SEACAR Discrete Water Quality Analysis: Surface Water Temperature

Last compiled on 04 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 6

Seasonal Kendall Tau Analysis 7

Appendix I: Scatter Plot of Entire Dataset 9

Appendix II: Dataset Summary Box Plots 12

Appendix III: Excluded Managed Areas 18

Appendix IV: Managed Area Trendlines 19

Appendix V: Managed Area Summary Box Plots 43

Purpose

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

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

https://github.com/FloridaSEACAR/SEACAR_Panzik

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

Adjustable Inputs

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

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

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

This includes: the raw data directory (`in_dir`), output file directory (`out_dit`), file prefix (`file_pref`), date the files were created from the database (`file_date`), the name of the parameter of interest (`param_name`), and 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-01"
param_name <- "Water_Temperature"
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.

```
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{
   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: 602666, Number Passed Filter: 602503
## I Codes: 0 (0%)
## Q Codes: 0 (0%)
## 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),
             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$Use_In_Analysis == TRUE, ] %>%
   group_by(ManagedAreaName, Year) %>%
   summarize(N = length(ResultValue),
             Min = min(ResultValue),
             Max = max(ResultValue),
             Median = median(ResultValue),
             Mean = mean(ResultValue),
             StandardDeviation = sd(ResultValue),
             ProgramIDs = paste(sort(unique(ProgramID), decreasing = FALSE),
                                collapse = ', '))
MA_Y_Stats <- as.data.table(MA_Y_Stats[order(MA_Y_Stats$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$Use_In_Analysis == TRUE, ] %>%
   group_by(ManagedAreaName, Month) %>%
   summarize(N = length(ResultValue),
             Min = min(ResultValue),
             Max = max(ResultValue),
             Median = median(ResultValue),
             Mean = mean(ResultValue),
             StandardDeviation = sd(ResultValue),
             ProgramIDs = paste(sort(unique(ProgramID), decreasing = FALSE),
                                collapse = ', '))
MA_M_Stats <- as.data.table(MA_M_Stats[order(MA_M_Stats$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))

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 that performed at The Water Atlas: https://sarasota.wateratlas.usf.edu/water-quality-trends/#analysis-overview

The following steps are performed:

1. Define the trend function.
2. Check to see if there are any groups to run analsis 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,
6. For each group, a temporary variable is created to run the `kendallSeasonalTrendTest` function using the `Year` values for year, and `Month` as the seasonal qualifier, and Trend.

- An independent.obs value of `TRUE` indicates that the data should be treated as not being serially auto-correlated. An independent.obs value of `FALSE` indicates that it is treated as being serially auto-correlated, but also requires one observation per season per year for the full time of observation.
- tau, Senn Slope (SennSlope), Senn Intercept (SennIntercept), and p are extracted from the model results.

7. The two stats tables are merged based on similar groups, and then Trend is determined from the user-defined function.

8. Write summary stats to a pipe-delimited .txt file in the output directory

    - Click this text to open Git directory with output files

```
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)
}

if(n==0){
   KT.Stats <- data.frame(matrix(ncol=14, nrow=0))
   colnames(KT.Stats) <- c("ManagedAreaName", "EarliestSampleDate",
                           "LastSampleDate", "N", "Min", "Max", "Median",
                           "Mean", "StandardDeviation", "tau", "SennSlope",
                           "SennIntercept", "p", "Trend")
   fwrite(KT.Stats, paste0(out_dir,"/", param_name, "_", file_date, "-", depth,
                           "-KendallTau_Stats.txt"), sep = "|")
} else {
KT.Stats <- data[data$Use_In_Analysis == TRUE, ] %>%
      group_by(ManagedAreaName) %>%
      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))
```

```
temp.stat <- data[data$Use_In_Analysis == TRUE, ] %>%
   group_by(ManagedAreaName) %>%
   summarize(z=kendallSeasonalTrendTest(y=ResultValue, season=Month, year=Year,
                                        independent.obs=TRUE)[c("estimate",
                                                               "p.value")]) %>%
   unnest_wider(z)

colnames(temp.stat)[6] <- "p"
temp.stat <- temp.stat %>% group_by(ManagedAreaName) %>%
   summarize(tau = unique(na.omit(tau)), SennSlope = unique(na.omit(slope)),
             SennIntercept = unique(na.omit(intercept)), p = unique(na.omit(p)))

KT.Stats <- merge(KT.Stats, temp.stat, by="ManagedAreaName")
rm(temp.stat)
KT.Stats <- KT.Stats %>% group_by(ManagedAreaName) %>%
   mutate(Trend = trend_calculator(slope=SennSlope, median_value=Median, p=p))
KT.Stats$tau <- round(KT.Stats$tau, digits=4)
KT.Stats$p <- round(KT.Stats$p, digits=4)

fwrite(KT.Stats, paste0(out_dir,"/", param_name, "_", file_date, "-", depth,
                        "-KendallTau_Stats.txt"), sep = "|")
}
```

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

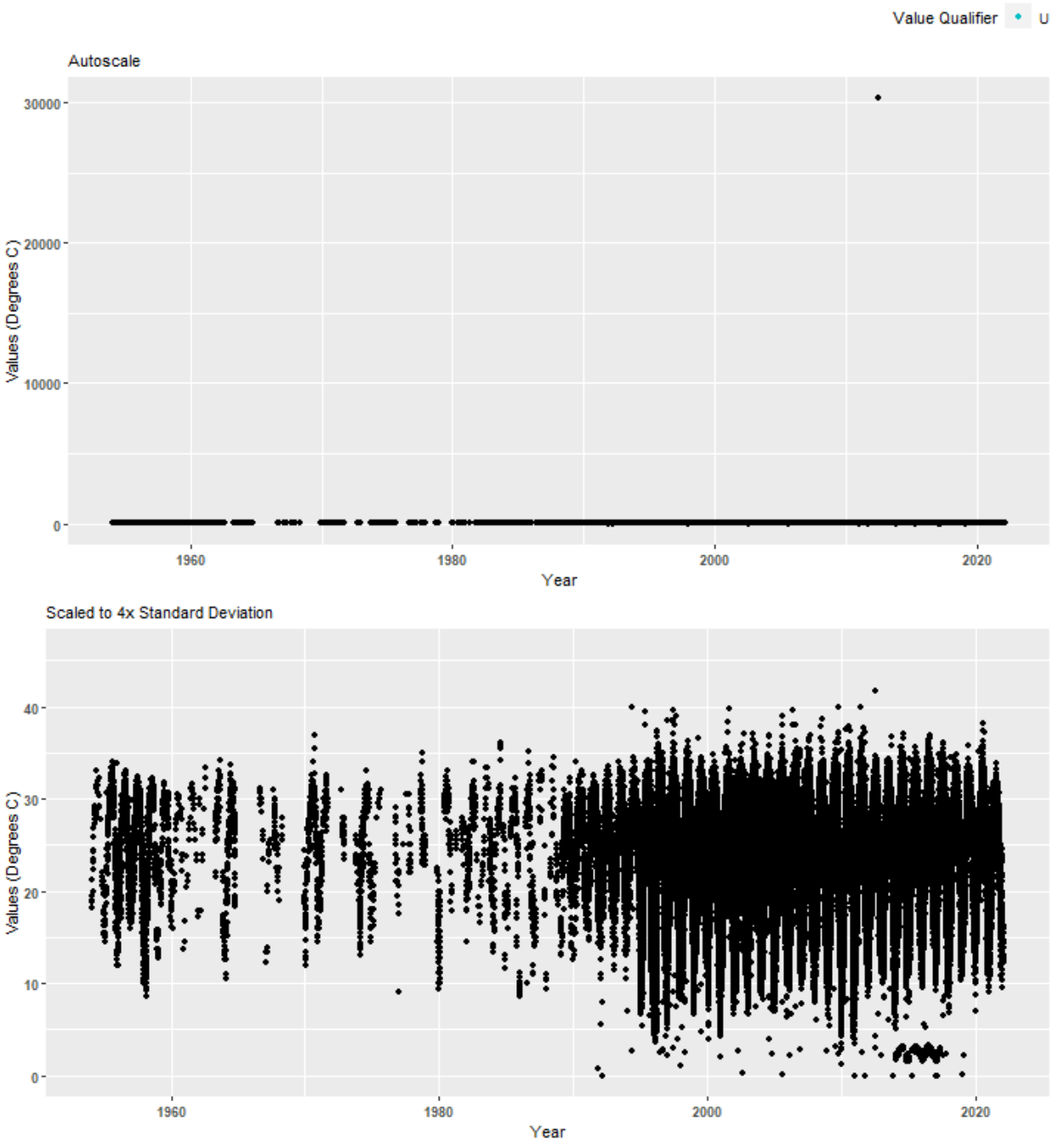
Appendix I: Scatter Plot of Entire Dataset

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 {
      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 {
      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"))
```

```
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.

```
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())
```

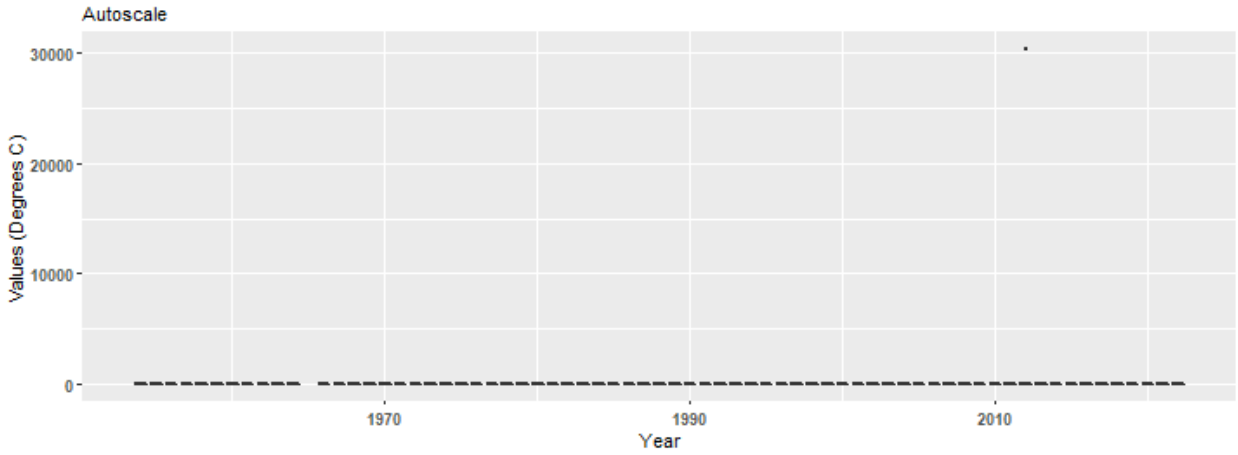
```
YMset <- ggarrange(p0, set, ncol=1, heights = c(0.07, 1))
```

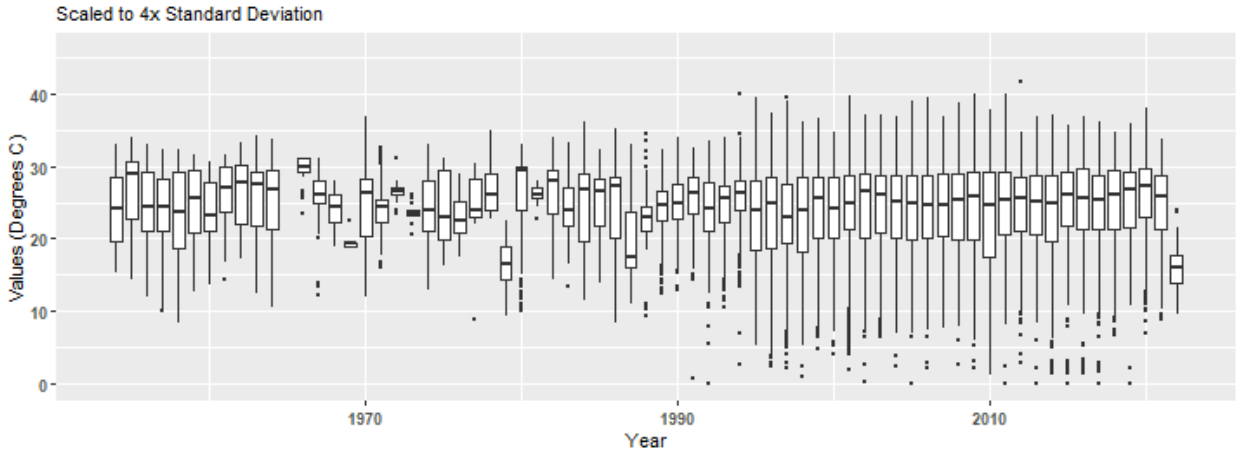
The following box plots are grouped by month with fill color being related to the month. This is designed to view potential seasonal trends.

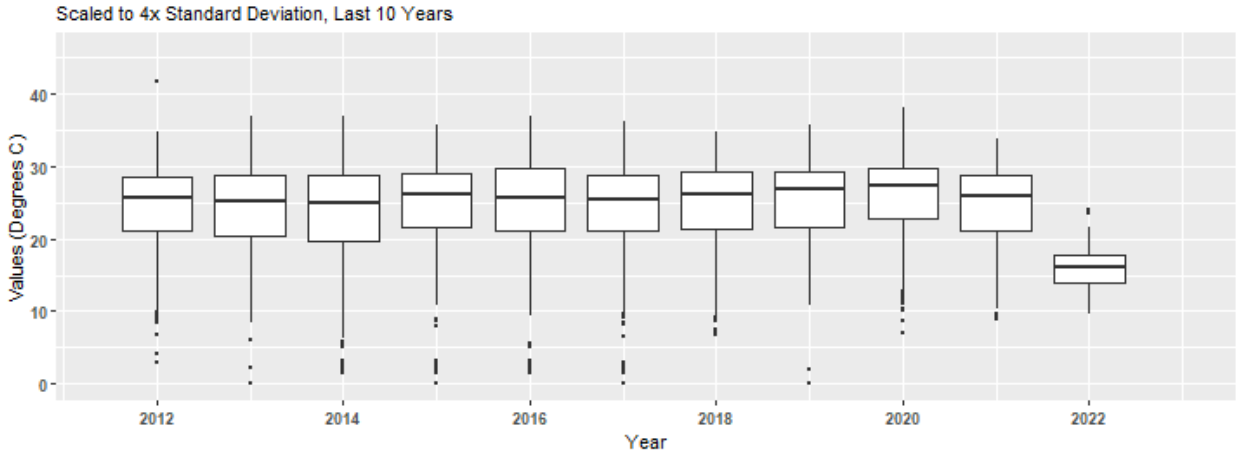
```
p1 <- ggplot(data = data[data$Include == TRUE, ],
             aes(x = Month, y = ResultValue,
                 group = Month, fill = as.factor(Month))) +
   geom_boxplot(outlier.size = 0.5) +
   labs(subtitle = "Autoscale", x = "Month",
        y = paste0("Values (", unit, ")"), fill="Month") +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

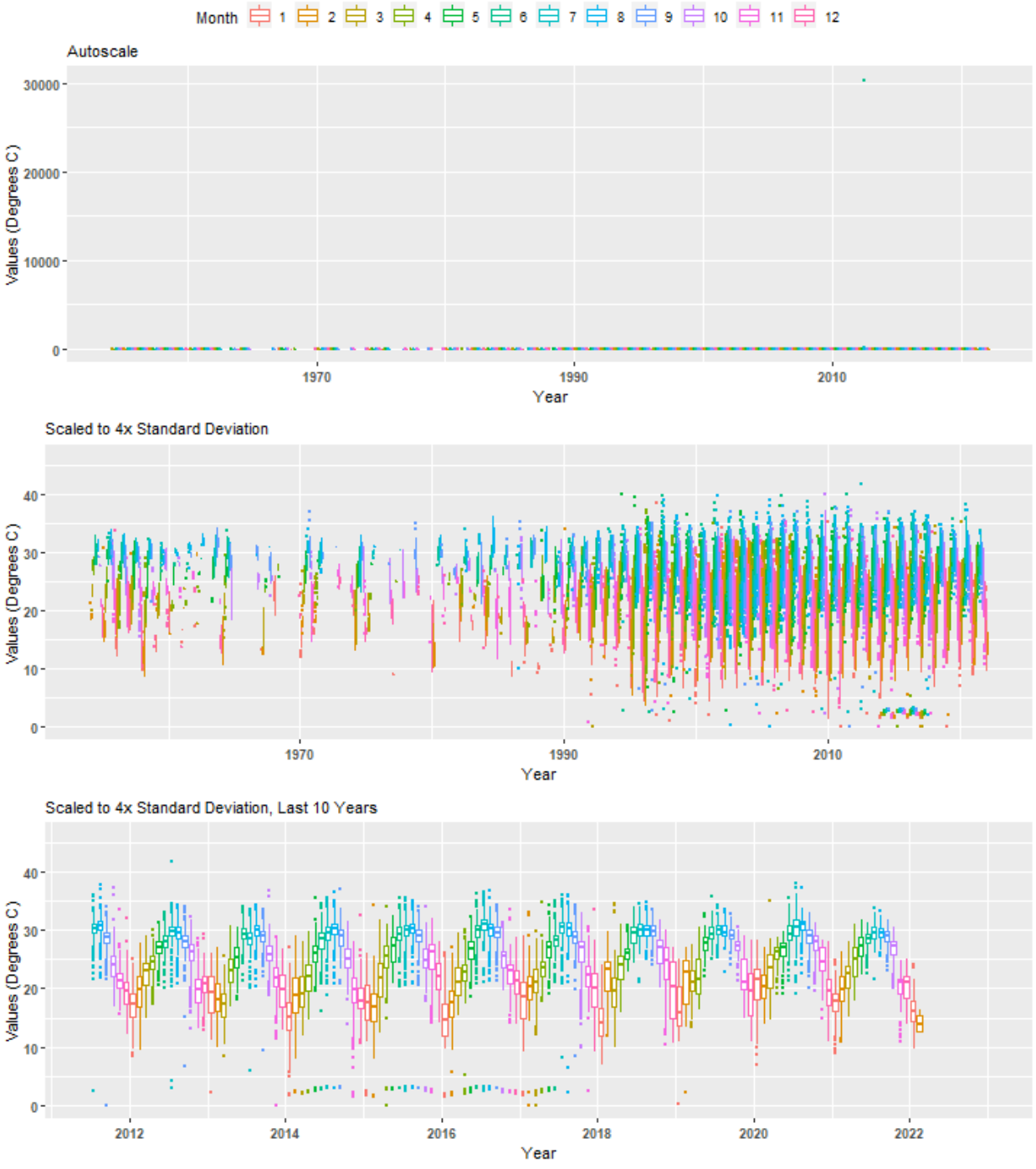






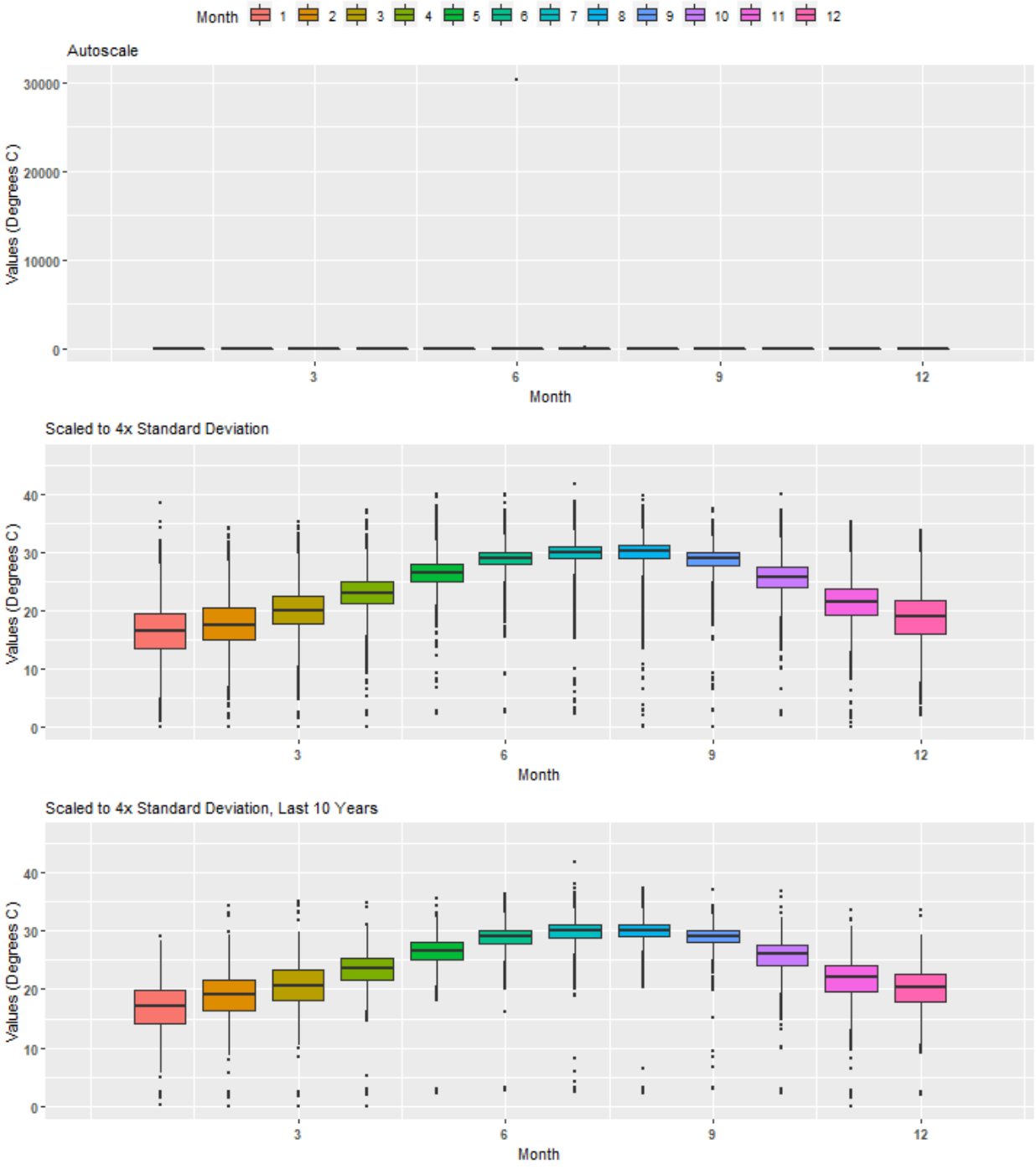
Summary Box Plots for Entire Data

By Year & Month



Summary Box Plots for Entire Data

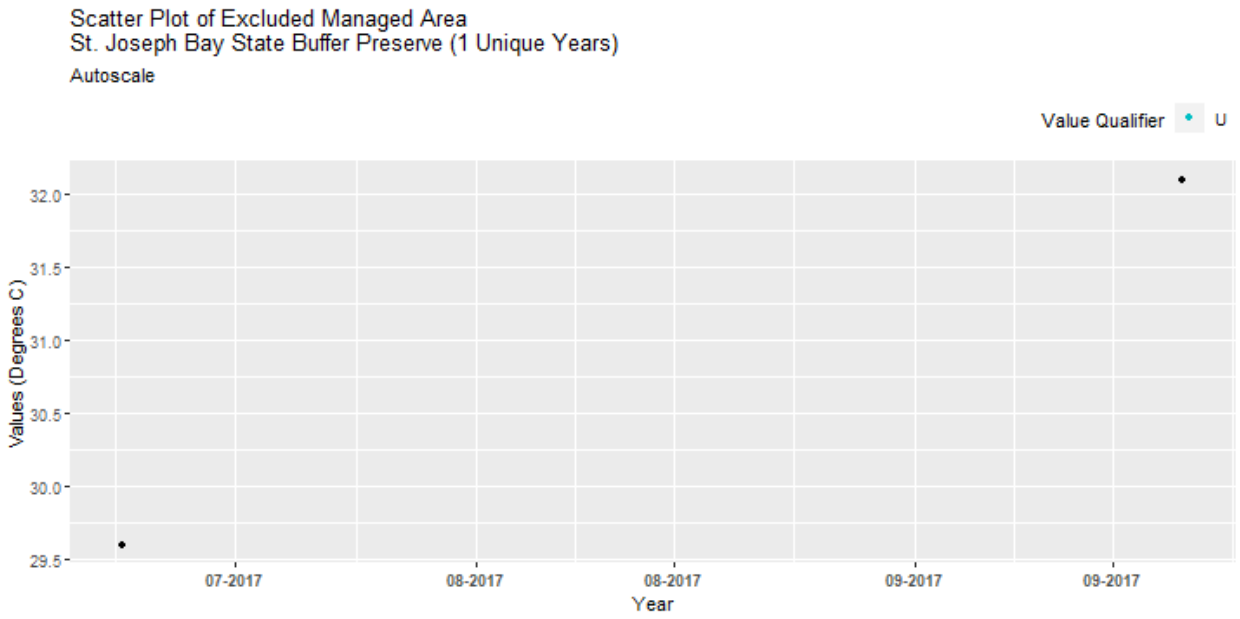
By Month



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 {
            scale_color_manual(values = c("U"= "#00BFC4"), na.value="black")
         }}
      print(p1)
   }
}
```



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

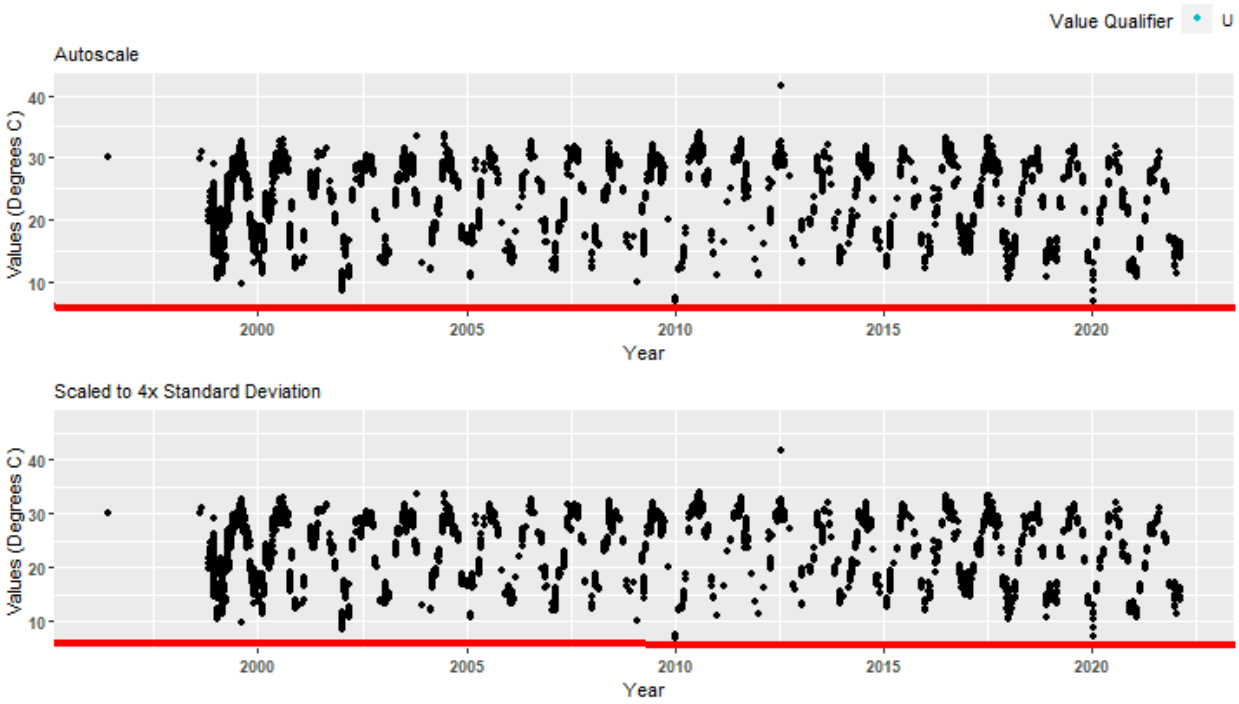
```
if(n==0){
   print("There are no managed areas that qualify.")
} else {
   for (i in 1:n) {
      year_lower <- min(data$Year[data$Use_In_Analysis == TRUE &
                                  data$ManagedAreaName ==
                                  MA_names[i]])
      year_upper <- max(data$Year[data$Use_In_Analysis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

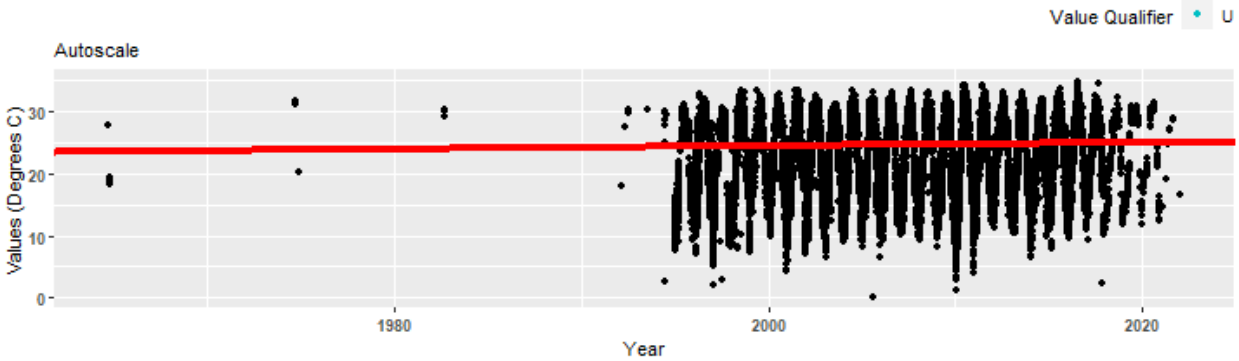
Data Points with Trendlines for Alligator Harbor Aquatic Preserve

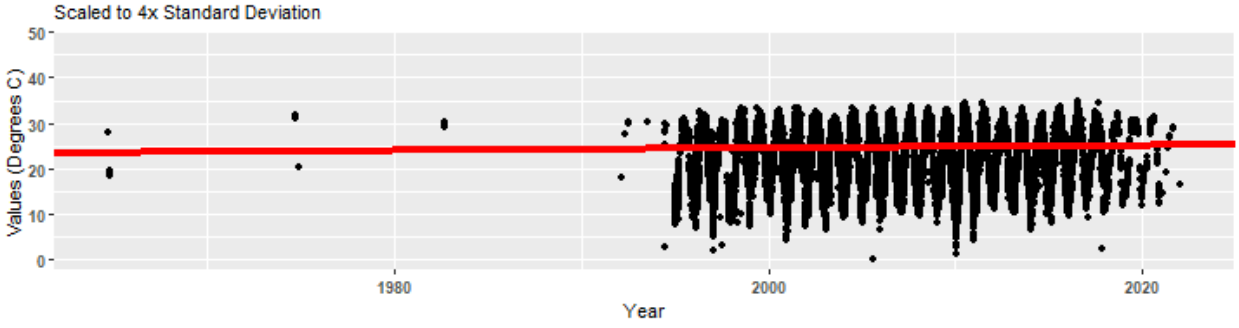
Senn Slope = -0.00909090909090922, Senn Intercept = 24.1833333333333
Trend = -1, tau = -0.0348, p = 0.019
Linear Trendline: y = 0.0765766330279603x + -129.639535875743



Data Points with Trendlines for Apalachicola Bay Aquatic Preserve

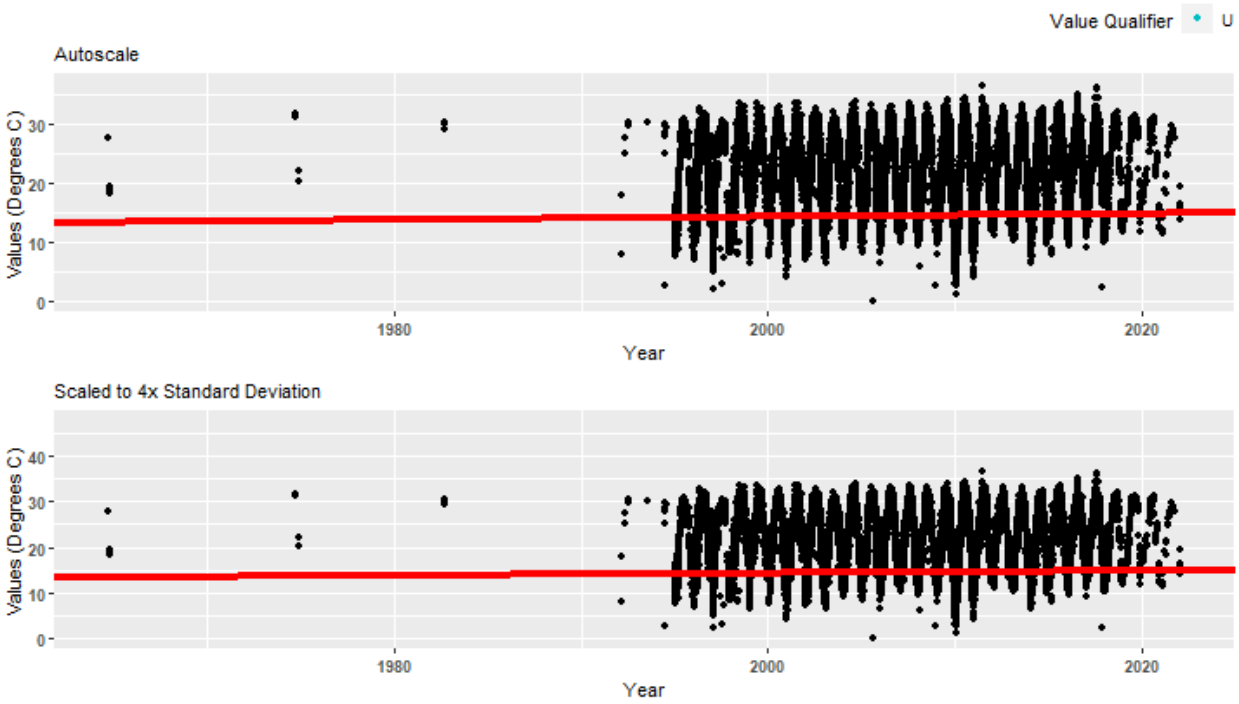
Senn Slope = 0.027272727272727, Senn Intercept = -29.9450000000001
Trend = 1, tau = 0.0581, p = 0
Linear Trendline: y = 0.11240621086787x + -203.006505119501





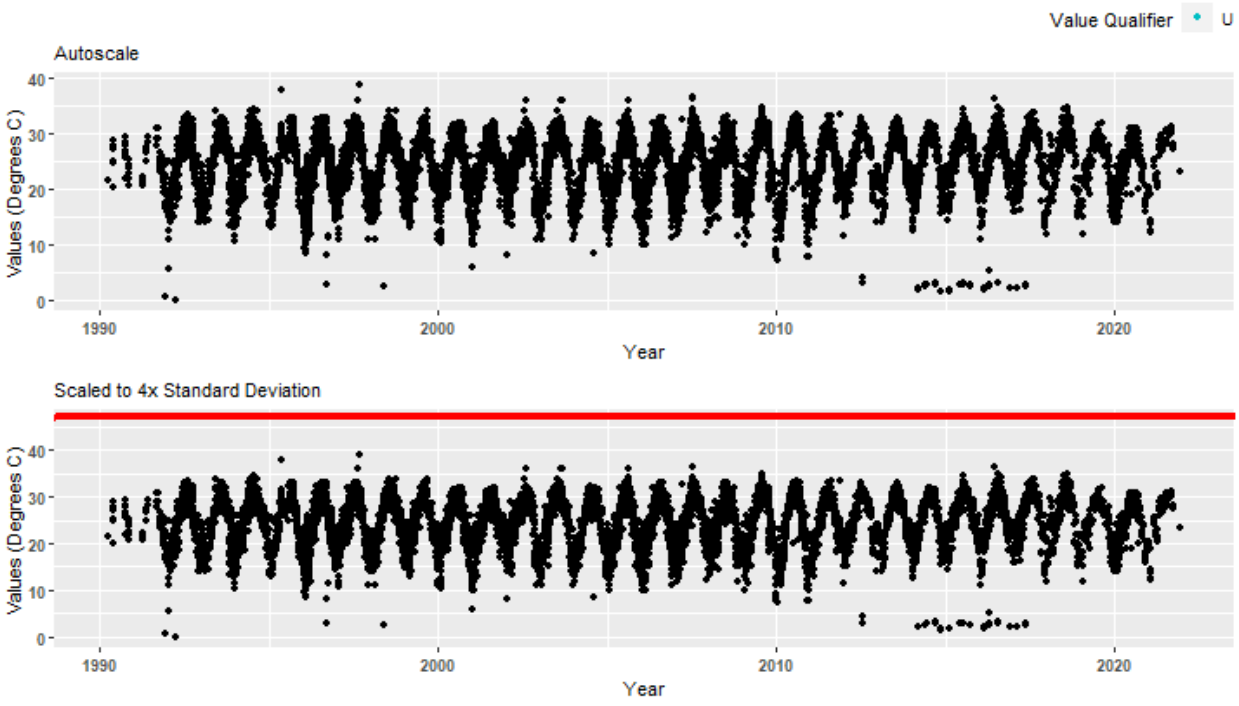
Data Points with Trendlines for Apalachicola National Estuarine Research Reserve

Senn Slope = 0.0272727272727273, Senn Intercept = -40.0260416666667
Trend = 1, tau = 0.0561, p = 0
Linear Trendline: y = 0.131684237596183x + -241.971007925226



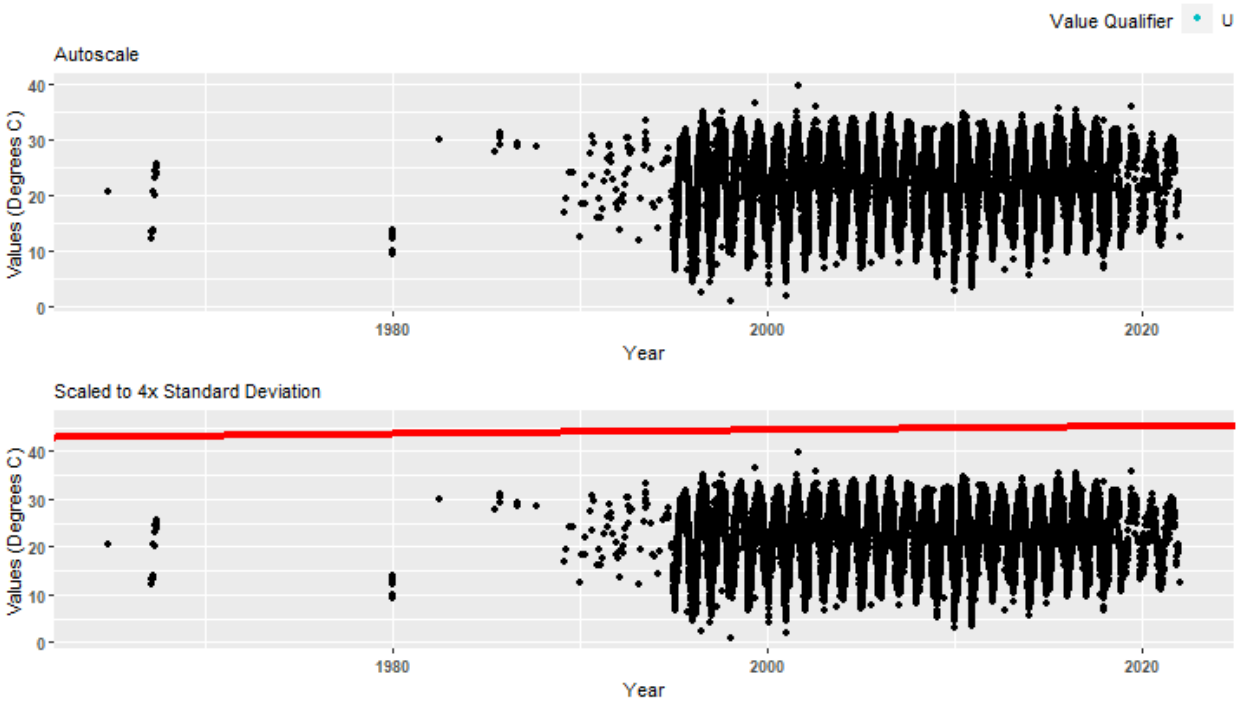
Data Points with Trendlines for Banana River Aquatic Preserve

Senn Slope = 0.0136363636363637, Senn Intercept = 20.05
Trend = 1, tau = 0.0311, p = 0
Linear Trendline: y = -0.0586968953262891x + 141.835637551065



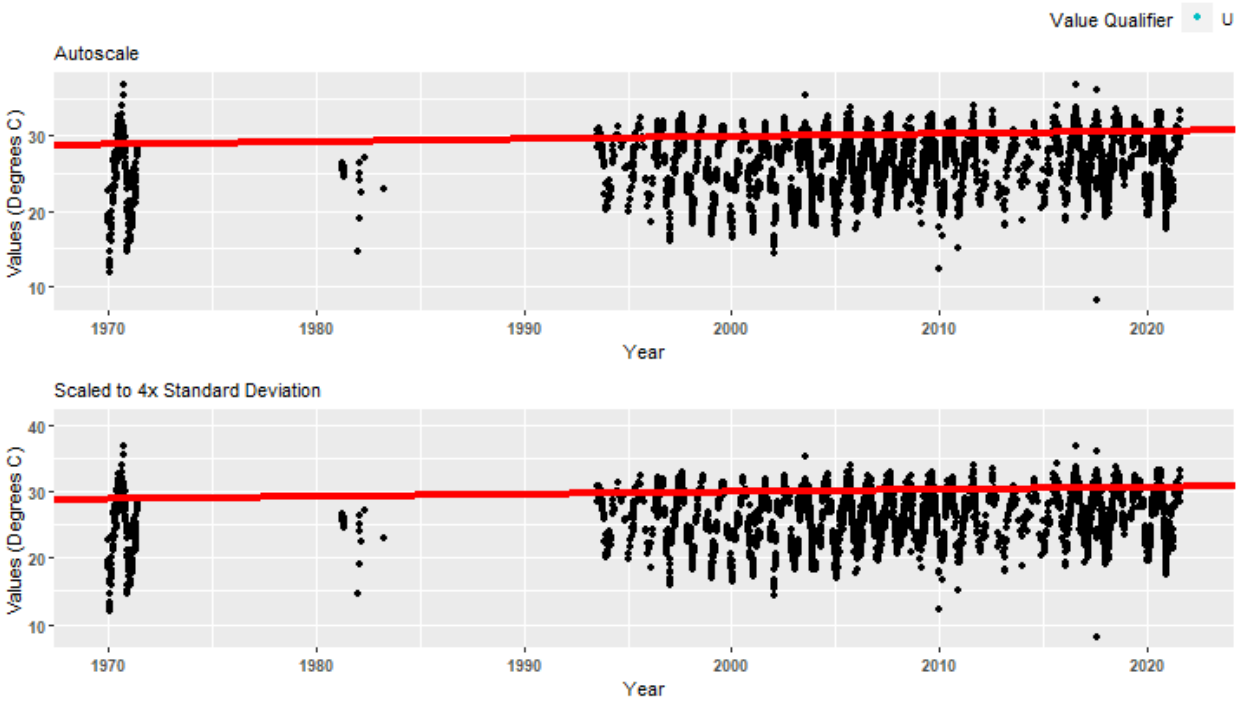
Data Points with Trendlines for Big Bend Seagrasses Aquatic Preserve

Senn Slope = 0.0399999999999999, Senn Intercept = -35.4100000000004
Trend = 1, tau = 0.0808, p = 0
Linear Trendline: y = 0.0749029467299095x + -127.286472365526



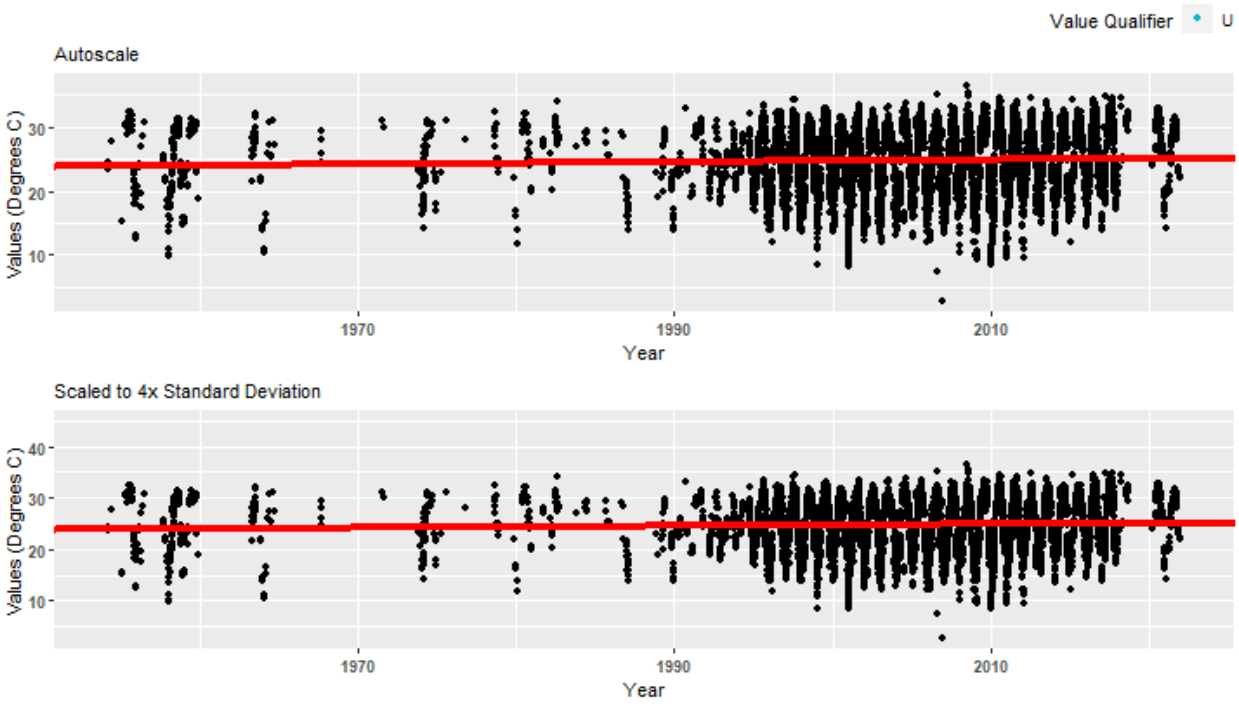
Data Points with Trendlines for Biscayne Bay Aquatic Preserve

Senn Slope = 0.0350000000000001, Senn Intercept = -40.0472500000003
Trend = 1, tau = 0.1333, p = 0
Linear Trendline: y = 0.060104441656788x + -94.0146752207948



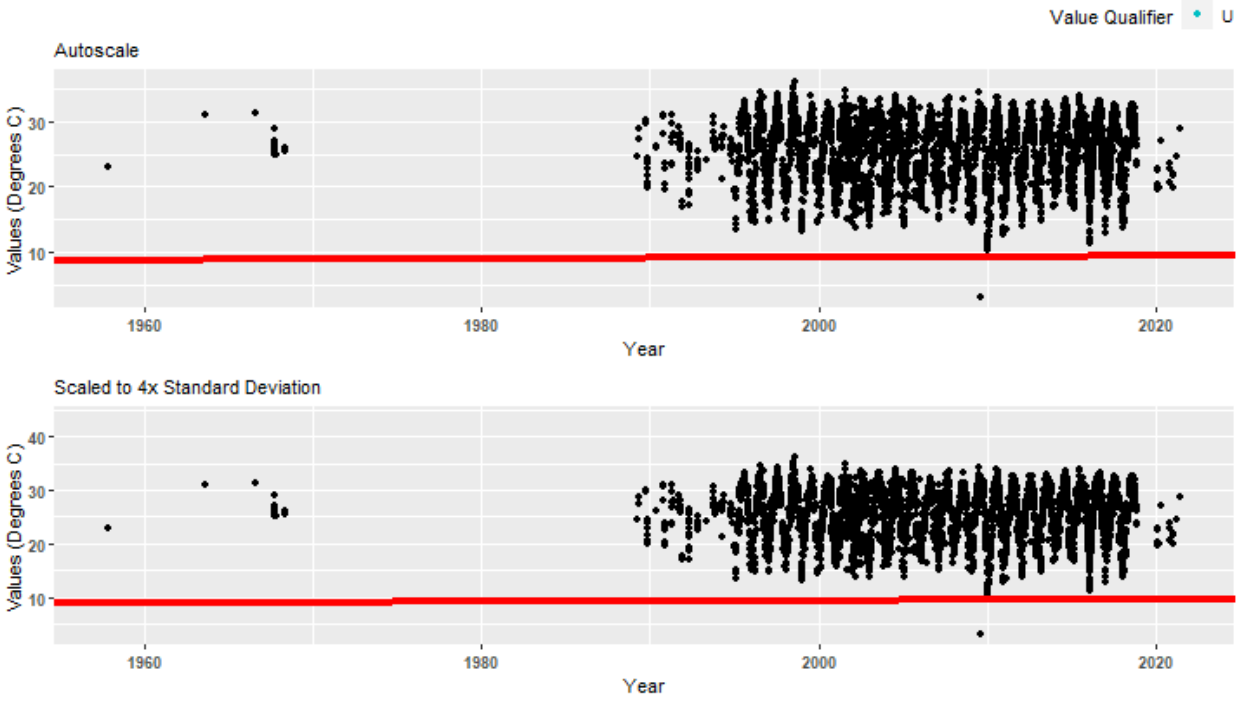
Data Points with Trendlines for Boca Ciega Bay Aquatic Preserve

Senn Slope = 0.0177777777777778, Senn Intercept = -10.7081182795695
Trend = 1, tau = 0.056, p = 0
Linear Trendline: y = 0.0165984671303678x + -8.19527163684843



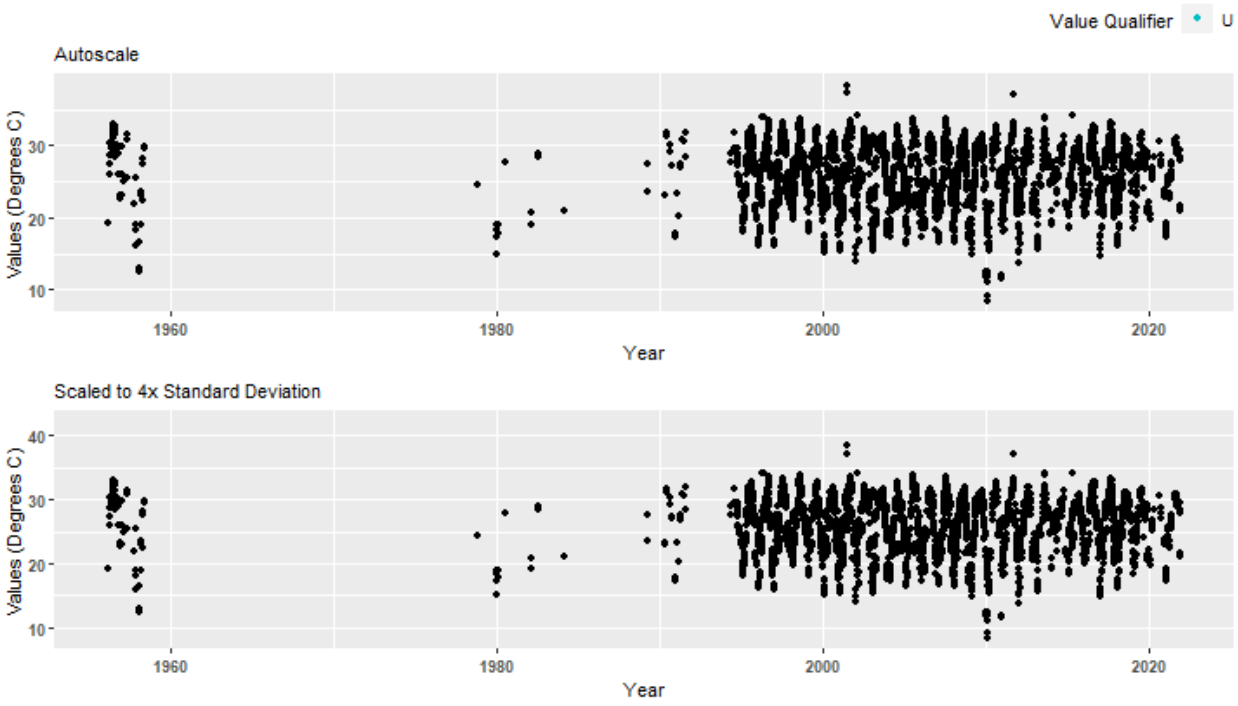
Data Points with Trendlines for Cape Haze Aquatic Preserve

Senn Slope = 0.00999999999999979, Senn Intercept = -10.5987681159418
Trend = 1, tau = 0.0249, p = 0.0026
Linear Trendline: y = 0.0373344683963208x + -49.1389321073719



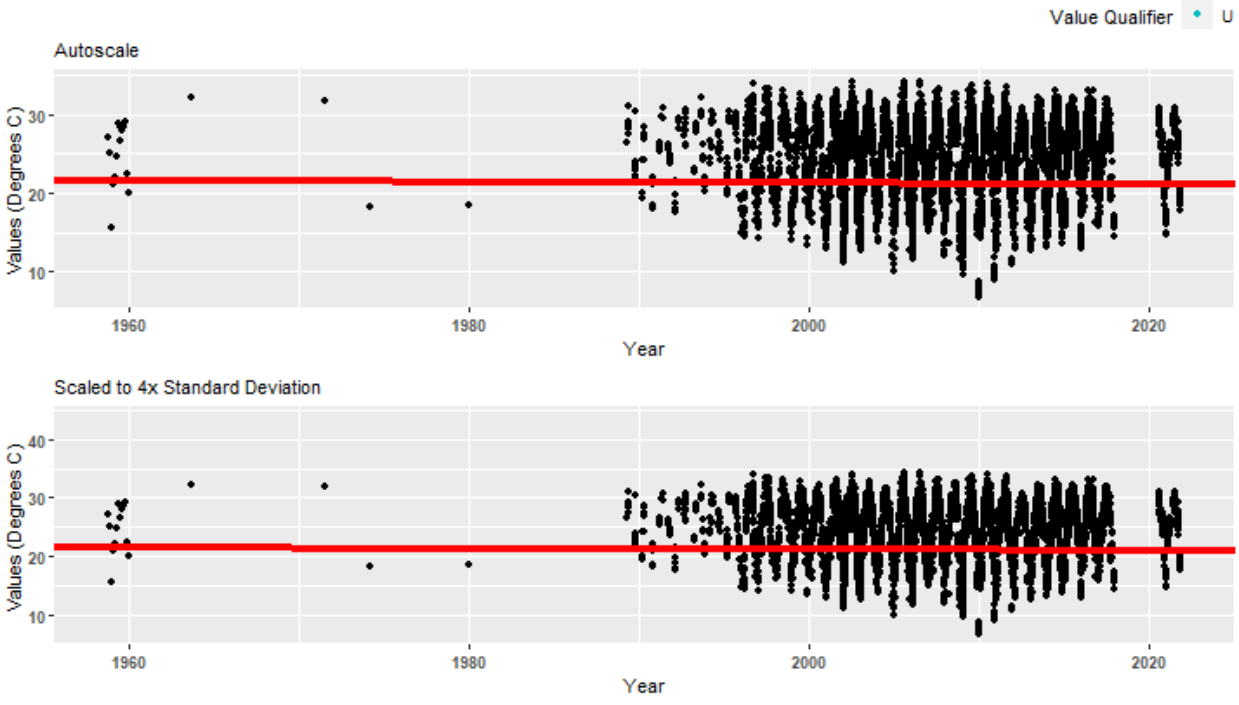
Data Points with Trendlines for Cape Romano-Ten Thousand Islands Aquatic Preserve

Senn Slope = 0.00588235294117635, Senn Intercept = 41.5281368221943
Trend = 0, tau = 0.0128, p = 0.0541
Linear Trendline: y = 0.00571265361473409x + 14.493765670841



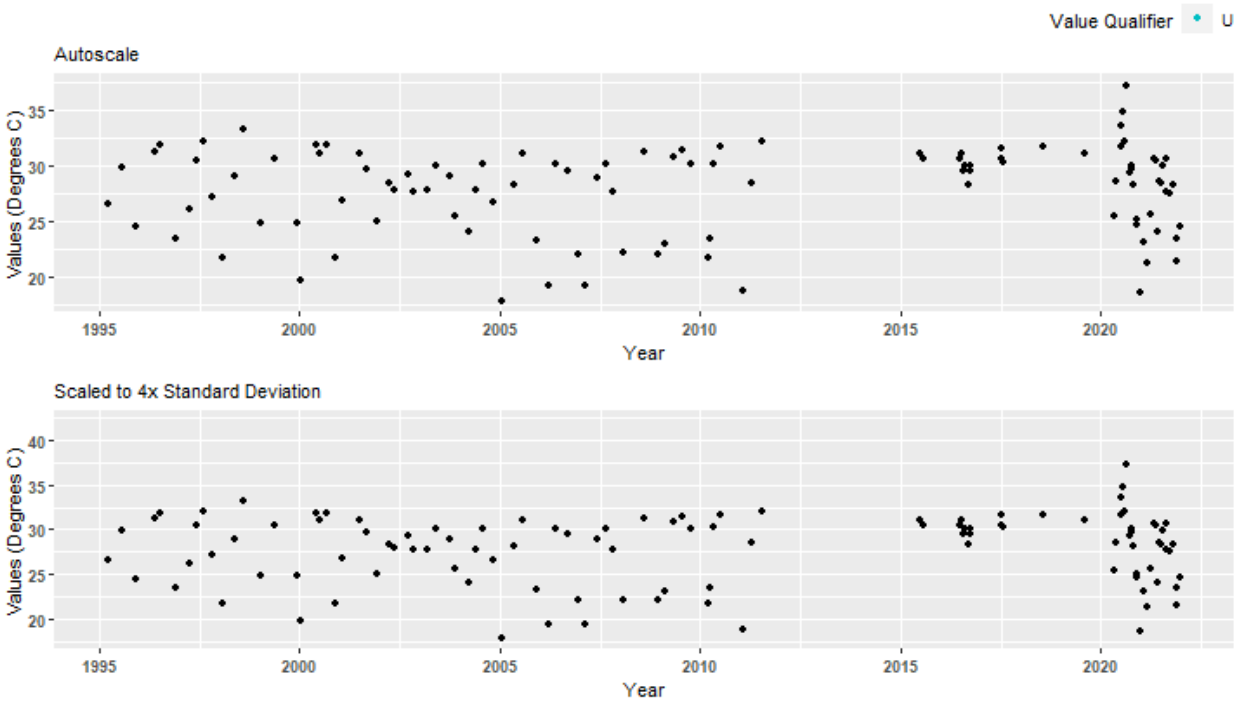
Data Points with Trendlines for Cockroach Bay Aquatic Preserve

Senn Slope = -0.00740740740740738, Senn Intercept = 36.1383333333334
Trend = -1, tau = -0.0059, p = 0.0072
Linear Trendline: y = -0.0672871482103434x + 160.251956559591



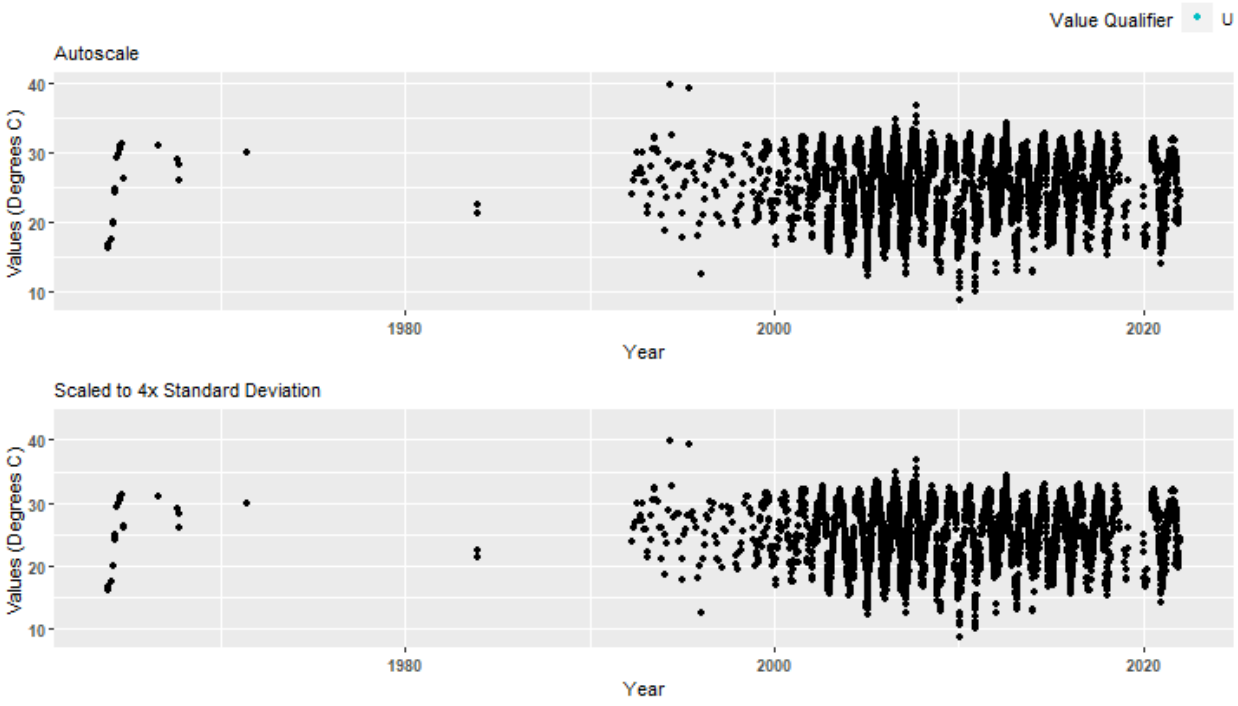
Data Points with Trendlines for Coupon Bight Aquatic Preserve

Senn Slope = -0.0389999999999979, Senn Intercept = 91.1205550480767
Trend = -1, tau = -0.1481, p = 0.0436
Linear Trendline: y = 0.0345853257526476x + -41.6814355768072



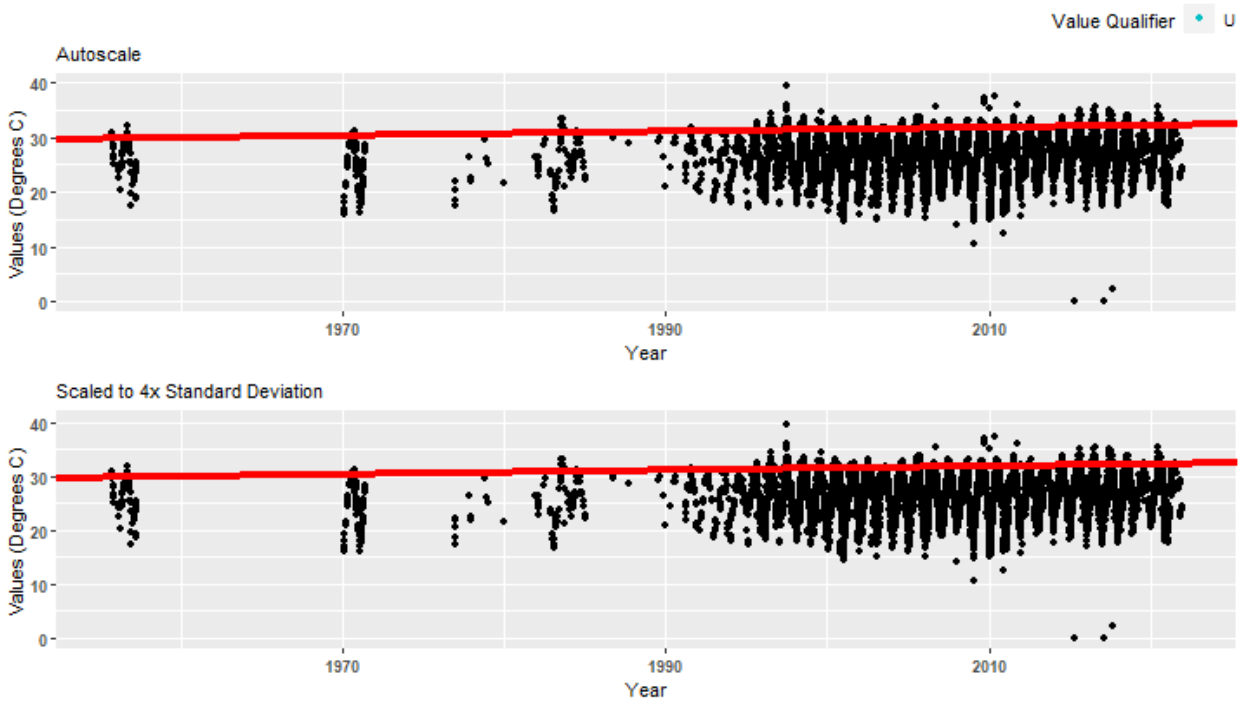
Data Points with Trendlines for Estero Bay Aquatic Preserve

Senn Slope = 0.0133333333333333, Senn Intercept = 19.5205555555557
Trend = 1, tau = 0.0298, p = 0.0004
Linear Trendline: y = 0.0169567110298942x + -8.50004653623



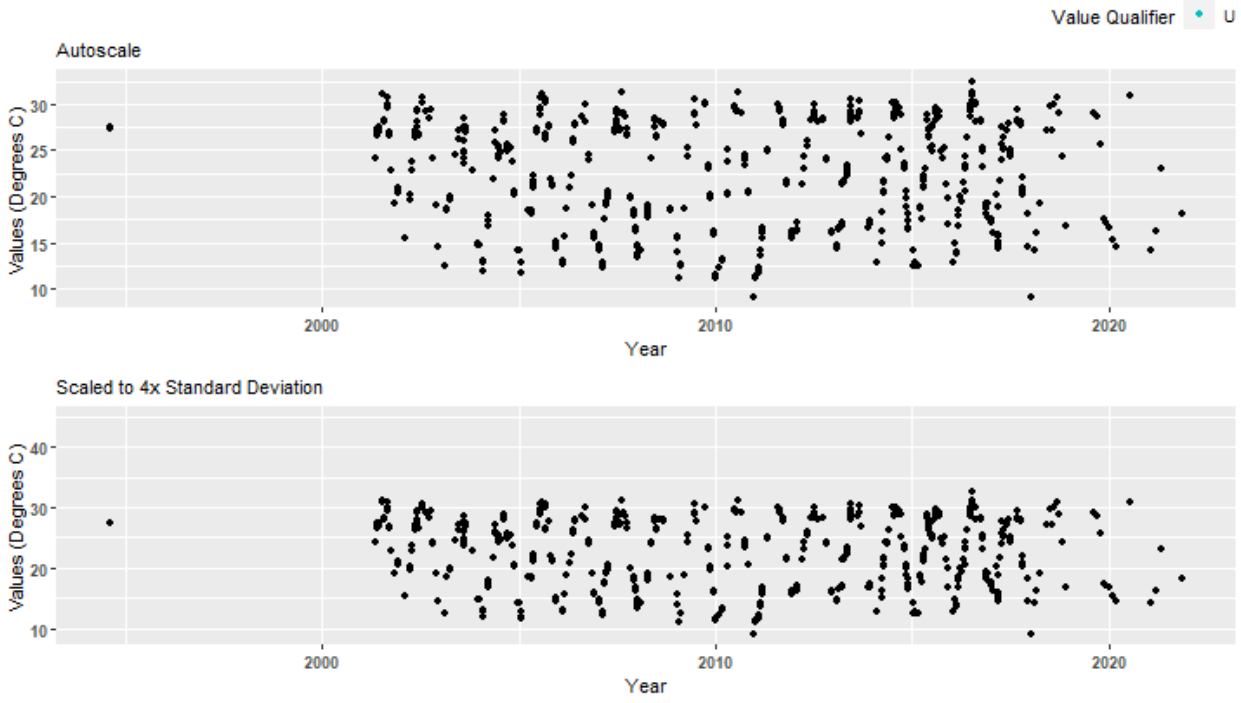
Data Points with Trendlines for Florida Keys National Marine Sanctuary

Senn Slope = 0.038290909090909, Senn Intercept = -45.0415621212116
Trend = 1, tau = 0.1323, p = 0
Linear Trendline: y = 0.0422969701159535x + -58.2187840216027



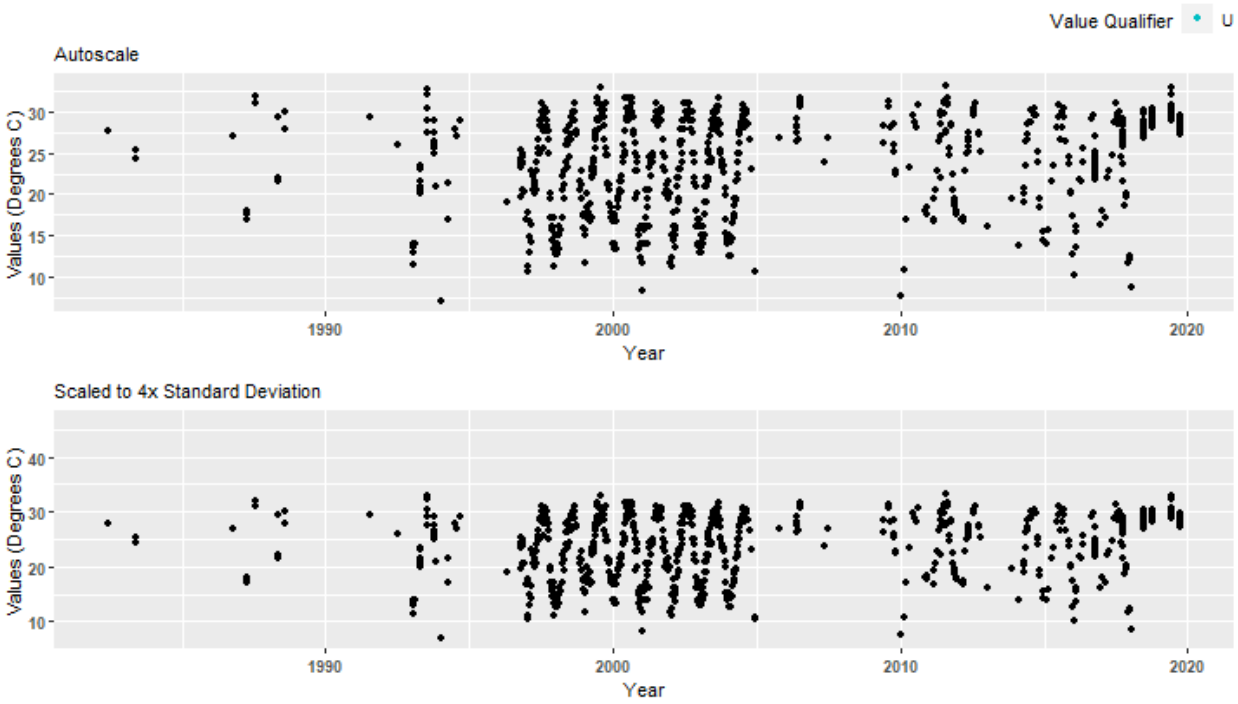
Data Points with Trendlines for Fort Clinch State Park Aquatic Preserve

Senn Slope = 0.0535897435897436, Senn Intercept = -149.925
Trend = 1, tau = 0.1011, p = 0
Linear Trendline: y = -0.0642145499336252x + 151.556782316318



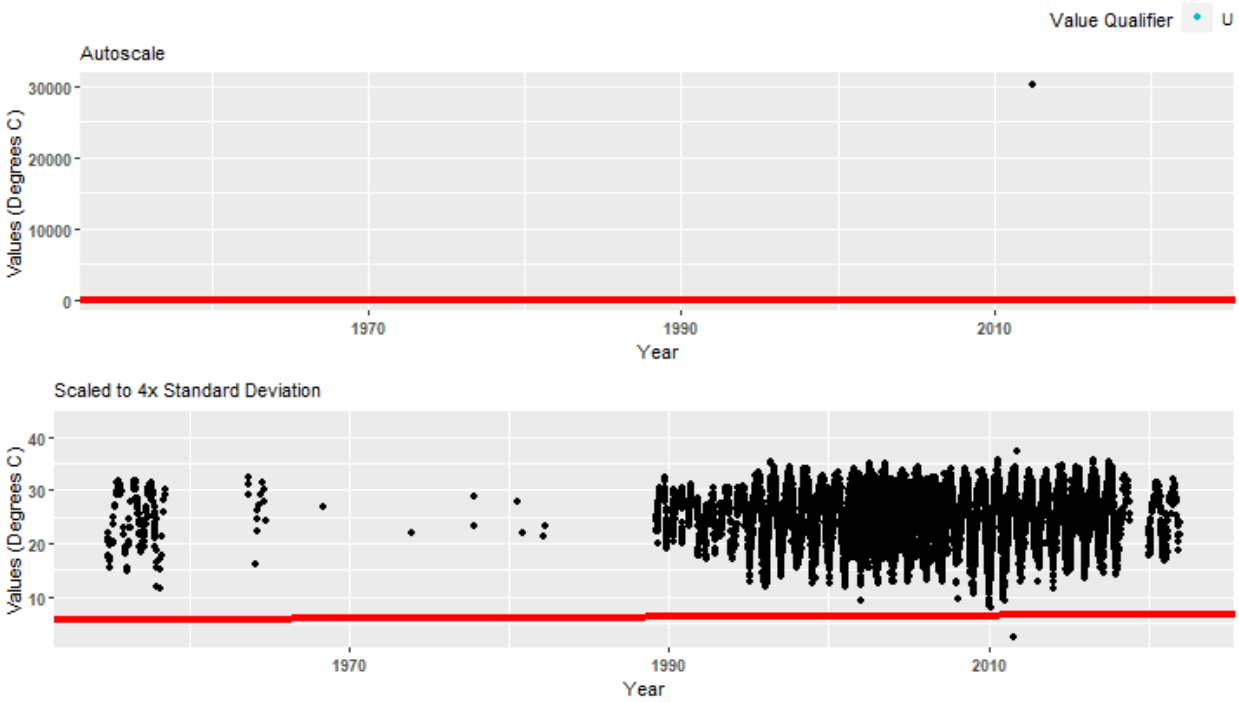
Data Points with Trendlines for Fort Pickens State Park Aquatic Preserve

Senn Slope = 0.0352173913043478, Senn Intercept = -1.94142878181821
Trend = 1, tau = 0.0763, p = 0.0002
Linear Trendline: y = 0.161256054703792x + -299.411531613316



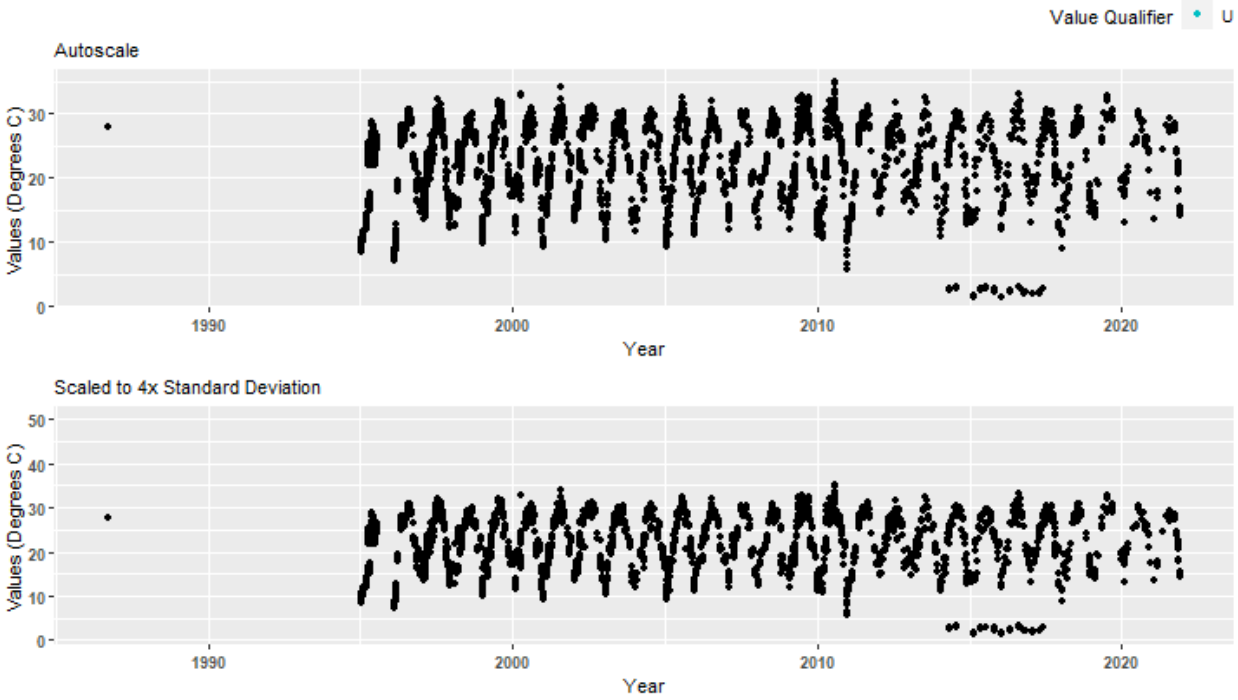
Data Points with Trendlines for Gasparilla Sound-Charlotte Harbor Aquatic Preserve

Senn Slope = 0.0146666666666666, Senn Intercept = -22.9173611111112
Trend = 1, tau = 0.0368, p = 0
Linear Trendline: y = 0.0843301640639236x + -143.111464010174



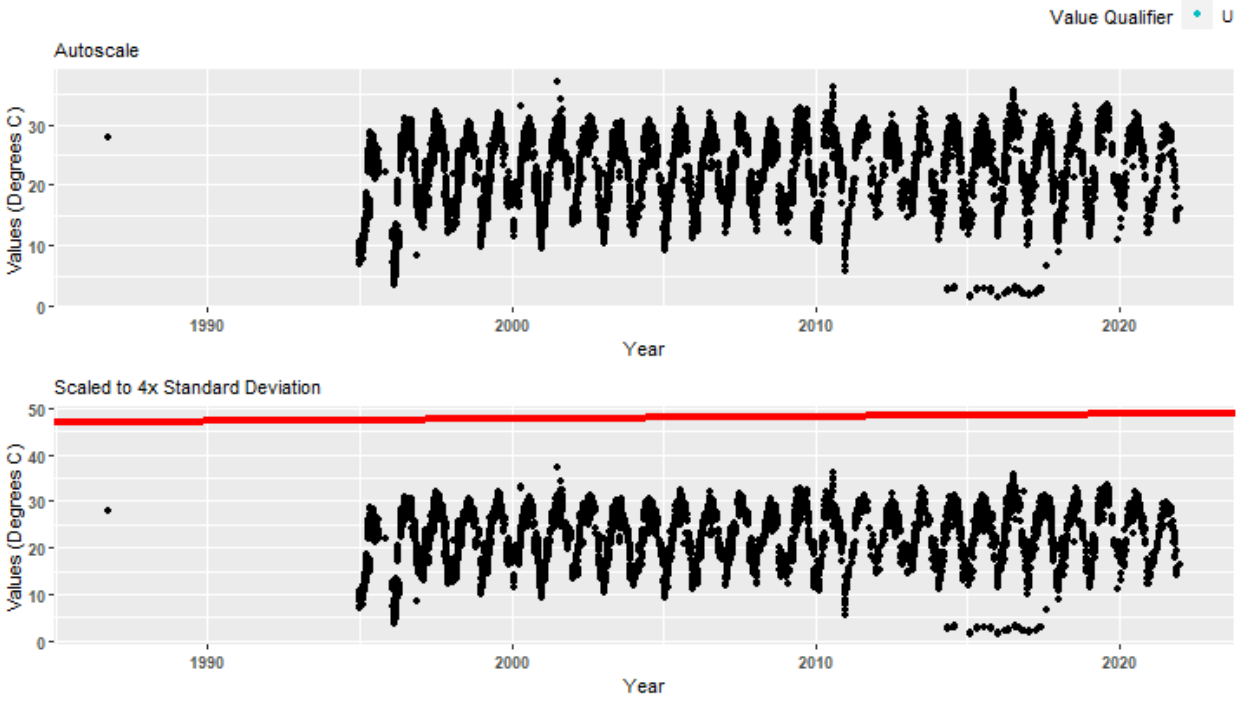
Data Points with Trendlines for Guana River Marsh Aquatic Preserve

Senn Slope = 0, Senn Intercept = 67.2794957983192
Trend = 0, tau = 0.0032, p = 0.3051
Linear Trendline: y = -0.19152004329333x + 405.500360747474



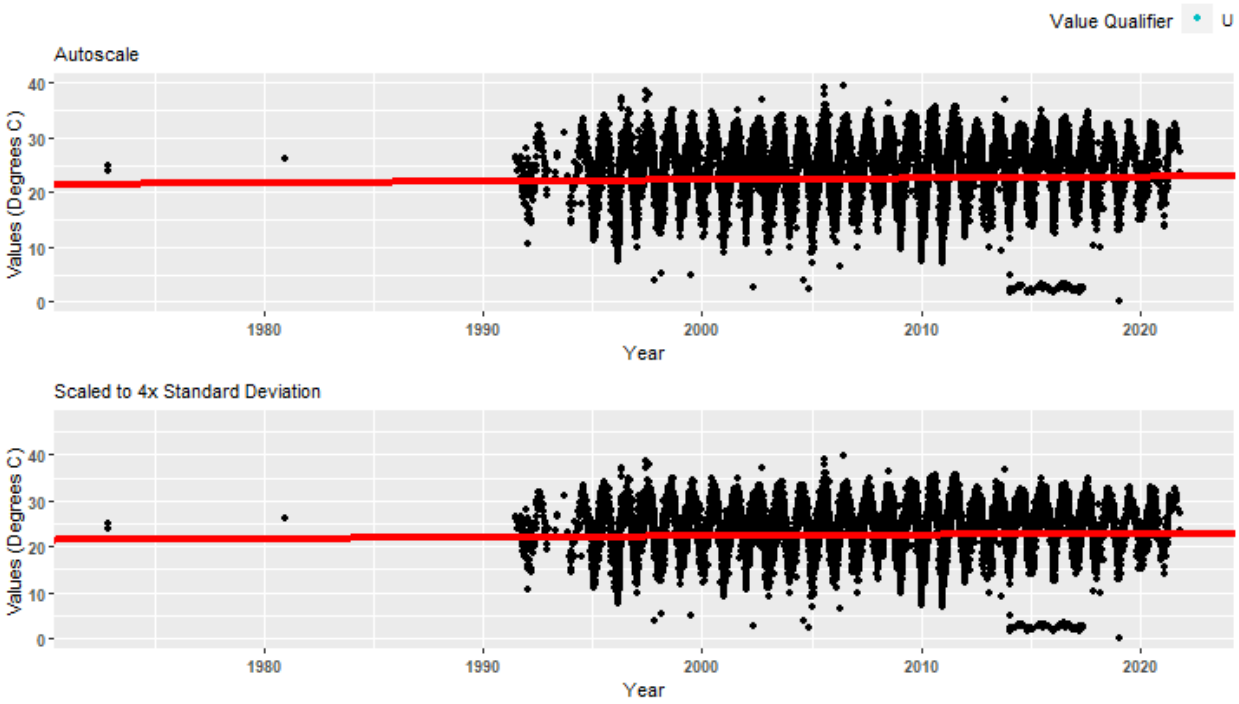
Data Points with Trendlines for Guana Tolomato Matanzas National Estuarine Research Reserve

Senn Slope = 0.0500000000000003, Senn Intercept = -52.1145454545455
Trend = 1, tau = 0.1043, p = 0
Linear Trendline: y = -0.052554150605416x + 127.297611263186



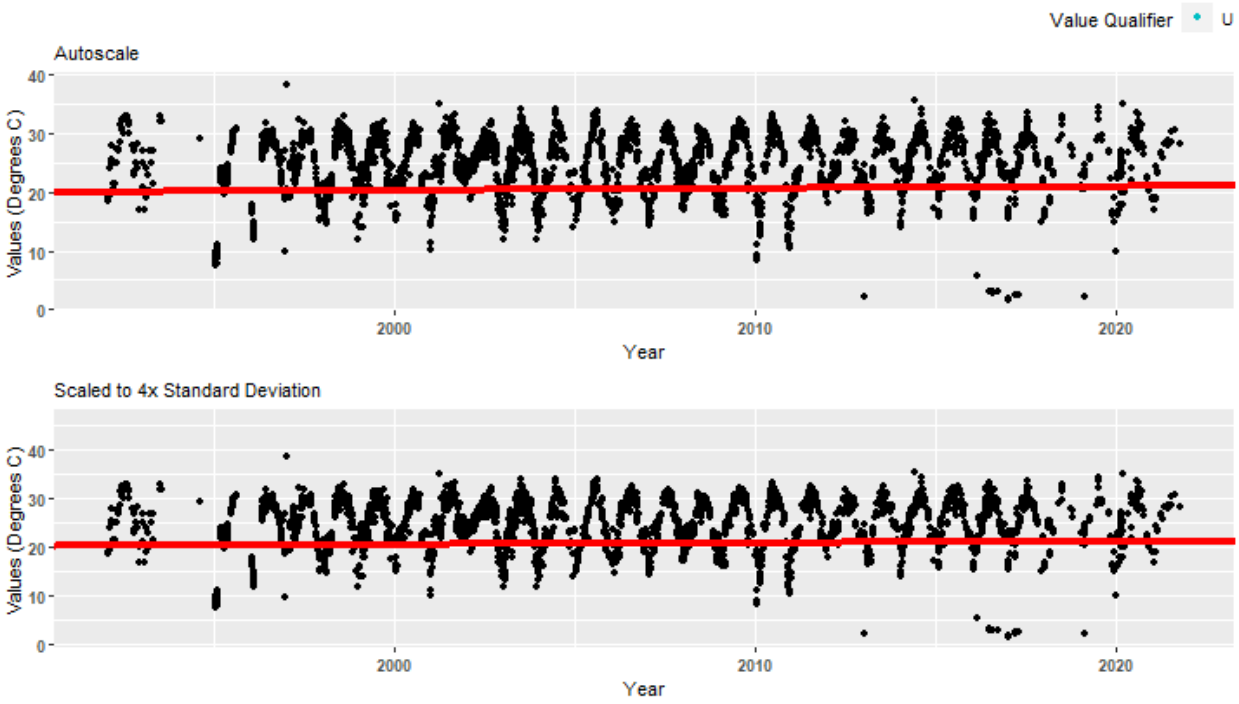
Data Points with Trendlines for Indian River-Malabar to Vero Beach Aquatic Preserve

Senn Slope = 0.0274999999999999, Senn Intercept = -32.6896428571427
Trend = 1, tau = 0.0536, p = 0
Linear Trendline: y = -0.0647554571165072x + 154.188156411678



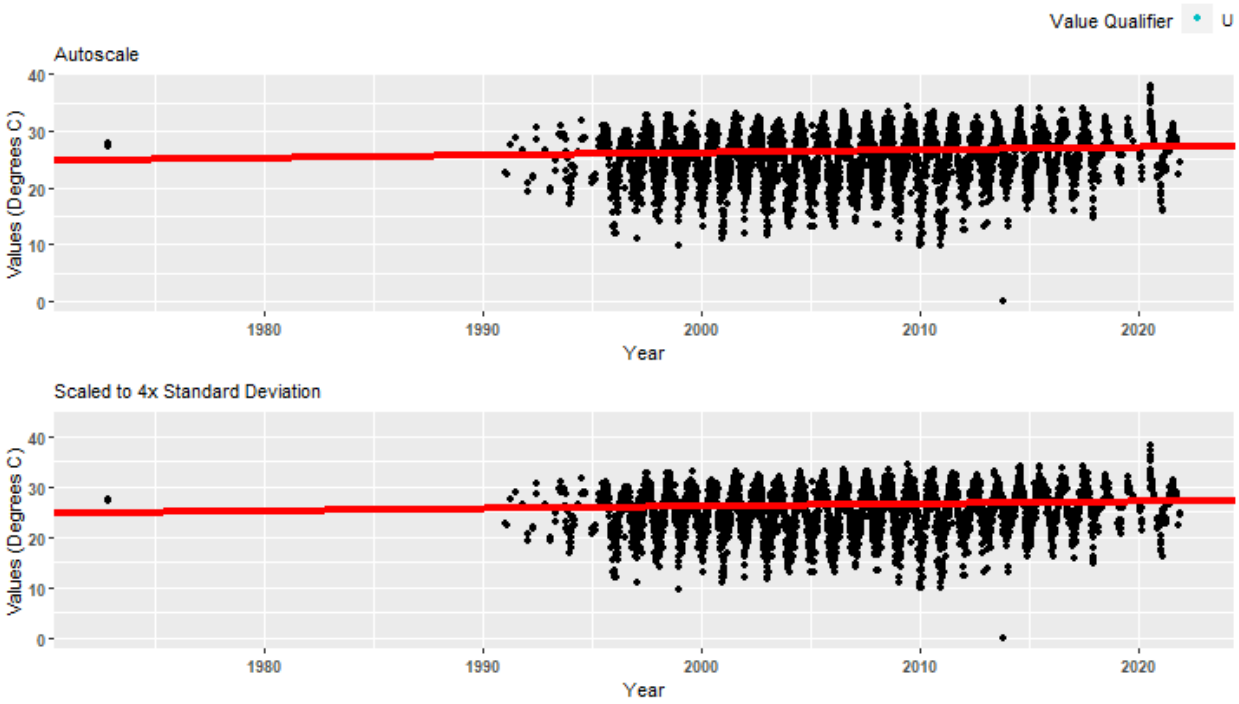
Data Points with Trendlines for Indian River-Vero Beach to Ft. Pierce Aquatic Preserve

Senn Slope = 0.0333333333333326, Senn Intercept = -46.0582608695651
Trend = 1, tau = 0.0781, p = 0
Linear Trendline: y = -0.0258794402939478x + 76.4974150584765



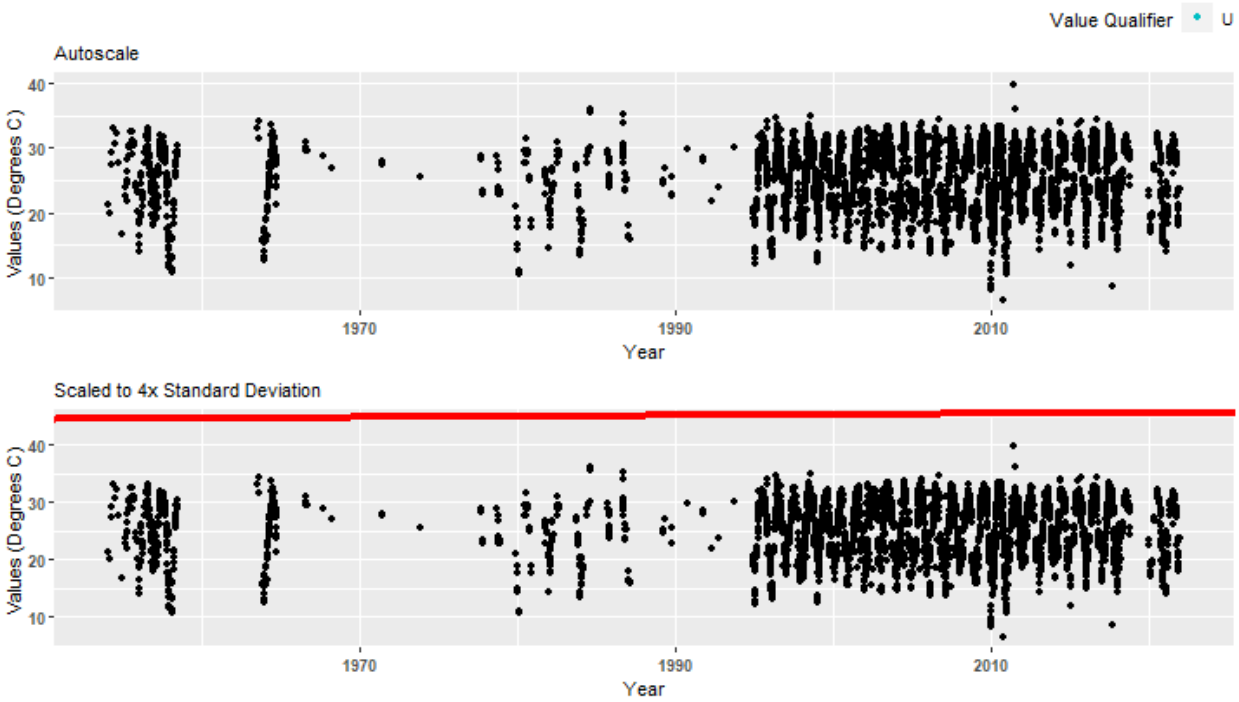
Data Points with Trendlines for Jensen Beach to Jupiter Inlet Aquatic Preserve

Senn Slope = 0.0470588235294118, Senn Intercept = -67.9357142857142
Trend = 1, tau = 0.0965, p = 0
Linear Trendline: y = 0.0397411302744567x + -54.7808288302511



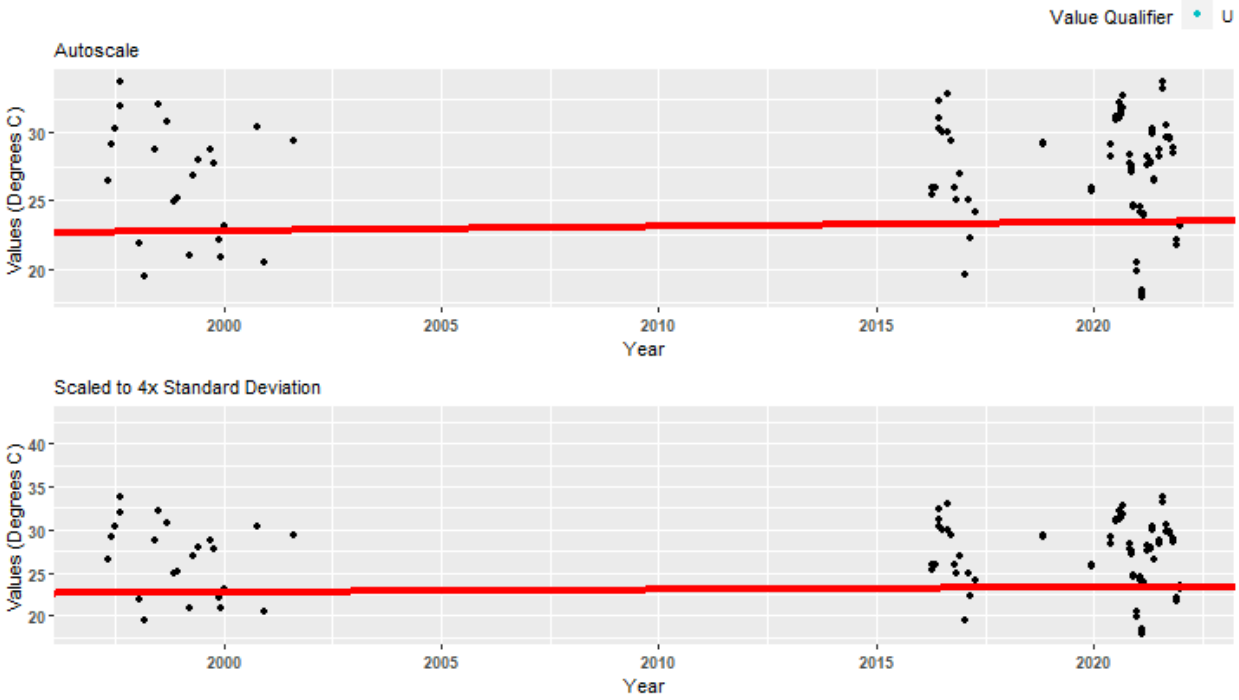
Data Points with Trendlines for Lemon Bay Aquatic Preserve

Senn Slope = 0.0153846153846156, Senn Intercept = 14.6662358974359
Trend = 1, tau = 0.0497, p = 0
Linear Trendline: y = 0.0218745308039911x + -18.2865028043796



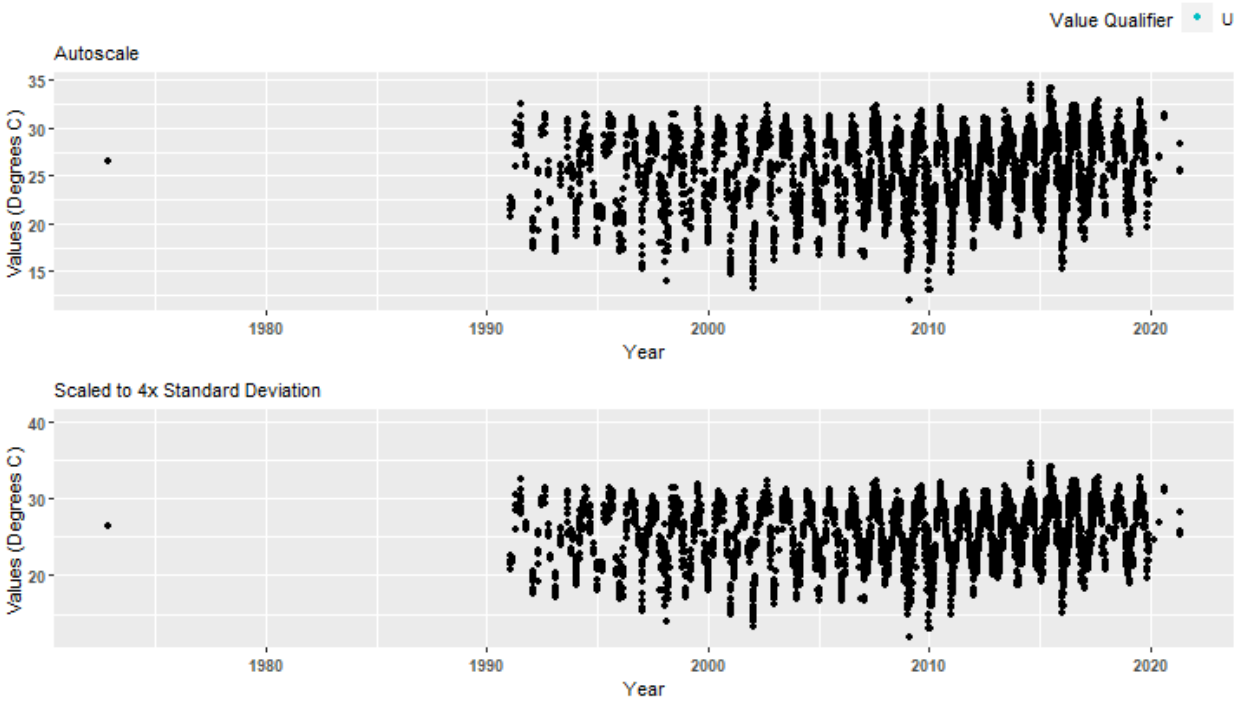
Data Points with Trendlines for Lignumvitae Key Aquatic Preserve

Senn Slope = 0.0304347826086958, Senn Intercept = -38.0145454545457
Trend = 0, tau = 0.1015, p = 0.6417
Linear Trendline: y = 0.00923725686947799x + 8.44436849287068



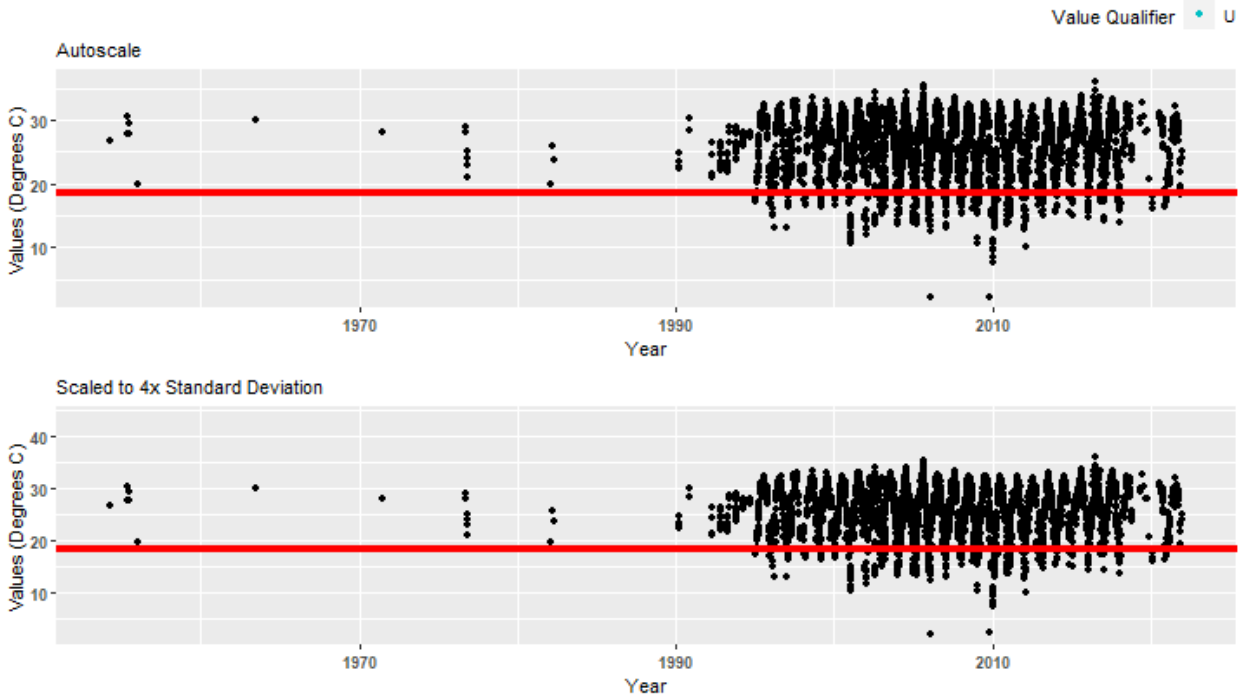
Data Points with Trendlines for Loxahatchee River-Lake Worth Creek Aquatic Preserve

Senn Slope = 0.0571428571428574, Senn Intercept = -137.775
Trend = 1, tau = 0.1549, p = 0
Linear Trendline: y = 0.0744395035695761x + -124.120046557086



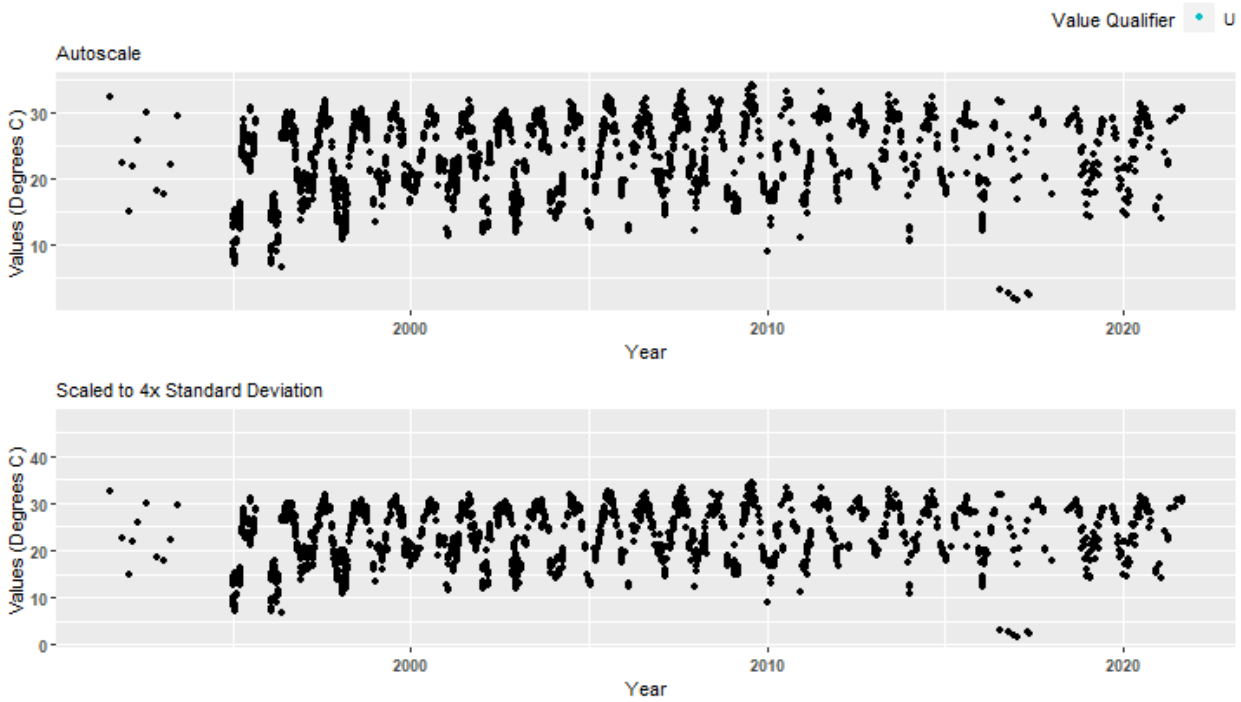
Data Points with Trendlines for Matlacha Pass Aquatic Preserve

Senn Slope = 0, Senn Intercept = 18.4999999999998
Trend = 0, tau = 0.0045, p = 0.645
Linear Trendline: y = -0.0163284708003389x + 58.2822239745043



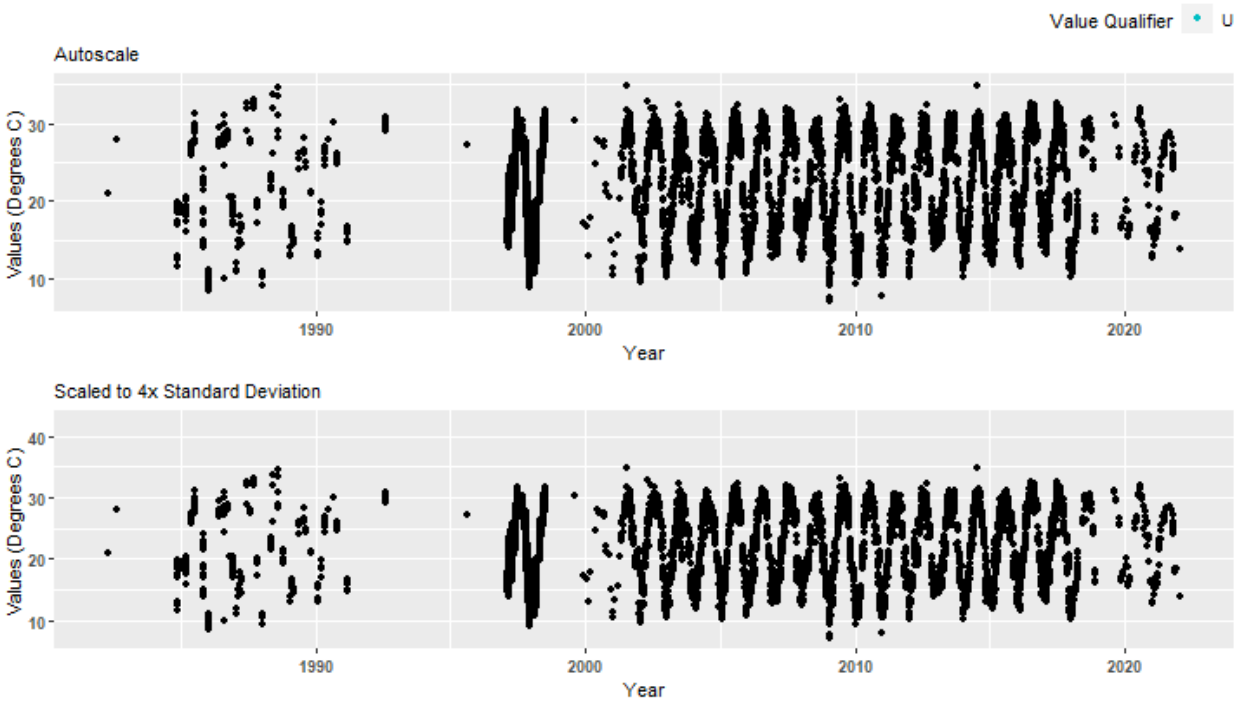
Data Points with Trendlines for Mosquito Lagoon Aquatic Preserve

Senn Slope = 0.0692307692307694, Senn Intercept = -151.88125
Trend = 1, tau = 0.1456, p = 0
Linear Trendline: y = 0.0224717220596644x + -21.3999652780378



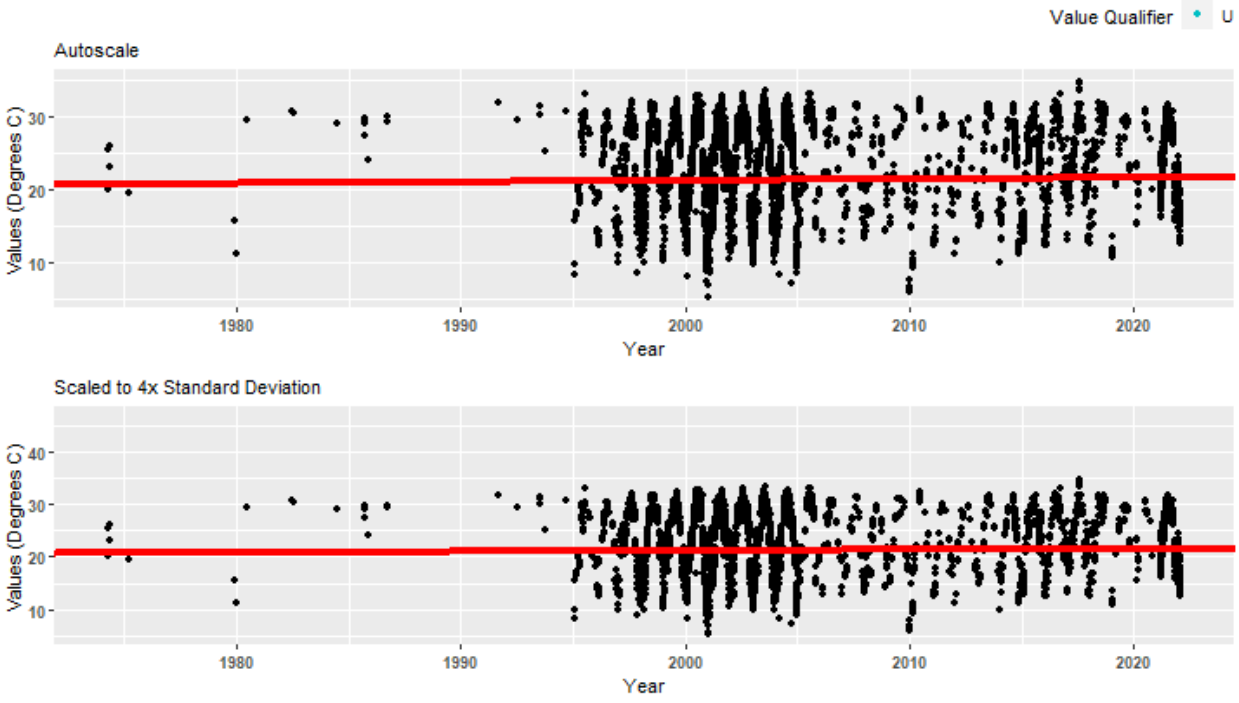
Data Points with Trendlines for Nassau River-St. Johns River Marshes Aquatic Preserve

Senn Slope = -0.0161538461538459, Senn Intercept = -45.0024999999998
Trend = -1, tau = -0.013, p = 0
Linear Trendline: y = 0.0767501918973709x + -132.030997869462



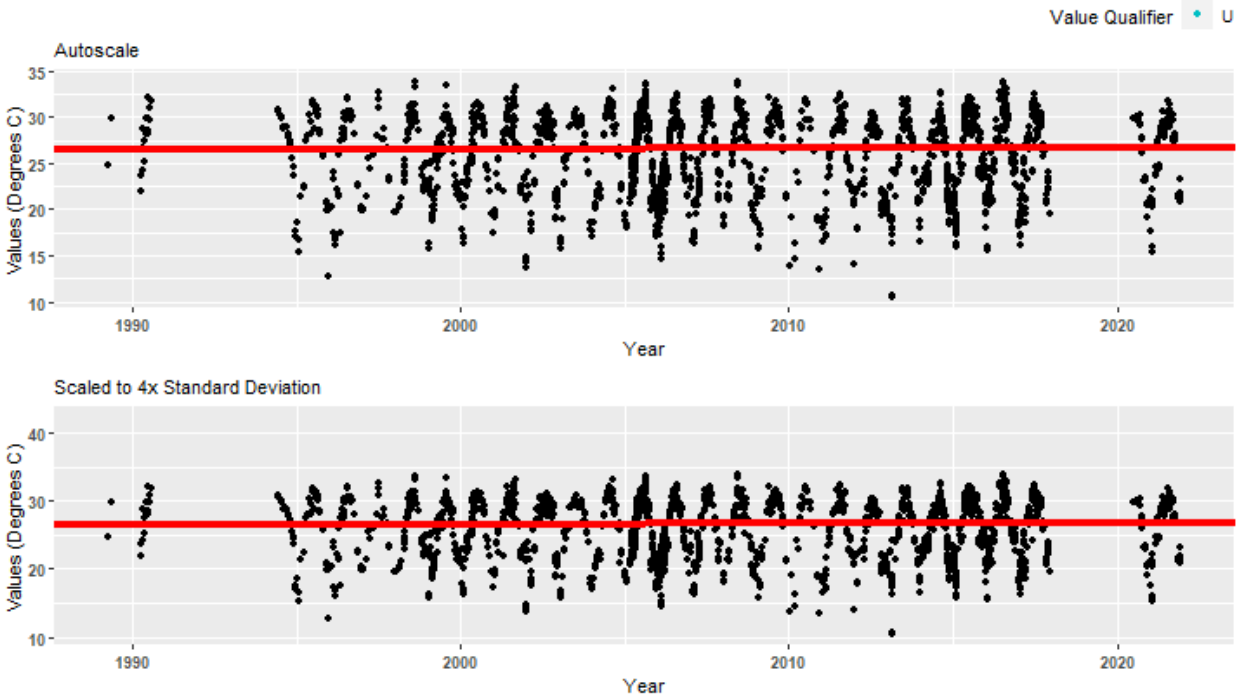
Data Points with Trendlines for Nature Coast Aquatic Preserve

Senn Slope = 0.0199999999999999, Senn Intercept = -18.6174999999998
Trend = 1, tau = 0.0434, p = 0
Linear Trendline: y = 0.0327484348606014x + -42.1845887078481



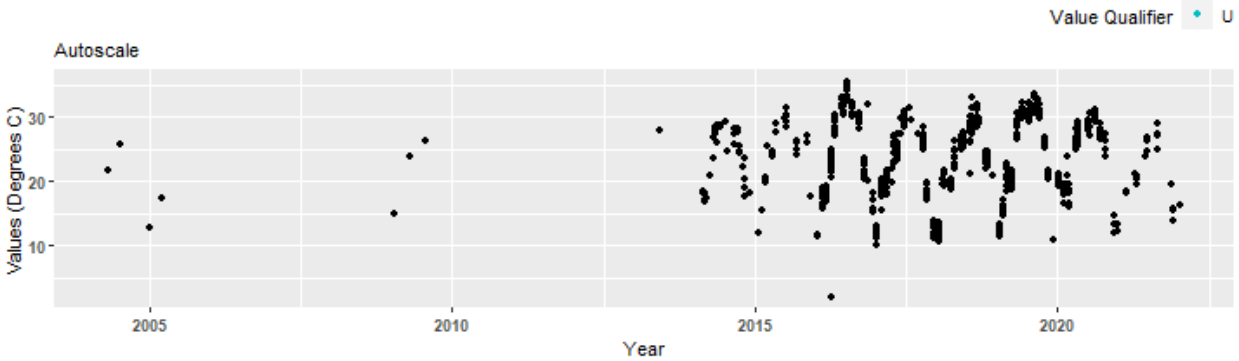
Data Points with Trendlines for North Fork St. Lucie Aquatic Preserve

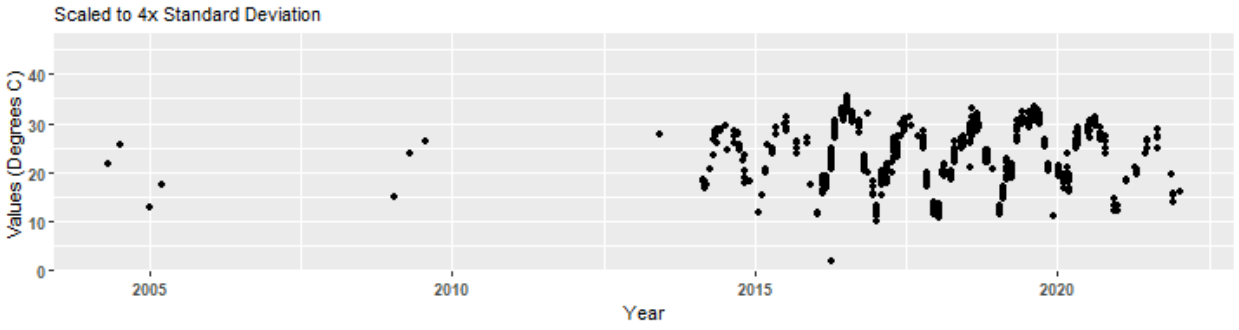
Senn Slope = 0.00499999999999989, Senn Intercept = 16.6375
Trend = 0, tau = 0.0151, p = 0.2068
Linear Trendline: y = -0.00303570081709087x + 31.5779696939685



Data Points with Trendlines for Pellicer Creek Aquatic Preserve

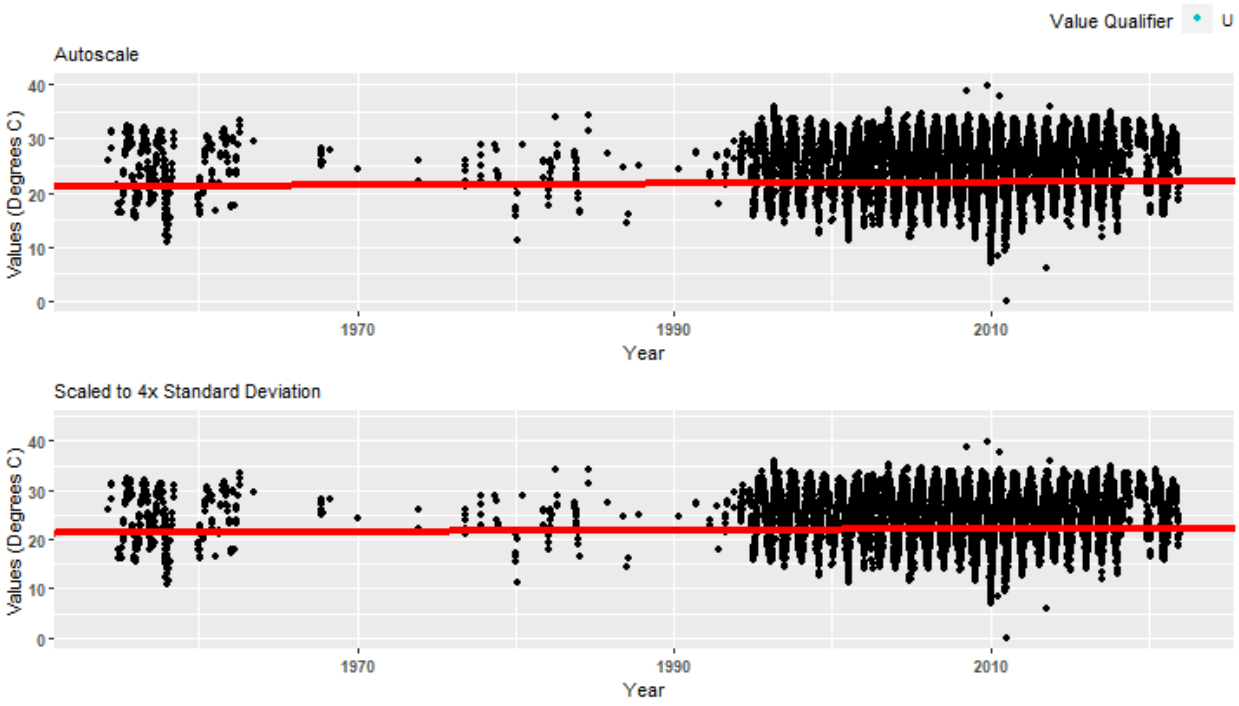
Senn Slope = 0.0552500000000005, Senn Intercept = 5.72958333333355
Trend = 0, tau = 0.0265, p = 0.3223
Linear Trendline: y = 0.144260617121818x + -267.471236725905





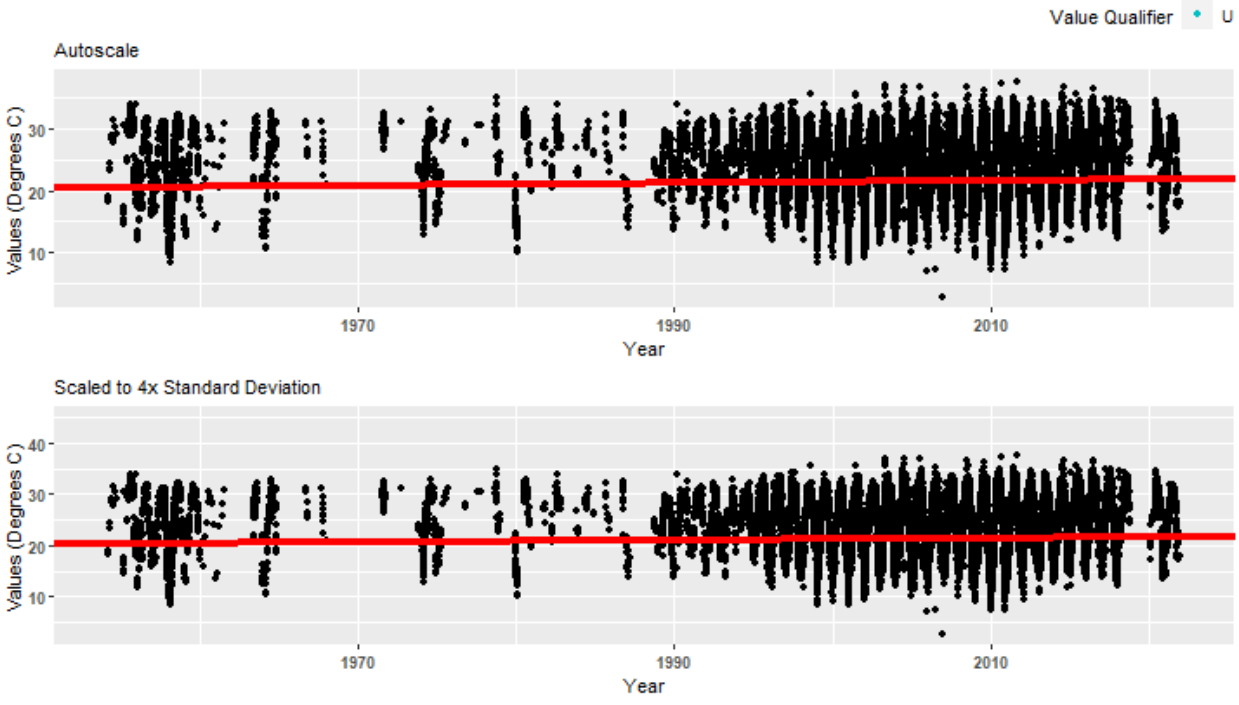
Data Points with Trendlines for Pine Island Sound Aquatic Preserve

Senn Slope = 0.0142857142857135, Senn Intercept = -6.46193181818155
Trend = 1, tau = 0.0396, p = 0
Linear Trendline: y = 0.0344251800818867x + -43.5203520094298



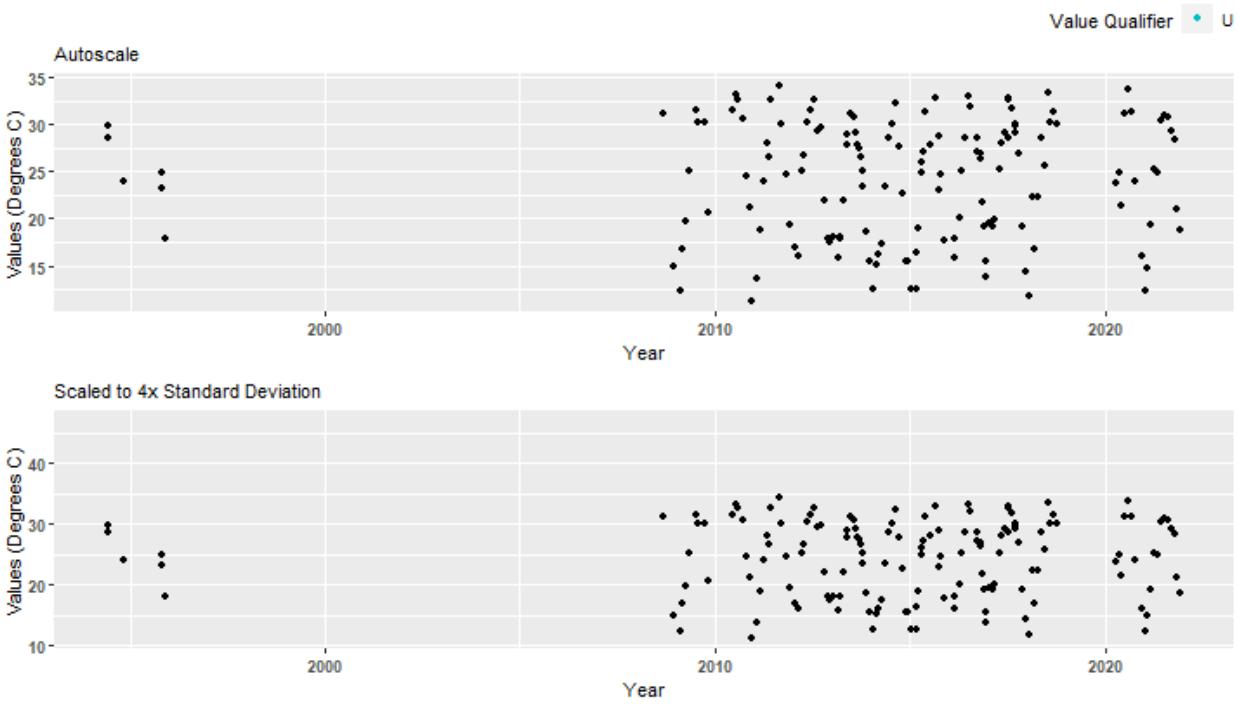
Data Points with Trendlines for Pinellas County Aquatic Preserve

Senn Slope = 0.0201754385964912, Senn Intercept = -18.9228571428572
Trend = 1, tau = 0.0686, p = 0
Linear Trendline: y = 0.0208986532929482x + -16.8848808842082



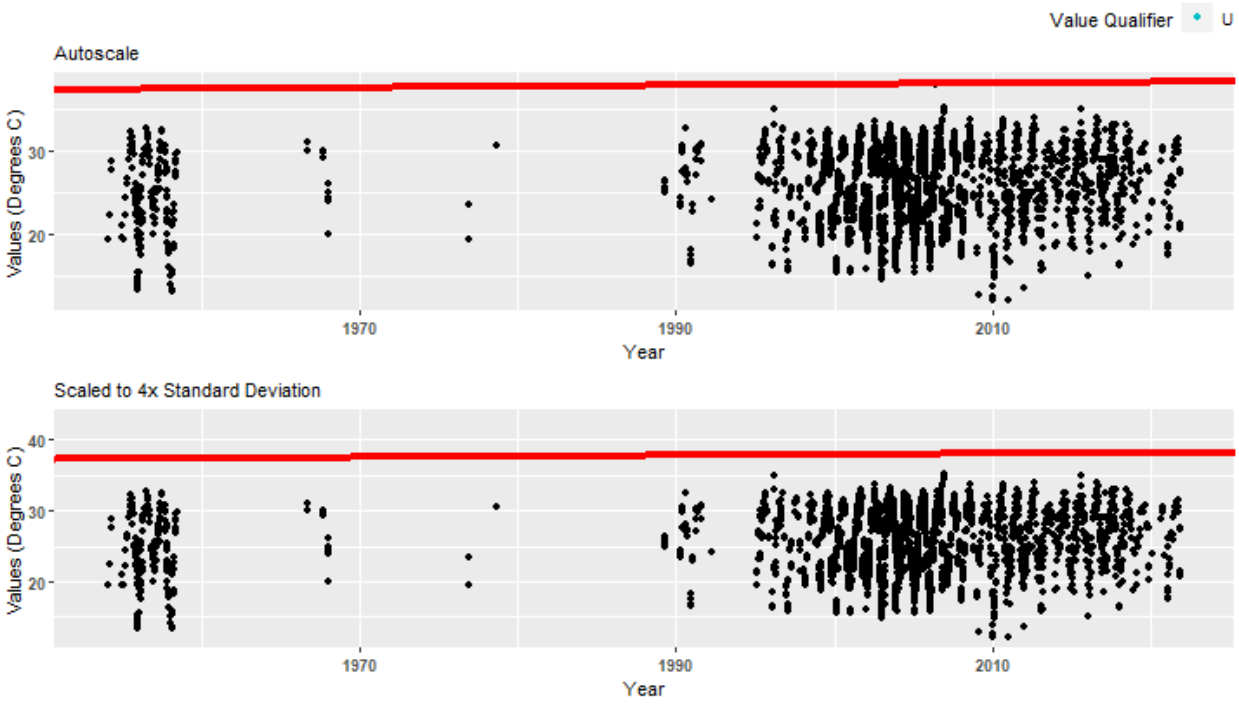
Data Points with Trendlines for Rocky Bayou State Park Aquatic Preserve

Senn Slope = 0.0575772727272727, Senn Intercept = -7.97840863095234
Trend = 0, tau = 0.061, p = 0.1257
Linear Trendline: y = 0.0183000871791618x + -12.6239561122326



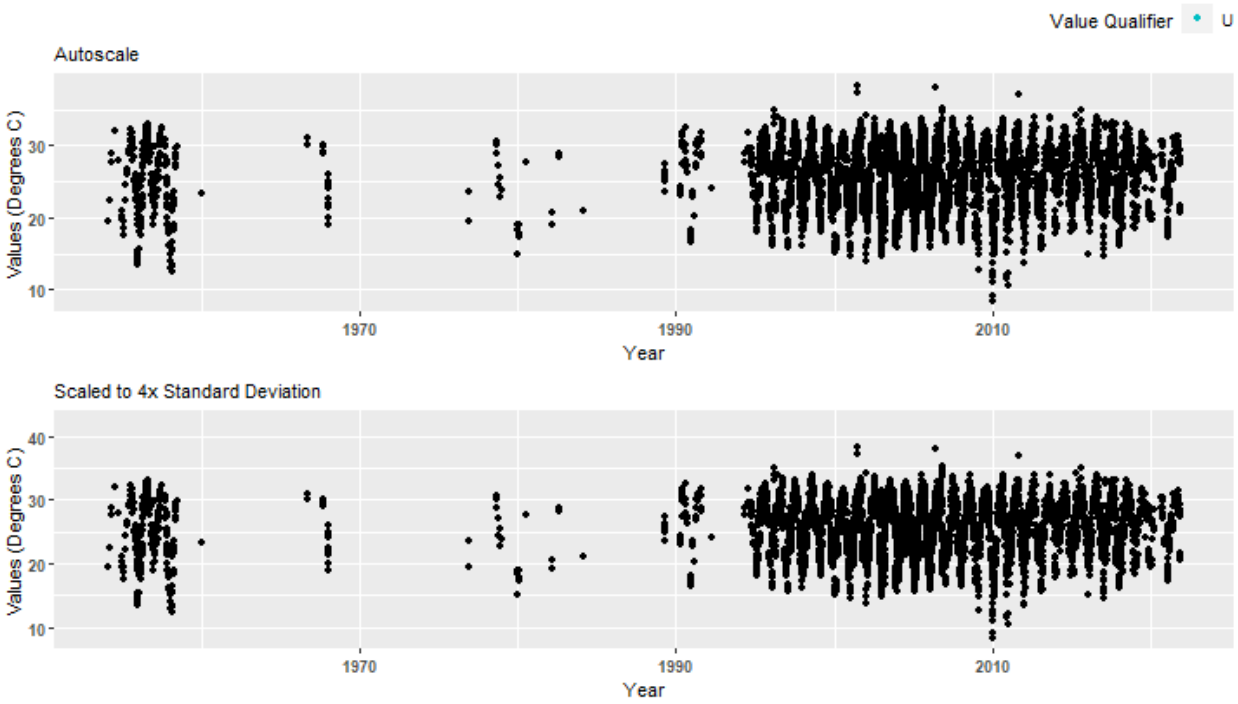
Data Points with Trendlines for Rookery Bay Aquatic Preserve

Senn Slope = 0.0133333333333334, Senn Intercept = 11.3017770034844
Trend = 1, tau = 0.0452, p = 0.0001
Linear Trendline: y = 0.026312389909253x + -26.8347568737919



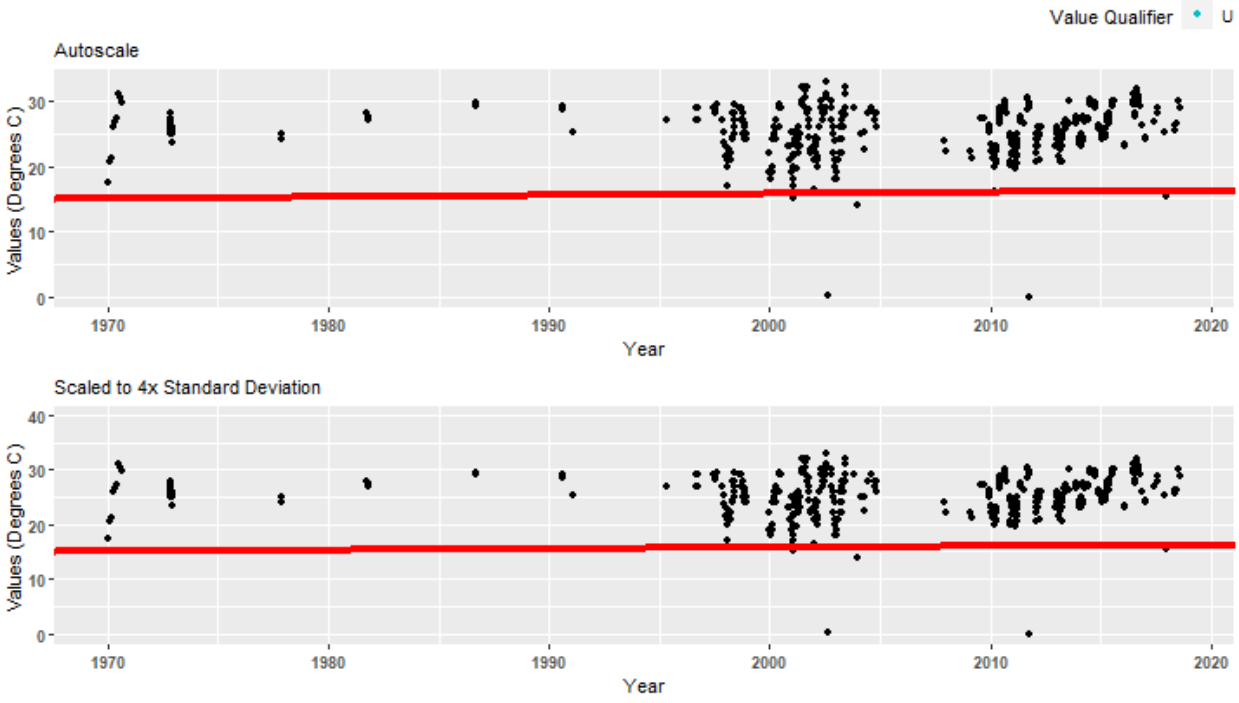
Data Points with Trendlines for Rookery Bay National Estuarine Research Reserve

Senn Slope = 0.00999999999999979, Senn Intercept = 34.1979545454545
Trend = 1, tau = 0.0269, p = 0
Linear Trendline: y = 0.0169178066961478x + -7.99666513341684



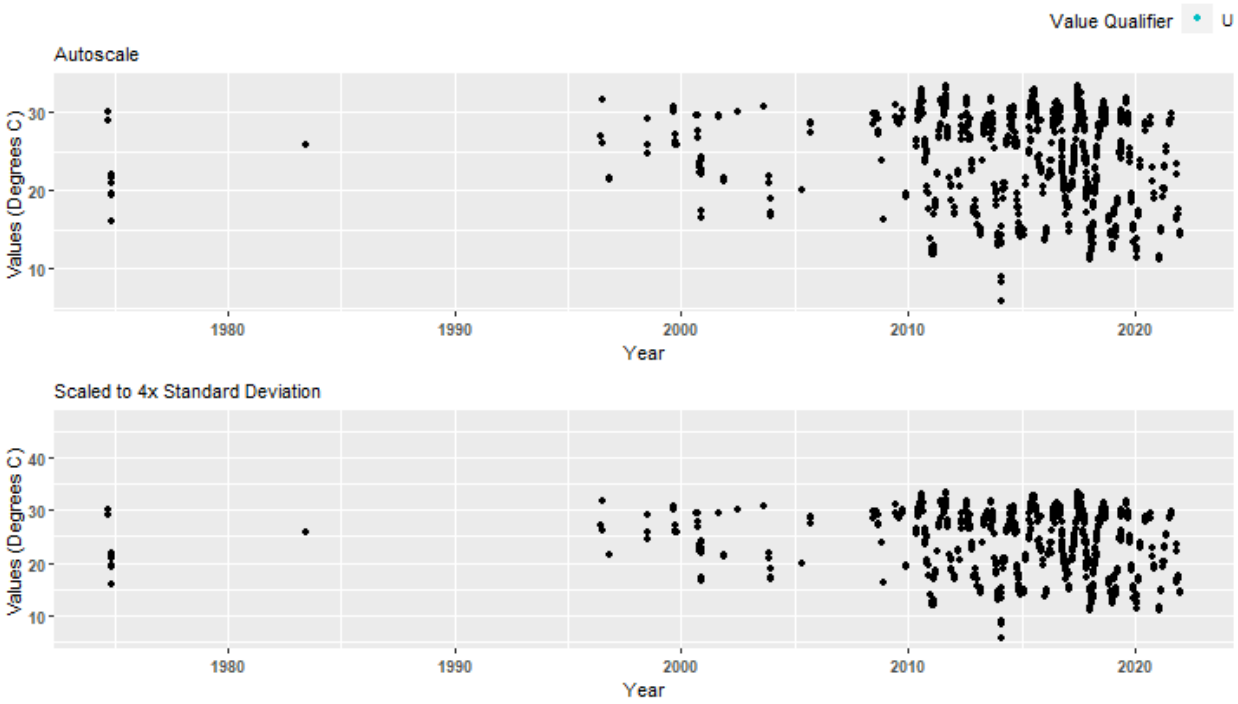
Data Points with Trendlines for Southeast Florida Coral Reef Ecosystem Conservation Area

Senn Slope = 0.0248780487804878, Senn Intercept = -33.8170576923076
Trend = 1, tau = 0.1158, p = 0.0006
Linear Trendline: y = 0.00392732585563837x + 17.8631172867807



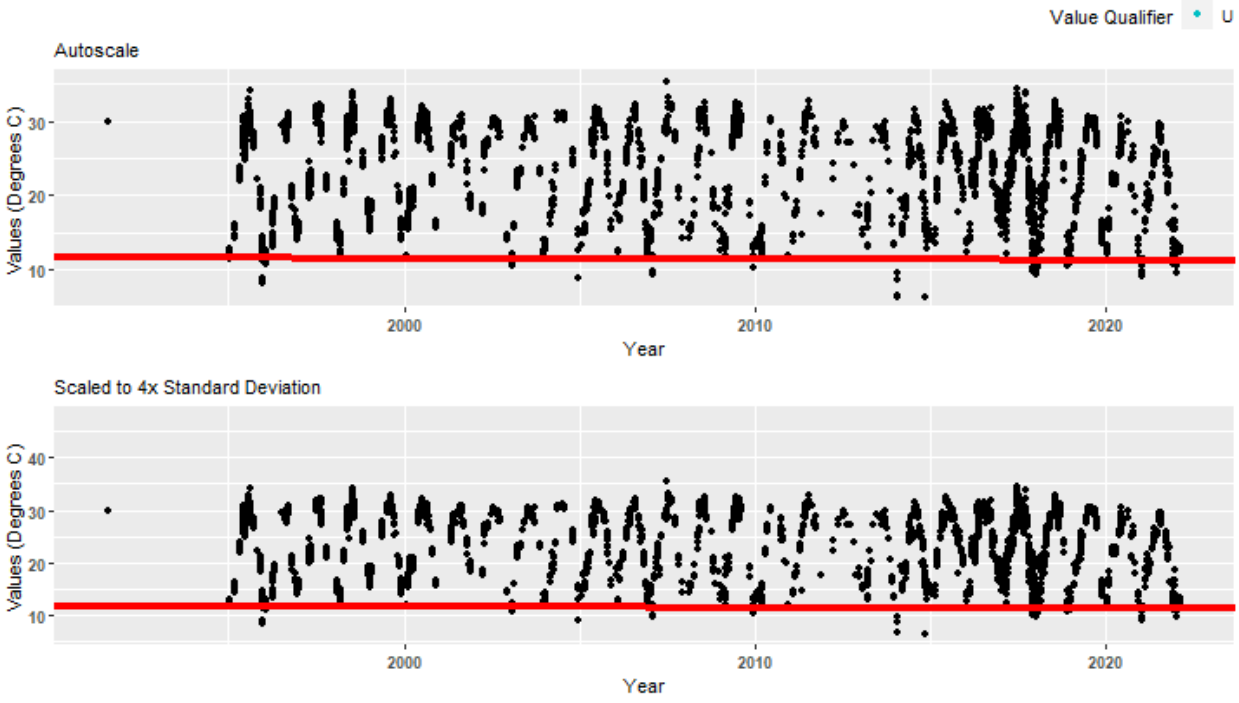
Data Points with Trendlines for St. Andrews State Park Aquatic Preserve

Senn Slope = 0, Senn Intercept = 56.7787142857143
Trend = 0, tau = -0.012, p = 0.864
Linear Trendline: y = -0.0569021812177698x + 139.150292729749



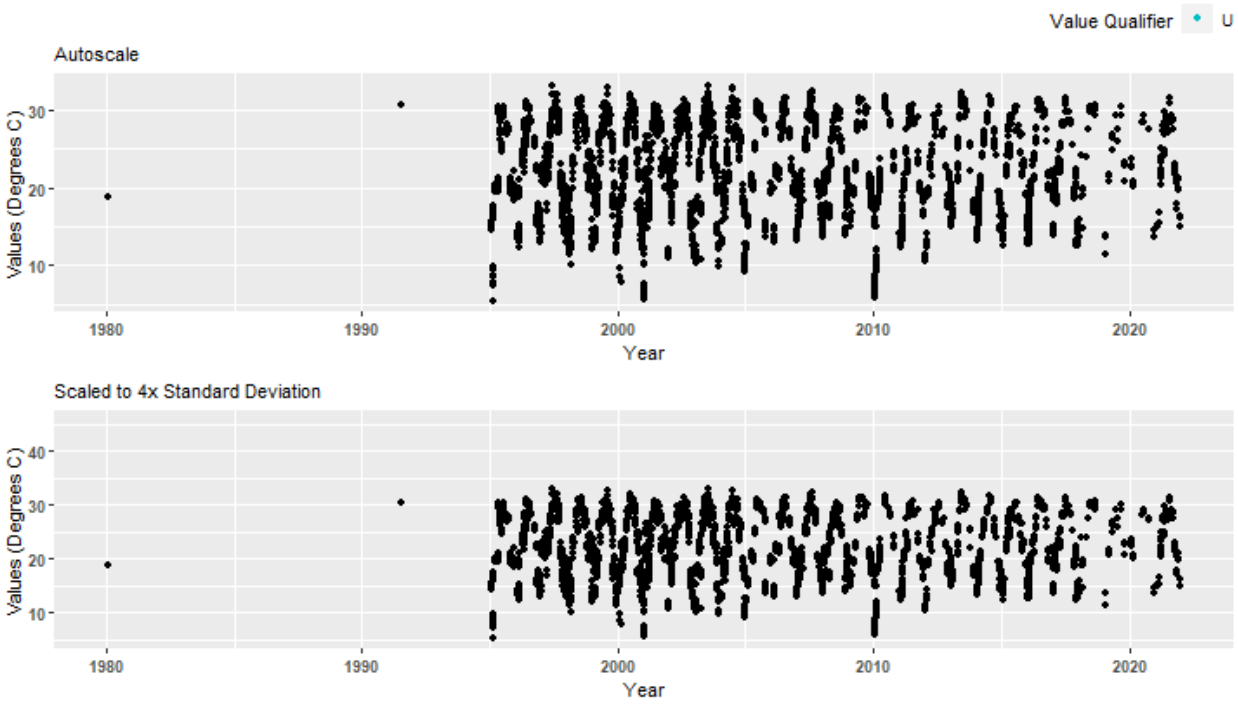
Data Points with Trendlines for St. Joseph Bay Aquatic Preserve

Senn Slope = -0.0111111111111113, Senn Intercept = 33.7916666666668
Trend = -1, tau = -0.0206, p = 0.0033
Linear Trendline: y = -0.0288026826319992x + 81.6353012749619



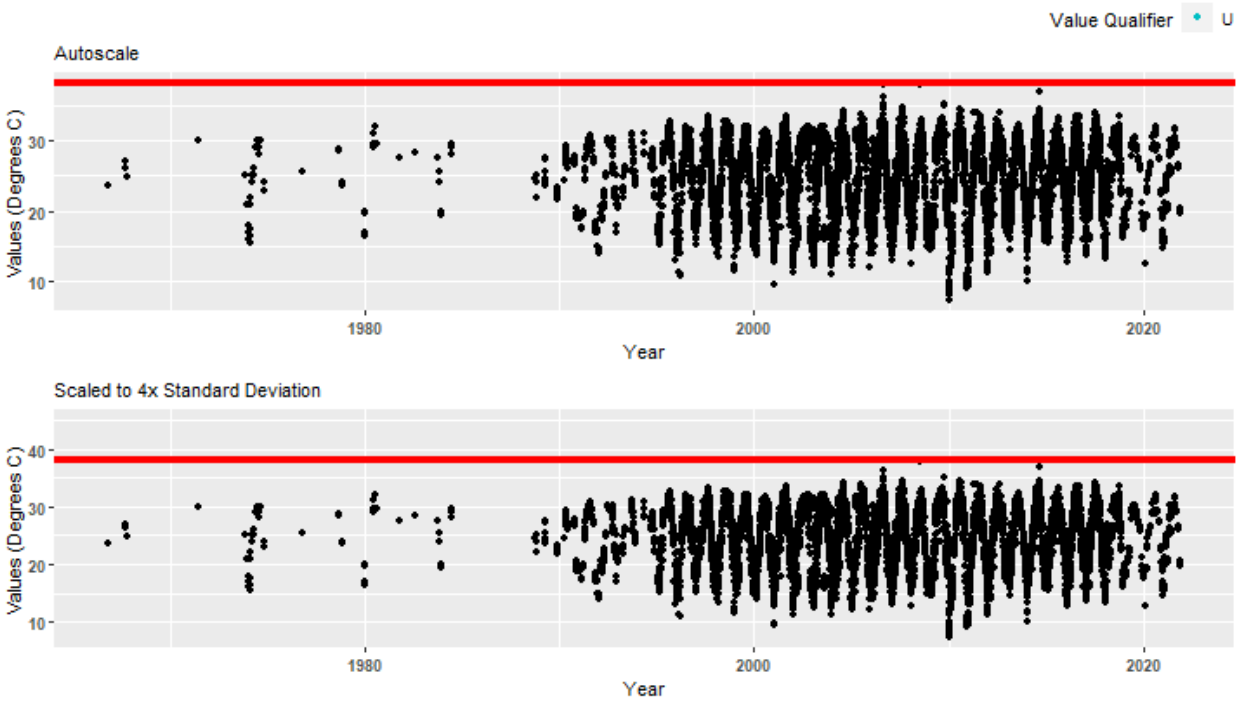
Data Points with Trendlines for St. Martins Marsh Aquatic Preserve

Senn Slope = 0.0416666666666667, Senn Intercept = -91.5791208791208
Trend = 1, tau = 0.0845, p = 0
Linear Trendline: y = -0.0198415499278492x + 62.1473779300912



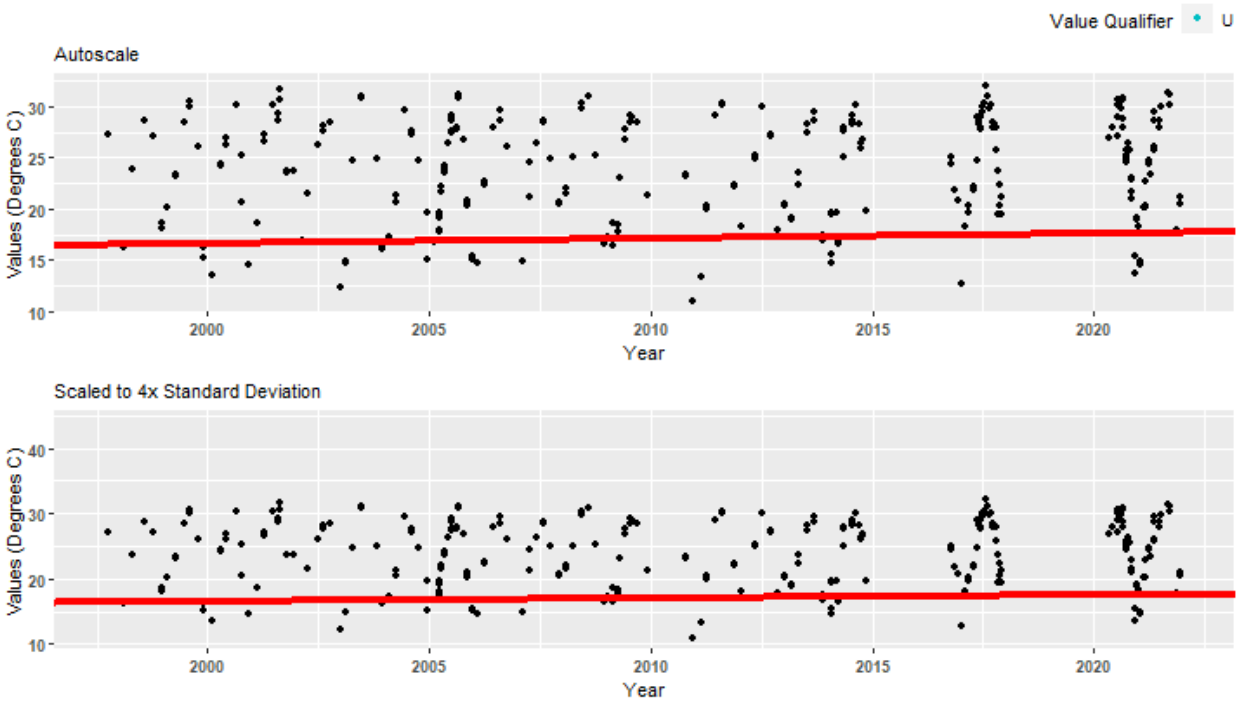
Data Points with Trendlines for Terra Ceia Aquatic Preserve

Senn Slope = 0, Senn Intercept = 38.0972321428571
Trend = 0, tau = -0.0035, p = 0.2226
Linear Trendline: y = 0.0241626051652632x + -23.3644111632024



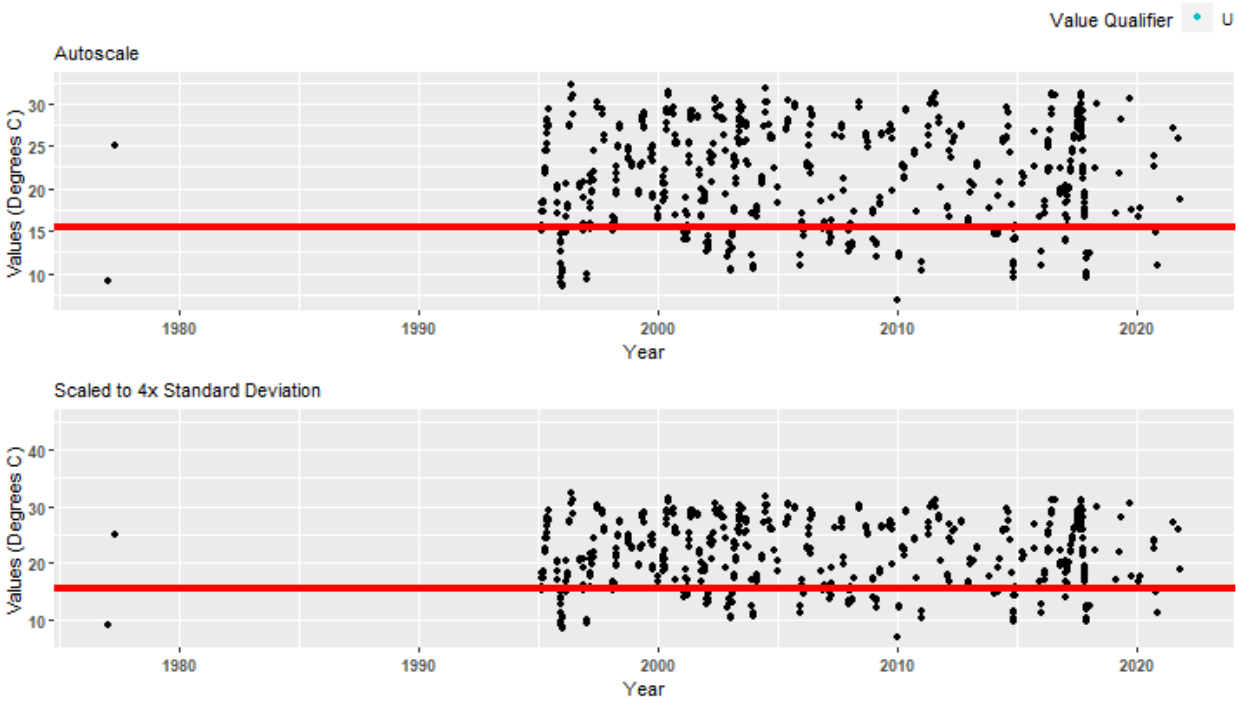
Data Points with Trendlines for Tomoka Marsh Aquatic Preserve

Senn Slope = 0.05, Senn Intercept = -83.3208076923079
Trend = 1, tau = 0.1036, p = 0.0116
Linear Trendline: y = 0.056366937228x + -89.4817218749535



Data Points with Trendlines for Yellow River Marsh Aquatic Preserve

Senn Slope = 0, Senn Intercept = 15.3350000000001
Trend = 0, tau = -0.0039, p = 0.918
Linear Trendline: y = 0.0193864921781496x + -17.3087147520867



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, ")")) +
```

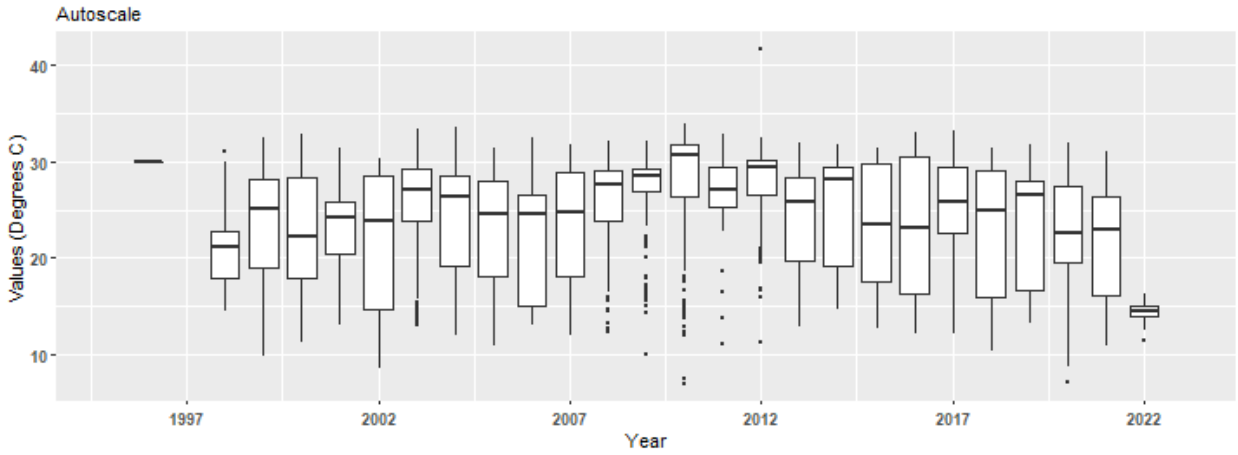
```
   scale_x_continuous(limits = c(year_lower - 1, year_upper + 1),
                      breaks = rev(seq(year_upper,
                                       year_lower, -x_scale))) +
   theme(axis.text.x = element_text(face = "bold"),
         axis.text.y = element_text(face = "bold"))

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))) +
```

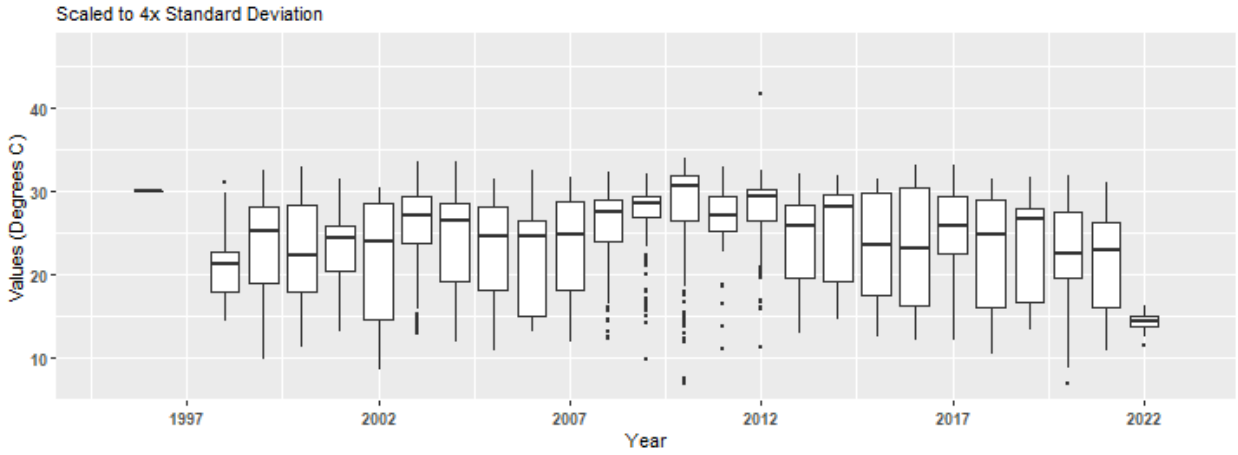
```
   theme(legend.position = "none",
         axis.text.x = element_text(face = "bold"),
         axis.text.y = element_text(face = "bold"))

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) +
```

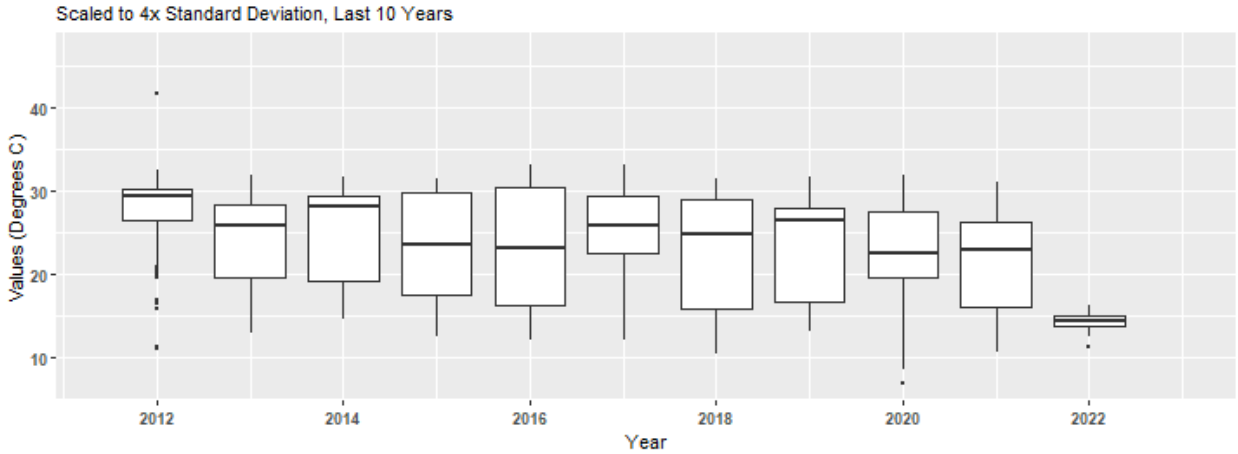
```
      labs(subtitle = "Autoscale",
           x = "Month", y = paste0("Values (", unit, ")"), fill = "Month") +
      scale_x_continuous(limits = c(0, 13), breaks = seq(3, 12, 3)) +
      theme(legend.position = "none",
            axis.text.x = element_text(face = "bold"),
            axis.text.y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

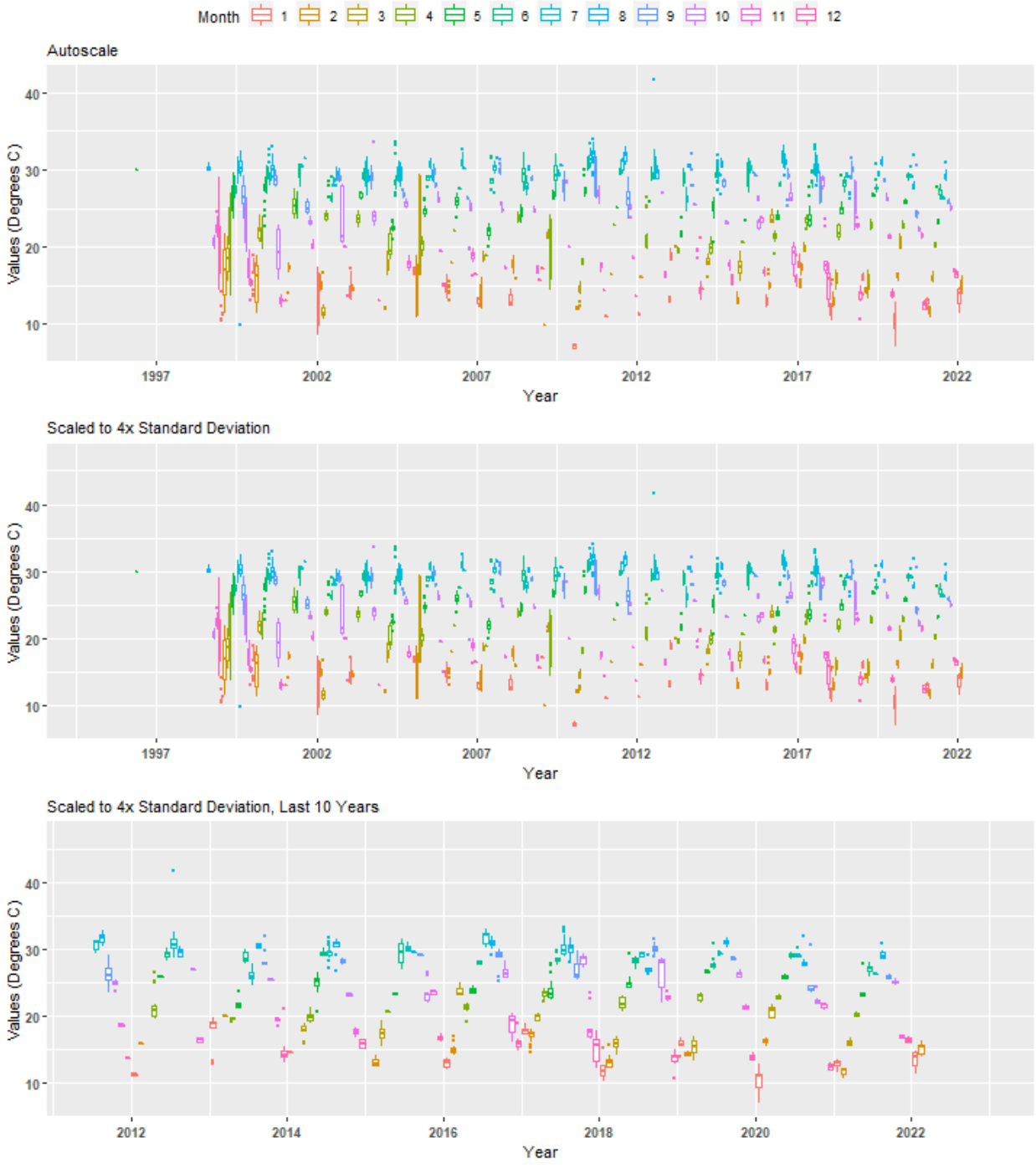






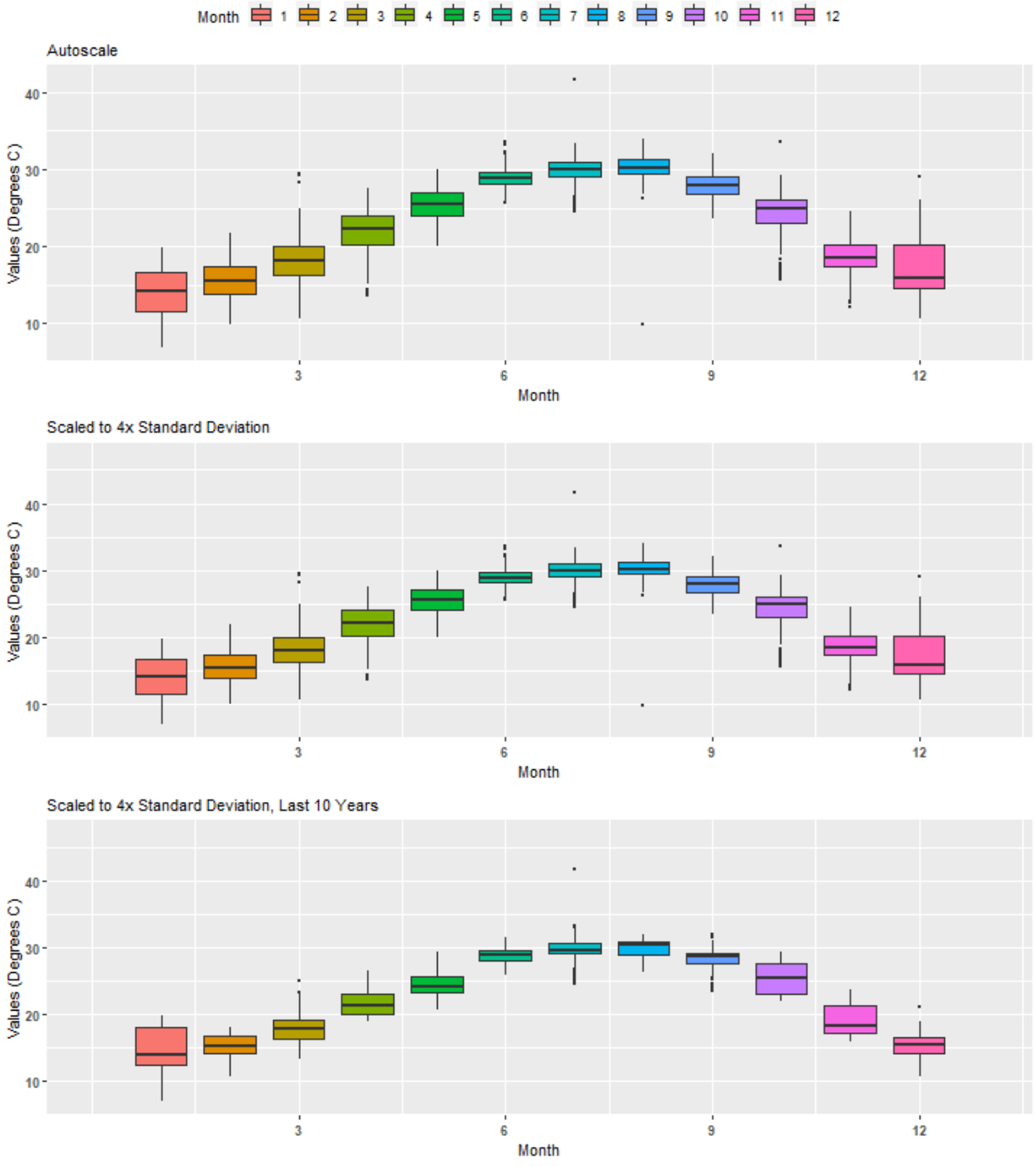
Summary Box Plots for Alligator Harbor Aquatic Preserve

By Year & Month



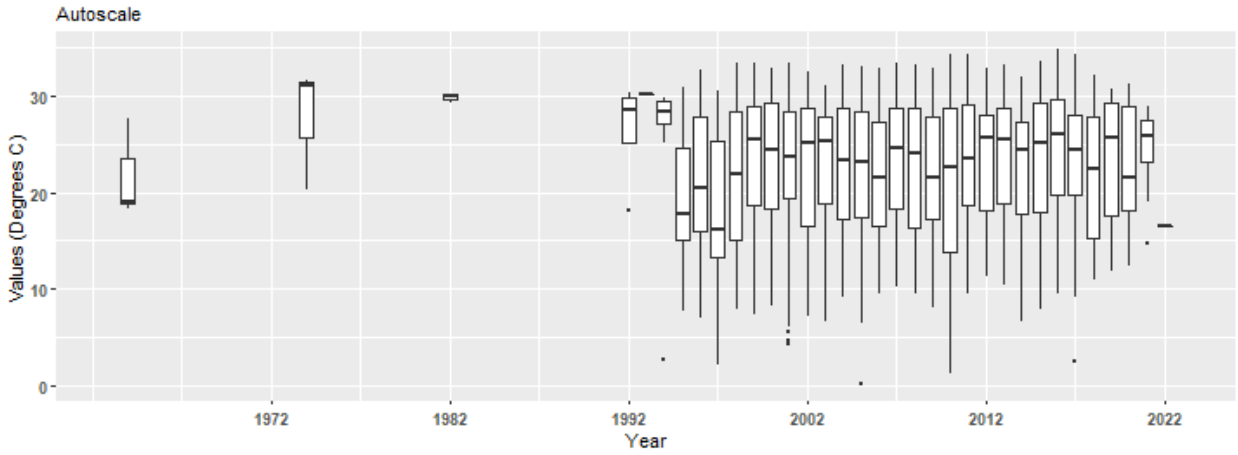
Summary Box Plots for Alligator Harbor Aquatic Preserve

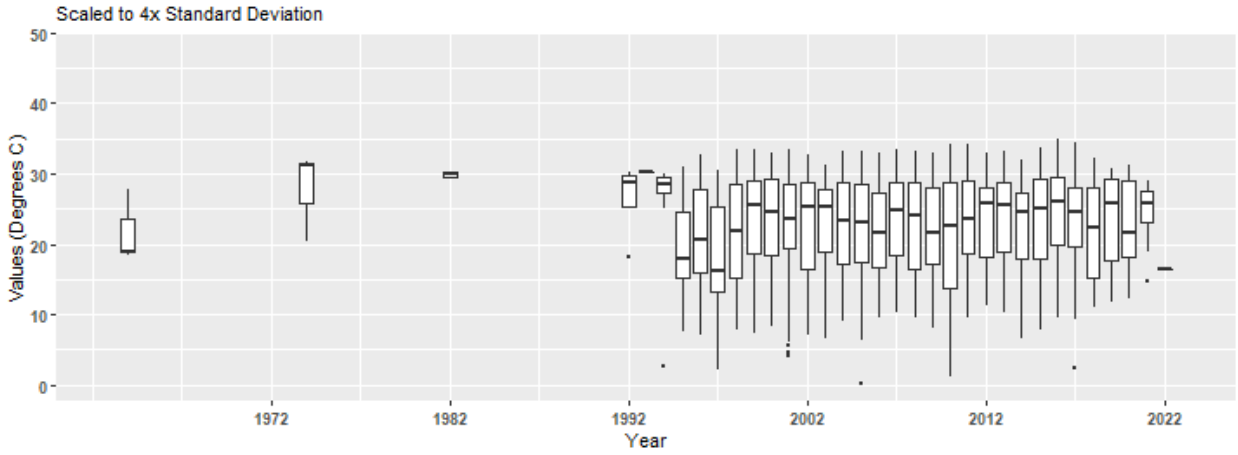
By Month

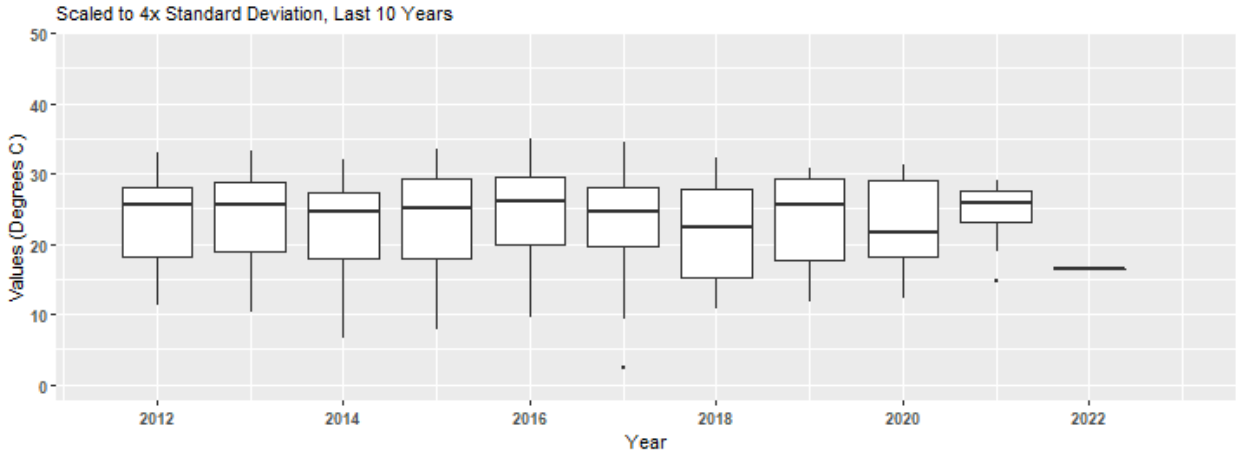


Summary Box Plots for Apalachicola Bay Aquatic Preserve

By Year

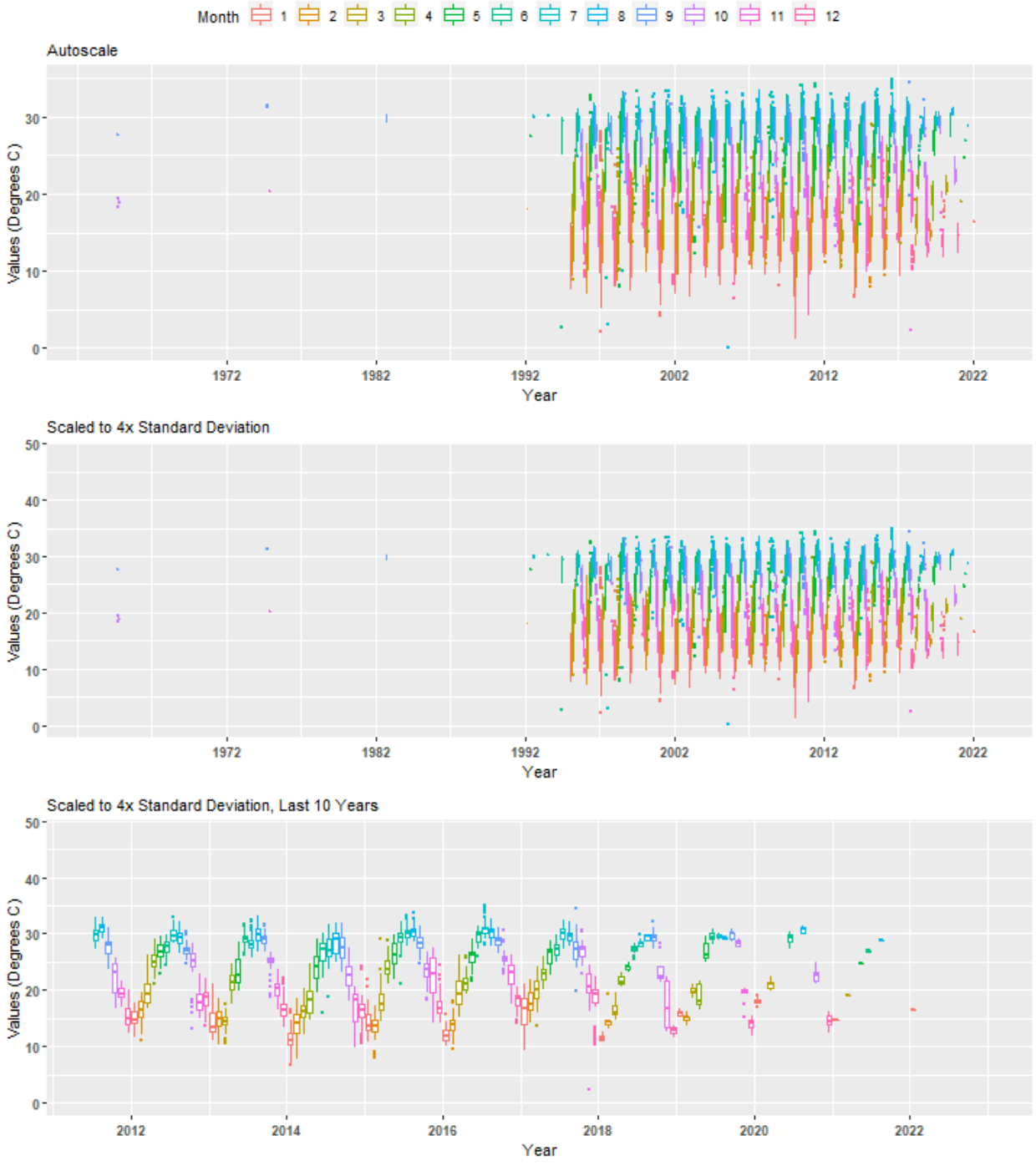






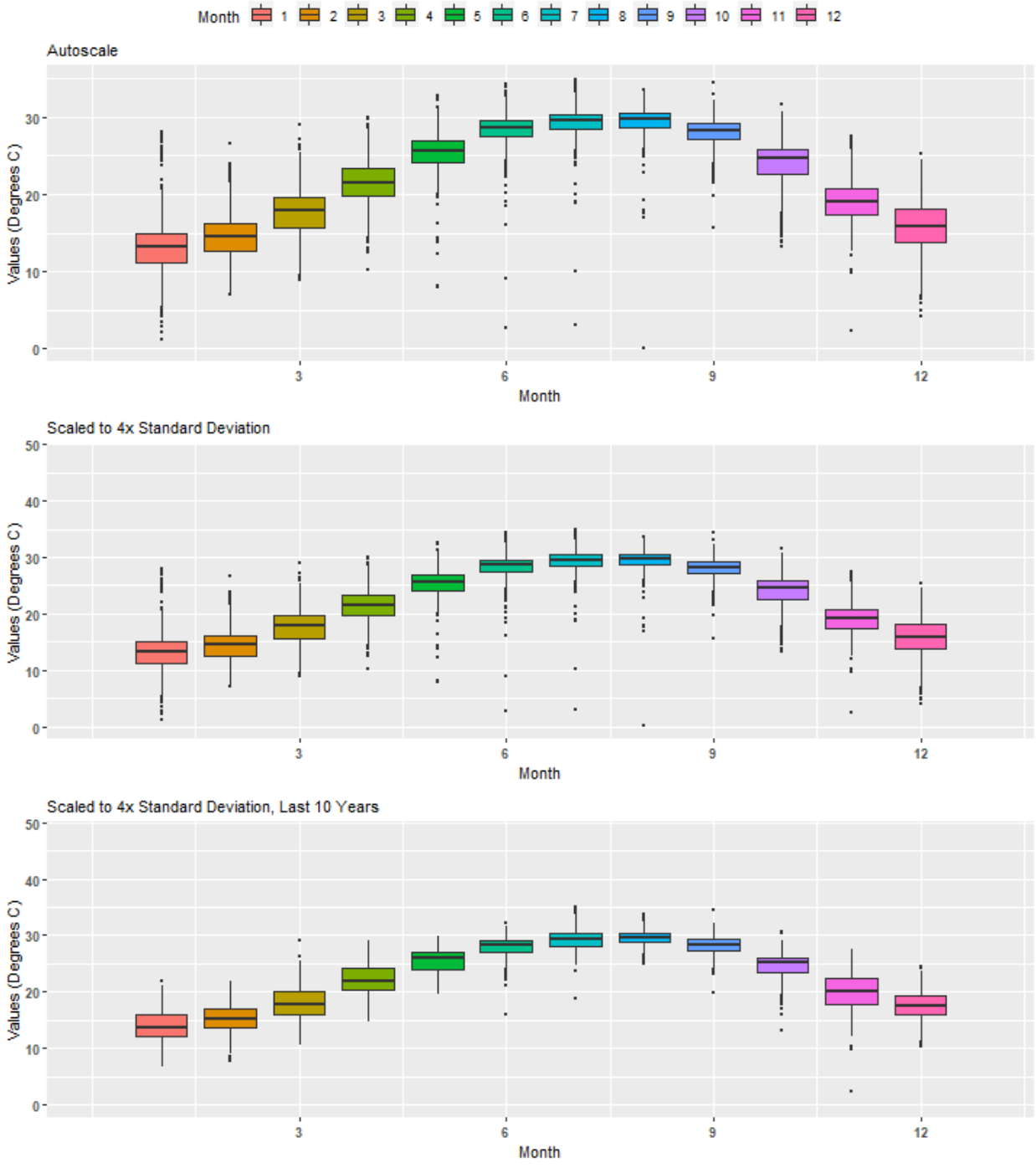
Summary Box Plots for Apalachicola Bay Aquatic Preserve

By Year & Month



Summary Box Plots for Apalachicola Bay Aquatic Preserve

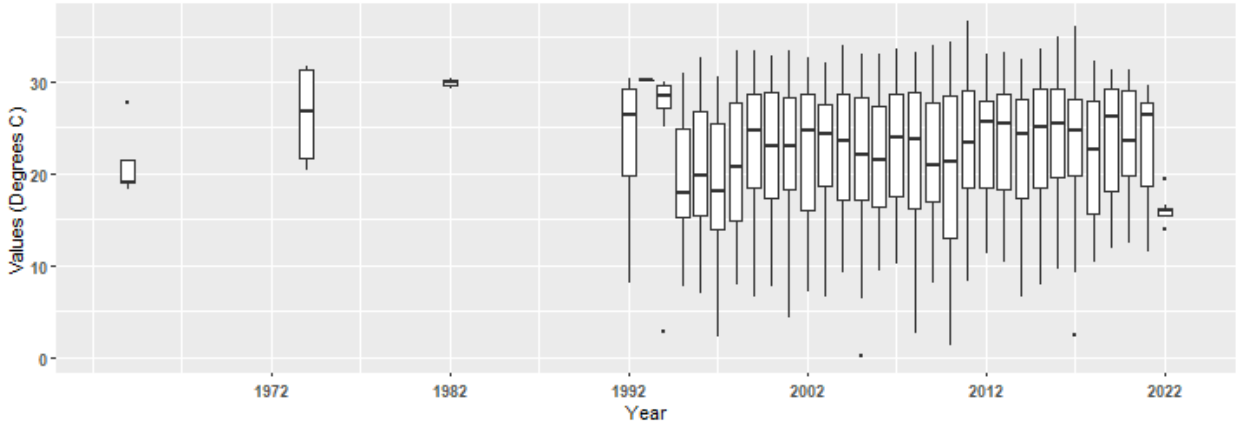
By Month



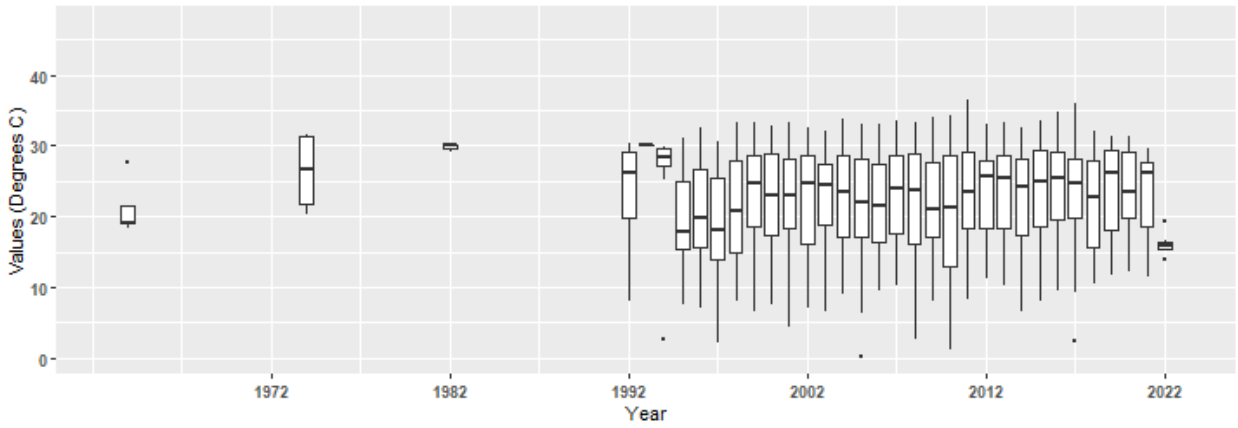
Summary Box Plots for Apalachicola National Estuarine Research Reserve

By Year

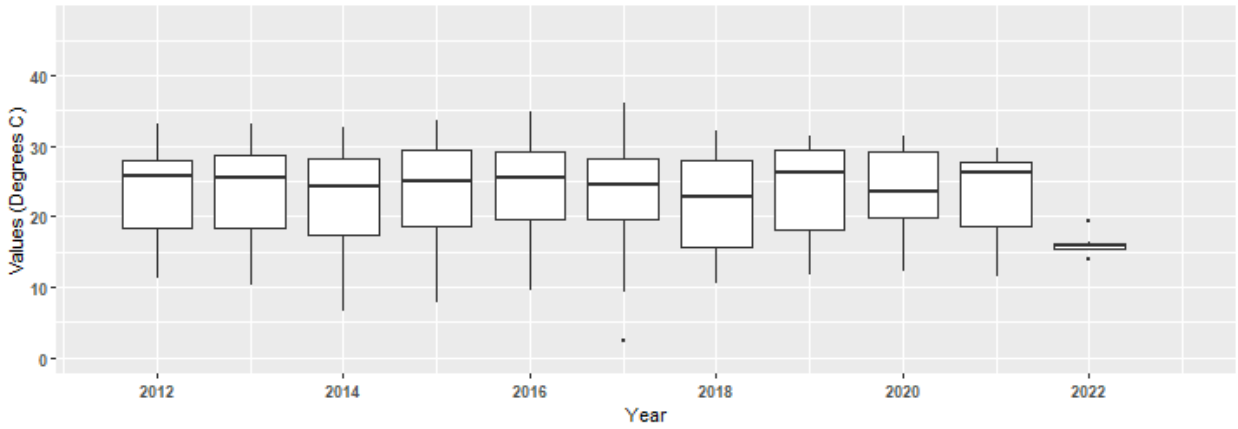
Autoscale



Scaled to 4x Standard Deviation

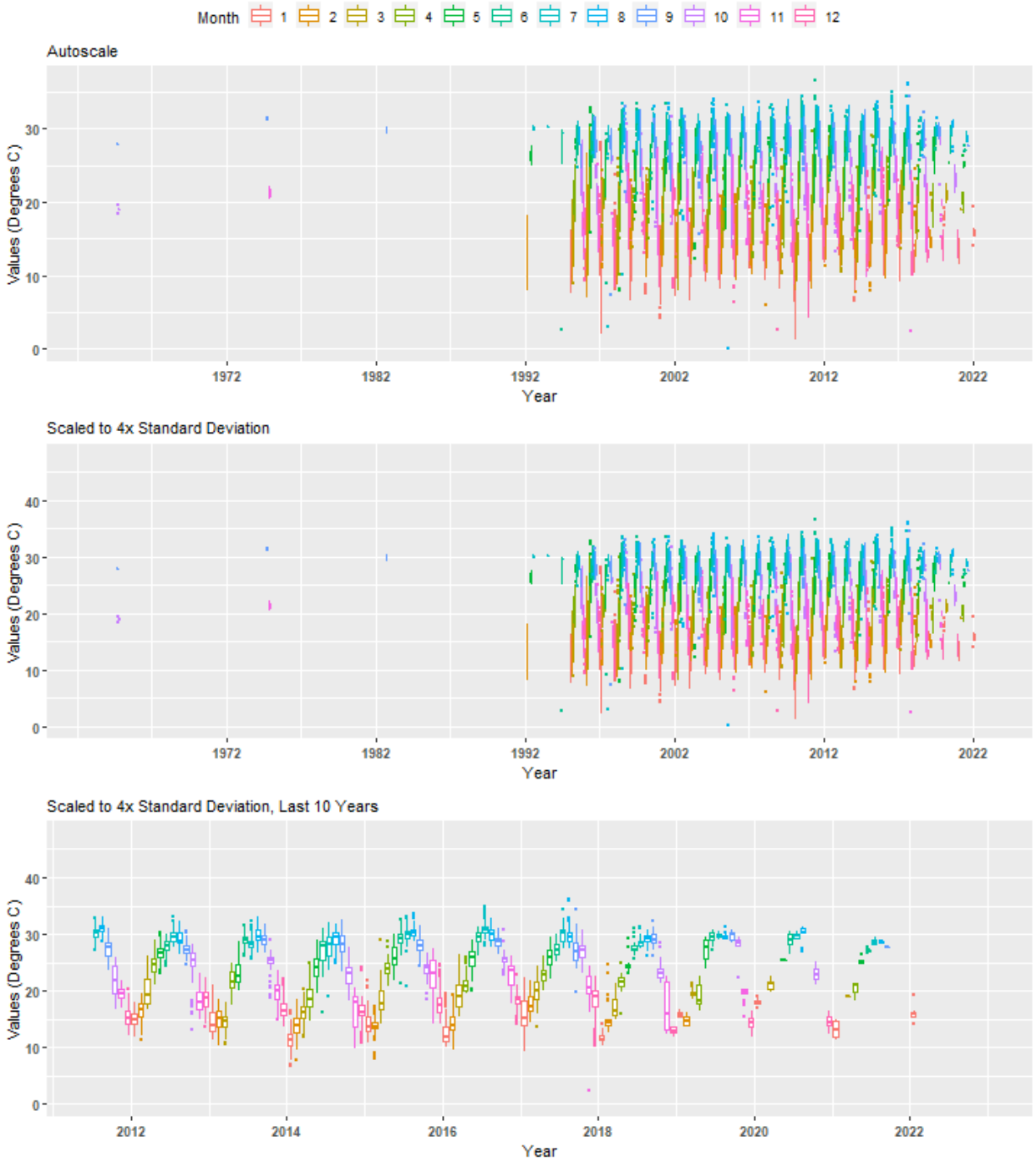


Scaled to 4x Standard Deviation, Last 10 Years



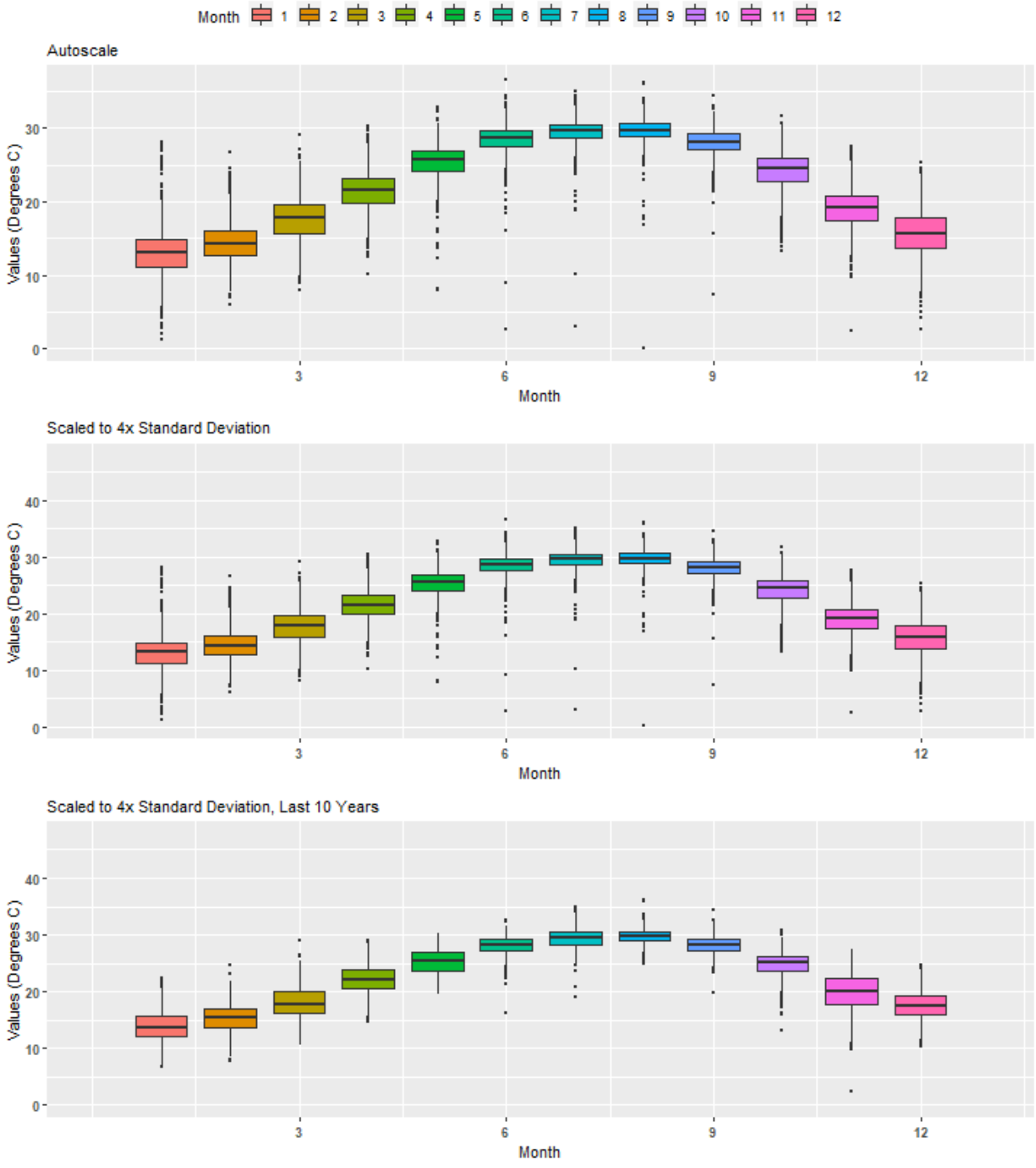
Summary Box Plots for Apalachicola National Estuarine Research Reserve

By Year & Month



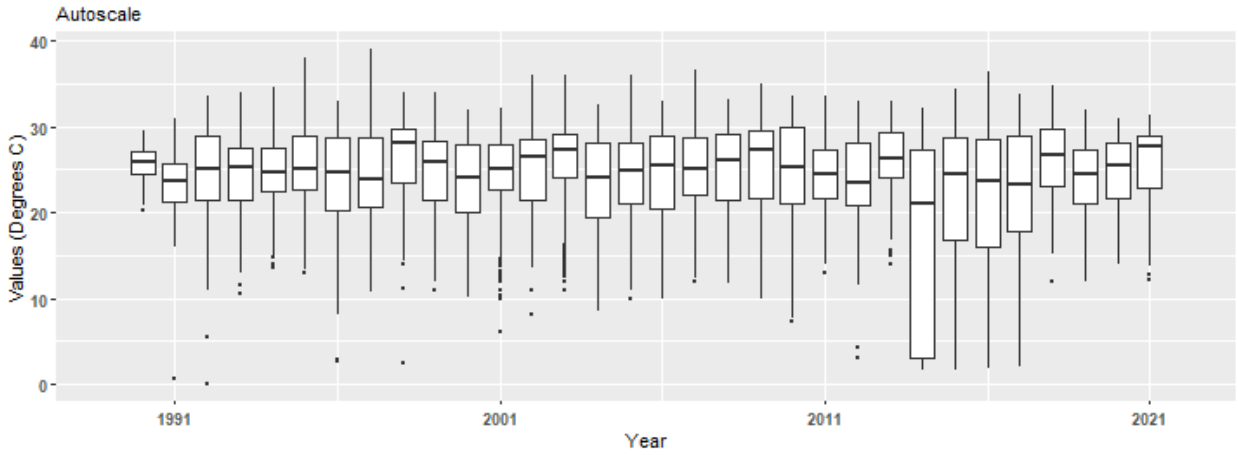
Summary Box Plots for Apalachicola National Estuarine Research Reserve

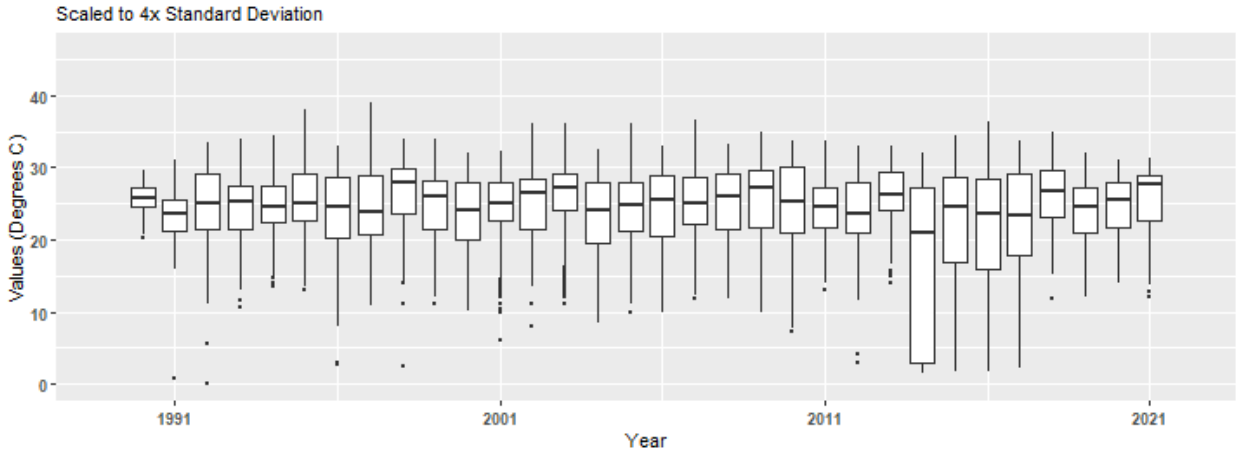
By Month

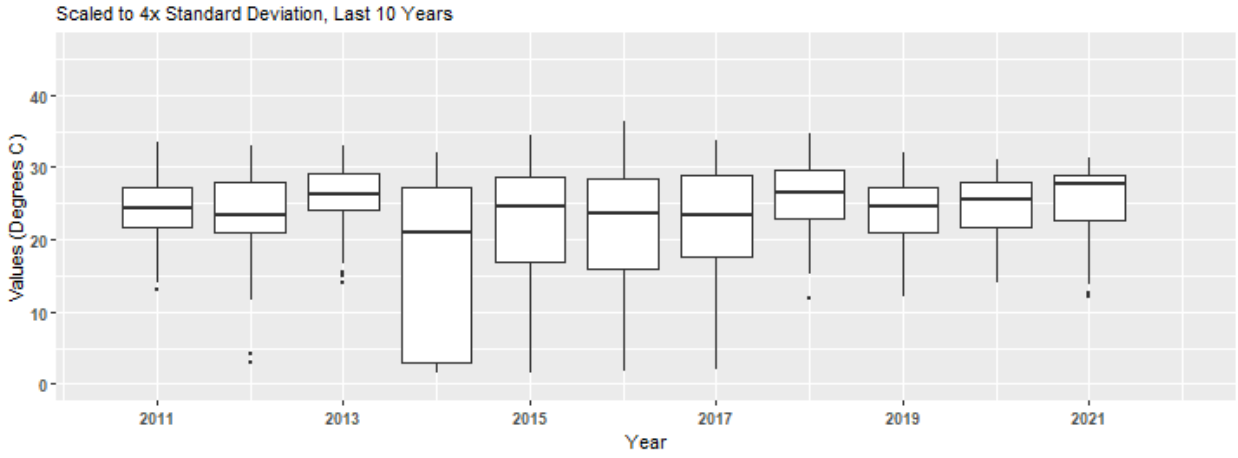


Summary Box Plots for Banana River Aquatic Preserve

By Year

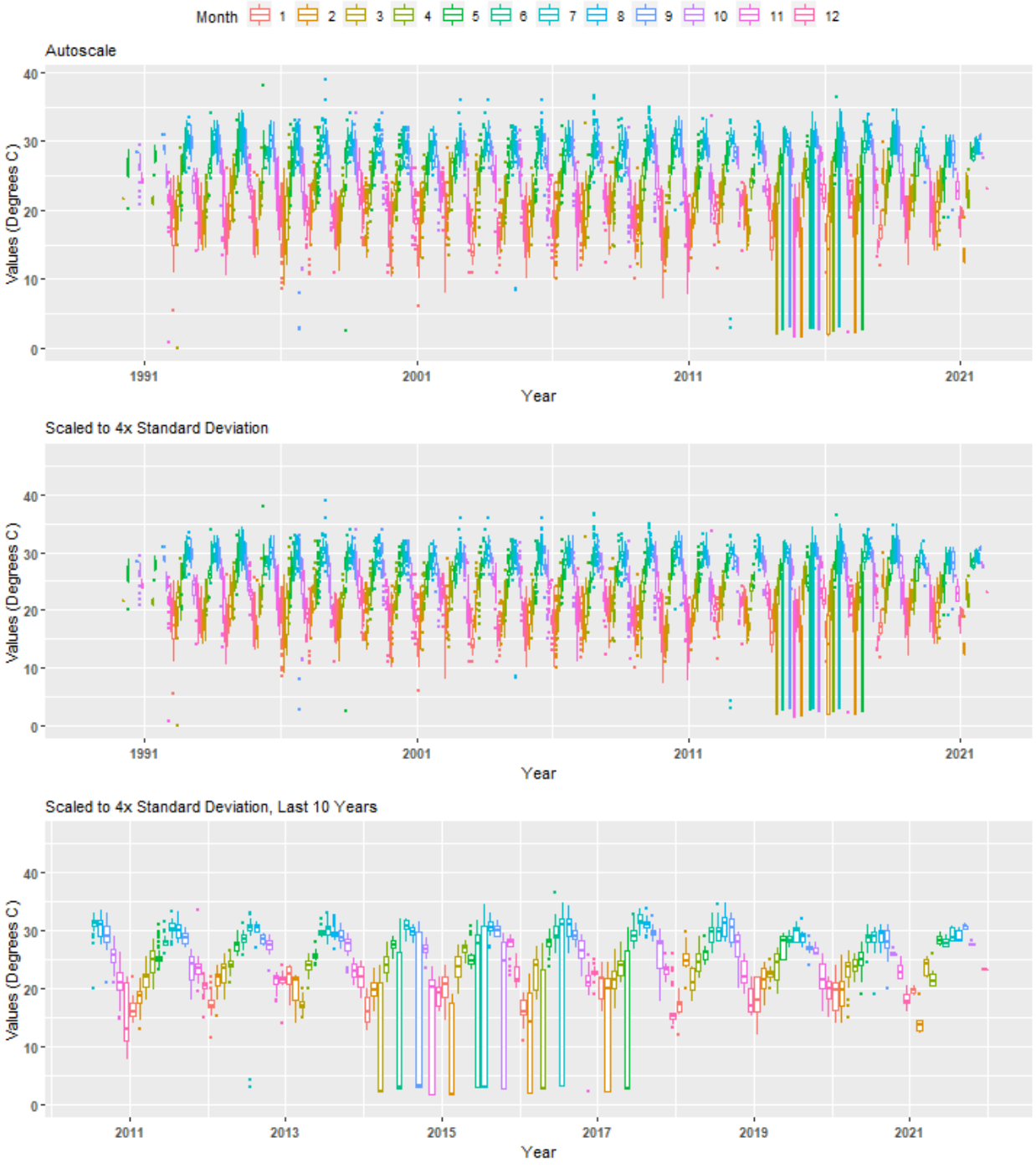






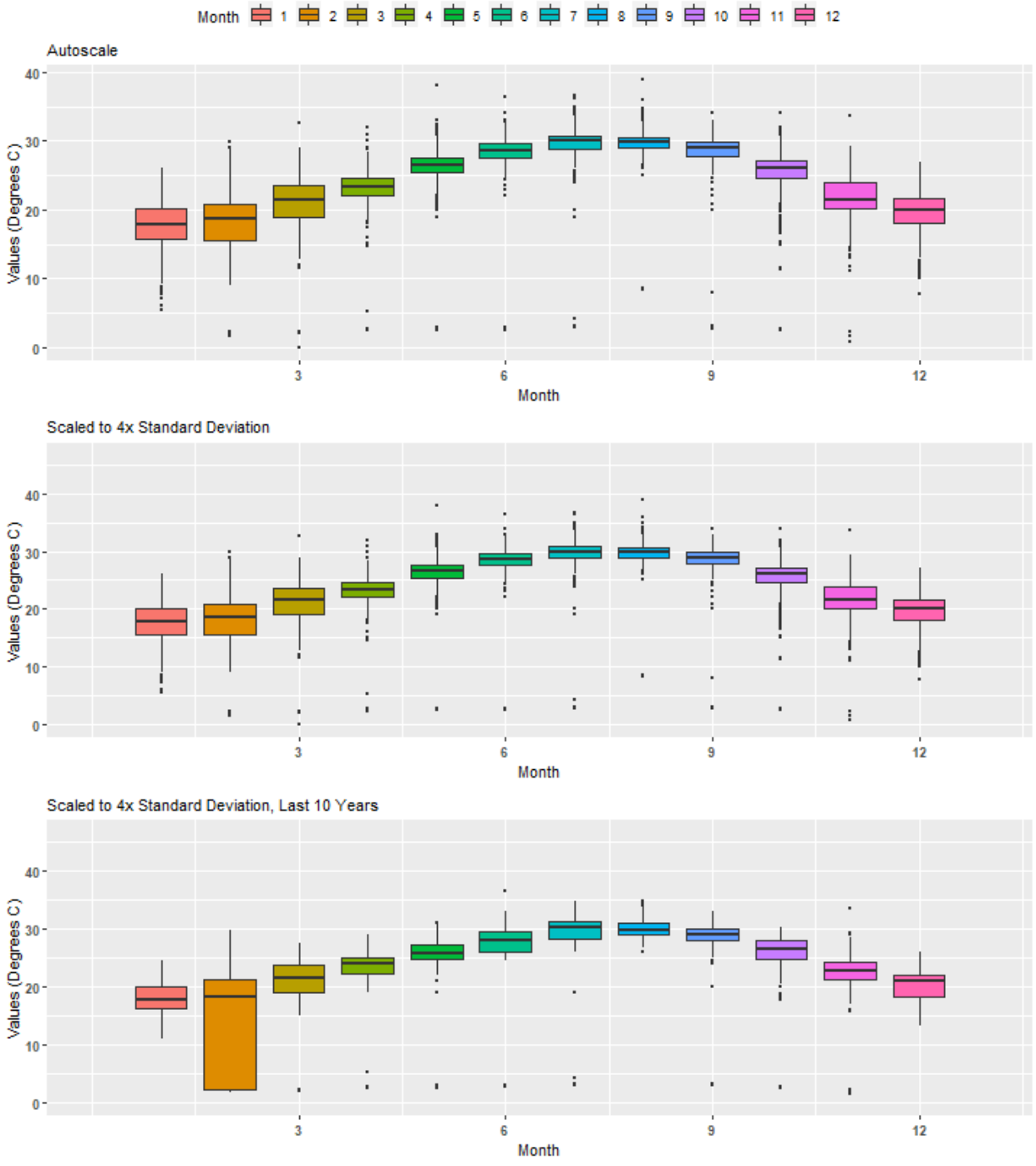
Summary Box Plots for Banana River Aquatic Preserve

By Year & Month



Summary Box Plots for Banana River Aquatic Preserve

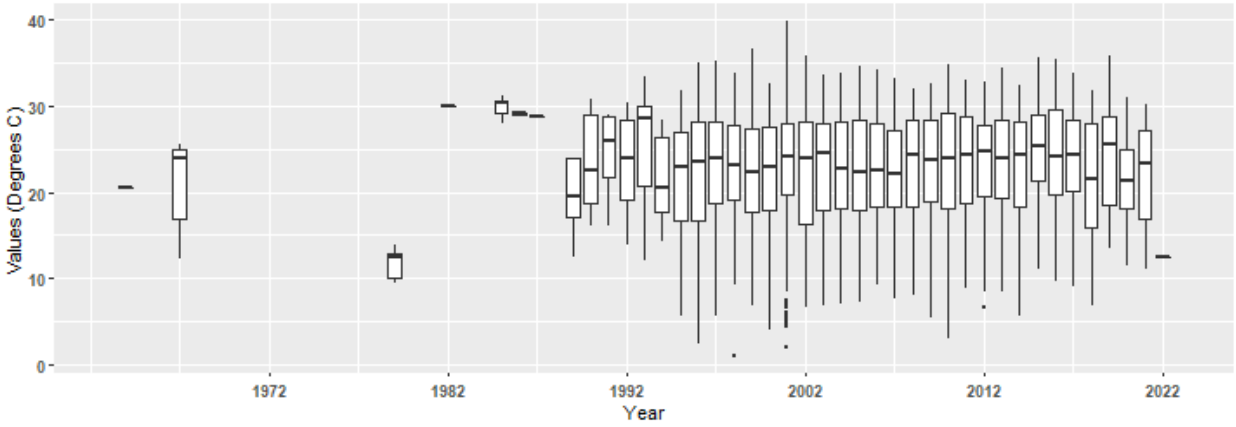
By Month



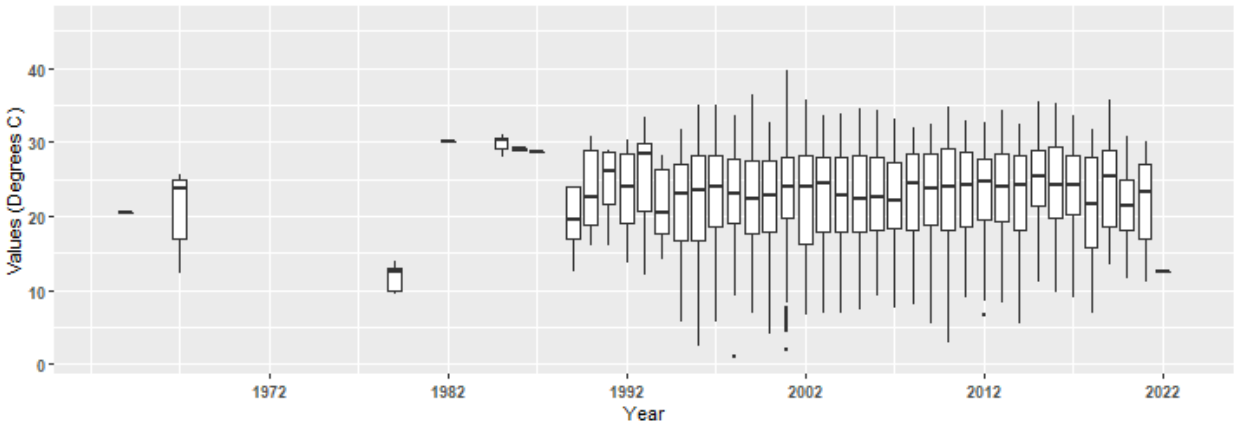
Summary Box Plots for Big Bend Seagrasses Aquatic Preserve

By Year

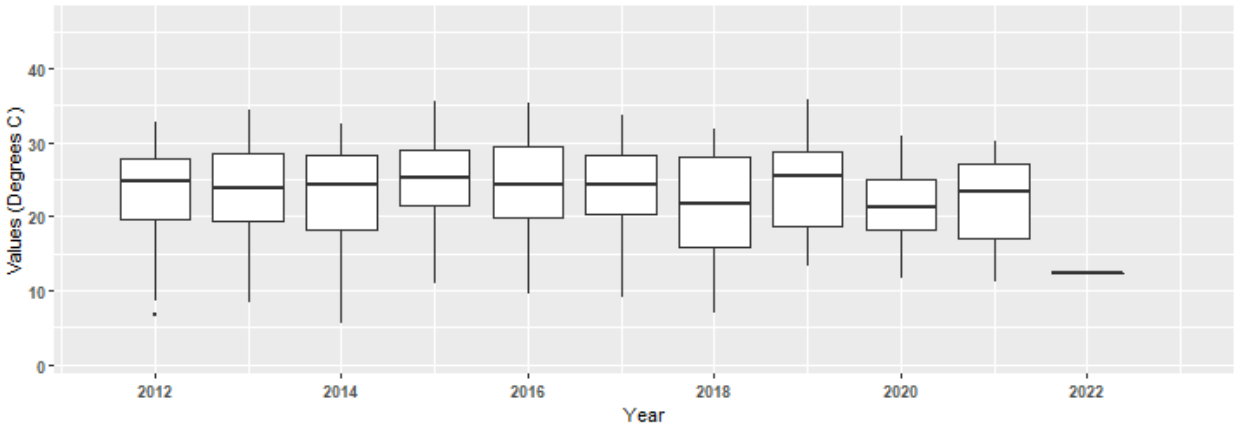
Autoscale



Scaled to 4x Standard Deviation

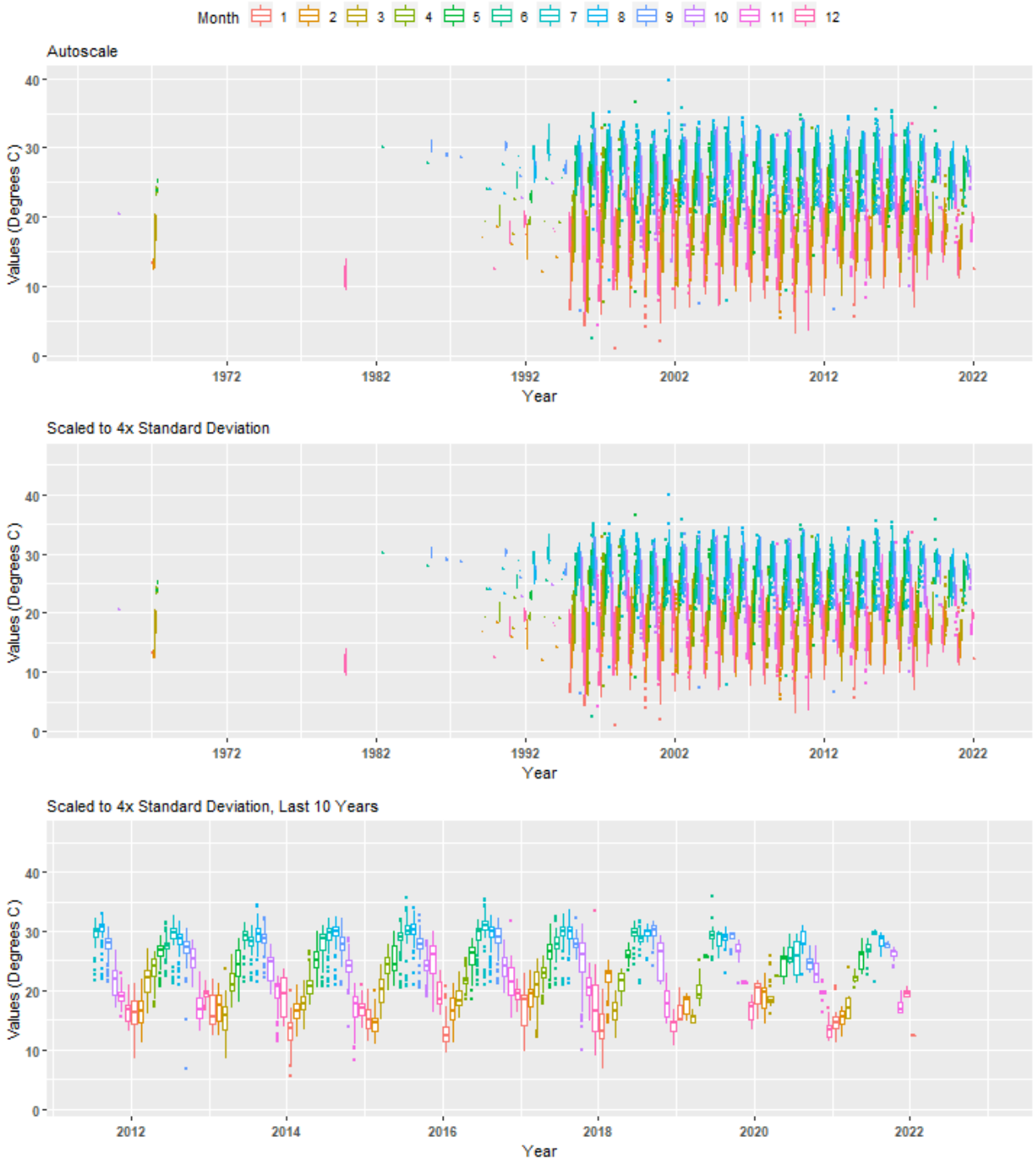


Scaled to 4x Standard Deviation, Last 10 Years



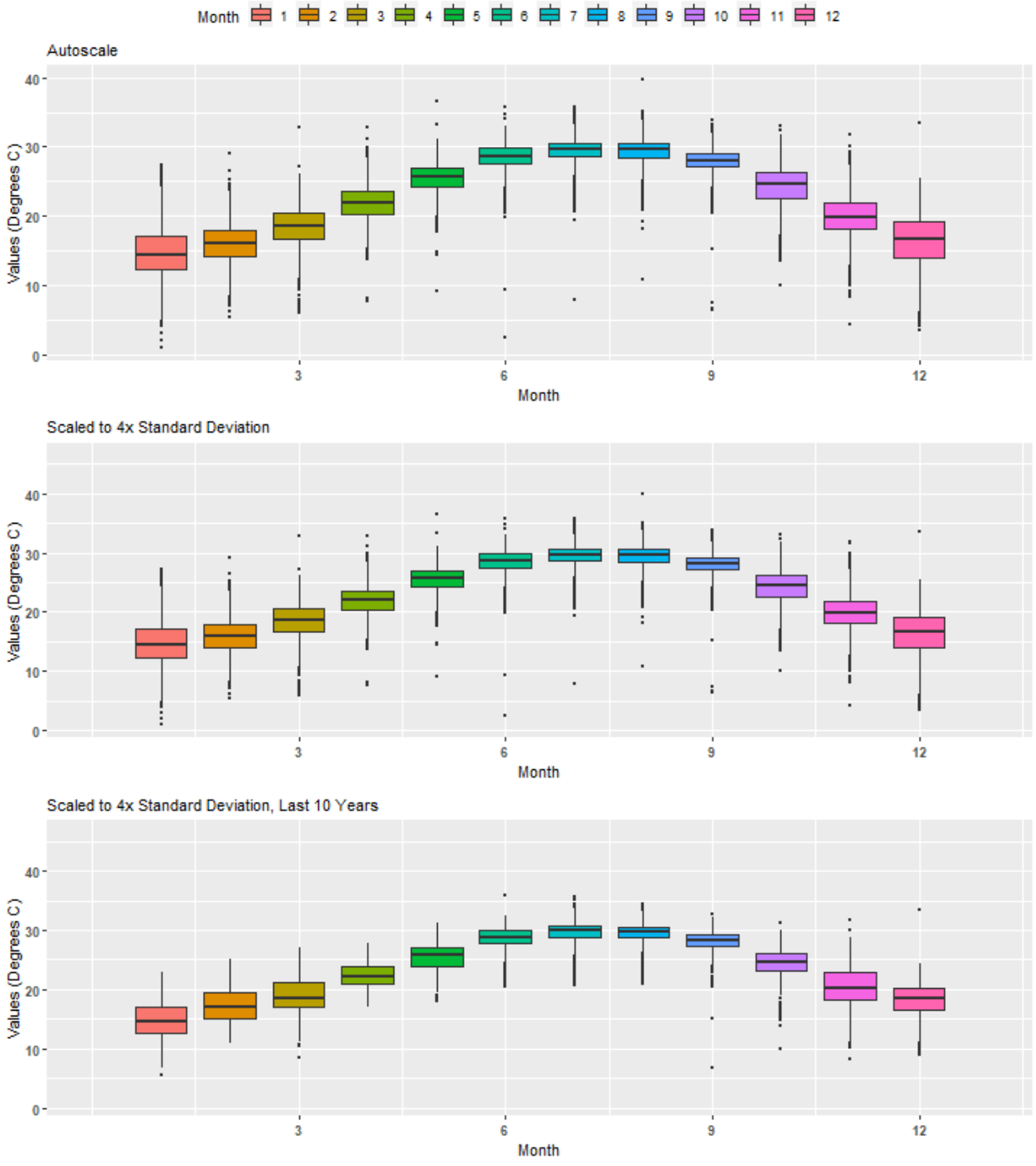
Summary Box Plots for Big Bend Seagrasses Aquatic Preserve

By Year & Month



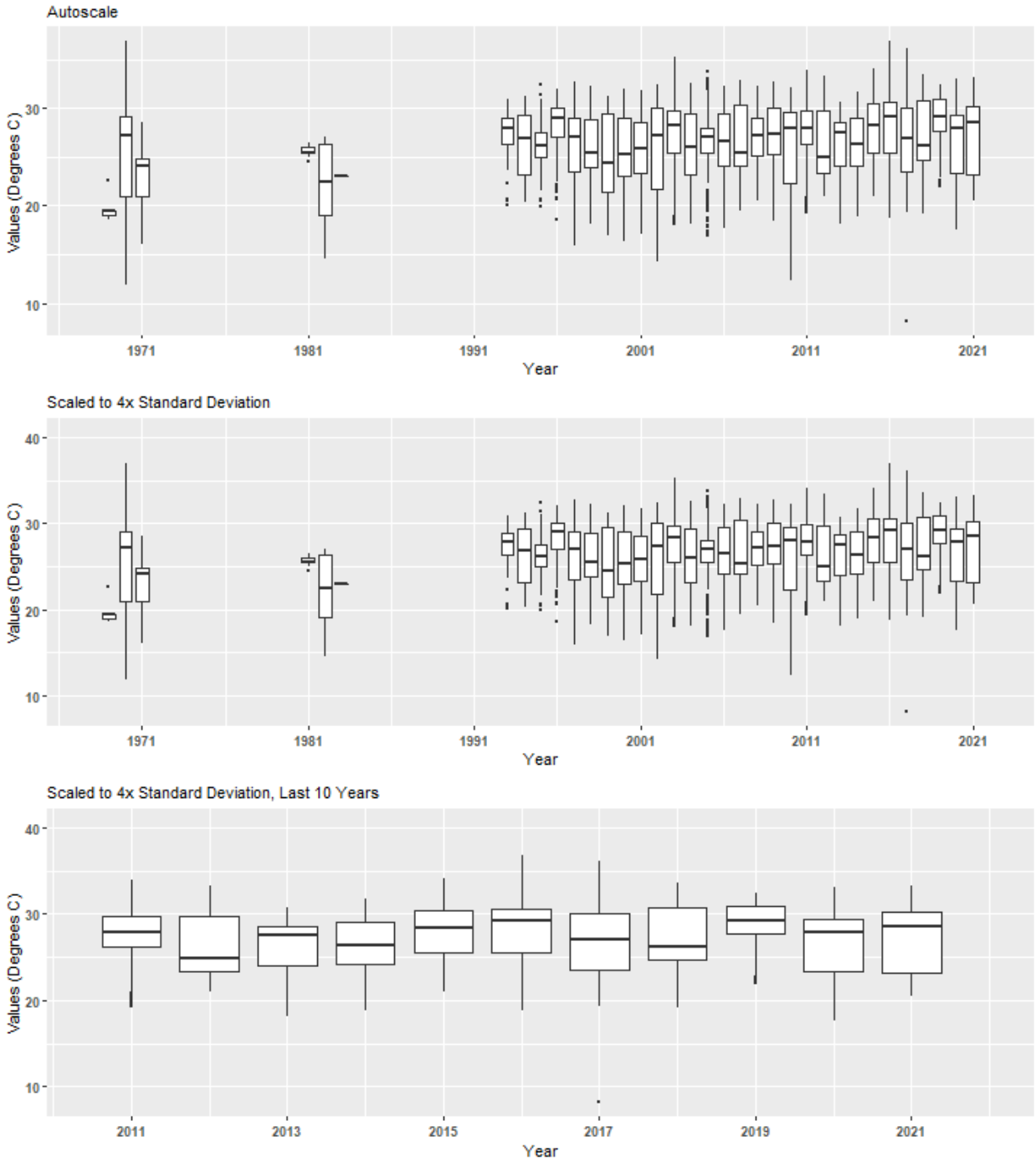
Summary Box Plots for Big Bend Seagrasses Aquatic Preserve

By Month



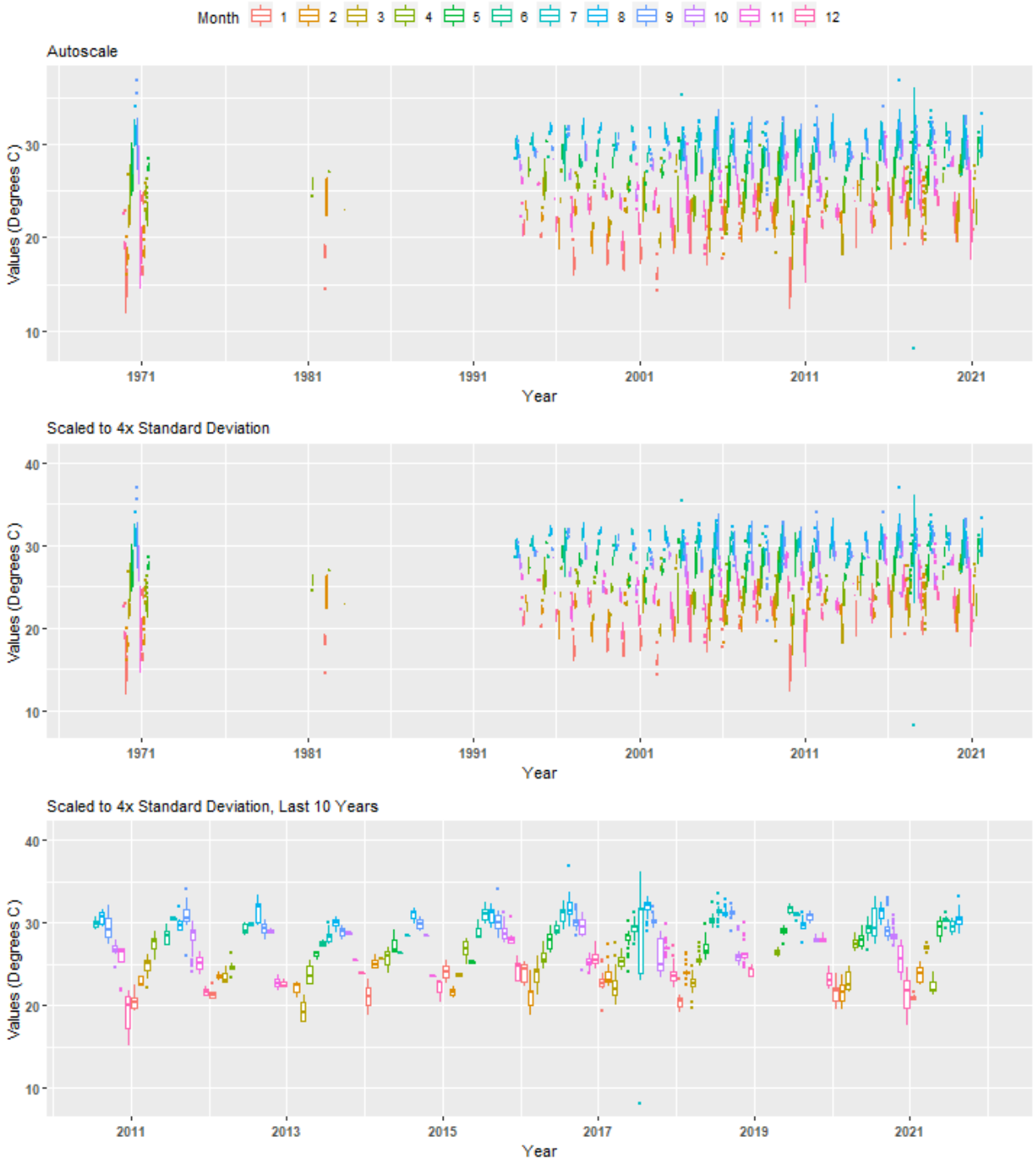
Summary Box Plots for Biscayne Bay Aquatic Preserve

By Year



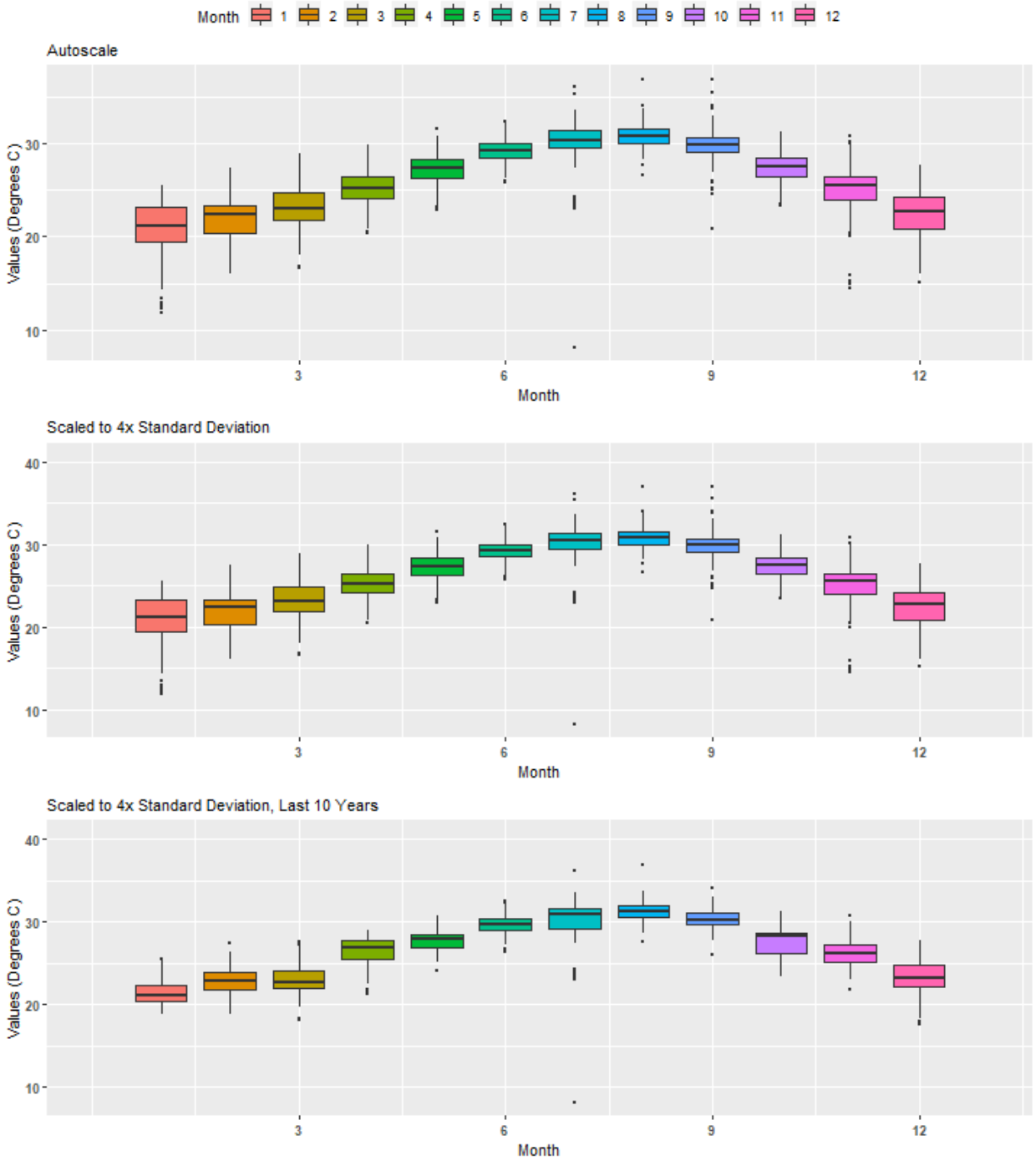
Summary Box Plots for Biscayne Bay Aquatic Preserve

By Year & Month



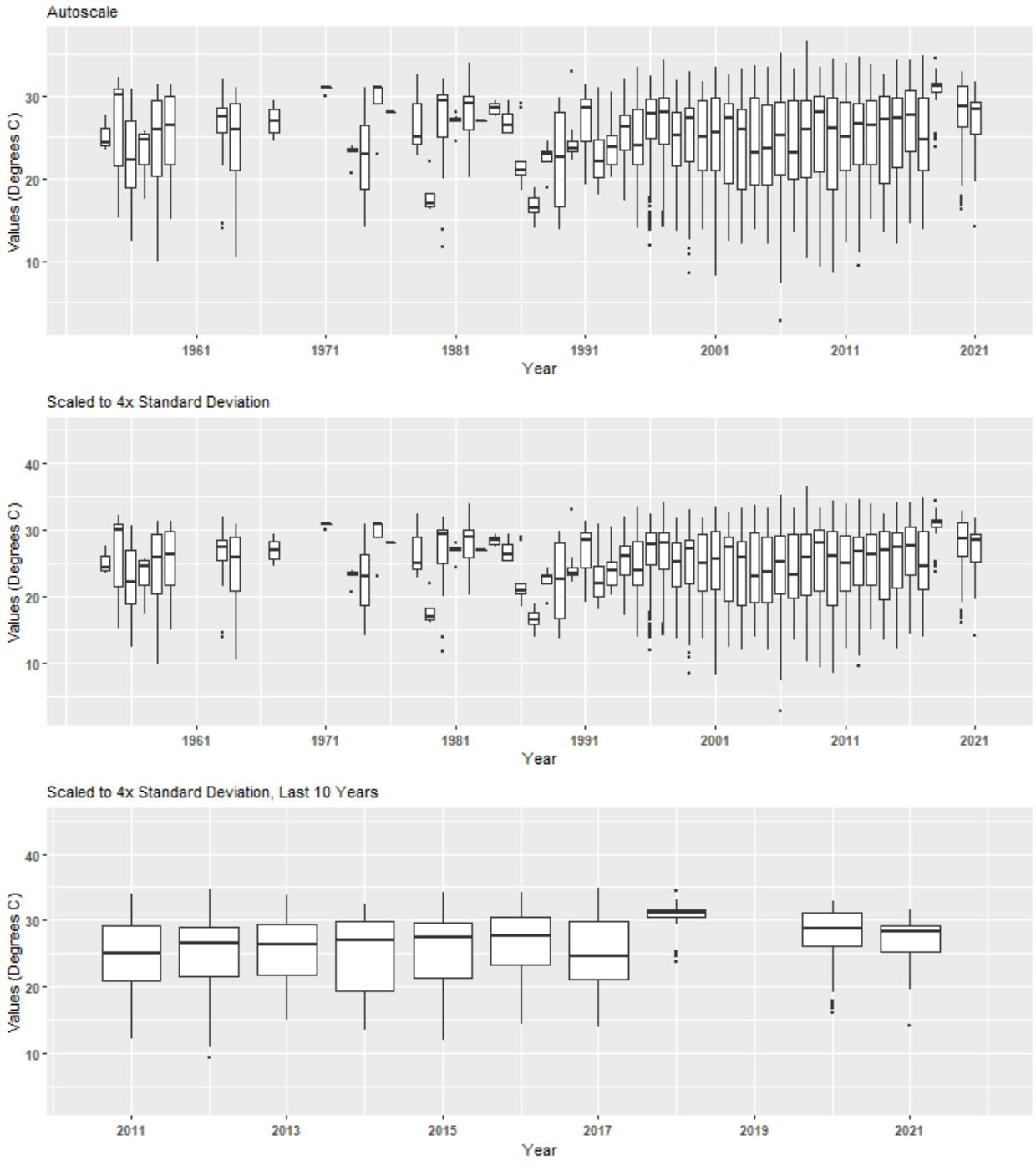
Summary Box Plots for Biscayne Bay Aquatic Preserve

By Month



Summary Box Plots for Boca Ciega Bay Aquatic Preserve

By Year

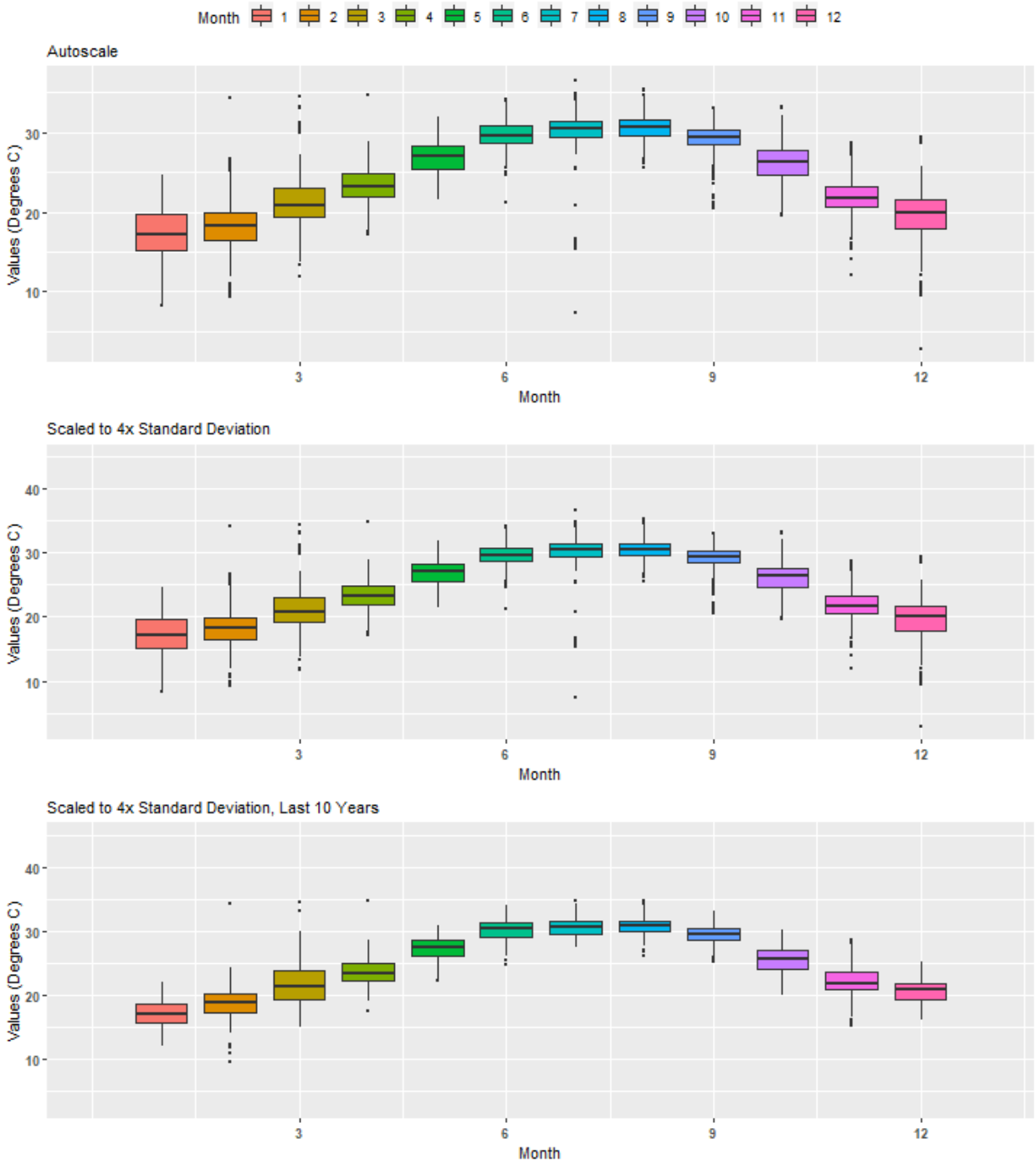


Summary Box Plots for Boca Ciega Bay Aquatic Preserve

By Year & Month

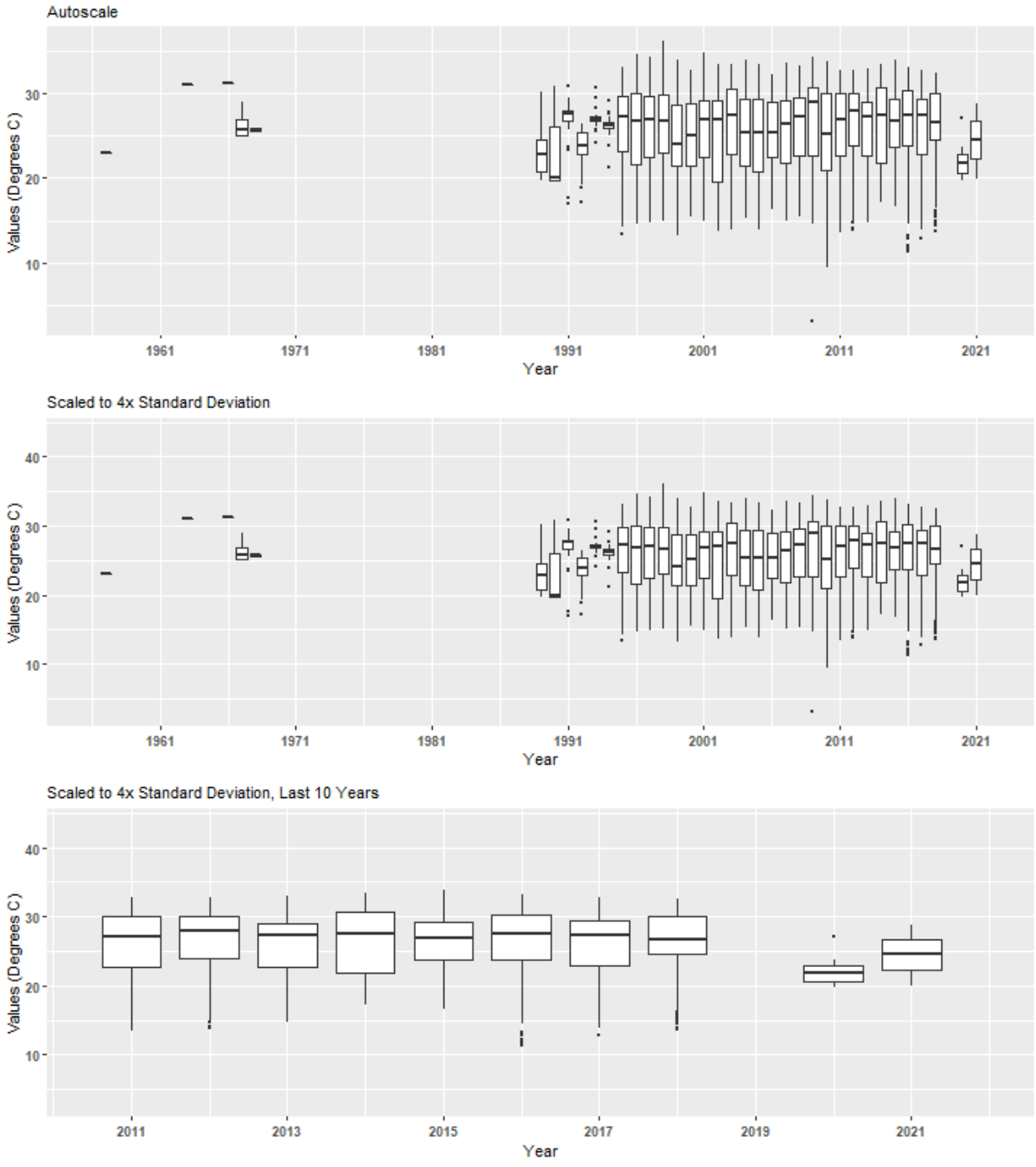
Summary Box Plots for Boca Ciega Bay Aquatic Preserve

By Month



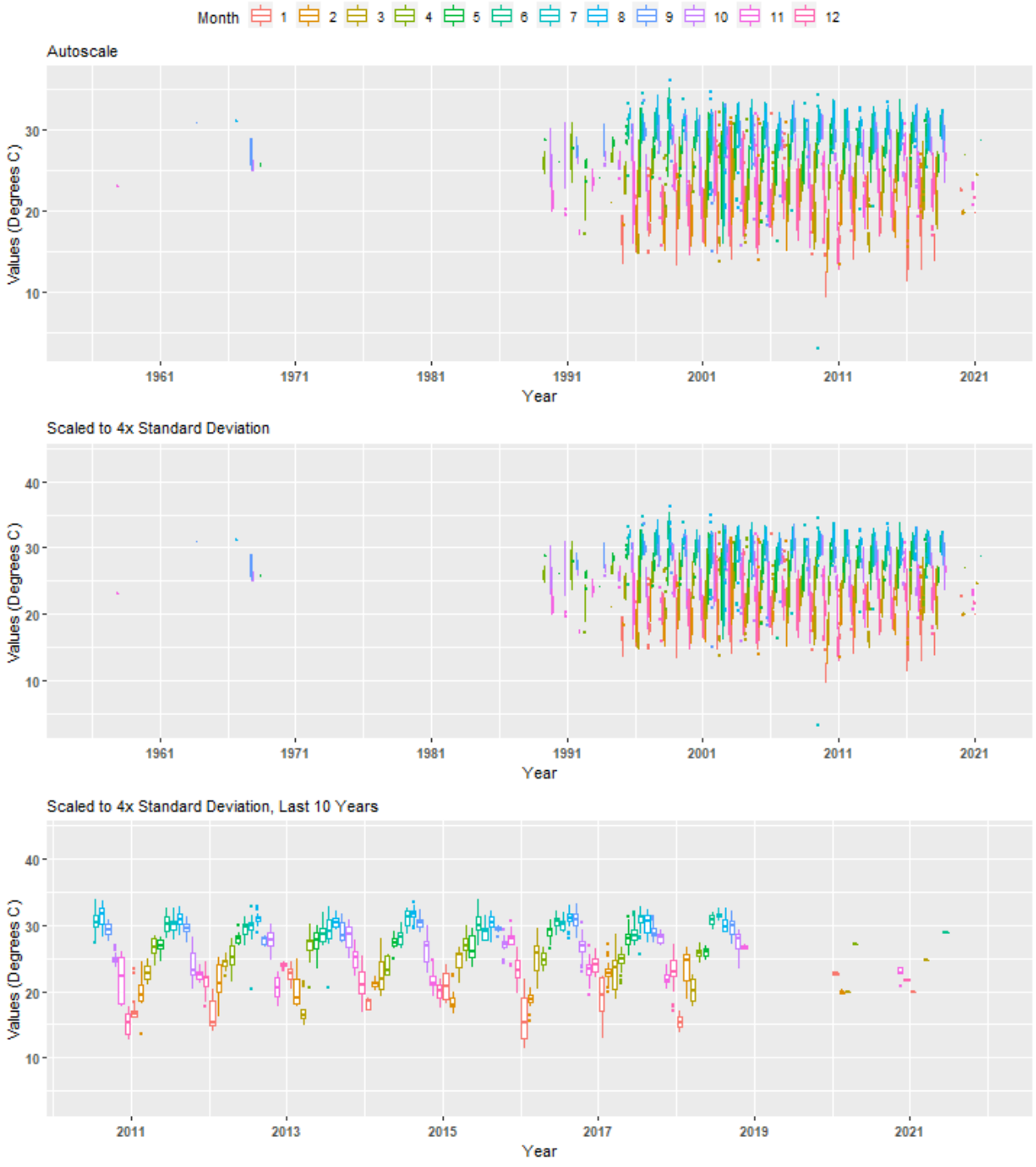
Summary Box Plots for Cape Haze Aquatic Preserve

By Year



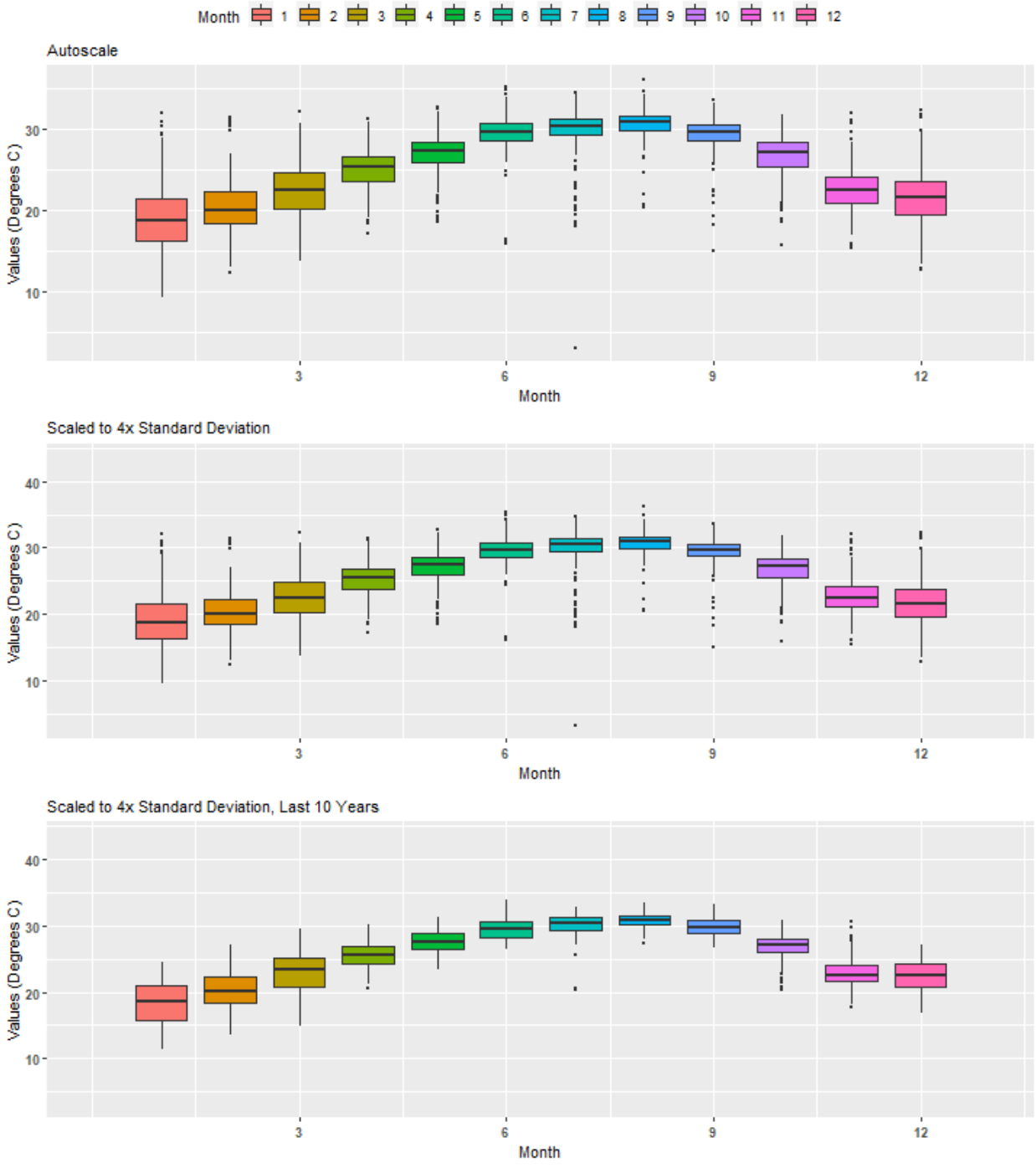
Summary Box Plots for Cape Haze Aquatic Preserve

By Year & Month



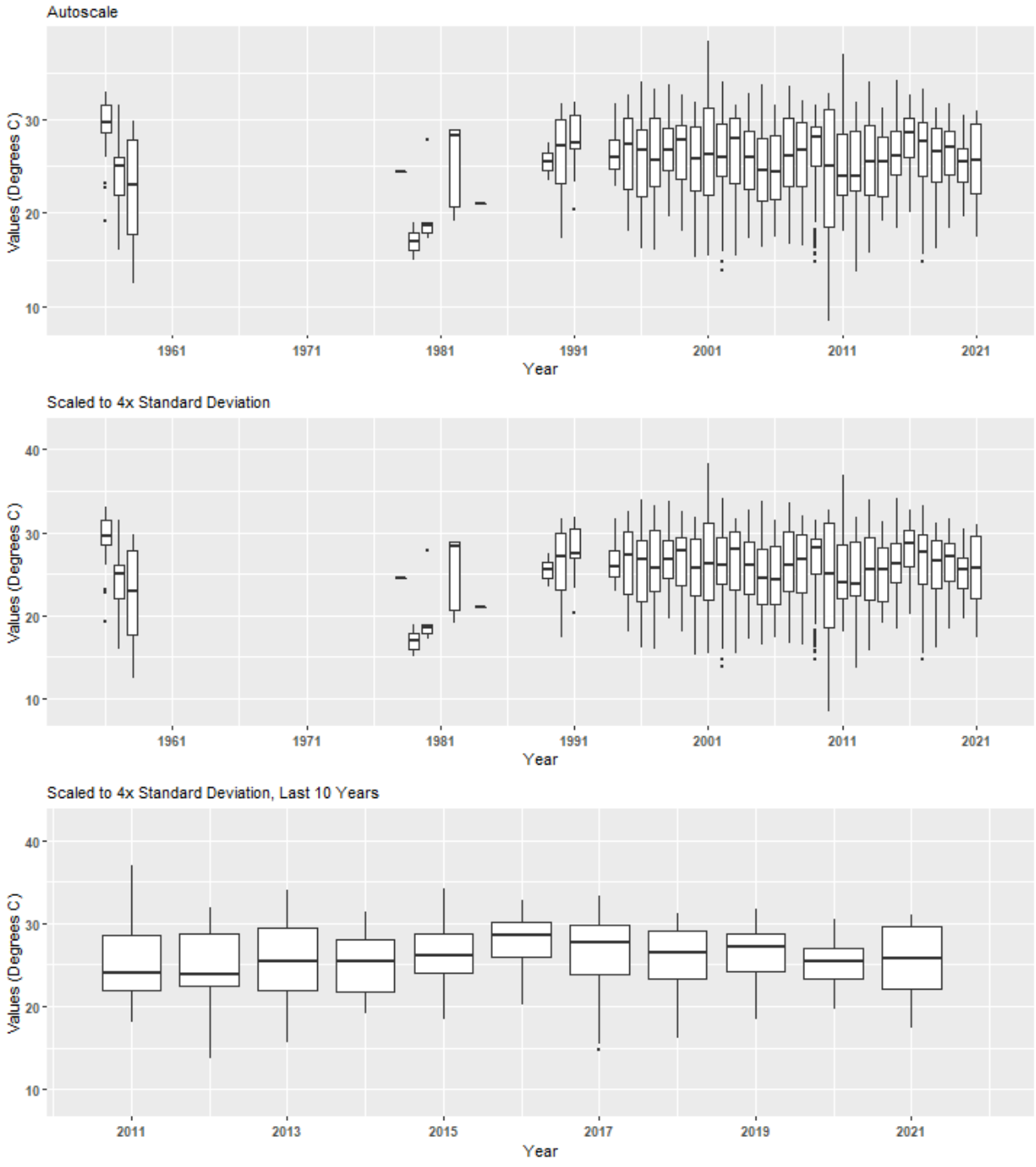
Summary Box Plots for Cape Haze Aquatic Preserve

By Month



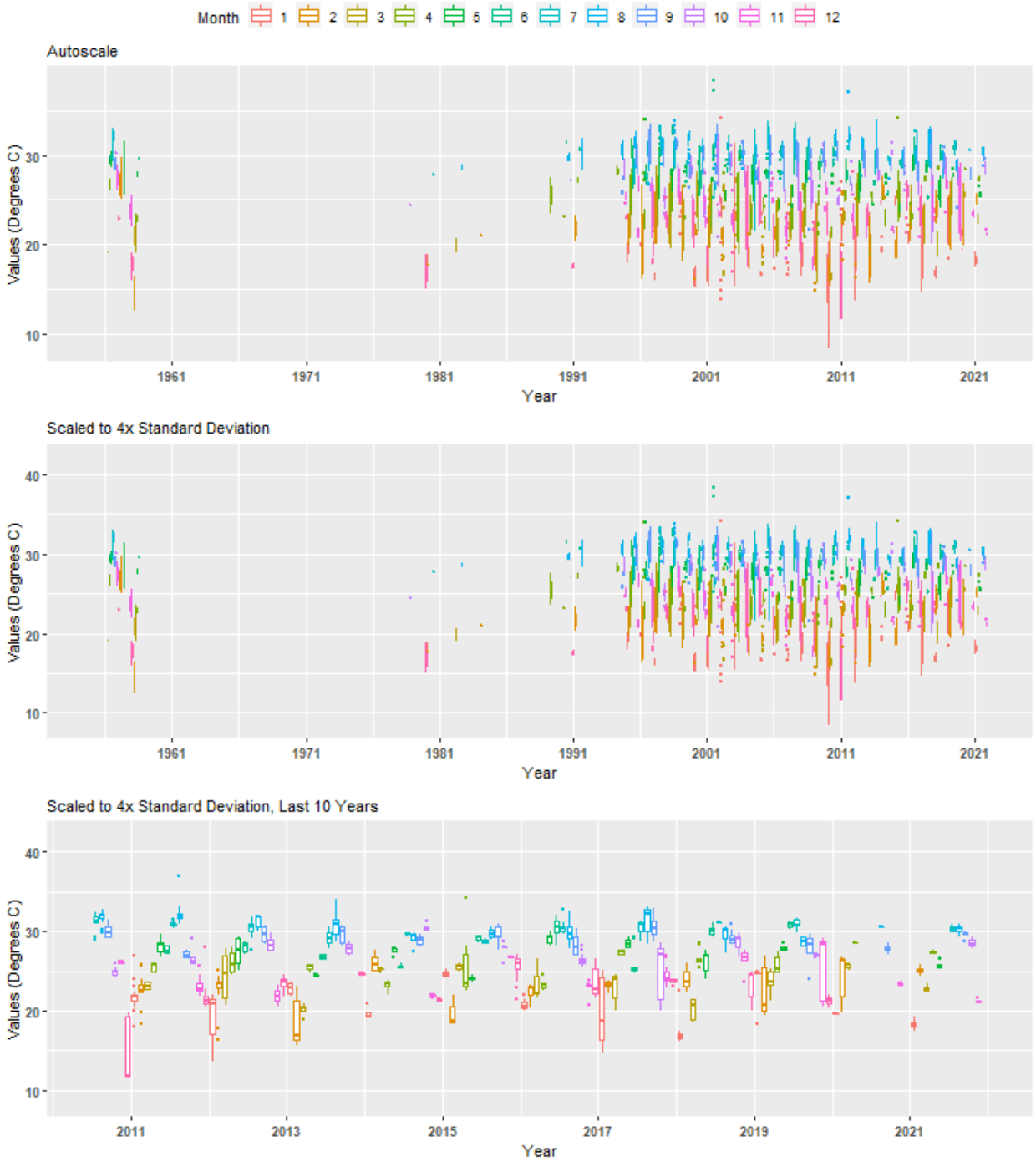
Summary Box Plots for Cape Romano-Ten Thousand Islands Aquatic Preserve

By Year



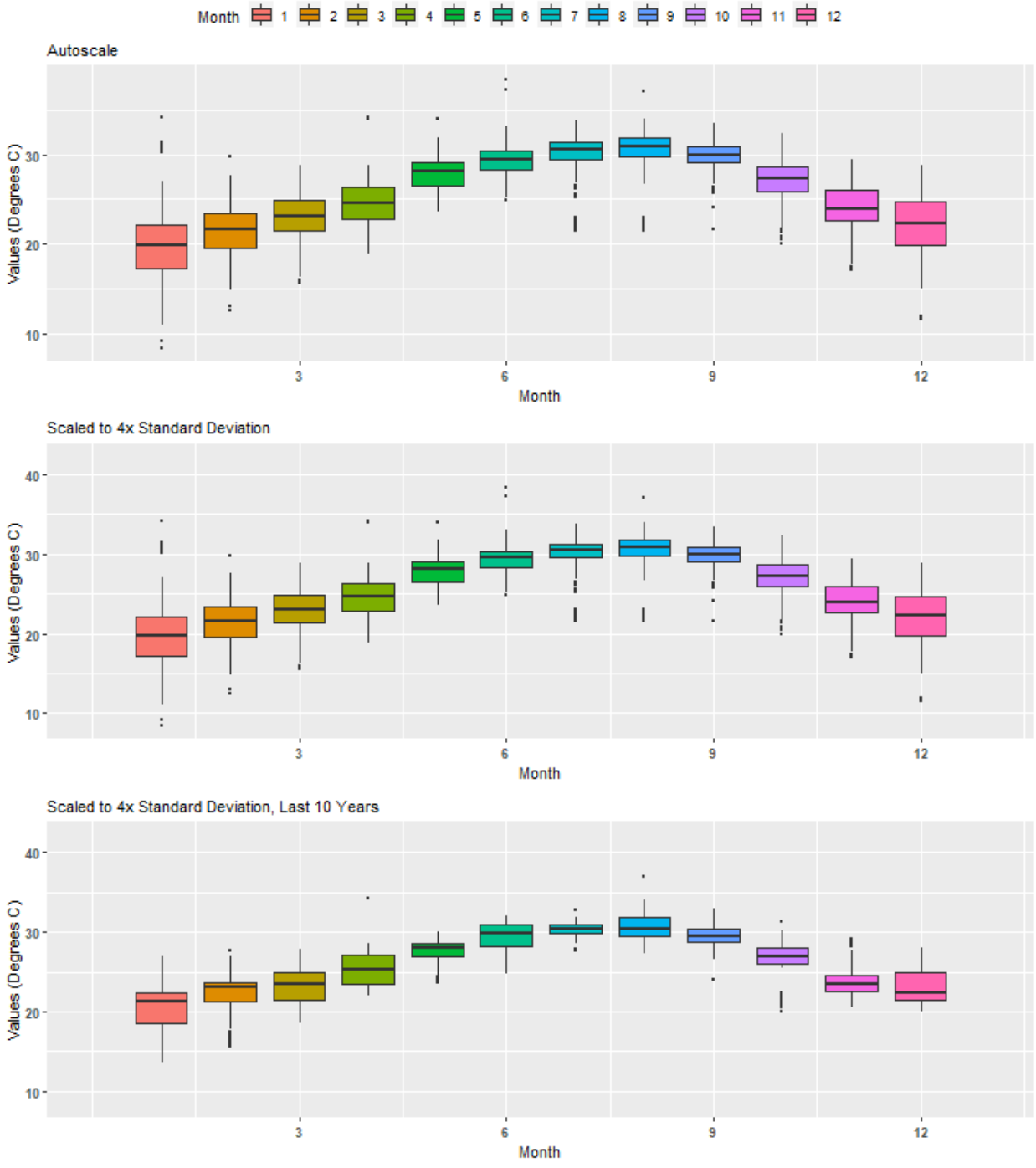
Summary Box Plots for Cape Romano-Ten Thousand Islands Aquatic Preserve

By Year & Month



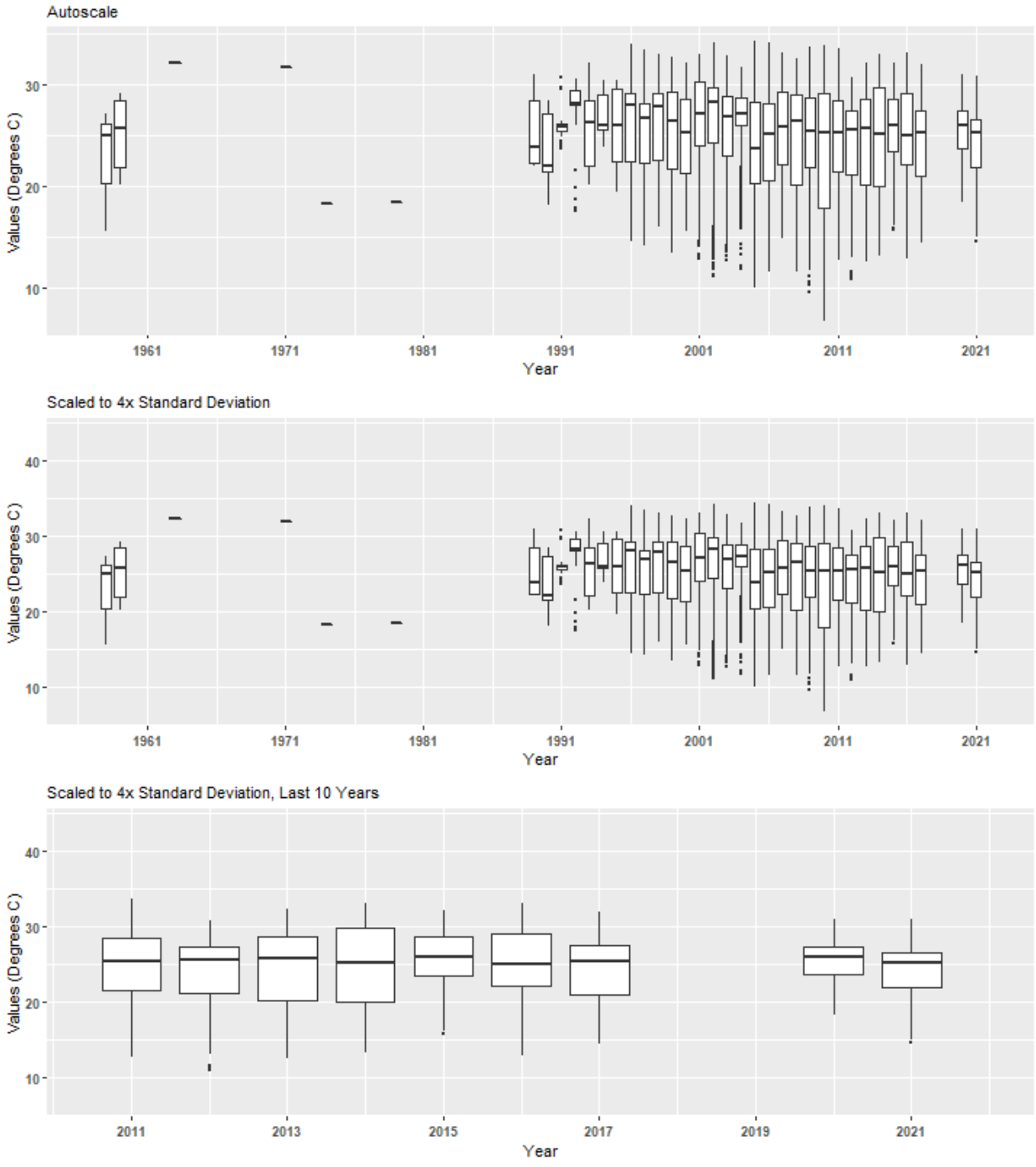
Summary Box Plots for Cape Romano-Ten Thousand Islands Aquatic Preserve

By Month



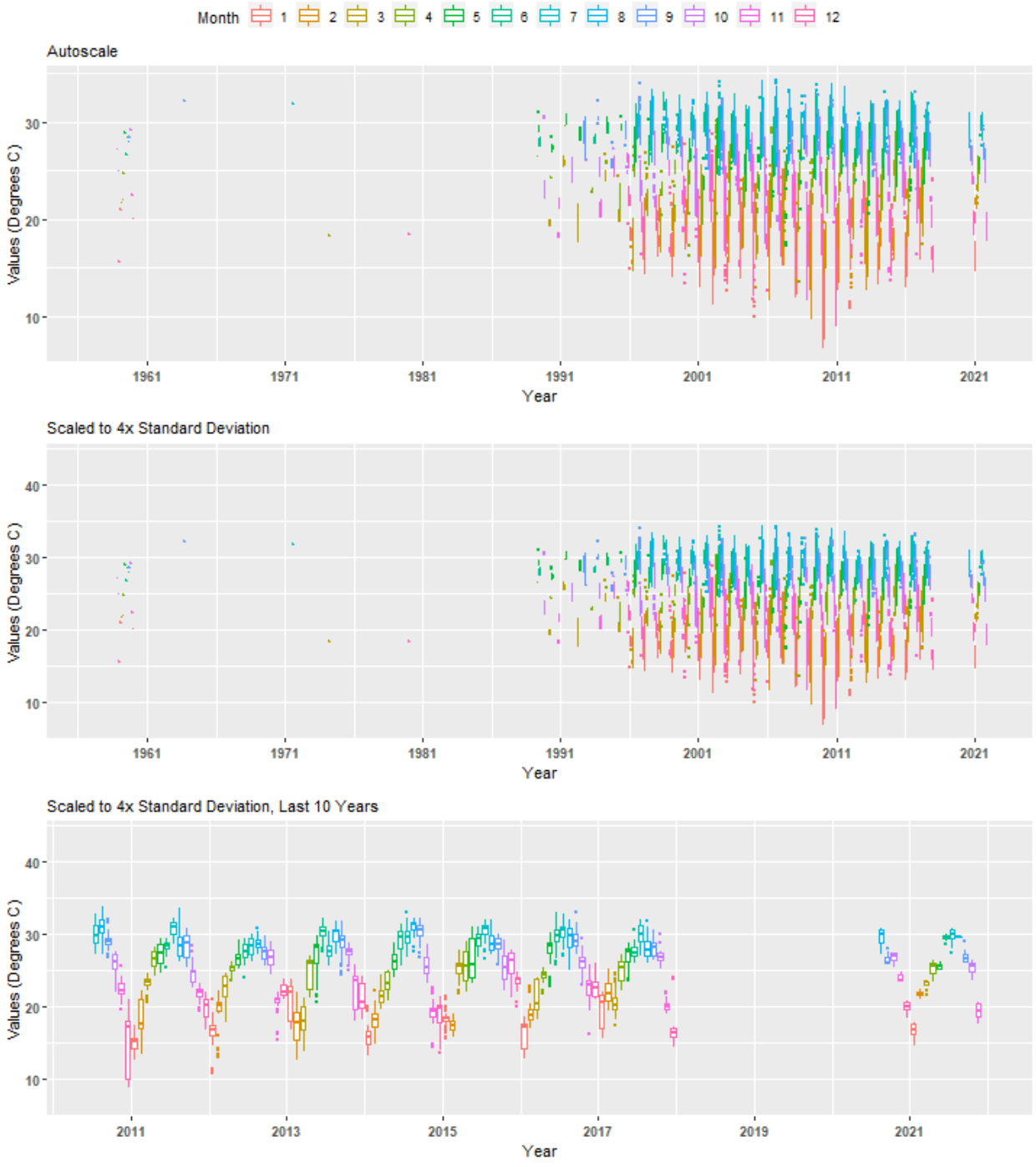
Summary Box Plots for Cockroach Bay Aquatic Preserve

By Year



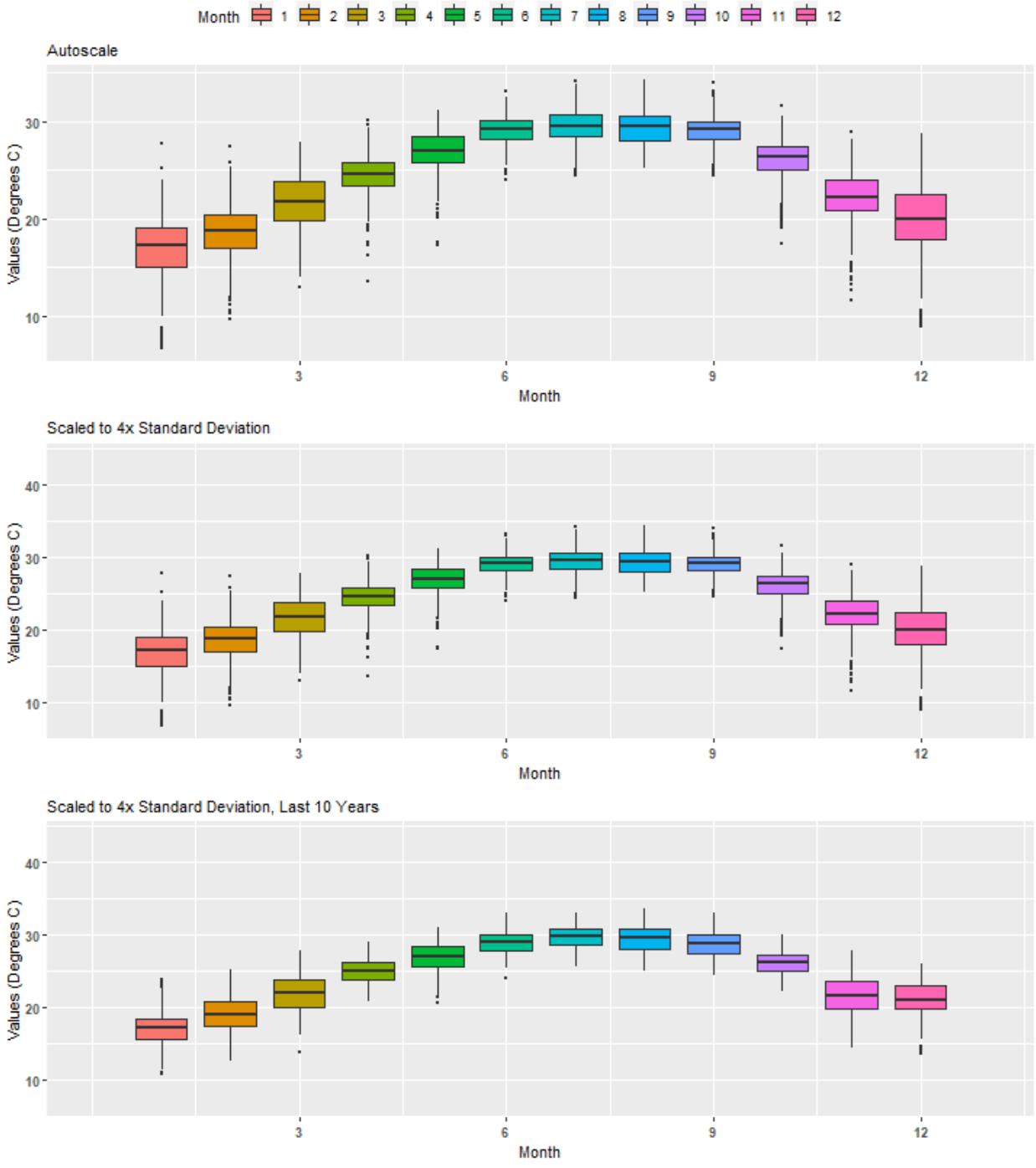
Summary Box Plots for Cockroach Bay Aquatic Preserve

By Year & Month



Summary Box Plots for Cockroach Bay Aquatic Preserve

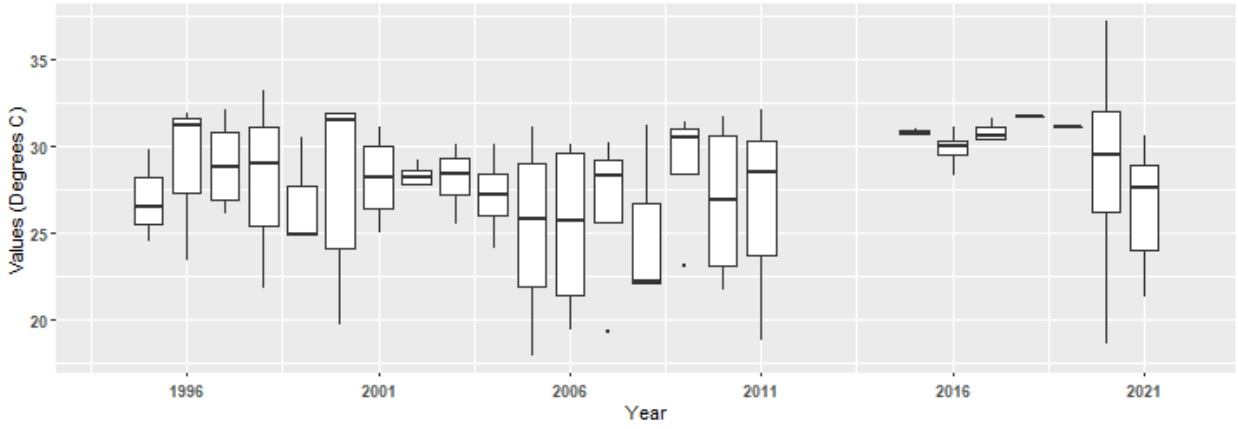
By Month



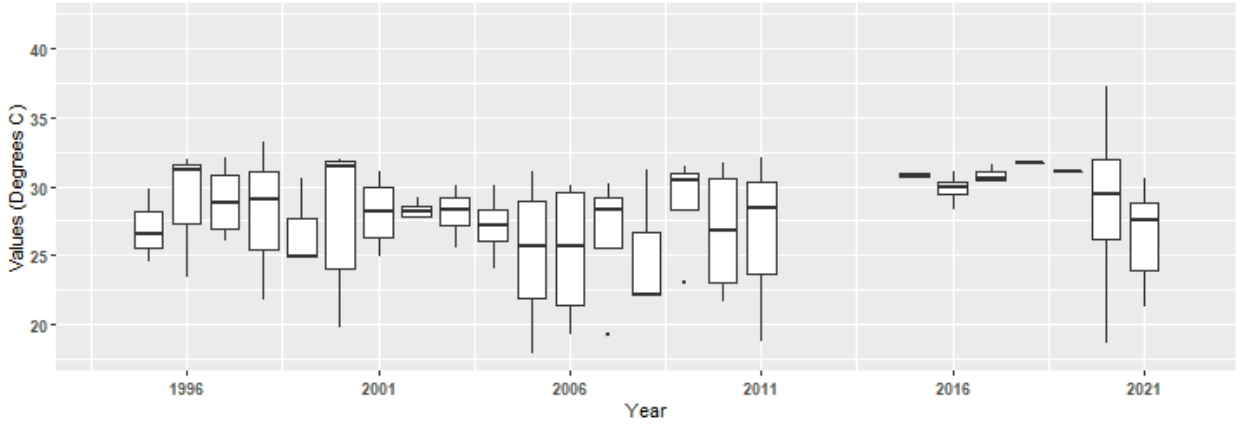
Summary Box Plots for Coupon Bight Aquatic Preserve

By Year

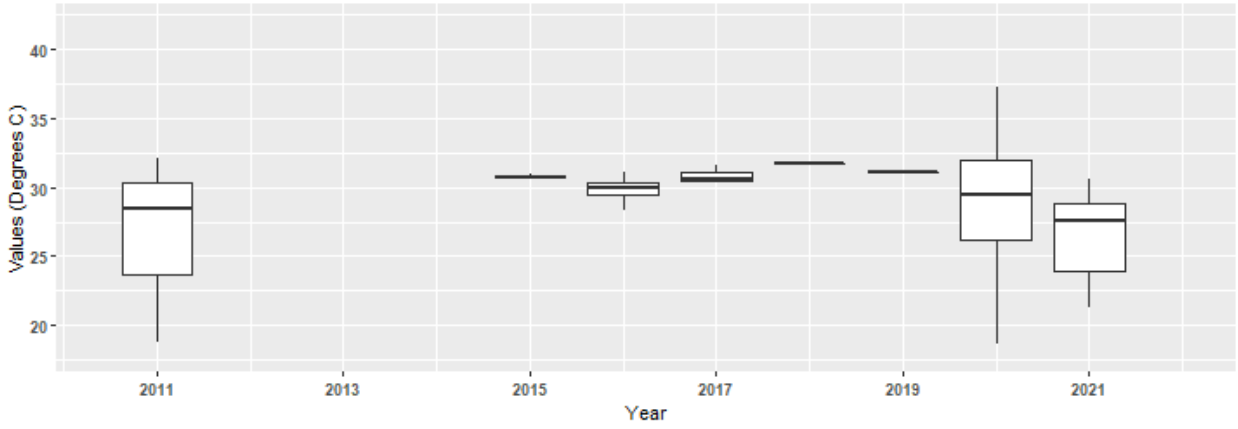
Autoscale



Scaled to 4x Standard Deviation



Scaled to 4x Standard Deviation, Last 10 Years

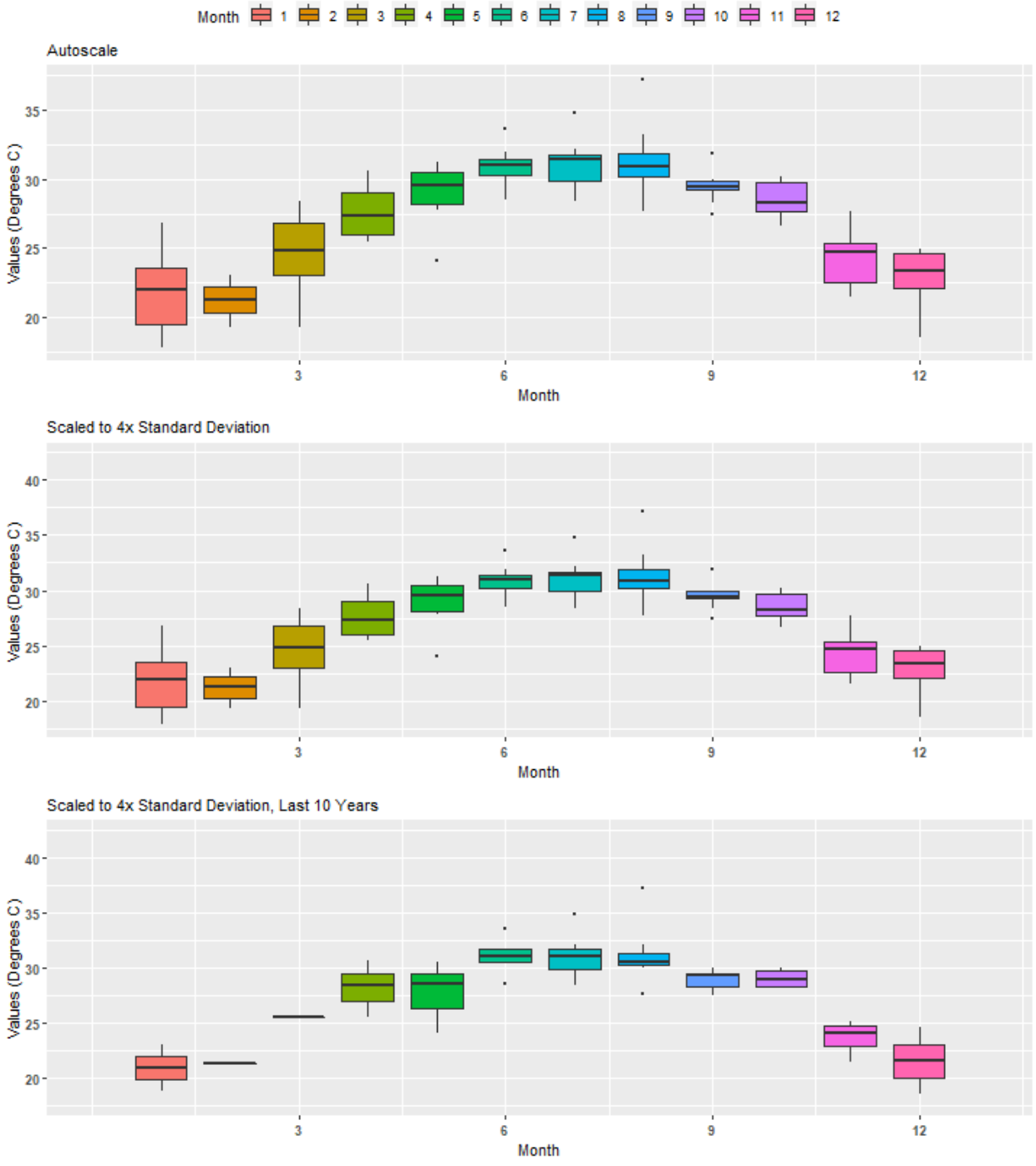


Summary Box Plots for Coupon Bight Aquatic Preserve

By Year & Month

Summary Box Plots for Coupon Bight Aquatic Preserve

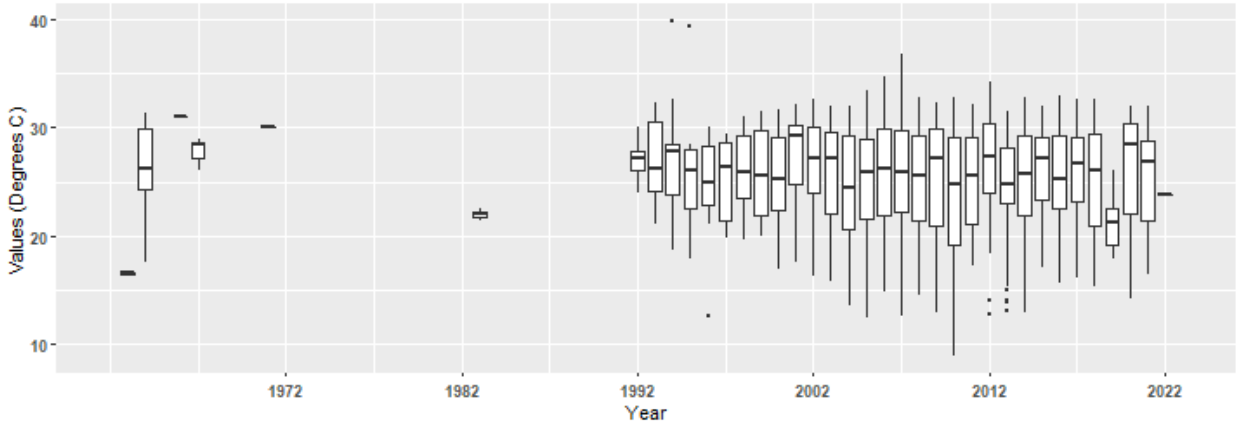
By Month



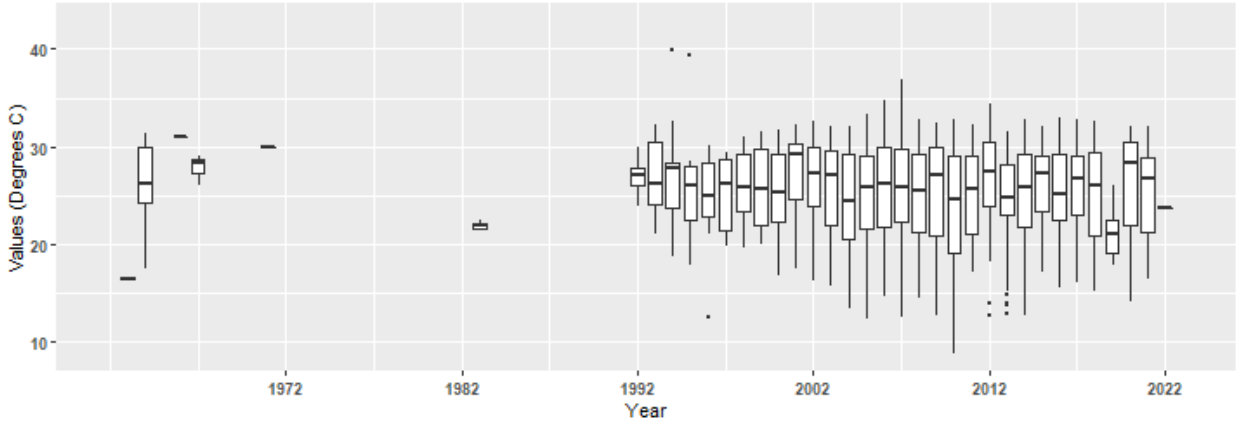
Summary Box Plots for Estero Bay Aquatic Preserve

By Year

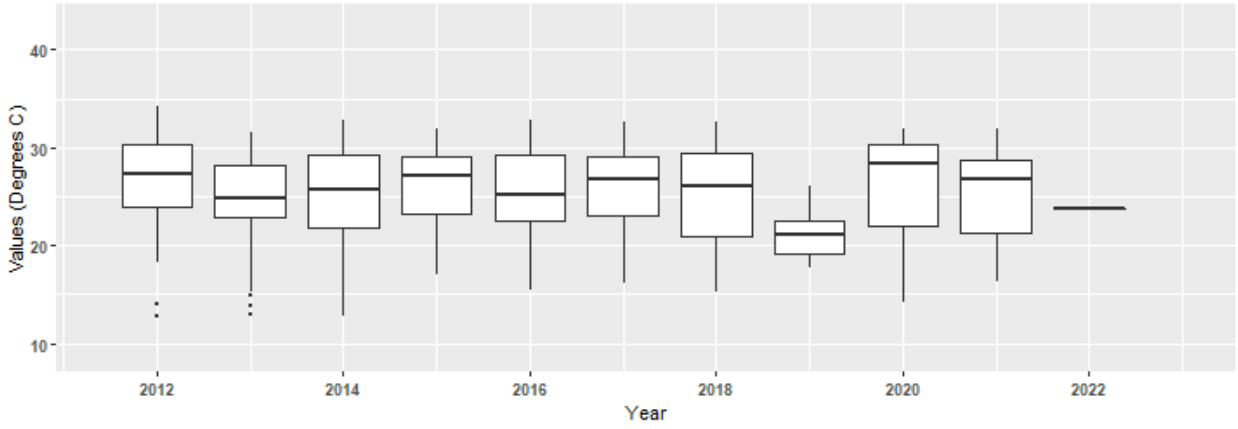
Autoscale



Scaled to 4x Standard Deviation

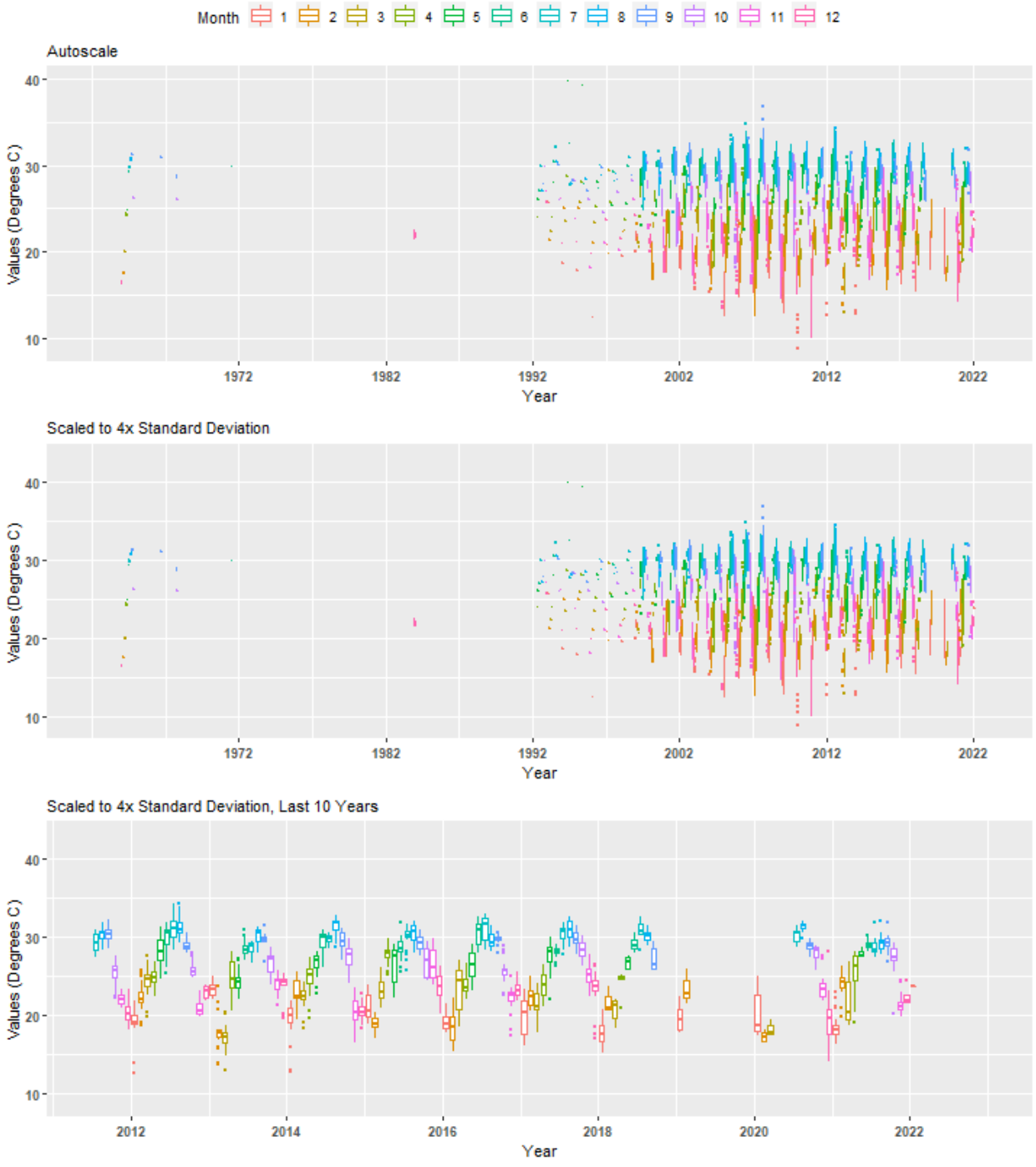


Scaled to 4x Standard Deviation, Last 10 Years



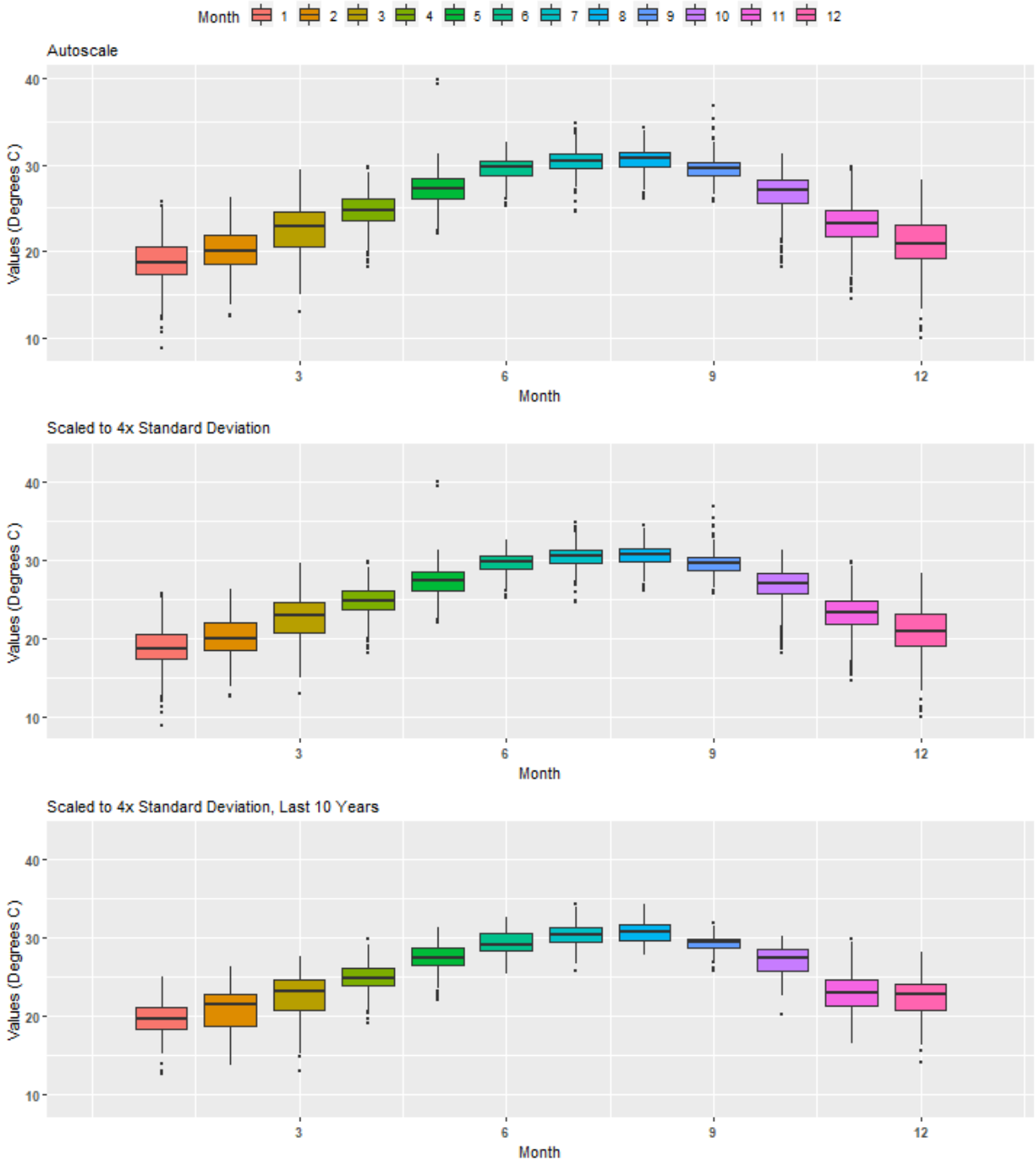
Summary Box Plots for Estero Bay Aquatic Preserve

By Year & Month



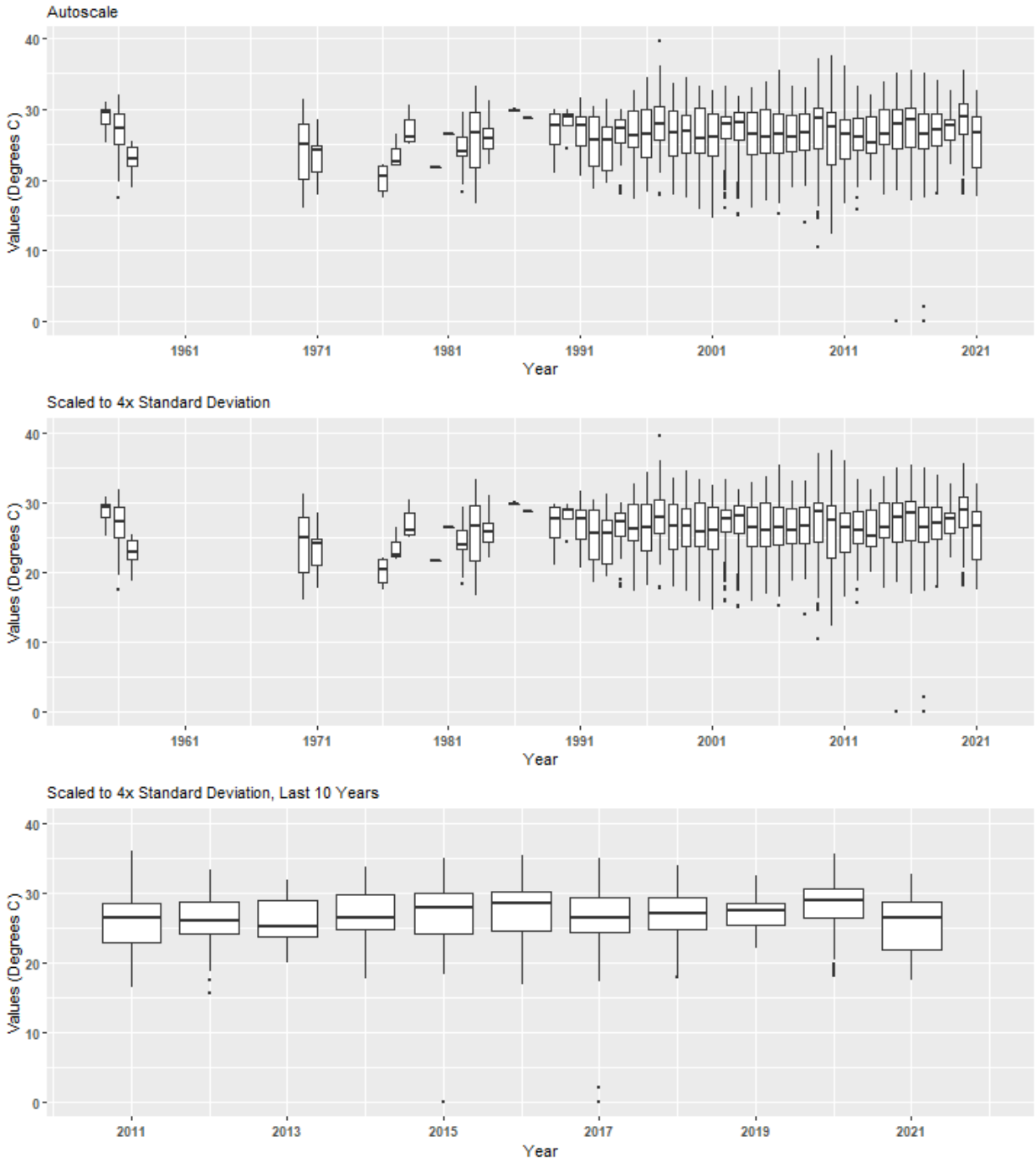
Summary Box Plots for Estero Bay Aquatic Preserve

By Month



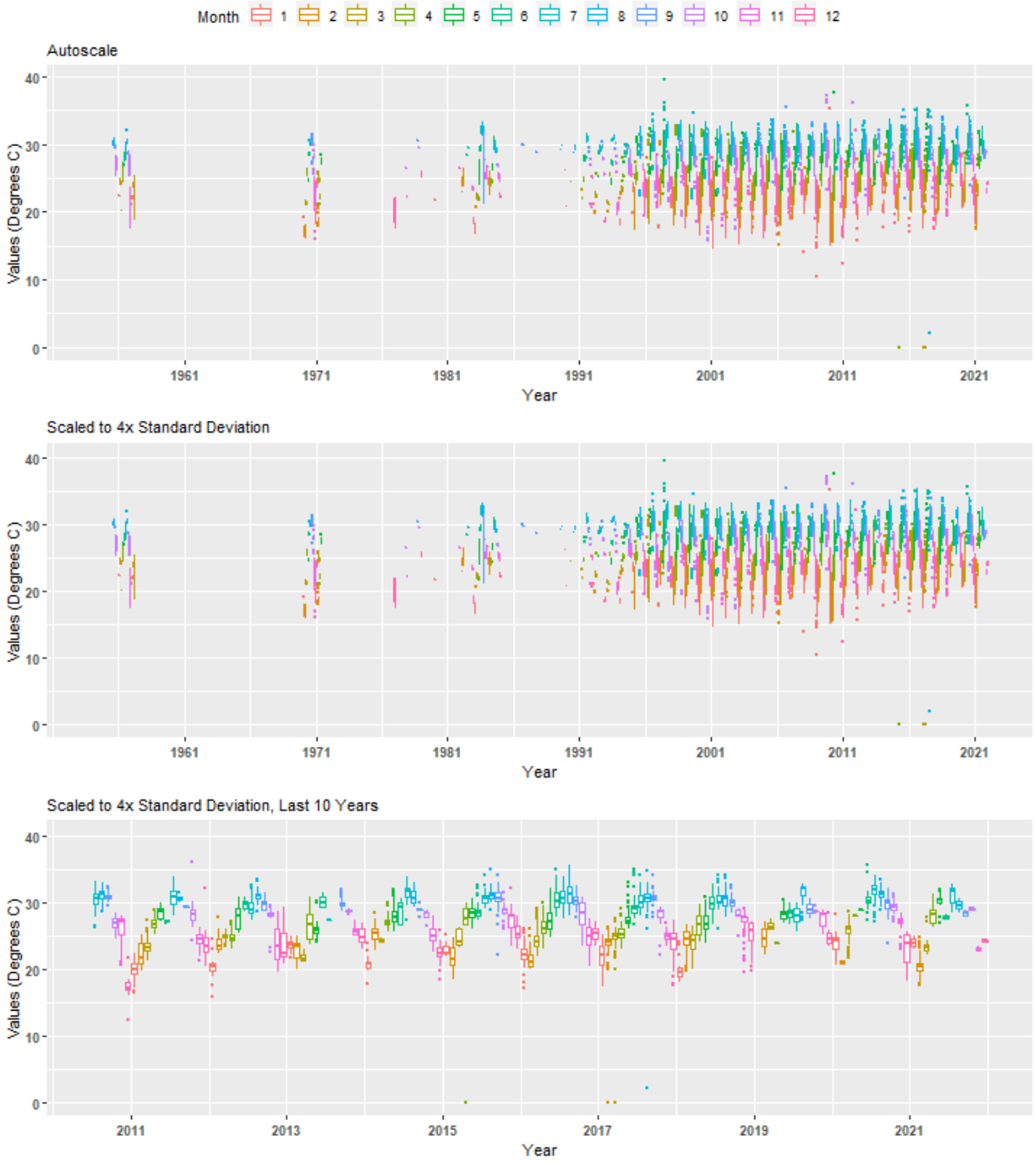
Summary Box Plots for Florida Keys National Marine Sanctuary

By Year



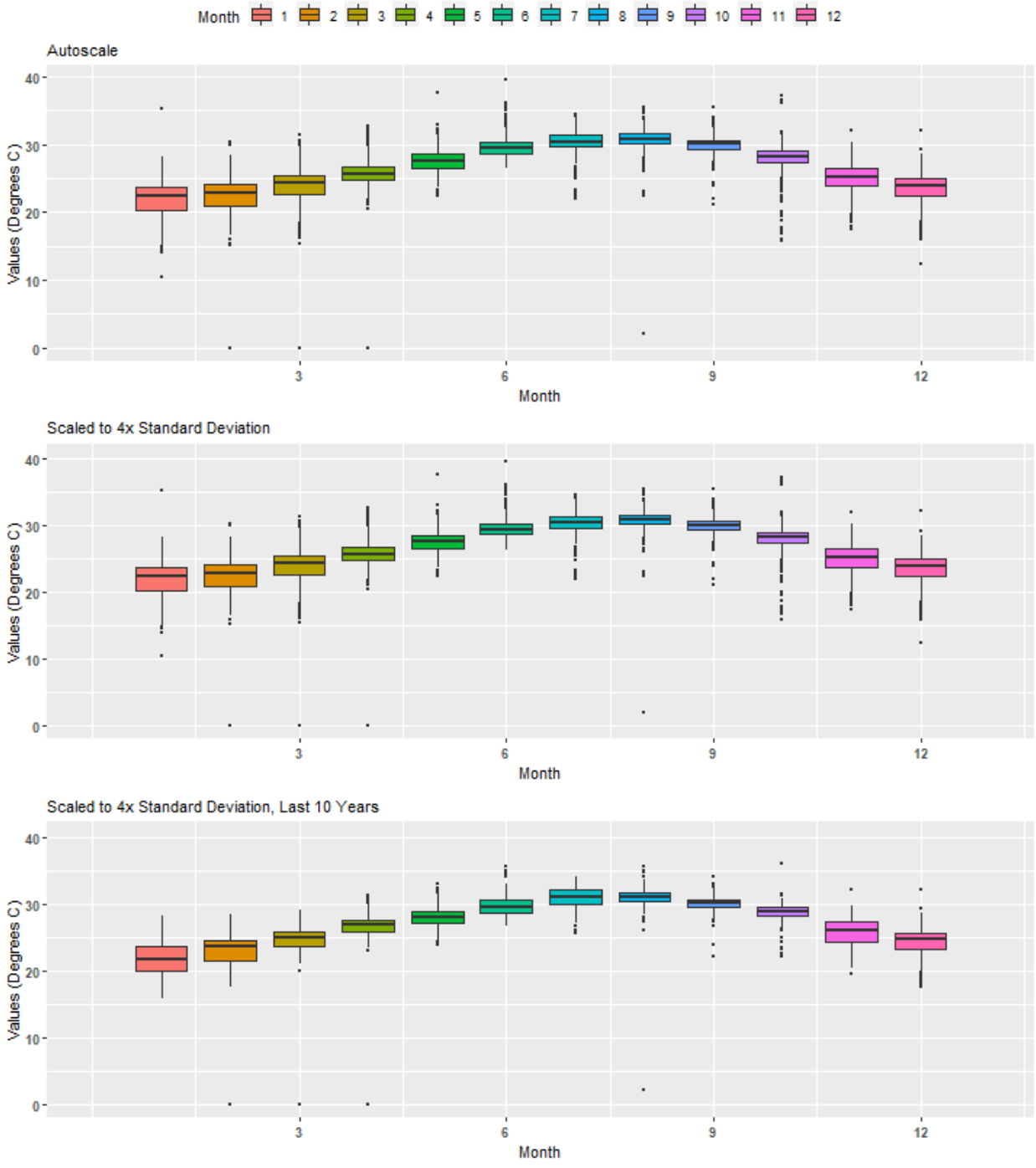
Summary Box Plots for Florida Keys National Marine Sanctuary

By Year & Month



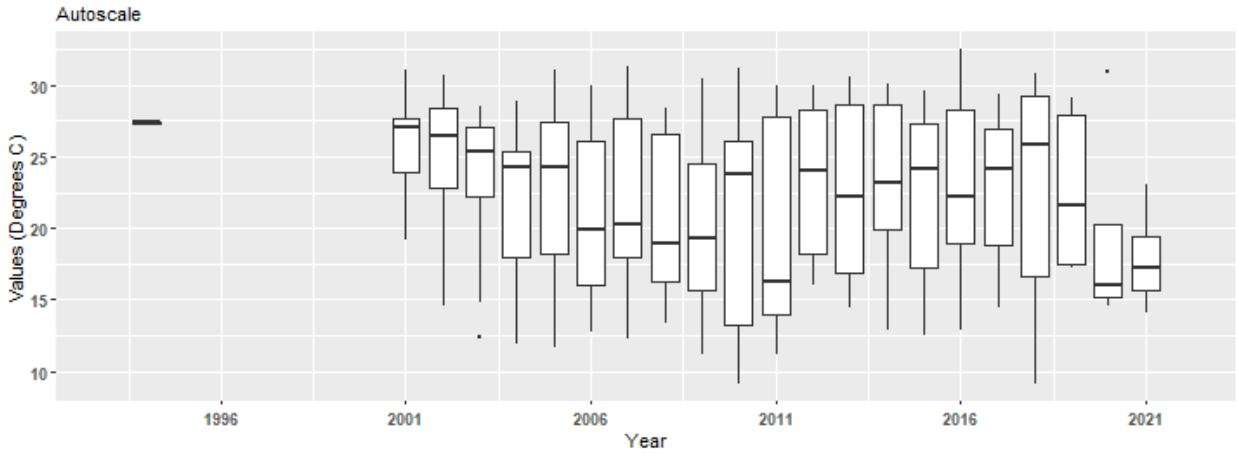
Summary Box Plots for Florida Keys National Marine Sanctuary

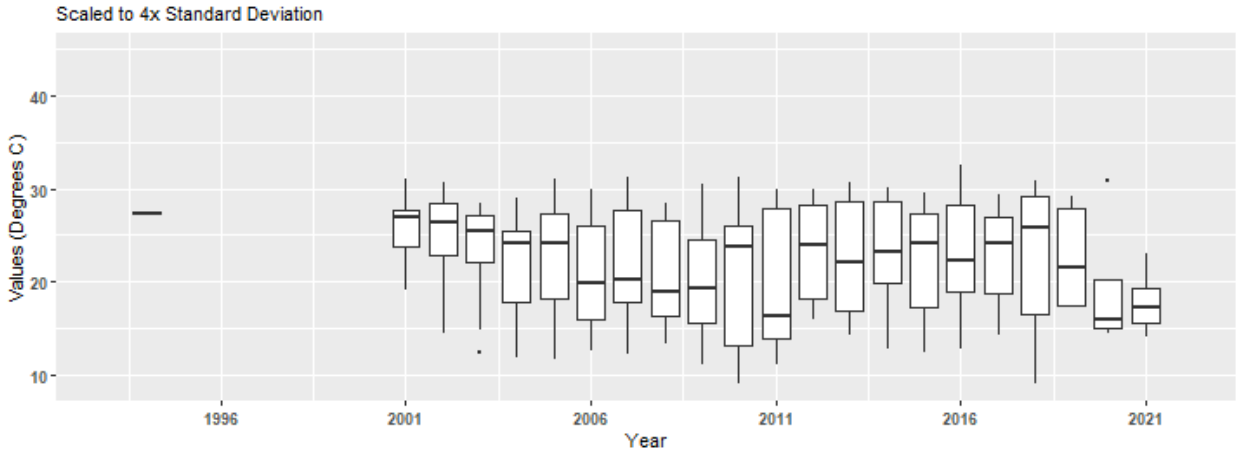
By Month

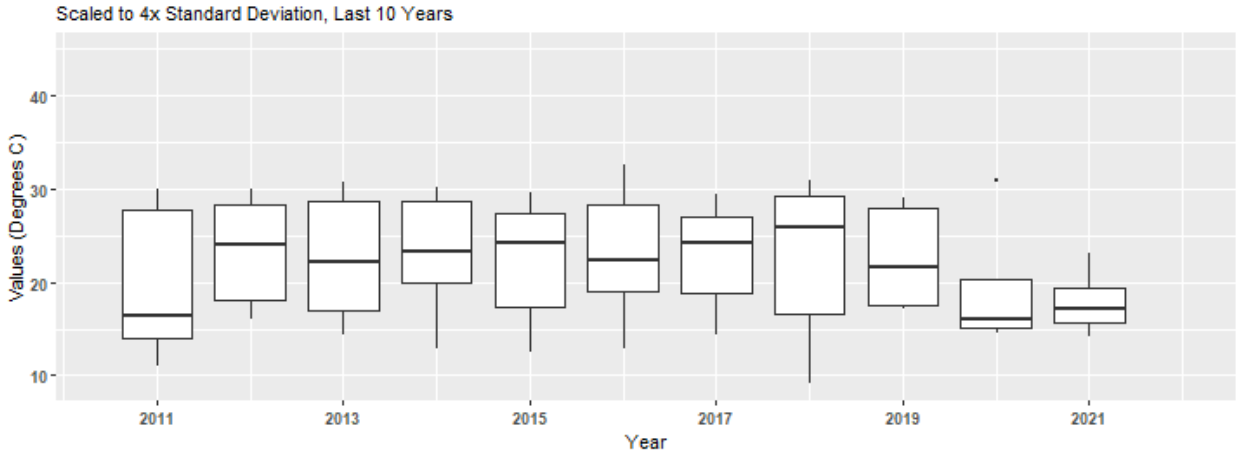


Summary Box Plots for Fort Clinch State Park Aquatic Preserve

By Year

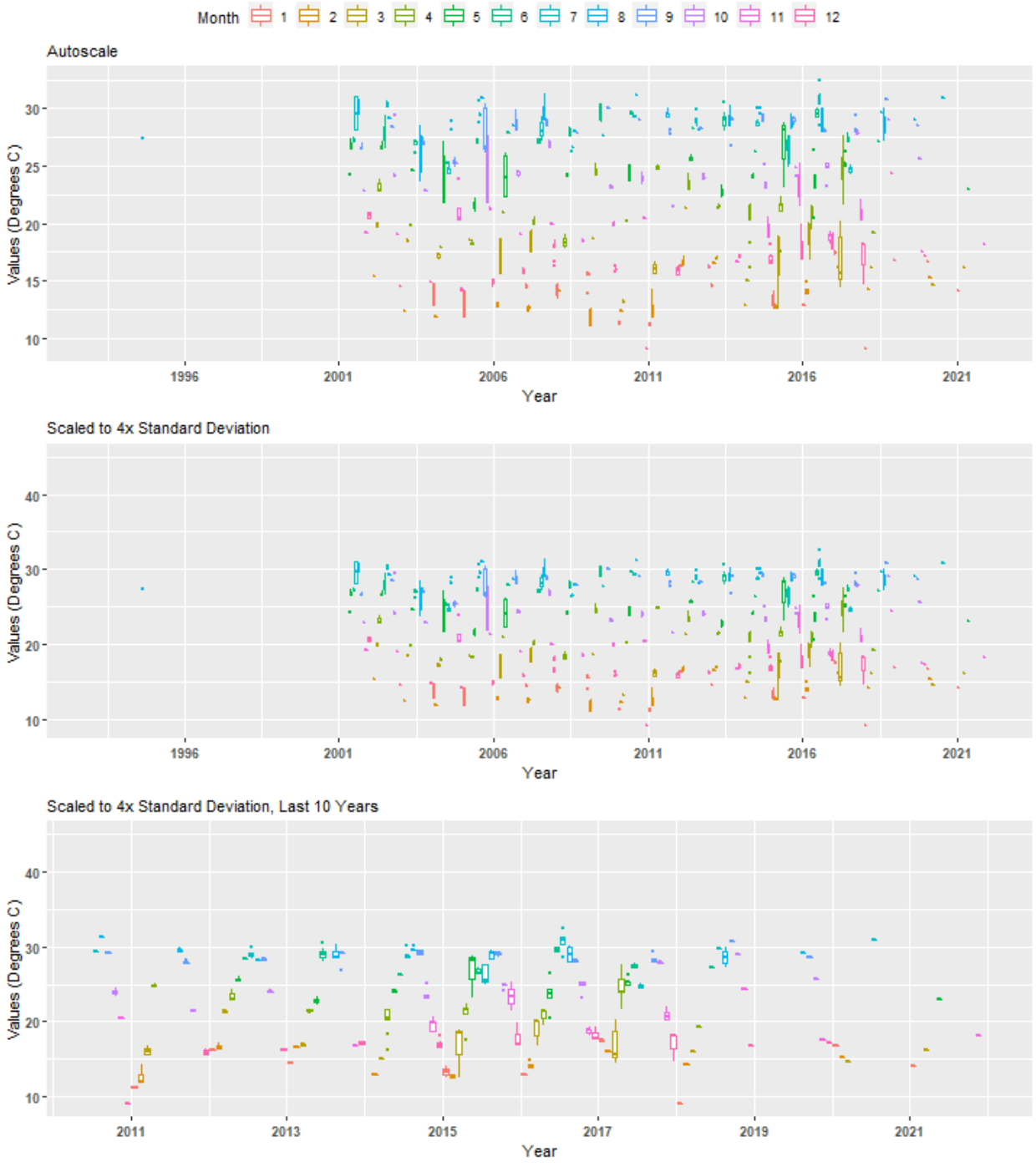






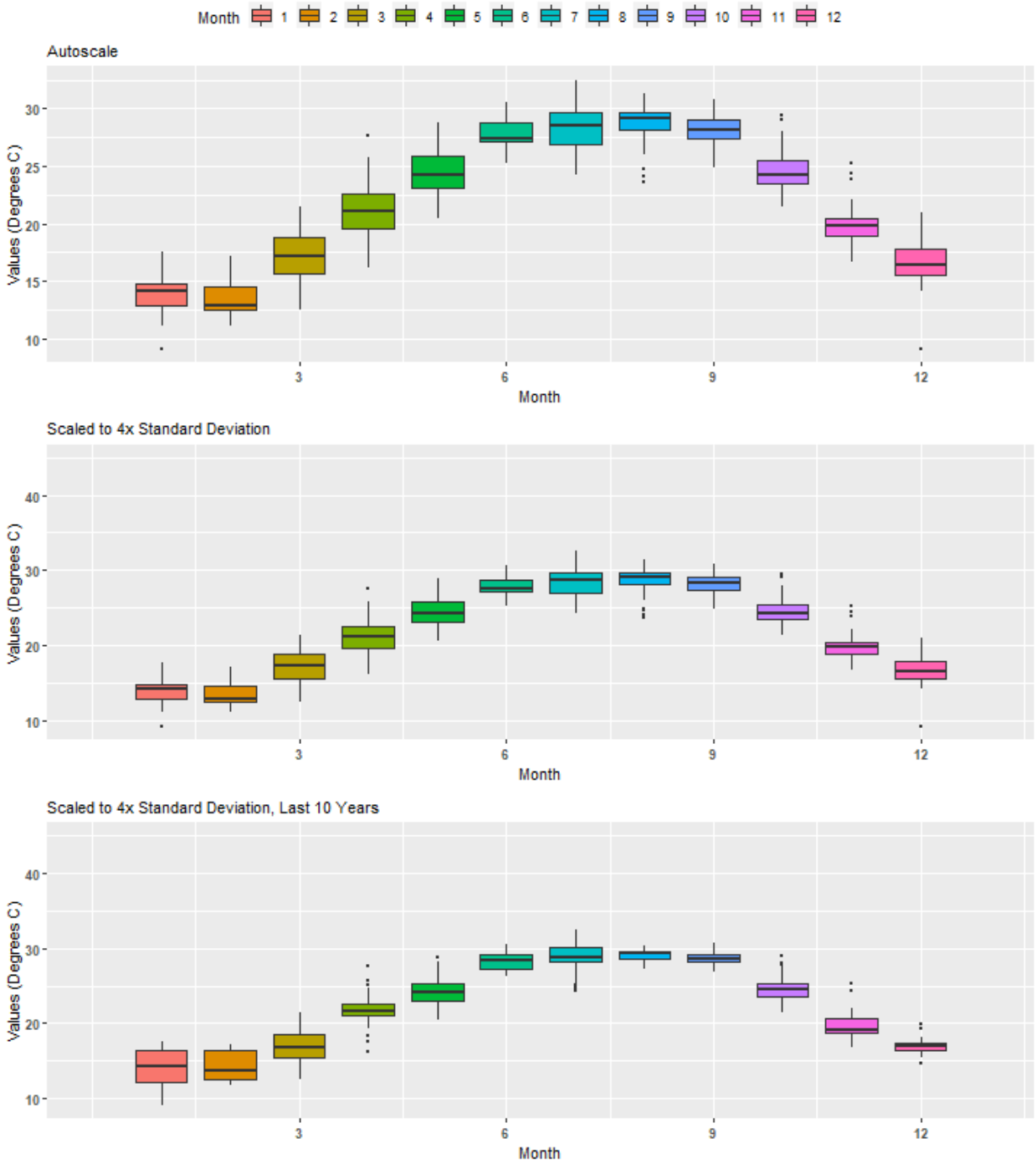
Summary Box Plots for Fort Clinch State Park Aquatic Preserve

By Year & Month



Summary Box Plots for Fort Clinch State Park Aquatic Preserve

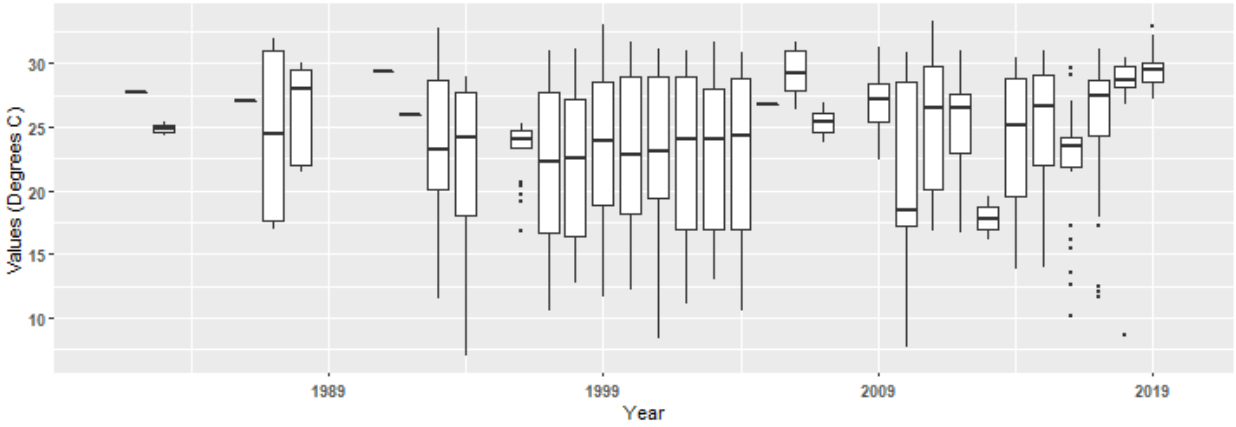
By Month



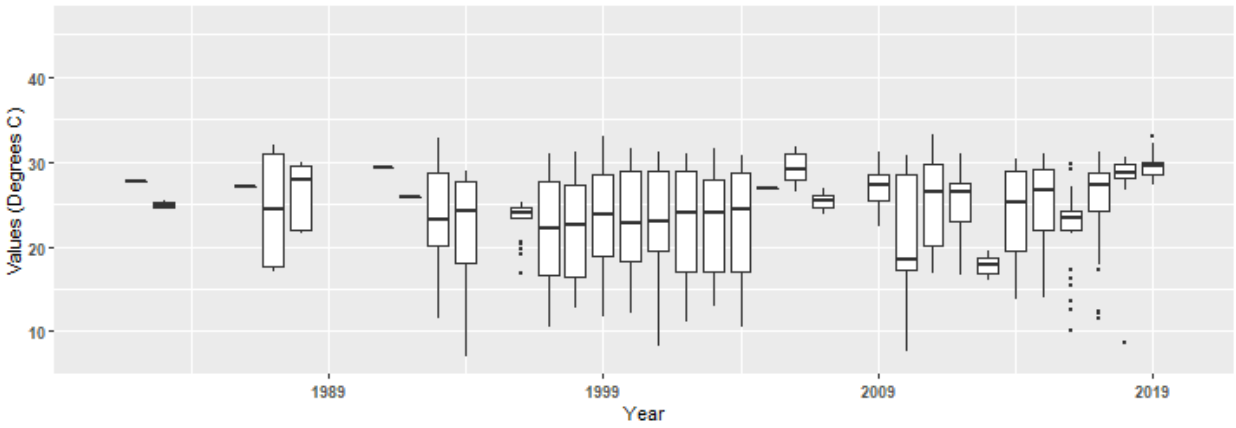
Summary Box Plots for Fort Pickens State Park Aquatic Preserve

By Year

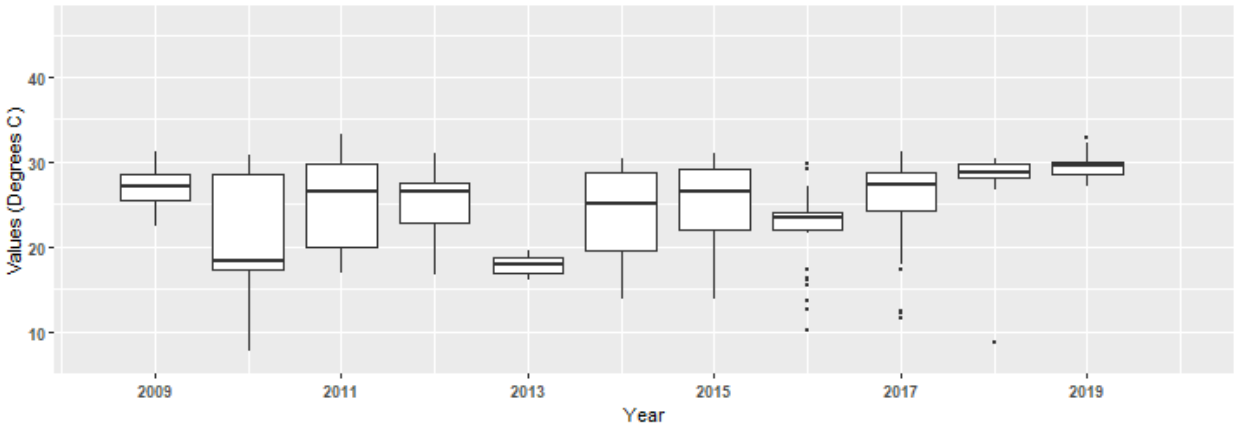
Autoscale



Scaled to 4x Standard Deviation



Scaled to 4x Standard Deviation, Last 10 Years

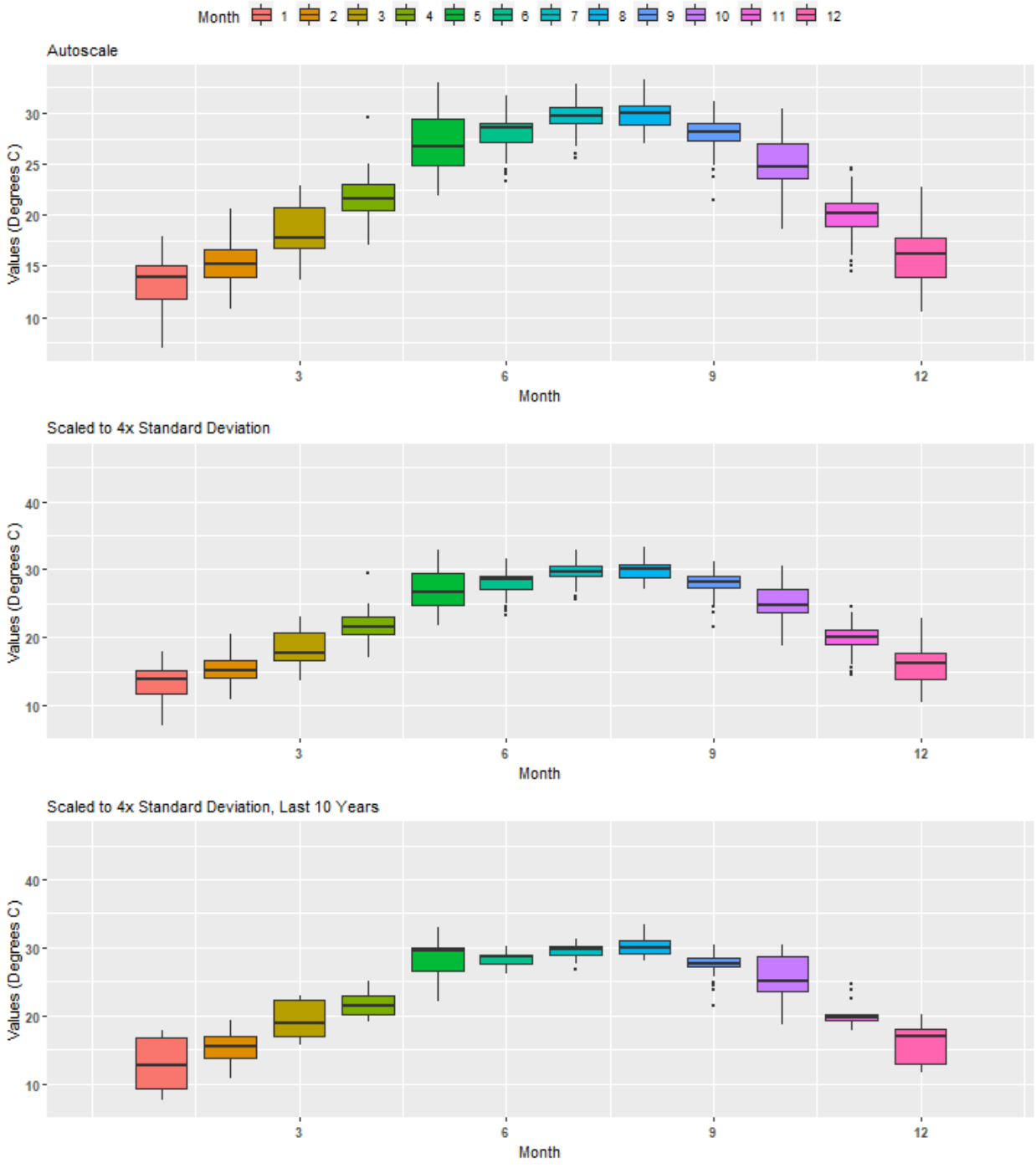


Summary Box Plots for Fort Pickens State Park Aquatic Preserve

By Year & Month

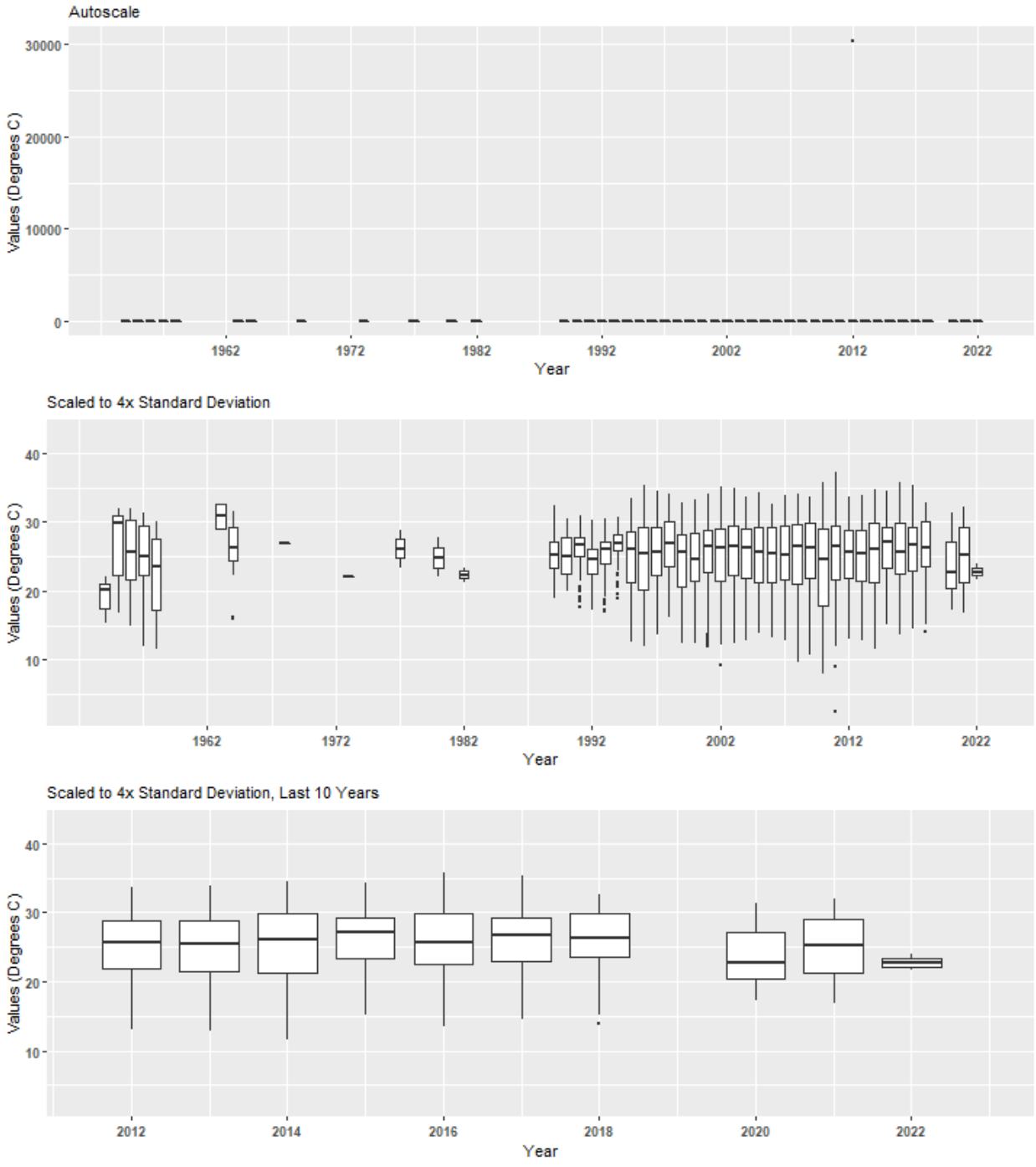
Summary Box Plots for Fort Pickens State Park Aquatic Preserve

By Month



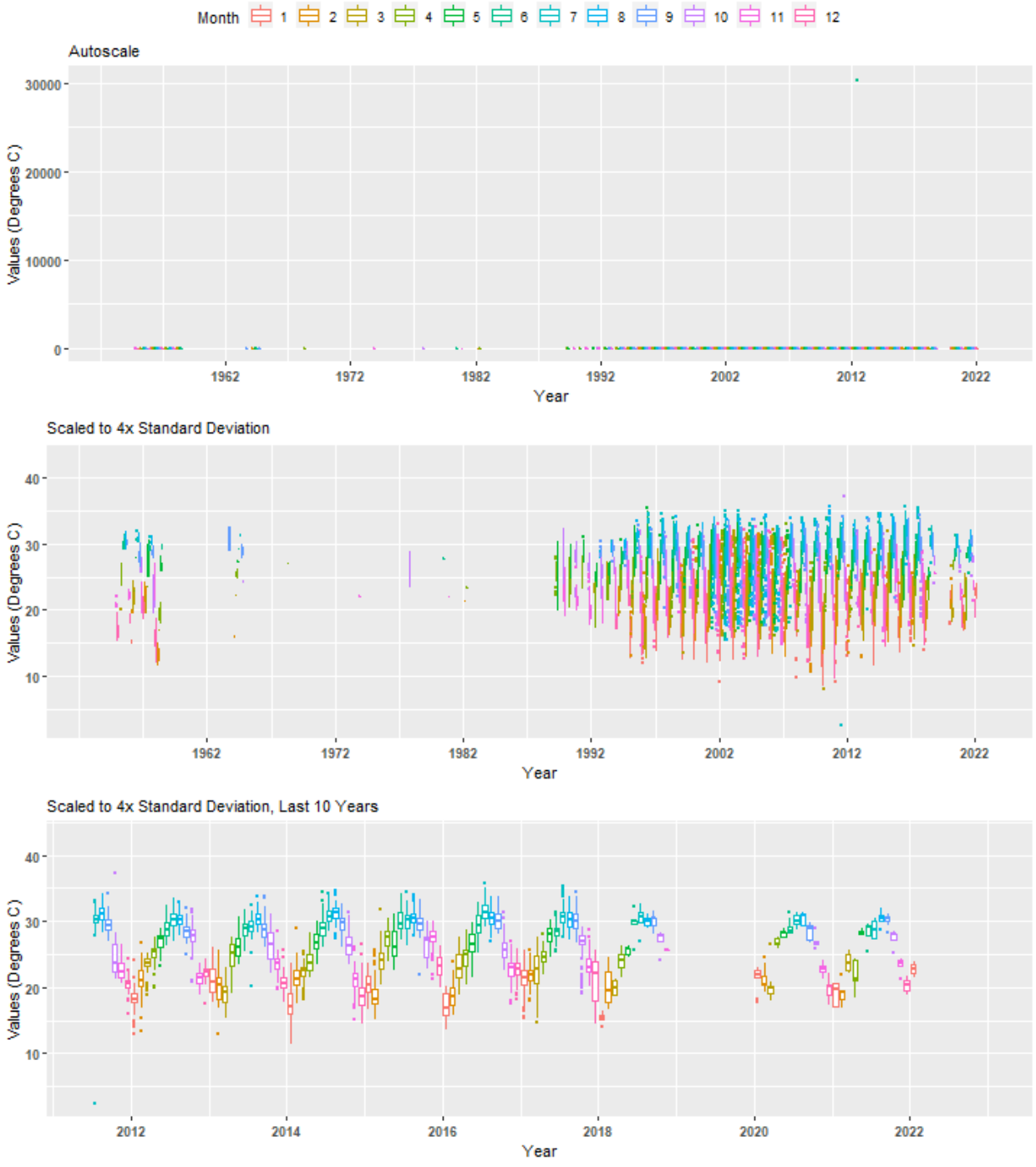
Summary Box Plots for Gasparilla Sound-Charlotte Harbor Aquatic Preserve

By Year



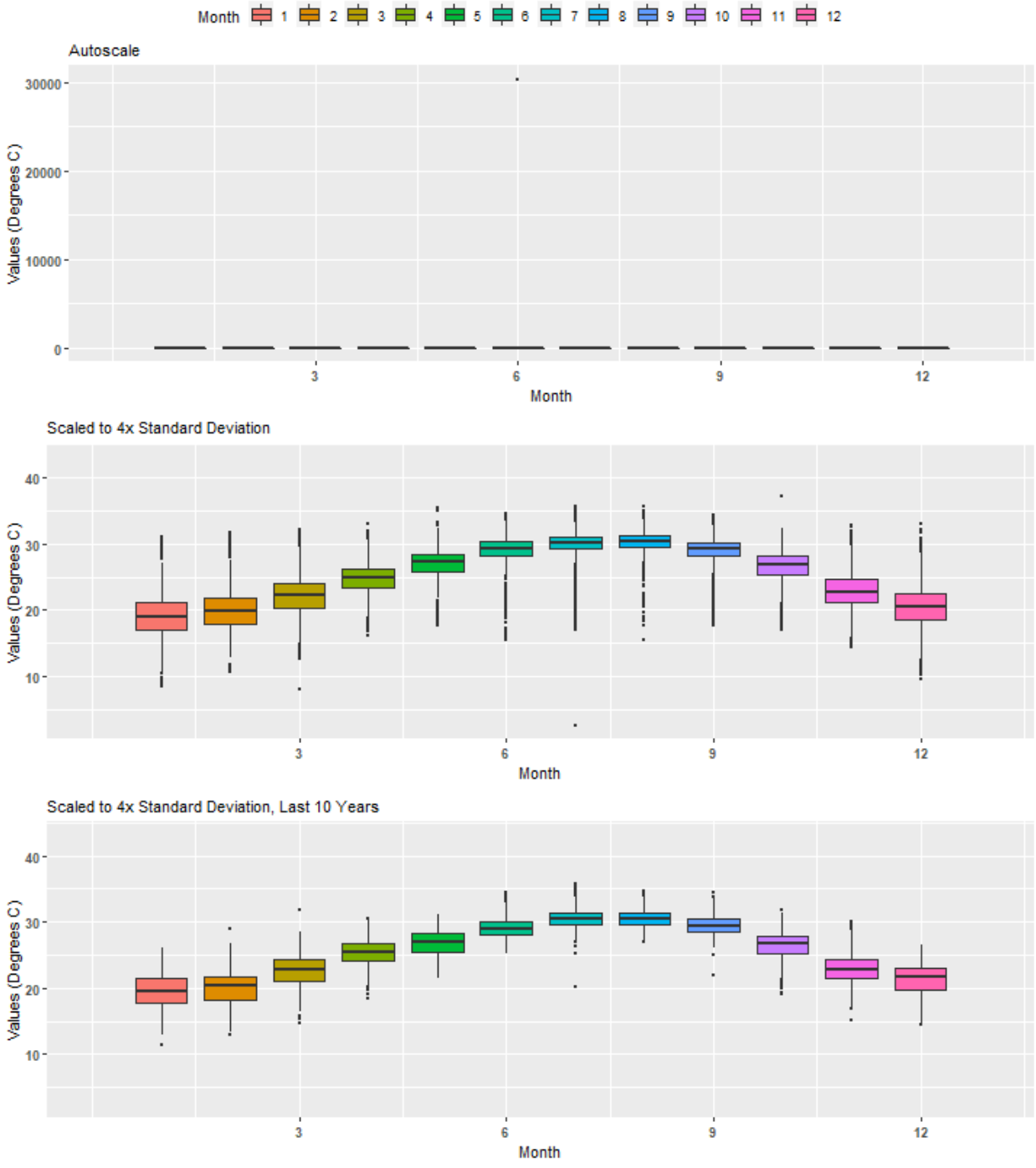
Summary Box Plots for Gasparilla Sound-Charlotte Harbor Aquatic Preserve

By Year & Month



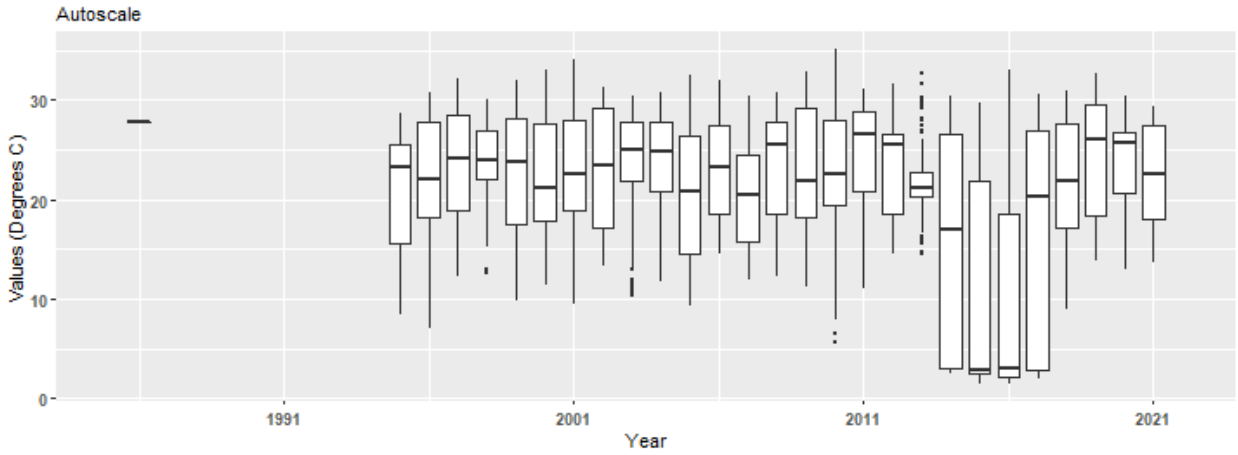
Summary Box Plots for Gasparilla Sound-Charlotte Harbor Aquatic Preserve

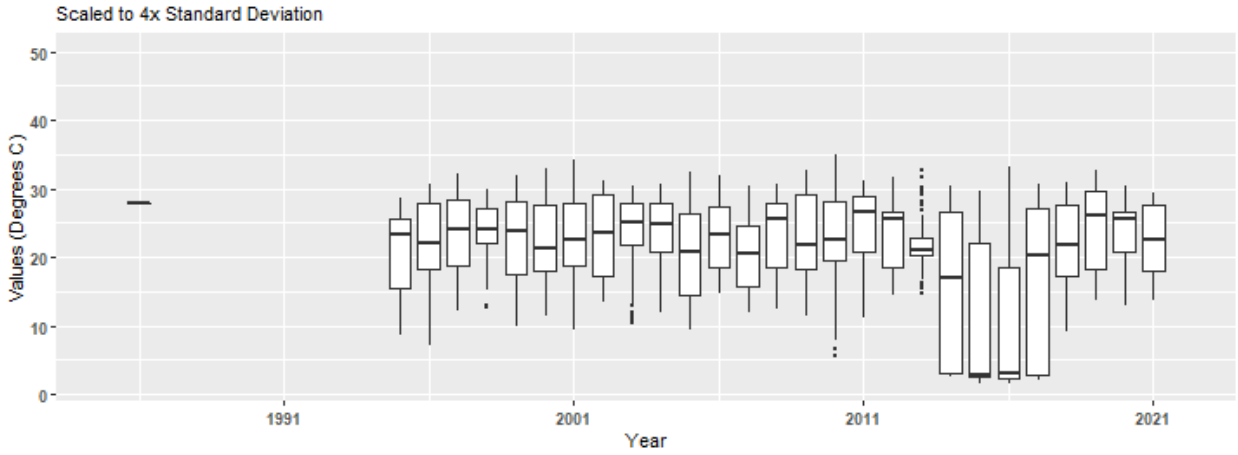
By Month

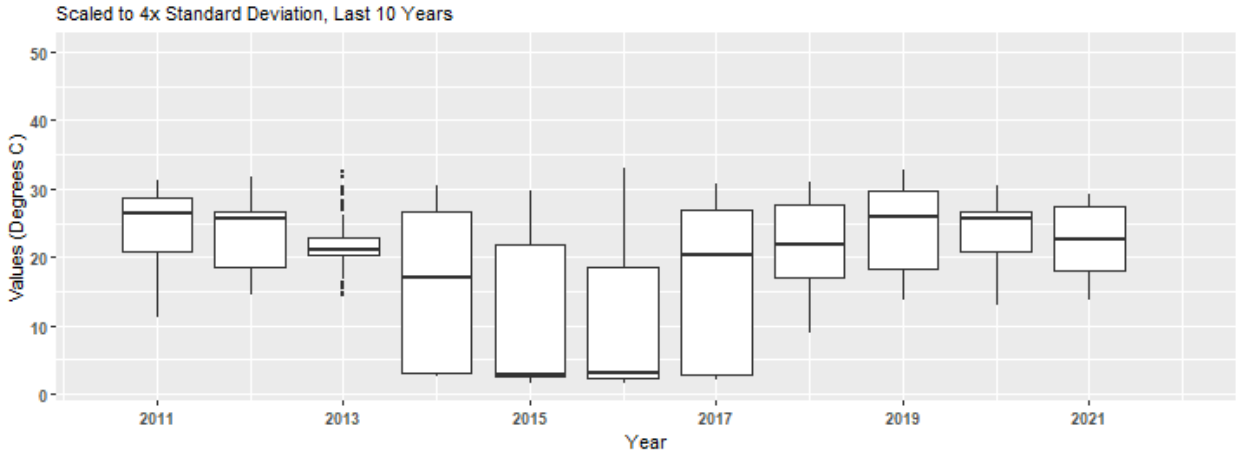


Summary Box Plots for Guana River Marsh Aquatic Preserve

By Year

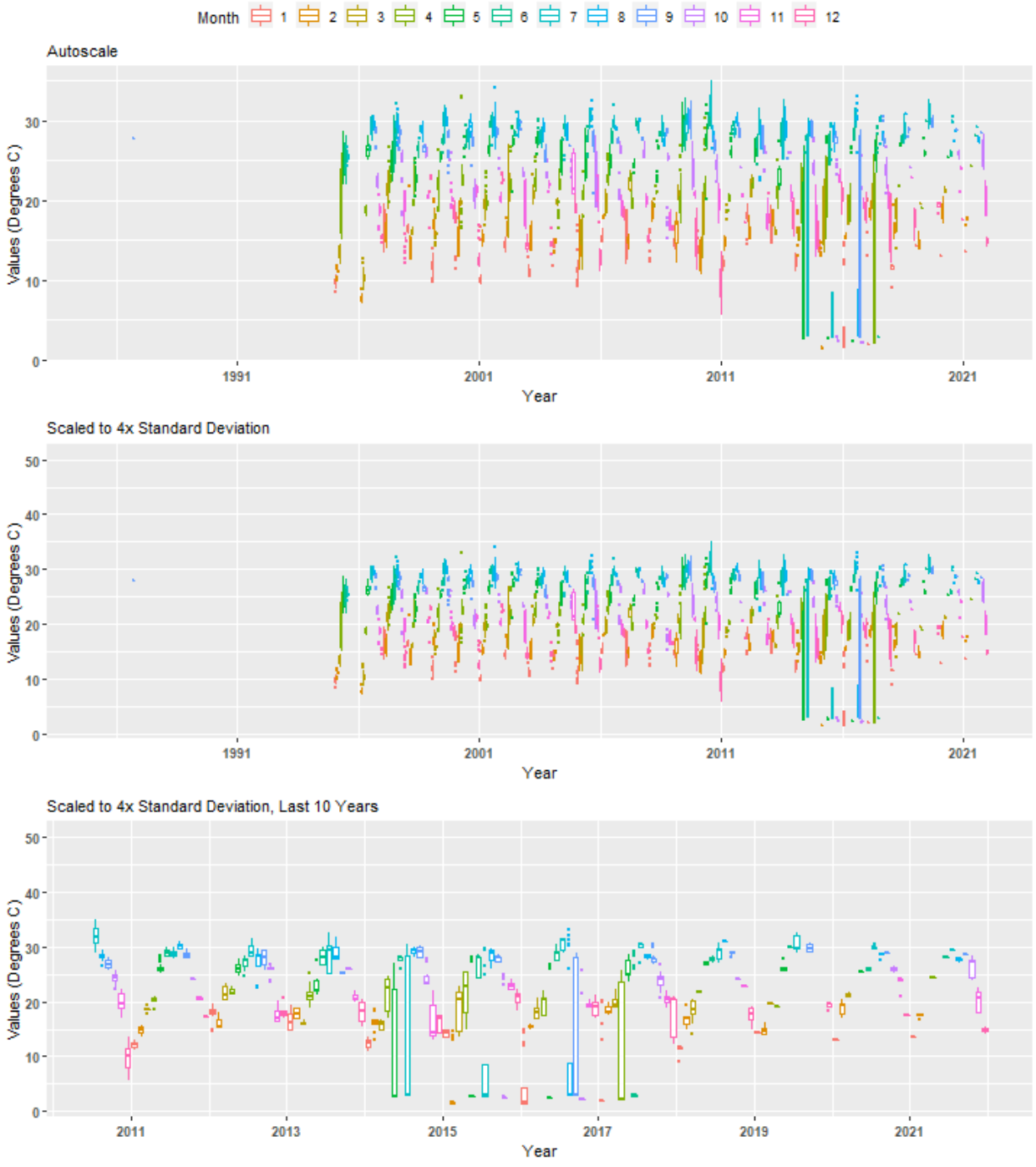






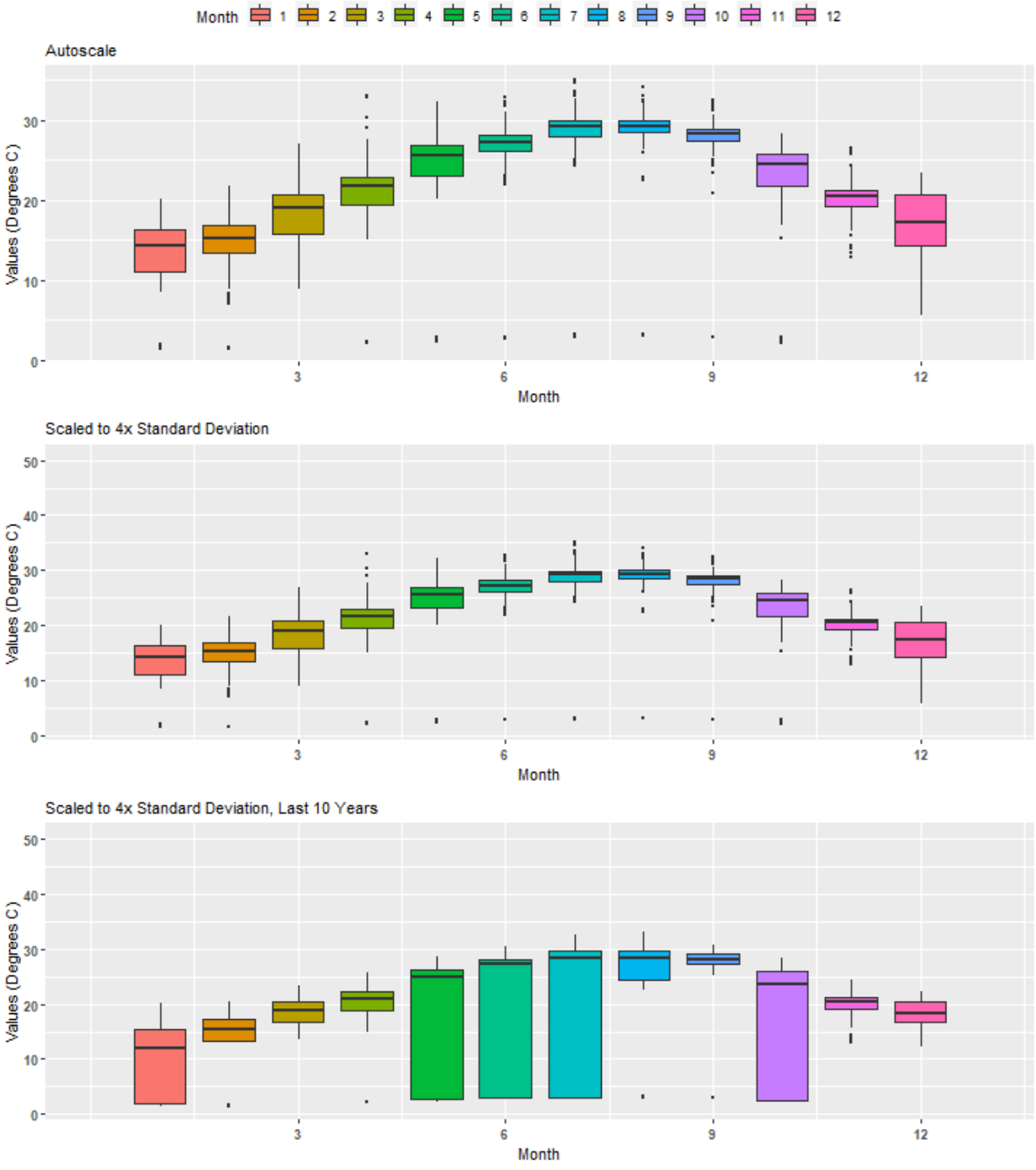
Summary Box Plots for Guana River Marsh Aquatic Preserve

By Year & Month



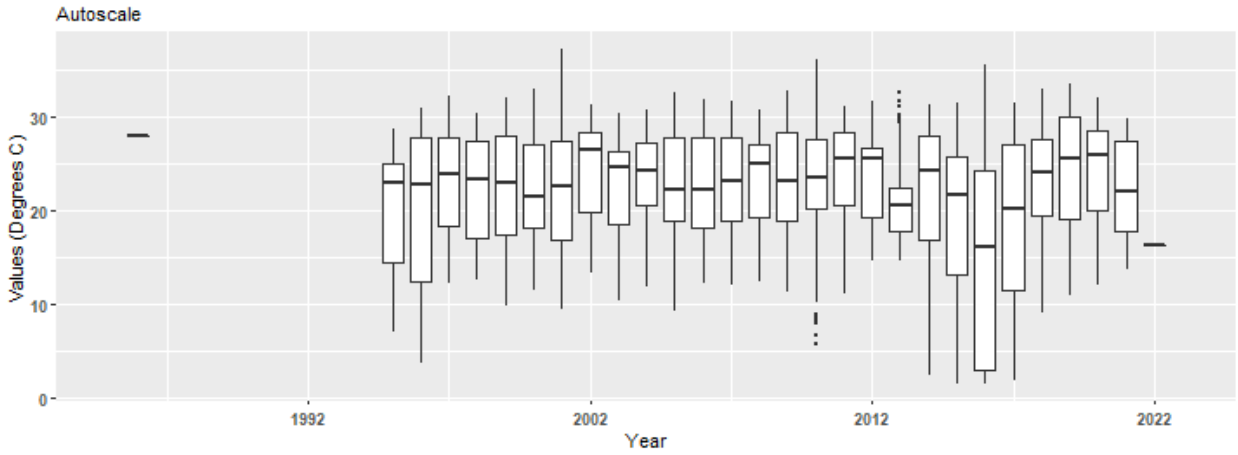
Summary Box Plots for Guana River Marsh Aquatic Preserve

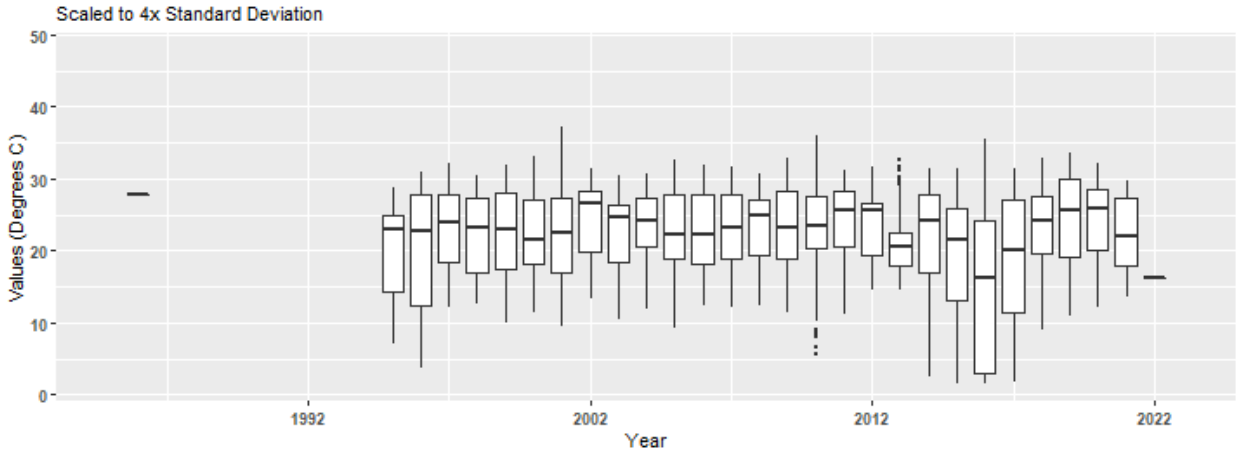
By Month

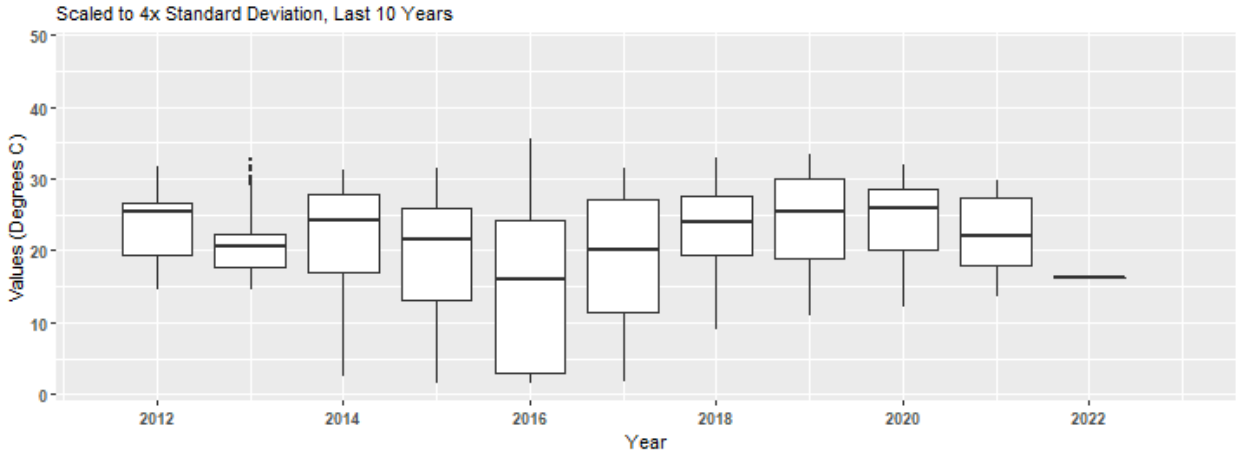


Summary Box Plots for Guana Tolomato Matanzas National Estuarine Research Reserve

By Year

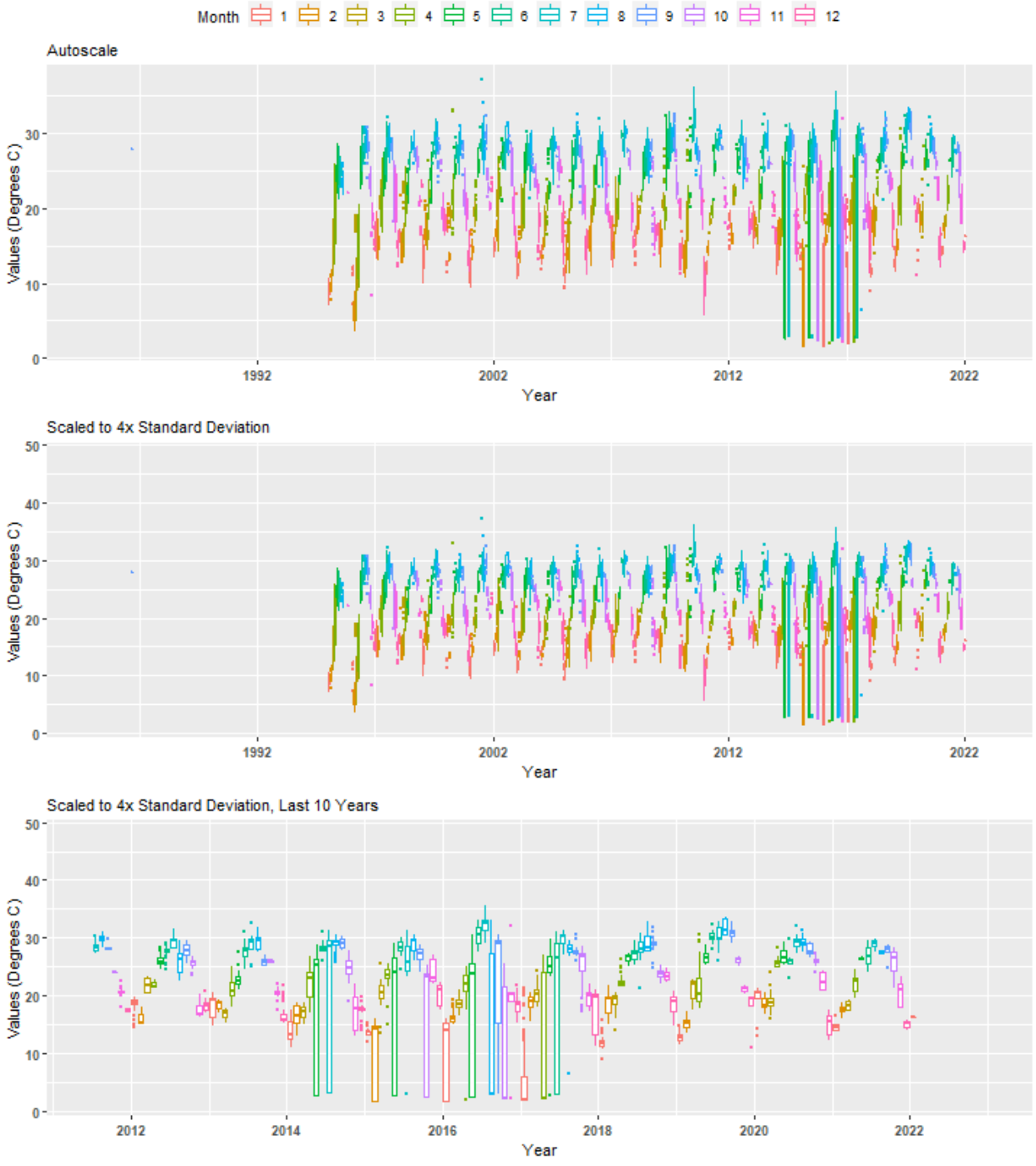






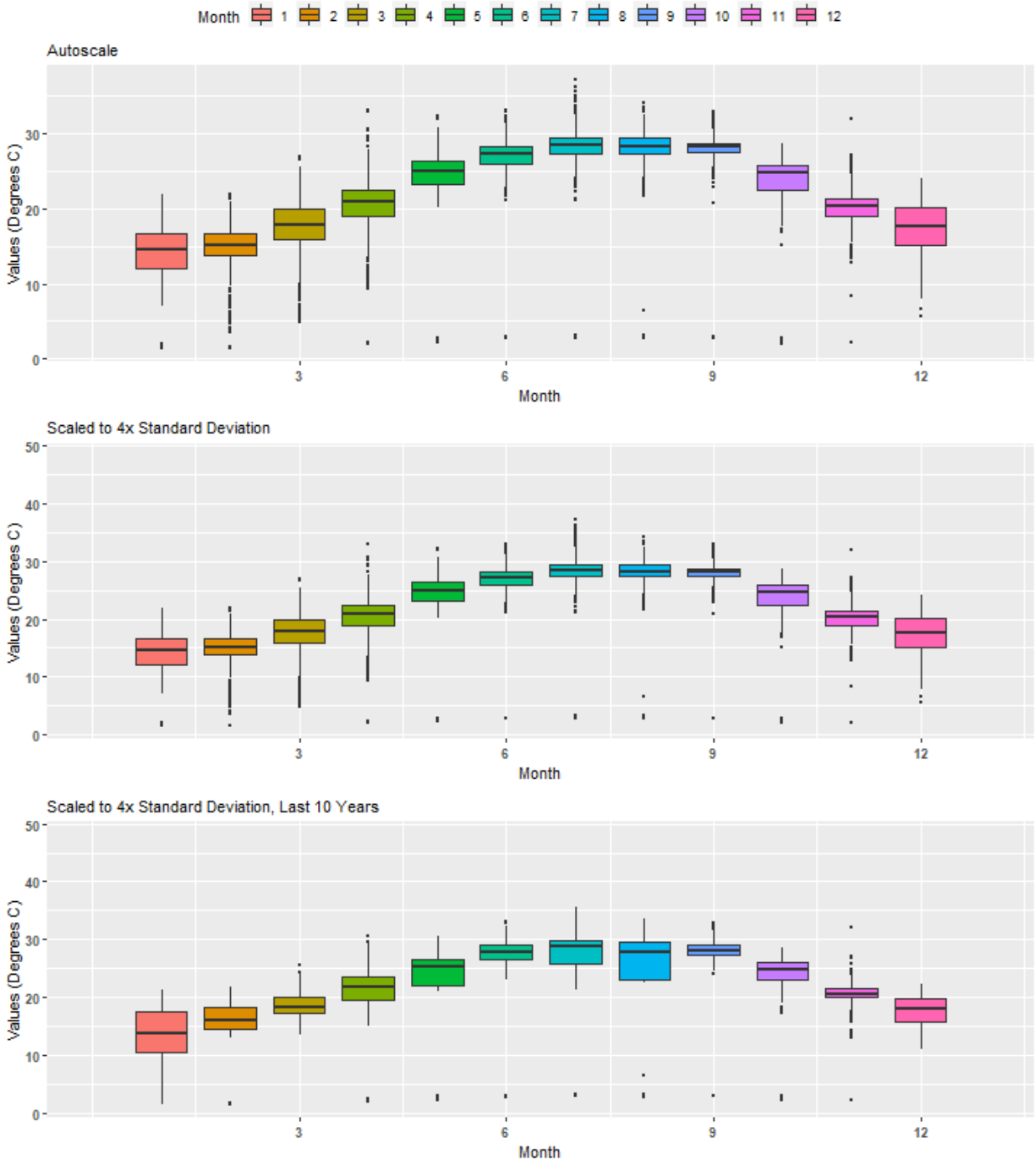
Summary Box Plots for Guana Tolomato Matanzas National Estuarine Research Reserve

By Year & Month



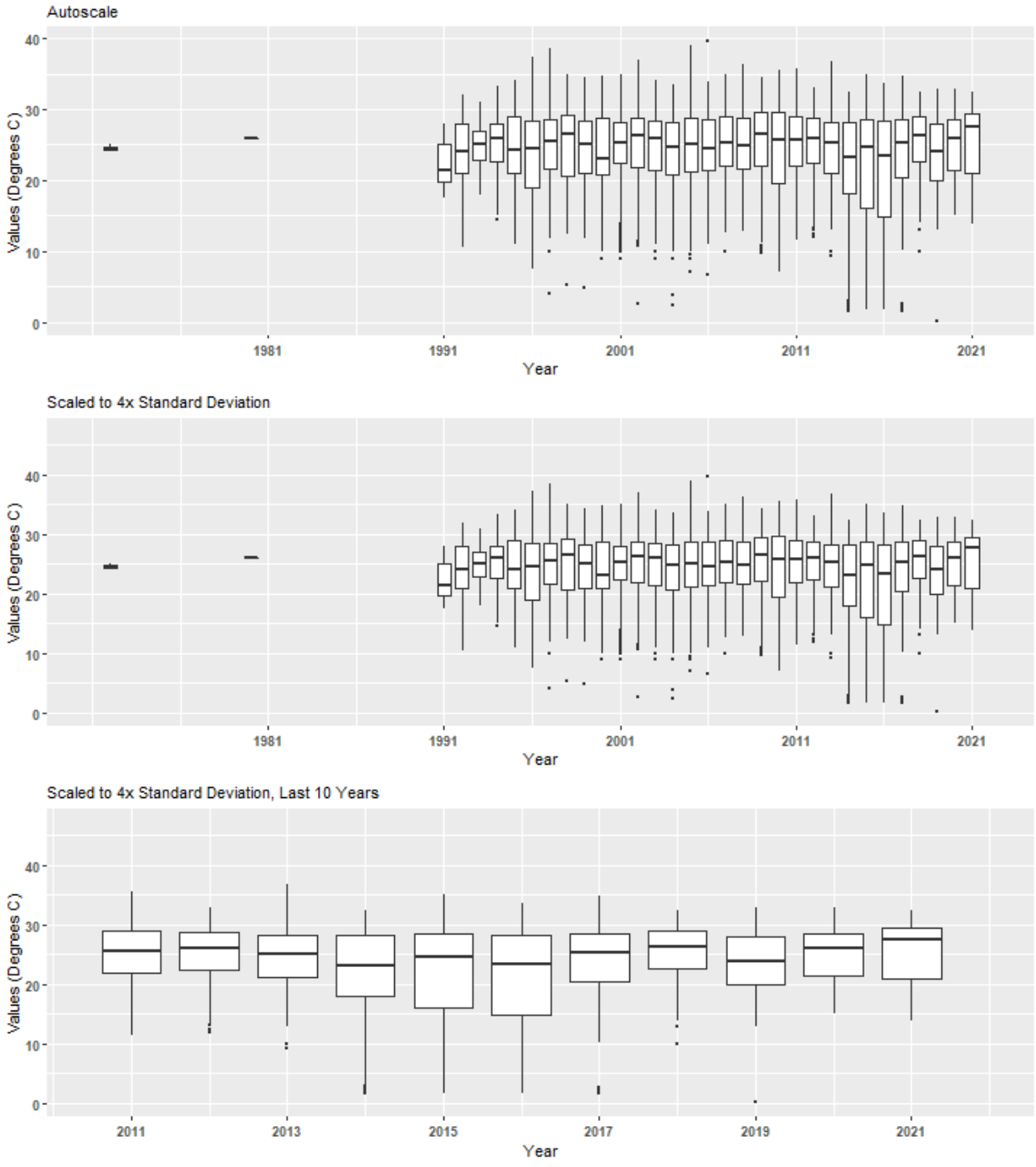
Summary Box Plots for Guana Tolomato Matanzas National Estuarine Research Reserve

By Month



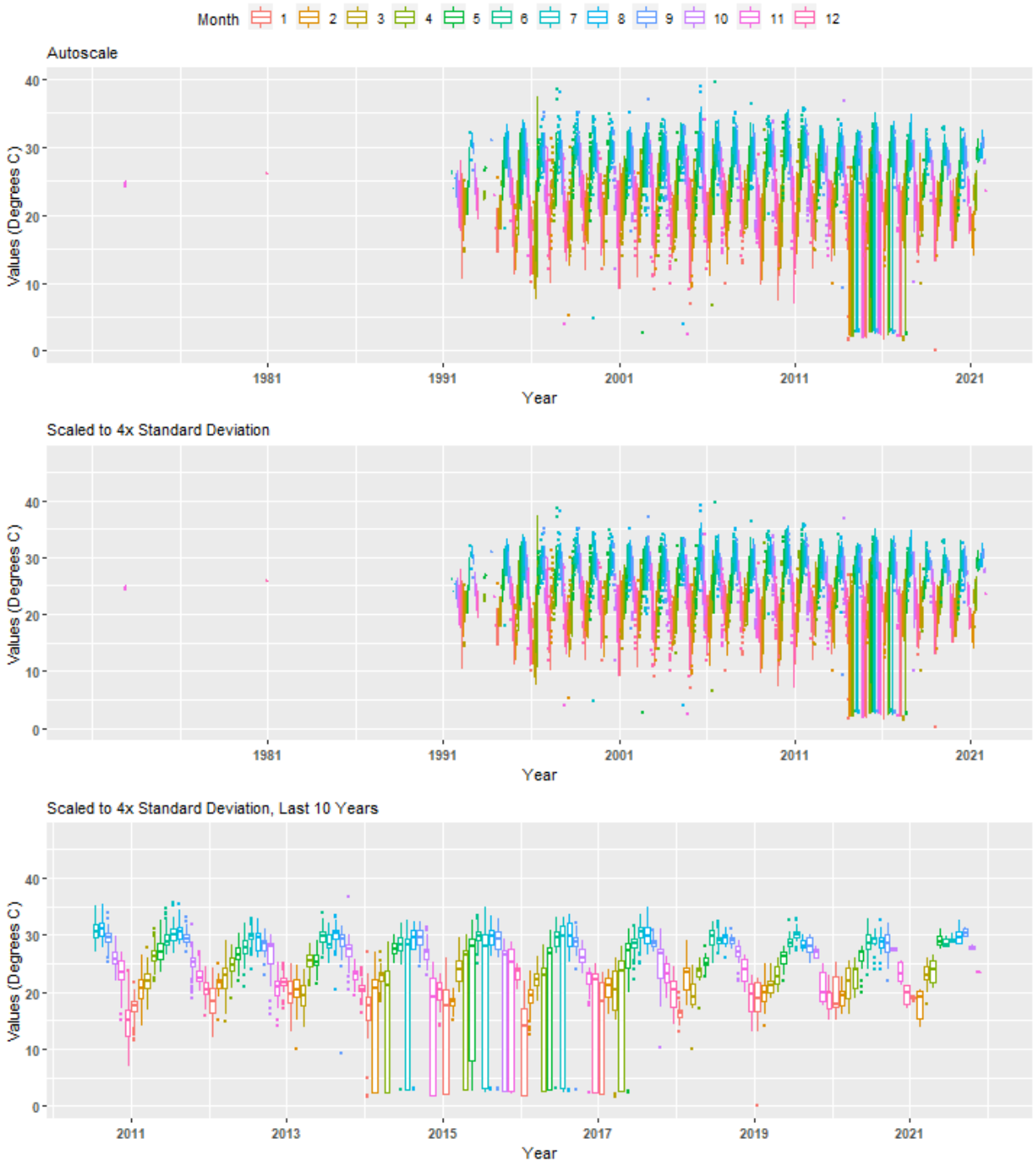
Summary Box Plots for Indian River-Malabar to Vero Beach Aquatic Preserve

By Year



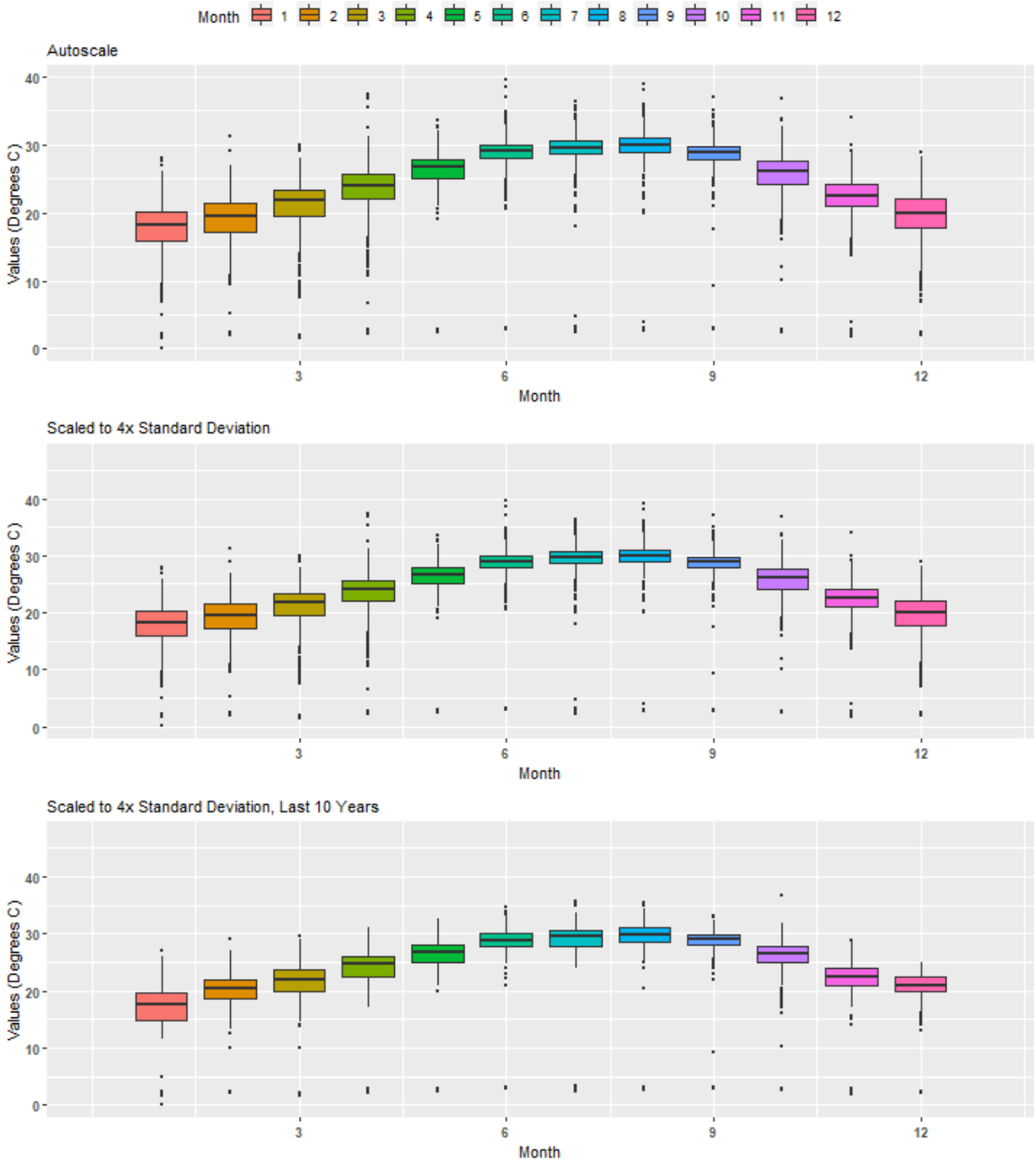
Summary Box Plots for Indian River-Malabar to Vero Beach Aquatic Preserve

By Year & Month



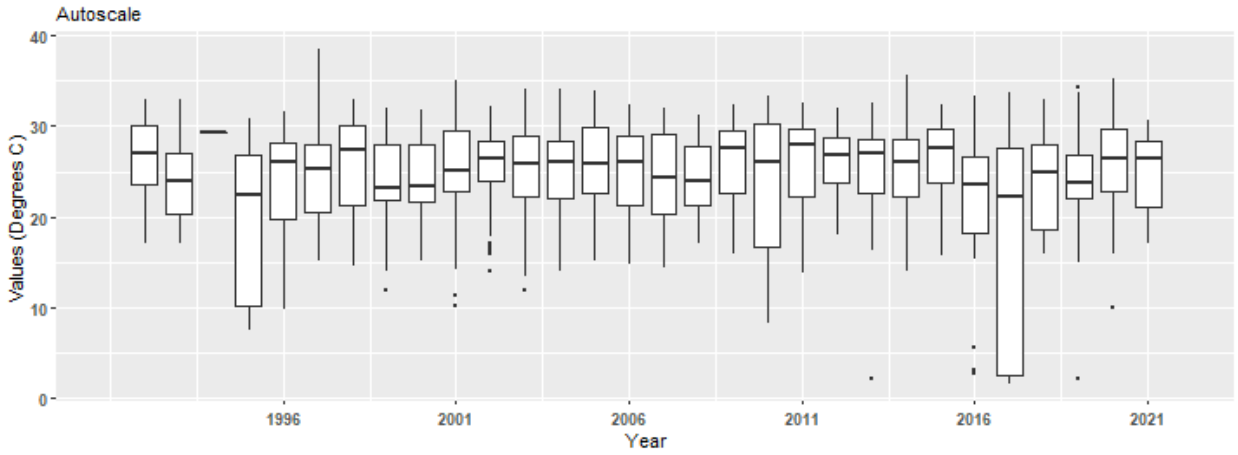
Summary Box Plots for Indian River-Malabar to Vero Beach Aquatic Preserve

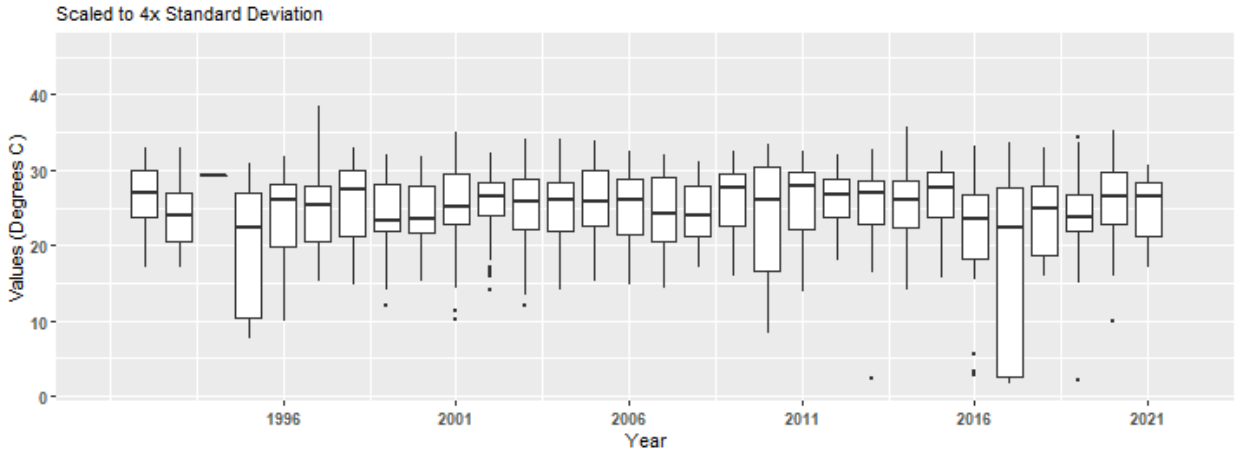
By Month

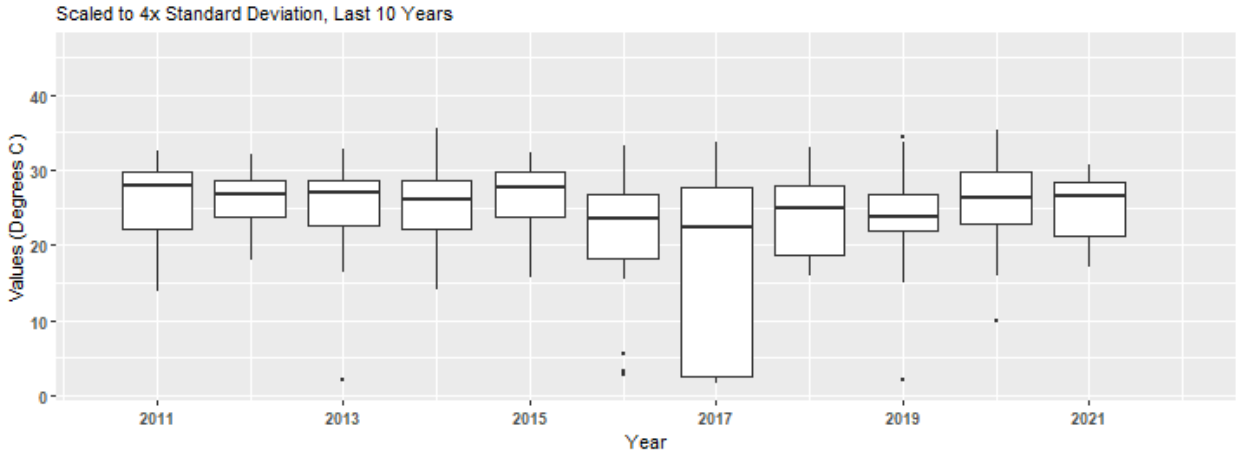


Summary Box Plots for Indian River-Vero Beach to Ft. Pierce Aquatic Preserve

By Year





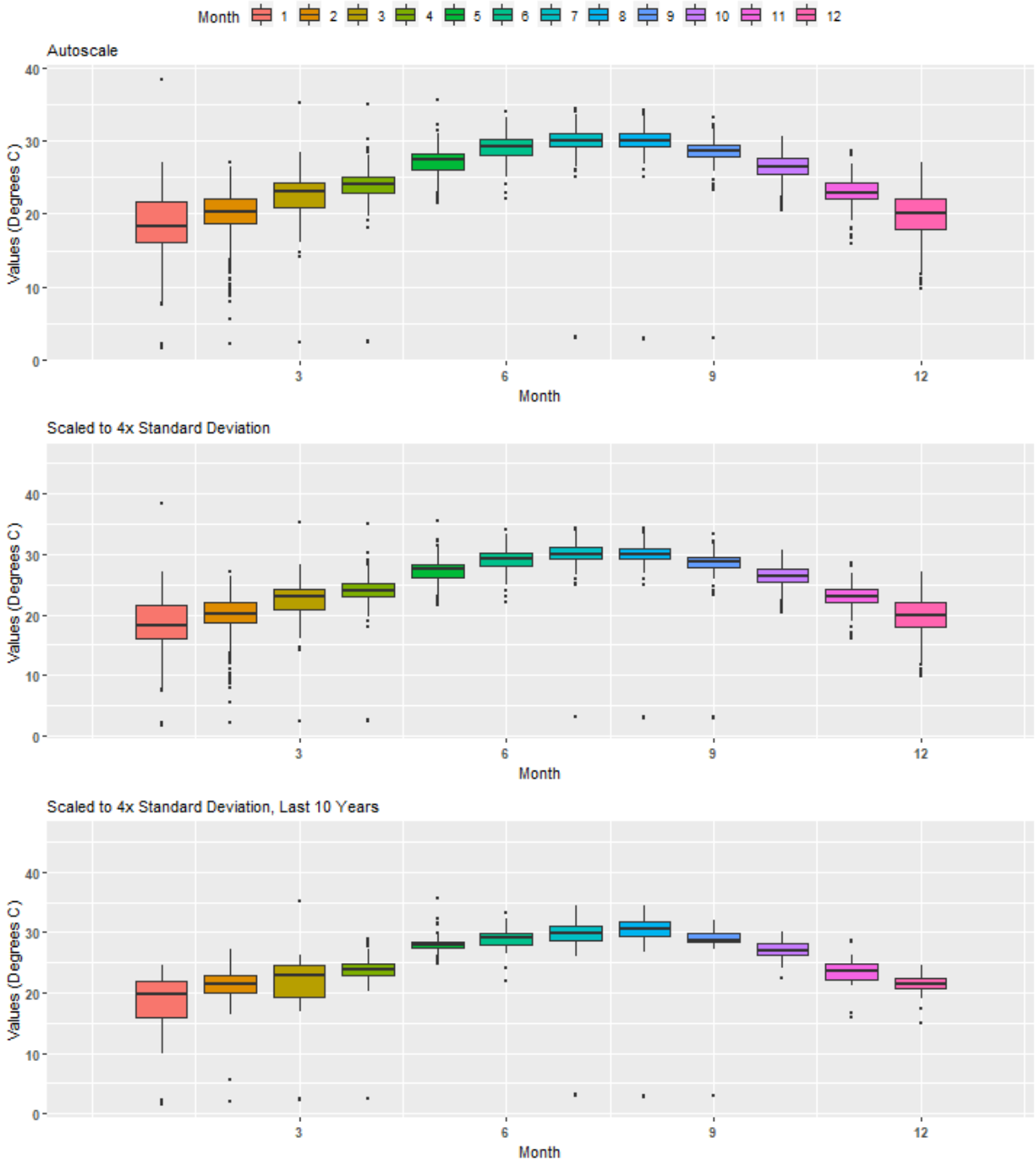


Summary Box Plots for Indian River-Vero Beach to Ft. Pierce Aquatic Preserve

By Year & Month

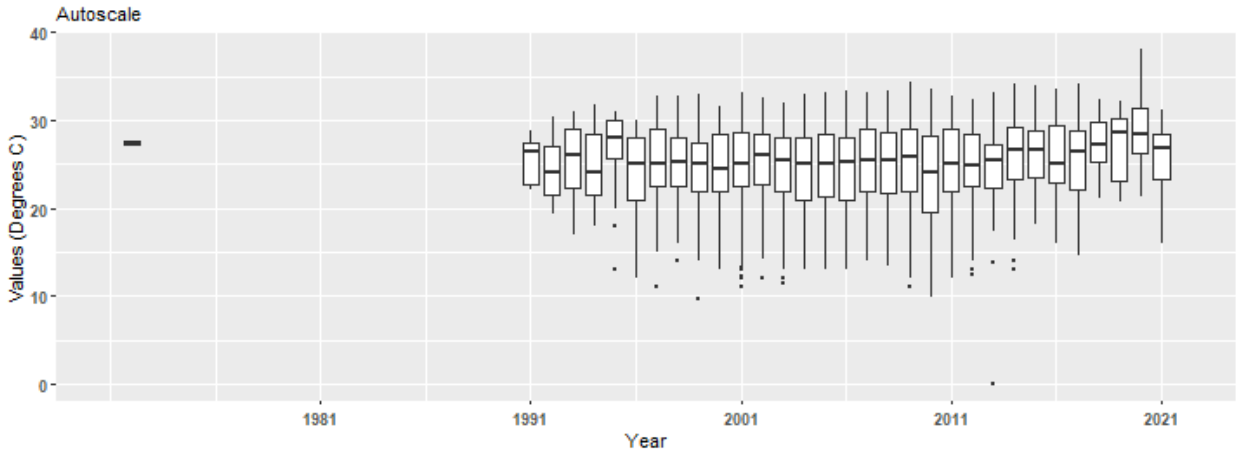
Summary Box Plots for Indian River-Vero Beach to Ft. Pierce Aquatic Preserve

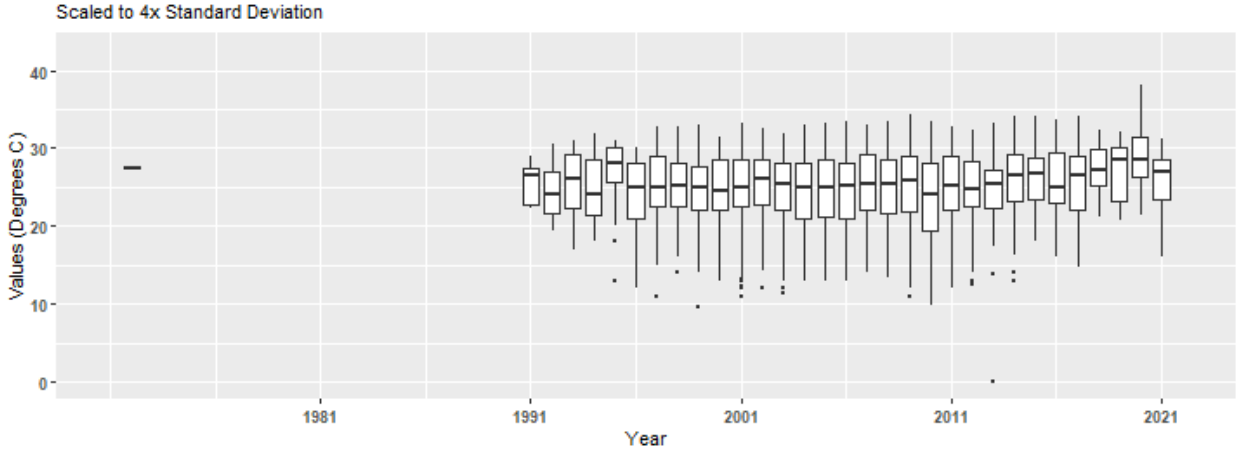
By Month

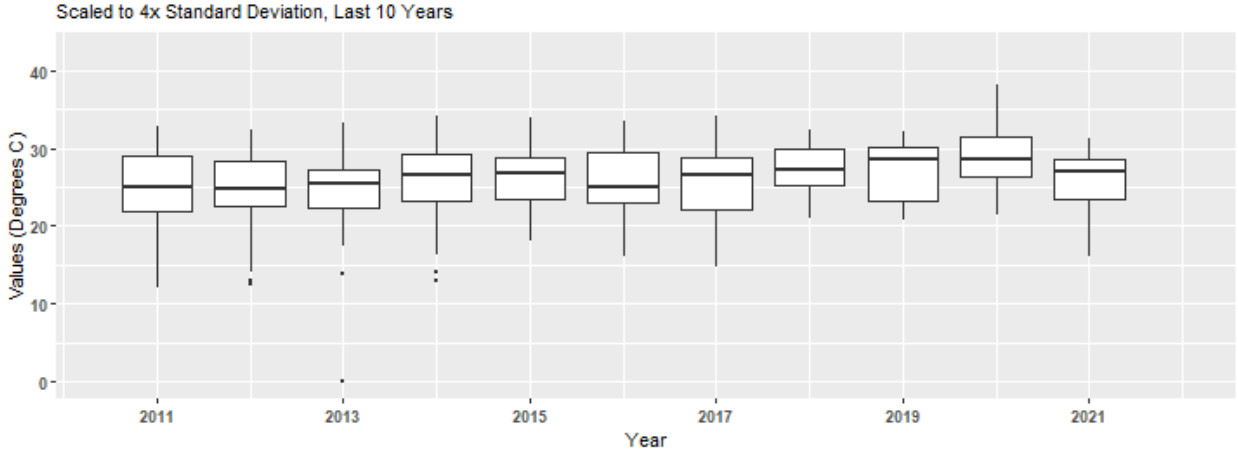


Summary Box Plots for Jensen Beach to Jupiter Inlet Aquatic Preserve

By Year

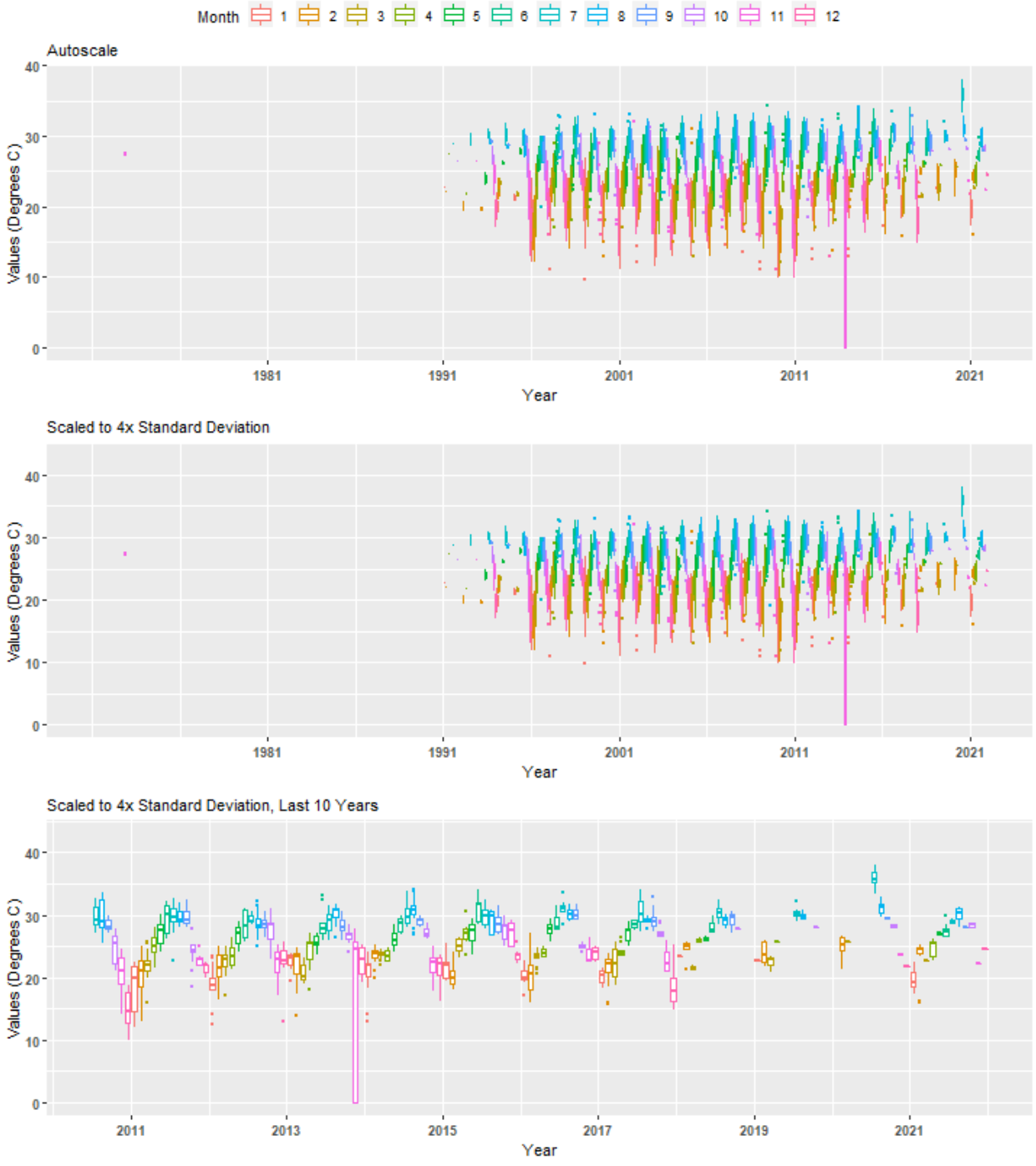






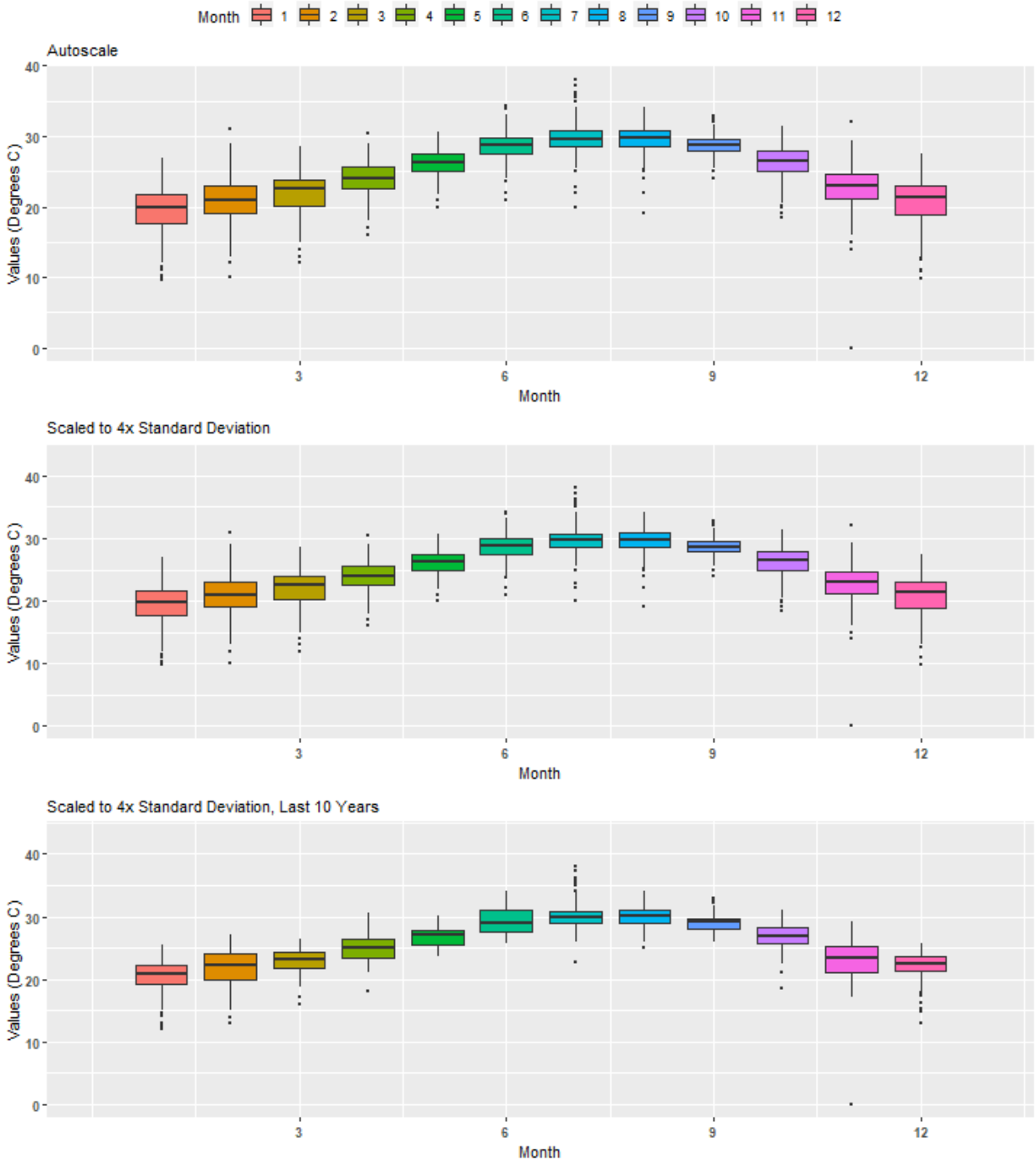
Summary Box Plots for Jensen Beach to Jupiter Inlet Aquatic Preserve

By Year & Month



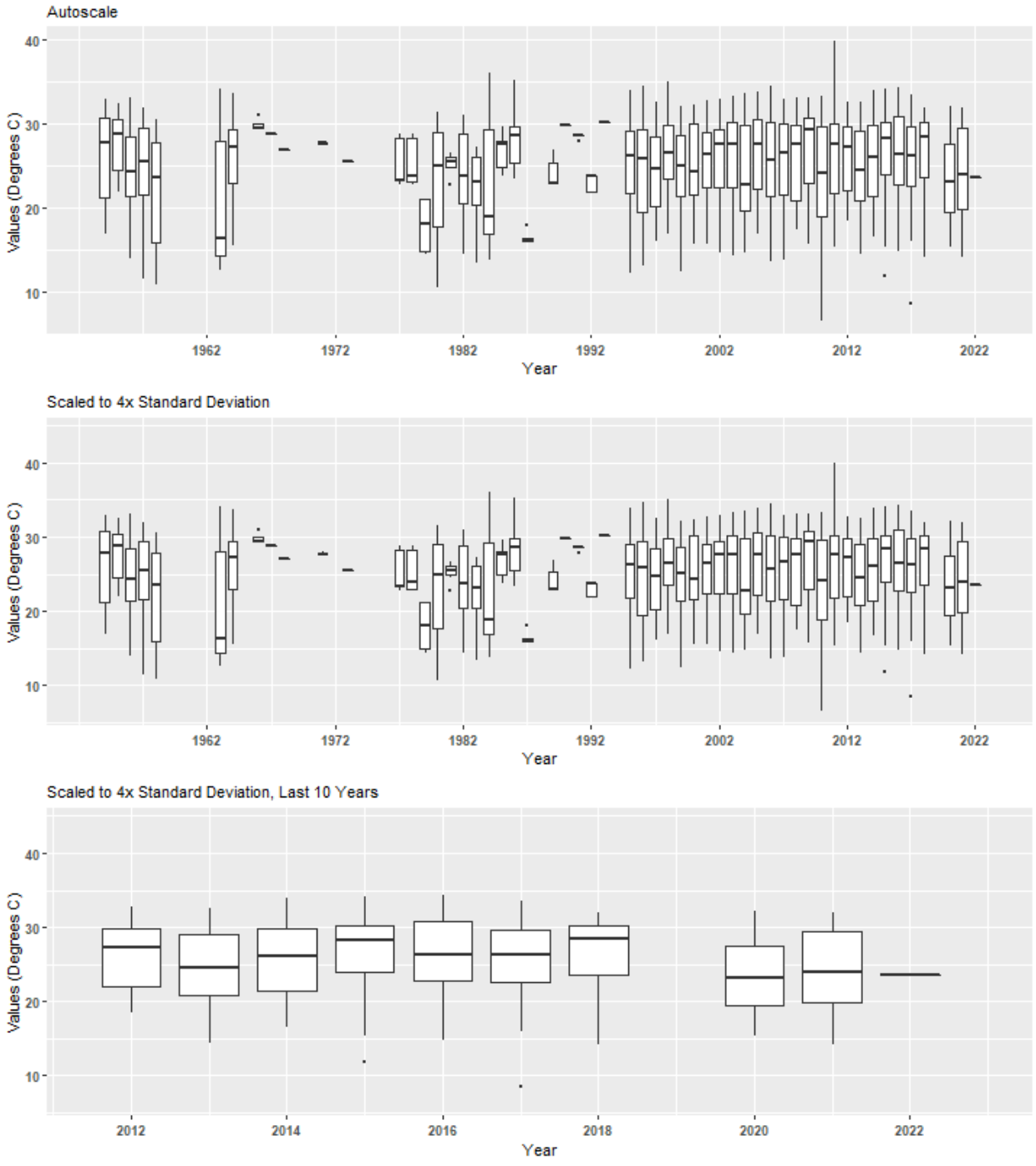
Summary Box Plots for Jensen Beach to Jupiter Inlet Aquatic Preserve

By Month



Summary Box Plots for Lemon Bay Aquatic Preserve

By Year

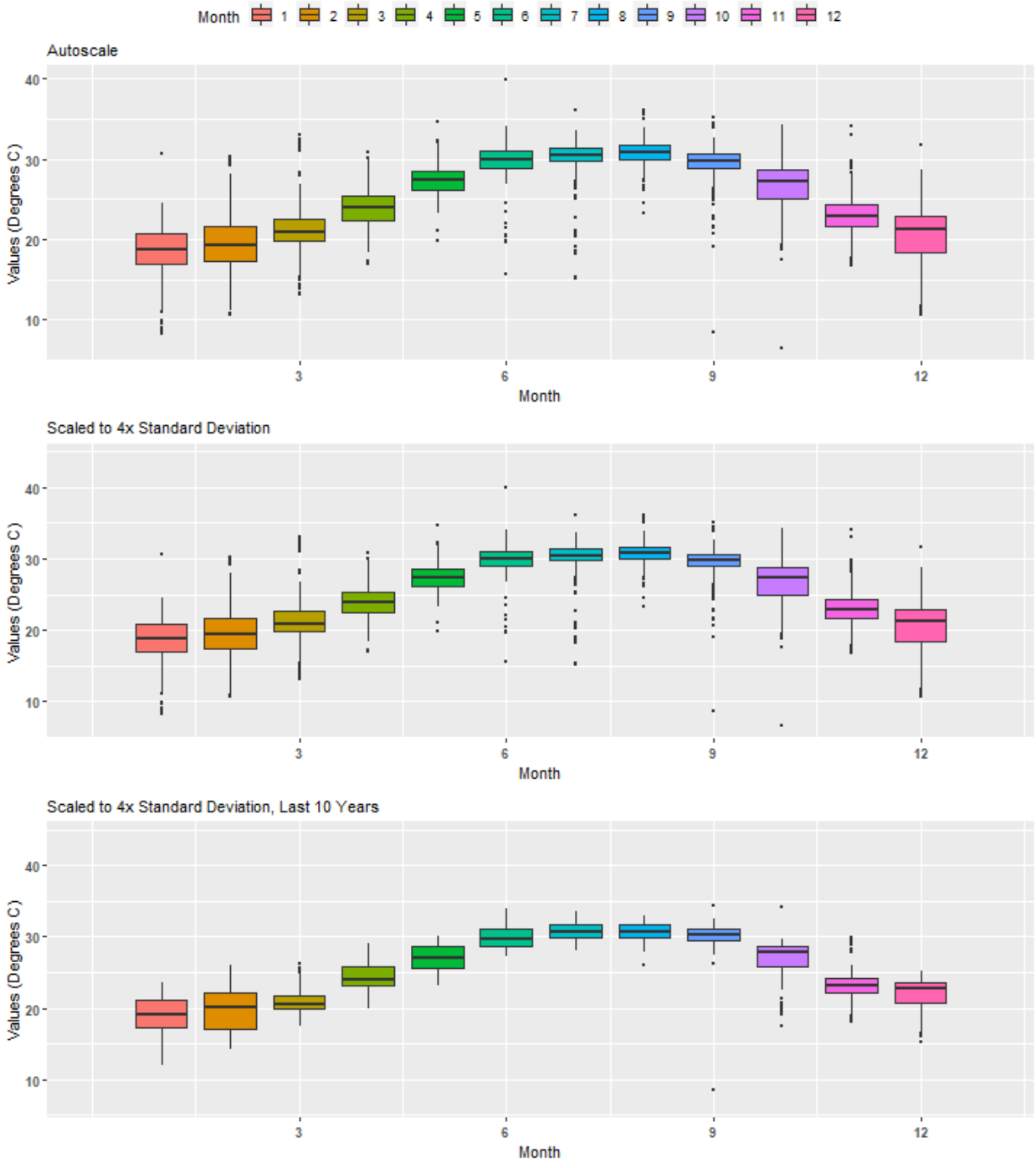


Summary Box Plots for Lemon Bay Aquatic Preserve

By Year & Month

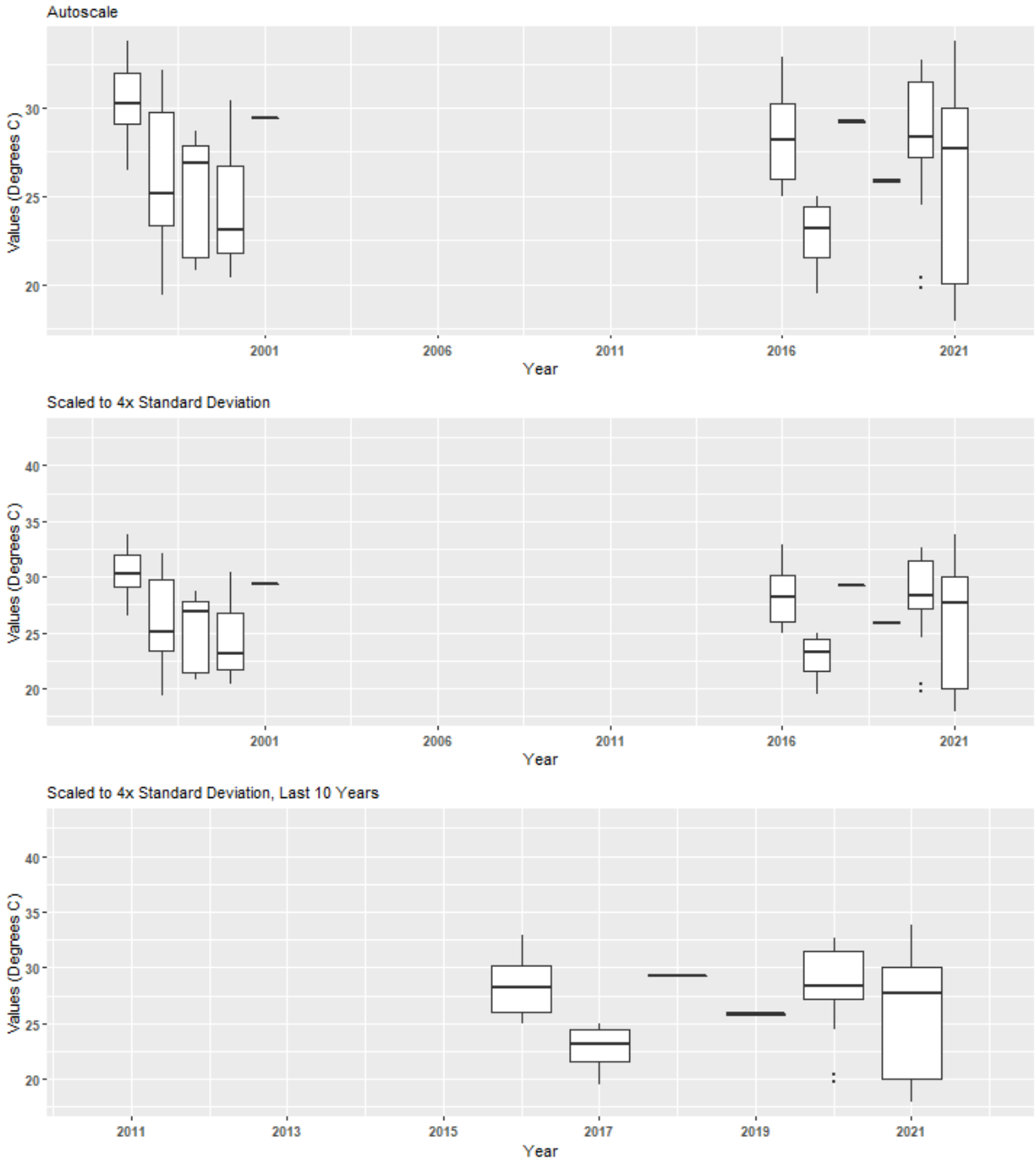
Summary Box Plots for Lemon Bay Aquatic Preserve

By Month



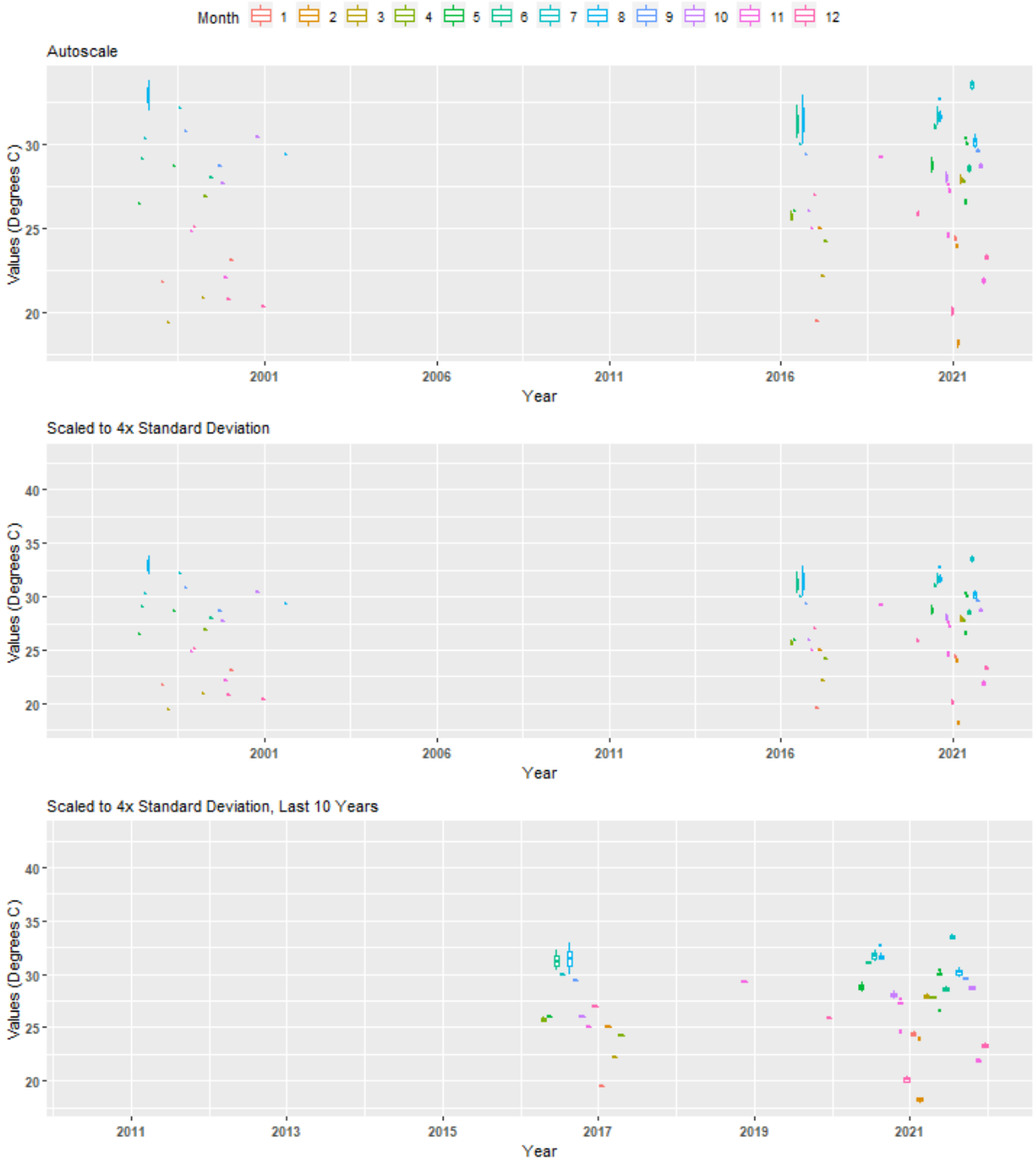
Summary Box Plots for Lignumvitae Key Aquatic Preserve

By Year



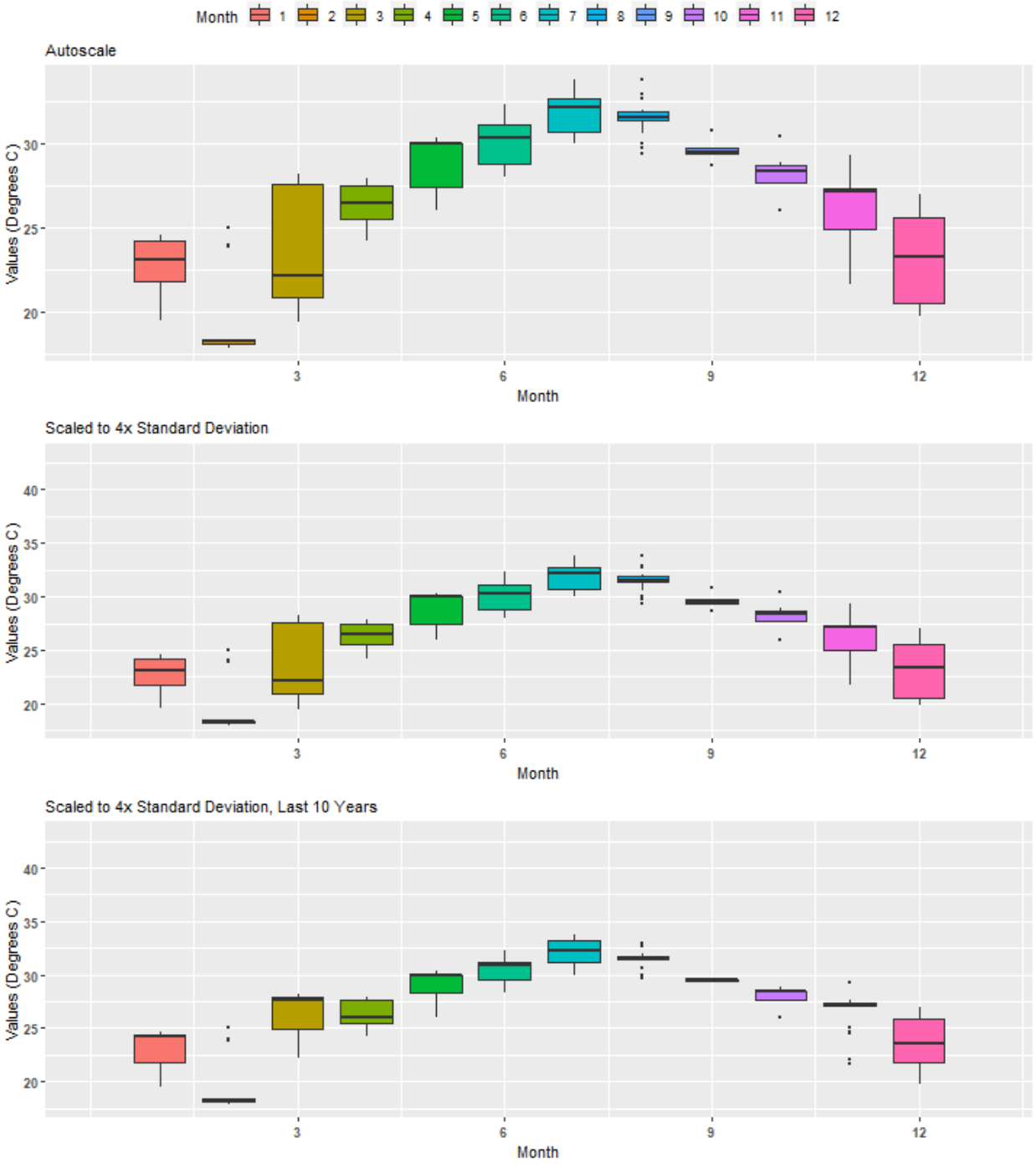
Summary Box Plots for Lignumvitae Key Aquatic Preserve

By Year & Month



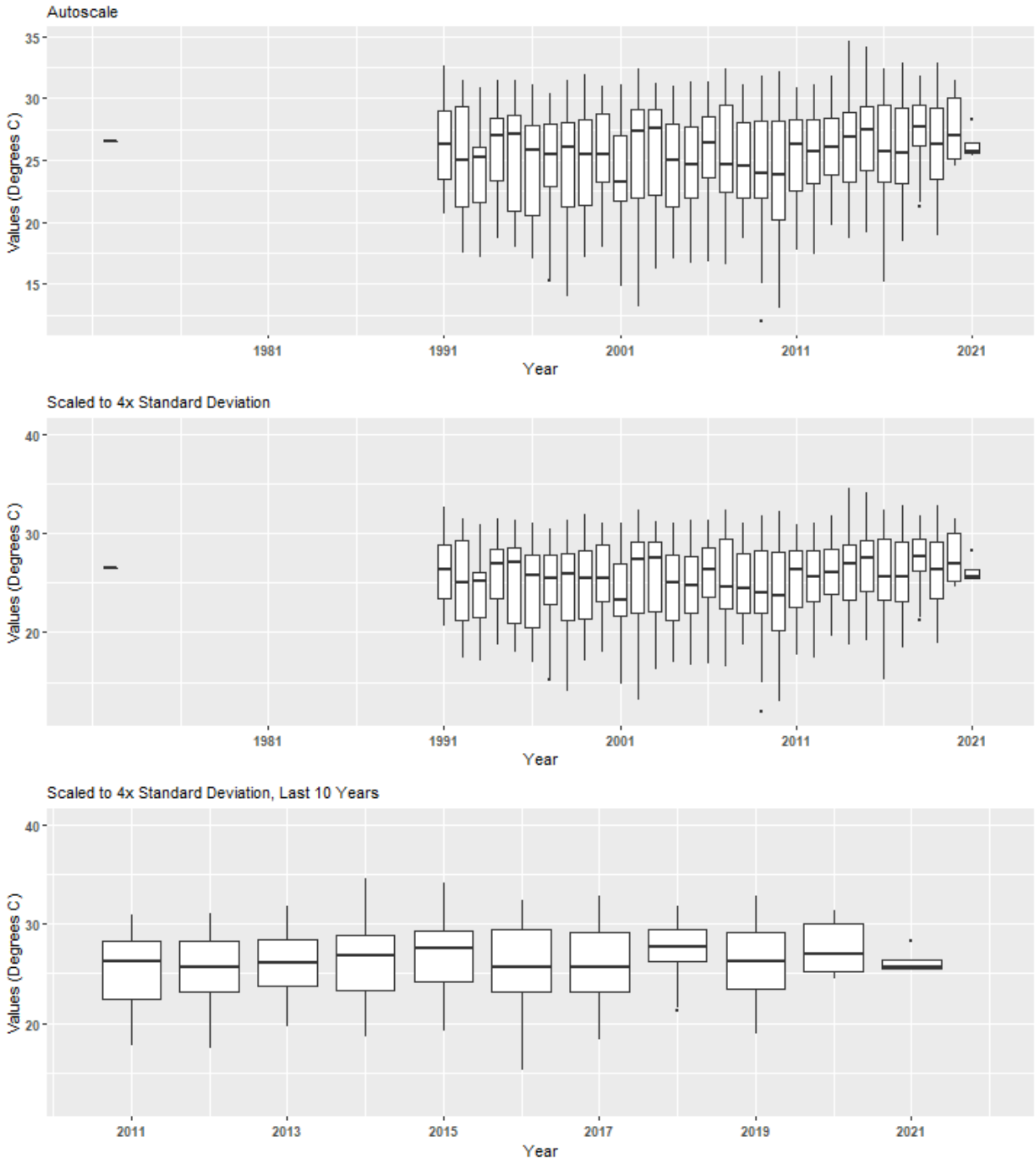
Summary Box Plots for Lignumvitae Key Aquatic Preserve

By Month



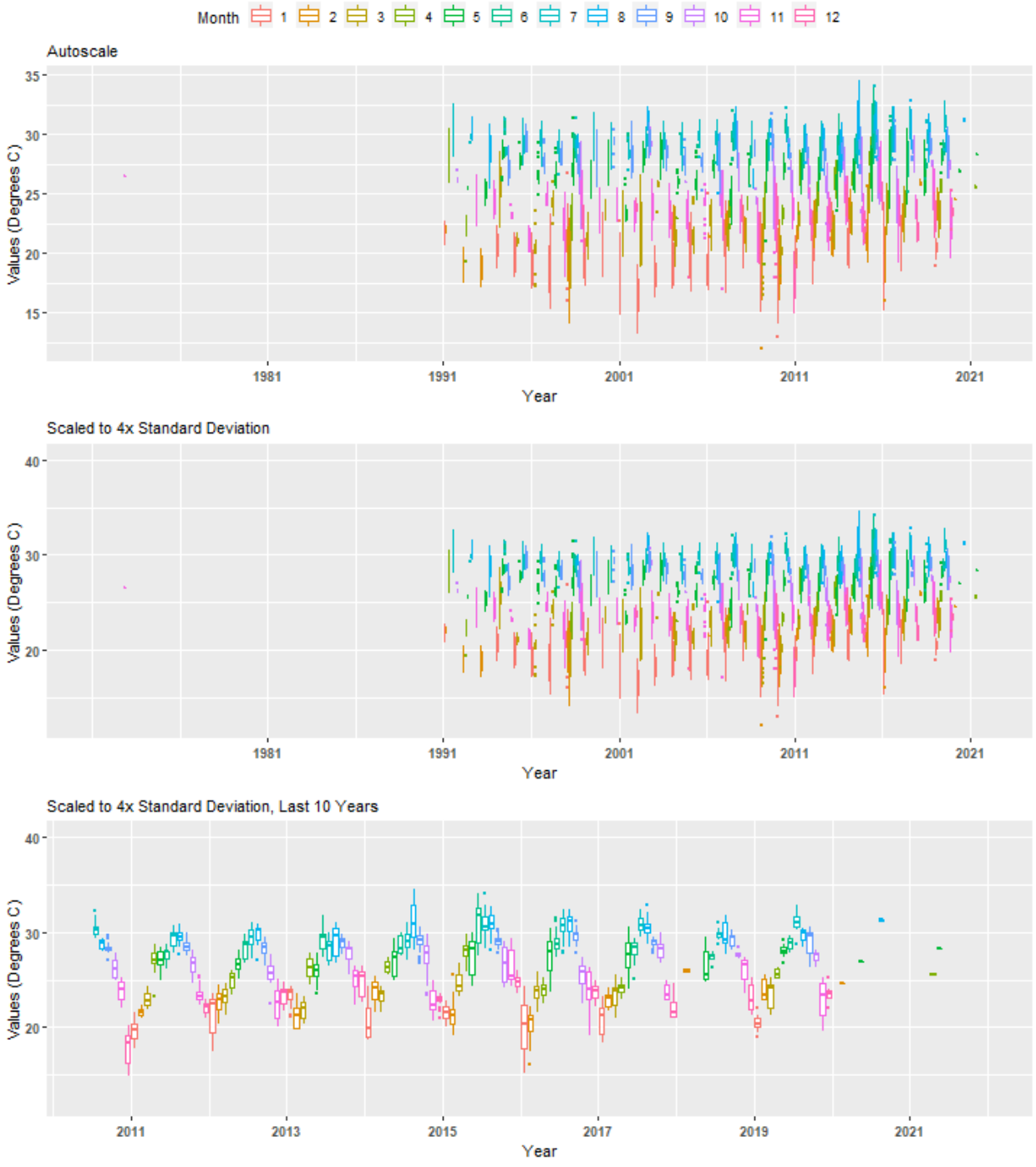
Summary Box Plots for Loxahatchee River-Lake Worth Creek Aquatic Preserve

By Year



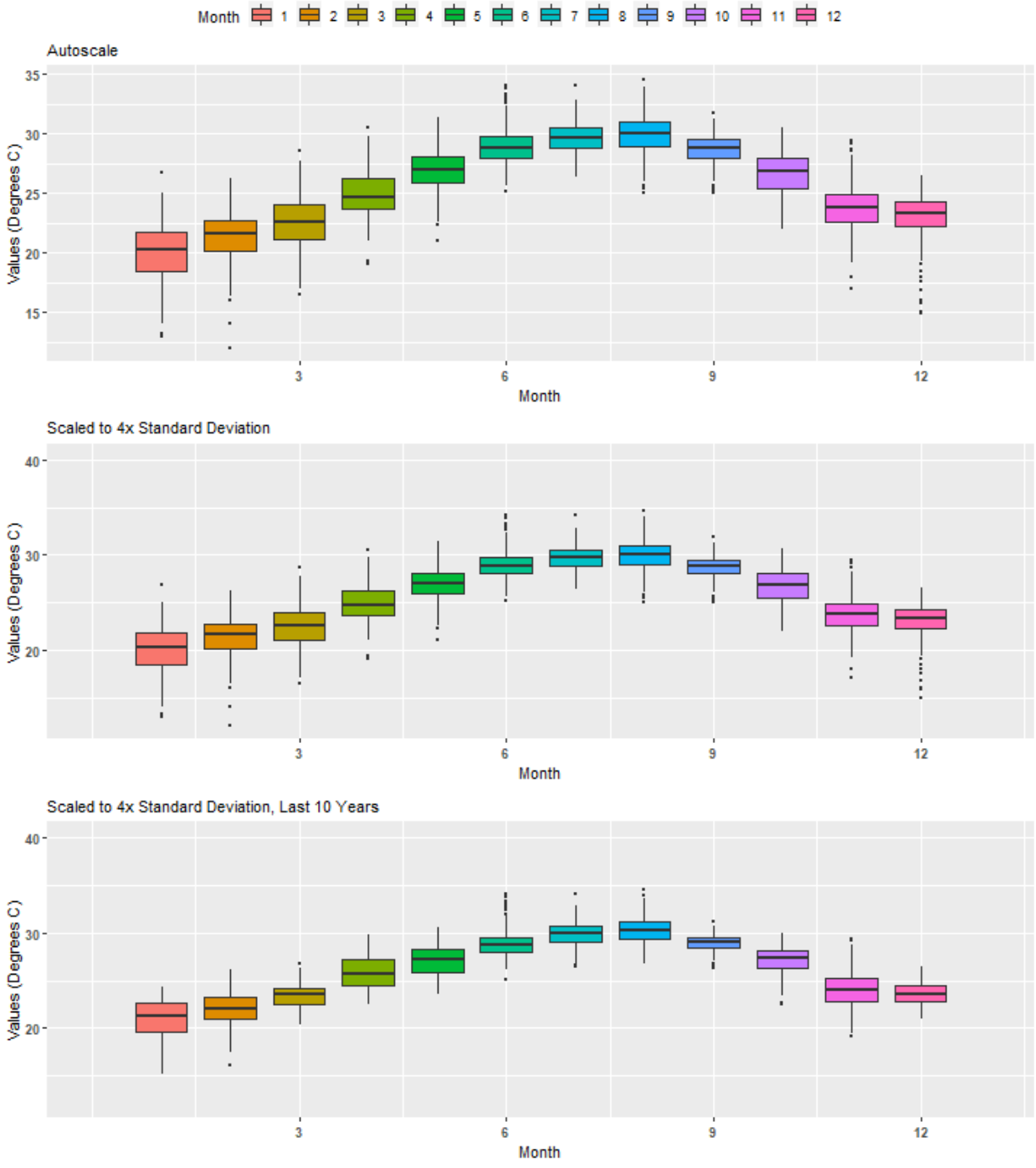
Summary Box Plots for Loxahatchee River-Lake Worth Creek Aquatic Preserve

By Year & Month



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

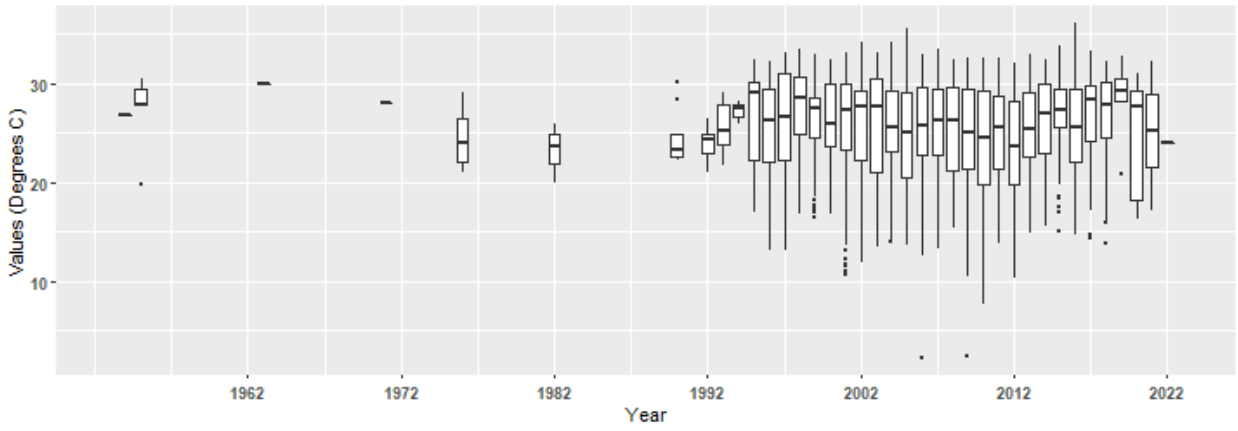
By Month



Summary Box Plots for Matlacha Pass Aquatic Preserve

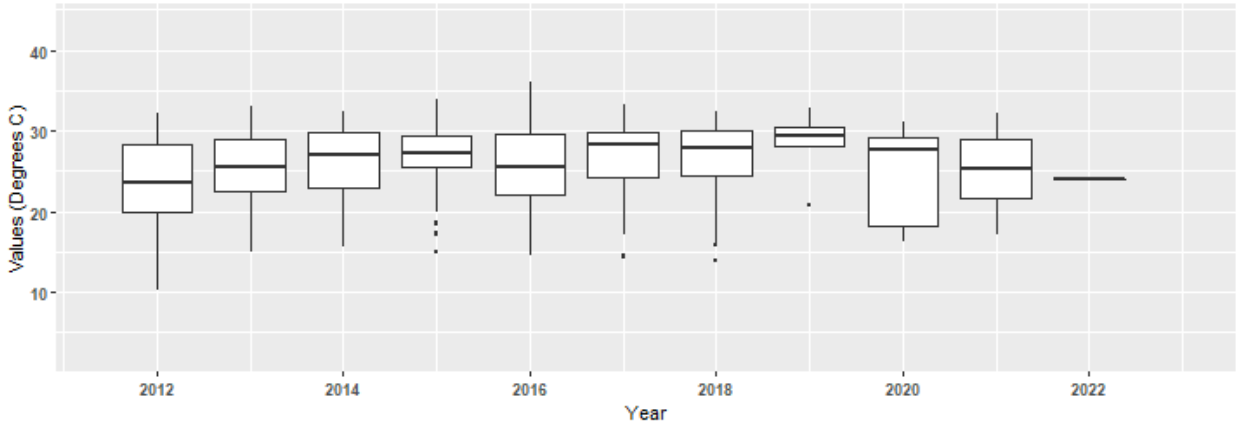
By Year

Autoscale



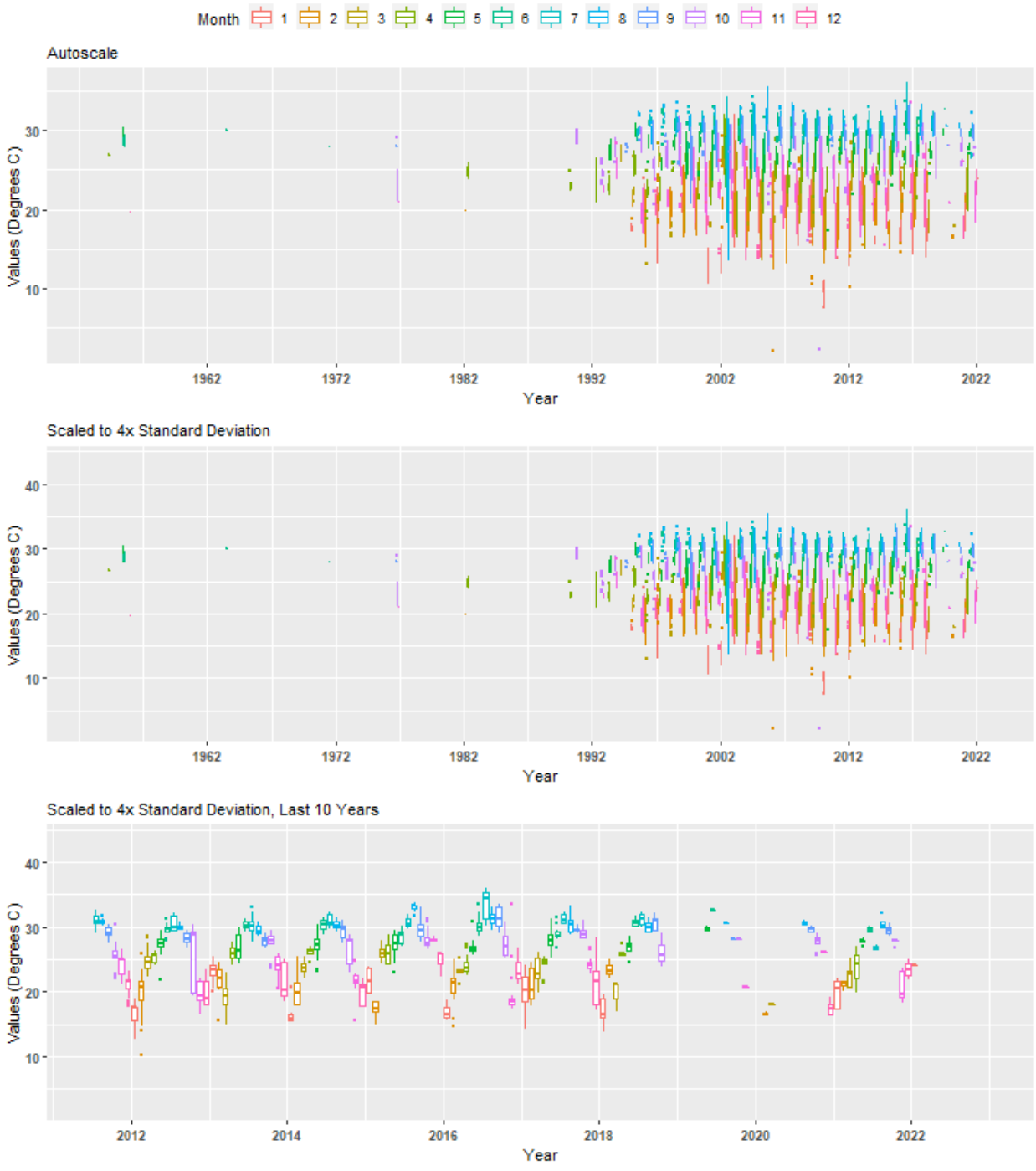
Scaled to 4x Standard Deviation

Scaled to 4x Standard Deviation, Last 10 Years



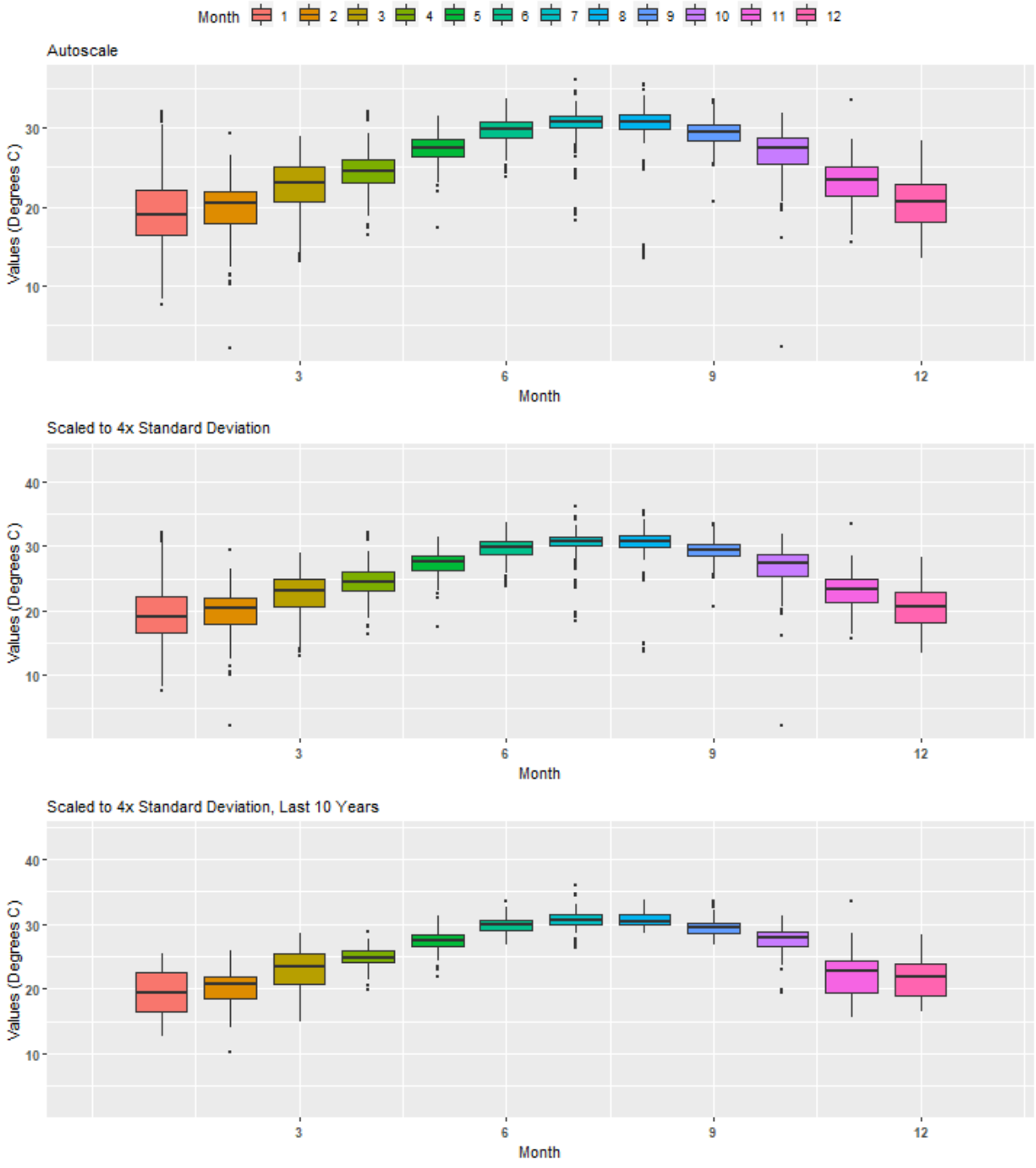
Summary Box Plots for Matlacha Pass Aquatic Preserve

By Year & Month



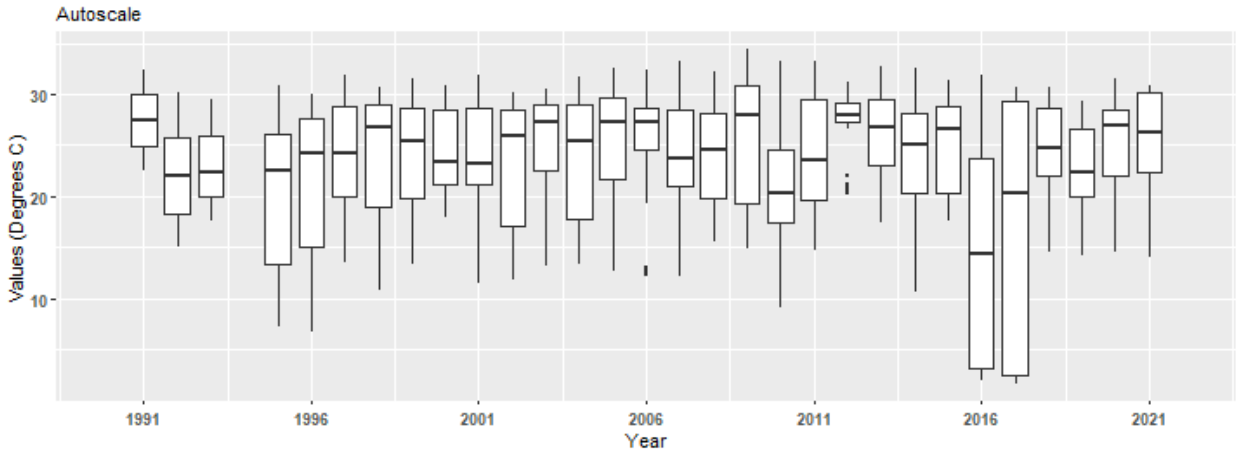
Summary Box Plots for Matlacha Pass Aquatic Preserve

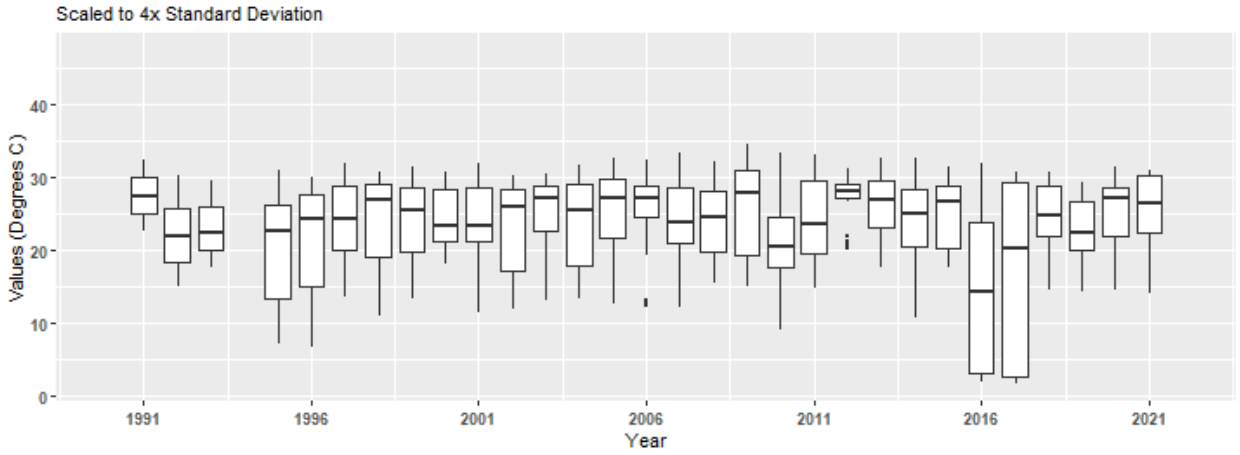
By Month

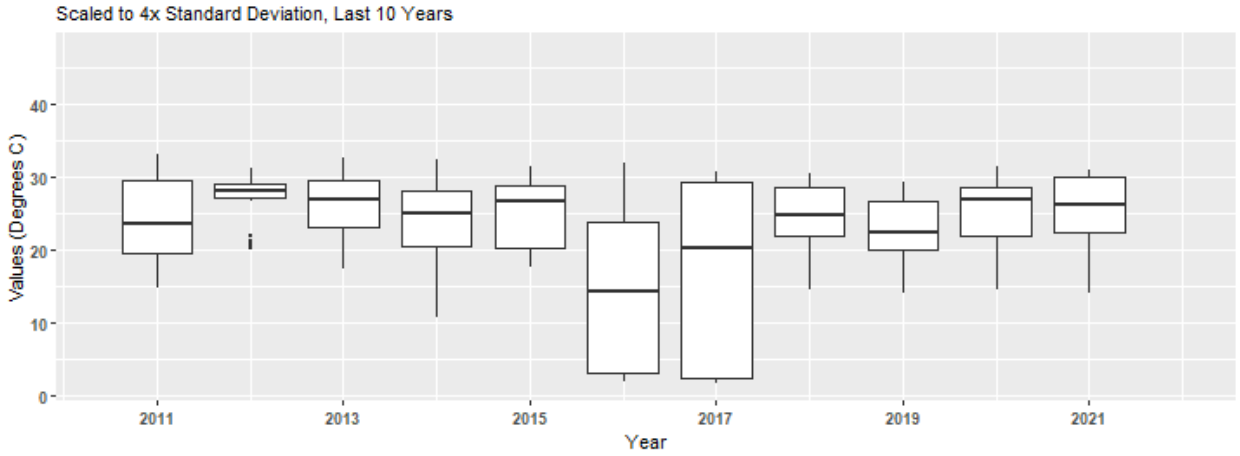


Summary Box Plots for Mosquito Lagoon Aquatic Preserve

By Year





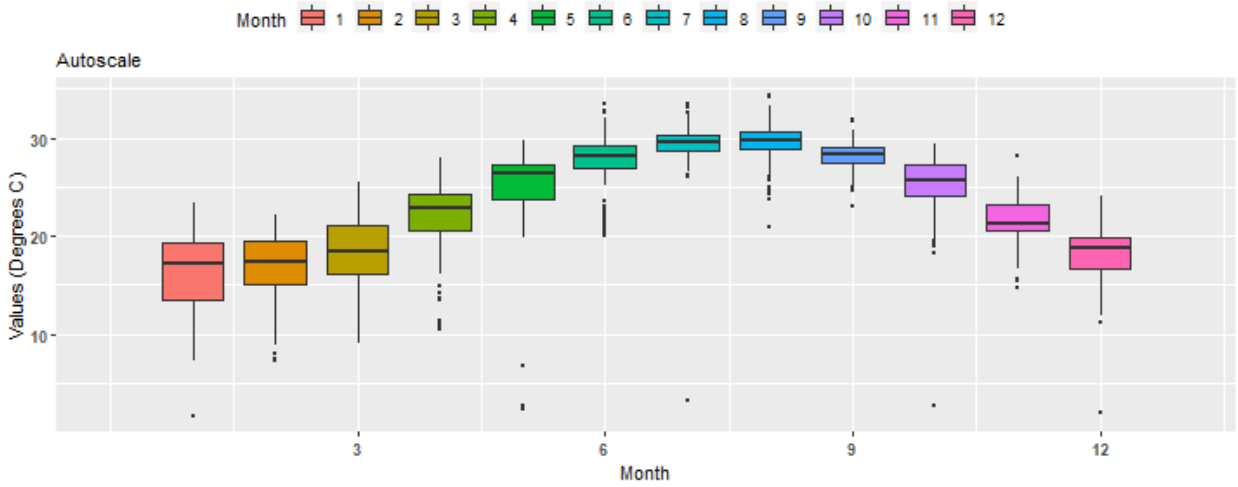


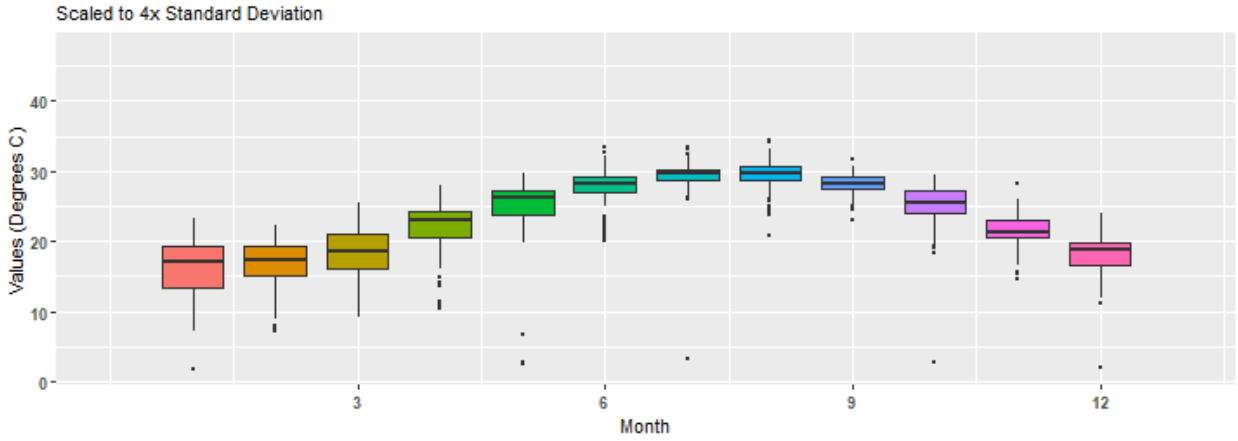
Summary Box Plots for Mosquito Lagoon Aquatic Preserve

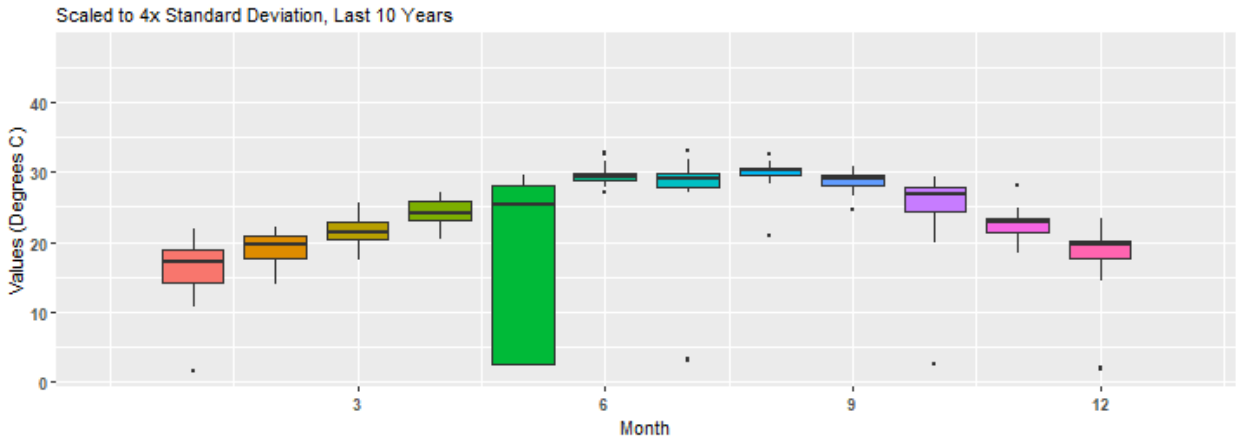
By Year & Month

Summary Box Plots for Mosquito Lagoon Aquatic Preserve

By Month

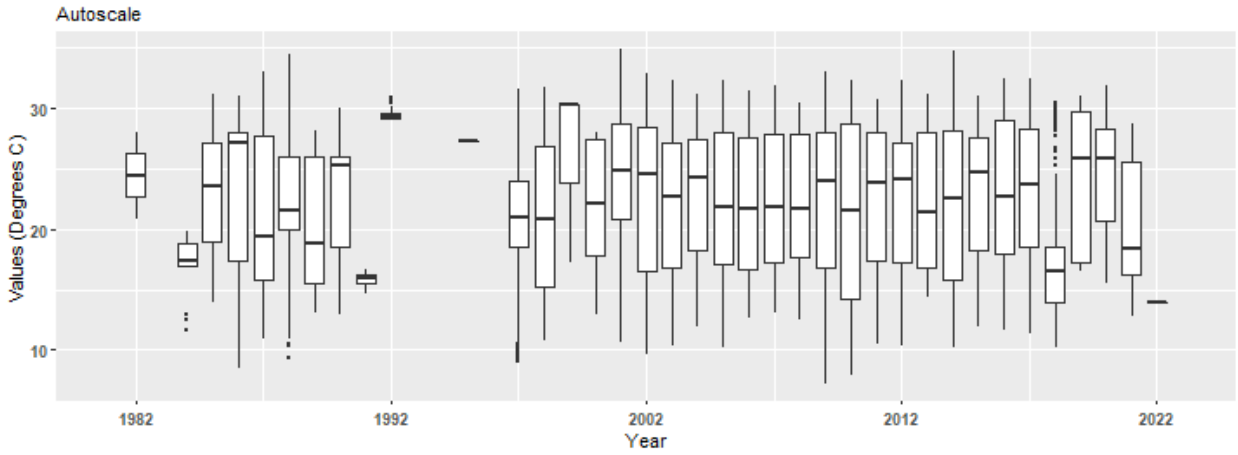


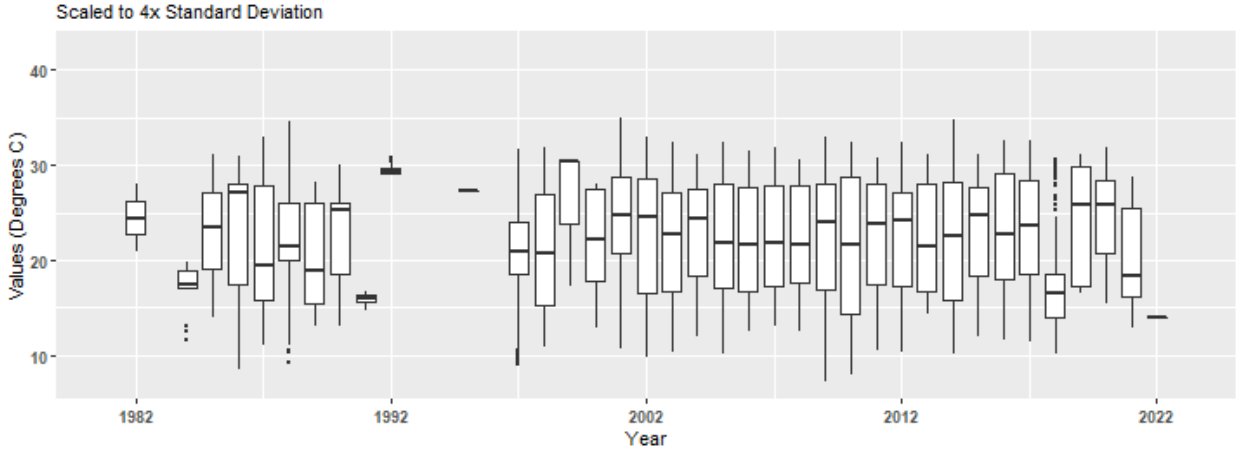


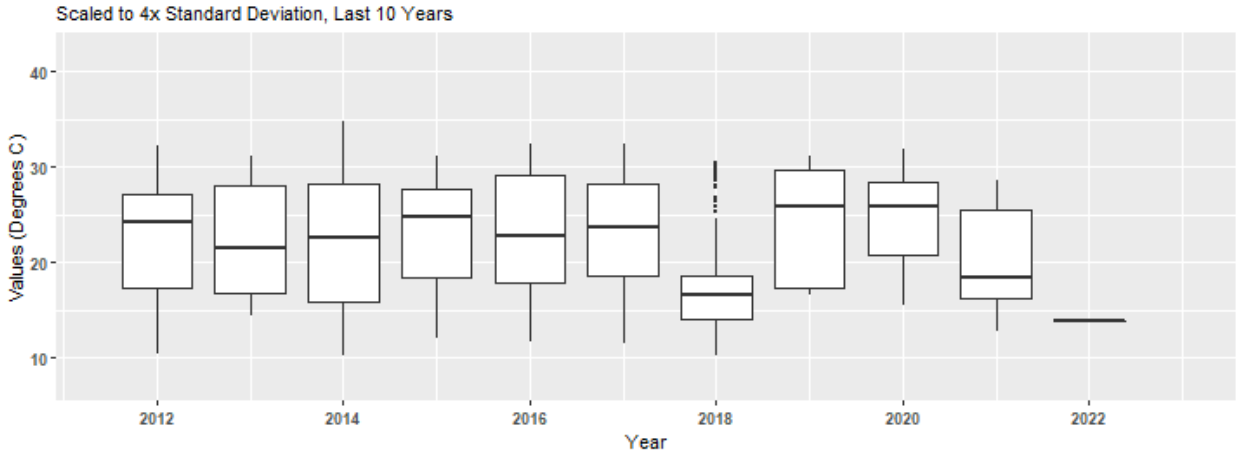


Summary Box Plots for Nassau River-St. Johns River Marshes Aquatic Preserve

By Year

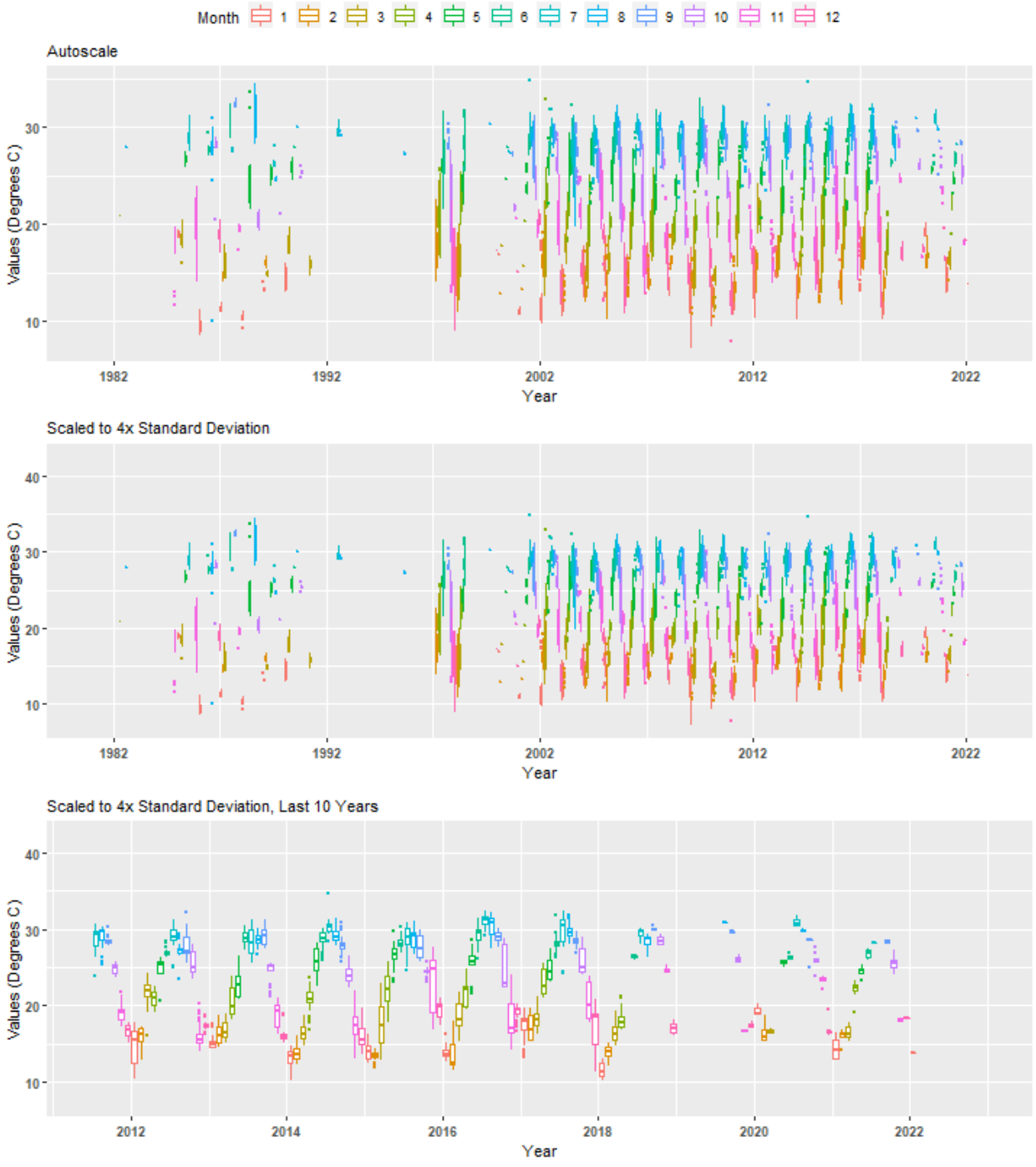






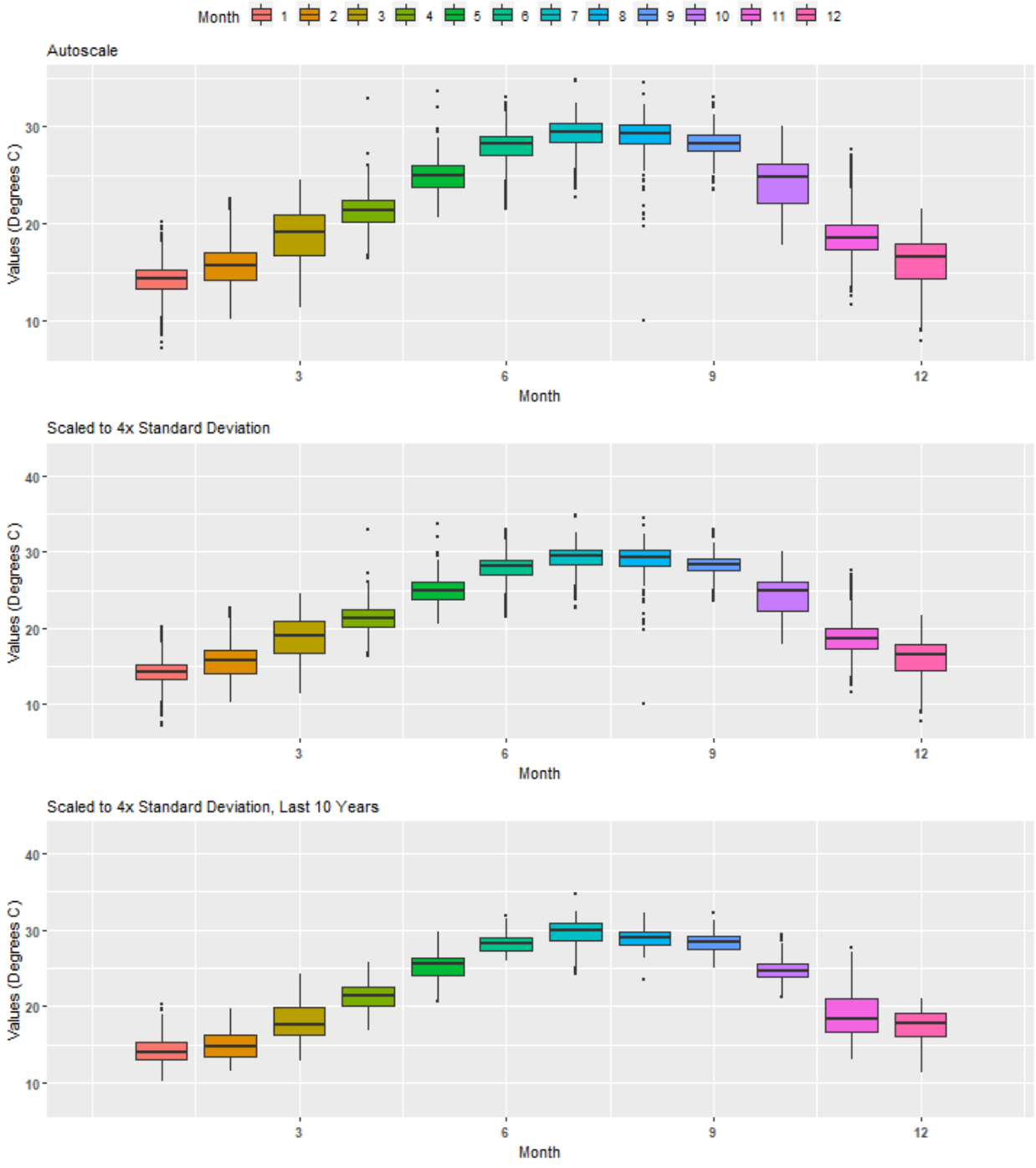
Summary Box Plots for Nassau River-St. Johns River Marshes Aquatic Preserve

By Year & Month



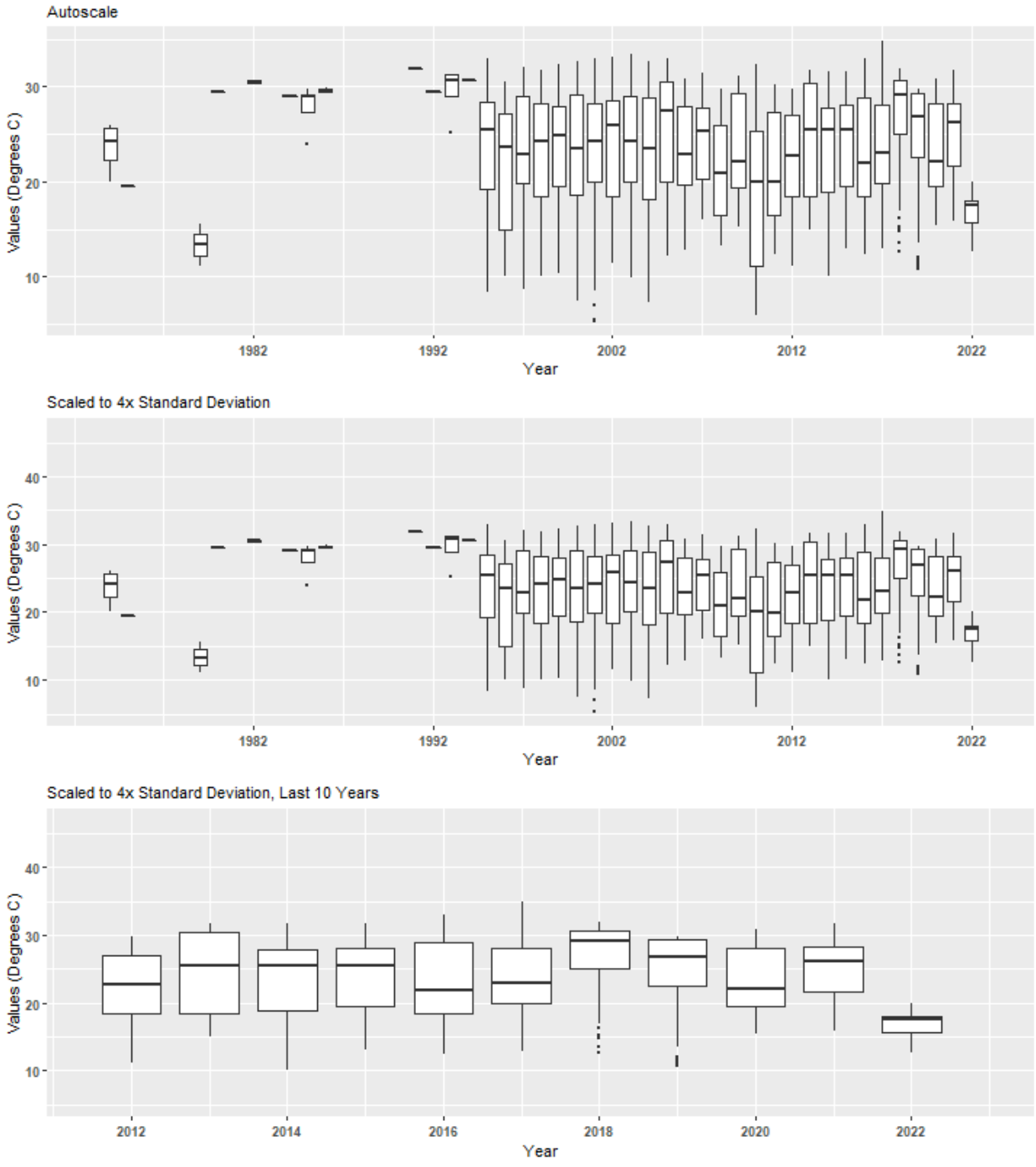
Summary Box Plots for Nassau River-St. Johns River Marshes Aquatic Preserve

By Month



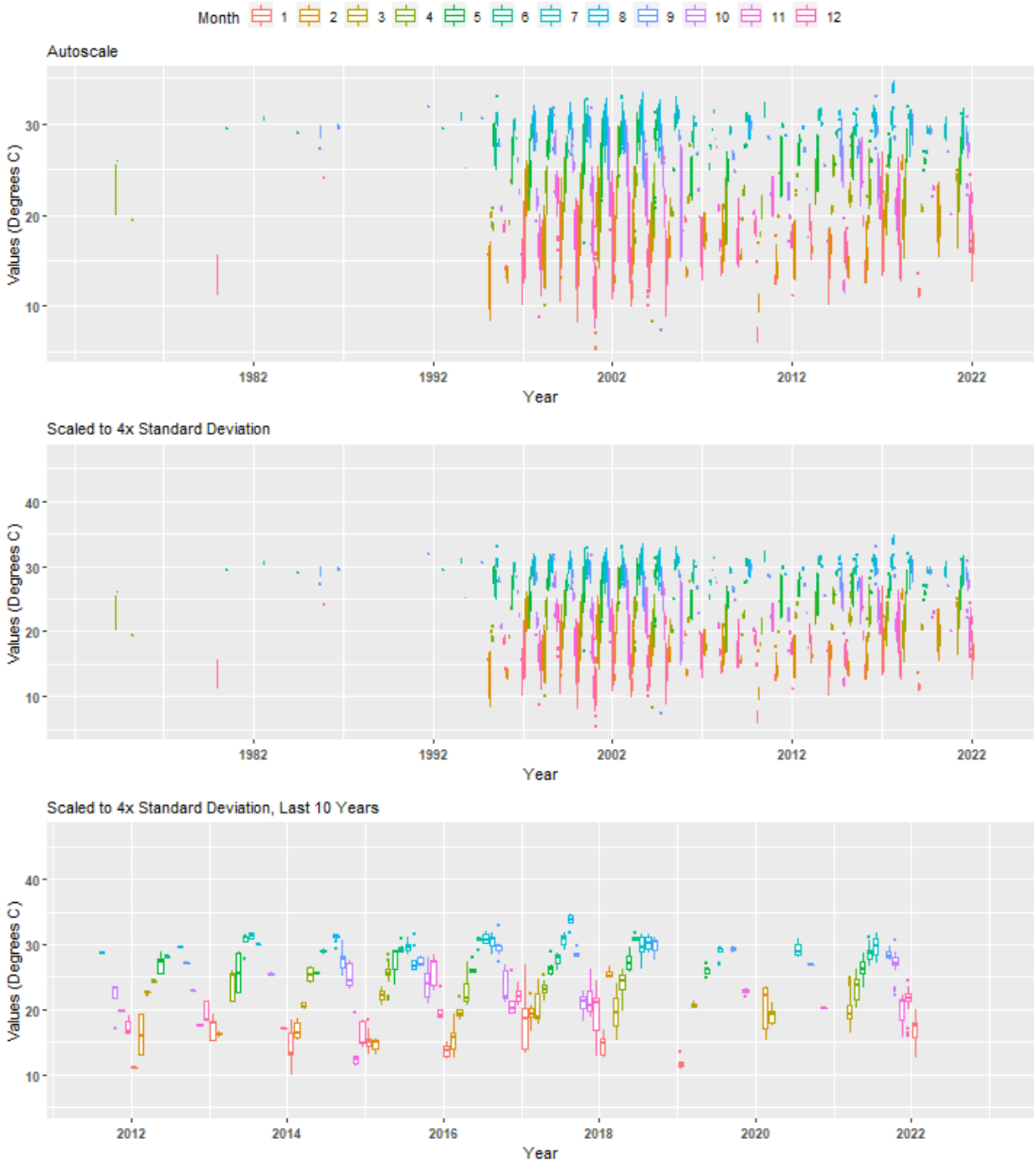
Summary Box Plots for Nature Coast Aquatic Preserve

By Year



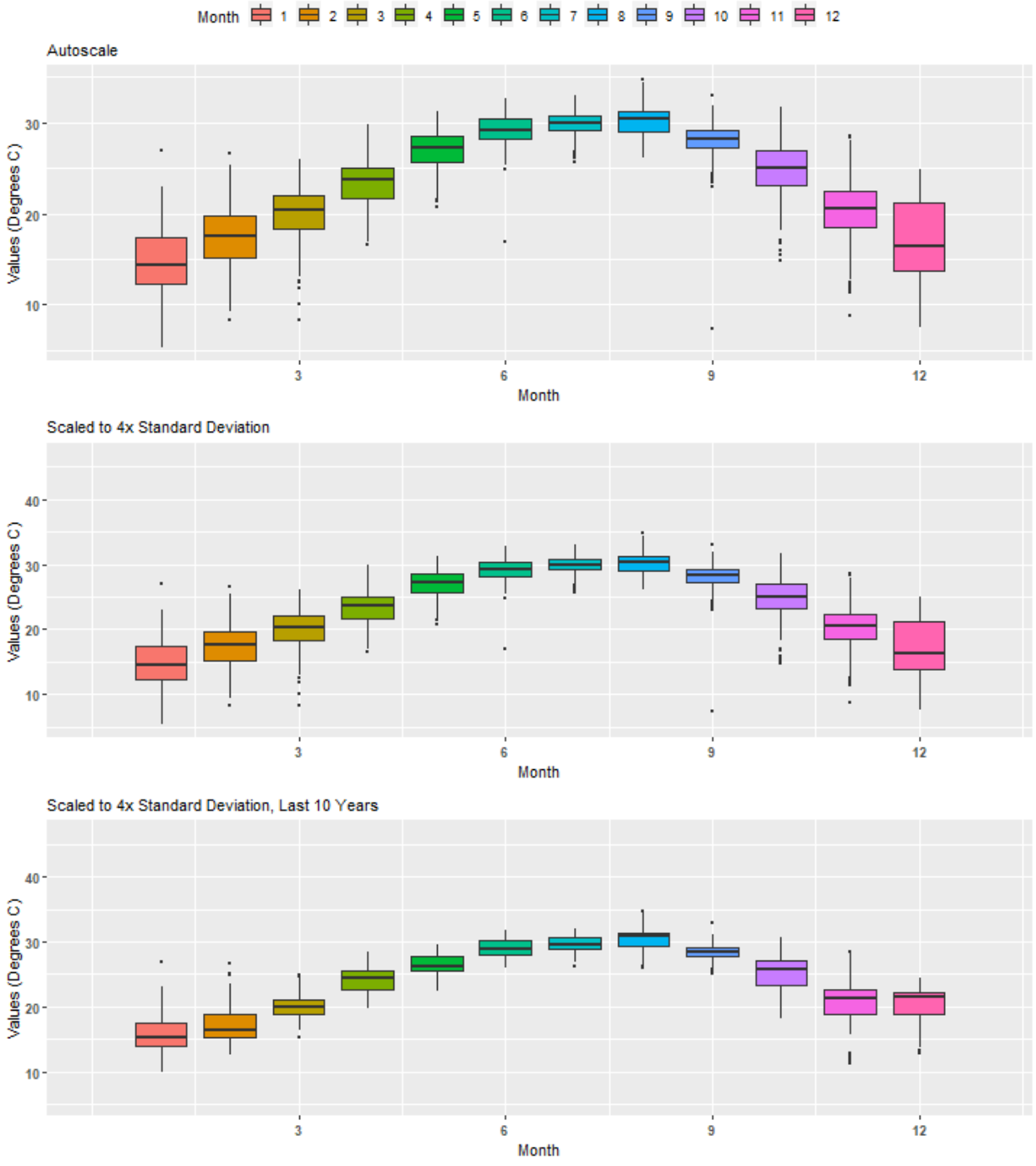
Summary Box Plots for Nature Coast Aquatic Preserve

By Year & Month



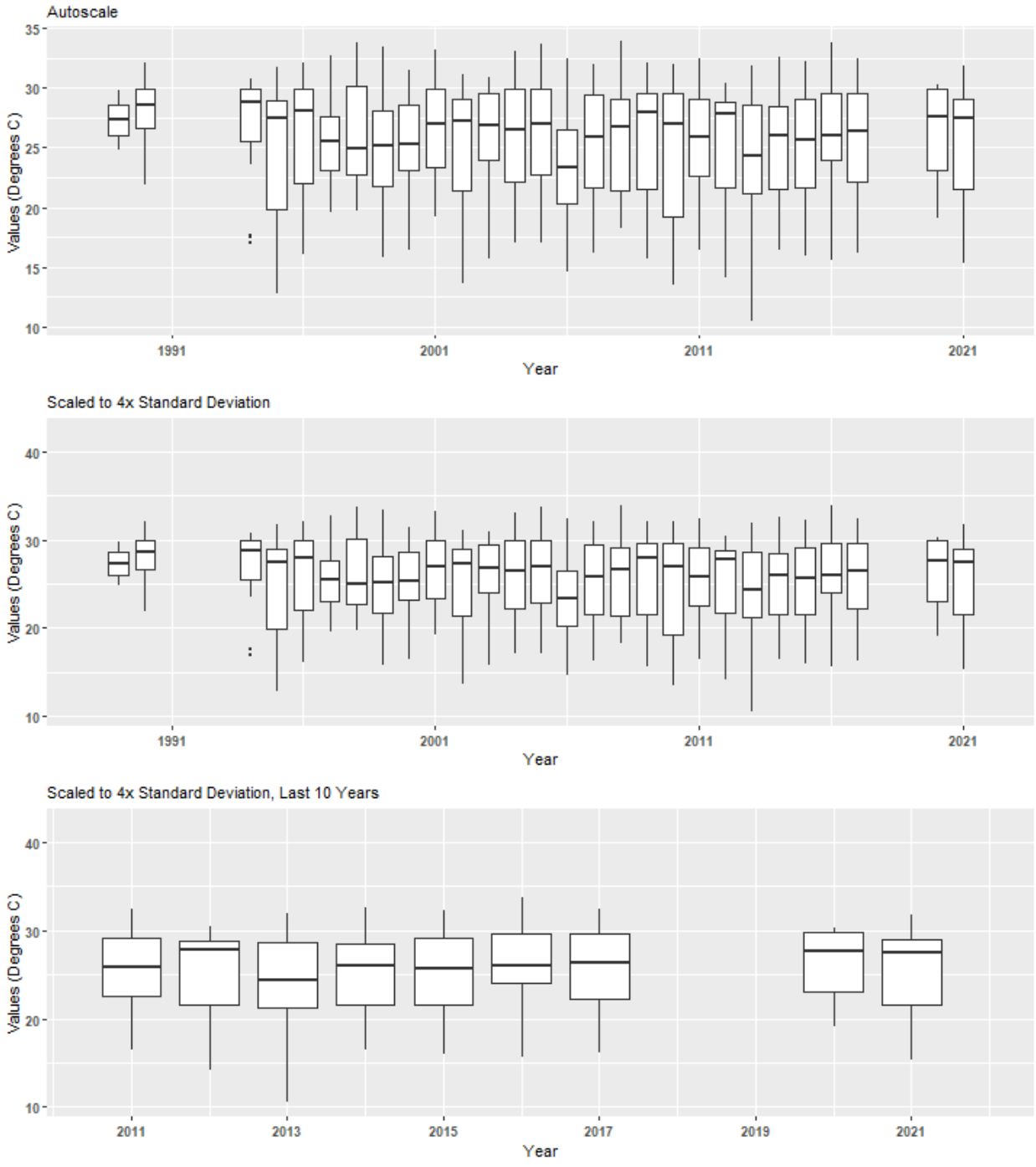
Summary Box Plots for Nature Coast Aquatic Preserve

By Month



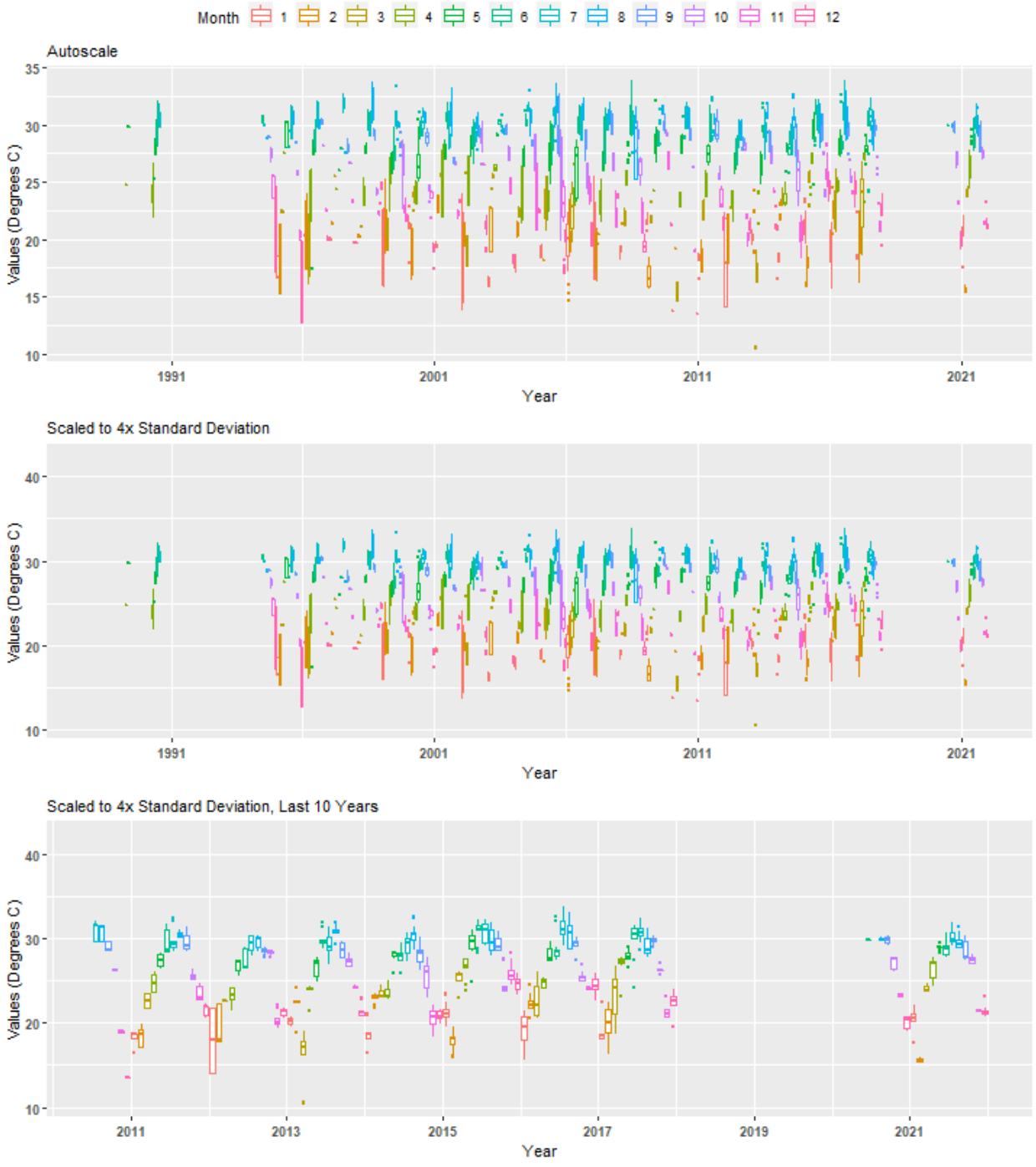
Summary Box Plots for North Fork St. Lucie Aquatic Preserve

By Year



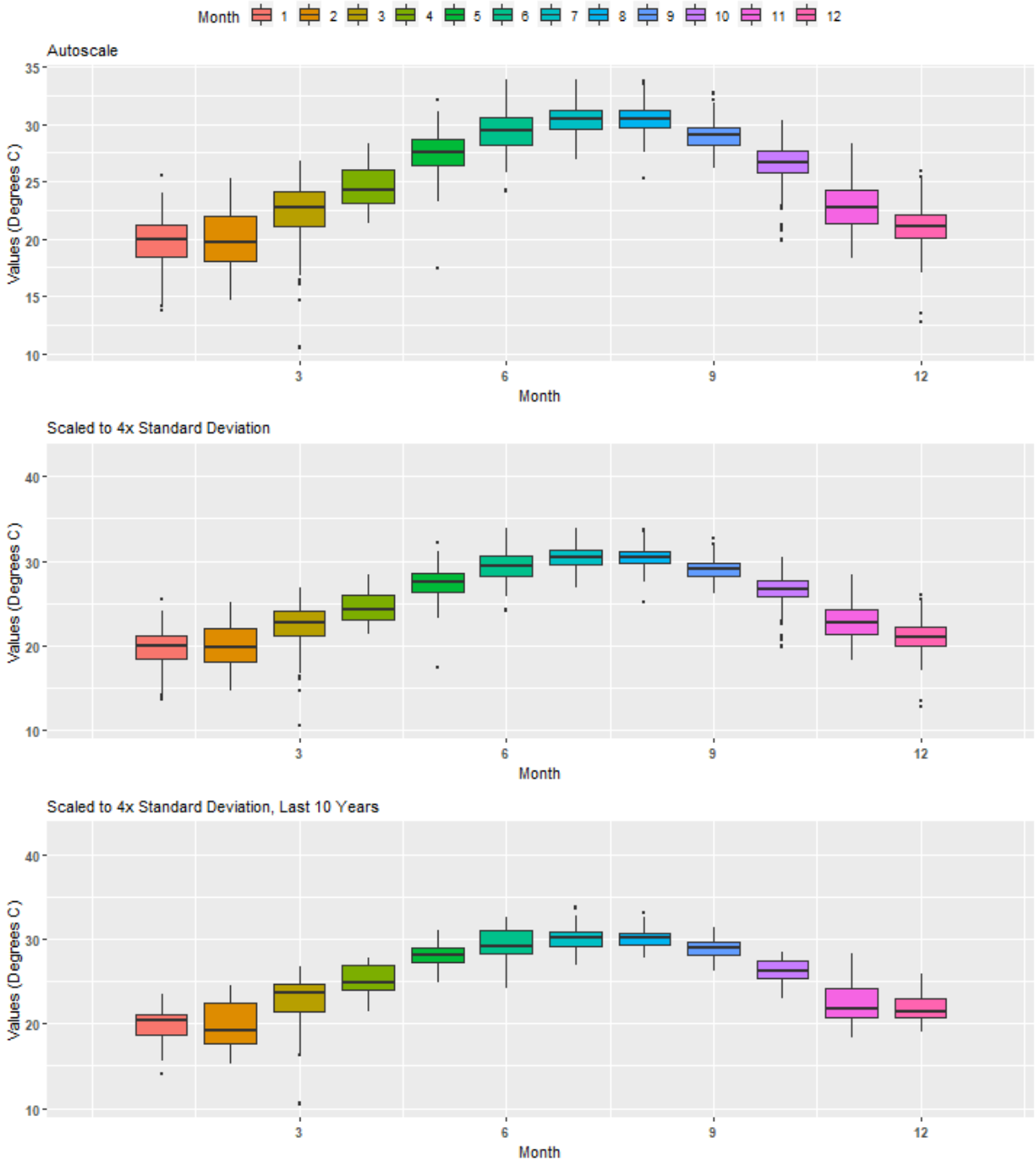
Summary Box Plots for North Fork St. Lucie Aquatic Preserve

By Year & Month



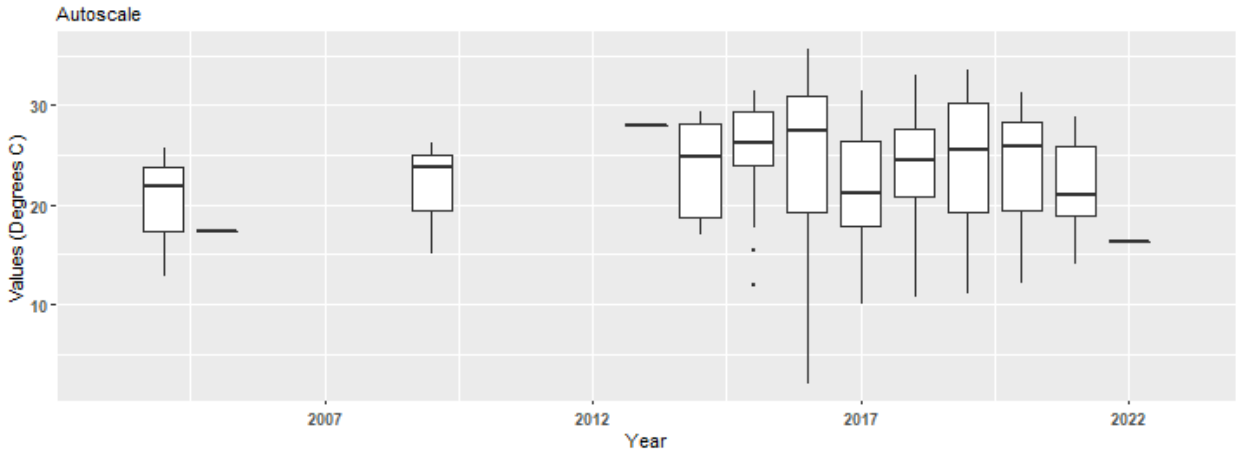
Summary Box Plots for North Fork St. Lucie Aquatic Preserve

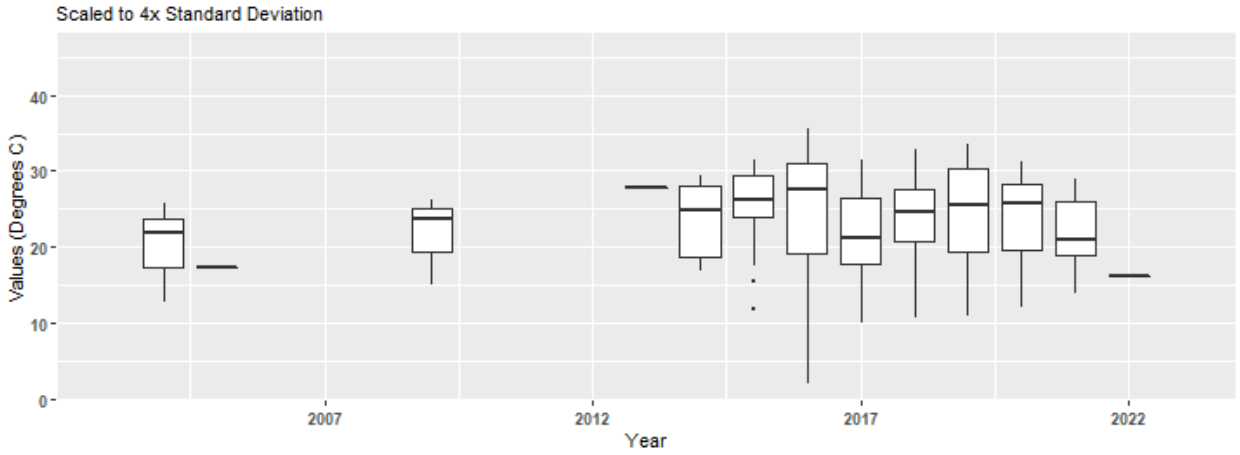
By Month

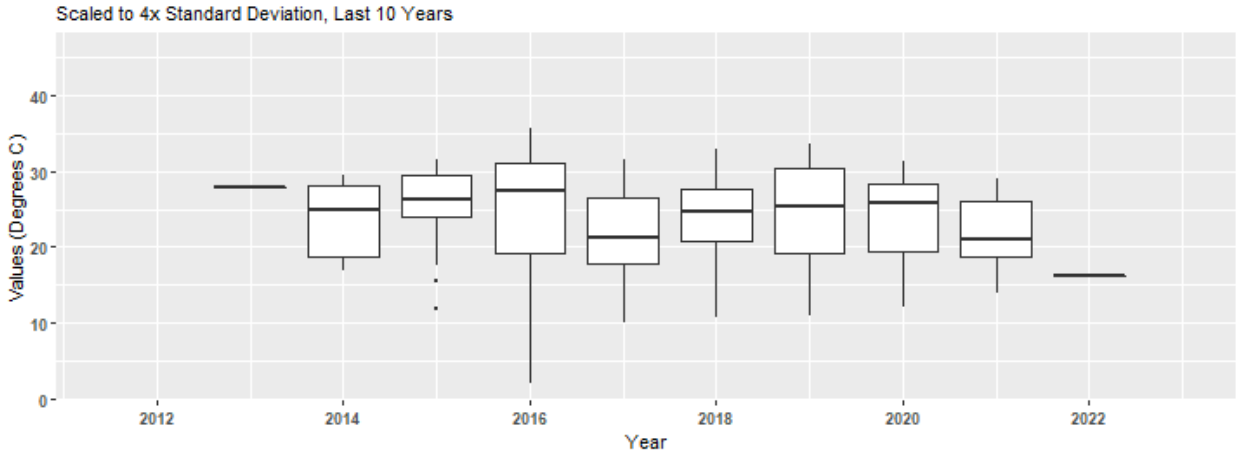


Summary Box Plots for Pellicer Creek Aquatic Preserve

By Year

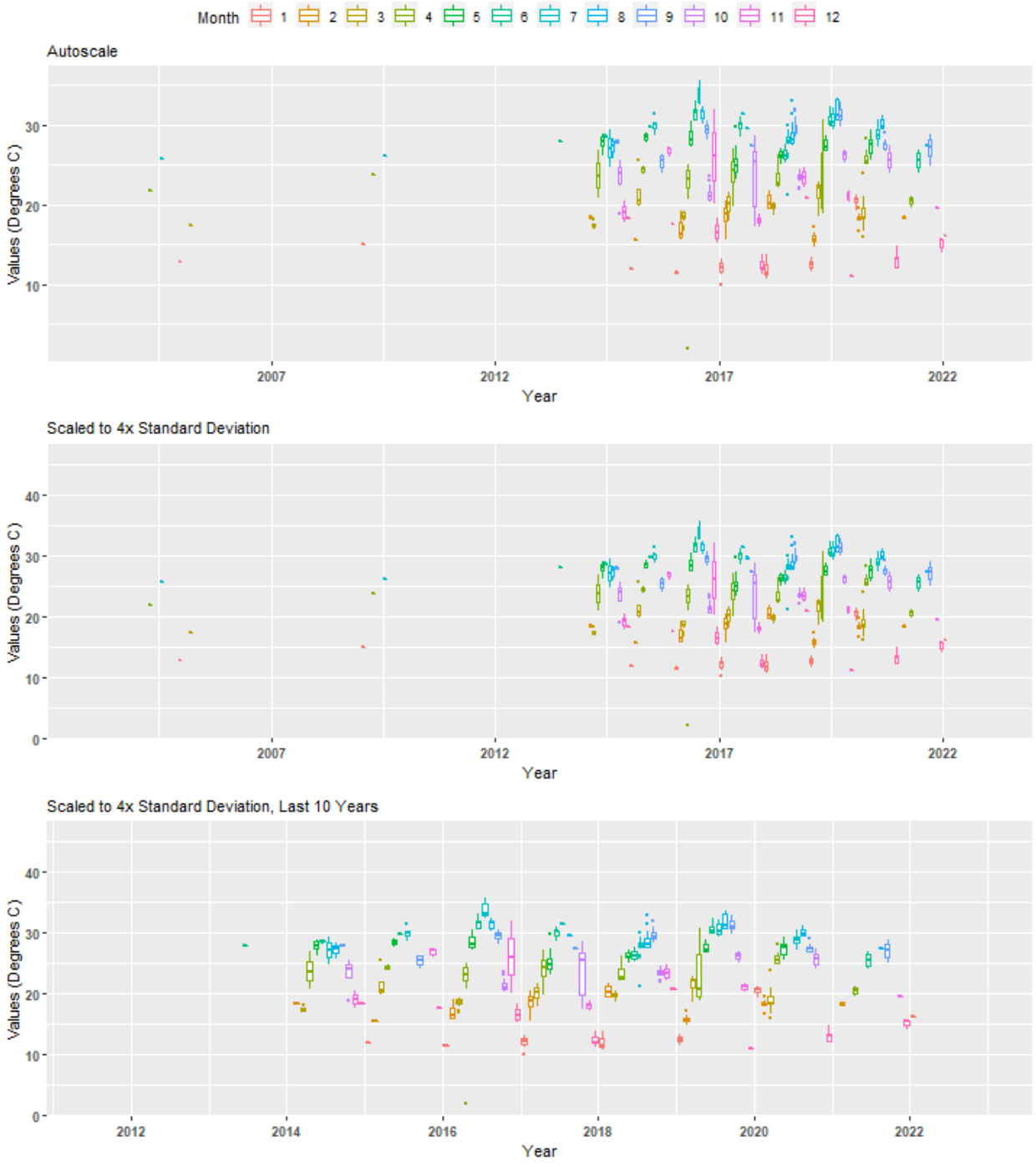






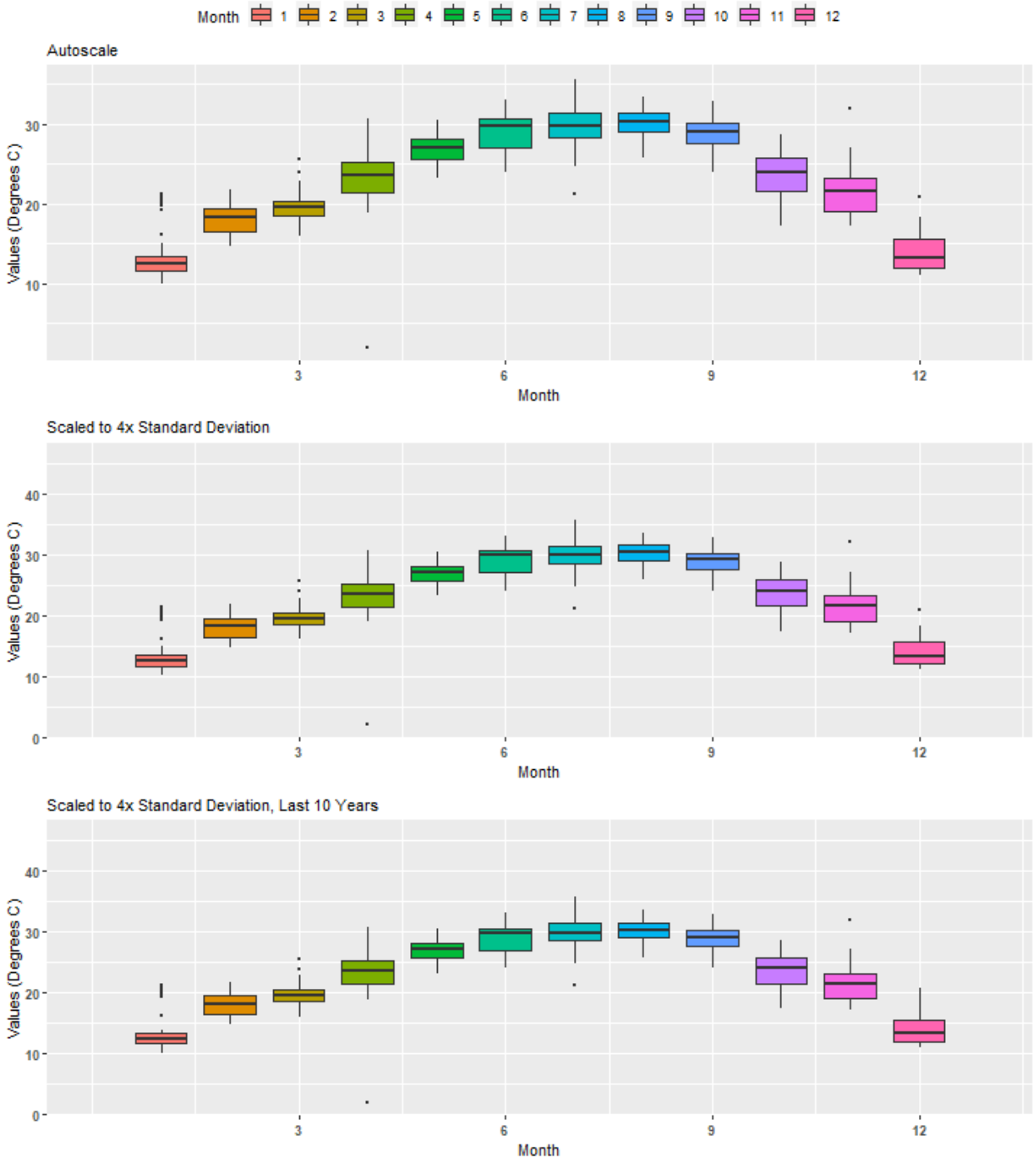
Summary Box Plots for Pellicer Creek Aquatic Preserve

By Year & Month



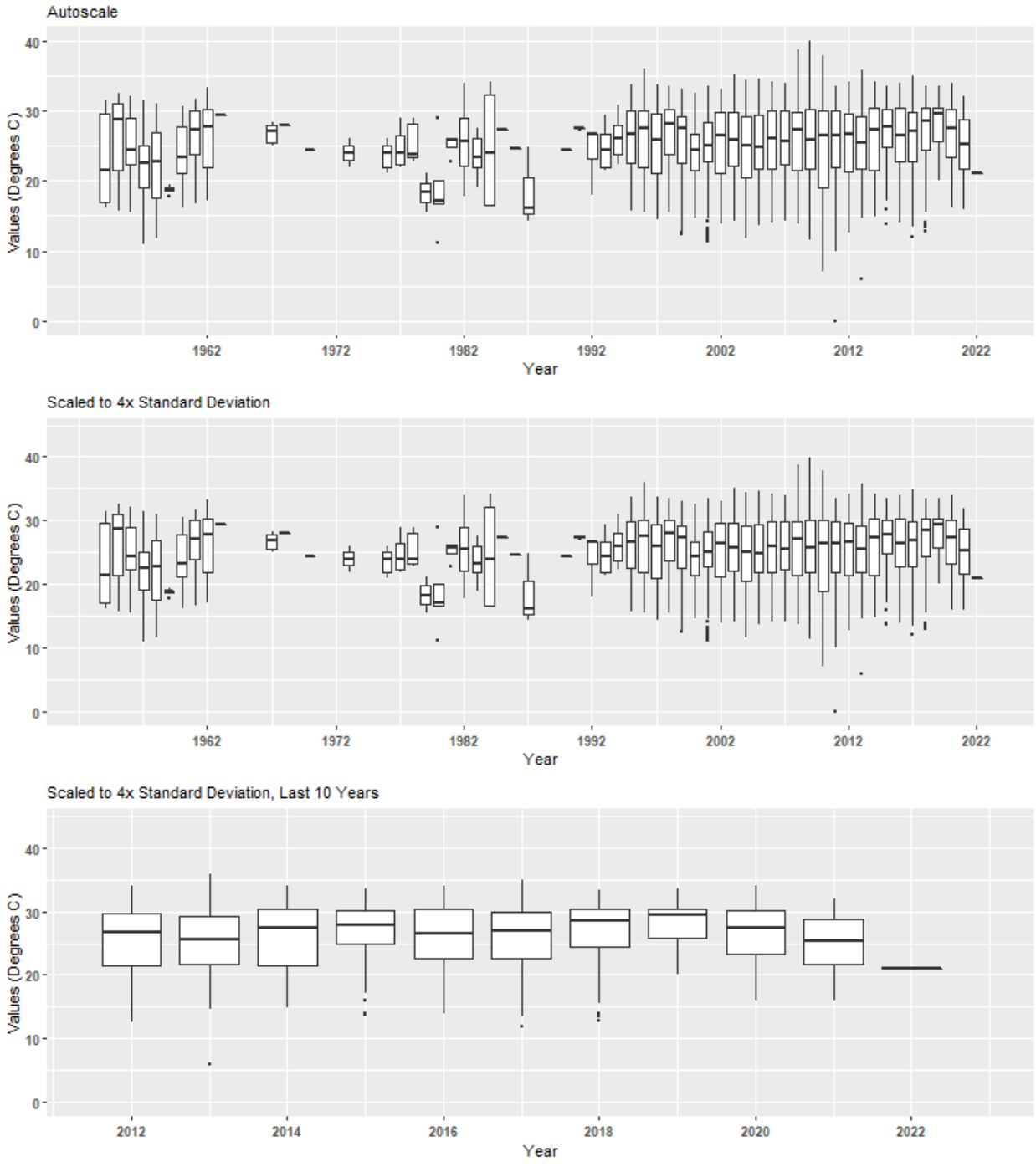
Summary Box Plots for Pellicer Creek Aquatic Preserve

By Month



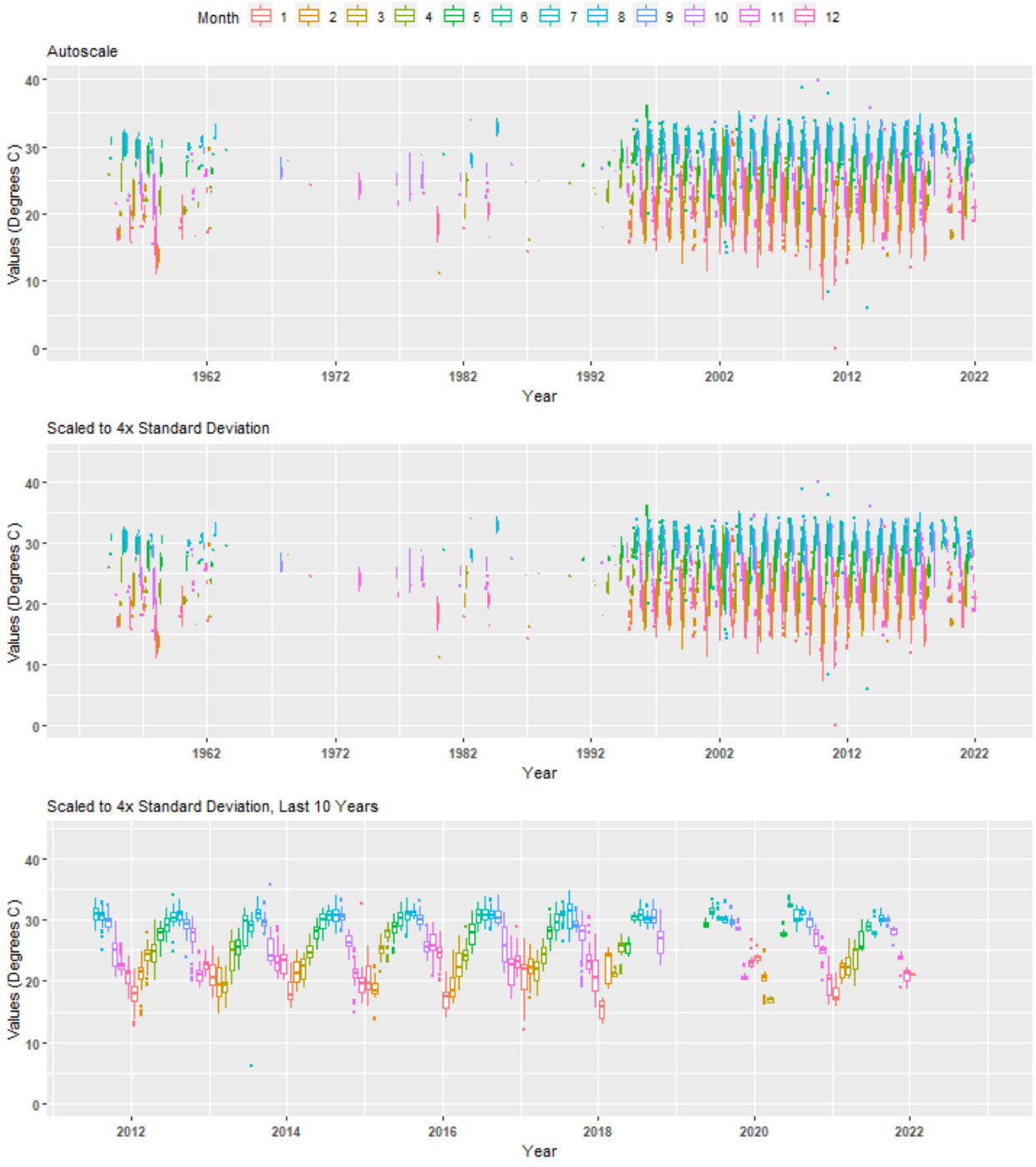
Summary Box Plots for Pine Island Sound Aquatic Preserve

By Year



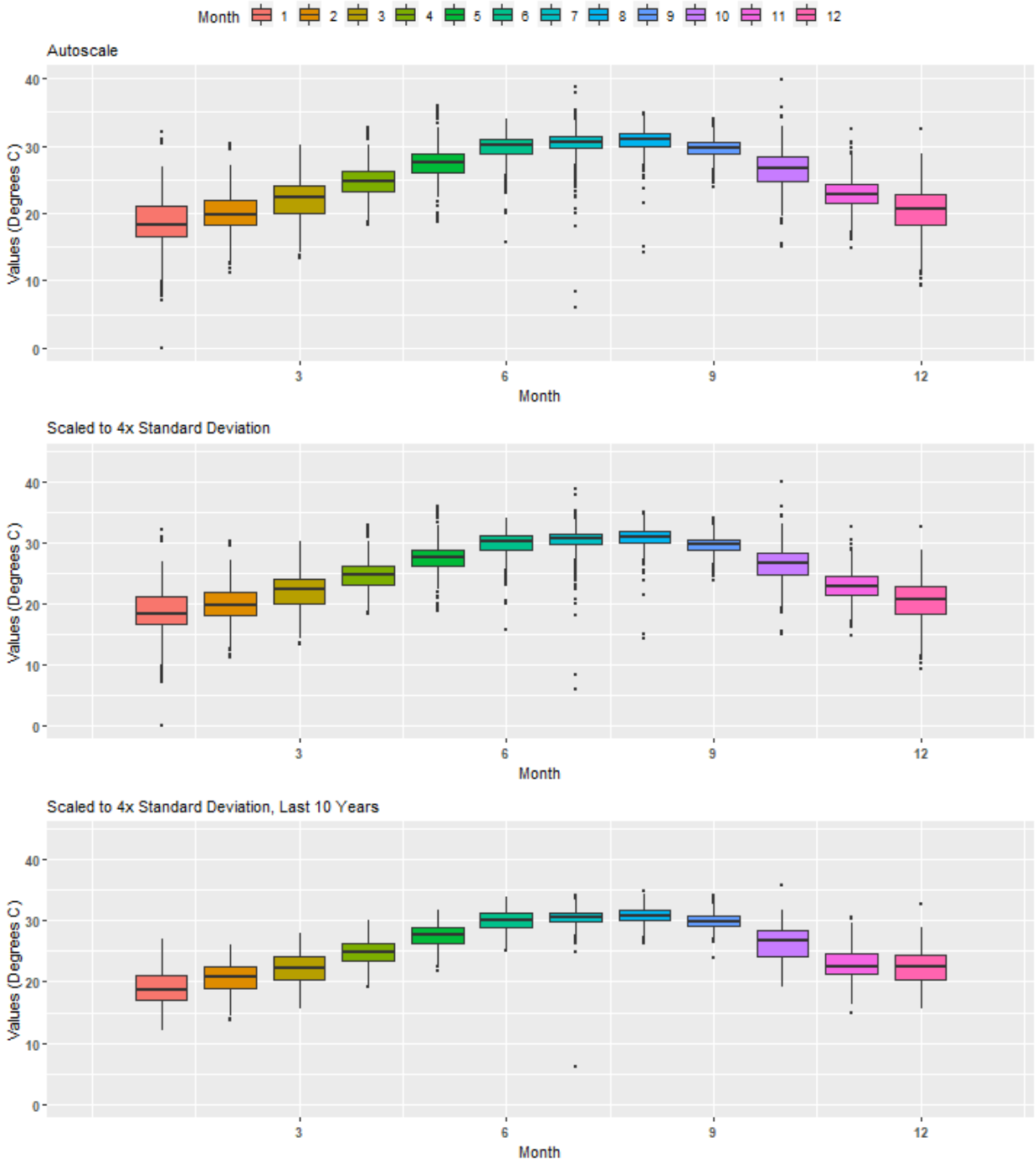
Summary Box Plots for Pine Island Sound Aquatic Preserve

By Year & Month



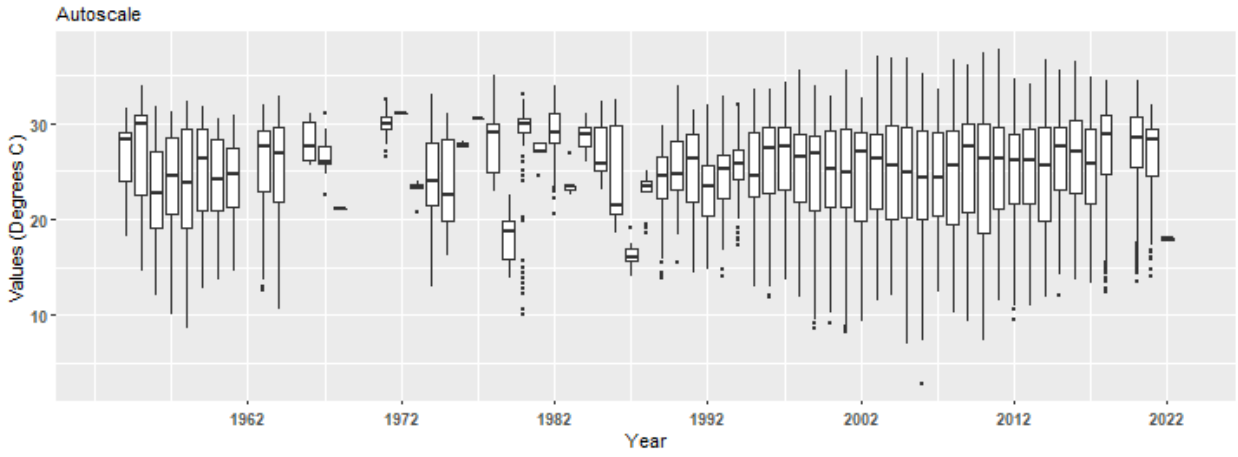
Summary Box Plots for Pine Island Sound Aquatic Preserve

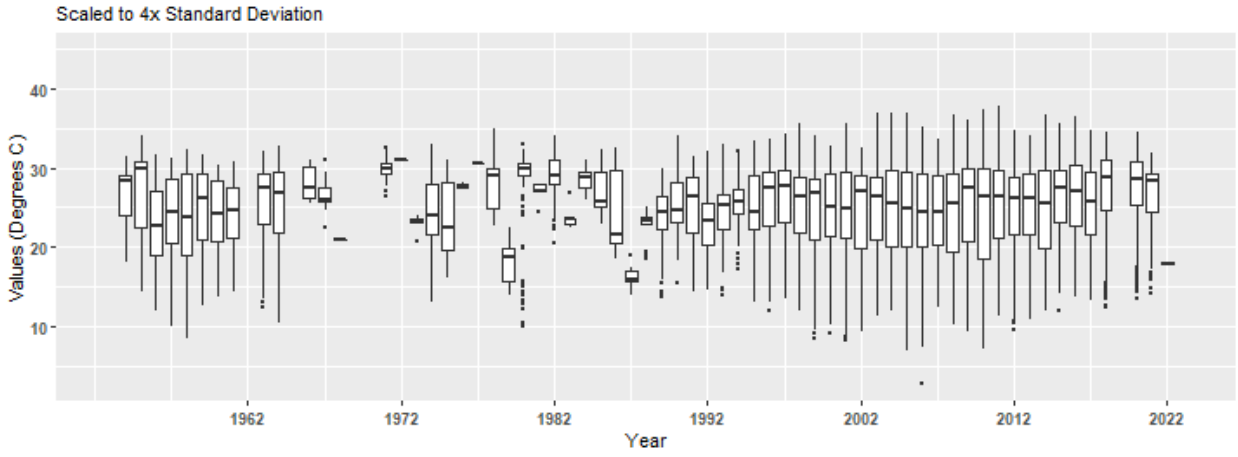
By Month

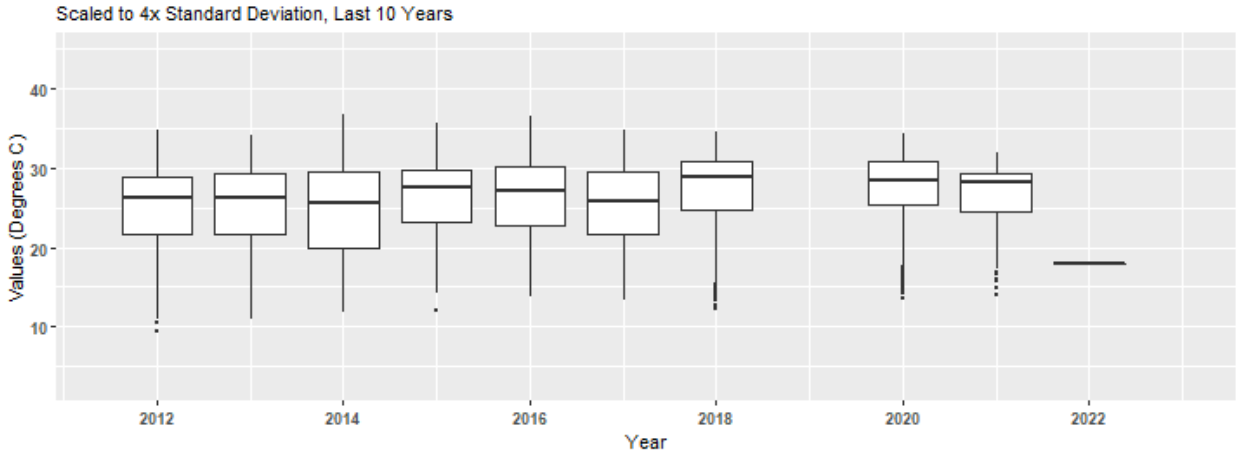


Summary Box Plots for Pinellas County Aquatic Preserve

By Year

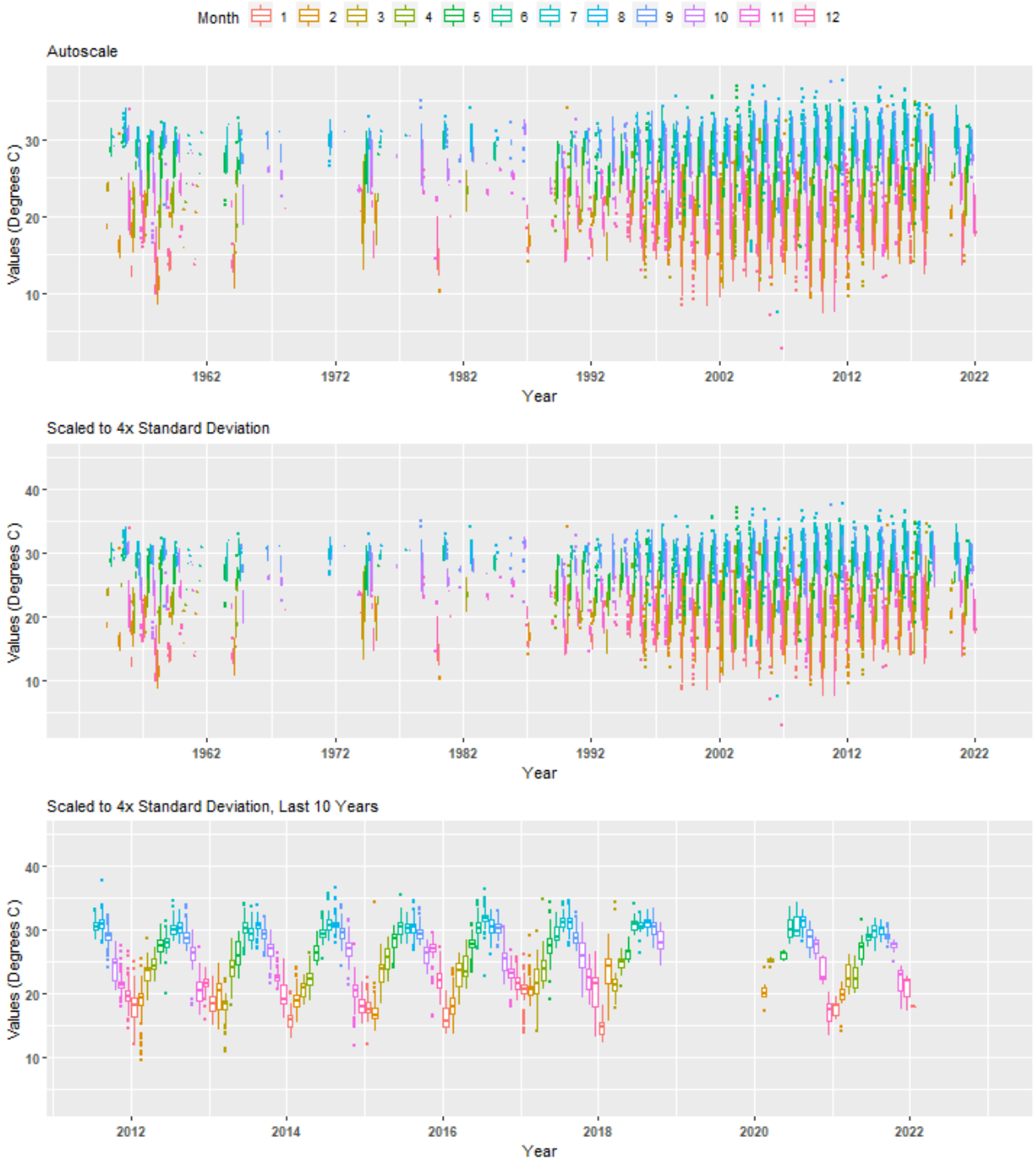






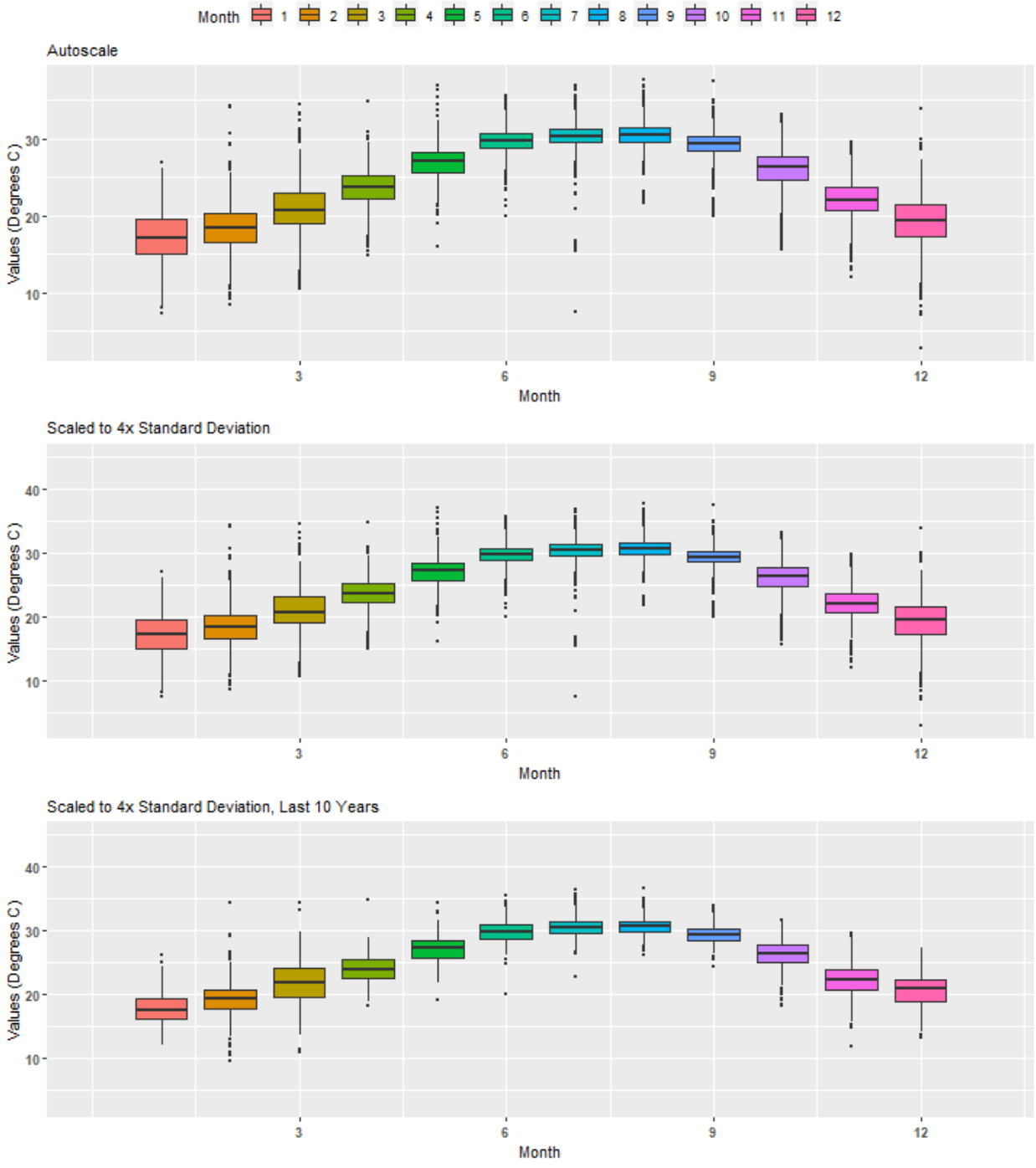
Summary Box Plots for Pinellas County Aquatic Preserve

By Year & Month



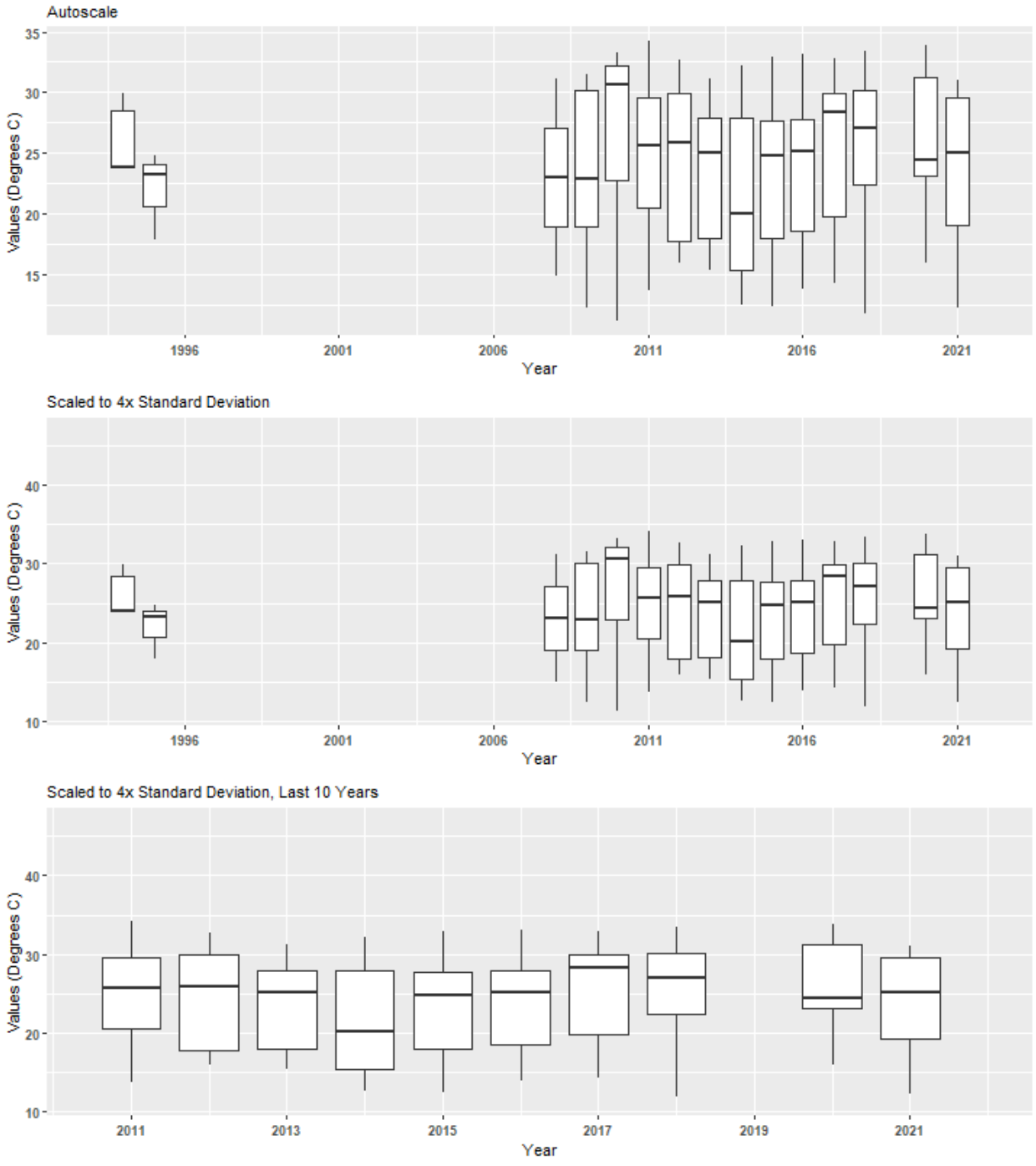
Summary Box Plots for Pinellas County Aquatic Preserve

By Month



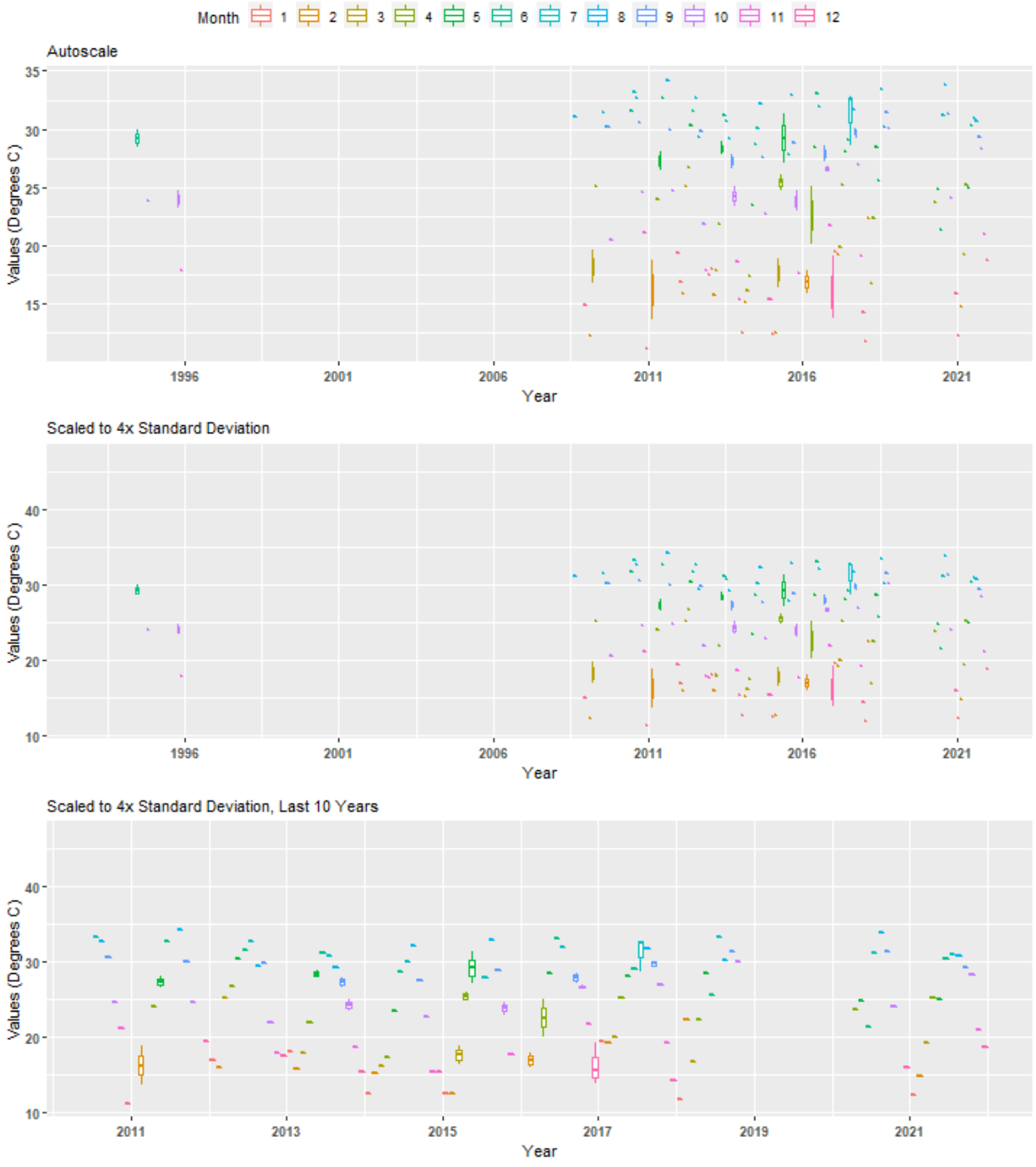
Summary Box Plots for Rocky Bayou State Park Aquatic Preserve

By Year



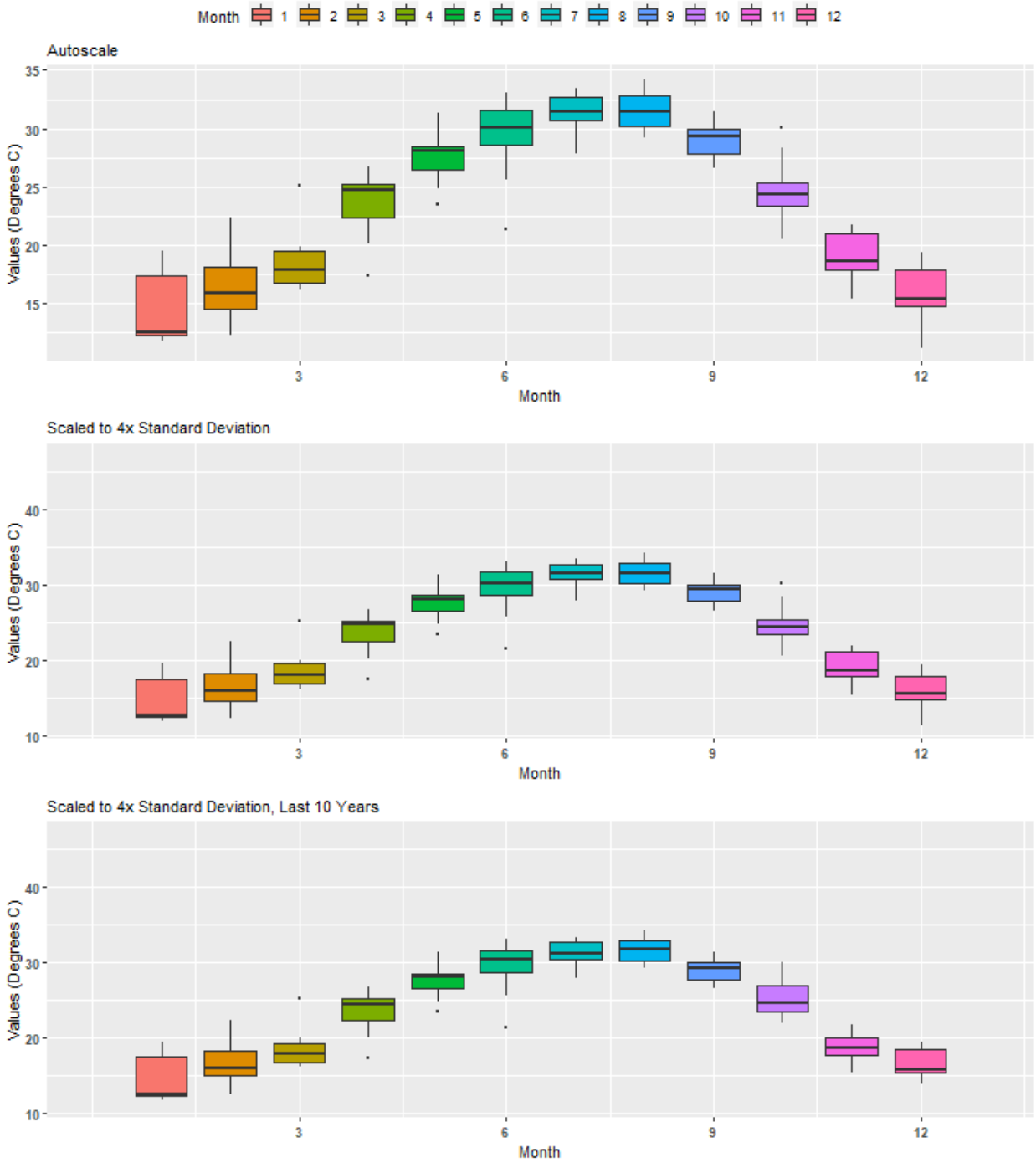
Summary Box Plots for Rocky Bayou State Park Aquatic Preserve

By Year & Month



Summary Box Plots for Rocky Bayou State Park Aquatic Preserve

By Month

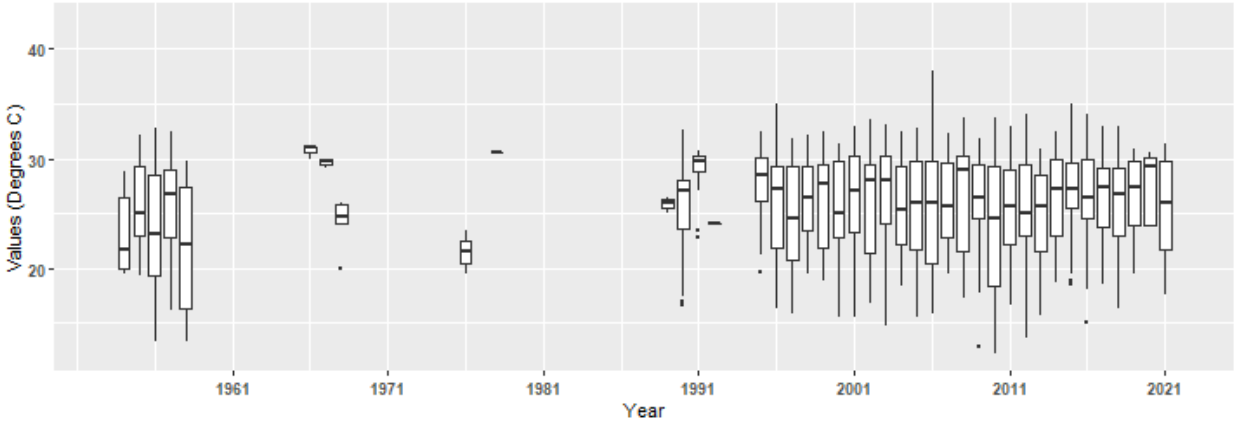


Summary Box Plots for Rookery Bay Aquatic Preserve

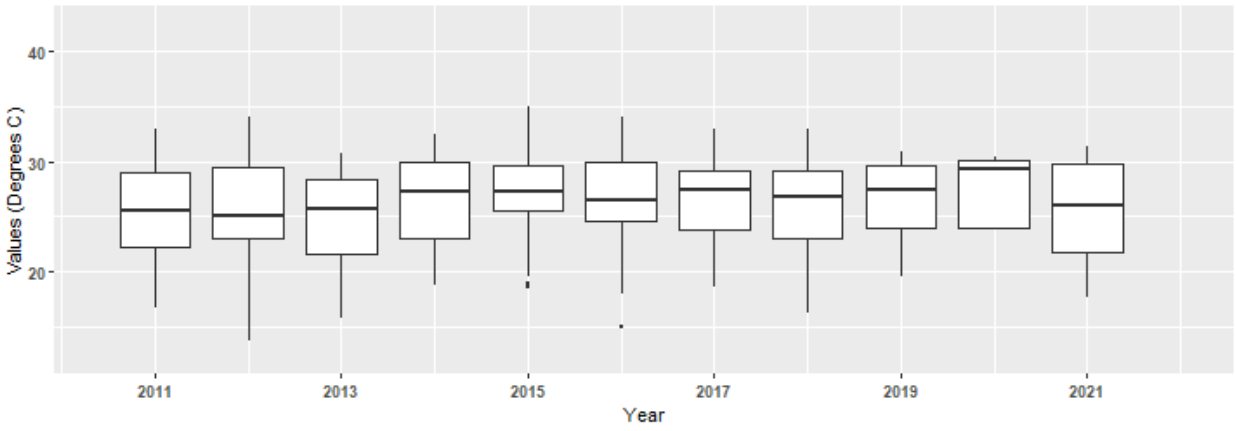
By Year

Autoscale

Scaled to 4x Standard Deviation

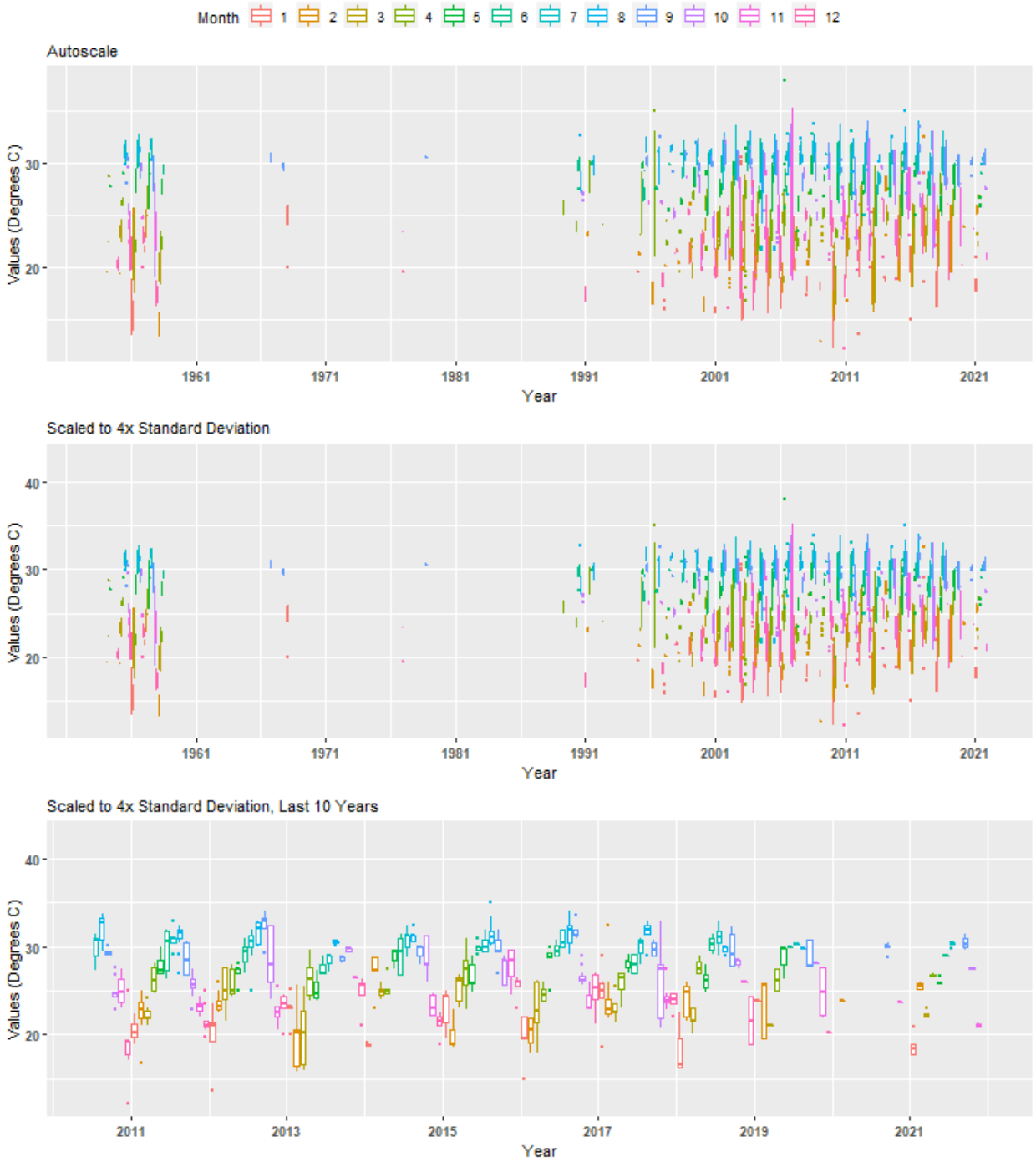


Scaled to 4x Standard Deviation, Last 10 Years



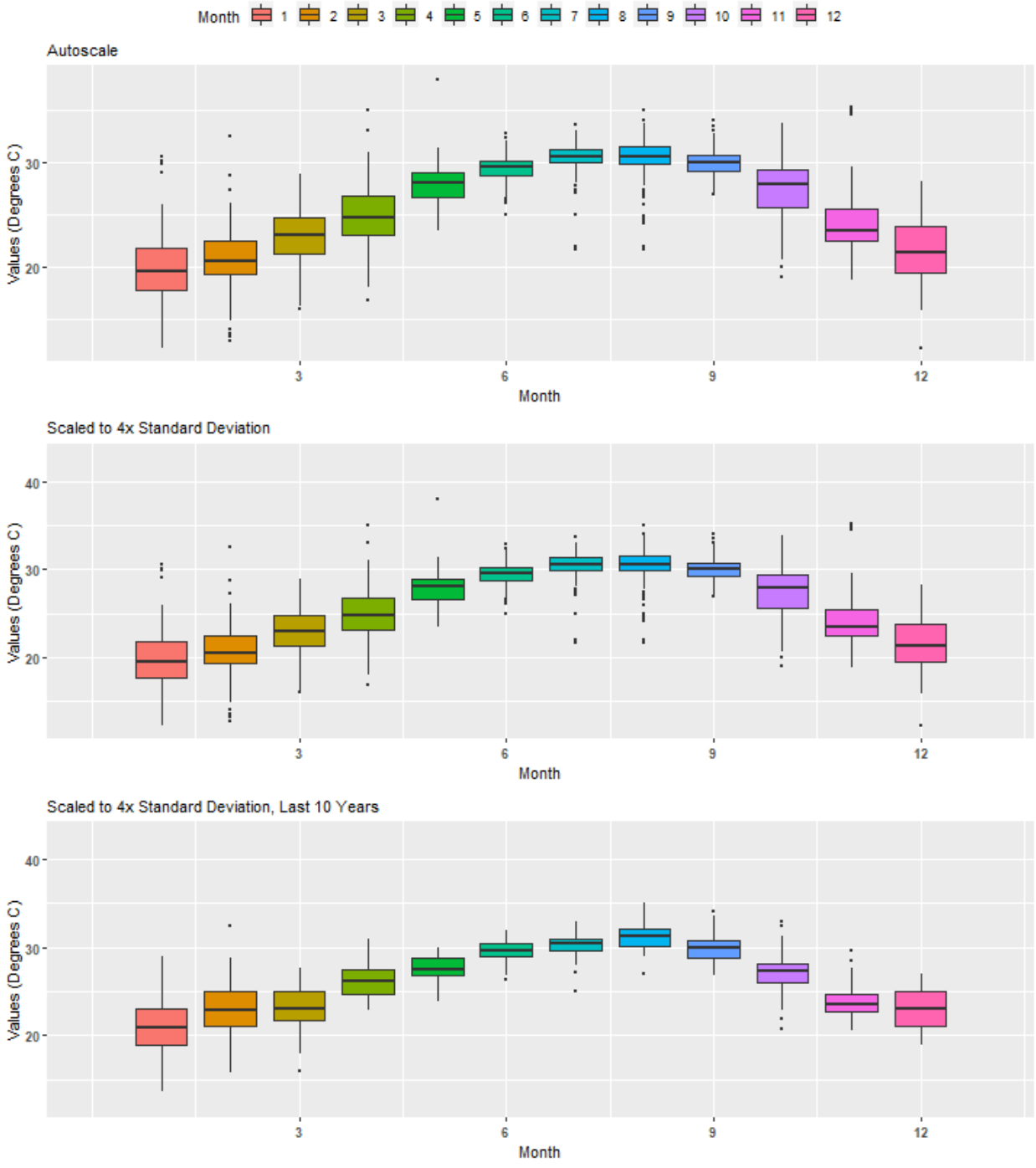
Summary Box Plots for Rookery Bay Aquatic Preserve

By Year & Month



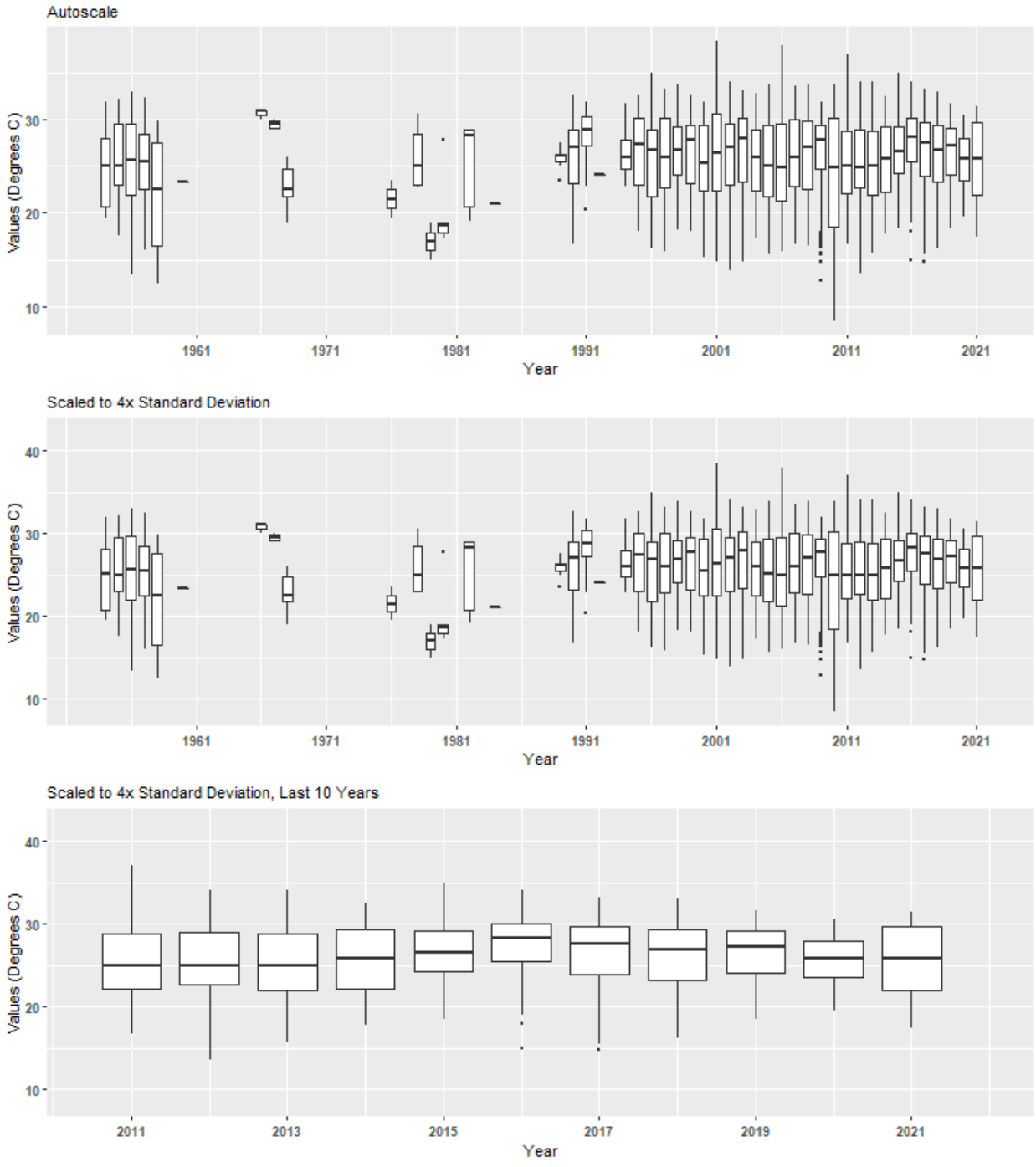
Summary Box Plots for Rookery Bay Aquatic Preserve

By Month



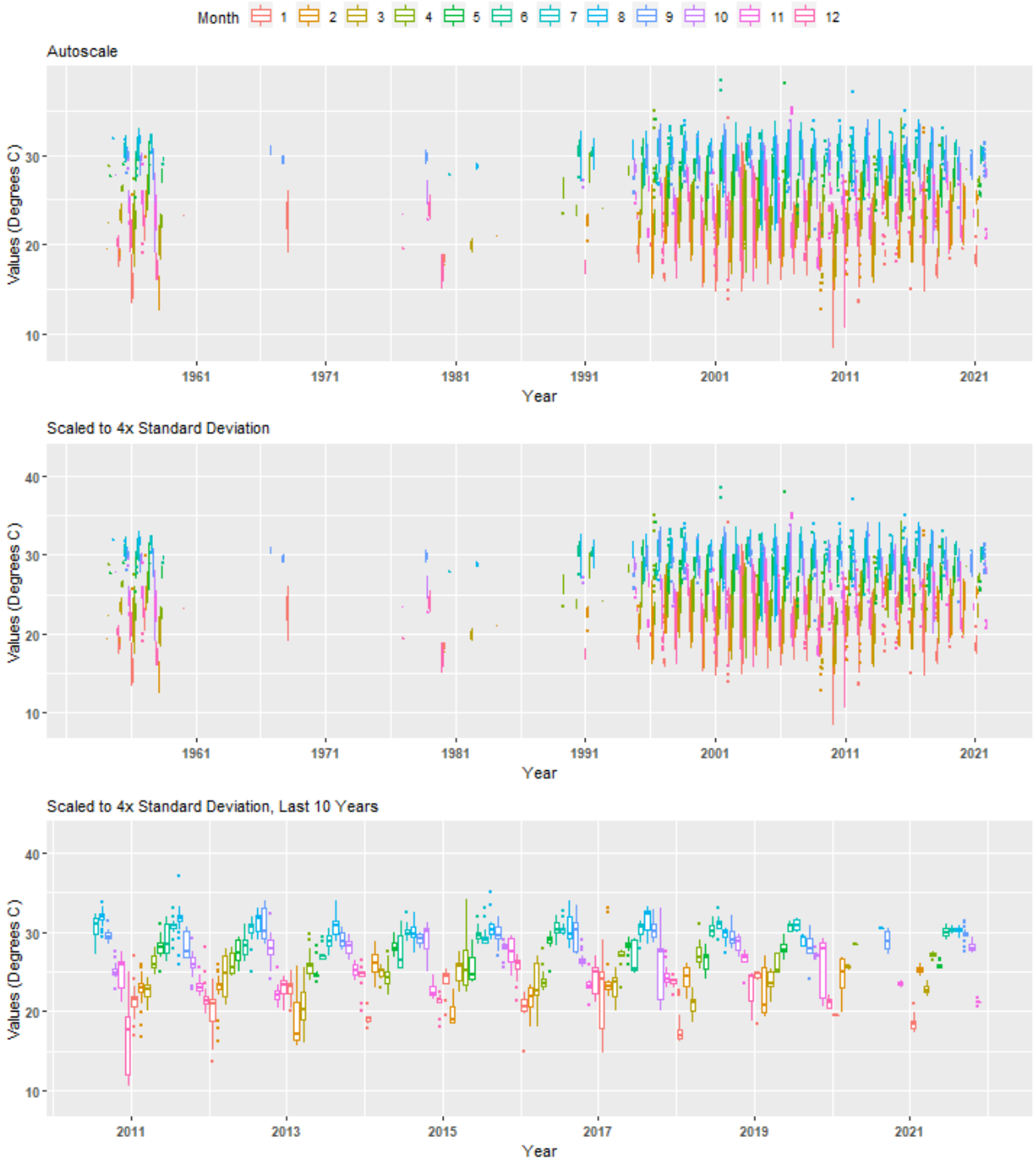
Summary Box Plots for Rookery Bay National Estuarine Research Reserve

By Year



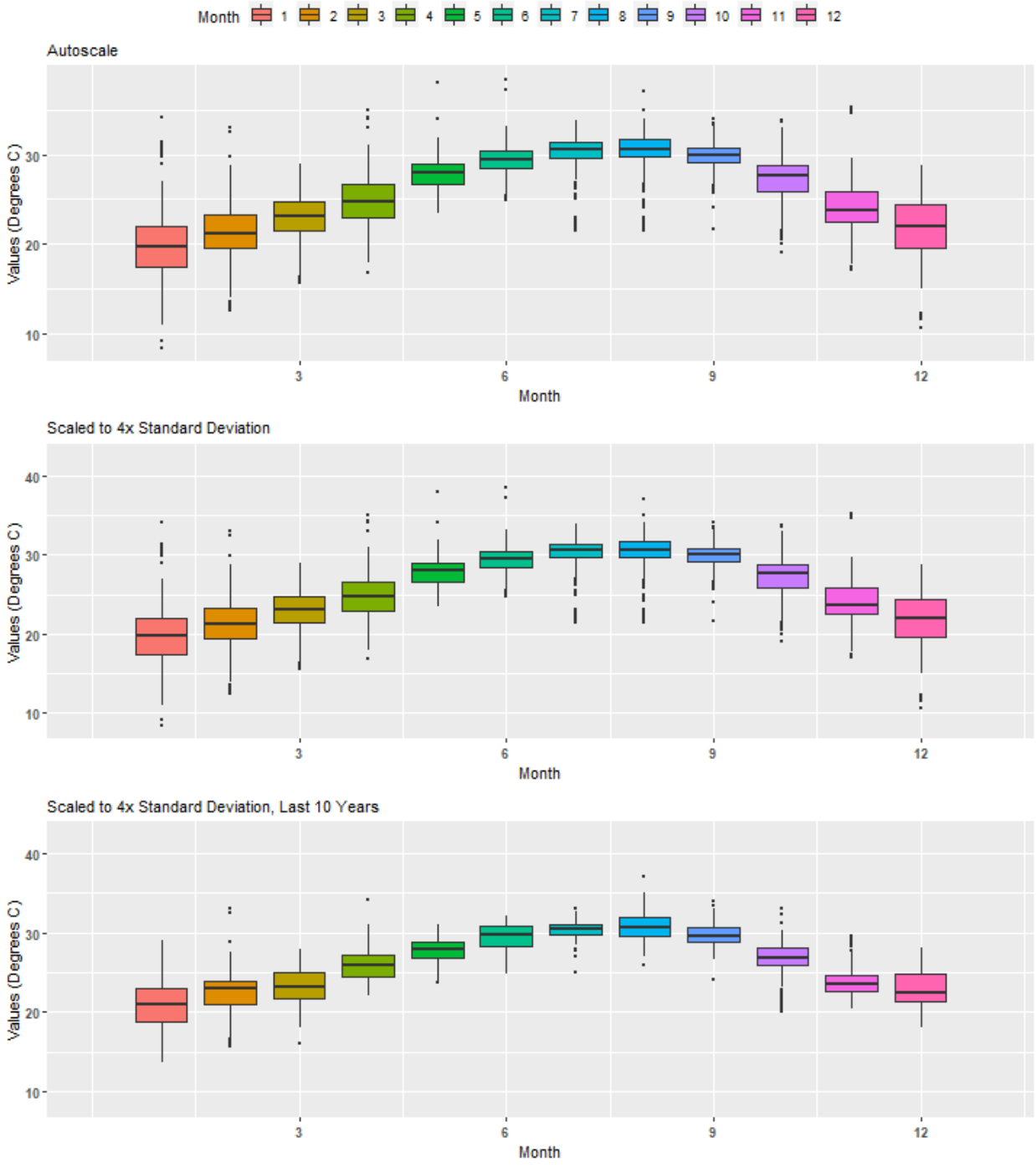
Summary Box Plots for Rookery Bay National Estuarine Research Reserve

By Year & Month



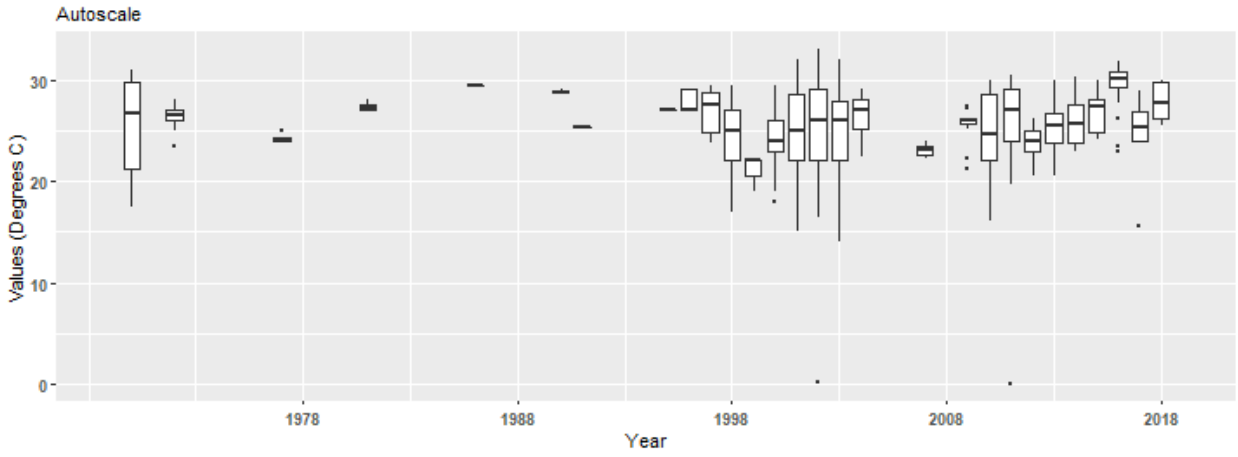
Summary Box Plots for Rookery Bay National Estuarine Research Reserve

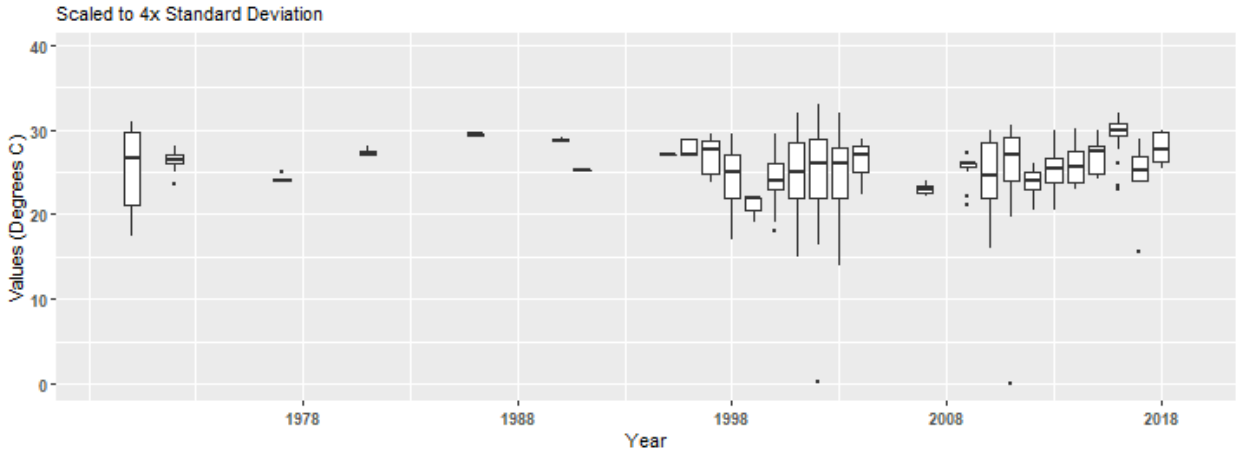
By Month

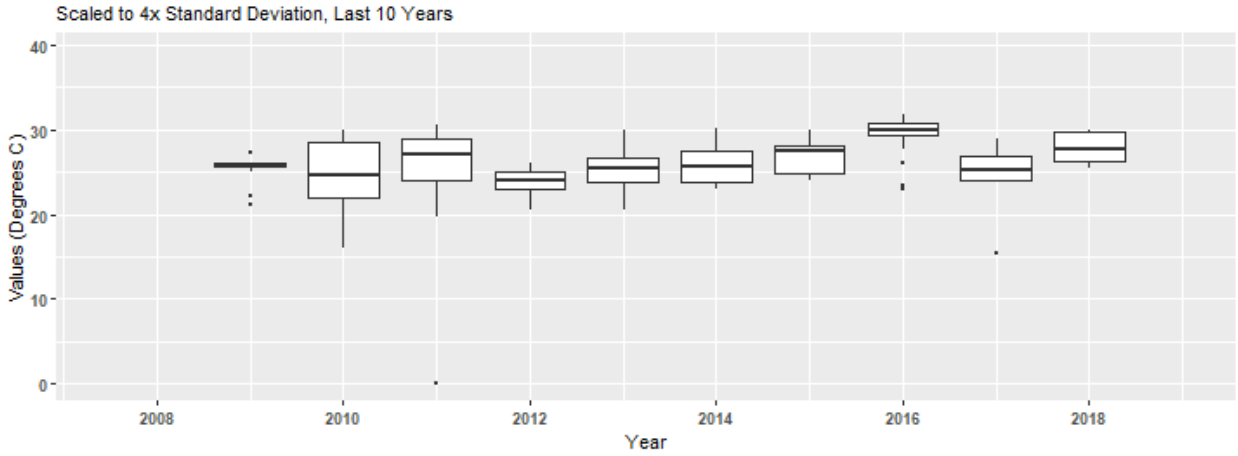


Summary Box Plots for Southeast Florida Coral Reef Ecosystem Conservation Area

By Year

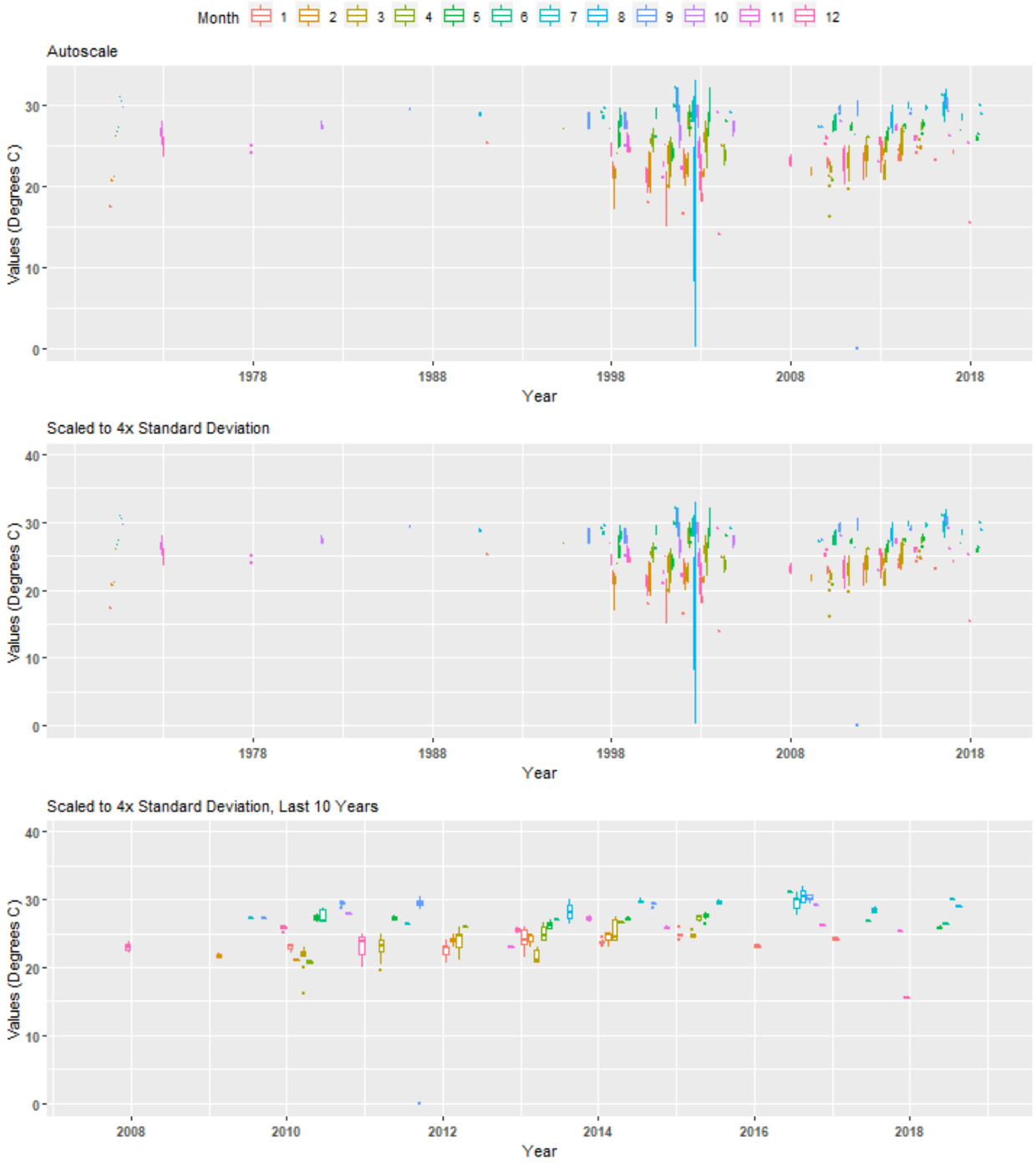






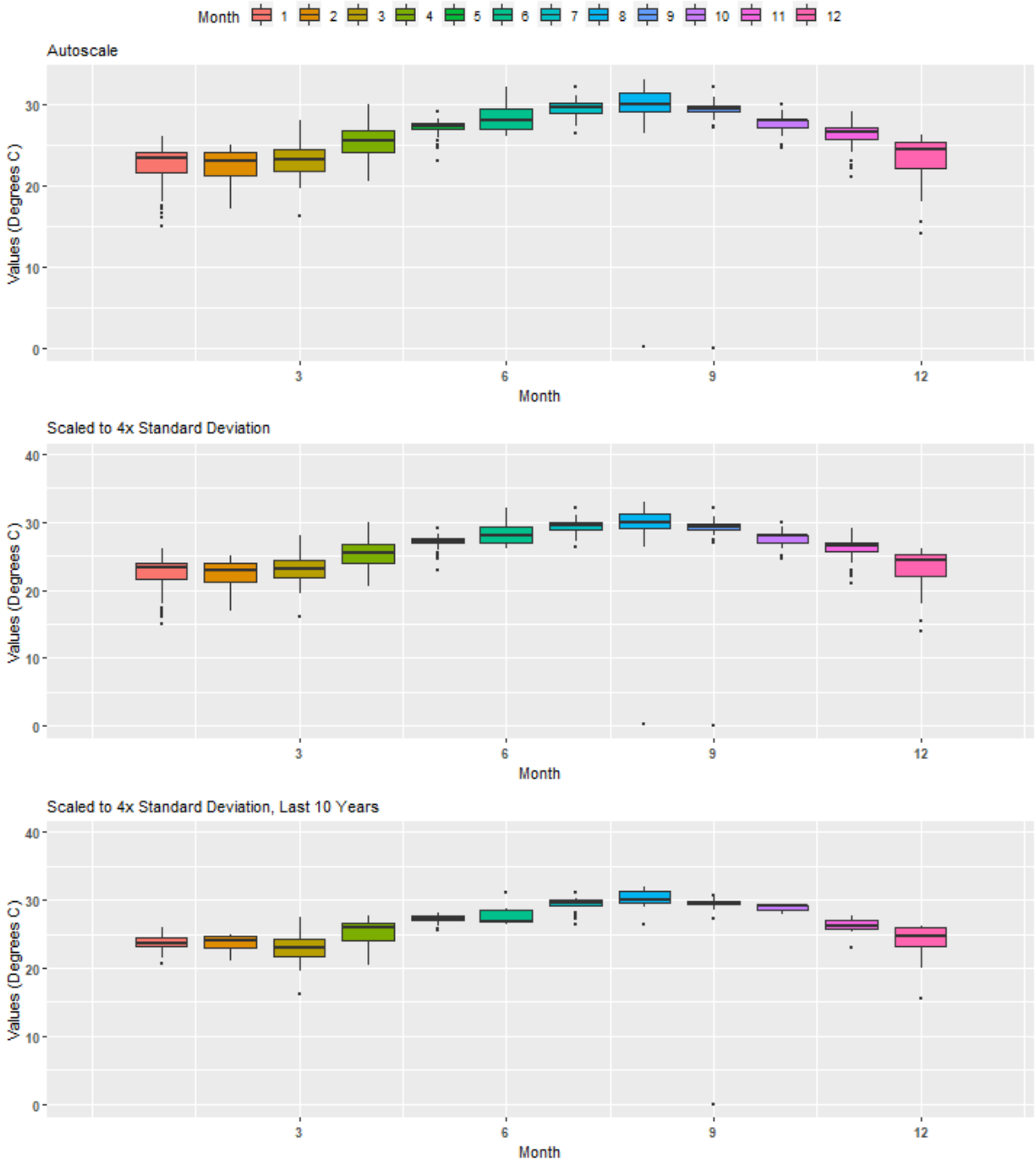
Summary Box Plots for Southeast Florida Coral Reef Ecosystem Conservation Area

By Year & Month



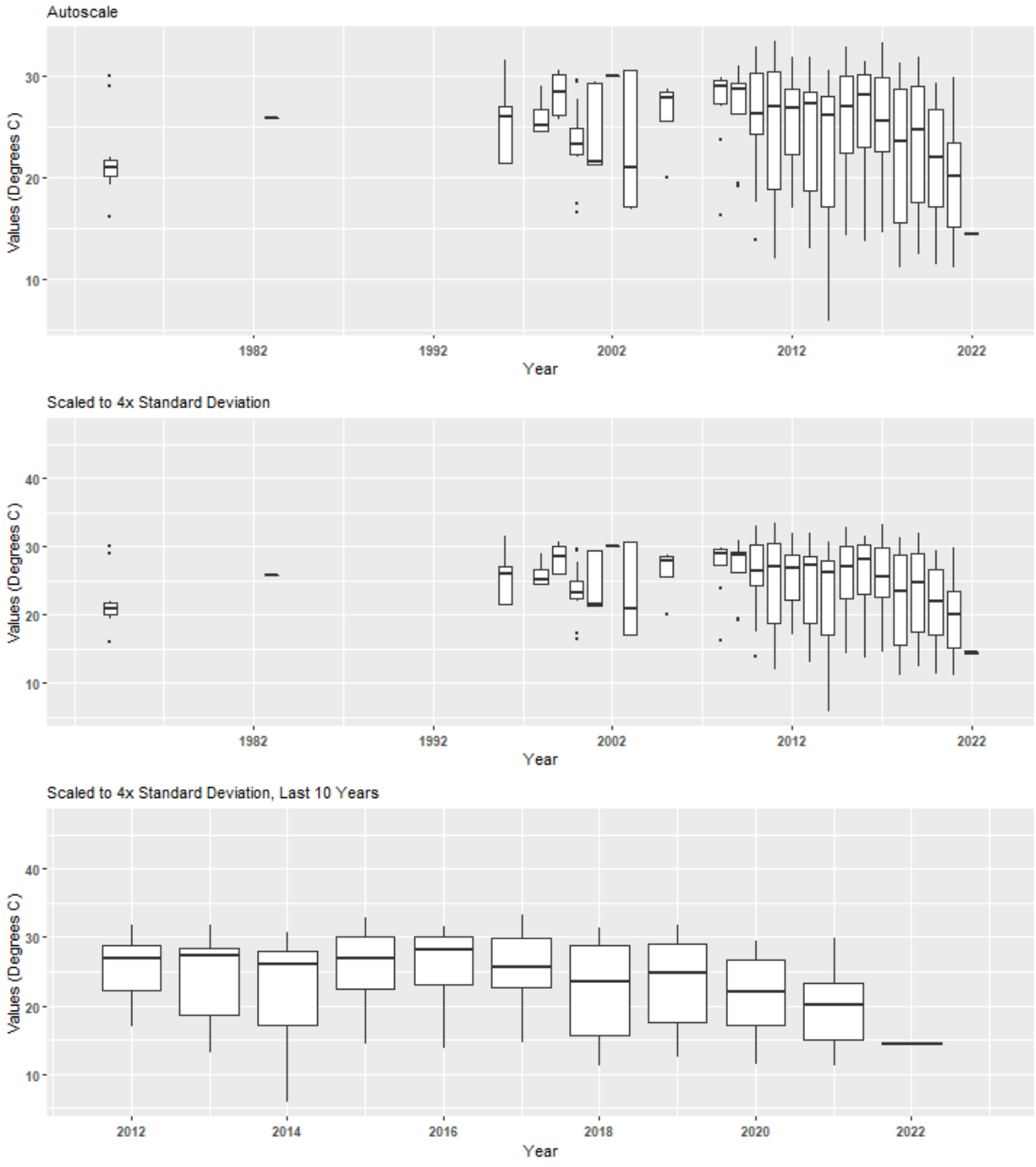
Summary Box Plots for Southeast Florida Coral Reef Ecosystem Conservation Area

By Month



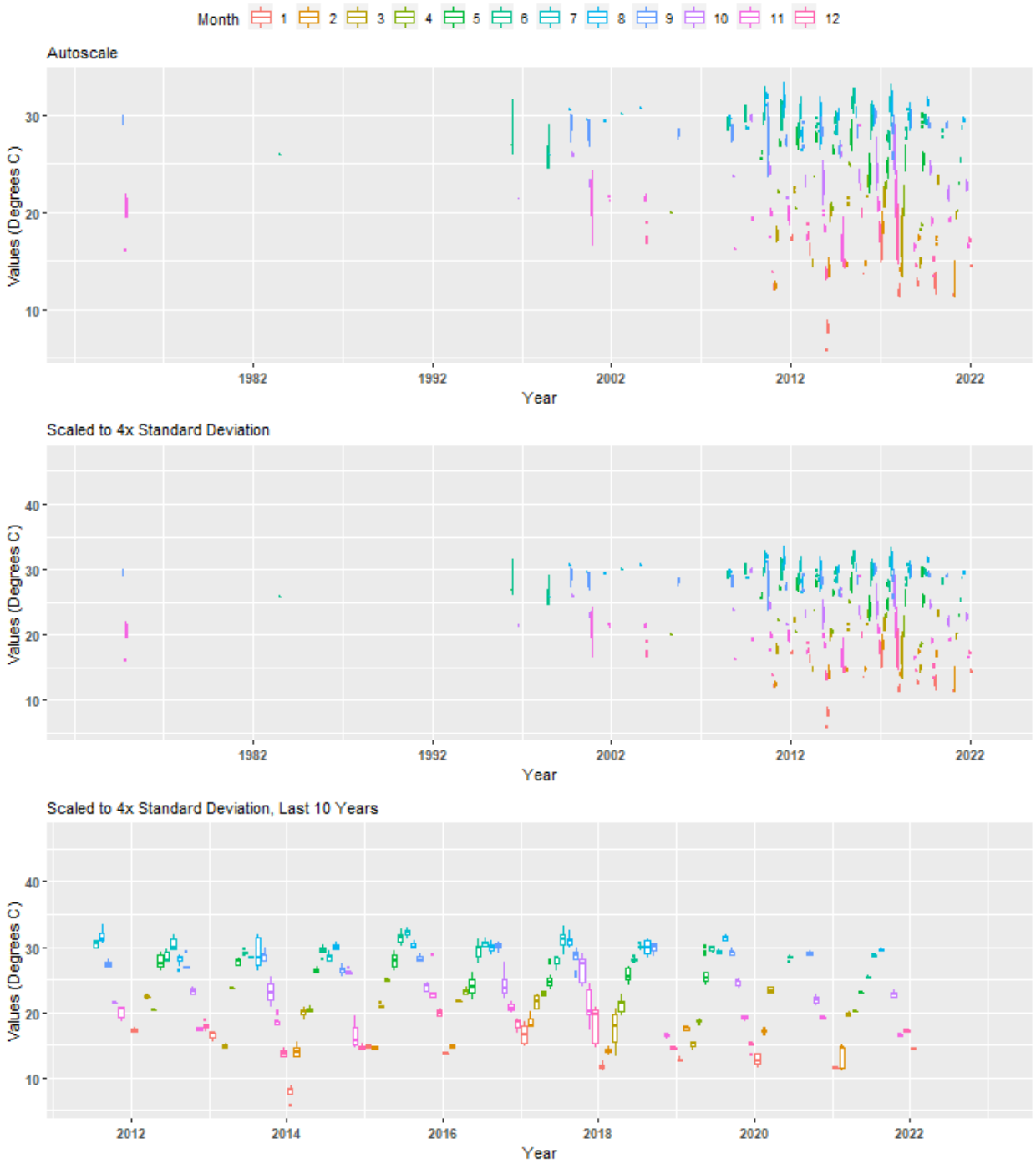
Summary Box Plots for St. Andrews State Park Aquatic Preserve

By Year



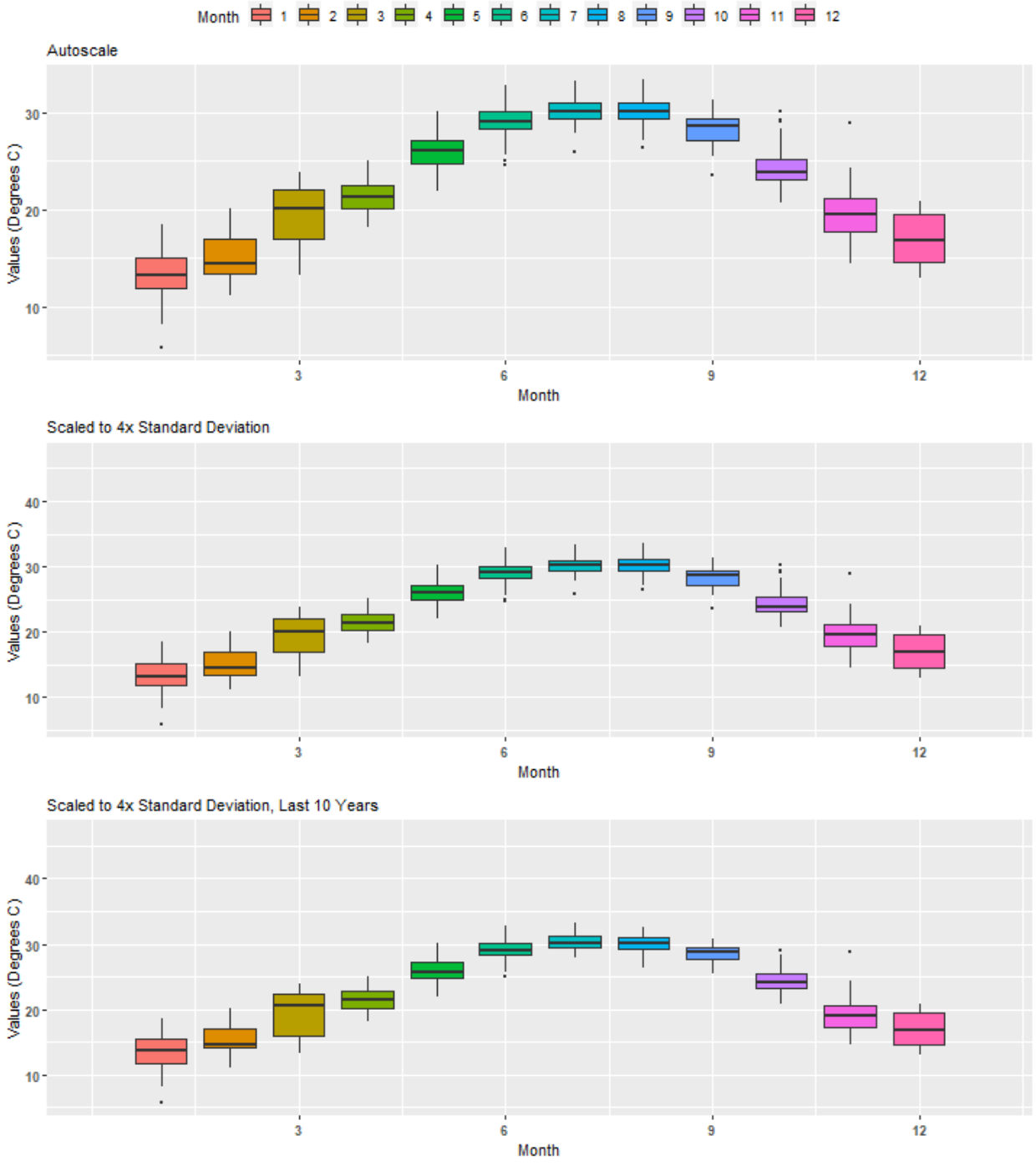
Summary Box Plots for St. Andrews State Park Aquatic Preserve

By Year & Month



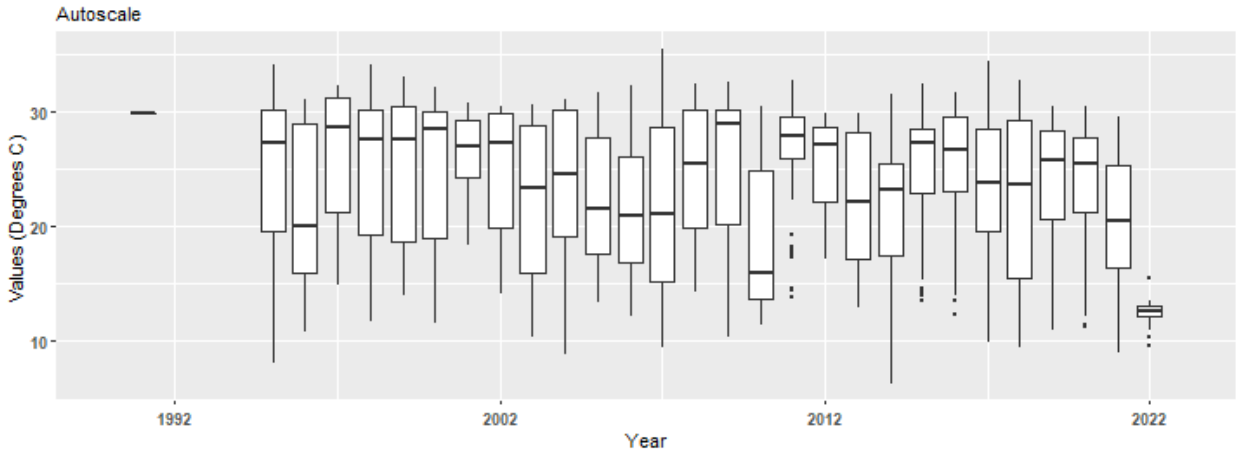
Summary Box Plots for St. Andrews State Park Aquatic Preserve

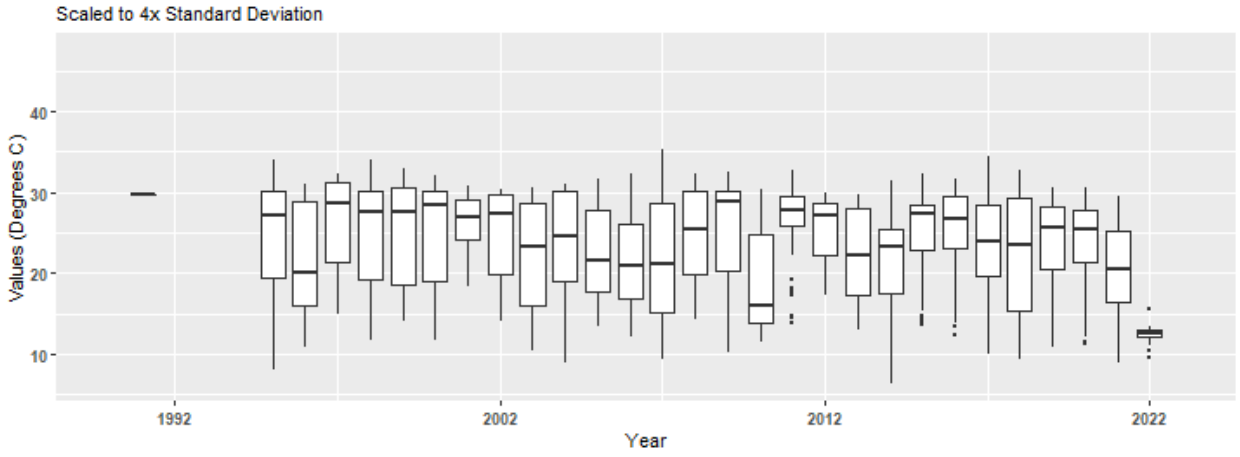
By Month

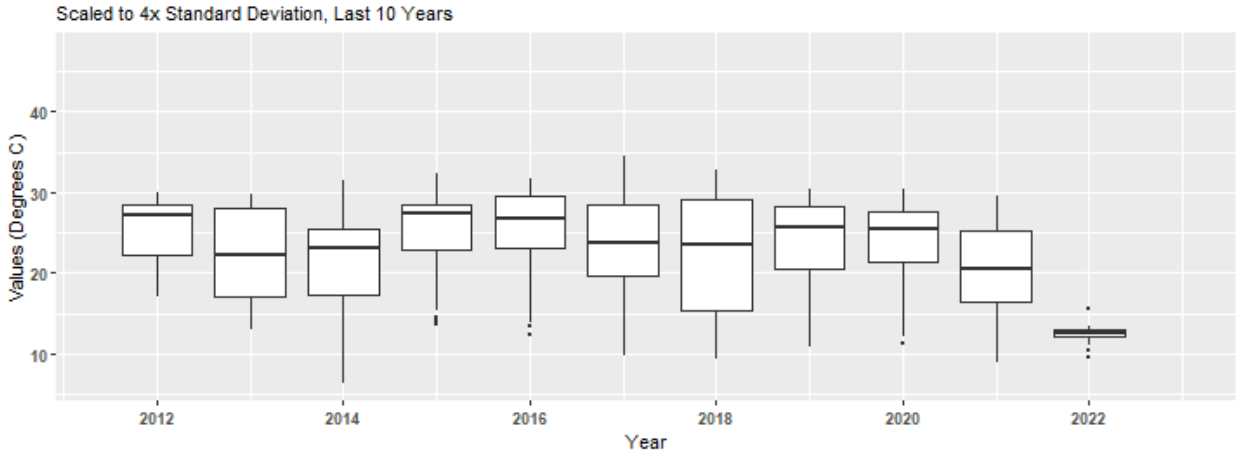


Summary Box Plots for St. Joseph Bay Aquatic Preserve

By Year

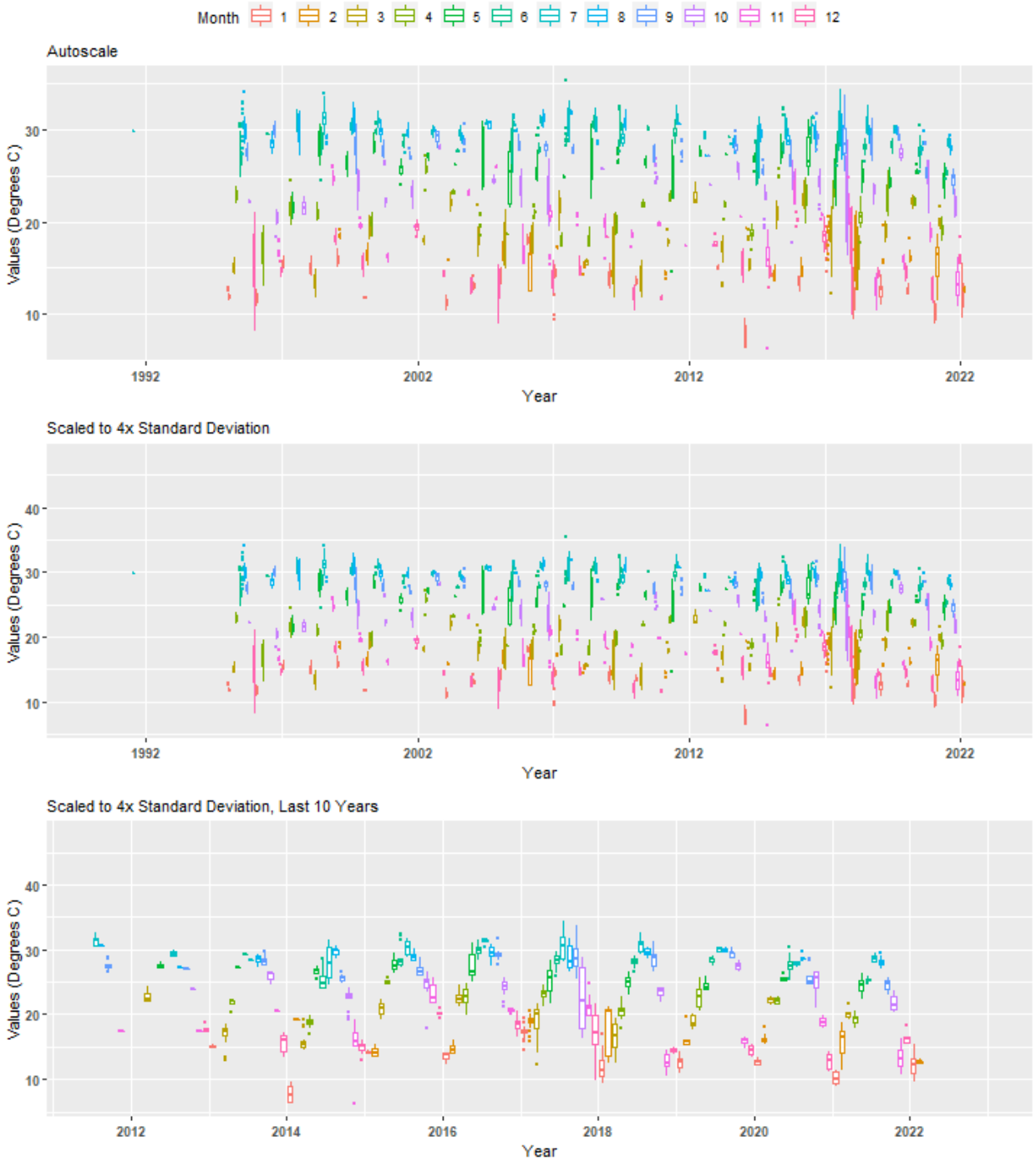






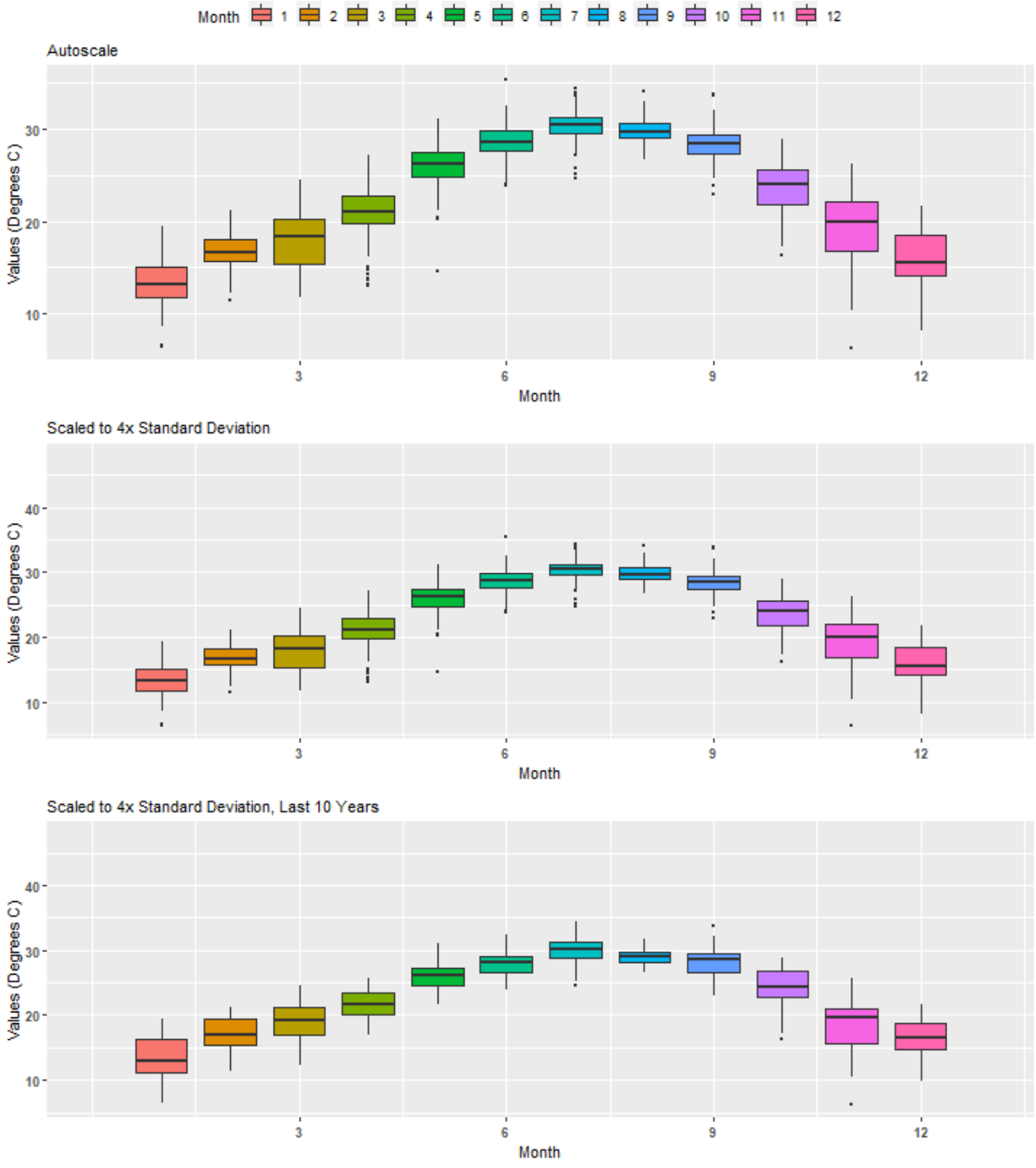
Summary Box Plots for St. Joseph Bay Aquatic Preserve

By Year & Month



Summary Box Plots for St. Joseph Bay Aquatic Preserve

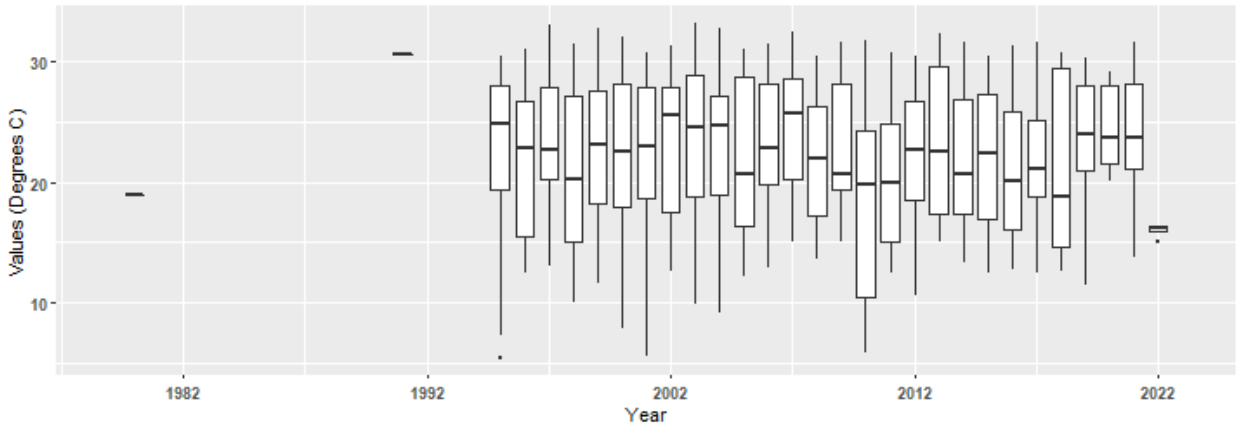
By Month



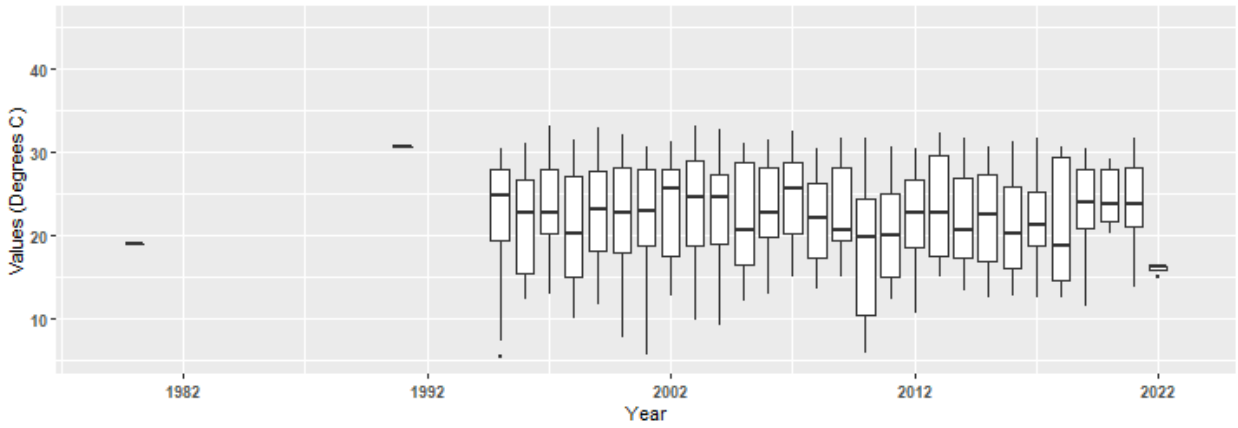
Summary Box Plots for St. Martins Marsh Aquatic Preserve

By Year

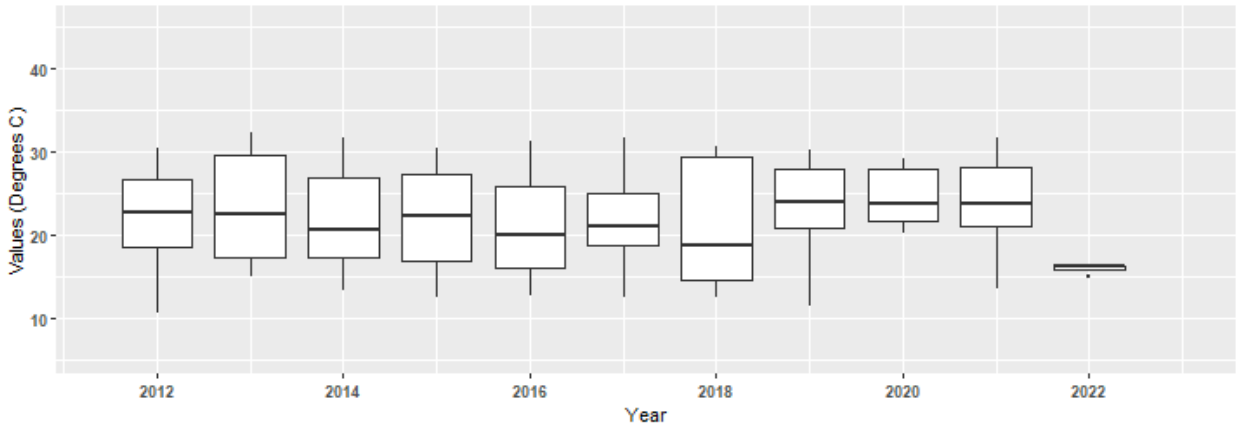
Autoscale



Scaled to 4x Standard Deviation

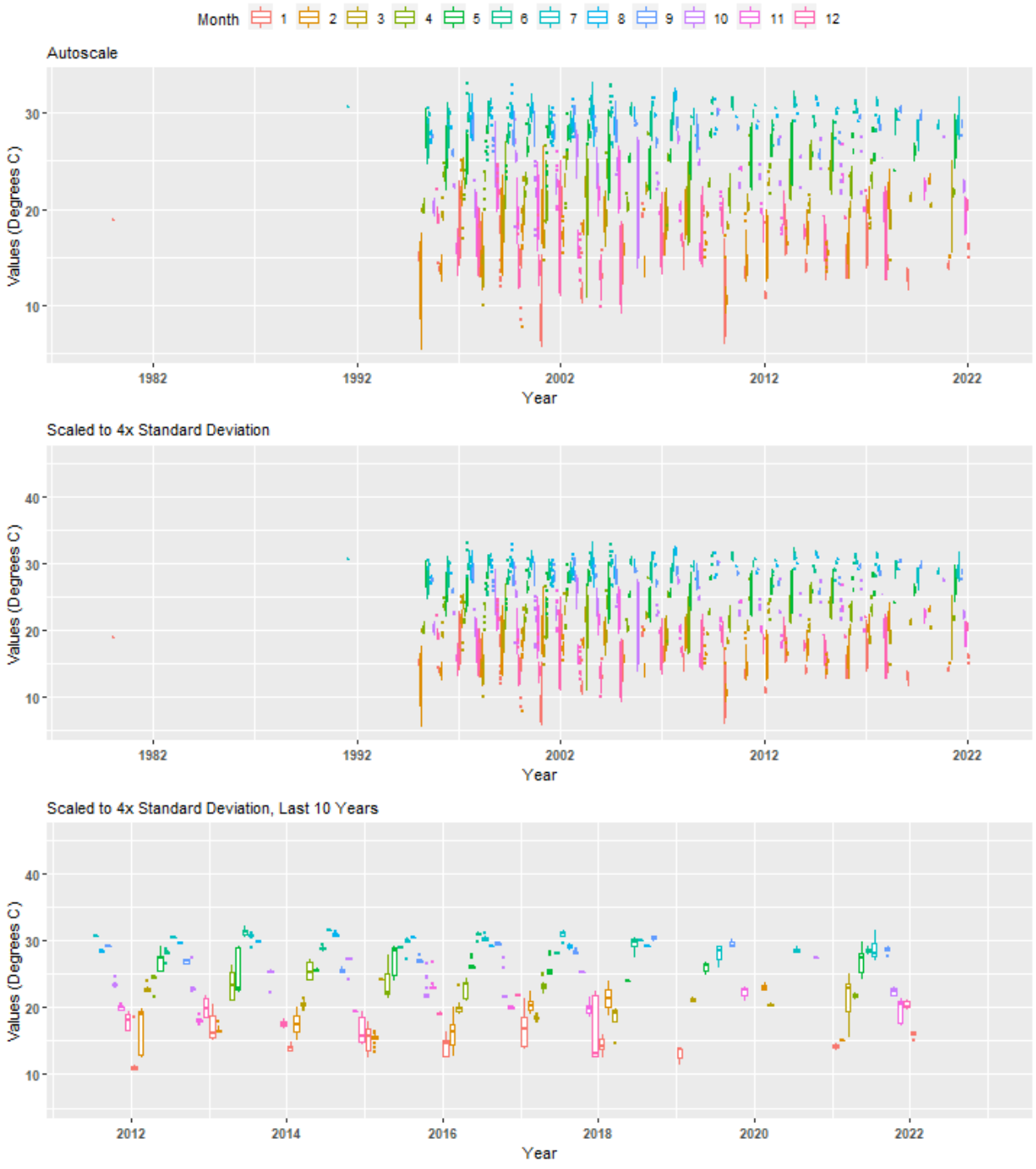


Scaled to 4x Standard Deviation, Last 10 Years



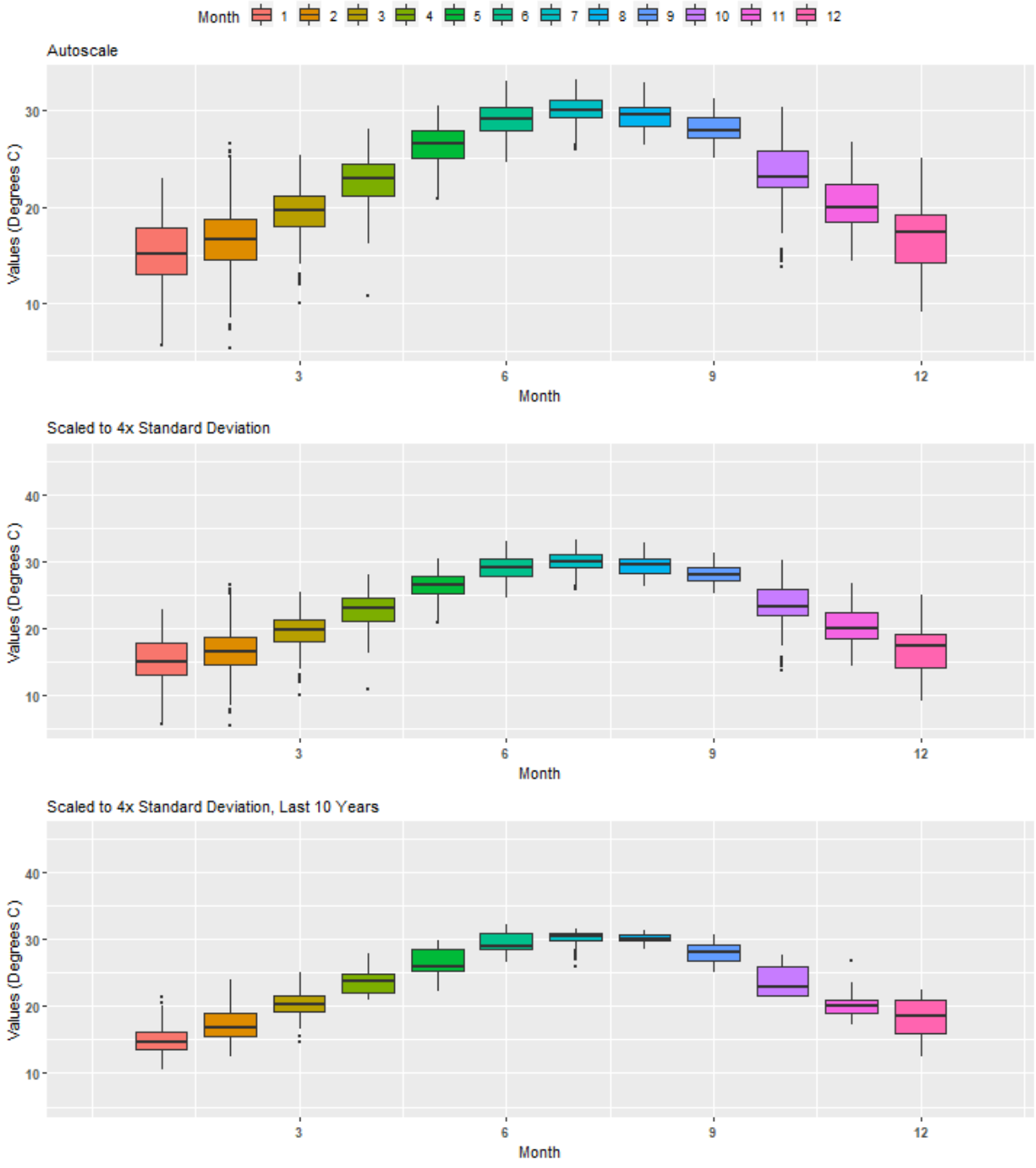
Summary Box Plots for St. Martins Marsh Aquatic Preserve

By Year & Month



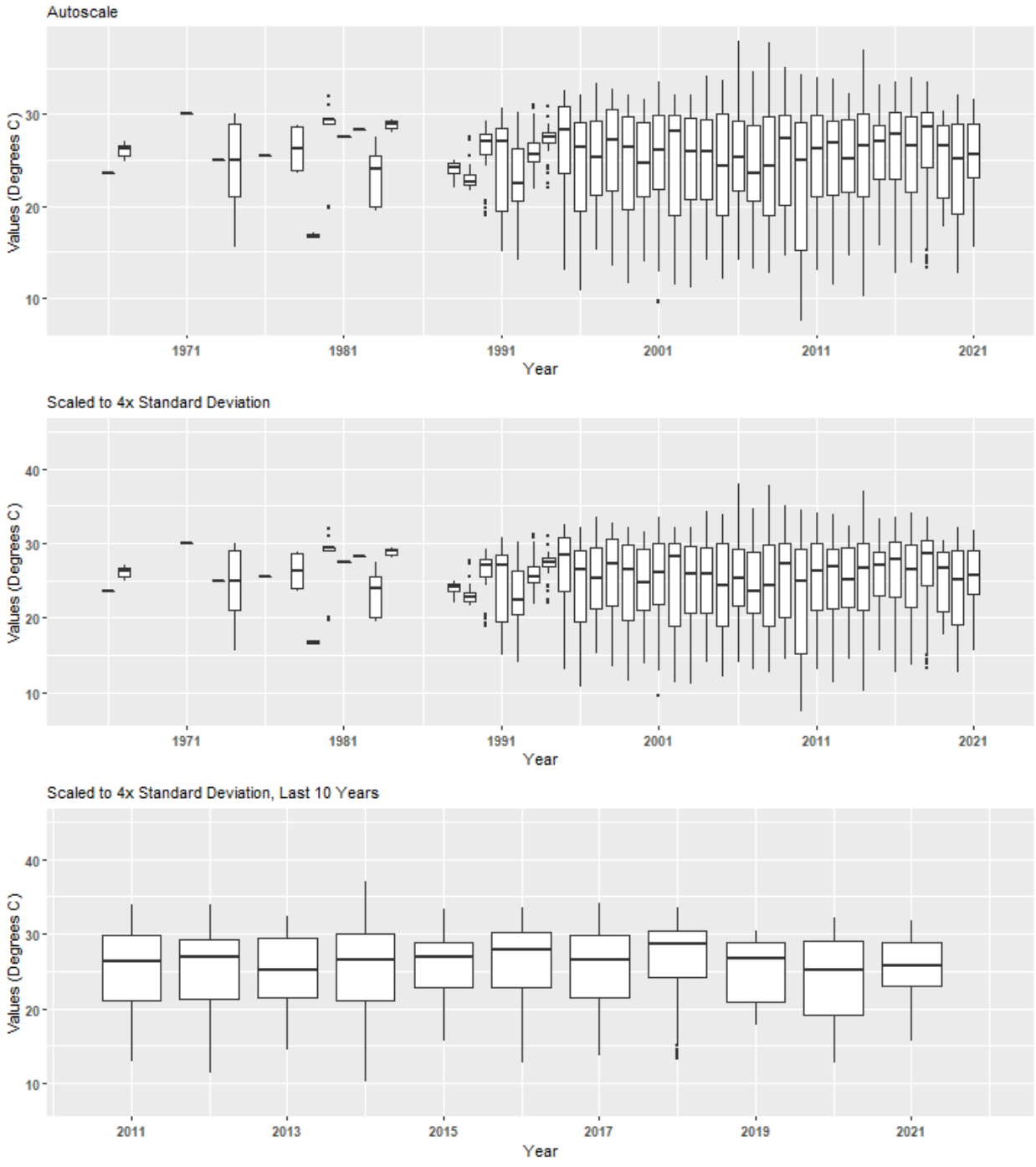
Summary Box Plots for St. Martins Marsh Aquatic Preserve

By Month



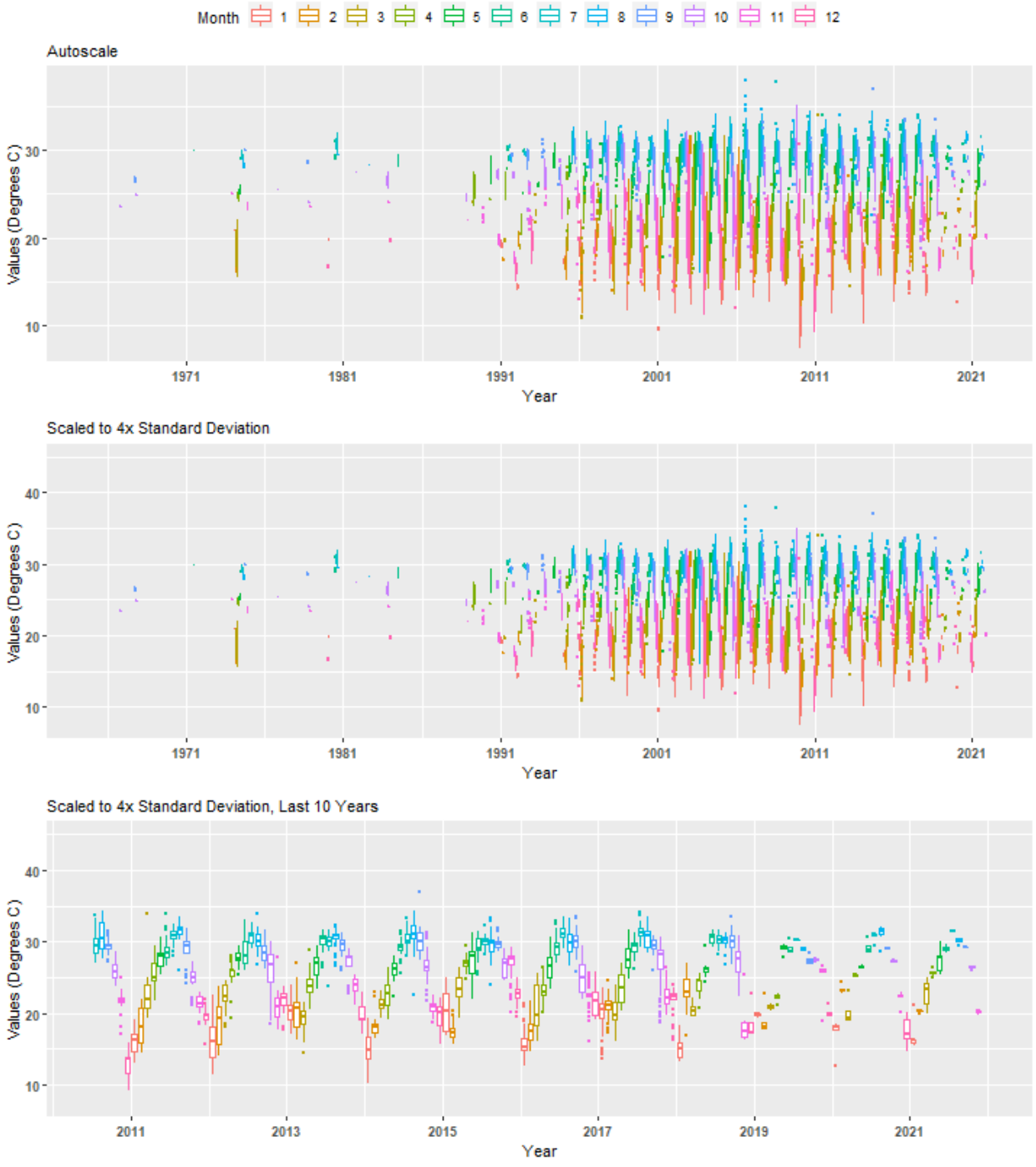
Summary Box Plots for Terra Ceia Aquatic Preserve

By Year



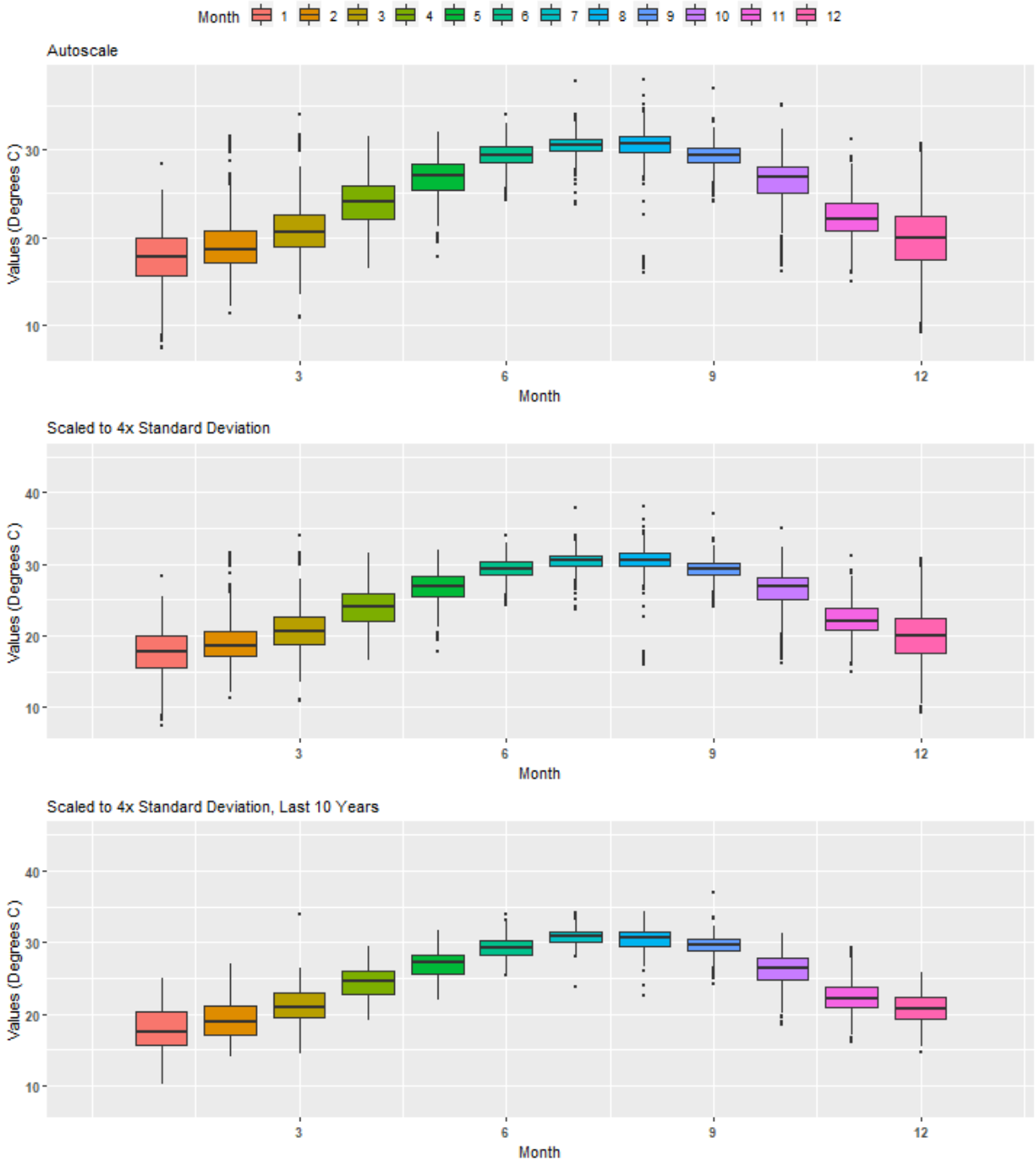
Summary Box Plots for Terra Ceia Aquatic Preserve

By Year & Month



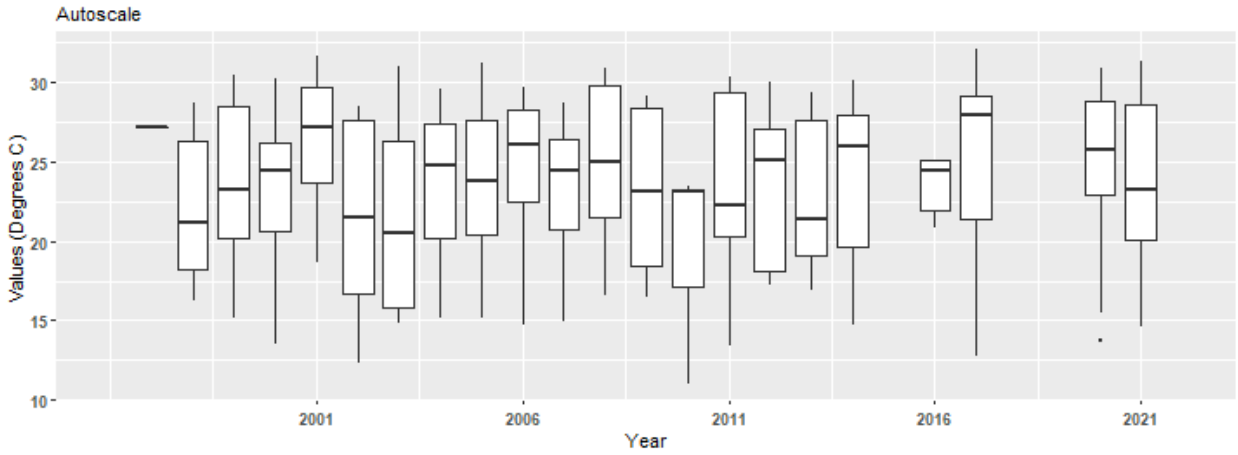
Summary Box Plots for Terra Ceia Aquatic Preserve

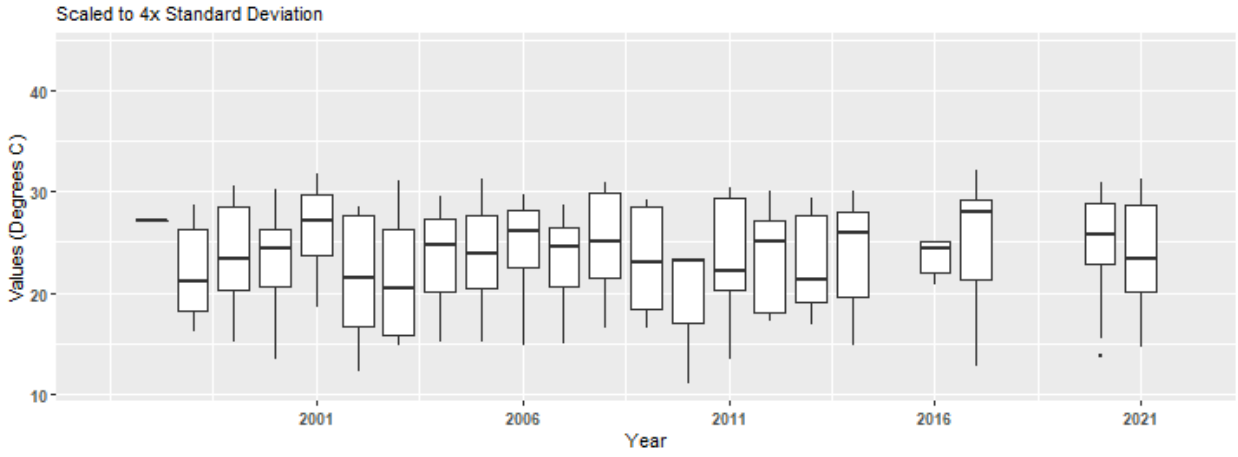
By Month

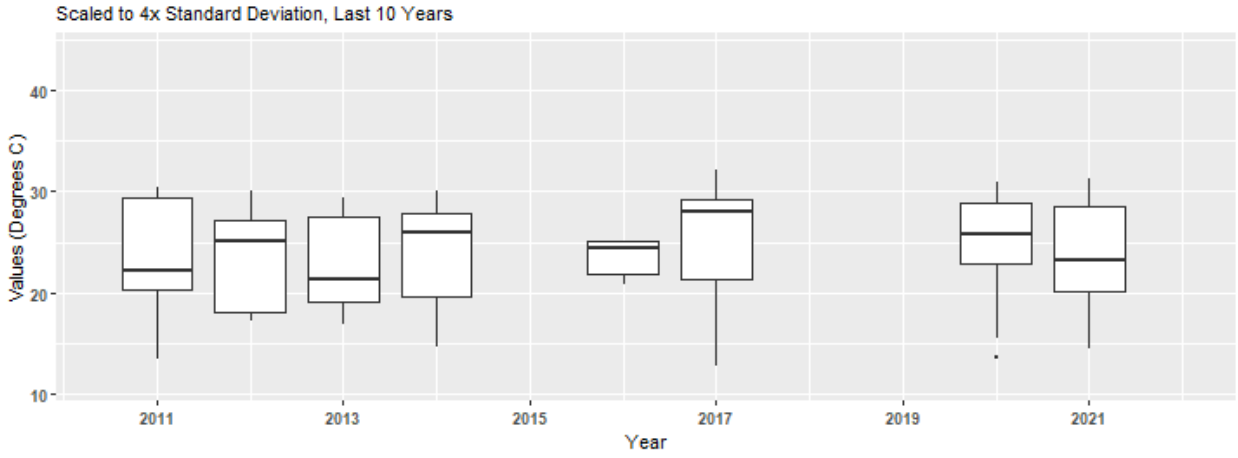


Summary Box Plots for Tomoka Marsh Aquatic Preserve

By Year





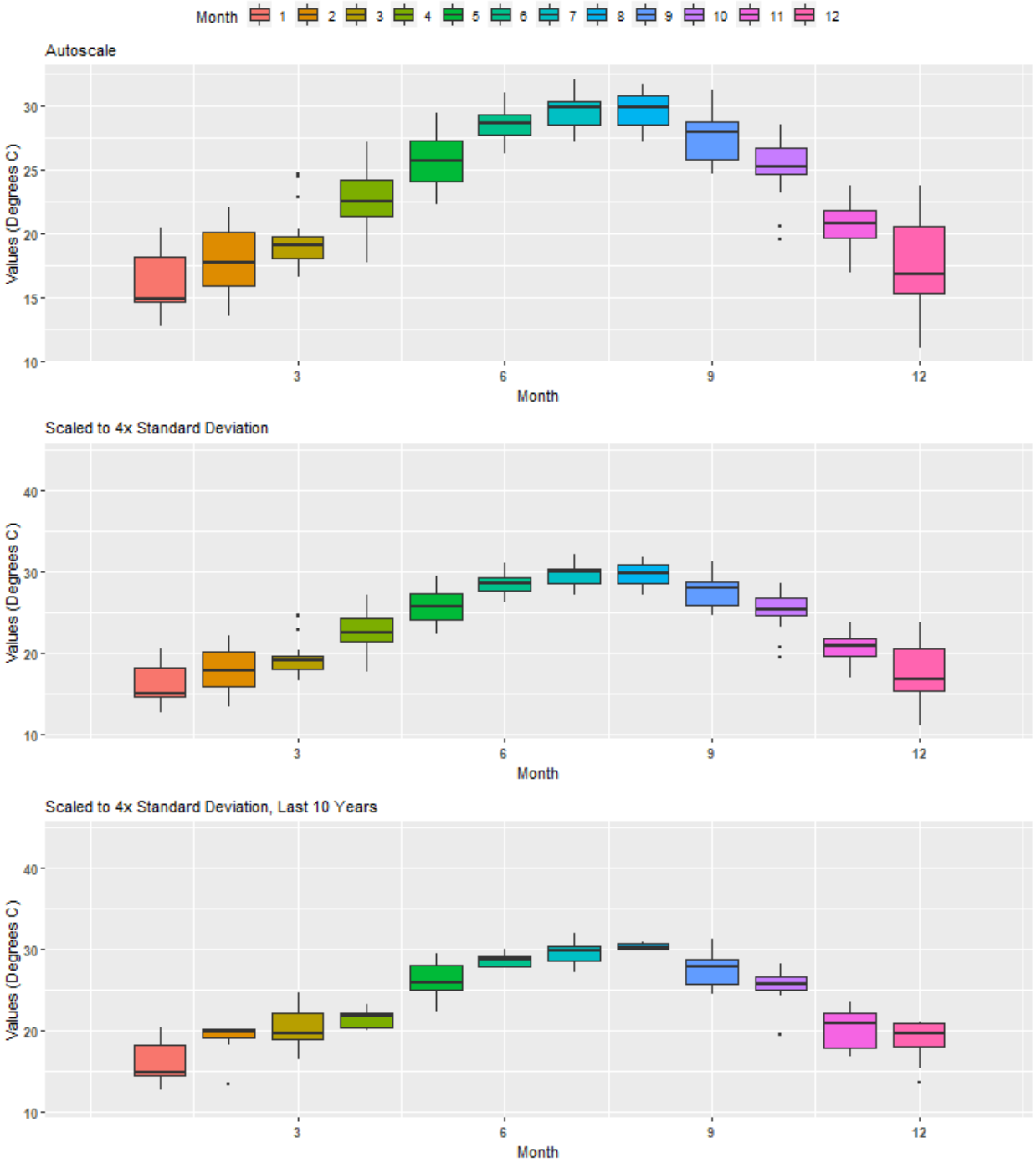


Summary Box Plots for Tomoka Marsh Aquatic Preserve

By Year & Month

Summary Box Plots for Tomoka Marsh Aquatic Preserve

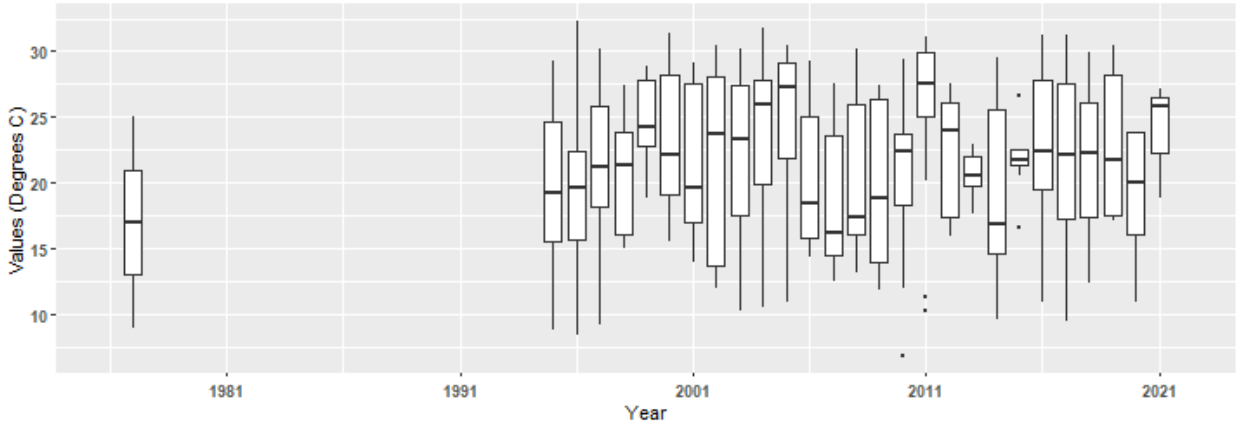
By Month



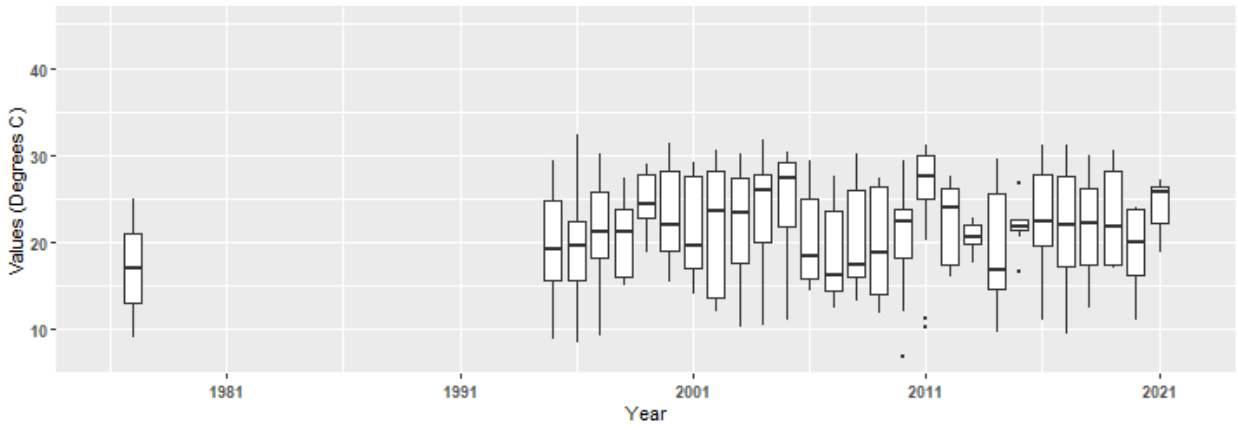
Summary Box Plots for Yellow River Marsh Aquatic Preserve

By Year

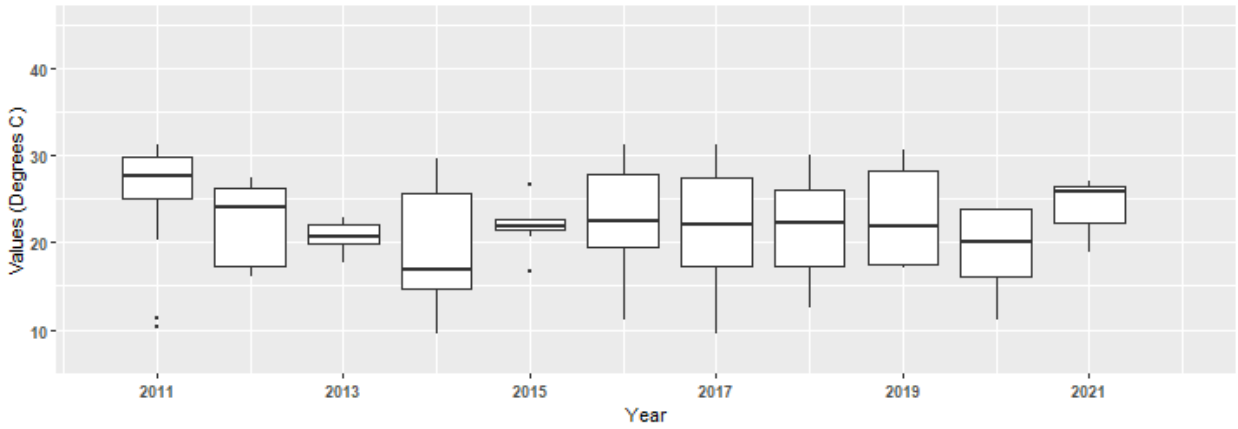
Autoscale



Scaled to 4x Standard Deviation

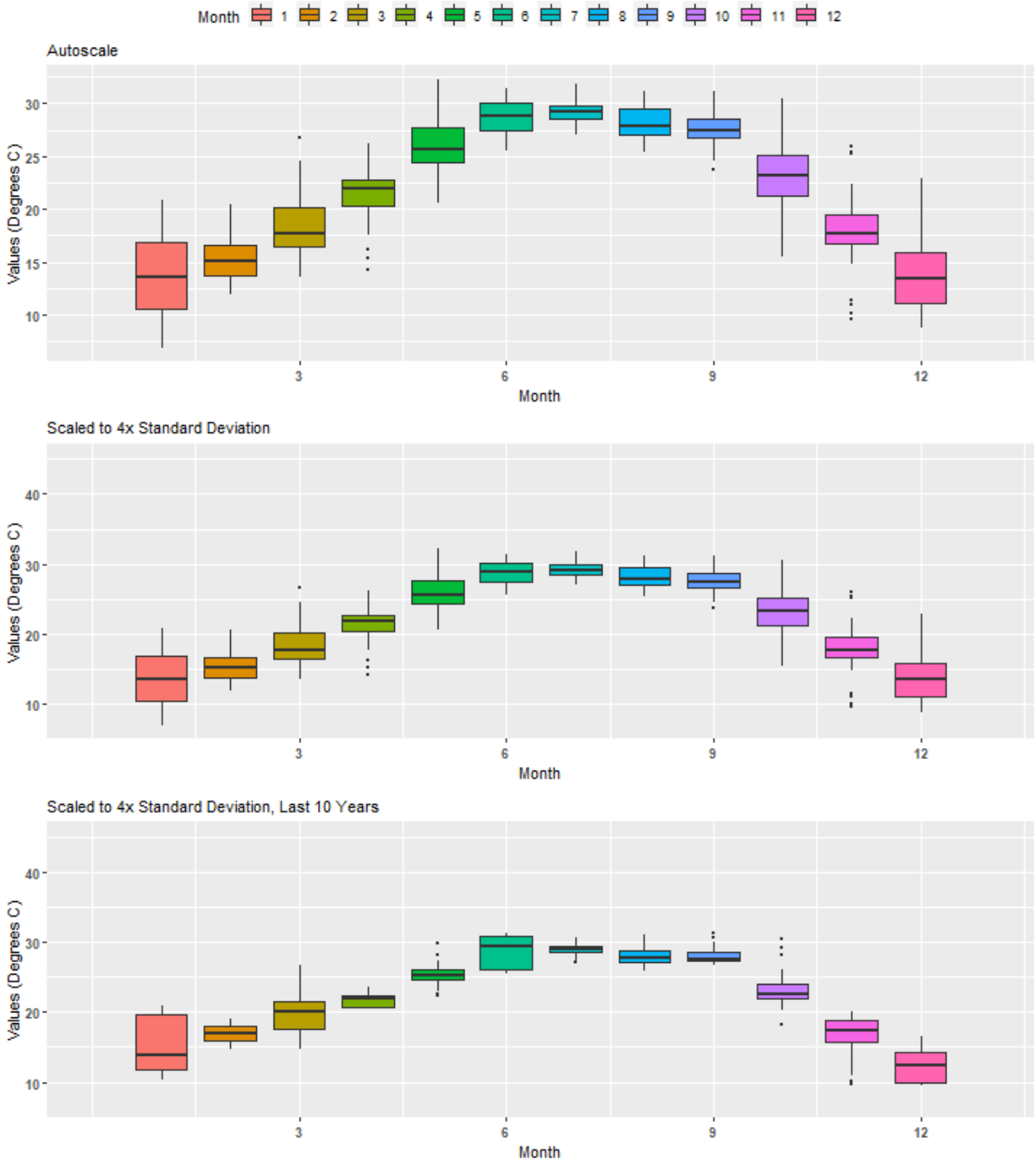


Scaled to 4x Standard Deviation, Last 10 Years



Summary Box Plots for Yellow River Marsh Aquatic Preserve

By Year & Month

Summary Box Plots for Yellow River Marsh Aquatic Preserve

By Month