SEACAR Continuous Water Quality Analysis: NE Region for Water Temperature

Last compiled on 24 June, 2022

Contents

Important Notes

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.

Libraries and Settings

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(plyr)
library(dplyr)
library(lubridate)
library(ggplot2)
library(ggpubr)
library(scales)
library(EnvStats)
library(tidyr)
library(kableExtra)

windowsFonts(`Segoe UI` = windowsFont('Segoe UI'))
options(scipen=999)
opts_chunk$set(warning=FALSE, message=FALSE, dpi=200)
```

File Import

Imports file that is determined in the WC_Continuous_parameter_ReportCompile.R script.

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 name of the parameter as it appears in the data file and units of the parameter.

```
data <- fread(file_in, sep="|", header=TRUE, stringsAsFactors=FALSE,
             select=c("ManagedAreaName", "ProgramID", "ProgramName",
                      "ProgramLocationID", "SampleDate", "Year", "Month",
                      "RelativeDepth", "ActivityType", "ParameterName",
                      "ResultValue", "ParameterUnits", "ValueQualifier",
                      "SEACAR_QAQCFlagCode", "Include"),
             na.strings="")
parameter <- unique(data$ParameterName)
unit <- unique(data$ParameterUnits)
```

Data Filtering

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

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

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

Because the continuous data is extensive and most measurements are taken every 15 minutes, a daily average is determined and used based on grouping `ManagedAreaName`, `ProgramID`, `ProgramName`, `ProgramLocationID`, and `SampleDate`. The new `ResultValue` is the mean of all values on that date from that specific monitoring location. Sets the `SampleDate` as a date object, and creates various scales of the date to be used by plotting functions.

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

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

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

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

   #temporarily removing FKNMS Temp. data because I think it might be causing R to run out of memory.
   # data <- data[data$ManagedAreaName != "Florida Keys National Marine Sanctuary"]
} else{
   data <- data[data$ResultValue>=0,]
}



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

data <- merge.data.frame(MA_All[,c("AreaID", "ManagedAreaName")],
                          data, by="ManagedAreaName", all=TRUE)

data$SampleDate <- as.Date(data$SampleDate)
data$YearMonth <- format(data$SampleDate, format = "%m-%Y")
data$YearMonthDec <- data$Year + ((data$Month-0.5) / 12)
data$DecDate <- decimal_date(data$SampleDate)

data <- data %>%
    group_by(ManagedAreaName, ProgramID, ProgramName, ProgramLocationID) %>%
    mutate(MonitoringID=cur_group_id())

Mon_Summ <- data %>%
   group_by(MonitoringID, AreaID, ManagedAreaName, ProgramID, ProgramName,
            ProgramLocationID) %>%
   dplyr::summarize(ParameterName=parameter,
             RelativeDepth=unique(RelativeDepth),
             N_Data=length(ResultValue[Include==TRUE & !is.na(ResultValue)]),
             N_Years=length(unique(Year[Include==TRUE & !is.na(Year)])),
             EarliestYear=min(Year[Include==TRUE]),
```

```
            LatestYear=max(Year[Include==TRUE]),
            SufficientData=ifelse(N_Data>0 & N_Years>=10, TRUE, FALSE))

Mon_Summ <- as.data.table(Mon_Summ[order(Mon_Summ$MonitoringID), ])

data <- merge.data.frame(data, Mon_Summ[,c("MonitoringID", "SufficientData")],
                         by="MonitoringID")

data$Use_In_Analysis <- ifelse(data$Include==TRUE &
                                  data$SufficientData==TRUE, TRUE, FALSE)
setDT(data)
data[, `:=` (relyear = Year - min(Year), relyear_dd = DecDate - min(DecDate)), by = "ManagedAreaName"]

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

Monitoring Location Statistics

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

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

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

```
                                                  Mon_YM_Stats$Month), ])
fwrite(Mon_YM_Stats, paste0(out_dir,"/", param_name, "_", region,
                            "_MonitoringLoc_YearMonth_Stats.txt"), sep="|")

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

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

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

Seasonal Kendall Tau Analysis

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

The following steps are performed:

1. Define the trend function.

2. Take the `data` variable and only include rows that have a `Use_In_Analysis` value of `TRUE`

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

```
tauSeasonal <- function(data, independent, stats.median, stats.minYear,
                        stats.maxYear, seasondata = Mon_M_Stats[Mon_M_Stats$ProgramLocationID==unique(da
  setDT(data)
  tau <- NULL
  tryCatch({ken <- kendallSeasonalTrendTest(
    y=data$ResultValue,
    season=data$Month,
    year=data$relyear,
    independent.obs=independent)

  tau <- ken$estimate[1]
  z <- ken$statistic[2]
  p_z <- ken$p.value[2]
  chi_sq <- ken$statistic[1]
  p_chi_sq <- ken$p.value[1]
  slope <- ken$estimate[2]
  intercept <- ken$estimate[3]
  trend <- trend_calculator(slope, stats.median, p_z)

  seasonresults <- as.data.table(ken$seasonal.estimates)
  rm(ken)
  }, warning = function(w) {
    print(w)
  }, error = function(e) {
    print(e)
  }, finally = {
    if (!exists("tau")) {
      tau <- NA
    }
   if (!exists("z")) {
      z <- NA
```

```
   }
   if (!exists("p_z")) {
      p_z <- NA
   }
   if (!exists("chi_sq")) {
      chi_sq <- NA
   }
   if (!exists("p_chi_sq")) {
      p_chi_sq <- NA
   }
   if (!exists("slope")) {
      slope <- NA
   }
   if (!exists("intercept")) {
      intercept <- NA
   }
   if (!exists("trend")) {
      trend <- NA
   }
})
KT <- data.table(MonitoringID = unique(data$MonitoringID),
                 season = "All",
                 stats.median = stats.median,
                 independent = independent,
                 tau = tau,
                 z = z,
                 p_z = p_z,
                 chi_sq = chi_sq,
                 p_chi_sq = p_chi_sq,
                 slope = slope,
                 intercept = intercept,
                 trend = trend)

seasonresults[, `:=` (MonitoringID = unique(data$MonitoringID),
                      season = sort(unique(data$Month)),
                      stats.median = as.numeric(NA),
                      independent = independent,
                      z = as.numeric(NA),
                      p_z = as.numeric(NA),
                      chi_sq = as.numeric(NA),
                      p_chi_sq = as.numeric(NA),
                      trend = as.integer(NA))]

for(s in as.integer(unique(seasonresults$season))){
  seasondat_s <- data[Month == s, ]
  if(!is.na(unique(seasondat_s$Month))){
    trend_s <- trend_calculator(seasonresults[season == s, slope], seasondata[Month == s, Median], p
    ken_s <- kendallTrendTest(ResultValue ~ relyear, data = seasondat_s)
    seasonresults[season == s, `:=` (stats.median = unique(seasondata[Month == s, Median]),
                                     z = ken_s$statistic,
                                     p_z = ken_s$p.value,
                                     chi_sq = NA,
                                     p_chi_sq = NA,
```

```r
                                         trend = trend_s)]
      } else{
        next
      }
    }

    seasonresults[, season := as.character(season)]

    KT <- rbind(KT, seasonresults)
    KT[, season := factor(season, levels = c("All", seq(1:12)), ordered = TRUE)]
    return(KT)
}
runStats <- function(data, Mon_M_Stats) {
    data$Index <- as.Date(data$SampleDate) # , "%Y-%m-%d")
    data$ResultValue <- as.numeric(data$ResultValue)
    # Calculate basic stats
    stats.median <- median(data$ResultValue, na.rm=TRUE)
    stats.minYear <- min(data$relyear, na.rm=TRUE)
    stats.maxYear <- max(data$relyear, na.rm=TRUE)
    # Calculate Kendall Tau and Slope stats,
    # then update appropriate columns and table
    seasondata <- Mon_M_Stats[Mon_M_Stats$ProgramLocationID==unique(data$ProgramLocationID[data$Monitori
    KT <- tauSeasonal(data, TRUE, stats.median,
                      stats.minYear, stats.maxYear, seasondata)
    #if (is.null(KT[8])) {
    if (is.na(KT[season == "All", trend])) {
       KT <- tauSeasonal(data, FALSE, stats.median,
                         stats.minYear, stats.maxYear, seasondata)
    }
    if (is.null(KT.Stats)==TRUE) {
       KT.Stats <- KT
    } else{
       KT.Stats <- rbind(KT.Stats, KT)
    }
    return(KT.Stats)
}
trend_calculator <- function(slope, median_value, p) {
    trend <-
       if (p < .05 & abs(slope) > abs(median_value) / 10.) {
          if (slope > 0) {
             2
          }
          else {
             -2
          }
       }
    else if (p < .05 & abs(slope) < abs(median_value) / 10.) {
       if (slope > 0) {
          1
       }
       else {
          -1
       }
```

```
   }
   else
      0
   return(trend)
}
KT.Stats <- NULL
# Loop that goes through each managed area.
# List of managed areas stored in MA_Years$ManagedAreaName
c_names <- c("MonitoringID", "Season", "Median", "Independent",
             "tau", "z", "p_z", "chi_sq", "p_chi_sq", "SennSlope", "SennIntercept", "Trend")
if(n==0){
   KT.Stats <- data.frame(matrix(ncol=length(c_names),
                                 nrow=nrow(Mon_Summ)))
   colnames(KT.Stats) <- c_names
   #KT.Stats[, c("MonitoringID")] <- Mon_Summ[, c("MonitoringID")]
} else{
   for (i in 1:n) {
      x <- nrow(data[data$Use_In_Analysis==TRUE &
                        data$MonitoringID==Mon_IDs[i], ])
      if (x>0) {
         KT.Stats <- runStats(data[data$Use_In_Analysis==TRUE &
                                      data$MonitoringID==Mon_IDs[i], ], Mon_M_Stats)
      }
   }
   KT.Stats <- as.data.frame(KT.Stats)

   if(dim(KT.Stats)[2]==1){
      KT.Stats <- as.data.frame(t(KT.Stats))
   }
   colnames(KT.Stats) <- c_names
   rownames(KT.Stats) <- seq(1:nrow(KT.Stats))
   KT.Stats$tau <- round(as.numeric(KT.Stats$tau), digits=4)
   KT.Stats$z <- round(as.numeric(KT.Stats$z), digits=4)
   KT.Stats$p_z <- round(as.numeric(KT.Stats$p_z), digits=4)
   KT.Stats$chi_sq <- round(as.numeric(KT.Stats$chi_sq), digits=4)
   KT.Stats$p_chi_sq <- round(as.numeric(KT.Stats$p_chi_sq), digits=4)
   KT.Stats$SennSlope <- as.numeric(KT.Stats$SennSlope)
   KT.Stats$SennIntercept <- as.numeric(KT.Stats$SennIntercept)
   KT.Stats$Trend <- as.integer(KT.Stats$Trend)
}

KT.Stats <-  merge.data.frame(Mon_Summ, KT.Stats,
                              by=c("MonitoringID"), all=TRUE)

KT.Stats <- as.data.table(KT.Stats[order(KT.Stats$MonitoringID), ])
KT.Stats2 <- copy(KT.Stats)
KT.Stats[, `:=` (Region = region, Units = unit)]
KT.Stats_all <- rbind(KT.Stats_all, KT.Stats)

KT.Stats2$MonitoringID <- NULL
fwrite(KT.Stats2, paste0(out_dir,"/", param_name, "_", region,
                           "_KendallTau_Stats.txt"), sep="|")
rm(KT.Stats2)
```

```
#KT.Stats$MonitoringID <- Mon_Summ$MonitoringID
data <- data[!is.na(data$ResultValue),]
```

Appendix I: Dataset Summary Box Plots

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

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

This set of box plots are grouped by year.

```
plot_theme <- theme_bw() +
   theme(text=element_text(family="Segoe UI"),
         title=element_text(face="bold"),
         plot.title=element_text(hjust=0.5, size=14, color="#314963"),
         plot.subtitle=element_text(hjust=0.5, size=10, color="#314963"),
         axis.title.x = element_text(margin = margin(t = 5, r = 0,
                                                     b = 10, l = 0)),
         axis.title.y = element_text(margin = margin(t = 0, r = 10,
                                                     b = 0, l = 0)),
         axis.text=element_text(size=10),
         #axis.text.x=element_text(face="bold", angle = 60, hjust = 1),
         axis.text.x=element_text(face="bold"),
         axis.text.y=element_text(face="bold"))

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=SampleDate, y=ResultValue, group=Year)) +
   geom_boxplot(color="#333333", fill="#cccccc", outlier.shape=21,
                outlier.size=3, outlier.color="#333333",
                outlier.fill="#cccccc", outlier.alpha=0.75) +
   labs(subtitle="Autoscale", x="Year",
        y=paste0("Values (", unit, ")")) +
   plot_theme
```

```r
p2 <- ggplot(data=data[data$Include==TRUE, ],
             aes(x=SampleDate, y=ResultValue, group=Year)) +
   geom_boxplot(color="#333333", fill="#cccccc", outlier.shape=21,
                outlier.size=3, outlier.color="#333333",
                outlier.fill="#cccccc", outlier.alpha=0.75) +
   labs(subtitle="Scaled to 4x Standard Deviation", x="Year",
        y=paste0("Values (", unit, ")")) +
   ylim(0, y_scale) +
   plot_theme

p3 <- ggplot(data=data[data$Include==TRUE, ],
             aes(x=Year, y=ResultValue, group=Year)) +
   geom_boxplot(color="#333333", fill="#cccccc", outlier.shape=21,
                outlier.size=3, outlier.color="#333333",
                outlier.fill="#cccccc", outlier.alpha=0.75) +
   labs(subtitle="Scaled to 4x Standard Deviation, Last 10 Years",
        x="Year", y=paste0("Values (", unit, ")")) +
   ylim(0, y_scale) +
   scale_x_continuous(limits=c(max(data$Year) - 10.5, max(data$Year)+0.5),
                      breaks=seq(max(data$Year) - 10, max(data$Year), 2)) +
   plot_theme

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

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

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

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

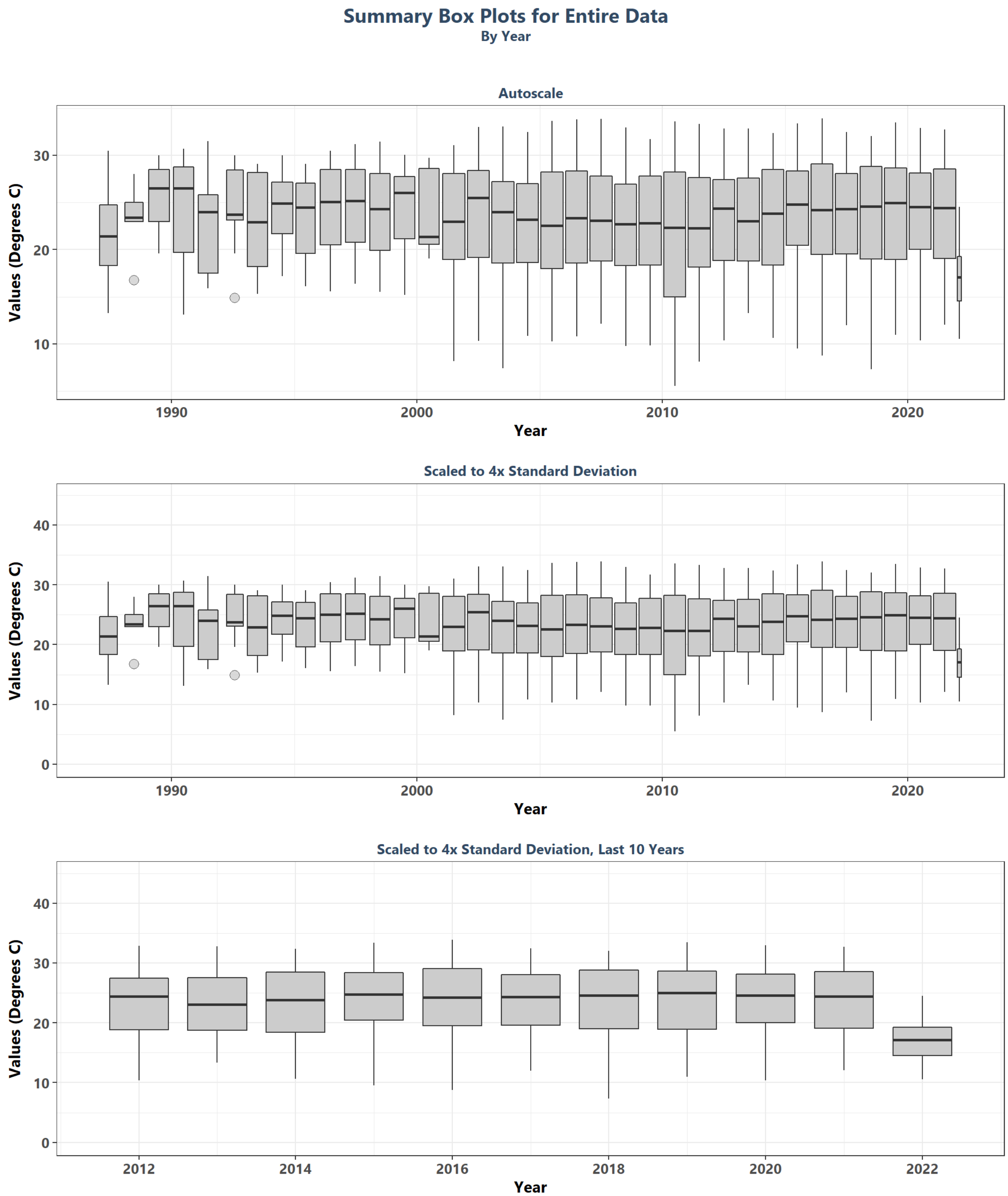
```r
p1 <- ggplot(data=data[data$Include==TRUE, ],
             aes(x=YearMonthDec, y=ResultValue,
                 group=YearMonth, color=as.factor(Month))) +
   geom_boxplot(fill="#cccccc", outlier.size=1.5, outlier.alpha=0.75) +
   labs(subtitle="Autoscale", x="Year",
        y=paste0("Values (", unit, ")"), color="Month") +
   plot_theme +
   theme(legend.position="top", legend.box="horizontal") +
   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(fill="#cccccc", outlier.size=1.5, outlier.alpha=0.75) +
   labs(subtitle="Scaled to 5x Standard Deviation",
        x="Year", y=paste0("Values (", unit, ")")) +
   ylim(0, y_scale) +
   plot_theme +
   theme(legend.position="none")
```

```
p3 <- ggplot(data=data[data$Include==TRUE, ],
             aes(x=YearMonthDec, y=ResultValue,
                 group=YearMonth, color=as.factor(Month))) +
   geom_boxplot(fill="#cccccc", outlier.size=1.5, outlier.alpha=0.75) +
   labs(subtitle="Scaled to 5x Standard Deviation, Last 10 Years",
        x="Year", y=paste0("Values (", unit, ")")) +
   ylim(0, y_scale) +
   scale_x_continuous(limits=c(max(data$Year) - 10.5, max(data$Year)+0.5),
                      breaks=seq(max(data$Year) - 10, max(data$Year), 2)) +
   plot_theme +
   theme(legend.position="none")
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") + plot_theme +
   theme(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))
```

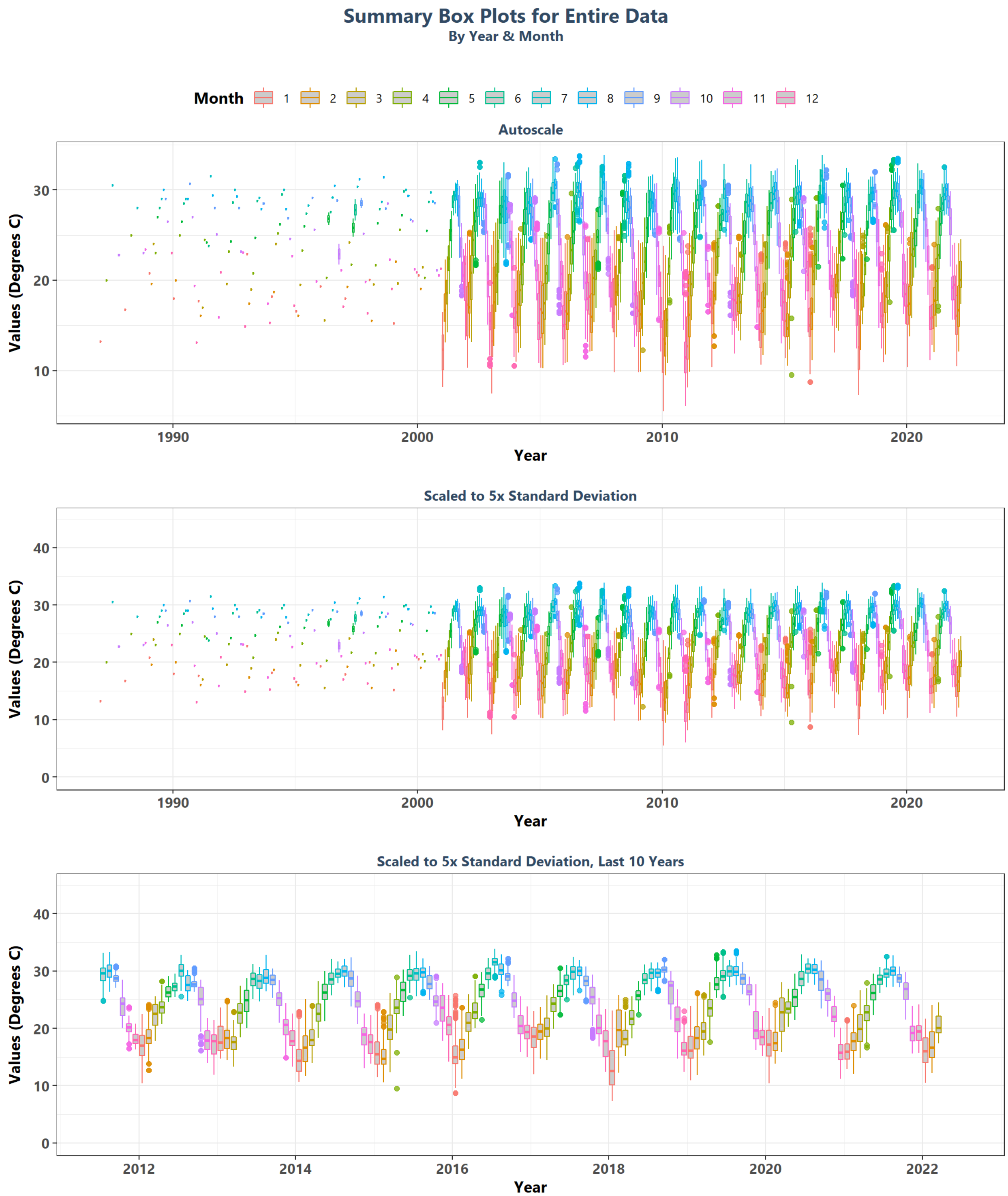
The following box plots are grouped by month with fill color being related to the month. This is designed to view potential seasonal trends.

```
p1 <- ggplot(data=data[data$Include==TRUE, ],
             aes(x=Month, y=ResultValue,
                 group=Month, fill=as.factor(Month))) +
   geom_boxplot(color="#333333", outlier.shape=21, outlier.size=3,
                outlier.color="#333333", outlier.alpha=0.75) +
   labs(subtitle="Autoscale", x="Month",
        y=paste0("Values (", unit, ")"), fill="Month") +
   scale_x_continuous(limits=c(0, 13), breaks=seq(3, 12, 3)) +
   plot_theme +
   theme(legend.position="top", legend.box="horizontal") +
   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(color="#333333", outlier.shape=21, outlier.size=3,
                outlier.color="#333333", outlier.alpha=0.75) +
   labs(subtitle="Scaled to 5x Standard Deviation",
        x="Month", y=paste0("Values (", unit, ")")) +
   ylim(0, y_scale) +
   scale_x_continuous(limits=c(0, 13), breaks=seq(3, 12, 3)) +
   plot_theme +
   theme(legend.position="none")

p3 <- ggplot(data=data[data$Include==TRUE &
                            data$Year >= max(data$Year) - 10, ],
             aes(x=Month, y=ResultValue,
                 group=Month, fill=as.factor(Month))) +
```

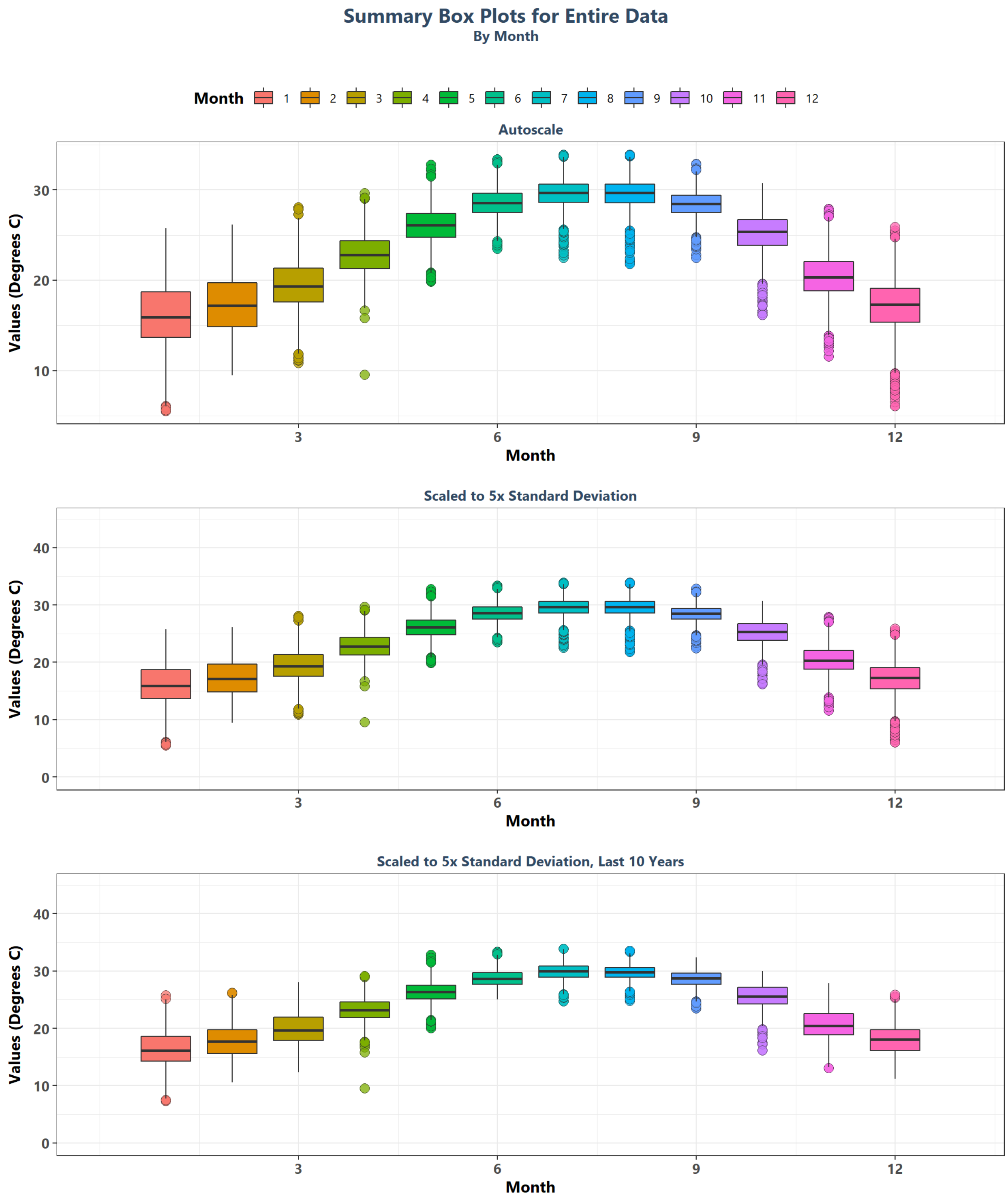
```r
  geom_boxplot(color="#333333", outlier.shape=21, outlier.size=3,
               outlier.color="#333333", outlier.alpha=0.75) +
  labs(subtitle="Scaled to 5x Standard Deviation, Last 10 Years",
       x="Month", y=paste0("Values (", unit, ")")) +
  ylim(0, y_scale) +
  scale_x_continuous(limits=c(0, 13), breaks=seq(3, 12, 3)) +
  plot_theme +
  theme(legend.position="none")
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") + plot_theme +
  theme(panel.border=element_blank(), panel.grid.major=element_blank(),
        panel.grid.minor=element_blank(), axis.line=element_blank())

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



Summary Box Plots for Entire Data
By Year
Autoscale
Values (Degrees C)
30
20
10
1990
2000
2010
2020
Year
Scaled to 4x Standard Deviation
Values (Degrees C)
40
30
20
10
0
1990
2000
2010
2020
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (Degrees C)
40
30
20
10
0
2012
2014
2016
2018
2020
2022
Year



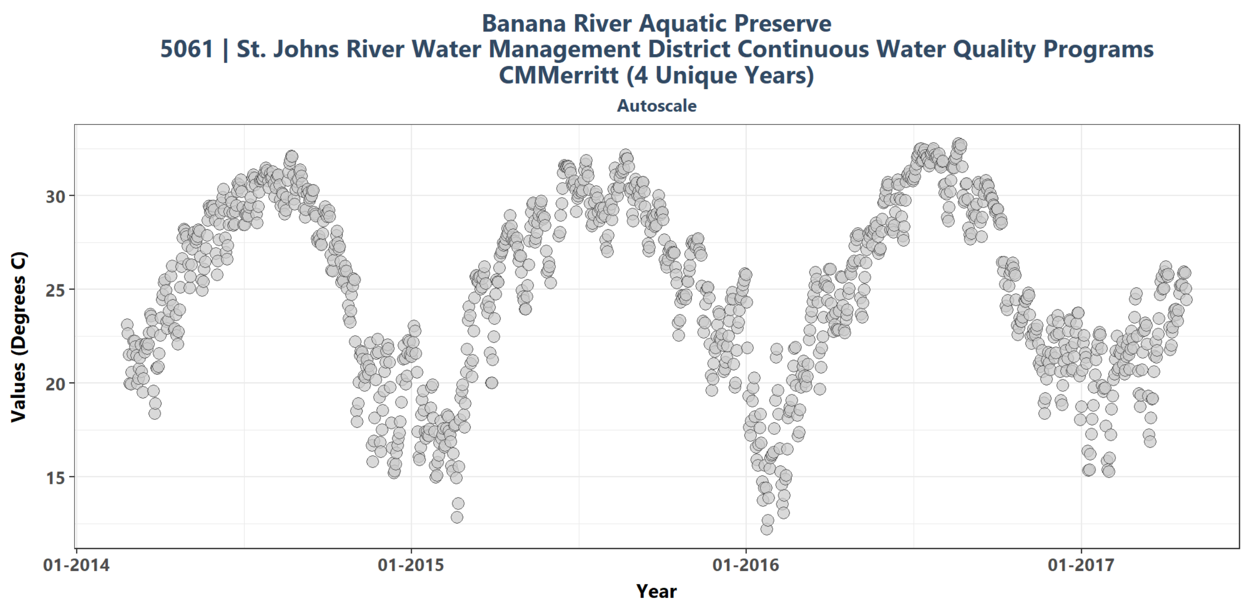
Summary Box Plots for Entire Data
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (Degrees C)
30
20
10
1990
2000
2010
2020
Year
Scaled to 5x Standard Deviation
Values (Degrees C)
40
30
20
10
0
1990
2000
2010
2020
Year
Scaled to 5x Standard Deviation, Last 10 Years
Values (Degrees C)
40
30
20
10
0
2012
2014
2016
2018
2020
2022
Year



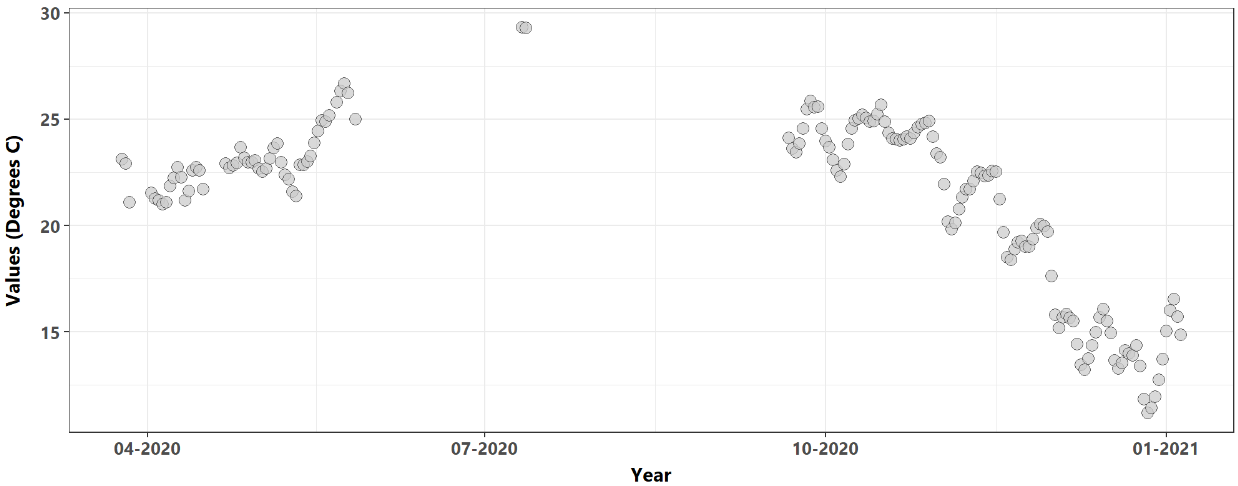
Appendix II: Excluded Monitoring Locations

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

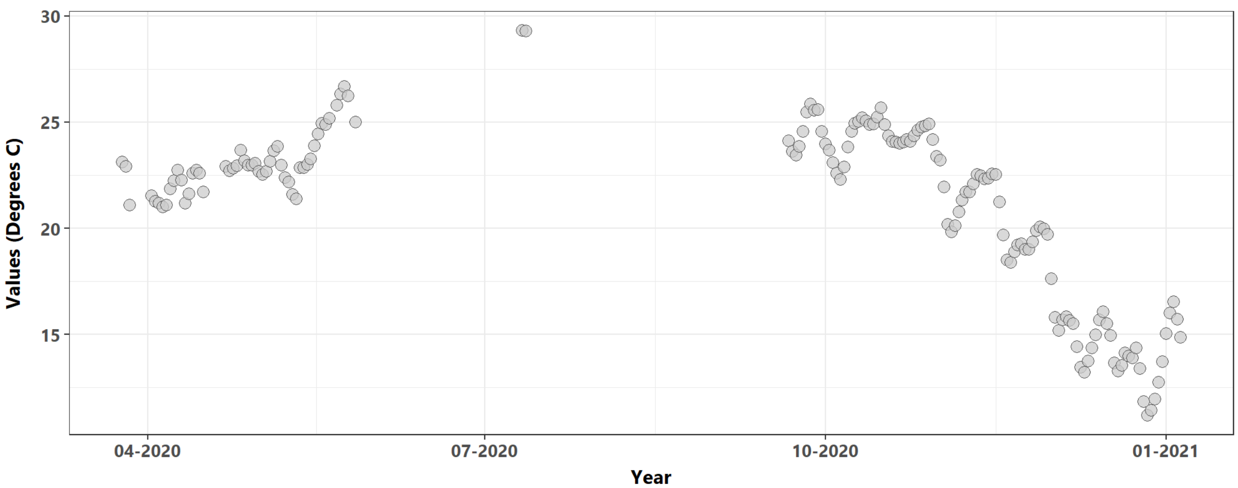
```
Mon_Exclude <- Mon_Summ[Mon_Summ$N_Years<5 & Mon_Summ$N_Years>0,]
Mon_Exclude <- Mon_Exclude[order(Mon_Exclude$MonitoringID),]
z=nrow(Mon_Exclude)

if(z==0){
   print("There are no monitoring locations that qualify.")
} else {
   for(i in 1:z){
      MA_name <- unique(data$ManagedAreaName[
         data$MonitoringID==Mon_Exclude$MonitoringID[i]])
      Mon_name <- paste0(unique(data$ProgramID[
         data$MonitoringID==Mon_Exclude$MonitoringID[i]]), " | ",
         unique(data$ProgramName[
            data$MonitoringID==Mon_Exclude$MonitoringID[i]]), "\n",
         unique(data$ProgramLocationID[
            data$MonitoringID==Mon_Exclude$MonitoringID[i]]))

      p1<-ggplot(data=data[data$MonitoringID==Mon_Exclude$MonitoringID[i]&
                             data$Include==TRUE, ],
                 aes(x=SampleDate, y=ResultValue)) +
         geom_point(shape=21, size=3, color="#333333", fill="#cccccc",
                    alpha=0.75) +
         labs(title=paste0(MA_name, "\n",
                           Mon_name, " (", Mon_Exclude$N_Years[i],
                        " Unique Years)"),
              subtitle="Autoscale", x="Year",
              y=paste0("Values (", unit, ")")) +
         plot_theme +
         scale_x_date(labels=date_format("%m-%Y"))
      print(p1)
   }
}
```

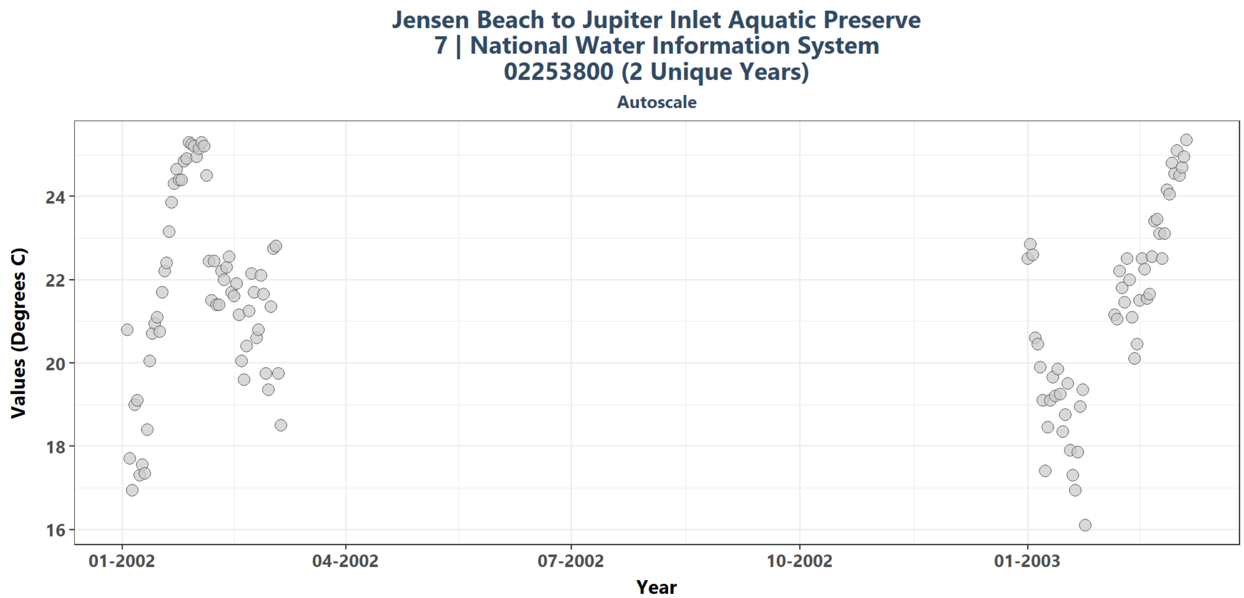


Guana River Marsh Aquatic Preserve
5062 | FDEP Bureau of Survey and Mapping Continuous Water Quality Program
872-0494 (2 Unique Years)
Autoscale

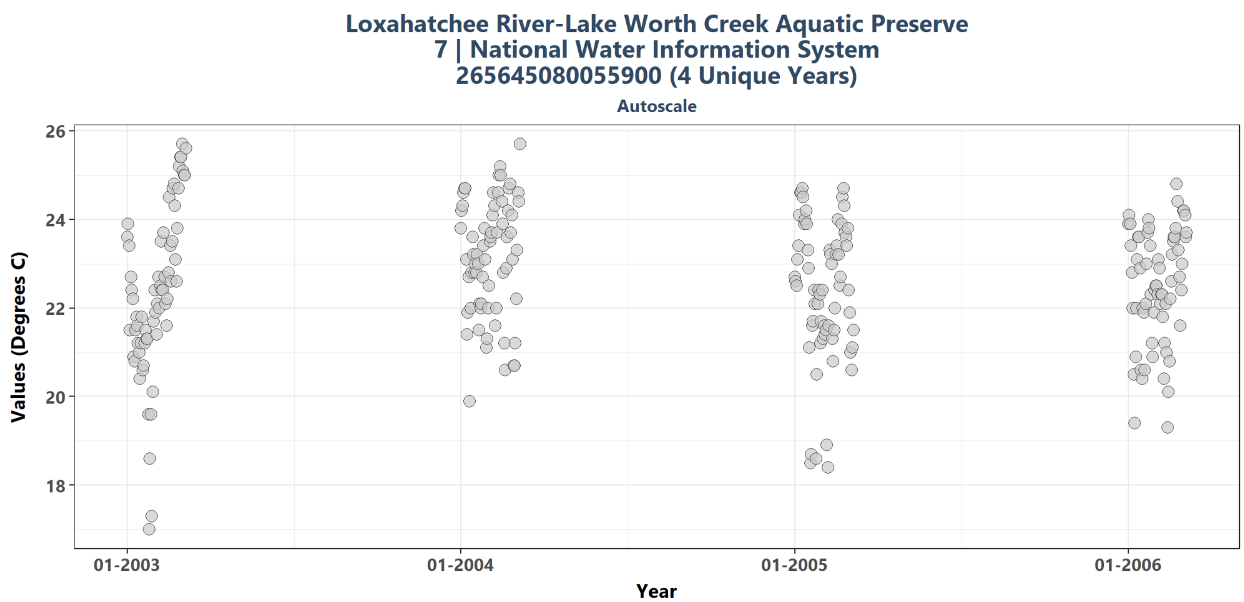


Guana Tolomato Matanzas National Estuarine Research Reserve
5062 | FDEP Bureau of Survey and Mapping Continuous Water Quality Program
872-0494 (2 Unique Years)
Autoscale

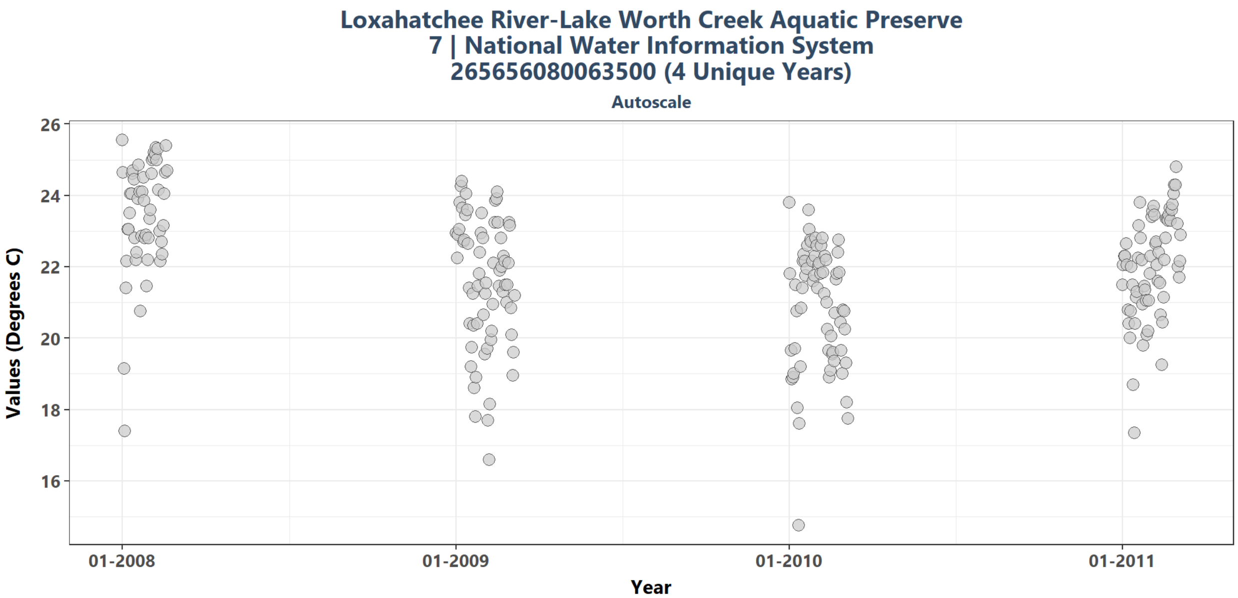


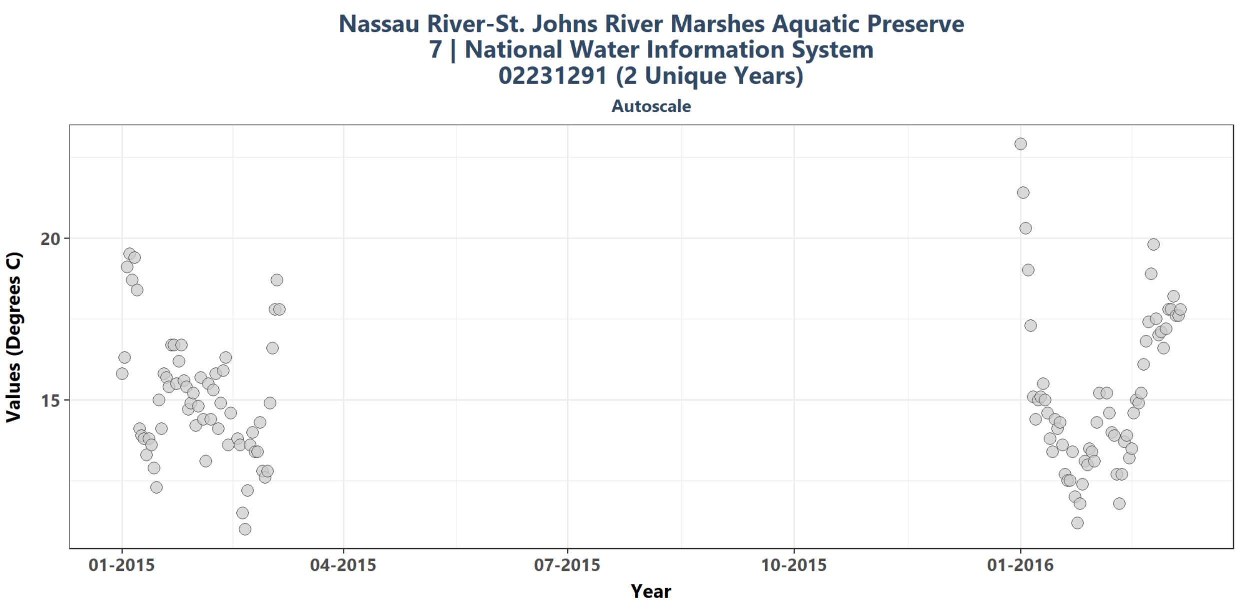


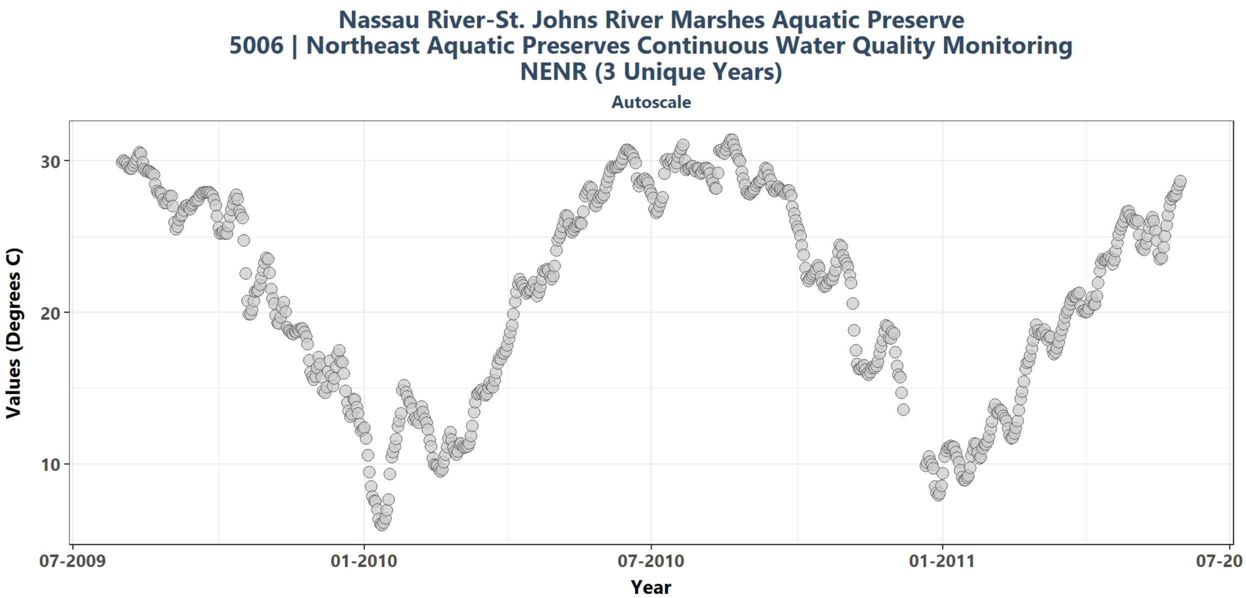
Jensen Beach to Jupiter Inlet Aquatic Preserve
7 | National Water Information System
02253800 (2 Unique Years)
Autoscale
Values (Degrees C)
16
18
20
22
24
01-2002
04-2002
07-2002
10-2002
01-2003
Year

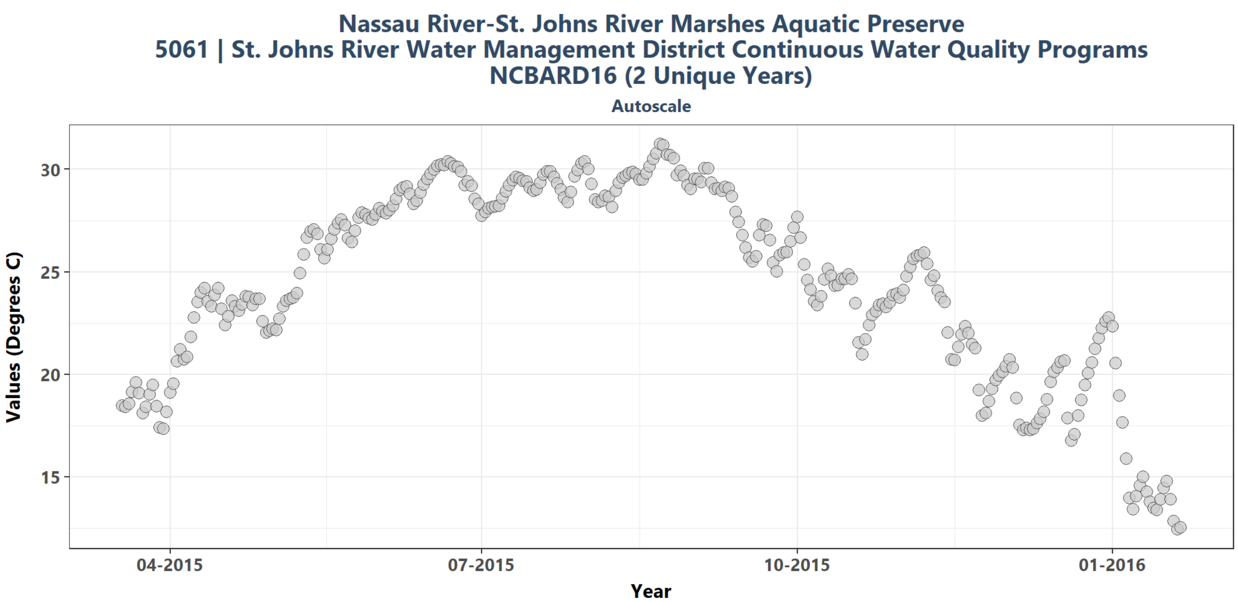


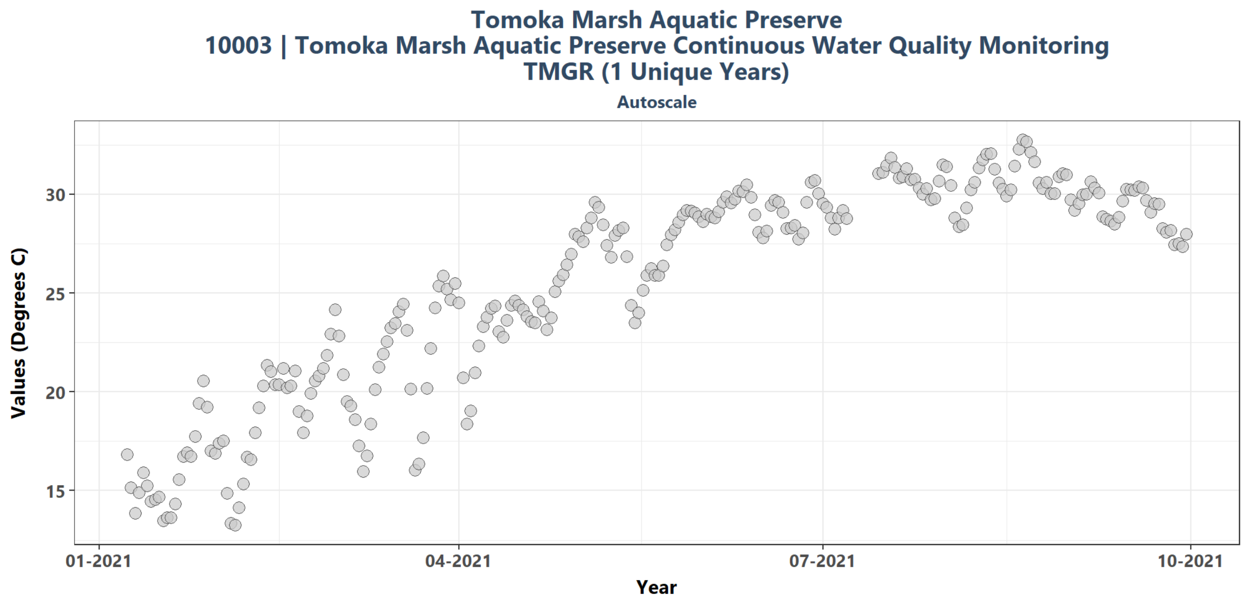
Loxahatchee River-Lake Worth Creek Aquatic Preserve
7 | National Water Information System
265645080055900 (4 Unique Years)
Autoscale
Values (Degrees C)
18
20
22
24
26
01-2003
01-2004
01-2005
01-2006
Year


Loxahatchee River-Lake Worth Creek Aquatic Preserve
7 | National Water Information System
265656080063500 (4 Unique Years)
Autoscale
Values (Degrees C)
26
24
22
20
18
16
01-2008
01-2009
01-2010
01-2011
Year


Nassau River-St. Johns River Marshes Aquatic Preserve
7 | National Water Information System
02231291 (2 Unique Years)
Autoscale
Values (Degrees C)
20
15
01-2015
04-2015
07-2015
10-2015
01-2016
Year


Nassau River-St. Johns River Marshes Aquatic Preserve
5006 | Northeast Aquatic Preserves Continuous Water Quality Monitoring
NENR (3 Unique Years)
Autoscale
Values (Degrees C)
30
20
10
07-2009
01-2010
07-2010
01-2011
07-20
Year


Nassau River-St. Johns River Marshes Aquatic Preserve
5061 | St. Johns River Water Management District Continuous Water Quality Programs
NCBARD16 (2 Unique Years)
Autoscale
Values (Degrees C)
30
25
20
15
04-2015
07-2015
10-2015
01-2016
Year



Appendix III: Monitoring Location Trendlines

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

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

```
if(n==0){
   print("There are no monitoring locations that qualify.")
} else {
   for (i in 1:n) {
      plot_data <- data[data$Use_In_Analysis==TRUE &
                                 data$MonitoringID==Mon_IDs[i],]
      plot_data$Season <- factor(plot_data$Month, levels = c("All", seq(1, 12)), ordered = TRUE)
      year_lower <- min(plot_data$relyear)
      year_upper <- max(plot_data$relyear)
      # relyear_dd_lower <- min(plot_data$relyear_dd)
      # relyear_dd_upper <- max(plot_data$relyear_dd)
```

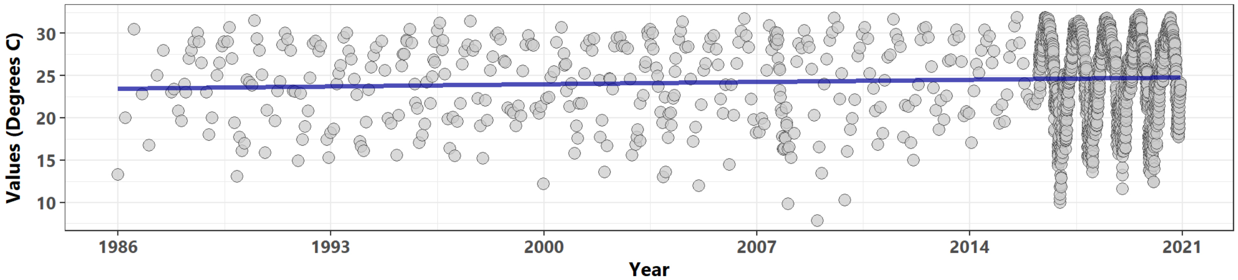
```
min_RV <- min(plot_data$ResultValue)
mn_RV <- mean(plot_data$ResultValue[plot_data$ResultValue <
                                  quantile(plot_data$ResultValue, 0.98)])
sd_RV <- sd(plot_data$ResultValue[plot_data$ResultValue <
                                quantile(plot_data$ResultValue, 0.98)])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

tau <- KT.Stats$tau[KT.Stats$MonitoringID==Mon_IDs[i]]
s_slope <- KT.Stats$SennSlope[KT.Stats$MonitoringID==Mon_IDs[i]]
s_int <- KT.Stats$SennIntercept[KT.Stats$MonitoringID==Mon_IDs[i]]
trend <- KT.Stats$Trend[KT.Stats$MonitoringID==Mon_IDs[i]]
z <- KT.Stats$z[KT.Stats$MonitoringID==Mon_IDs[i]]
p_z <- KT.Stats$p_z[KT.Stats$MonitoringID==Mon_IDs[i]]
chi_sq <- KT.Stats$chi_sq[KT.Stats$MonitoringID==Mon_IDs[i]]
p_chi_sq <- KT.Stats$p_chi_sq[KT.Stats$MonitoringID==Mon_IDs[i]]


MA_name <- KT.Stats$ManagedAreaName[KT.Stats$MonitoringID==Mon_IDs[i]]
Mon_name <- paste0(KT.Stats$ProgramID[KT.Stats$MonitoringID==Mon_IDs[i]],
   " | ", KT.Stats$ProgramName[KT.Stats$MonitoringID==Mon_IDs[i]], "\n",
   KT.Stats$ProgramLocationID[KT.Stats$MonitoringID==Mon_IDs[i]])

xbrks <- seq(round_any(min(plot_data$relyear_dd), 5, floor), round_any(max(plot_data$relyear_dd),
             by = (round_any(max(plot_data$relyear_dd), 5, ceiling) - round_any(min(plot_data$rely

xlabs <- seq(max(plot_data$Year) - round_any(max(plot_data$relyear_dd), 5, ceiling),
             max(plot_data$Year),
             by = (max(plot_data$Year) - (max(plot_data$Year) - round_any(max(plot_data$relyear_dd

# x1 <- min(unique(data$relyear_dd[data$Year == KT.Stats[ManagedAreaName==MA_Include[i] & Season =
# y1 <- min(unique(data$relyear_dd[data$Year == KT.Stats[ManagedAreaName==MA_Include[i] & Season =
# x_end1 <- max(unique(data$relyear_dd[data$Year == KT.Stats[ManagedAreaName==MA_Include[i] & Sea
# y_end1 <- max(unique(data$relyear_dd[data$Year == KT.Stats[ManagedAreaName==MA_Include[i] & Sea
# x1 <- relyear_dd_lower
# y1 <- relyear_dd_lower * KT.Stats[ManagedAreaName==MA_Include[i] & Season == "All", SennSlope] +
# x_end1 <- relyear_dd_upper
# y_end1 <- relyear_dd_upper * KT.Stats[ManagedAreaName==MA_Include[i] & Season == "All", SennSlo

KT.Stats[, season := Season]
KT.Stats[MonitoringID == Mon_IDs[i] & season != "All", `:=` (N_Data = nrow(plot_data[Season == sea
KT.Stats[MonitoringID == Mon_IDs[i] & season == "All", `:=` (relyear_dd_lower = min(plot_data[Moni
KT.Stats[, season := NULL]


p1 <- ggplot(data=plot_data,
             aes(x=relyear_dd, y=ResultValue)) +
   geom_point(shape=21, size=3, color="#333333", fill="#cccccc",
              alpha=0.75) +
   #geom_abline(data = KT.Stats[ManagedAreaName==MA_Include[i] & Season == "All", ], aes(slope=Se
   #             color="#000099", size=1.2, alpha=0.7) +
   geom_segment(data = KT.Stats[MonitoringID == Mon_IDs[i] & Season == "All", ], aes(x = relyear_
                                                                                    y = relyear_
```

```
                                                                                    xend = relyea
                                                                                    yend = relyea
                 color="#000099", size=1.2, alpha=0.7) +
    labs(subtitle="Autoscale",
         x="Year", y=paste0("Values (", unit, ")")) +
    plot_theme +
    scale_x_continuous(breaks = xbrks,
                       labels = xlabs)


  p2 <- ggplot(data=plot_data,
               aes(x=relyear_dd, y=ResultValue)) +
    geom_point(shape=21, size=3, color="#333333", fill="#cccccc",
               alpha=0.75) +
    # geom_abline(aes(slope=s_slope, intercept=s_int),
    #             color="#000099", size=1.2, alpha=0.7) +
    geom_segment(data = KT.Stats[MonitoringID == Mon_IDs[i] & Season == "All", ], aes(x = relyear_d
                                                                                    y = relyear_d
                                                                                    xend = relyea
                                                                                    yend = relyea
                 color="#000099", size=1.2, alpha=0.7) +
    ylim(min_RV-0.1*y_scale, y_scale) +
    labs(subtitle="Scaled to 4x Standard Deviation",
         x="Year", y=paste0("Values (", unit, ")")) +
    plot_theme +
    scale_x_continuous(breaks = xbrks,
                       labels = xlabs)

  splot <- ggplot(plot_data, aes(x = relyear_dd, y = ResultValue)) +
      geom_point(shape = 21, size = 1.5, color="#333333", fill="#cccccc", alpha=0.75) +
      geom_segment(data = KT.Stats[MonitoringID == Mon_IDs[i] & Season != "All", ], aes(x = relyea
                                                                                      y = relyea
                                                                                      xend = rel
                                                                                      yend = rel
                 color="#000099", size=1.2, alpha=0.7) +
      #ylim(min_RV-0.1*y_scale, y_scale) +
      scale_x_continuous(breaks = xbrks,
                         labels = xlabs) +
      labs(y = paste0("Values (", unit, ")"), x = "Year", subtitle = "Results for Individual Seas
      facet_wrap(~Season, ncol = 3) +
      plot_theme

  KTset <- ggarrange(p1, p2, splot, ncol=1, heights=c(1, 1, 1.5))

  p0 <- ggplot() + labs(title=paste0(MA_name, "\n", Mon_name)) +
    plot_theme + theme(panel.border=element_blank(),
                       panel.grid.major=element_blank(),
                       panel.grid.minor=element_blank(),
                       axis.line=element_blank())

  KT.Stats[MonitoringID==Mon_IDs[i], `:=` (N = N_Data,
                                           Median = round(Median, 2),
                                           Slope = round(SennSlope, 4),
```

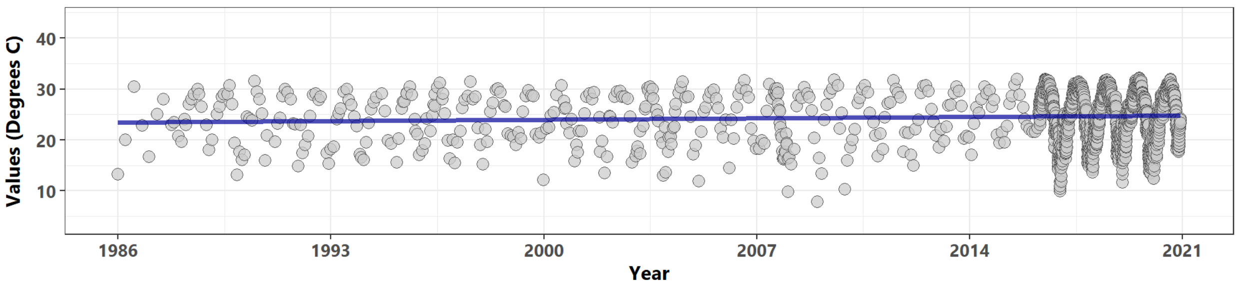
```
                                            Int. = round(SennIntercept, 4),
                                            z = round(z, 1),
                                            chi_sq = round(chi_sq, 1))]

      print(ggarrange(p0, KTset, ncol=1, heights=c(0.1, 1.25)))
      cat('\n')
      print(KT.Stats[KT.Stats$MonitoringID==Mon_IDs[i], ] %>%
         select(Season, N, Median, tau, Slope, Int., z, p_z, chi_sq, p_chi_sq, Trend) %>%
         kable(format="latex") %>%
            row_spec(0,bold=TRUE) %>%
            kable_styling(latex_options = "HOLD_position",
                          font_size = 7) %>%
            add_footnote(
            "p < 0.00005 appear as 0 due to rounding"))
      cat('\n')
      rm(plot_data)
      rm(KTset, leg)
   }
}
```

Banana River Aquatic Preserve
5061 | St. Johns River Water Management District Continuous Water Quality Programs
IRLB04

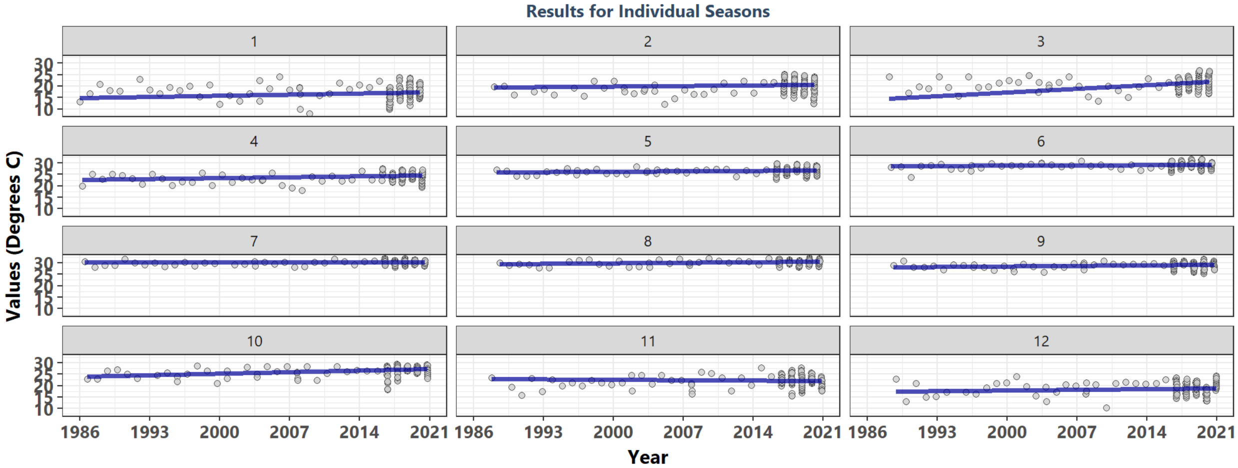
Autoscale



Scaled to 4x Standard Deviation



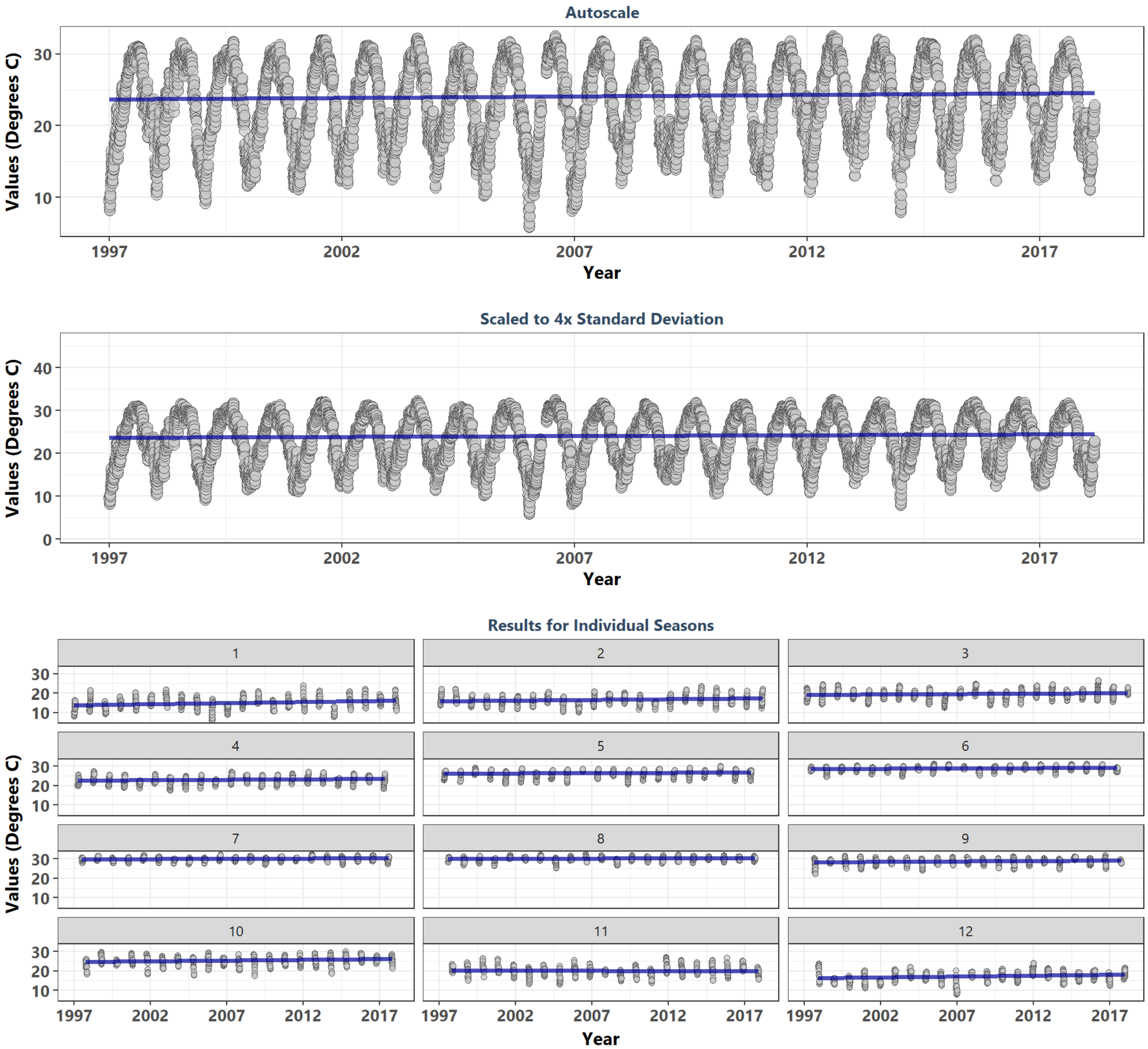
Results for Individual Seasons



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	2083	25.82	0.0806	0.0377	23.4554	5.4	0.0000	23	0.0176	1
1	158	17.08	0.0840	0.0693	14.8643	1.6	0.1118	NA	NA	1
2	143	20.41	0.0356	0.0327	19.3618	0.6	0.5231	NA	NA	1
3	155	21.30	0.2416	0.2286	13.9801	4.5	0.0000	NA	NA	1
4	157	24.43	0.1160	0.0582	22.5659	2.2	0.0287	NA	NA	1
5	187	26.72	0.0748	0.0303	25.7794	1.5	0.1242	NA	NA	1
6	180	29.10	0.0518	0.0169	28.5627	1.0	0.2963	NA	NA	1
7	186	30.22	0.0021	0.0004	30.2048	0.0	0.9667	NA	NA	1
8	185	30.45	0.1081	0.0371	29.2676	2.2	0.0270	NA	NA	1
9	184	29.17	0.0890	0.0340	28.1124	1.8	0.0699	NA	NA	1
10	189	26.85	0.1722	0.0967	23.8561	3.6	0.0004	NA	NA	1
11	182	22.05	-0.0387	-0.0324	23.0548	-0.8	0.4336	NA	NA	-1
12	177	18.73	0.0471	0.0444	17.3506	0.9	0.3465	NA	NA	1

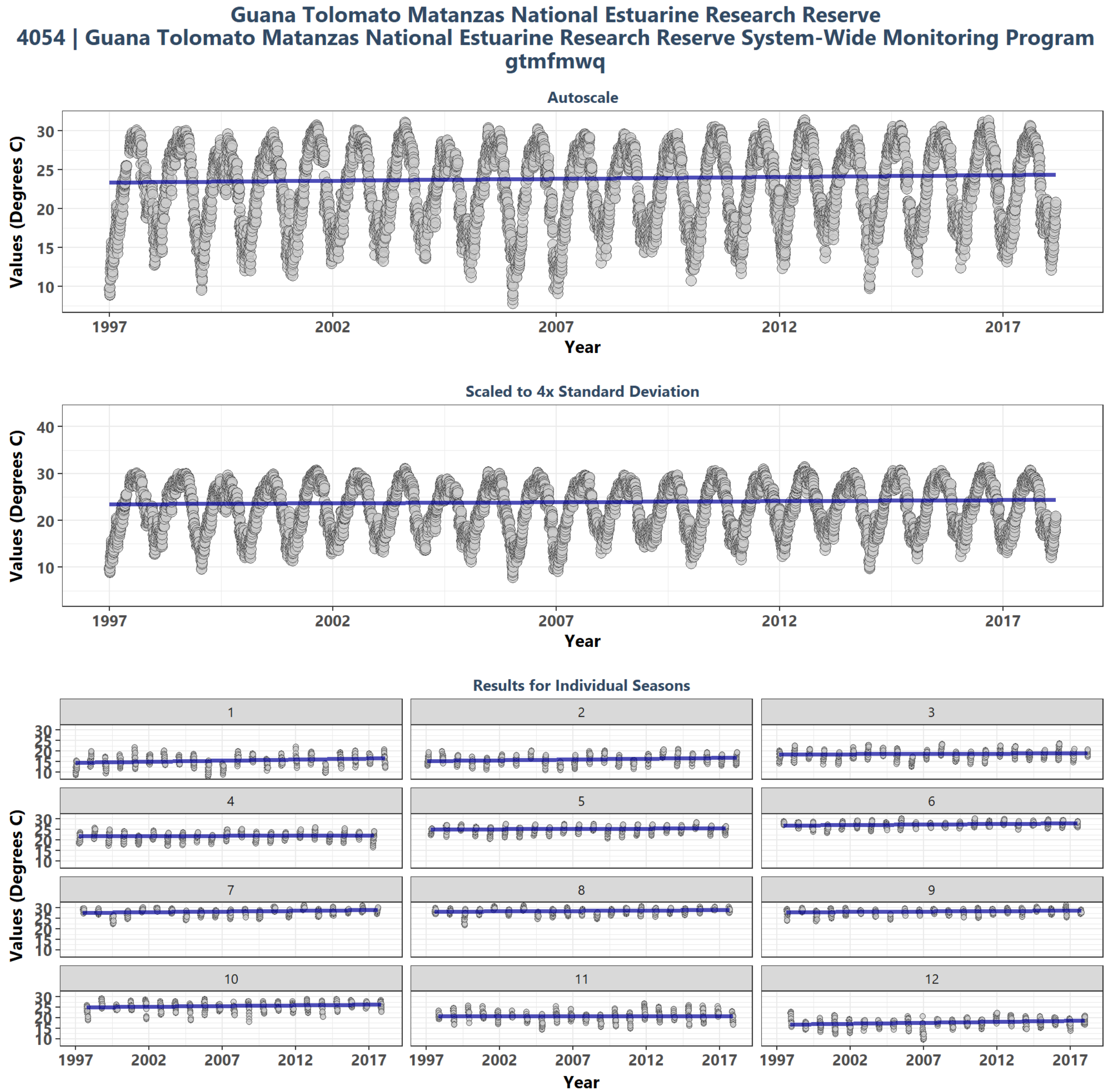
[a] $p < 0.00005$ appear as 0 due to rounding

Guana River Marsh Aquatic Preserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmpiwq



Season	N	Median	tau	Slope	Int.	z	p__z	chi__sq	p__chi__sq	Trend
All	7531	24.12	0.0941	0.0419	23.6668	12.3	0.0000	34.5	0.0003	1
1	661	14.79	0.1566	0.1135	13.6510	6.0	0.0000	NA	NA	1
2	605	16.57	0.1206	0.0772	15.7198	4.4	0.0000	NA	NA	1
3	652	19.49	0.0563	0.0375	19.1151	2.2	0.0312	NA	NA	1
4	608	23.08	0.0921	0.0460	22.6205	3.4	0.0007	NA	NA	1
5	605	26.46	0.0652	0.0274	26.1633	2.4	0.0163	NA	NA	1
6	630	28.93	0.0758	0.0229	28.7026	2.9	0.0044	NA	NA	1
7	634	30.03	0.1223	0.0307	29.7233	4.6	0.0000	NA	NA	1
8	634	30.20	0.0728	0.0192	30.0084	2.7	0.0060	NA	NA	1
9	611	28.81	0.1192	0.0440	28.3649	4.4	0.0000	NA	NA	1
10	651	25.42	0.1305	0.0712	24.7132	5.0	0.0000	NA	NA	1
11	612	20.14	-0.0171	-0.0115	20.2542	-0.6	0.5254	NA	NA	-1
12	628	17.19	0.1292	0.0943	16.2438	4.9	0.0000	NA	NA	1

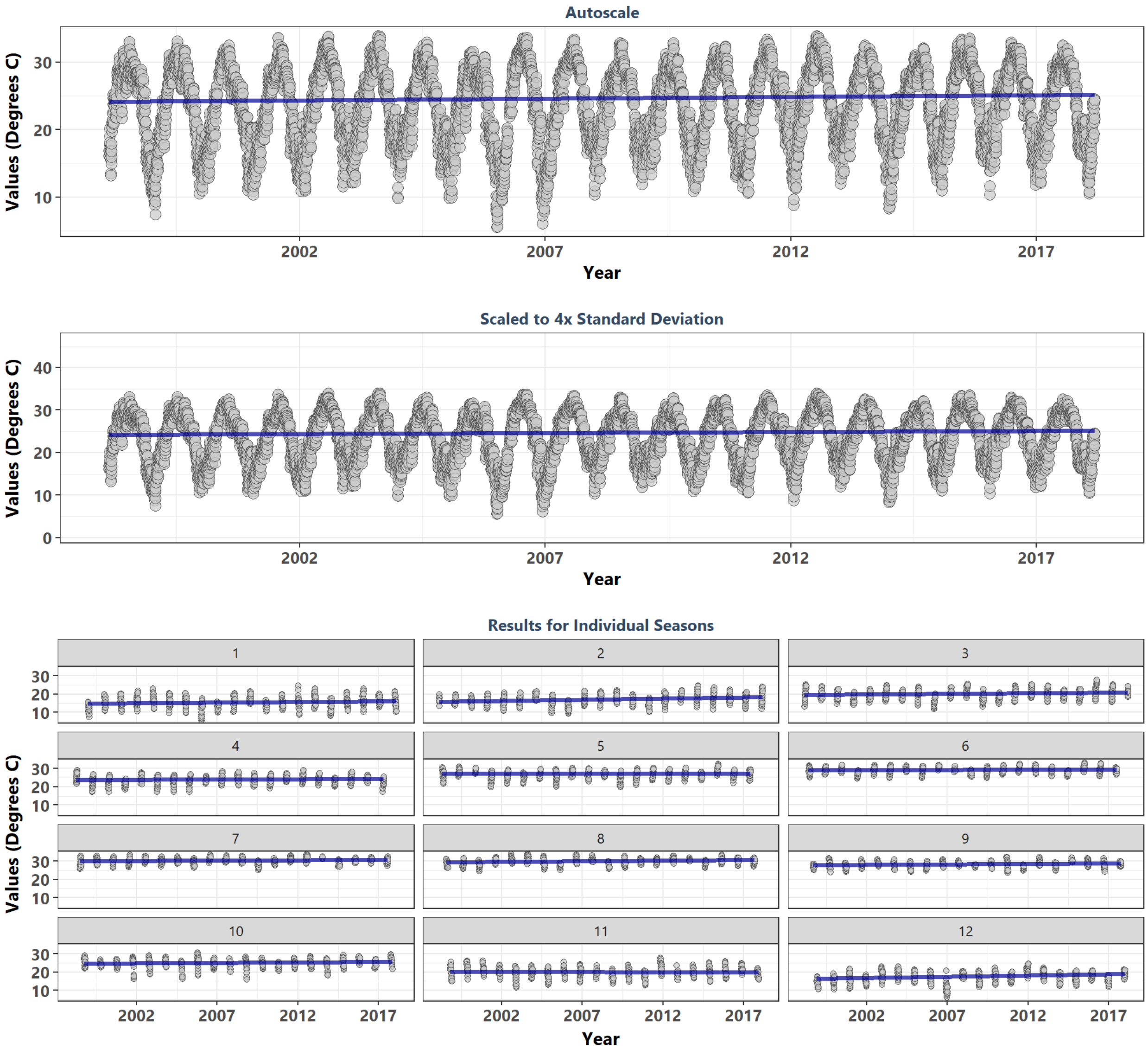
[a] $p < 0.00005$ appear as 0 due to rounding



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	7481	23.52	0.1097	0.0460	23.3967	14.2	0.0000	47.8	0	1
1	646	15.52	0.1521	0.0940	14.5759	5.8	0.0000	NA	NA	1
2	591	16.16	0.1402	0.0709	15.3811	5.1	0.0000	NA	NA	1
3	632	18.72	0.0485	0.0254	18.4441	1.8	0.0678	NA	NA	1
4	630	22.00	0.0450	0.0201	21.8030	1.7	0.0909	NA	NA	1
5	640	25.28	0.0759	0.0290	24.9905	2.9	0.0040	NA	NA	1
6	597	27.39	0.1582	0.0492	26.8932	5.8	0.0000	NA	NA	1
7	616	28.27	0.1524	0.0569	27.6995	5.7	0.0000	NA	NA	1
8	641	28.66	0.1218	0.0394	28.2688	4.6	0.0000	NA	NA	1
9	602	28.35	0.1389	0.0374	27.9721	5.1	0.0000	NA	NA	1
10	647	25.61	0.1212	0.0573	25.0406	4.6	0.0000	NA	NA	1
11	614	20.71	-0.0070	-0.0039	20.7509	-0.3	0.7939	NA	NA	-1
12	625	17.56	0.1726	0.0913	16.6500	6.5	0.0000	NA	NA	1

[a] $p < 0.00005$ appear as 0 due to rounding

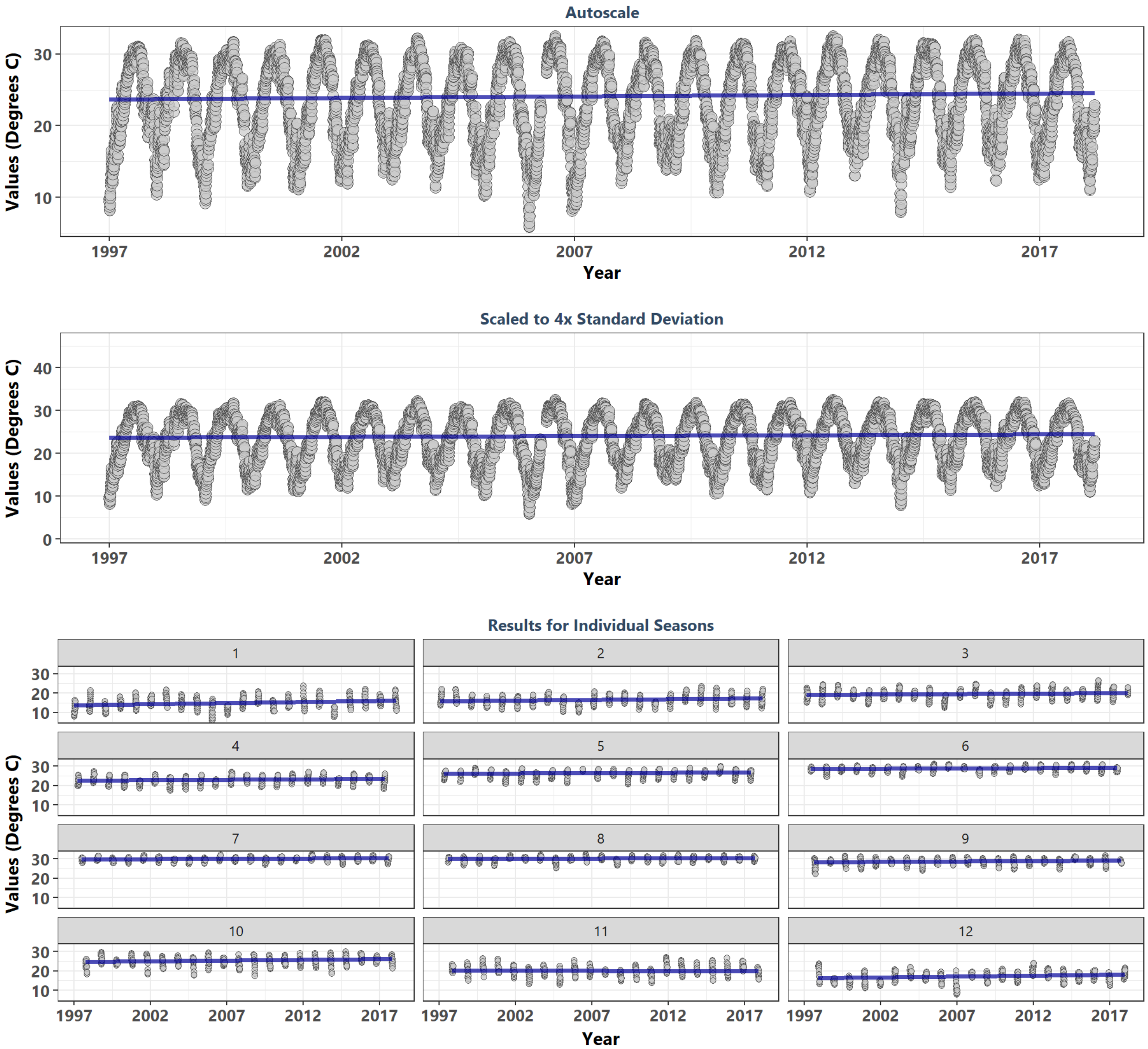
Guana Tolomato Matanzas National Estuarine Research Reserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmpcwq



Season	N	Median	tau	Slope	Int.	z	p__z	chi__sq	p__chi__sq	Trend
All	7130	24.42	0.0793	0.0478	24.1288	10.0	0.0000	59.6	0	1
1	604	15.42	0.0778	0.0663	14.6912	2.9	0.0042	NA	NA	1
2	567	17.11	0.1696	0.1298	15.6842	6.0	0.0000	NA	NA	1
3	627	20.26	0.0861	0.0676	19.5170	3.2	0.0013	NA	NA	1
4	588	24.07	0.0480	0.0283	23.7585	1.7	0.0814	NA	NA	1
5	608	27.32	-0.0022	-0.0010	27.3271	-0.1	0.9360	NA	NA	-1
6	570	29.24	0.0136	0.0062	29.1729	0.5	0.6258	NA	NA	1
7	578	30.46	0.0749	0.0340	30.1222	2.7	0.0070	NA	NA	1
8	578	29.93	0.1361	0.0672	29.2589	4.9	0.0000	NA	NA	1
9	588	28.22	0.1373	0.0588	27.6343	5.0	0.0000	NA	NA	1
10	620	25.02	0.0854	0.0498	24.4991	3.2	0.0015	NA	NA	1
11	586	20.10	-0.0318	-0.0221	20.3172	-1.2	0.2492	NA	NA	-1
12	616	17.40	0.1571	0.1272	16.1313	5.8	0.0000	NA	NA	1

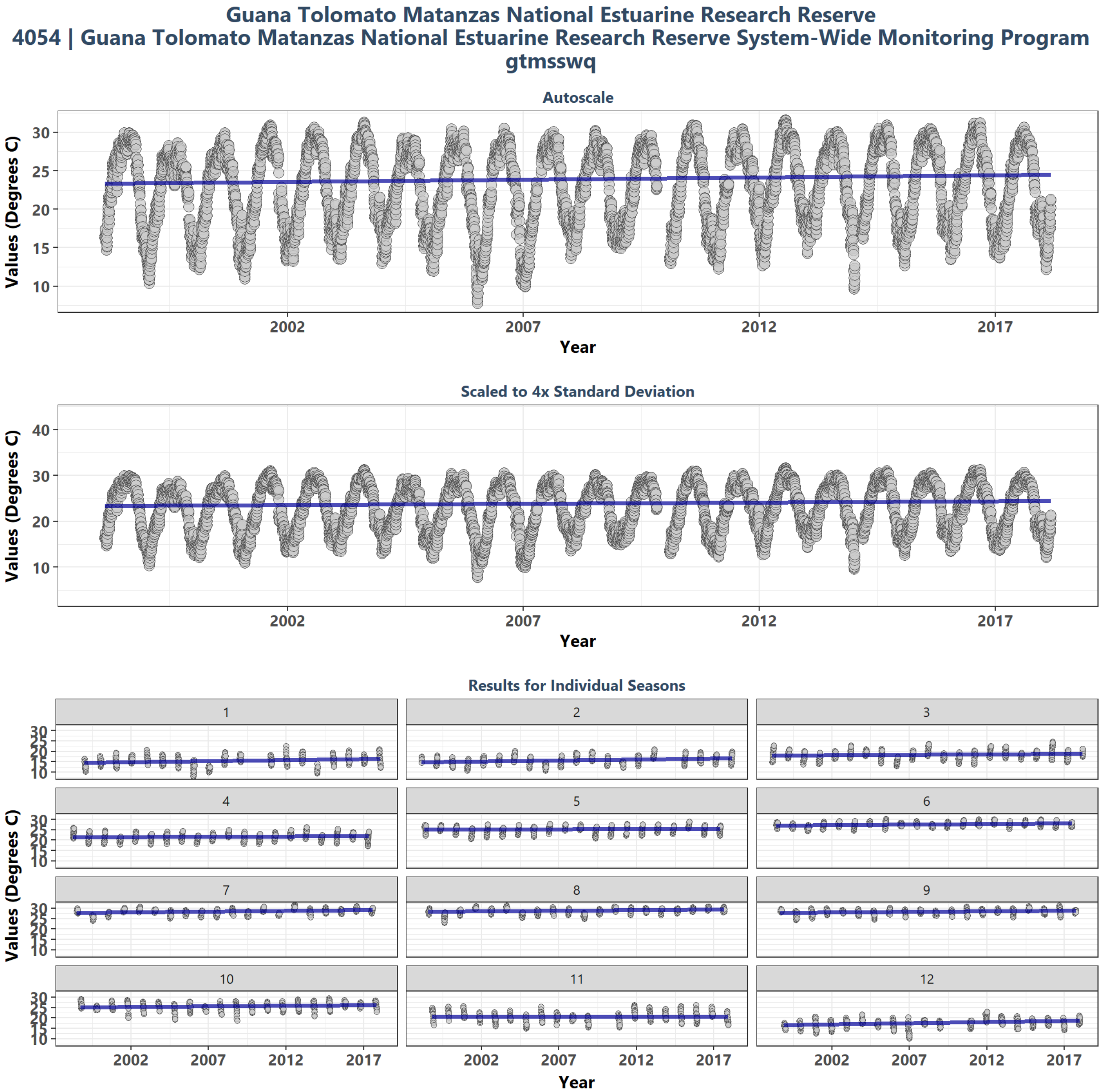
[a] $p < 0.00005$ appear as 0 due to rounding

Guana Tolomato Matanzas National Estuarine Research Reserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmpiwq



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	7531	24.12	0.0941	0.0419	23.6668	12.3	0.0000	34.5	0.0003	1
1	661	14.79	0.1566	0.1135	13.6510	6.0	0.0000	NA	NA	1
2	605	16.57	0.1206	0.0772	15.7198	4.4	0.0000	NA	NA	1
3	652	19.49	0.0563	0.0375	19.1151	2.2	0.0312	NA	NA	1
4	608	23.08	0.0921	0.0460	22.6205	3.4	0.0007	NA	NA	1
5	605	26.46	0.0652	0.0274	26.1633	2.4	0.0163	NA	NA	1
6	630	28.93	0.0758	0.0229	28.7026	2.9	0.0044	NA	NA	1
7	634	30.03	0.1223	0.0307	29.7233	4.6	0.0000	NA	NA	1
8	634	30.20	0.0728	0.0192	30.0084	2.7	0.0060	NA	NA	1
9	611	28.81	0.1192	0.0440	28.3649	4.4	0.0000	NA	NA	1
10	651	25.42	0.1305	0.0712	24.7132	5.0	0.0000	NA	NA	1
11	612	20.14	-0.0171	-0.0115	20.2542	-0.6	0.5254	NA	NA	-1
12	628	17.19	0.1292	0.0943	16.2438	4.9	0.0000	NA	NA	1

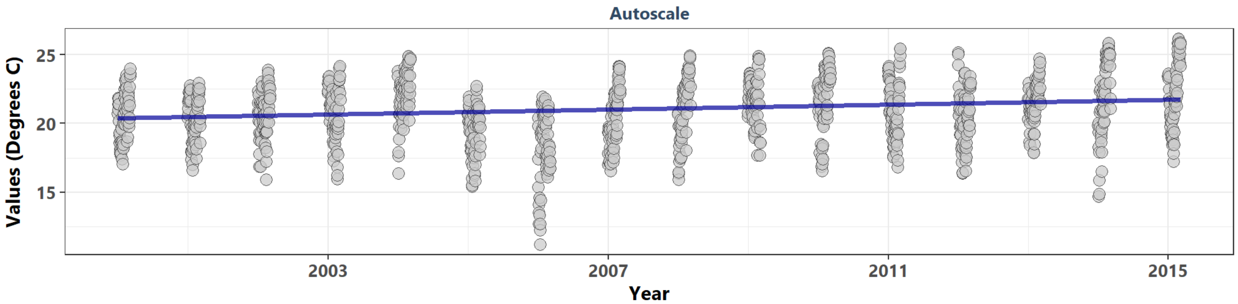
[a] $p < 0.00005$ appear as 0 due to rounding

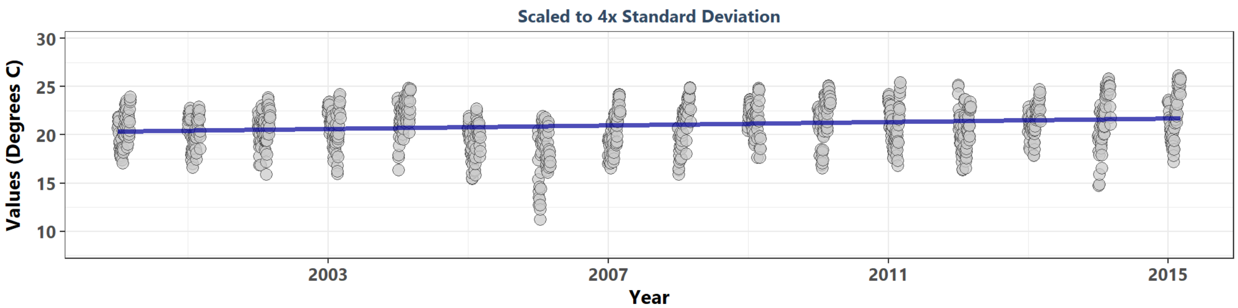


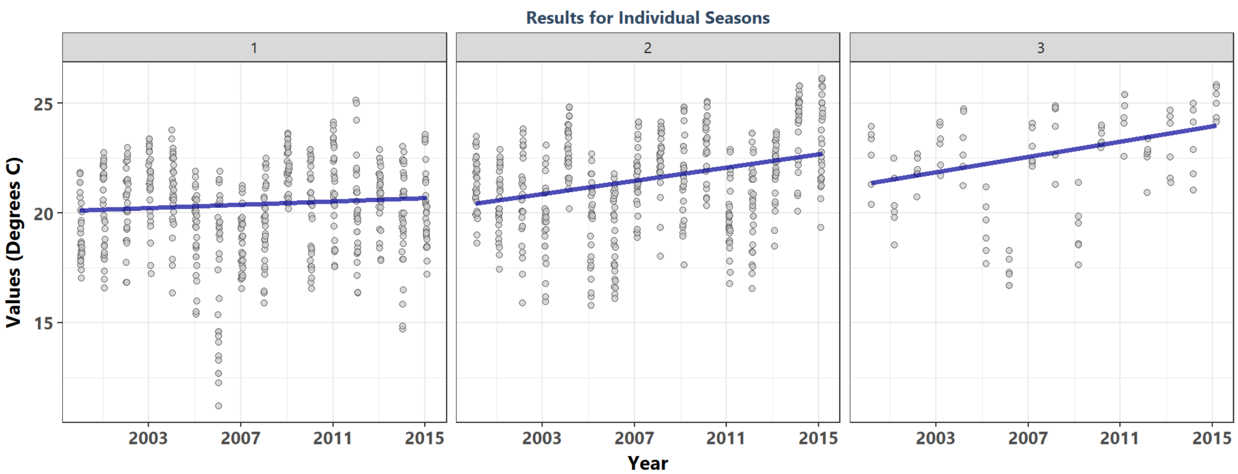
Season	N	Median	tau	Slope	Int.	z	p__z	chi__sq	p__chi__sq	Trend
All	6998	23.80	0.1310	0.0559	23.3127	16.4	0.0000	81.6	0	1
1	569	15.46	0.1661	0.1050	14.3071	5.9	0.0000	NA	NA	1
2	540	15.72	0.1876	0.1040	14.5809	6.5	0.0000	NA	NA	1
3	618	18.40	0.0534	0.0301	18.0956	2.0	0.0469	NA	NA	1
4	589	21.82	0.0541	0.0260	21.5323	2.0	0.0492	NA	NA	1
5	580	25.39	0.0424	0.0167	25.2219	1.5	0.1267	NA	NA	1
6	580	27.68	0.2053	0.0594	27.0292	7.4	0.0000	NA	NA	1
7	594	28.54	0.1933	0.0710	27.7578	7.1	0.0000	NA	NA	1
8	596	28.87	0.1849	0.0654	28.2170	6.8	0.0000	NA	NA	1
9	594	28.42	0.1706	0.0481	27.8870	6.2	0.0000	NA	NA	1
10	612	25.63	0.1050	0.0489	25.0932	3.9	0.0001	NA	NA	1
11	542	20.62	-0.0028	-0.0018	20.6386	-0.1	0.9231	NA	NA	-1
12	584	17.46	0.2115	0.1045	16.4119	7.7	0.0000	NA	NA	1

[a] p < 0.00005 appear as 0 due to rounding

Loxahatchee River-Lake Worth Creek Aquatic Preserve
7 | National Water Information System
265906080093500



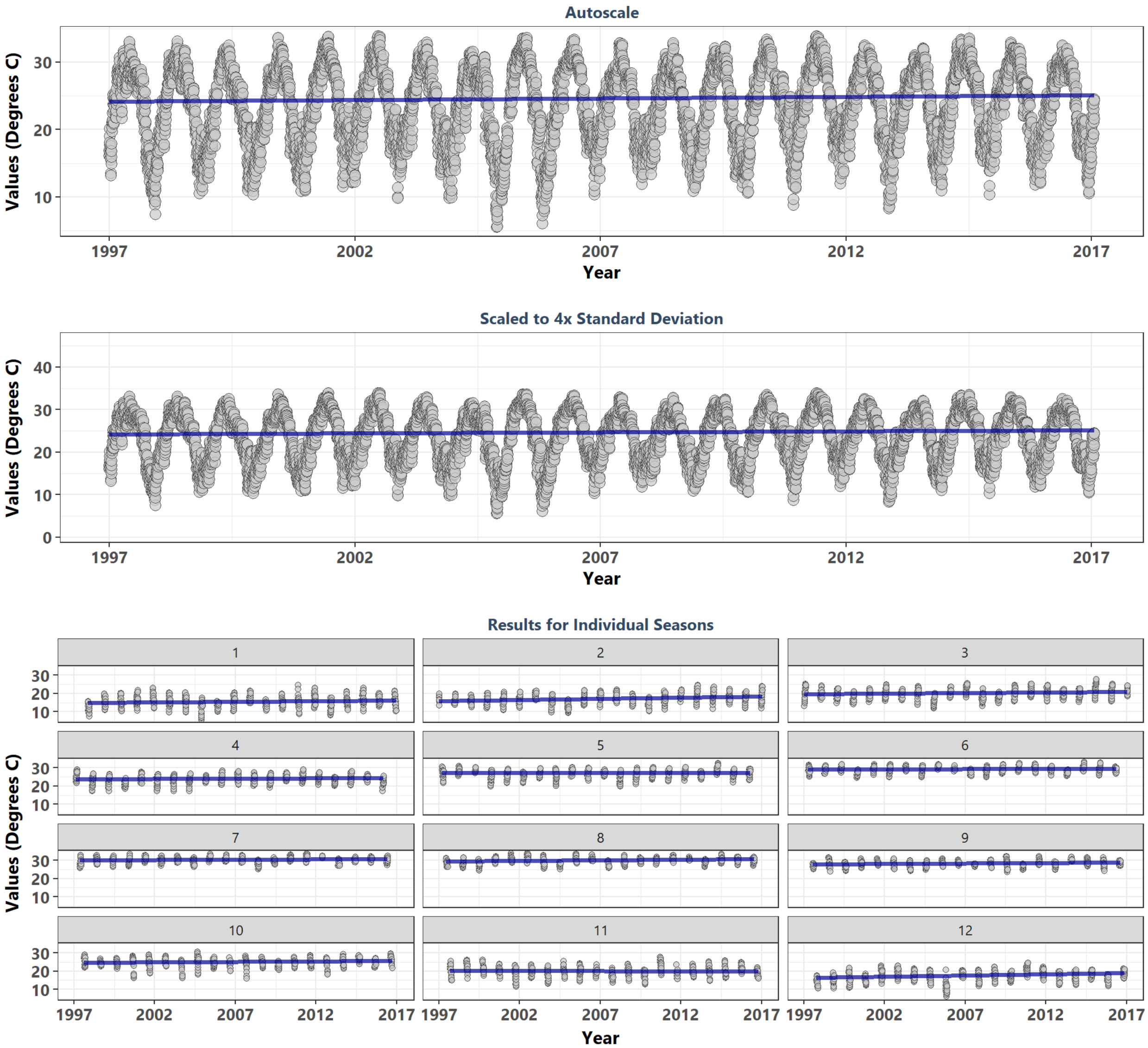




Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	1044	21.00	0.1409	0.0900	20.2750	6.0	0.0000	11.2	0.0036	1
1	496	20.40	0.0557	0.0375	20.0812	1.9	0.0632	NA	NA	1
2	452	21.55	0.2069	0.1500	20.2750	6.6	0.0000	NA	NA	1
3	96	22.65	0.2704	0.1739	21.1722	3.9	0.0001	NA	NA	1

[a] $p < 0.00005$ appear as 0 due to rounding

Pellicer Creek Aquatic Preserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmpcwq



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	7130	24.42	0.0793	0.0478	24.1679	10.0	0.0000	59.6	0	1
1	604	15.42	0.0778	0.0663	14.7575	2.9	0.0042	NA	NA	1
2	567	17.11	0.1696	0.1298	15.8140	6.0	0.0000	NA	NA	1
3	627	20.26	0.0861	0.0676	19.5846	3.2	0.0013	NA	NA	1
4	588	24.07	0.0480	0.0283	23.7868	1.7	0.0814	NA	NA	1
5	608	27.32	-0.0022	-0.0010	27.3260	-0.1	0.9360	NA	NA	-1
6	570	29.24	0.0136	0.0062	29.1792	0.5	0.6258	NA	NA	1
7	578	30.46	0.0749	0.0340	30.1562	2.7	0.0070	NA	NA	1
8	578	29.93	0.1361	0.0672	29.3260	4.9	0.0000	NA	NA	1
9	588	28.22	0.1373	0.0588	27.6931	5.0	0.0000	NA	NA	1
10	620	25.02	0.0854	0.0498	24.5490	3.2	0.0015	NA	NA	1
11	586	20.10	-0.0318	-0.0221	20.2951	-1.2	0.2492	NA	NA	-1
12	616	17.40	0.1571	0.1272	16.2586	5.8	0.0000	NA	NA	1

[a] $p < 0.00005$ appear as 0 due to rounding

Appendix IV: Monitoring Location Summary Box Plots

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

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

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

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

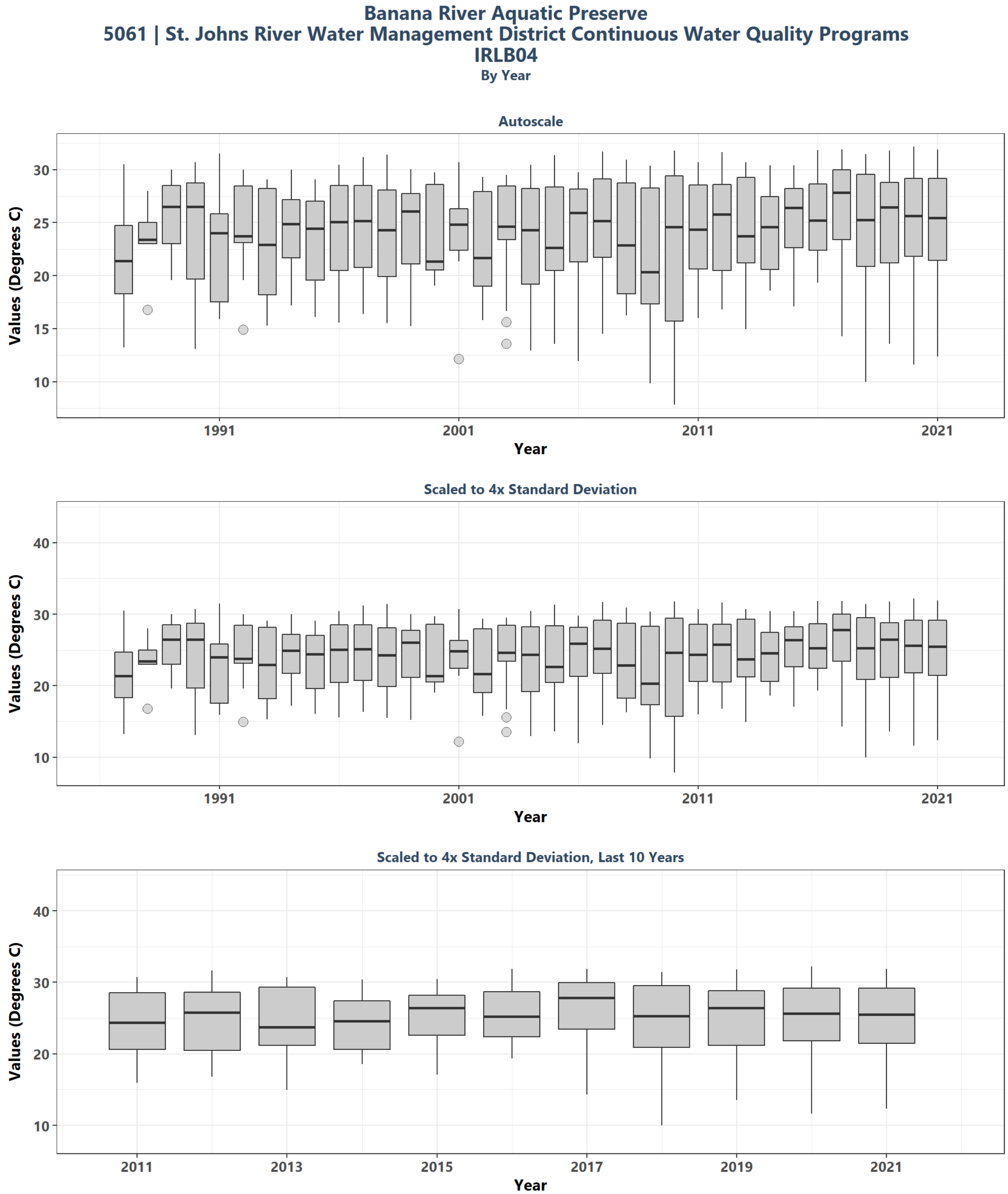
```
if(n==0){
   print("There are no monitoring locations that qualify.")
} else {
   for (i in 1:n) {
      year_lower <- min(data$Year[data$Use_In_Analysis==TRUE &
                                  data$MonitoringID==Mon_IDs[i]])
      year_upper <- max(data$Year[data$Use_In_Analysis==TRUE &
                                  data$MonitoringID==Mon_IDs[i]])
      min_RV <- min(data$ResultValue[data$Use_In_Analysis==TRUE &
                                     data$MonitoringID==Mon_IDs[i]])
      mn_RV <- mean(data$ResultValue[data$Use_In_Analysis==TRUE &
                                     data$MonitoringID==Mon_IDs[i] &
                                     data$ResultValue <
                                     quantile(data$ResultValue, 0.98)])
      sd_RV <- sd(data$ResultValue[data$Use_In_Analysis==TRUE &
                                   data$MonitoringID==Mon_IDs[i] &
                                   data$ResultValue <
                                   quantile(data$ResultValue, 0.98)])
      x_scale <- ifelse(year_upper - year_lower > 30, 10, 5)
      y_scale <- mn_RV + 4 * sd_RV
      MA_name <- KT.Stats$ManagedAreaName[KT.Stats$MonitoringID==Mon_IDs[i]]
      Mon_name <- paste0(KT.Stats$ProgramID[KT.Stats$MonitoringID==Mon_IDs[i]],
         " | ", KT.Stats$ProgramName[KT.Stats$MonitoringID==Mon_IDs[i]], "\n",
         KT.Stats$ProgramLocationID[KT.Stats$MonitoringID==Mon_IDs[i]])

      ##Year plots
      p1 <- ggplot(data=data[data$Use_In_Analysis==TRUE &
                             data$MonitoringID==Mon_IDs[i], ],
```

```
            aes(x=Year, y=ResultValue, group=Year)) +
  geom_boxplot(color="#333333", fill="#cccccc", outlier.shape=21,
               outlier.size=3, outlier.color="#333333",
               outlier.fill="#cccccc", outlier.alpha=0.75) +
  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))) +
  plot_theme

p2 <- ggplot(data=data[data$Use_In_Analysis==TRUE &
                            data$MonitoringID==Mon_IDs[i], ],
             aes(x=Year, y=ResultValue, group=Year)) +
  geom_boxplot(color="#333333", fill="#cccccc", outlier.shape=21,
               outlier.size=3, outlier.color="#333333",
               outlier.fill="#cccccc", outlier.alpha=0.75) +
  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))) +
  plot_theme

p3 <- ggplot(data=data[data$Use_In_Analysis==TRUE &
                            data$MonitoringID==Mon_IDs[i] &
                            data$Year>=year_upper-10, ],
             aes(x=Year, y=ResultValue, group=Year)) +
  geom_boxplot(color="#333333", fill="#cccccc", outlier.shape=21,
               outlier.size=3, outlier.color="#333333",
               outlier.fill="#cccccc", outlier.alpha=0.75) +
  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))) +
  plot_theme

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

p0 <- ggplot() + labs(title=paste0(MA_name, "\n", Mon_name),
                     subtitle="By Year") +
  plot_theme + theme(panel.border=element_blank(),
                     panel.grid.major=element_blank(),
                     panel.grid.minor=element_blank(),
                     axis.line=element_blank())


## Year & Month Plots
p4 <- ggplot(data=data[data$Use_In_Analysis==TRUE &
                            data$MonitoringID==Mon_IDs[i], ],
             aes(x=YearMonthDec, y=ResultValue,
```

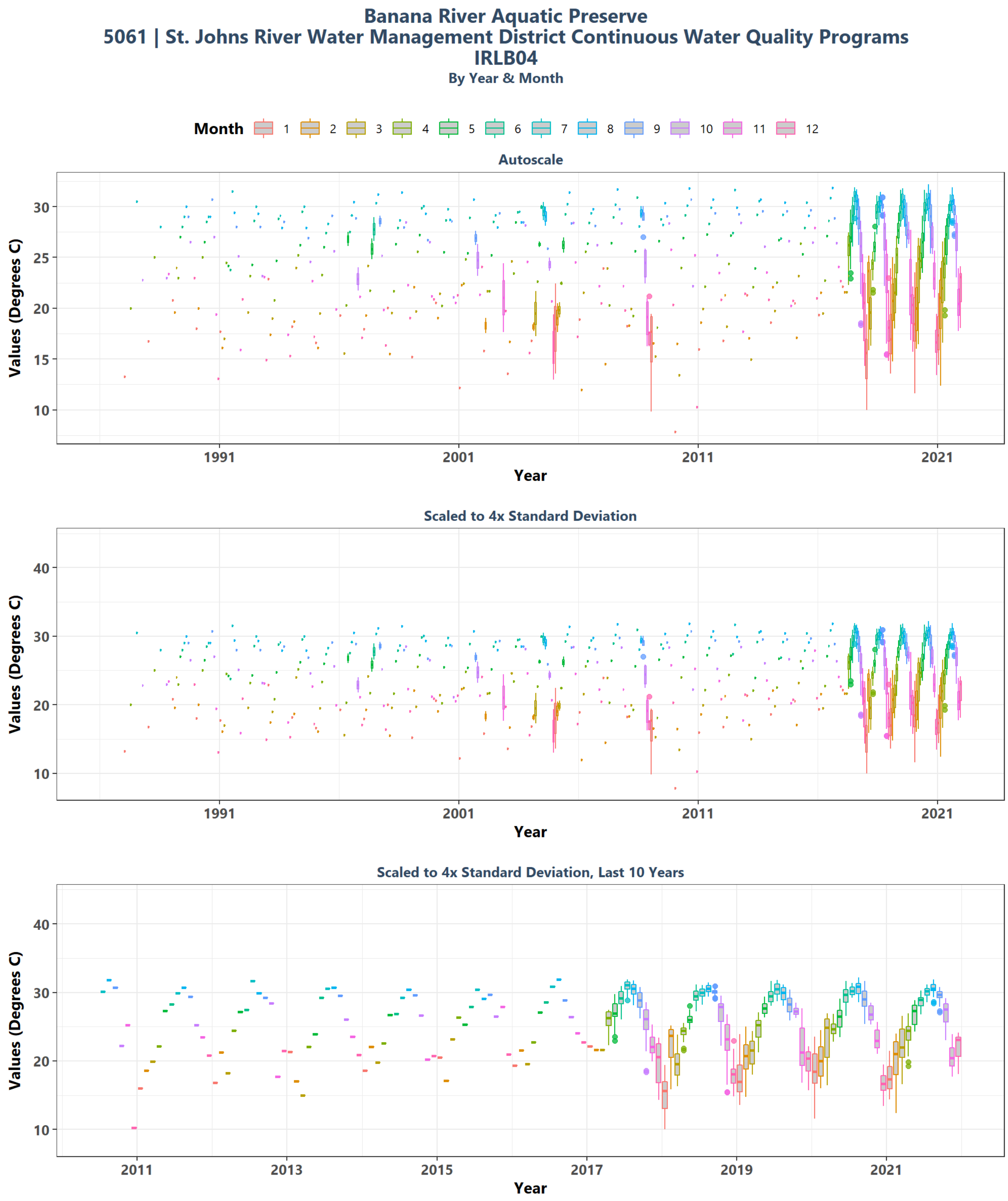
```
                group=YearMonth, color=as.factor(Month))) +
    geom_boxplot(fill="#cccccc", outlier.size=1.5, outlier.alpha=0.75) +
    labs(subtitle="Autoscale",
         x="Year", y=paste0("Values (", unit, ")"), color="Month") +
    scale_x_continuous(limits=c(year_lower - 1, year_upper + 1),
                       breaks=rev(seq(year_upper,
                                      year_lower, -x_scale))) +
    plot_theme +
    theme(legend.position="none")

  p5 <- ggplot(data=data[data$Use_In_Analysis==TRUE &
                            data$MonitoringID==Mon_IDs[i], ],
              aes(x=YearMonthDec, y=ResultValue,
                  group=YearMonth, color=as.factor(Month))) +
    geom_boxplot(fill="#cccccc", outlier.size=1.5, outlier.alpha=0.75) +
    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))) +
    plot_theme +
    theme(legend.position="top", legend.box="horizontal") +
    guides(color=guide_legend(nrow=1))

  p6 <- ggplot(data=data[data$Use_In_Analysis==TRUE &
                            data$MonitoringID==Mon_IDs[i], ],
              aes(x=YearMonthDec, y=ResultValue,
                  group=YearMonth, color=as.factor(Month))) +
    geom_boxplot(fill="#cccccc", outlier.size=1.5, outlier.alpha=0.75) +
    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))) +
    plot_theme +
    theme(legend.position="none")

  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(MA_name, "\n", Mon_name),
                         subtitle="By Year & Month") + plot_theme +
    theme(panel.border=element_blank(),
          panel.grid.major=element_blank(),
          panel.grid.minor=element_blank(), axis.line=element_blank())

  ## Month Plots
  p7 <- ggplot(data=data[data$Use_In_Analysis==TRUE &
                            data$MonitoringID==Mon_IDs[i], ],
              aes(x=Month, y=ResultValue,
                  group=Month, fill=as.factor(Month))) +
```

```
    geom_boxplot(color="#333333", outlier.shape=21, outlier.size=3,
                 outlier.color="#333333", outlier.alpha=0.75) +
    labs(subtitle="Autoscale",
         x="Month", y=paste0("Values (", unit, ")"), fill="Month") +
    scale_x_continuous(limits=c(0, 13), breaks=seq(3, 12, 3)) +
    plot_theme +
    theme(legend.position="none")

  p8 <- ggplot(data=data[data$Use_In_Analysis==TRUE &
                            data$MonitoringID==Mon_IDs[i], ],
              aes(x=Month, y=ResultValue,
                  group=Month, fill=as.factor(Month))) +
    geom_boxplot(color="#333333", outlier.shape=21, outlier.size=3,
                 outlier.color="#333333", outlier.alpha=0.75) +
    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)) +
    plot_theme +
    theme(legend.position="top", legend.box="horizontal") +
    guides(fill=guide_legend(nrow=1))

  p9 <- ggplot(data=data[data$Use_In_Analysis==TRUE &
                            data$MonitoringID==Mon_IDs[i] &
                            data$Year >= year_upper - 10, ],
              aes(x=Month, y=ResultValue,
                  group=Month, fill=as.factor(Month))) +
    geom_boxplot(color="#333333", outlier.shape=21, outlier.size=3,
                 outlier.color="#333333", outlier.alpha=0.75) +
    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)) +
    plot_theme +
    theme(legend.position="none")

  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(MA_name, "\n", Mon_name),
                          subtitle="By Month") + plot_theme +
    theme(panel.border=element_blank(),
          panel.grid.major=element_blank(),
          panel.grid.minor=element_blank(), axis.line=element_blank())

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

  rm(plot_data)
  rm(p1, p2, p3, p4, p5, p6, p7, p8, p9, p0, p00, p000, leg1, leg2,
    Yset, YMset, Mset)
```

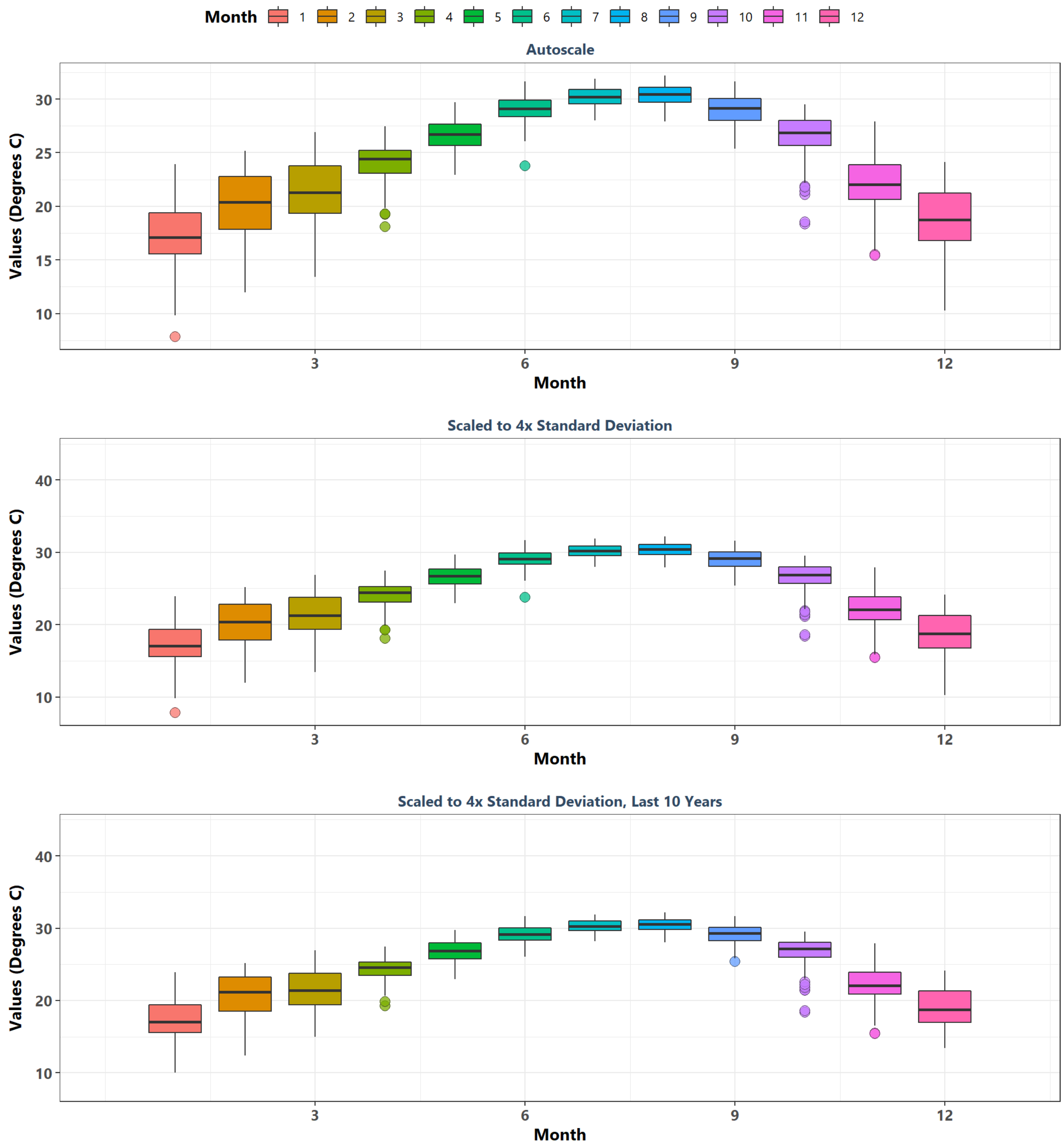
```
    }
}
```



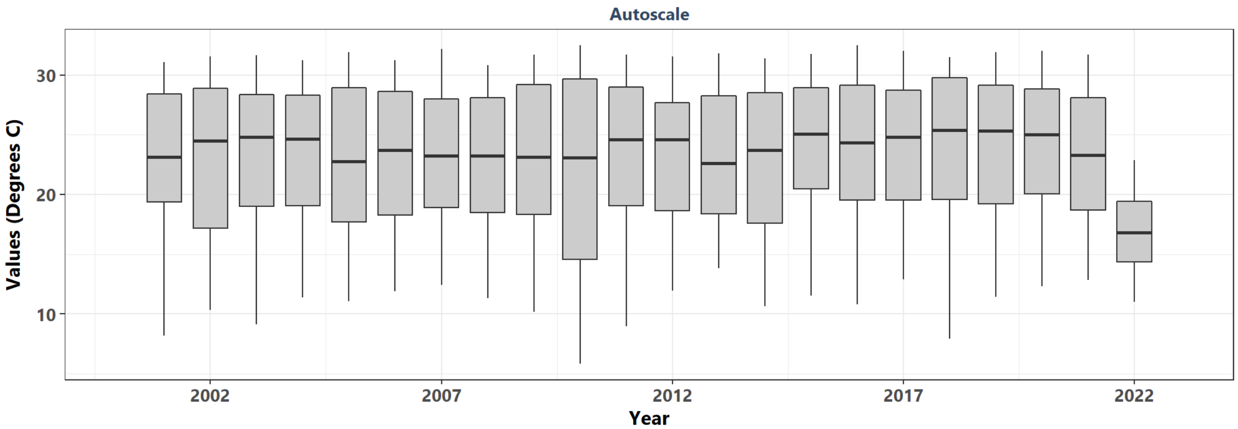
Banana River Aquatic Preserve
5061 | St. Johns River Water Management District Continuous Water Quality Programs
IRLB04
By Year
Autoscale
Values (Degrees C)
10
15
20
25
30
1991
2001
2011
2021
Year
Scaled to 4x Standard Deviation
Values (Degrees C)
10
20
30
40
1991
2001
2011
2021
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (Degrees C)
10
20
30
40
2011
2013
2015
2017
2019
2021
Year

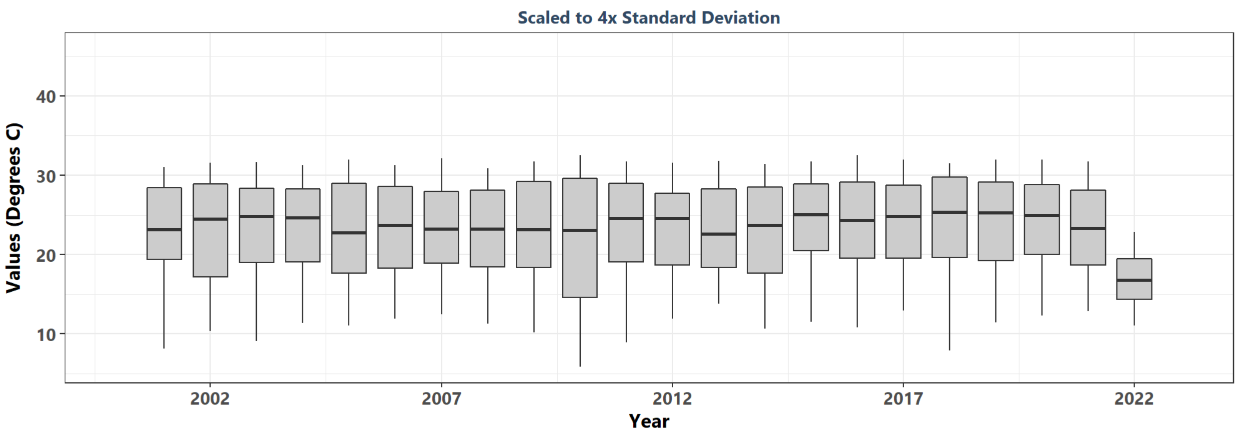

Banana River Aquatic Preserve
5061 | St. Johns River Water Management District Continuous Water Quality Programs
IRLB04
By Year & Month
Month
1
2
3
4
5
6
7
8
9
10
11
12
Autoscale
Values (Degrees C)
10
15
20
25
30
1991
2001
2011
2021
Year
Scaled to 4x Standard Deviation
Values (Degrees C)
10
20
30
40
1991
2001
2011
2021
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (Degrees C)
10
20
30
40
2011
2013
2015
2017
2019
2021
Year

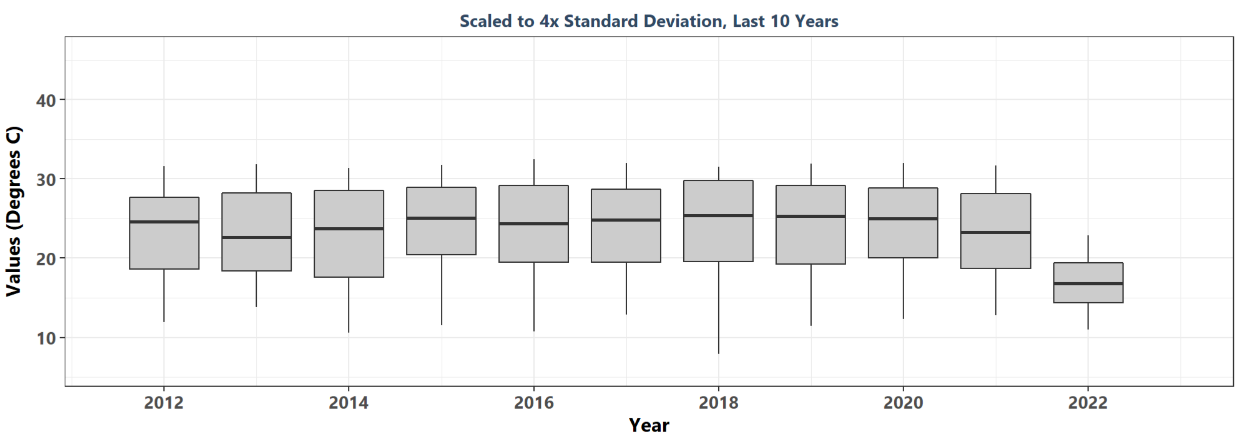
Banana River Aquatic Preserve
5061 | St. Johns River Water Management District Continuous Water Quality Programs
IRLB04
By Month



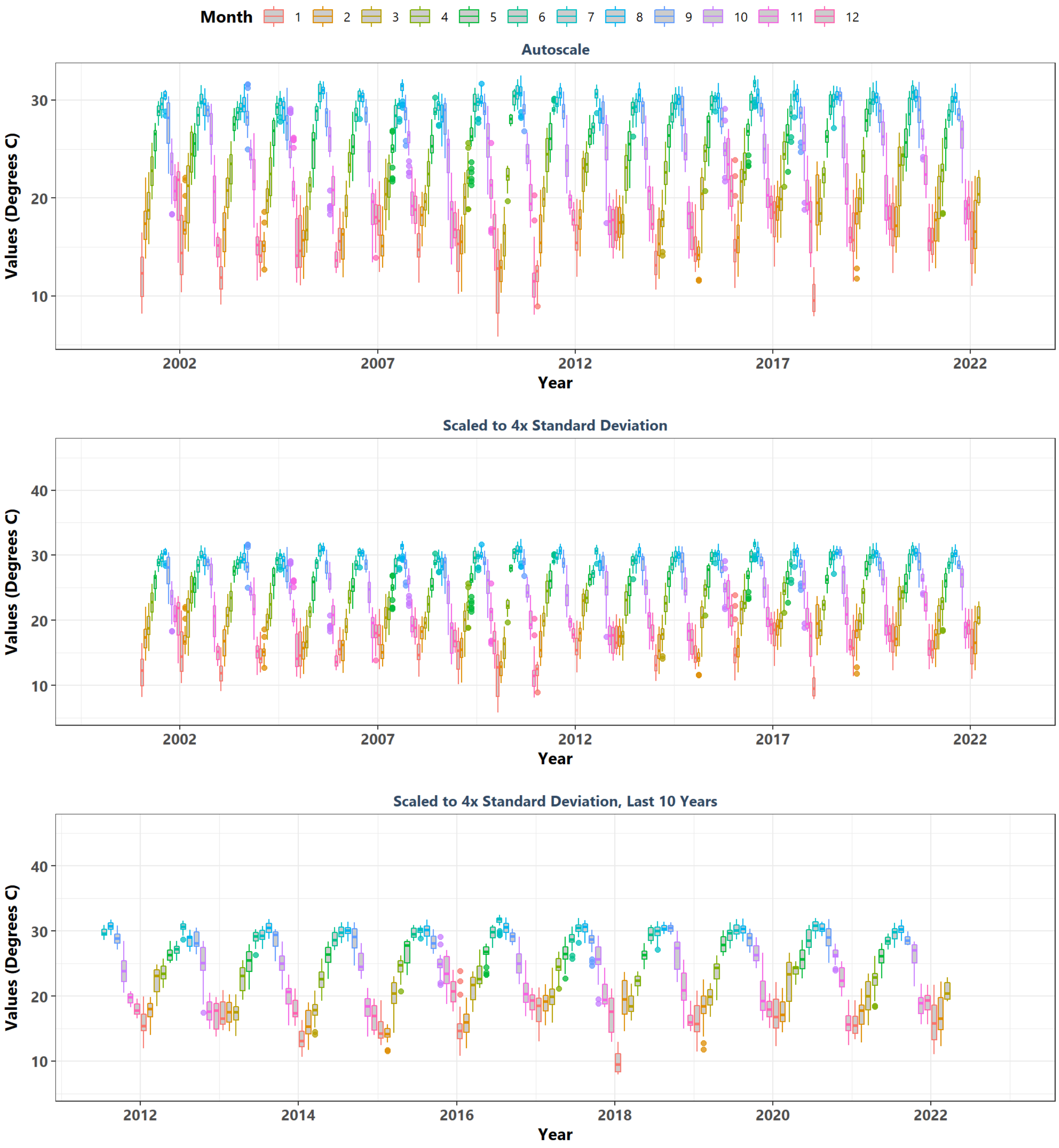
Guana River Marsh Aquatic Preserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmpiwq
By Year



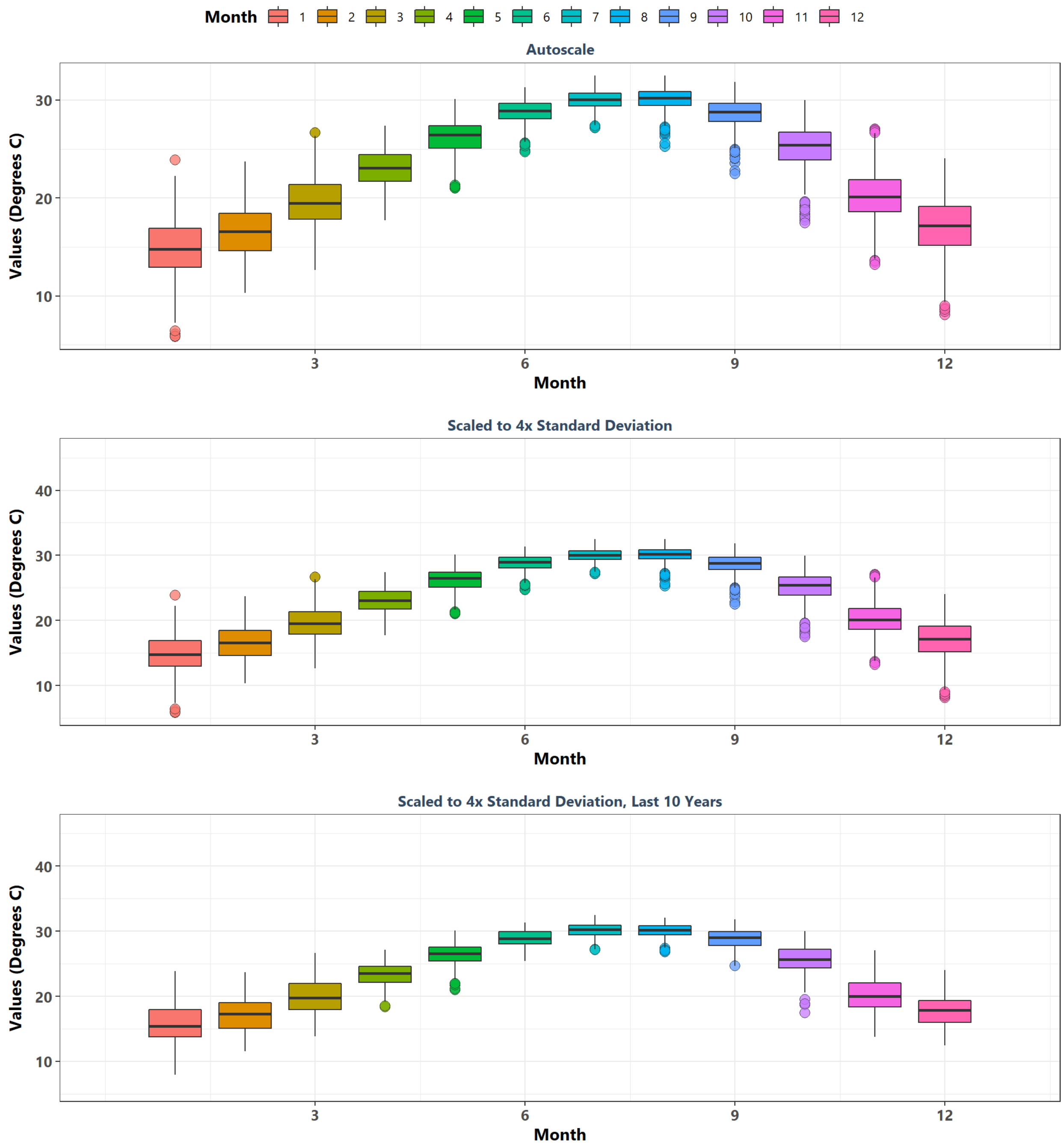




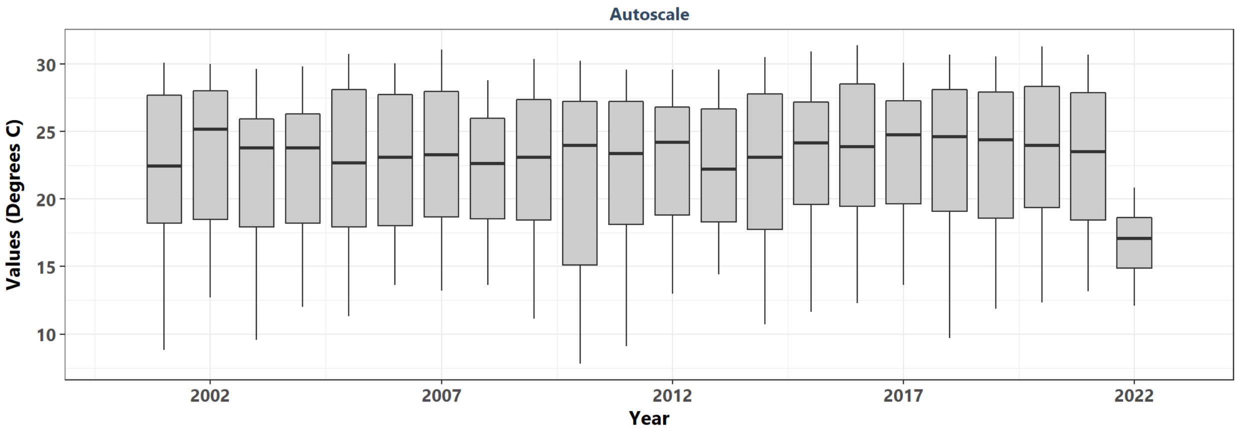
Guana River Marsh Aquatic Preserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmpiwq
By Year & Month

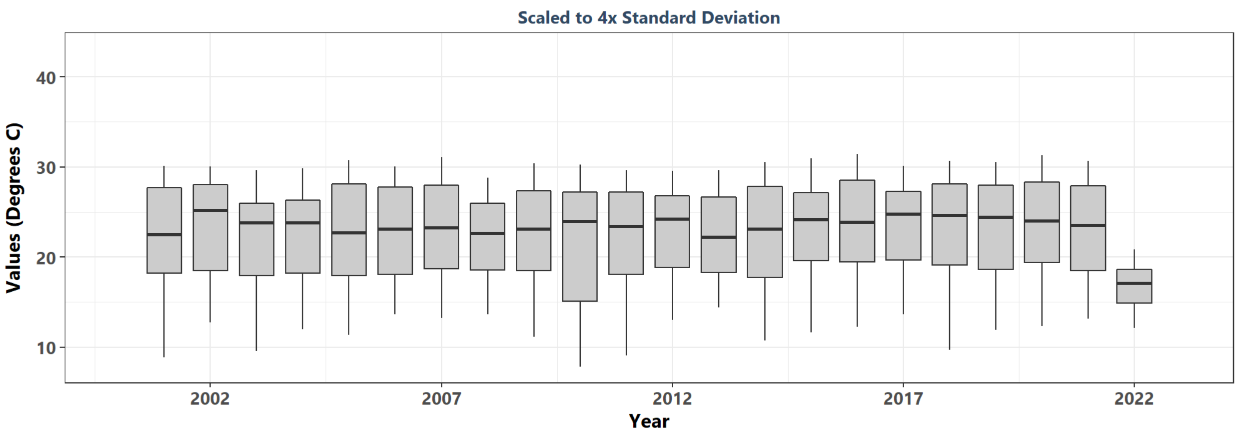


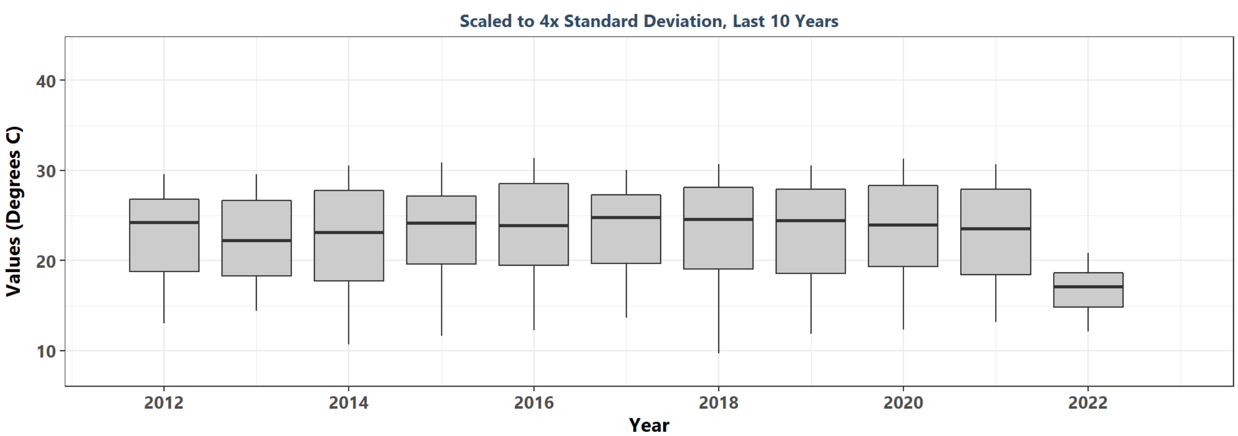
Guana River Marsh Aquatic Preserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmpiwq
By Month



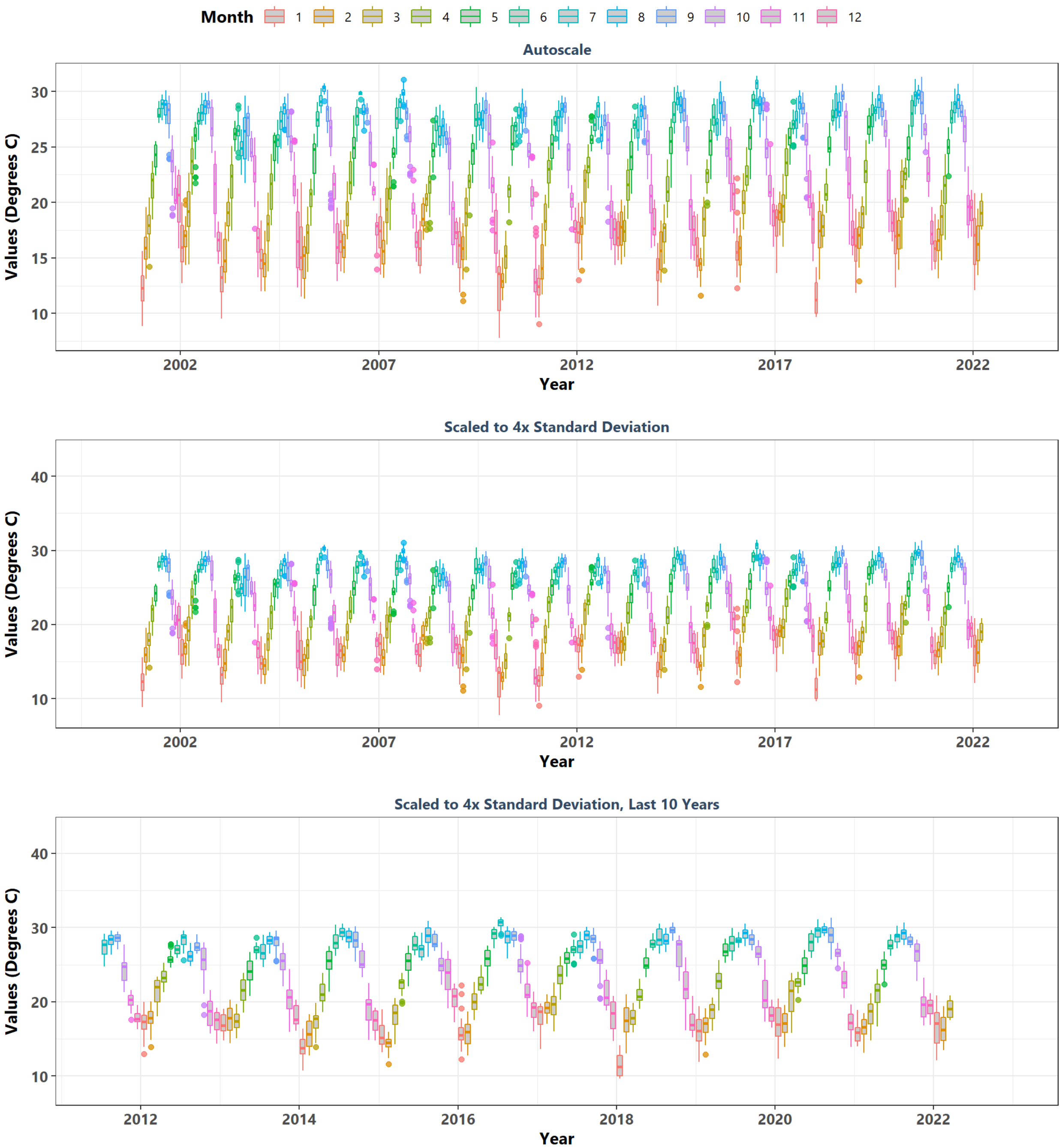
Guana Tolomato Matanzas National Estuarine Research Reserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmfmwq
By Year



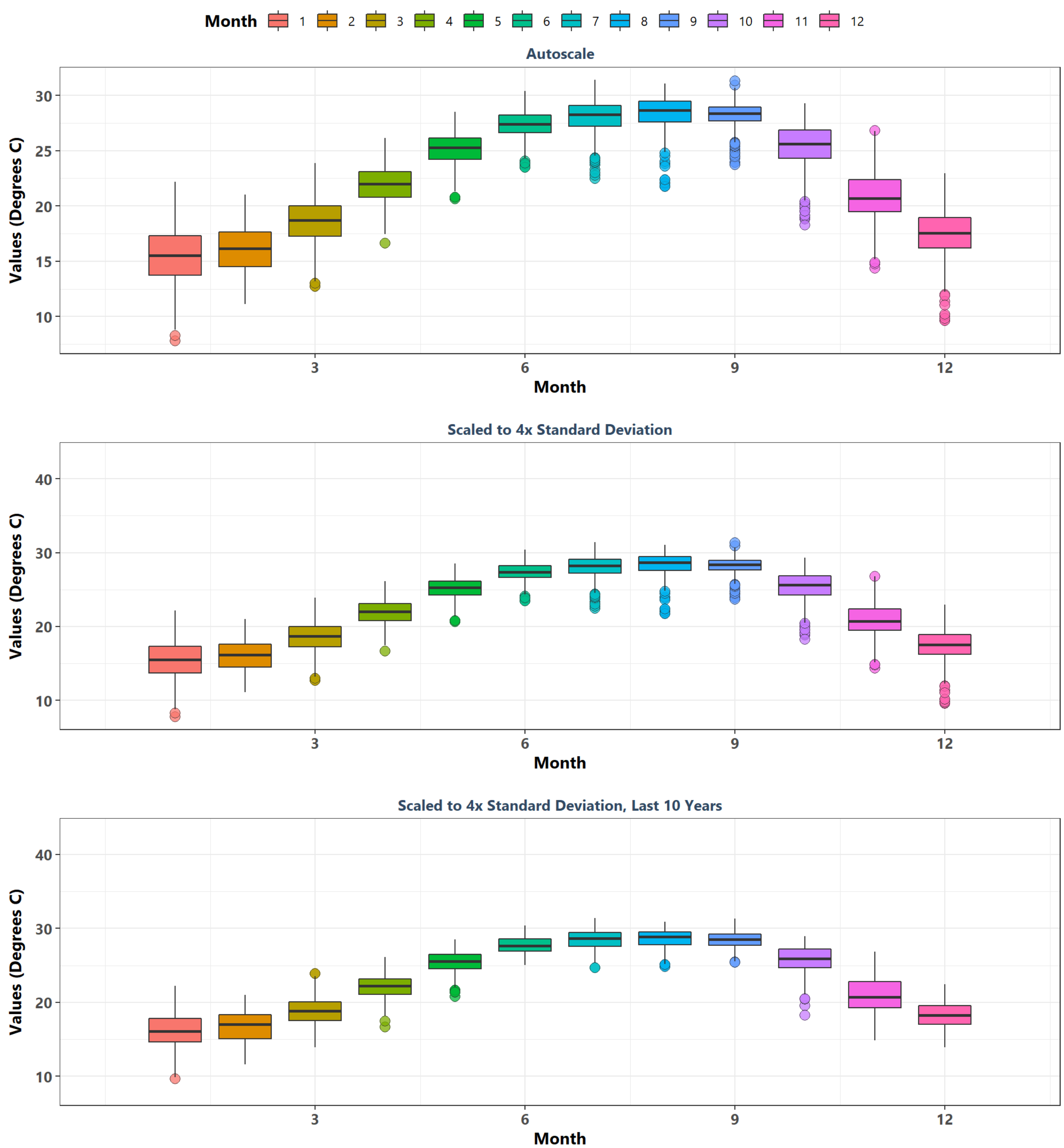




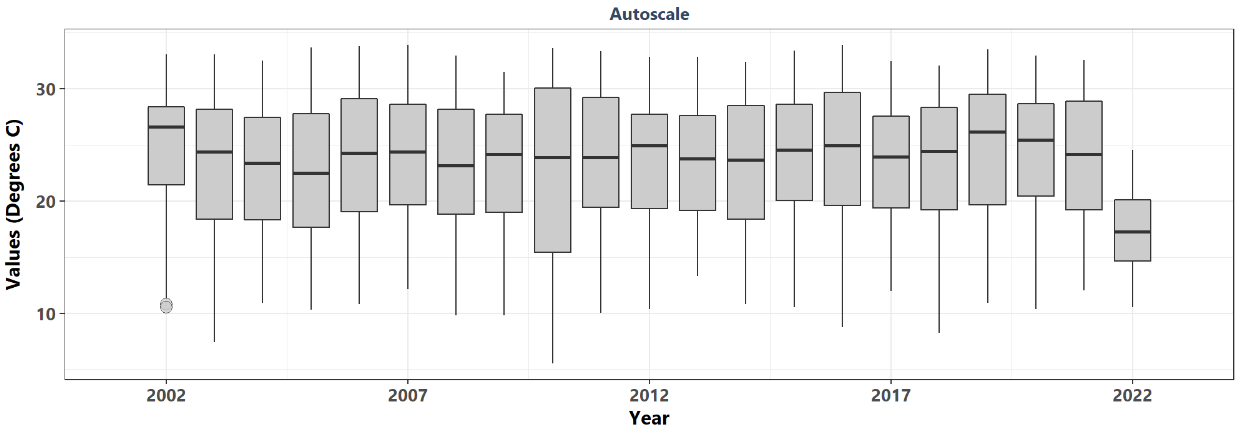
Guana Tolomato Matanzas National Estuarine Research Reserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmfmwq
By Year & Month

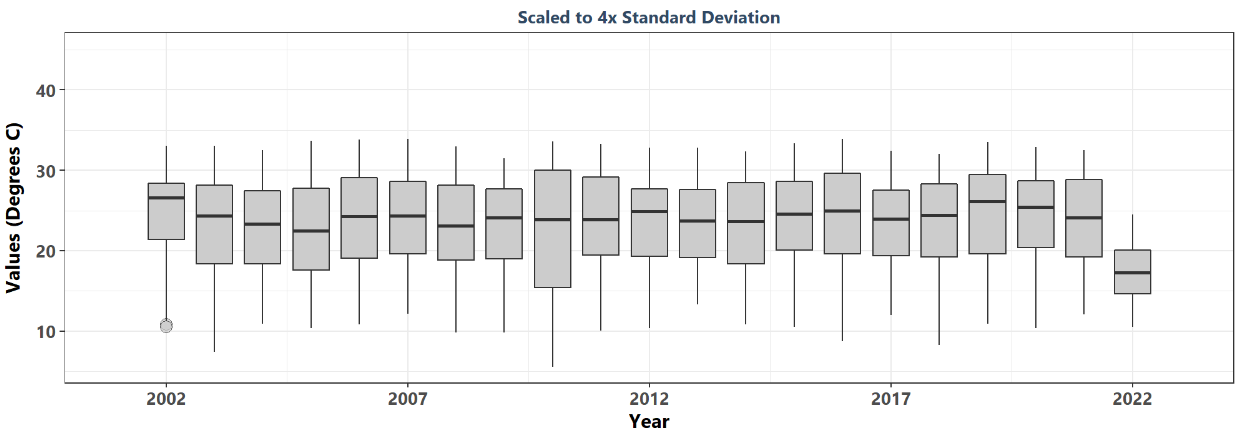


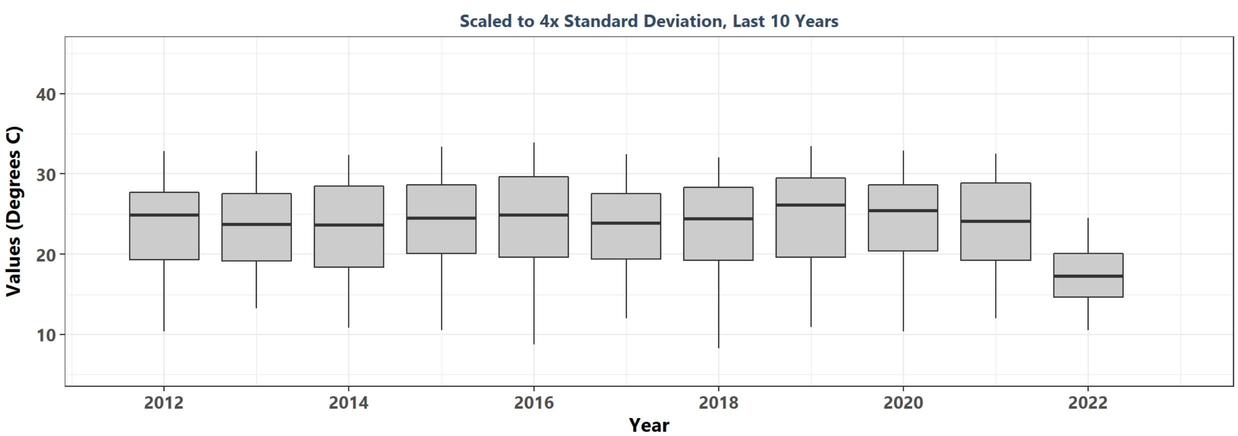
Guana Tolomato Matanzas National Estuarine Research Reserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmfmwq
By Month



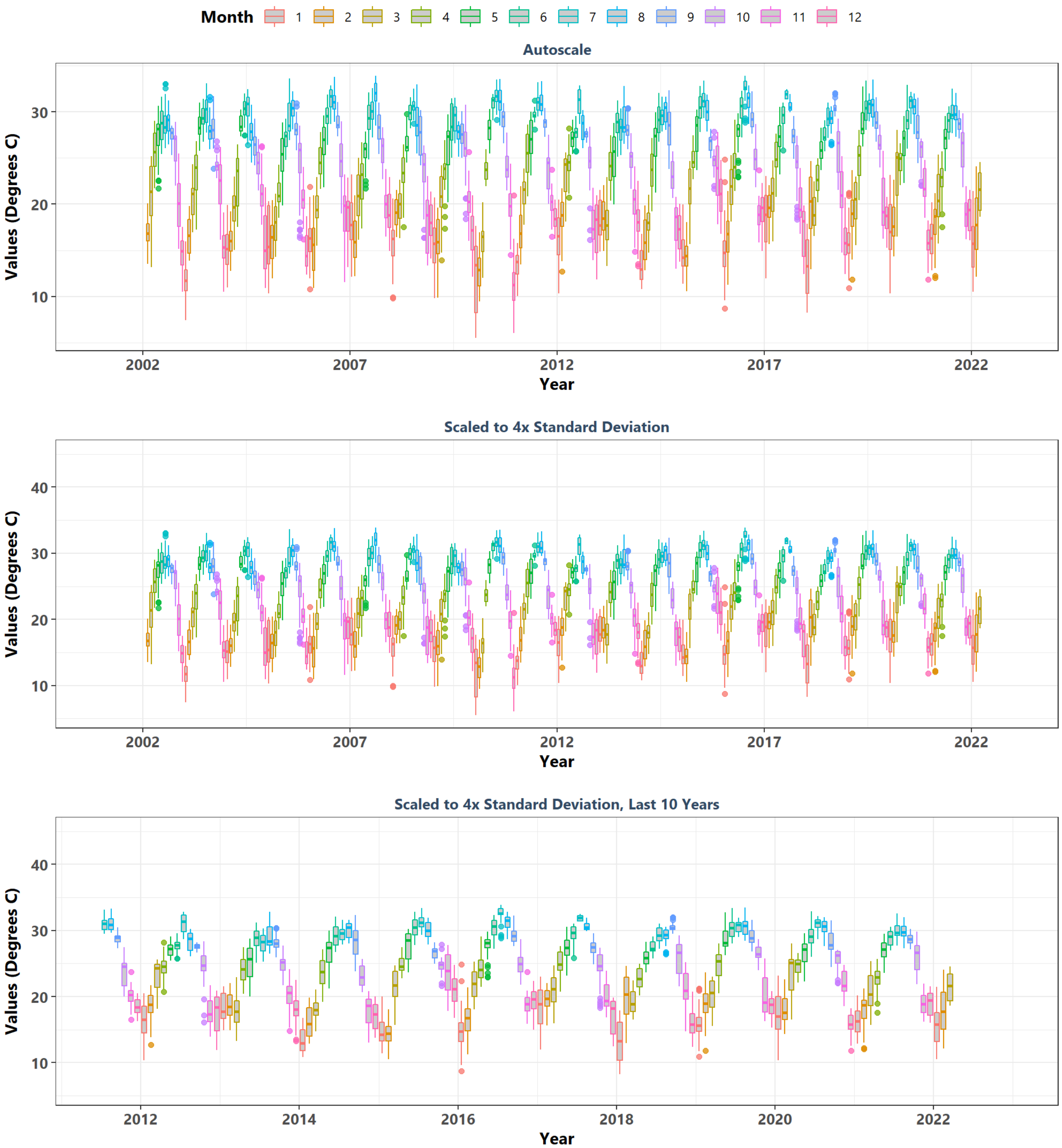
Guana Tolomato Matanzas National Estuarine Research Reserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmpcwq
By Year



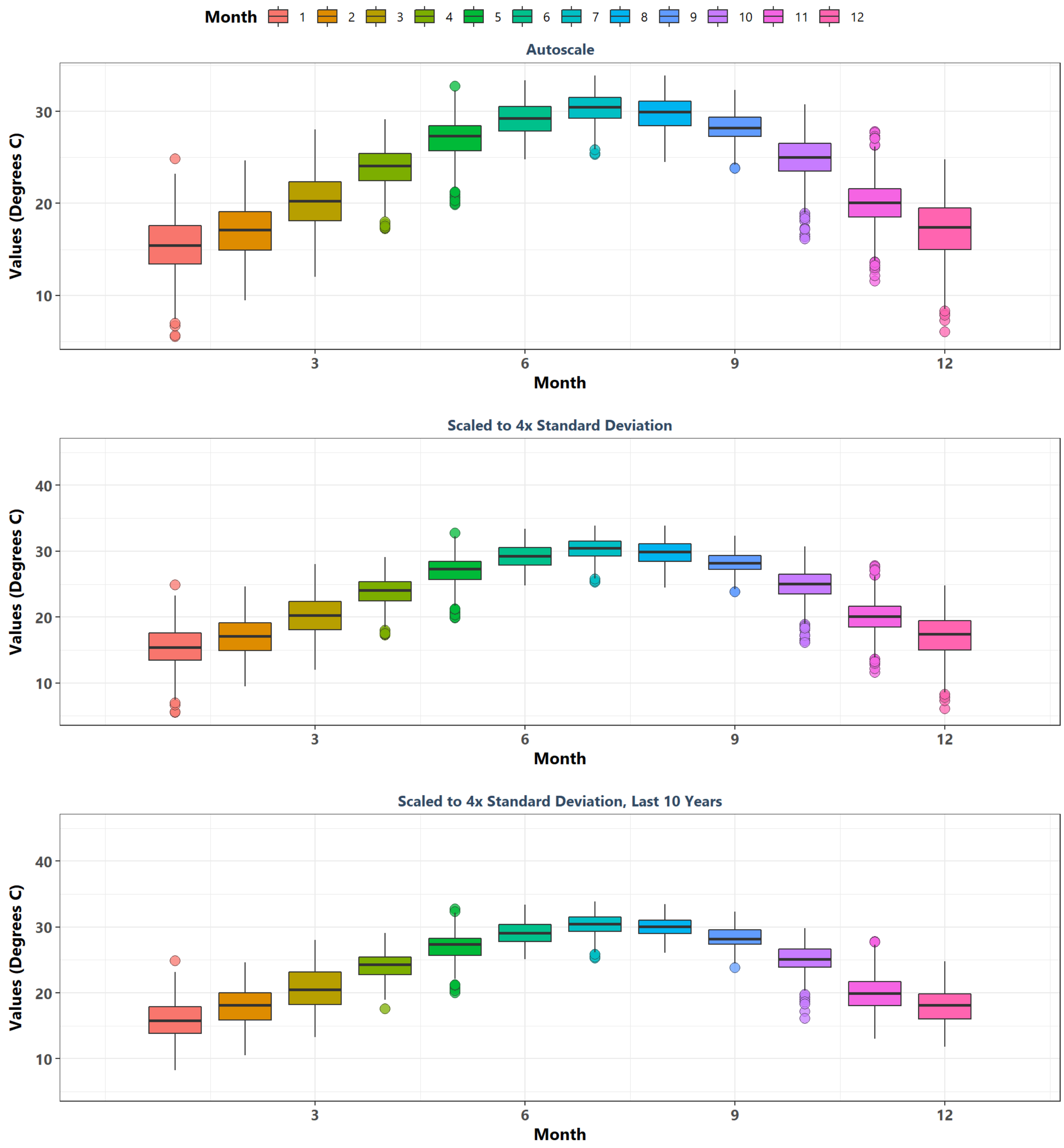




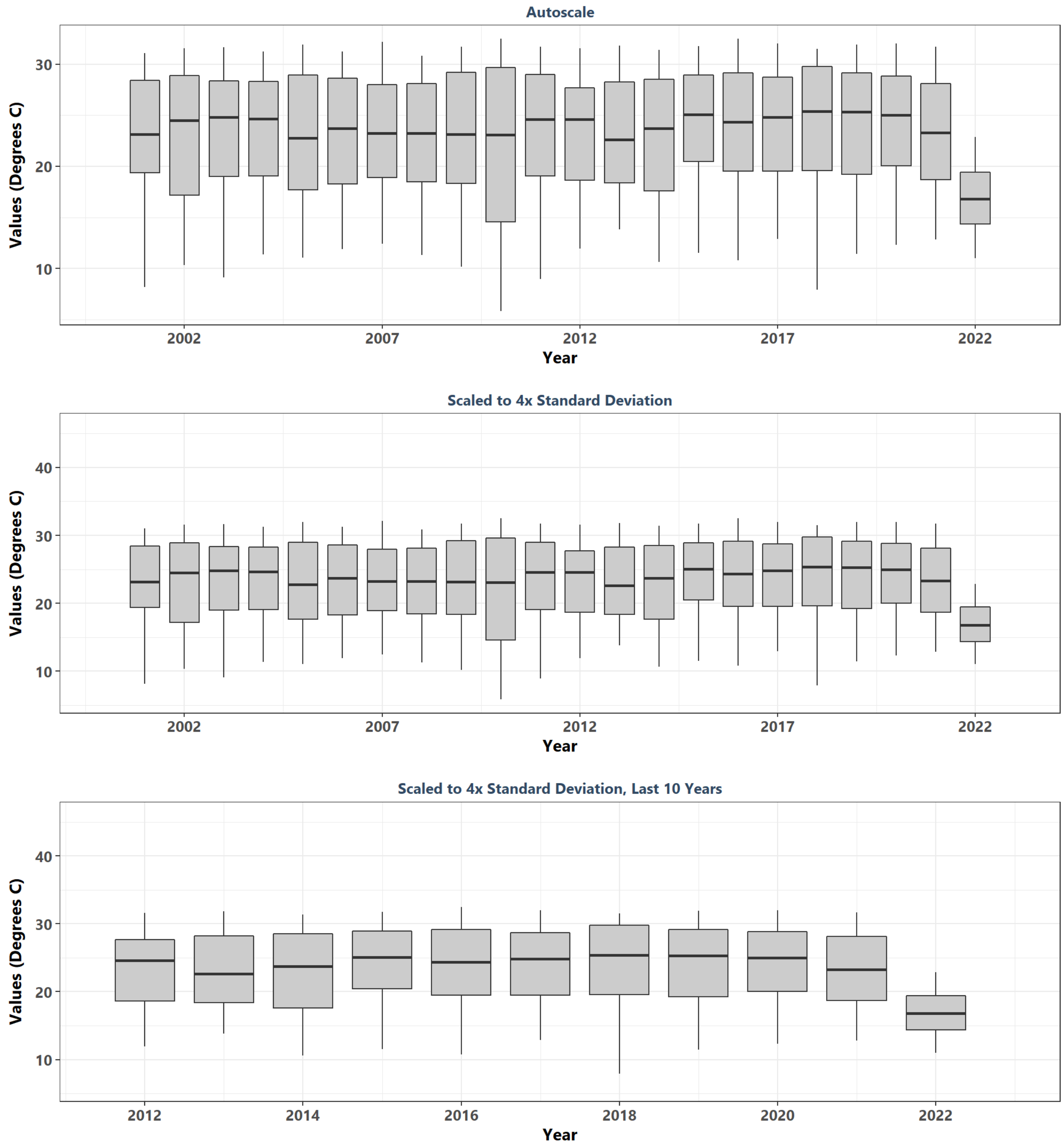
Guana Tolomato Matanzas National Estuarine Research Reserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmpcwq
By Year & Month



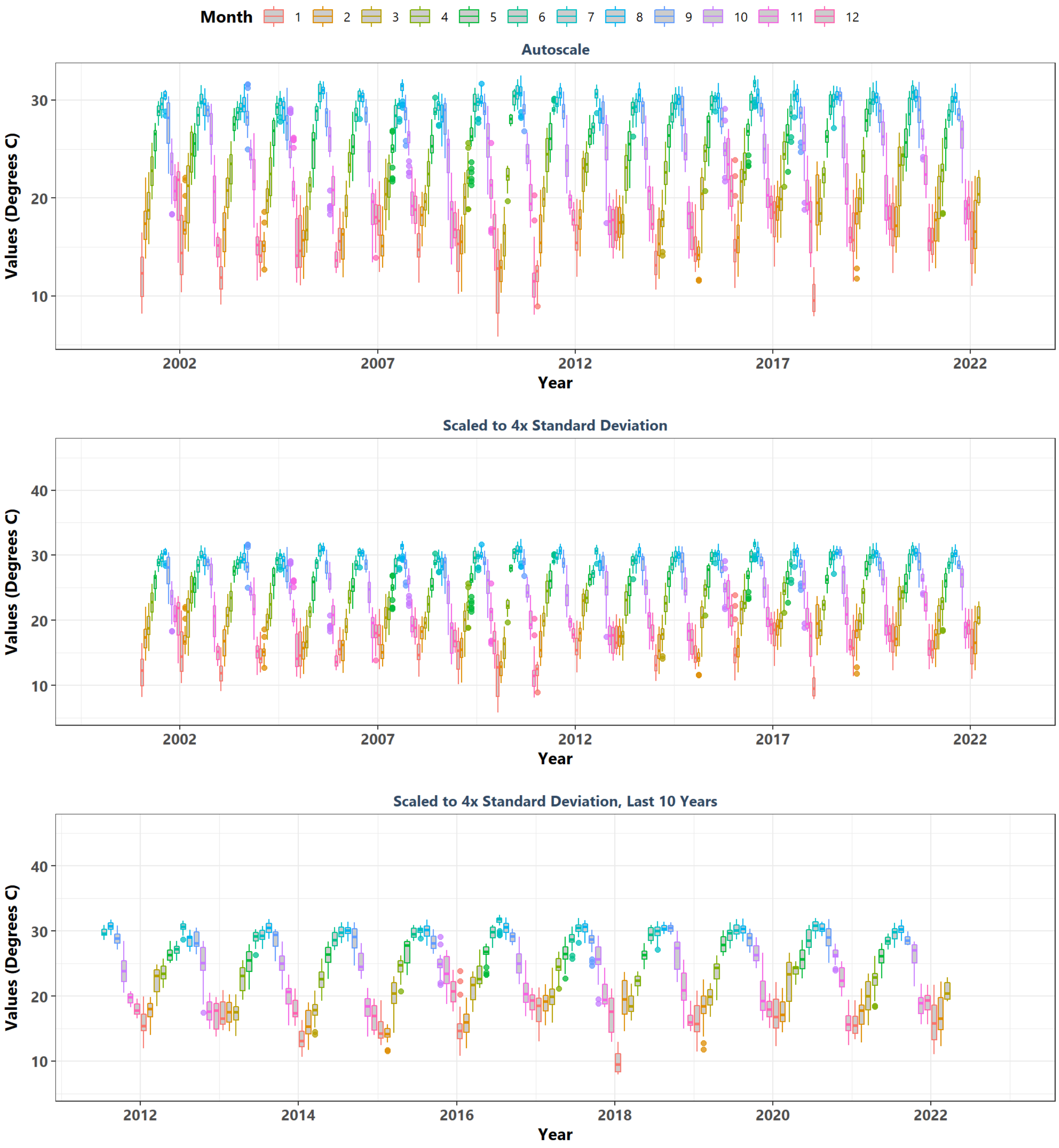
Guana Tolomato Matanzas National Estuarine Research Reserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmpcwq
By Month



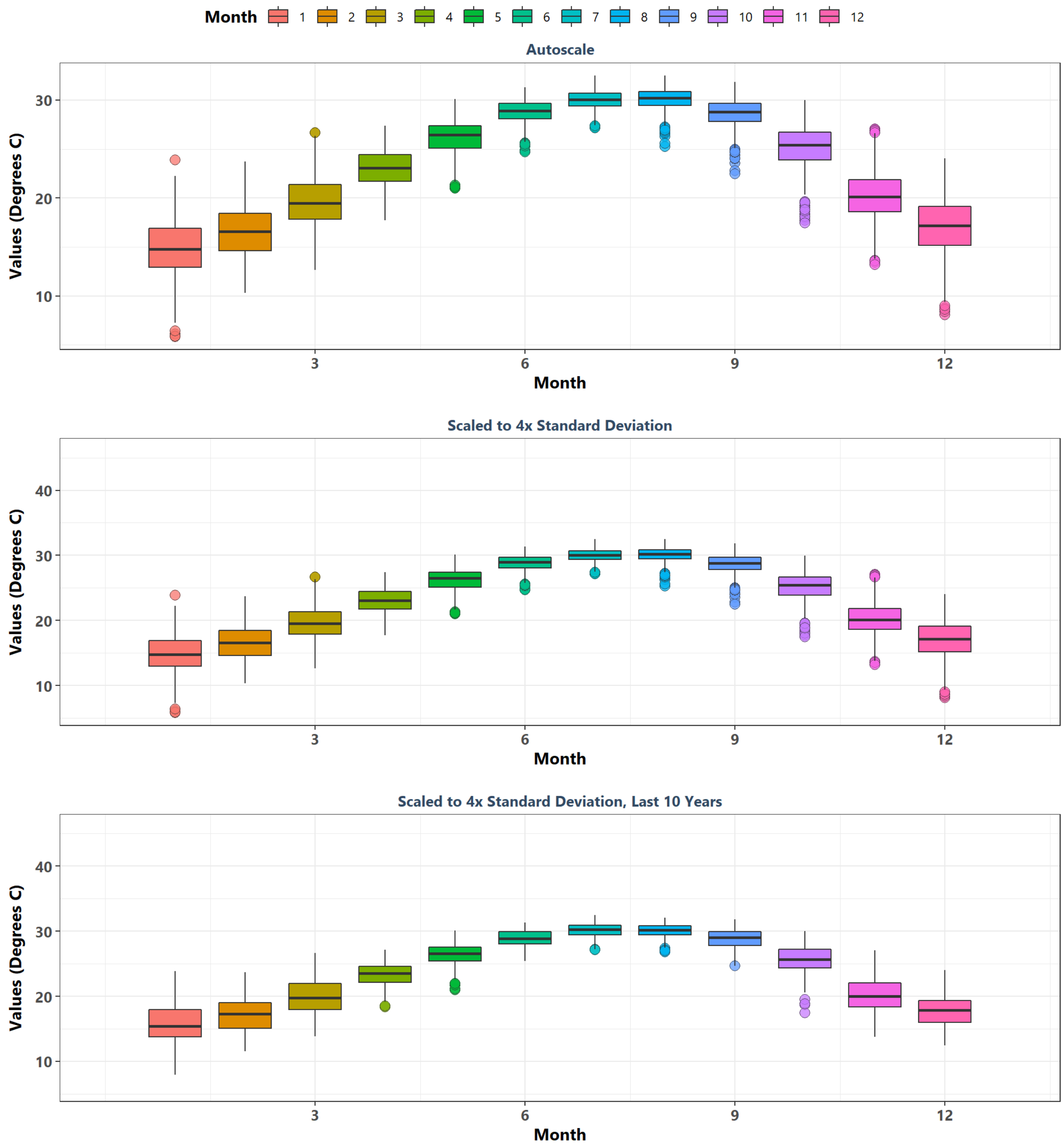
Guana Tolomato Matanzas National Estuarine Research Reserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmpiwq
By Year



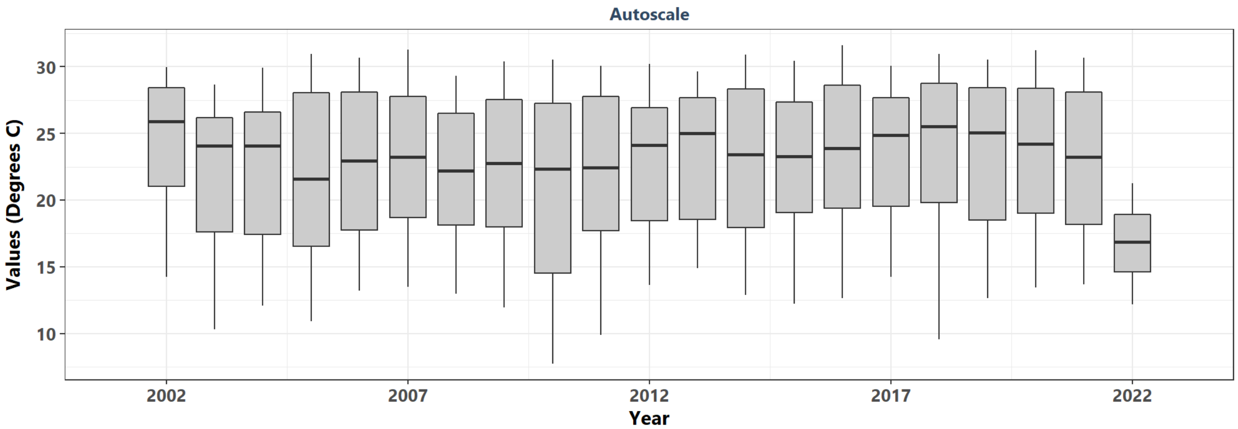
Guana Tolomato Matanzas National Estuarine Research Reserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmpiwq
By Year & Month

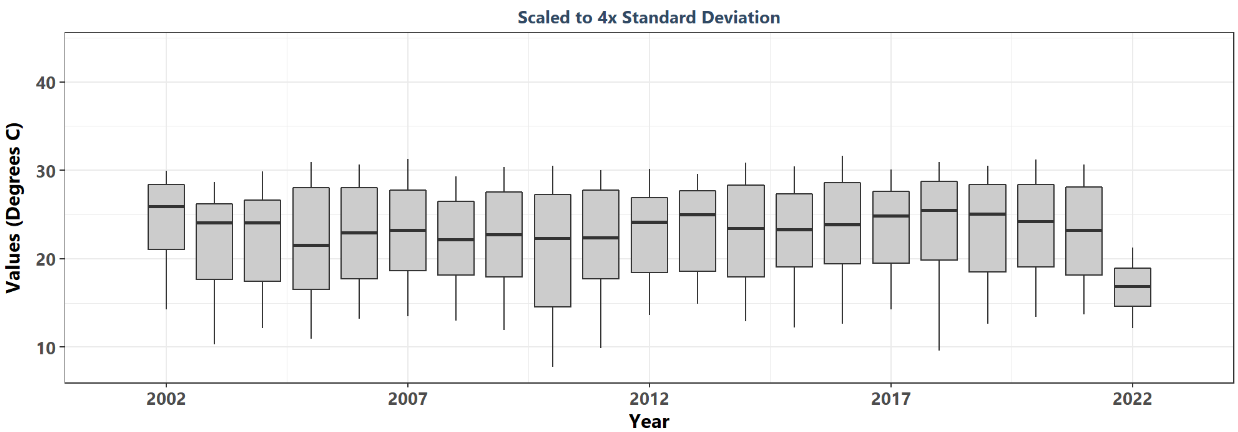


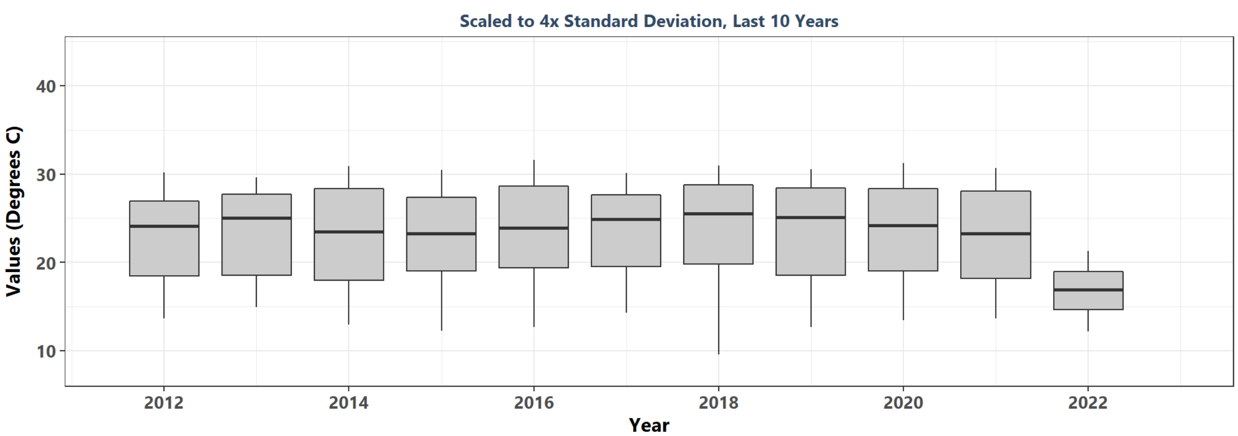
Guana Tolomato Matanzas National Estuarine Research Reserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmpiwq
By Month

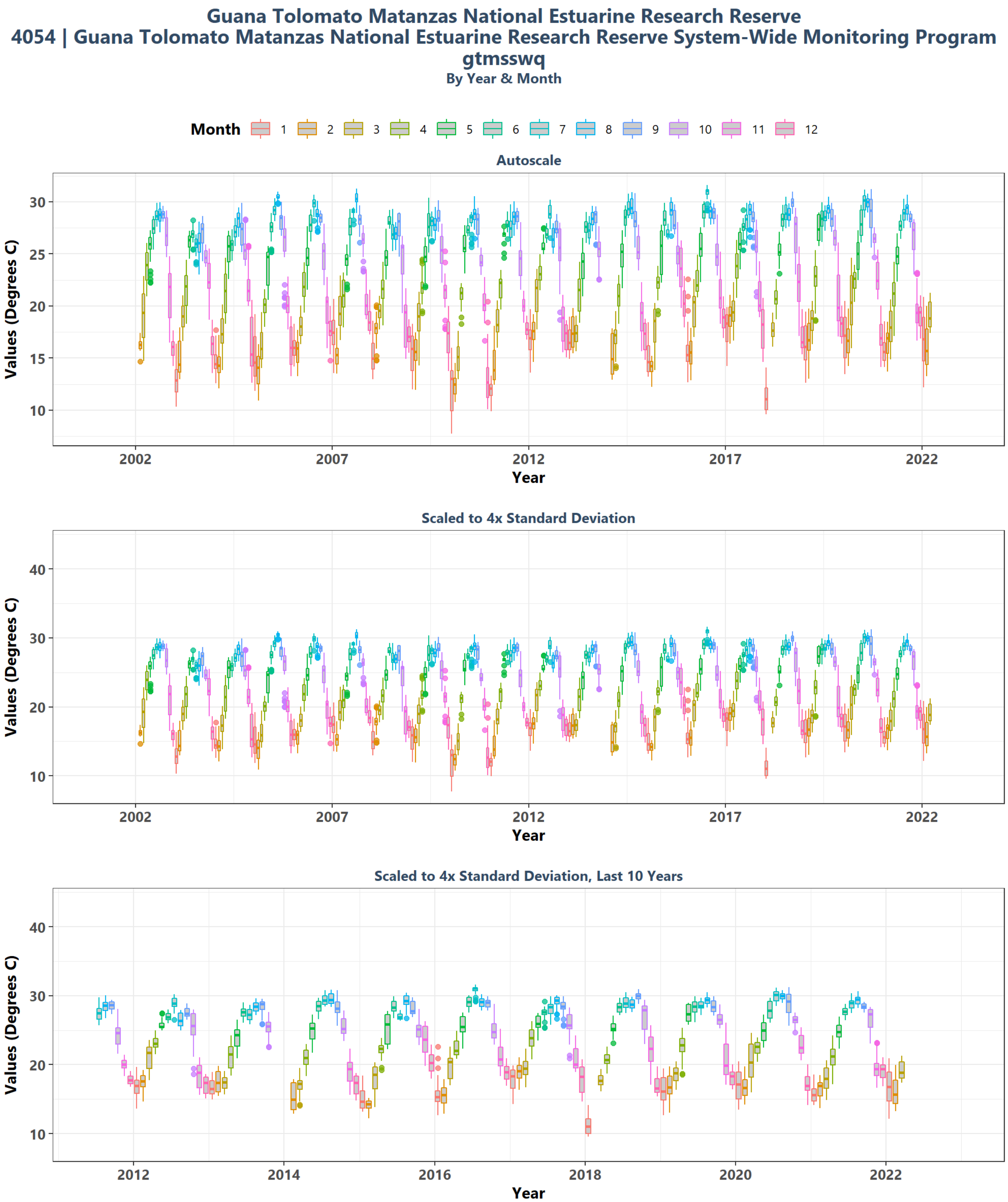


Guana Tolomato Matanzas National Estuarine Research Reserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmsswq
By Year

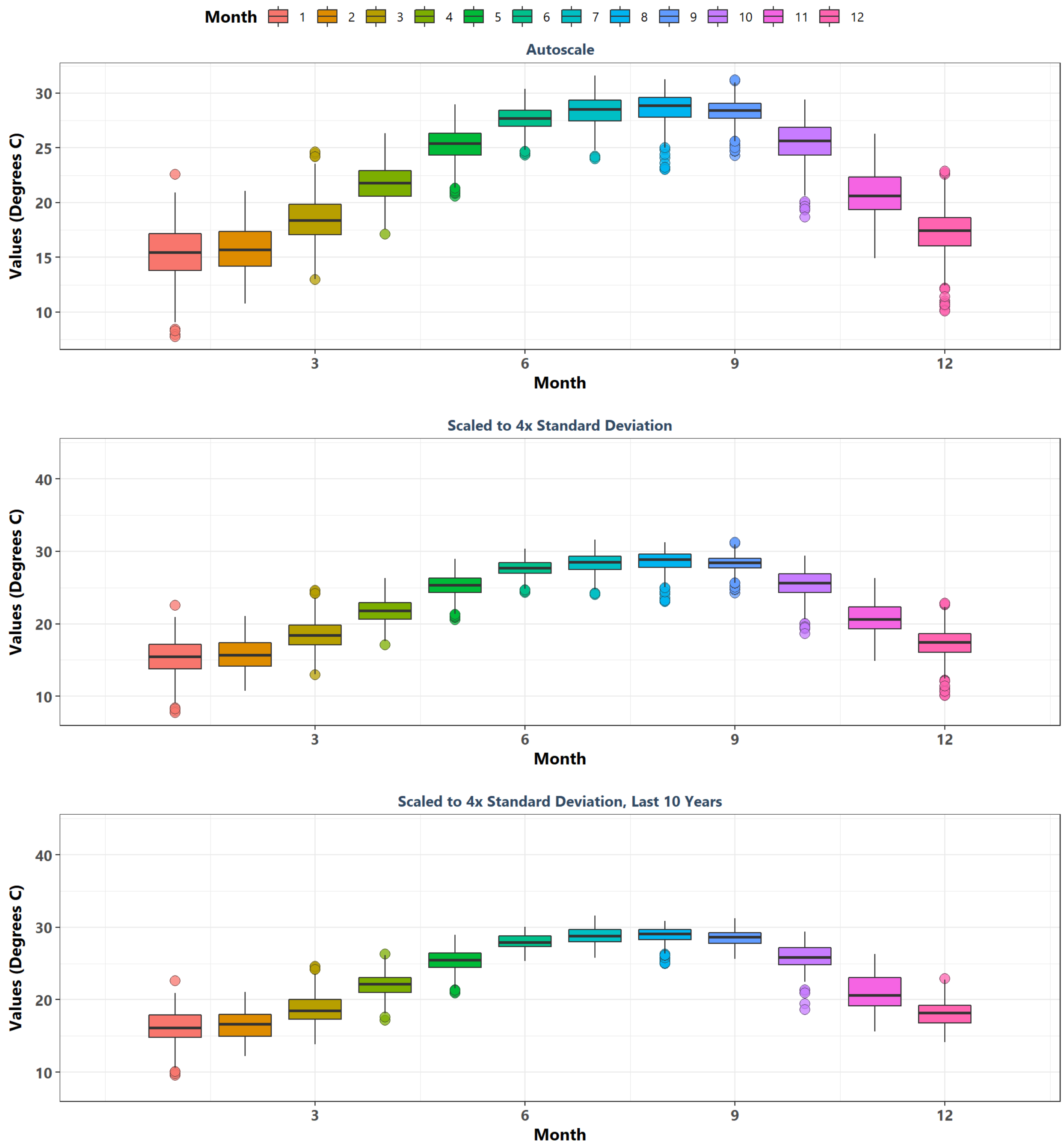




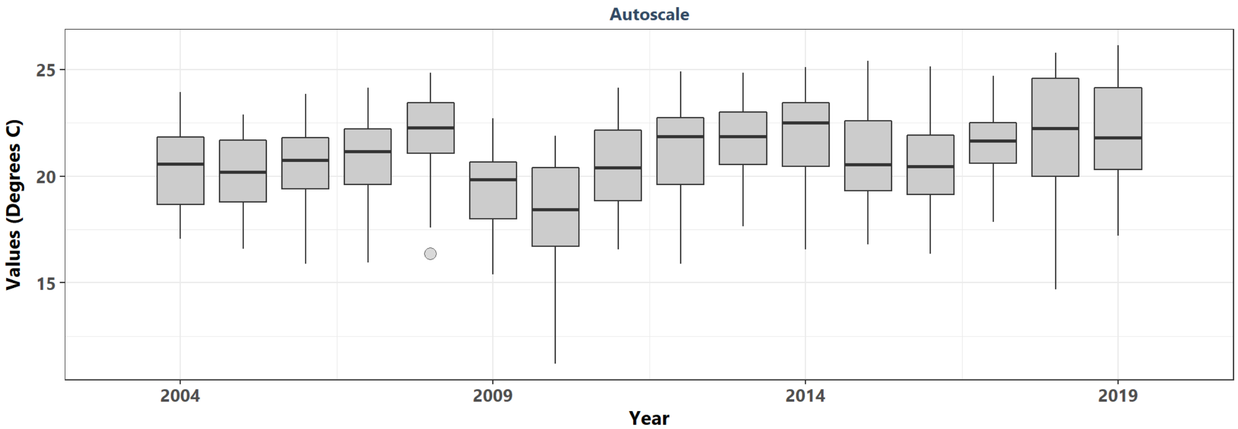


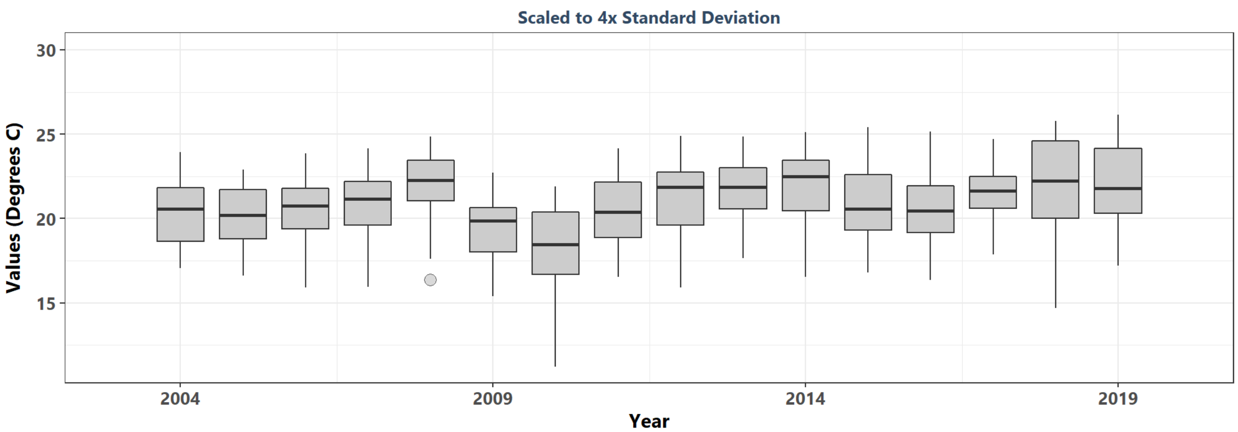

Guana Tolomato Matanzas National Estuarine Research Reserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmsswq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (Degrees C)
10
15
20
25
30
2002
2007
2012
2017
2022
Year
Scaled to 4x Standard Deviation
Values (Degrees C)
10
20
30
40
2002
2007
2012
2017
2022
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (Degrees C)
10
20
30
40
2012
2014
2016
2018
2020
2022
Year

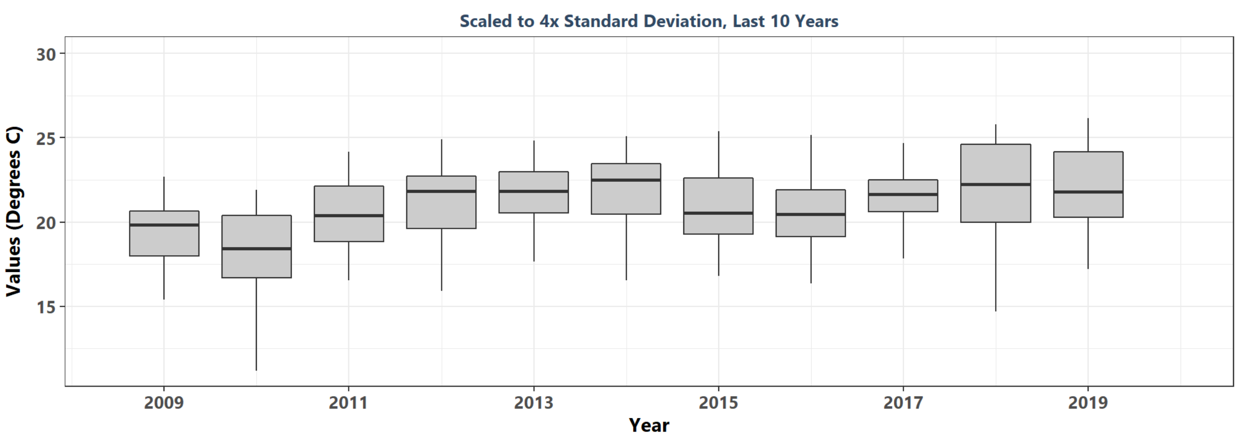
Guana Tolomato Matanzas National Estuarine Research Reserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmsswq
By Month



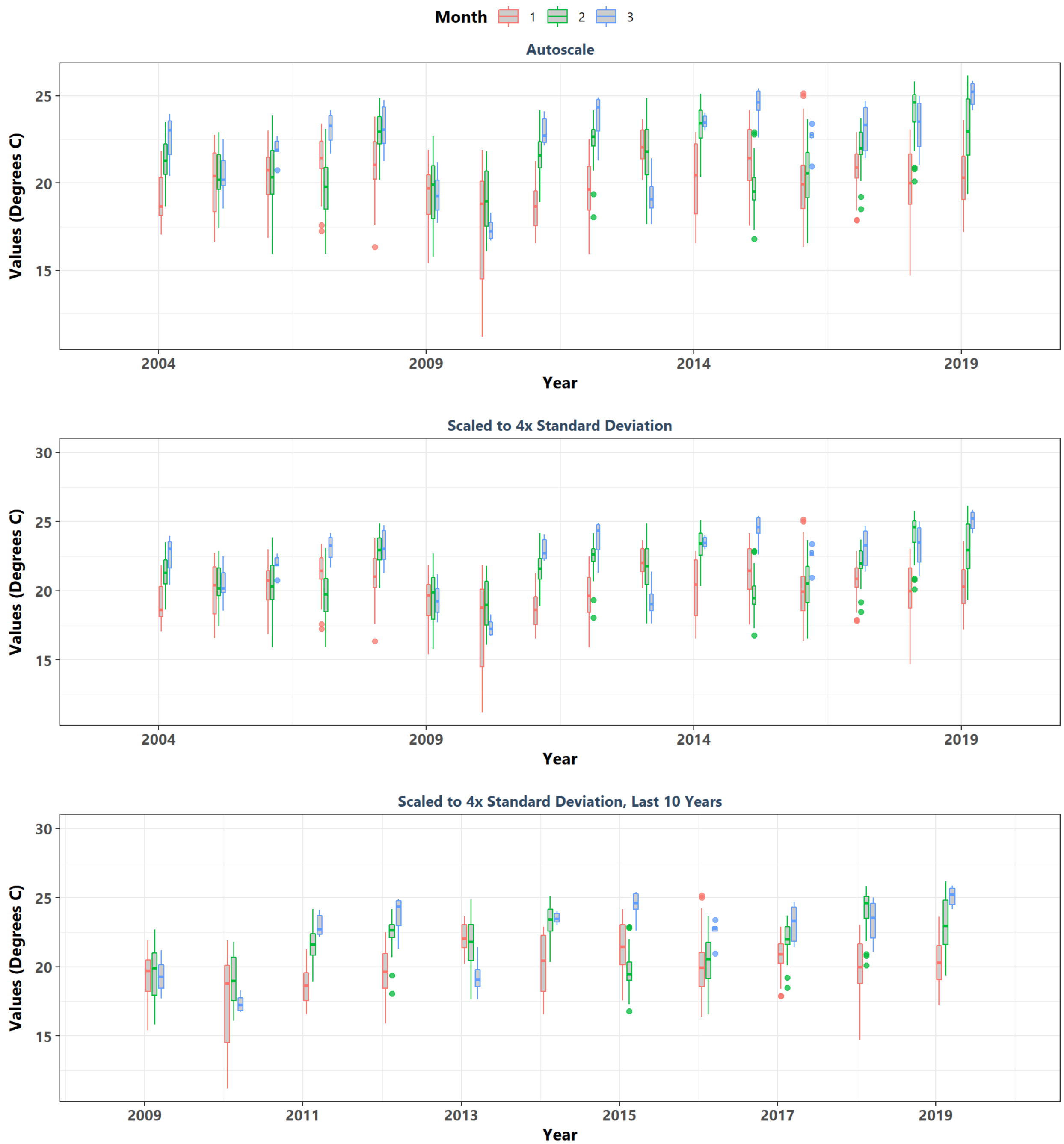
Loxahatchee River-Lake Worth Creek Aquatic Preserve
7 | National Water Information System
265906080093500
By Year





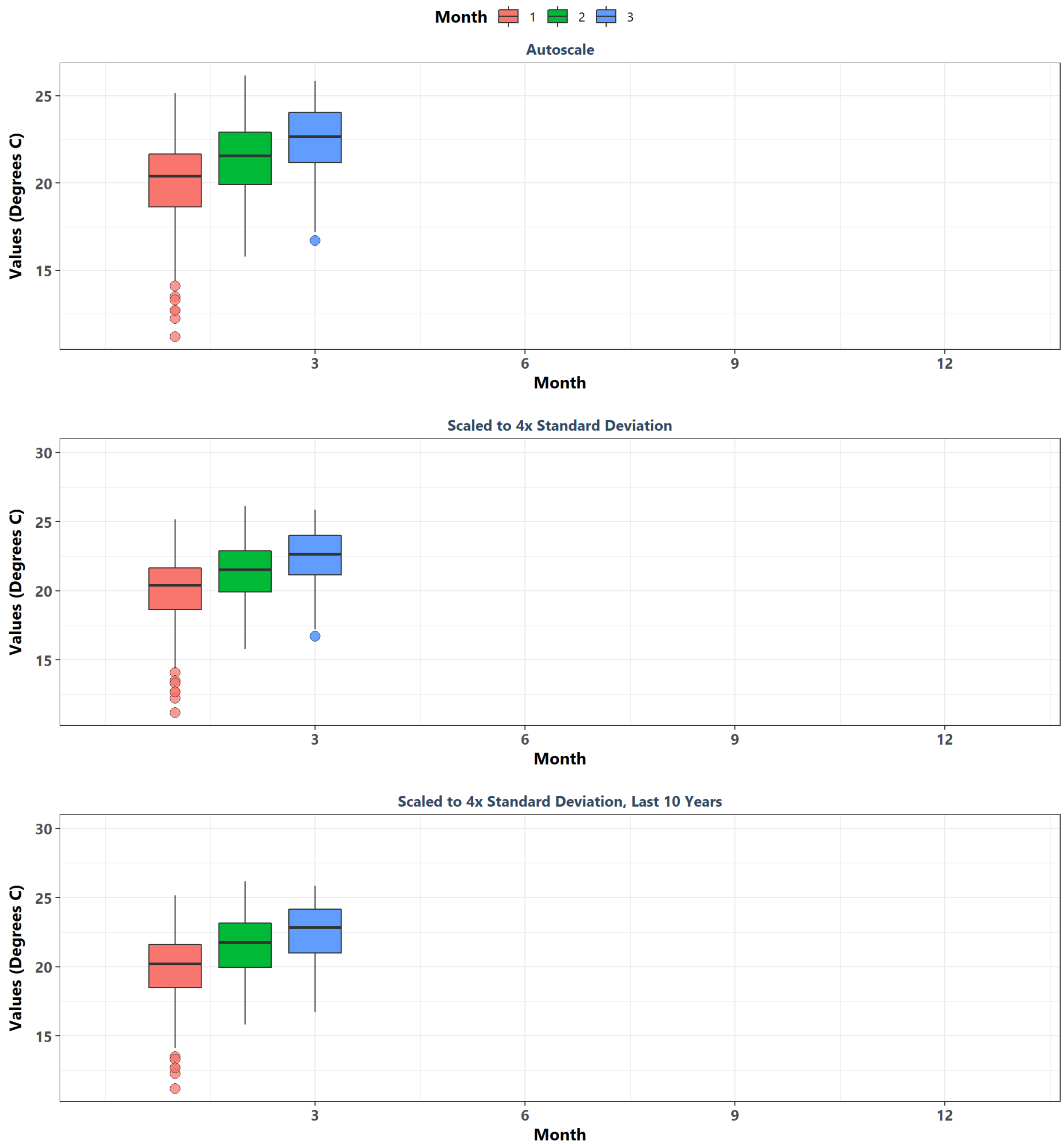


Loxahatchee River-Lake Worth Creek Aquatic Preserve
7 | National Water Information System
265906080093500
By Year & Month

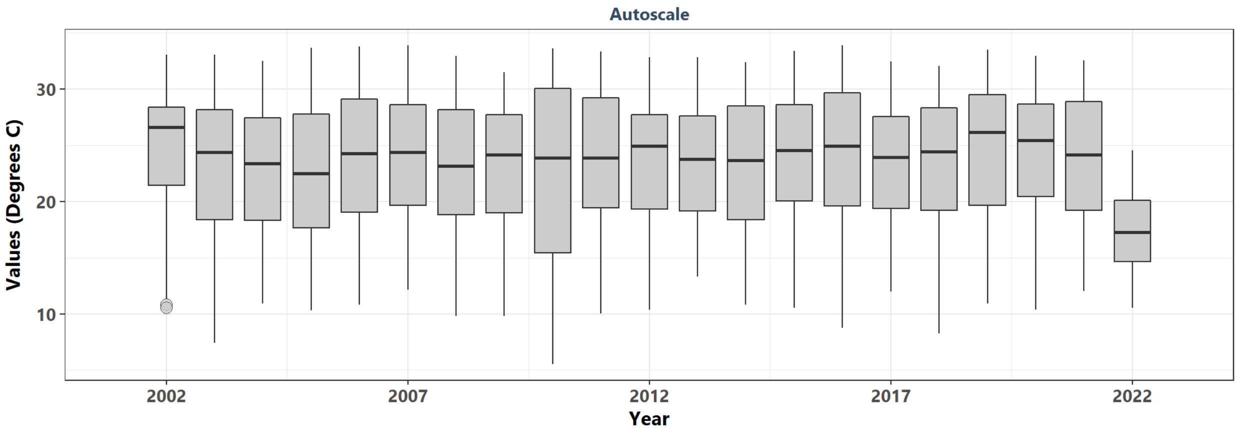


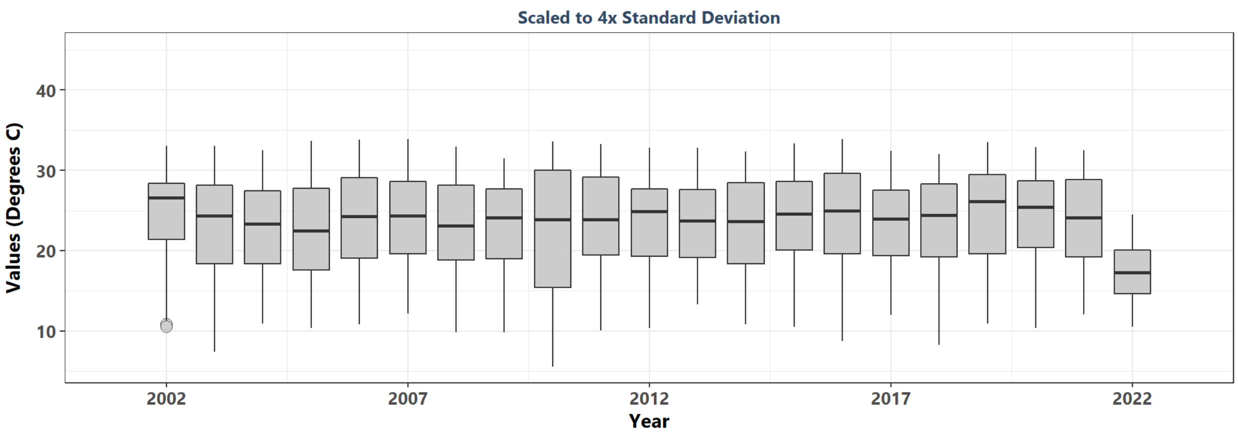
Loxahatchee River-Lake Worth Creek Aquatic Preserve
7 | National Water Information System
265906080093500

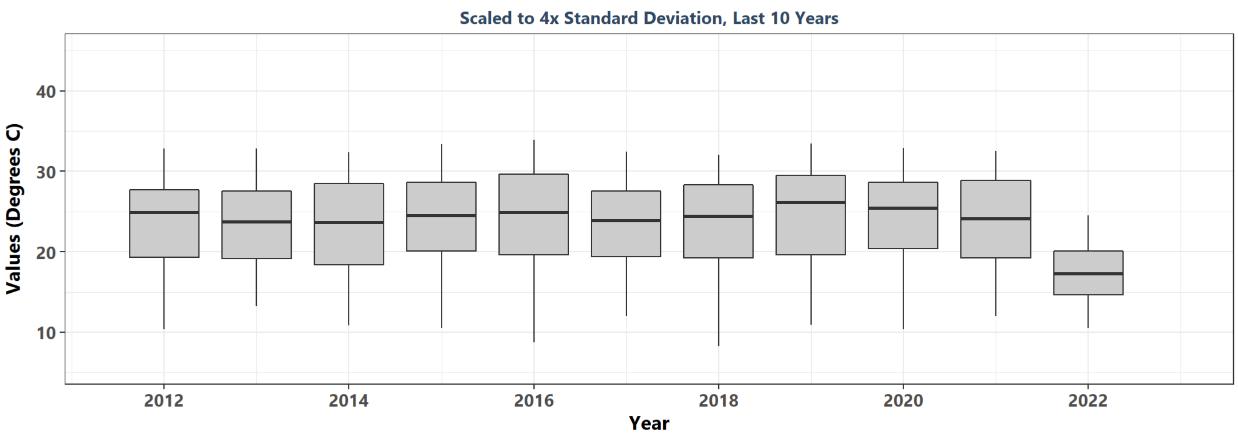
By Month



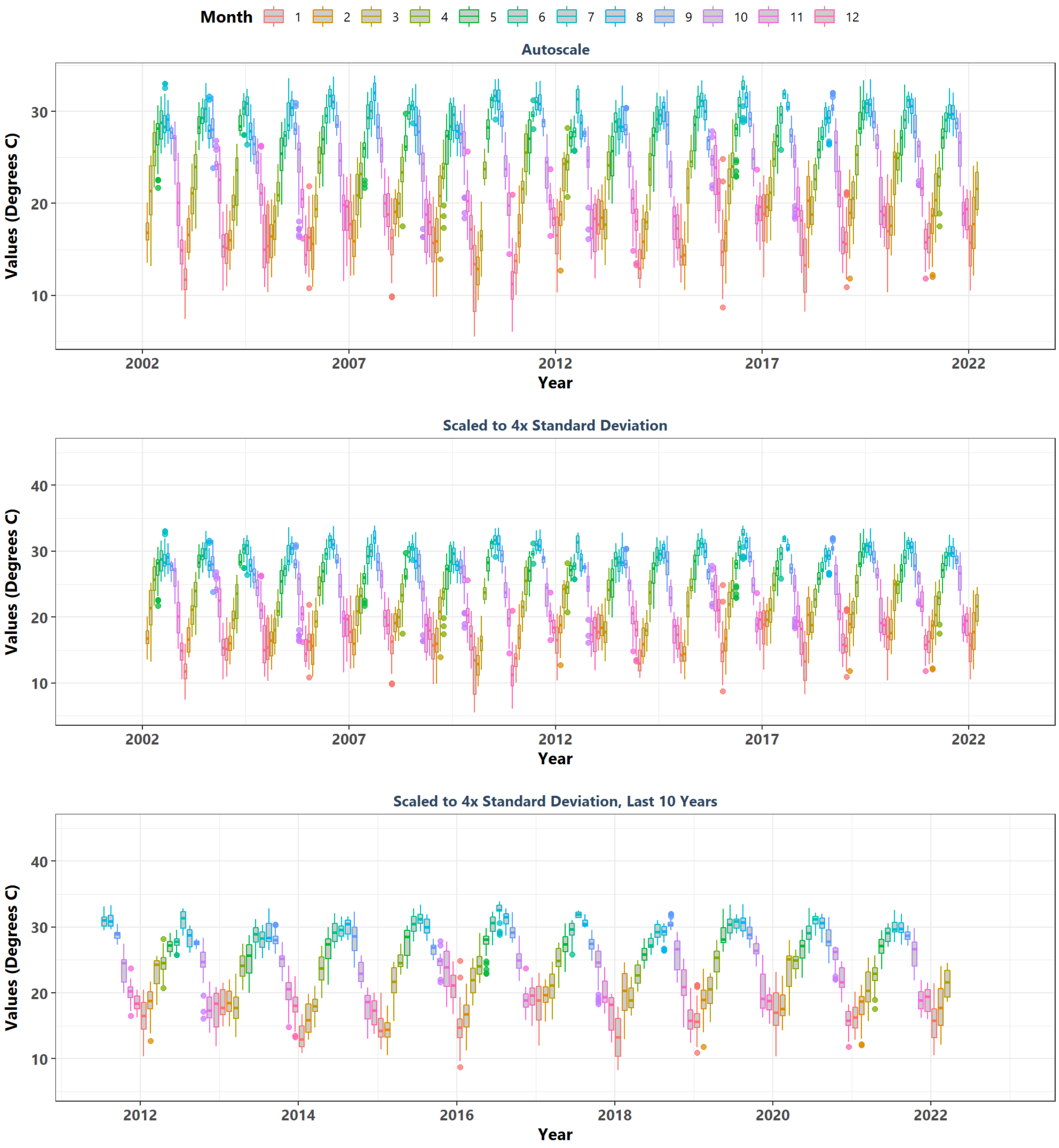
Pellicer Creek Aquatic Preserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmpcwq
By Year

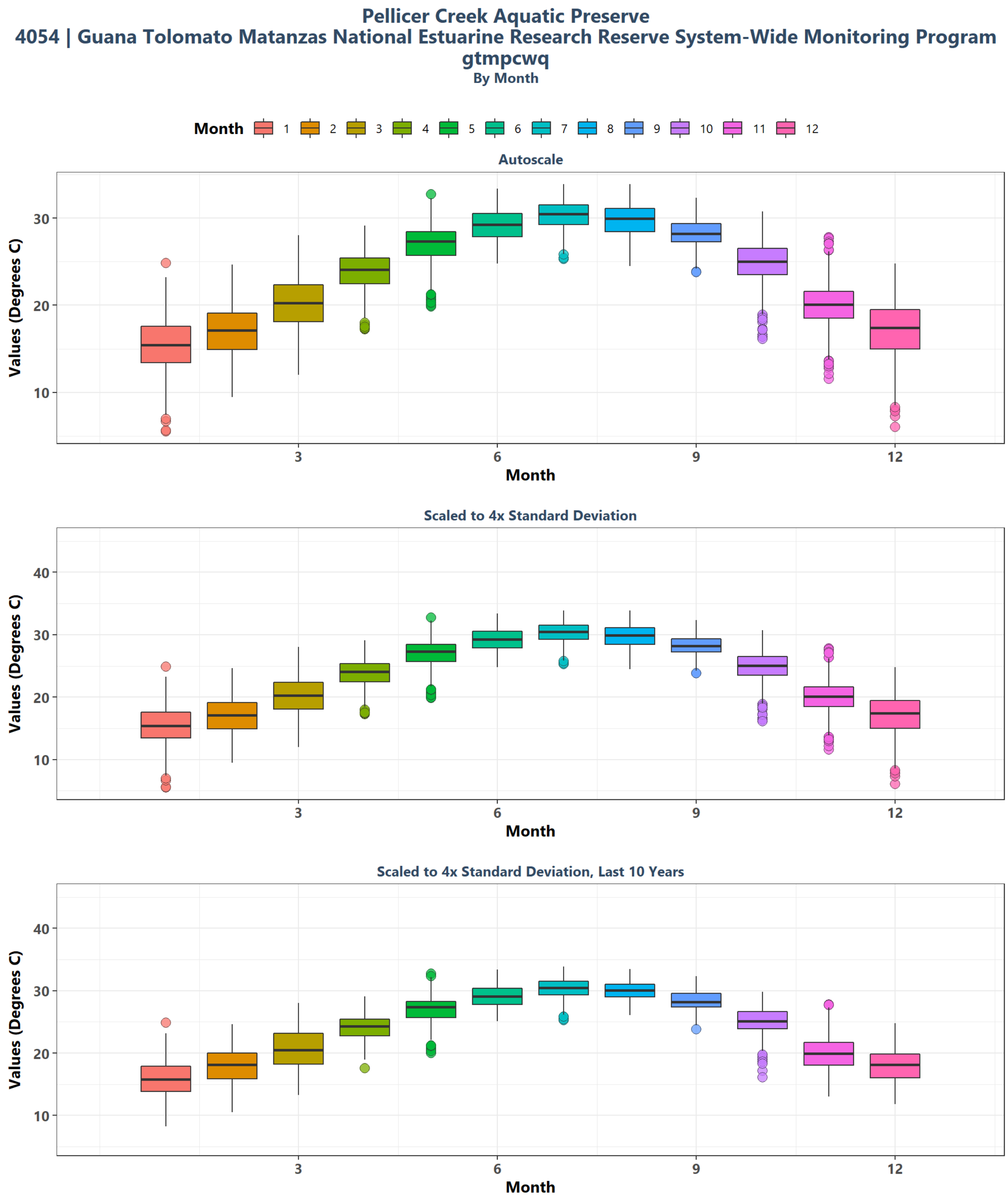






Pellicer Creek Aquatic Preserve
4054 | Guana Tolomato Matanzas National Estuarine Research Reserve System-Wide Monitoring Program
gtmpcwq
By Year & Month






```
rm(list = setdiff(ls(), c("param_name", "all_regions", "file_list", "KT.Stats_all", "MA_All", "APP_Plot
```