SEACAR Discrete Water Quality Analysis: Secchi Depth

Last compiled on 07 April, 2022

Contents

Purpose 1

Adjustable Inputs 2

Libraries 2

File Import 3

Data Filtering and Data Impacted by Specific Value Qualifiers 3

Managed Area Statistics 5

Monitoring Location Statistics 7

Seasonal Kendall Tau Analysis 7

Appendix I: Scatter Plot of Entire Dataset 10

Appendix II: Dataset Summary Box Plots 14

Appendix III: Excluded Managed Areas 20

Appendix IV: Managed Area Trendlines 24

Appendix V: Managed Area Summary Box Plots 45

Purpose

The purpose of this script is to analyze the discrete secchi depth 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 the relative depth of interest (`depth`). 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_"
file_date <- "2022-Apr-05"
param_name <- "Secchi_Depth"
depth <- "Surface"
```

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.

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.

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

file_in <- paste0(in_dir, file_pref, param_name, "-", 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 = "")

unit <- unique(data$ParameterUnits)
data$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 Filtering and Data Impacted by Specific Value Qualifiers

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 only keeps data that is measured at the relative depth (surface, bottom, etc.) of interest. This is partly handled on export with the `RelativeDepth` variable, but there are some measurements that are considered both surface and bottom based on measurement depth and total depth. By default, these are marked as `Surface` for `RelativeDepth` and receive a `SEACAR_QAQCFlag` indicator of `12Q`. Data passes the filtering the process if it is from the correct depth and has an `Include` value of 1.

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 10 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.

After filtering, the amount of data impacted by the H (for dissolved oxygen & pH in program 476), I, Q, and U value qualifiers. A variable is also created that determines if scatter plot points should be a different color based on value qualifiers of interest. The value qualifier S indicates that the Secchi depth was visible on bottom.

```r
if(depth=="Bottom"){
   data$RelativeDepth[grep("12Q", data$SEACAR_QAQCFlagCode[
      data$RelativeDepth == "Surface"])] <- "Bottom"
}

data$Include <- as.logical(data$Include)
data <- data[!is.na(data$ResultValue),]
data <- data[!is.na(data$RelativeDepth) & data$RelativeDepth==depth,]
if(param_name == "Water_Temperature"){
   data <- data[data$ResultValue>=-2,]
} else{
   data <- data[data$ResultValue>=0,]
}
data$Include[grep("H", data$ValueQualifier[data$ProgramID==476])] <- TRUE

MA_Years <- data[data$Include == TRUE, ] %>%
   group_by(ManagedAreaName) %>%
   summarize(N = length(unique(Year)))
MA_Years <- as.data.table(MA_Years[order(MA_Years$ManagedAreaName), ])
MA_Years$Enough_Time <- ifelse(MA_Years$N < 10, FALSE, TRUE)
data$Exclude_ManagedArea <- is.element(data$ManagedAreaName,
                                       MA_Years$ManagedAreaName[
                                          MA_Years$Enough_Time == FALSE])

data$Use_In_Analysis <- ifelse(data$Include == TRUE &
                                  data$Exclude_ManagedArea == FALSE,
                               TRUE, FALSE)
total <- length(data$Include)
pass_filter <- length(data$Include[data$Include==TRUE])

count_H <- length(grep("H", data$ValueQualifier[data$ProgramID==476]))
perc_H <- 100*count_H/length(data$ValueQualifier)

count_I <- length(grep("I", data$ValueQualifier))
perc_I <- 100*count_I/length(data$ValueQualifier)

count_Q <- length(grep("Q", data$ValueQualifier))
perc_Q <- 100*count_Q/length(data$ValueQualifier)

count_U <- length(grep("U", data$ValueQualifier))
perc_U <- 100*count_U/length(data$ValueQualifier)


data$VQ_Plot <- data$ValueQualifier

inc_H <- ifelse(param_name=="pH" | param_name=="Dissolved_Oxygen" |
                   param_name=="Dissolved_Oxygen_Saturation", TRUE, FALSE)

if (inc_H==TRUE){
   data$VQ_Plot <- gsub("[^HU]+", "", data$VQ_Plot)
   data$VQ_Plot <- gsub("UH", "HU", data$VQ_Plot)
   data$VQ_Plot[data$ProgramID!=476] <- gsub("[^U]+", "",
                                             data$VQ_Plot[data$ProgramID!=476])
```

```
    data$VQ_Plot[data$VQ_Plot==""] <- NA
    cat(paste0("Number of Measurements: ", total,
               ", Number Passed Filter: ", pass_filter, "\n",
               "Program 476 H Codes: ", count_H, " (", round(perc_H, 6), "%)\n",
               "I Codes: ", count_I, " (", round(perc_I, 6), "%)\n",
               "Q Codes: ", count_Q, " (", round(perc_Q, 6), "%)\n",
               "U Codes: ", count_U, " (", round(perc_U, 6), "%)"))
} else if (param_name=="Secchi_Depth") {
    count_S <- length(grep("S", data$ValueQualifier))
    perc_S <- 100*count_S/length(data$ValueQualifier)
    data$VQ_Plot <- gsub("[^SU]+", "", data$VQ_Plot)
    data$VQ_Plot <- gsub("US", "SU", data$VQ_Plot)
    data$VQ_Plot[data$VQ_Plot==""] <- NA
    cat(paste0("Number of Measurements: ", total,
               ", Number Passed Filter: ", pass_filter, "\n",
               "I Codes: ", count_I, " (", round(perc_I, 6), "%)\n",
               "Q Codes: ", count_Q, " (", round(perc_Q, 6), "%)\n",
               "S Codes: ", count_S, " (", round(perc_S, 6), "%)\n",
               "U Codes: ", count_U, " (", round(perc_U, 6), "%)"))
} else {
    data$VQ_Plot <- gsub("[^U]+", "", data$VQ_Plot)
    data$VQ_Plot[data$VQ_Plot==""] <- NA
    cat(paste0("Number of Measurements: ", total,
               ", Number Passed Filter: ", pass_filter, "\n",
               "I Codes: ", count_I, " (", round(perc_I, 6), "%)\n",
               "Q Codes: ", count_Q, " (", round(perc_Q, 6), "%)\n",
               "U Codes: ", count_U, " (", round(perc_U, 6), "%)"))
}
```

```
## Number of Measurements: 227868, Number Passed Filter: 227851
## I Codes: 0 (0%)
## Q Codes: 0 (0%)
## S Codes: 1985 (0.871118%)
## U Codes: 0 (0%)
```

Managed Area Statistics

Gets summary statistics for each managed area. Excluded managed areas 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`, `Year`, and `Month`.
    - Second summary statistics do not use the `Month` grouping and are only for `ManagedAreaName` and `Year`.
    - Third summary statistics do not use `Year` grouping and are only for `ManagedAreaName` and `Month`
3. For each group, provide the following information: 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` then `Year` then `Month`
5. Write summary stats to a pipe-delimited .txt file in the output directory

   - Click this text to open Git directory with output files

```
MA_names <- unique(data$ManagedAreaName[data$Use_In_Analysis == TRUE])
MA_names <- MA_names[order(MA_names)]
n <- length(MA_names)

MA_YM_Stats <- data[data$Use_In_Analysis == TRUE, ] %>%
   group_by(ManagedAreaName, Year, Month) %>%
   summarize(N = length(ResultValue),
             Min = min(ResultValue),
             Max = max(ResultValue),
             Median = median(ResultValue),
             Mean = mean(ResultValue),
             StandardDeviation = sd(ResultValue),
             N_VoB = length(grep("S", ValueQualifier)),
             ProgramIDs = paste(sort(unique(ProgramID), decreasing = FALSE),
                                collapse = ', '))
MA_YM_Stats <- as.data.table(MA_YM_Stats[order(MA_YM_Stats$ManagedAreaName,
                                               MA_YM_Stats$Year,
                                               MA_YM_Stats$Month), ])
fwrite(MA_YM_Stats, paste0(out_dir,"/", param_name, "_", file_date, "_", depth,
                           "_ManagedArea_YearMonth_Stats.txt"), sep = "|")

MA_Y_Stats <- data[data$Use_In_Analysis == TRUE, ] %>%
   group_by(ManagedAreaName, Year) %>%
   summarize(N = length(ResultValue),
             Min = min(ResultValue),
             Max = max(ResultValue),
             Median = median(ResultValue),
             Mean = mean(ResultValue),
             StandardDeviation = sd(ResultValue),
             N_VoB = length(grep("S", ValueQualifier)),
             ProgramIDs = paste(sort(unique(ProgramID), decreasing = FALSE),
                                collapse = ', '))
MA_Y_Stats <- as.data.table(MA_Y_Stats[order(MA_Y_Stats$ManagedAreaName,
                                             MA_Y_Stats$Year), ])
fwrite(MA_Y_Stats, paste0(out_dir,"/", param_name, "_", file_date, "_", depth,
                          "_ManagedArea_Year_Stats.txt"), sep = "|")

MA_M_Stats <- data[data$Use_In_Analysis == TRUE, ] %>%
   group_by(ManagedAreaName, Month) %>%
   summarize(N = length(ResultValue),
             Min = min(ResultValue),
             Max = max(ResultValue),
             Median = median(ResultValue),
             Mean = mean(ResultValue),
             StandardDeviation = sd(ResultValue),
             N_VoB = length(grep("S", ValueQualifier)),
             ProgramIDs = paste(sort(unique(ProgramID), decreasing = FALSE),
                                collapse = ', '))
MA_M_Stats <- as.data.table(MA_M_Stats[order(MA_M_Stats$ManagedAreaName,
```

```
                                                  MA_M_Stats$Month), ])
fwrite(MA_M_Stats, paste0(out_dir,"/", param_name, "_", file_date, "_", depth,
                          "_ManagedArea_Month_Stats.txt"), sep = "|")
```

Monitoring Location Statistics

Gets monitoring location statistics, which is defined as a unique combination of `ManagedAreaName`, `ProgramID`, `ProgramAreaName`, and `ProgramLocationID`, using piping from dplyr package. 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`, and `ProgramLocationID`.
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, and Standard Deviation.
4. Sort the data in ascending (A to Z and 0 to 9) order based on `ManagedAreaName` then `ProgramName` then `ProgramID` then `ProgramLocationID`
5. Write summary stats to a pipe-delimited .txt file in the output directory
    - Click this text to open Git directory with output files

```
Mon_Stats <- data[data$Use_In_Analysis == TRUE, ] %>%
   group_by(ManagedAreaName, ProgramID, ProgramName, ProgramLocationID) %>%
   summarize(EarliestSampleDate = min(SampleDate),
             LastSampleDate = max(SampleDate),
             N = length(ResultValue),
             Min = min(ResultValue),
             Max = max(ResultValue),
             Median = median(ResultValue),
             Mean = mean(ResultValue),
             StandardDeviation = sd(ResultValue),
             N_VoB = length(grep("S", ValueQualifier)))

Mon_Stats <- as.data.table(Mon_Stats[order(Mon_Stats$ManagedAreaName,
                                             Mon_Stats$ProgramName,
                                             Mon_Stats$ProgramID,
                                             Mon_Stats$ProgramLocationID), ])
fwrite(Mon_Stats, paste0(out_dir,"/", param_name, "_", file_date, "_", depth,
                          "_MonitoringLoc_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 code created by Jason Scolaro that performed at The Water Atlas: https://sarasota.wateratlas.usf.edu/water-quality-trends/#analysis-overview

The following steps are performed:

1. Define the functions used in the analysis
2. Check to see if there are any groups to run analysis on.
3. Take the `data` variable and only include rows that have a `Use_In_Analysis` value of `TRUE`
4. Group data that have the same `ManagedAreaName`.
5. 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, tau, Senn Slope (SennSlope), Senn Intercept (SennIntercept), and p.
   - The analysis is run with 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.
6. Reformat columns in the data frame from export.
7. Write summary stats to a pipe-delimited .txt file in the output directory
   - Click this text to open Git directory with output files

```
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 = {
      7
      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$ManagedAreaName),
         independent,
         stats.median,
         nrow(data),
         stats.minYear,
         stats.maxYear,
         tau,
         p,
         slope,
         intercept,
         trend)
  return(KT)
}
runStats <- function(data) {
  data$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$ManagedAreaName
c_names <- c("ManagedAreaName", "Independent", "Median", "N", "EarliestYear",
             "LatestYear", "tau", "p", "SennSlope", "SennIntercept", "Trend")
if(n==0){
   KT.Stats <- data.frame(matrix(ncol=11, nrow=0))
   colnames(KT.Stats) <- c_names
   fwrite(KT.Stats, paste0(out_dir,"/", param_name, "_", file_date, "_", depth,
                           "_KendallTau_Stats.txt"), sep = "|")
} else{
   for (i in 1:n) {
      values <- data[data$Use_In_Analysis == TRUE &
                        data$ManagedAreaName == MA_names[i], ]
      if (nrow(values) > 0) {
         KT.Stats <- runStats(values)
      }
   }
   KT.Stats <- as.data.frame(KT.Stats)
   c_names <- c("ManagedAreaName", "Independent", "Median", "N", "EarliestYear",
                "LatestYear", "tau", "p", "SennSlope", "SennIntercept", "Trend")
   if(dim(KT.Stats)[2]==1){
      KT.Stats <- as.data.frame(t(KT.Stats))
   }
   colnames(KT.Stats) <- c_names
   rownames(KT.Stats) <- seq(1:nrow(KT.Stats))
   KT.Stats$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)
   fwrite(KT.Stats, paste0(out_dir,"/", param_name, "_", file_date, "_", depth,
                           "_KendallTau_Stats.txt"), sep = "|")
}
```

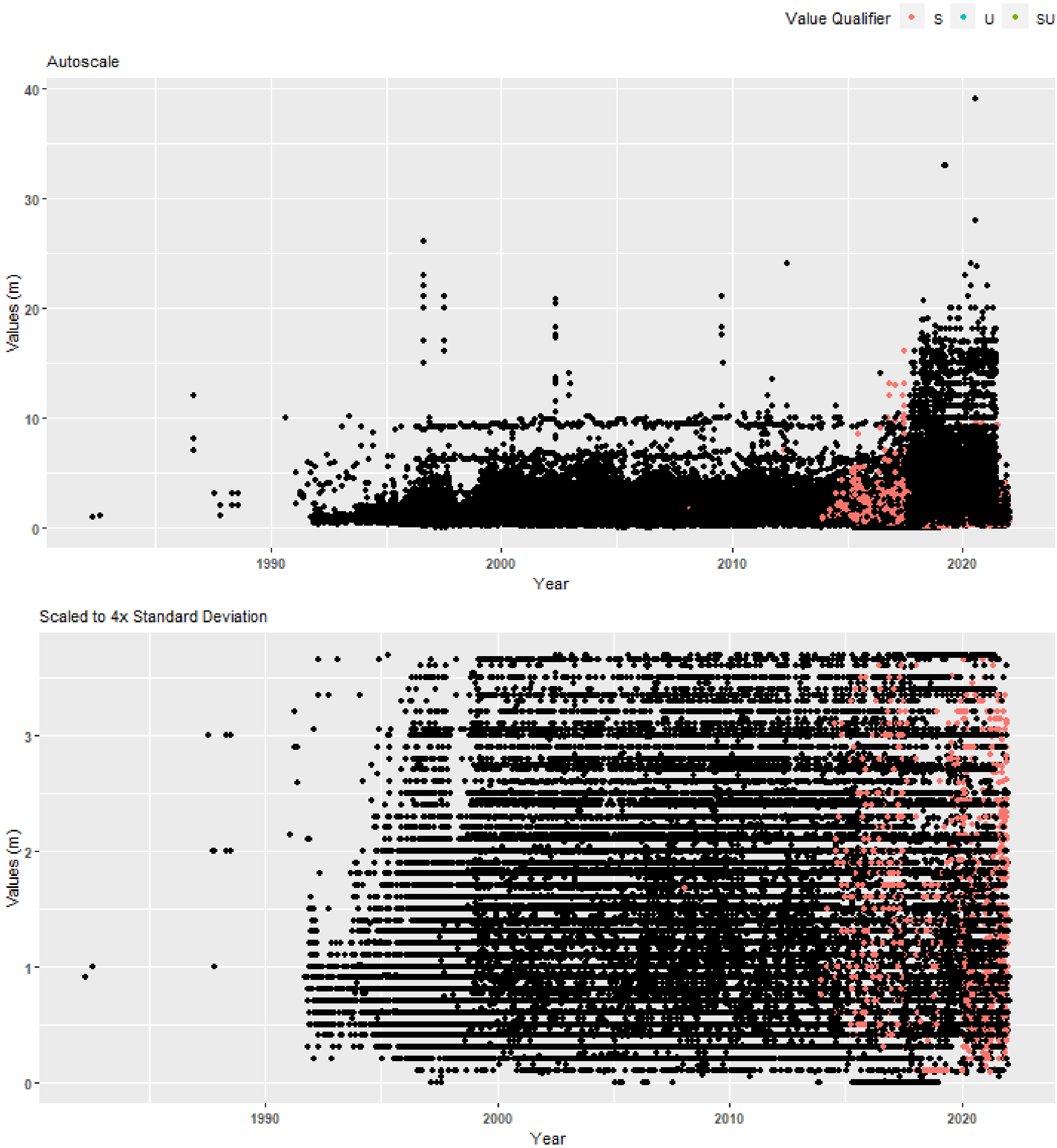
Appendix I: Scatter Plot of Entire Dataset

This part will create a scatter plot of the all data that passed initial filtering criteria with points colored based on specific value qualifiers. The values determined at the beginning (`year_lower`, `year_upper`, `min_RV`, mn_RV, x_scale, and y_scale) are solely for use by the plotting functions and are not output as part of the computed statistics.

```
year_lower <- min(data$Year)
year_upper <- max(data$Year)
min_RV <- min(data$ResultValue)
mn_RV <- mean(data$ResultValue[data$ResultValue <
                                  quantile(data$ResultValue, 0.98)])
sd_RV <- sd(data$ResultValue[data$ResultValue <
                                quantile(data$ResultValue, 0.98)])
x_scale <- ifelse(year_upper - year_lower > 30, 10, 5)
y_scale <- mn_RV + 4 * sd_RV


p1 <- ggplot(data = data[data$Include==TRUE,],
             aes(x = SampleDate, y = ResultValue,
                 color=VQ_Plot)) +
   geom_point(size = 1.5) +
   labs(subtitle = "Autoscale",
        x = "Year", y = paste0("Values (", unit, ")"),
        color="Value Qualifier") +
   theme(legend.position = "top", legend.box = "horizontal",
         legend.justification = "right",
         axis.text.x = element_text(face = "bold"),
         axis.text.y = element_text(face="bold")) +
   scale_x_date(labels = date_format("%Y")) +
   {if(inc_H==TRUE){
      scale_color_manual(values = c("H"= "#F8766D", "U"= "#00BFC4",
                                    "HU" = "#7CAE00"), na.value="black")
   } else if (param_name=="Secchi_Depth") {
      scale_color_manual(values = c("S"= "#F8766D", "U"= "#00BFC4",
                                    "SU" = "#7CAE00"), na.value="black")
   } else {
      scale_color_manual(values = c("U"= "#00BFC4"), na.value="black")
   }}


p2 <- ggplot(data = data[data$Include==TRUE,],
             aes(x = SampleDate, y = ResultValue,
                 color=VQ_Plot)) +
   geom_point(size = 1.5) +
   ylim(min_RV, y_scale) +
   labs(subtitle = "Scaled to 4x Standard Deviation",
        x = "Year", y = paste0("Values (", unit, ")")) +
   theme(legend.position = "none",
         axis.text.x = element_text(face = "bold"),
         axis.text.y = element_text(face = "bold")) +
   scale_x_date(labels = date_format("%Y")) +
   {if(inc_H==TRUE){
      scale_color_manual(values = c("H"= "#F8766D", "U"= "#00BFC4",
                                    "HU" = "#7CAE00"), na.value="black")
   } else if (param_name=="Secchi_Depth") {
      scale_color_manual(values = c("S"= "#F8766D", "U"= "#00BFC4",
                                    "SU" = "#7CAE00"), na.value="black")
```

```
  } else {
    scale_color_manual(values = c("U"= "#00BFC4"), na.value="black")
  }}

leg <- get_legend(p1)
pset <- ggarrange(leg, p1 + theme(legend.position = "none"), p2,
                  ncol = 1, heights = c(0.1, 1, 1))

p0 <- ggplot() + labs(title = "Scatter Plot for Entire Dataset") +
  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())

ggarrange(p0, pset, ncol = 1, heights = c(0.1, 1))
```

Scatter Plot for Entire Dataset



Appendix II: 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(min_RV, y_scale) +
   theme(axis.text.x = element_text(face = "bold"),
         axis.text.y = element_text(face = "bold"))

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

```r
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 4x Standard Deviation",
        x = "Year", y = paste0("Values (", unit, ")")) +
   ylim(min_RV, 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$Include == TRUE, ],
             aes(x = YearMonthDec, y = ResultValue,
                 group = YearMonth, color = as.factor(Month))) +
   geom_boxplot(outlier.size = 0.5) +
   labs(subtitle = "Scaled to 4x Standard Deviation, Last 10 Years",
        x = "Year", y = paste0("Values (", unit, ")")) +
   ylim(min_RV, y_scale) +
   scale_x_continuous(limits = c(max(data$Year) - 10.5, max(data$Year)+1),
                      breaks = seq(max(data$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())
```

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

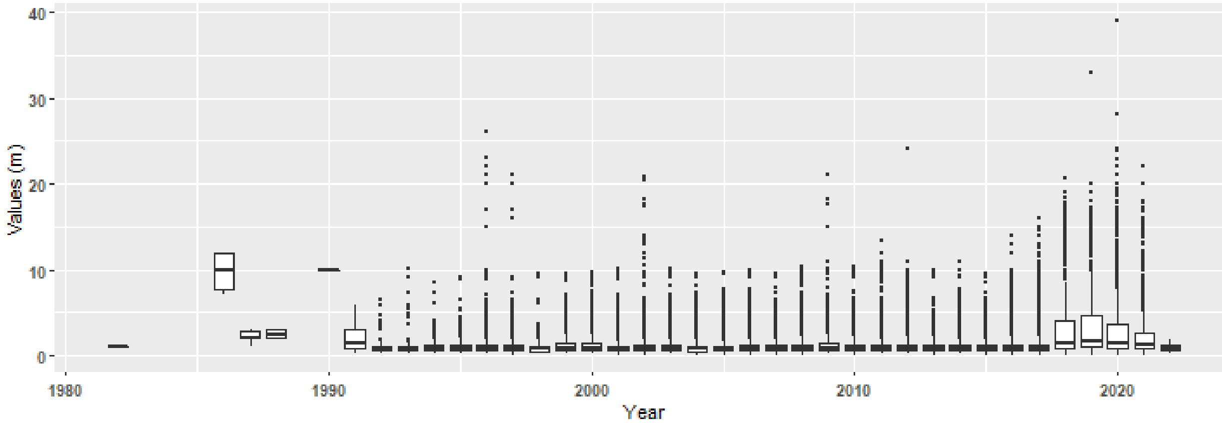
The following box plots are grouped by month with fill color being related to the month. This is designed to view potential seasonal trends.

```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 4x Standard Deviation",
        x = "Month", y = paste0("Values (", unit, ")")) +
   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"))

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 4x Standard Deviation, Last 10 Years",
        x = "Month", y = paste0("Values (", unit, ")")) +
   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"))
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))
```

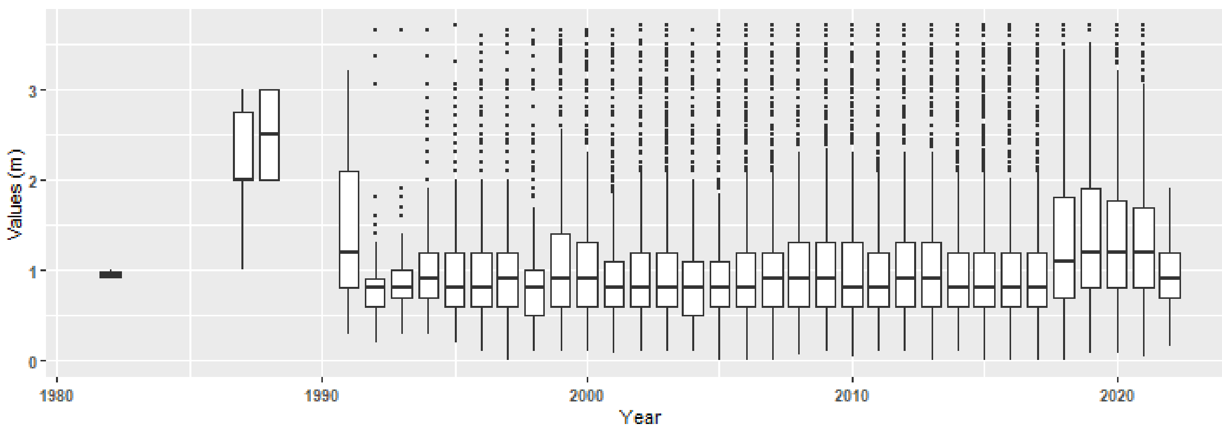
Summary Box Plots for Entire Data

By Year

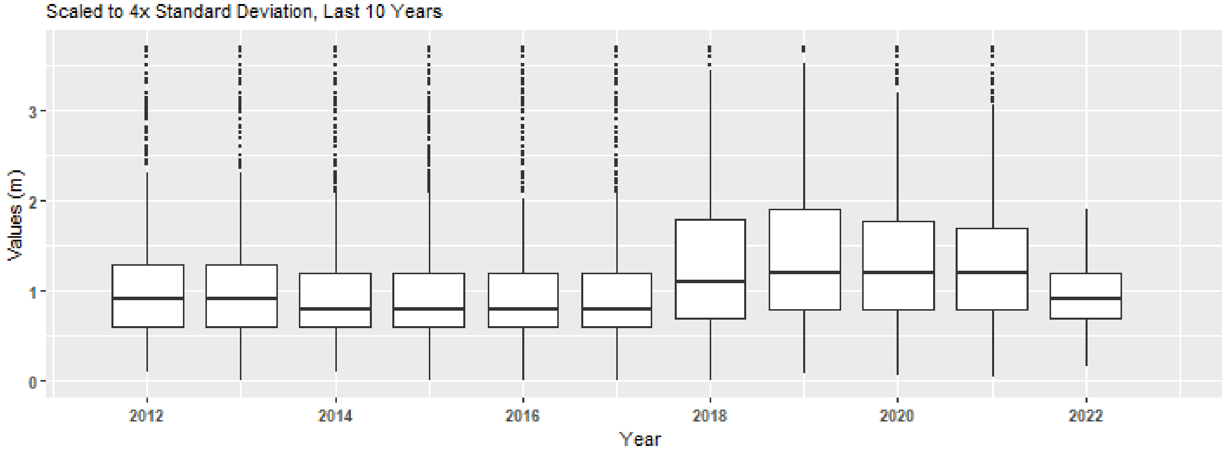
Autoscale



Scaled to 4x Standard Deviation

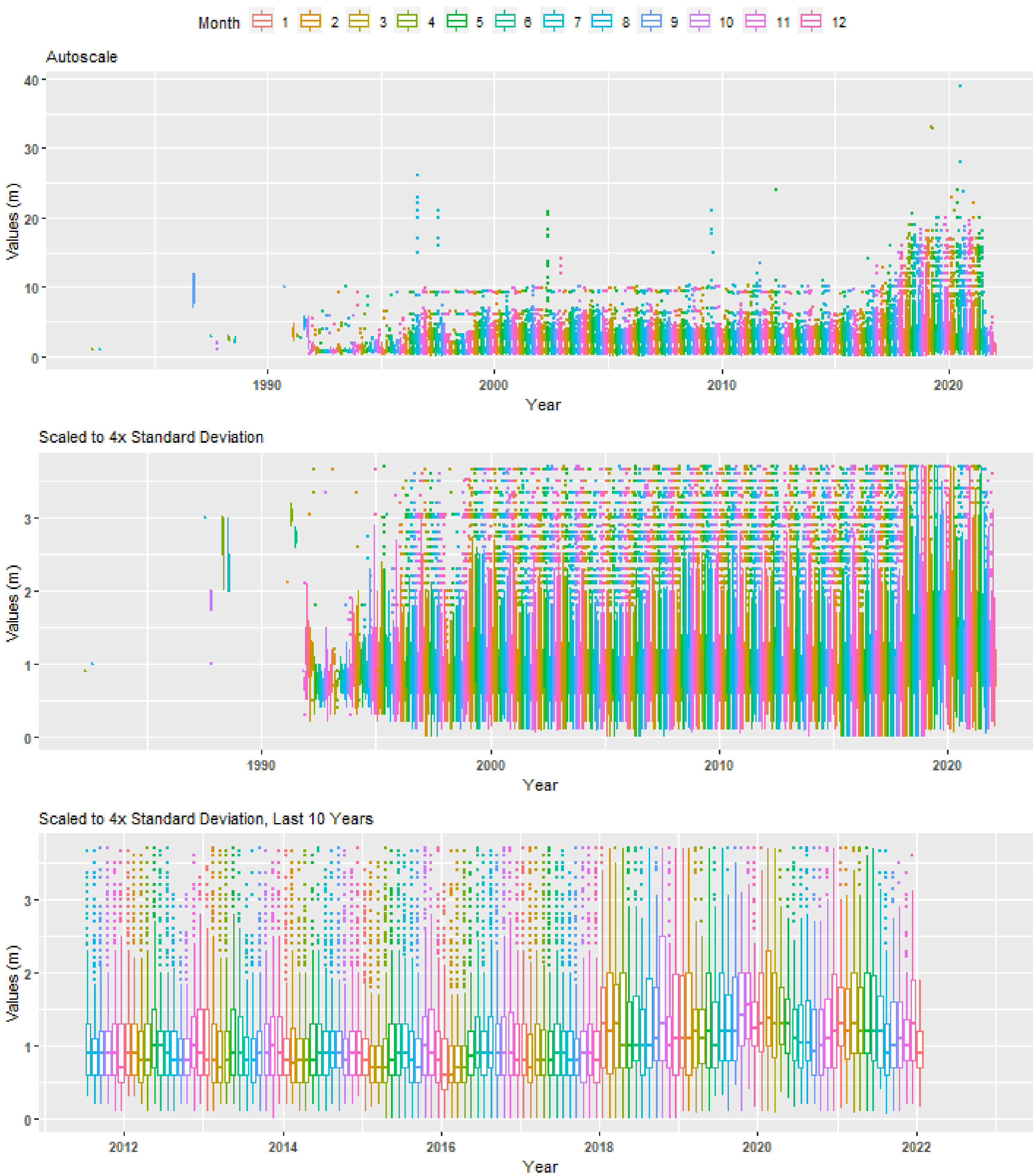


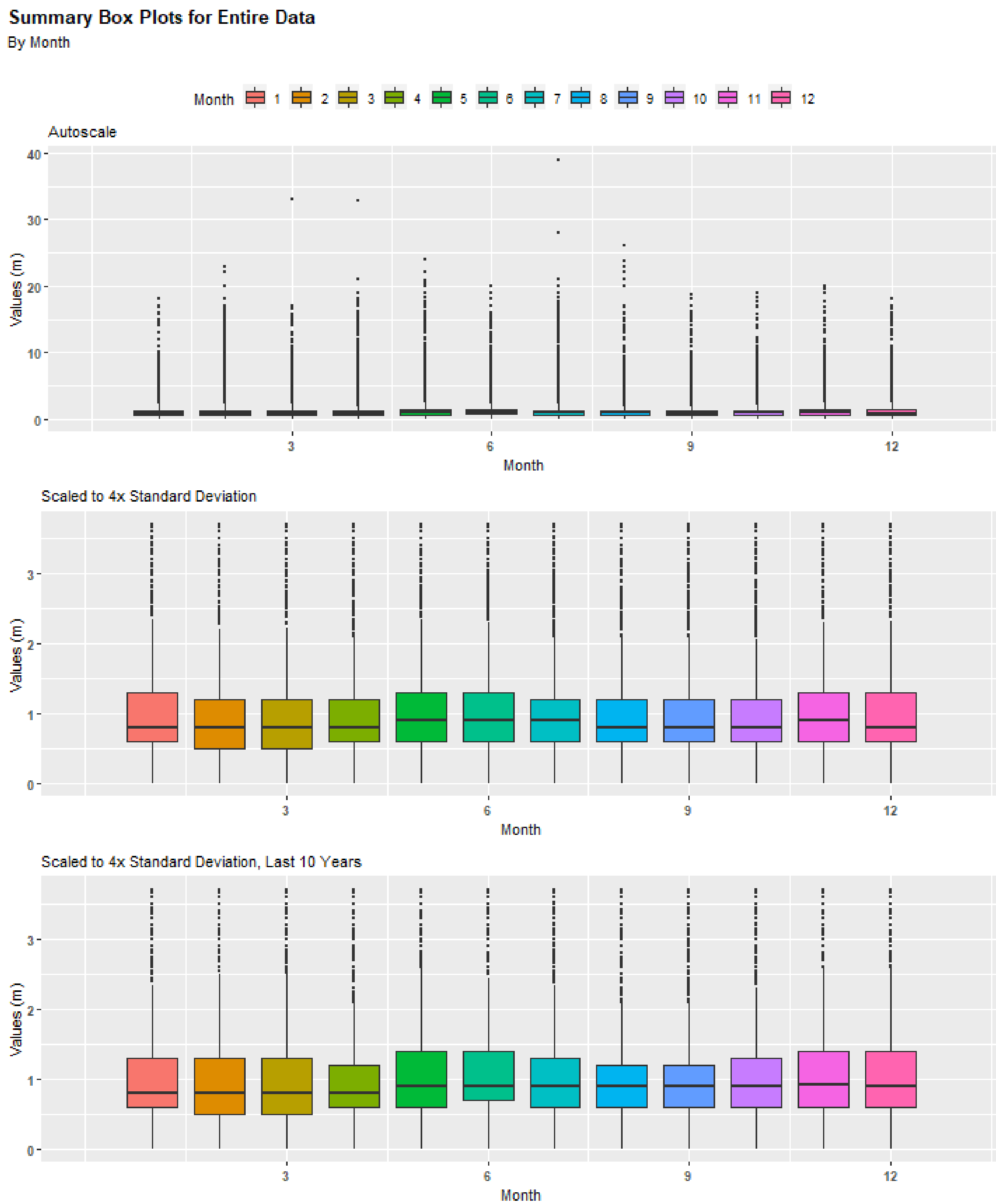
Scaled to 4x Standard Deviation, Last 10 Years



Summary Box Plots for Entire Data

By Year & Month



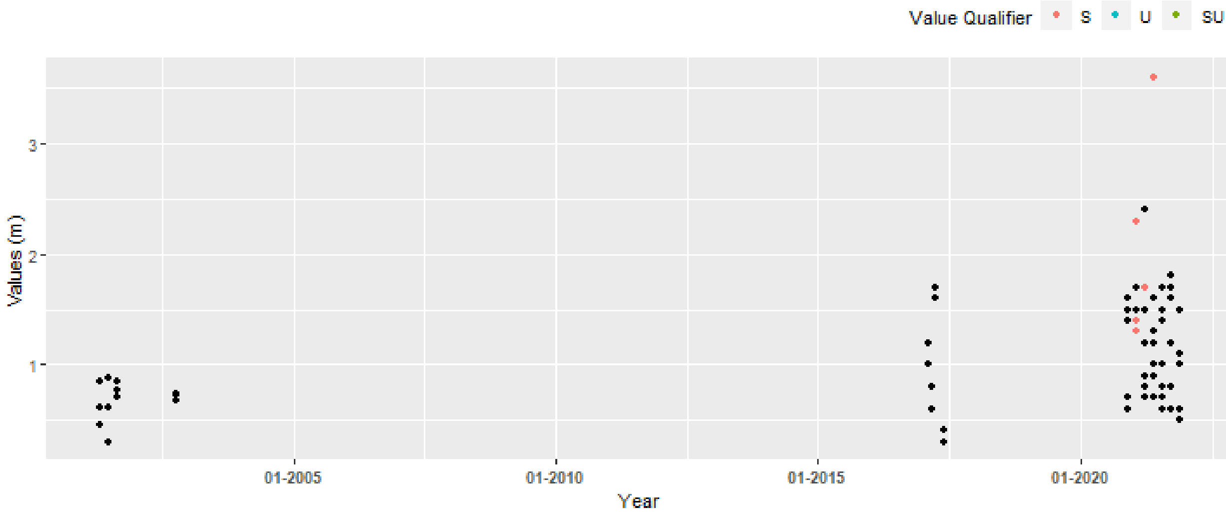

Summary Box Plots for Entire Data
By Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (m)
0
10
20
30
40
3
6
9
12
Month
Scaled to 4x Standard Deviation
Values (m)
0
1
2
3
3
6
9
12
Month
Scaled to 4x Standard Deviation, Last 10 Years
Values (m)
0
1
2
3
3
6
9
12
Month

Appendix III: Excluded Managed Areas

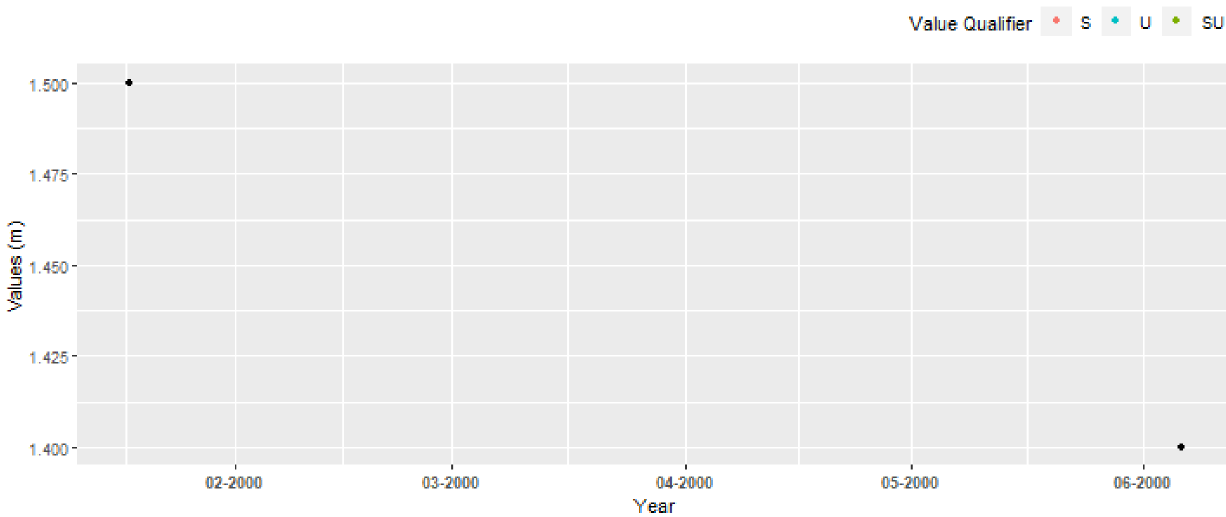
Scatter plots of data values are created for managed areas that have fewer than 10 separate years of data entries. Data points are colored based on specific value qualifiers of interest.

```
MA_Exclude <- MA_Years[MA_Years$Enough_Time==FALSE,]
MA_Exclude <- MA_Exclude[order(MA_Exclude$ManagedAreaName),]
z=length(MA_Exclude$ManagedAreaName)

if(z==0){
   print("There are no managed areas that qualify.")
} else {
   for(i in 1:z){

      p1<-ggplot(data=data[data$ManagedAreaName==MA_Exclude$ManagedAreaName[i]&
                              data$Include == TRUE, ],
                 aes(x = SampleDate, y = ResultValue, color=VQ_Plot)) +
         geom_point() +
         labs(title = paste0("Scatter Plot of Excluded Managed Area\n",
                             MA_Exclude$ManagedAreaName[i], " (",
                             MA_Exclude$N[i], " Unique Years)"),
              subtitle="Autoscale", x = "Year",
              y = paste0("Values (", unit, ")"), color="Value Qualifier") +
         theme(legend.position = "top", legend.box = "horizontal",
               legend.justification = "right",
               axis.text.x = element_text(face = "bold")) +
         scale_x_date(labels = date_format("%m-%Y")) +
         {if(inc_H==TRUE){
            scale_color_manual(values = c("H"= "#F8766D", "U"= "#00BFC4",
                                          "HU" = "#7CAE00"), na.value="black")
         } else if (param_name=="Secchi_Depth") {
            scale_color_manual(values = c("S"= "#F8766D", "U"= "#00BFC4",
                                          "SU" = "#7CAE00"), na.value="black")
         } else {
            scale_color_manual(values = c("U"= "#00BFC4"), na.value="black")
         }}
      print(p1)
   }
}
```

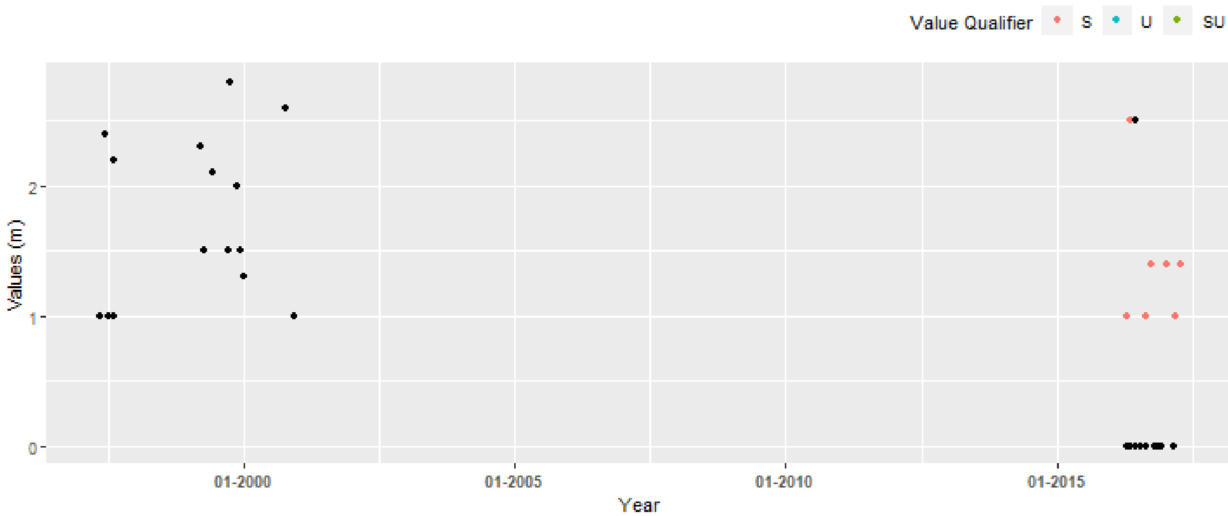
Scatter Plot of Excluded Managed Area
Cape Romano-Ten Thousand Islands Aquatic Preserve (5 Unique Years)
Autoscale



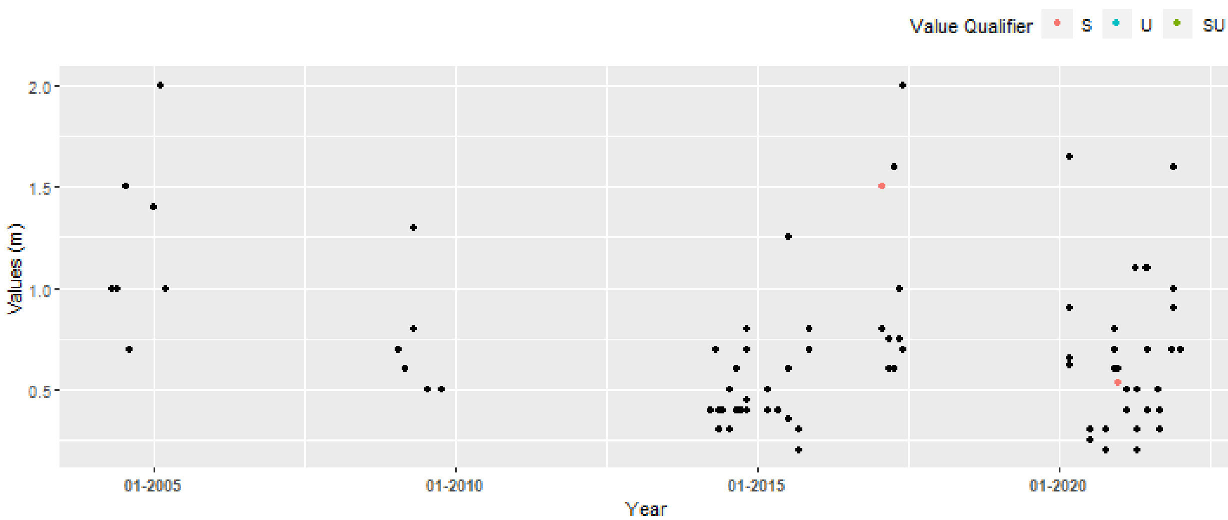
Scatter Plot of Excluded Managed Area
Coupon Bight Aquatic Preserve (1 Unique Years)
Autoscale

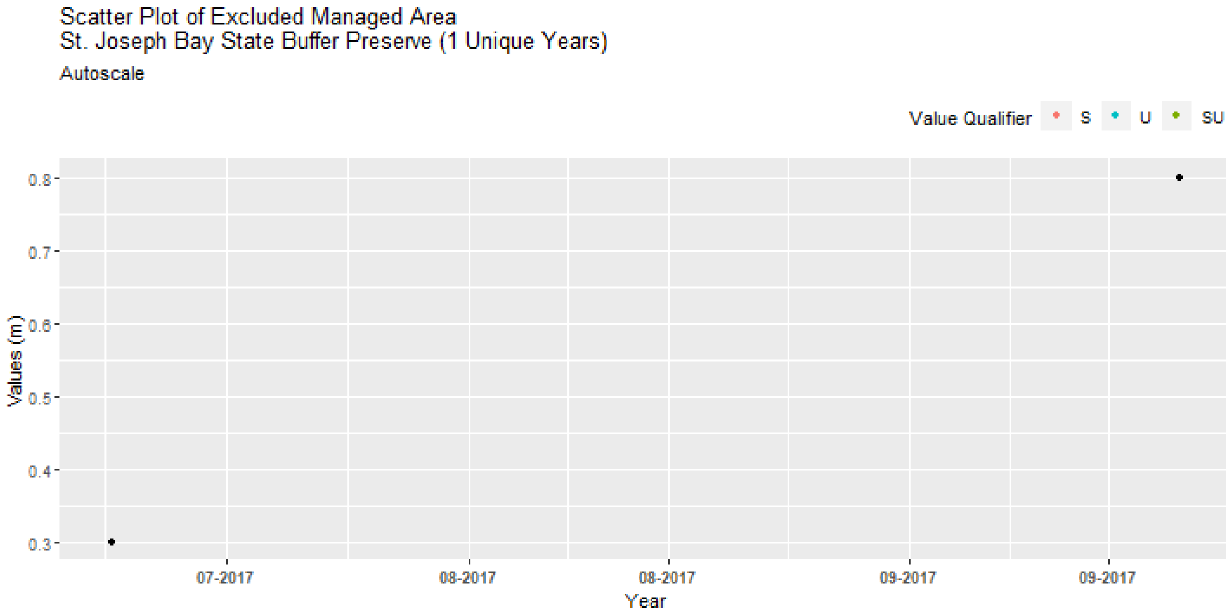


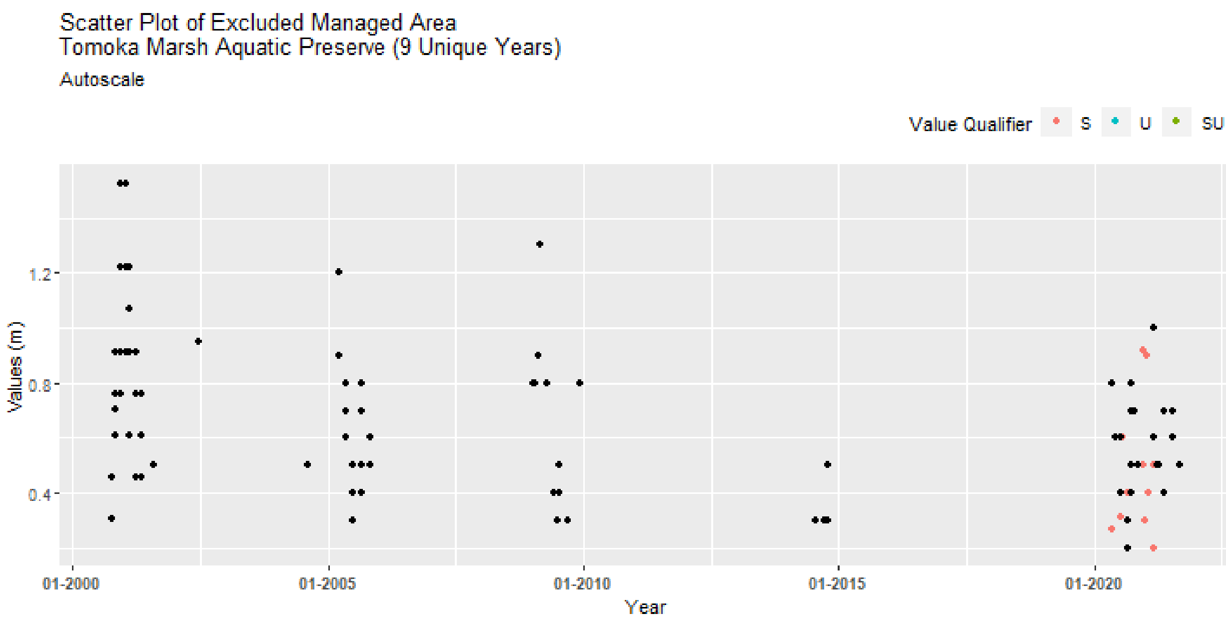
Scatter Plot of Excluded Managed Area
Lignumvitae Key Aquatic Preserve (5 Unique Years)
Autoscale

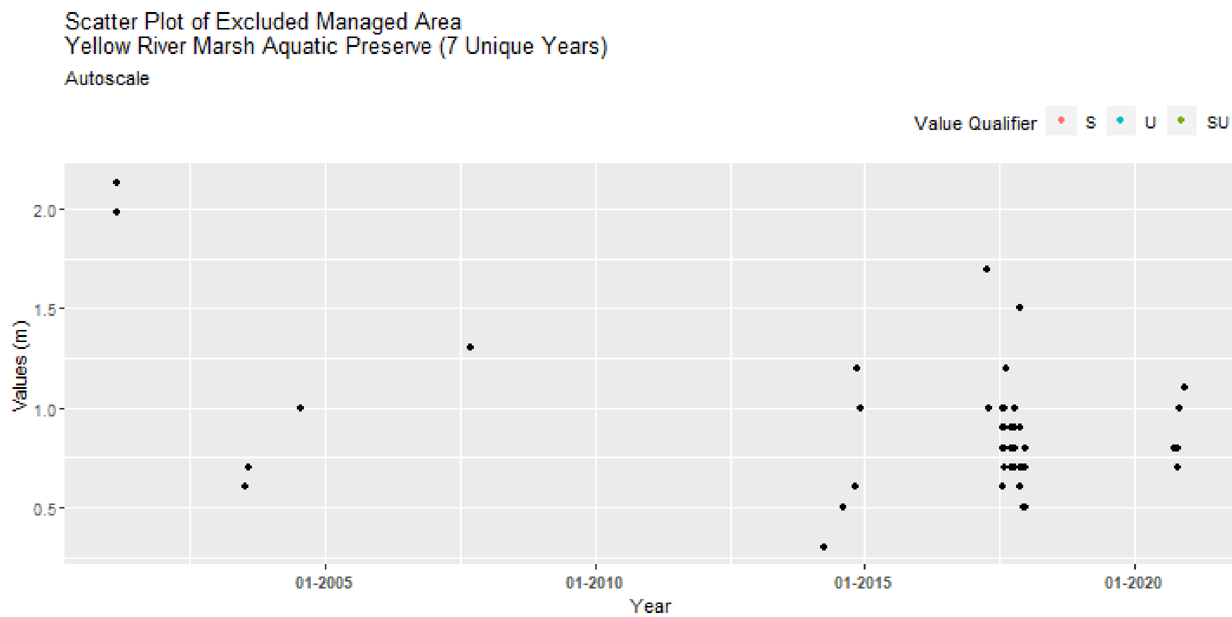


Scatter Plot of Excluded Managed Area
Pellicer Creek Aquatic Preserve (9 Unique Years)
Autoscale




Scatter Plot of Excluded Managed Area
St. Joseph Bay State Buffer Preserve (1 Unique Years)
Autoscale
Value Qualifier S U SU
Values (m)
0.8
0.7
0.6
0.5
0.4
0.3
07-2017
08-2017
08-2017
09-2017
09-2017
Year


Scatter Plot of Excluded Managed Area
Tomoka Marsh Aquatic Preserve (9 Unique Years)
Autoscale
Value Qualifier S U SU
Values (m)
1.2
0.8
0.4
01-2000
01-2005
01-2010
01-2015
01-2020
Year



Appendix IV: Managed Area Trendlines

The plots created in this section are designed to show the general trend of the data. Data is taken and grouped by `ManagedAreaName`. 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 managed area
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

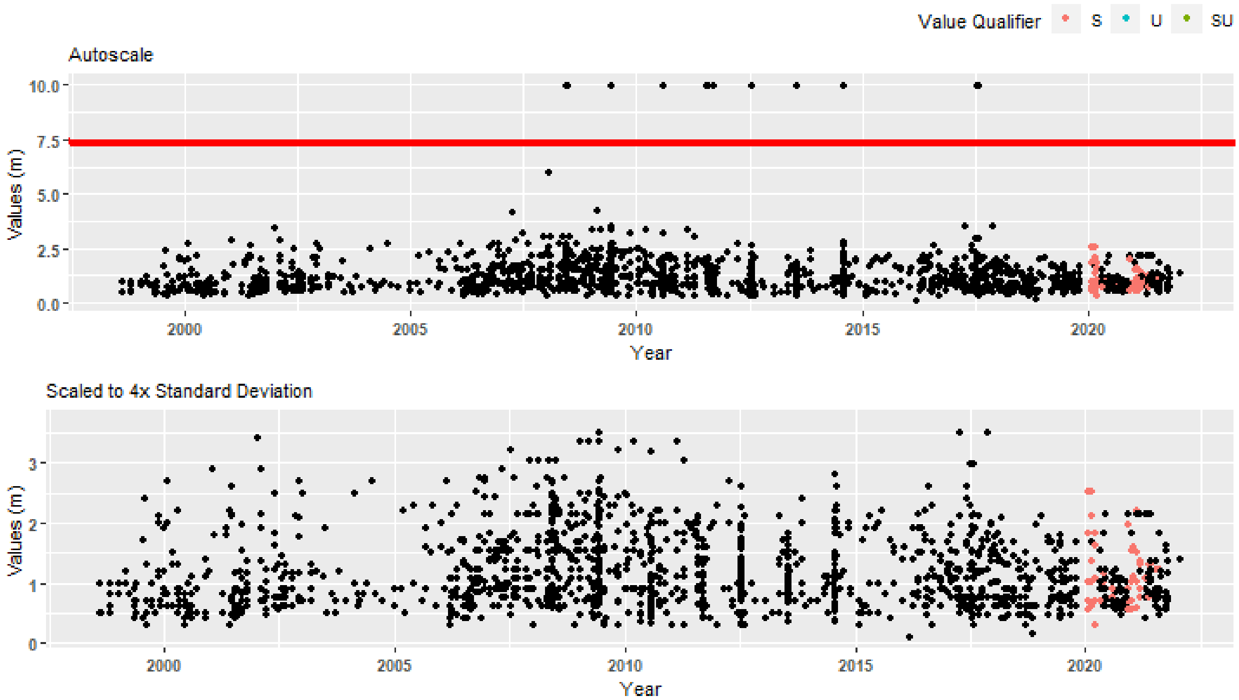
```
if(n==0){
  print("There are no managed areas that qualify.")
} else {
  for (i in 1:n) {
    year_lower <- min(data$Year[data$Use_In_Analysis == TRUE &
                                  data$ManagedAreaName ==
                                  MA_names[i]])
    year_upper <- max(data$Year[data$Use_In_Analysis == TRUE &
                                  data$ManagedAreaName ==
```

```
                                 MA_names[i]])
min_RV <- min(data$ResultValue[data$Use_In_Analysis == TRUE &
                                  data$ManagedAreaName == MA_names[i]])
mn_RV <- mean(data$ResultValue[data$Use_In_Analysis == TRUE &
                                  data$ManagedAreaName == MA_names[i] &
                                  data$ResultValue <
                                  quantile(data$ResultValue, 0.98)])
sd_RV <- sd(data$ResultValue[data$Use_In_Analysis == TRUE &
                                data$ManagedAreaName == MA_names[i] &
                                data$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$tau[KT.Stats$ManagedAreaName==MA_names[i]]
s_slope <- KT.Stats$SennSlope[KT.Stats$ManagedAreaName==MA_names[i]]
s_int <- KT.Stats$SennIntercept[KT.Stats$ManagedAreaName==MA_names[i]]
trend <- KT.Stats$Trend[KT.Stats$ManagedAreaName==MA_names[i]]
p <- KT.Stats$p[KT.Stats$ManagedAreaName==MA_names[i]]

model <- lm(ResultValue ~ DecDate,
            data = data[data$Use_In_Analysis == TRUE &
                           data$ManagedAreaName == MA_names[i]])
m_int <- coef(model)[[1]]
m_slope <- coef(model)[[2]]
p1 <- ggplot(data = data[data$Use_In_Analysis == TRUE &
                            data$ManagedAreaName == MA_names[i], ],
             aes(x = DecDate, y = ResultValue,
                 color=VQ_Plot)) +
   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, ")"),
        color="Value Qualifier") +
   theme(legend.position = "top", legend.box = "horizontal",
         legend.justification = "right",
         axis.text.x = element_text(face = "bold"),
         axis.text.y = element_text(face="bold")) +
   {if(inc_H==TRUE){
      scale_color_manual(values = c("H"= "#F8766D", "U"= "#00BFC4",
                                    "HU" = "#7CAE00"), na.value="black")
   } else if (param_name=="Secchi_Depth") {
      scale_color_manual(values = c("S"= "#F8766D", "U"= "#00BFC4",
                                    "SU" = "#7CAE00"), na.value="black")
   } else {
      scale_color_manual(values = c("U"= "#00BFC4"), na.value="black")
   }}

p2 <- ggplot(data = data[data$Use_In_Analysis == TRUE &
                            data$ManagedAreaName == MA_names[i], ],
             aes(x = DecDate, y = ResultValue,
                 color=VQ_Plot)) +
```

```r
        geom_point(size = 1.5) +
        geom_abline(aes(slope=s_slope, intercept=s_int),
                    color="red", size=1.5) +
        ylim(min_RV, y_scale) +
        labs(subtitle = "Scaled to 4x Standard Deviation",
             x = "Year", y = paste0("Values (", unit, ")")) +
        theme(legend.position = "none",
              axis.text.x = element_text(face = "bold"),
              axis.text.y = element_text(face="bold")) +
        {if(inc_H==TRUE){
           scale_color_manual(values = c("H"= "#F8766D", "U"= "#00BFC4",
                                         "HU" = "#7CAE00"), na.value="black")
        } else if (param_name=="Secchi_Depth") {
           scale_color_manual(values = c("S"= "#F8766D", "U"= "#00BFC4",
                                         "SU" = "#7CAE00"), na.value="black")
        } else {
           scale_color_manual(values = c("U"= "#00BFC4"), na.value="black")
        }}
      leg <- get_legend(p1)
      KTset <- ggarrange(leg, p1 + theme(legend.position = "none"), p2,
                         ncol = 1, heights = c(0.1, 1, 1))

      p0 <- ggplot() + labs(title = paste0("Data Points with Trendlines for ",
                                           MA_names[i]),
                            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(0.15, 1)))
   }
}
```

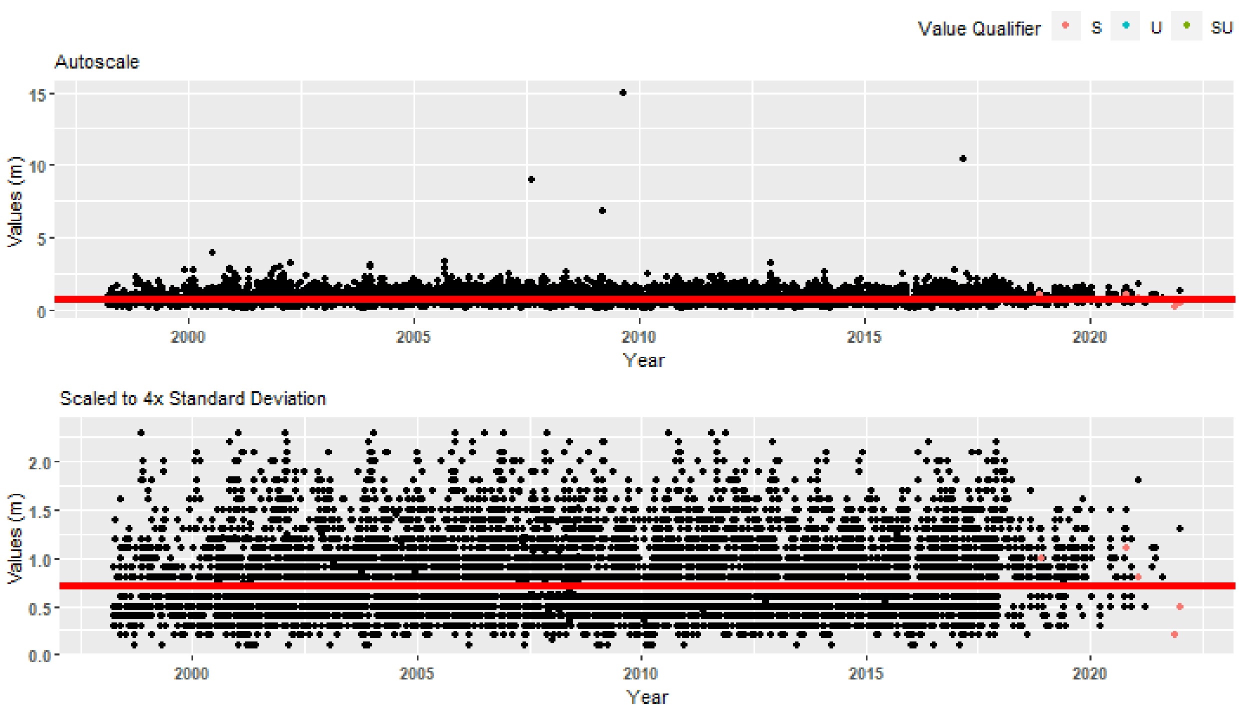
Data Points with Trendlines for Alligator Harbor Aquatic Preserve

Senn Slope = -0.00240185333333333,   Senn Intercept = 12.1891458173333
Trend = 0,   tau = -0.0254,   p = 0.1338
Linear Trendline: y = -0.00653170638700217x + 14.6237799260595



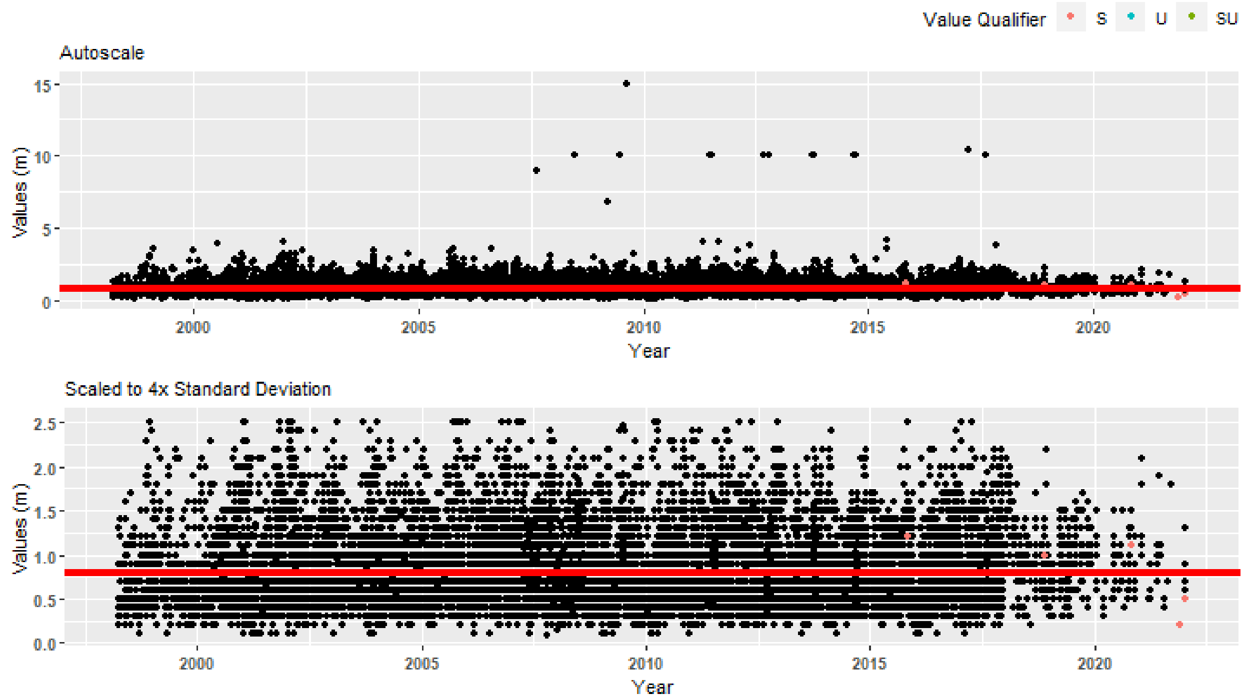
Data Points with Trendlines for Apalachicola Bay Aquatic Preserve

Senn Slope = 0,   Senn Intercept = 0.7
Trend = -1,   tau = 0.0135,   p = 0.0021
Linear Trendline: y = 0.000901416703220044x + -1.01337584330384



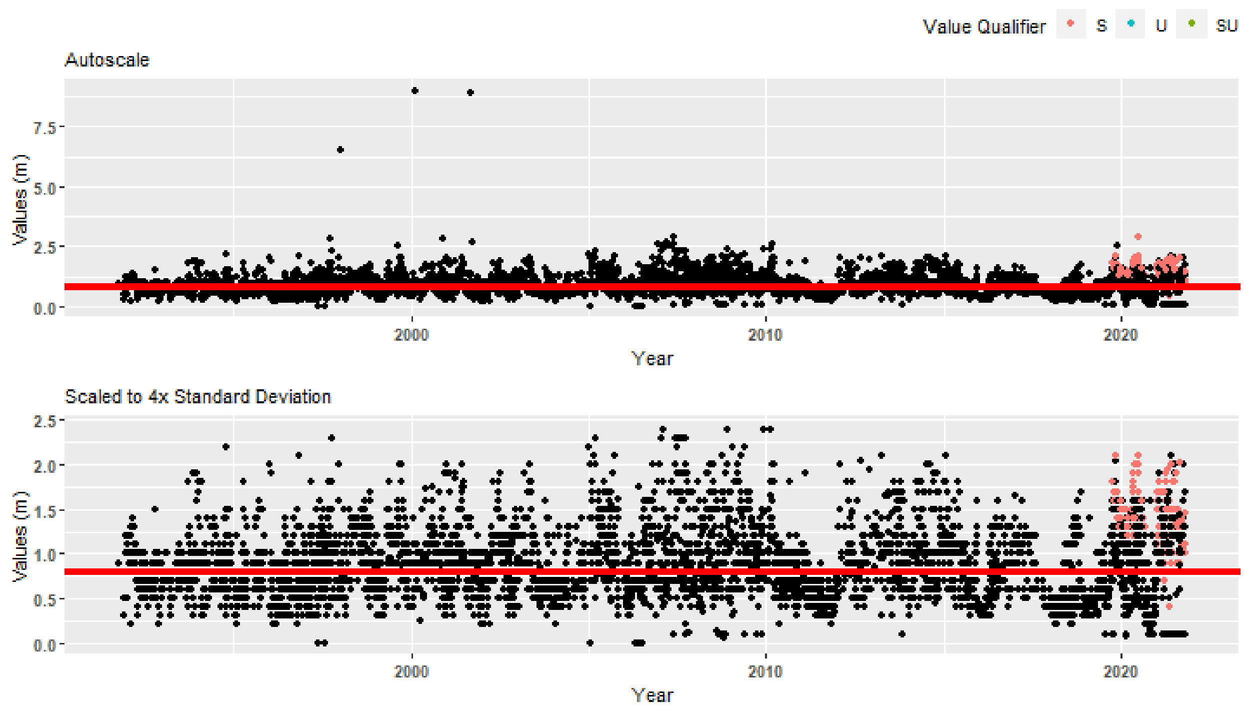
Data Points with Trendlines for Apalachicola National Estuarine Research Reserve

Senn Slope = 0, Senn Intercept = 0.8
Trend = 0, tau = 0.0014, p = 0.3599
Linear Trendline: y = 0.0010053325369741x + -1.16343807105137



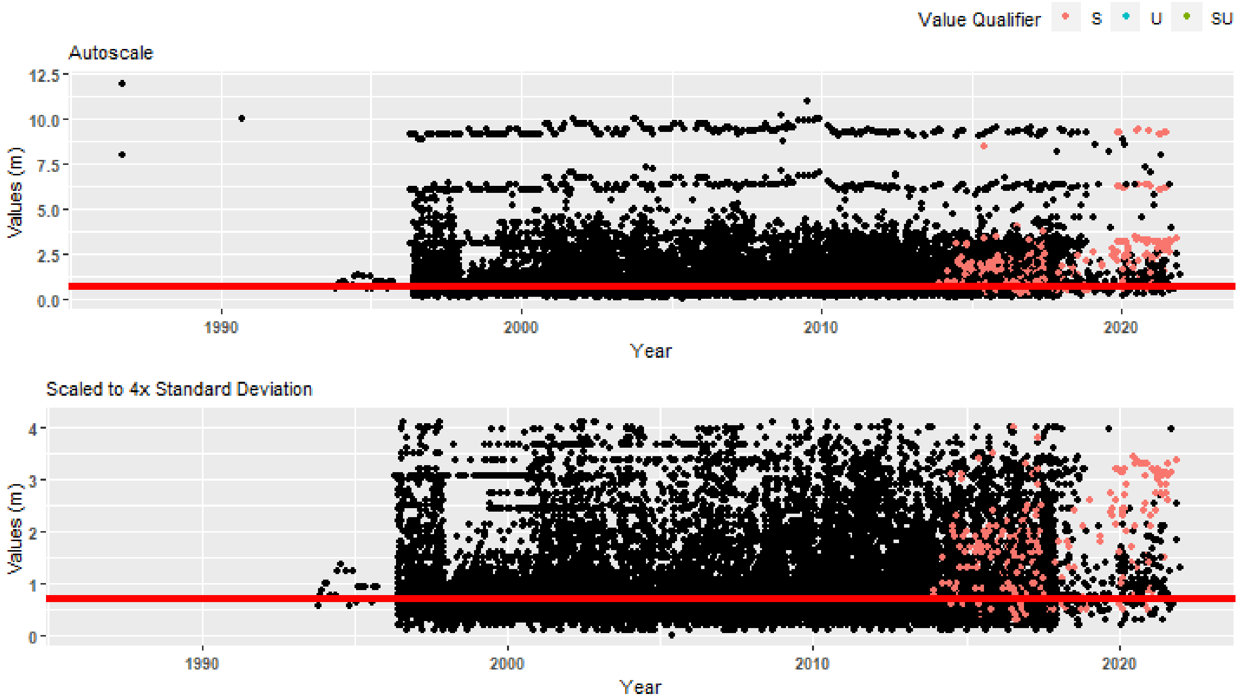
Data Points with Trendlines for Banana River Aquatic Preserve

Senn Slope = 0, Senn Intercept = 0.8
Trend = -1, tau = -0.0258, p = 0.0003
Linear Trendline: y = -0.000842237931283848x + 2.55244778585619



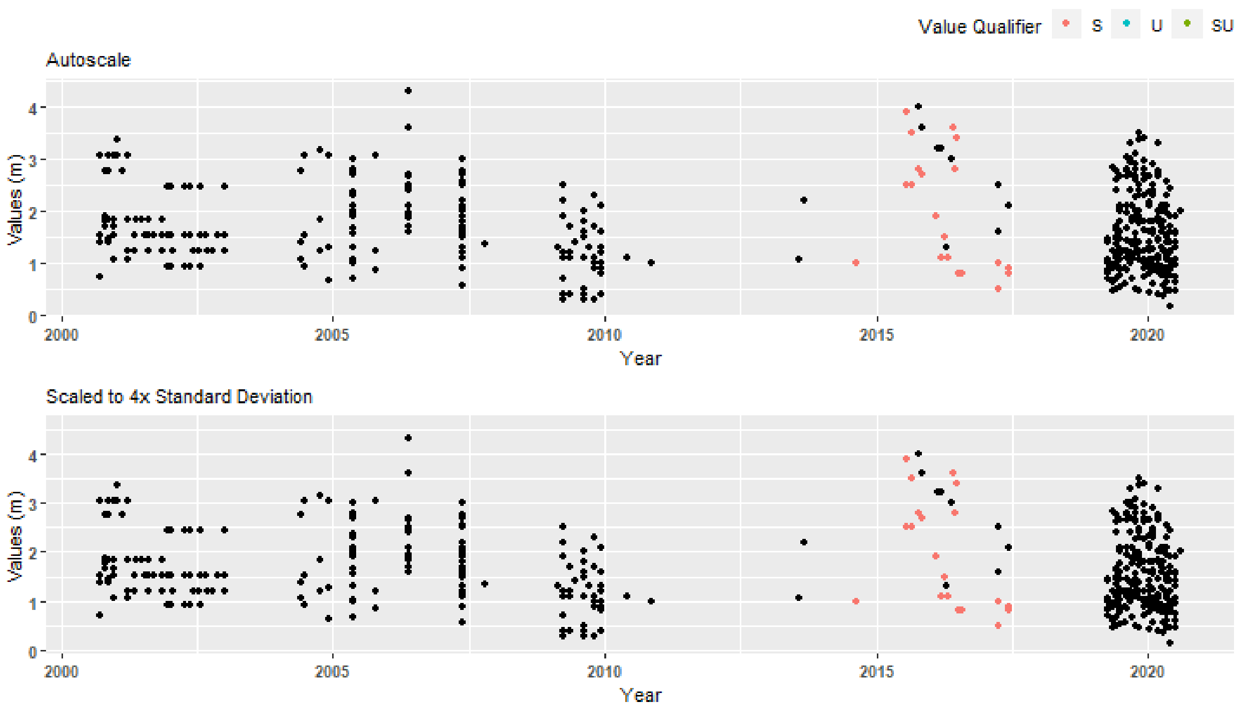
Data Points with Trendlines for Big Bend Seagrasses Aquatic Preserve

Senn Slope = 0,   Senn Intercept = 0.7
Trend = -1,   tau = 0.0184,   p = 0
Linear Trendline: y = 0.0000443411152385459x + 1.06047273986278



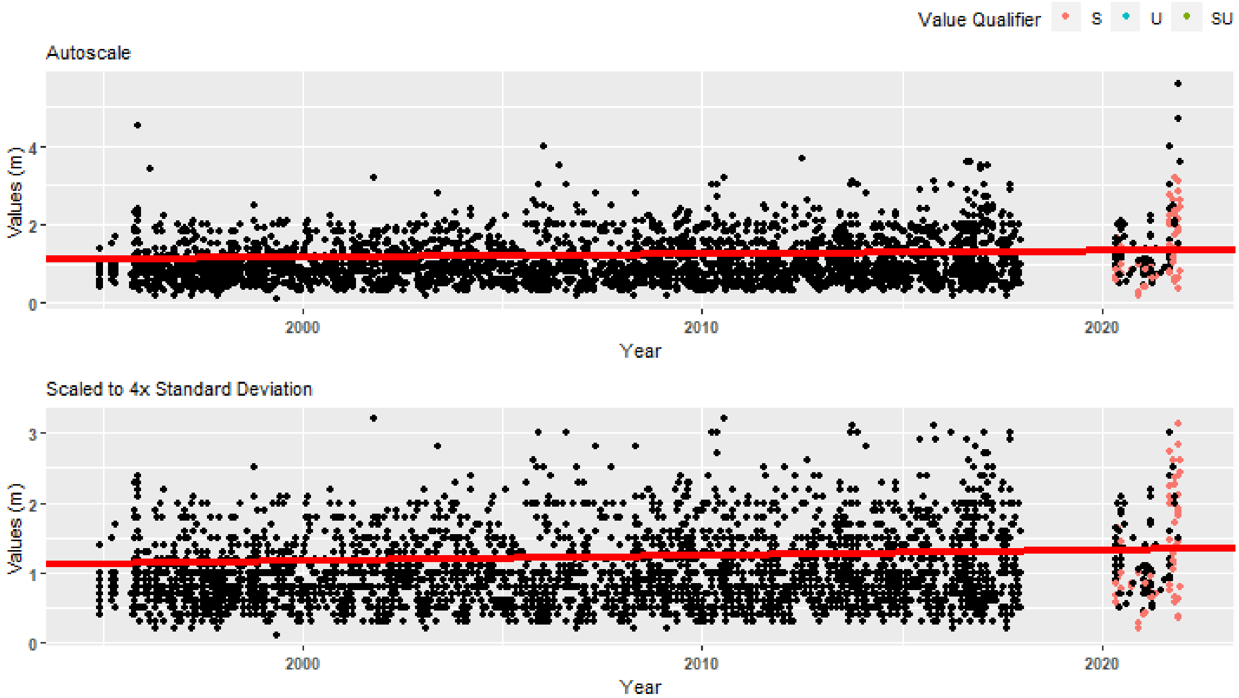
Data Points with Trendlines for Biscayne Bay Aquatic Preserve

Senn Slope = -0.0317666666666667,   Senn Intercept = 20.1233879071827
Trend = -1,   tau = -0.1274,   p = 0
Linear Trendline: y = -0.0191966044633891x + 40.2387699500989



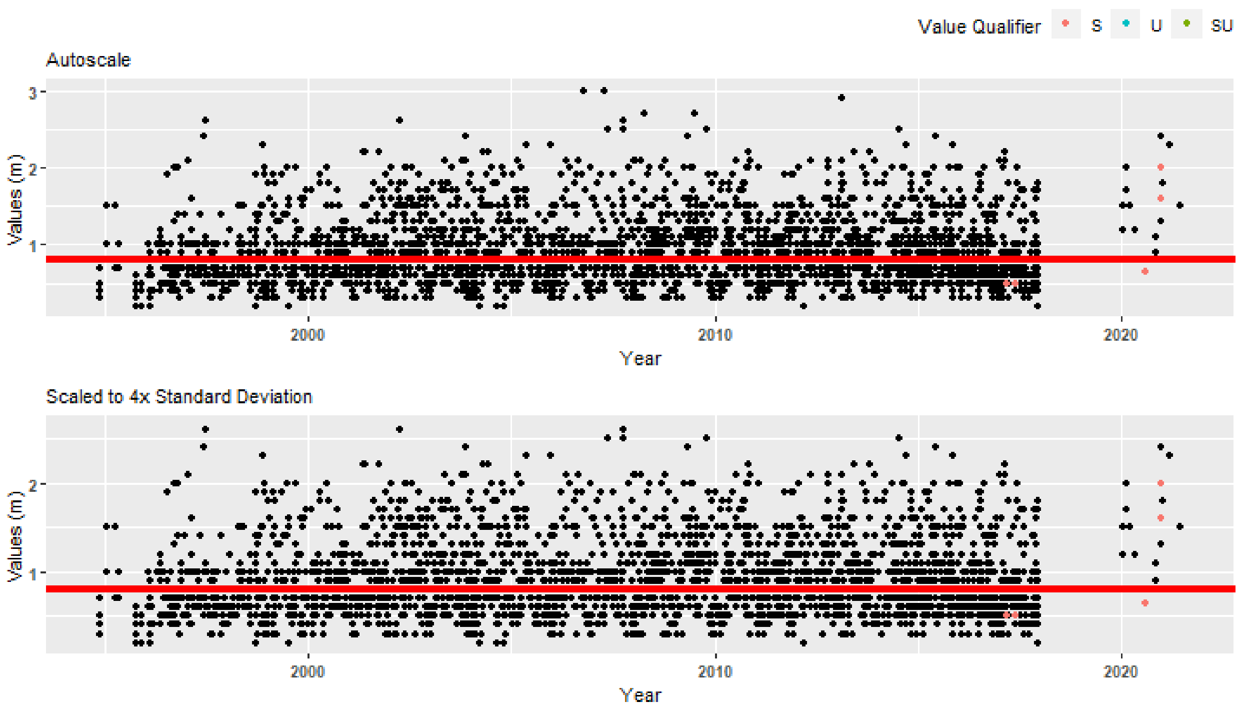
Data Points with Trendlines for Boca Ciega Bay Aquatic Preserve

Senn Slope = 0.00769230769230769,   Senn Intercept = -14.211
Trend = 1,   tau = 0.0861,   p = 0
Linear Trendline: y = 0.0146175673899761x + -28.3045974789245



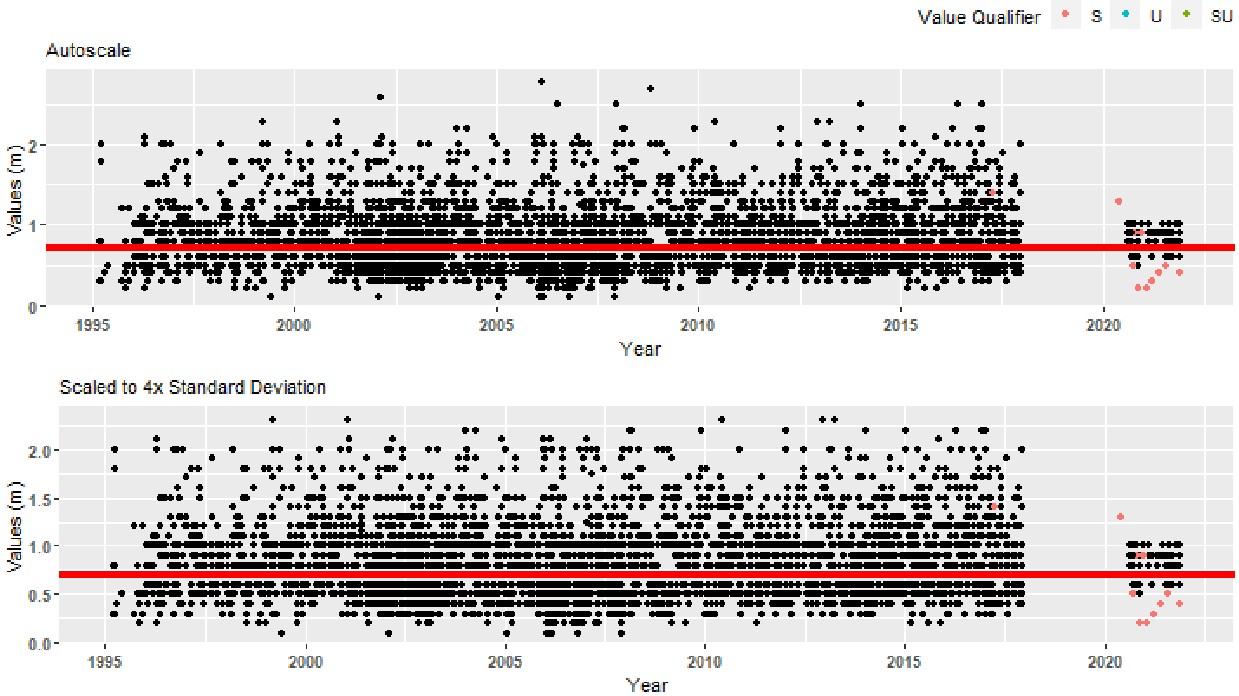
Data Points with Trendlines for Cape Haze Aquatic Preserve

Senn Slope = 0,   Senn Intercept = 0.8
Trend = 0,   tau = 0.0047,   p = 0.6892
Linear Trendline: y = 0.00124303635836936x + -1.56659804984407



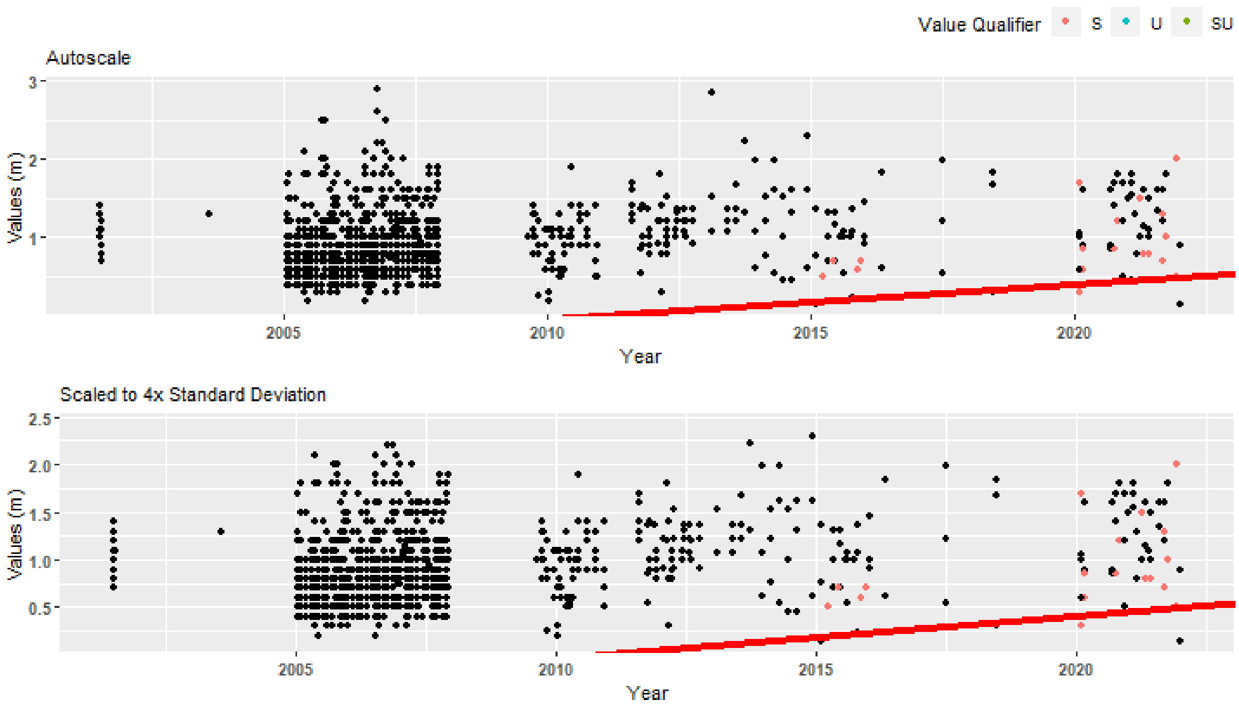
Data Points with Trendlines for Cockroach Bay Aquatic Preserve

Senn Slope = 0,   Senn Intercept = 0.7
Trend = -1,   tau = 0.0739,   p = 0
Linear Trendline: y = 0.0070526273080718x + -13.3355371314318



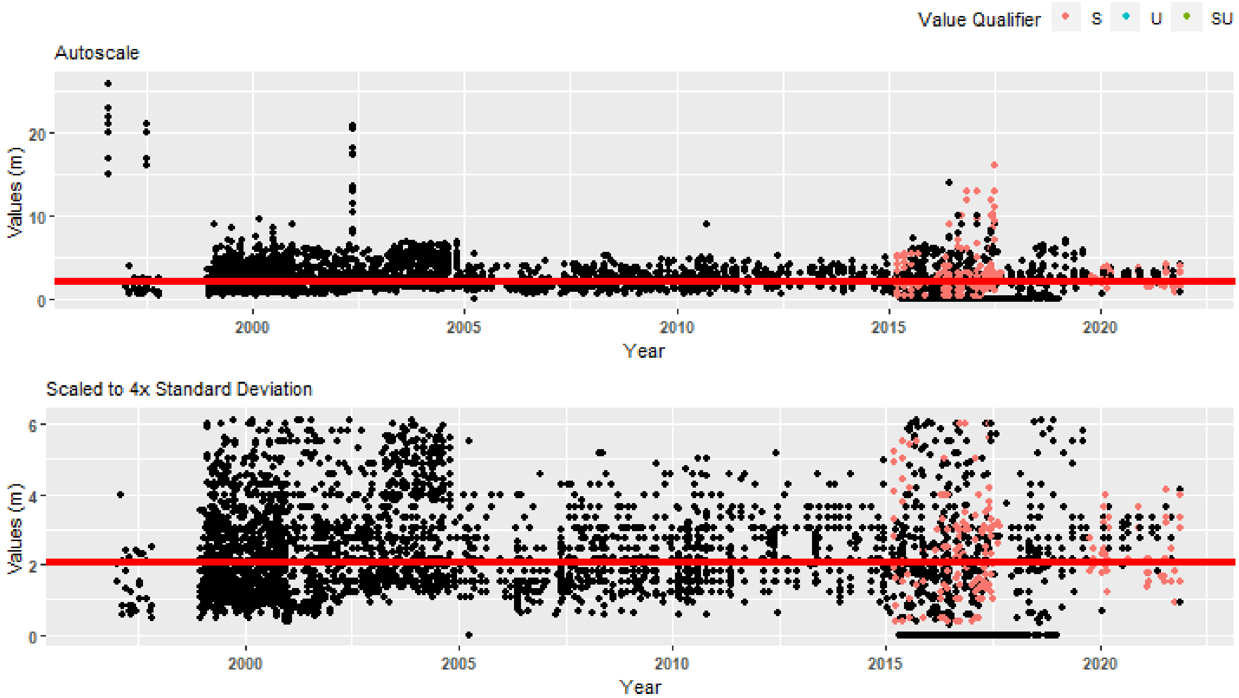
Data Points with Trendlines for Estero Bay Aquatic Preserve

Senn Slope = 0.0444688283333333,   Senn Intercept = -89.42
Trend = 1,   tau = 0.1527,   p = 0
Linear Trendline: y = 0.0245480617944246x + -48.3624994615316



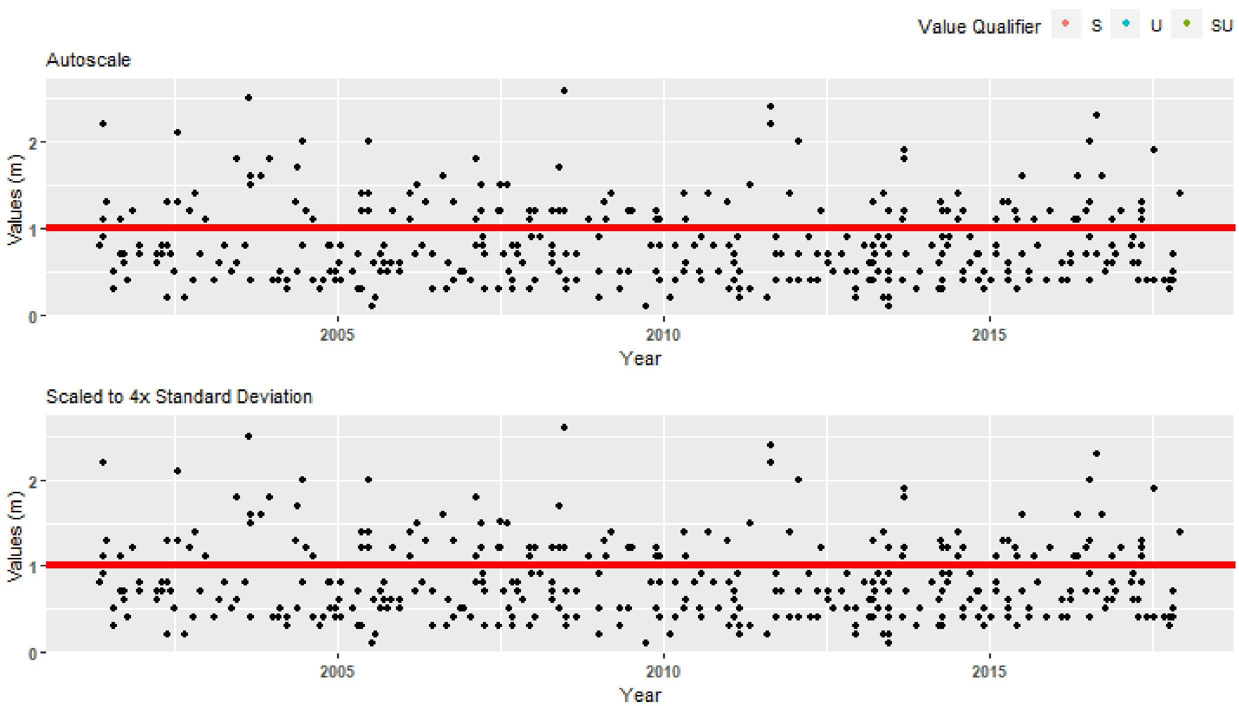
Data Points with Trendlines for Florida Keys National Marine Sanctuary

Senn Slope = 0, Senn Intercept = 2.06681297
Trend = 0, tau = 0.0053, p = 0.1459
Linear Trendline: y = -0.020244216911024x + 43.0823694211575



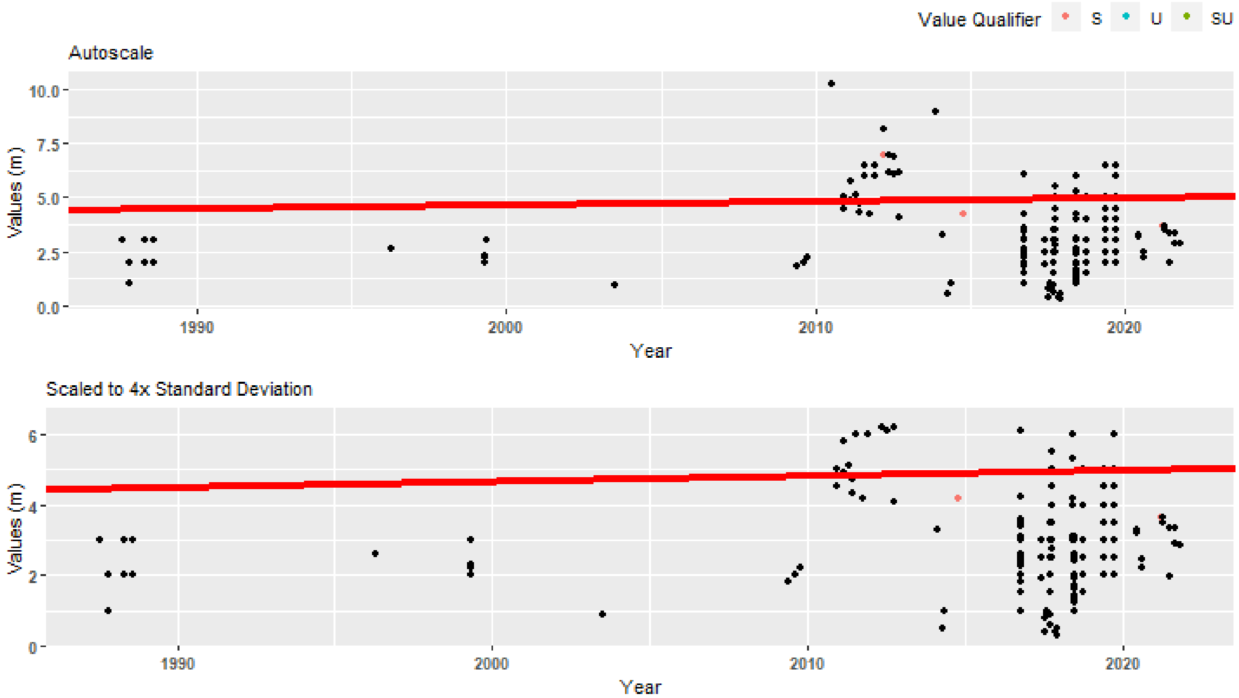
Data Points with Trendlines for Fort Clinch State Park Aquatic Preserve

Senn Slope = 0, Senn Intercept = 1
Trend = 0, tau = -0.028, p = 0.2636
Linear Trendline: y = -0.00613460225989455x + 13.2103726275615



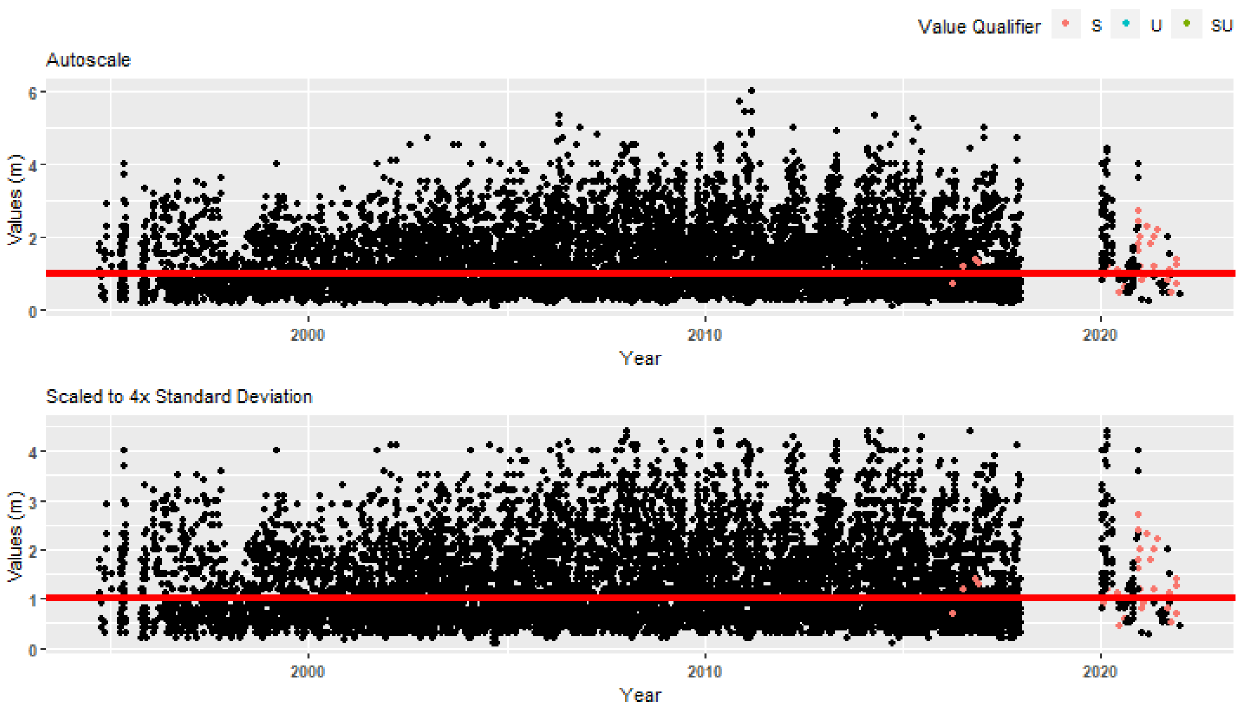
Data Points with Trendlines for Fort Pickens State Park Aquatic Preserve

Senn Slope = 0.0161290322580645, Senn Intercept = -27.5939133333333
Trend = 0, tau = 0.0159, p = 0.2758
Linear Trendline: y = 0.00514012601779392x + -7.32635477314295



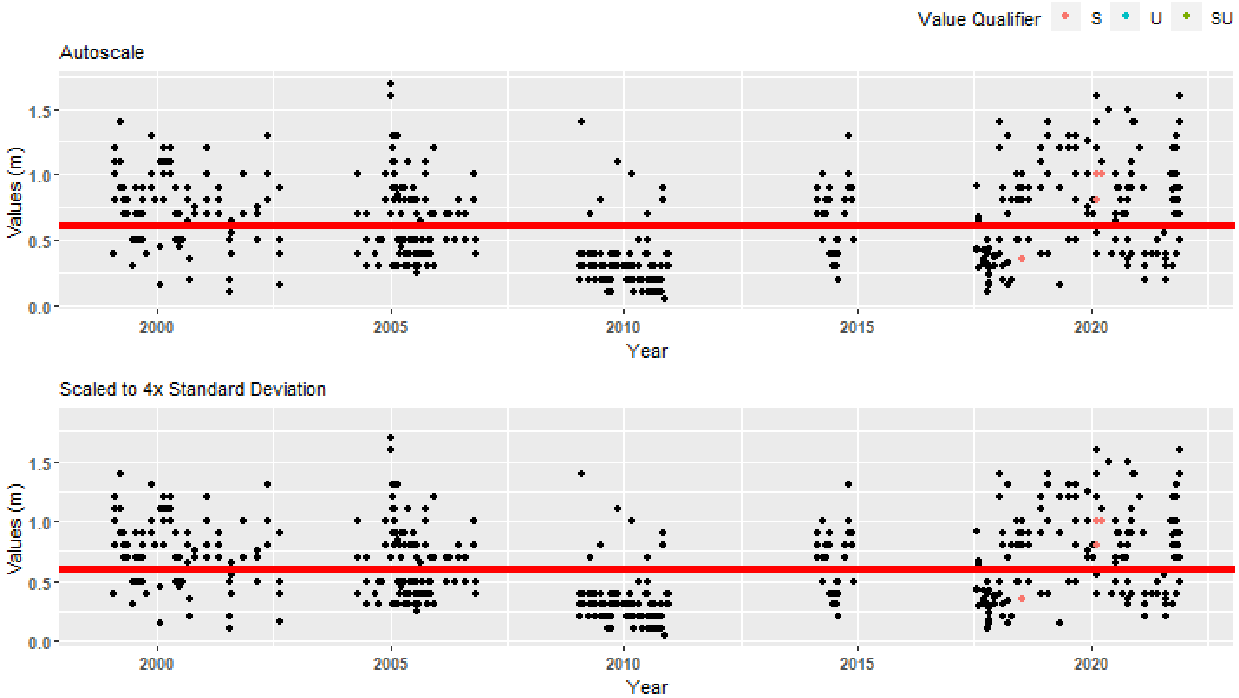
Data Points with Trendlines for Gasparilla Sound-Charlotte Harbor Aquatic Preserve

Senn Slope = 0, Senn Intercept = 1
Trend = 0, tau = -0.0025, p = 0.4217
Linear Trendline: y = 0.00442133824022264x + -7.61577881111404



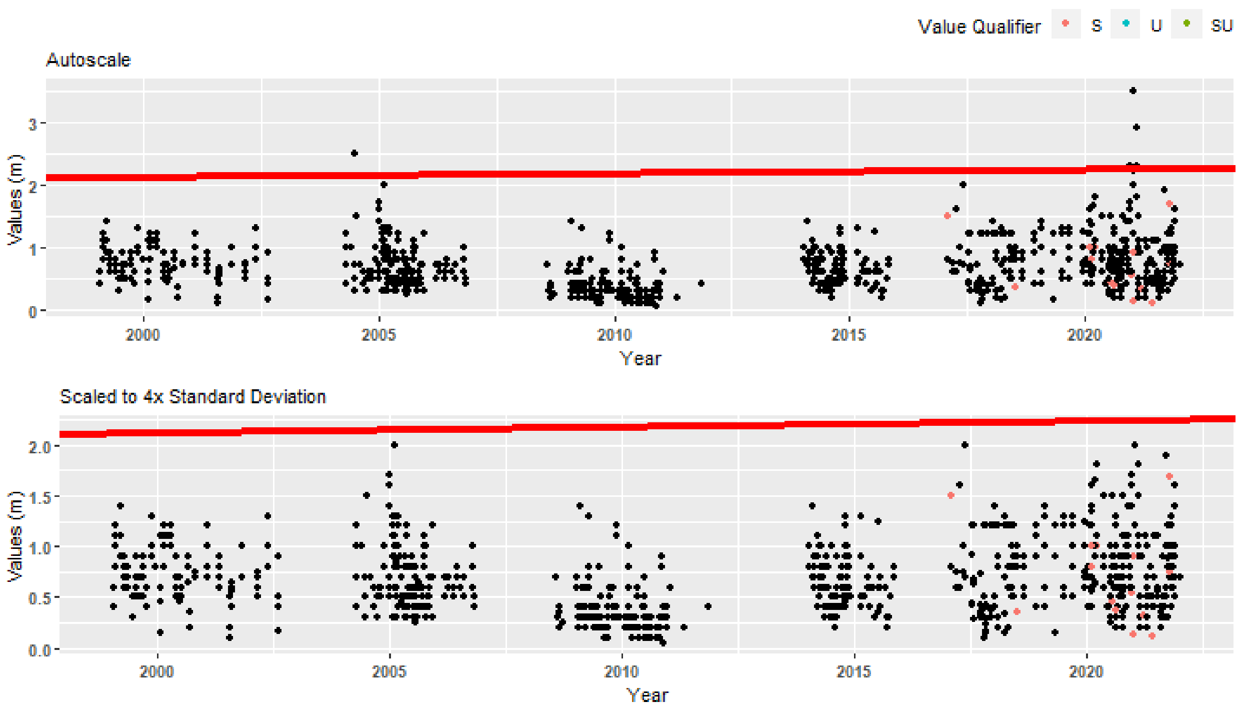
Data Points with Trendlines for Guana River Marsh Aquatic Preserve

Senn Slope = 0,   Senn Intercept = 0.6
Trend = 0,   tau = 0.018,   p = 0.5417
Linear Trendline: y = 0.00210473384914424x + -3.64956728303718



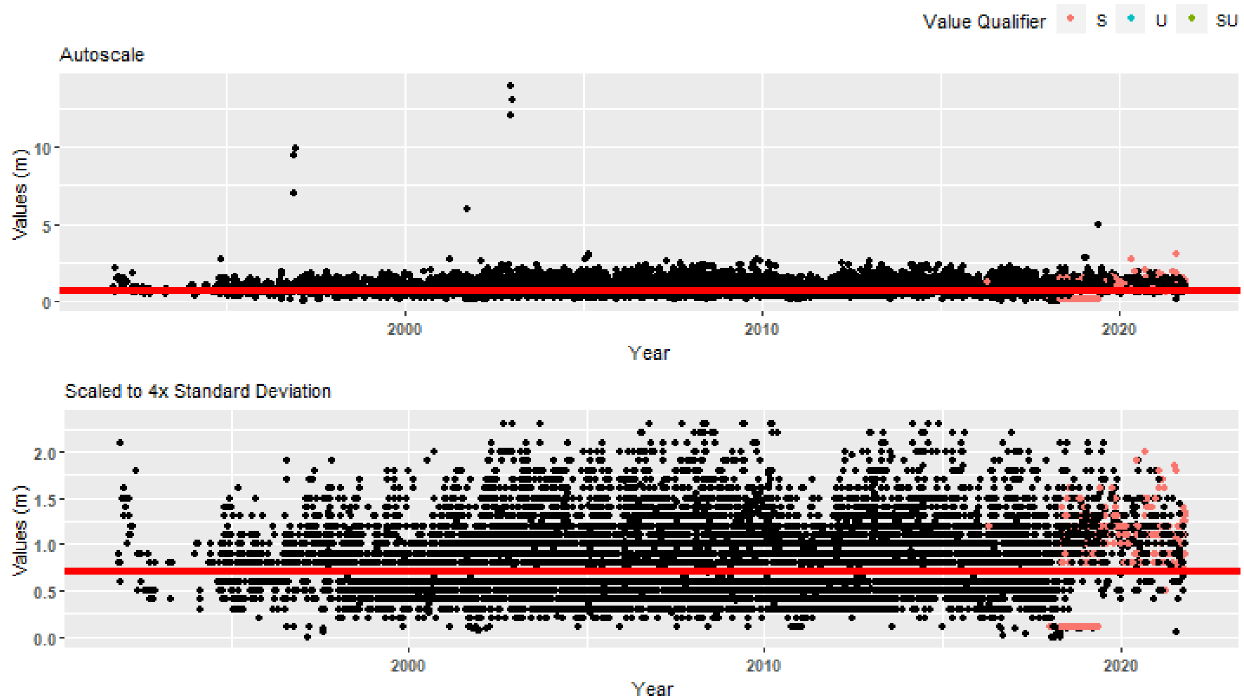
Data Points with Trendlines for Guana Tolomato Matanzas National Estuarine Research Reserve

Senn Slope = 0.00588235294117647,   Senn Intercept = -9.64750000000001
Trend = 1,   tau = 0.0893,   p = 0
Linear Trendline: y = 0.00736313711178493x + -14.1788149137756



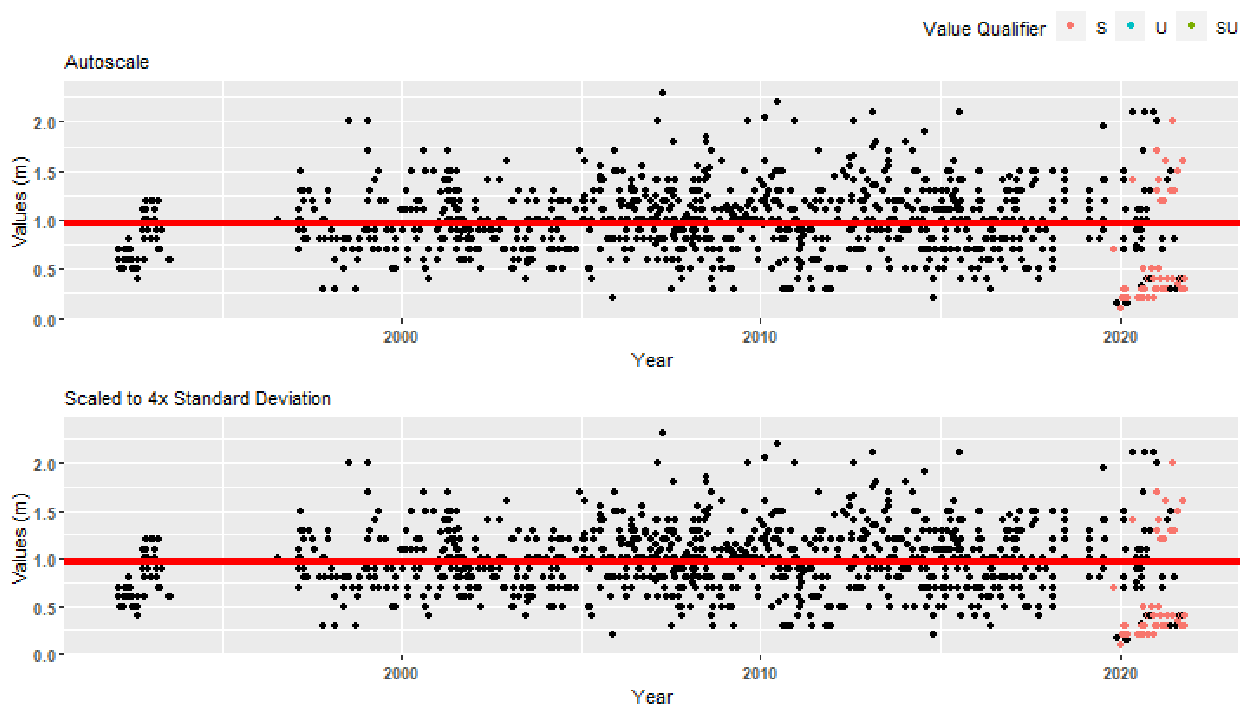
Data Points with Trendlines for Indian River-Malabar to Vero Beach Aquatic Preserve

Senn Slope = 0, Senn Intercept = 0.7
Trend = -1, tau = 0.0558, p = 0
Linear Trendline: y = 0.00454472363182281x + -8.30163905037541



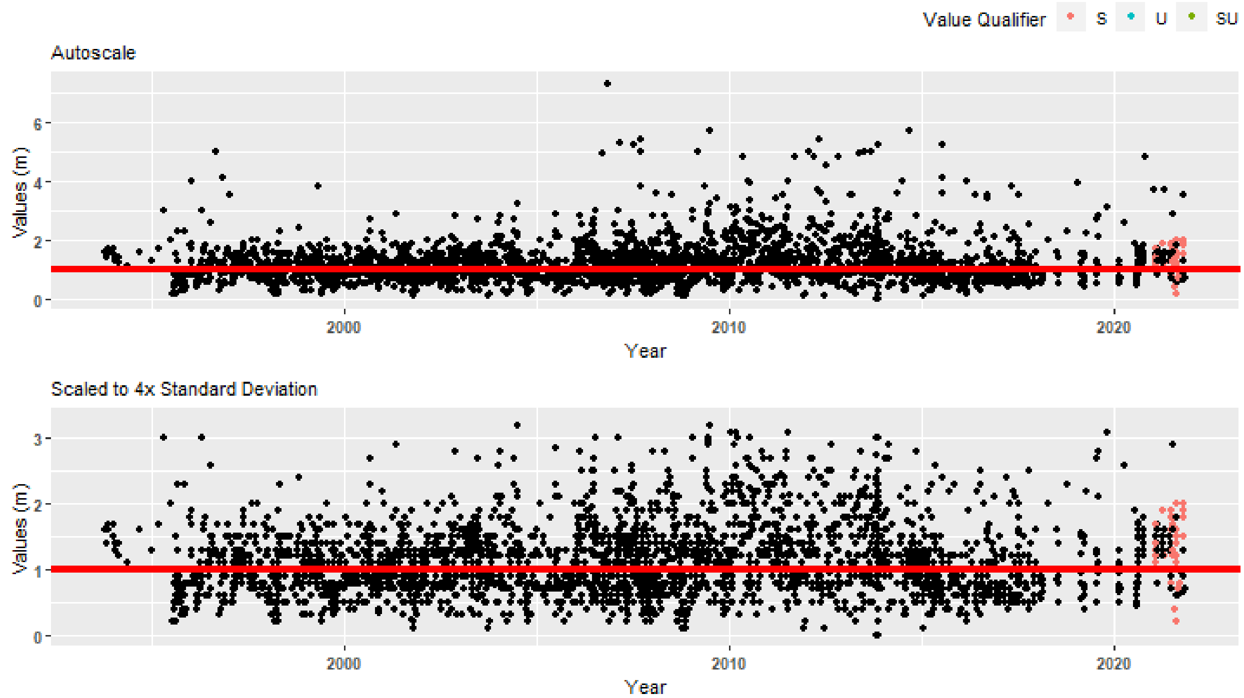
Data Points with Trendlines for Indian River-Vero Beach to Ft. Pierce Aquatic Preserve

Senn Slope = 0, Senn Intercept = 0.9625
Trend = 0, tau = -0.0007, p = 0.4214
Linear Trendline: y = -0.000186756395143288x + 1.31941558846201



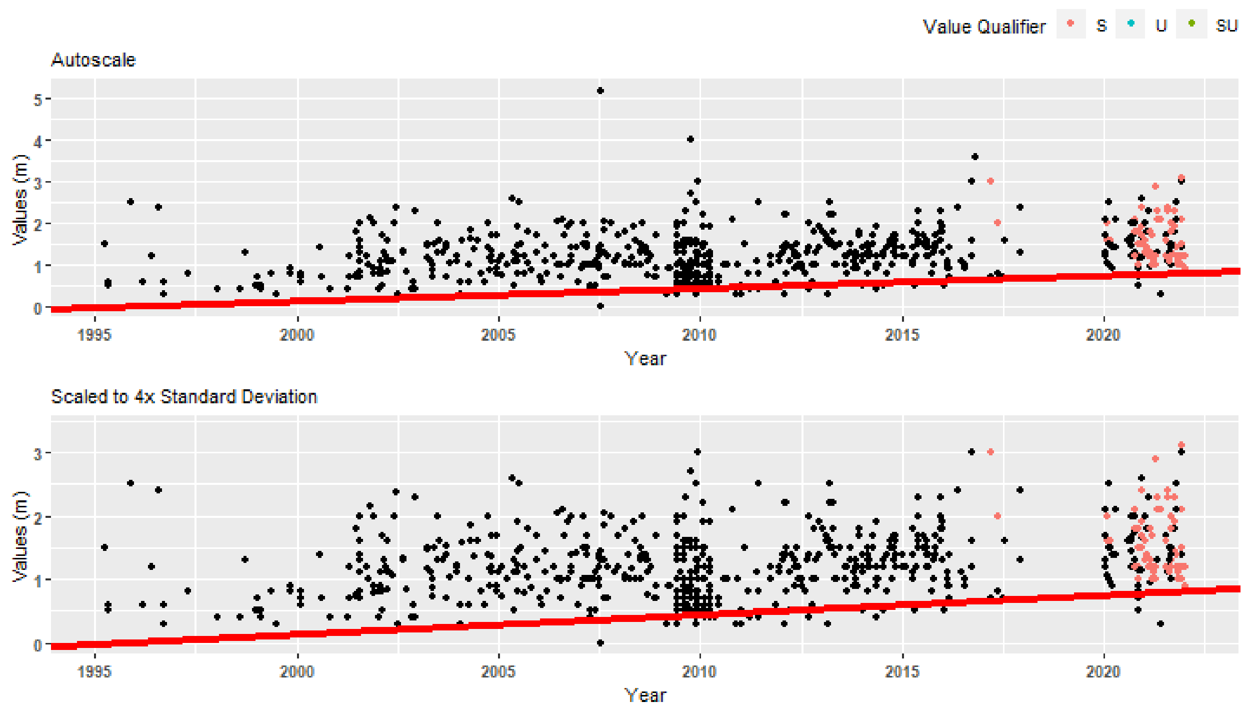
Data Points with Trendlines for Jensen Beach to Jupiter Inlet Aquatic Preserve

Senn Slope = 0,   Senn Intercept = 1
Trend = 0,   tau = 0.0135,   p = 0.1381
Linear Trendline: y = 0.00625881836749657x + -11.4389013224943



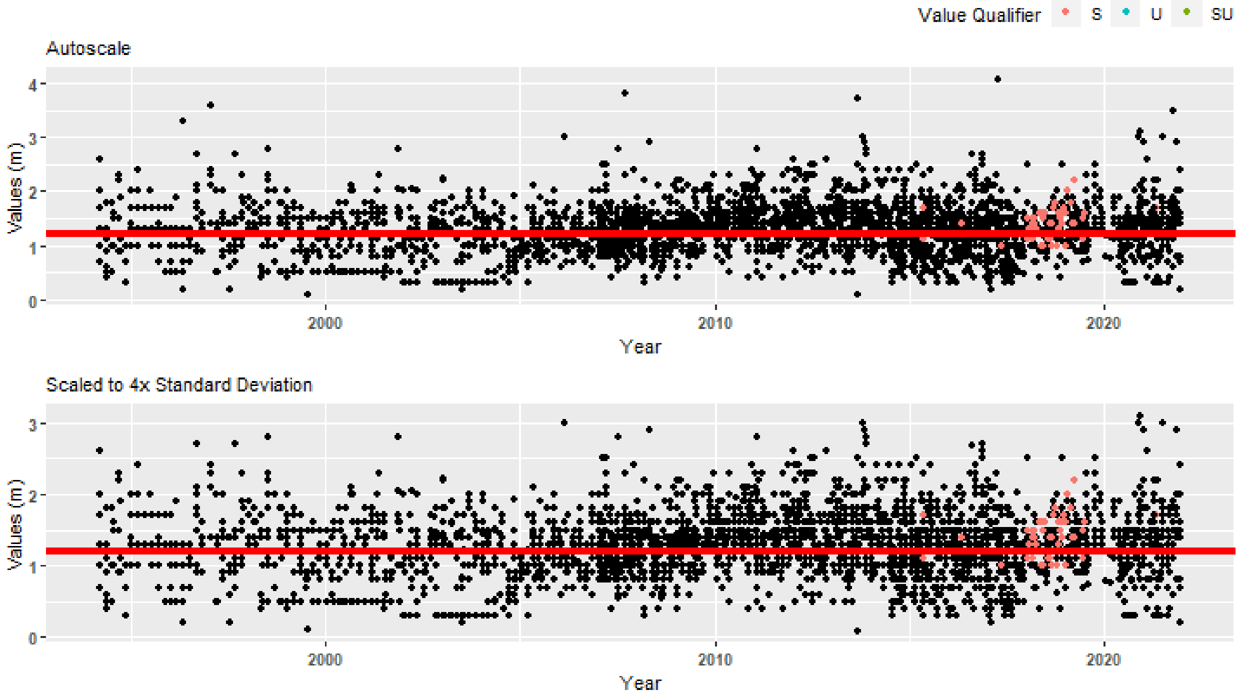
Data Points with Trendlines for Lemon Bay Aquatic Preserve

Senn Slope = 0.0307692307692308,   Senn Intercept = -61.3963636363636
Trend = 1,   tau = 0.1828,   p = 0
Linear Trendline: y = 0.0261194106579497x + -51.3149554843349



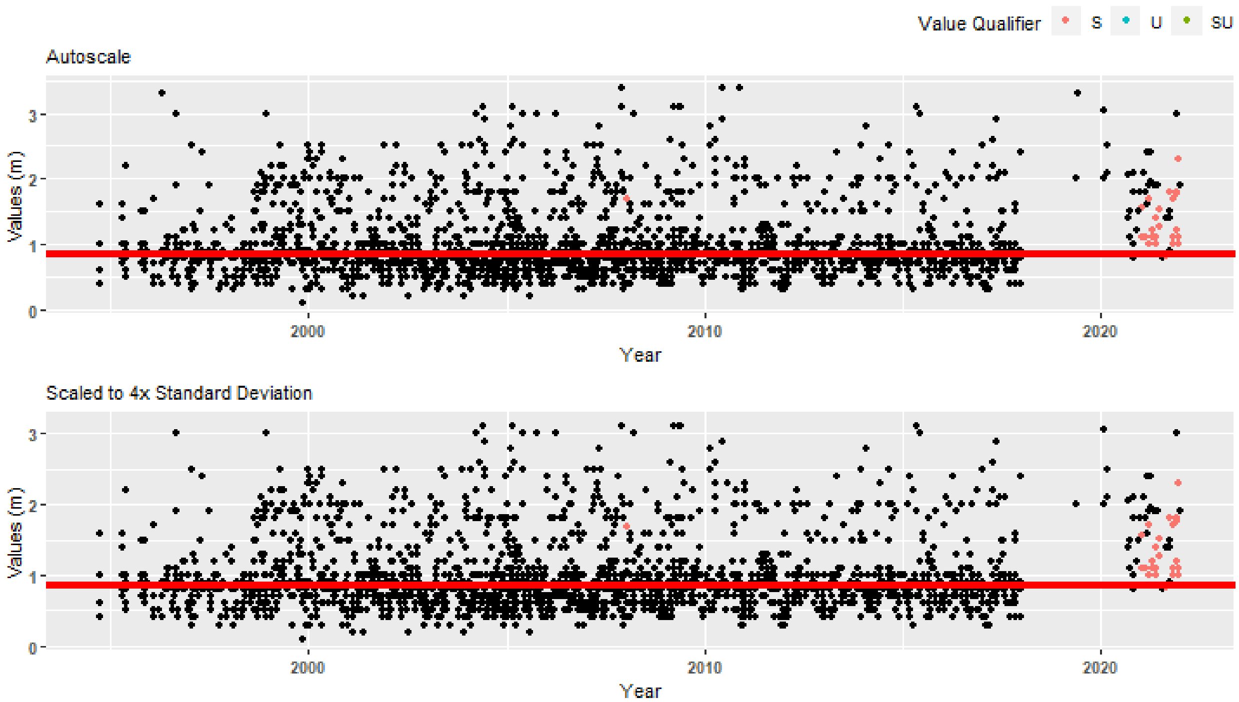
Data Points with Trendlines for Loxahatchee River-Lake Worth Creek Aquatic Preserve

Senn Slope = 0, Senn Intercept = 1.209607415
Trend = -1, tau = 0.03, p = 0.0005
Linear Trendline: y = 0.00353055777240741x + -5.84581027341538



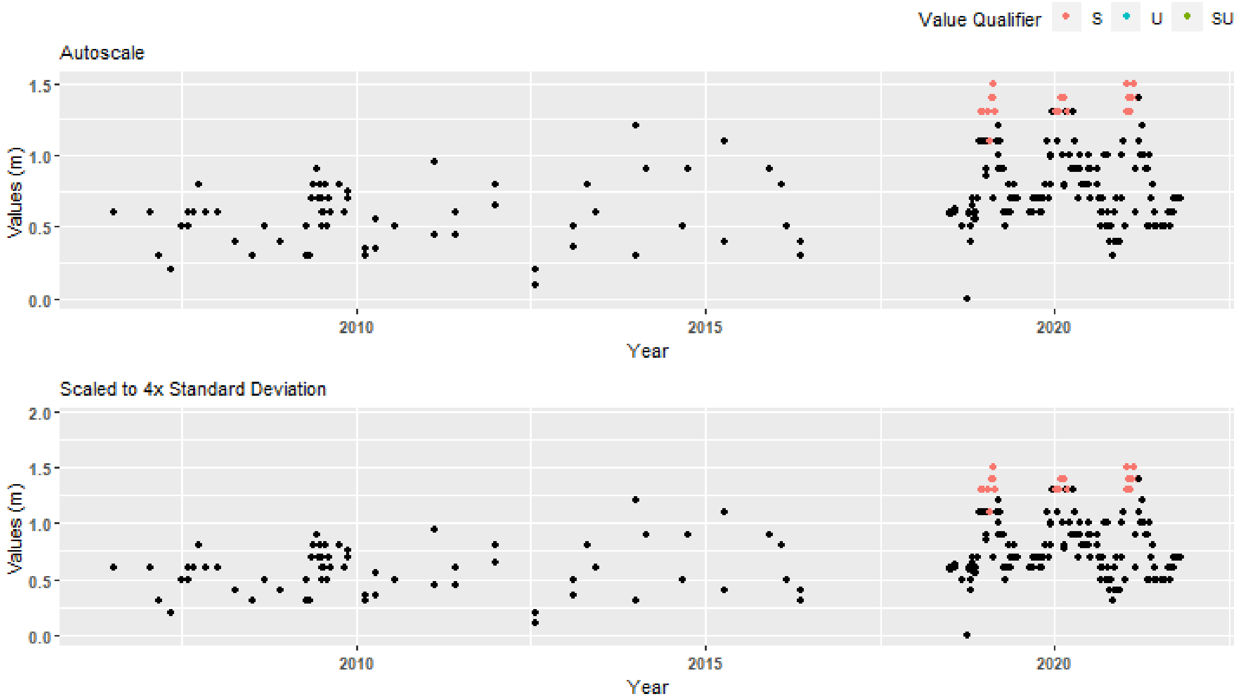
Data Points with Trendlines for Matlacha Pass Aquatic Preserve

Senn Slope = 0, Senn Intercept = 0.85
Trend = -1, tau = 0.0462, p = 0
Linear Trendline: y = 0.00477743465118122x + -8.59548086990689



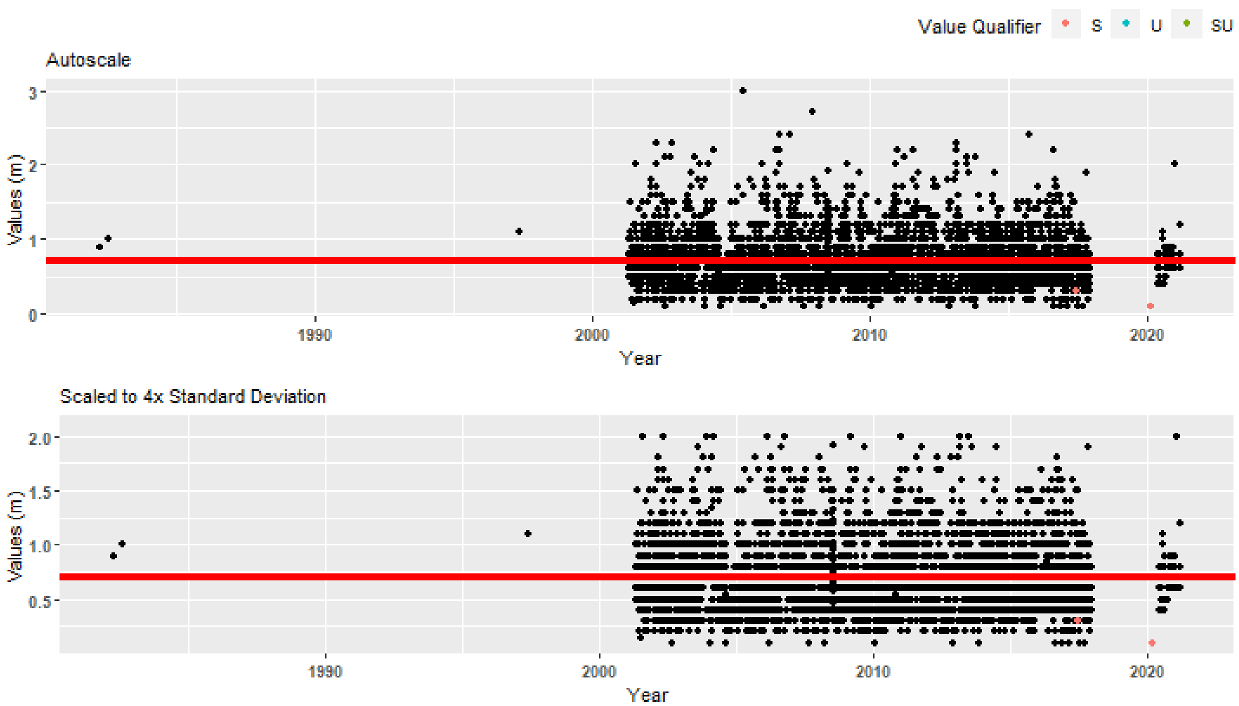
Data Points with Trendlines for Mosquito Lagoon Aquatic Preserve

Senn Slope = 0.0111111111111111, Senn Intercept = -7.16788461538461
Trend = 1, tau = 0.1948, p = 0
Linear Trendline: y = 0.0231274257745496x + -45.8969238102385



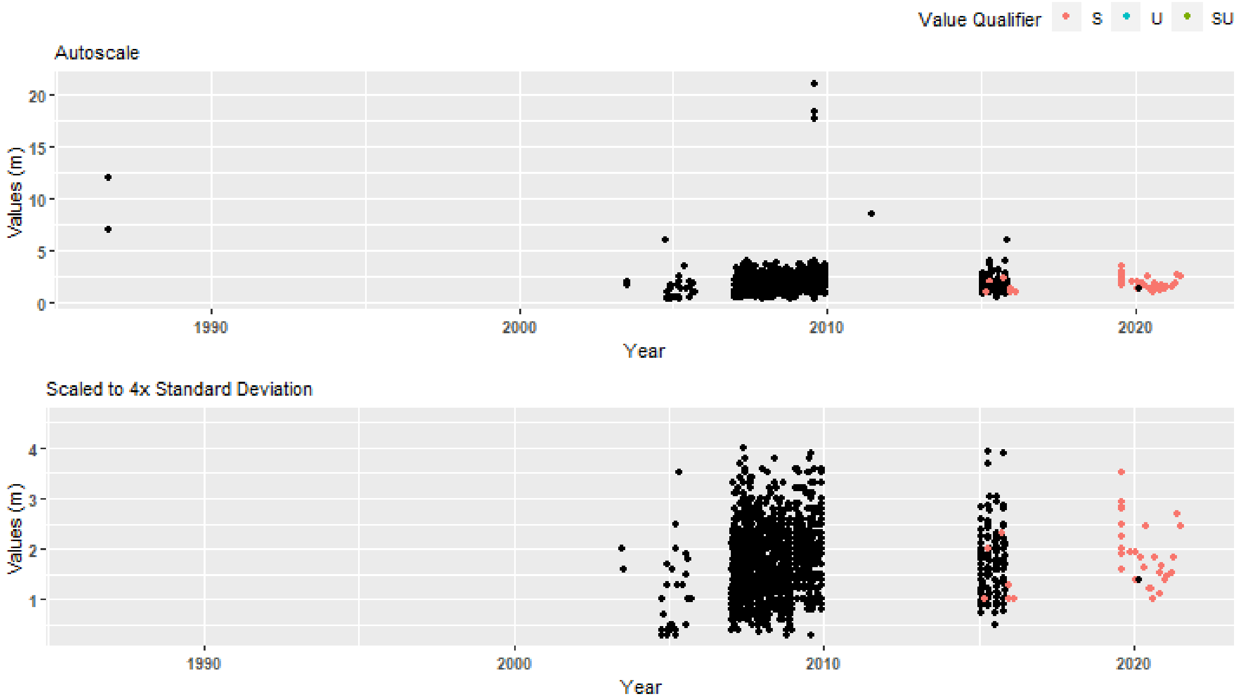
Data Points with Trendlines for Nassau River-St. Johns River Marshes Aquatic Preserve

Senn Slope = 0, Senn Intercept = 0.7
Trend = 0, tau = -0.0124, p = 0.102
Linear Trendline: y = -0.00177779638762175x + 4.2897174883583



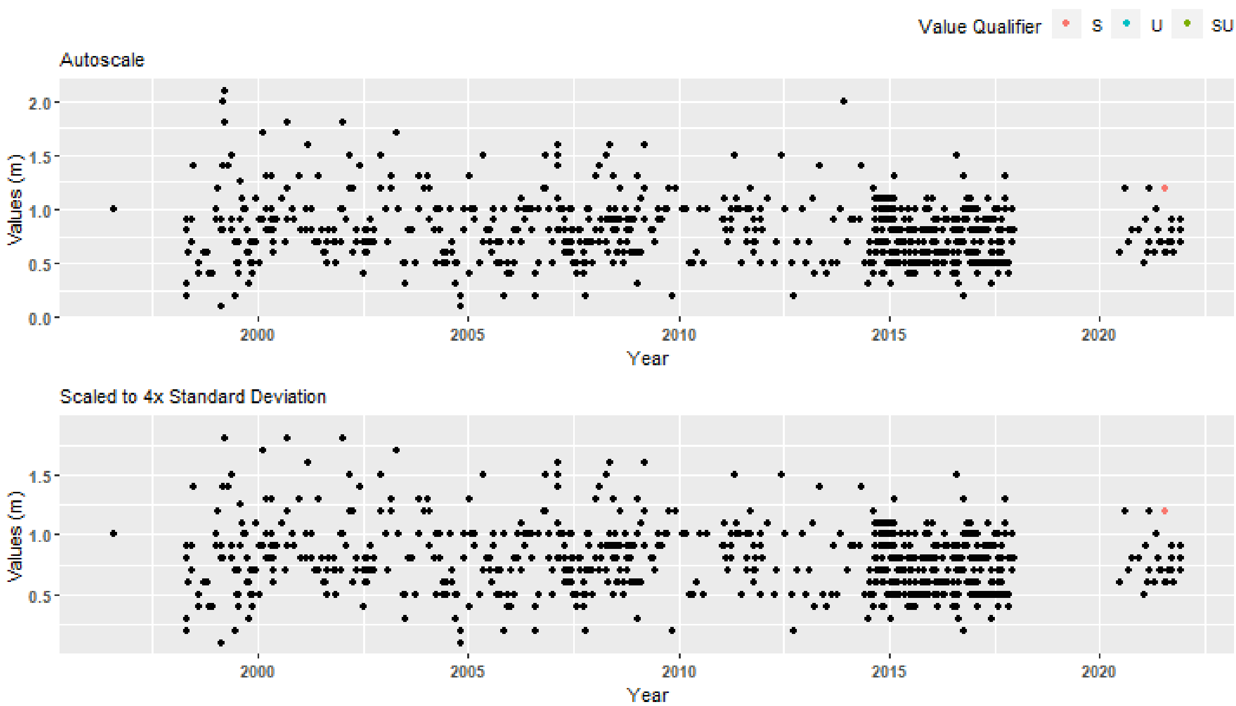
Data Points with Trendlines for Nature Coast Aquatic Preserve

Senn Slope = 0.0381818181818182, Senn Intercept = -92.6466523076921
Trend = 1, tau = 0.0809, p = 0
Linear Trendline: y = -0.00808069011289821x + 18.0612688869089



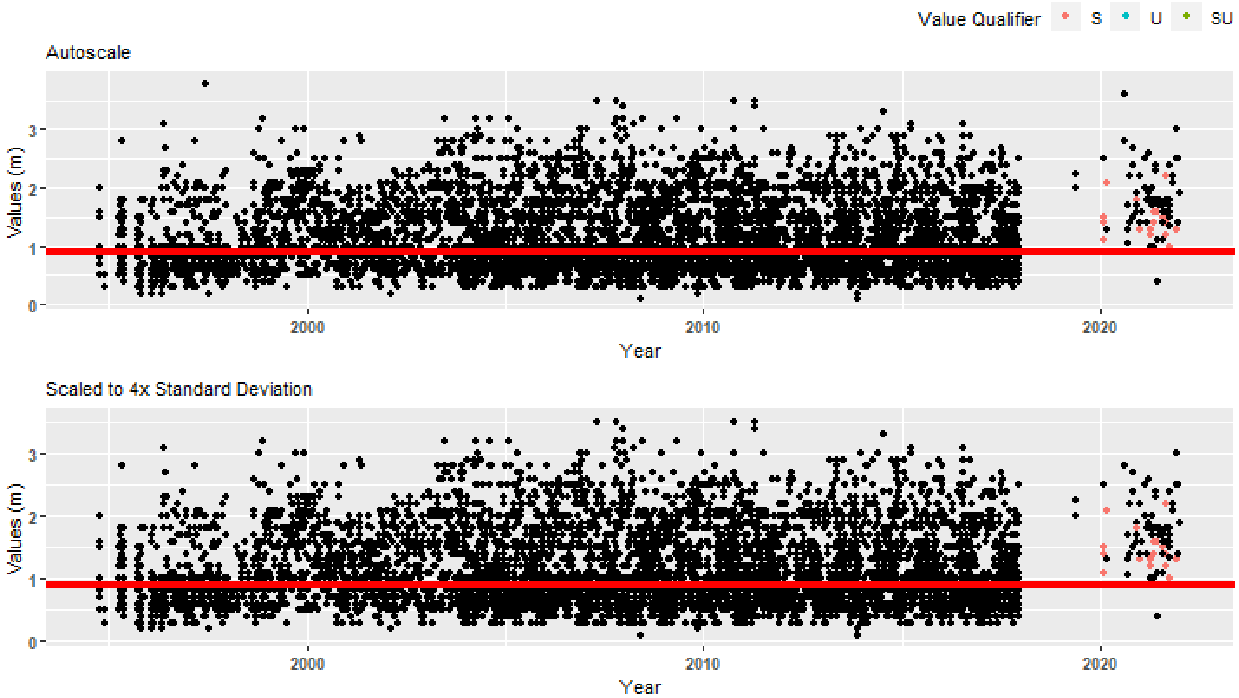
Data Points with Trendlines for North Fork St. Lucie Aquatic Preserve

Senn Slope = 0, Senn Intercept = 6.38611111111111
Trend = -1, tau = -0.0848, p = 0.0002
Linear Trendline: y = -0.00620429152189867x + 13.266157100693



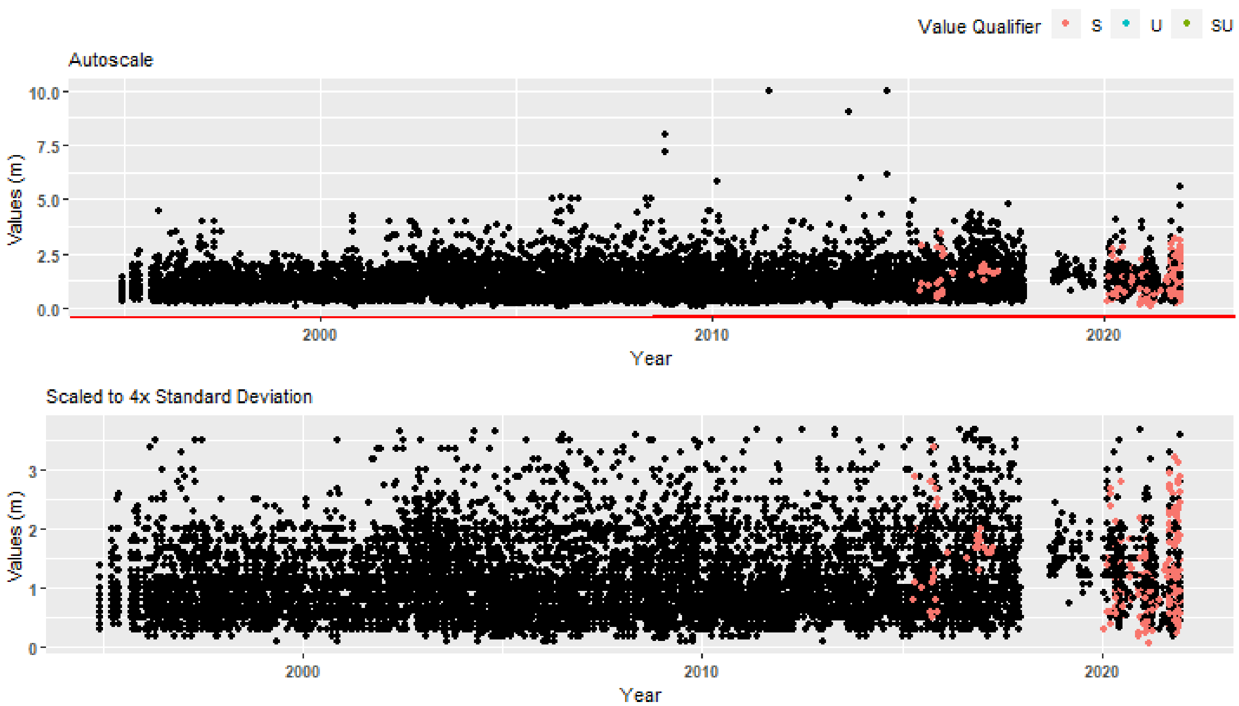
Data Points with Trendlines for Pine Island Sound Aquatic Preserve

Senn Slope = 0,   Senn Intercept = 0.9
Trend = -1,   tau = 0.0159,   p = 0.0027
Linear Trendline: y = 0.0032527229805786x + -5.35191503166282



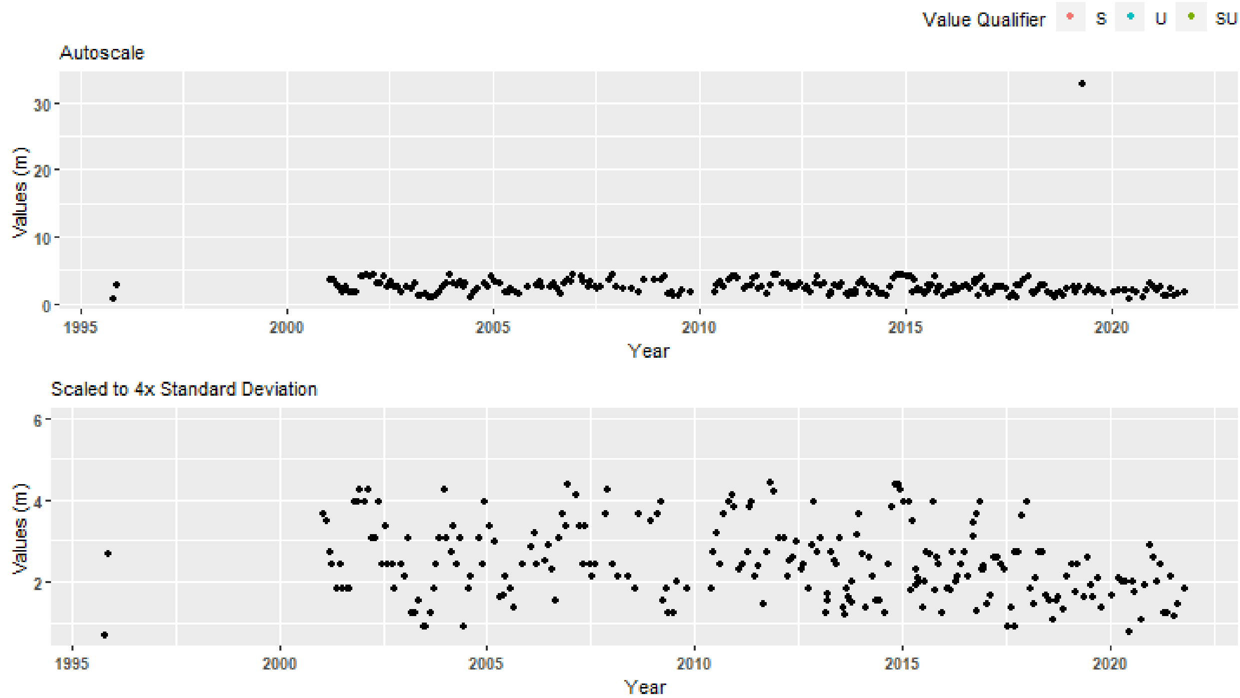
Data Points with Trendlines for Pinellas County Aquatic Preserve

Senn Slope = 0.00555555555555556,   Senn Intercept = -11.6517647058823
Trend = 1,   tau = 0.0618,   p = 0
Linear Trendline: y = 0.013023723432215x + -25.0073587276863



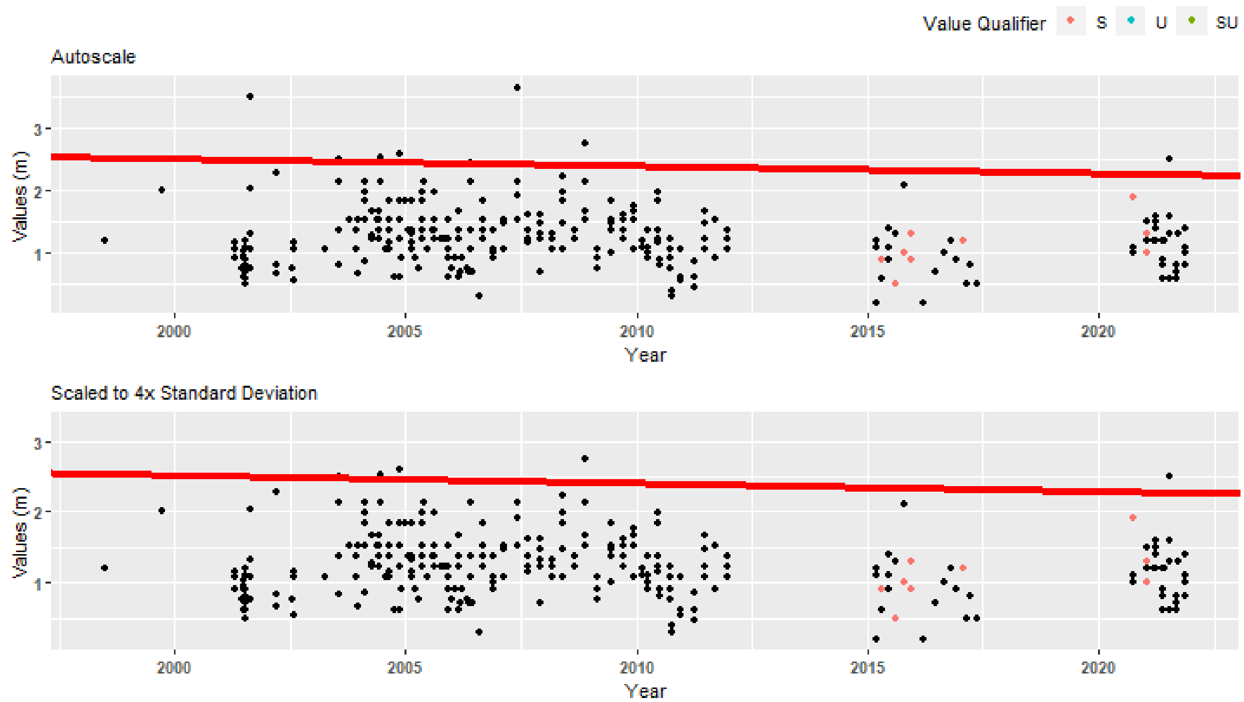
Data Points with Trendlines for Rocky Bayou State Park Aquatic Preserve

Senn Slope = -0.0254, Senn Intercept = 39.9702487488889
Trend = -1, tau = -0.1321, p = 0.0003
Linear Trendline: y = -0.00803296515658958x + 18.6961410823995



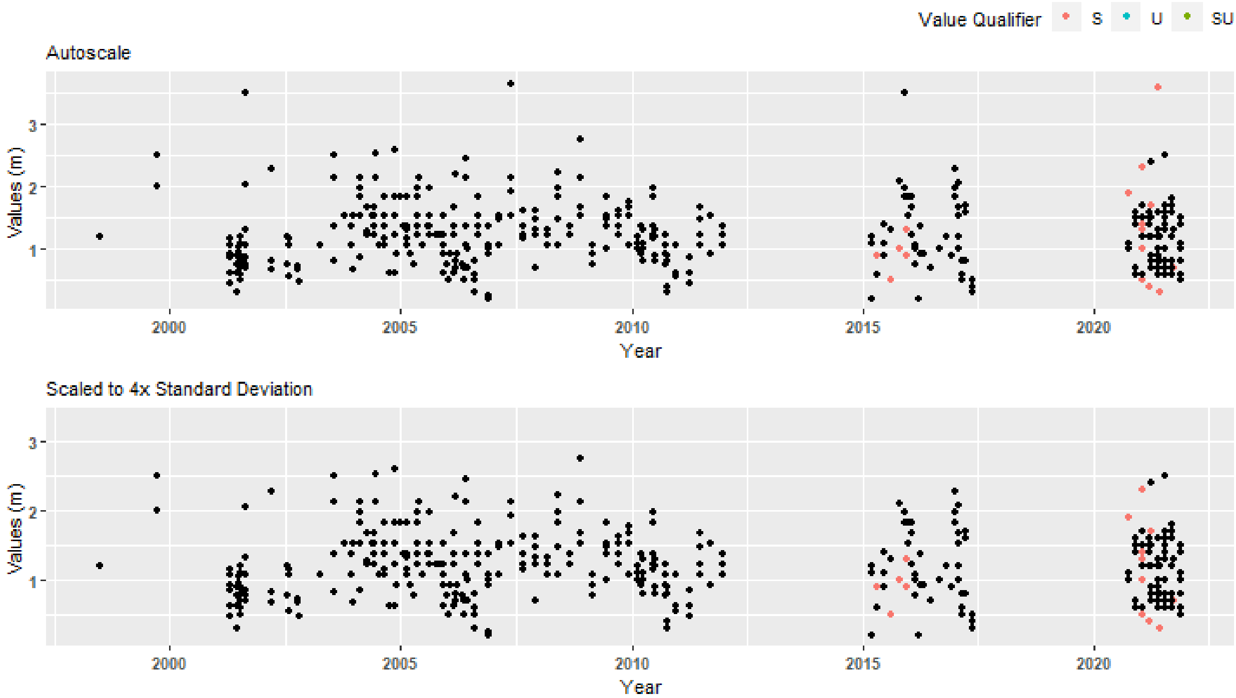
Data Points with Trendlines for Rookery Bay Aquatic Preserve

Senn Slope = -0.011441112, Senn Intercept = 25.391796243052
Trend = -1, tau = -0.0947, p = 0.0339
Linear Trendline: y = -0.0121758923654509x + 25.6911722814603



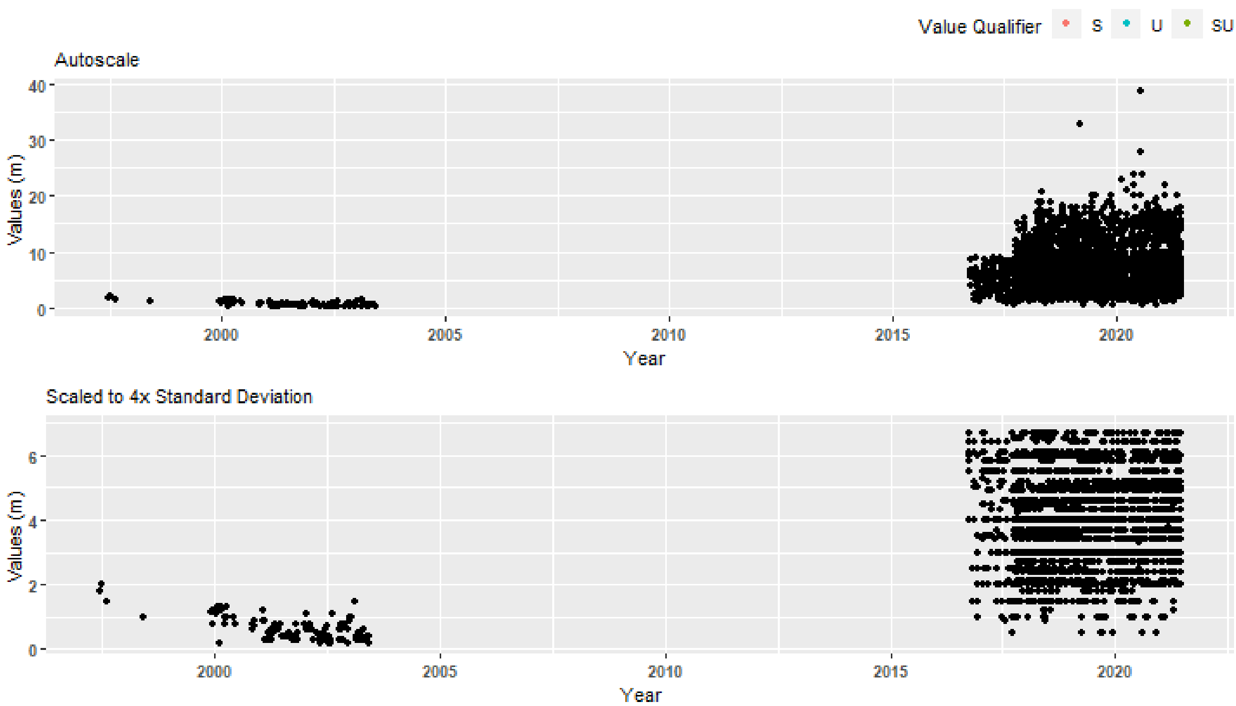
Data Points with Trendlines for Rookery Bay National Estuarine Research Reserve

Senn Slope = -0.00184757923076923, Senn Intercept = 1.259607415
Trend = 0, tau = -0.0383, p = 0.2254
Linear Trendline: y = -0.00180541811035784x + 4.82943644626888



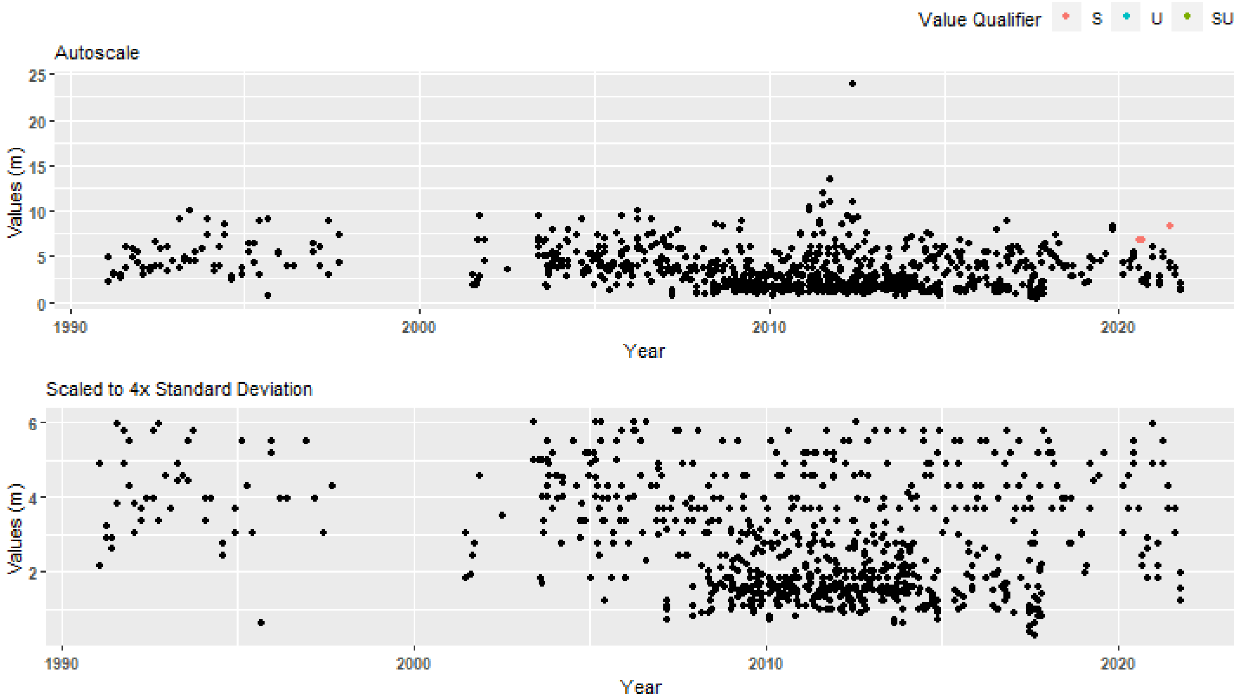
Data Points with Trendlines for Southeast Florida Coral Reef Ecosystem Conservation Area

Senn Slope = 0.3, Senn Intercept = -653.45507518797
Trend = 1, tau = 0.1193, p = 0
Linear Trendline: y = 0.355731117169095x + -711.954216822515



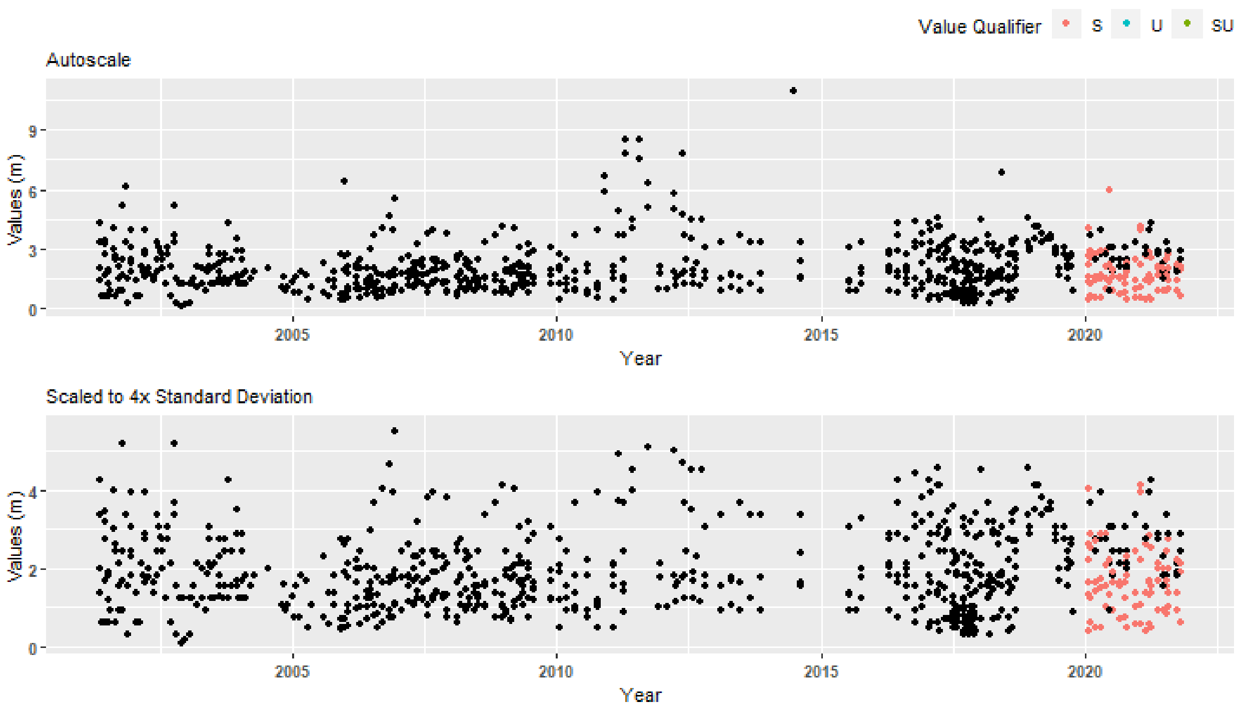
Data Points with Trendlines for St. Andrews State Park Aquatic Preserve

Senn Slope = -0.08,   Senn Intercept = 139.40686728
Trend = -1,   tau = -0.1656,   p = 0
Linear Trendline: y = -0.10005320279 2075x + 204.318812390955



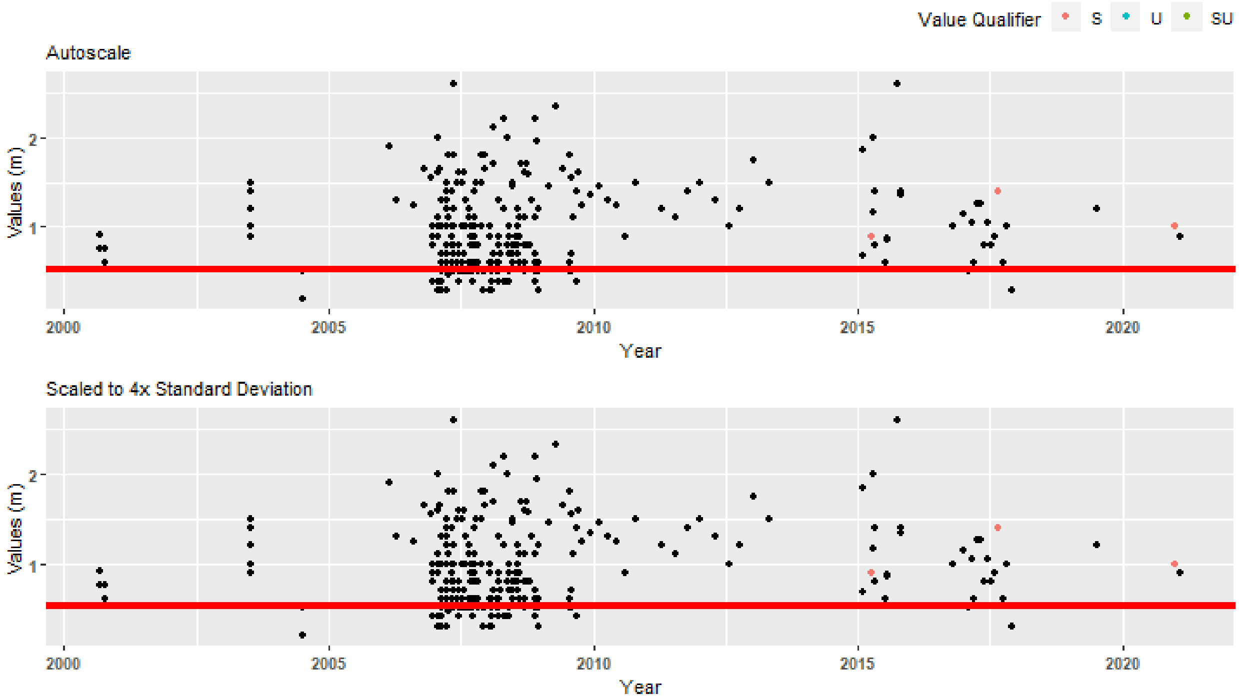
Data Points with Trendlines for St. Joseph Bay Aquatic Preserve

Senn Slope = -0.00831259272727272,   Senn Intercept = 1.9050221425
Trend = 0,   tau = -0.0228,   p = 0.1118
Linear Trendline: y = -0.00963864357226749x + 21.3028624645801



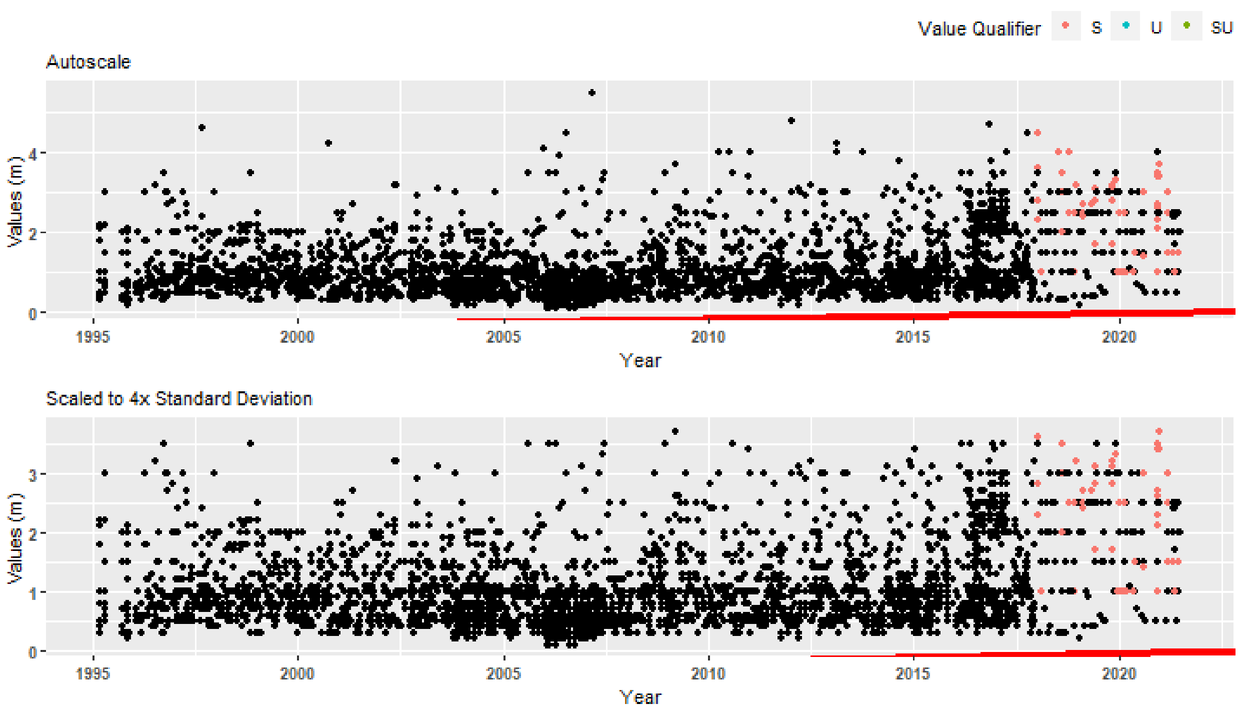
Data Points with Trendlines for St. Martins Marsh Aquatic Preserve

Senn Slope = 0,   Senn Intercept = 0.523757492142845
Trend = 0,   tau = 0.0081,   p = 0.8604
Linear Trendline: y = 0.0143539039012905x + -27.8882533102971



Data Points with Trendlines for Terra Ceia Aquatic Preserve

Senn Slope = 0.0142857142857143,   Senn Intercept = -28.8818681318681
Trend = 1,   tau = 0.1281,   p = 0
Linear Trendline: y = 0.0303734882117154x + -59.9880152112915



Appendix V: Managed Area Summary Box Plots

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

1. Use the data set that only has `Use_In_Analysis` of `TRUE` for the desired managed area
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 `ManagedAreaName` with data grouped by `Year`, then `Year` and `Month`, then finally `Month` only. Each managed 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 + 5 time the standard deviation
3. Y-axis set to be mean + 5 time the standard deviation for most recent 10 years of data

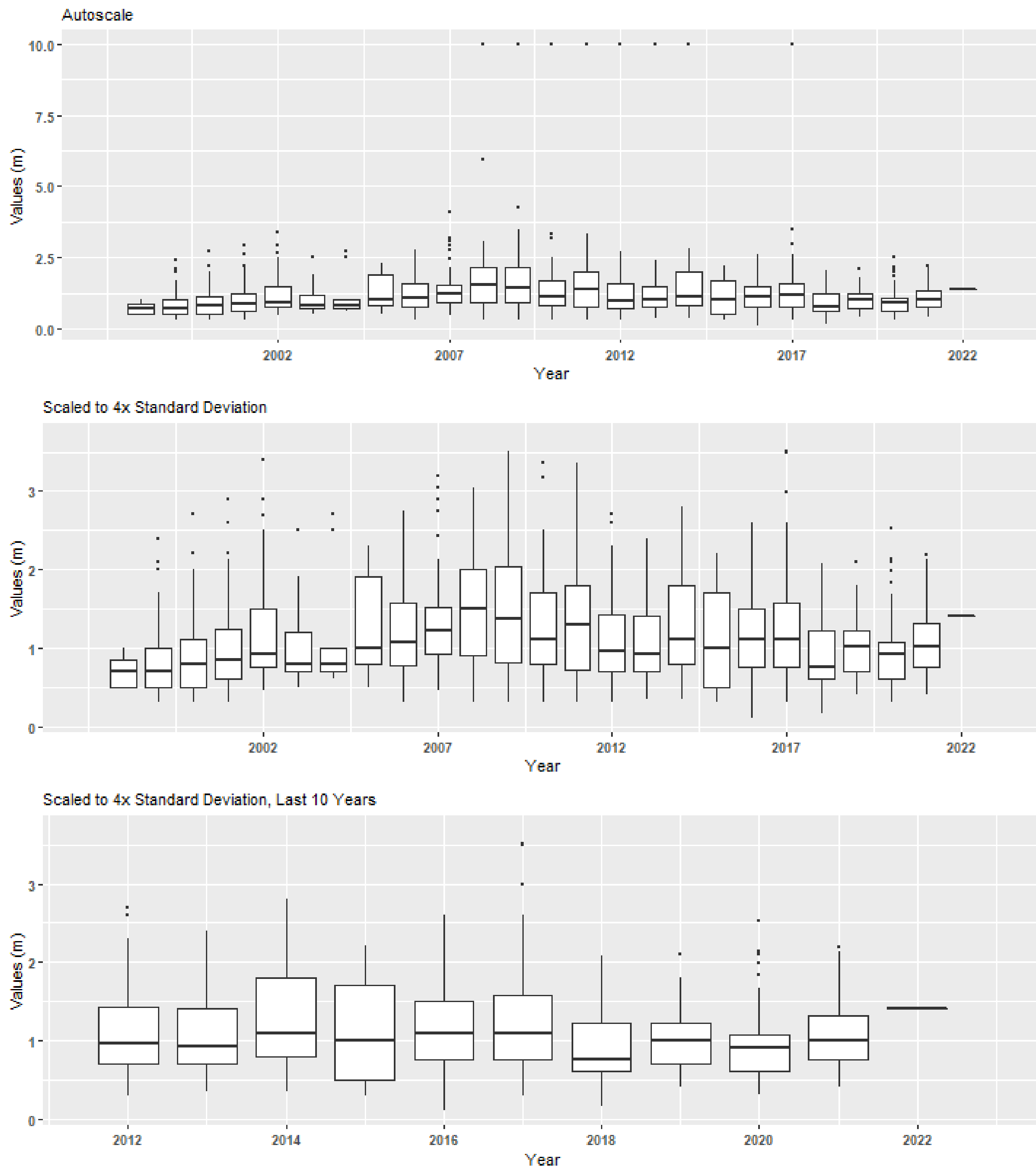
```
if(n==0){
   print("There are no managed areas that qualify.")
} else {
   for (i in 1:n) {
      year_lower <- min(data$Year[data$Use_In_Analysis == TRUE &
                                   data$ManagedAreaName == MA_names[i]])
      year_upper <- max(data$Year[data$Use_In_Analysis == TRUE &
 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

      ##Year plots
      p1 <- ggplot(data = data[data$Use_In_Analysis == TRUE &
                                 data$ManagedAreaName == MA_names[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$Use_In_Analysis == TRUE &
                              data$ManagedAreaName == MA_names[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$Use_In_Analysis == TRUE &
                              data$ManagedAreaName == MA_names[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_names[i]), 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$Use_In_Analysis == TRUE &
                              data$ManagedAreaName == MA_names[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$Use_In_Analysis == TRUE &
                             data$ManagedAreaName == MA_names[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$Use_In_Analysis == TRUE &
                             data$ManagedAreaName == MA_names[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_names[i]),
                       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$Use_In_Analysis == TRUE &
                             data$ManagedAreaName == MA_names[i], ],
             aes(x = Month, y = ResultValue,
                 group = Month, fill = as.factor(Month))) +
    geom_boxplot(outlier.size = 0.5) +
```

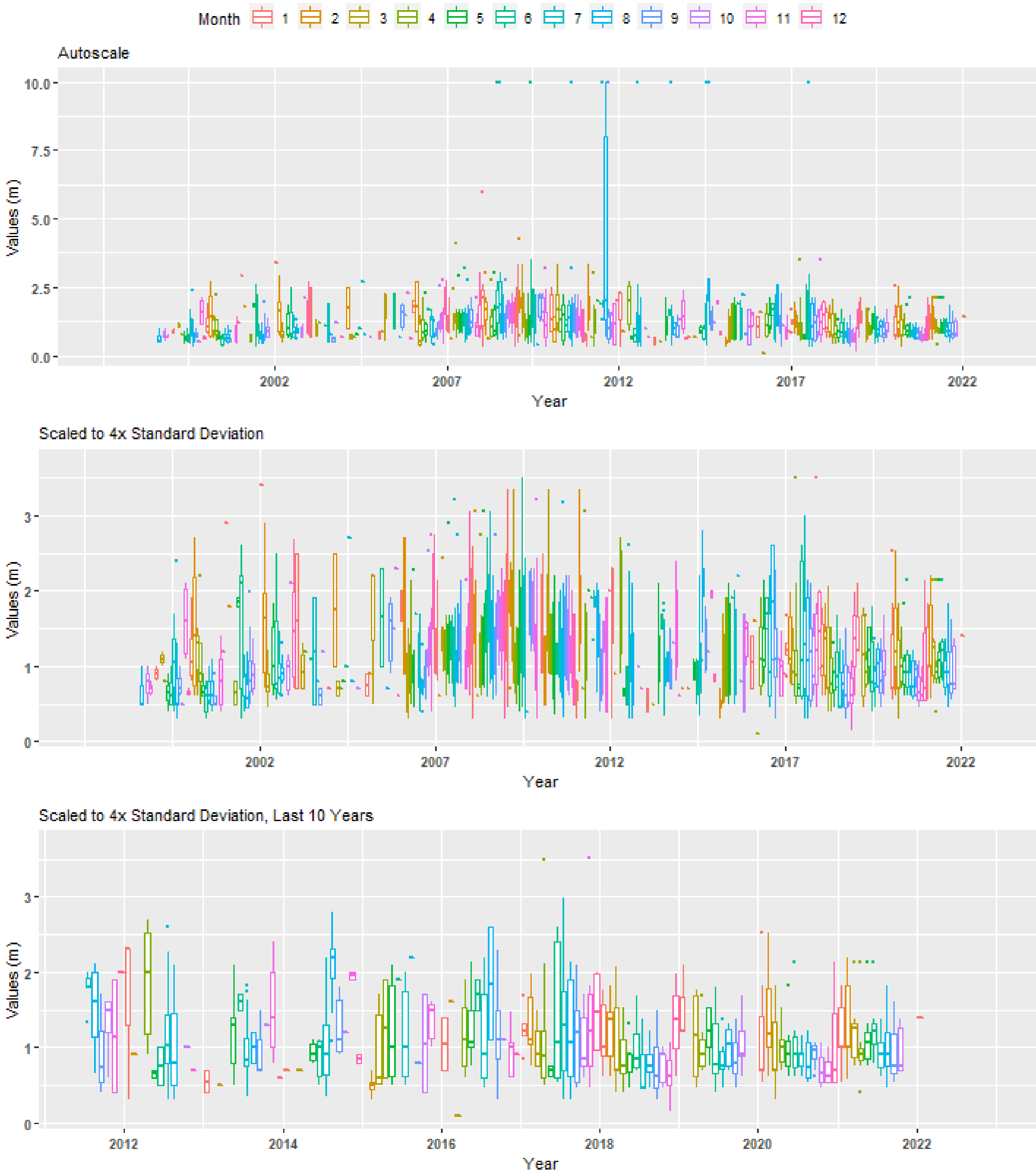
```
        labs(subtitle = "Autoscale",
             x = "Month", y = paste0("Values (", unit, ")"), fill = "Month") +
        scale_x_continuous(limits = c(0, 13), breaks = seq(3, 12, 3)) +
        theme(legend.position = "none",
              axis.text.x = element_text(face = "bold"),
              axis.text.y = element_text(face = "bold"))

    p8 <- ggplot(data = data[data$Use_In_Analysis == TRUE &
                                 data$ManagedAreaName == MA_names[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$Use_In_Analysis == TRUE &
                                 data$ManagedAreaName == MA_names[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_names[i]),
                            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(0.07, 1)))
    print(ggarrange(p00, YMset, ncol = 1, heights = c(0.07, 1)))
    print(ggarrange(p000, Mset, ncol = 1, heights = c(0.07, 1, 0.7)))
  }
}
```

Summary Box Plots for Alligator Harbor Aquatic Preserve
By Year

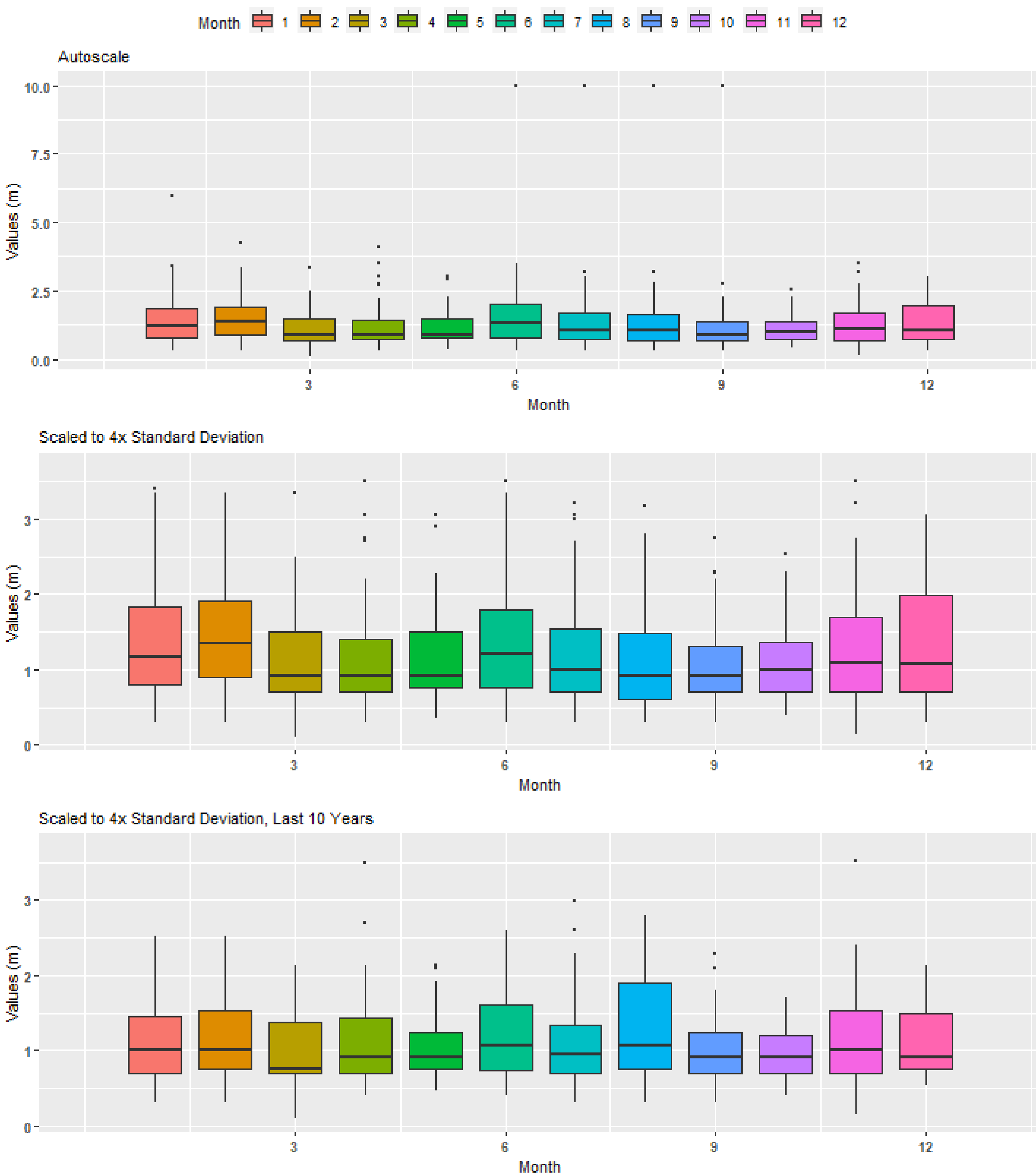


Summary Box Plots for Alligator Harbor Aquatic Preserve

By Year & Month

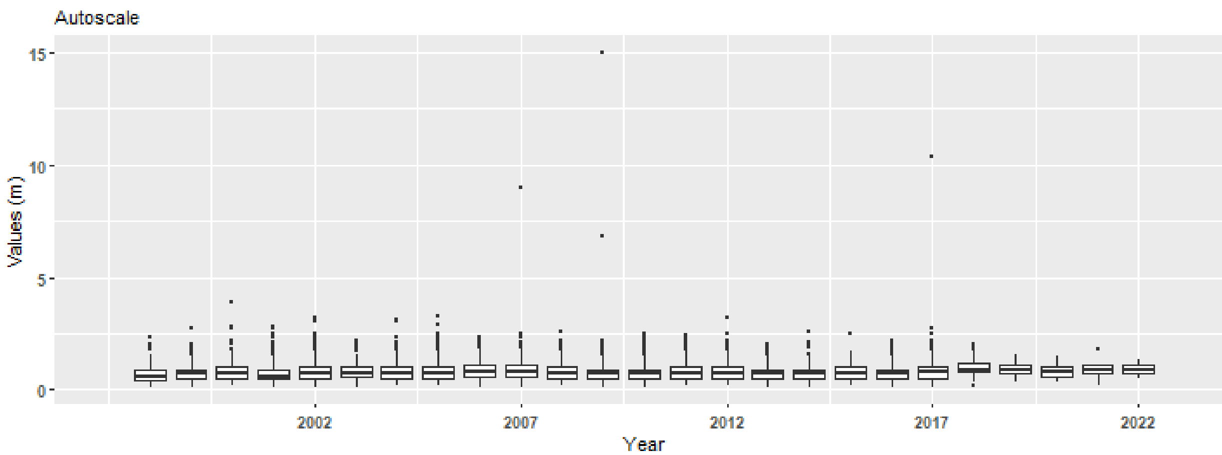


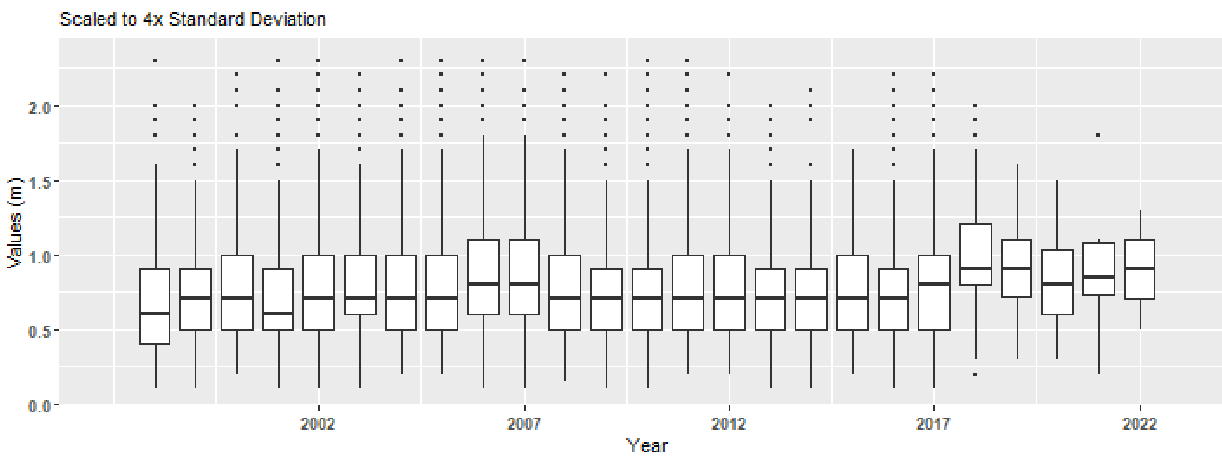
Summary Box Plots for Alligator Harbor Aquatic Preserve
By Month

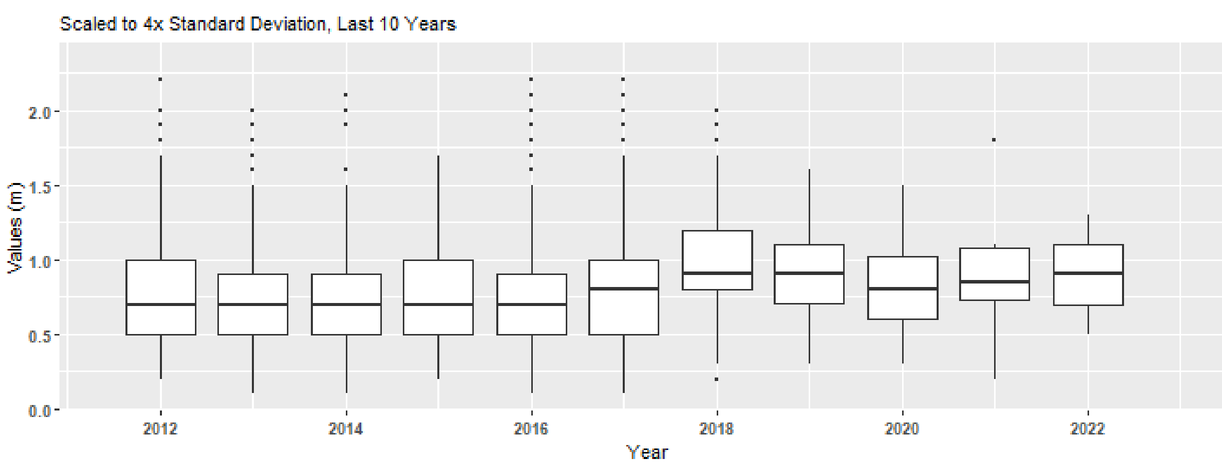


Summary Box Plots for Apalachicola Bay Aquatic Preserve

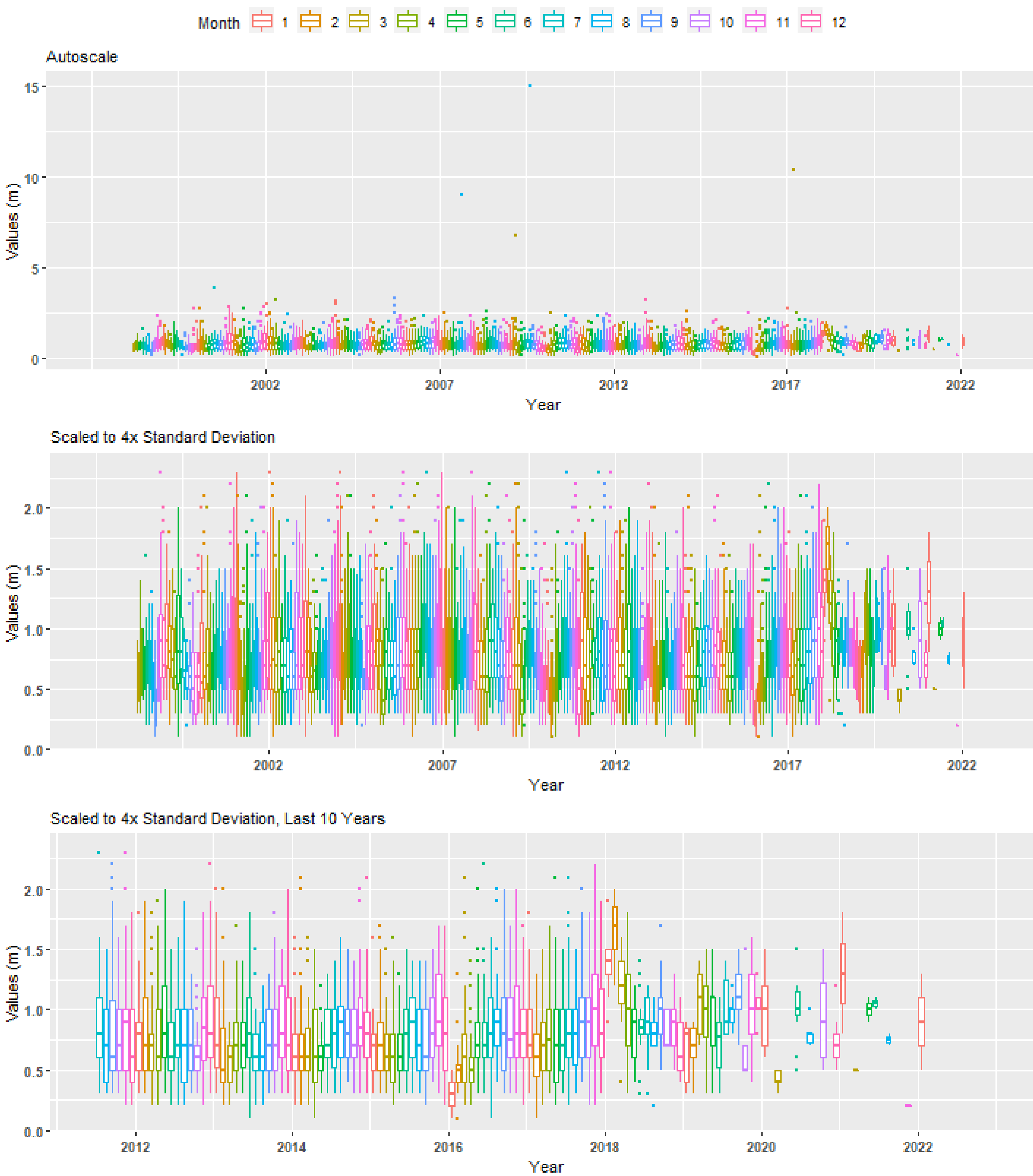
By Year

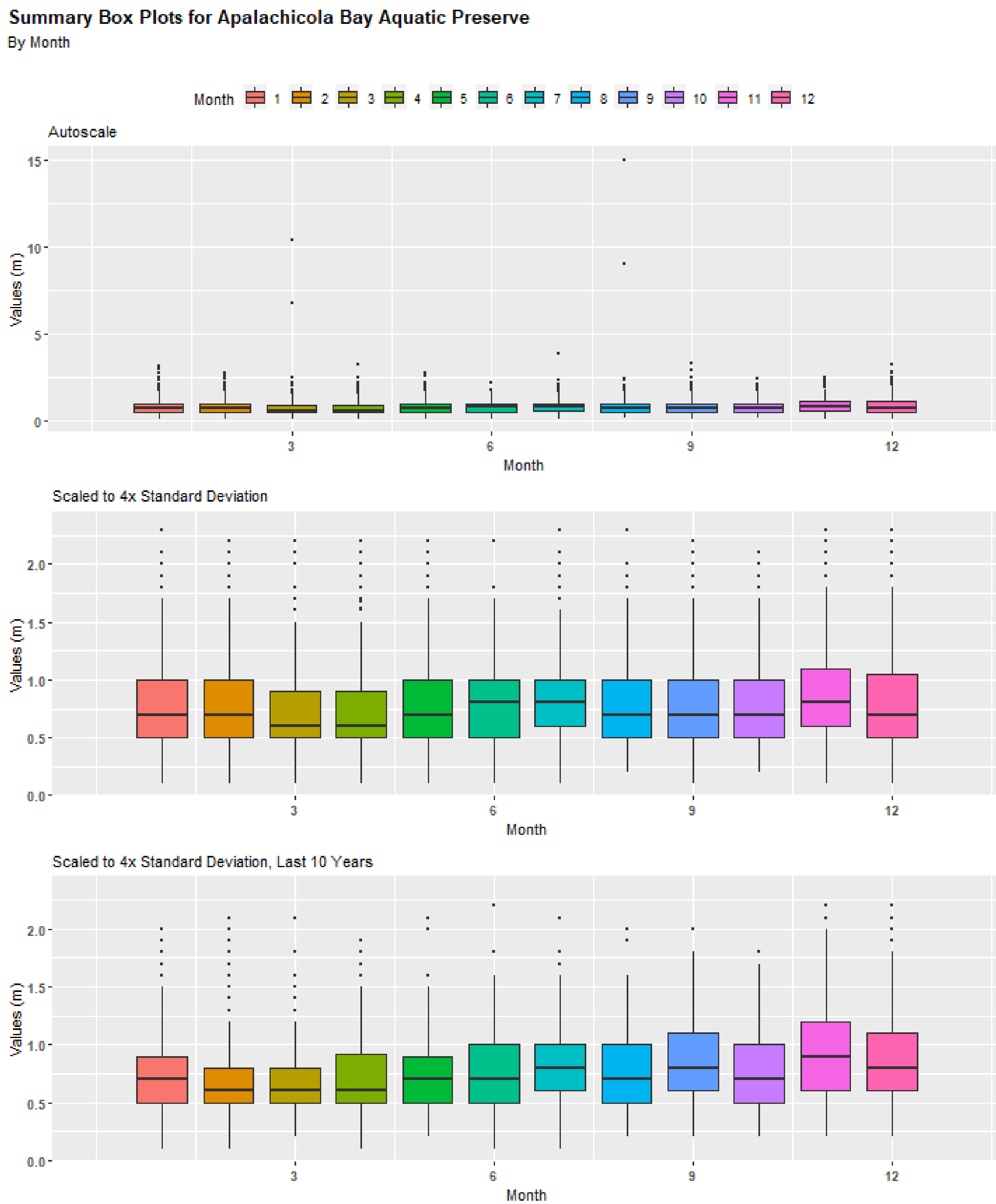




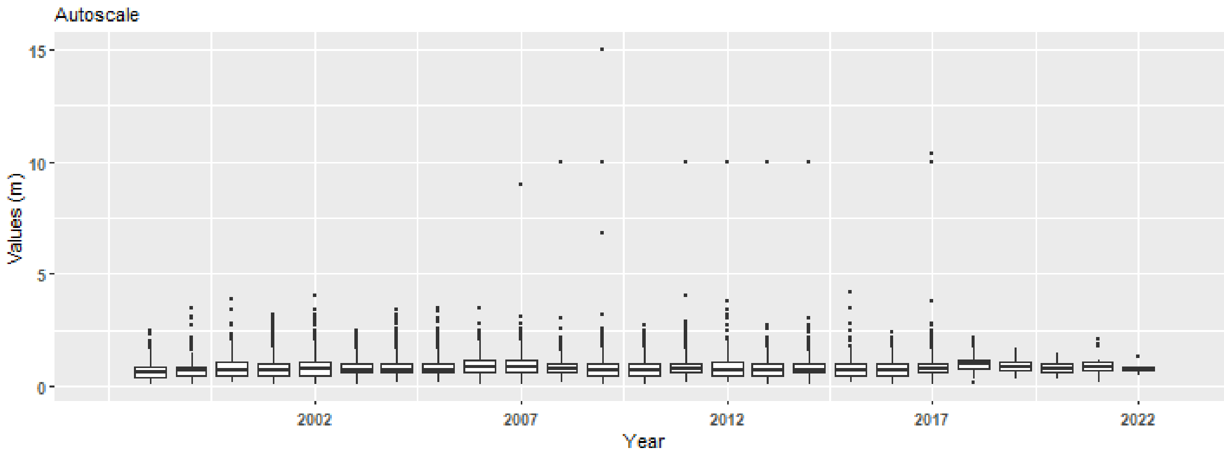


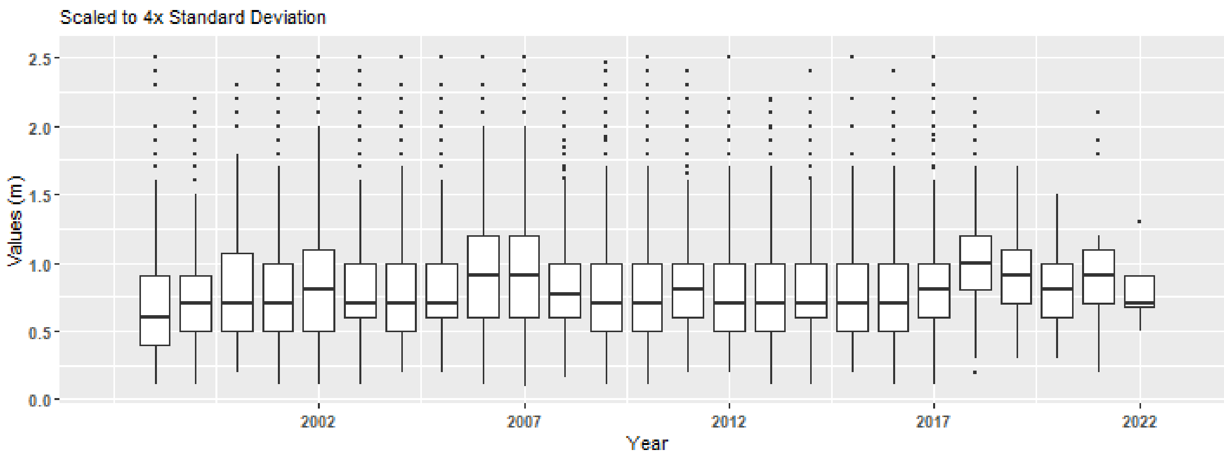
Summary Box Plots for Apalachicola Bay Aquatic Preserve
By Year & Month

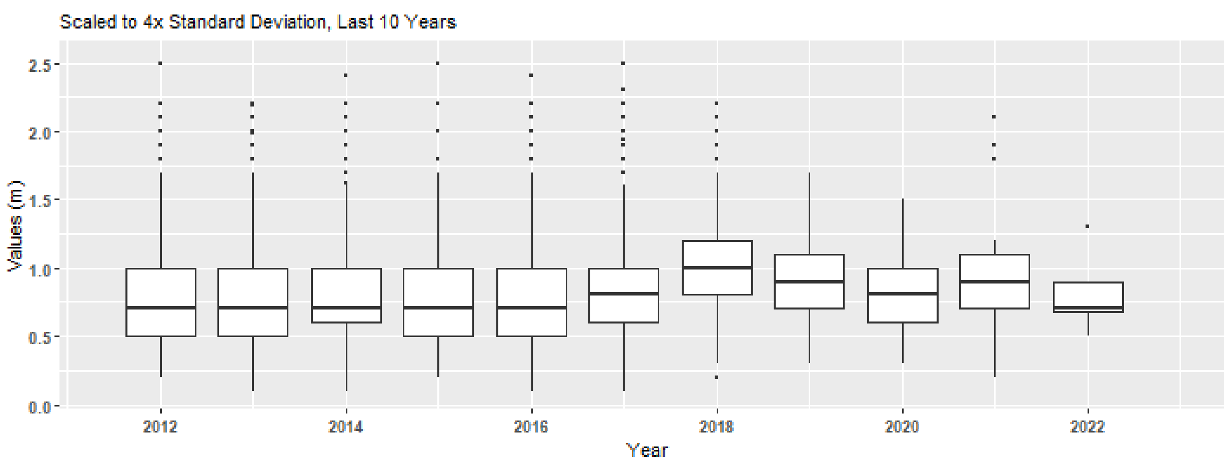



Summary Box Plots for Apalachicola Bay Aquatic Preserve
By Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (m)
15
10
5
0
3
6
9
12
Month
Scaled to 4x Standard Deviation
Values (m)
2.0
1.5
1.0
0.5
0.0
3
6
9
12
Month
Scaled to 4x Standard Deviation, Last 10 Years
Values (m)
2.0
1.5
1.0
0.5
0.0
3
6
9
12
Month

Summary Box Plots for Apalachicola National Estuarine Research Reserve
By Year

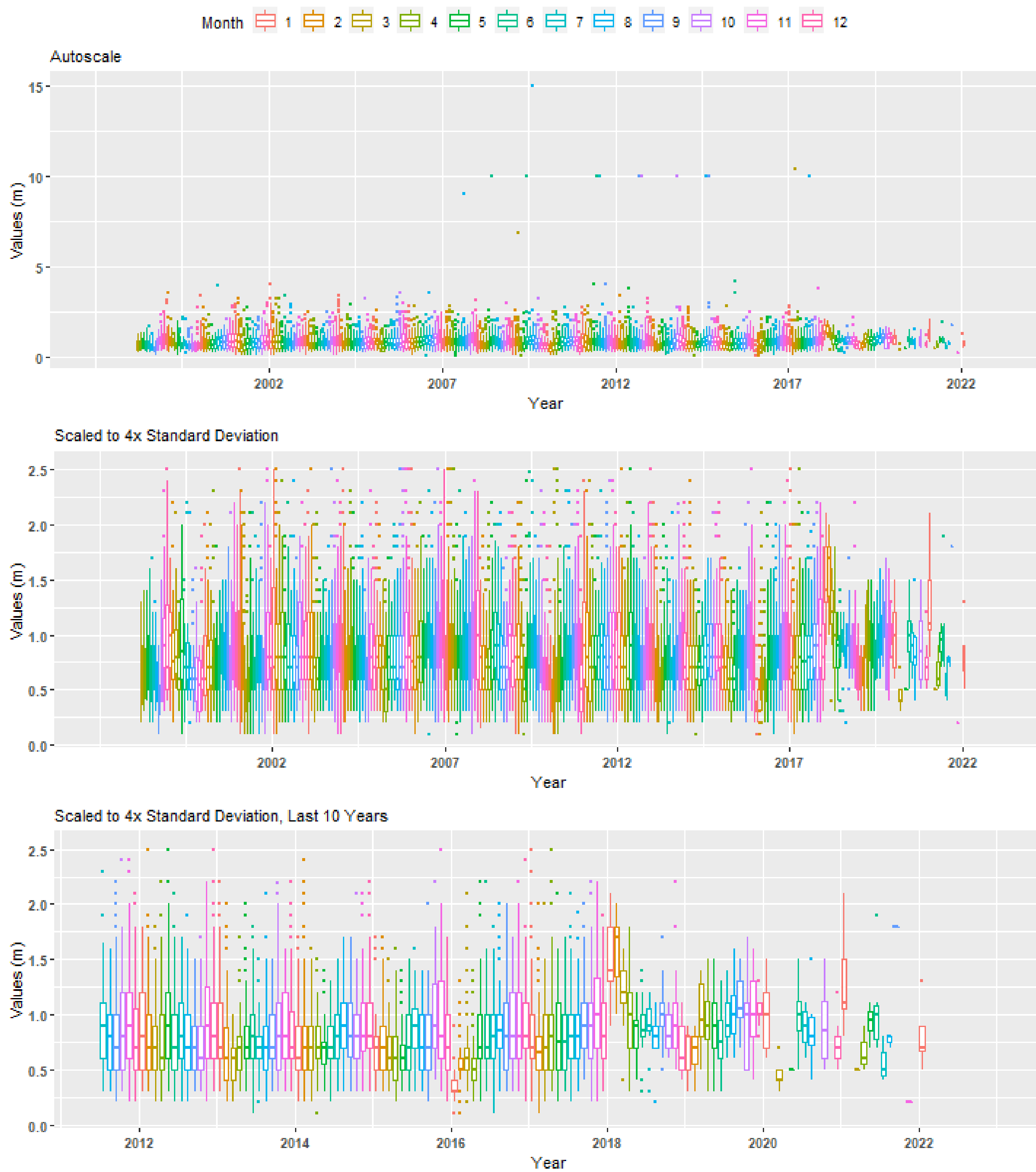






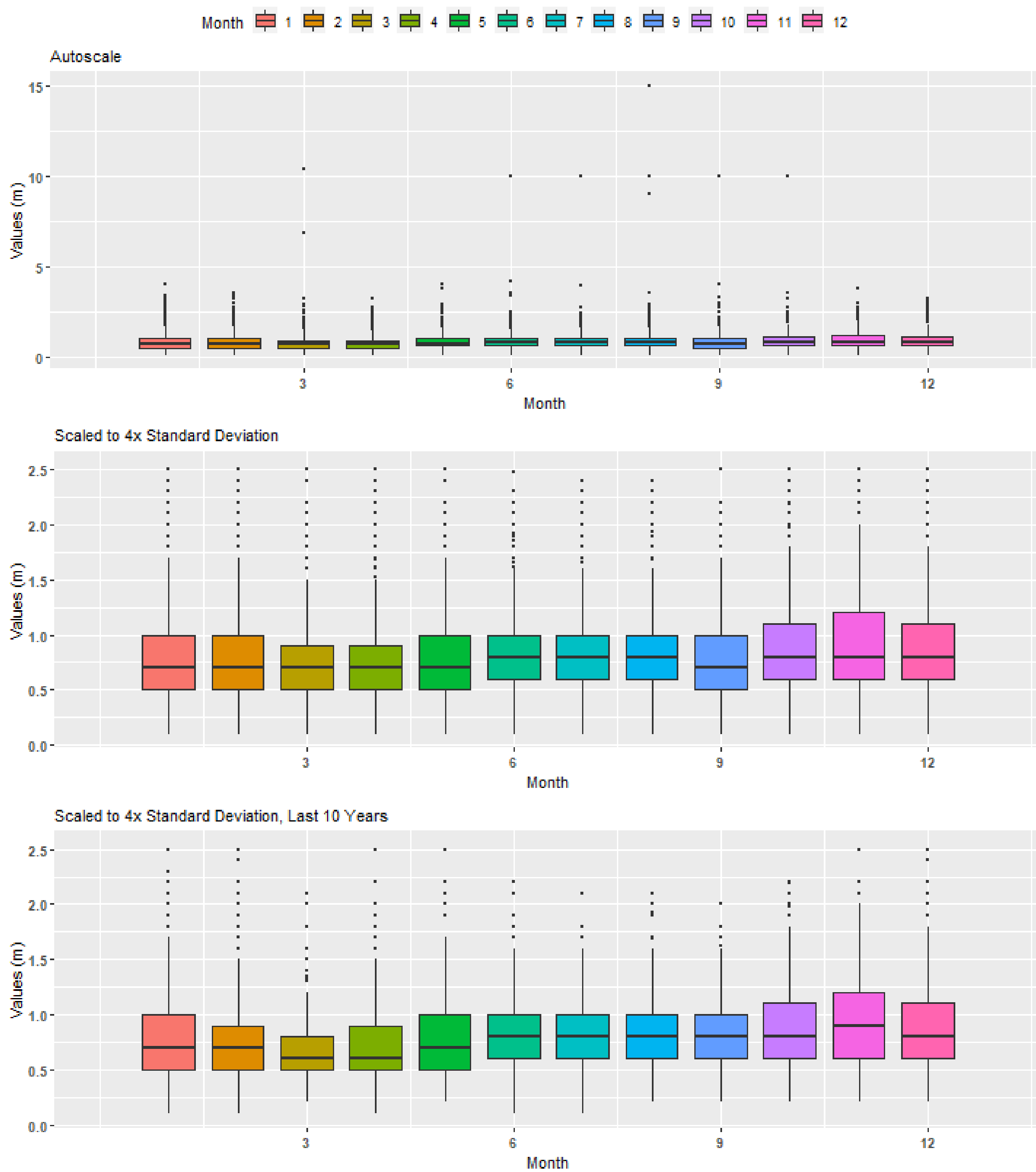
Summary Box Plots for Apalachicola National Estuarine Research Reserve

By Year & Month

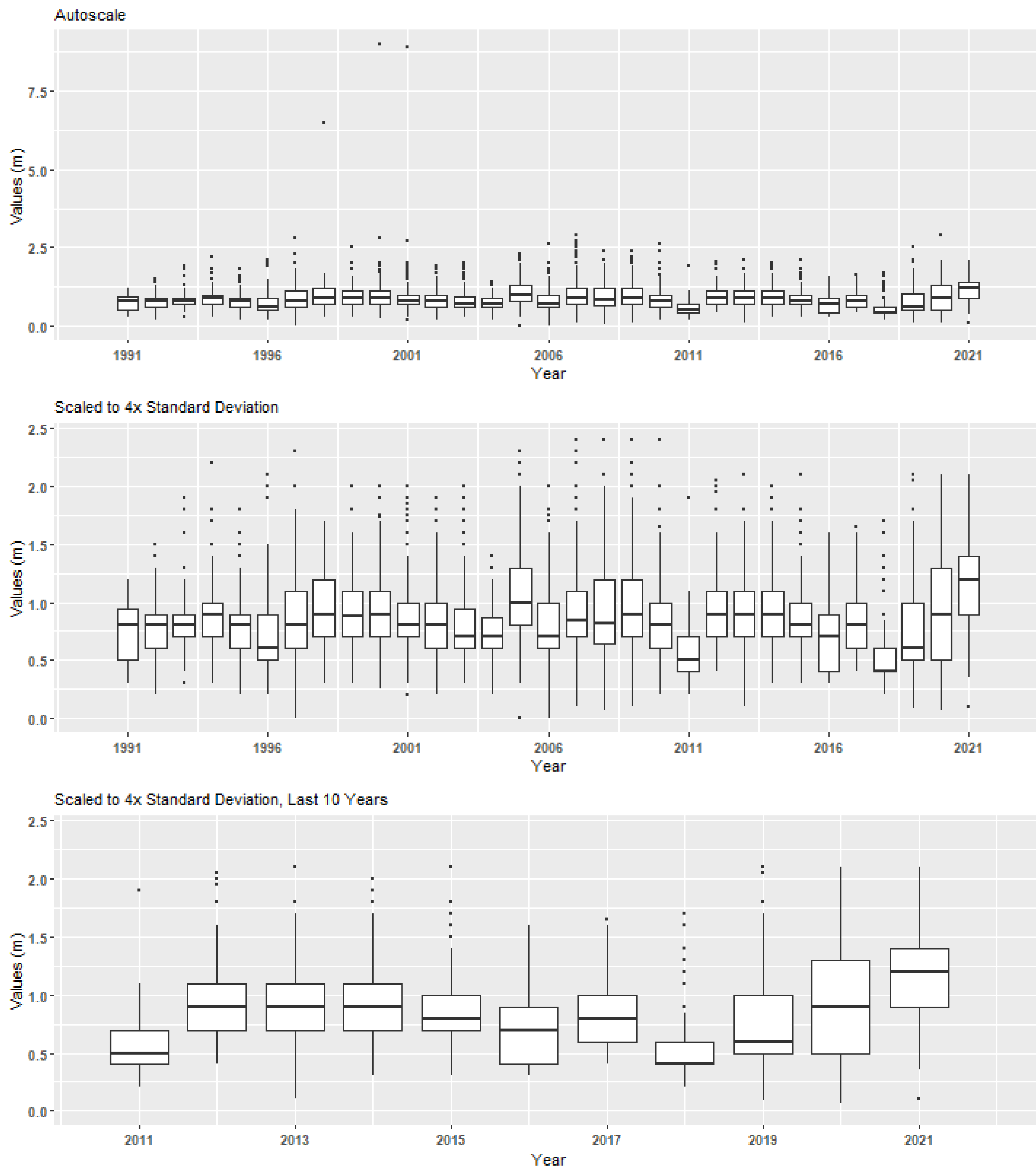


Summary Box Plots for Apalachicola National Estuarine Research Reserve

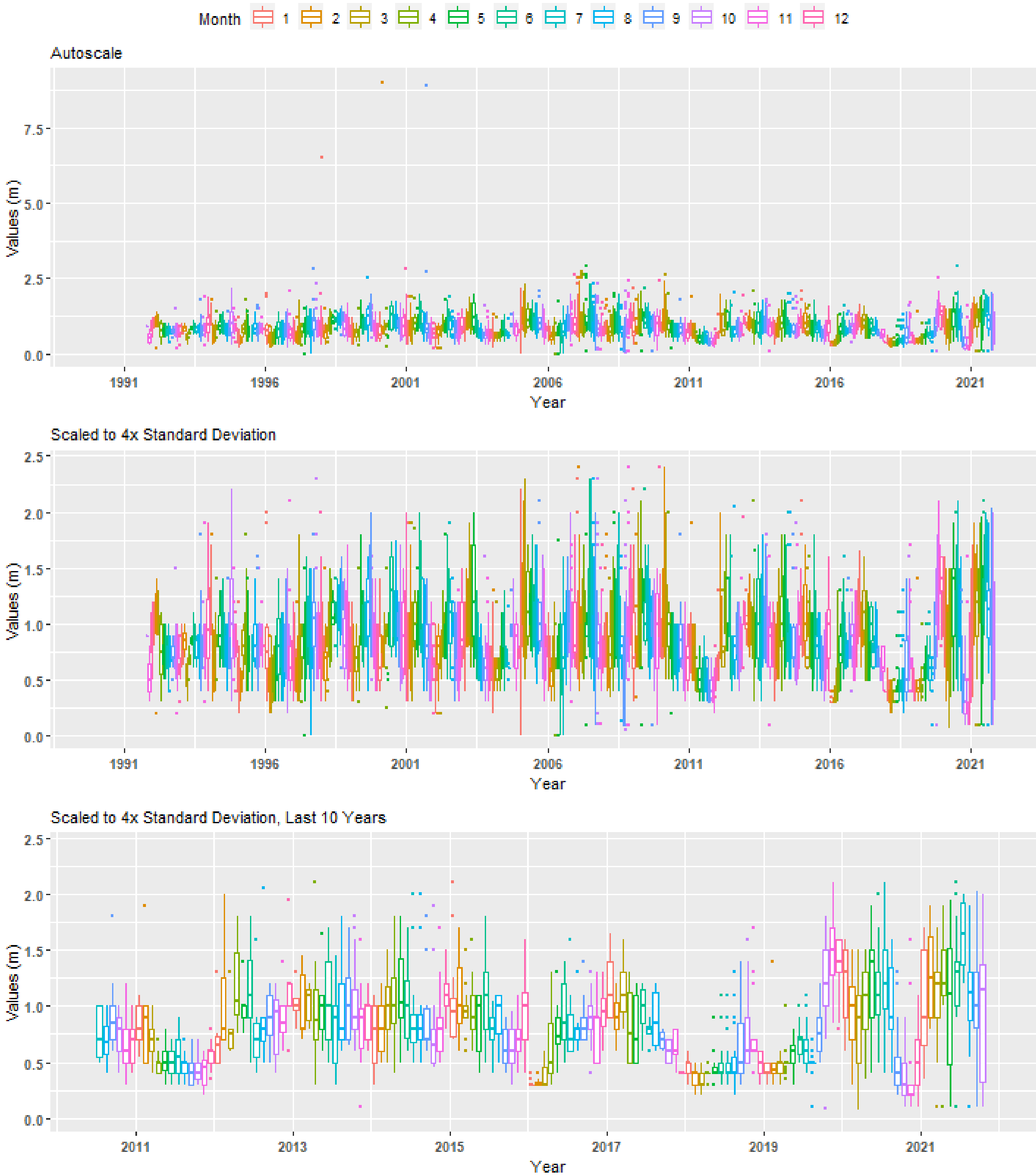
By Month



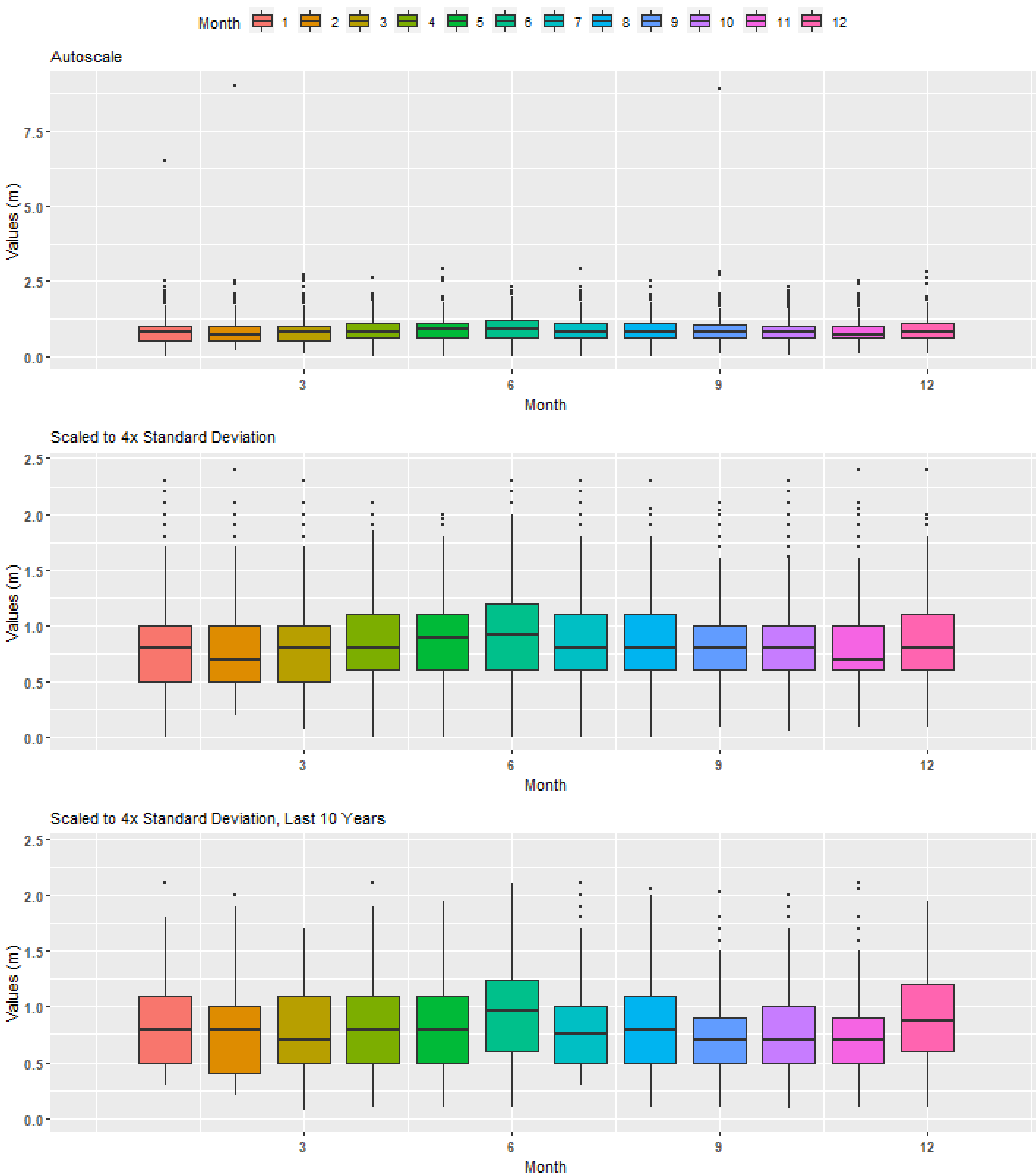
Summary Box Plots for Banana River Aquatic Preserve
By Year



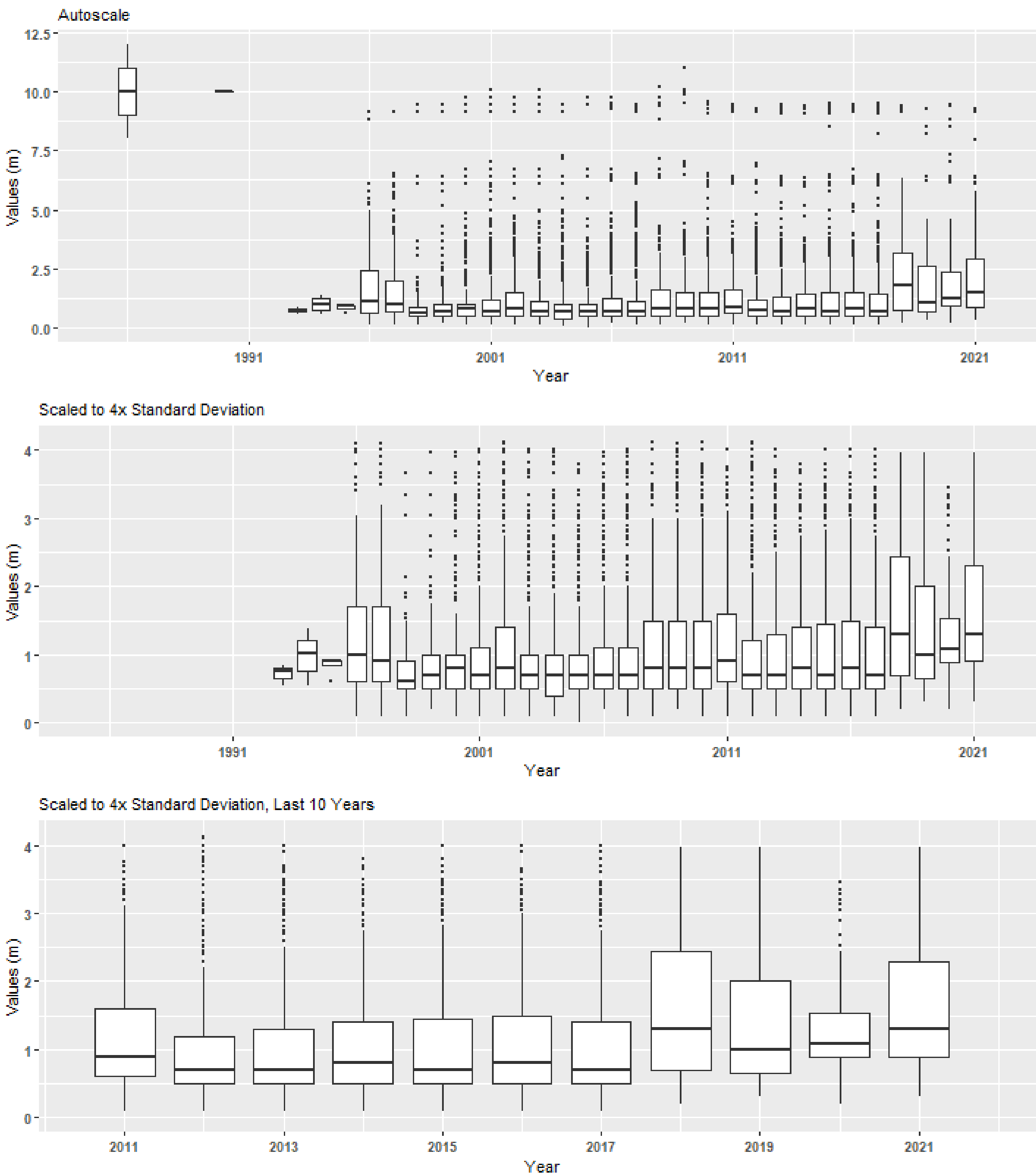
Summary Box Plots for Banana River Aquatic Preserve
By Year & Month



Summary Box Plots for Banana River Aquatic Preserve
By Month

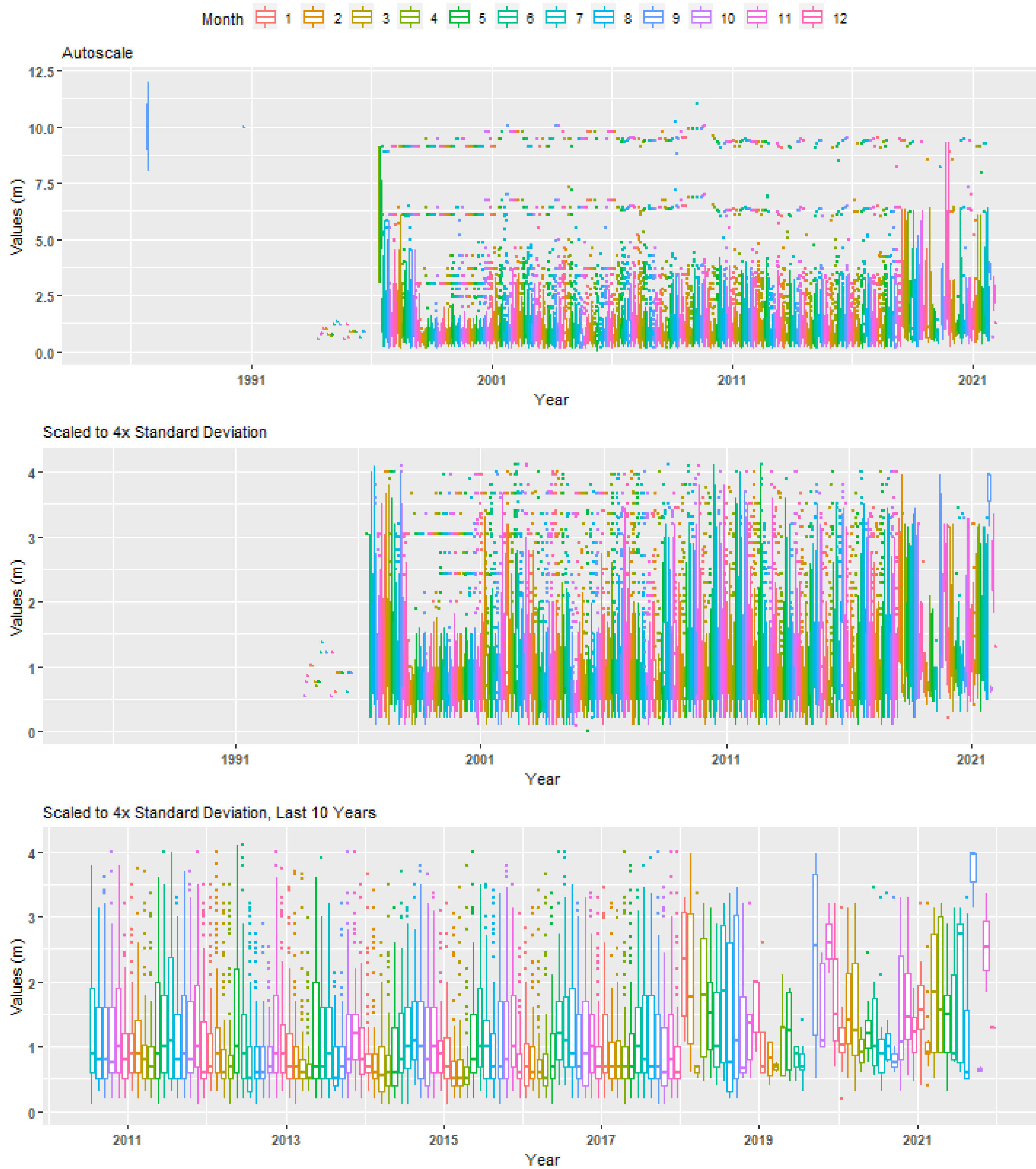


Summary Box Plots for Big Bend Seagrasses Aquatic Preserve
By Year



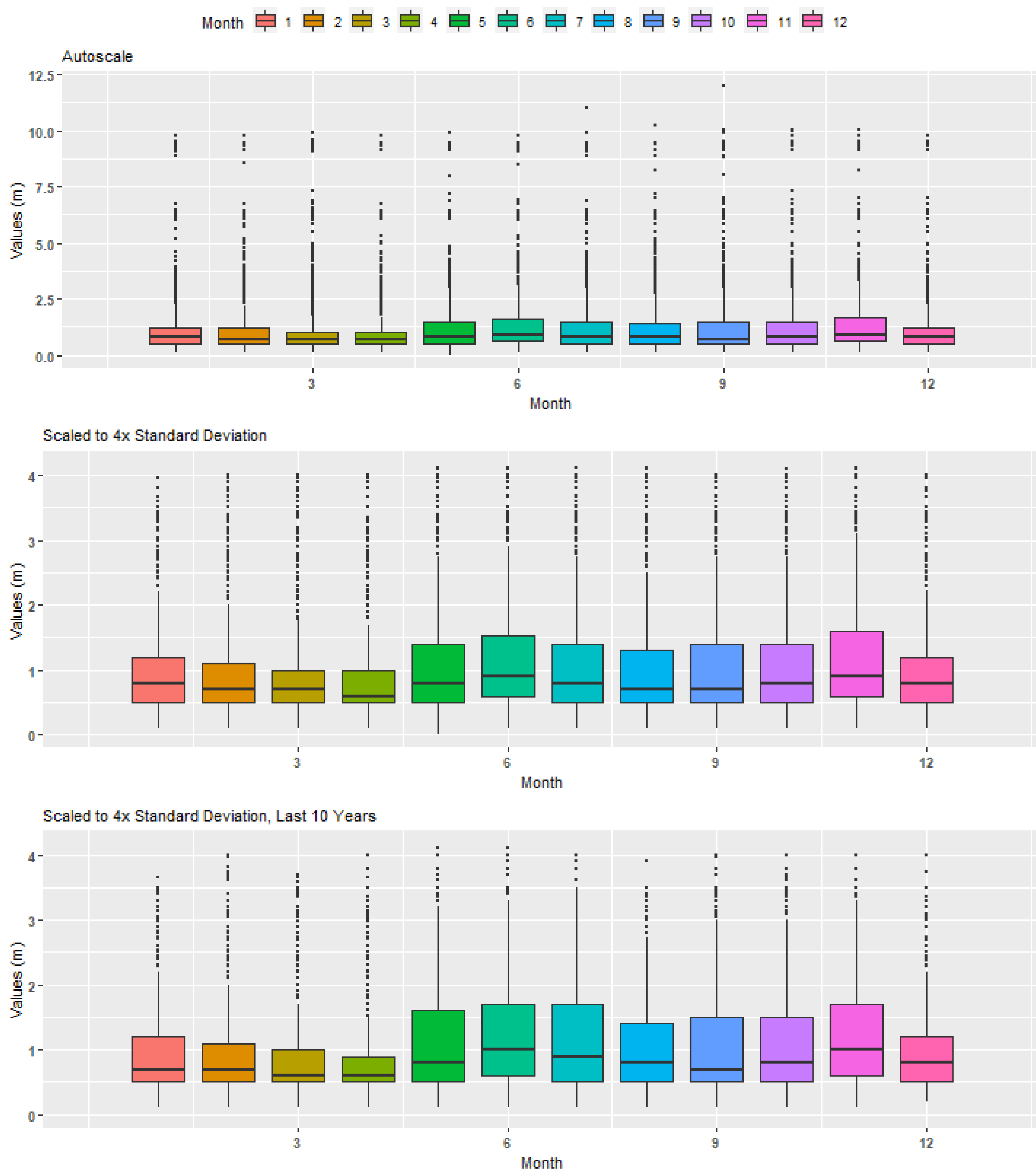
Summary Box Plots for Big Bend Seagrasses Aquatic Preserve

By Year & Month



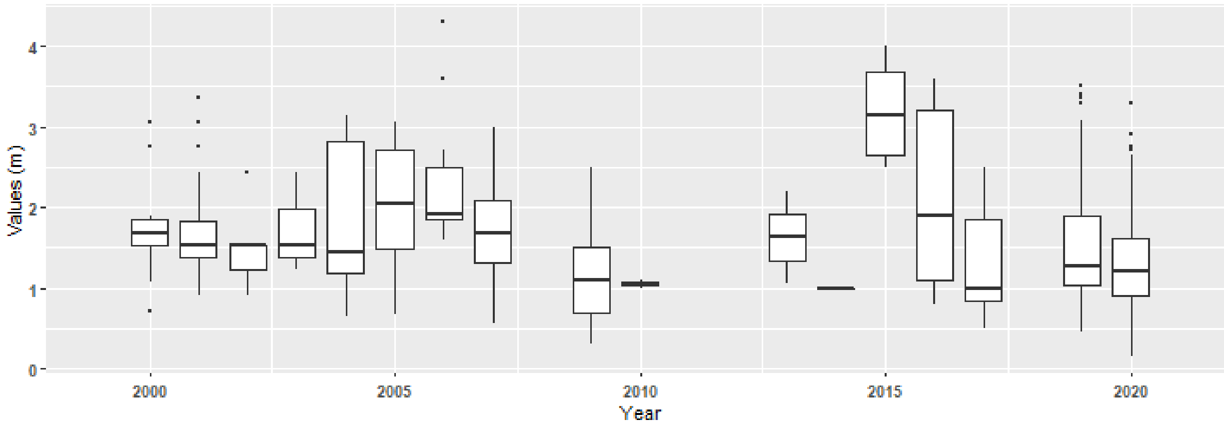
Summary Box Plots for Big Bend Seagrasses Aquatic Preserve

By Month

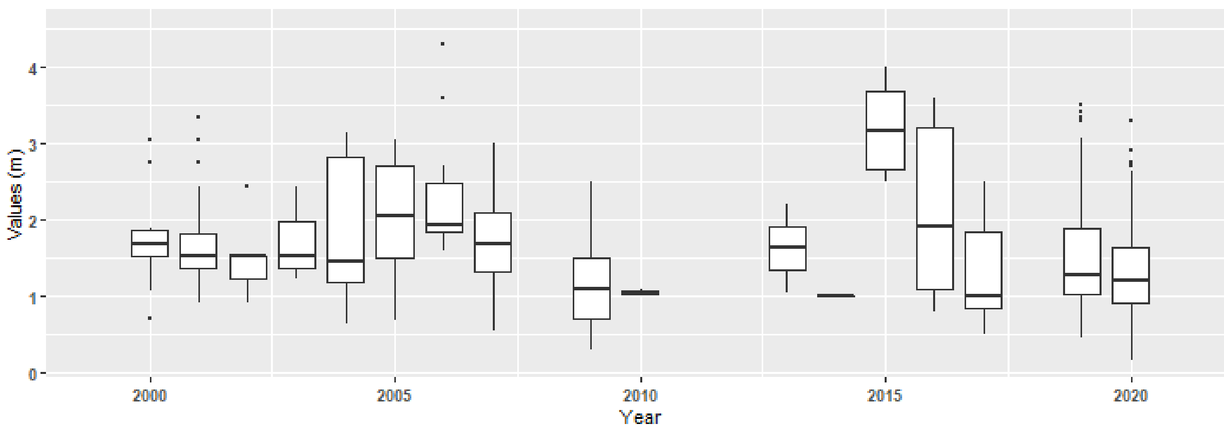


Summary Box Plots for Biscayne Bay Aquatic Preserve
By Year

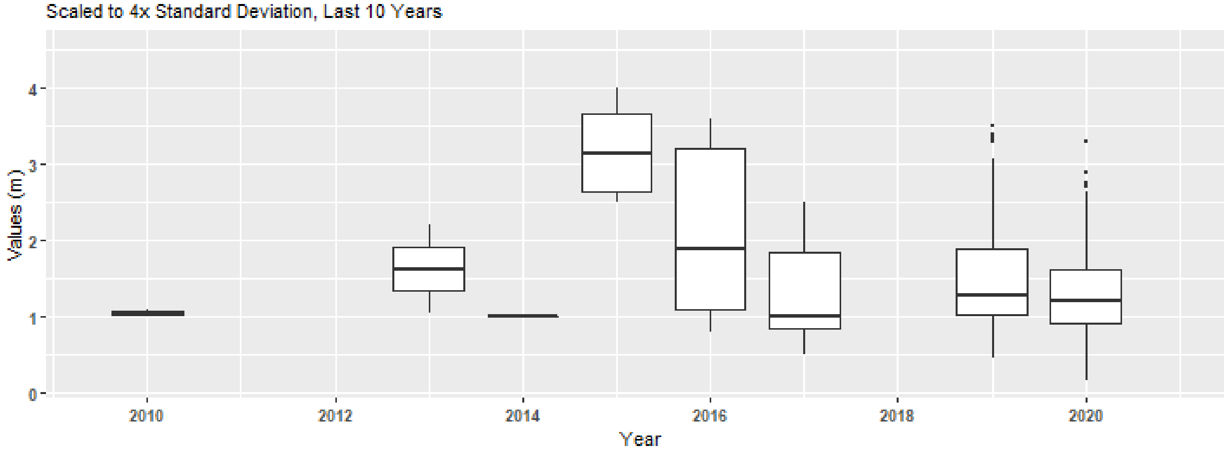
Autoscale

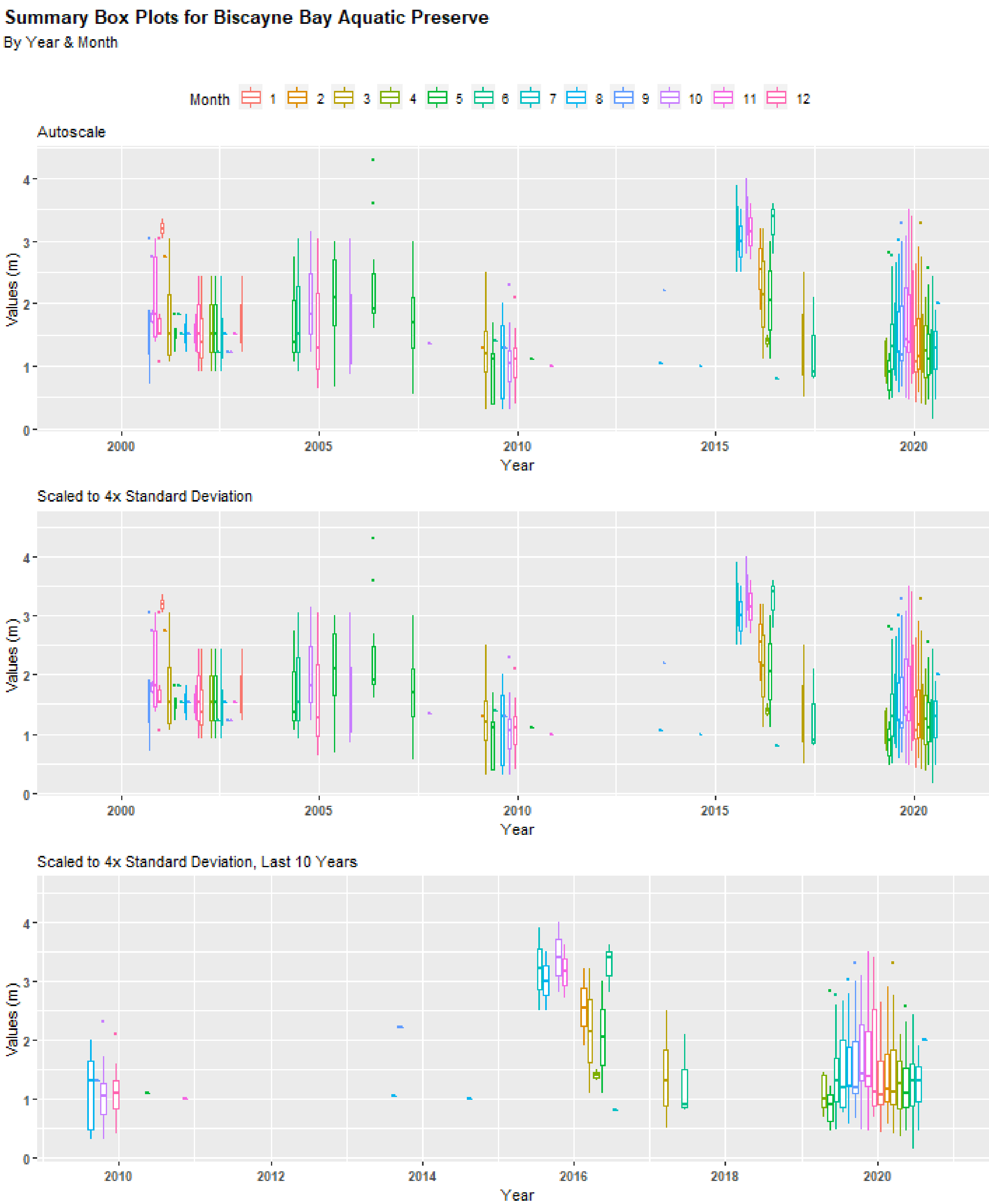


Scaled to 4x Standard Deviation

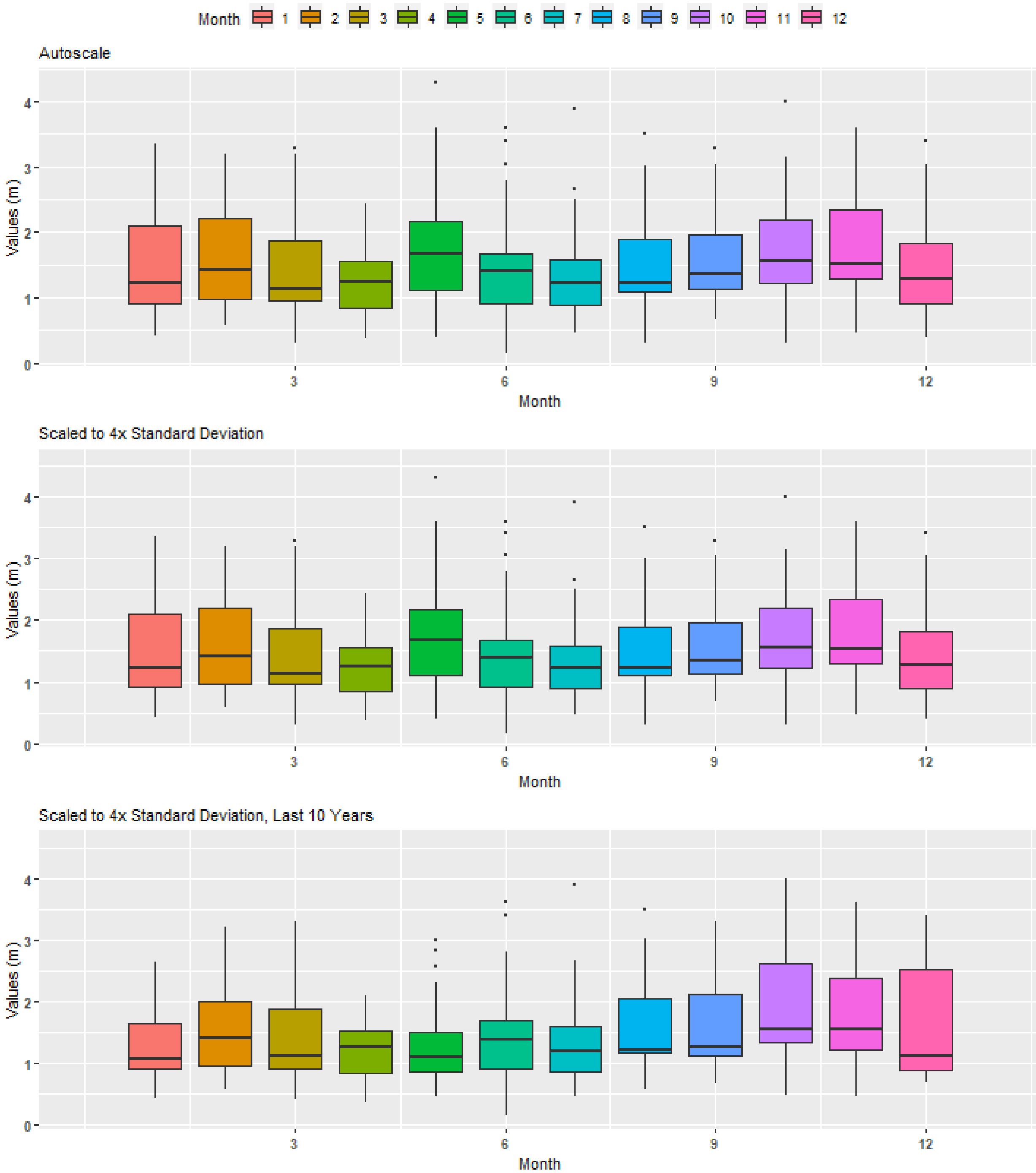


Scaled to 4x Standard Deviation, Last 10 Years



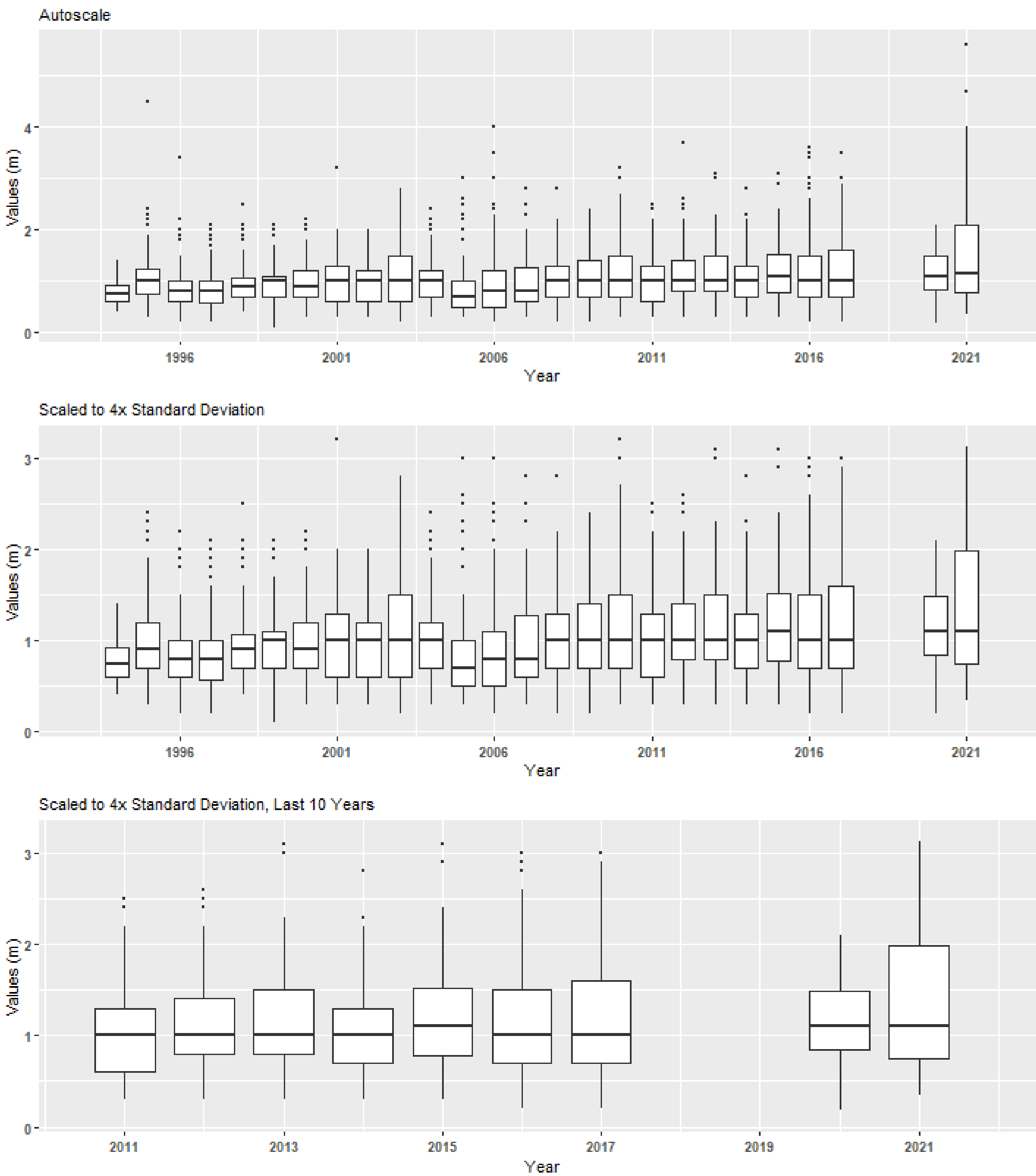

Summary Box Plots for Biscayne Bay Aquatic Preserve
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (m)
0
1
2
3
4
2000
2005
2010
2015
2020
Year
Scaled to 4x Standard Deviation
Values (m)
0
1
2
3
4
2000
2005
2010
2015
2020
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (m)
0
1
2
3
4
2010
2012
2014
2016
2018
2020
Year

Summary Box Plots for Biscayne Bay Aquatic Preserve
By Month

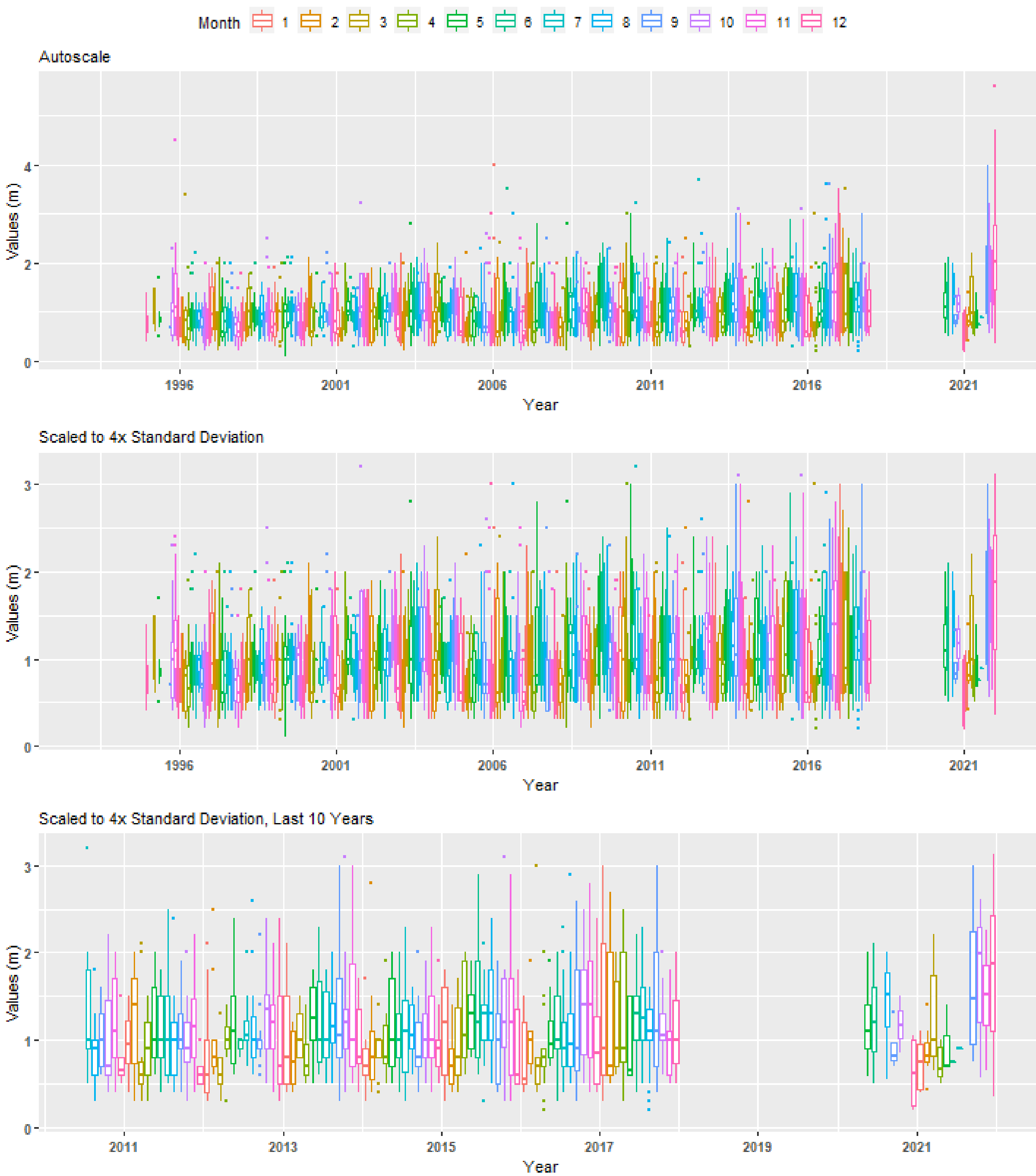


Summary Box Plots for Boca Ciega Bay Aquatic Preserve

By Year

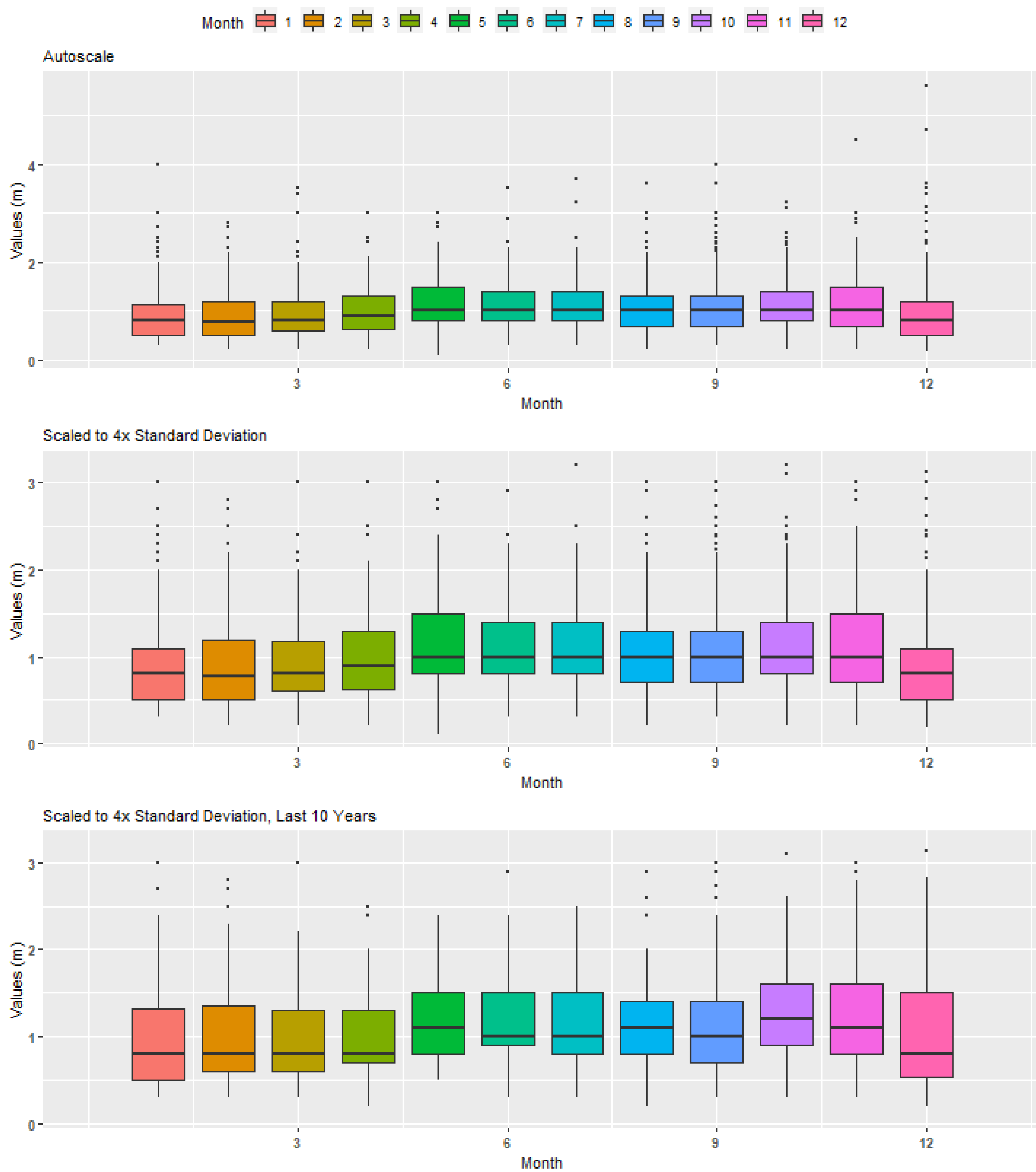


Summary Box Plots for Boca Ciega Bay Aquatic Preserve
By Year & Month



Summary Box Plots for Boca Ciega Bay Aquatic Preserve

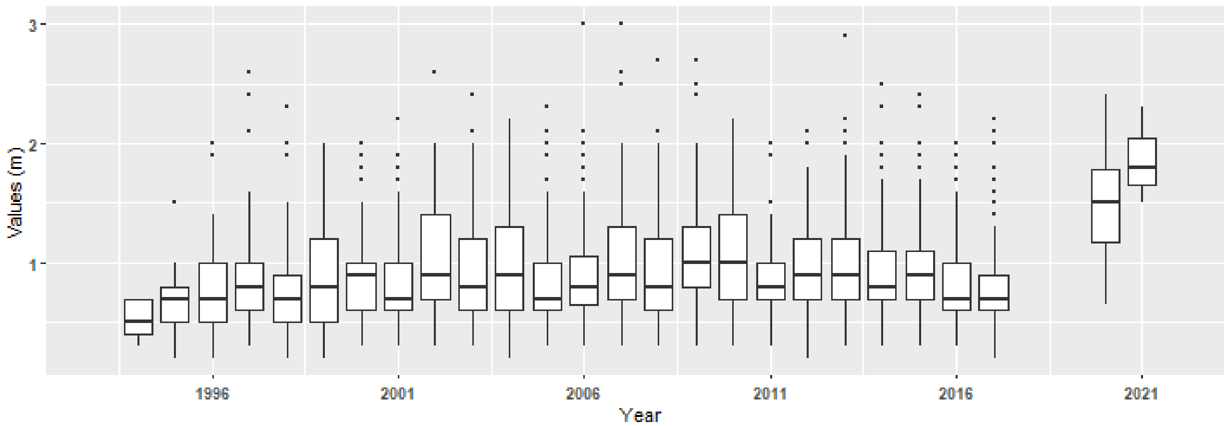
By Month



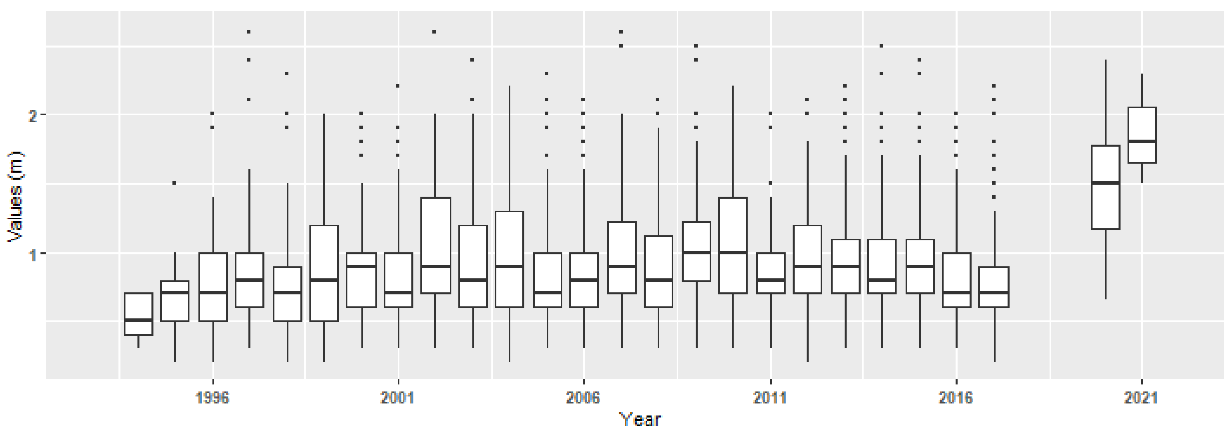
Summary Box Plots for Cape Haze Aquatic Preserve

By Year

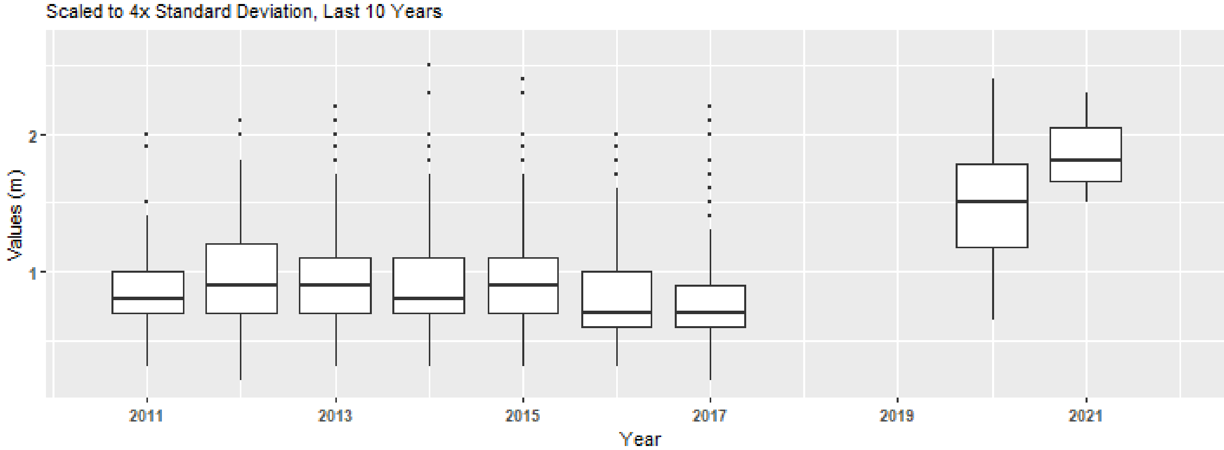
Autoscale



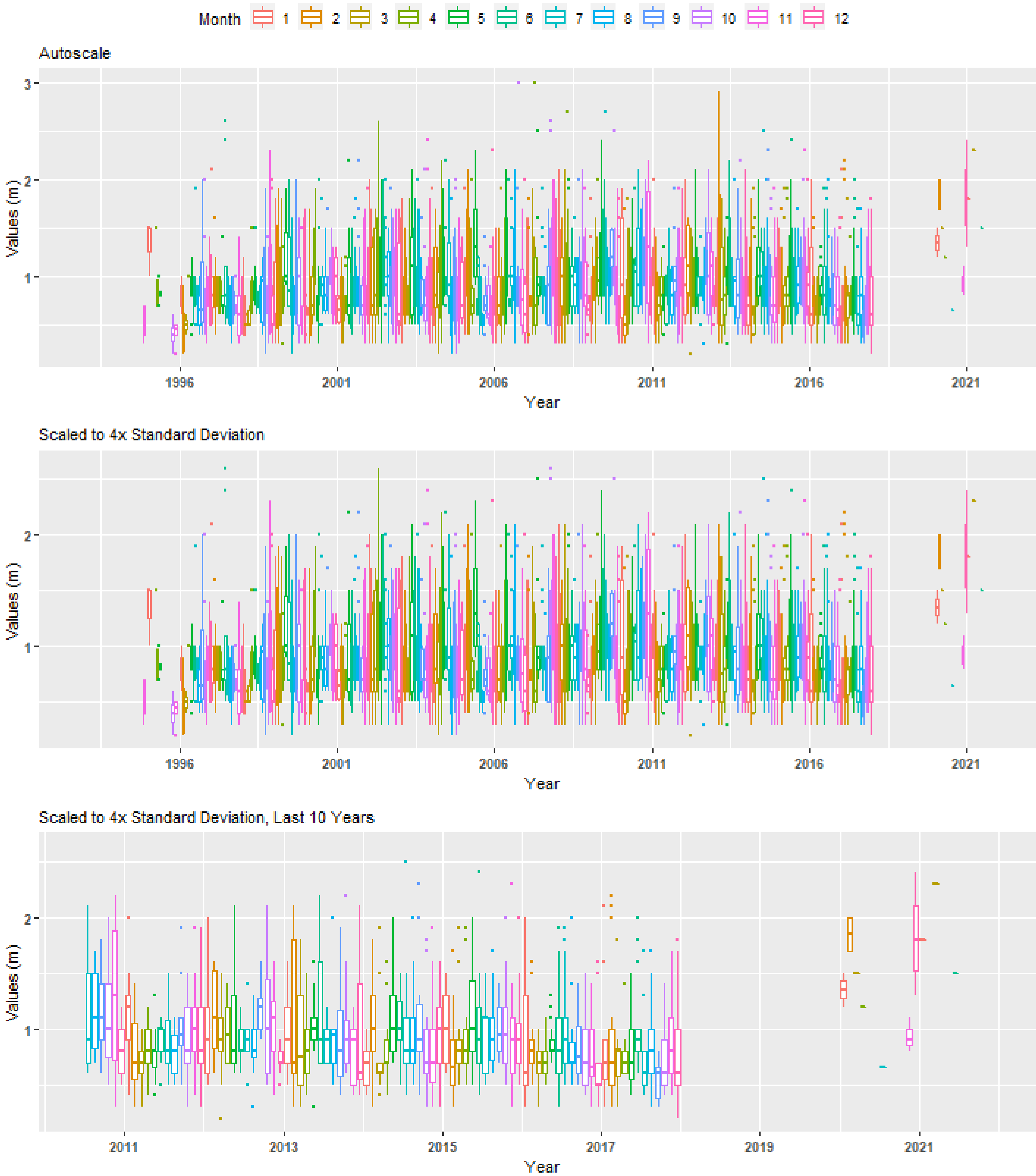
Scaled to 4x Standard Deviation



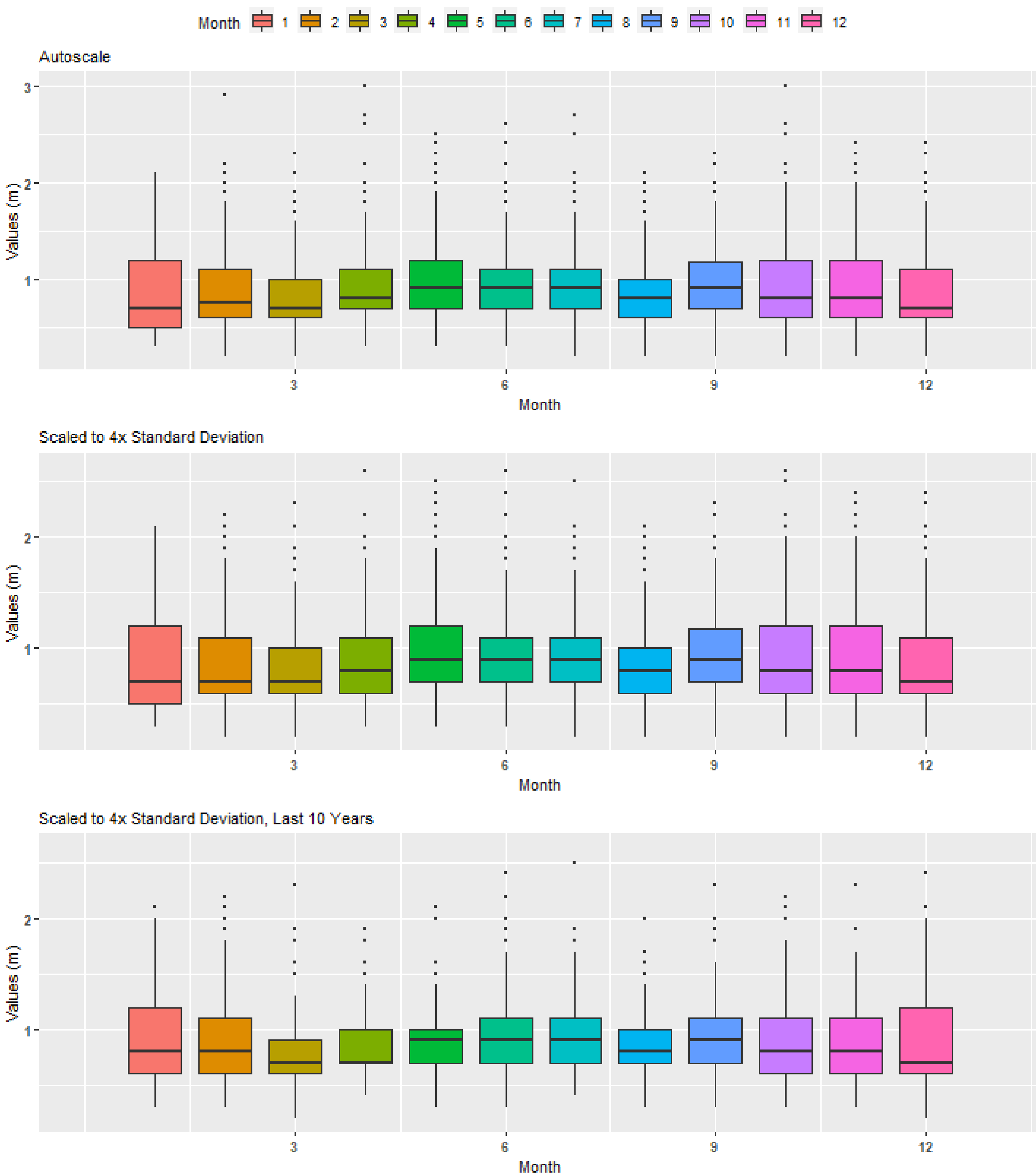
Scaled to 4x Standard Deviation, Last 10 Years



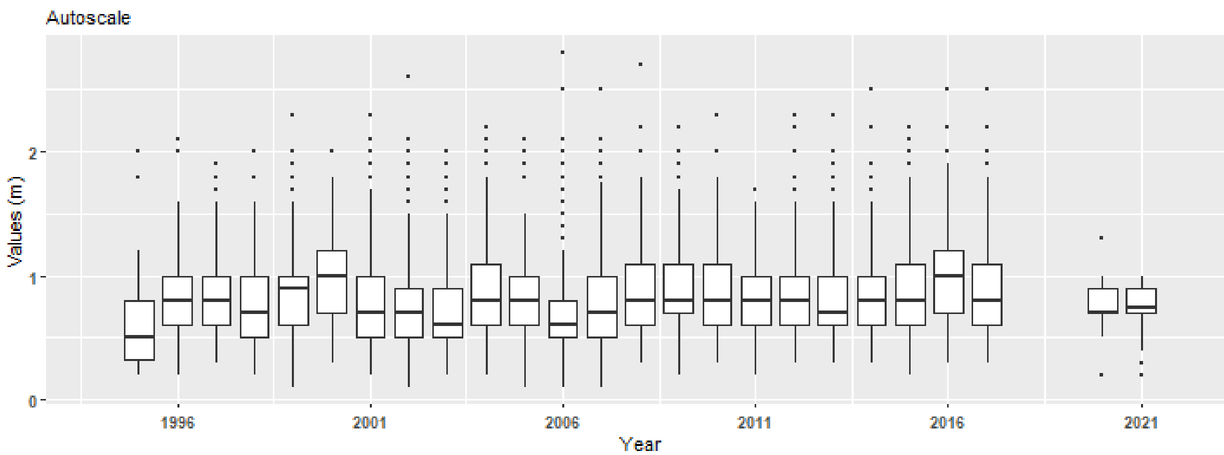
Summary Box Plots for Cape Haze Aquatic Preserve
By Year & Month

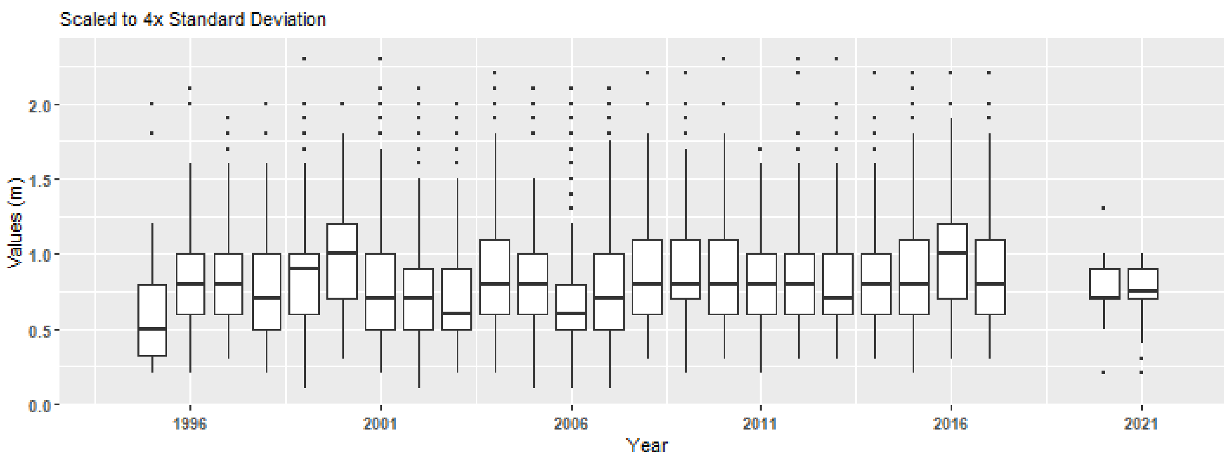


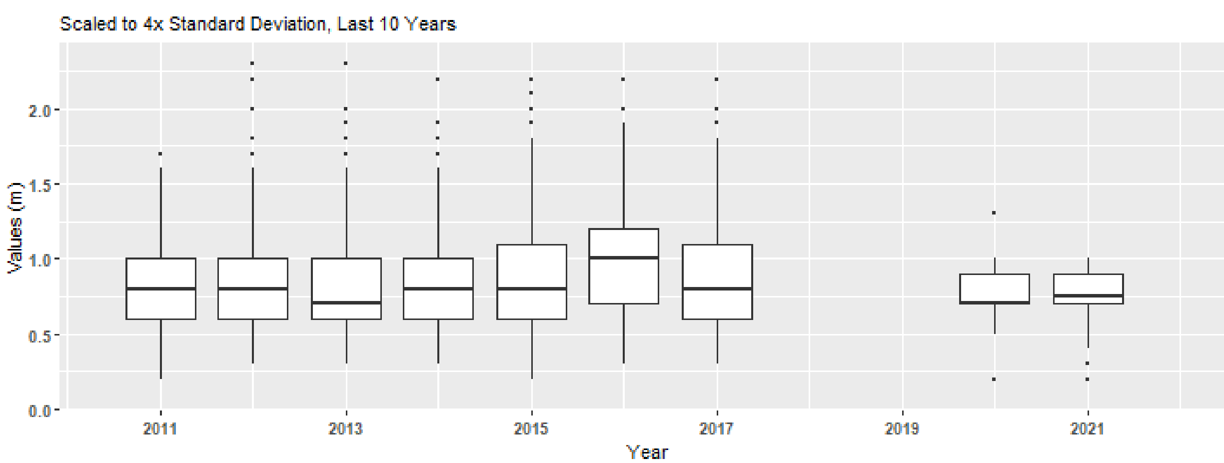
Summary Box Plots for Cape Haze Aquatic Preserve
By Month



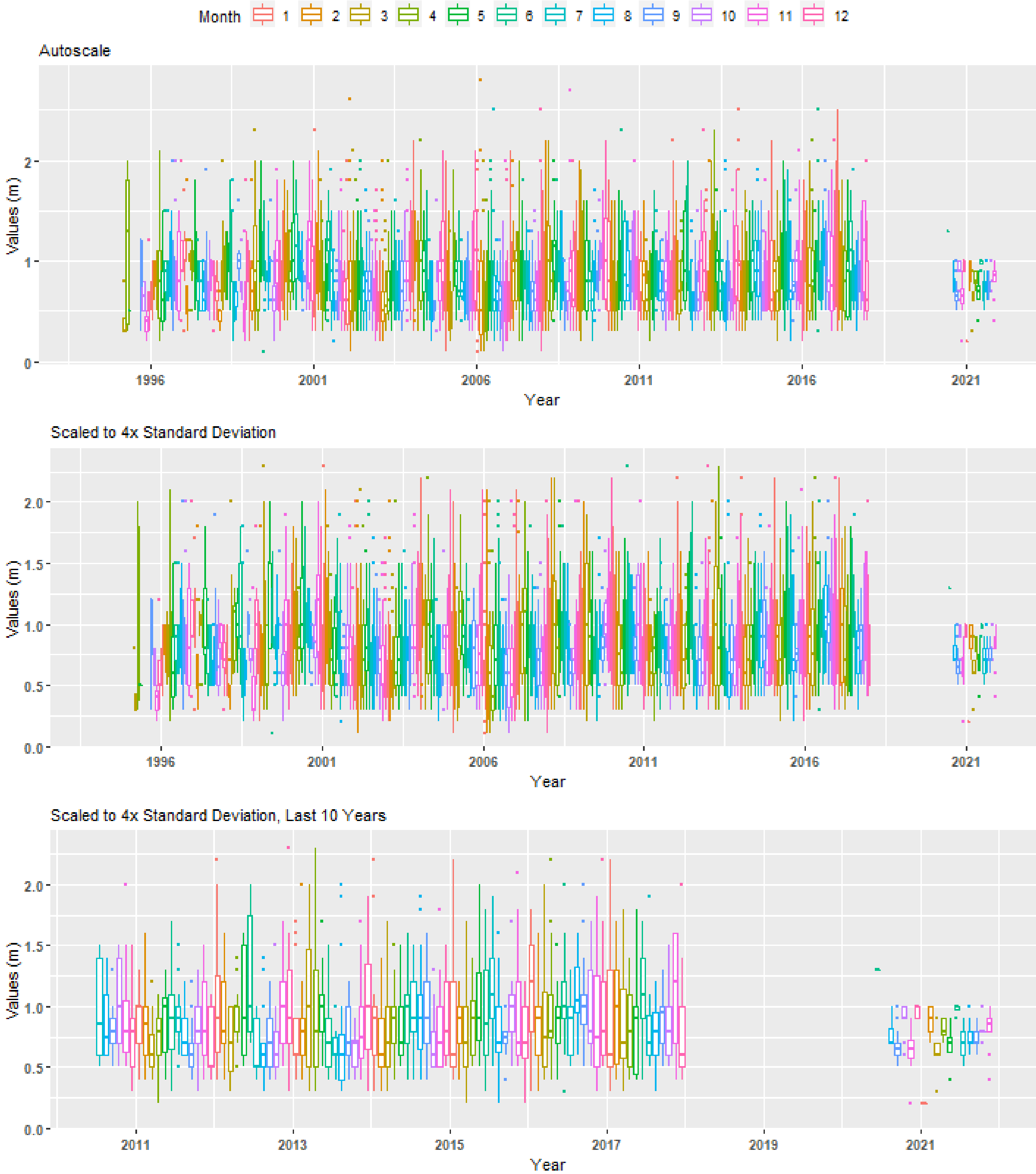
Summary Box Plots for Cockroach Bay Aquatic Preserve
By Year



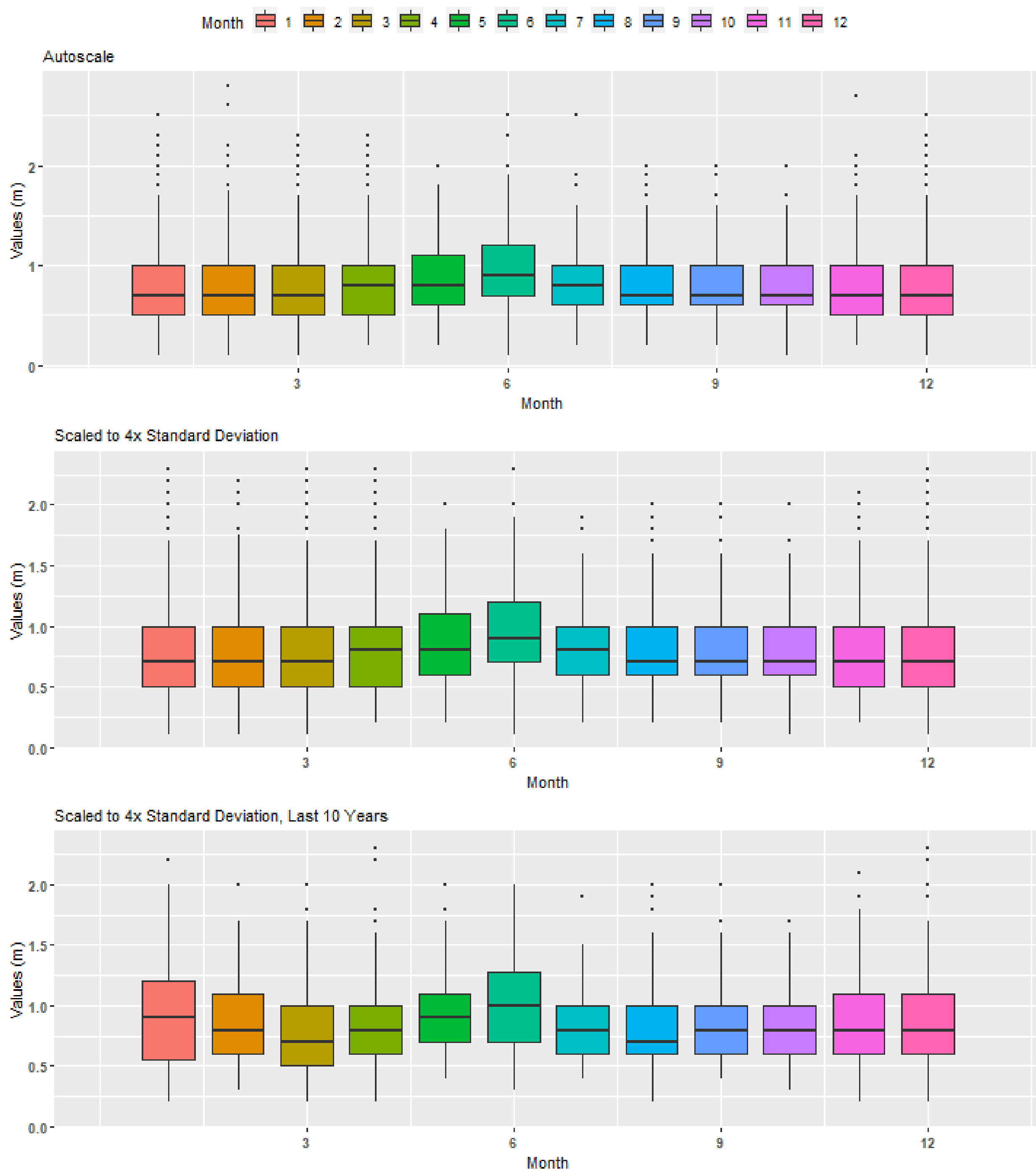




Summary Box Plots for Cockroach Bay Aquatic Preserve
By Year & Month

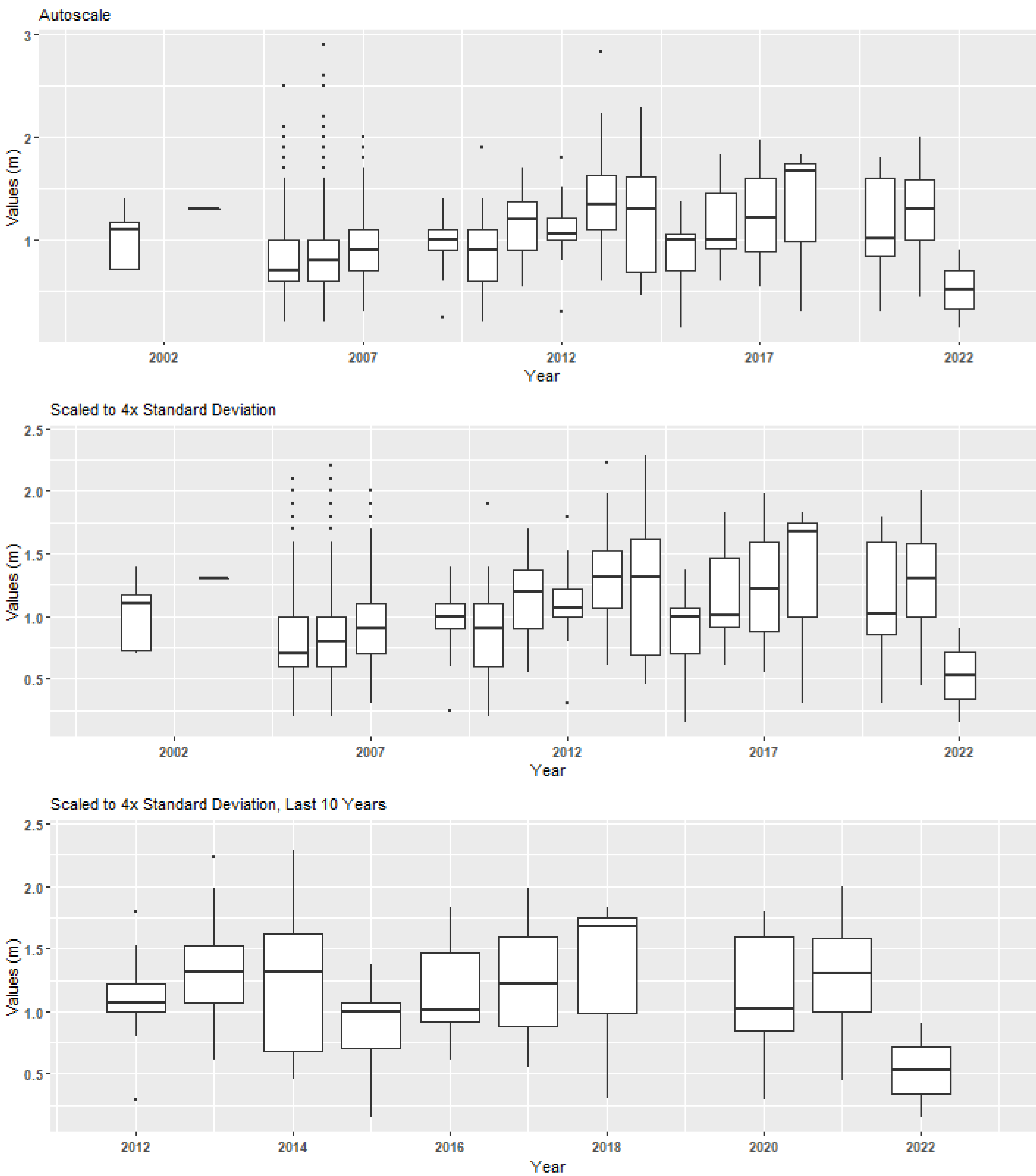


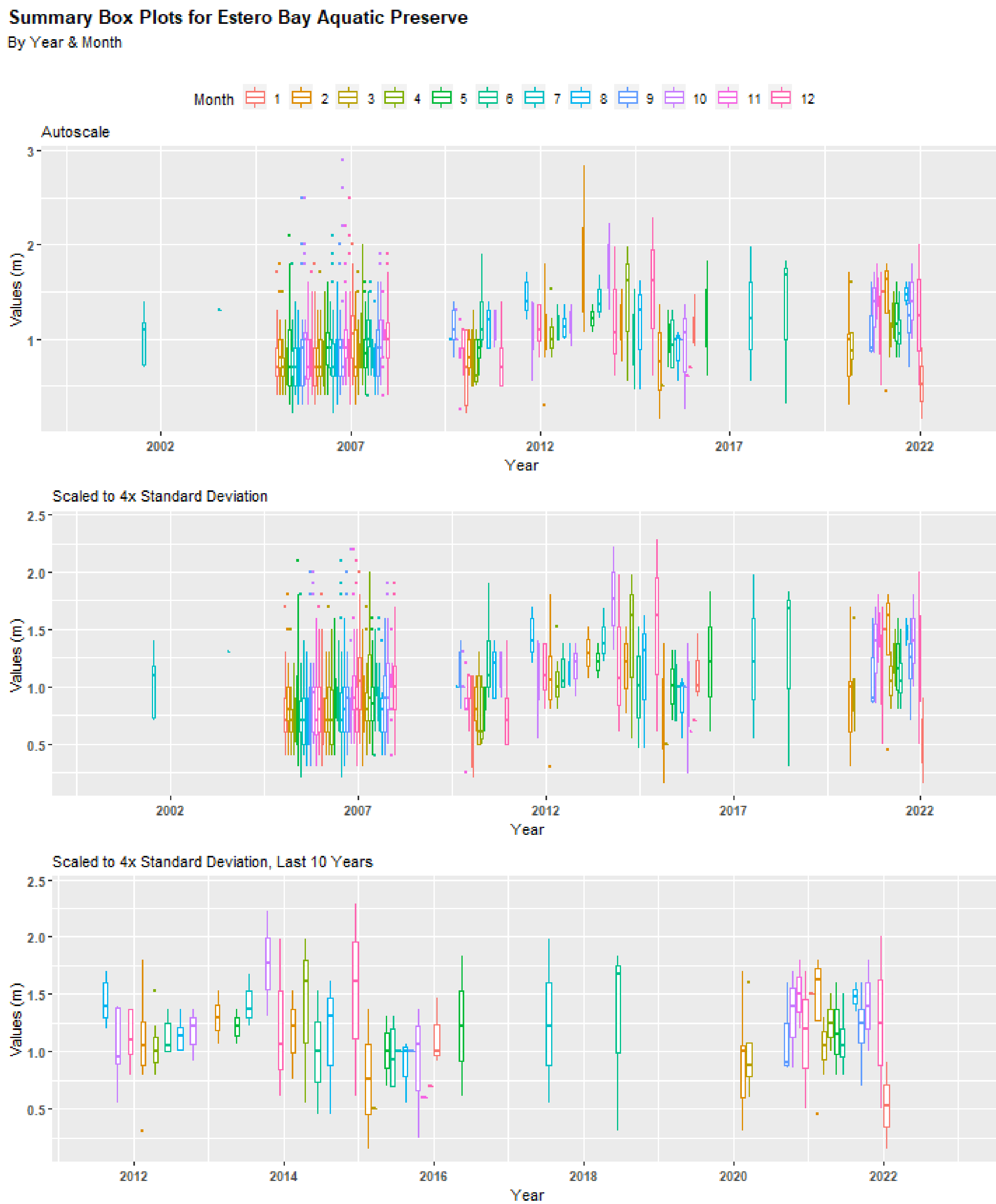
Summary Box Plots for Cockroach Bay Aquatic Preserve
By Month

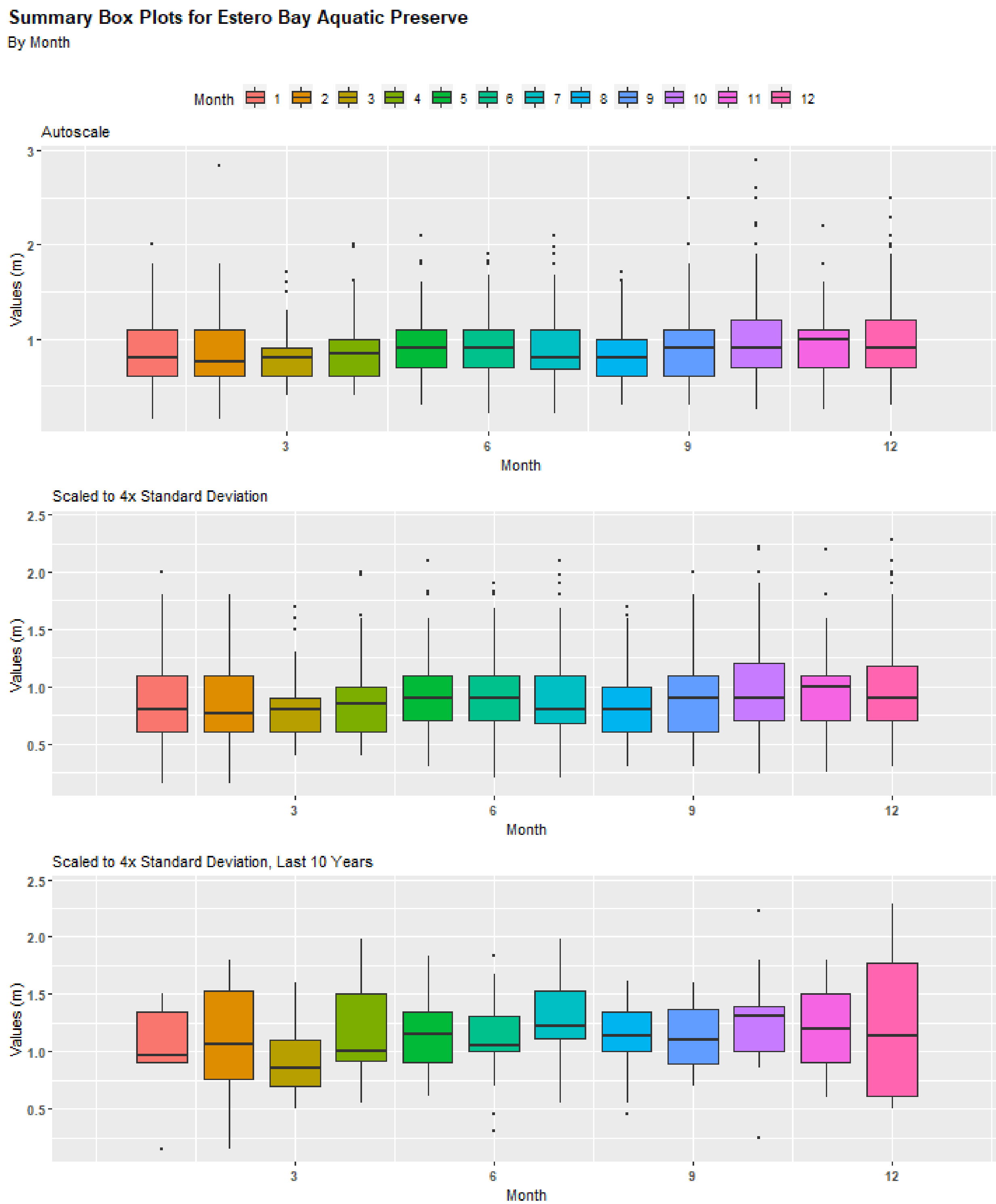


Summary Box Plots for Estero Bay Aquatic Preserve

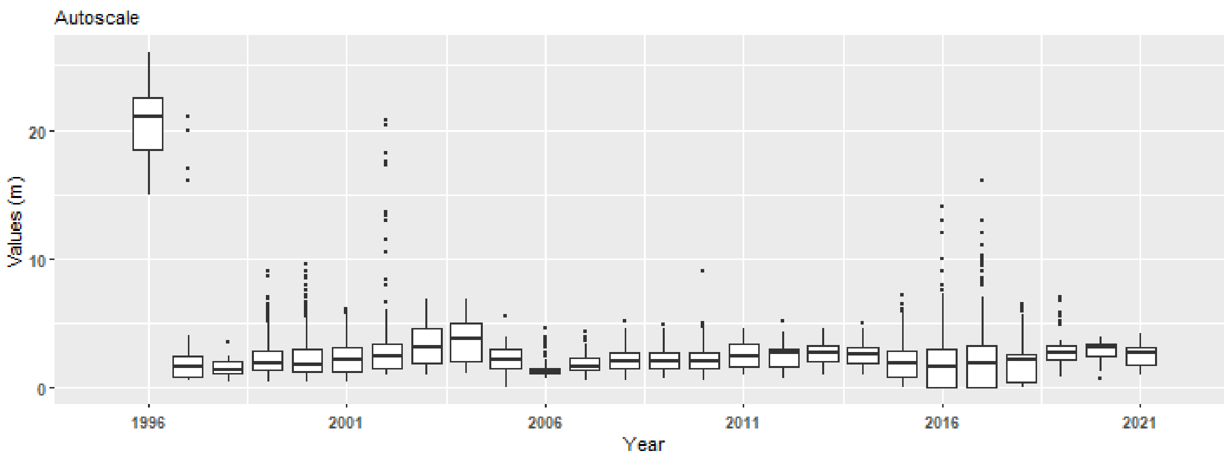
By Year

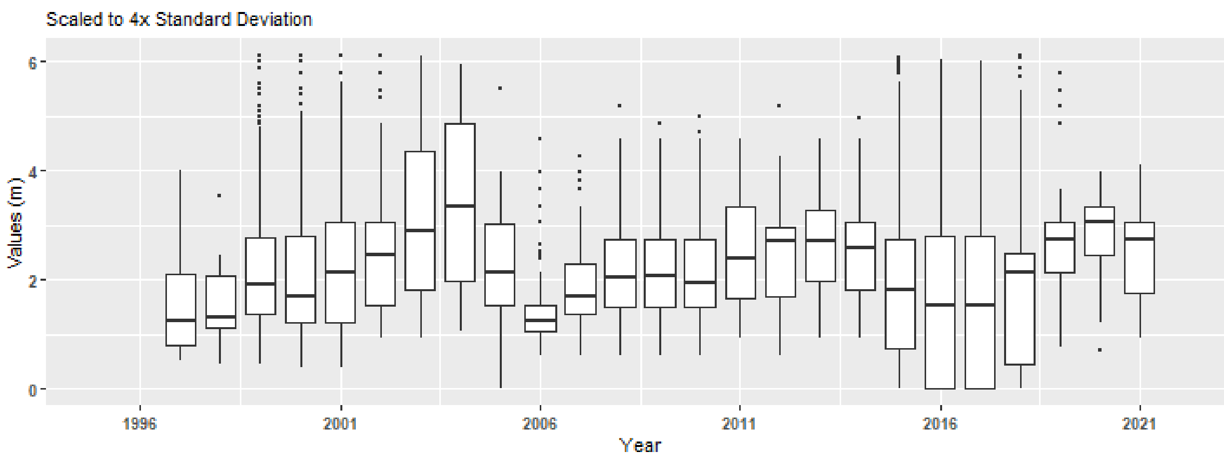


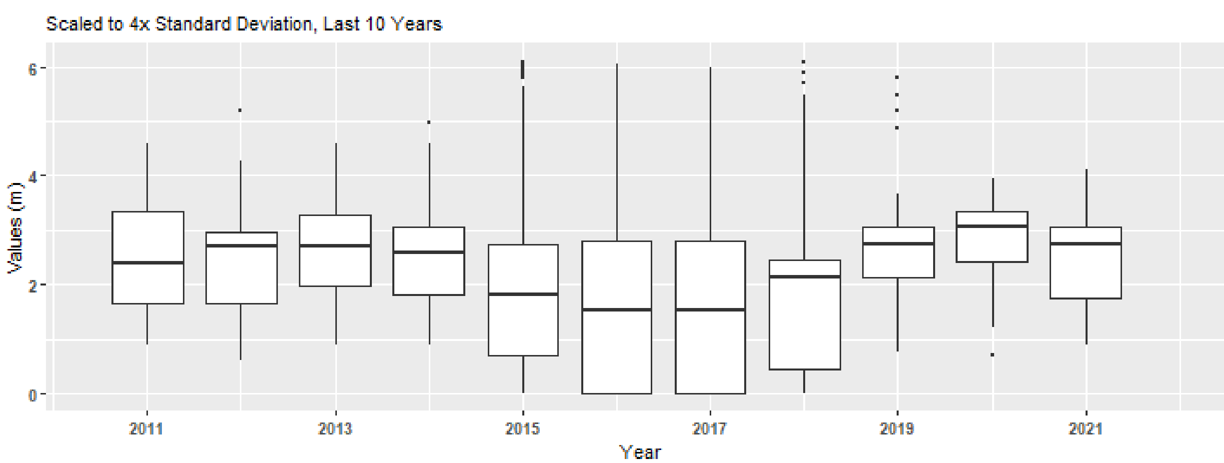

Summary Box Plots for Estero Bay Aquatic Preserve
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (m)
3
2
1
2002
2007
2012
2017
2022
Year
Scaled to 4x Standard Deviation
Values (m)
2.5
2.0
1.5
1.0
0.5
2002
2007
2012
2017
2022
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (m)
2.5
2.0
1.5
1.0
0.5
2012
2014
2016
2018
2020
2022
Year

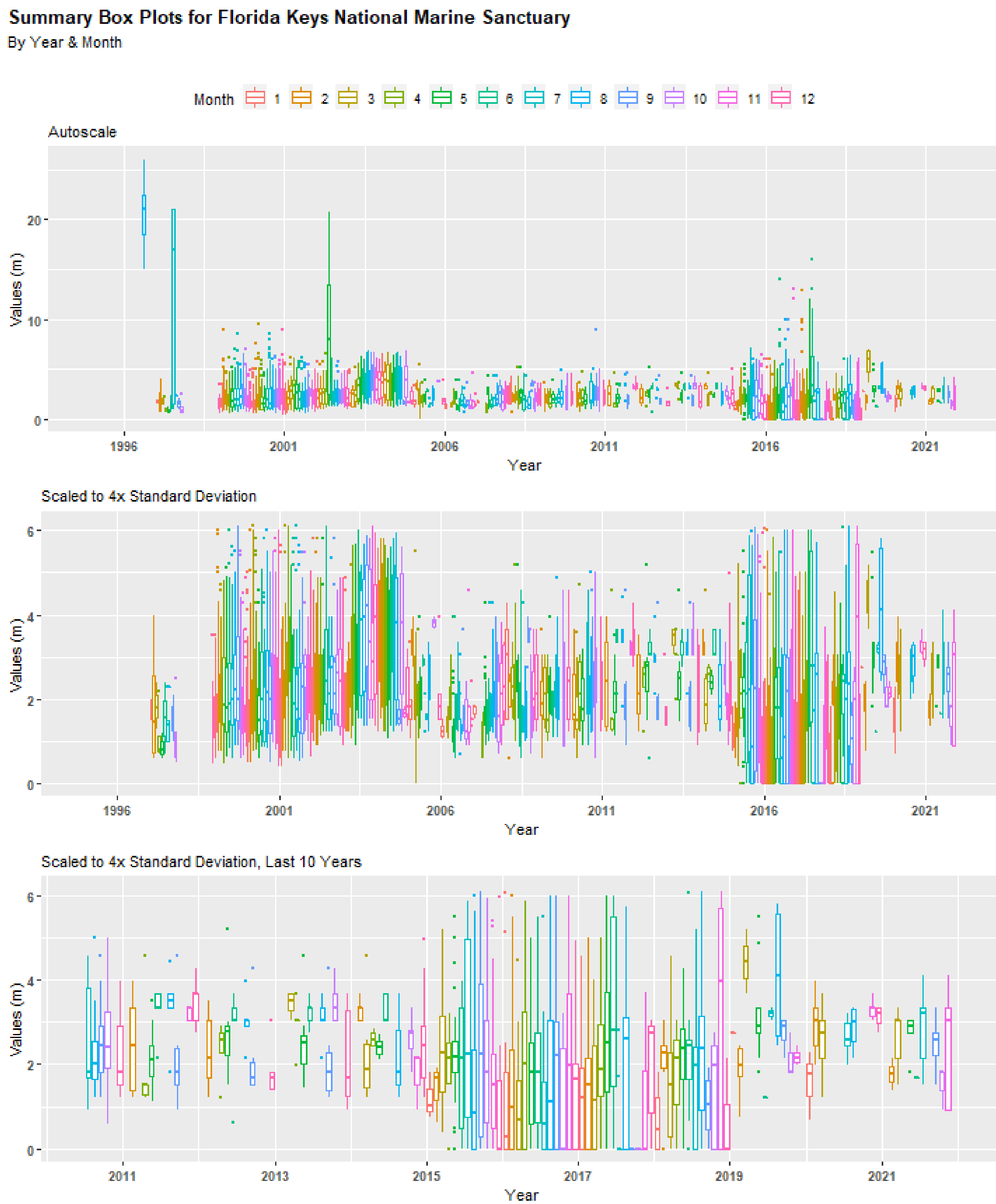

Summary Box Plots for Estero Bay Aquatic Preserve
By Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (m)
3
2
1
3
6
9
12
Month
Scaled to 4x Standard Deviation
Values (m)
2.5
2.0
1.5
1.0
0.5
3
6
9
12
Month
Scaled to 4x Standard Deviation, Last 10 Years
Values (m)
2.5
2.0
1.5
1.0
0.5
3
6
9
12
Month

Summary Box Plots for Florida Keys National Marine Sanctuary
By Year



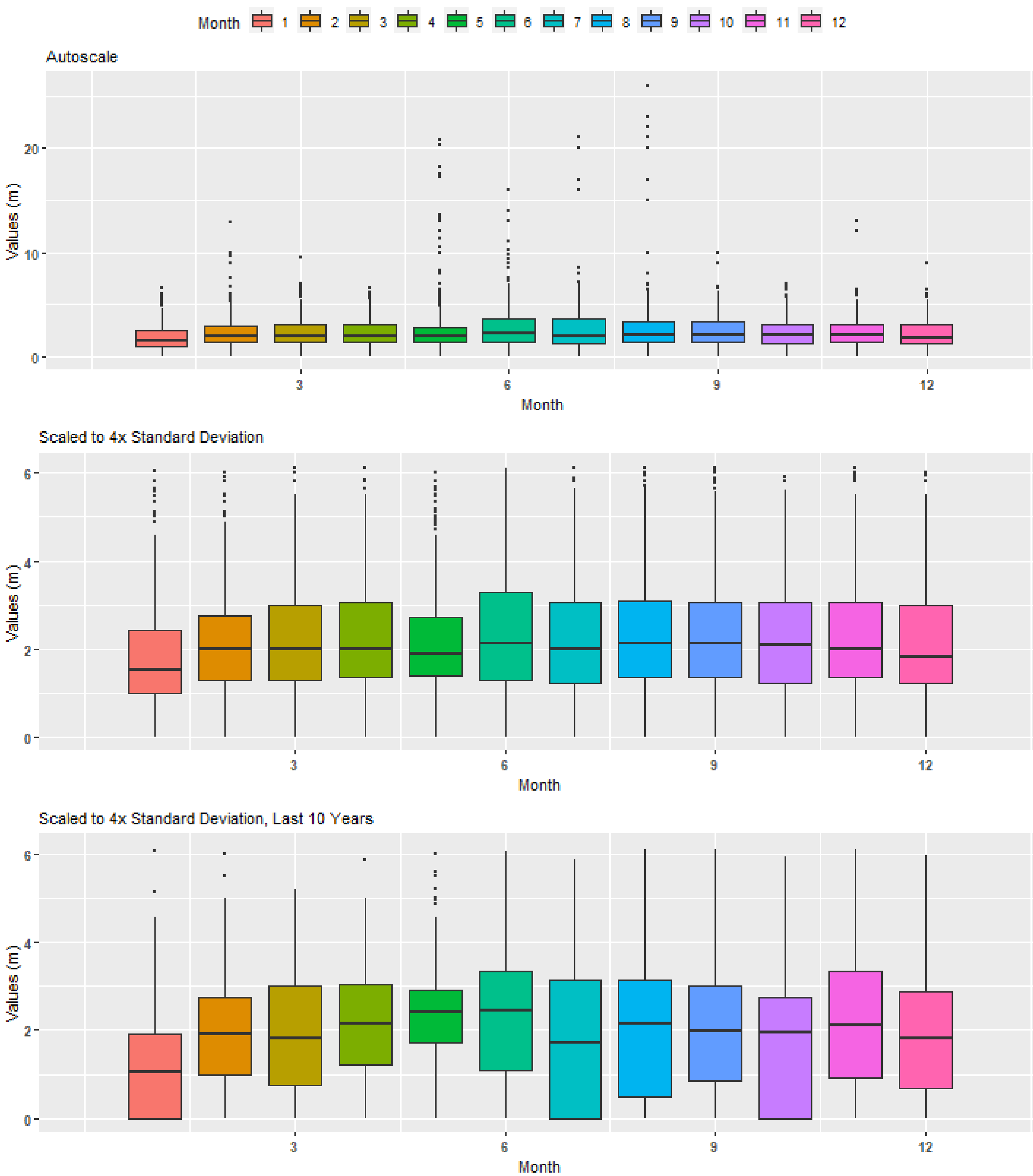






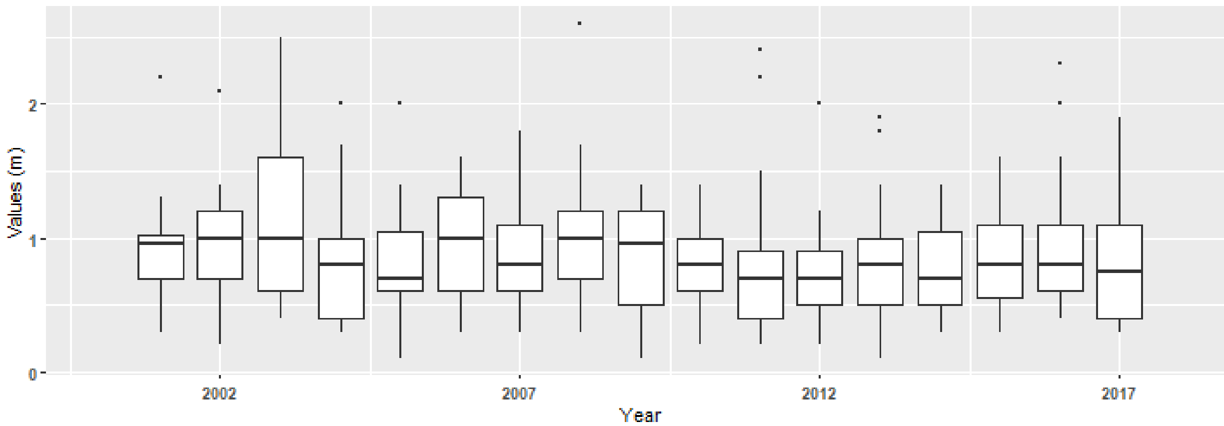
Summary Box Plots for Florida Keys National Marine Sanctuary
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (m)
0
10
20
1996
2001
2006
2011
2016
2021
Year
Scaled to 4x Standard Deviation
Values (m)
0
2
4
6
1996
2001
2006
2011
2016
2021
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (m)
0
2
4
6
2011
2013
2015
2017
2019
2021
Year

Summary Box Plots for Florida Keys National Marine Sanctuary
By Month

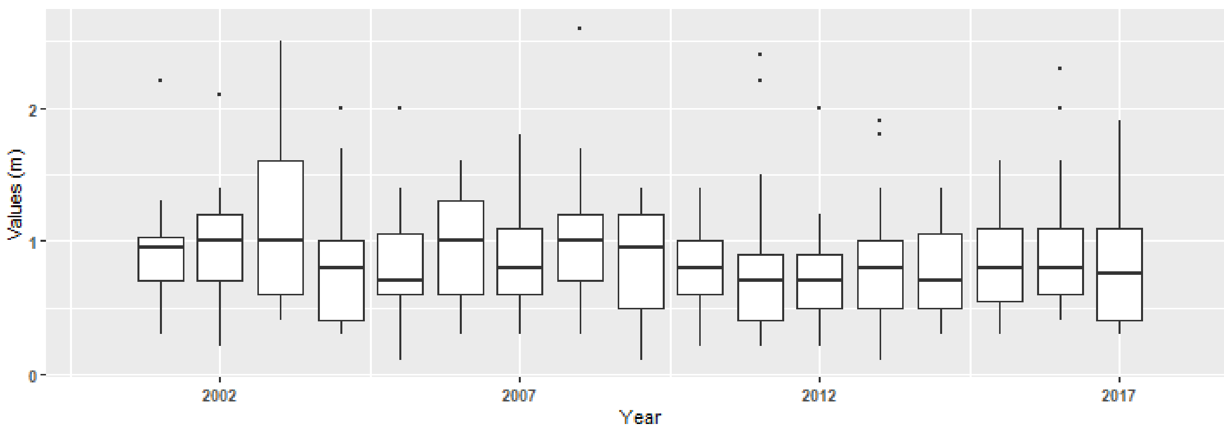


Summary Box Plots for Fort Clinch State Park Aquatic Preserve
By Year

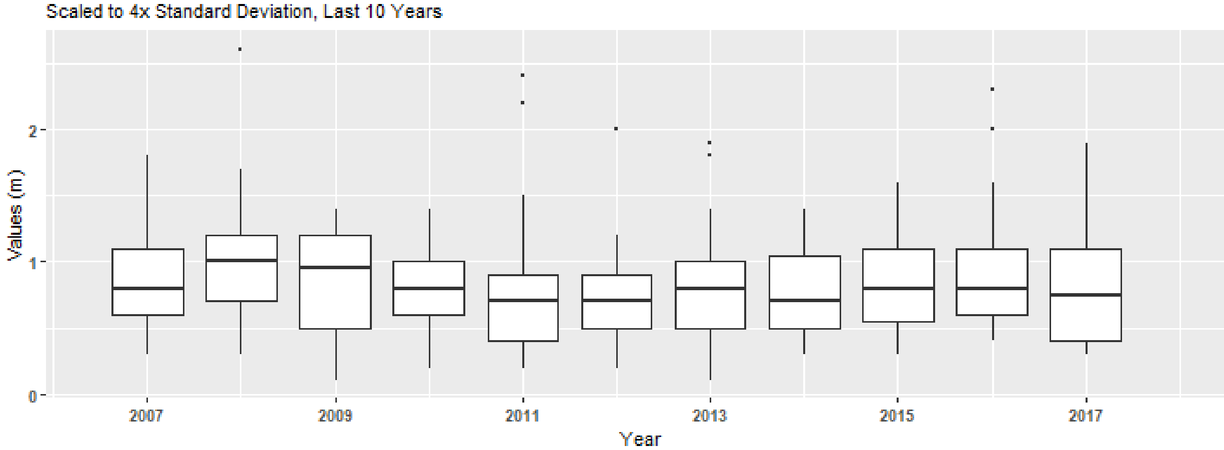
Autoscale



Scaled to 4x Standard Deviation

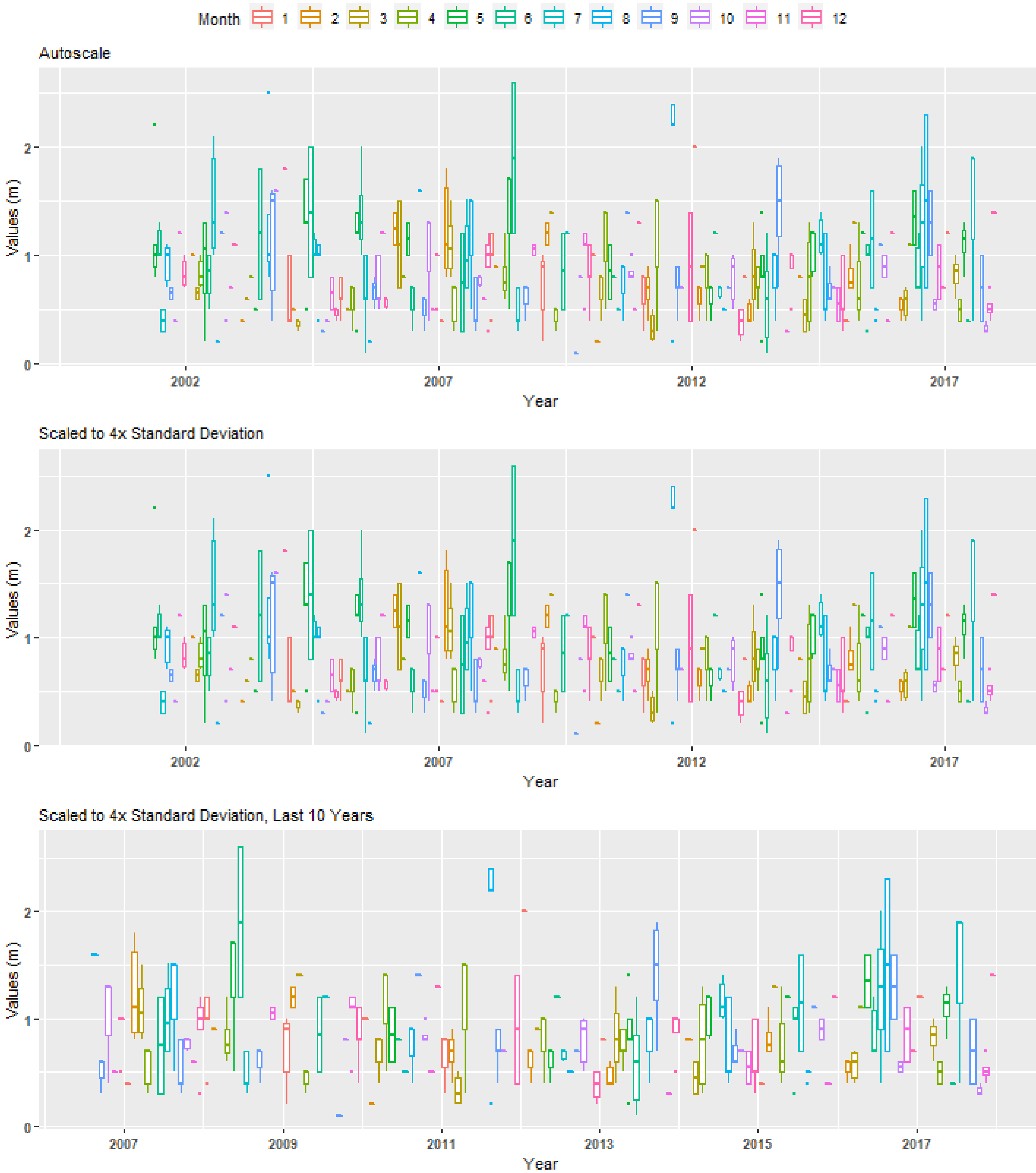


Scaled to 4x Standard Deviation, Last 10 Years

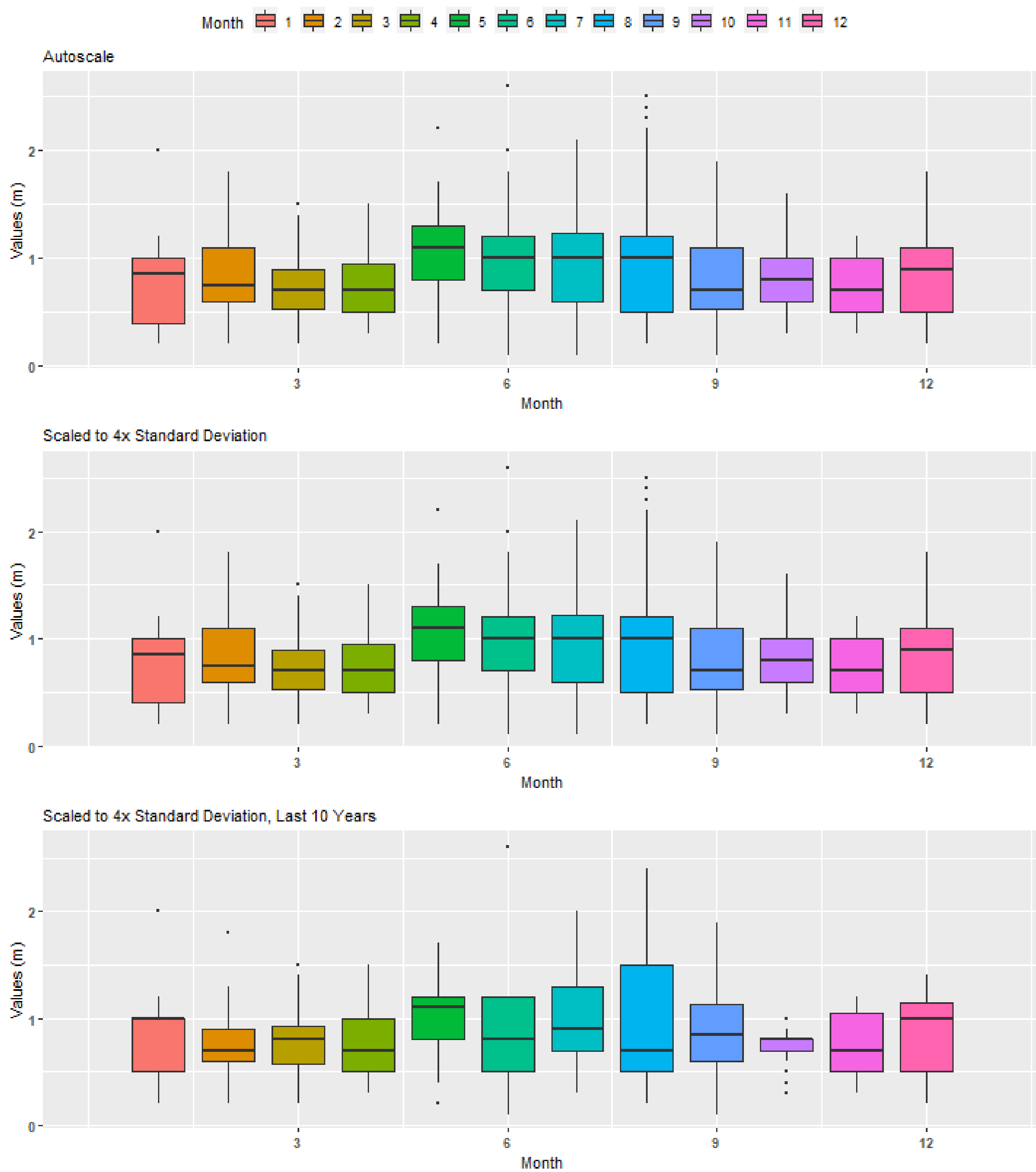


Summary Box Plots for Fort Clinch State Park Aquatic Preserve

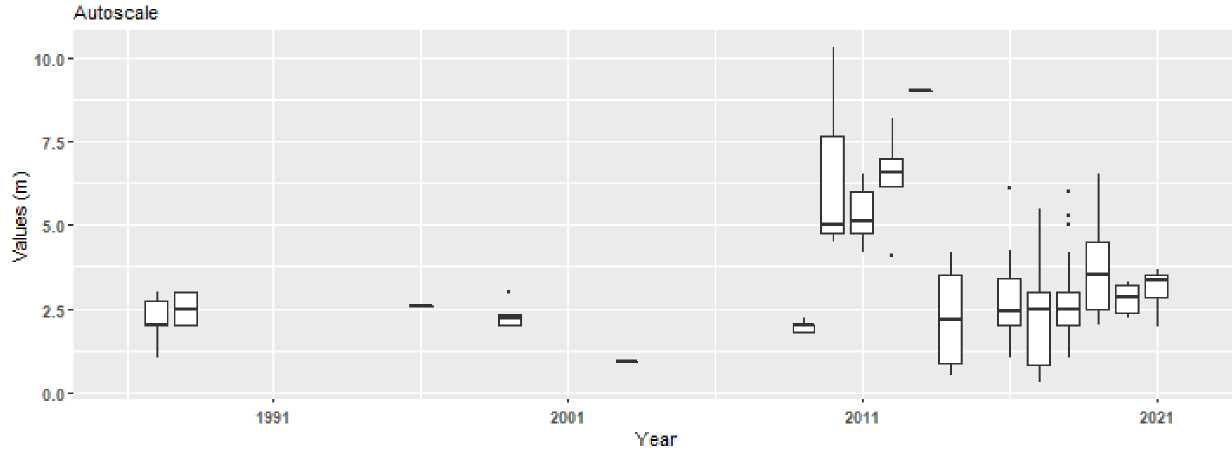
By Year & Month

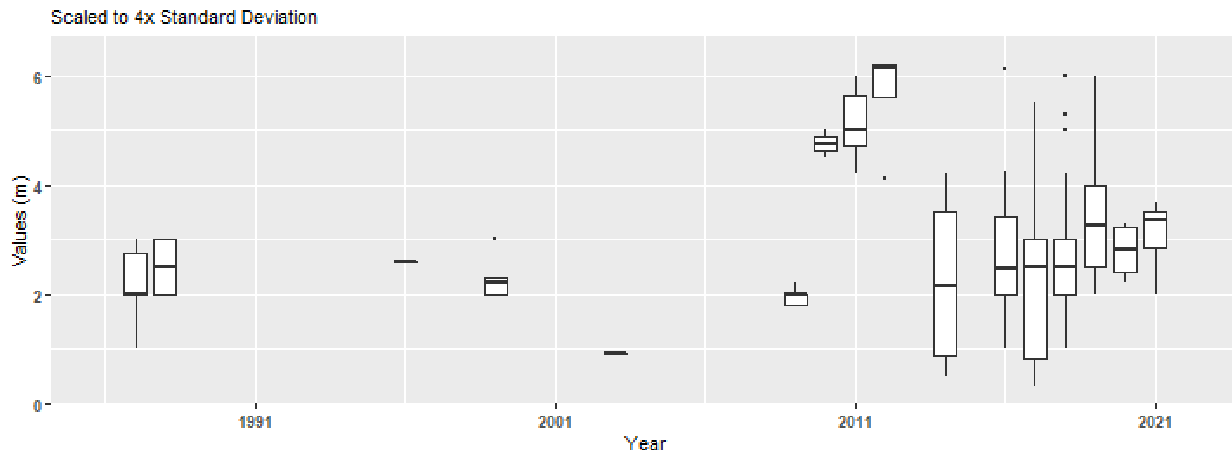


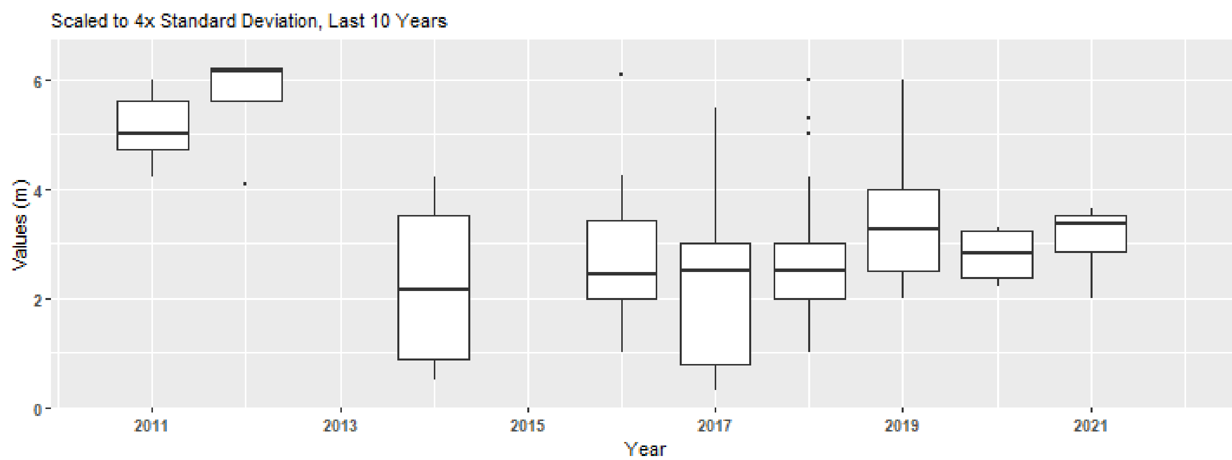
Summary Box Plots for Fort Clinch State Park Aquatic Preserve
By Month



Summary Box Plots for Fort Pickens State Park Aquatic Preserve
By Year

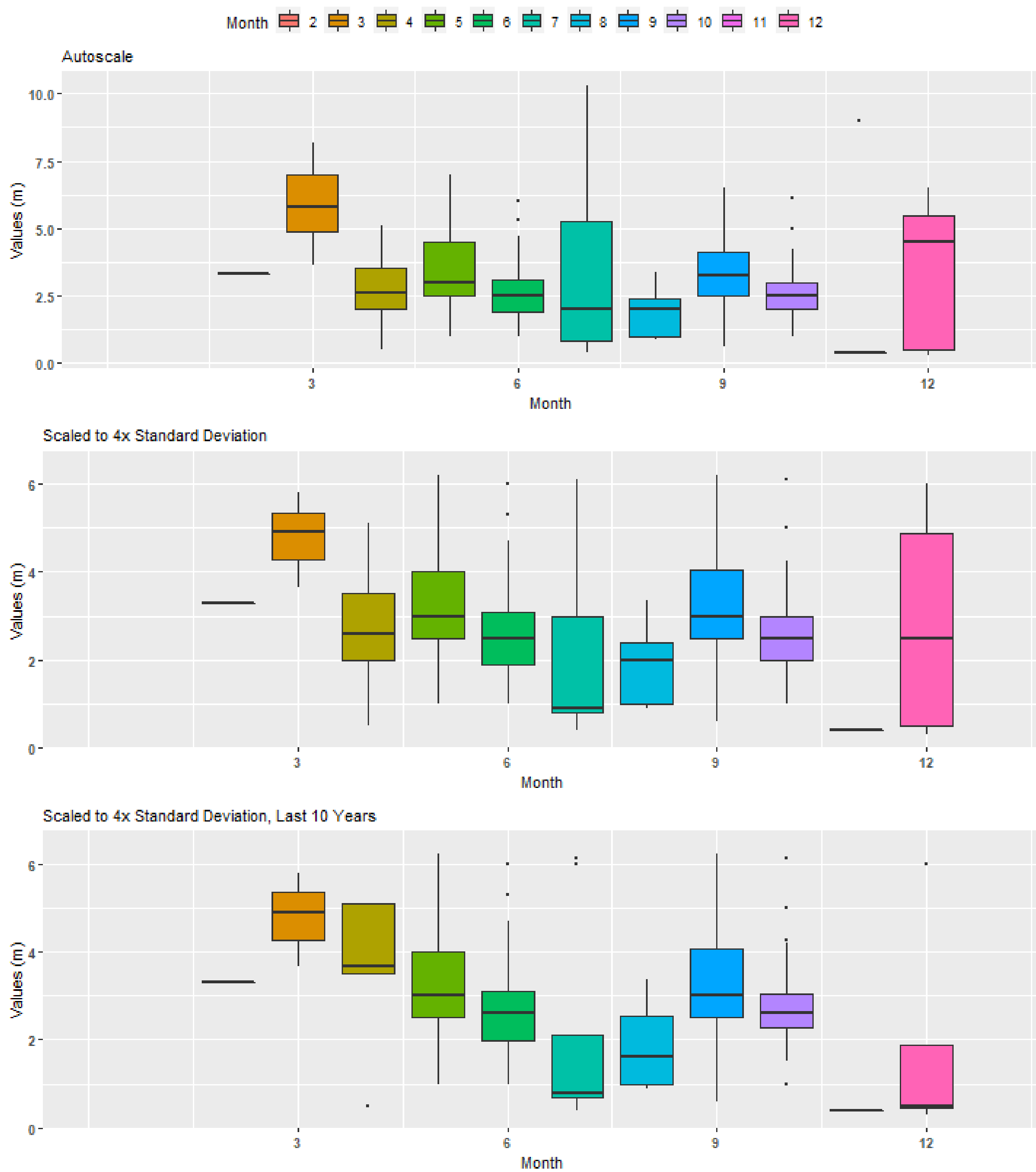






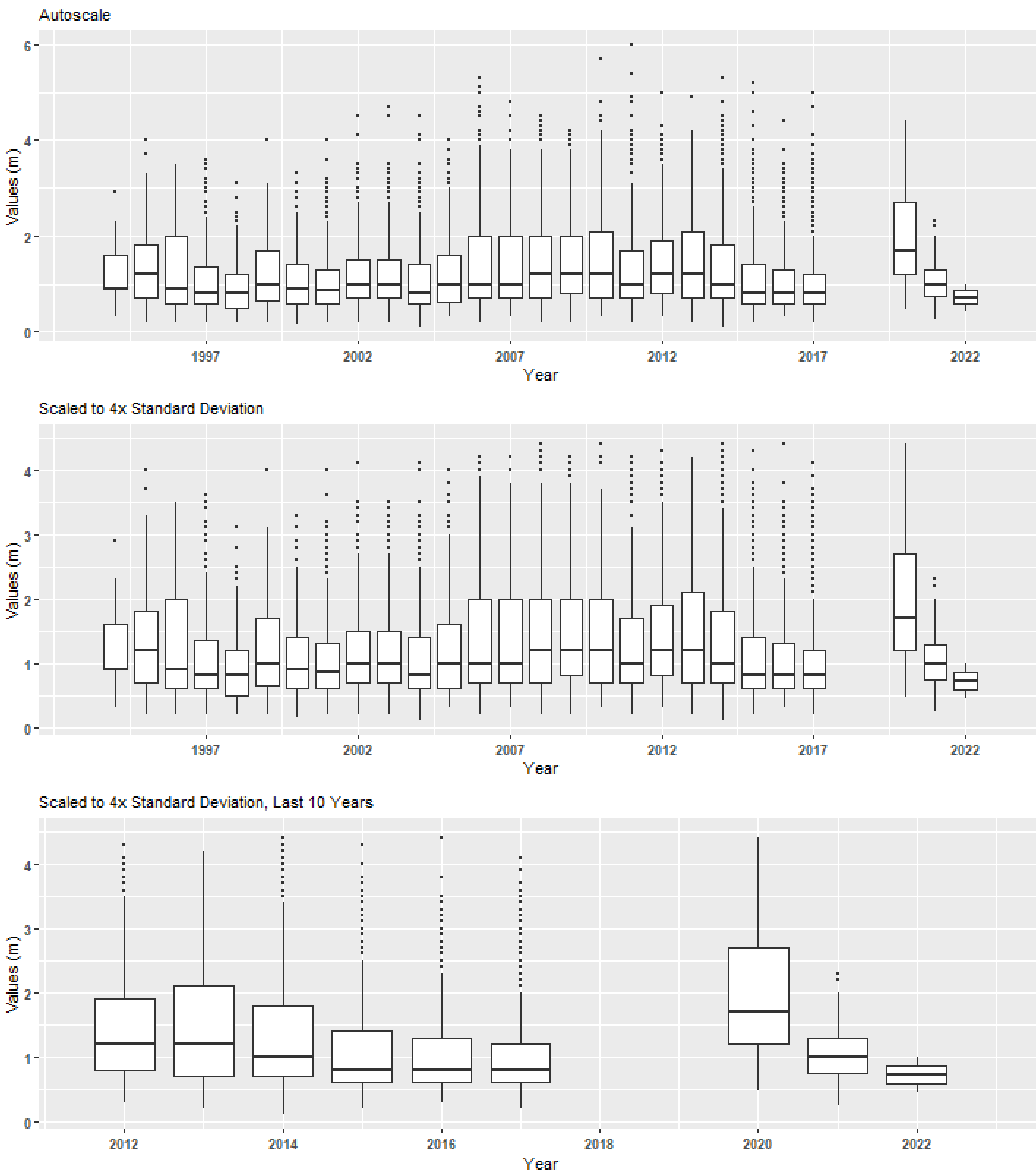

Summary Box Plots for Fort Pickens State Park Aquatic Preserve
By Year & Month
Month 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (m)
10.0
7.5
5.0
2.5
0.0
1991
2001
2011
2021
Year
Scaled to 4x Standard Deviation
Values (m)
6
4
2
0
1991
2001
2011
2021
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (m)
6
4
2
0
2011
2013
2015
2017
2019
2021
Year

Summary Box Plots for Fort Pickens State Park Aquatic Preserve
By Month



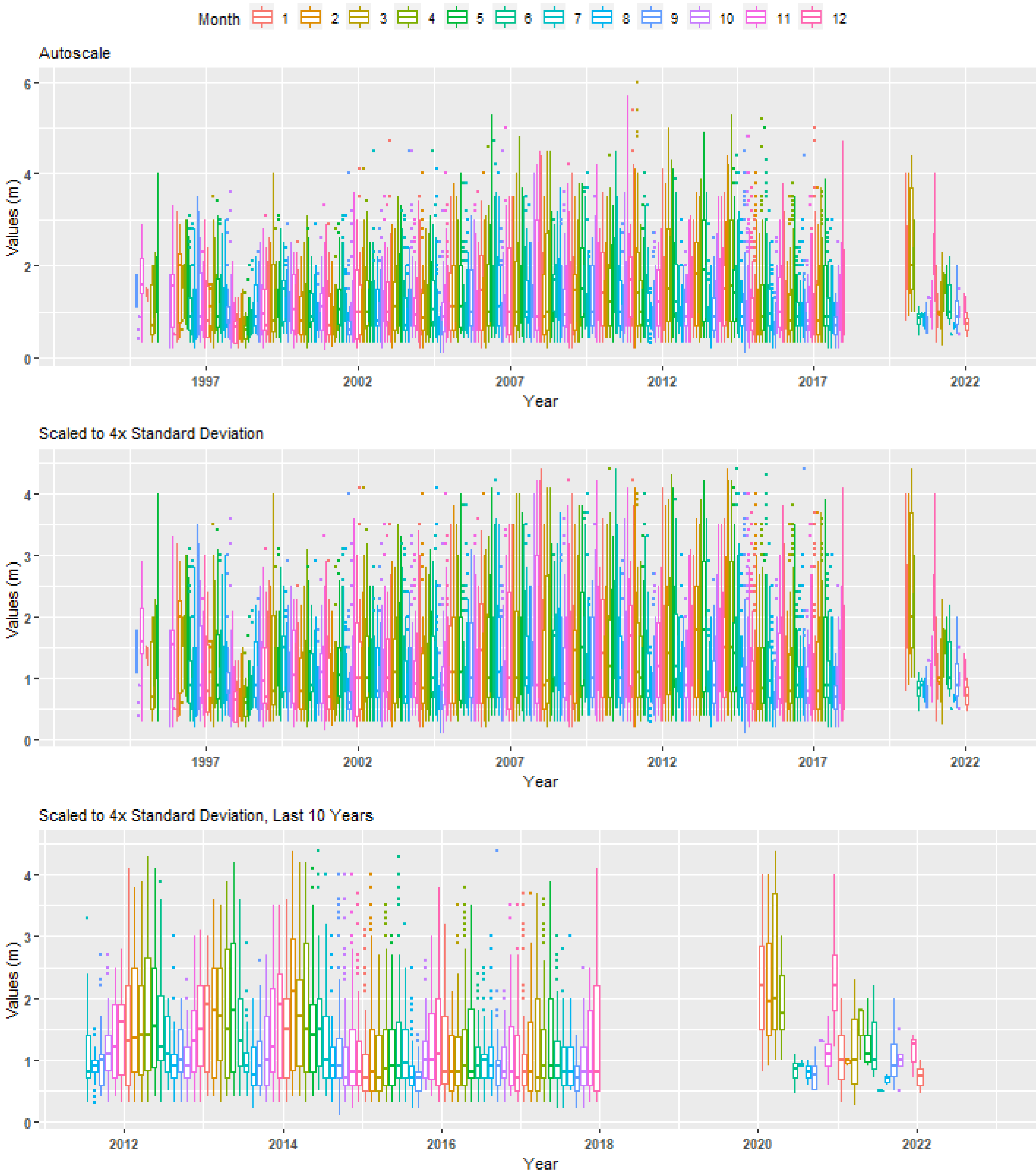
Summary Box Plots for Gasparilla Sound-Charlotte Harbor Aquatic Preserve

By Year

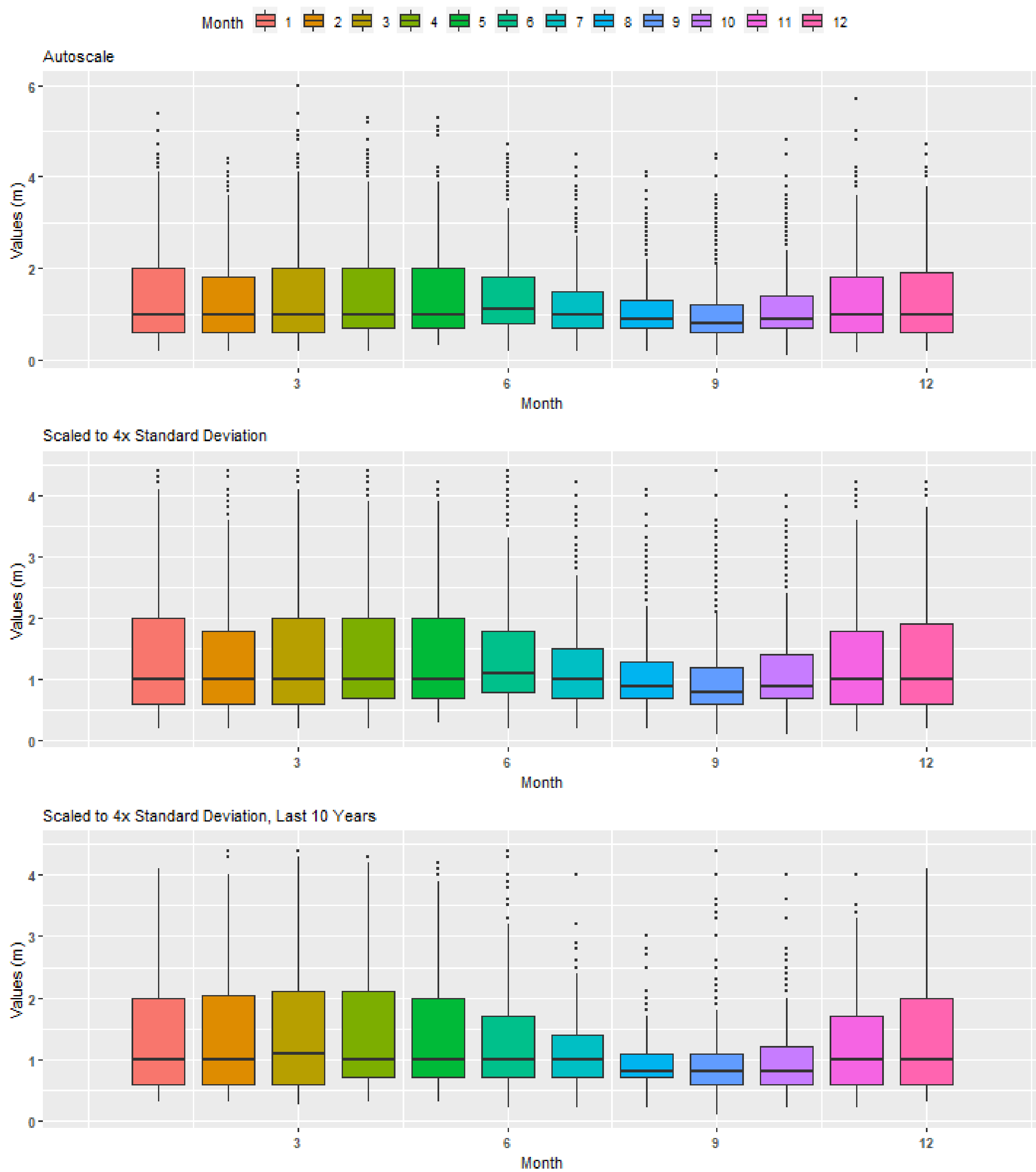


Summary Box Plots for Gasparilla Sound-Charlotte Harbor Aquatic Preserve

By Year & Month

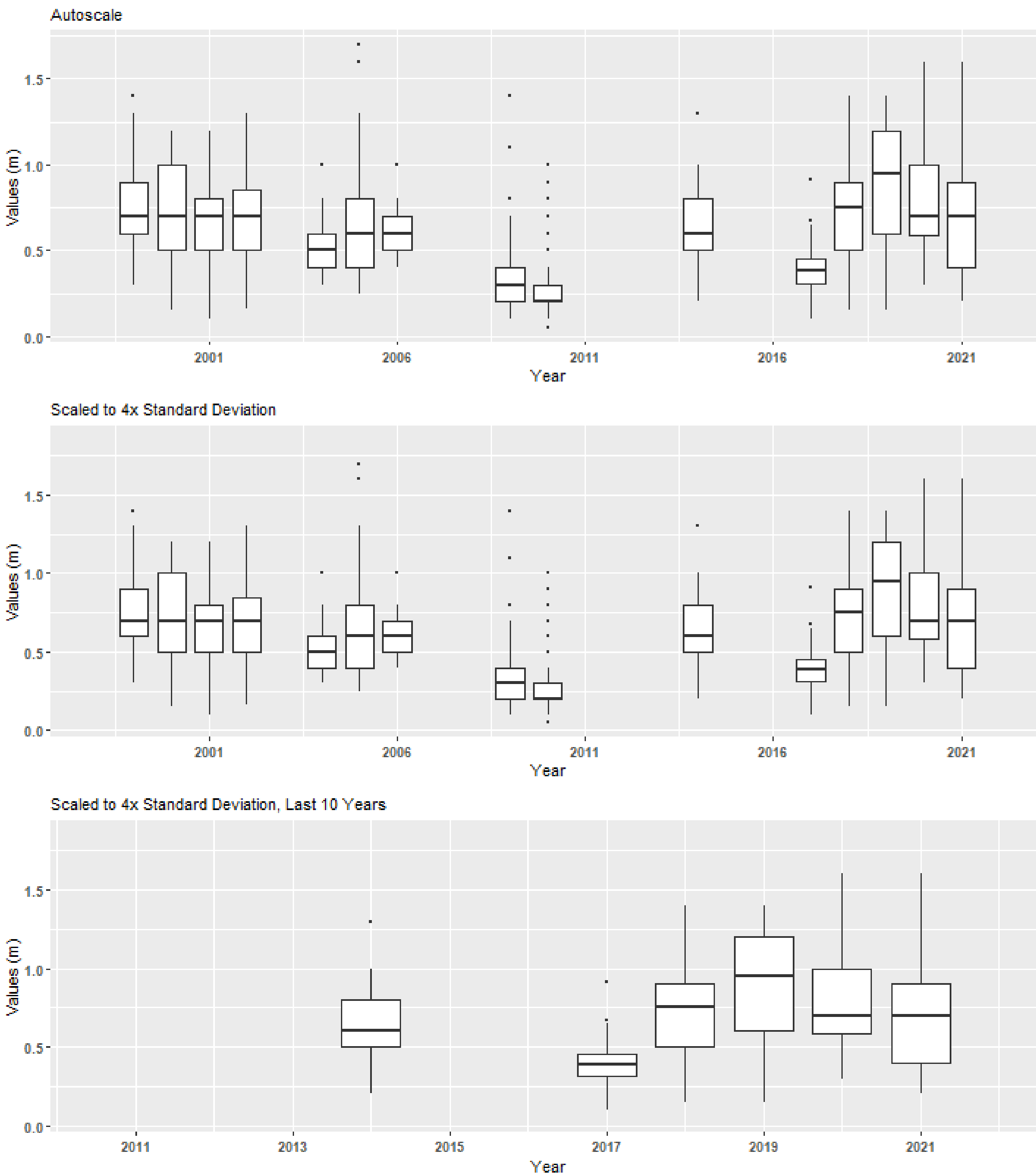


Summary Box Plots for Gasparilla Sound-Charlotte Harbor Aquatic Preserve
By Month



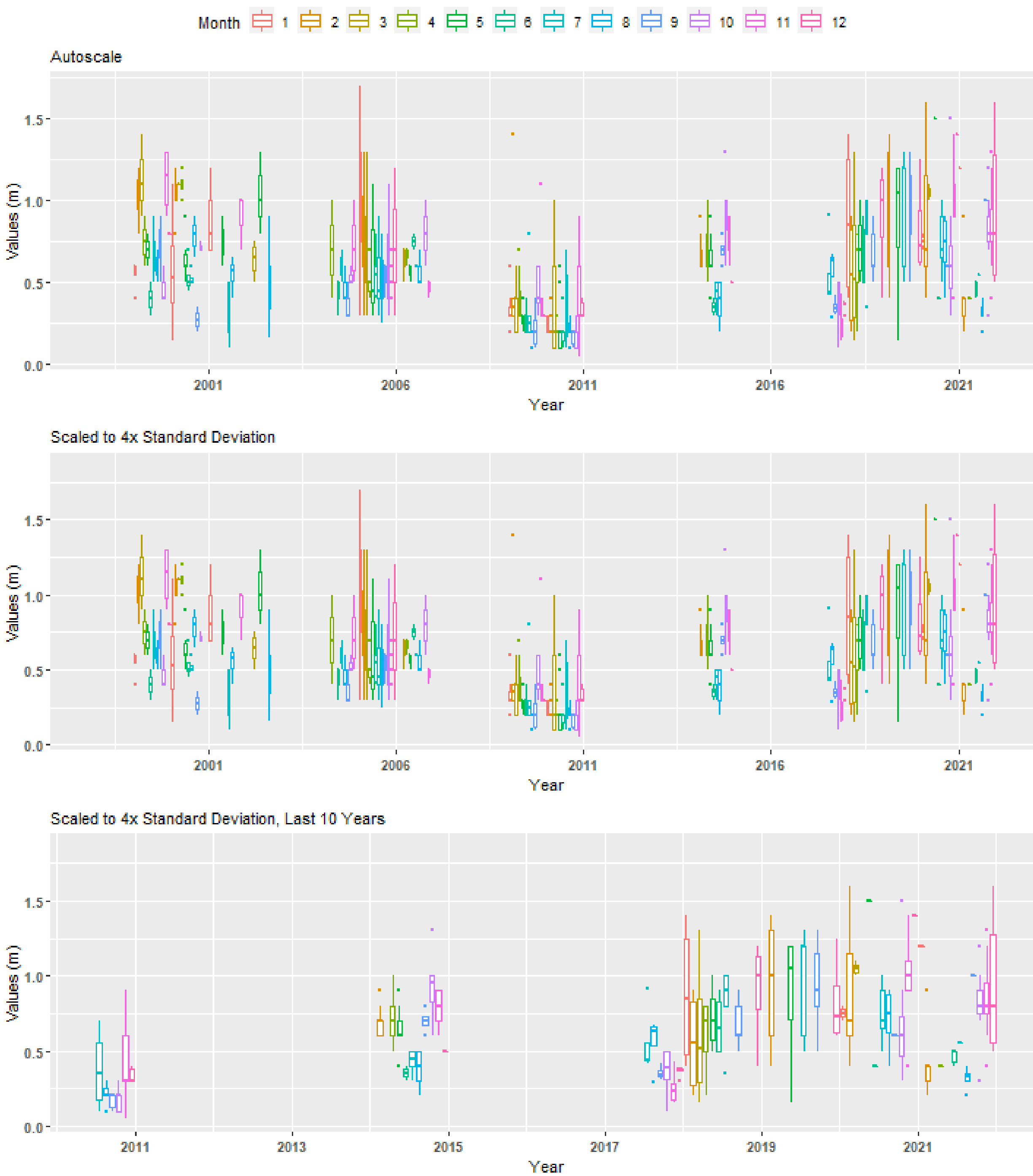
Summary Box Plots for Guana River Marsh Aquatic Preserve

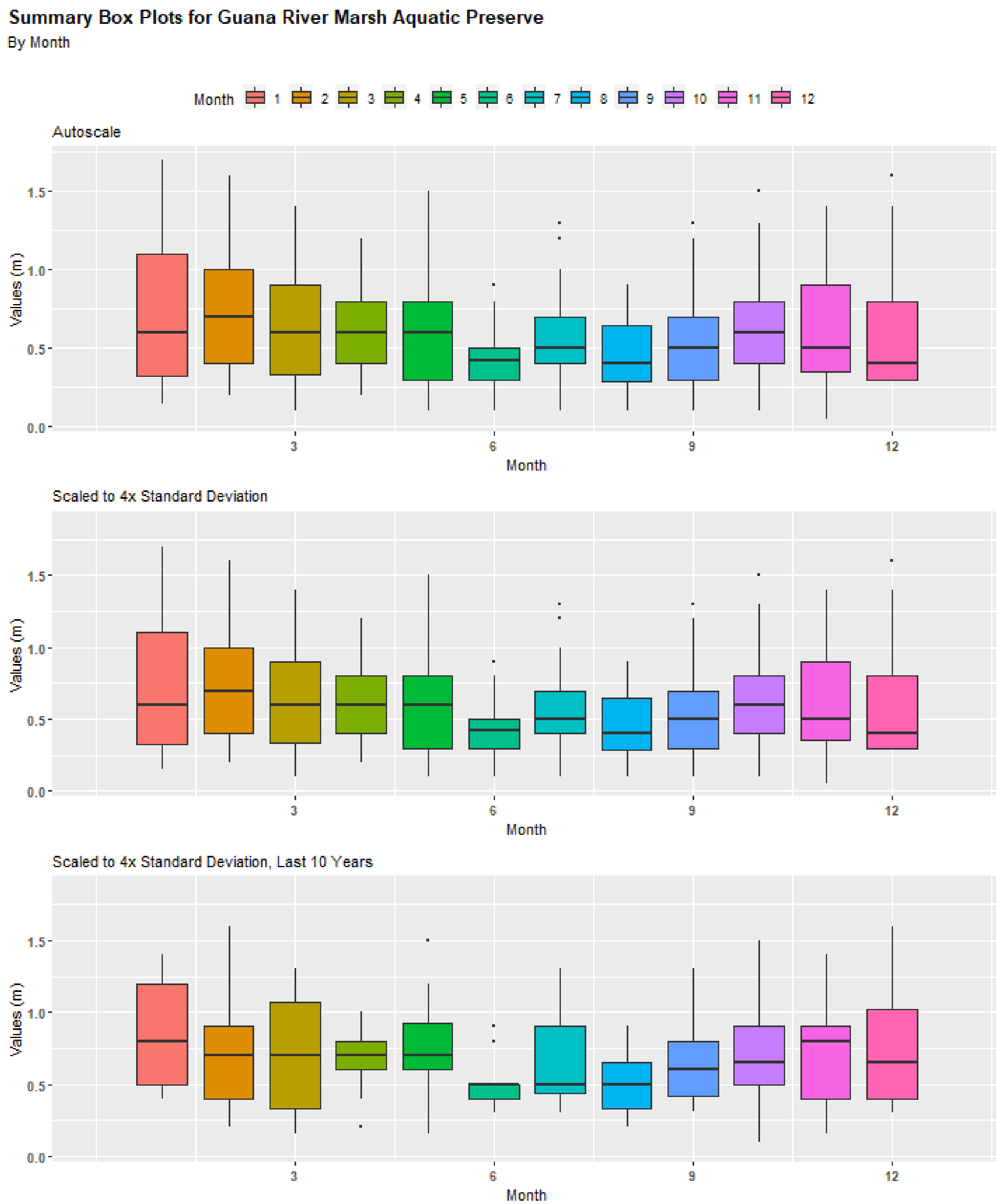
By Year



Summary Box Plots for Guana River Marsh Aquatic Preserve

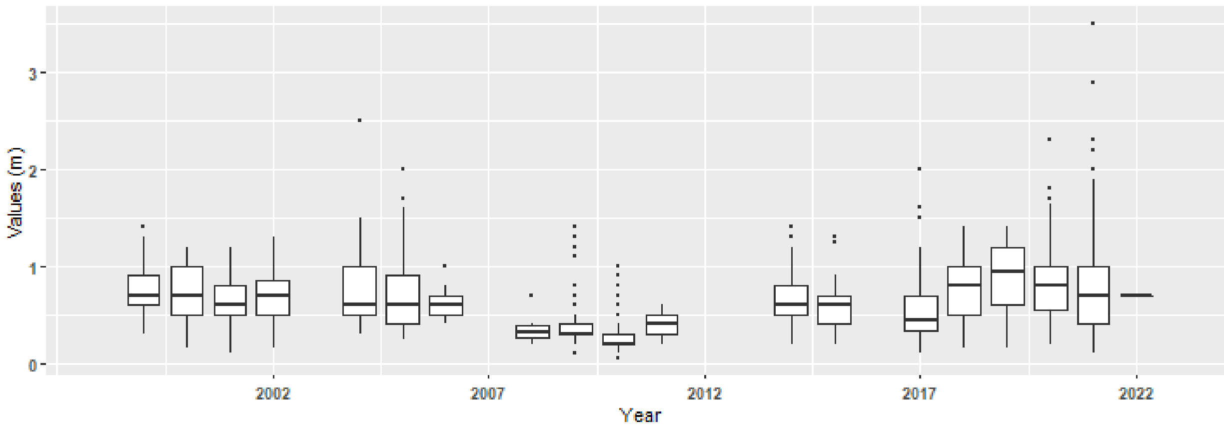
By Year & Month



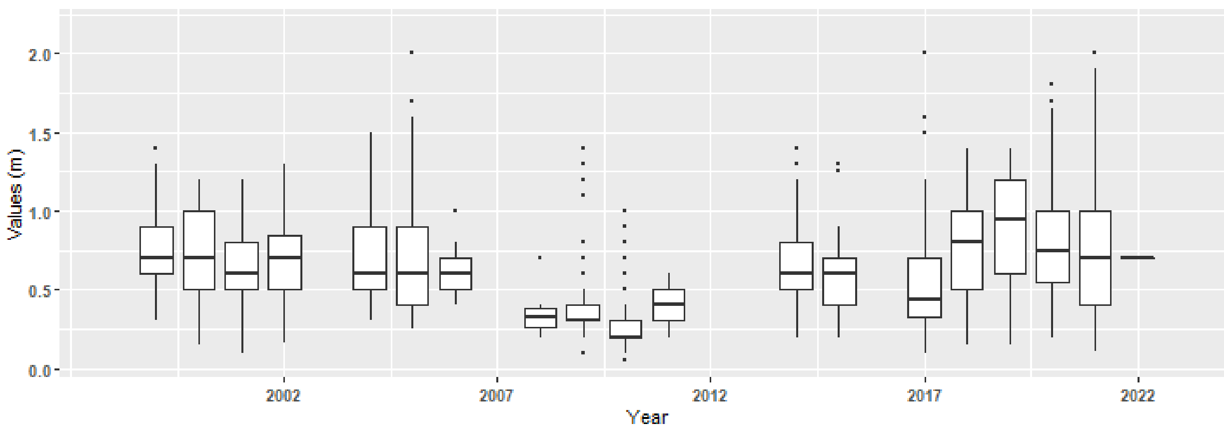

Summary Box Plots for Guana River Marsh Aquatic Preserve
By Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (m)
0.0
0.5
1.0
1.5
3
6
9
12
Month
Scaled to 4x Standard Deviation
Values (m)
0.0
0.5
1.0
1.5
3
6
9
12
Month
Scaled to 4x Standard Deviation, Last 10 Years
Values (m)
0.0
0.5
1.0
1.5
3
6
9
12
Month

Summary Box Plots for Guana Tolomato Matanzas National Estuarine Research Reserve
By Year

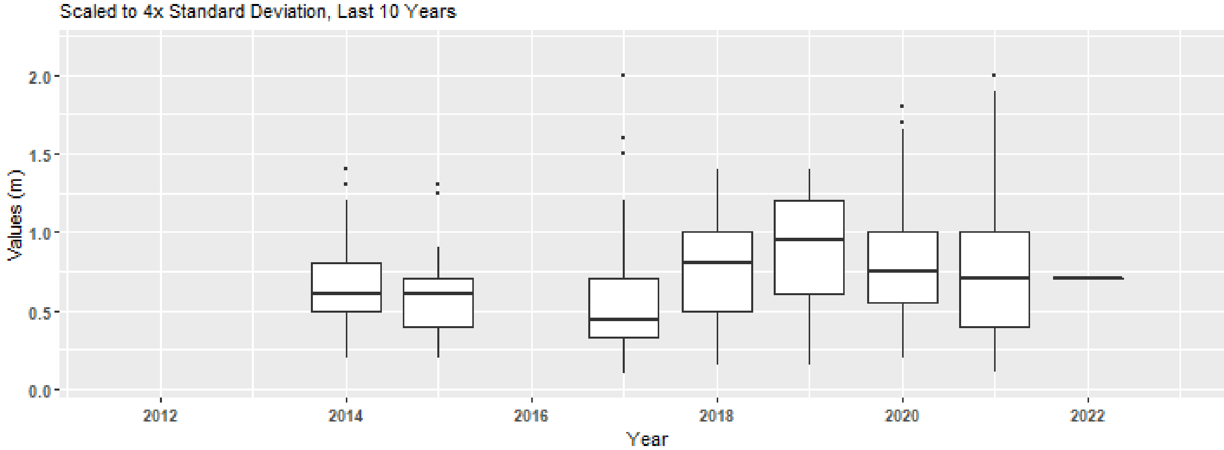
Autoscale



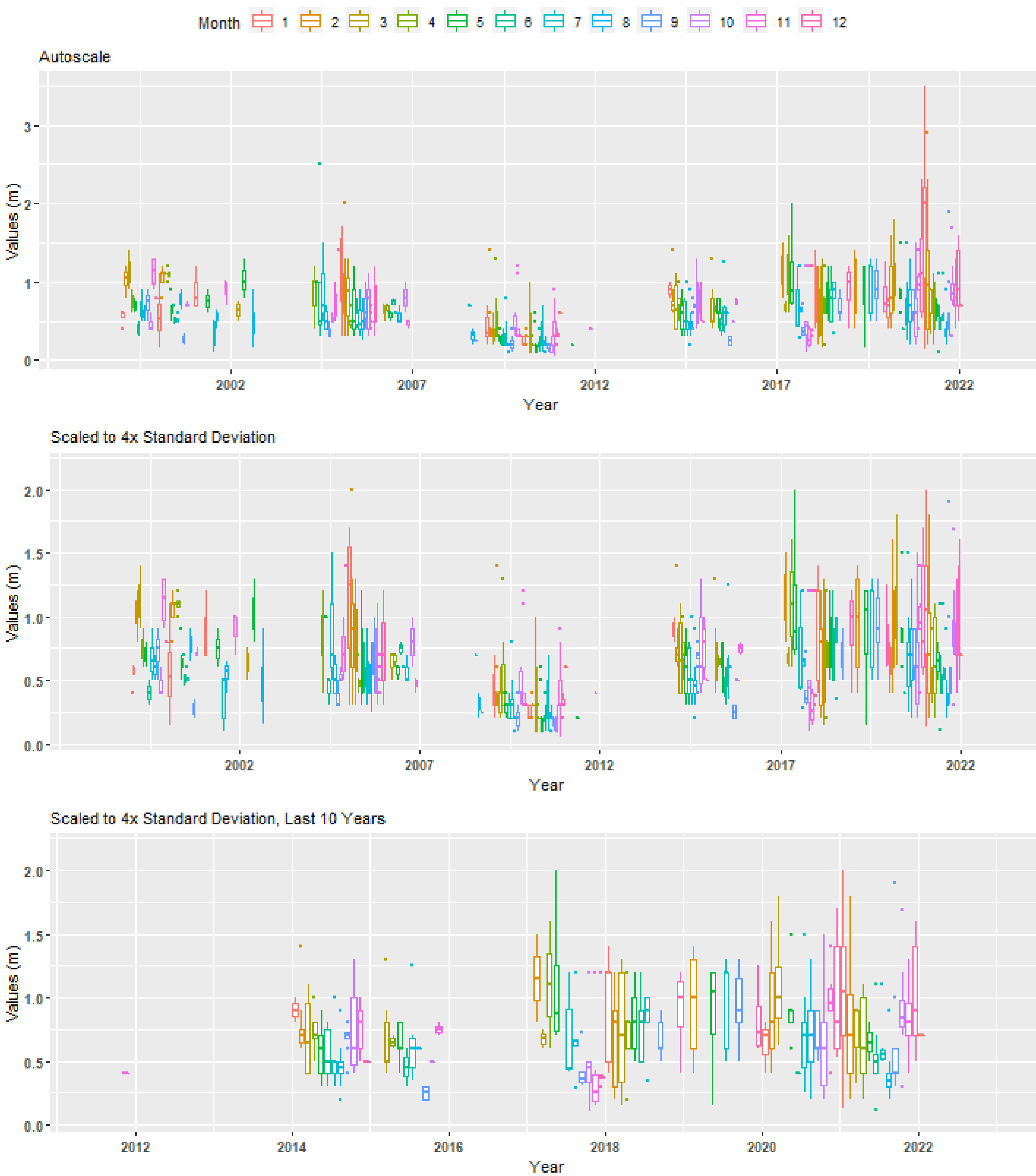
Scaled to 4x Standard Deviation



Scaled to 4x Standard Deviation, Last 10 Years

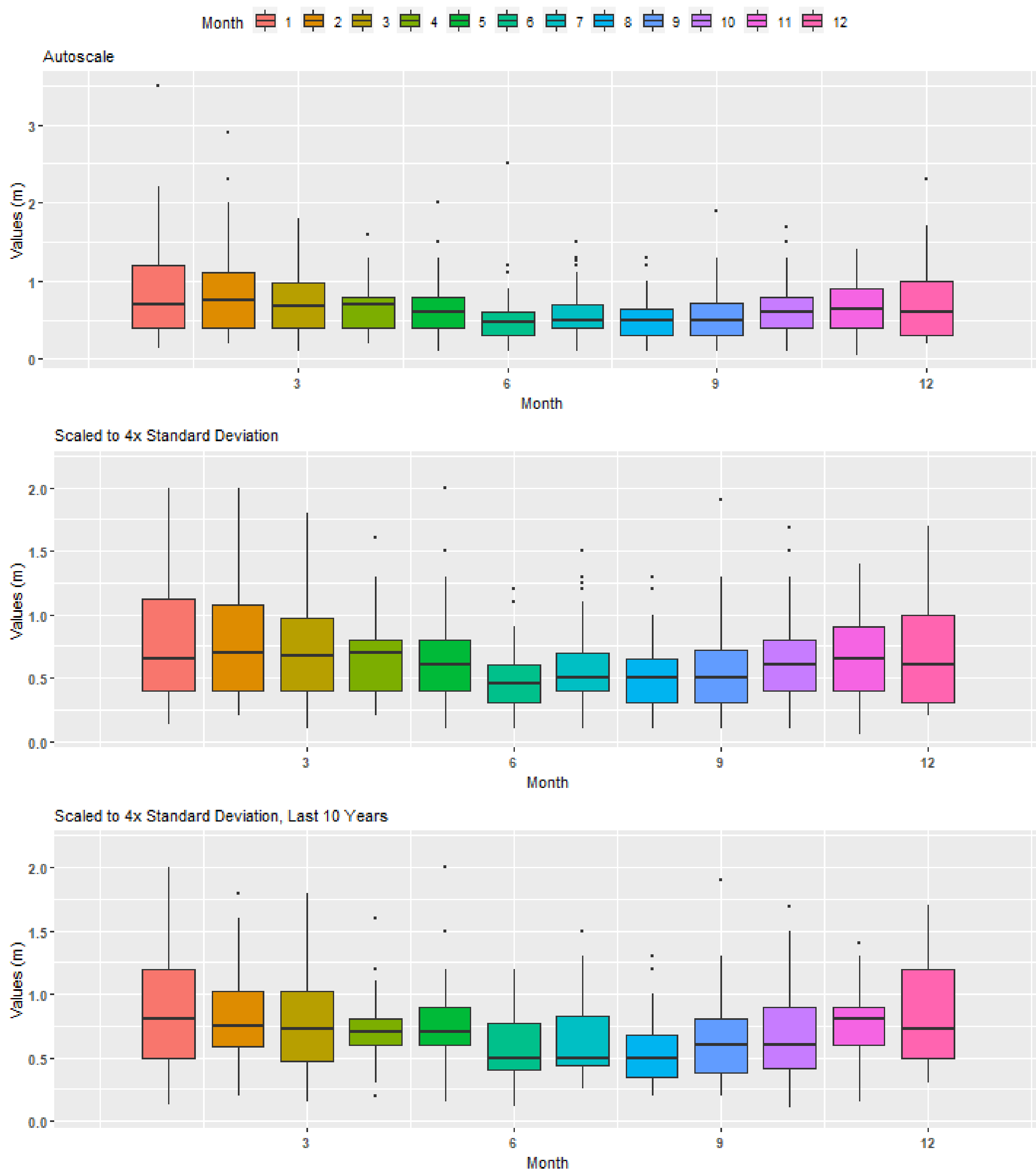


Summary Box Plots for Guana Tolomato Matanzas National Estuarine Research Reserve
By Year & Month



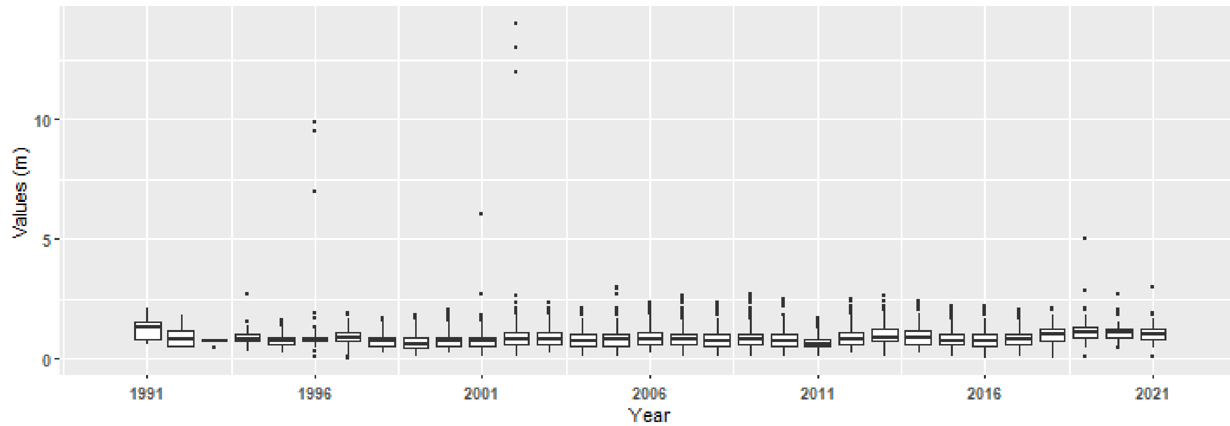
Summary Box Plots for Guana Tolomato Matanzas National Estuarine Research Reserve

By Month

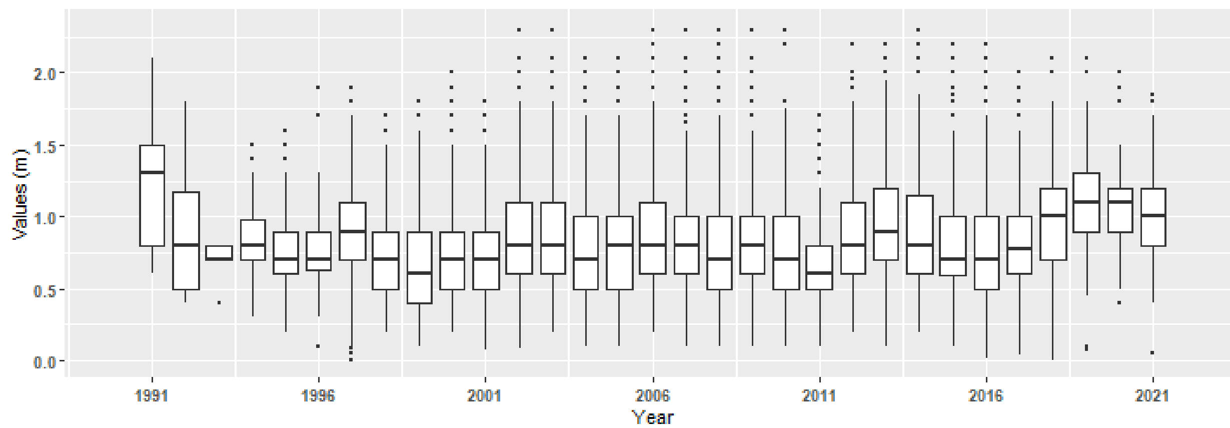


Summary Box Plots for Indian River-Malabar to Vero Beach Aquatic Preserve
By Year

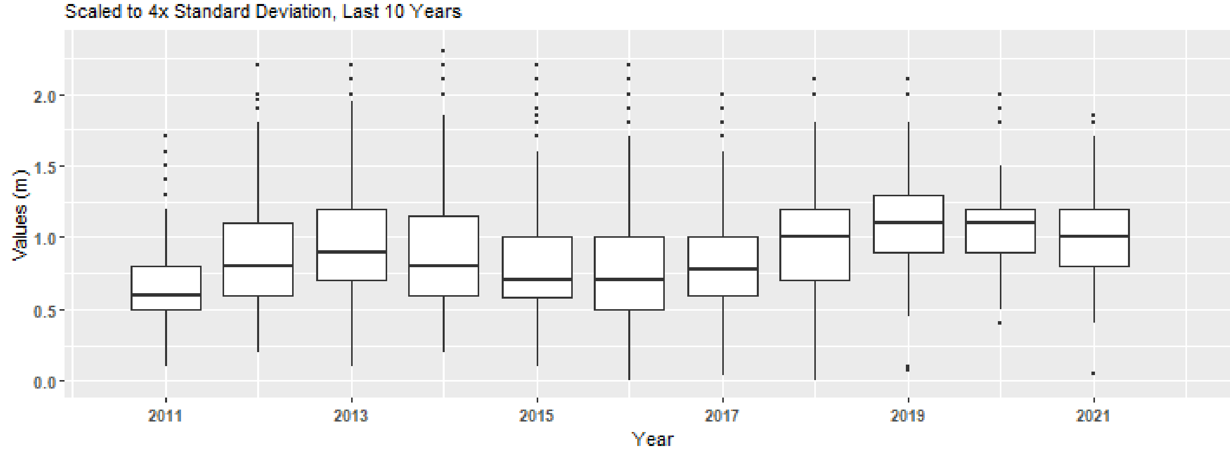
Autoscale



Scaled to 4x Standard Deviation

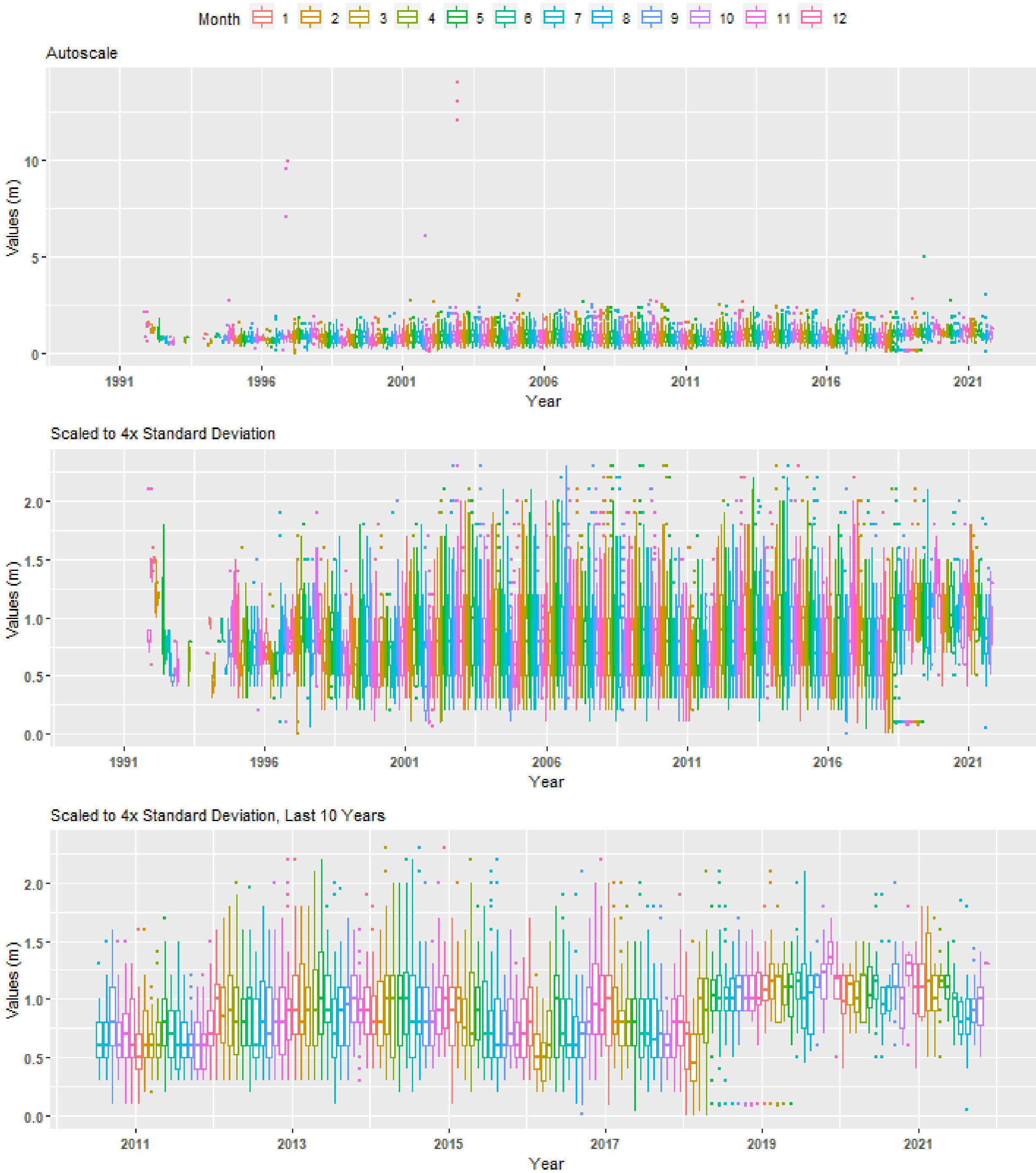


Scaled to 4x Standard Deviation, Last 10 Years

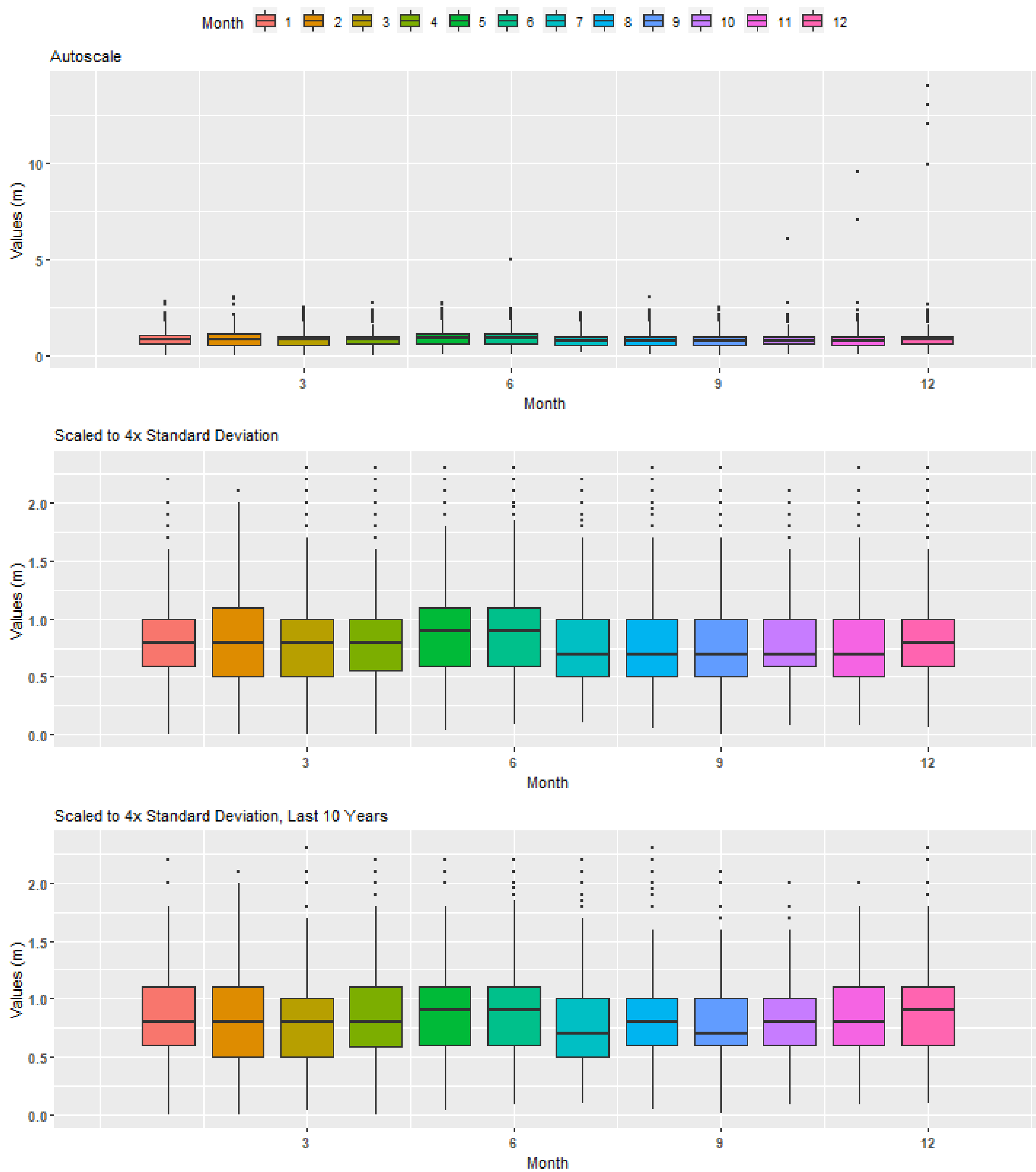


Summary Box Plots for Indian River-Malabar to Vero Beach Aquatic Preserve

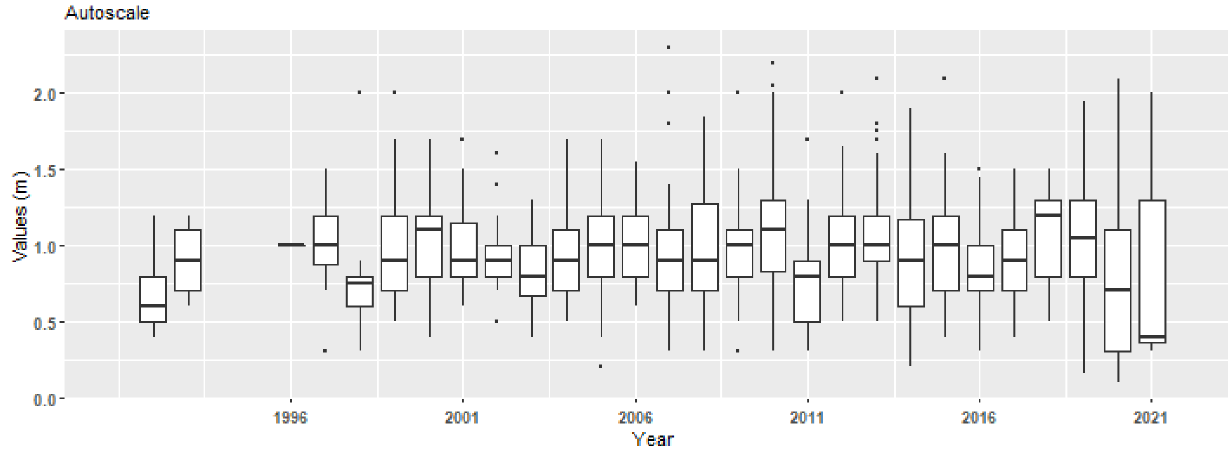
By Year & Month

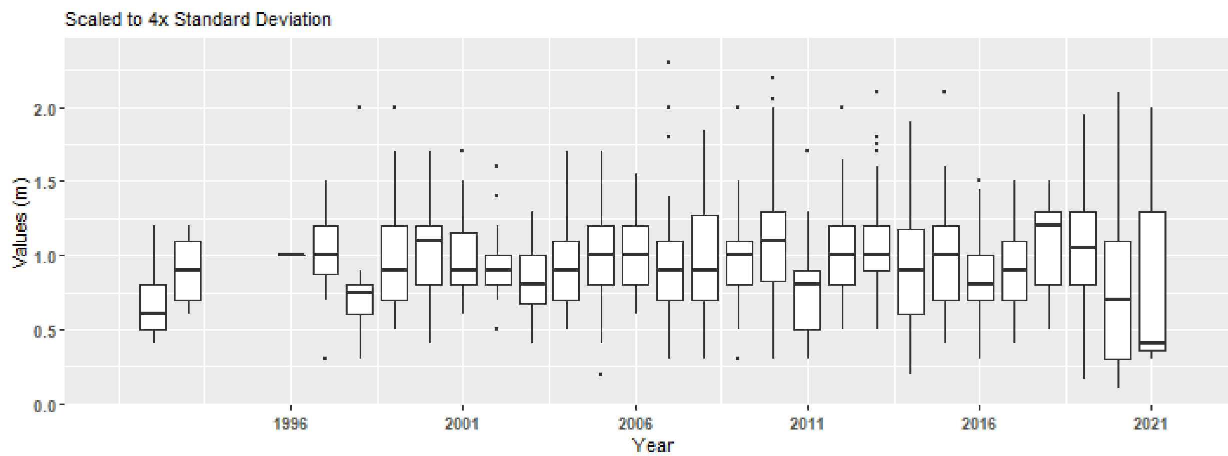


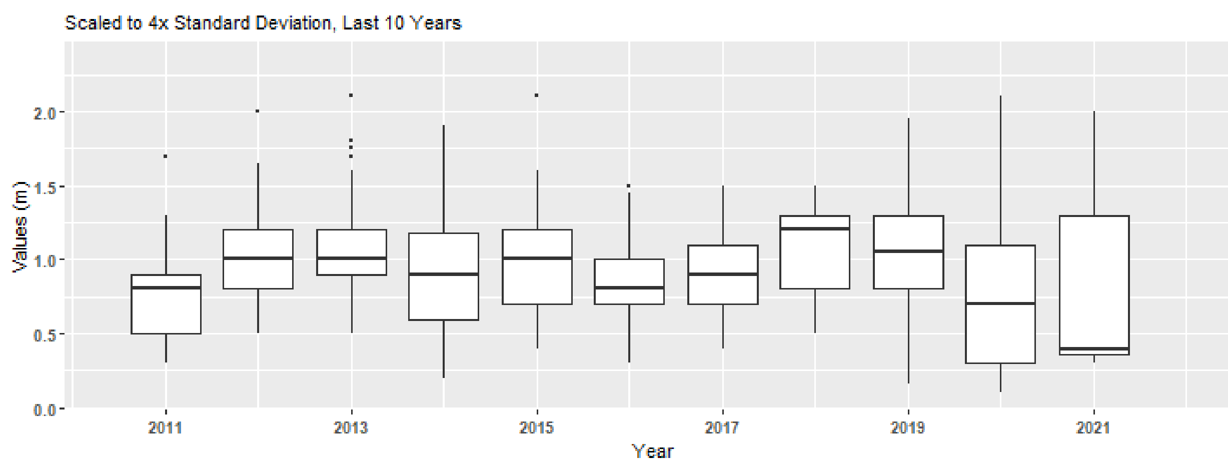
Summary Box Plots for Indian River-Malabar to Vero Beach Aquatic Preserve
By Month



Summary Box Plots for Indian River-Vero Beach to Ft. Pierce Aquatic Preserve
By Year

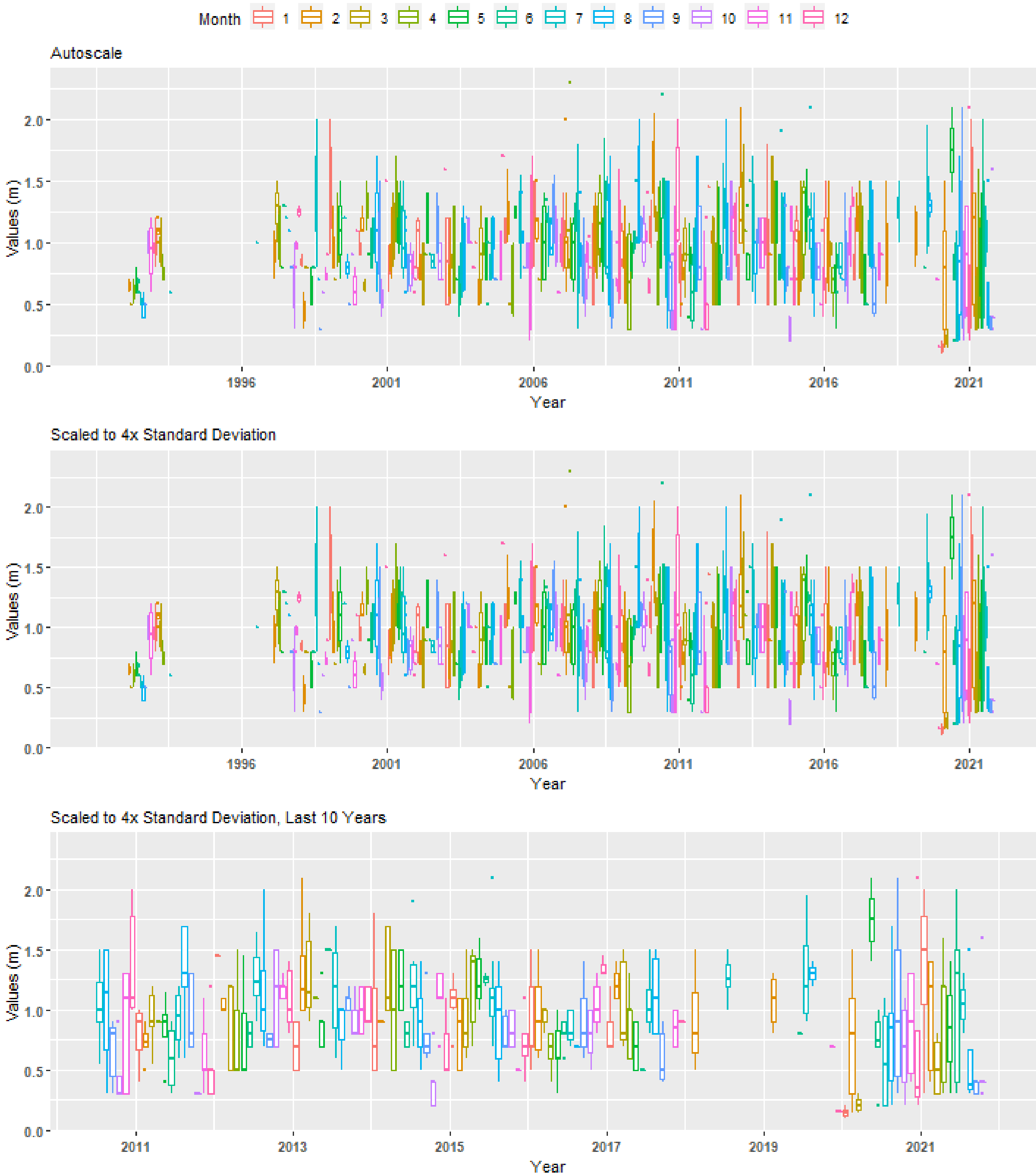




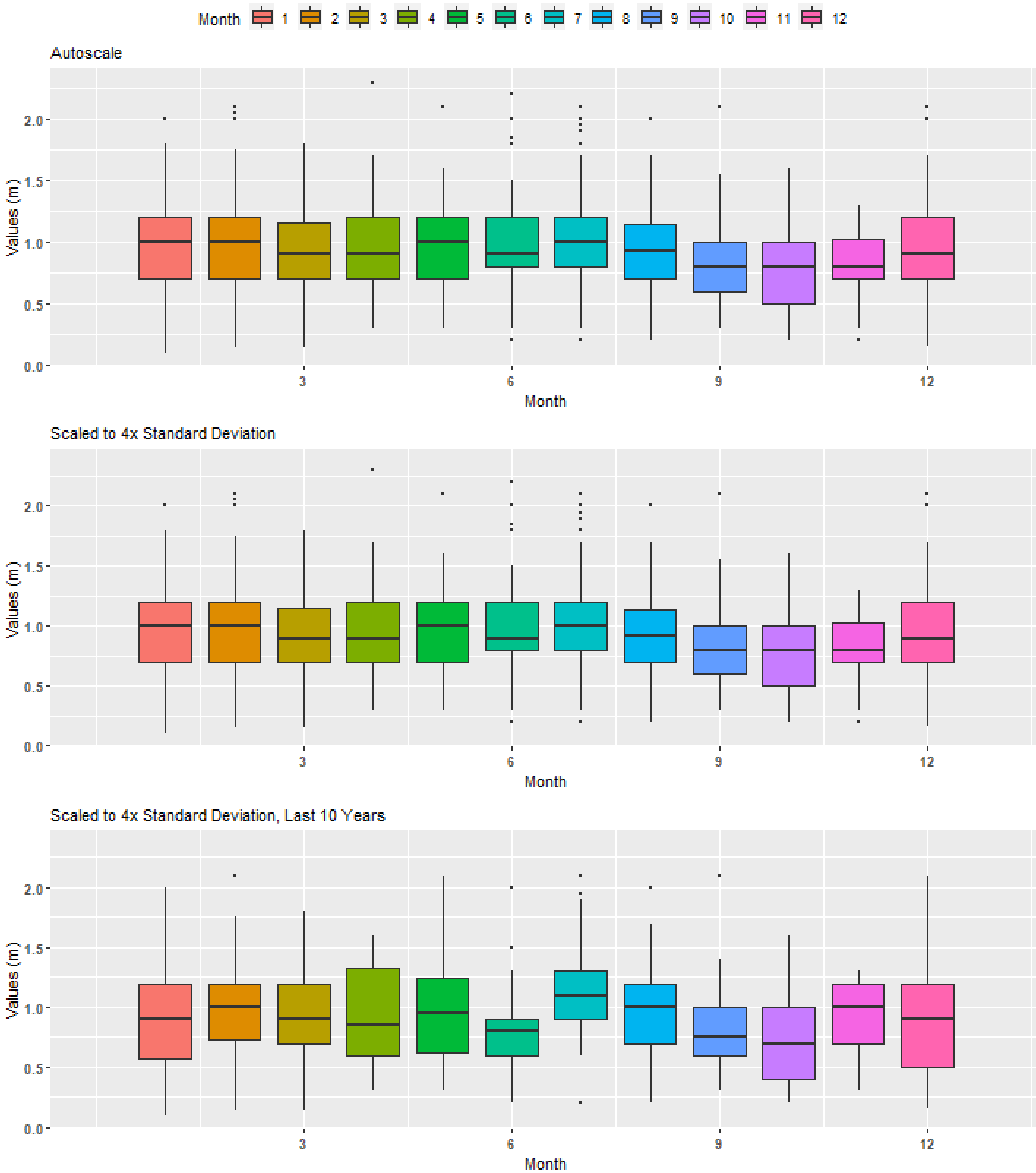


Summary Box Plots for Indian River-Vero Beach to Ft. Pierce Aquatic Preserve

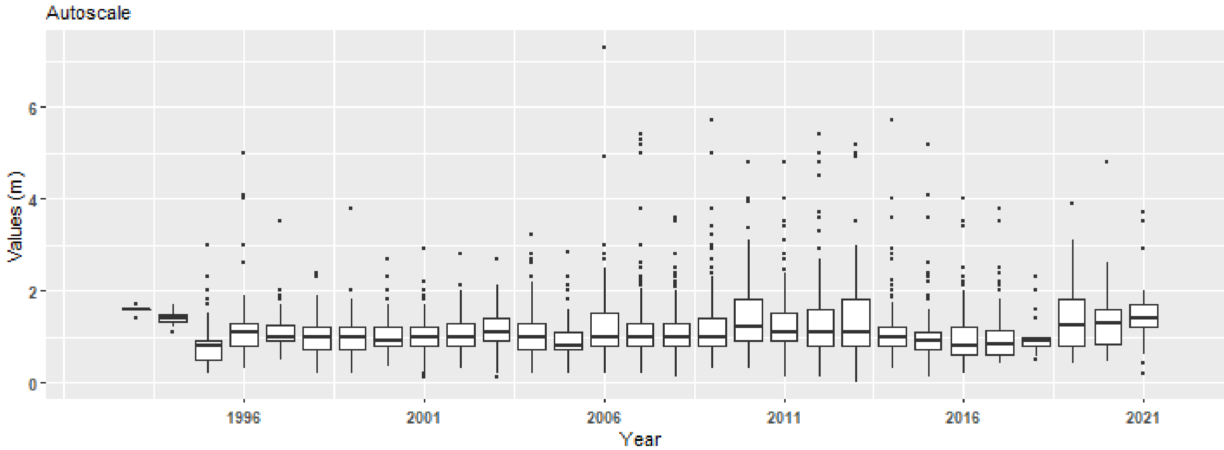
By Year & Month

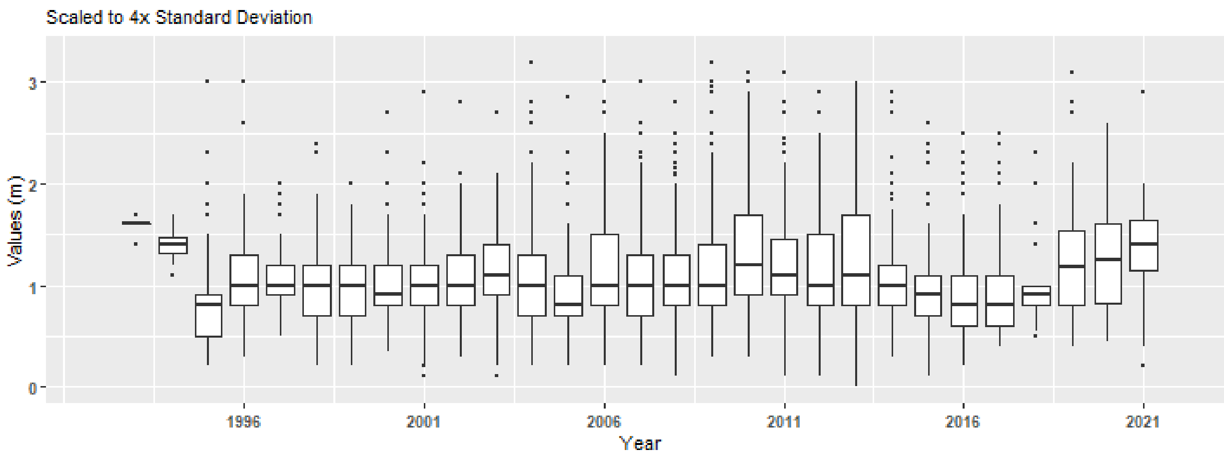


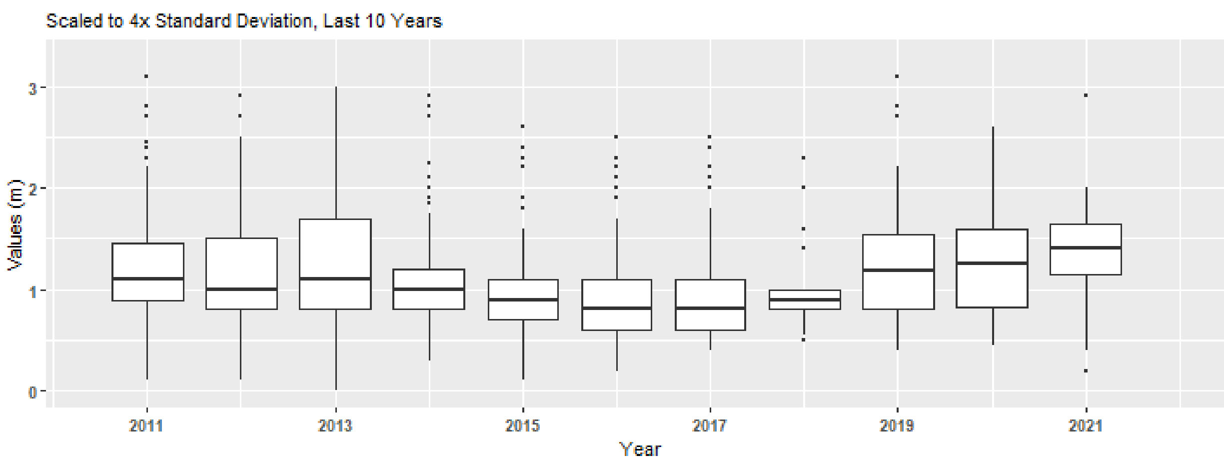
Summary Box Plots for Indian River-Vero Beach to Ft. Pierce Aquatic Preserve
By Month



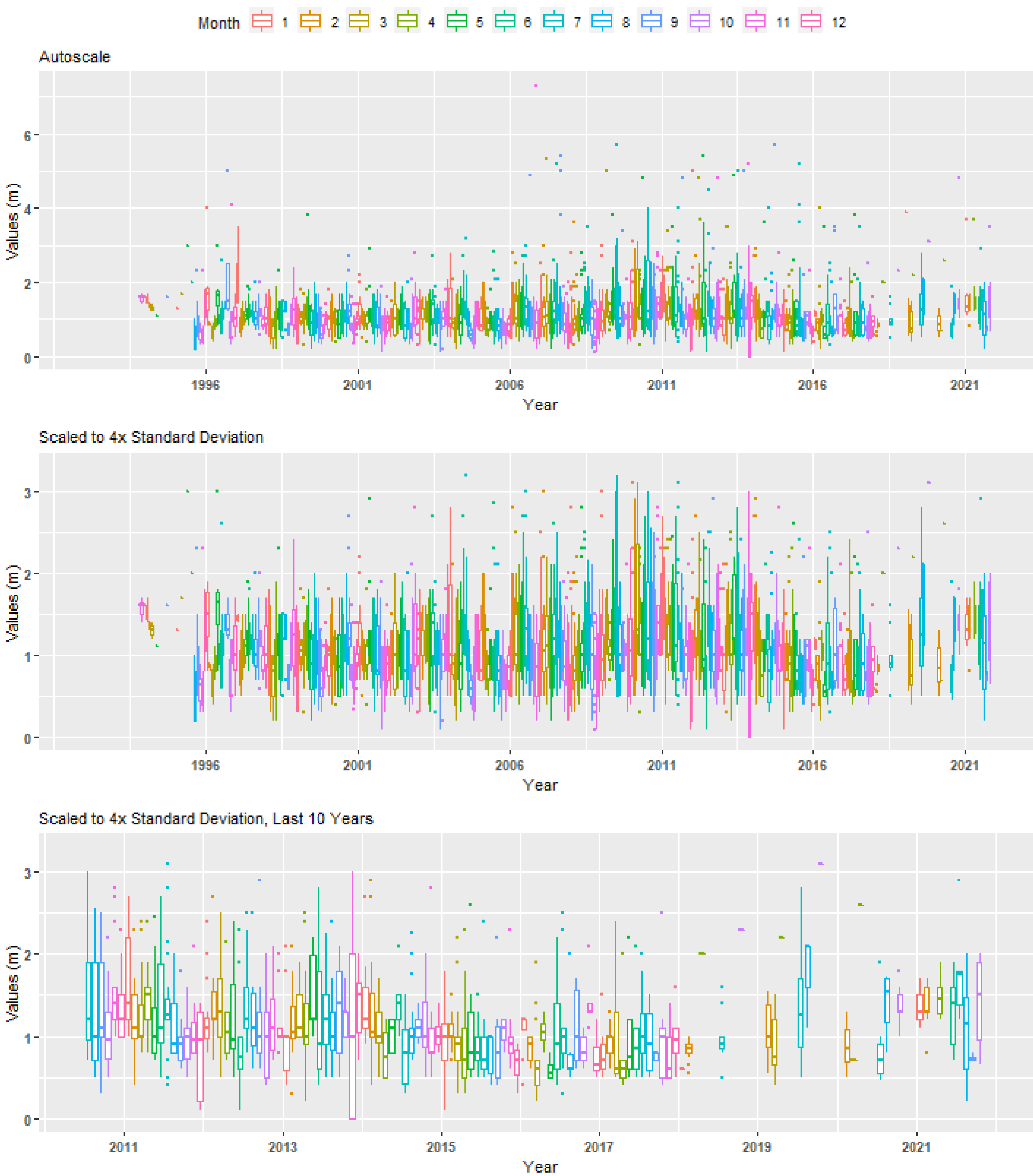
Summary Box Plots for Jensen Beach to Jupiter Inlet Aquatic Preserve
By Year

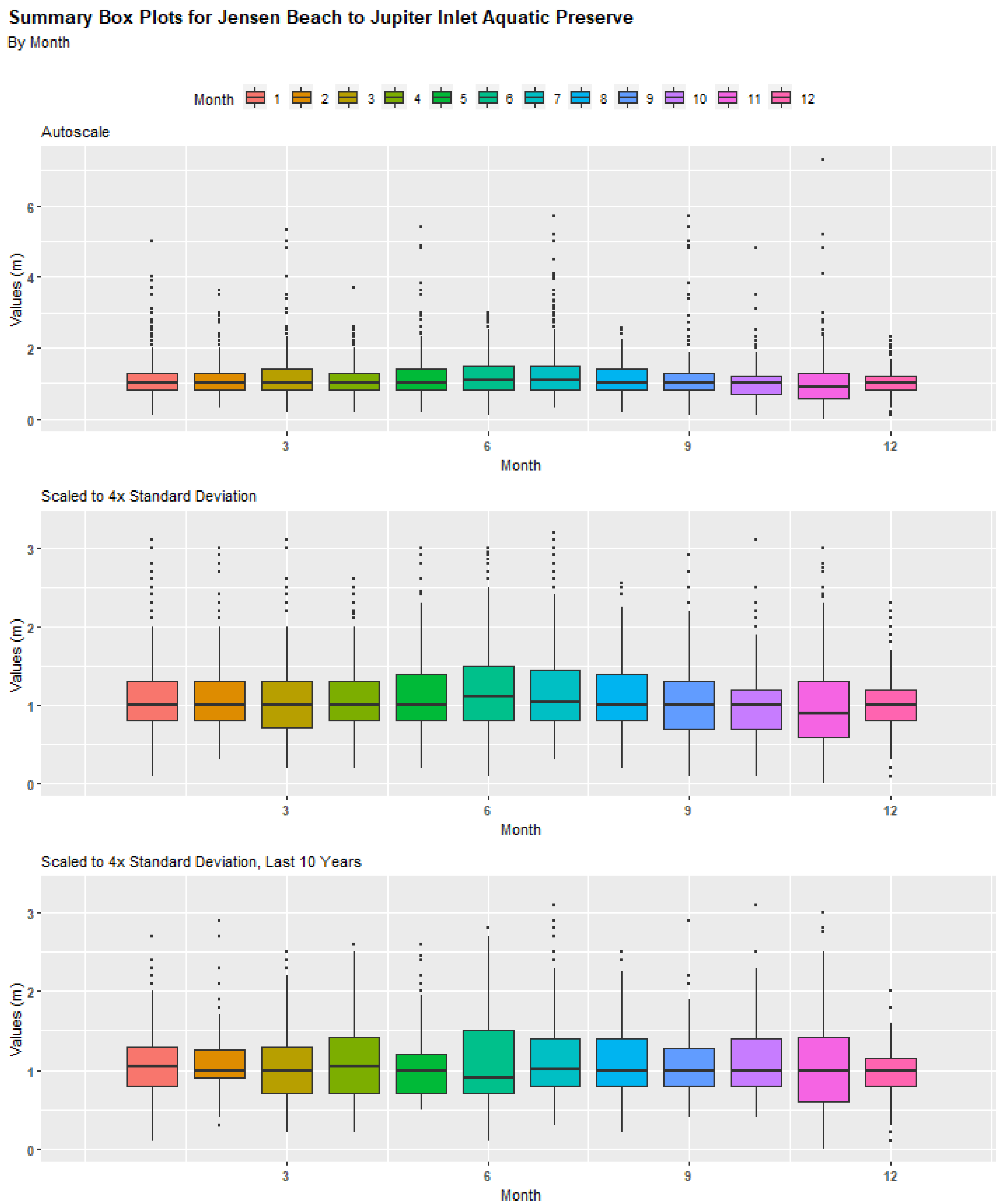






Summary Box Plots for Jensen Beach to Jupiter Inlet Aquatic Preserve
By Year & Month

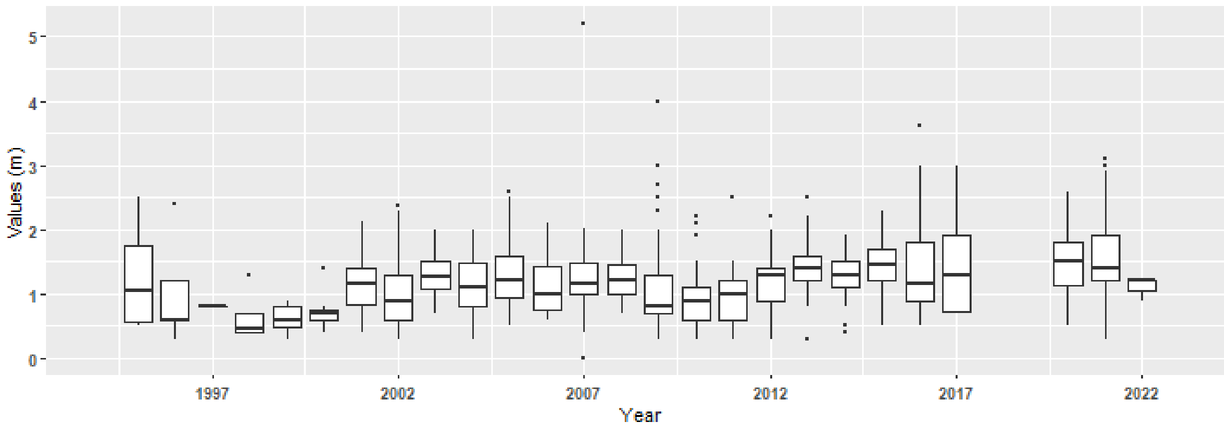




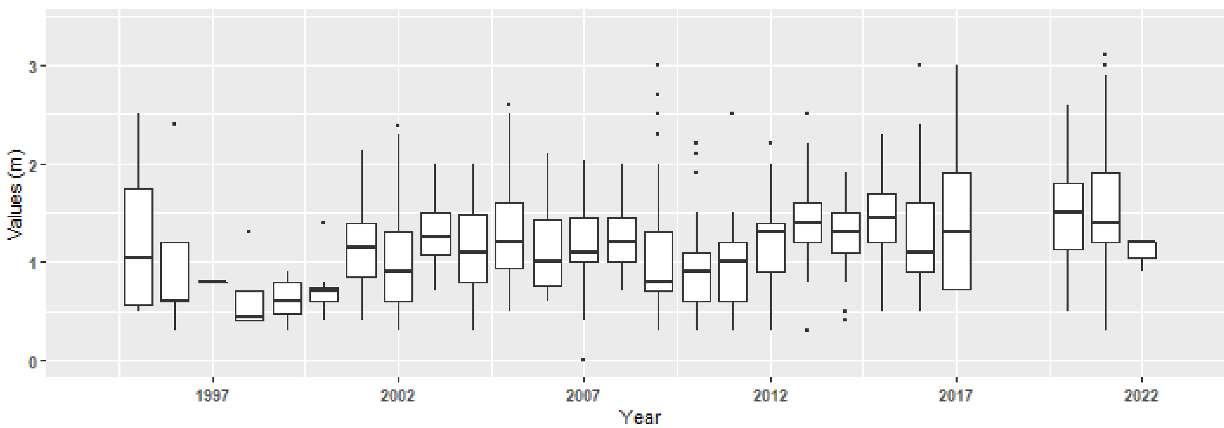
Summary Box Plots for Jensen Beach to Jupiter Inlet Aquatic Preserve
By Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (m)
0 2 4 6
3 6 9 12
Month
Scaled to 4x Standard Deviation
Values (m)
0 1 2 3
3 6 9 12
Month
Scaled to 4x Standard Deviation, Last 10 Years
Values (m)
0 1 2 3
3 6 9 12
Month

Summary Box Plots for Lemon Bay Aquatic Preserve
By Year

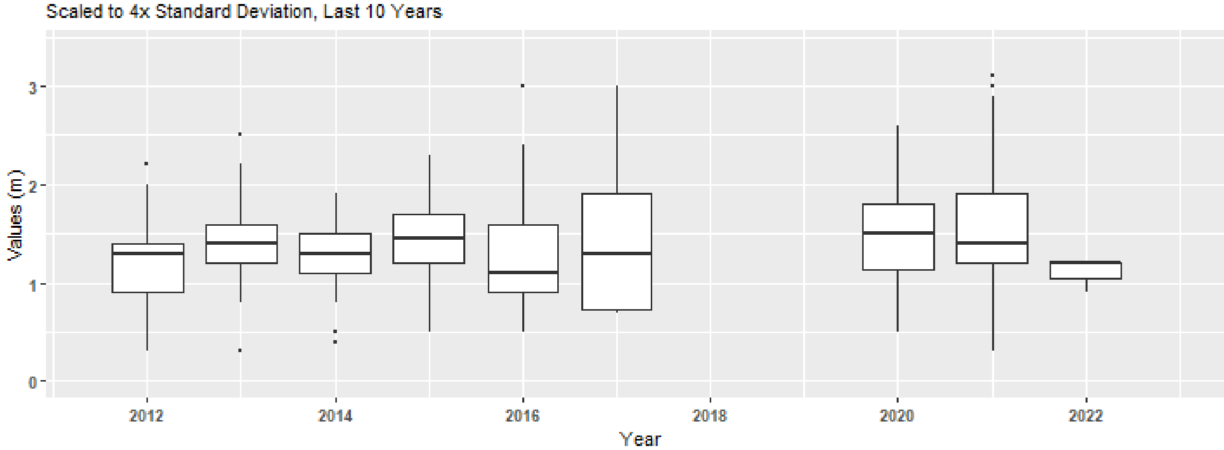
Autoscale



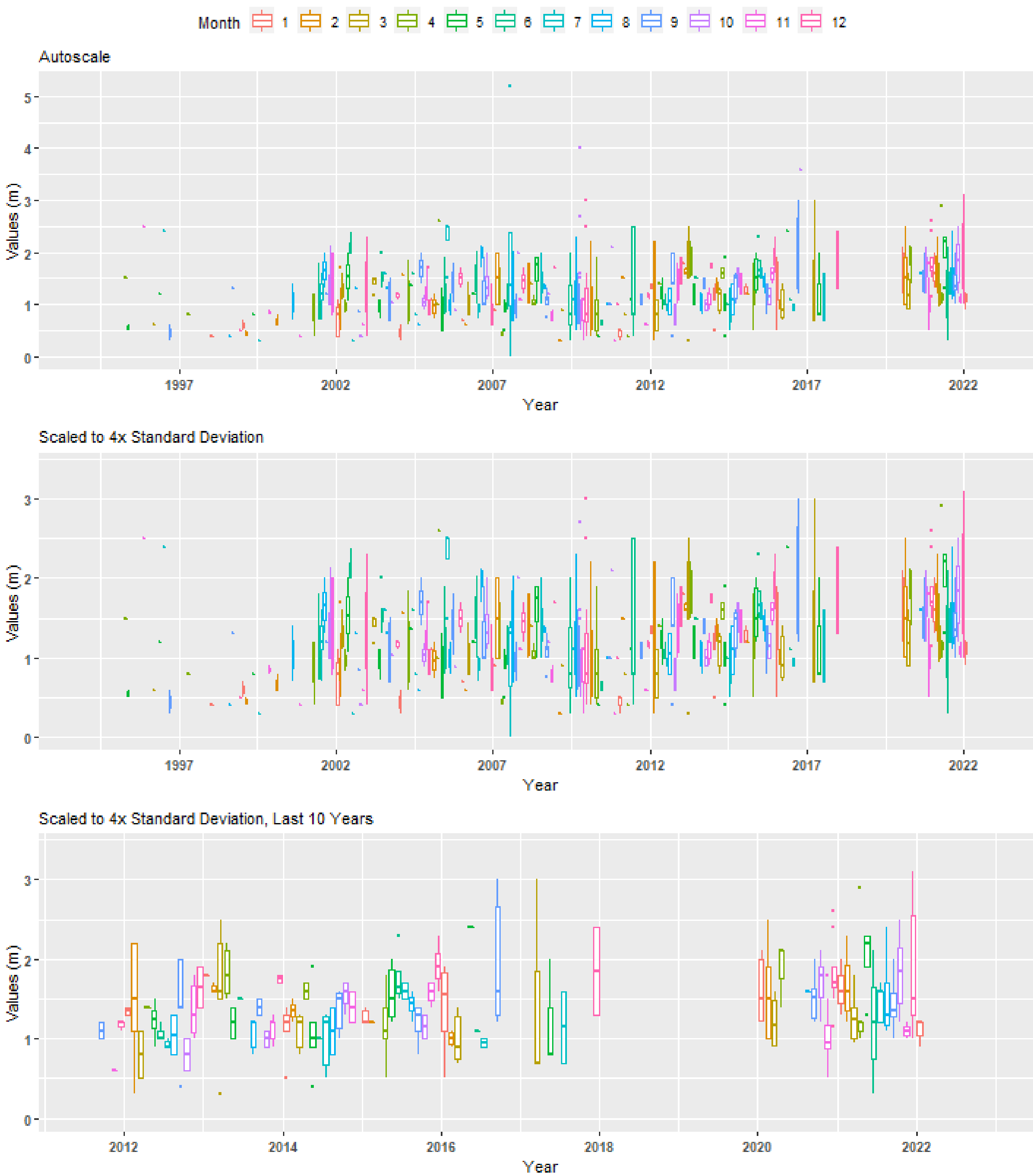
Scaled to 4x Standard Deviation



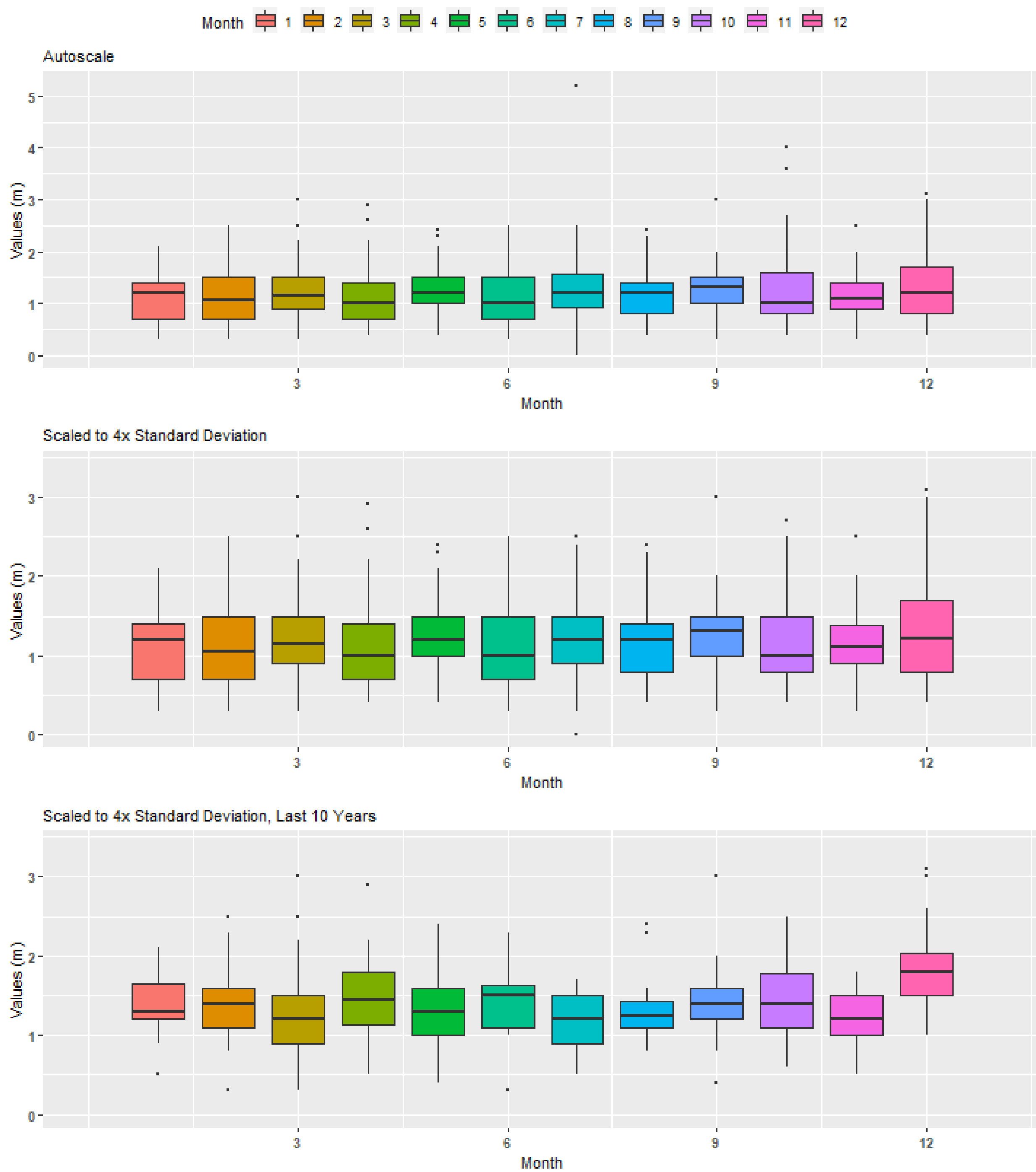
Scaled to 4x Standard Deviation, Last 10 Years



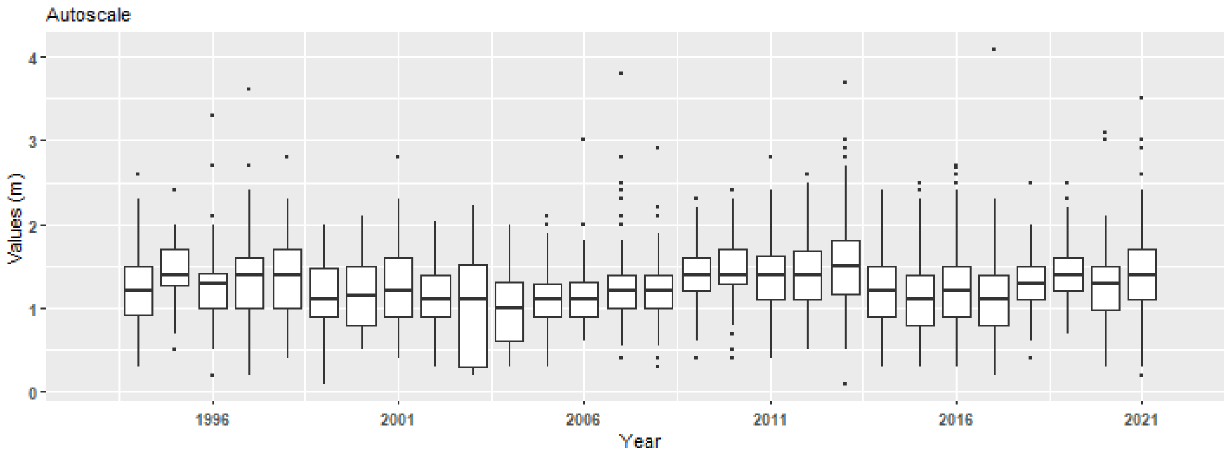
Summary Box Plots for Lemon Bay Aquatic Preserve
By Year & Month

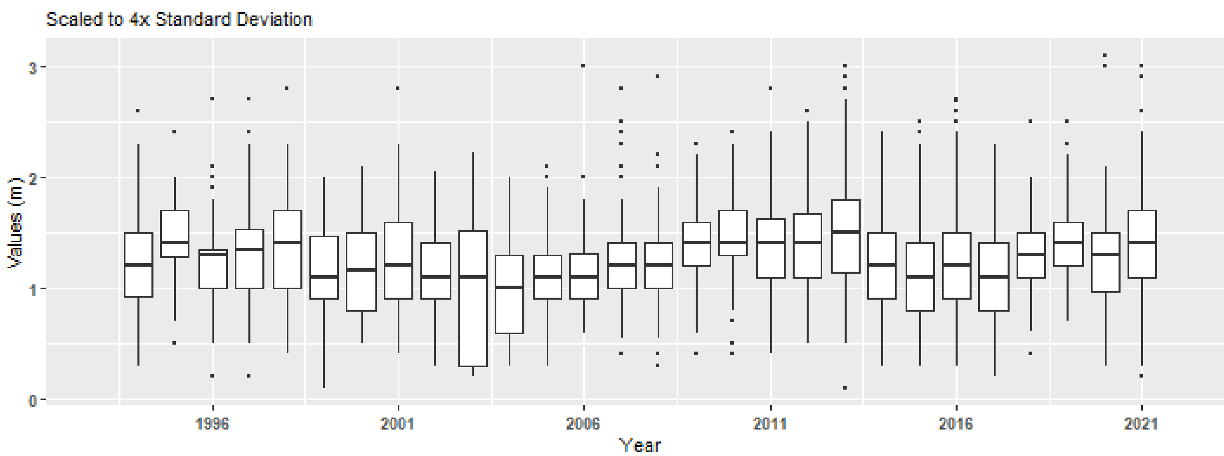


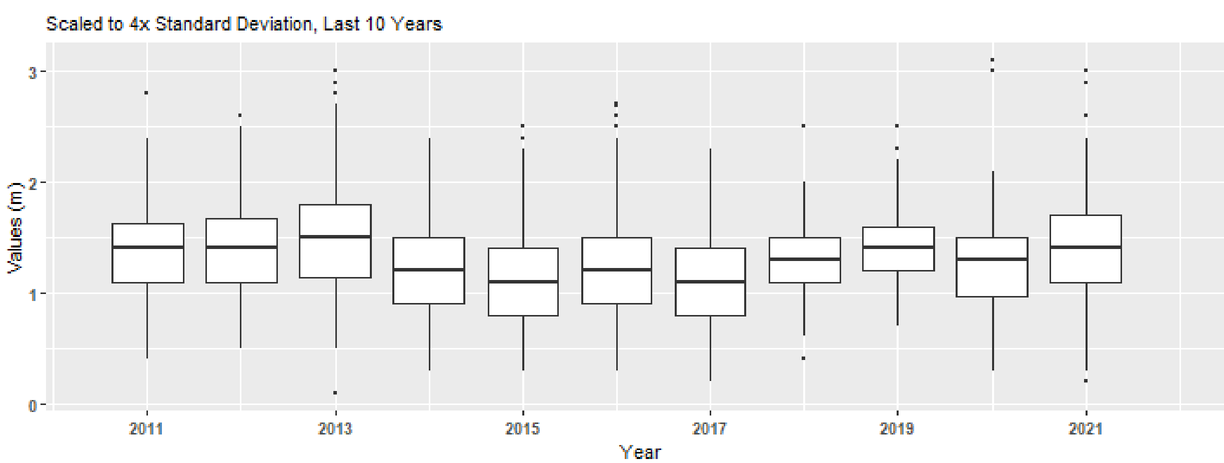
Summary Box Plots for Lemon Bay Aquatic Preserve
By Month



Summary Box Plots for Loxahatchee River-Lake Worth Creek Aquatic Preserve
By Year

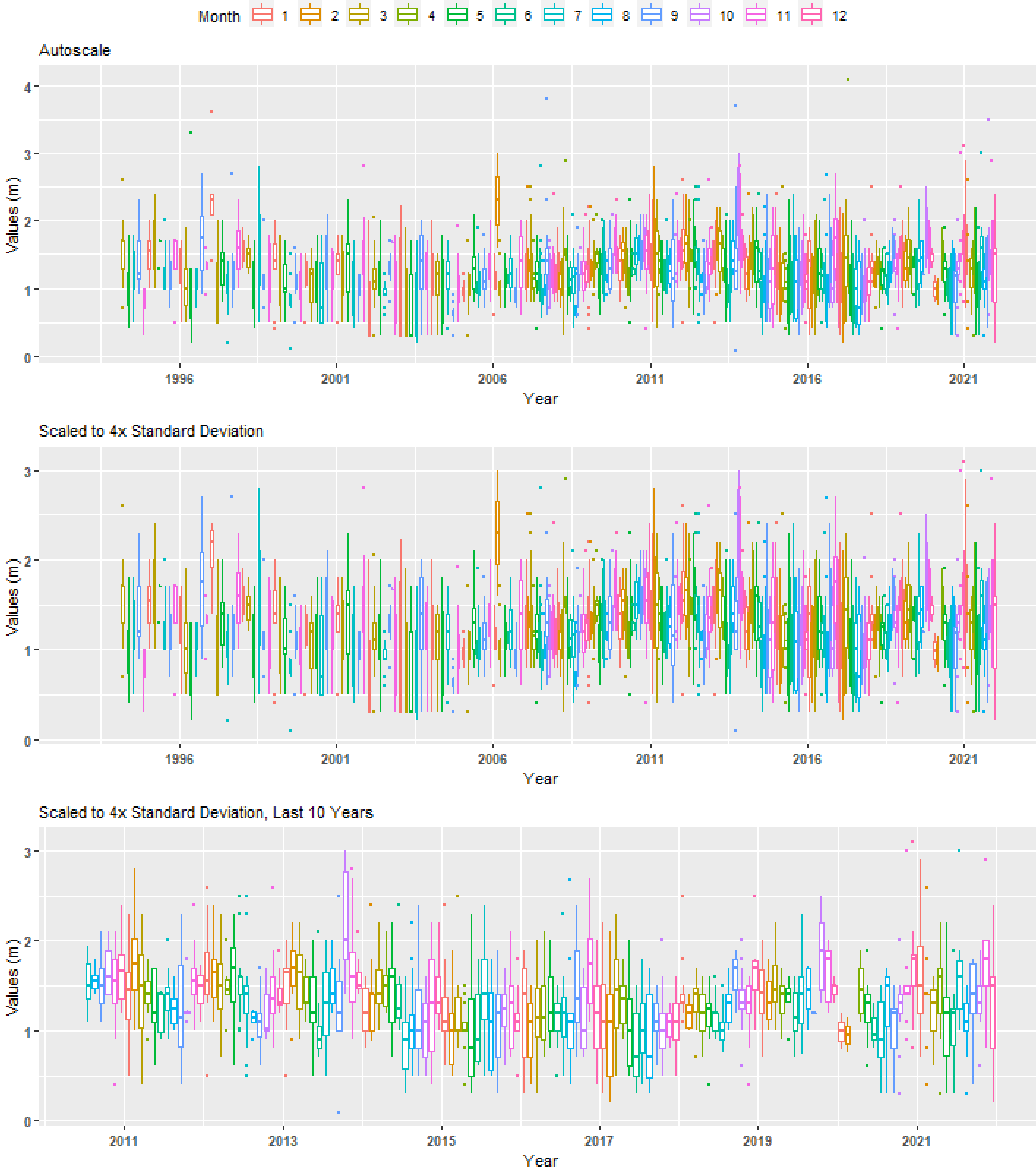




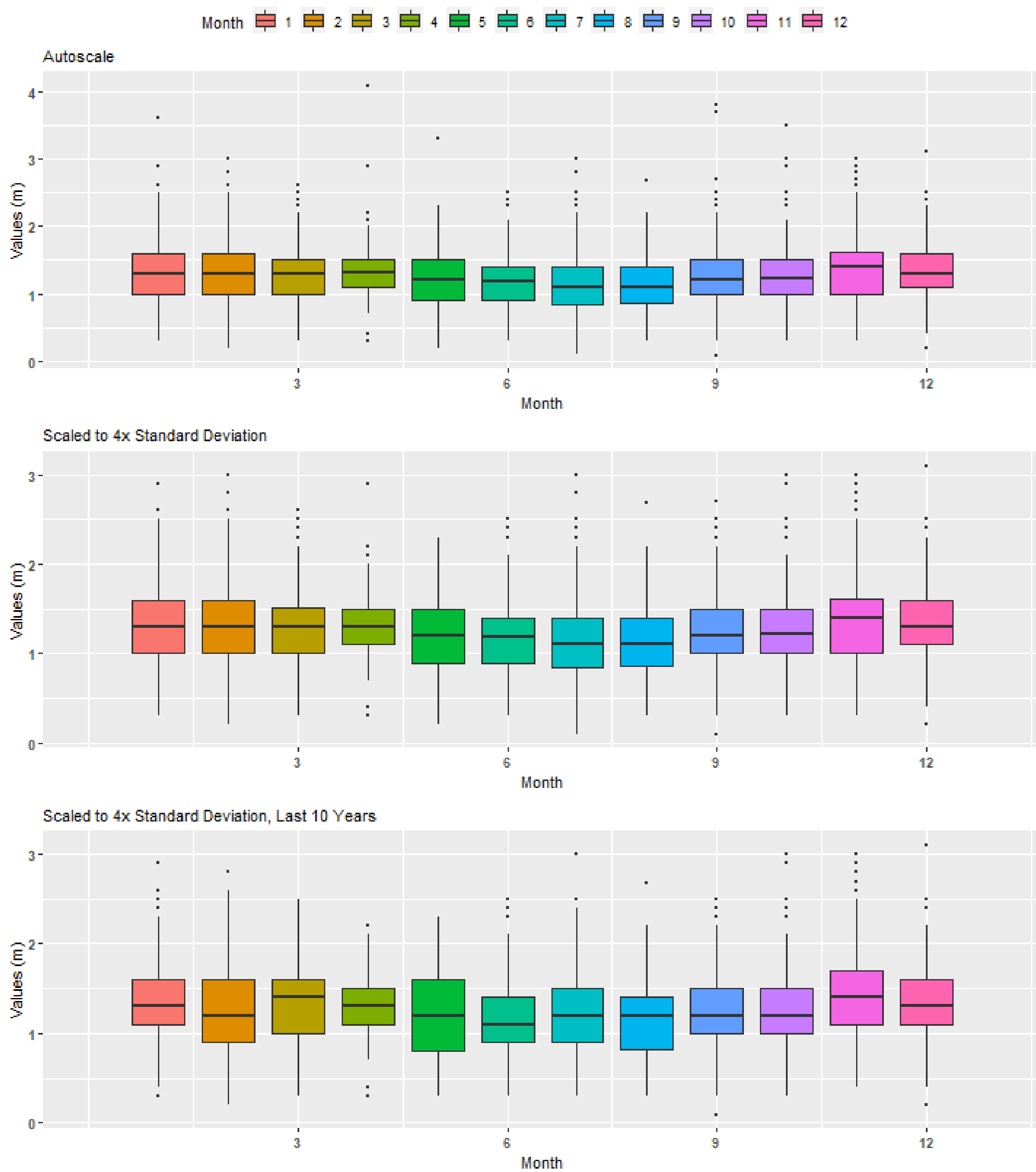


Summary Box Plots for Loxahatchee River-Lake Worth Creek Aquatic Preserve

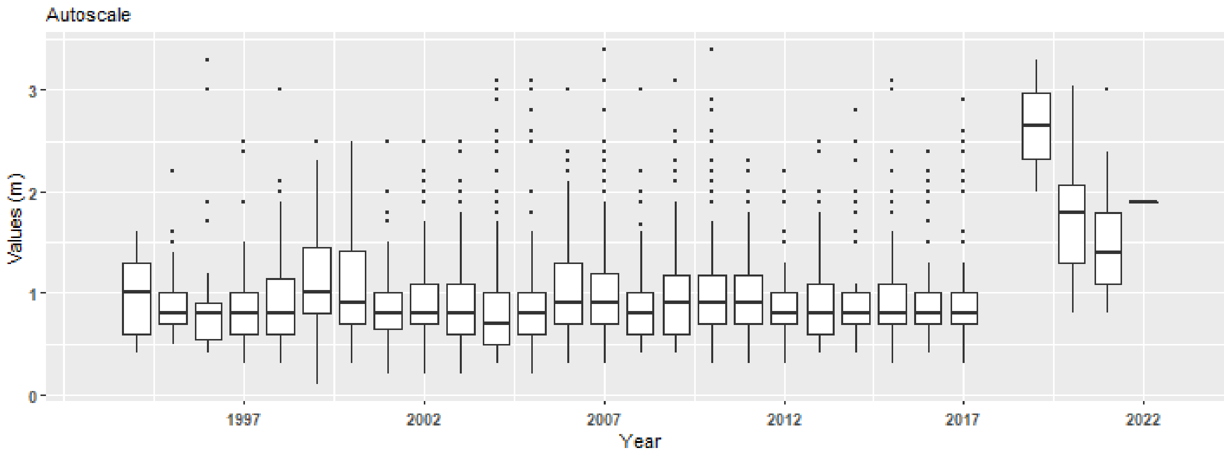
By Year & Month

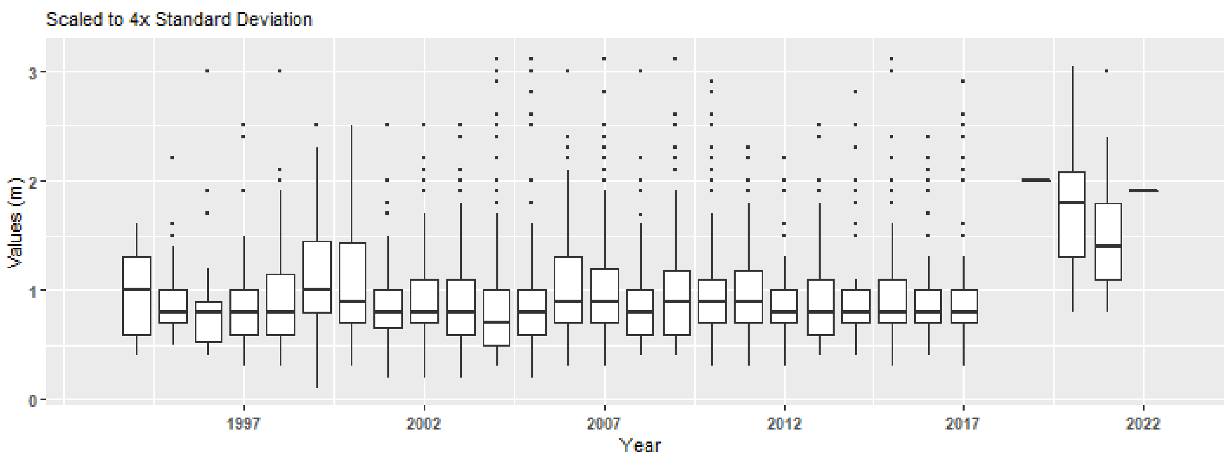


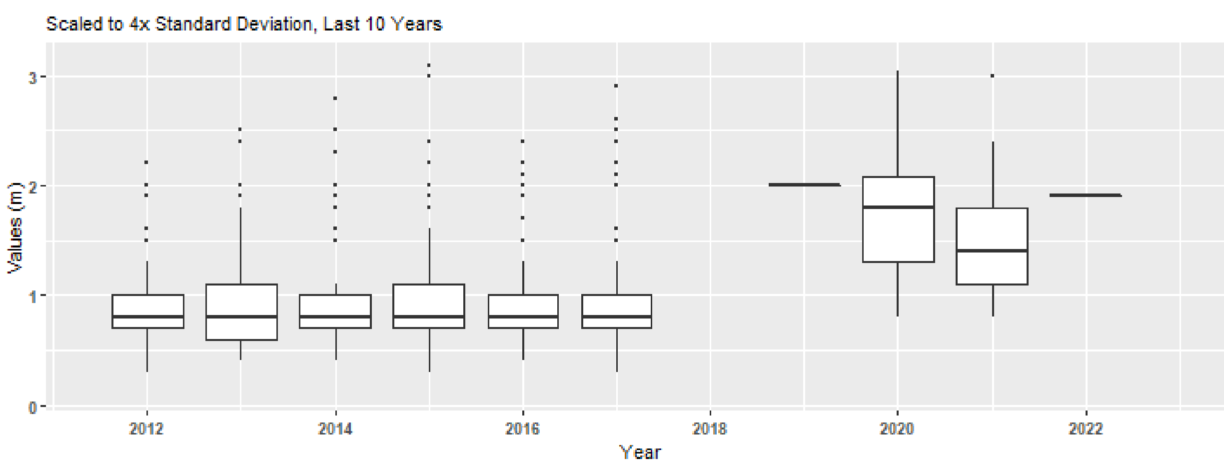
Summary Box Plots for Loxahatchee River-Lake Worth Creek Aquatic Preserve
By Month



Summary Box Plots for Matlacha Pass Aquatic Preserve
By Year

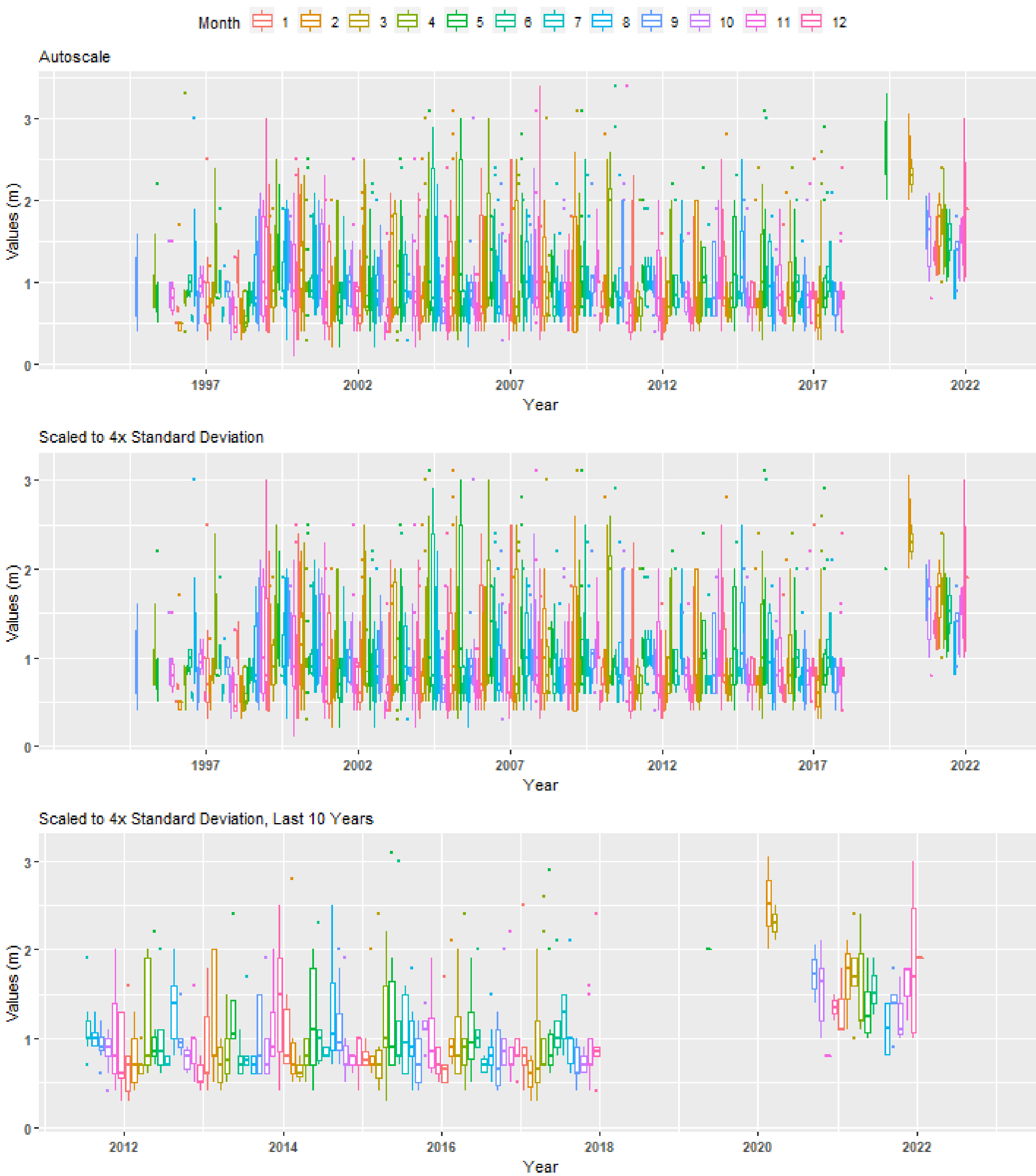




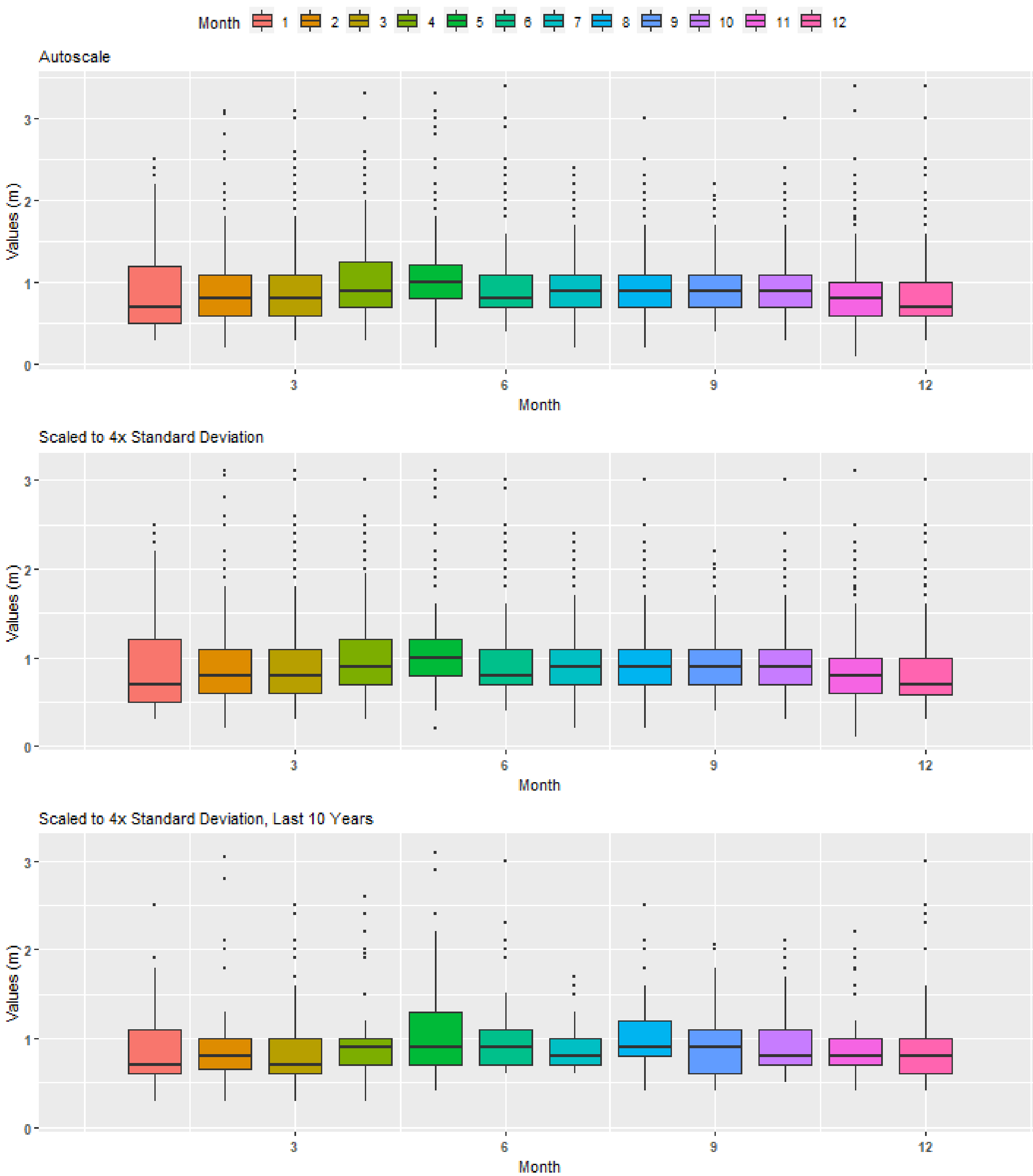


Summary Box Plots for Matlacha Pass Aquatic Preserve

By Year & Month

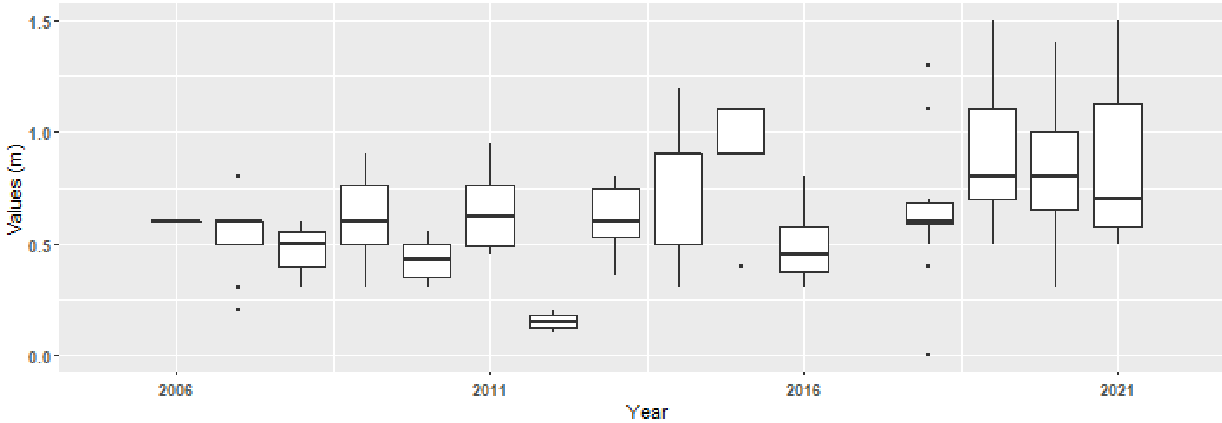


Summary Box Plots for Matlacha Pass Aquatic Preserve
By Month

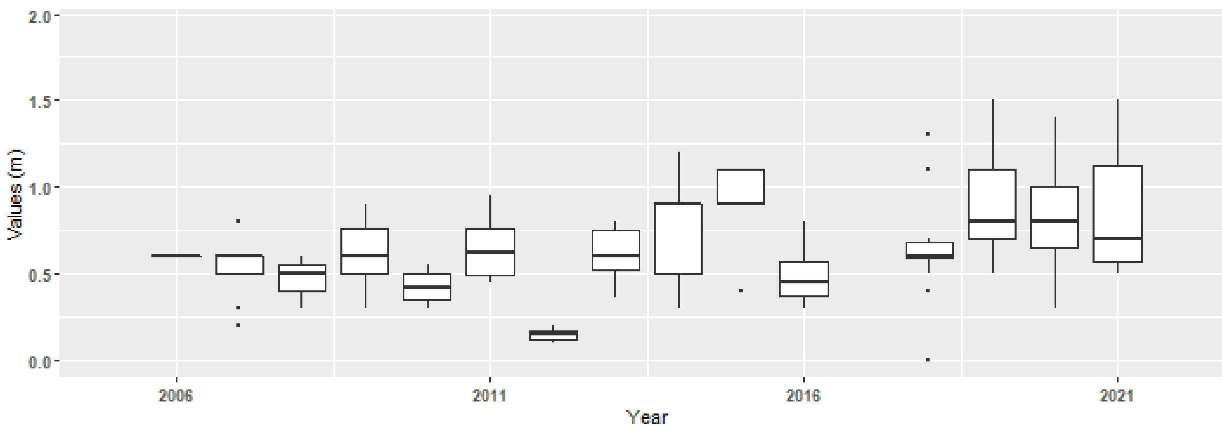


Summary Box Plots for Mosquito Lagoon Aquatic Preserve
By Year

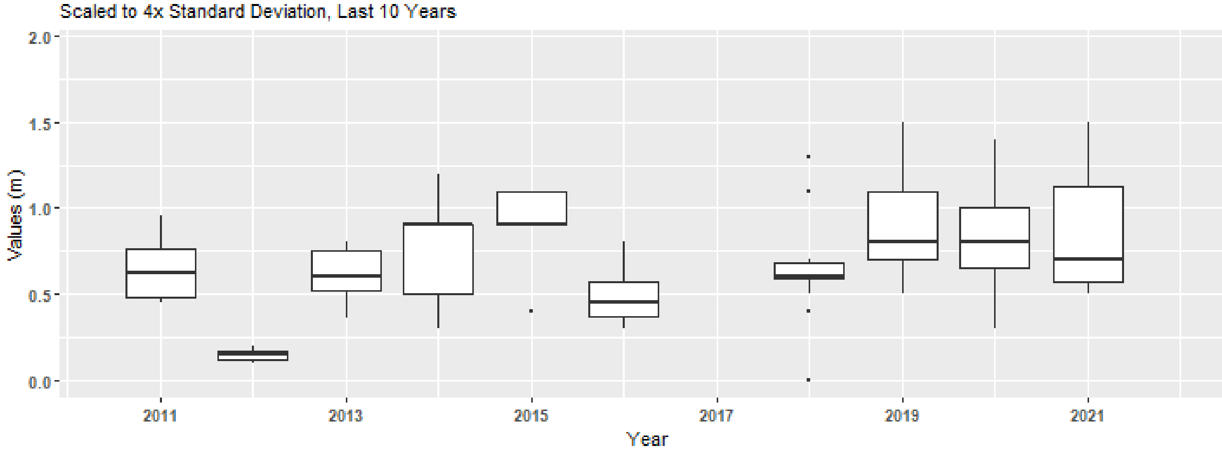
Autoscale



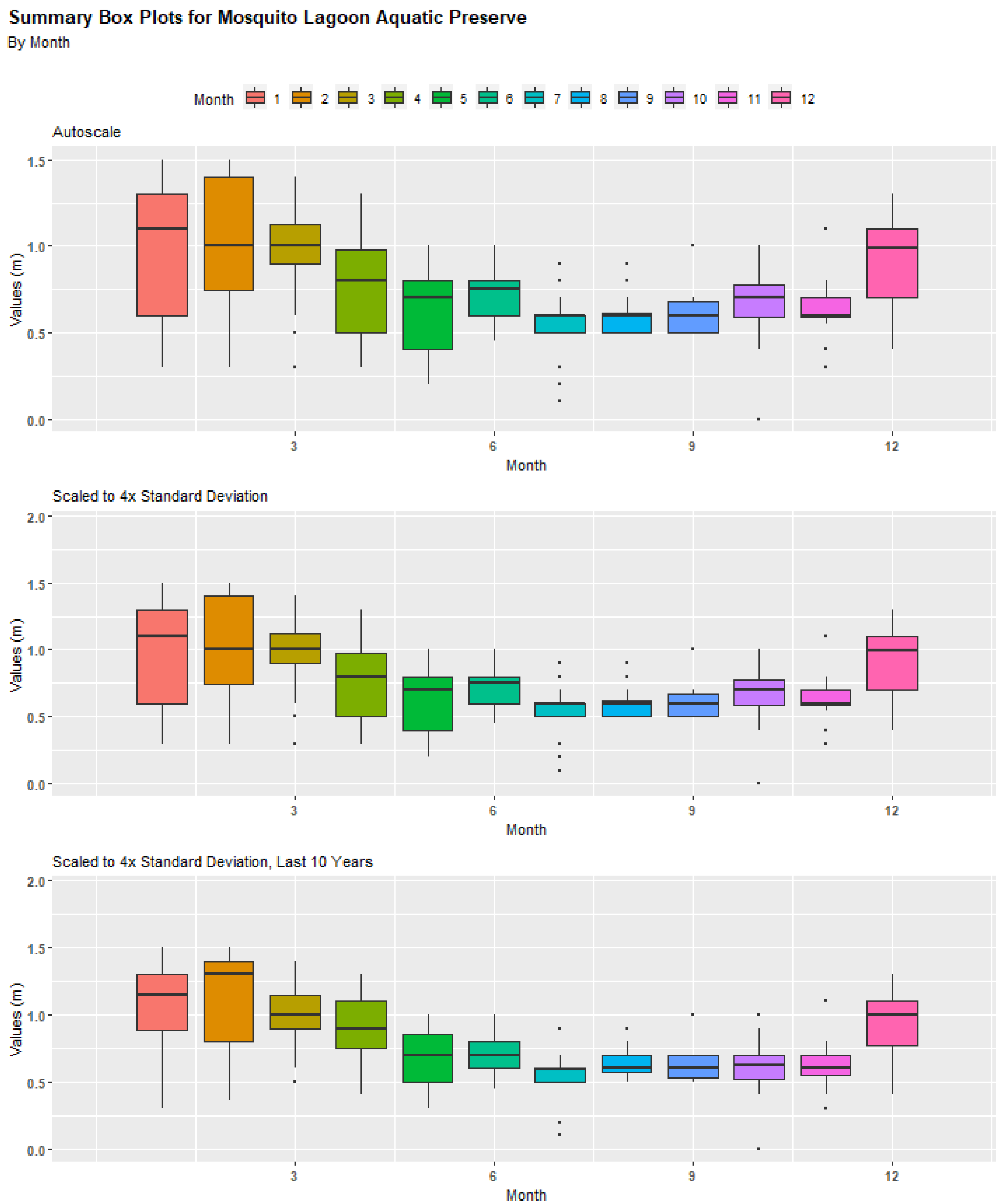
Scaled to 4x Standard Deviation



Scaled to 4x Standard Deviation, Last 10 Years

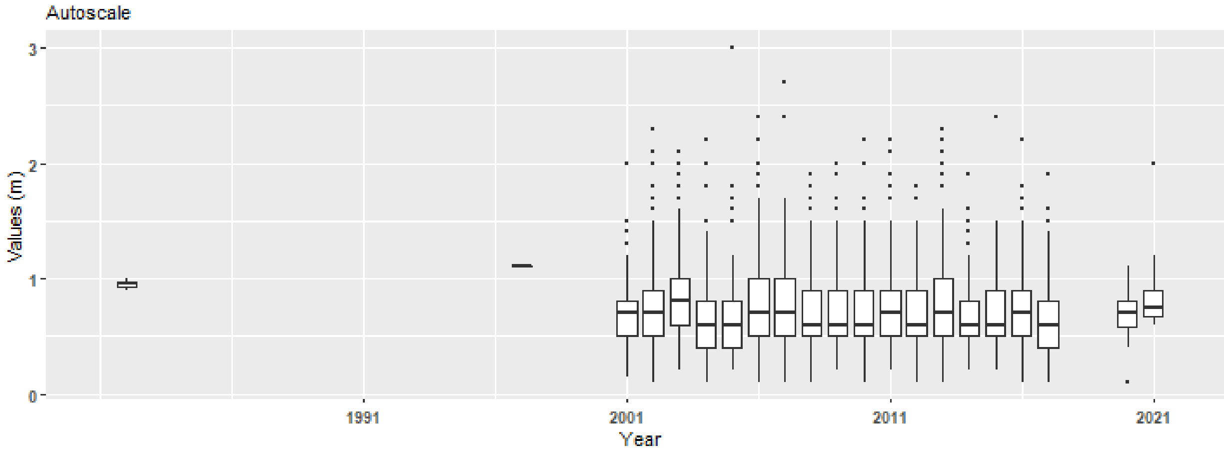


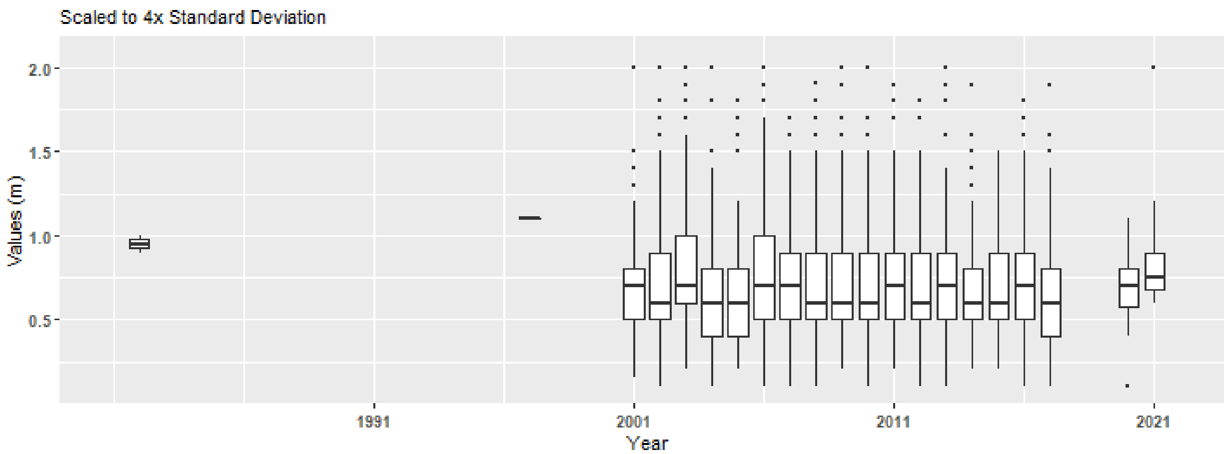

Summary Box Plots for Mosquito Lagoon Aquatic Preserve
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (m)
1.5
1.0
0.5
0.0
2006
2011
2016
2021
Year
Scaled to 4x Standard Deviation
Values (m)
2.0
1.5
1.0
0.5
0.0
2006
2011
2016
2021
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (m)
2.0
1.5
1.0
0.5
0.0
2011
2013
2015
2017
2019
2021
Year

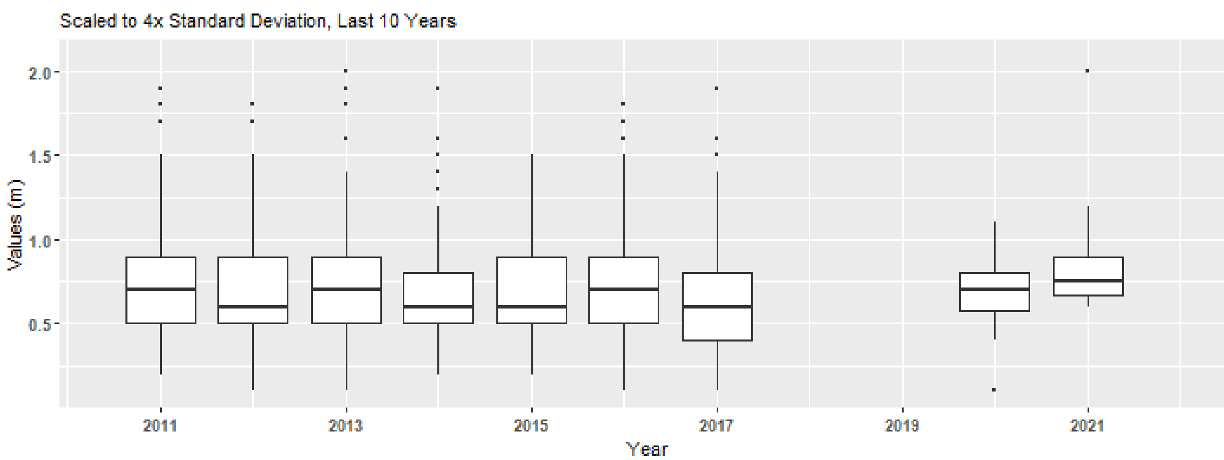


Summary Box Plots for Mosquito Lagoon Aquatic Preserve
By Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (m)
1.5
1.0
0.5
0.0
3
6
9
12
Month
Scaled to 4x Standard Deviation
Values (m)
2.0
1.5
1.0
0.5
0.0
3
6
9
12
Month
Scaled to 4x Standard Deviation, Last 10 Years
Values (m)
2.0
1.5
1.0
0.5
0.0
3
6
9
12
Month

Summary Box Plots for Nassau River-St. Johns River Marshes Aquatic Preserve
By Year

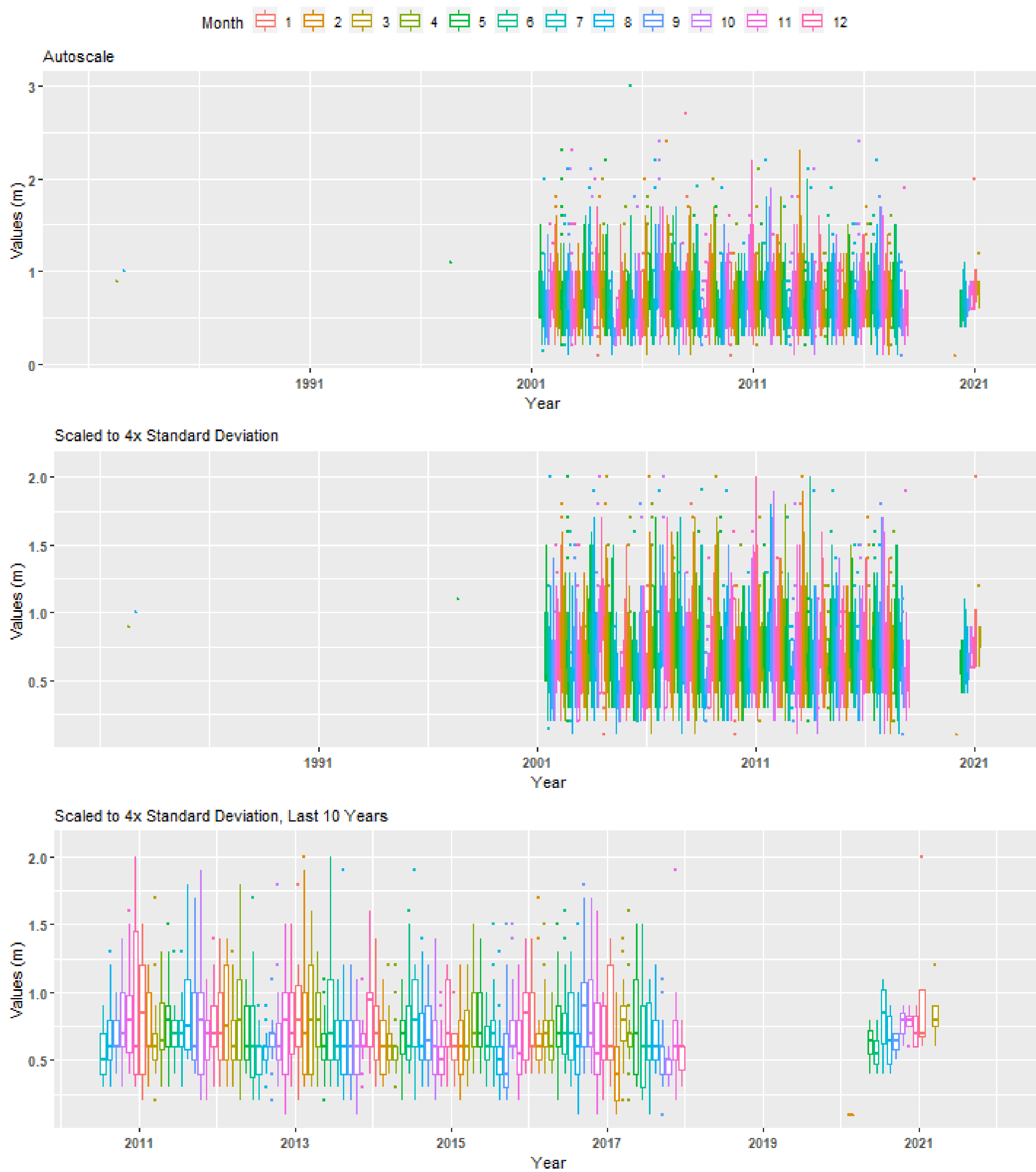




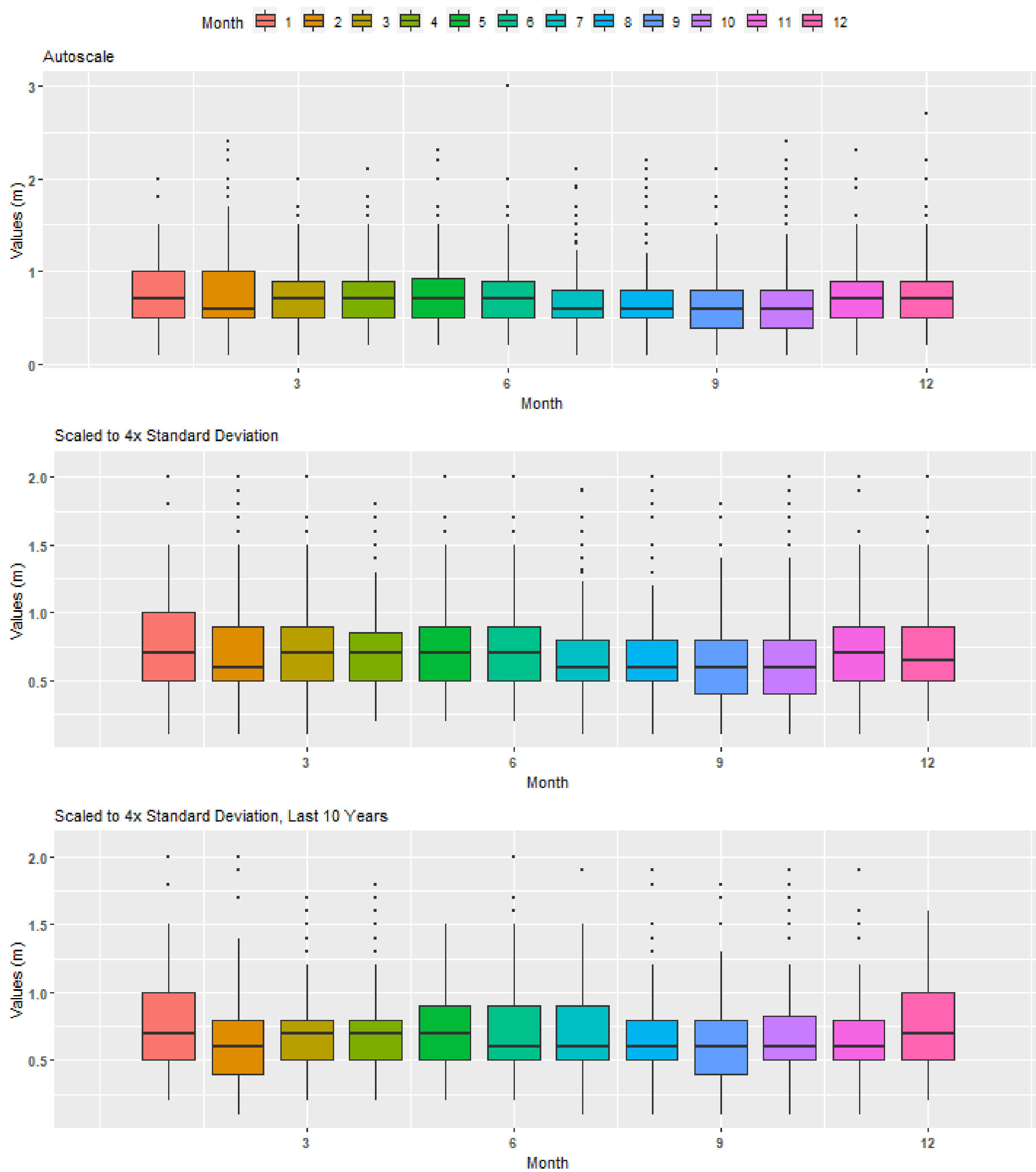


Summary Box Plots for Nassau River-St. Johns River Marshes Aquatic Preserve

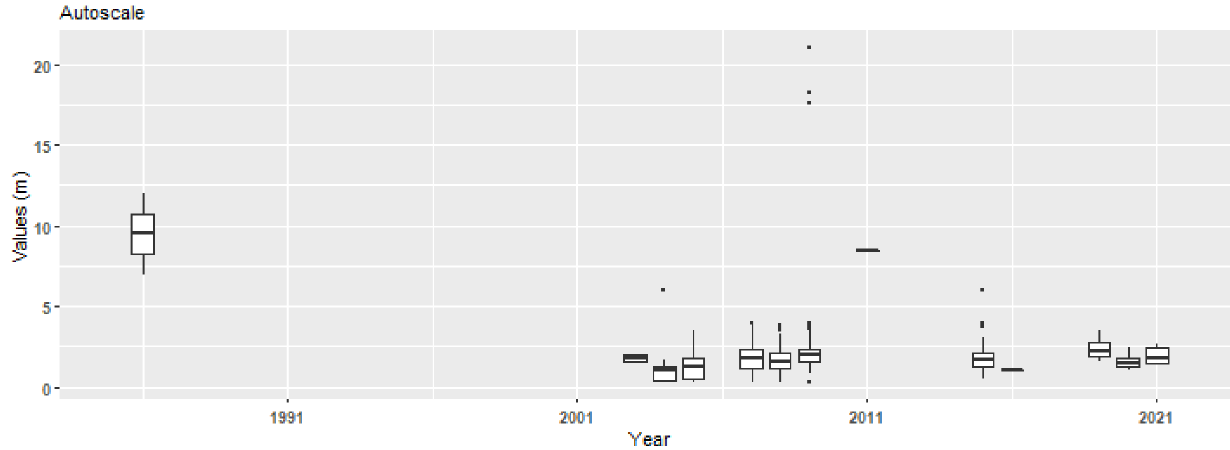
By Year & Month

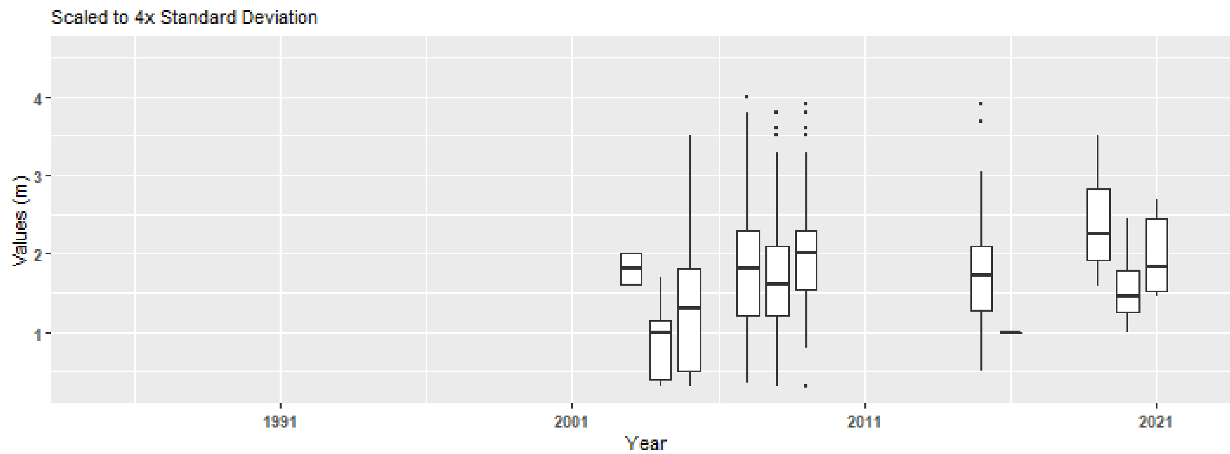


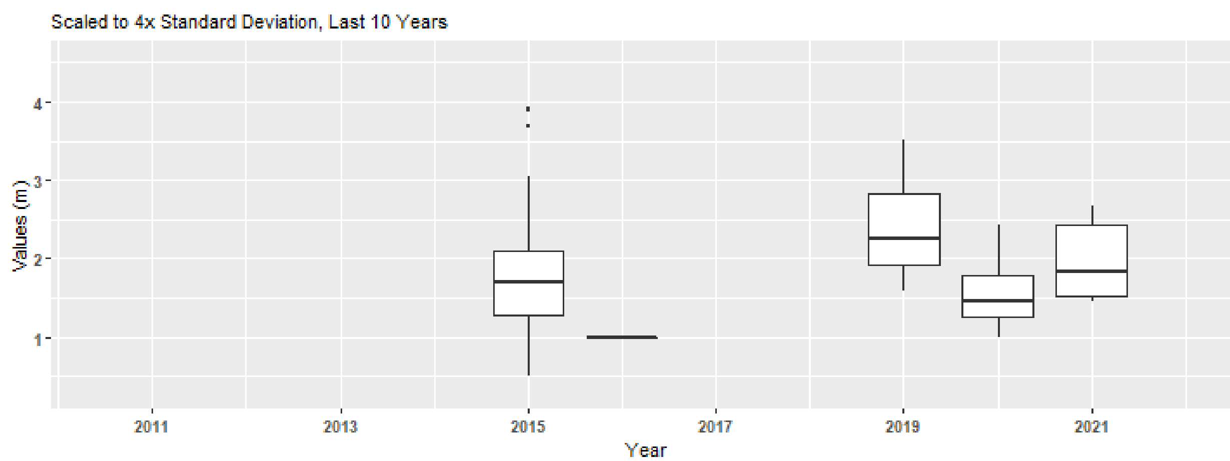
Summary Box Plots for Nassau River-St. Johns River Marshes Aquatic Preserve
By Month



Summary Box Plots for Nature Coast Aquatic Preserve
By Year

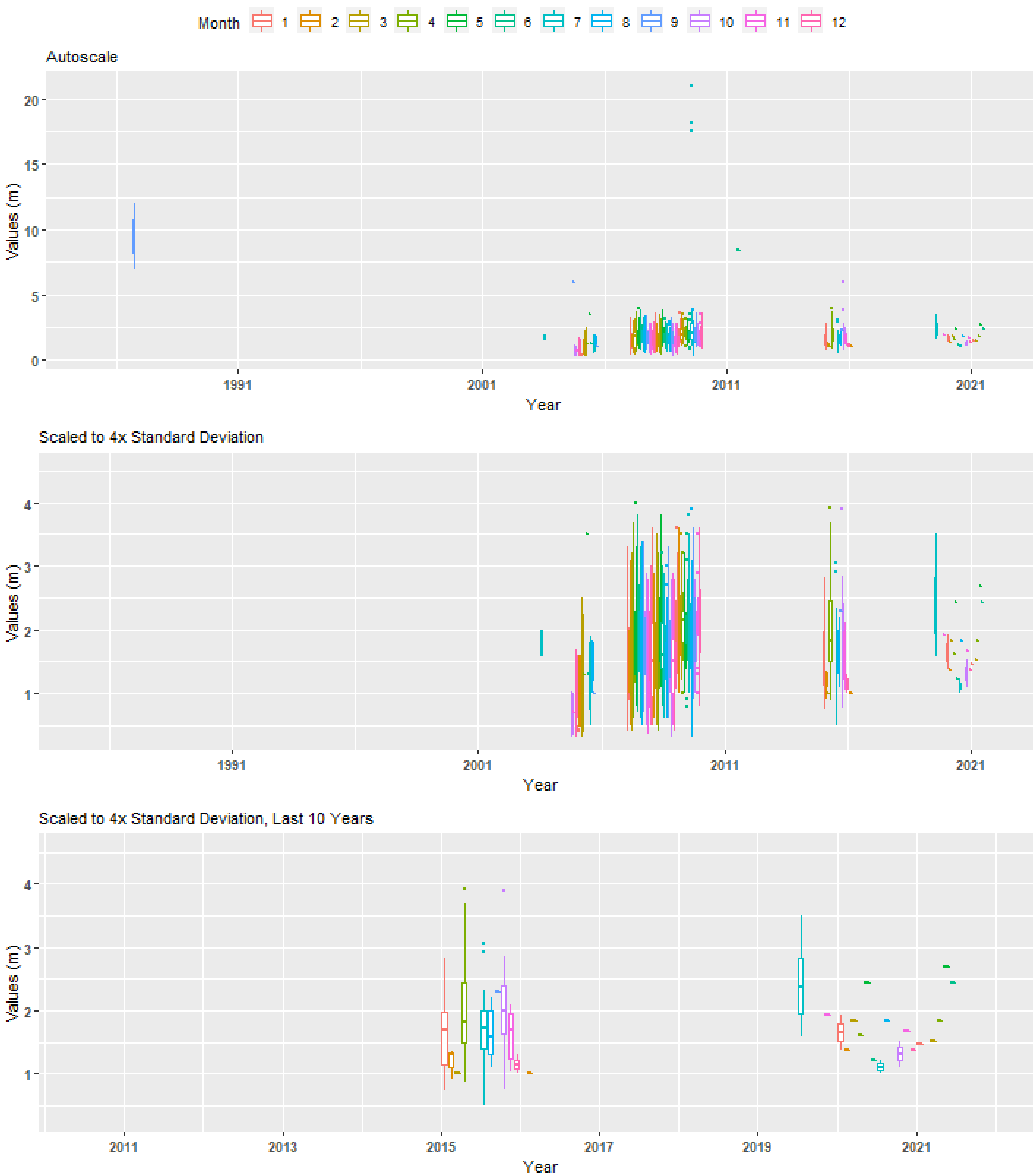




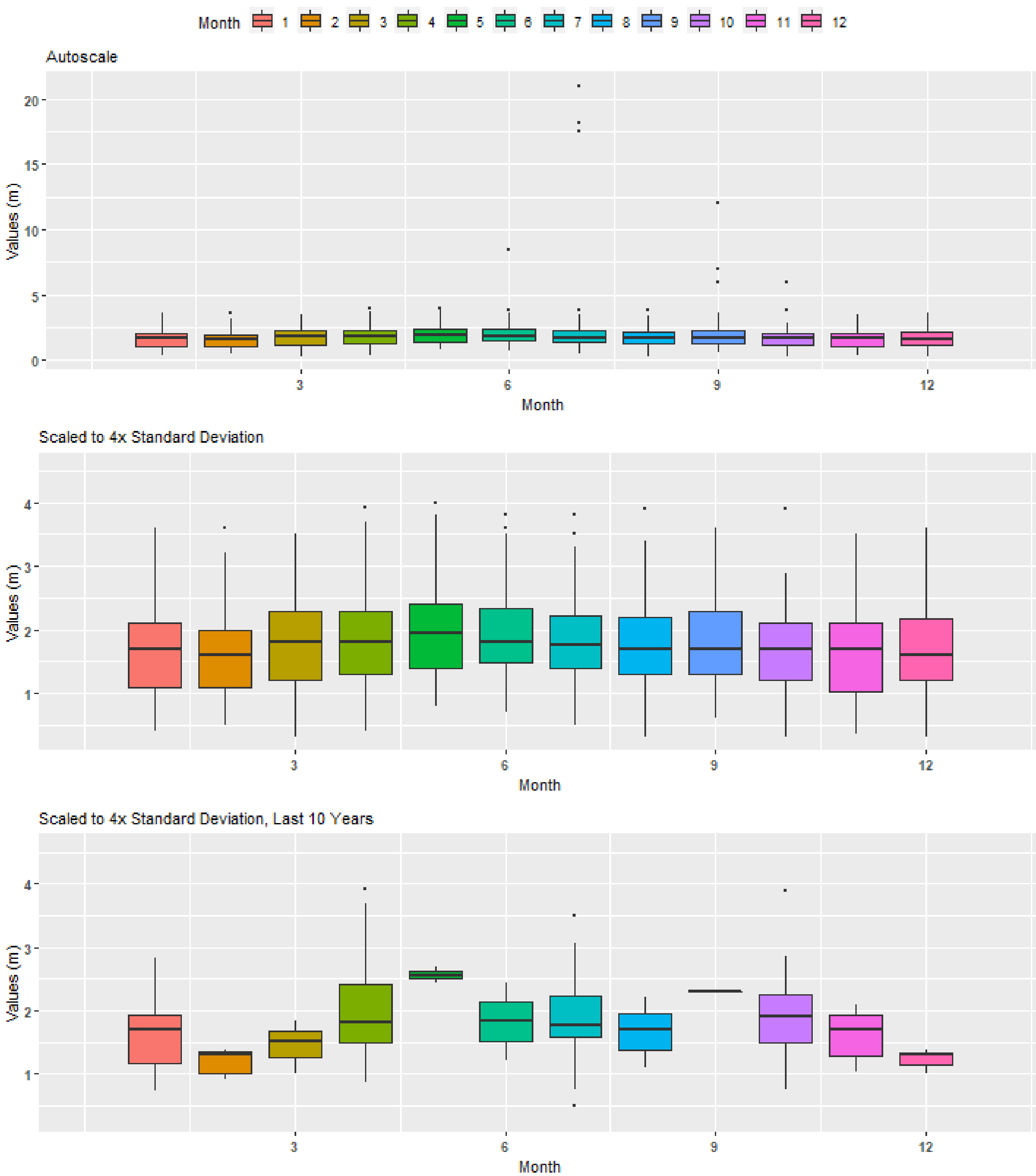


Summary Box Plots for Nature Coast Aquatic Preserve

By Year & Month



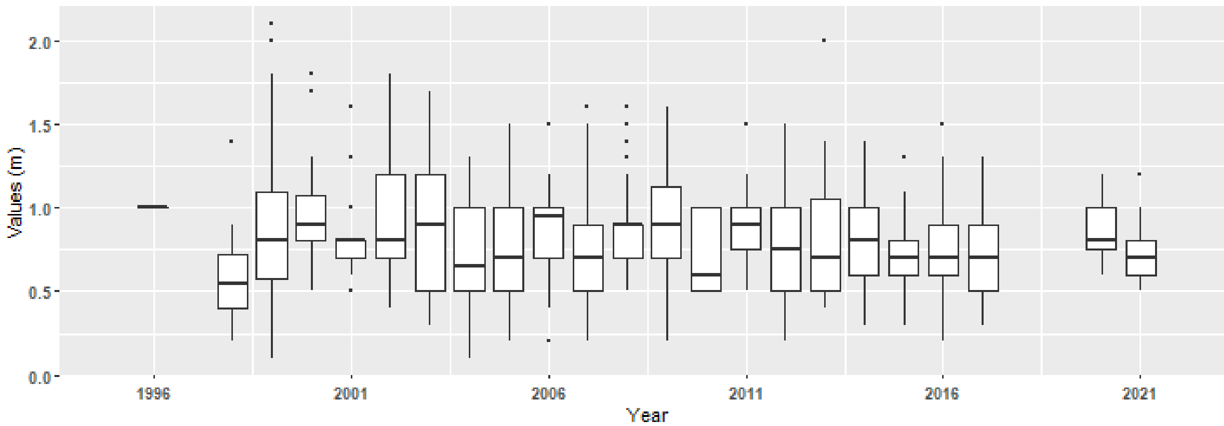
Summary Box Plots for Nature Coast Aquatic Preserve
By Month



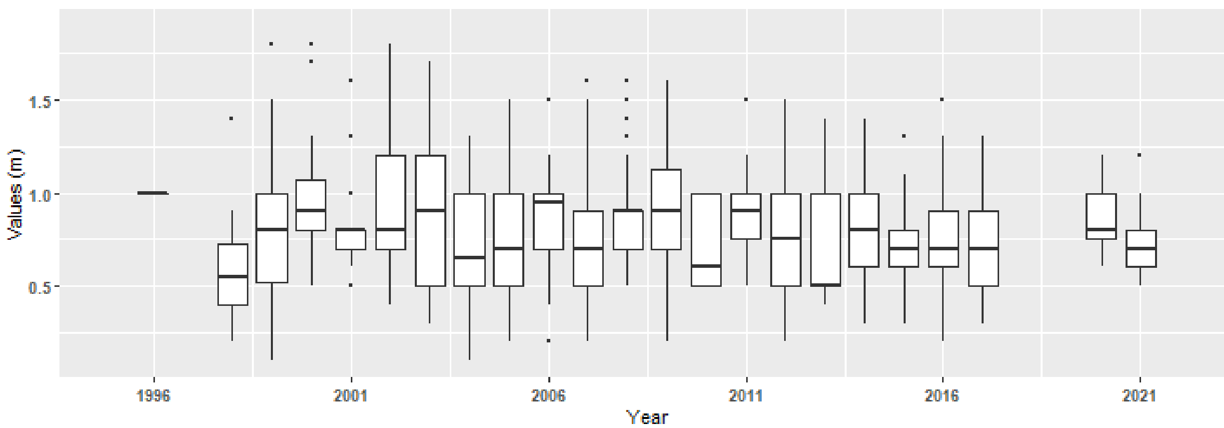
Summary Box Plots for North Fork St. Lucie Aquatic Preserve

By Year

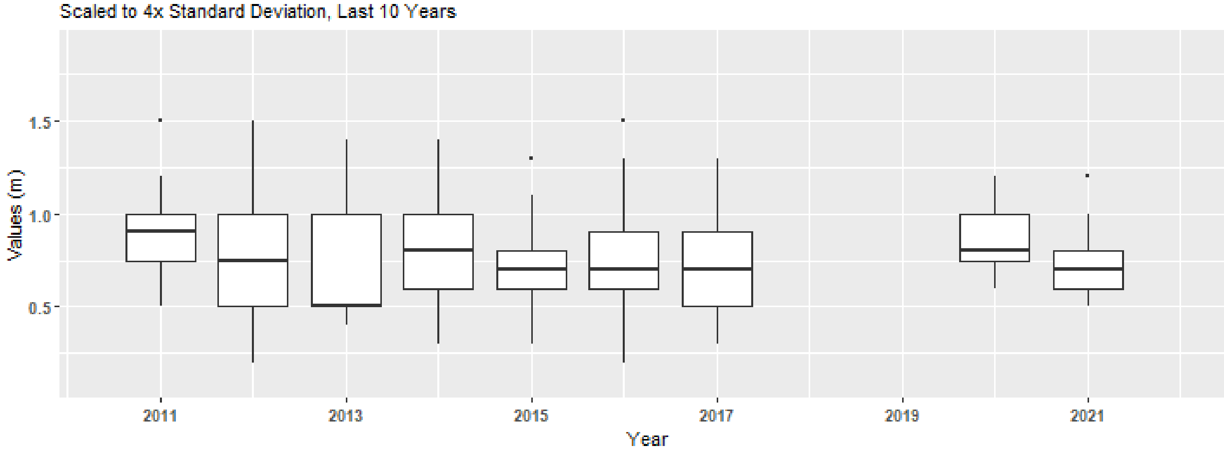
Autoscale



Scaled to 4x Standard Deviation

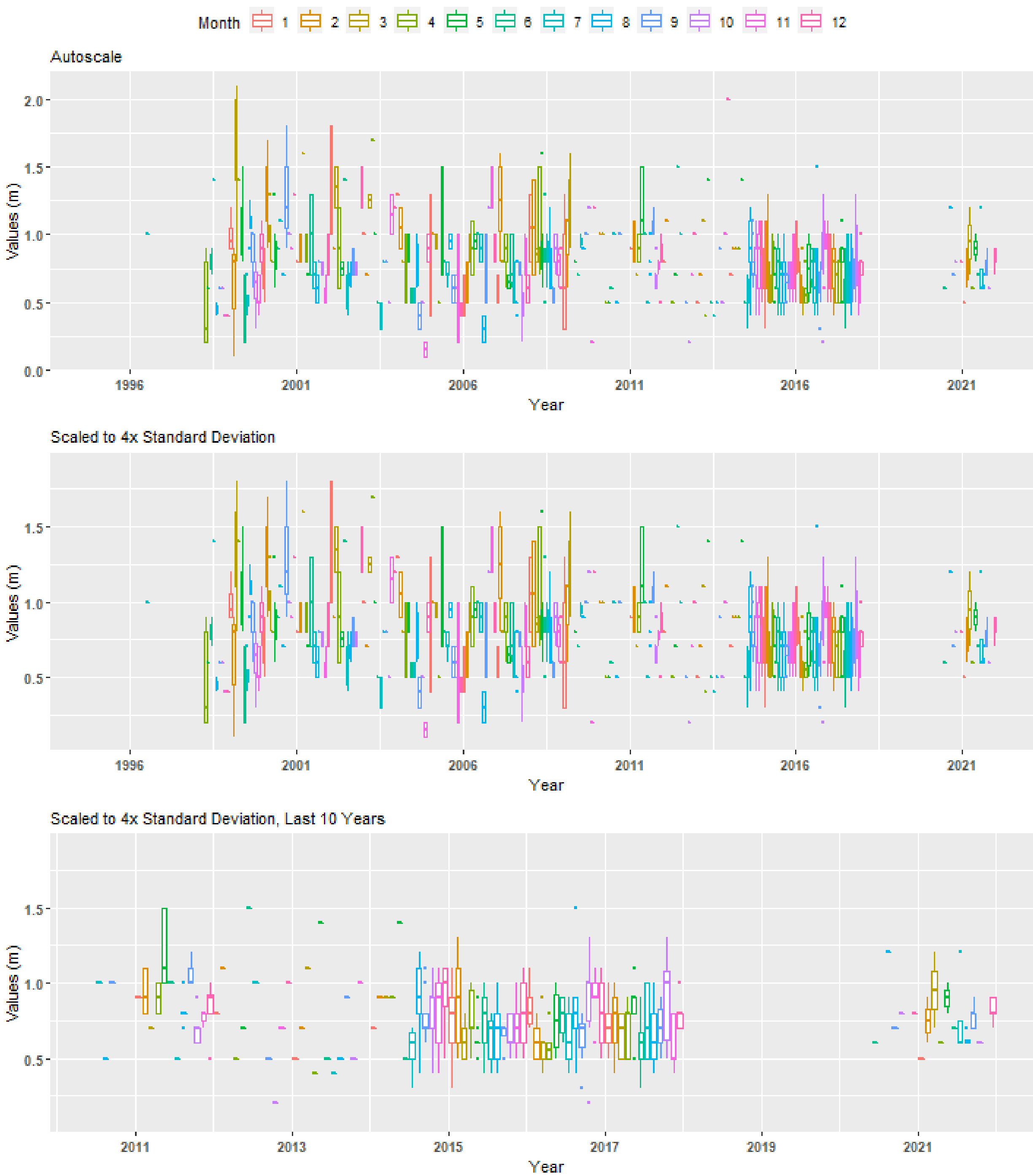


Scaled to 4x Standard Deviation, Last 10 Years

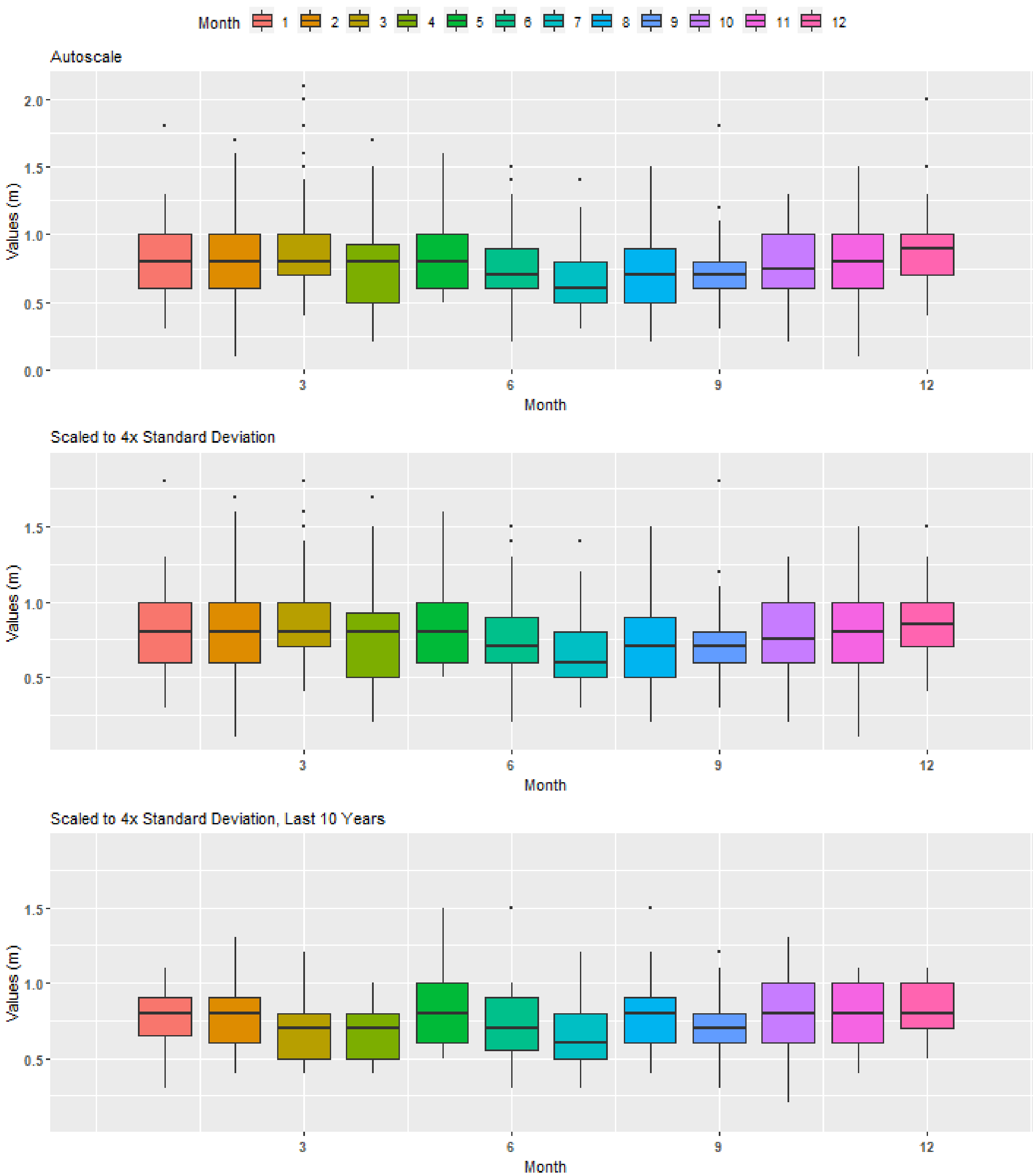


Summary Box Plots for North Fork St. Lucie Aquatic Preserve

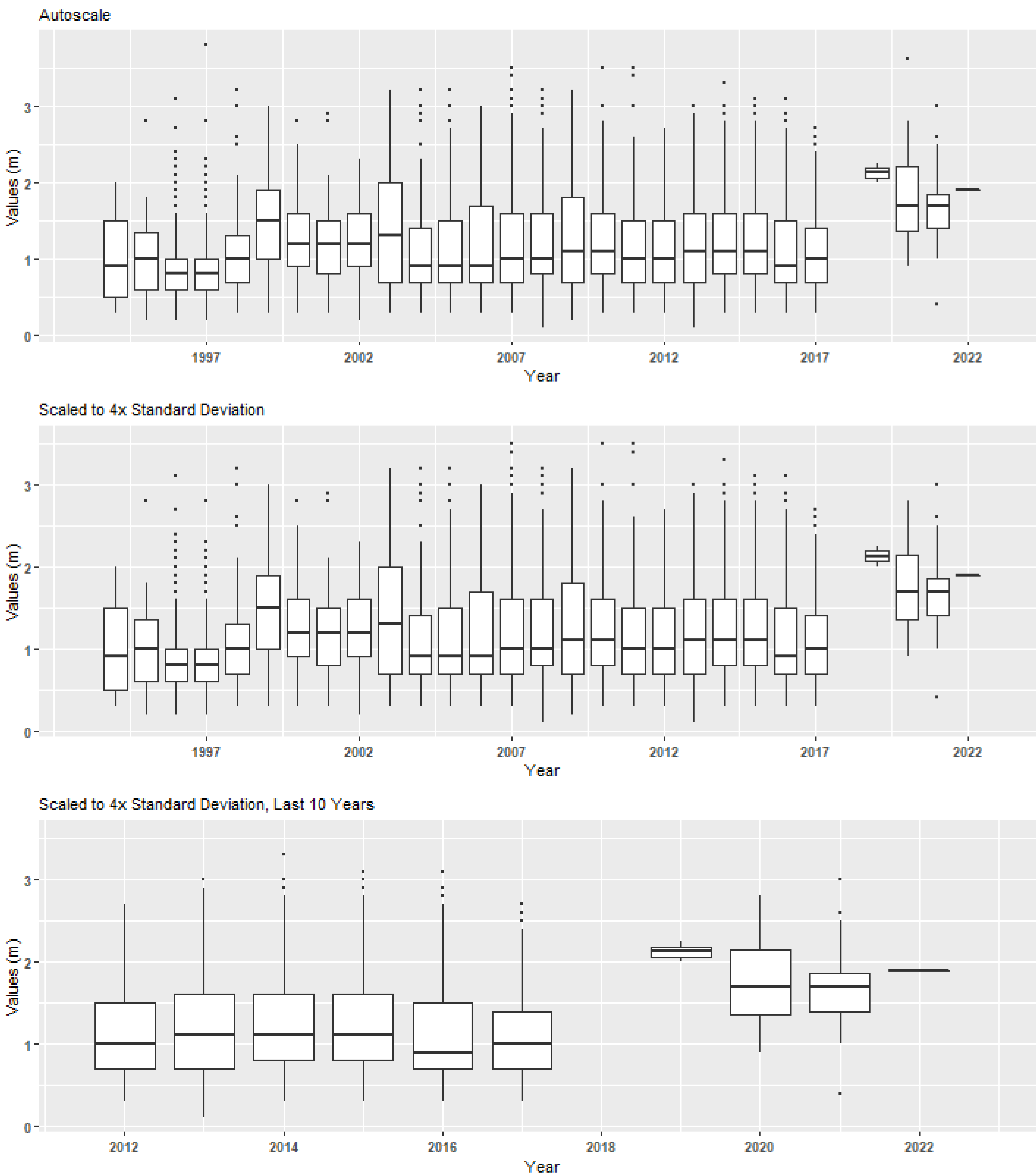
By Year & Month



Summary Box Plots for North Fork St. Lucie Aquatic Preserve
By Month

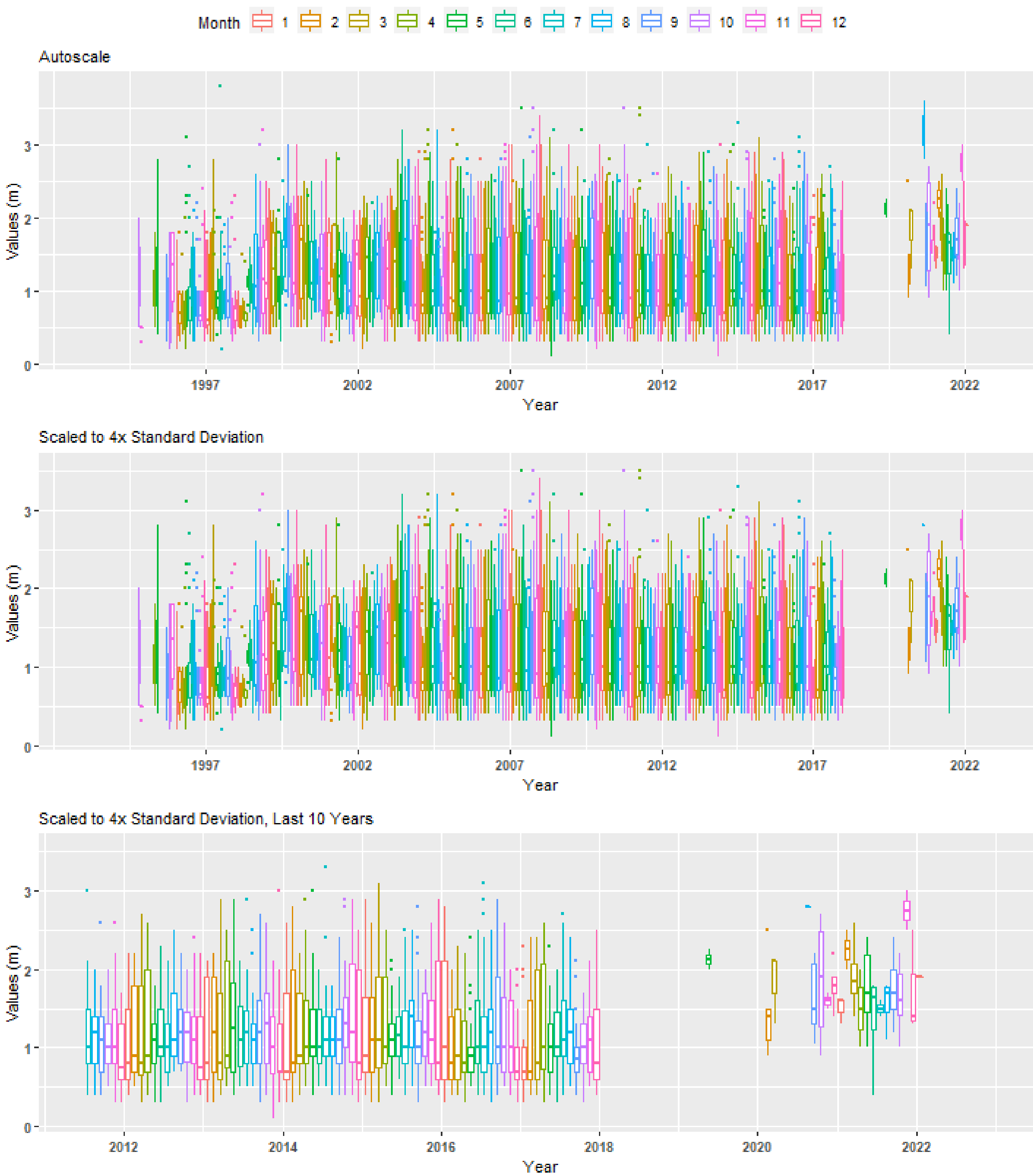


Summary Box Plots for Pine Island Sound Aquatic Preserve
By Year

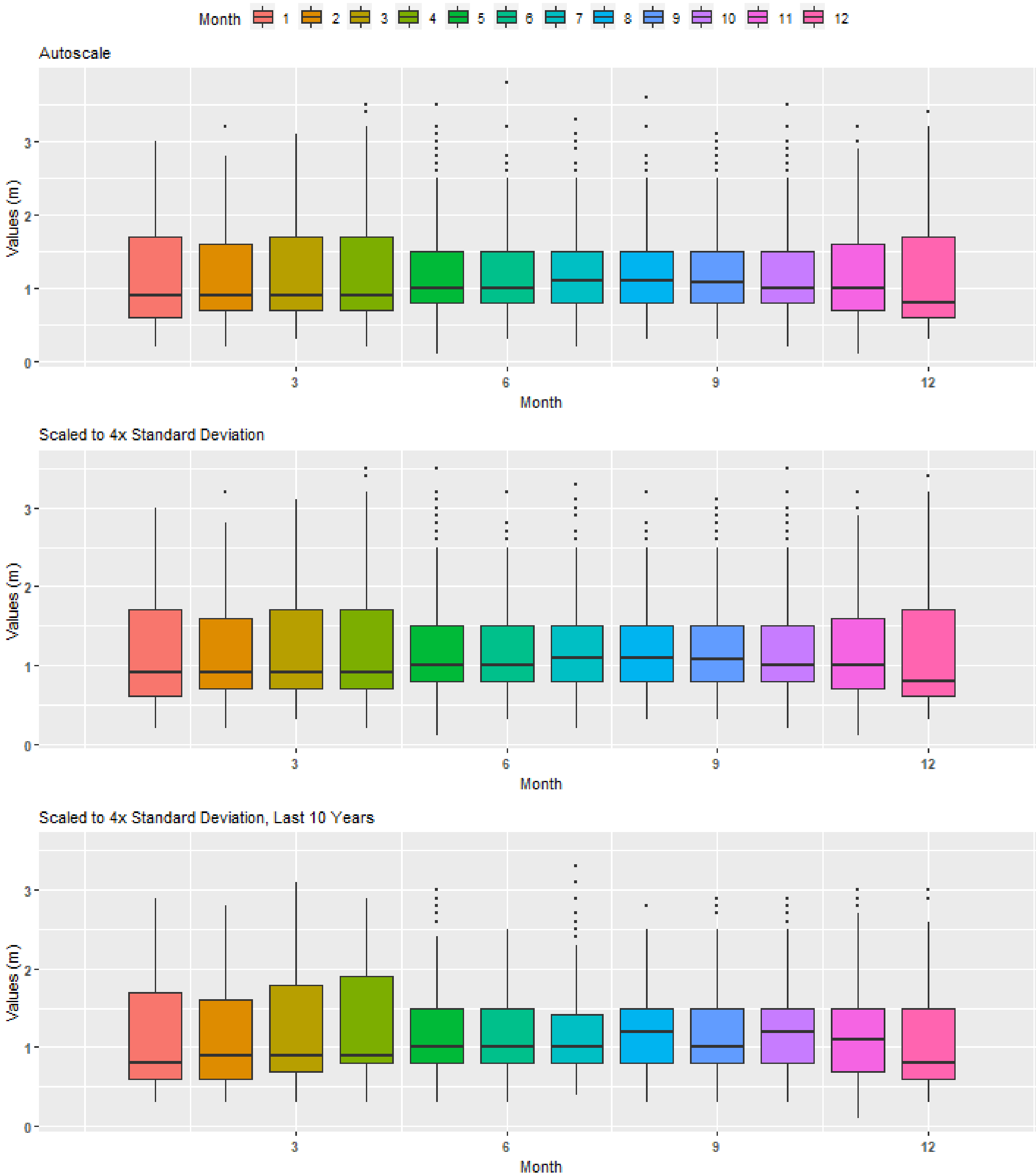


Summary Box Plots for Pine Island Sound Aquatic Preserve

By Year & Month

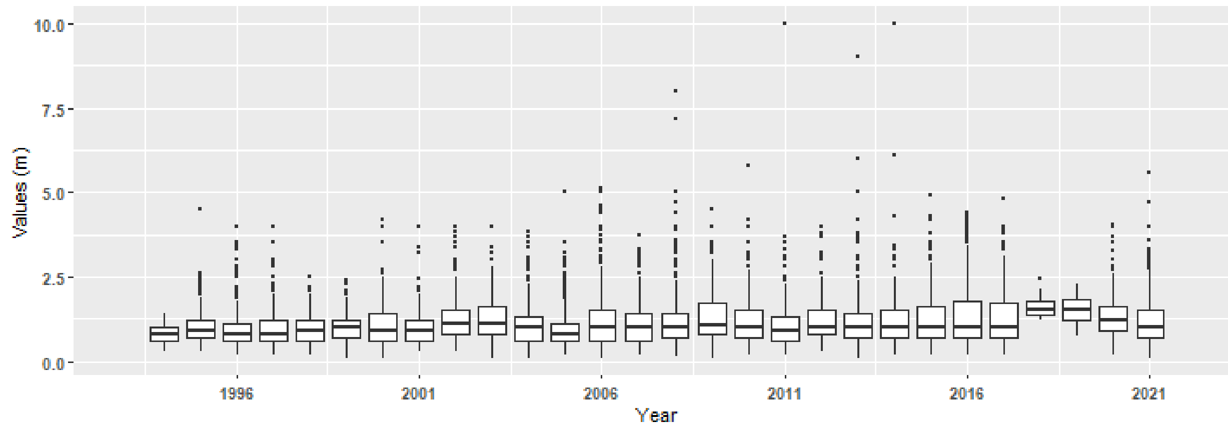


Summary Box Plots for Pine Island Sound Aquatic Preserve
By Month

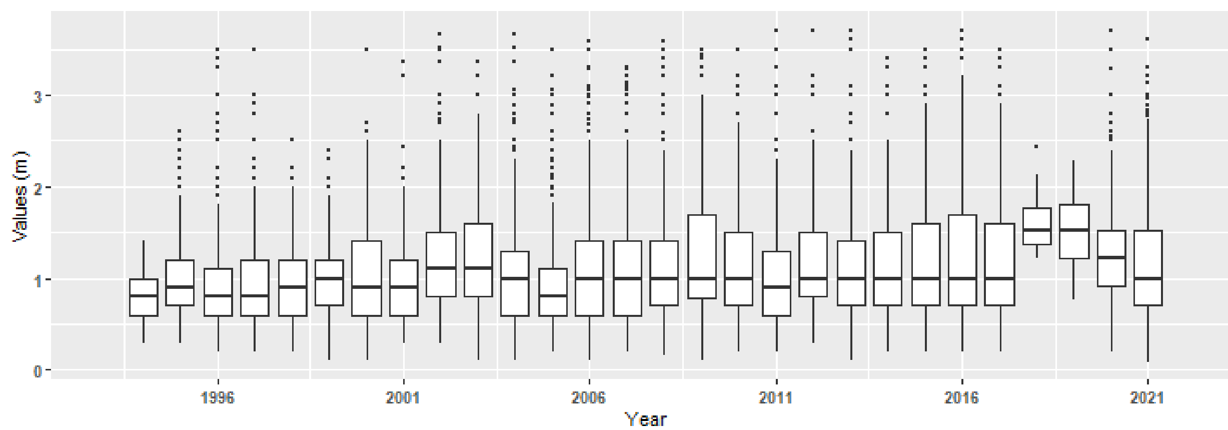


Summary Box Plots for Pinellas County Aquatic Preserve
By Year

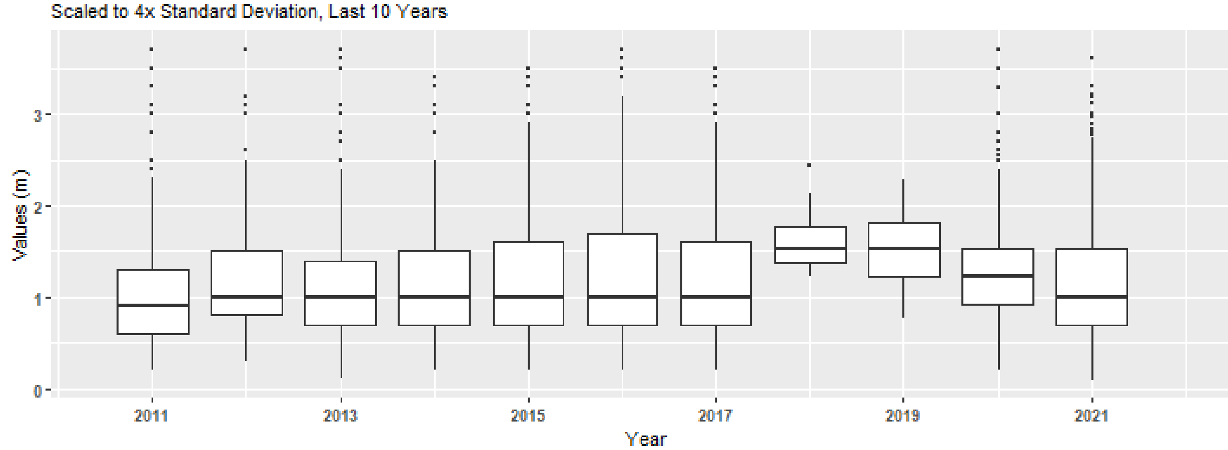
Autoscale



Scaled to 4x Standard Deviation

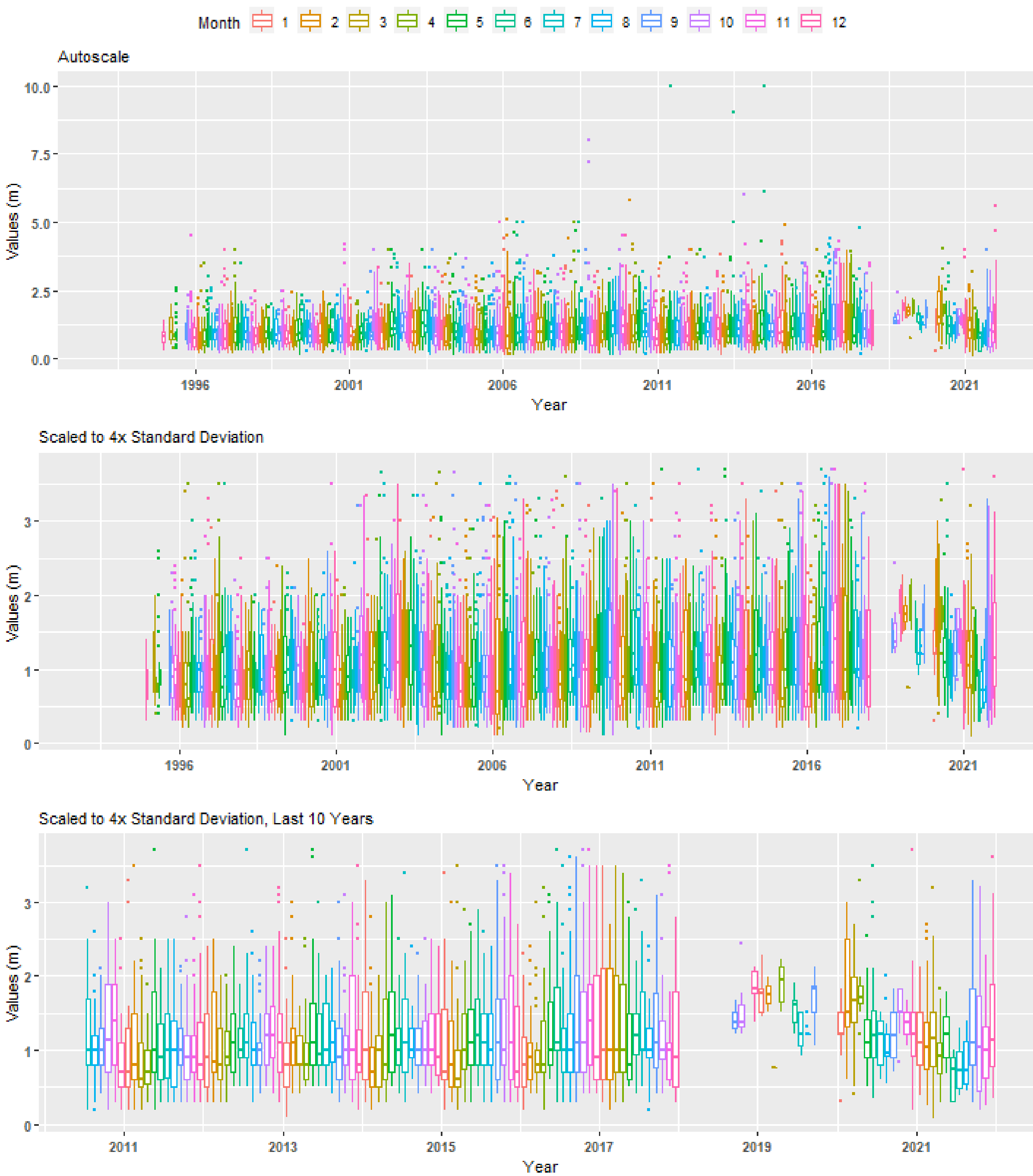


Scaled to 4x Standard Deviation, Last 10 Years



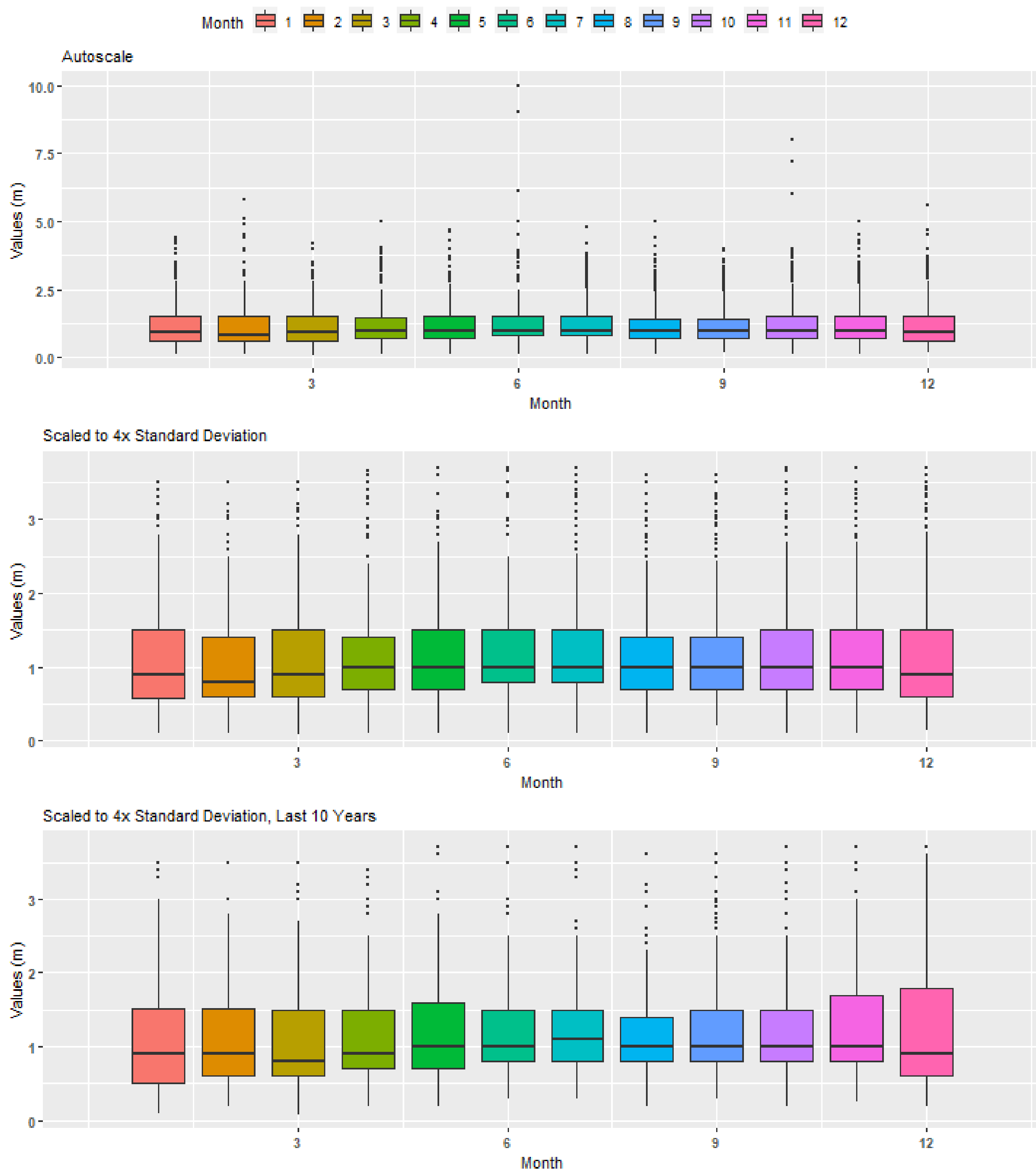
Summary Box Plots for Pinellas County Aquatic Preserve

By Year & Month



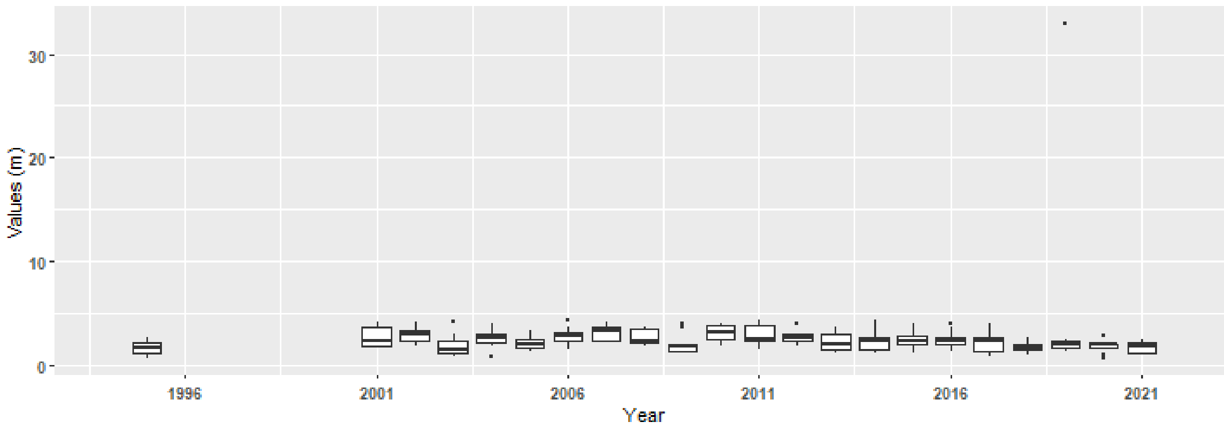
Summary Box Plots for Pinellas County Aquatic Preserve

By Month

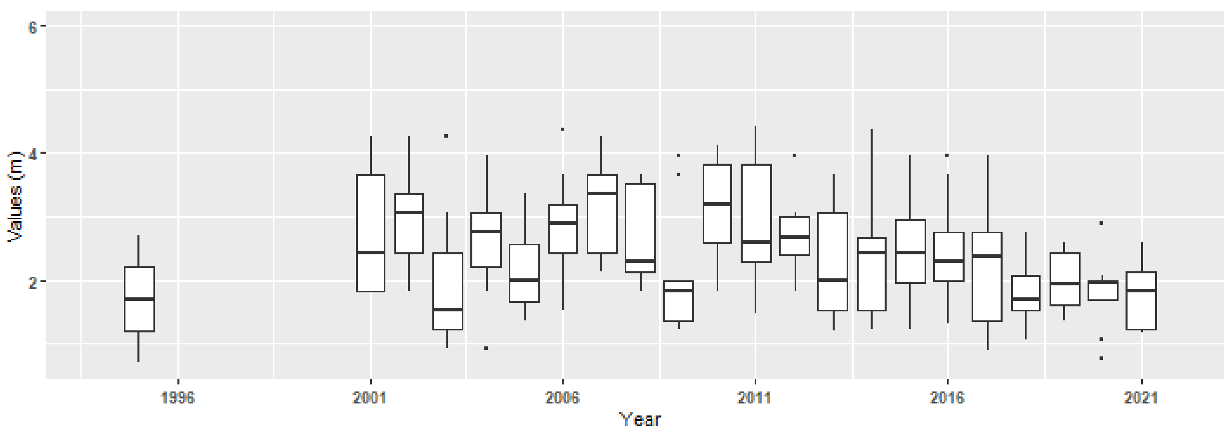


Summary Box Plots for Rocky Bayou State Park Aquatic Preserve
By Year

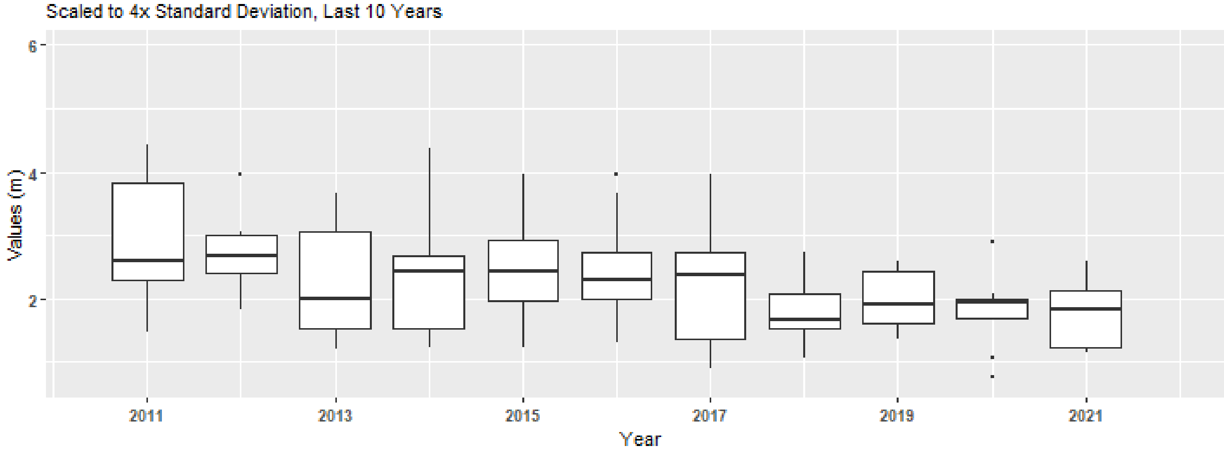
Autoscale



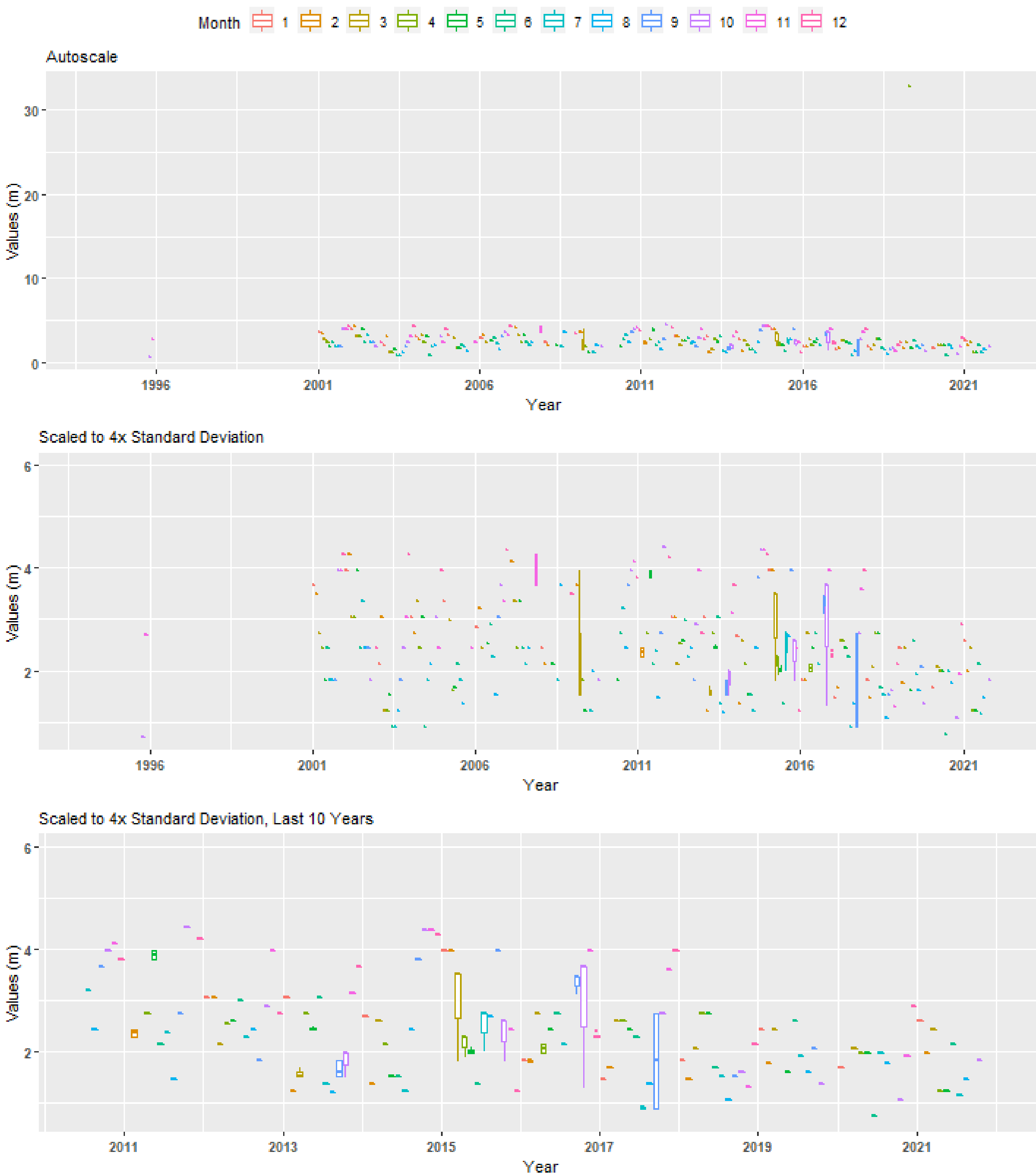
Scaled to 4x Standard Deviation



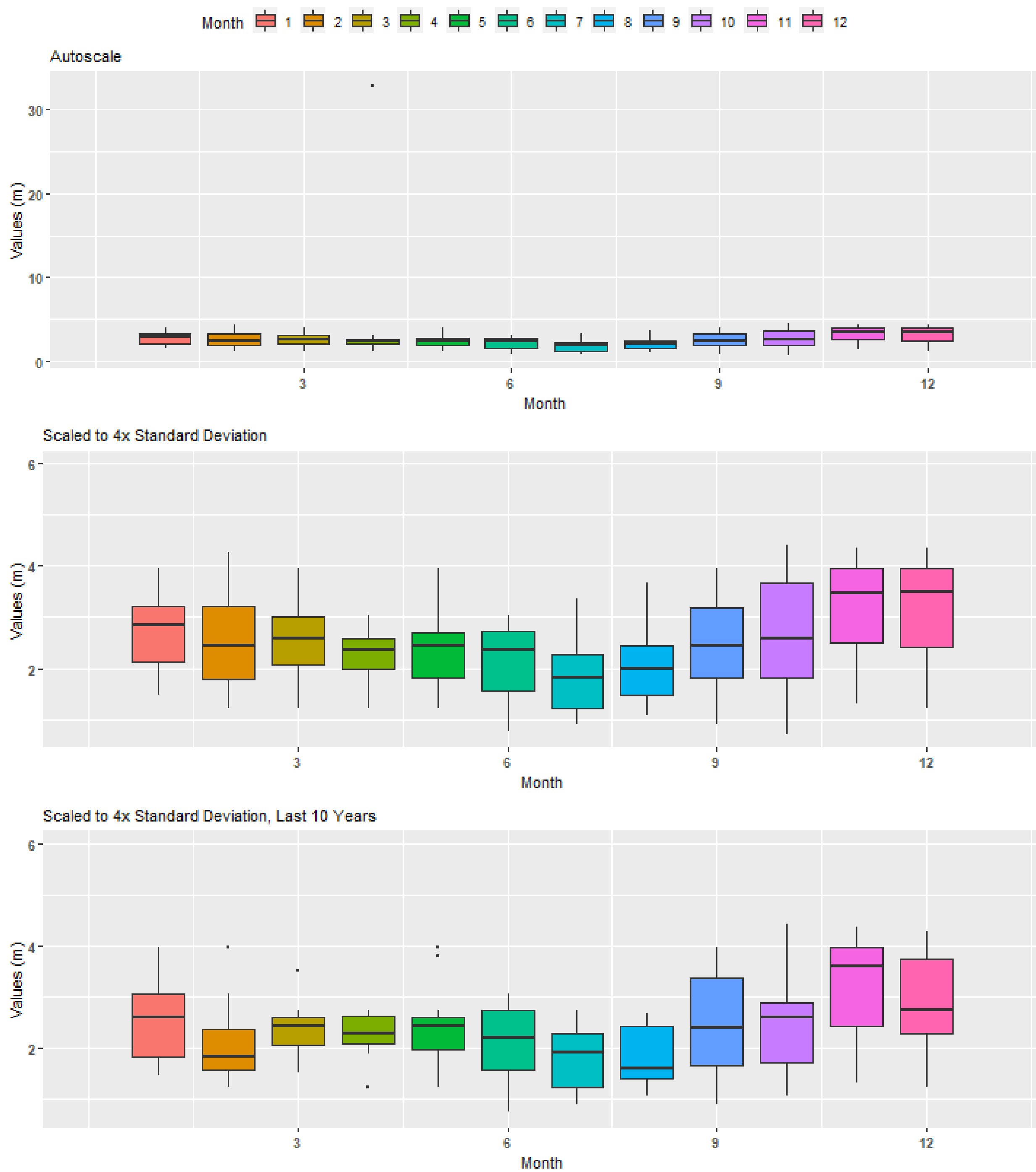
Scaled to 4x Standard Deviation, Last 10 Years



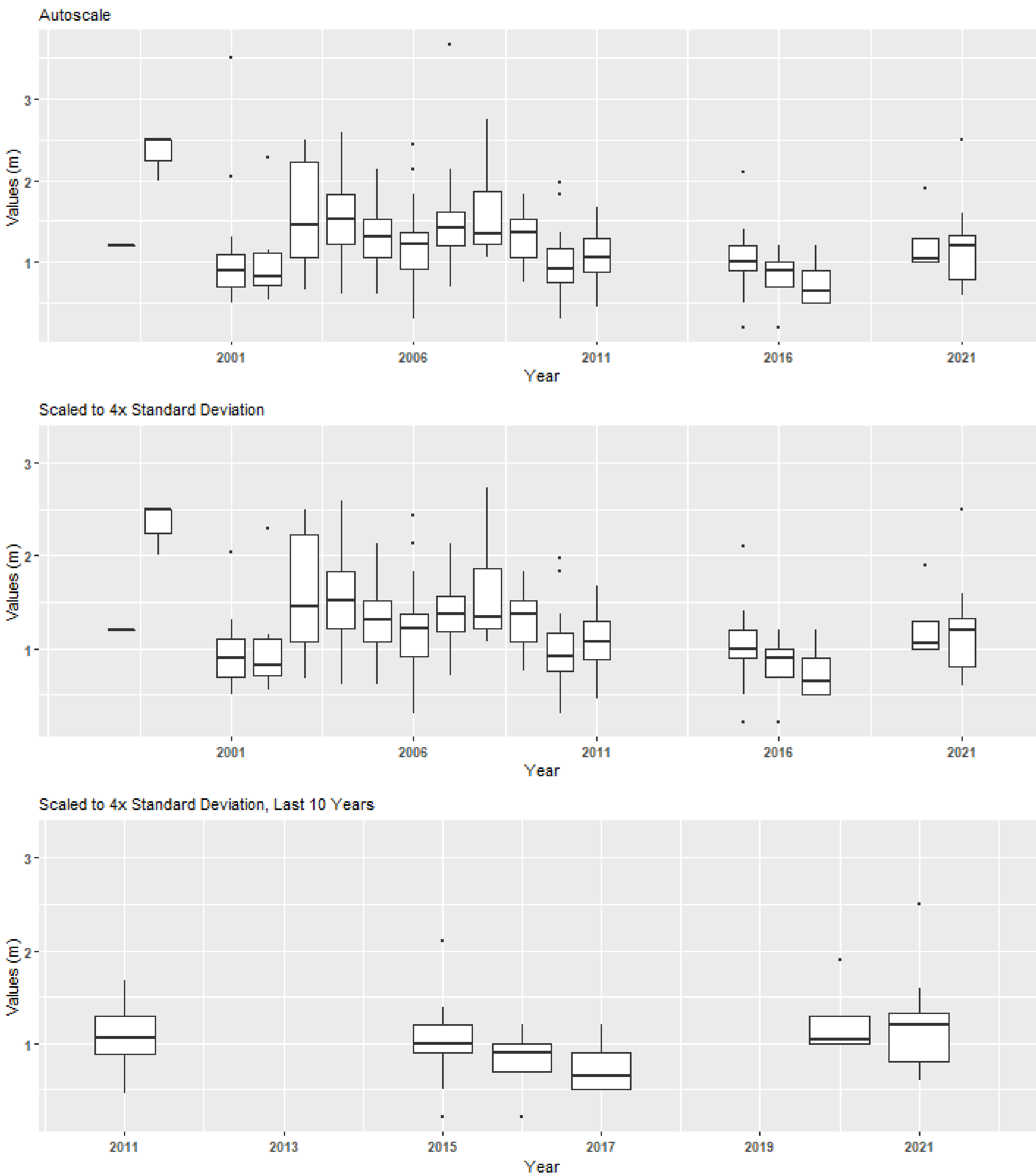
Summary Box Plots for Rocky Bayou State Park Aquatic Preserve
By Year & Month



Summary Box Plots for Rocky Bayou State Park Aquatic Preserve
By Month



Summary Box Plots for Rookery Bay Aquatic Preserve
By Year

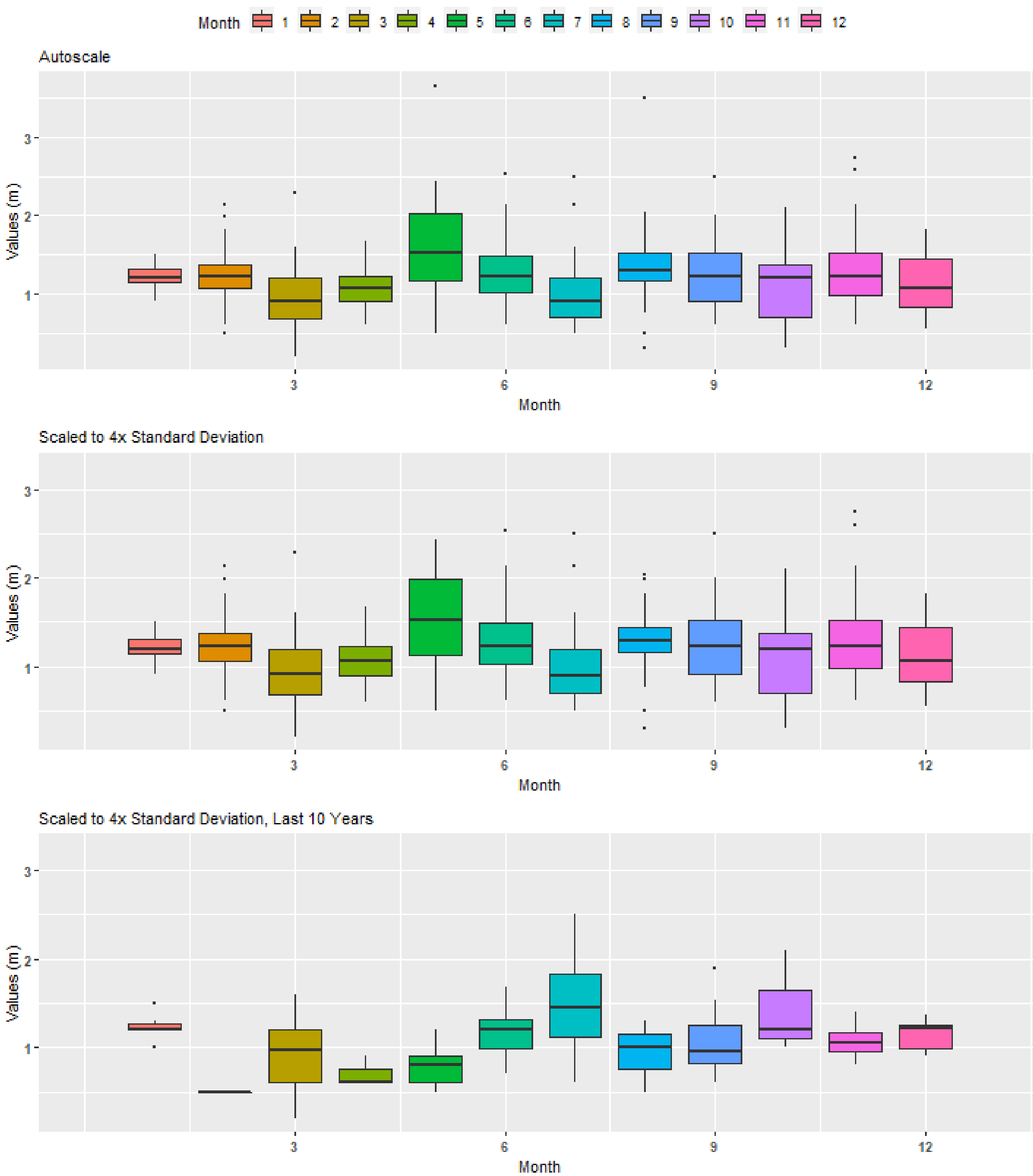


Summary Box Plots for Rookery Bay Aquatic Preserve

By Year & Month

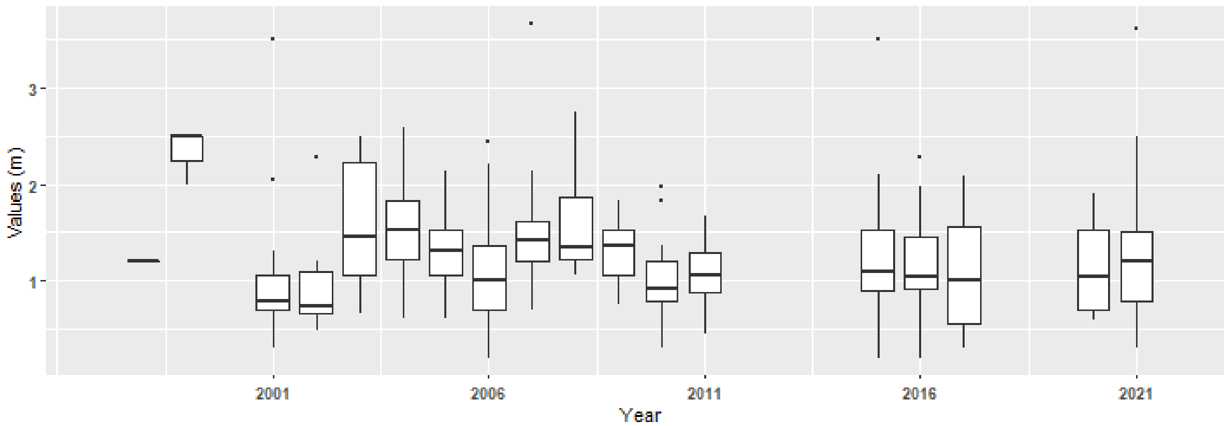
Summary Box Plots for Rookery Bay Aquatic Preserve

By Month

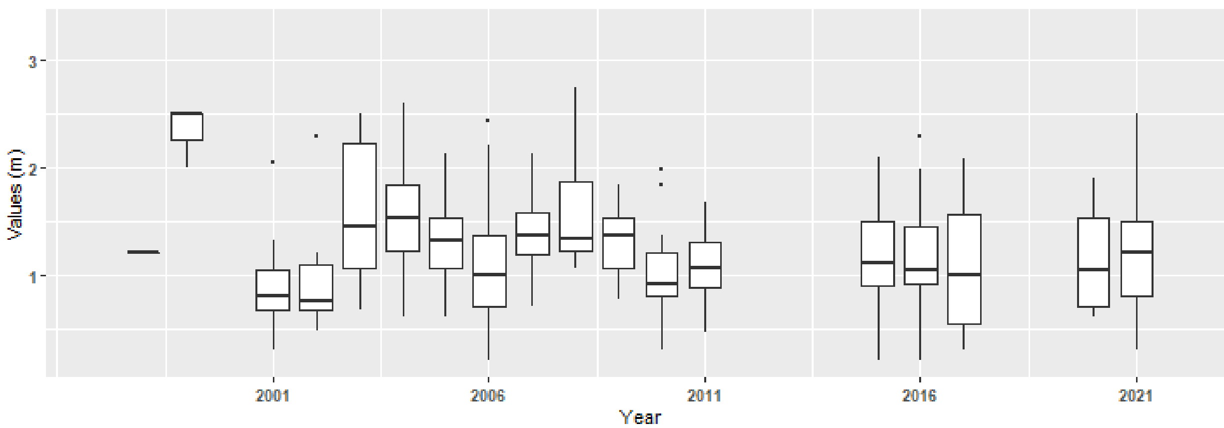


Summary Box Plots for Rookery Bay National Estuarine Research Reserve
By Year

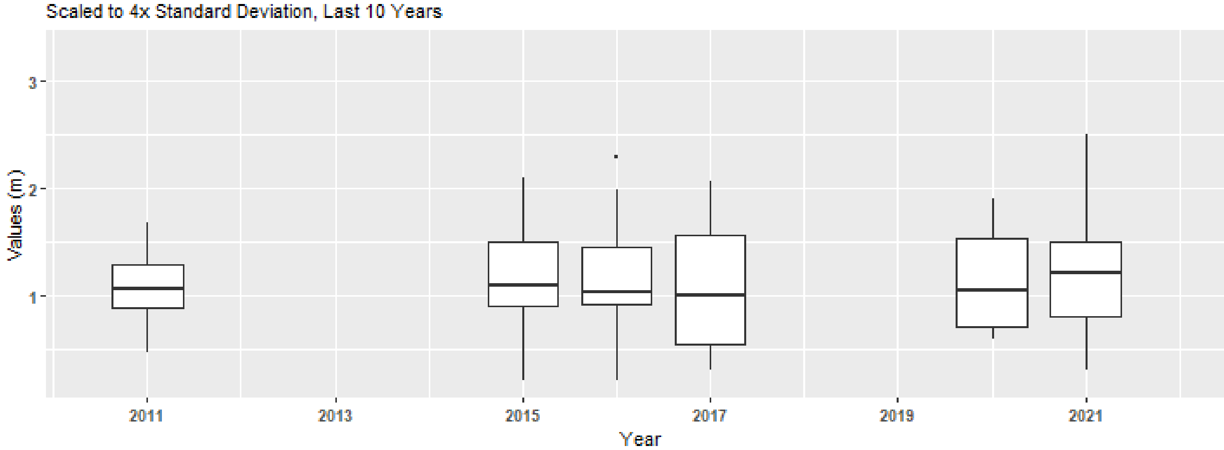
Autoscale

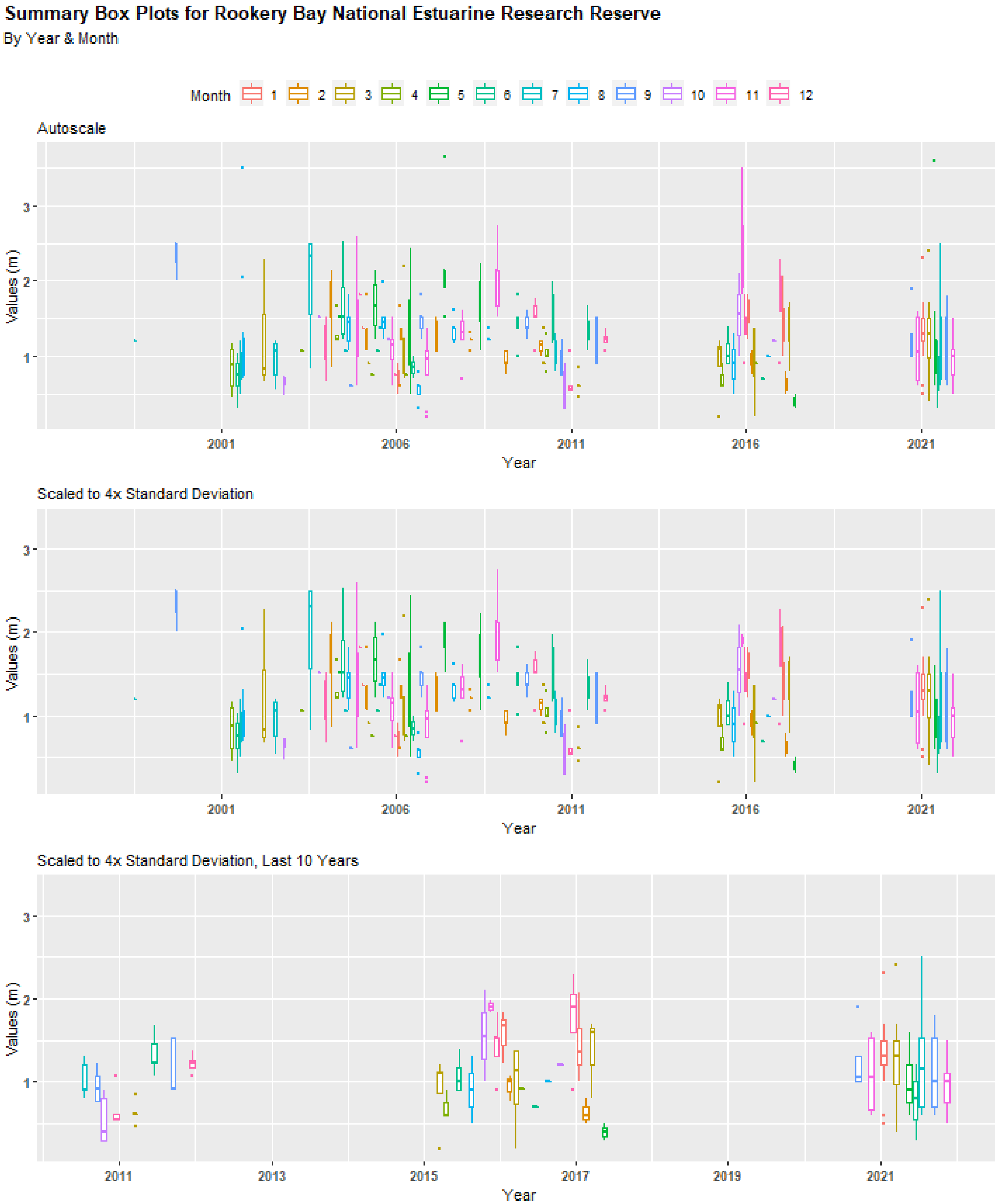


Scaled to 4x Standard Deviation



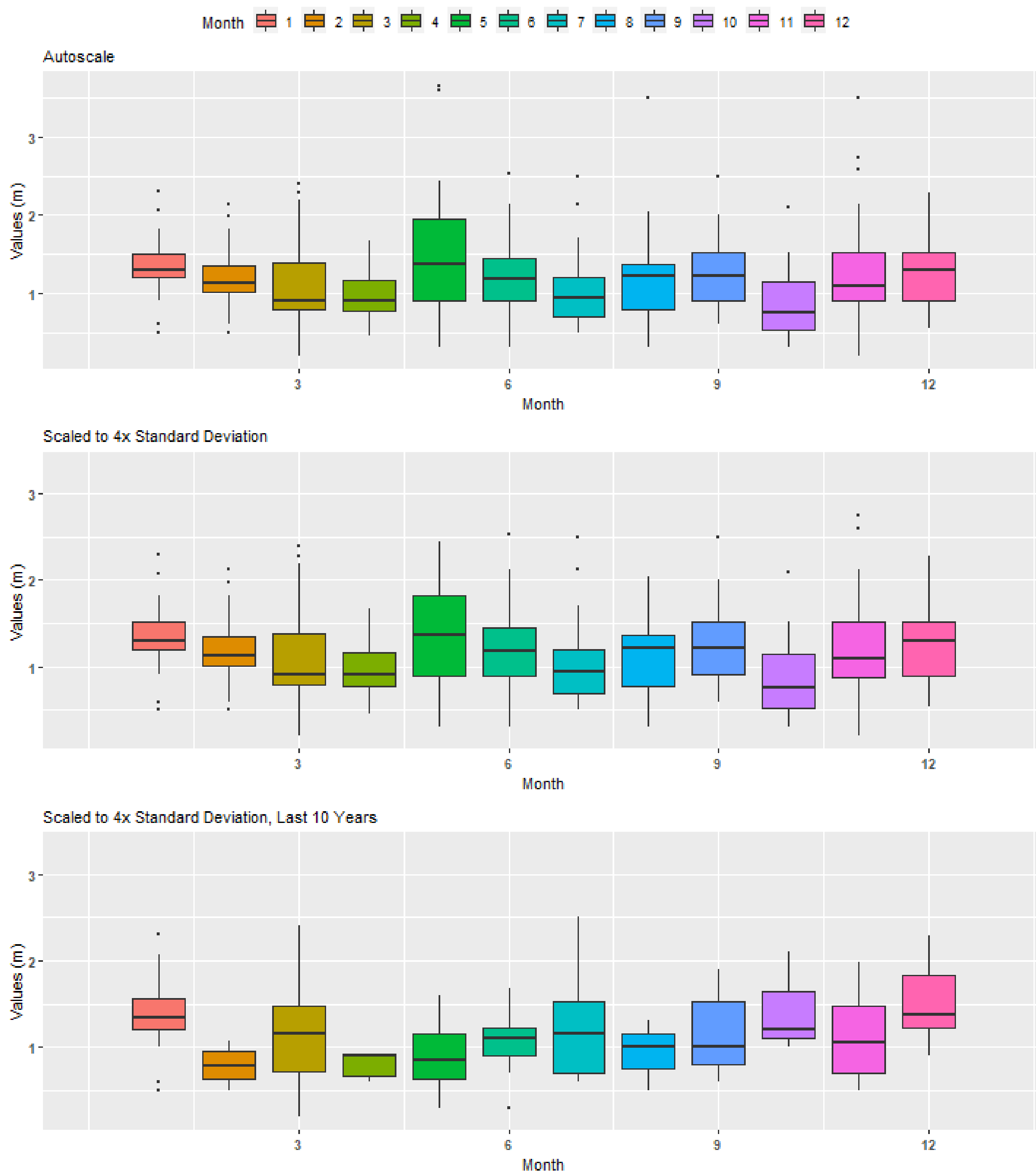
Scaled to 4x Standard Deviation, Last 10 Years




Summary Box Plots for Rookery Bay National Estuarine Research Reserve
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (m)
3
2
1
2001
2006
2011
2016
2021
Year
Scaled to 4x Standard Deviation
Values (m)
3
2
1
2001
2006
2011
2016
2021
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (m)
3
2
1
2011
2013
2015
2017
2019
2021
Year

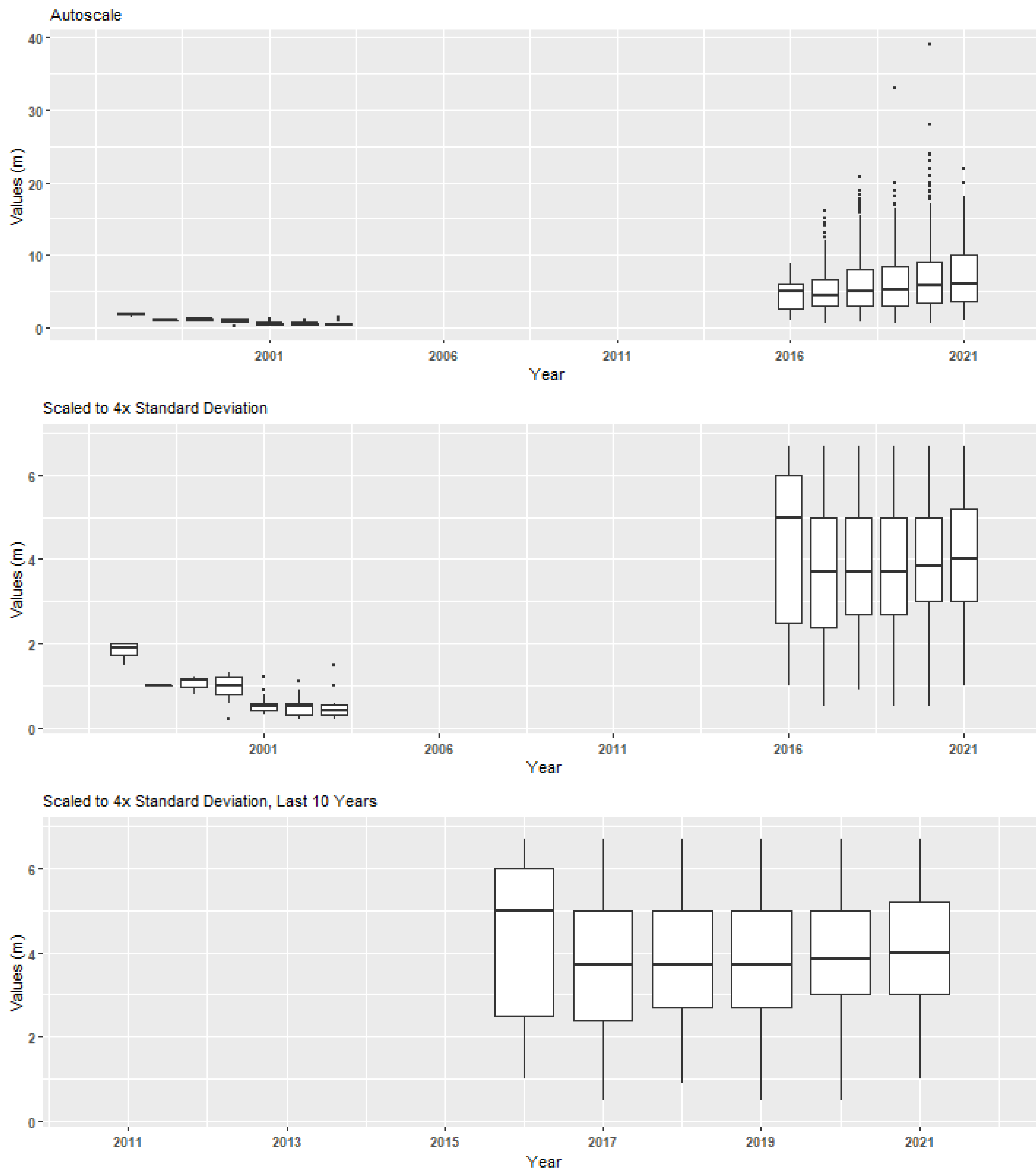
Summary Box Plots for Rookery Bay National Estuarine Research Reserve

By Month

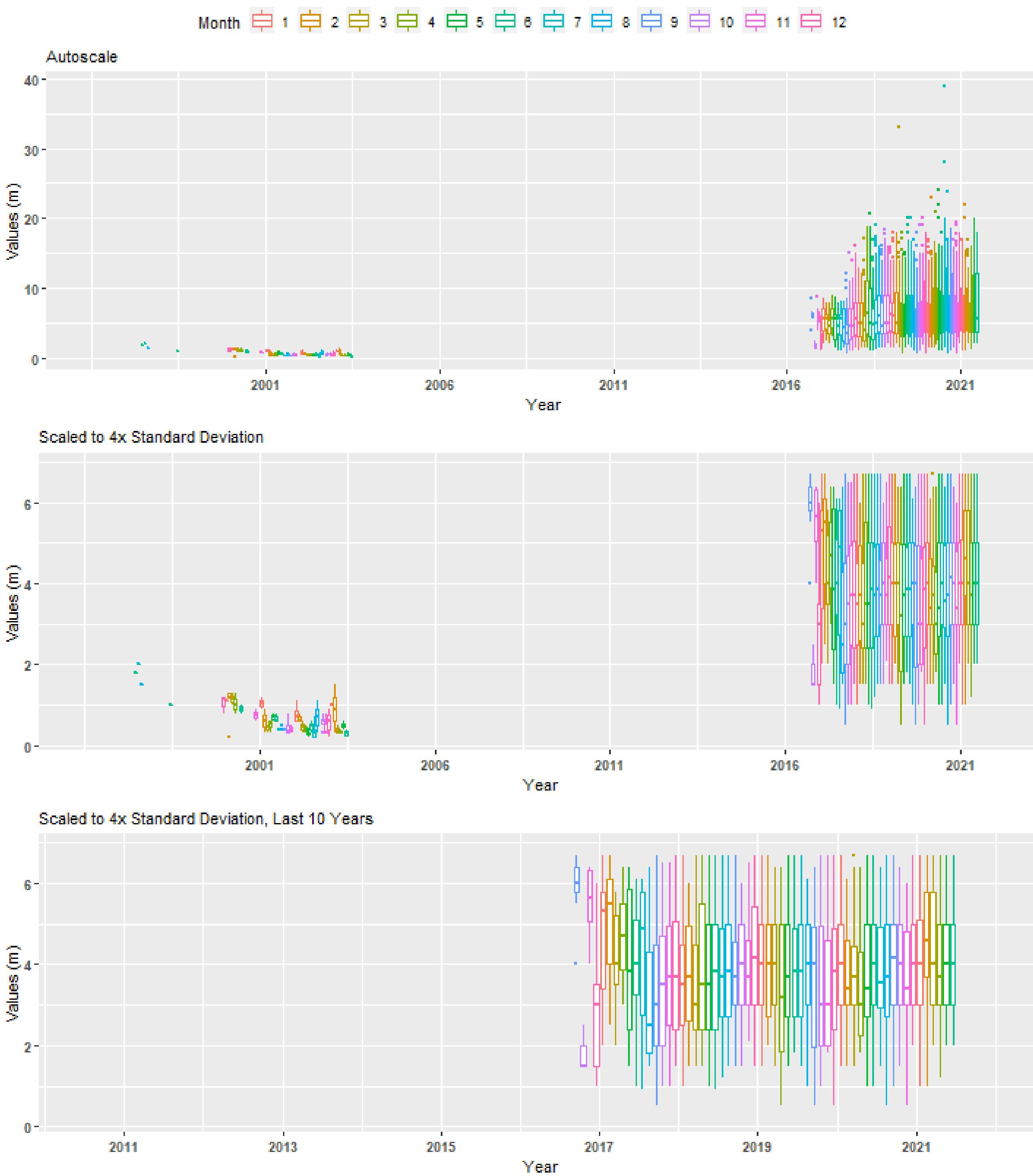


Summary Box Plots for Southeast Florida Coral Reef Ecosystem Conservation Area

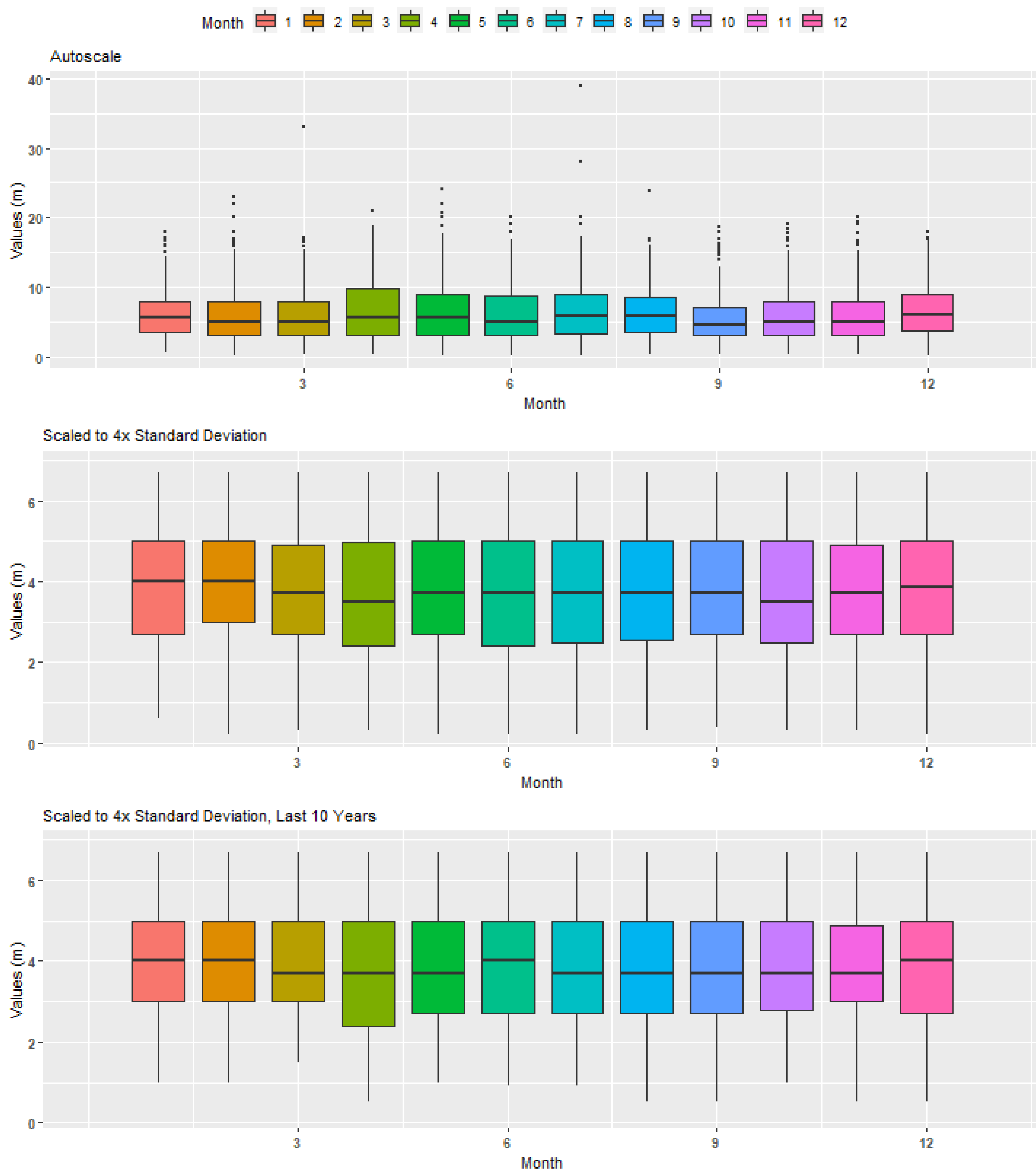
By Year



Summary Box Plots for Southeast Florida Coral Reef Ecosystem Conservation Area
By Year & Month

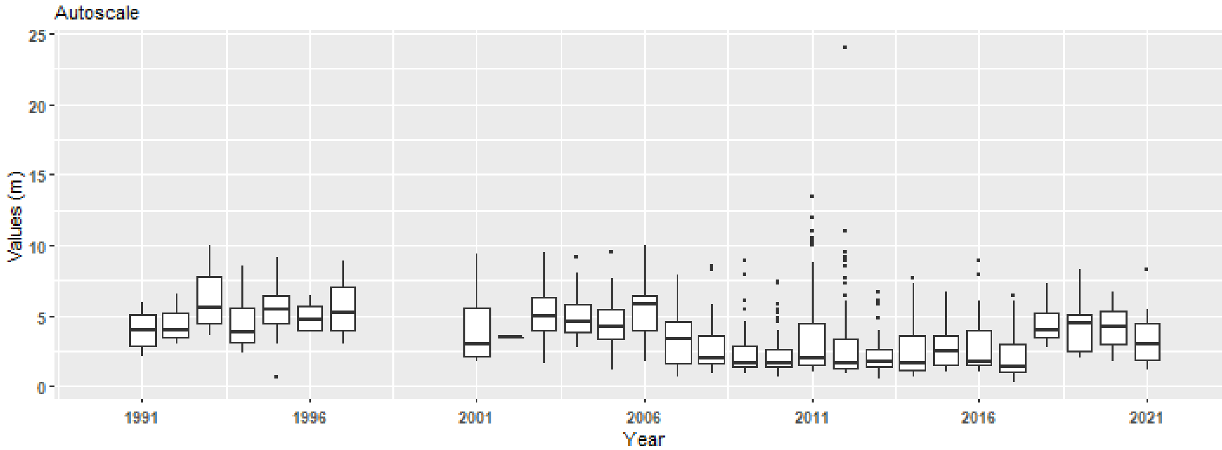


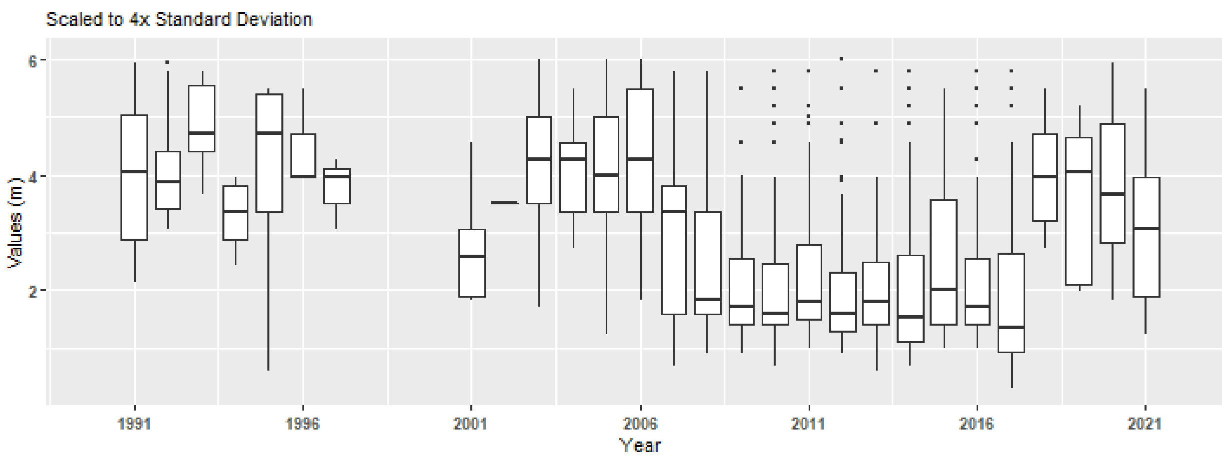
Summary Box Plots for Southeast Florida Coral Reef Ecosystem Conservation Area
By Month

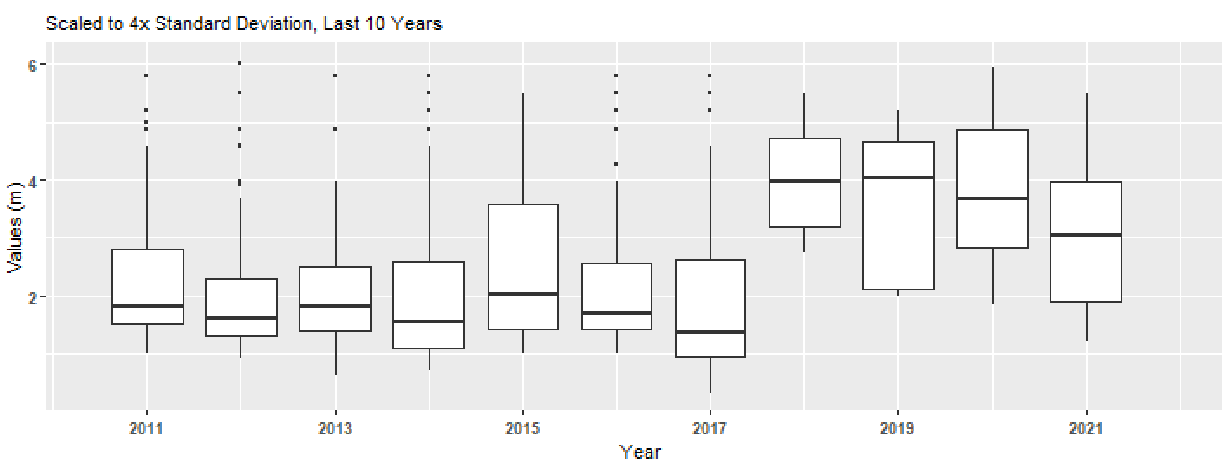


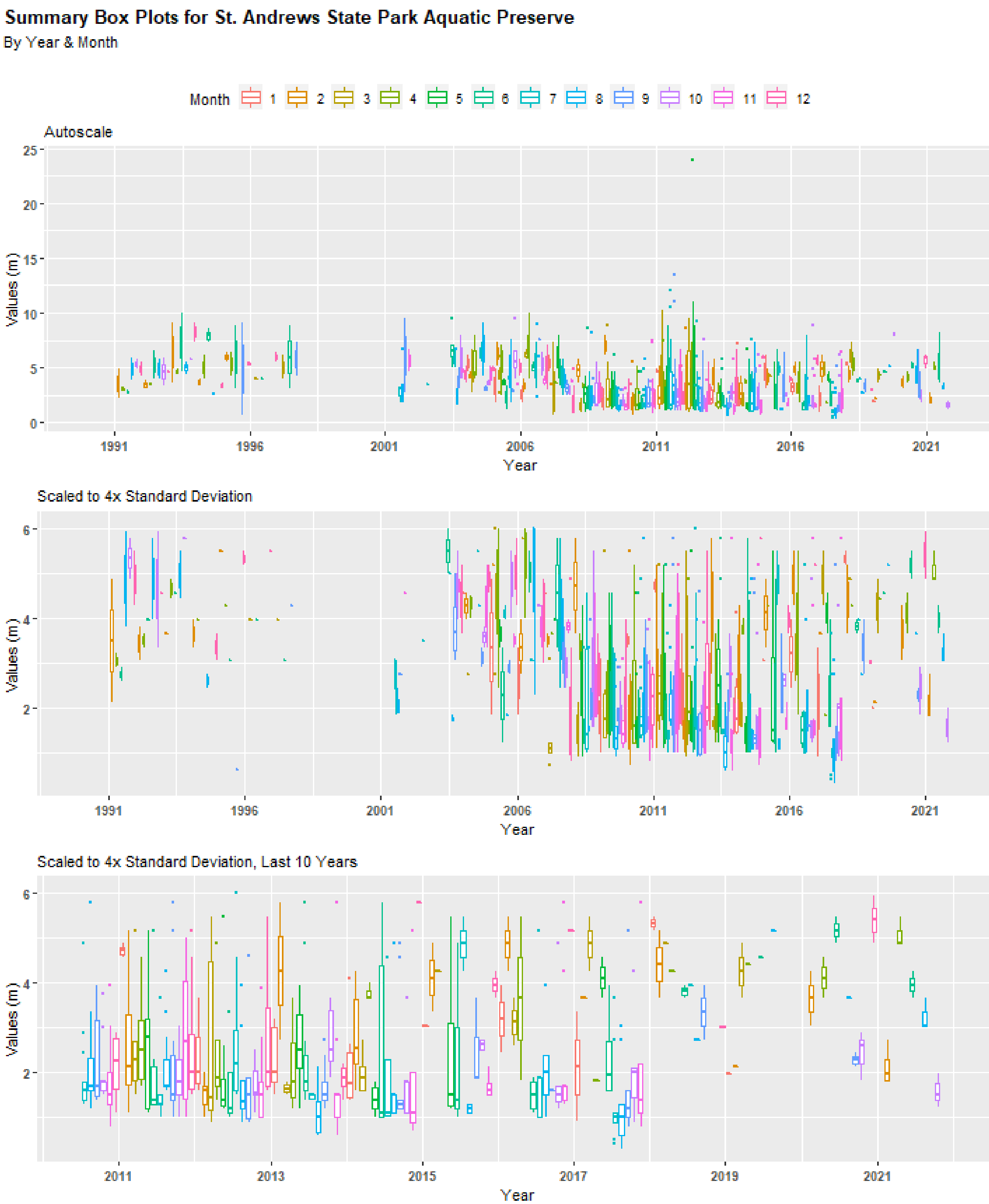
Summary Box Plots for St. Andrews State Park Aquatic Preserve

By Year





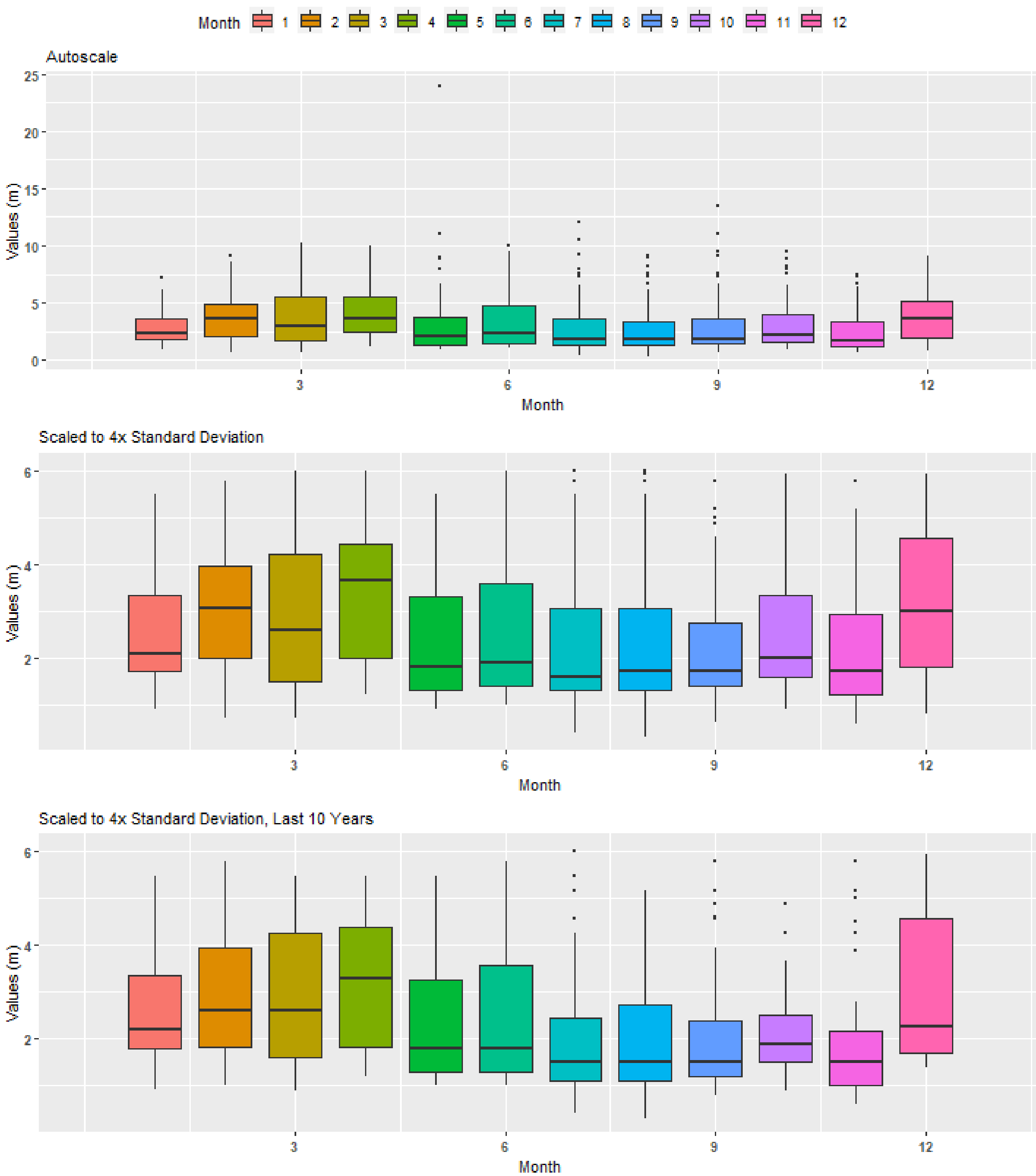




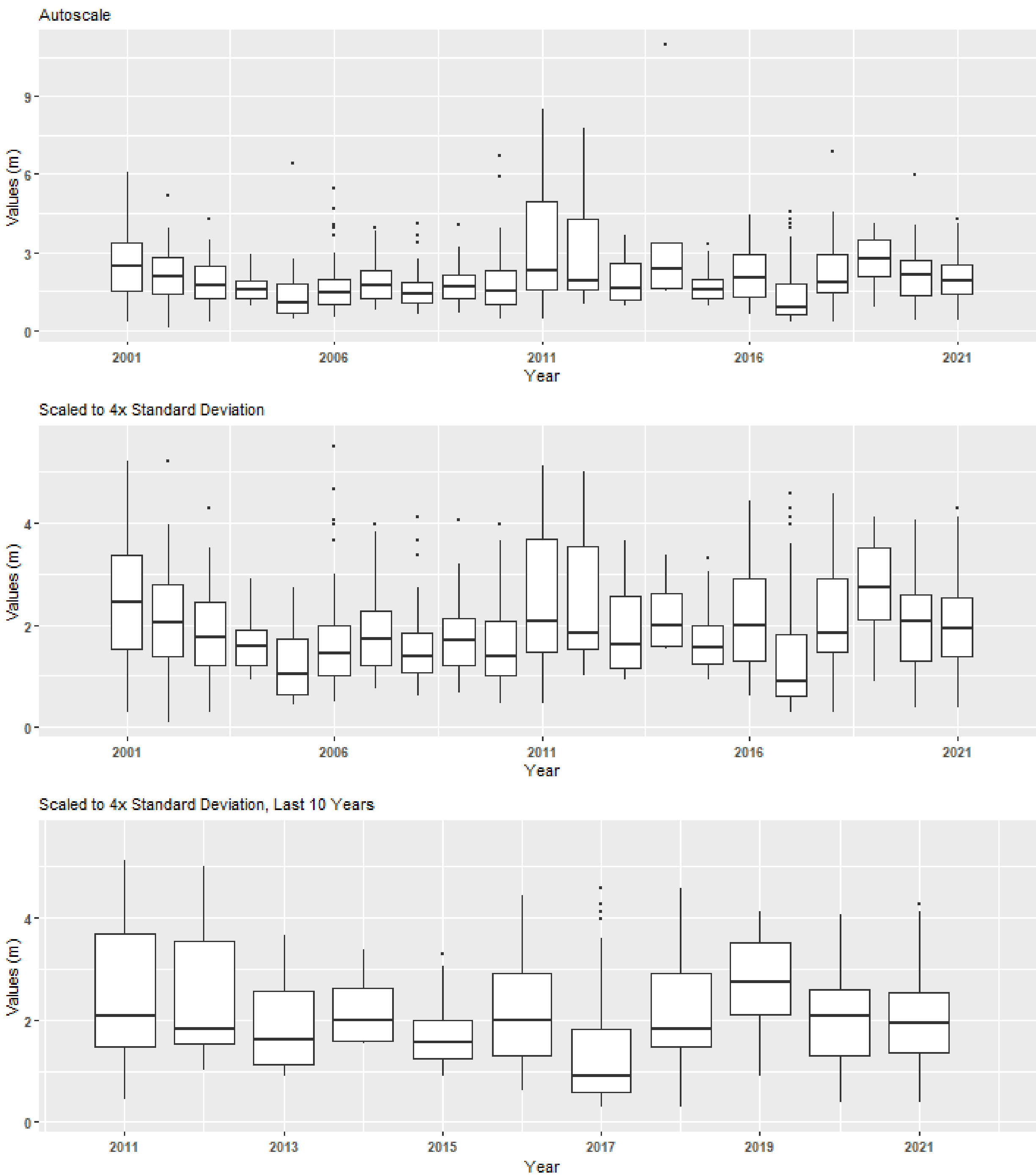
Summary Box Plots for St. Andrews State Park Aquatic Preserve
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (m)
Year
Scaled to 4x Standard Deviation
Values (m)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (m)
Year

Summary Box Plots for St. Andrews State Park Aquatic Preserve

By Month

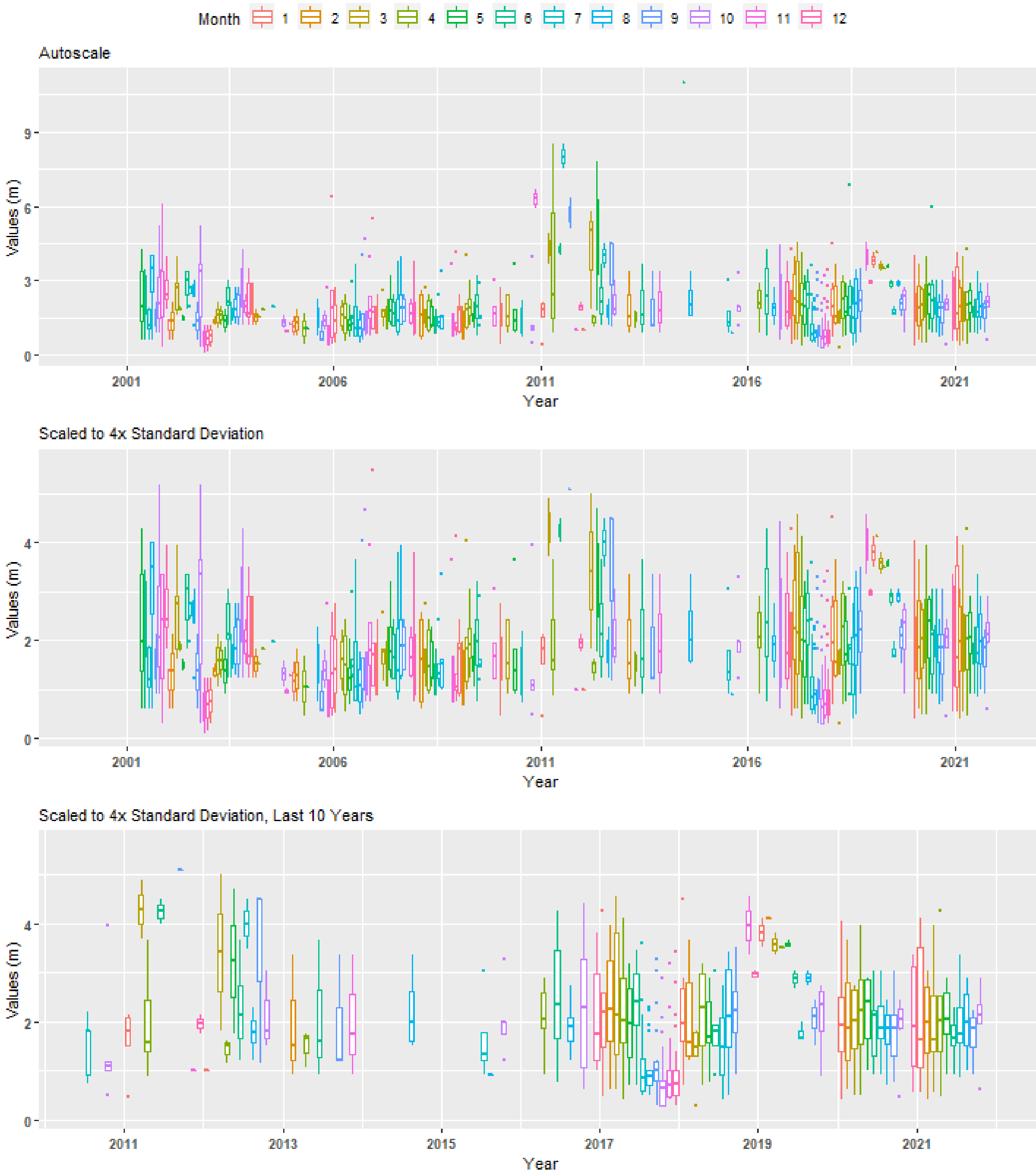


Summary Box Plots for St. Joseph Bay Aquatic Preserve
By Year



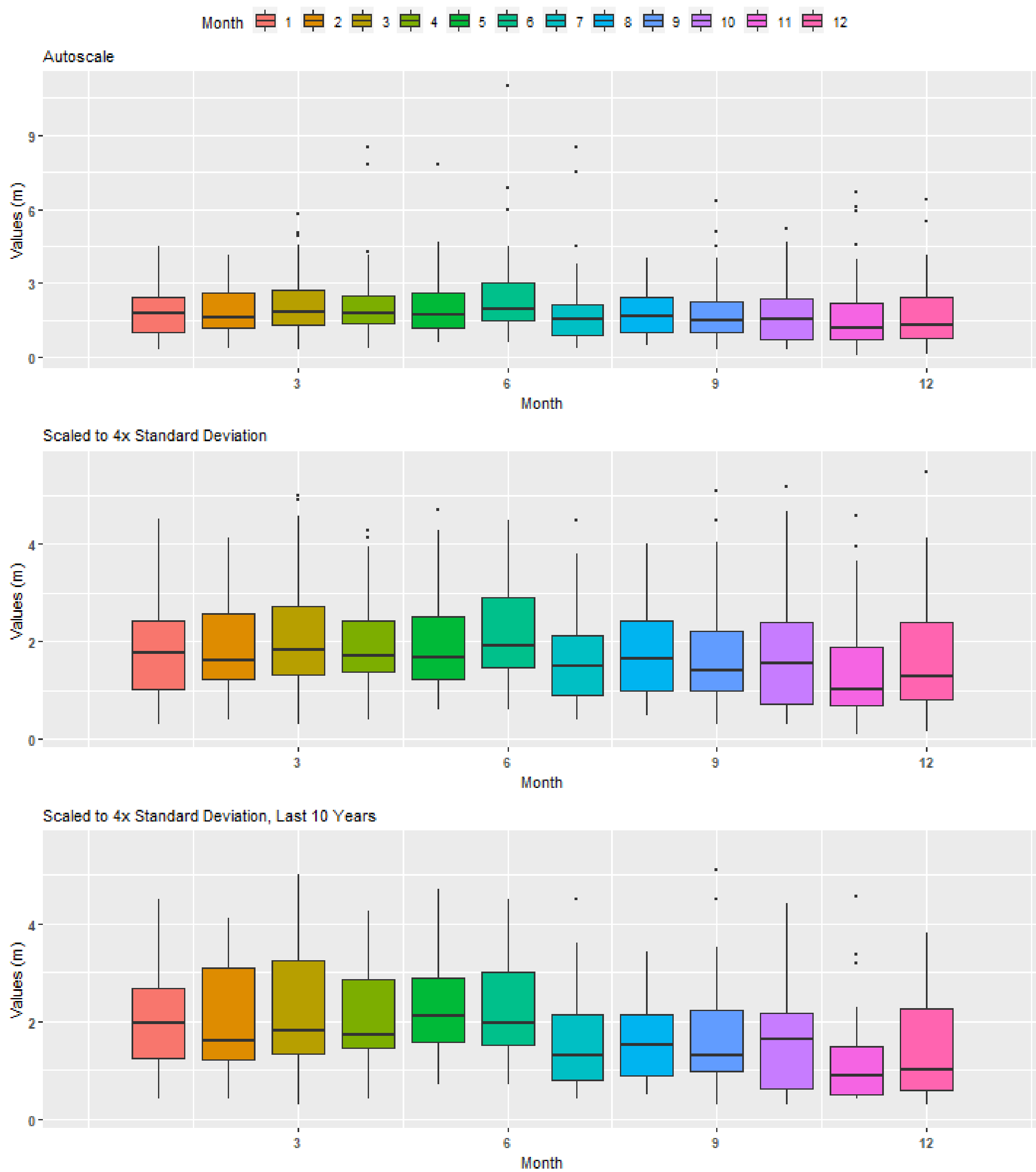
Summary Box Plots for St. Joseph Bay Aquatic Preserve

By Year & Month



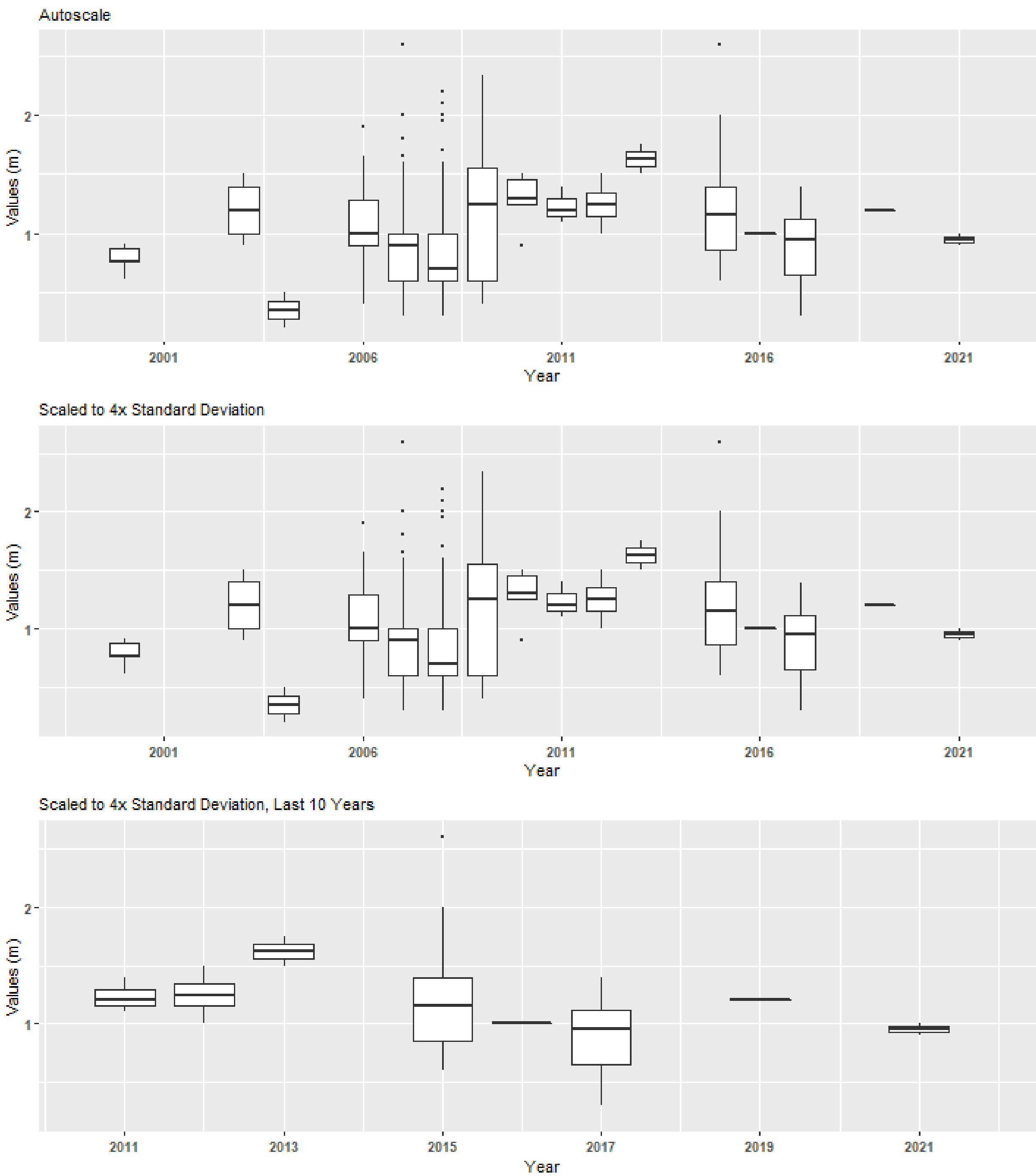
Summary Box Plots for St. Joseph Bay Aquatic Preserve

By Month

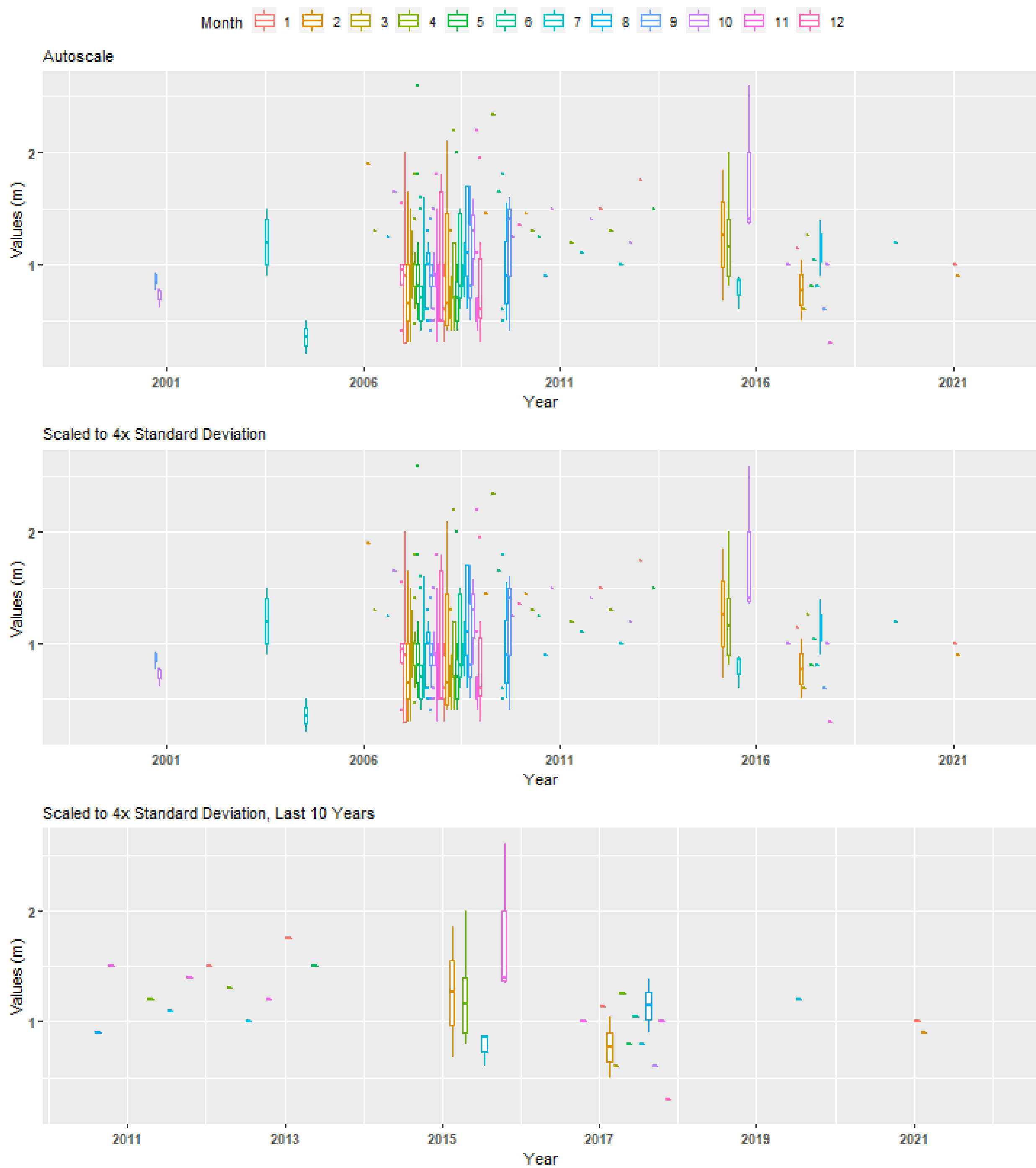


Summary Box Plots for St. Martins Marsh Aquatic Preserve

By Year

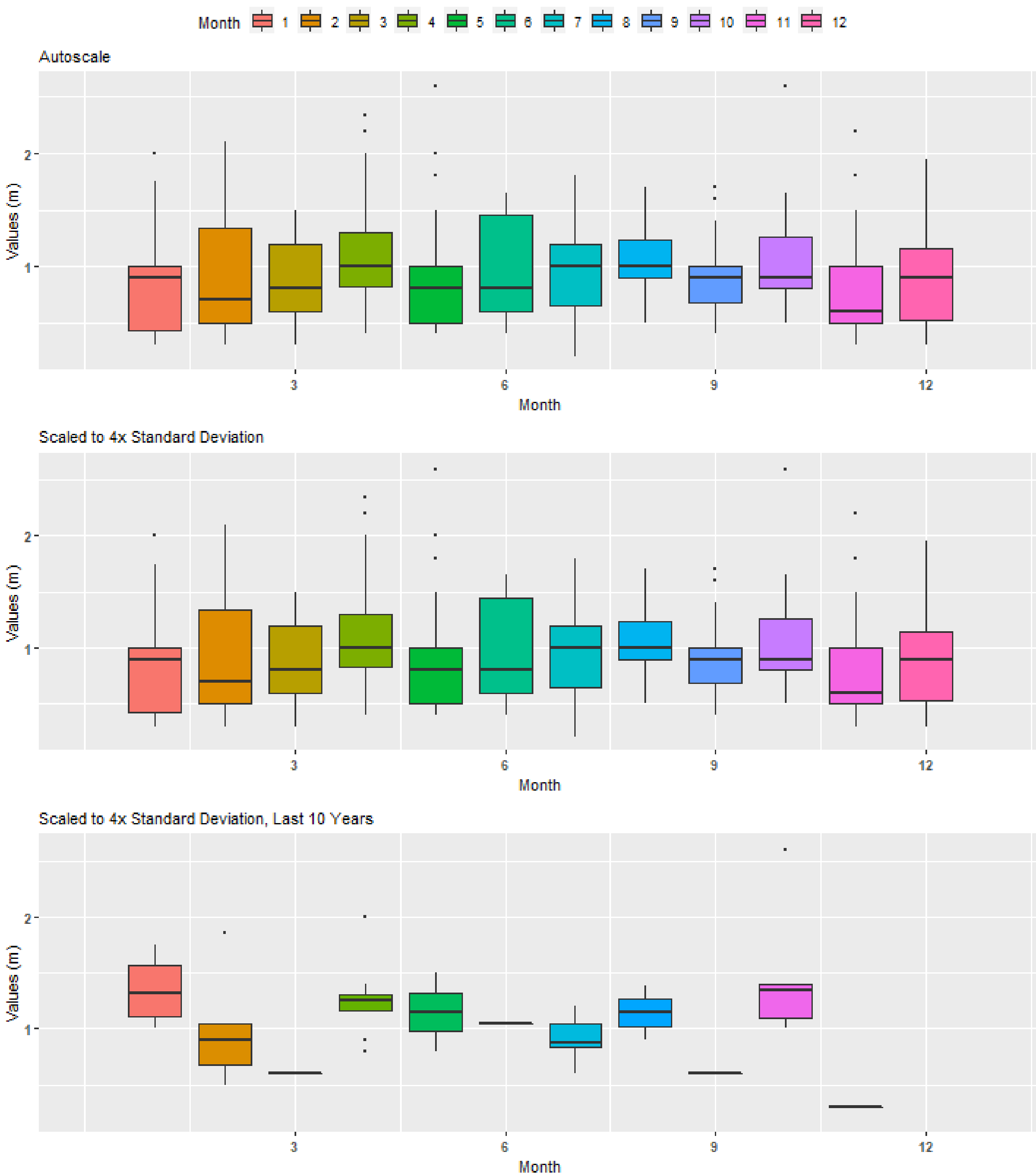


Summary Box Plots for St. Martins Marsh Aquatic Preserve
By Year & Month



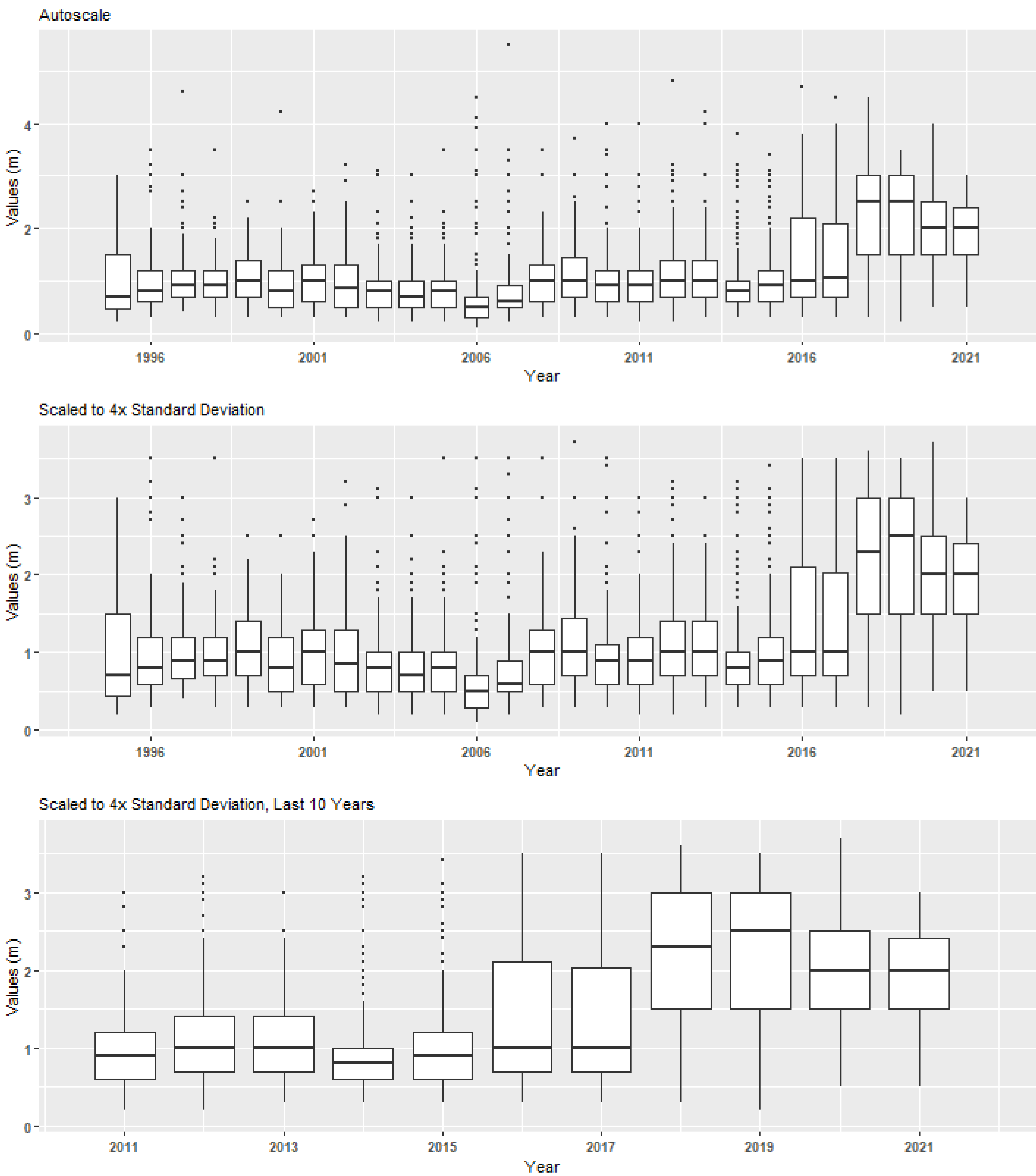
Summary Box Plots for St. Martins Marsh Aquatic Preserve

By Month



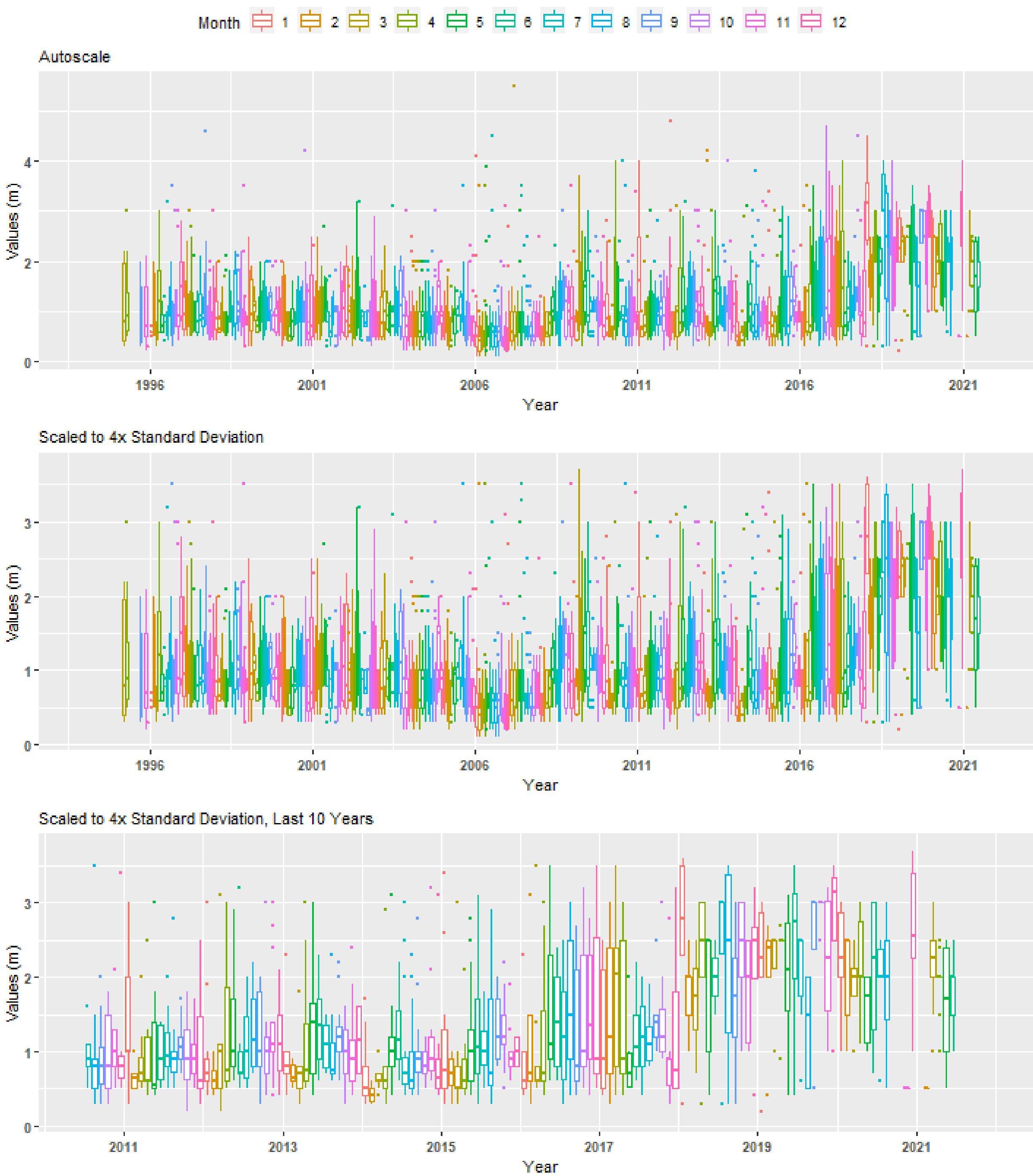
Summary Box Plots for Terra Ceia Aquatic Preserve

By Year



Summary Box Plots for Terra Ceia Aquatic Preserve

By Year & Month



Summary Box Plots for Terra Ceia Aquatic Preserve
By Month