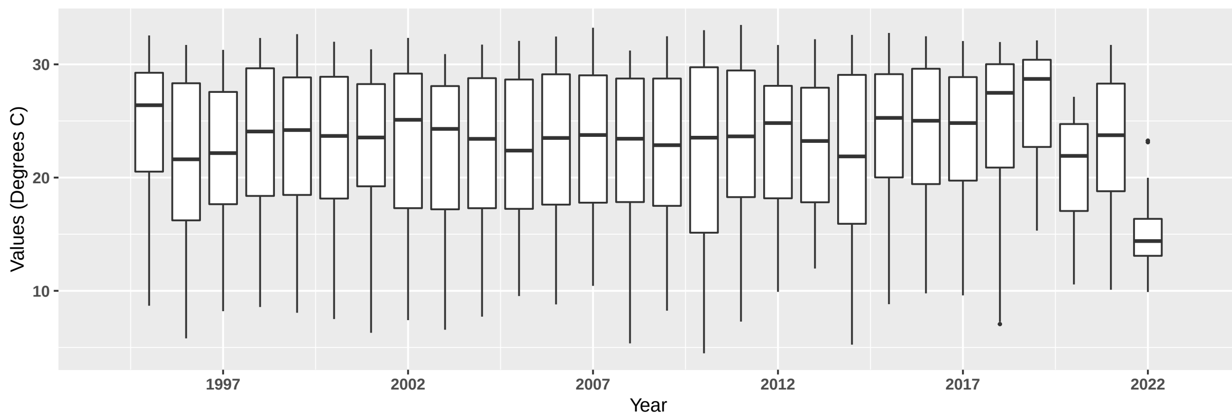


Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaebwq
By Month

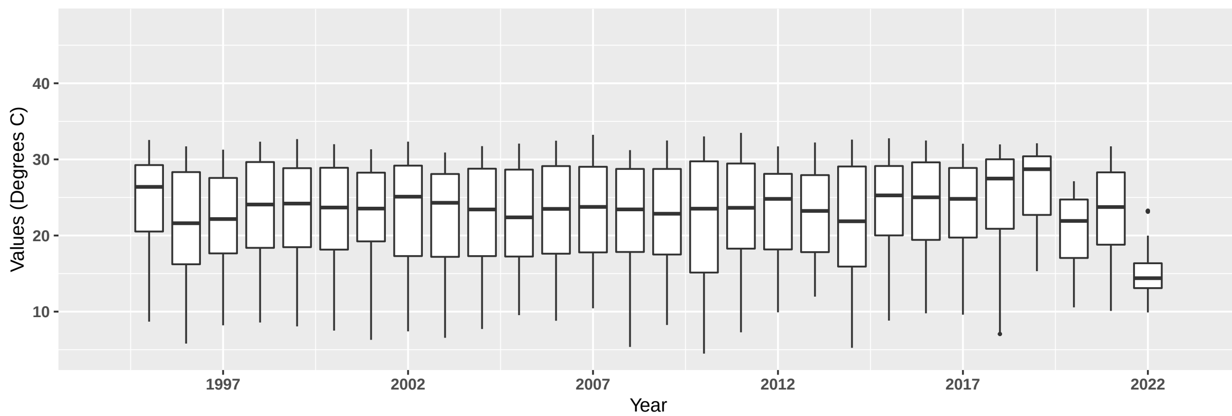


Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaeswq
By Year

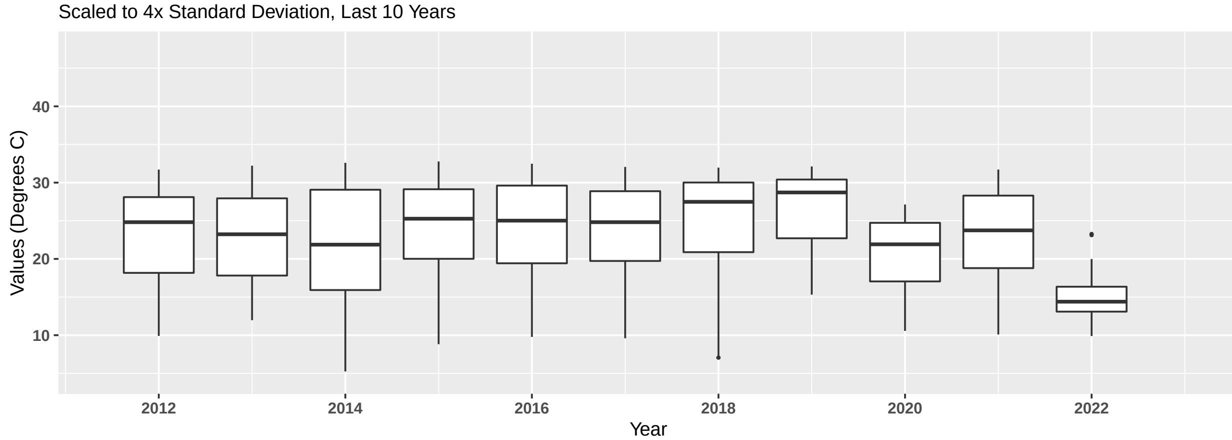
Autoscale



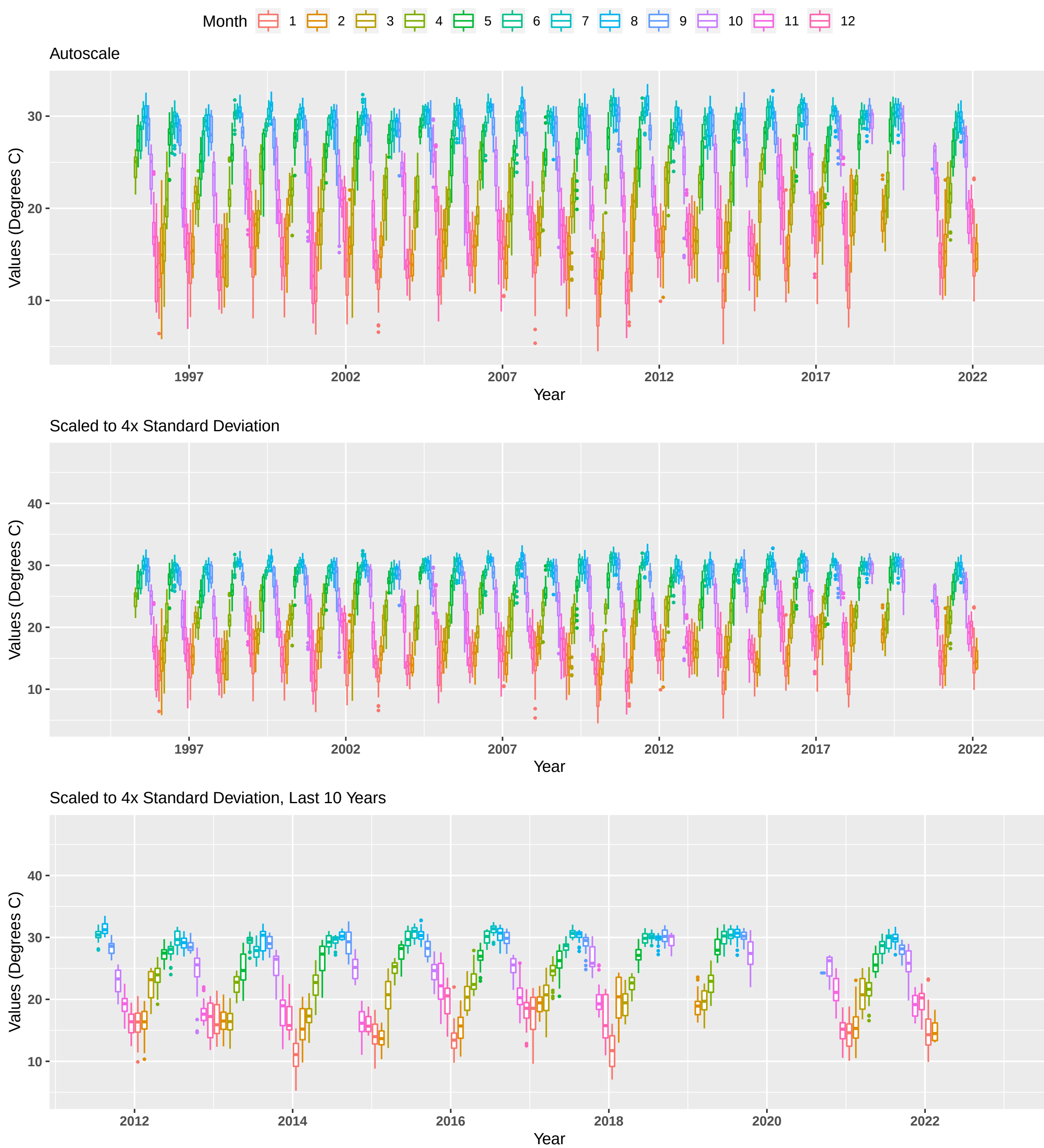
Scaled to 4x Standard Deviation



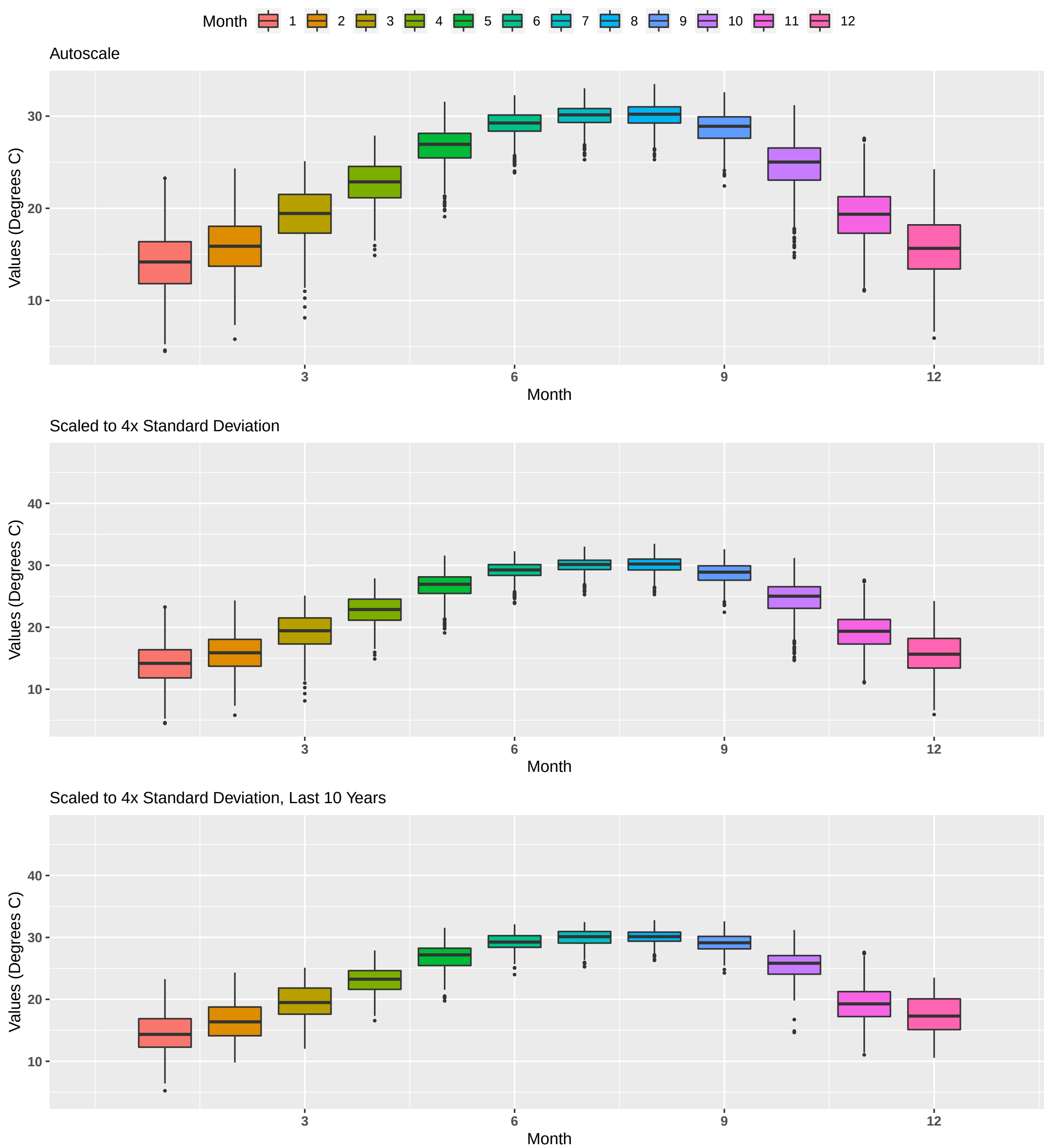
Scaled to 4x Standard Deviation, Last 10 Years



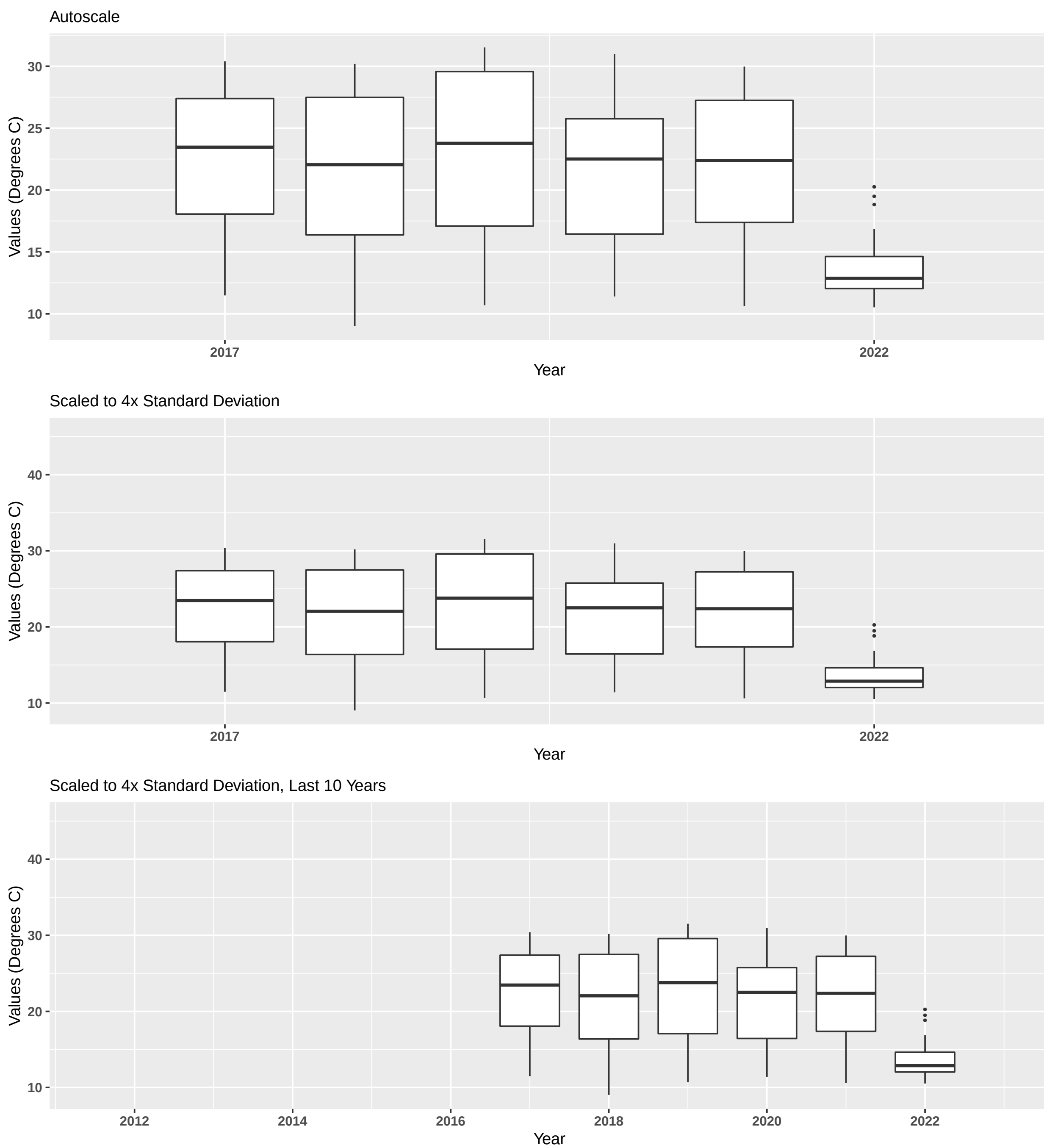
Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaeswq
By Year & Month



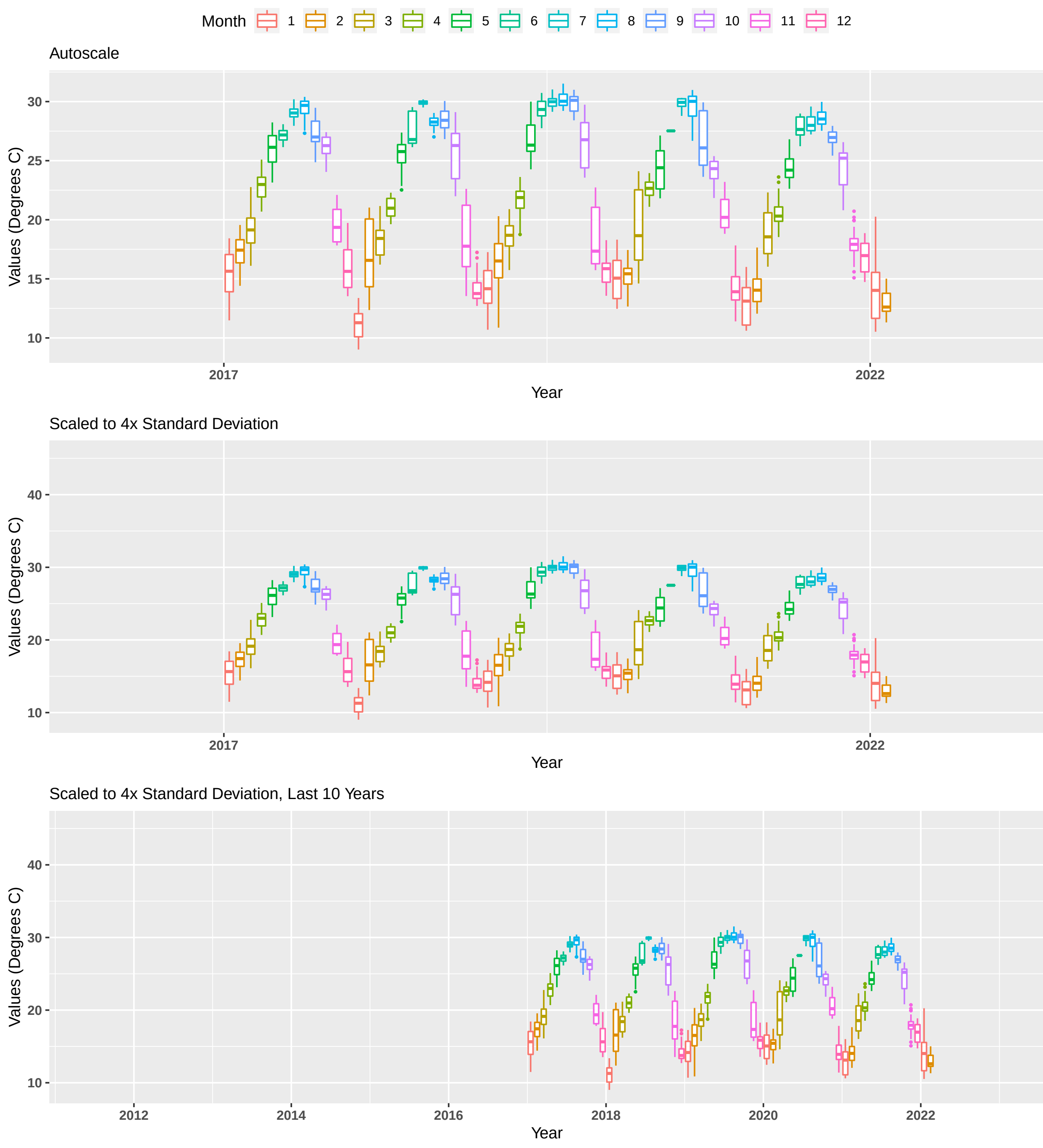
Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apaeswq
By Month



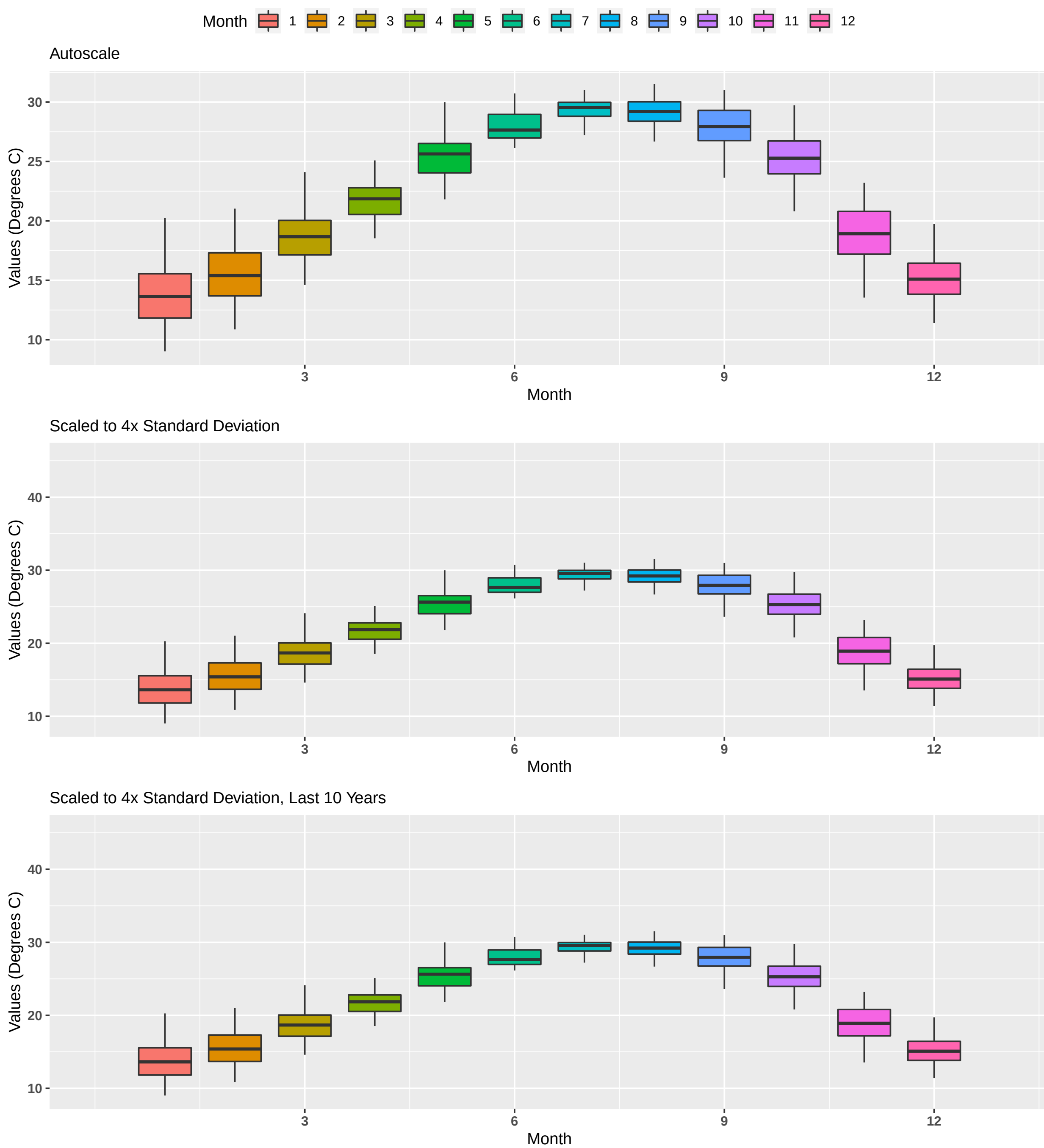
Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apalmwq
By Year



Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apalmwq
By Year & Month

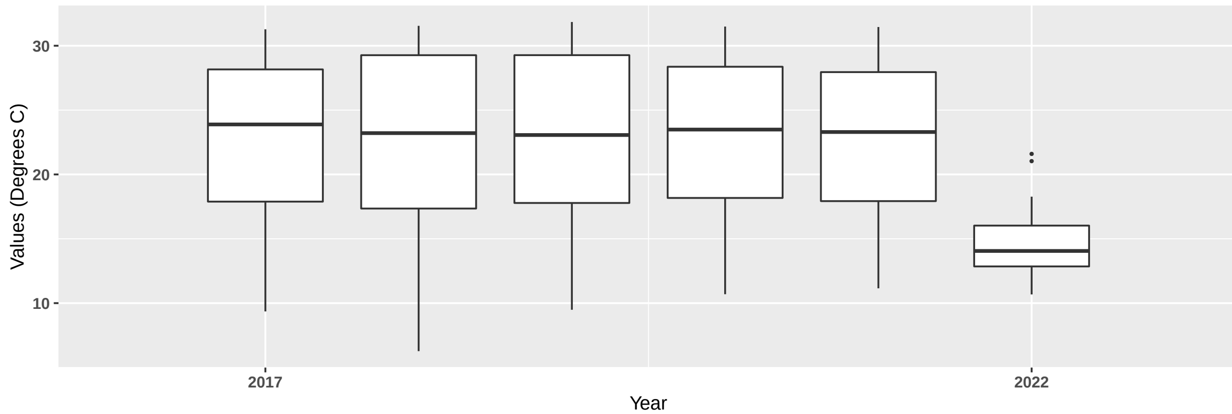


Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apalmwq
By Month

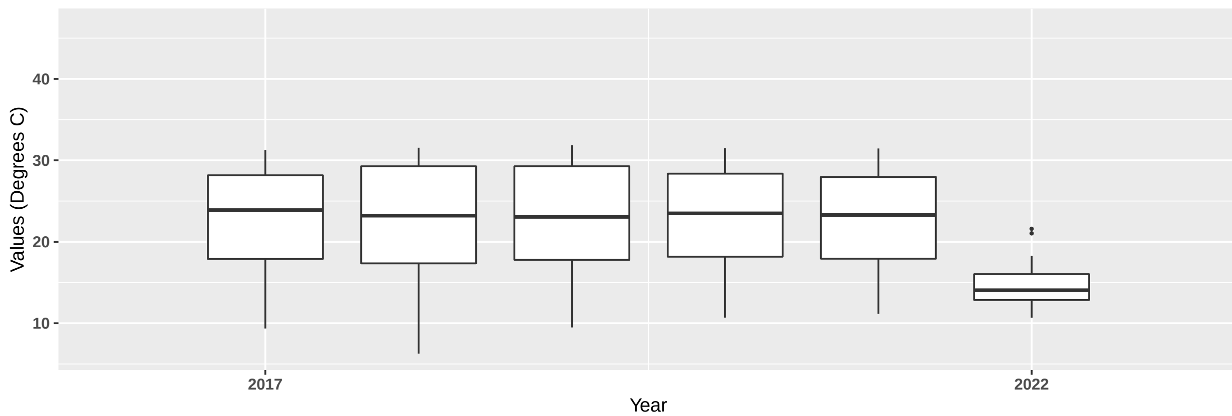


Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apapcwq
By Year

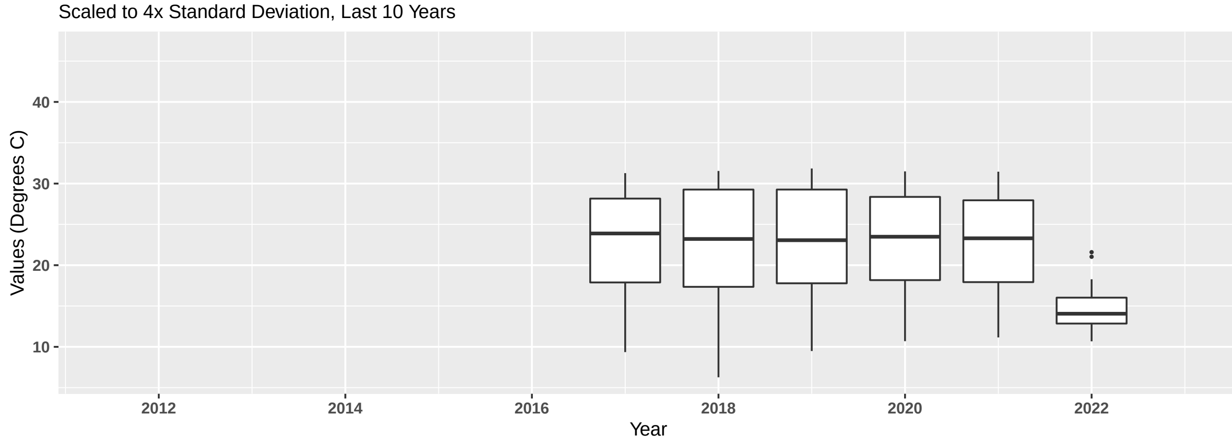
Autoscale



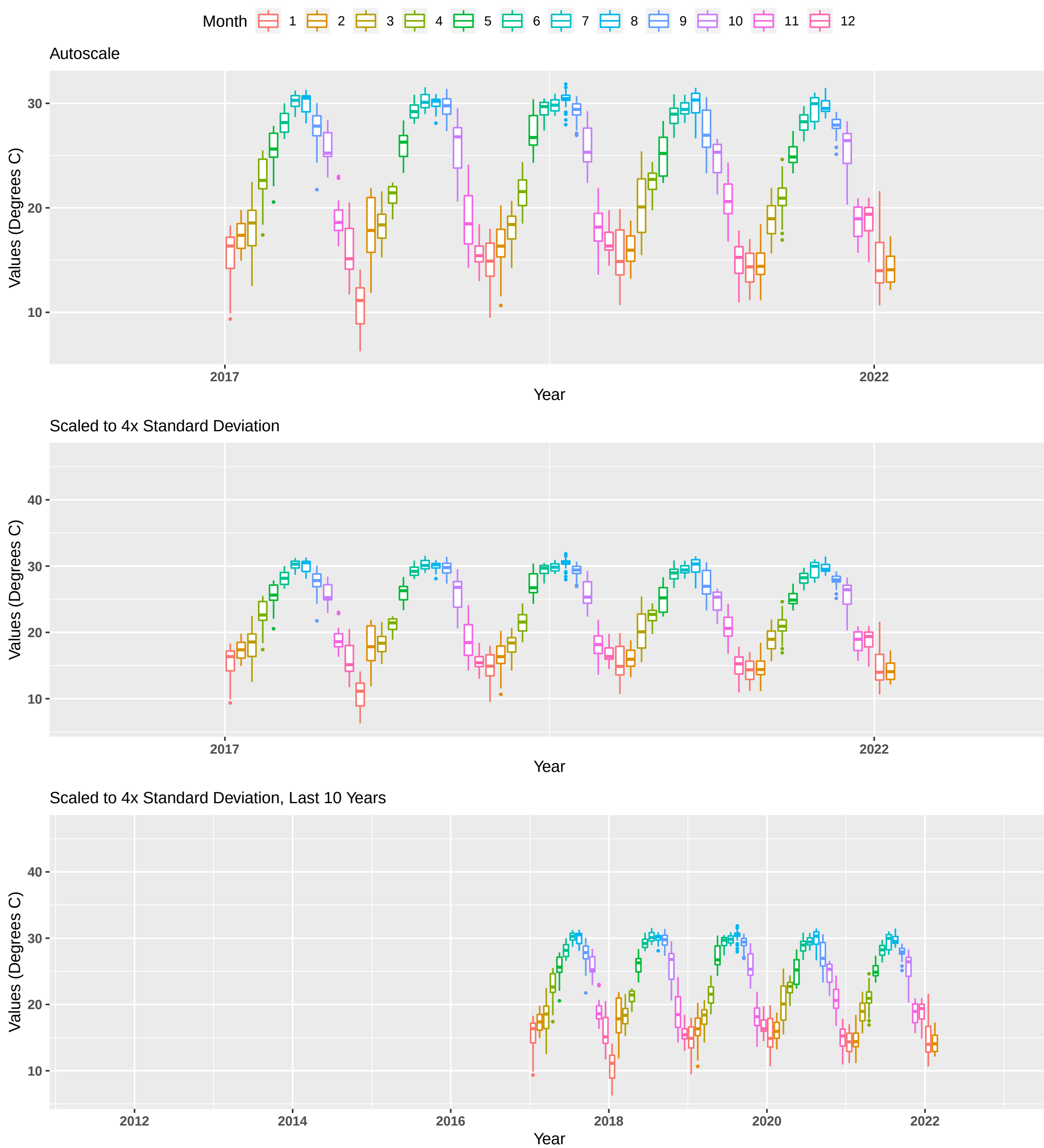
Scaled to 4x Standard Deviation



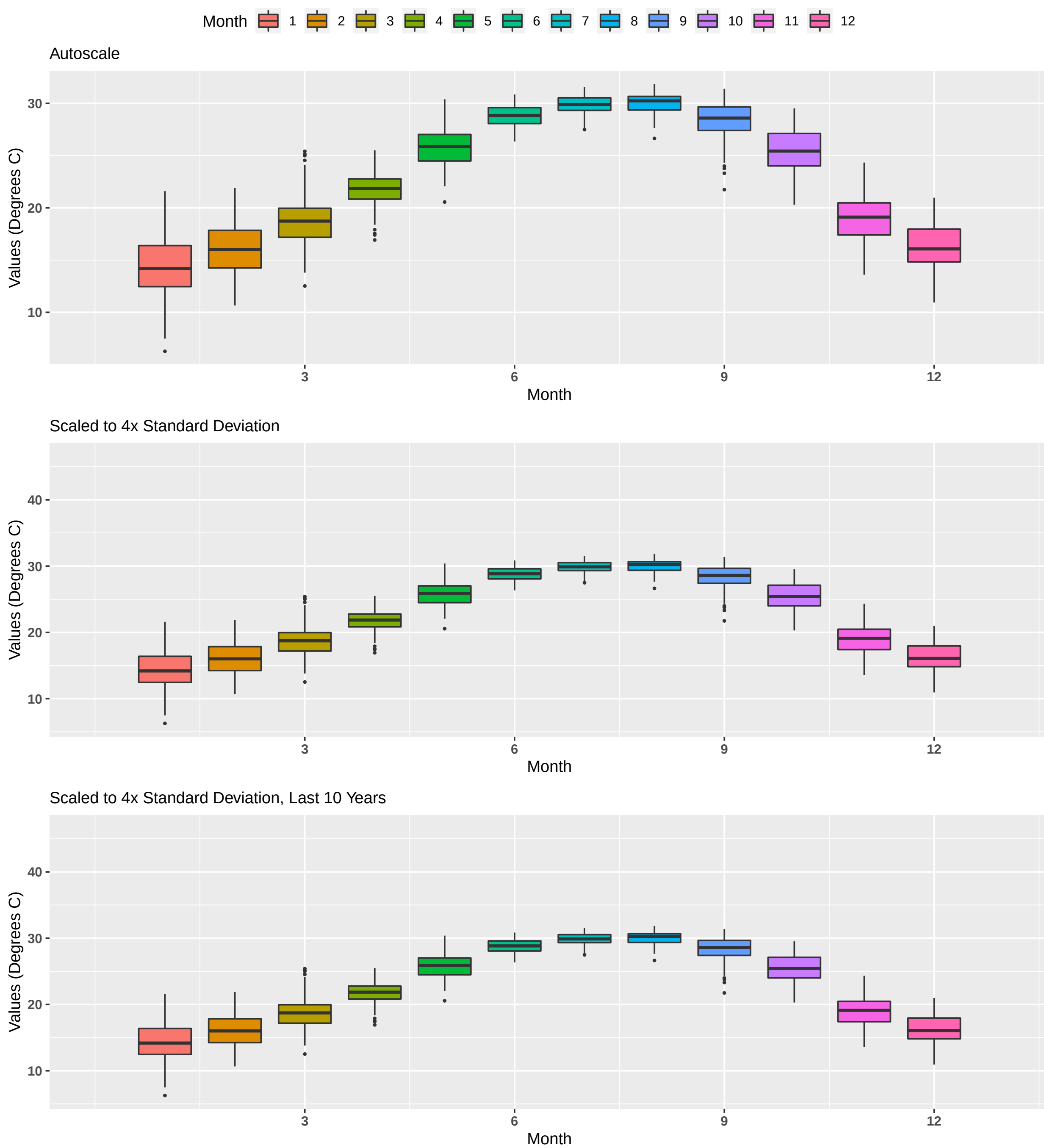
Scaled to 4x Standard Deviation, Last 10 Years



Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apapcwq
By Year & Month

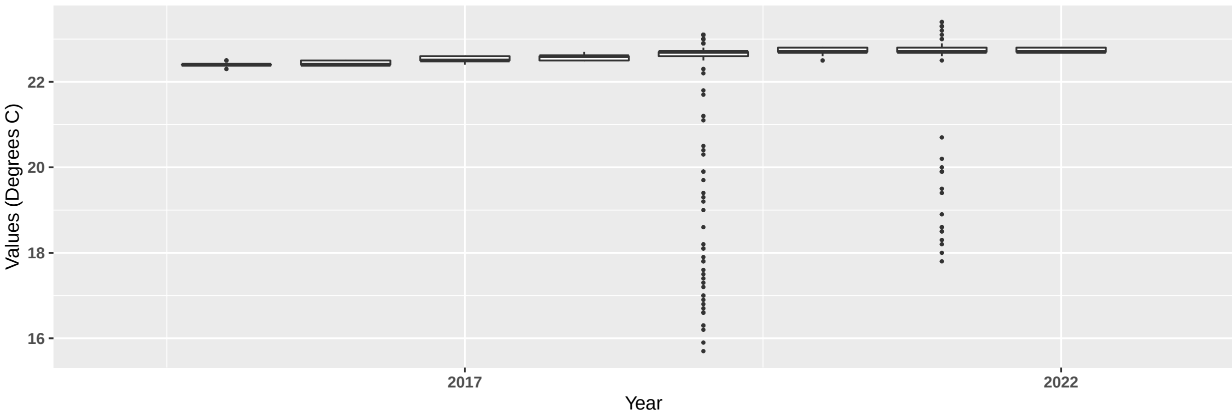


Summary Box Plots for Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System−Wide Monitoring Program | apapcwq
By Month

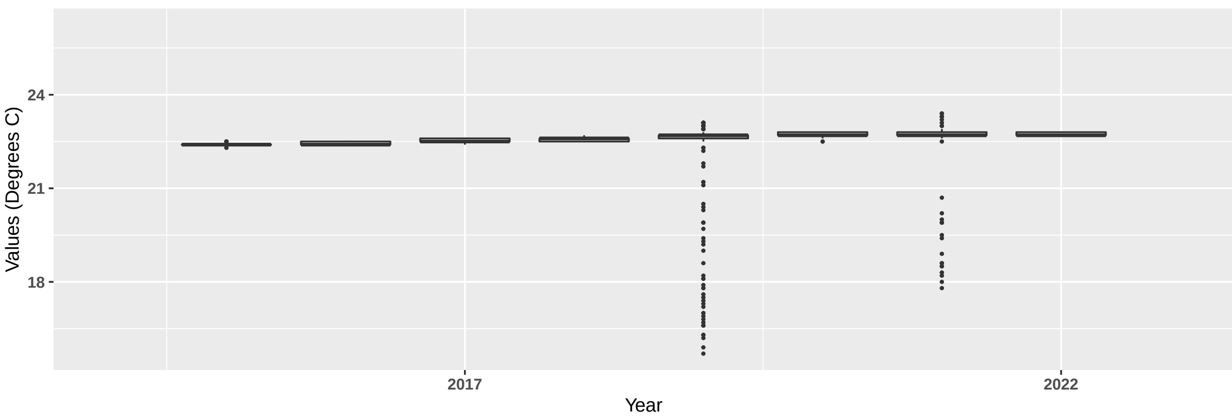


Summary Box Plots for Big Bend Seagrasses Aquatic Preserve
7 | National Water Information System | 02323566
By Year

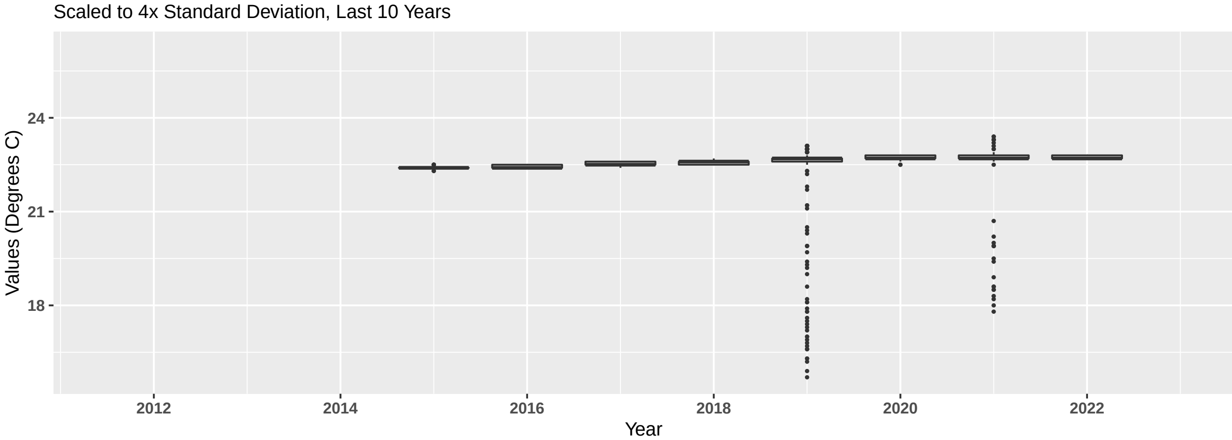
Autoscale



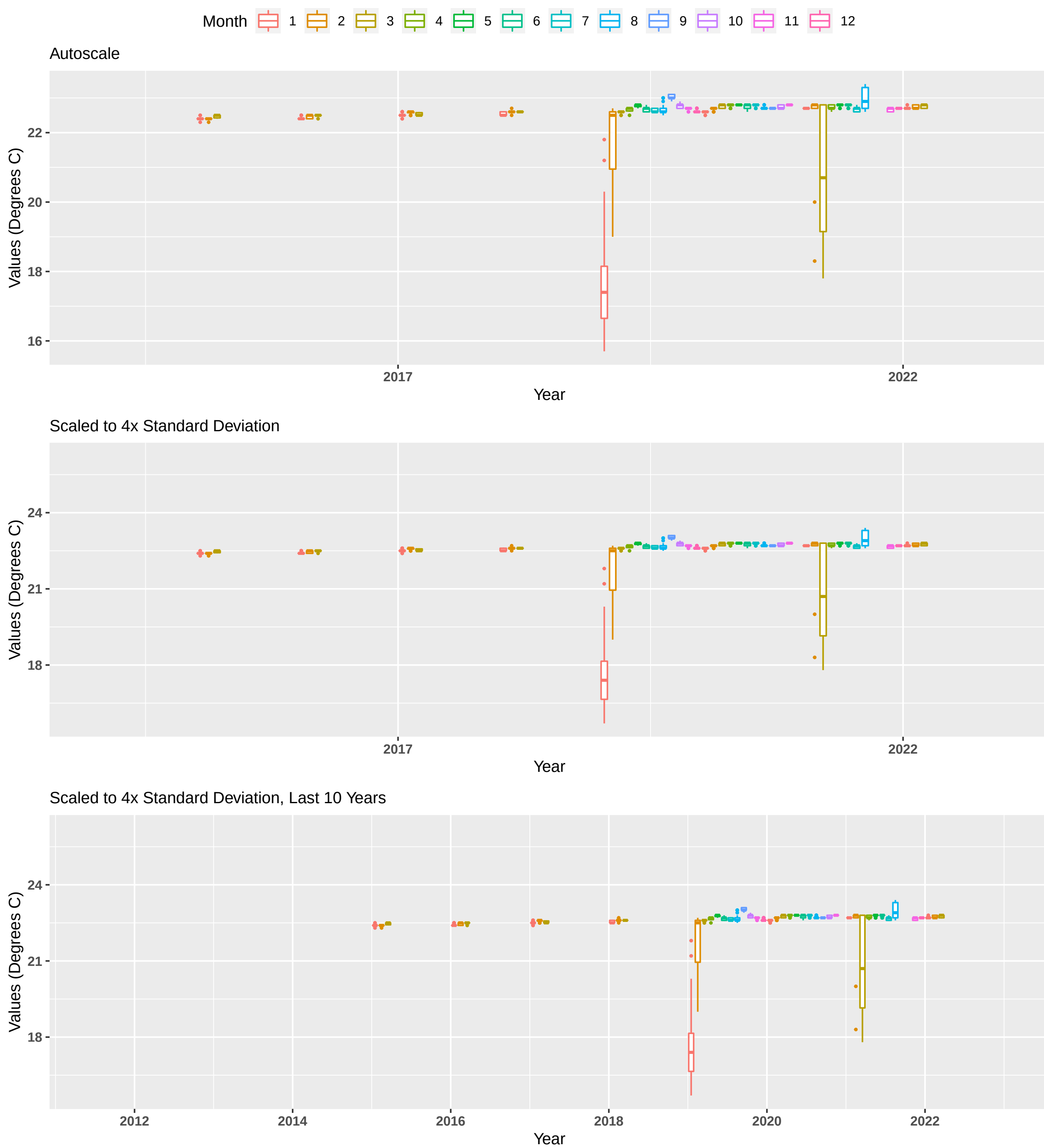
Scaled to 4x Standard Deviation



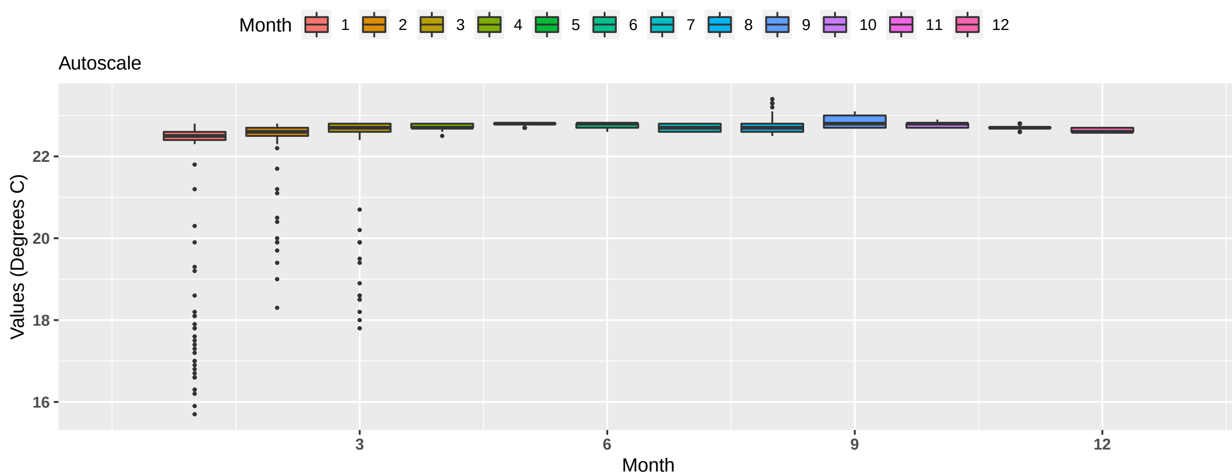
Scaled to 4x Standard Deviation, Last 10 Years

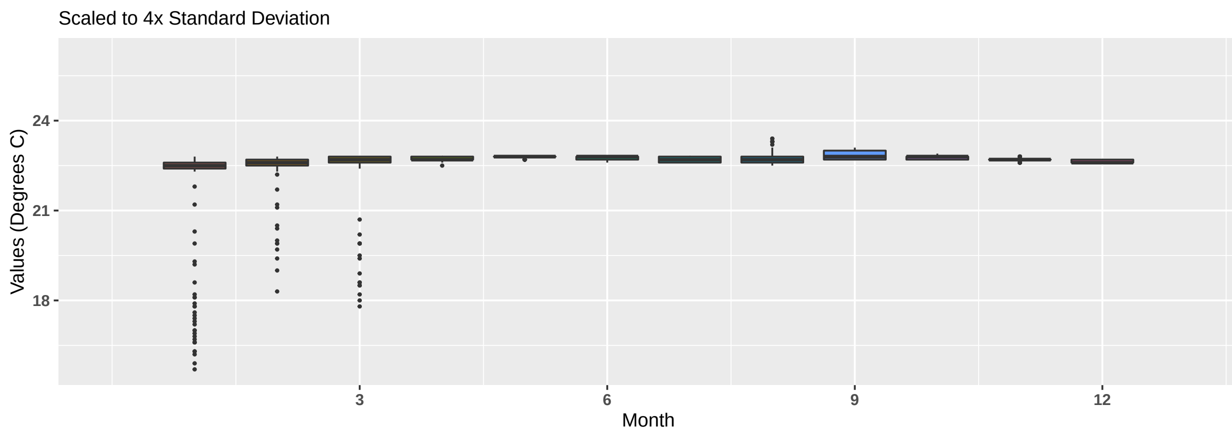


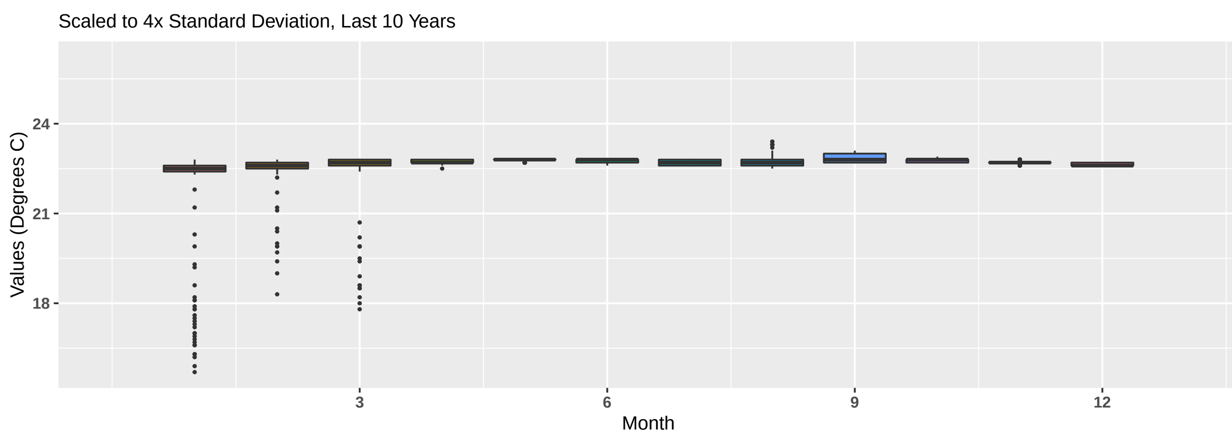
Summary Box Plots for Big Bend Seagrasses Aquatic Preserve
7 | National Water Information System | 02323566
By Year & Month



Summary Box Plots for Big Bend Seagrasses Aquatic Preserve
7 | National Water Information System | 02323566
By Month

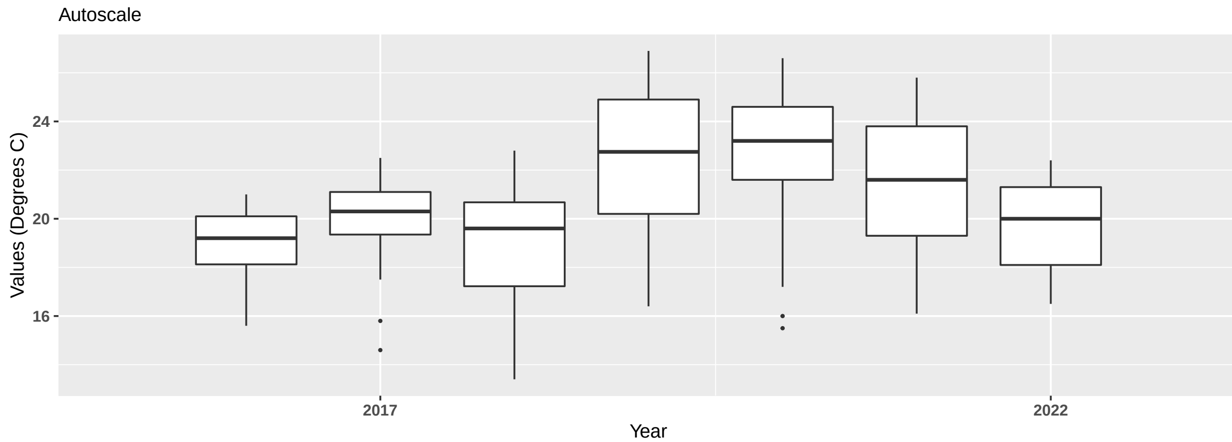


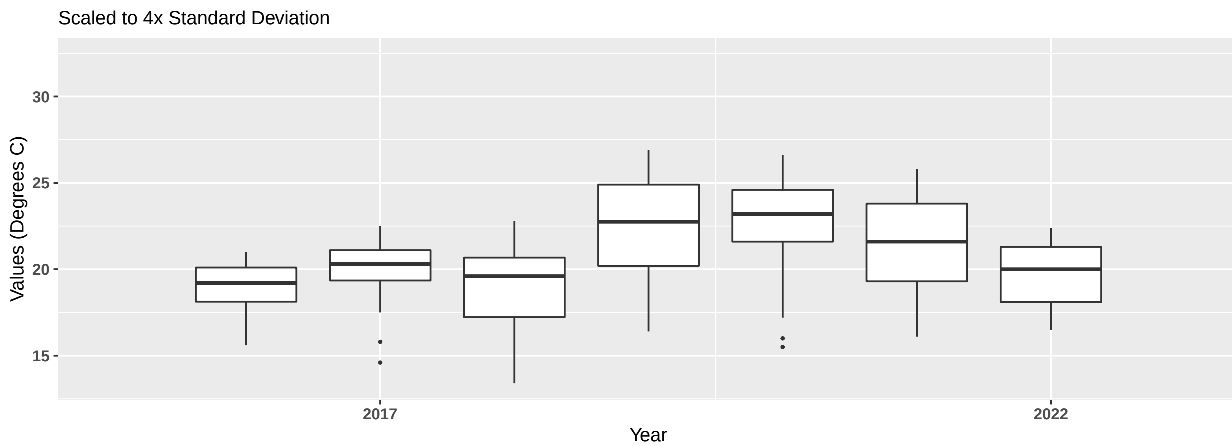


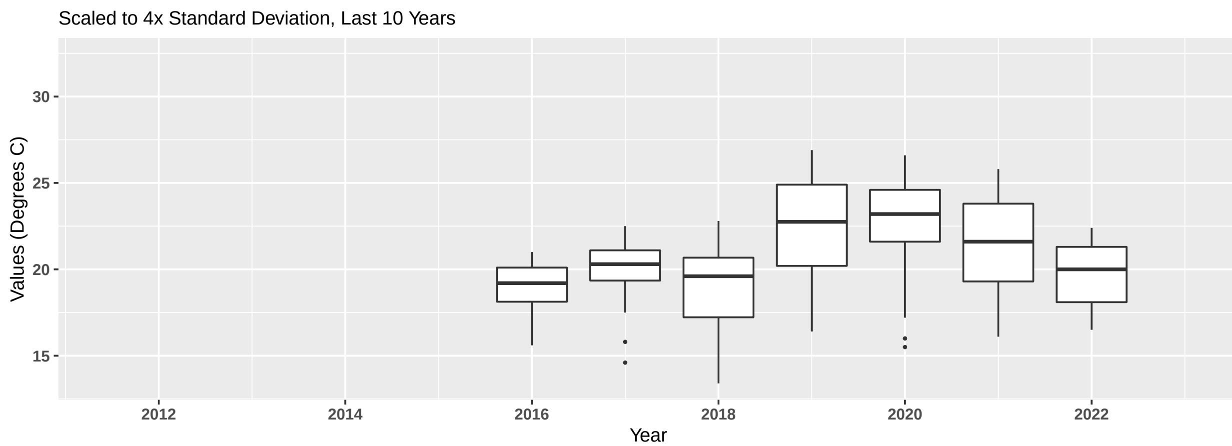


Summary Box Plots for Big Bend Seagrasses Aquatic Preserve 7 | National Water Information System | 02326526

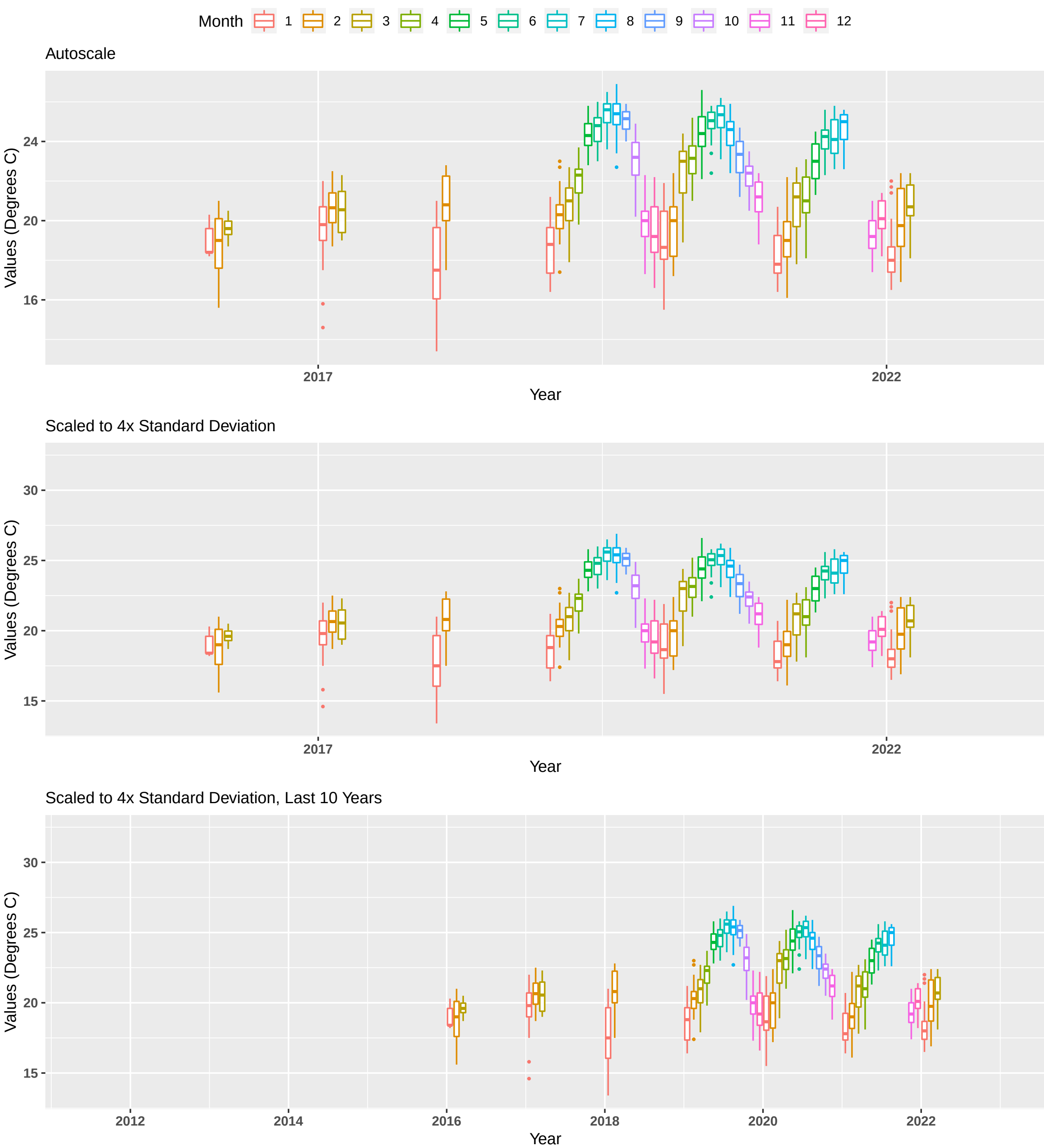
By Year



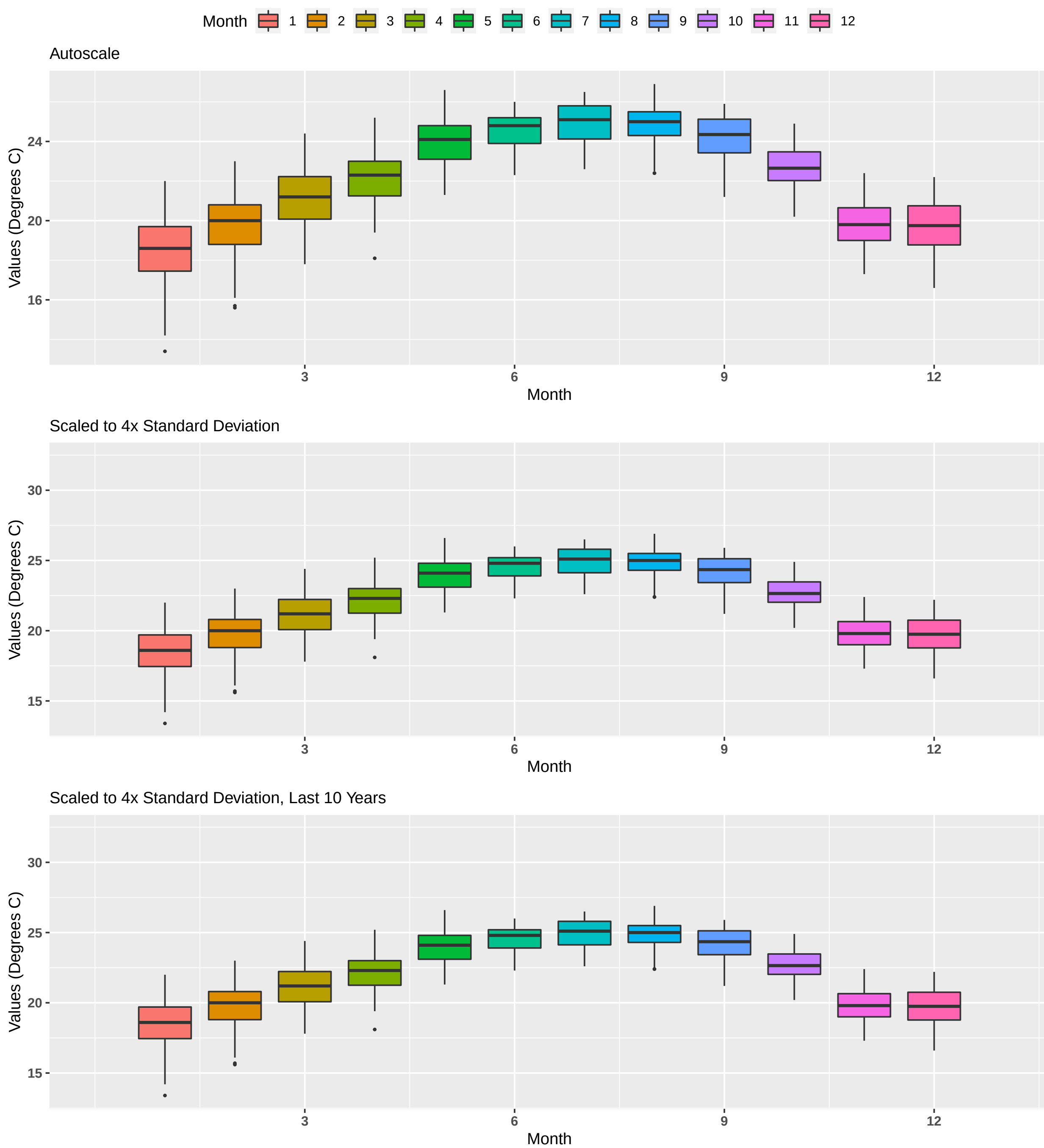




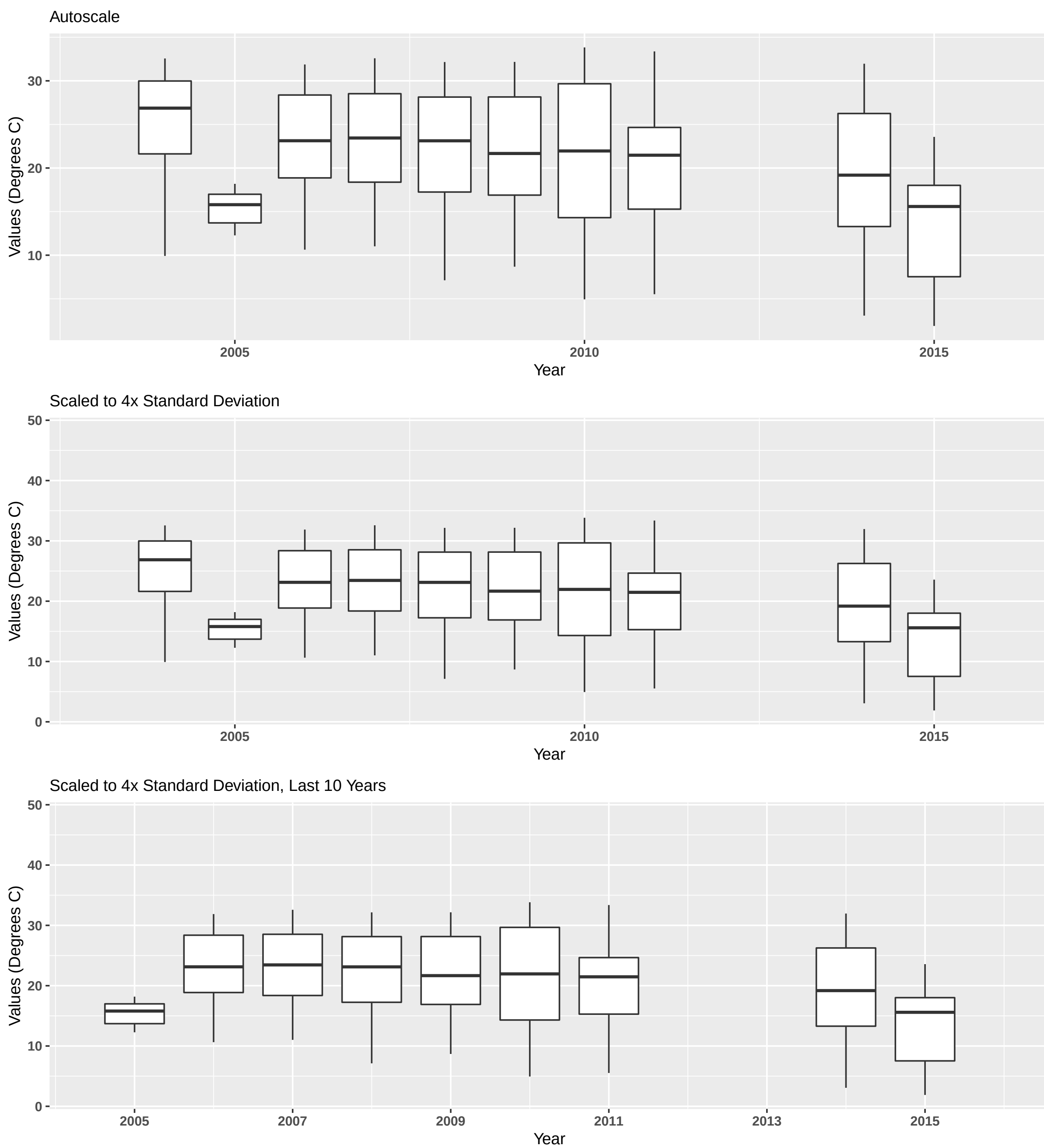
Summary Box Plots for Big Bend Seagrasses Aquatic Preserve
7 | National Water Information System | 02326526
By Year & Month



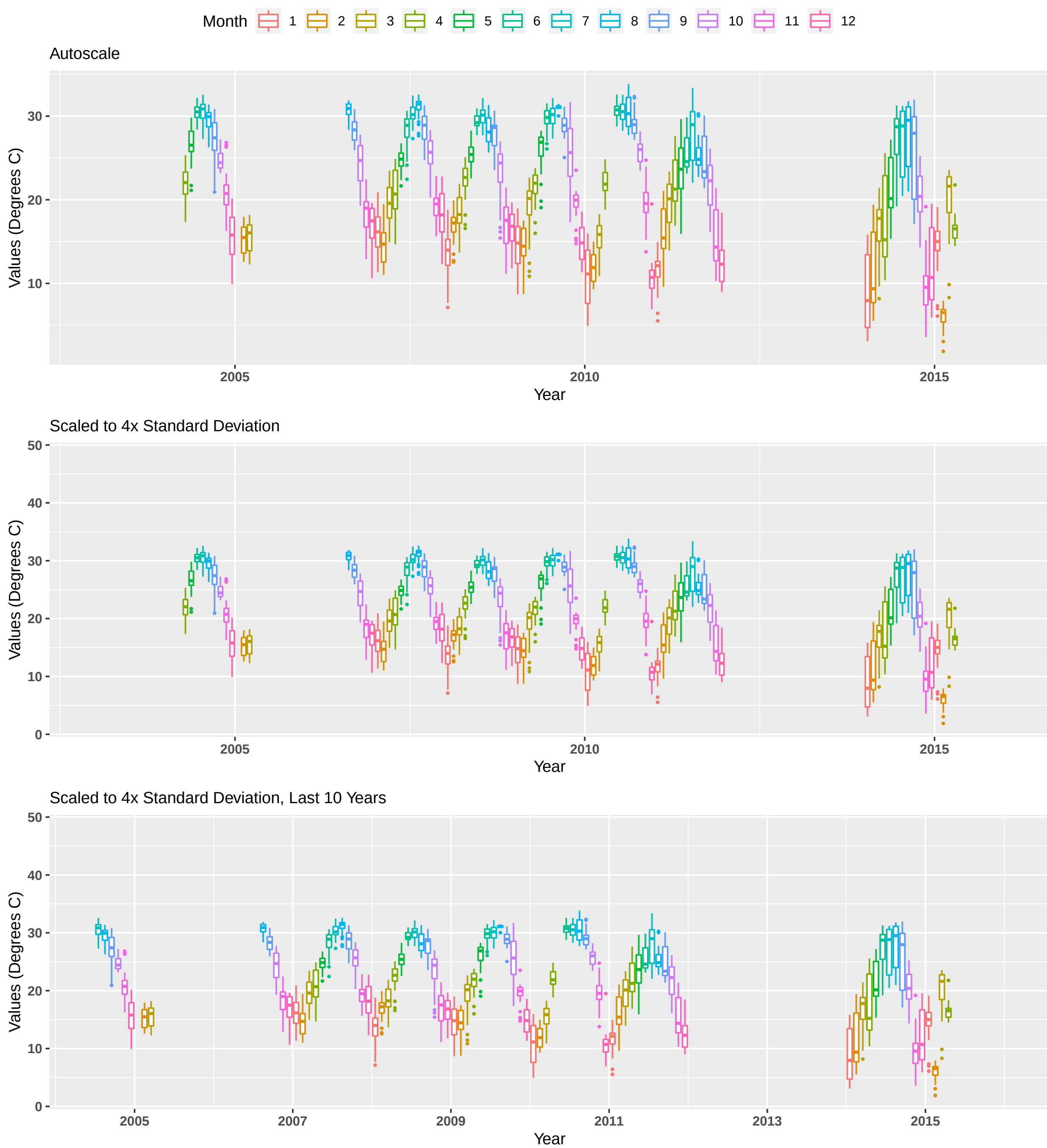
Summary Box Plots for Big Bend Seagrasses Aquatic Preserve
7 | National Water Information System | 02326526
By Month



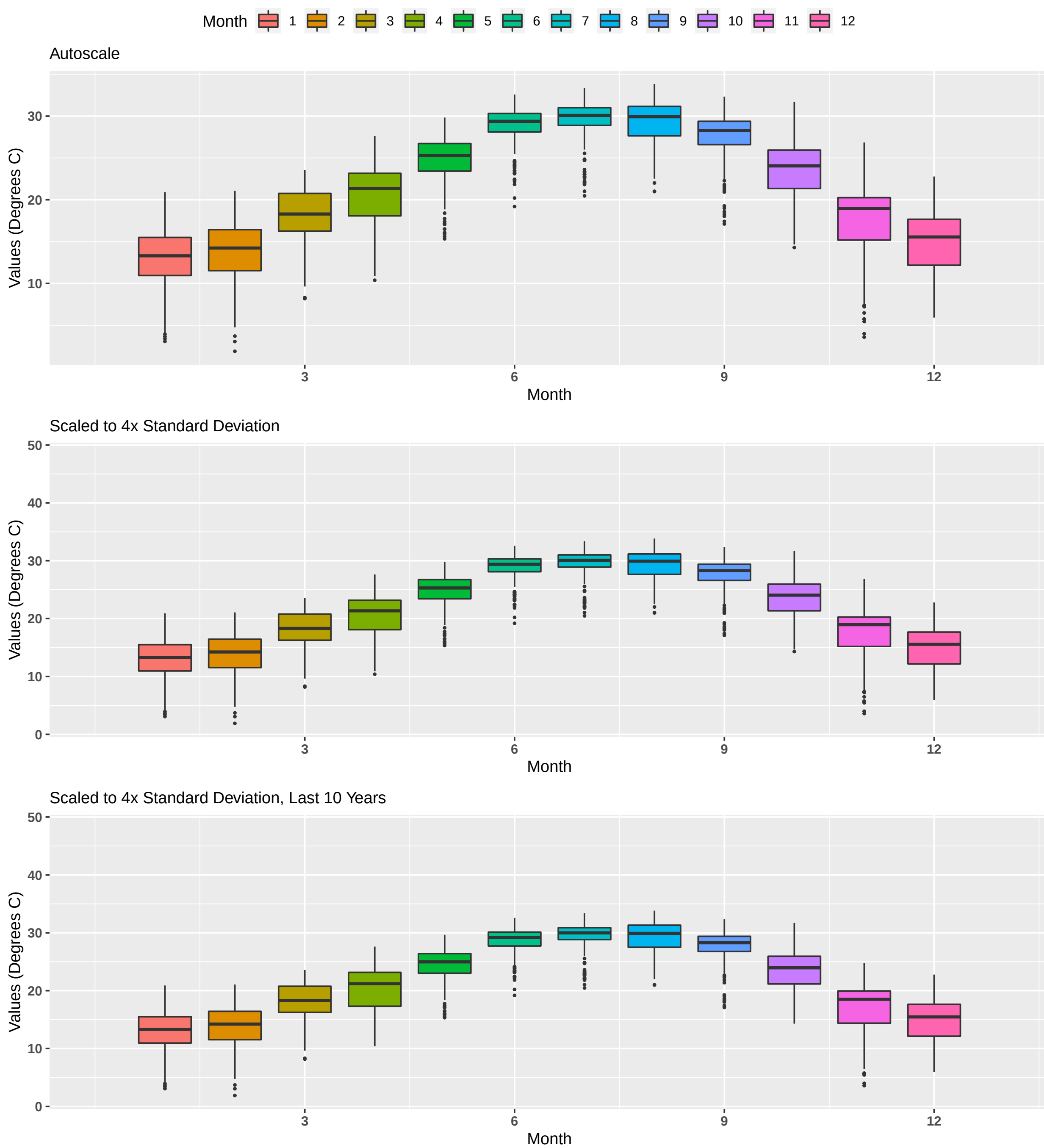
Summary Box Plots for Big Bend Seagrasses Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring | BBSSK
By Year



Summary Box Plots for Big Bend Seagrasses Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring | BBSSK
By Year & Month

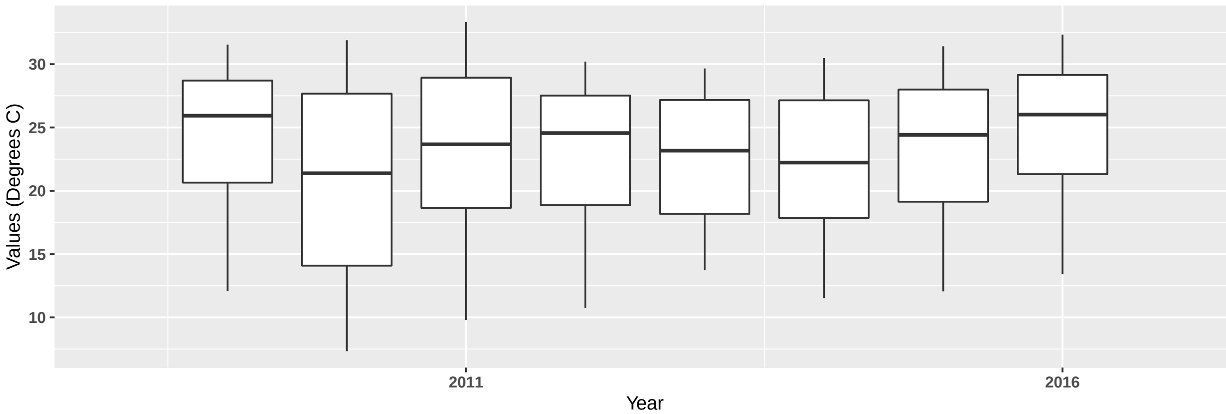


Summary Box Plots for Big Bend Seagrasses Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring | BBSSK
By Month

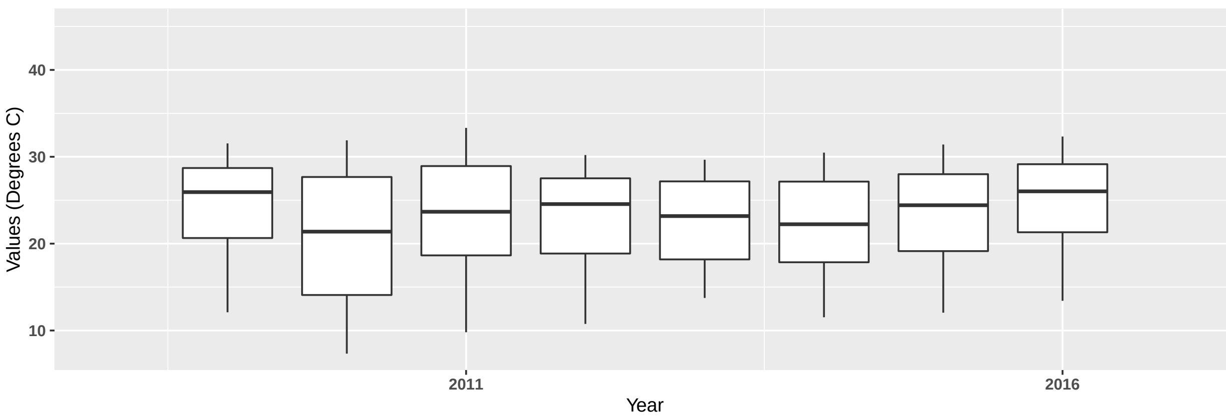


Summary Box Plots for Big Bend Seagrasses Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring | BBSSW
By Year

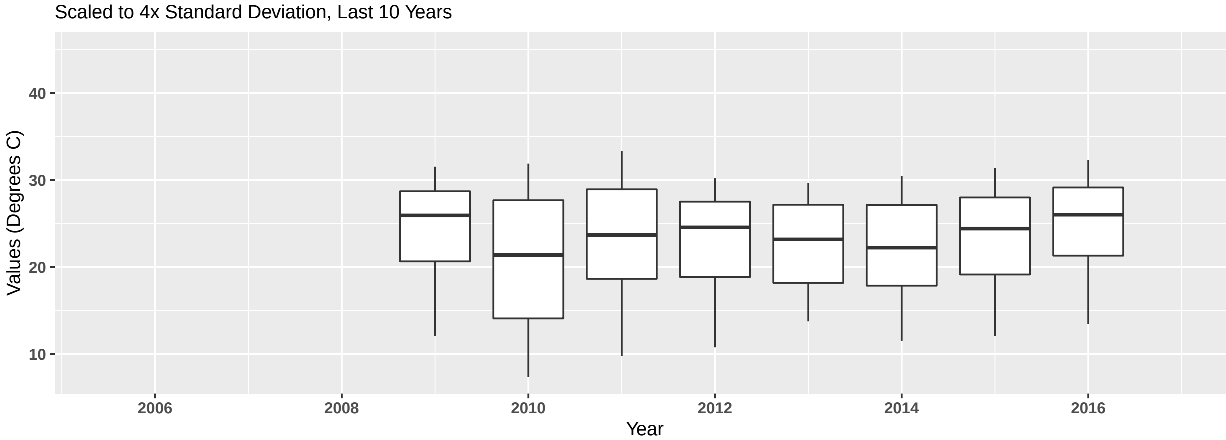
Autoscale



Scaled to 4x Standard Deviation

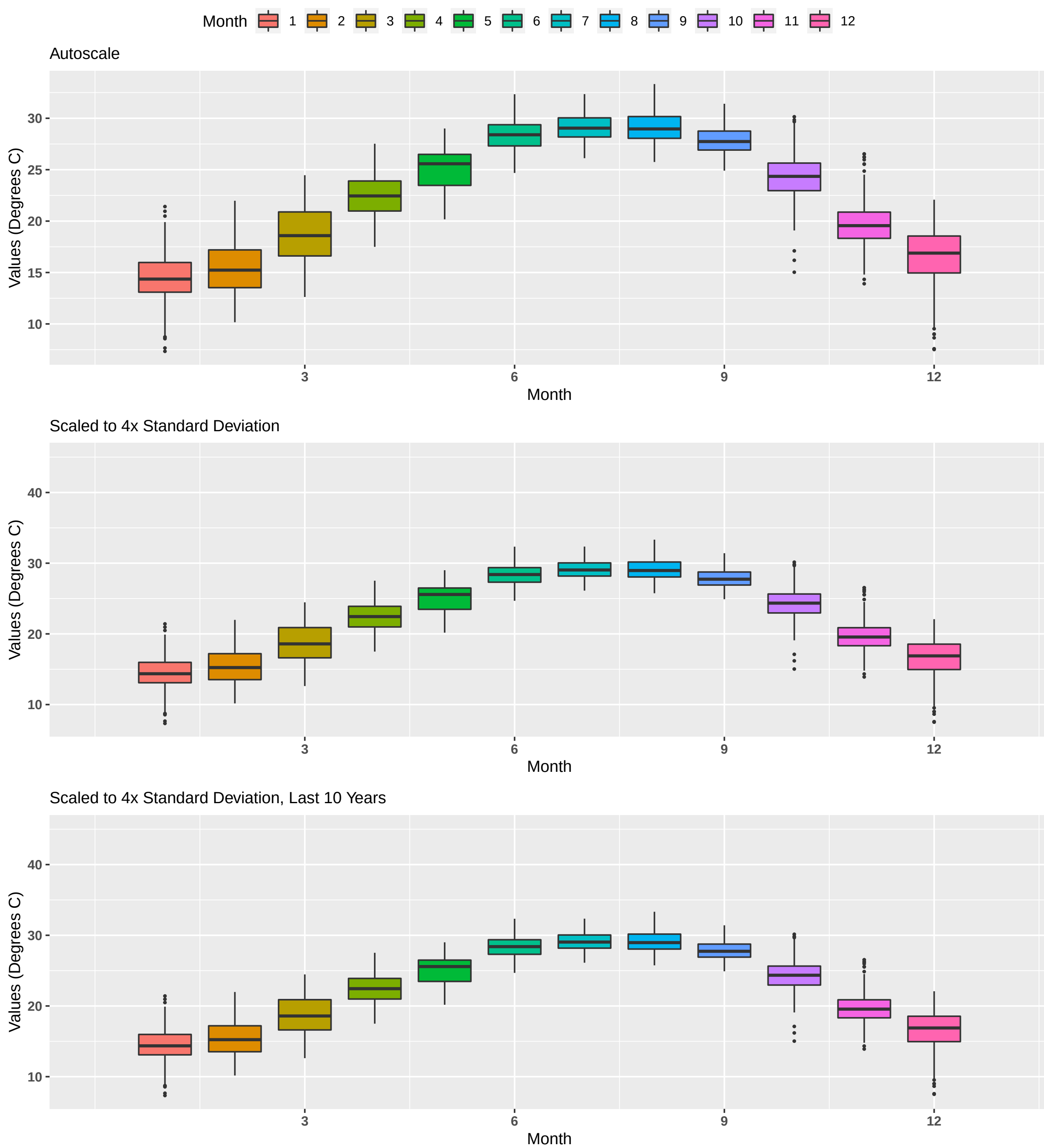


Scaled to 4x Standard Deviation, Last 10 Years

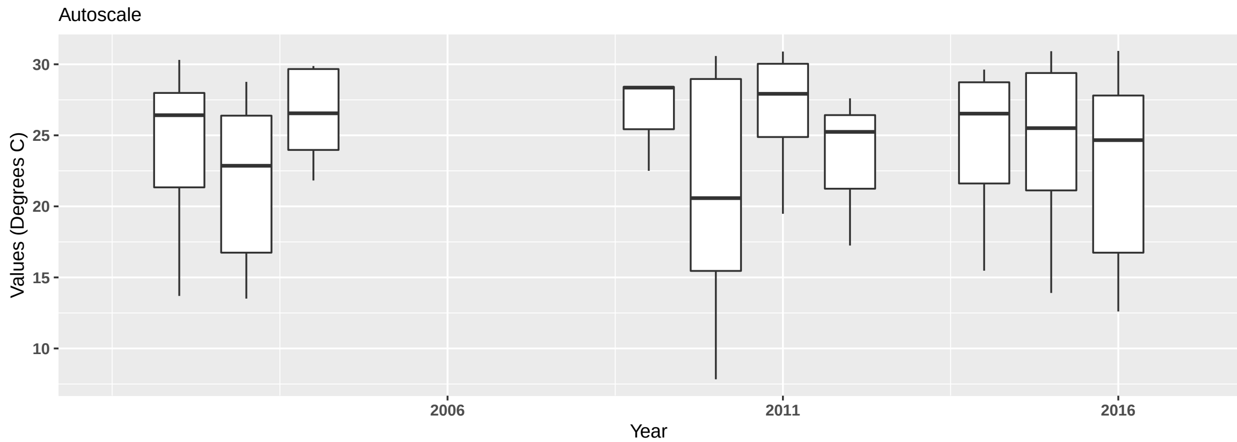


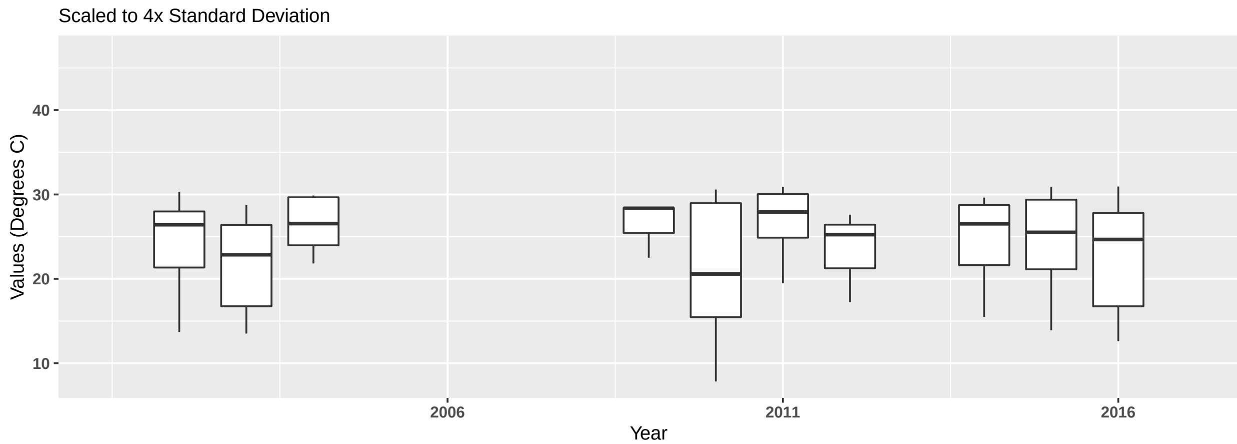
Summary Box Plots for Big Bend Seagrasses Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring | BBSSW
By Year & Month

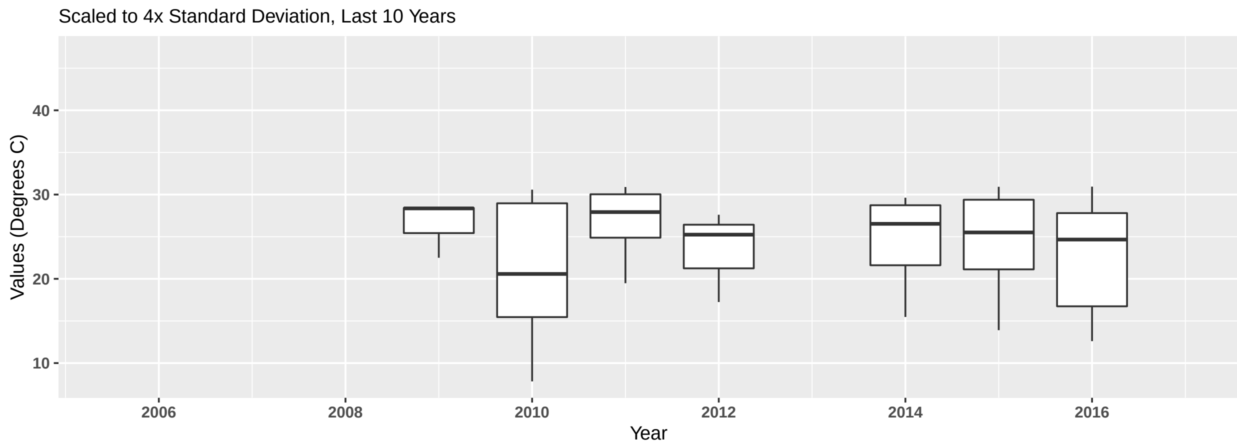
Summary Box Plots for Big Bend Seagrasses Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring | BBSSW
By Month



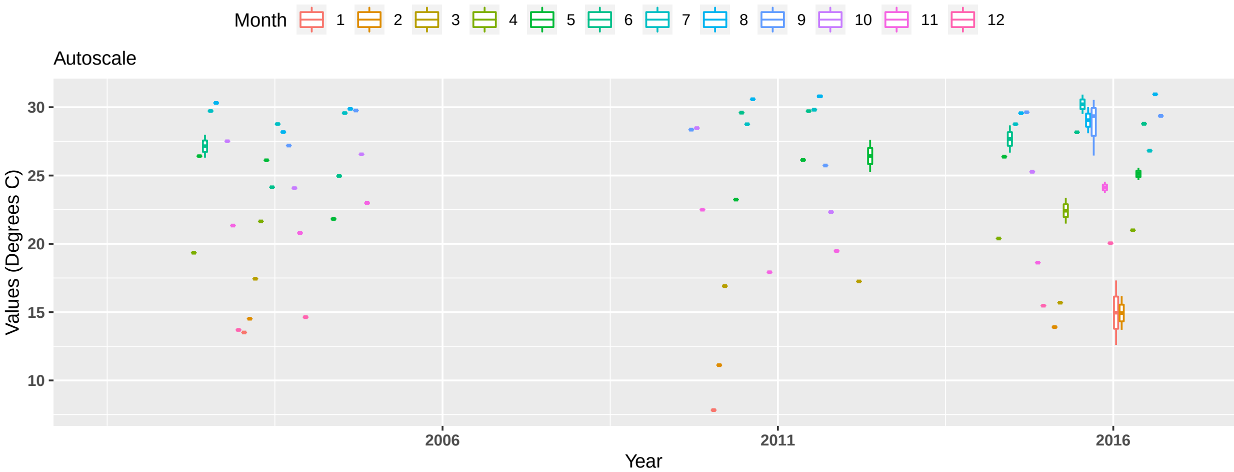
Summary Box Plots for Fort Pickens State Park Aquatic Preserve
505 | Pensacola Bay Water Quality Monitoring Program | P09
By Year

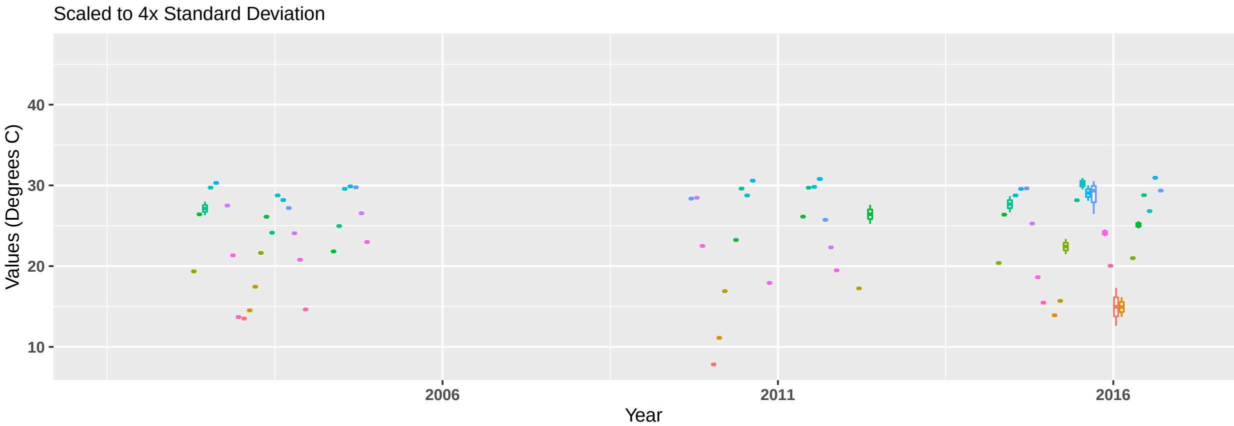


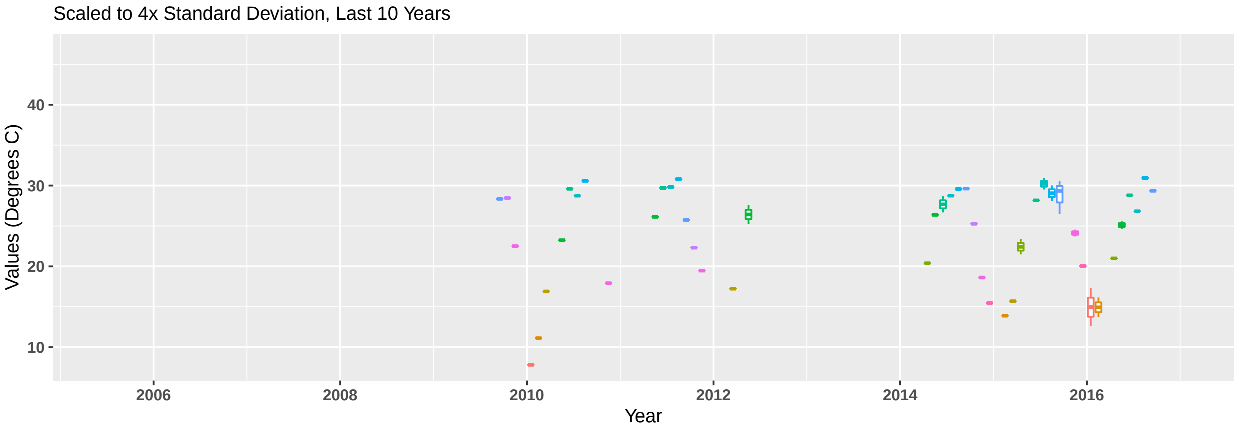




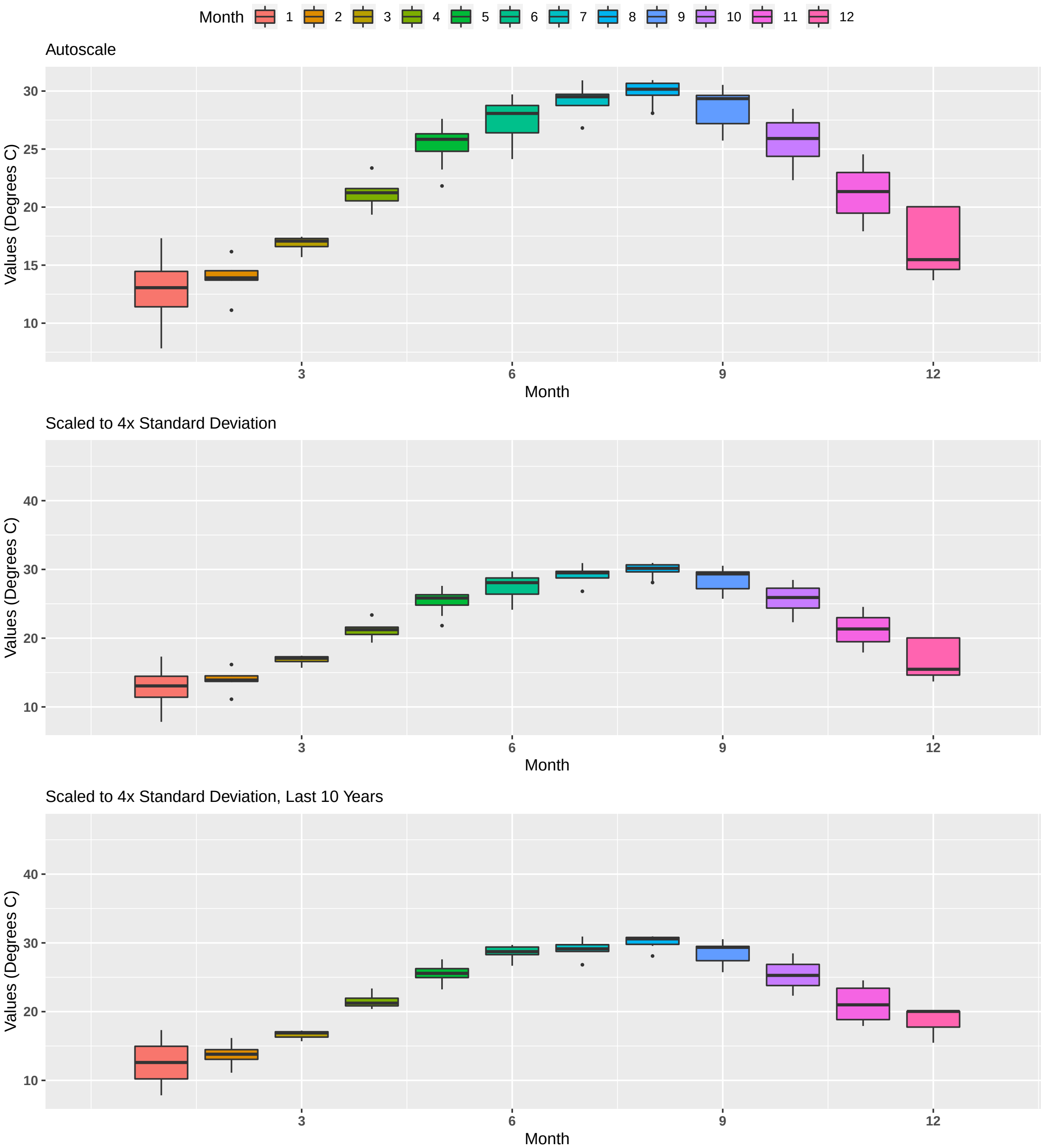
Summary Box Plots for Fort Pickens State Park Aquatic Preserve
505 | Pensacola Bay Water Quality Monitoring Program | P09
By Year & Month





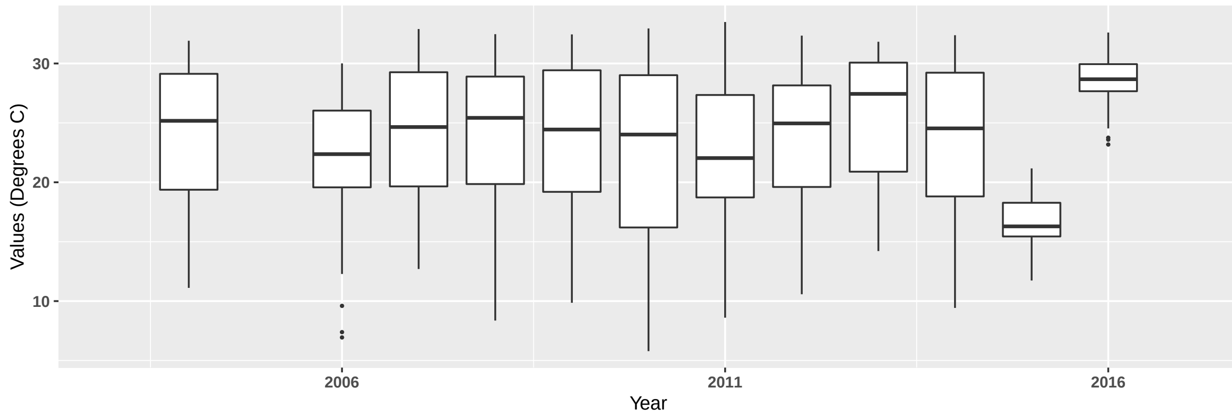


Summary Box Plots for Fort Pickens State Park Aquatic Preserve
505 | Pensacola Bay Water Quality Monitoring Program | P09
By Month

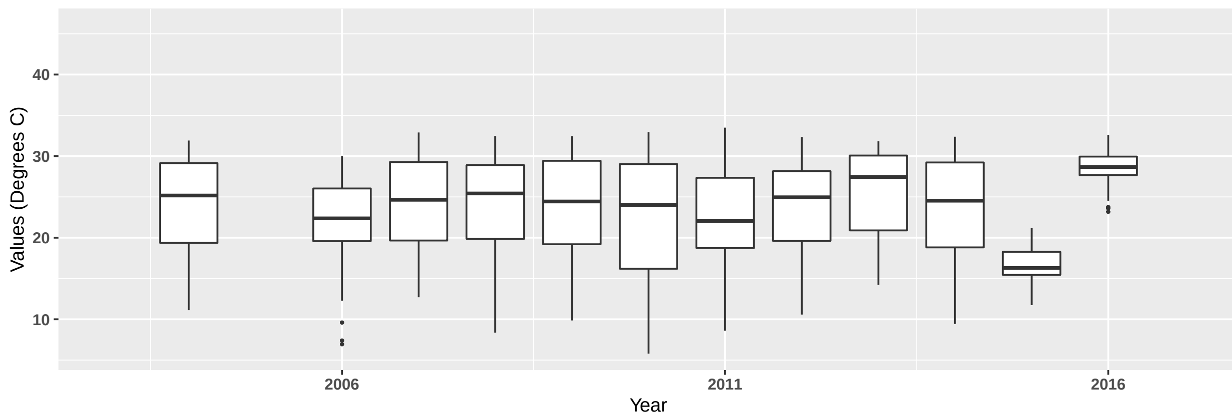


Summary Box Plots for Nature Coast Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring | BBSHS
By Year

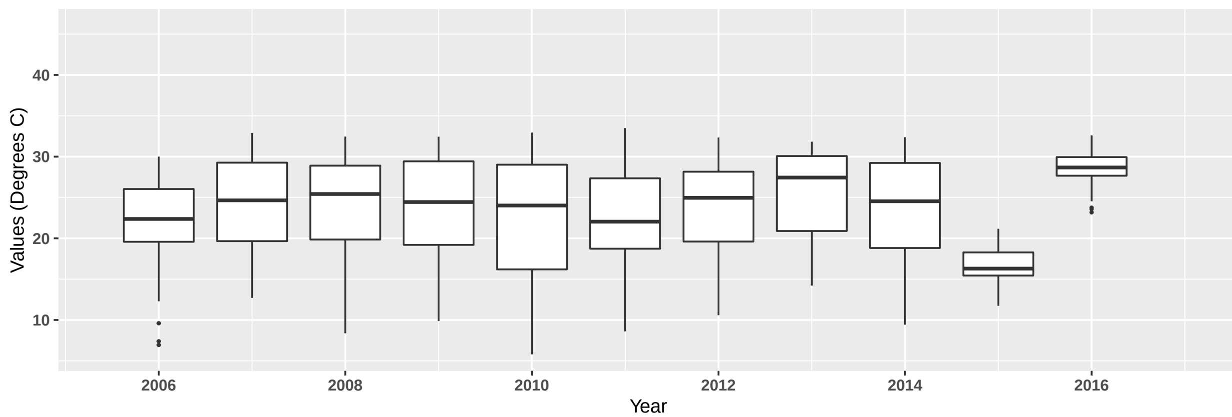
Autoscale



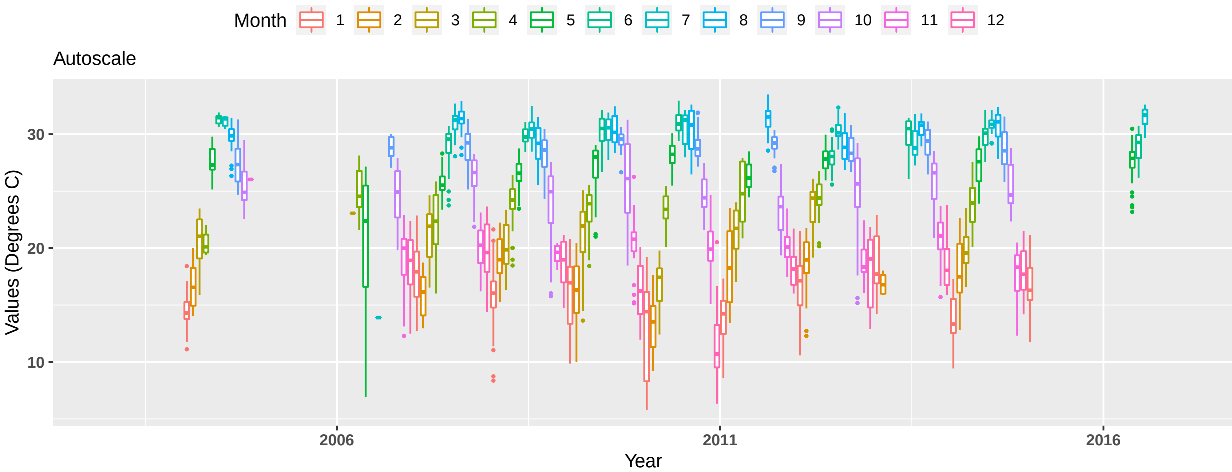
Scaled to 4x Standard Deviation

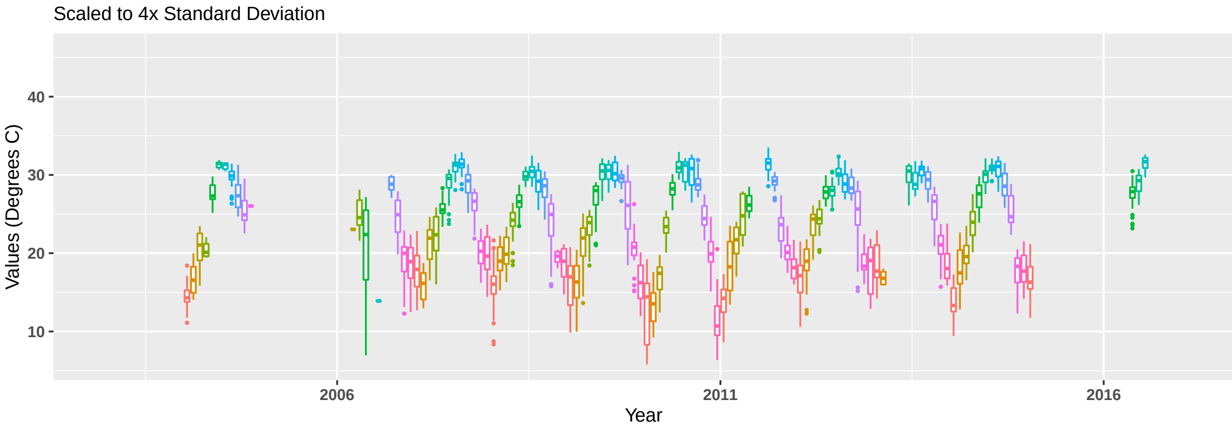


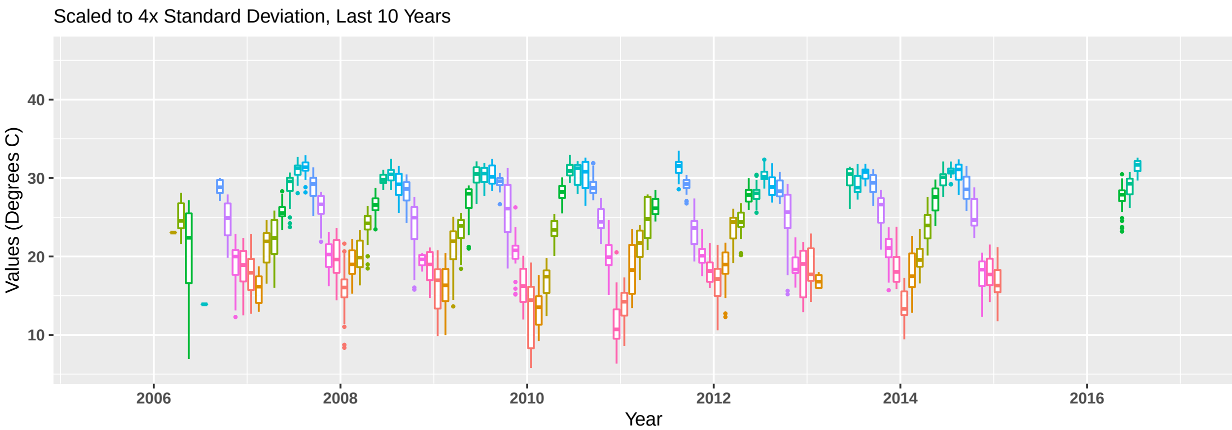
Scaled to 4x Standard Deviation, Last 10 Years



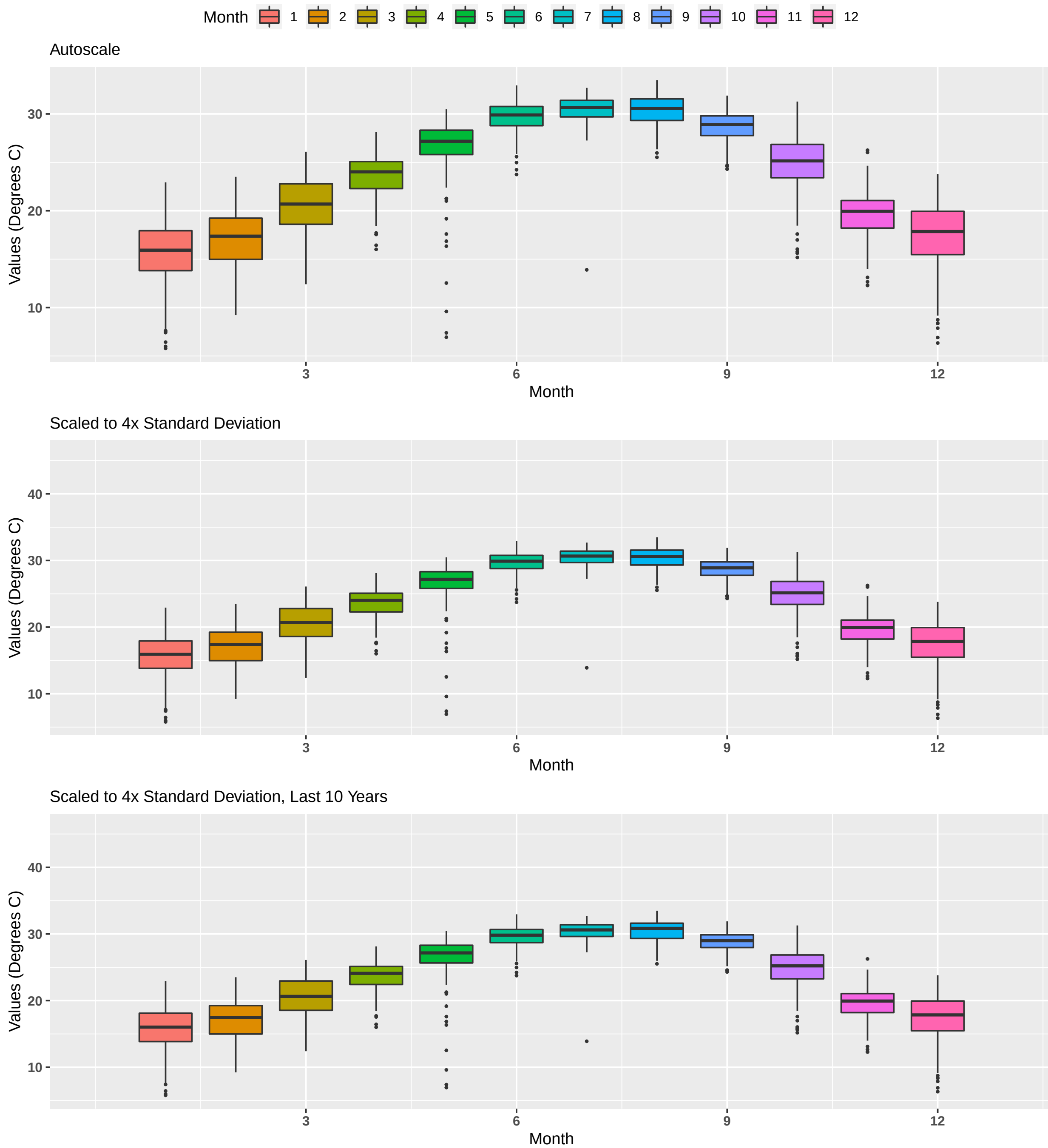
Summary Box Plots for Nature Coast Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring | BBSHS
By Year & Month



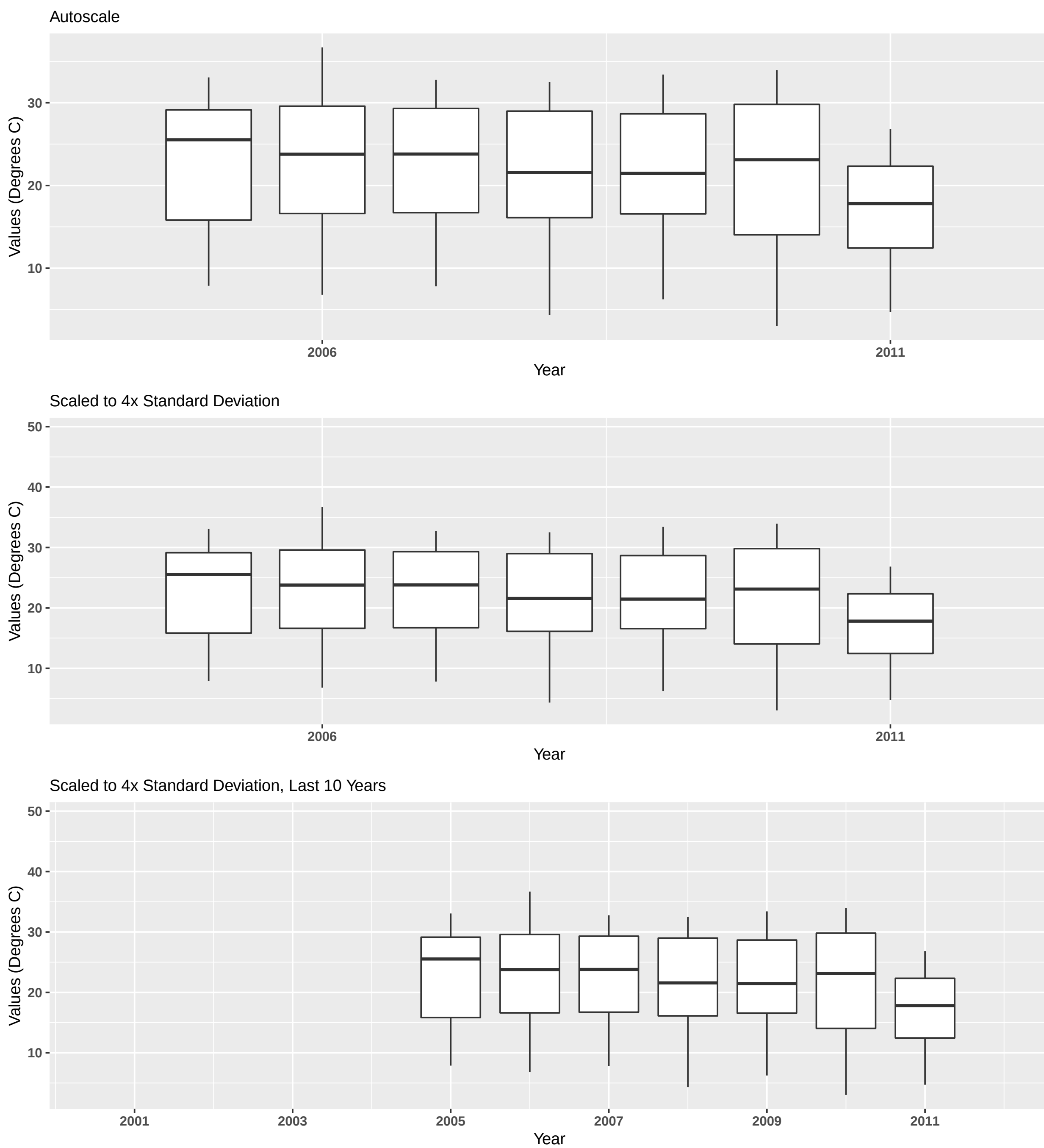




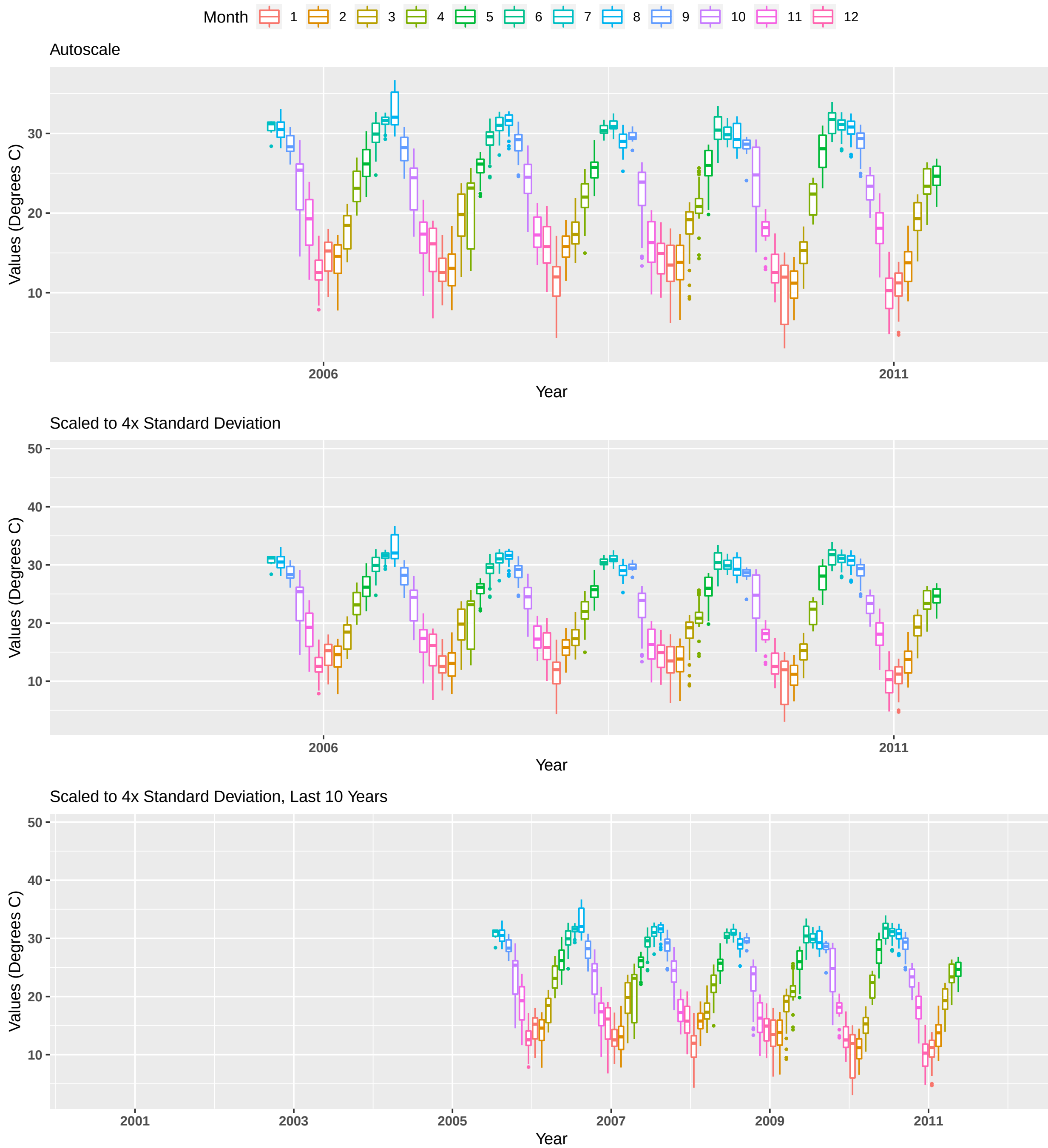
Summary Box Plots for Nature Coast Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring | BBSHS
By Month



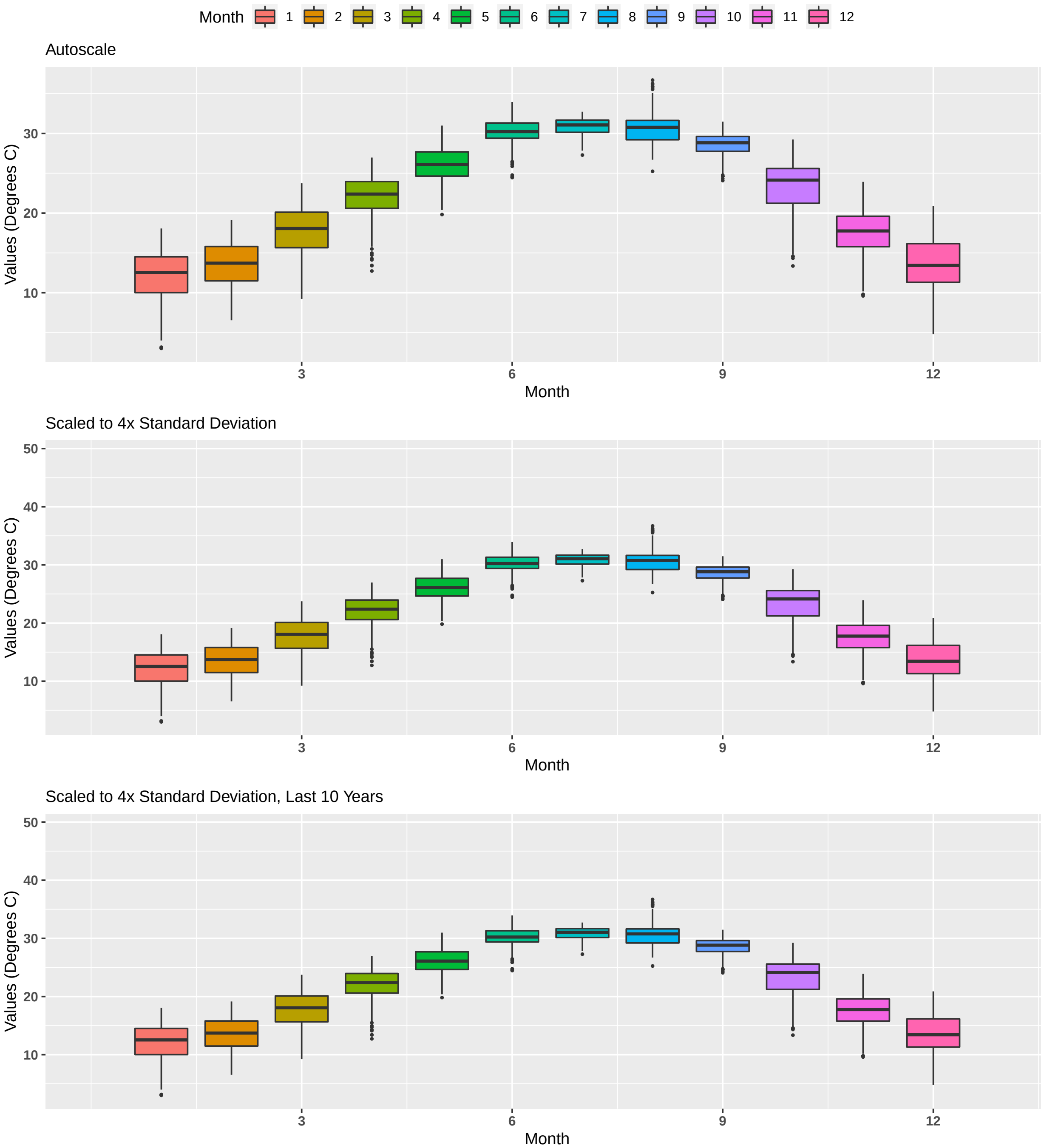
Summary Box Plots for St. Joseph Bay Aquatic Preserve
468 | Central Panhandle Aquatic Preserves Continuous Water Quality Monitoring | CPRH
By Year



Summary Box Plots for St. Joseph Bay Aquatic Preserve
468 | Central Panhandle Aquatic Preserves Continuous Water Quality Monitoring | CPRH
By Year & Month

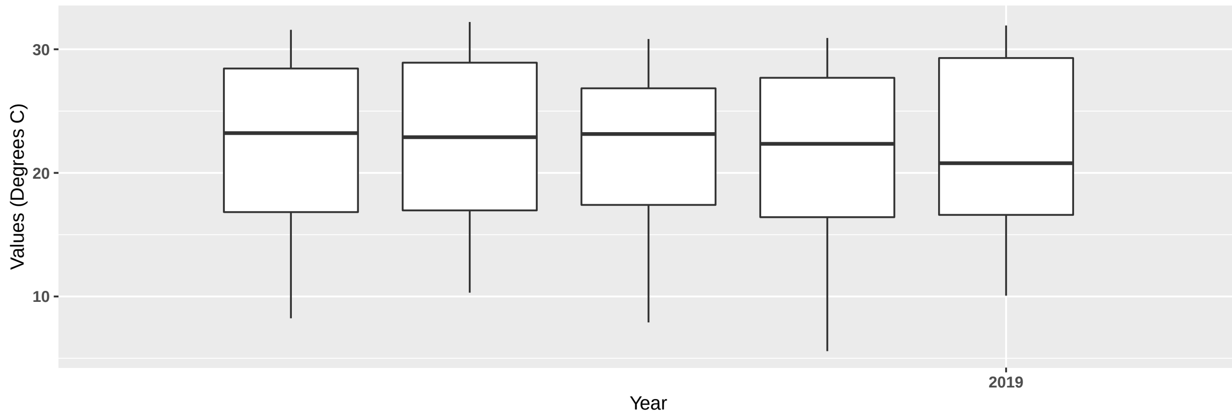


Summary Box Plots for St. Joseph Bay Aquatic Preserve
468 | Central Panhandle Aquatic Preserves Continuous Water Quality Monitoring | CPRH
By Month

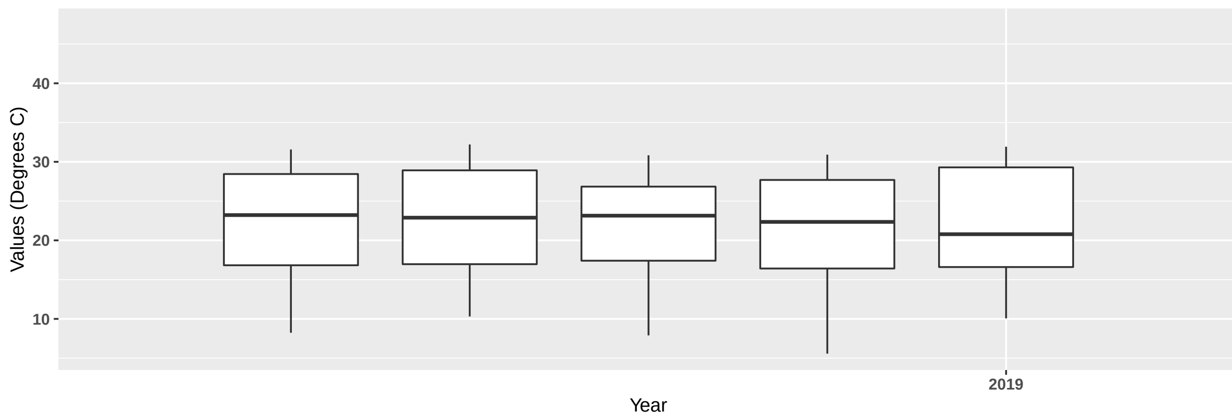


Summary Box Plots for Yellow River Marsh Aquatic Preserve
467 | Yellow River Marsh Aquatic Preserve Continuous Water Quality Monitoring | YRMAP1
By Year

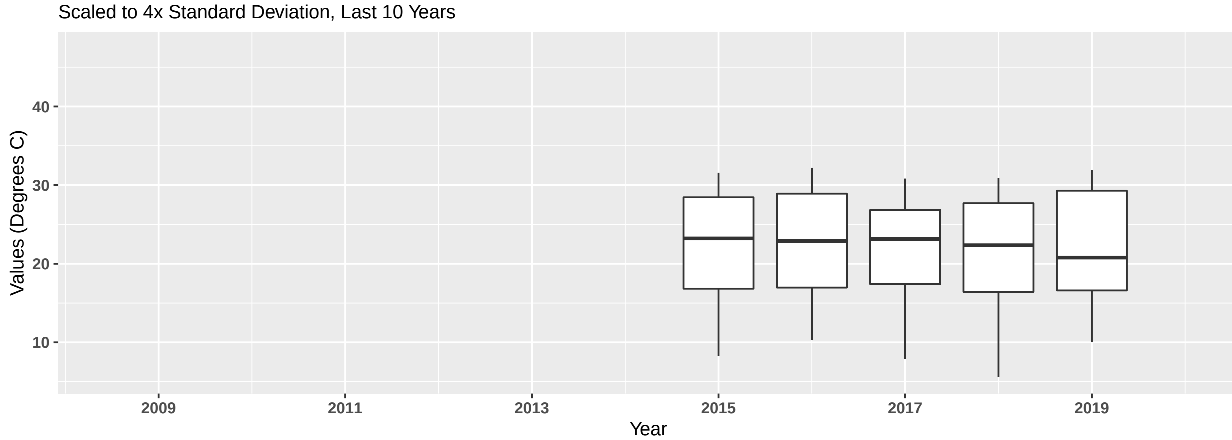
Autoscale



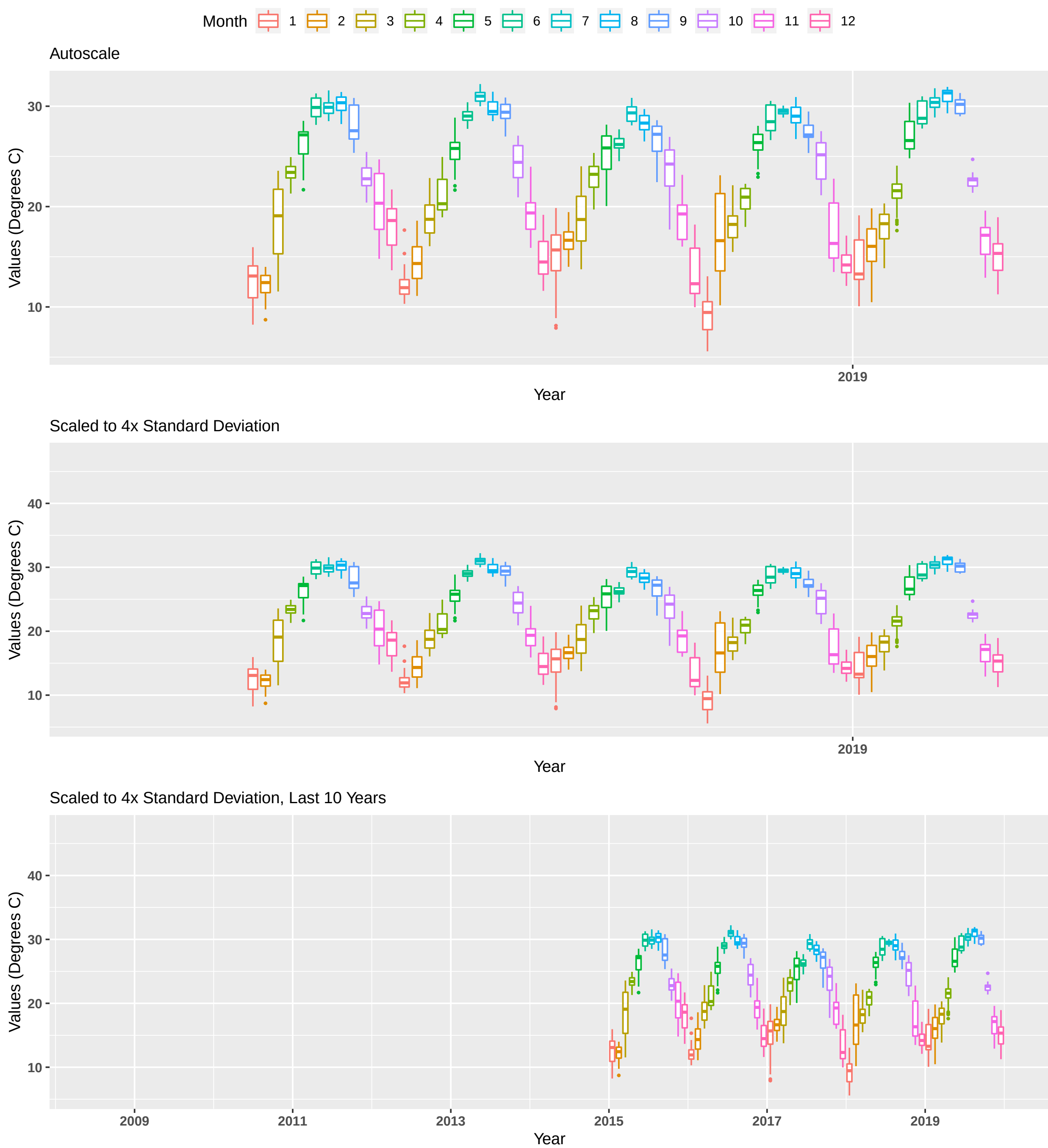
Scaled to 4x Standard Deviation



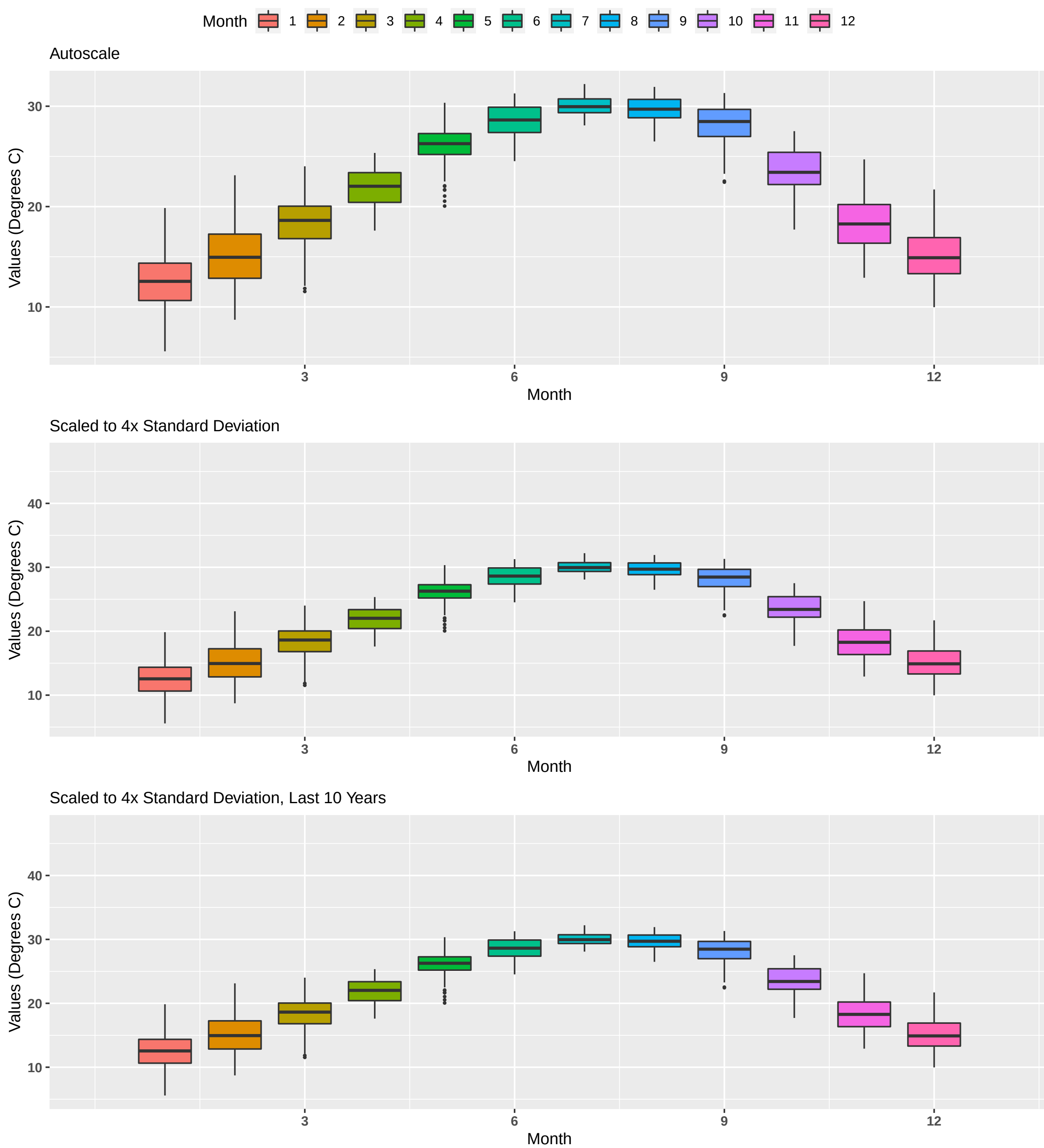
Scaled to 4x Standard Deviation, Last 10 Years



Summary Box Plots for Yellow River Marsh Aquatic Preserve
467 | Yellow River Marsh Aquatic Preserve Continuous Water Quality Monitoring | YRMAP1
By Year & Month

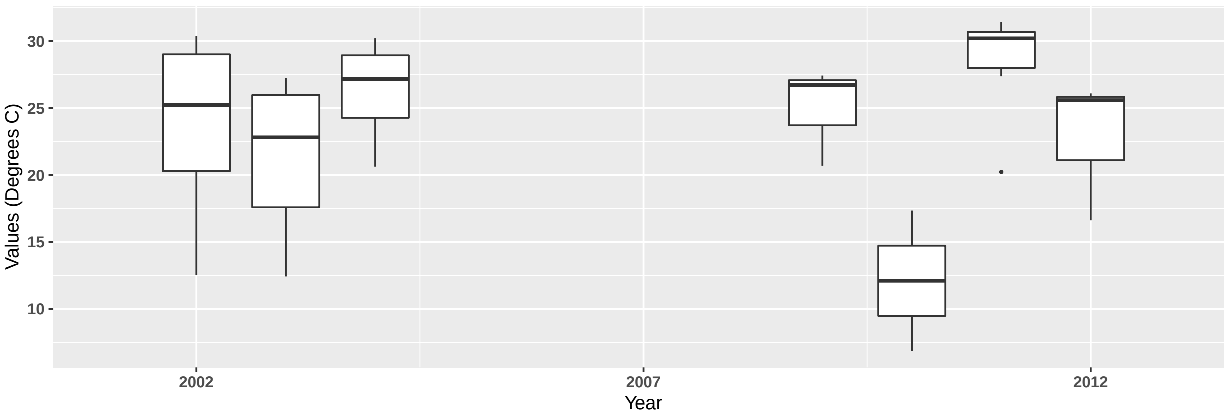


Summary Box Plots for Yellow River Marsh Aquatic Preserve
467 | Yellow River Marsh Aquatic Preserve Continuous Water Quality Monitoring | YRMAP1
By Month

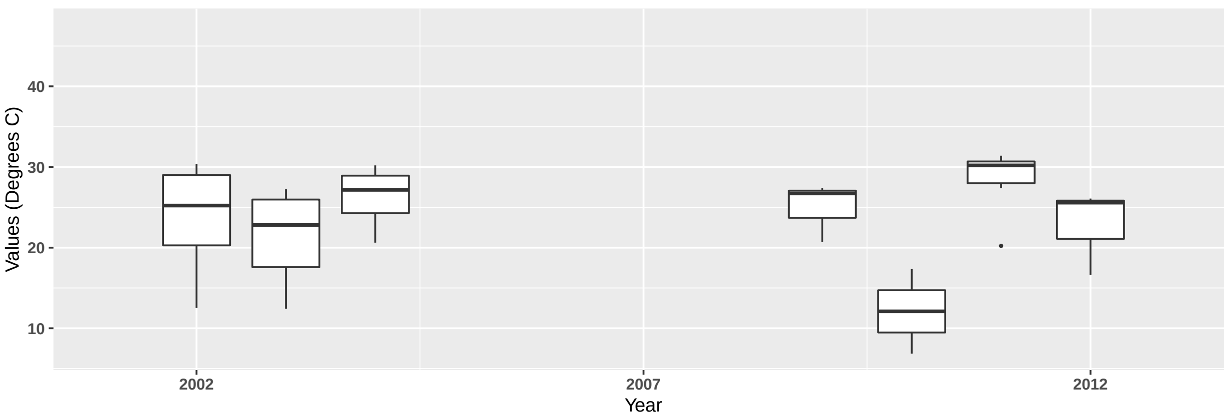


Summary Box Plots for Yellow River Marsh Aquatic Preserve
505 | Pensacola Bay Water Quality Monitoring Program | P11
By Year

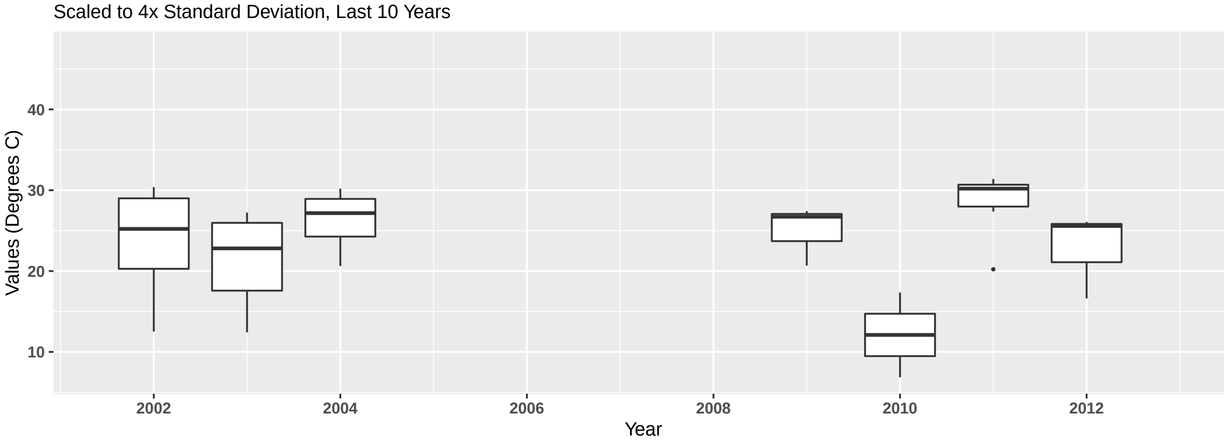
Autoscale



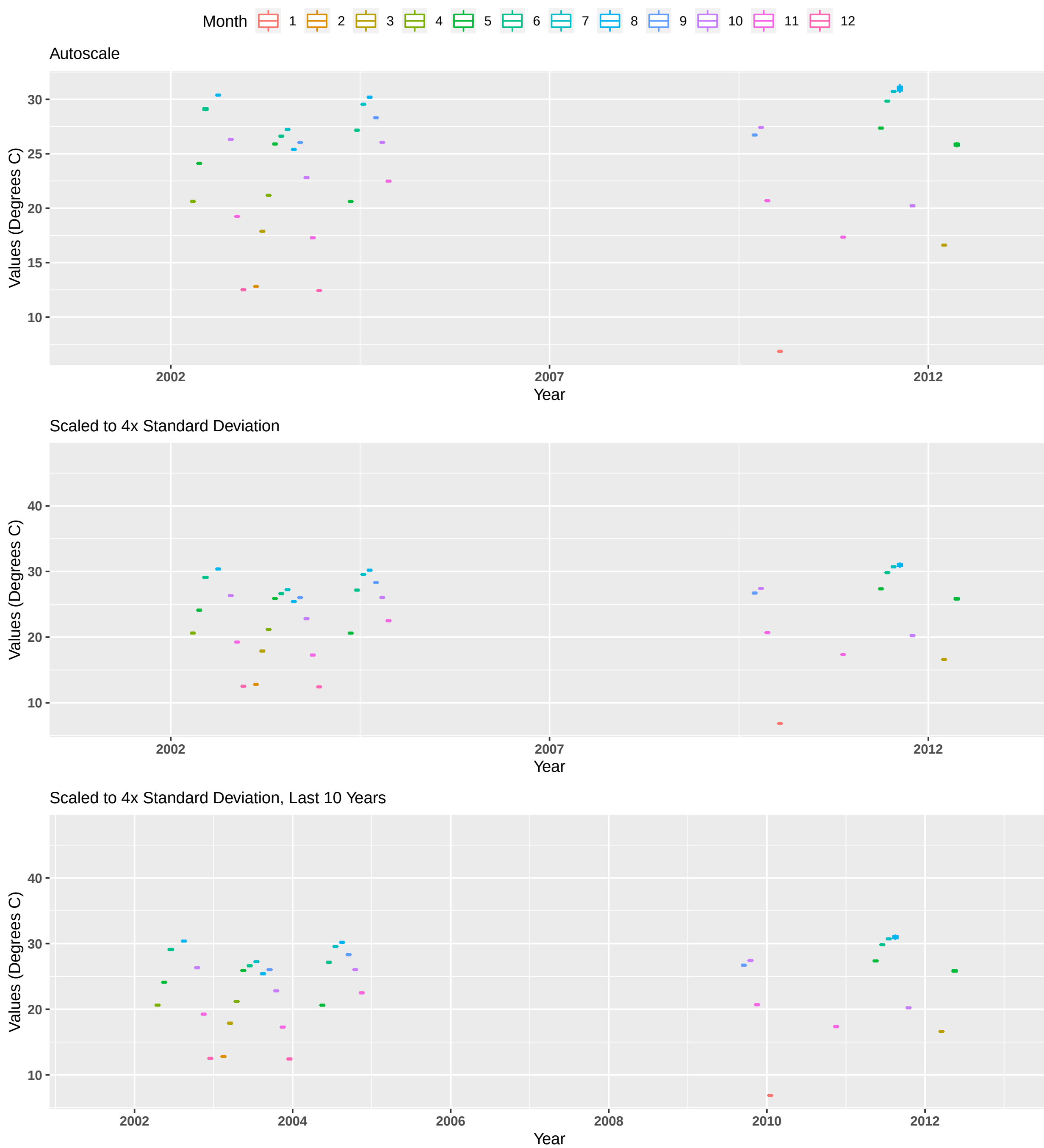
Scaled to 4x Standard Deviation



Scaled to 4x Standard Deviation, Last 10 Years



Summary Box Plots for Yellow River Marsh Aquatic Preserve
505 | Pensacola Bay Water Quality Monitoring Program | P11
By Year & Month



Summary Box Plots for Yellow River Marsh Aquatic Preserve
505 | Pensacola Bay Water Quality Monitoring Program | P11
By Month