SEACAR Continuous Water Quality Analysis: SW Region for Salinity

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,
```

```
                                        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
```

```
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.

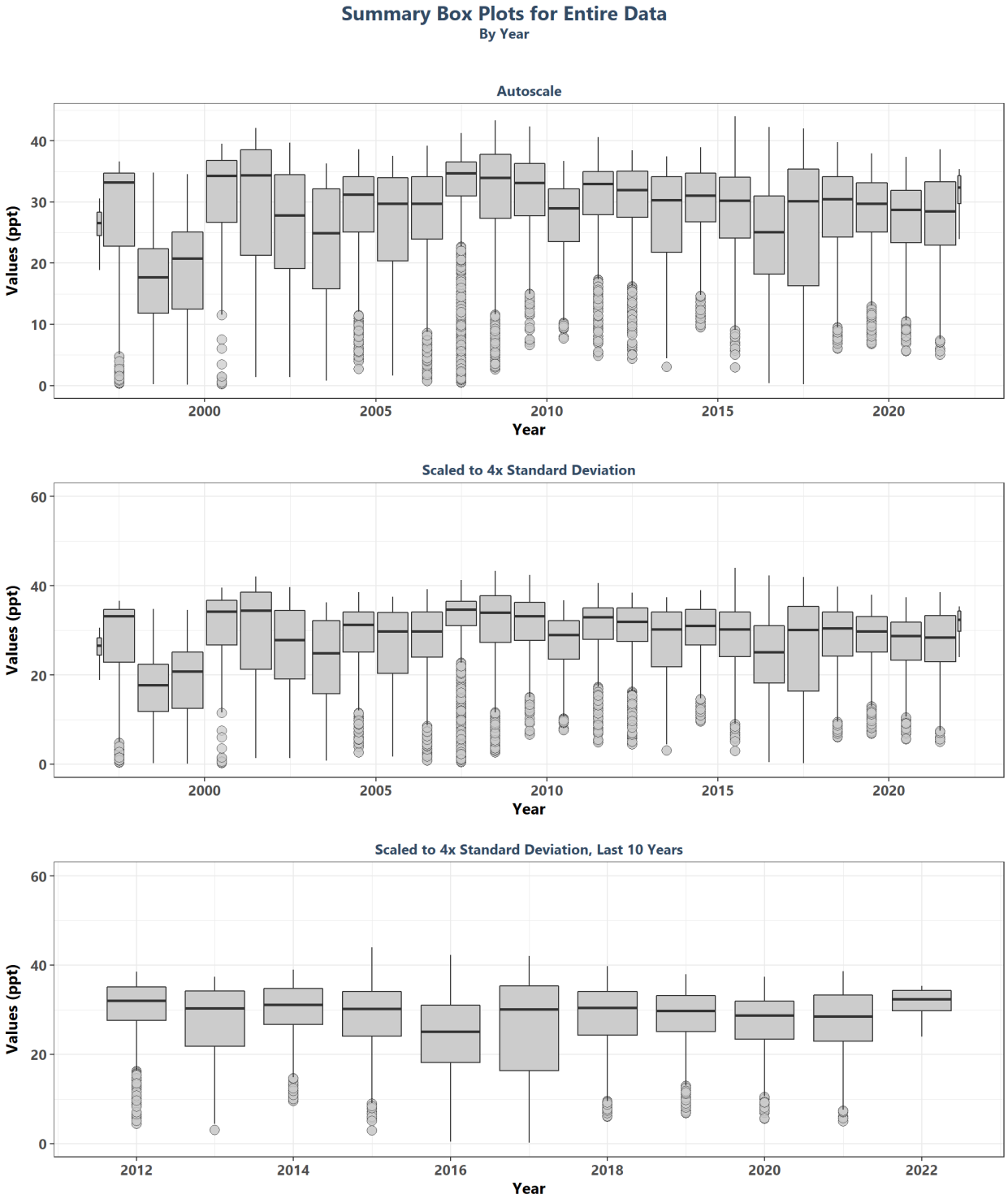
```
p1 <- ggplot(data=data[data$Include==TRUE, ],
             aes(x=YearMonthDec, y=ResultValue,
                 group=YearMonth, color=as.factor(Month))) +
   geom_boxplot(fill="#cccccc", outlier.size=1.5, outlier.alpha=0.75) +
   labs(subtitle="Autoscale", x="Year",
        y=paste0("Values (", unit, ")"), 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")
```

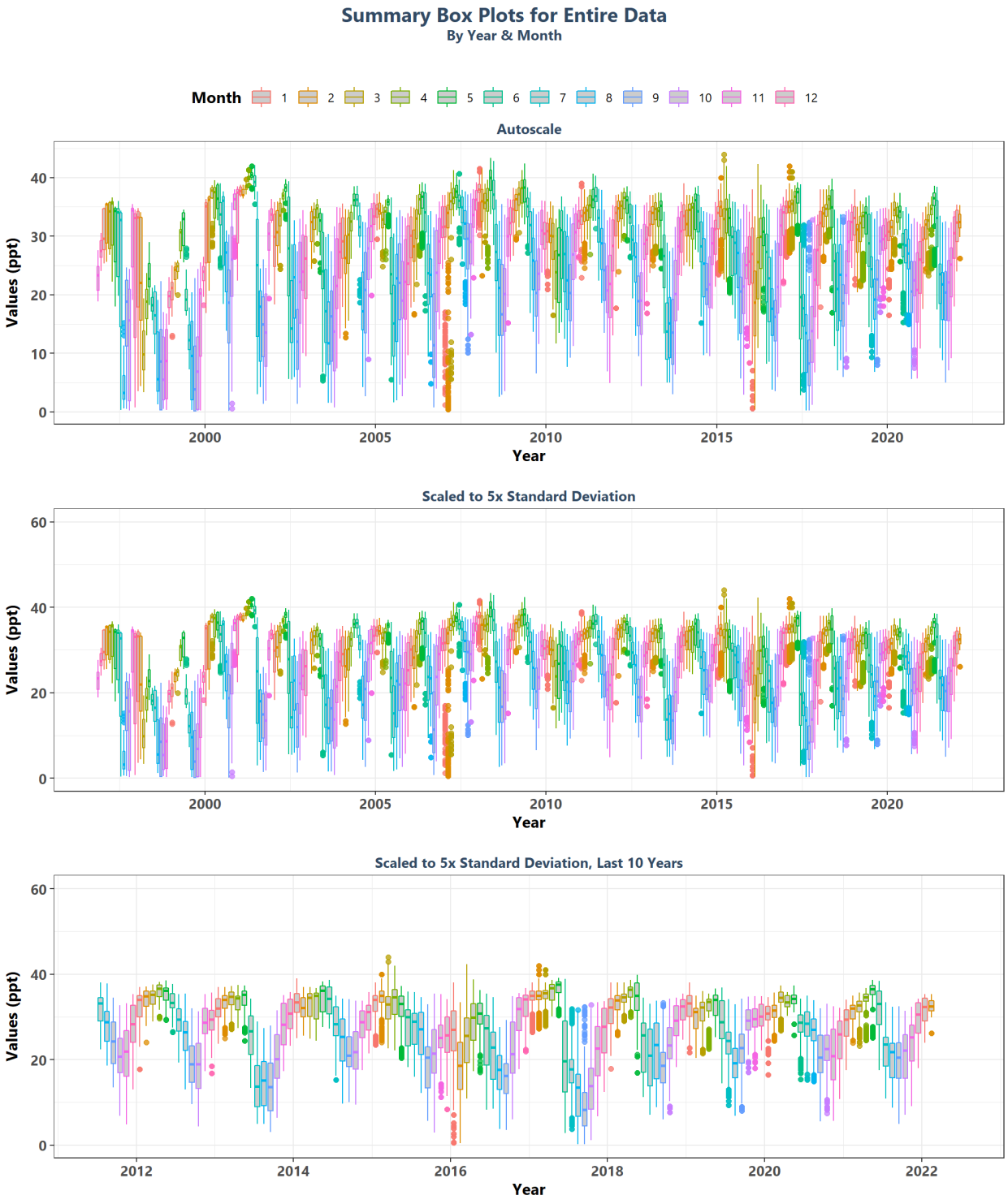
```
p3 <- ggplot(data=data[data$Include==TRUE, ],
             aes(x=YearMonthDec, y=ResultValue,
                 group=YearMonth, color=as.factor(Month))) +
   geom_boxplot(fill="#cccccc", outlier.size=1.5, outlier.alpha=0.75) +
   labs(subtitle="Scaled to 5x Standard Deviation, Last 10 Years",
        x="Year", y=paste0("Values (", unit, ")")) +
   ylim(0, y_scale) +
   scale_x_continuous(limits=c(max(data$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))
```

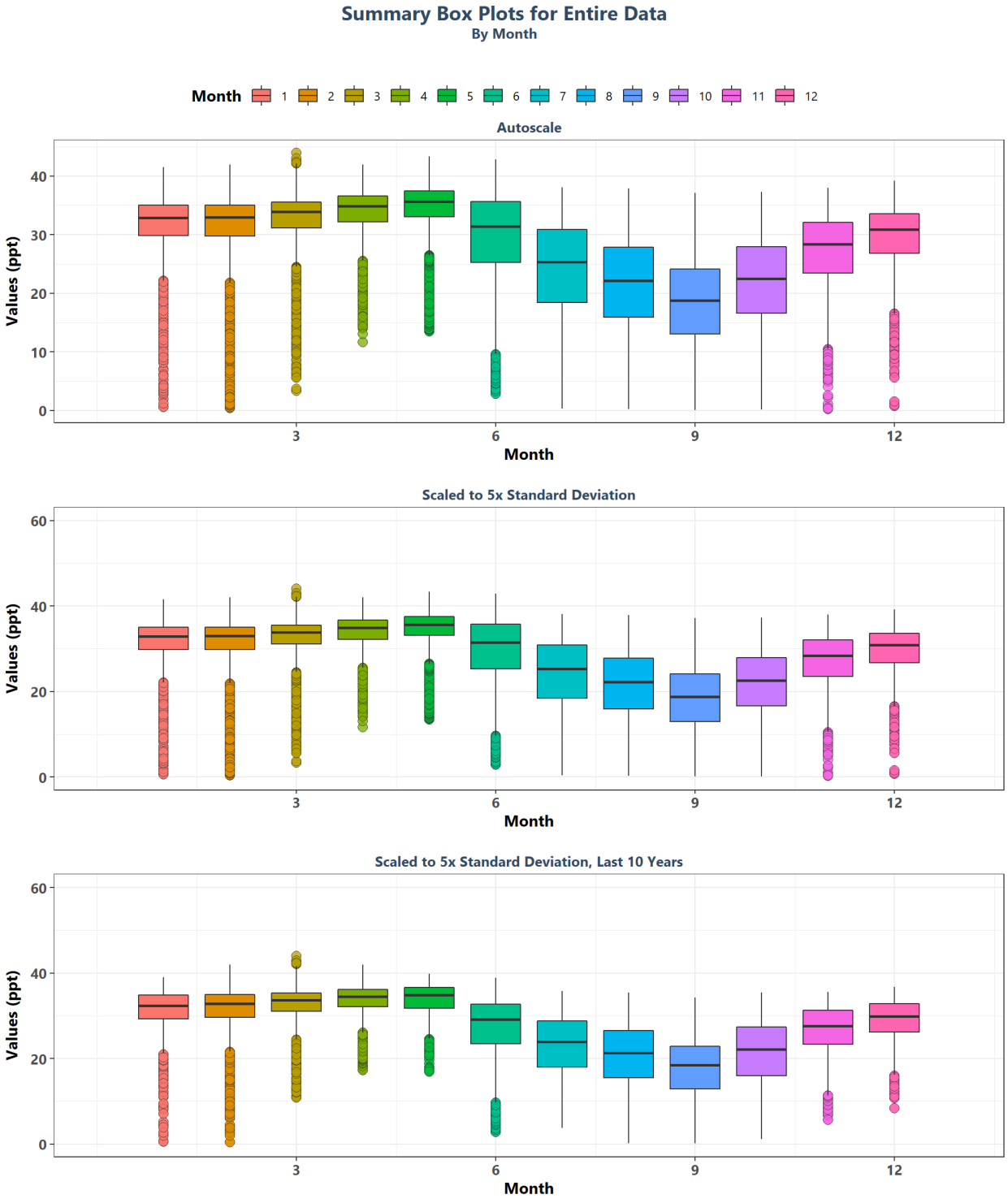
The following box plots are grouped by month with fill color being related to the month. This is designed to view potential seasonal trends.

```
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))) +
```

```
   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 (ppt)
0
10
20
30
40
2000
2005
2010
2015
2020
Year
Scaled to 4x Standard Deviation
Values (ppt)
0
20
40
60
2000
2005
2010
2015
2020
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
0
20
40
60
2012
2014
2016
2018
2020
2022
Year

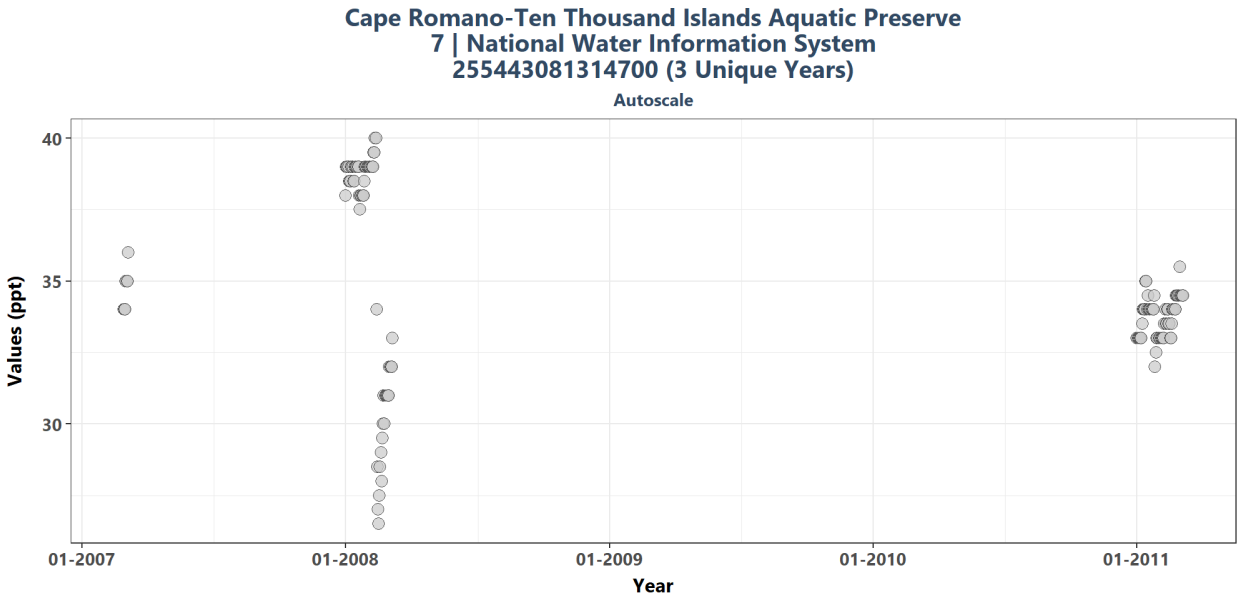

Summary Box Plots for Entire Data
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 5x Standard Deviation
Values (ppt)
Year
Scaled to 5x Standard Deviation, Last 10 Years
Values (ppt)
Year

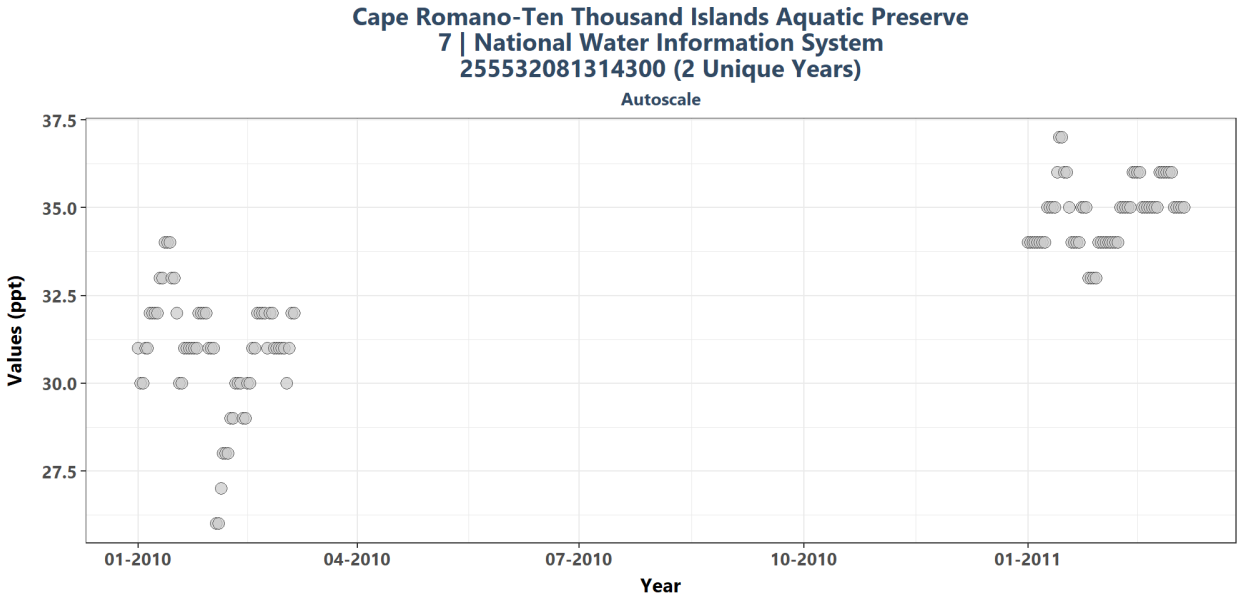


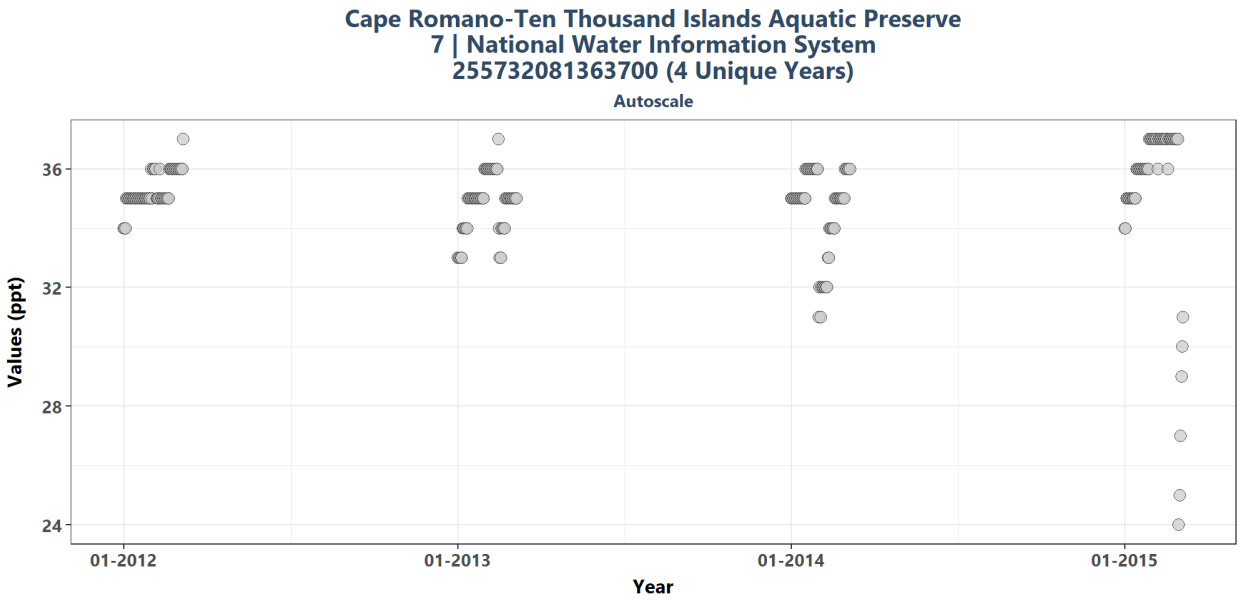
Appendix II: Excluded Monitoring Locations

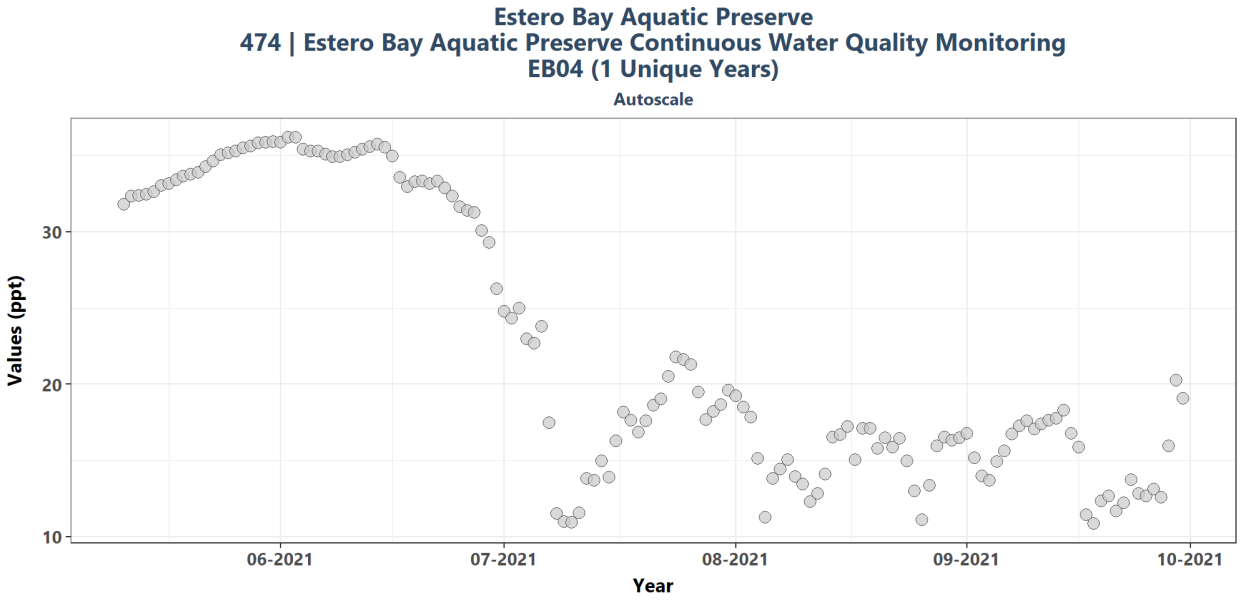
Scatter plots of data values are created for monitoring locations that have fewer than 5 separate years of data entries.

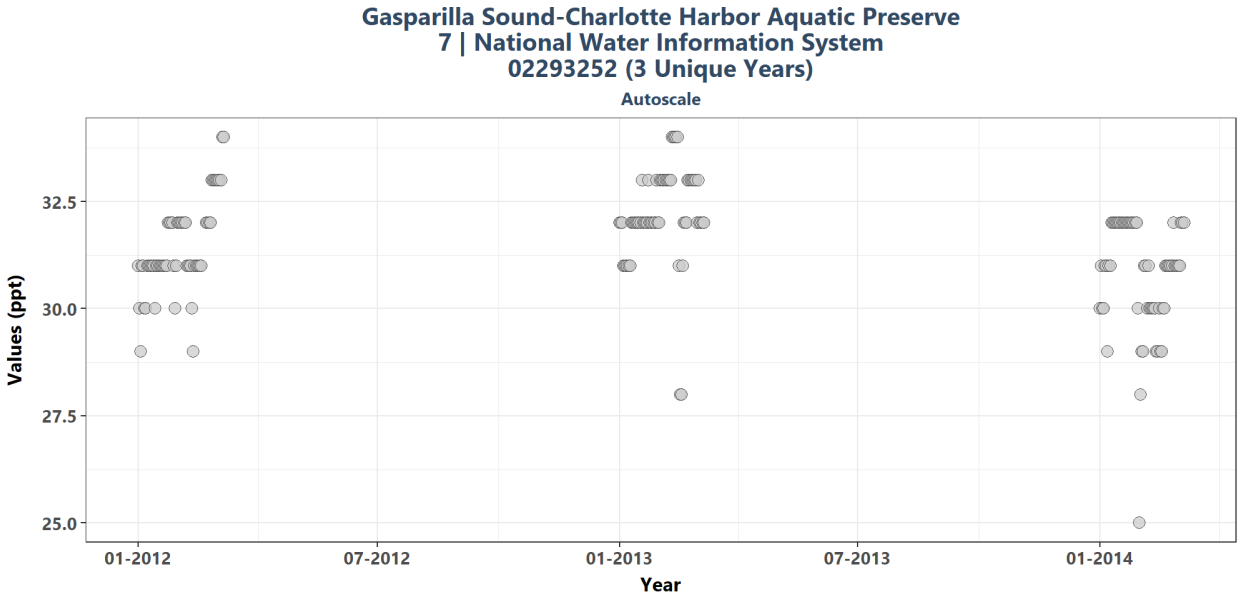
```
Mon_Exclude <- Mon_Summ[Mon_Summ$N_Years<5 & Mon_Summ$N_Years>0,]
Mon_Exclude <- Mon_Exclude[order(Mon_Exclude$MonitoringID),]
z=nrow(Mon_Exclude)

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)
   }
}
```

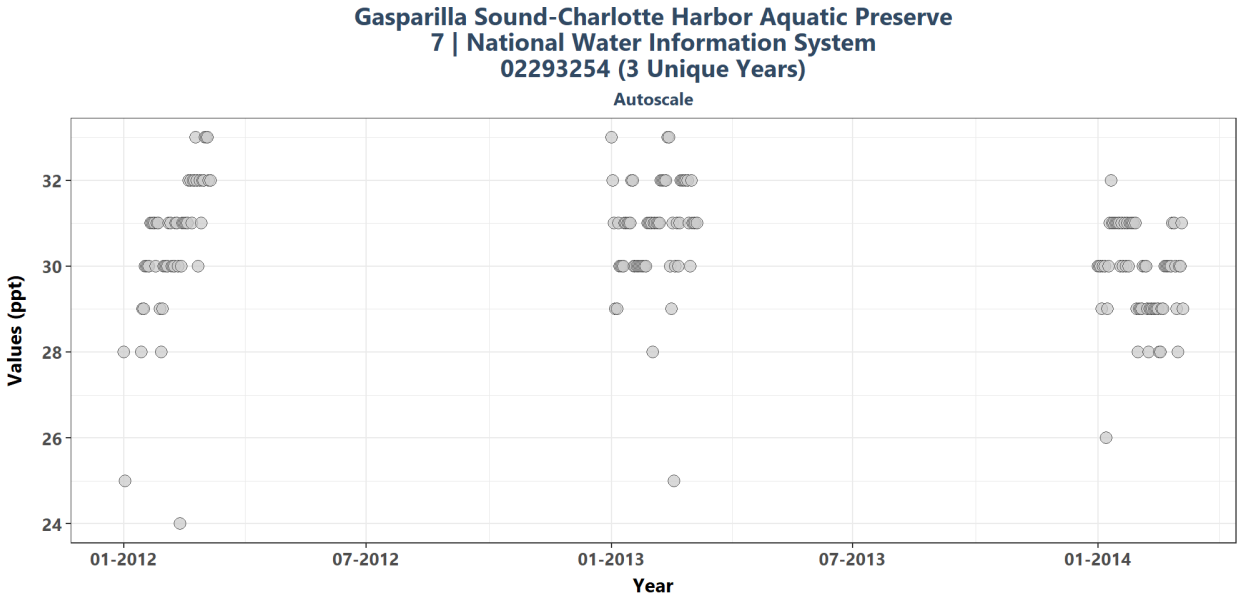


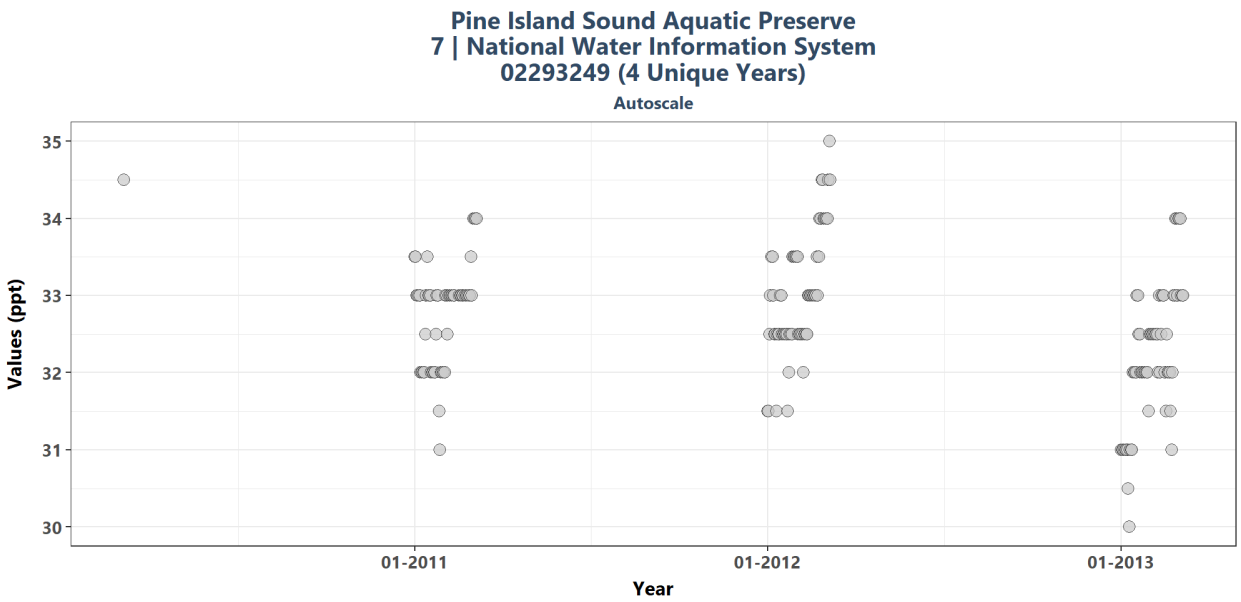

Cape Romano-Ten Thousand Islands Aquatic Preserve
7 | National Water Information System
255532081314300 (2 Unique Years)
Autoscale
Values (ppt)
37.5
35.0
32.5
30.0
27.5
01-2010
04-2010
07-2010
10-2010
01-2011
Year

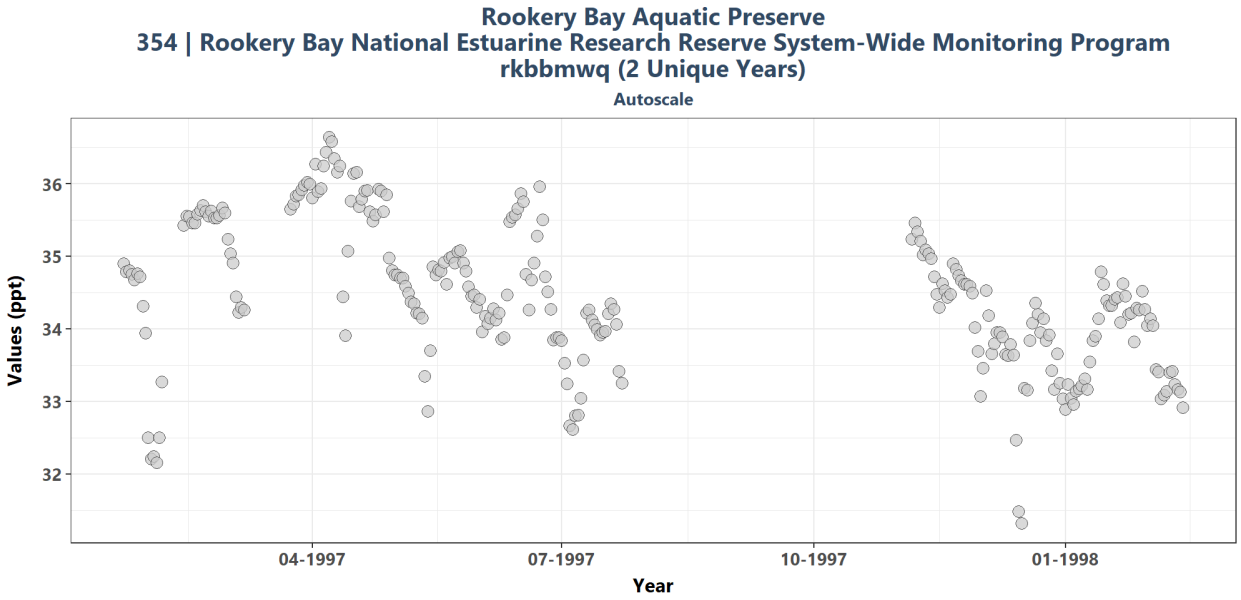

Cape Romano-Ten Thousand Islands Aquatic Preserve
7 | National Water Information System
255732081363700 (4 Unique Years)
Autoscale
Values (ppt)
36
32
28
24
01-2012
01-2013
01-2014
01-2015
Year

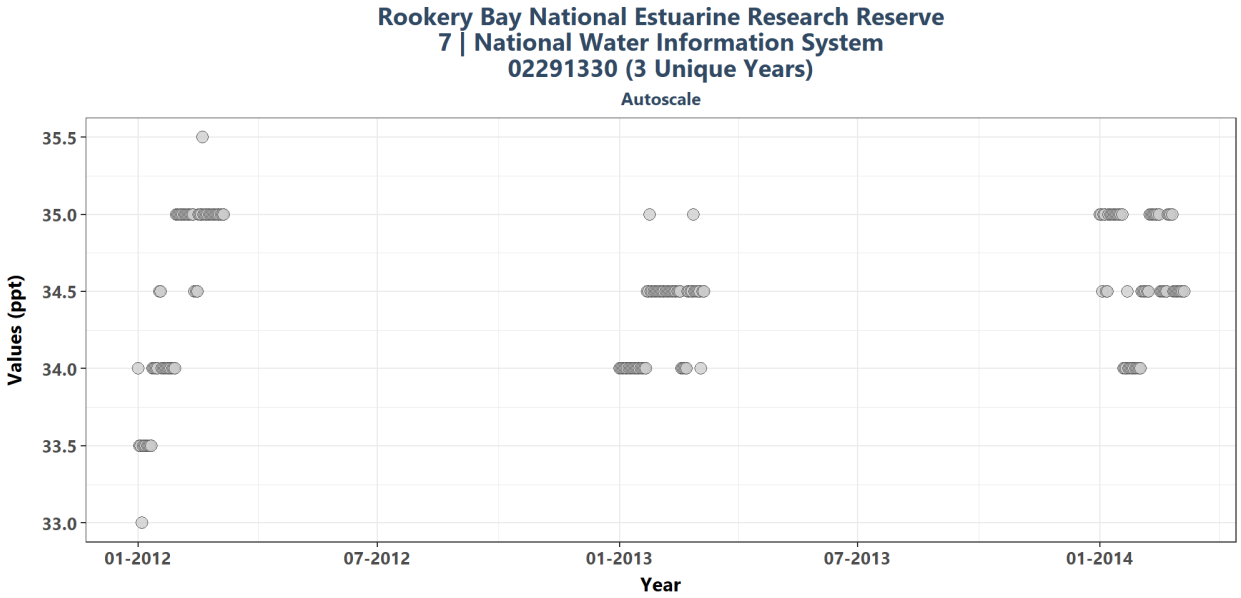

Estero Bay Aquatic Preserve
474 | Estero Bay Aquatic Preserve Continuous Water Quality Monitoring
EB04 (1 Unique Years)
Autoscale
Values (ppt)
30
20
10
06-2021
07-2021
08-2021
09-2021
10-2021
Year

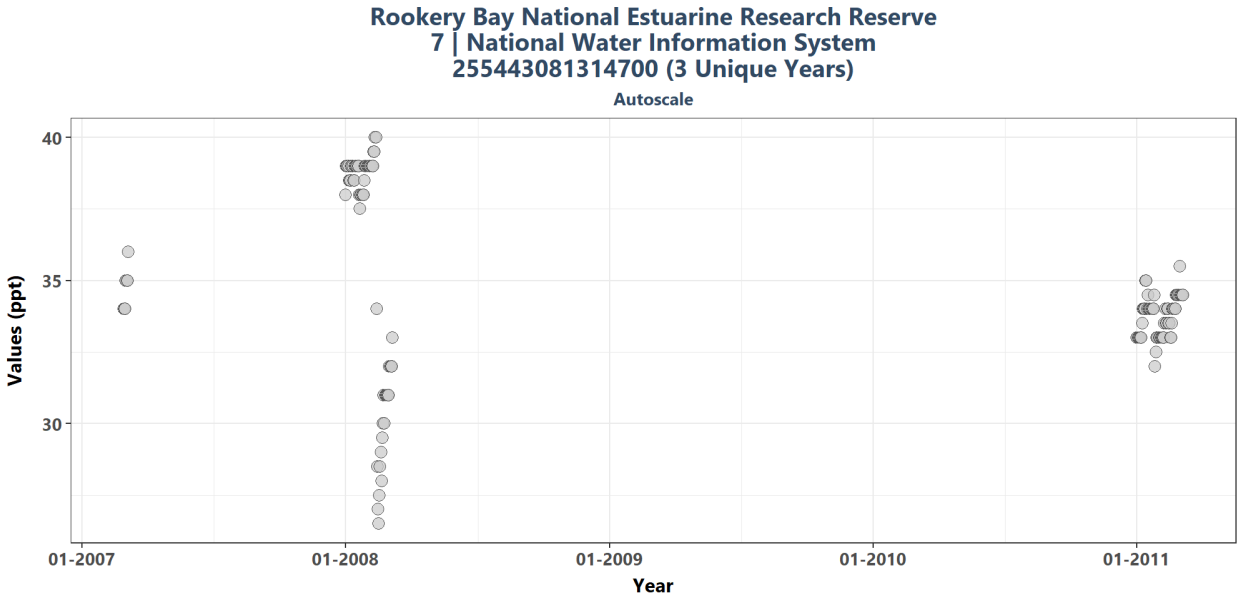

Gasparilla Sound-Charlotte Harbor Aquatic Preserve
7 | National Water Information System
02293252 (3 Unique Years)
Autoscale
Values (ppt)
32.5
30.0
27.5
25.0
01-2012
07-2012
01-2013
07-2013
01-2014
Year

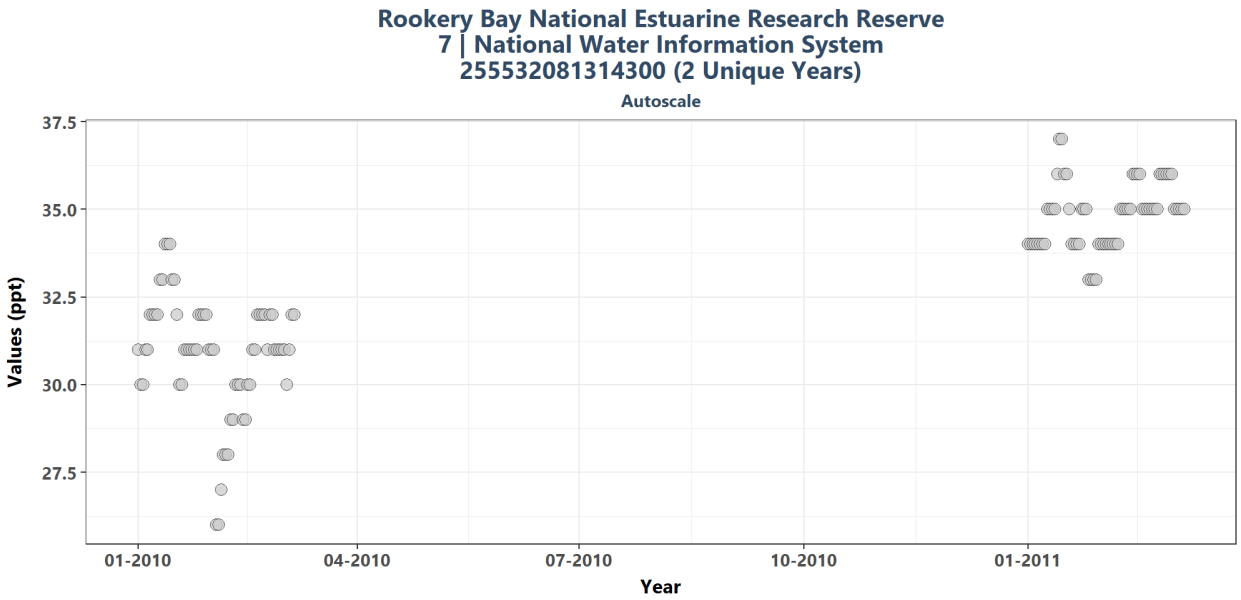

Gasparilla Sound-Charlotte Harbor Aquatic Preserve
7 | National Water Information System
02293254 (3 Unique Years)
Autoscale
Values (ppt)
32
30
28
26
24
01-2012
07-2012
01-2013
07-2013
01-2014
Year

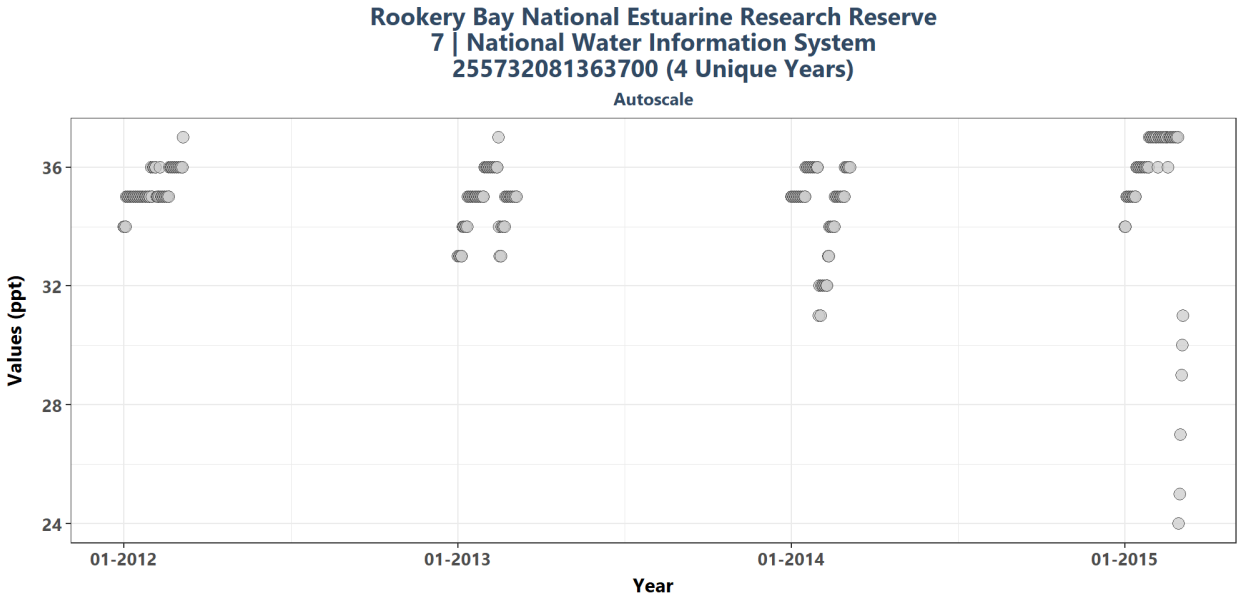

Pine Island Sound Aquatic Preserve
7 | National Water Information System
02293249 (4 Unique Years)
Autoscale
Values (ppt)
35
34
33
32
31
30
01-2011
01-2012
01-2013
Year

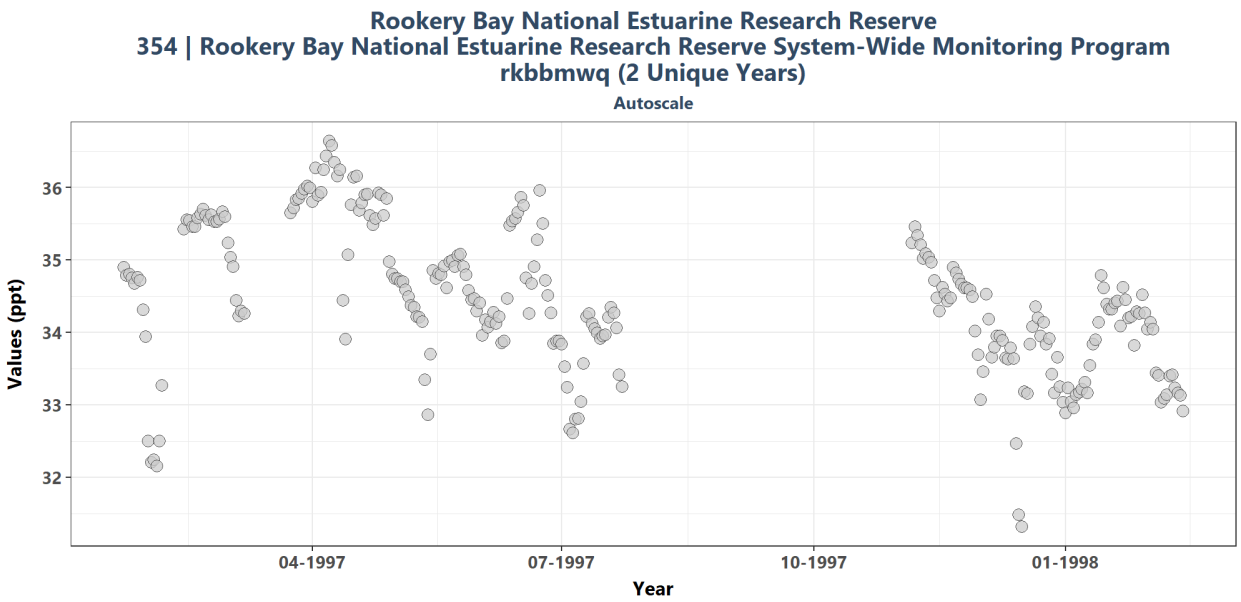

Rookery Bay Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbbmwq (2 Unique Years)
Autoscale
Values (ppt)
36
35
34
33
32
04-1997
07-1997
10-1997
01-1998
Year

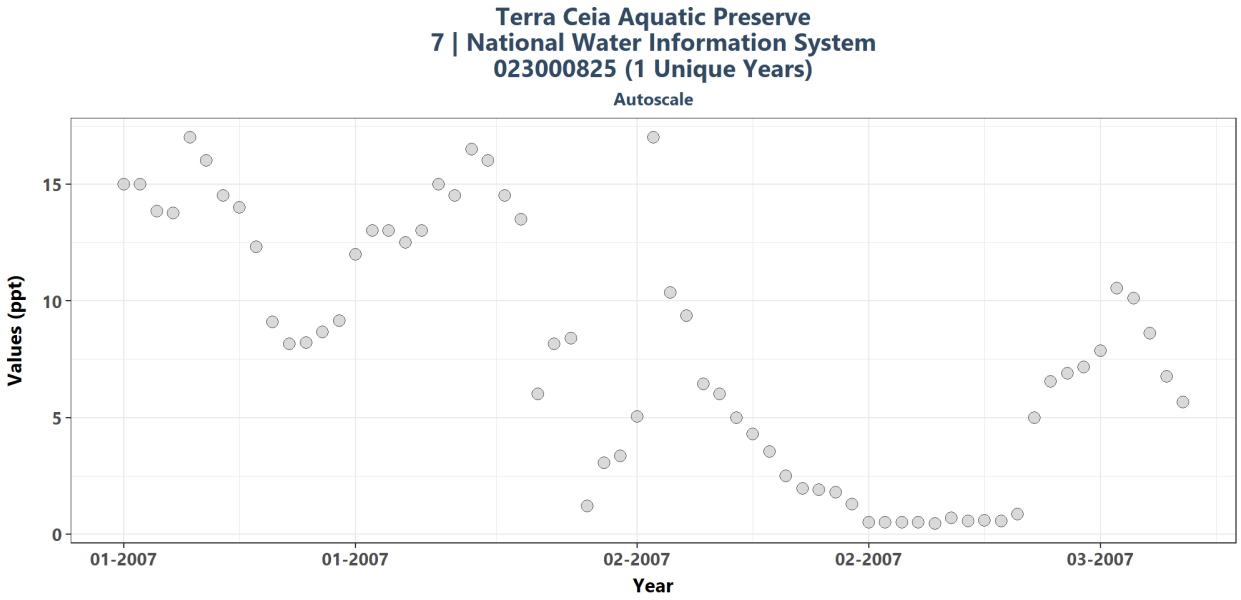

Rookery Bay National Estuarine Research Reserve
7 | National Water Information System
02291330 (3 Unique Years)
Autoscale
Values (ppt)
35.5
35.0
34.5
34.0
33.5
33.0
01-2012
07-2012
01-2013
07-2013
01-2014
Year

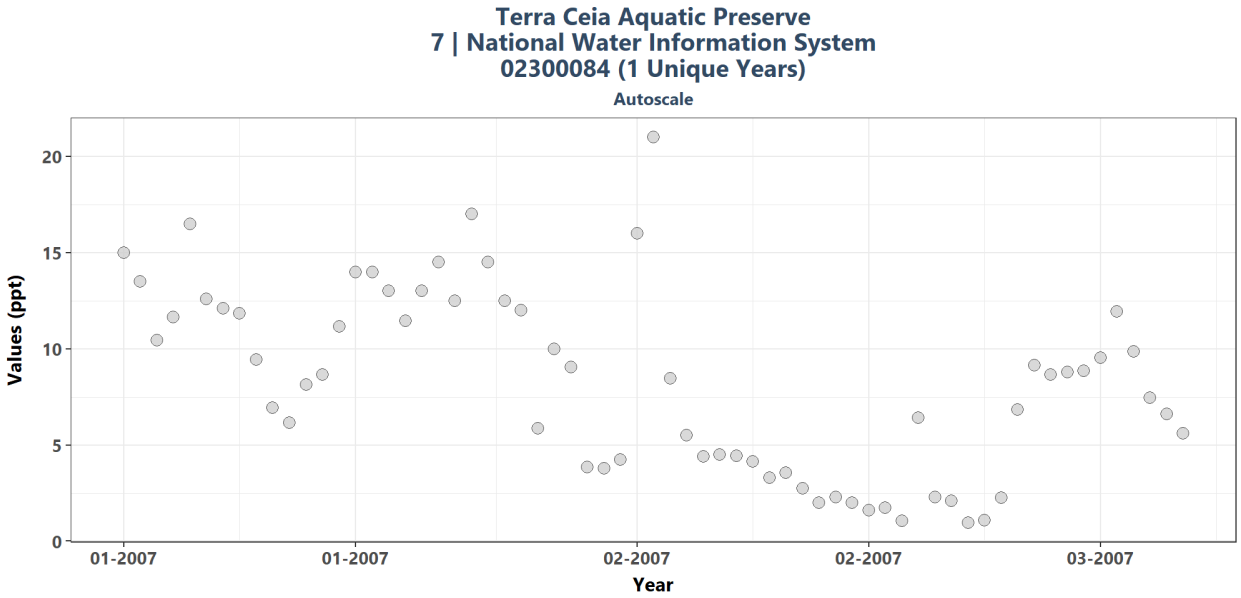

Rookery Bay National Estuarine Research Reserve
7 | National Water Information System
255443081314700 (3 Unique Years)
Autoscale
Values (ppt)
40
35
30
01-2007
01-2008
01-2009
01-2010
01-2011
Year

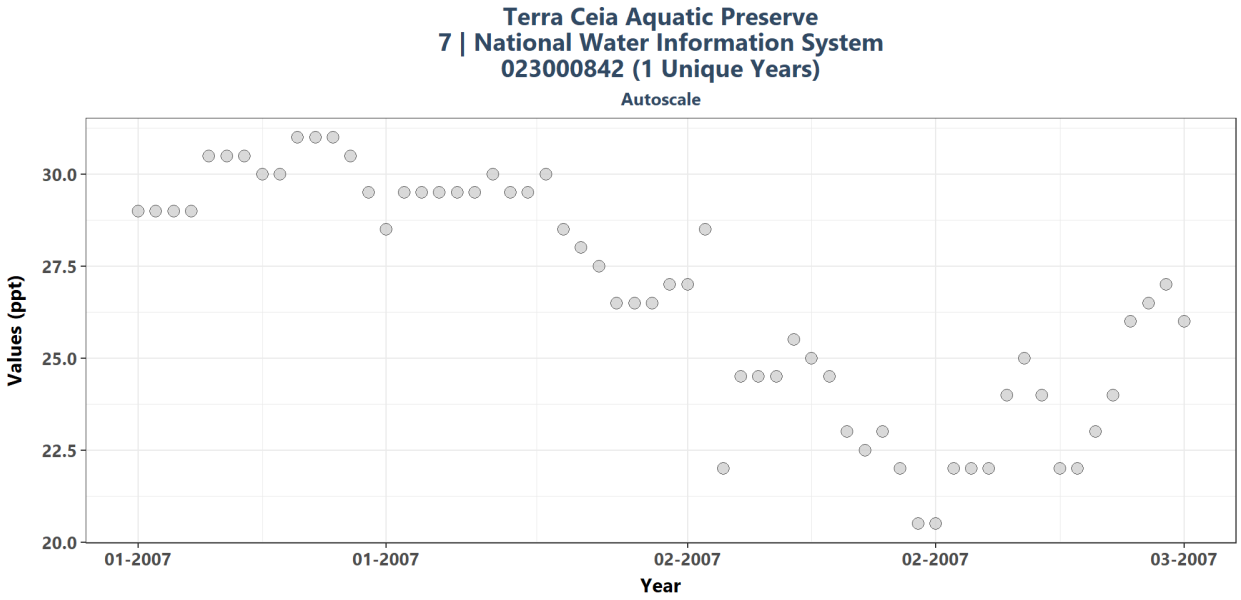

Rookery Bay National Estuarine Research Reserve
7 | National Water Information System
255532081314300 (2 Unique Years)
Autoscale
Values (ppt)
37.5
35.0
32.5
30.0
27.5
01-2010
04-2010
07-2010
10-2010
01-2011
Year


Rookery Bay National Estuarine Research Reserve
7 | National Water Information System
255732081363700 (4 Unique Years)
Autoscale
Values (ppt)
36
32
28
24
01-2012
01-2013
01-2014
01-2015
Year


Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbbmwq (2 Unique Years)
Autoscale
Values (ppt)
36
35
34
33
32
04-1997
07-1997
10-1997
01-1998
Year


Terra Ceia Aquatic Preserve
7 | National Water Information System
023000825 (1 Unique Years)
Autoscale
Values (ppt)
0
5
10
15
01-2007
01-2007
02-2007
02-2007
03-2007
Year


Terra Ceia Aquatic Preserve
7 | National Water Information System
02300084 (1 Unique Years)
Autoscale
Values (ppt)
0
5
10
15
20
01-2007
01-2007
02-2007
02-2007
03-2007
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

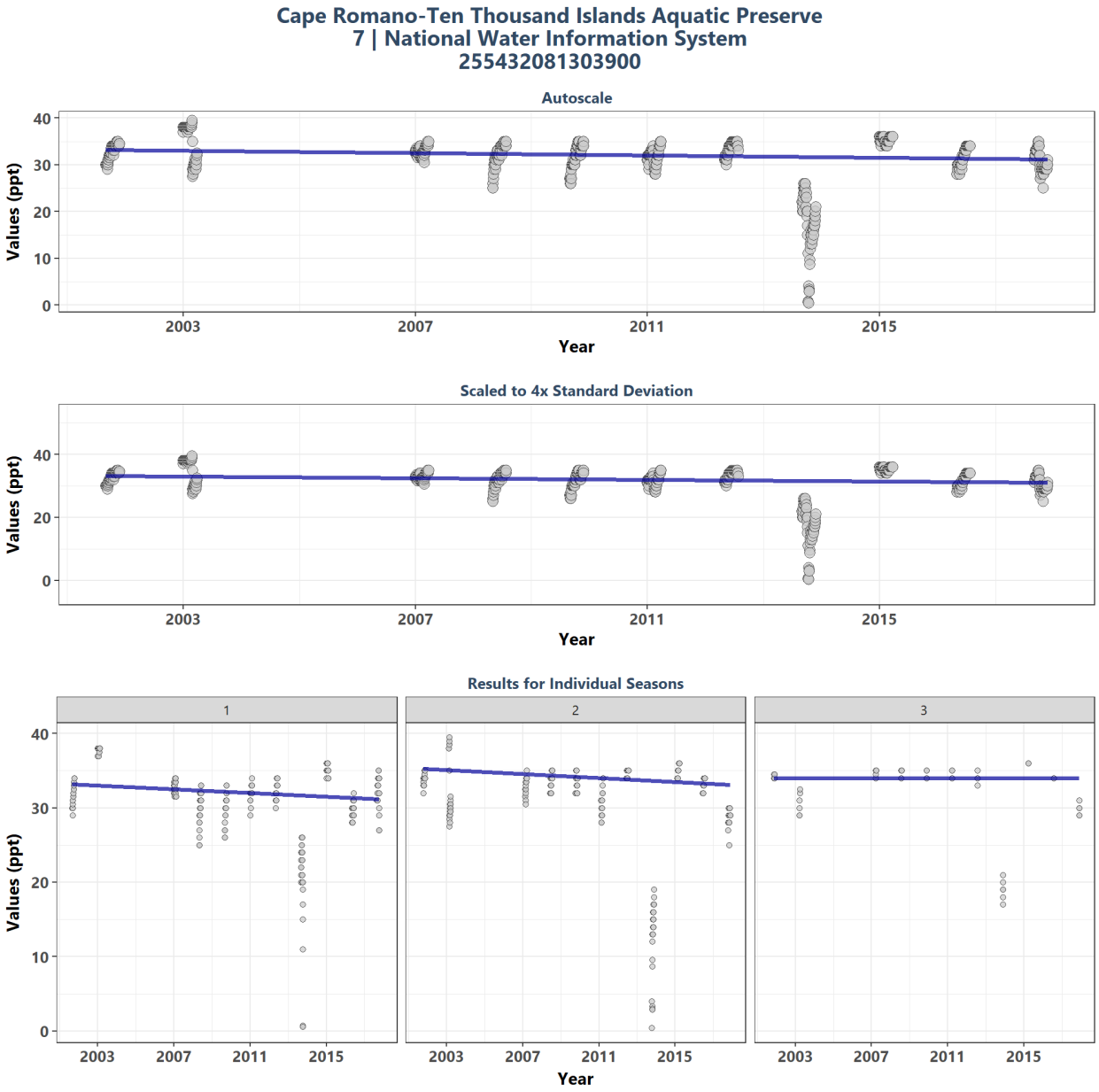
1. Use the data set that only has `Use_In_Analysis` of `TRUE` for the desired monitoring location
2. Determine the earliest and latest year of the data to create x-axis scale and intervals
3. Determine the minimum, mean, and standard deviation for the data to be used for y-axis scales
    - Excludes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

      # 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]
```

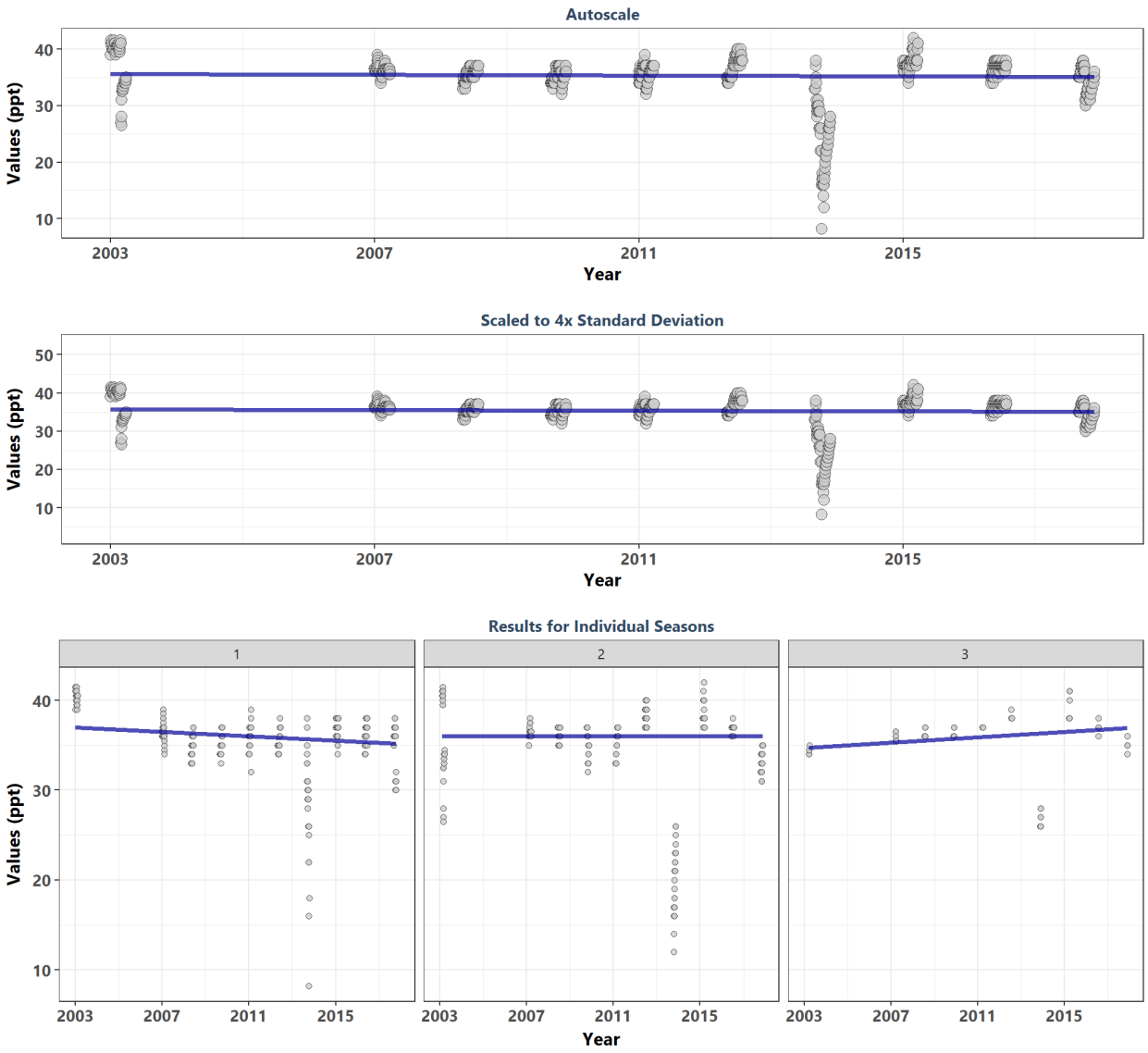
```
# x_end1 <- relyear_dd_upper
# y_end1 <- relyear_dd_upper * KT.Stats[ManagedAreaName==MA_Include[i] & Season == "All", SennSlo

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[Mon
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_
                                                                                     y = relyear_
                                                                                     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)
   }
}
```



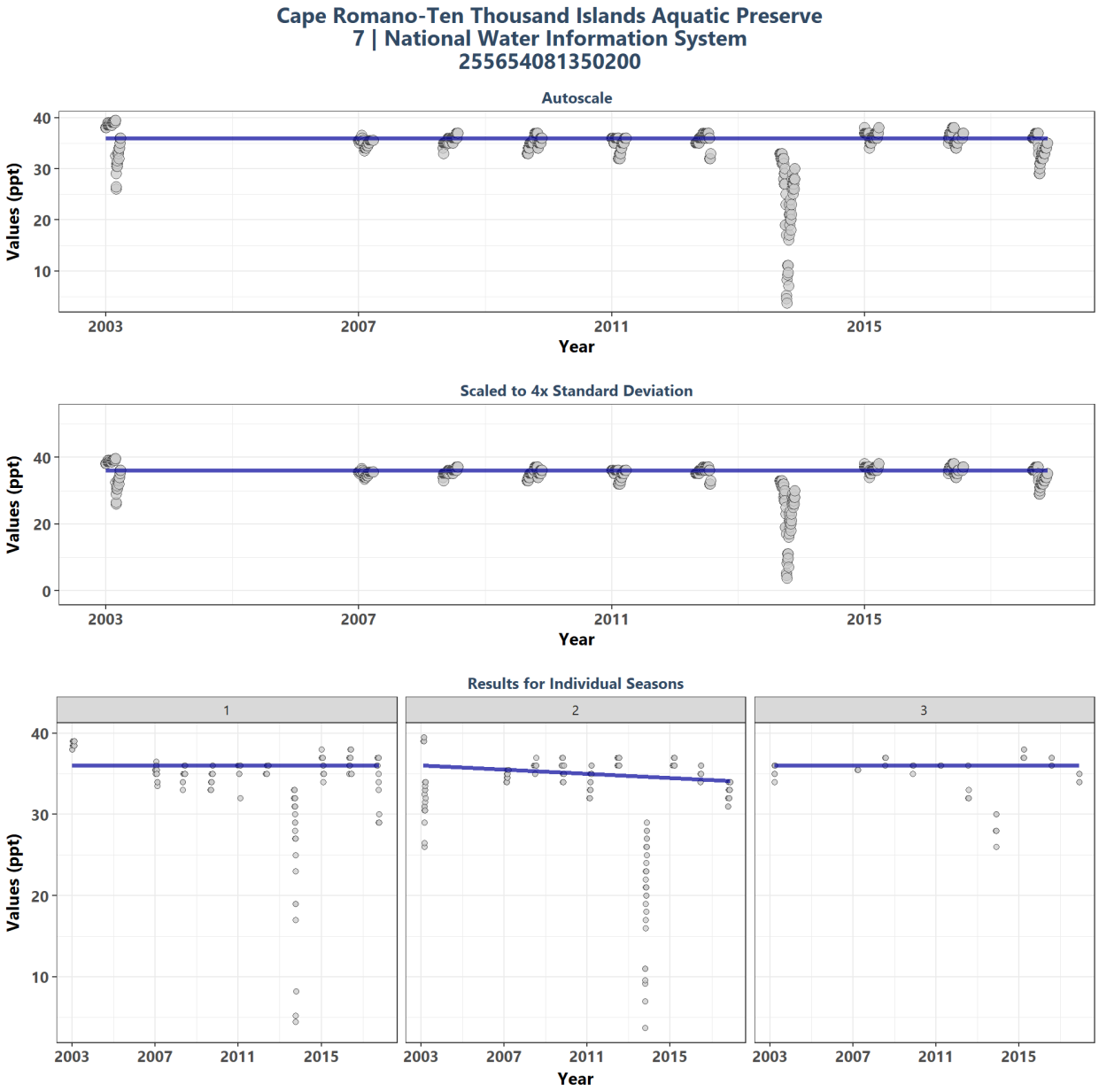
Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	718	33	-0.1307	-0.1667	34.3333	-5.3	0.0000	5.5	0.0652	-1
1	341	32	-0.1182	-0.1667	34.3333	-3.3	0.0010	NA	NA	-1
2	311	34	-0.1556	-0.1818	36.5455	-4.2	0.0000	NA	NA	-1
3	66	34	-0.0779	0.0000	34.0000	-1.0	0.3383	NA	NA	-1

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



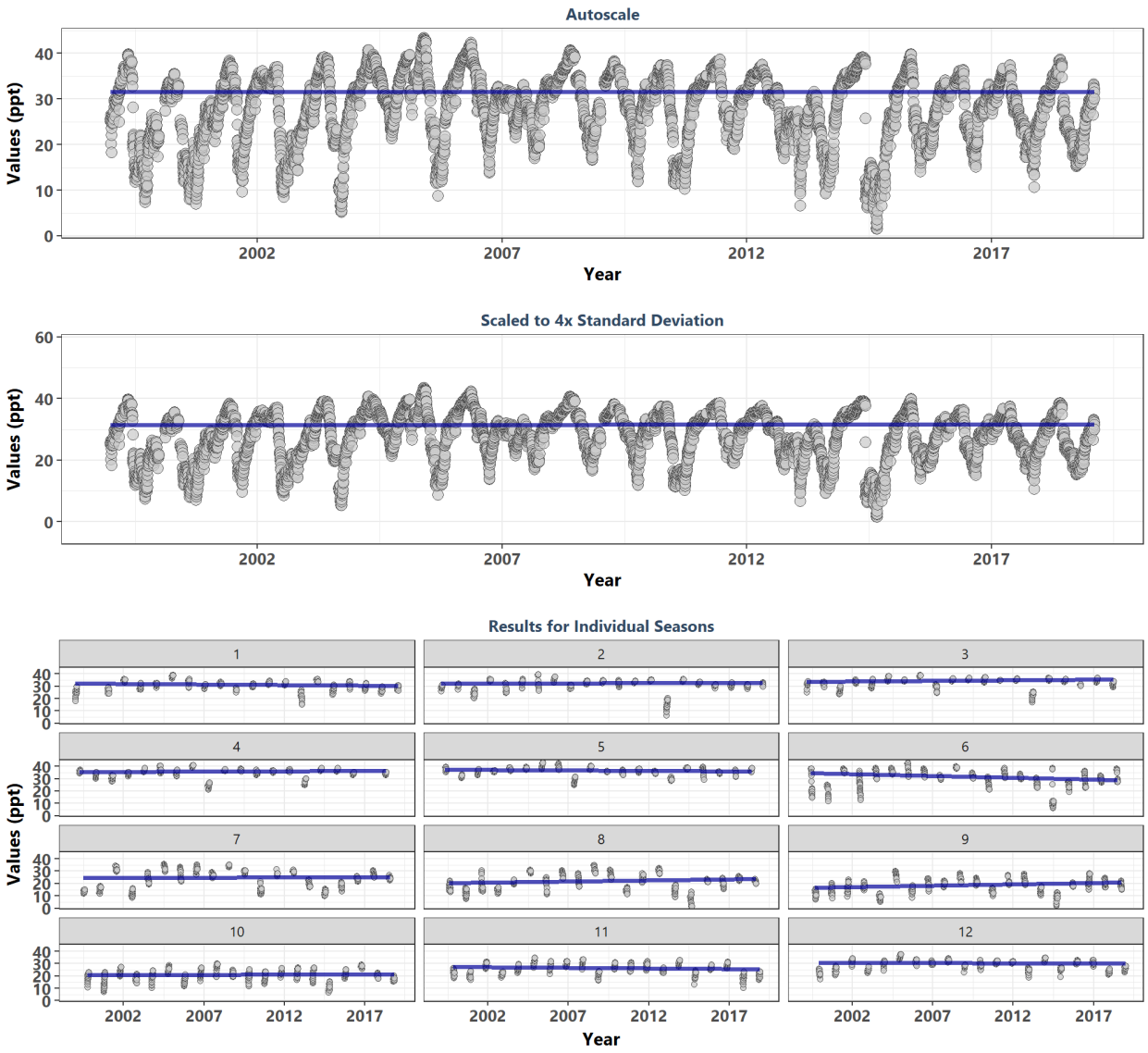
Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	645	36	-0.0636	-0.0455	36.0000	-3.2	0.0013	16.5	0.0003	-1
1	307	36	-0.1280	-0.1583	38.2167	-3.4	0.0007	NA	NA	-1
2	278	36	-0.0492	0.0000	36.0000	-1.2	0.2163	NA	NA	-1
3	60	36	0.1994	0.2000	33.1000	2.3	0.0222	NA	NA	1

[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	626	35.5	-0.0501	0.0000	36.0000	-1.8	0.0718	5.4	0.0667	0
1	301	36.0	0.0033	0.0000	36.0000	0.1	0.9308	NA	NA	0
2	265	35.0	-0.1178	-0.1667	37.3333	-2.9	0.0037	NA	NA	0
3	60	36.0	-0.0186	0.0000	36.0000	-0.2	0.8340	NA	NA	0

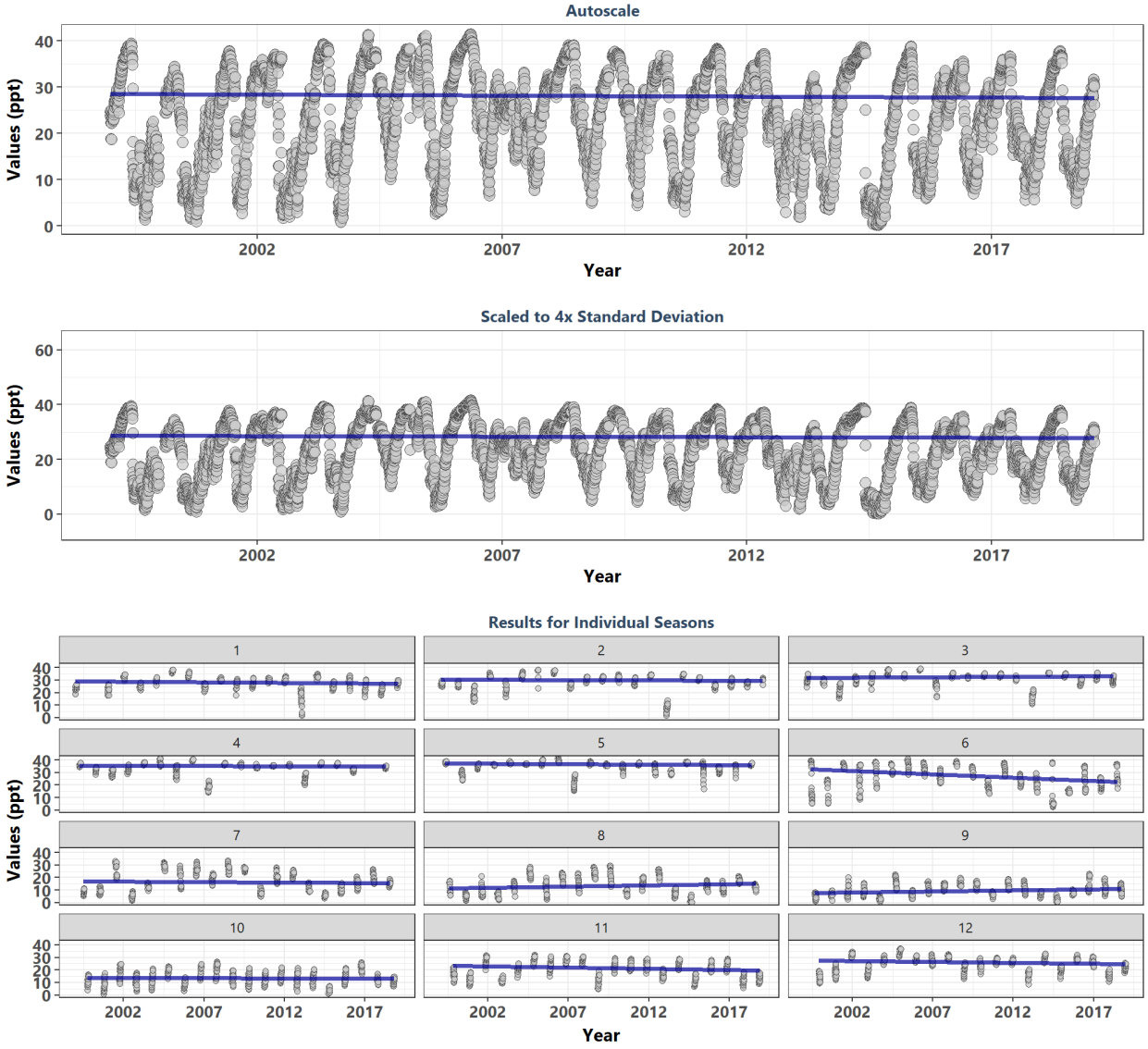
[a] $p < 0.00005$ appear as 0 due to rounding



Season	N	Median	tau	Slope	Int.	z	p__z	chi__sq	p__chi__sq	Trend
All	6826	29.50	0.0051	0.0044	31.4974	0.5	0.6492	120.1	0	0
1	587	31.10	-0.0735	-0.0792	32.1290	-2.7	0.0077	NA	NA	0
2	555	32.41	0.0257	0.0251	32.1072	0.9	0.3641	NA	NA	0
3	550	34.42	0.1242	0.0932	33.3901	4.4	0.0000	NA	NA	0
4	506	35.80	0.0853	0.0643	35.0925	2.9	0.0041	NA	NA	0
5	548	36.69	-0.0982	-0.0737	37.5729	-3.4	0.0006	NA	NA	0
6	563	31.62	-0.1437	-0.2942	35.1487	-5.1	0.0000	NA	NA	0
7	588	24.87	0.0029	0.0063	24.8008	0.1	0.9175	NA	NA	0
8	574	21.73	0.0858	0.1695	19.8612	3.1	0.0021	NA	NA	0
9	556	18.35	0.1527	0.2195	15.9341	5.4	0.0000	NA	NA	0
10	603	20.97	0.0244	0.0308	20.6301	0.9	0.3691	NA	NA	0
11	594	26.60	-0.0681	-0.0929	27.7102	-2.5	0.0129	NA	NA	0
12	602	30.43	-0.0385	-0.0381	30.8877	-1.4	0.1573	NA	NA	0

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

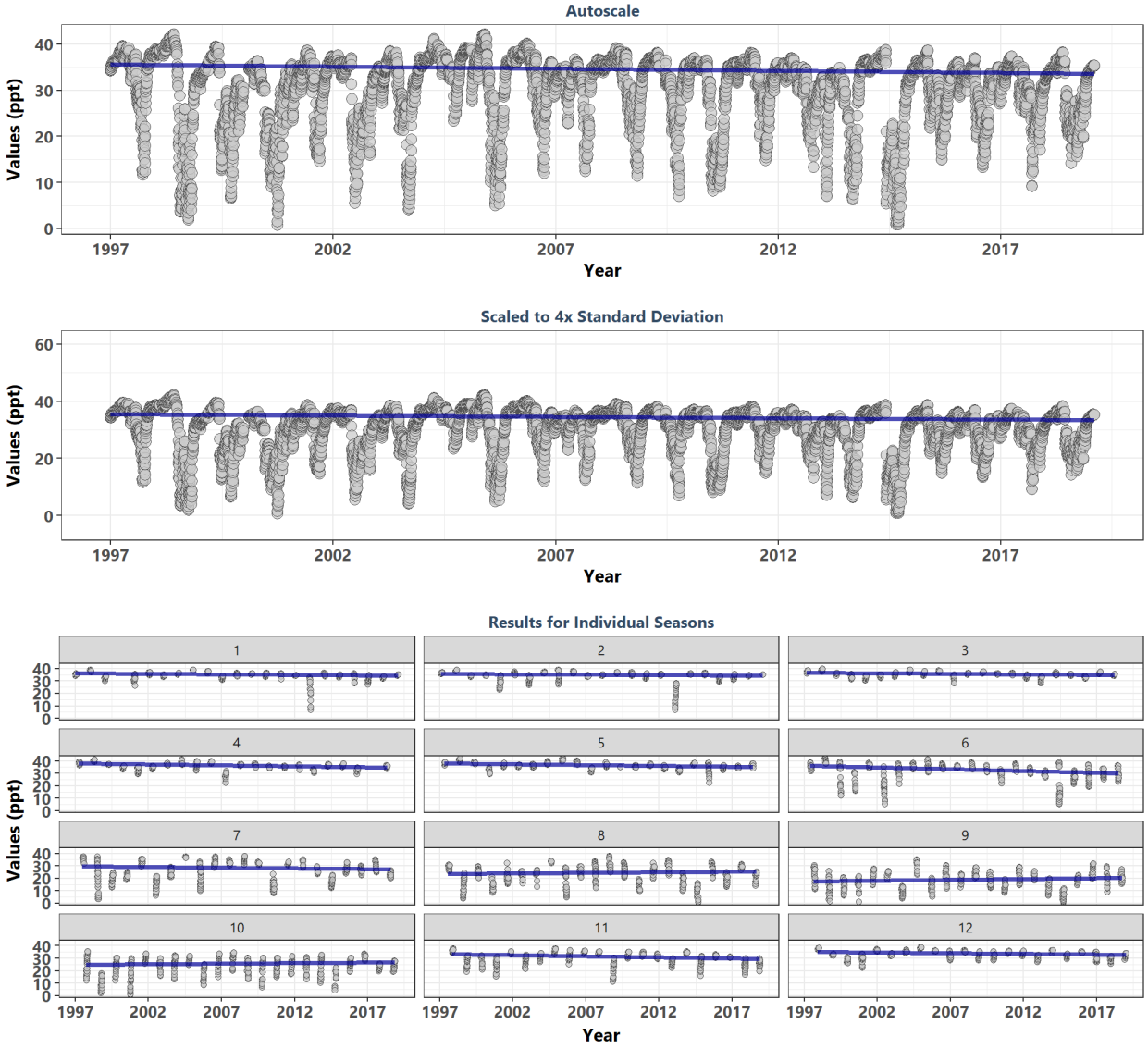
Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfuwq



Season	N	Median	tau	Slope	Int.	z	p__z	chi__sq	p__chi__sq	Trend
All	6984	24.79	-0.0295	-0.0420	28.6541	-3.6	0.0003	124.1	0	-1
1	604	28.18	-0.0648	-0.0924	29.3380	-2.4	0.0170	NA	NA	-1
2	544	29.84	-0.0312	-0.0486	30.4281	-1.1	0.2764	NA	NA	-1
3	545	32.59	0.0732	0.0786	31.7304	2.6	0.0105	NA	NA	1
4	546	35.35	-0.0146	-0.0134	35.4966	-0.5	0.6092	NA	NA	-1
5	568	36.55	-0.1106	-0.0778	37.4802	-3.9	0.0001	NA	NA	-1
6	556	27.57	-0.2095	-0.5485	34.1527	-7.4	0.0000	NA	NA	-1
7	602	16.19	-0.0261	-0.0632	16.8821	-1.0	0.3381	NA	NA	-1
8	605	13.08	0.0905	0.1798	11.1033	3.3	0.0009	NA	NA	1
9	600	9.35	0.1215	0.1602	7.5117	4.5	0.0000	NA	NA	1
10	619	13.46	-0.0107	-0.0176	13.6561	-0.4	0.6890	NA	NA	-1
11	592	21.44	-0.0955	-0.1845	23.6523	-3.5	0.0005	NA	NA	-1
12	603	26.30	-0.0852	-0.1516	27.9703	-3.1	0.0017	NA	NA	-1

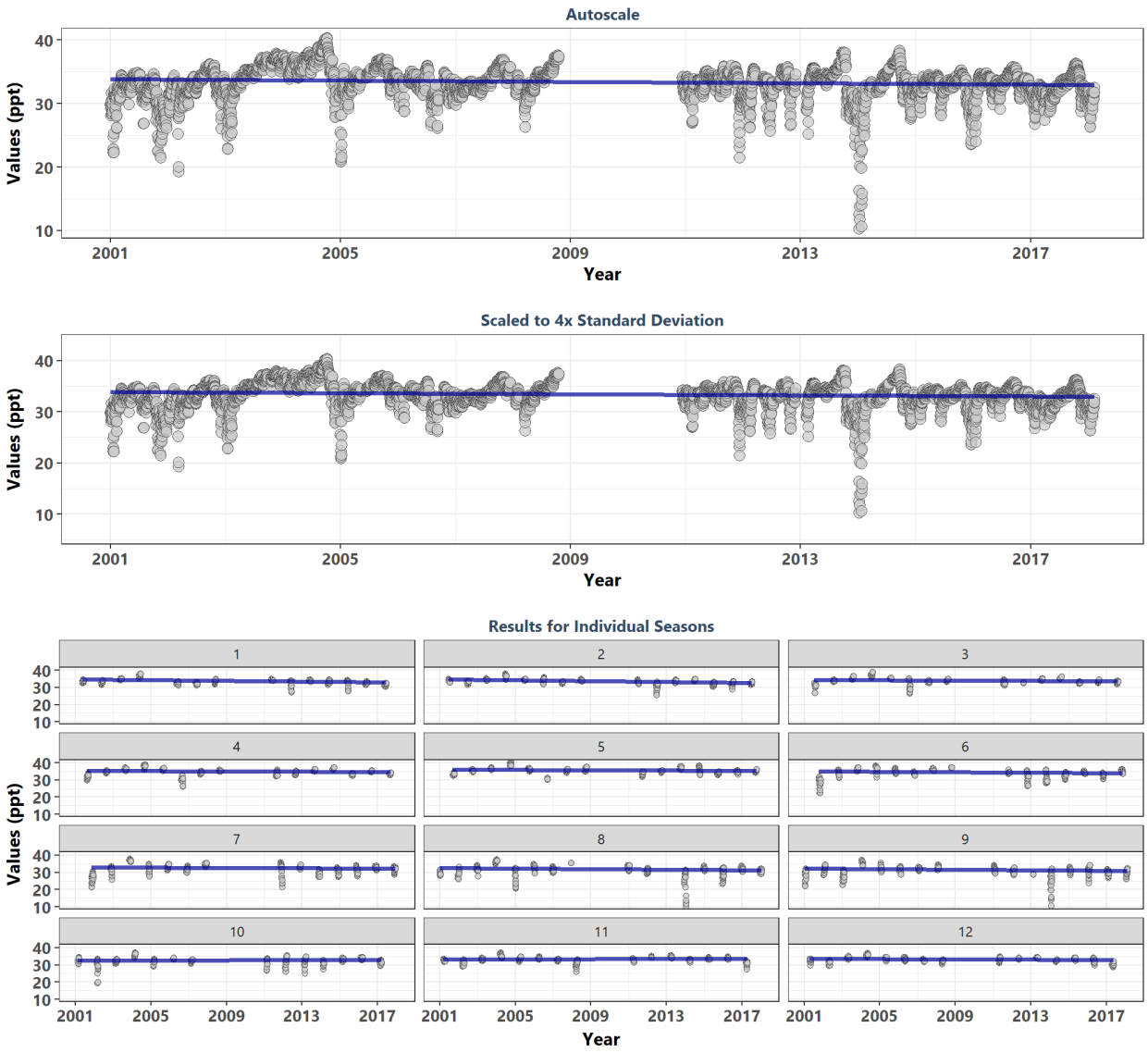
[a] $p < 0.00005$ appear as 0 due to rounding

Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbmbwq



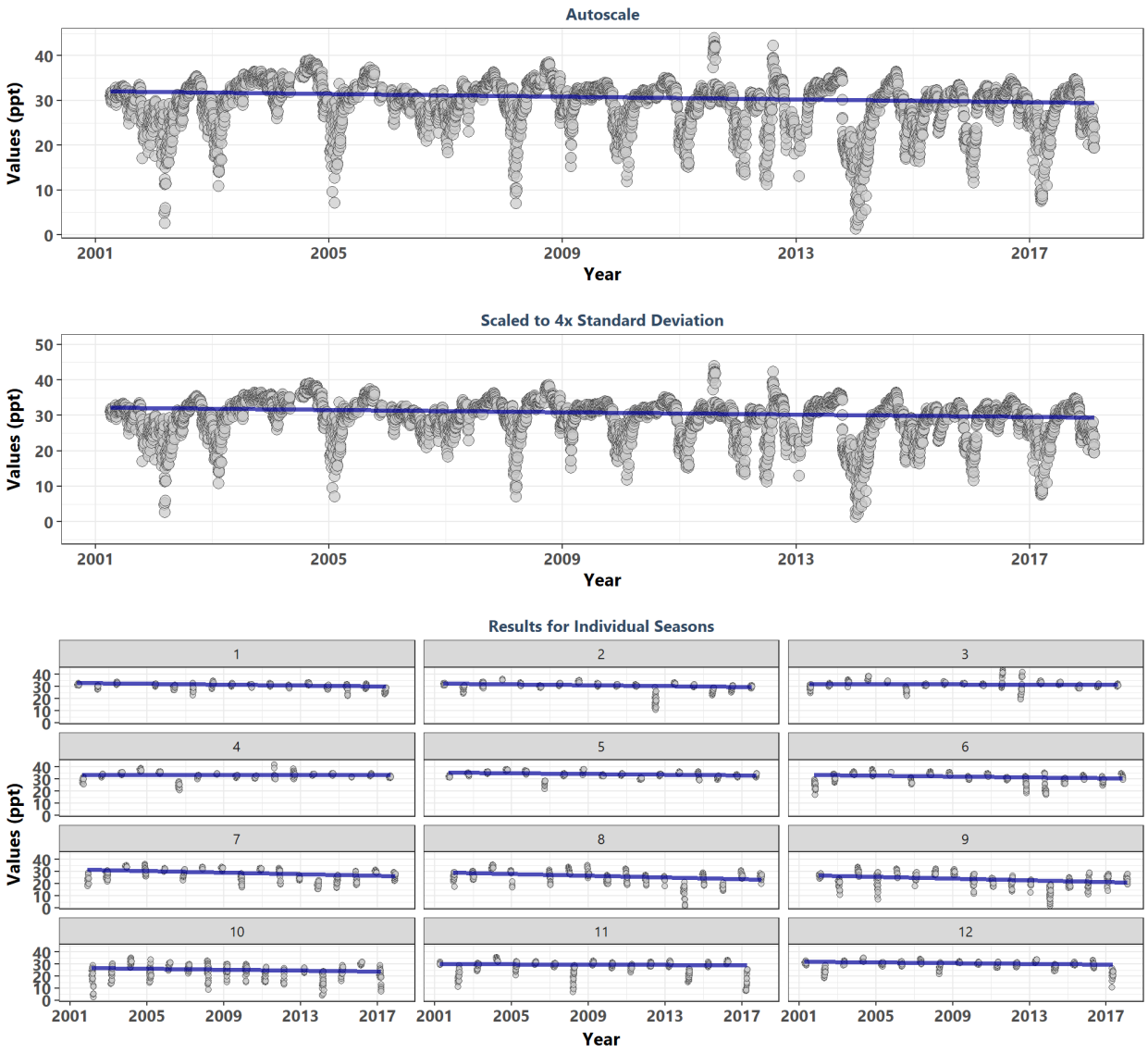
Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	7514	33.24	-0.1148	-0.0916	35.5562	-14.7	0.0000	214.7	0	-1
1	669	35.04	-0.2072	-0.0924	36.1527	-8.0	0.0000	NA	NA	-1
2	595	35.08	-0.1436	-0.0717	35.8718	-5.2	0.0000	NA	NA	-1
3	656	35.74	-0.1823	-0.0883	36.7078	-7.0	0.0000	NA	NA	-1
4	580	36.46	-0.2475	-0.1341	37.7969	-8.9	0.0000	NA	NA	-1
5	589	36.74	-0.2308	-0.1442	38.1827	-8.4	0.0000	NA	NA	-1
6	601	33.44	-0.2167	-0.2928	36.3715	-8.0	0.0000	NA	NA	-1
7	624	28.46	-0.0511	-0.1010	29.4667	-1.9	0.0561	NA	NA	-1
8	653	24.43	0.0536	0.0954	23.3768	2.0	0.0404	NA	NA	1
9	617	19.02	0.0718	0.1195	17.7025	2.7	0.0076	NA	NA	1
10	653	25.68	0.0574	0.0929	24.7500	2.2	0.0279	NA	NA	1
11	610	31.61	-0.1497	-0.1735	33.3460	-5.5	0.0000	NA	NA	-1
12	667	34.13	-0.1546	-0.1115	35.2406	-6.0	0.0000	NA	NA	-1

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



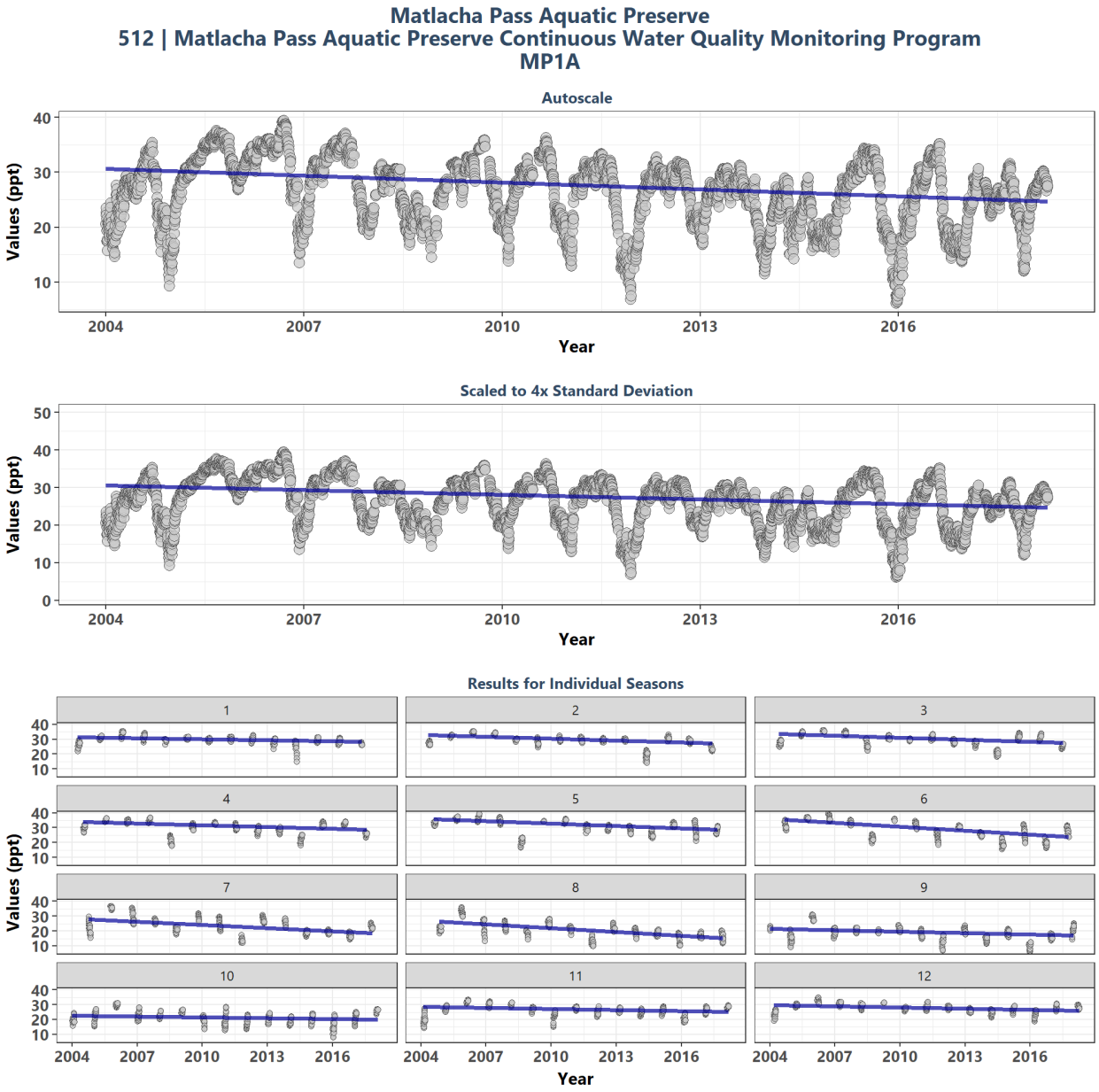
Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	4922	33.44	-0.1043	-0.0556	33.8662	-11.1	0.0000	102.7	0	-1
1	427	33.60	-0.2565	-0.1001	34.7035	-7.9	0.0000	NA	NA	-1
2	421	33.76	-0.3023	-0.1227	34.7417	-9.3	0.0000	NA	NA	-1
3	394	33.94	-0.0983	-0.0454	34.3019	-2.9	0.0035	NA	NA	-1
4	450	35.06	-0.0667	-0.0366	35.3499	-2.1	0.0340	NA	NA	-1
5	422	35.45	-0.0900	-0.0486	35.9900	-2.8	0.0056	NA	NA	-1
6	403	34.41	-0.0874	-0.0651	34.9344	-2.6	0.0086	NA	NA	-1
7	408	32.43	-0.0456	-0.0362	32.8252	-1.4	0.1683	NA	NA	-1
8	390	31.72	-0.1343	-0.0853	32.5743	-4.0	0.0001	NA	NA	-1
9	422	31.39	-0.1312	-0.0853	32.2440	-4.0	0.0001	NA	NA	-1
10	361	32.64	0.0335	0.0154	32.4870	1.0	0.3412	NA	NA	1
11	409	33.30	0.0397	0.0177	33.1917	1.2	0.2293	NA	NA	1
12	415	33.21	-0.0880	-0.0372	33.4306	-2.7	0.0073	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	5423	30.57	-0.1706	-0.1590	32.1374	-18.7	0.0000	93.5	0	-1
1	413	31.33	-0.2693	-0.1560	32.8888	-8.2	0.0000	NA	NA	-1
2	431	31.06	-0.2478	-0.1756	32.8129	-7.7	0.0000	NA	NA	-1
3	488	31.88	-0.0629	-0.0429	32.3094	-2.1	0.0376	NA	NA	-1
4	422	33.38	-0.0065	-0.0047	33.4177	-0.2	0.8417	NA	NA	-1
5	483	33.94	-0.2236	-0.1503	35.4441	-7.4	0.0000	NA	NA	-1
6	451	31.81	-0.1877	-0.2070	33.8786	-6.0	0.0000	NA	NA	-1
7	460	28.83	-0.2485	-0.3486	31.9653	-8.0	0.0000	NA	NA	-1
8	430	26.29	-0.2702	-0.3795	29.7026	-8.4	0.0000	NA	NA	-1
9	447	23.95	-0.2213	-0.3738	27.3156	-7.0	0.0000	NA	NA	-1
10	441	25.21	-0.0988	-0.1880	26.7101	-3.1	0.0019	NA	NA	-1
11	474	29.54	-0.0485	-0.0555	29.9792	-1.6	0.1140	NA	NA	-1
12	483	30.52	-0.1759	-0.1372	31.7558	-5.8	0.0000	NA	NA	-1

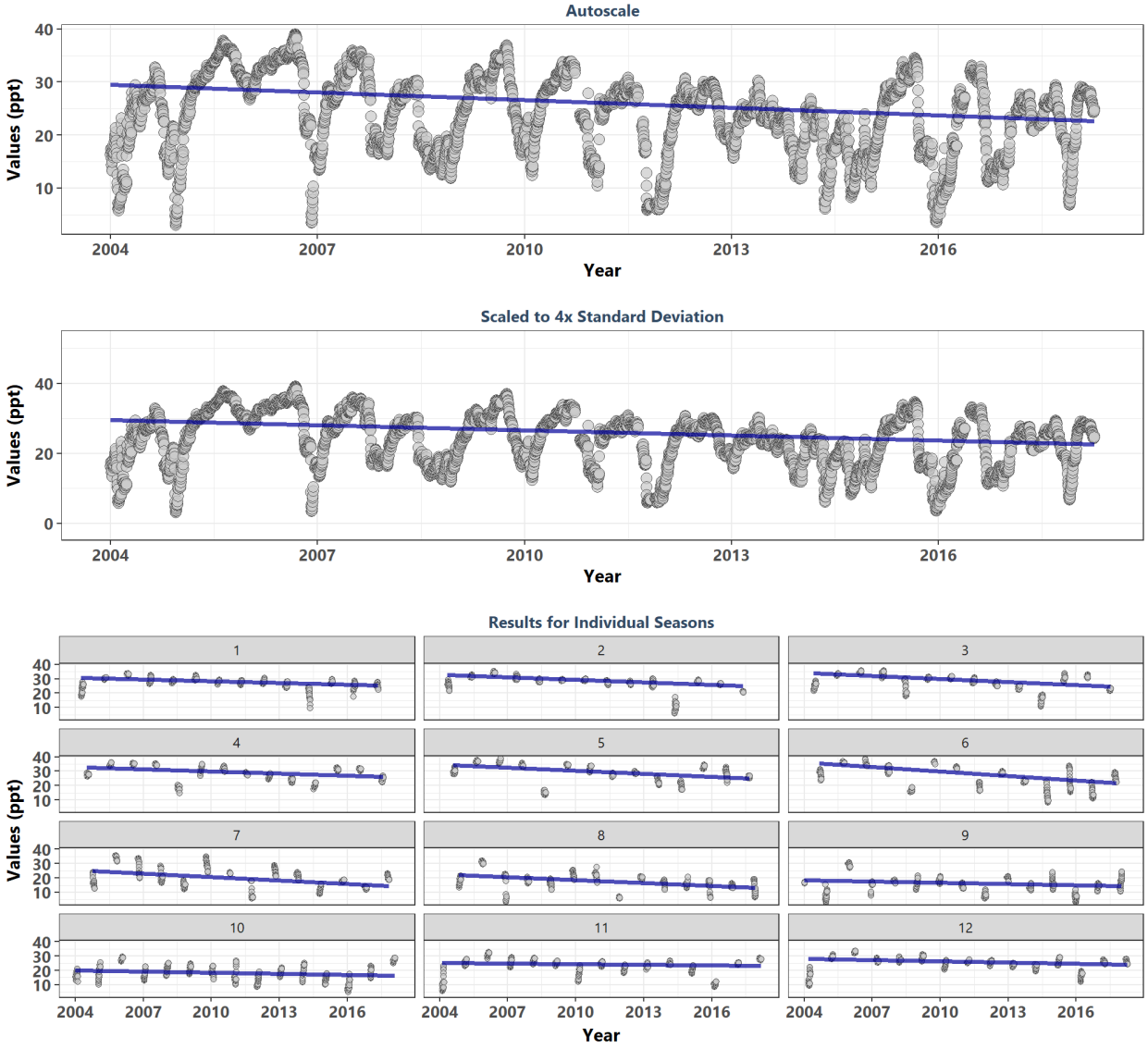
[a] $p < 0.00005$ appear as 0 due to rounding



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	4872	27.36	-0.2793	-0.4148	30.6112	-28.9	0.0000	107.7	0	-1
1	420	29.81	-0.2512	-0.2201	31.3540	-7.7	0.0000	NA	NA	-1
2	374	29.63	-0.3181	-0.4177	32.9729	-9.2	0.0000	NA	NA	-1
3	408	30.07	-0.2707	-0.4654	33.7964	-8.2	0.0000	NA	NA	-1
4	403	30.80	-0.2696	-0.3995	33.9990	-8.1	0.0000	NA	NA	-1
5	417	31.77	-0.3493	-0.5302	36.0158	-10.7	0.0000	NA	NA	-1
6	385	28.98	-0.4268	-0.8854	36.0677	-12.5	0.0000	NA	NA	-1
7	370	22.66	-0.3305	-0.7291	28.4952	-9.5	0.0000	NA	NA	-1
8	411	20.41	-0.4715	-0.8531	27.2396	-14.3	0.0000	NA	NA	-1
9	381	18.99	-0.1726	-0.3370	21.6871	-5.0	0.0000	NA	NA	-1
10	445	21.38	-0.1056	-0.1840	22.6705	-3.3	0.0009	NA	NA	-1
11	438	26.86	-0.1646	-0.2381	28.5220	-5.2	0.0000	NA	NA	-1
12	420	27.92	-0.2526	-0.2779	29.8685	-7.8	0.0000	NA	NA	-1

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

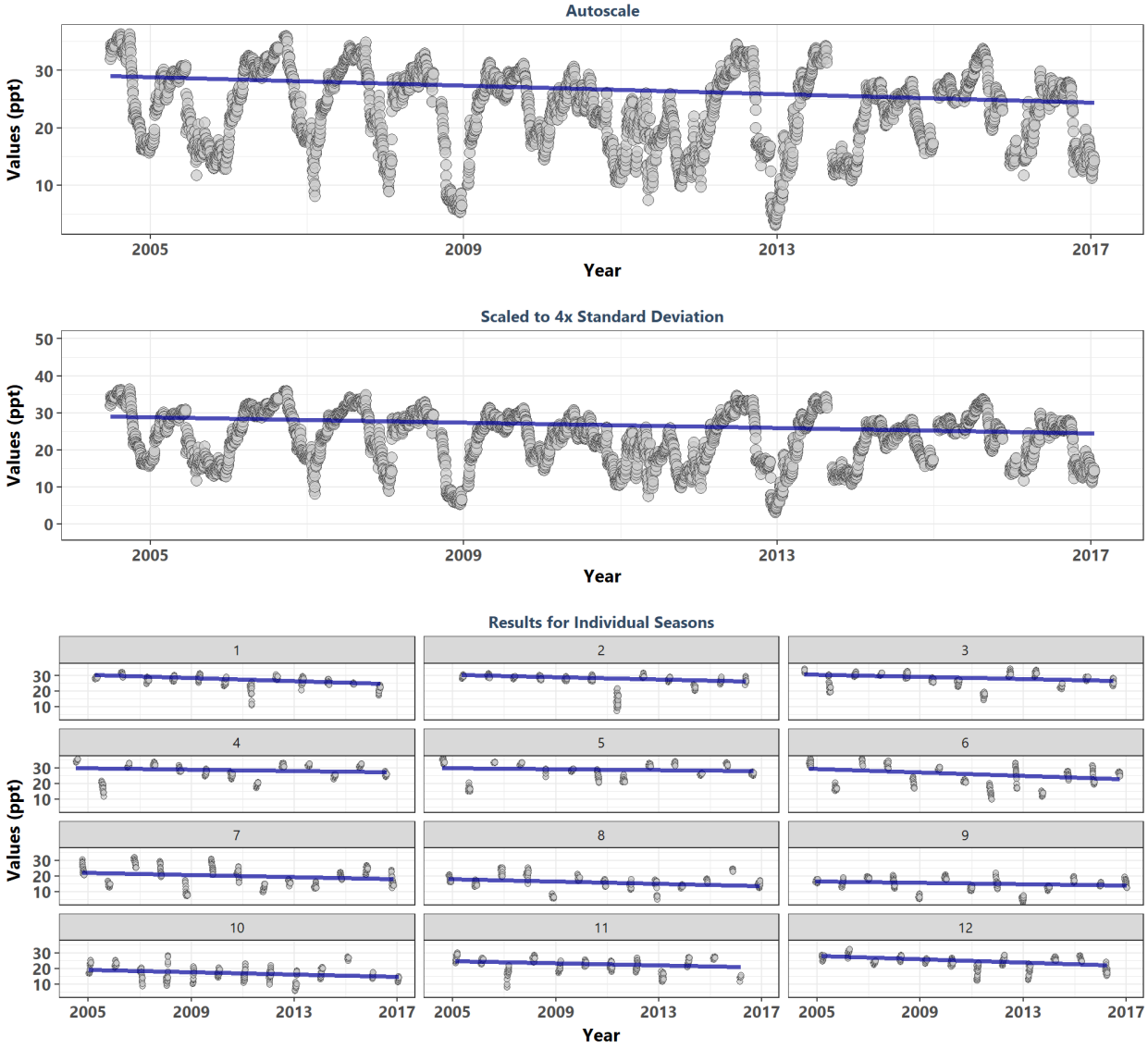
Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP2B



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	4965	24.95	-0.2792	-0.4849	29.5461	-28.9	0.0000	126.6	0	-1
1	433	27.70	-0.3723	-0.3955	30.8628	-11.6	0.0000	NA	NA	-1
2	364	28.65	-0.4659	-0.5747	32.6732	-13.3	0.0000	NA	NA	-1
3	400	29.30	-0.3264	-0.7146	34.3000	-9.8	0.0000	NA	NA	-1
4	416	29.40	-0.2981	-0.4769	32.7398	-9.1	0.0000	NA	NA	-1
5	417	29.58	-0.3333	-0.6943	34.4443	-10.2	0.0000	NA	NA	-1
6	376	27.68	-0.3696	-1.0452	36.0429	-10.7	0.0000	NA	NA	-1
7	406	19.23	-0.2440	-0.8047	25.6714	-7.4	0.0000	NA	NA	-1
8	402	18.04	-0.4022	-0.6746	22.7630	-12.1	0.0000	NA	NA	-1
9	412	16.04	-0.1448	-0.2993	18.4314	-4.4	0.0000	NA	NA	-1
10	447	18.08	-0.1186	-0.2549	19.8634	-3.8	0.0002	NA	NA	-1
11	433	24.28	-0.0980	-0.1691	25.4668	-3.1	0.0023	NA	NA	-1
12	459	26.12	-0.2333	-0.3016	28.2293	-7.5	0.0000	NA	NA	-1

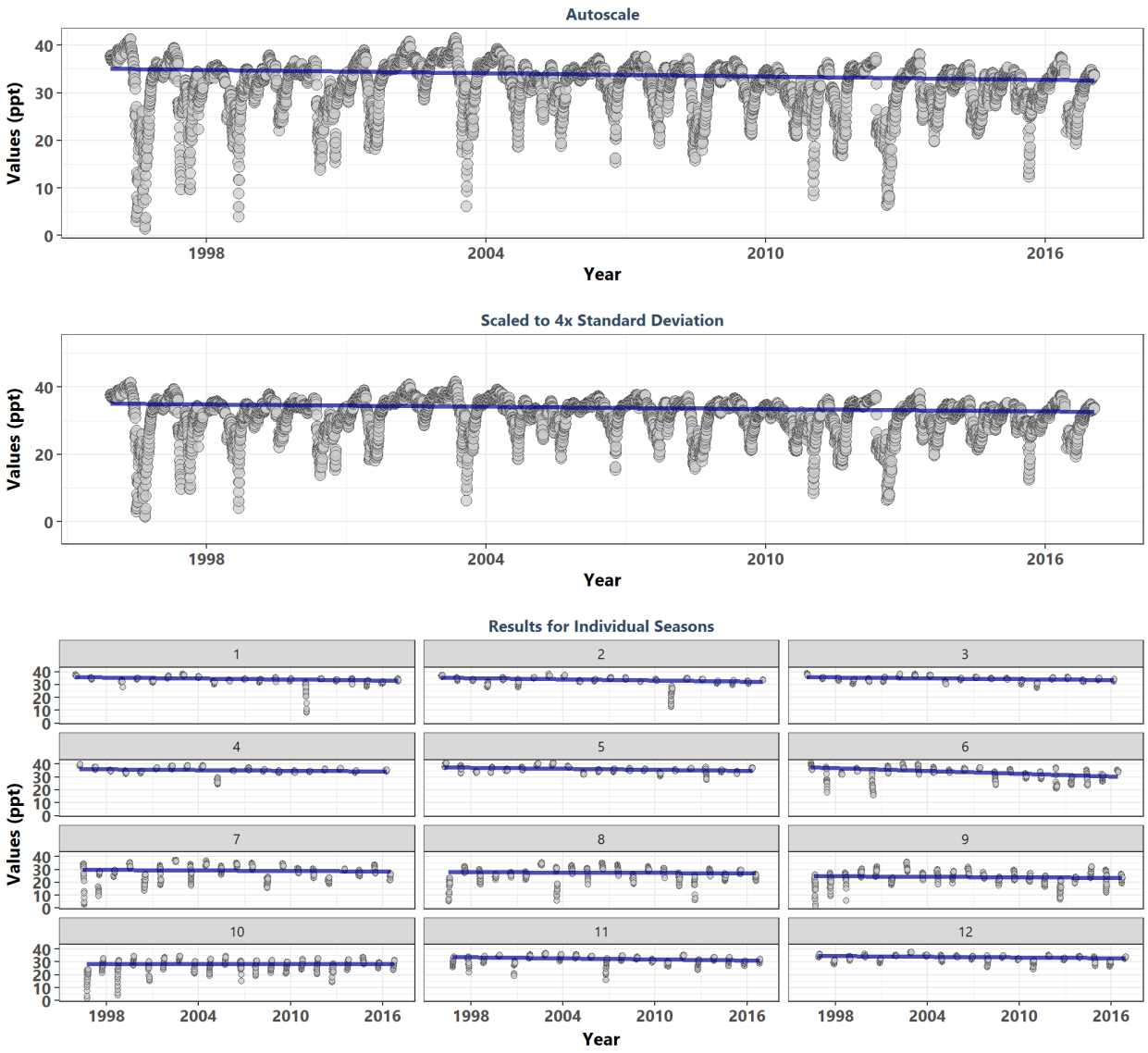
[a] $p < 0.00005$ appear as 0 due to rounding

Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP3C



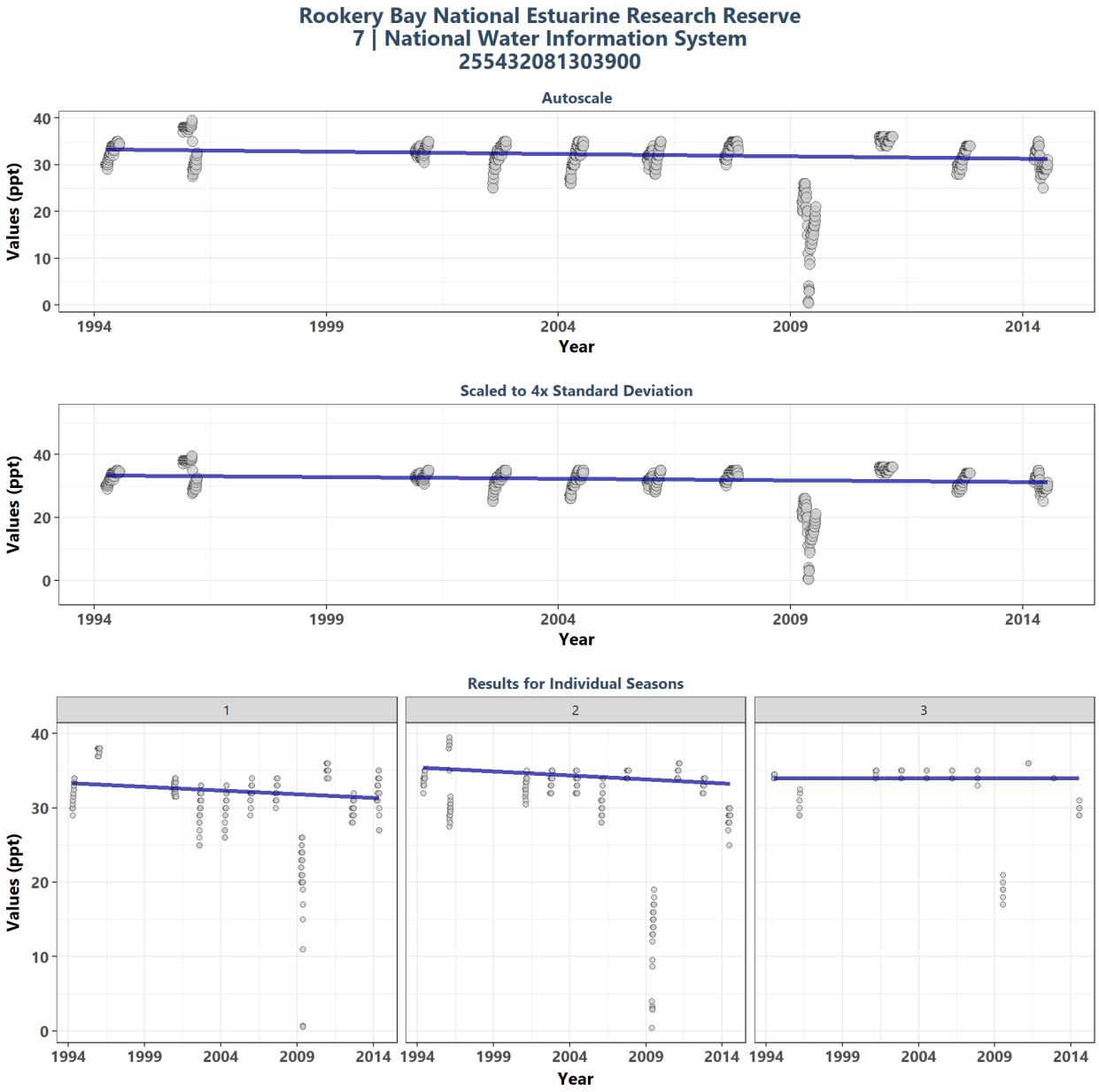
Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	4310	24.03	-0.2067	-0.3639	30.2564	-20.1	0.0000	98.1	0	-1
1	355	27.41	-0.4443	-0.5155	32.5683	-12.5	0.0000	NA	NA	-1
2	319	28.33	-0.3811	-0.3729	32.0563	-10.2	0.0000	NA	NA	-1
3	370	28.59	-0.1787	-0.3252	31.8457	-5.2	0.0000	NA	NA	-1
4	384	28.54	-0.1133	-0.1969	30.5115	-3.3	0.0009	NA	NA	-1
5	351	28.98	-0.1031	-0.1687	30.6677	-2.9	0.0038	NA	NA	-1
6	356	26.04	-0.2022	-0.5620	31.6594	-5.7	0.0000	NA	NA	-1
7	403	19.97	-0.1152	-0.3376	23.3469	-3.5	0.0005	NA	NA	-1
8	367	16.30	-0.2306	-0.3667	19.5990	-6.6	0.0000	NA	NA	-1
9	351	15.12	-0.1467	-0.2153	17.2755	-4.1	0.0000	NA	NA	-1
10	338	17.26	-0.1674	-0.3785	20.6677	-4.6	0.0000	NA	NA	-1
11	344	23.01	-0.1474	-0.3519	26.1799	-4.1	0.0000	NA	NA	-1
12	372	25.12	-0.2795	-0.5138	30.0013	-8.1	0.0000	NA	NA	-1

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



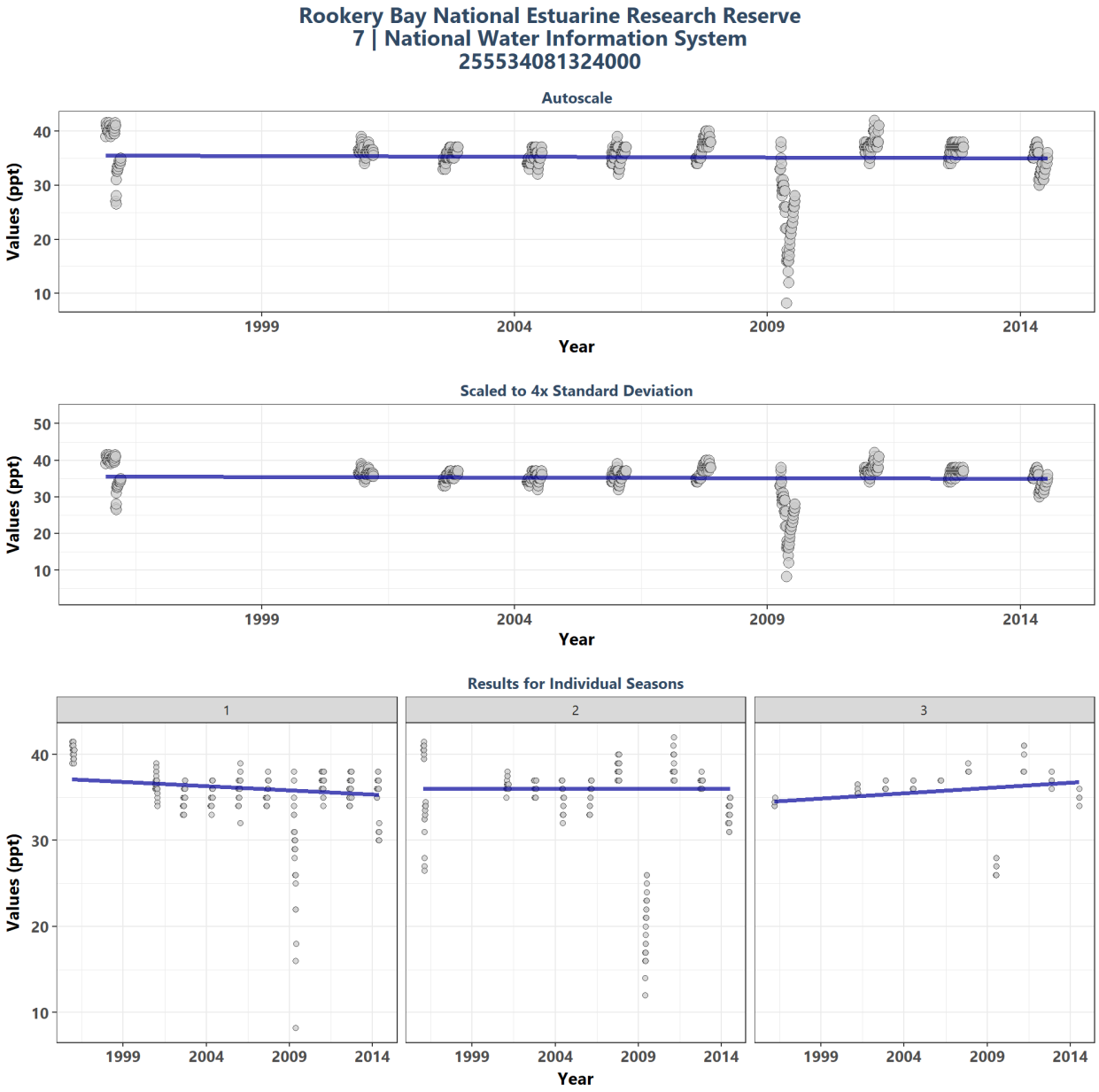
Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	7024	33.05	-0.1667	-0.1169	35.5727	-20.6	0.0000	170.6	0	-1
1	637	34.26	-0.3037	-0.1392	36.3521	-11.5	0.0000	NA	NA	-1
2	555	33.88	-0.2560	-0.1470	35.9399	-9.0	0.0000	NA	NA	-1
3	598	34.68	-0.2002	-0.1010	36.0964	-7.3	0.0000	NA	NA	-1
4	513	35.38	-0.1899	-0.0974	36.6490	-6.4	0.0000	NA	NA	-1
5	520	36.21	-0.2527	-0.1557	38.3860	-8.6	0.0000	NA	NA	-1
6	542	34.01	-0.3128	-0.3628	39.0873	-10.9	0.0000	NA	NA	-1
7	553	29.30	-0.0393	-0.0656	30.2177	-1.4	0.1664	NA	NA	-1
8	613	27.58	-0.0457	-0.0552	28.3542	-1.7	0.0899	NA	NA	-1
9	591	24.38	-0.0467	-0.0592	25.2126	-1.7	0.0888	NA	NA	-1
10	638	28.54	0.0020	0.0021	28.5135	0.1	0.9399	NA	NA	1
11	618	32.67	-0.1780	-0.1277	34.4624	-6.6	0.0000	NA	NA	-1
12	646	33.83	-0.2025	-0.0982	35.2054	-7.7	0.0000	NA	NA	-1

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



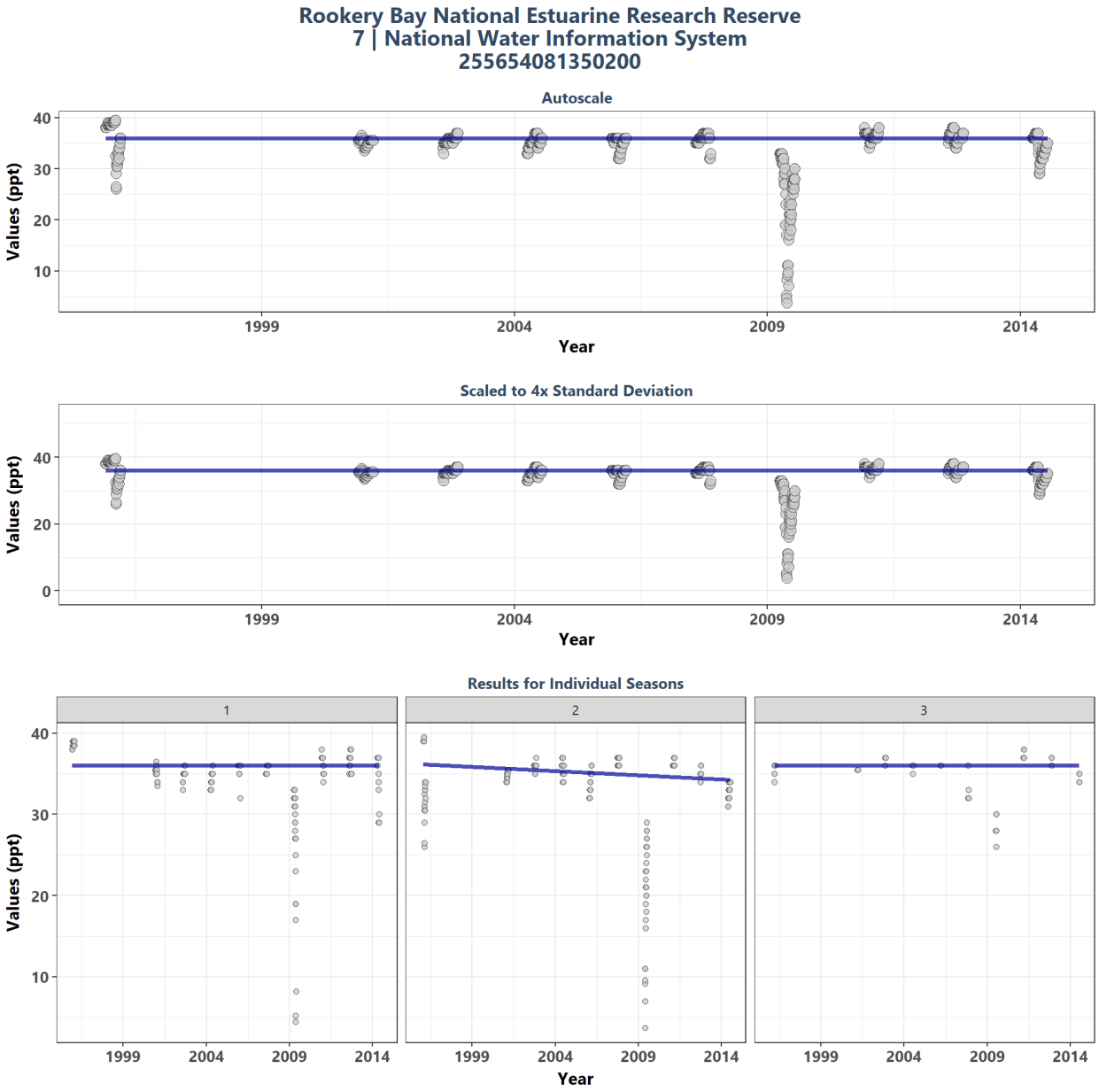
Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	718	33	-0.1307	-0.1667	35.0000	-5.3	0.0000	5.5	0.0652	-1
1	341	32	-0.1182	-0.1667	35.0000	-3.3	0.0010	NA	NA	-1
2	311	34	-0.1556	-0.1818	37.2727	-4.2	0.0000	NA	NA	-1
3	66	34	-0.0779	0.0000	34.0000	-1.0	0.3383	NA	NA	-1

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



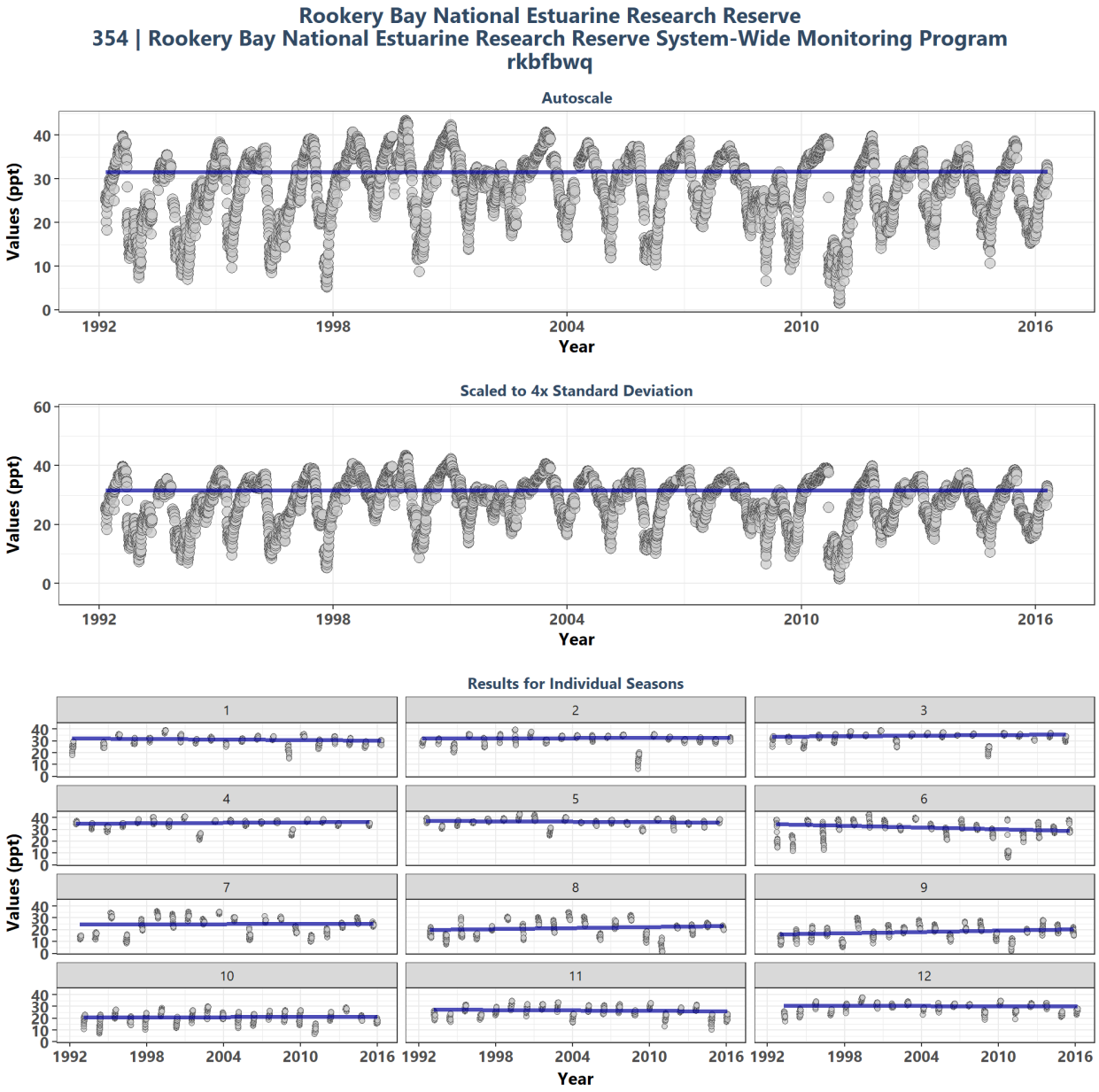
Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	645	36	-0.0636	-0.0455	36.00	-3.2	0.0013	16.5	0.0003	-1
1	307	36	-0.1280	-0.1583	38.85	-3.4	0.0007	NA	NA	-1
2	278	36	-0.0492	0.0000	36.00	-1.2	0.2163	NA	NA	-1
3	60	36	0.1994	0.2000	32.30	2.3	0.0222	NA	NA	1

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



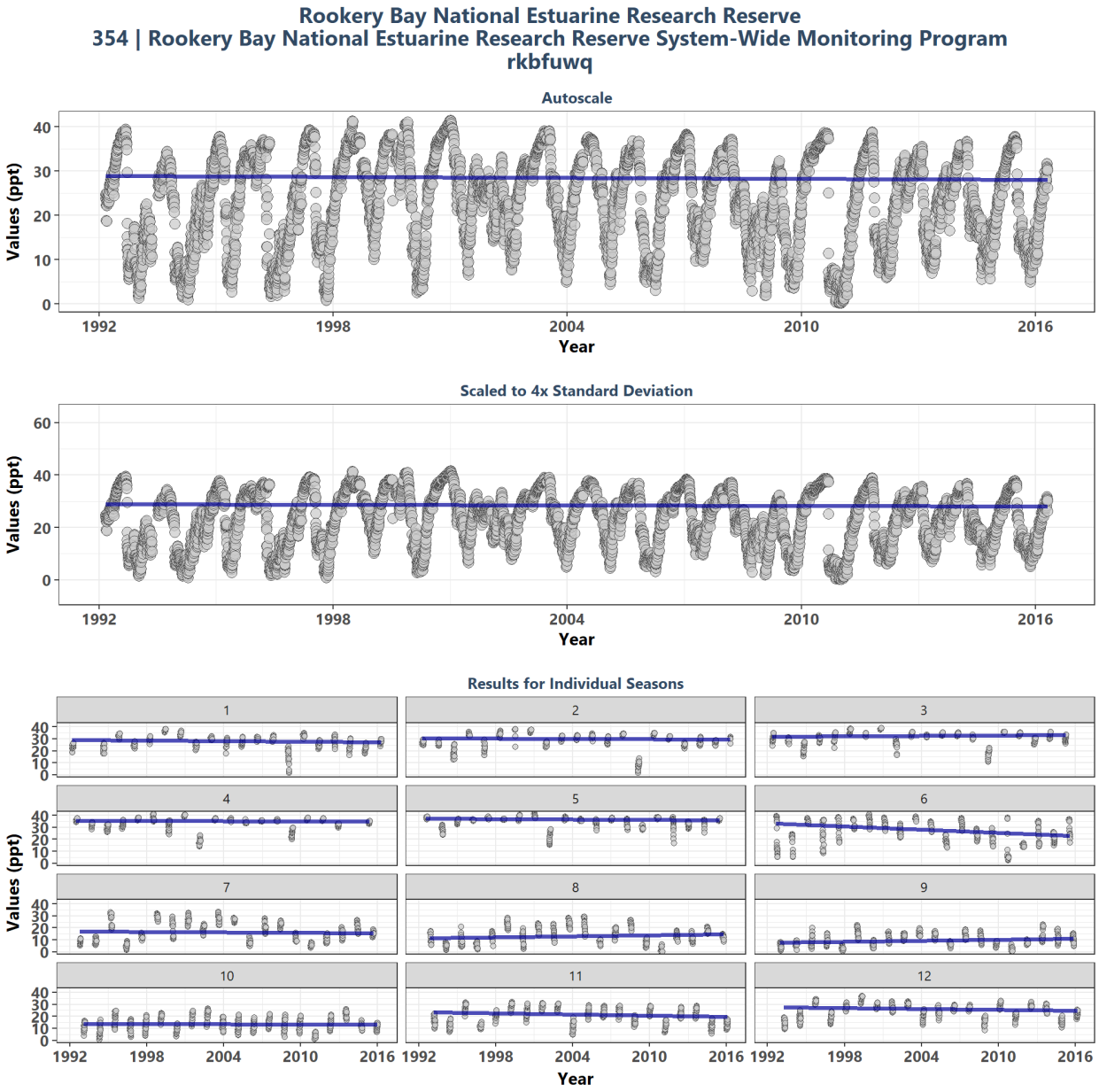
Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	626	35.5	-0.0501	0.0000	36	-1.8	0.0718	5.4	0.0667	0
1	301	36.0	0.0033	0.0000	36	0.1	0.9308	NA	NA	0
2	265	35.0	-0.1178	-0.1667	38	-2.9	0.0037	NA	NA	0
3	60	36.0	-0.0186	0.0000	36	-0.2	0.8340	NA	NA	0

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



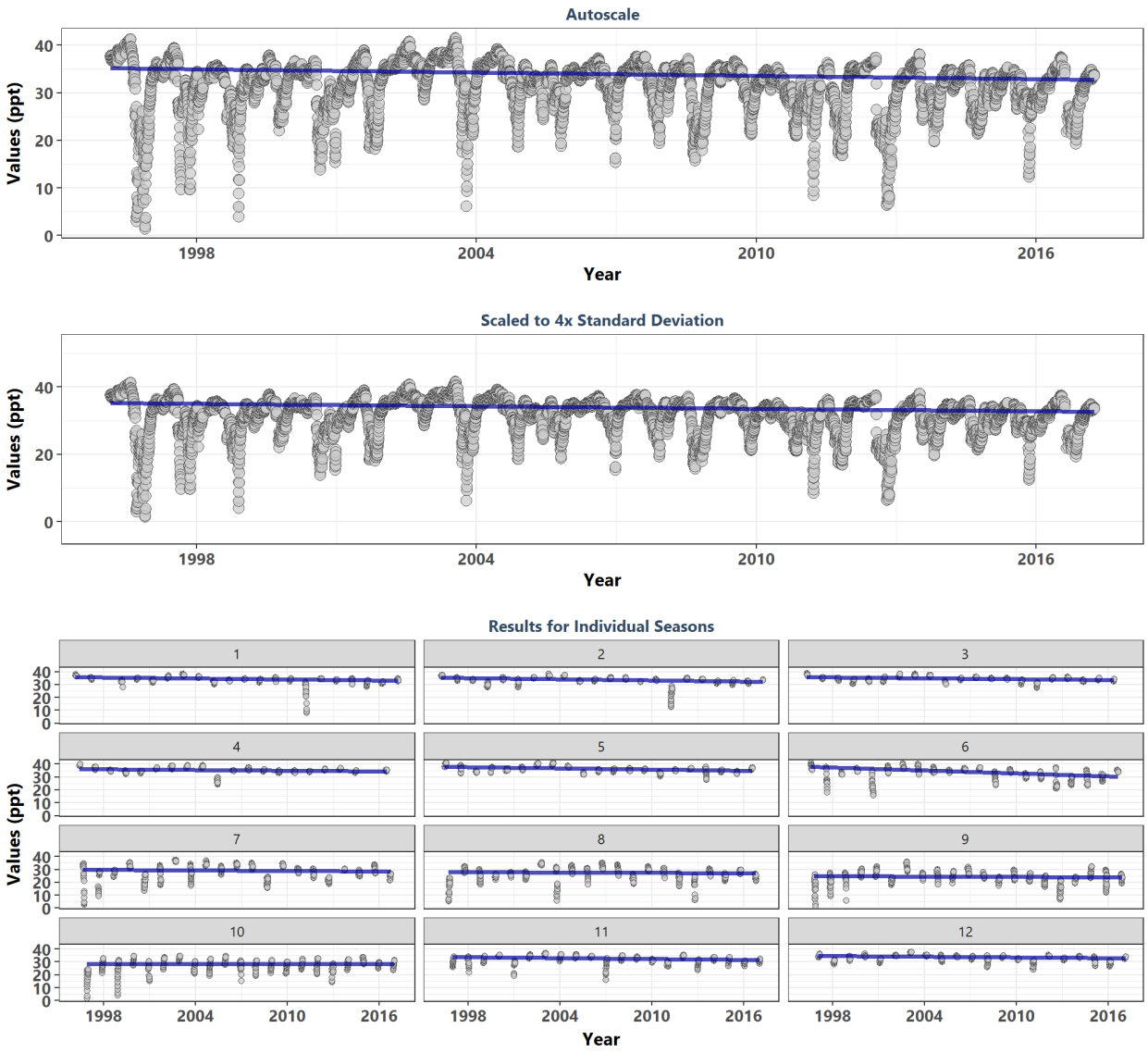
Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	6826	29.50	0.0051	0.0044	31.5235	0.5	0.6492	120.1	0	0
1	587	31.10	-0.0735	-0.0792	32.4460	-2.7	0.0077	NA	NA	0
2	555	32.41	0.0257	0.0251	32.0068	0.9	0.3641	NA	NA	0
3	550	34.42	0.1242	0.0932	33.0173	4.4	0.0000	NA	NA	0
4	506	35.80	0.0853	0.0643	34.8354	2.9	0.0041	NA	NA	0
5	548	36.69	-0.0982	-0.0737	37.8677	-3.4	0.0006	NA	NA	0
6	563	31.62	-0.1437	-0.2942	36.3253	-5.1	0.0000	NA	NA	0
7	588	24.87	0.0029	0.0063	24.7755	0.1	0.9175	NA	NA	0
8	574	21.73	0.0858	0.1695	19.1831	3.1	0.0021	NA	NA	0
9	556	18.35	0.1527	0.2195	15.0560	5.4	0.0000	NA	NA	0
10	603	20.97	0.0244	0.0308	20.5069	0.9	0.3691	NA	NA	0
11	594	26.60	-0.0681	-0.0929	28.0819	-2.5	0.0129	NA	NA	0
12	602	30.43	-0.0385	-0.0381	31.0402	-1.4	0.1573	NA	NA	0

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



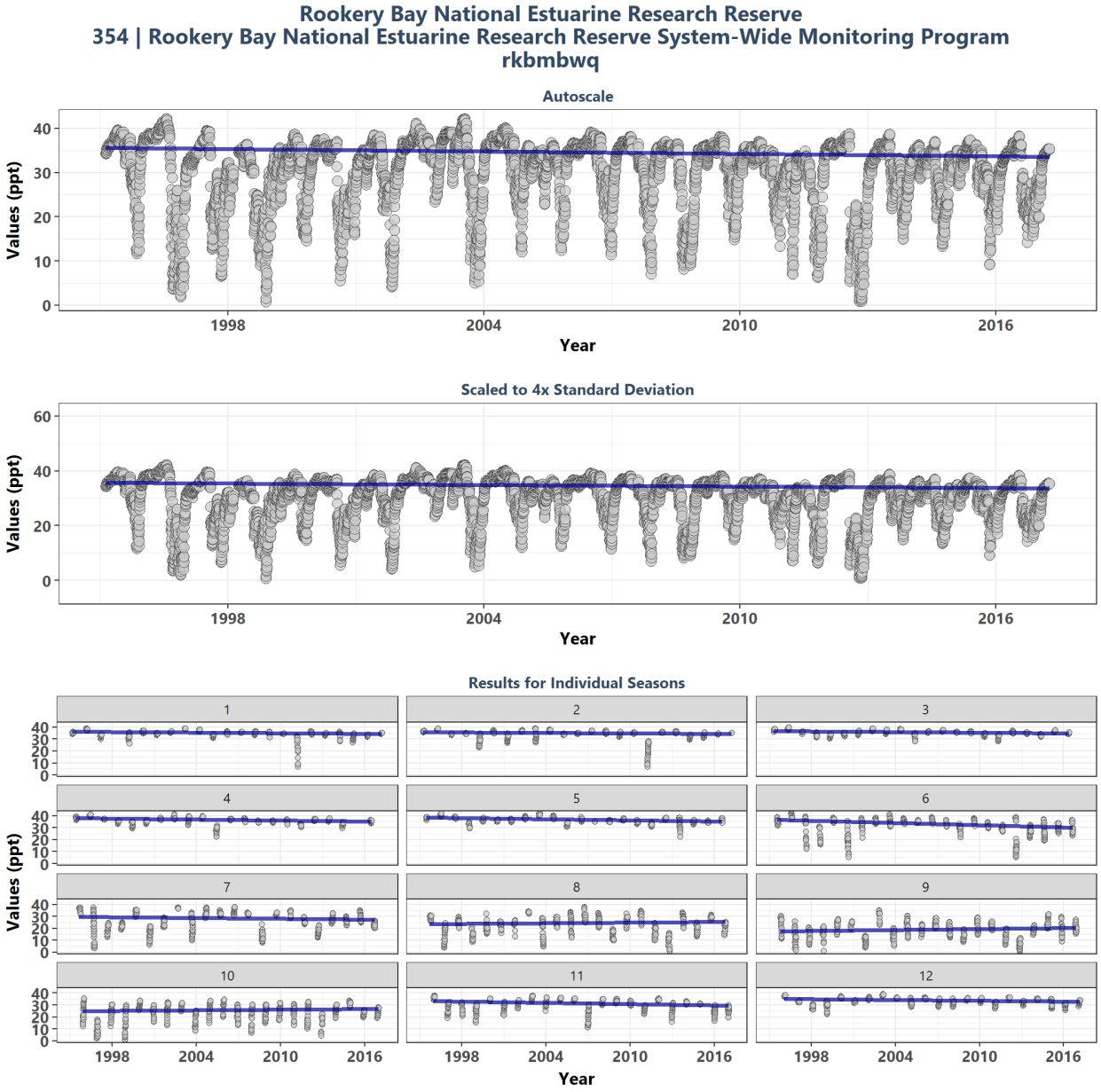
Season	N	Median	tau	Slope	Int.	z	p__z	chi__sq	p__chi__sq	Trend
All	6984	24.79	-0.0295	-0.0420	29.1420	-3.6	0.0003	124.1	0	-1
1	604	28.18	-0.0648	-0.0924	29.7075	-2.4	0.0170	NA	NA	-1
2	544	29.84	-0.0312	-0.0486	30.6226	-1.1	0.2764	NA	NA	-1
3	545	32.59	0.0732	0.0786	31.4161	2.6	0.0105	NA	NA	1
4	546	35.35	-0.0146	-0.0134	35.5502	-0.5	0.6092	NA	NA	-1
5	568	36.55	-0.1106	-0.0778	37.7913	-3.9	0.0001	NA	NA	-1
6	556	27.57	-0.2095	-0.5485	36.3466	-7.4	0.0000	NA	NA	-1
7	602	16.19	-0.0261	-0.0632	17.1349	-1.0	0.3381	NA	NA	-1
8	605	13.08	0.0905	0.1798	10.3840	3.3	0.0009	NA	NA	1
9	600	9.35	0.1215	0.1602	6.8708	4.5	0.0000	NA	NA	1
10	619	13.46	-0.0107	-0.0176	13.7265	-0.4	0.6890	NA	NA	-1
11	592	21.44	-0.0955	-0.1845	24.3905	-3.5	0.0005	NA	NA	-1
12	603	26.30	-0.0852	-0.1516	28.5766	-3.1	0.0017	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	7024	33.05	-0.1667	-0.1169	35.6953	-20.6	0.0000	170.6	0	-1
1	637	34.26	-0.3037	-0.1392	36.4913	-11.5	0.0000	NA	NA	-1
2	555	33.88	-0.2560	-0.1470	36.0870	-9.0	0.0000	NA	NA	-1
3	598	34.68	-0.2002	-0.1010	36.1974	-7.3	0.0000	NA	NA	-1
4	513	35.38	-0.1899	-0.0974	36.7464	-6.4	0.0000	NA	NA	-1
5	520	36.21	-0.2527	-0.1557	38.5417	-8.6	0.0000	NA	NA	-1
6	542	34.01	-0.3128	-0.3628	39.4502	-10.9	0.0000	NA	NA	-1
7	553	29.30	-0.0393	-0.0656	30.2833	-1.4	0.1664	NA	NA	-1
8	613	27.58	-0.0457	-0.0552	28.4094	-1.7	0.0899	NA	NA	-1
9	591	24.38	-0.0467	-0.0592	25.2718	-1.7	0.0888	NA	NA	-1
10	638	28.54	0.0020	0.0021	28.5115	0.1	0.9399	NA	NA	1
11	618	32.67	-0.1780	-0.1277	34.5901	-6.6	0.0000	NA	NA	-1
12	646	33.83	-0.2025	-0.0982	35.3035	-7.7	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	7514	33.24	-0.1148	-0.0916	35.9225	-14.7	0.0000	214.7	0	-1
1	669	35.04	-0.2072	-0.0924	36.5223	-8.0	0.0000	NA	NA	-1
2	595	35.08	-0.1436	-0.0717	36.1585	-5.2	0.0000	NA	NA	-1
3	656	35.74	-0.1823	-0.0883	37.0610	-7.0	0.0000	NA	NA	-1
4	580	36.46	-0.2475	-0.1341	38.3333	-8.9	0.0000	NA	NA	-1
5	589	36.74	-0.2308	-0.1442	38.7595	-8.4	0.0000	NA	NA	-1
6	601	33.44	-0.2167	-0.2928	37.5426	-8.0	0.0000	NA	NA	-1
7	624	28.46	-0.0511	-0.1010	29.8706	-1.9	0.0561	NA	NA	-1
8	653	24.43	0.0536	0.0954	22.9952	2.0	0.0404	NA	NA	1
9	617	19.02	0.0718	0.1195	17.2246	2.7	0.0076	NA	NA	1
10	653	25.68	0.0574	0.0929	24.3783	2.2	0.0279	NA	NA	1
11	610	31.61	-0.1497	-0.1735	34.0400	-5.5	0.0000	NA	NA	-1
12	667	34.13	-0.1546	-0.1115	35.6865	-6.0	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

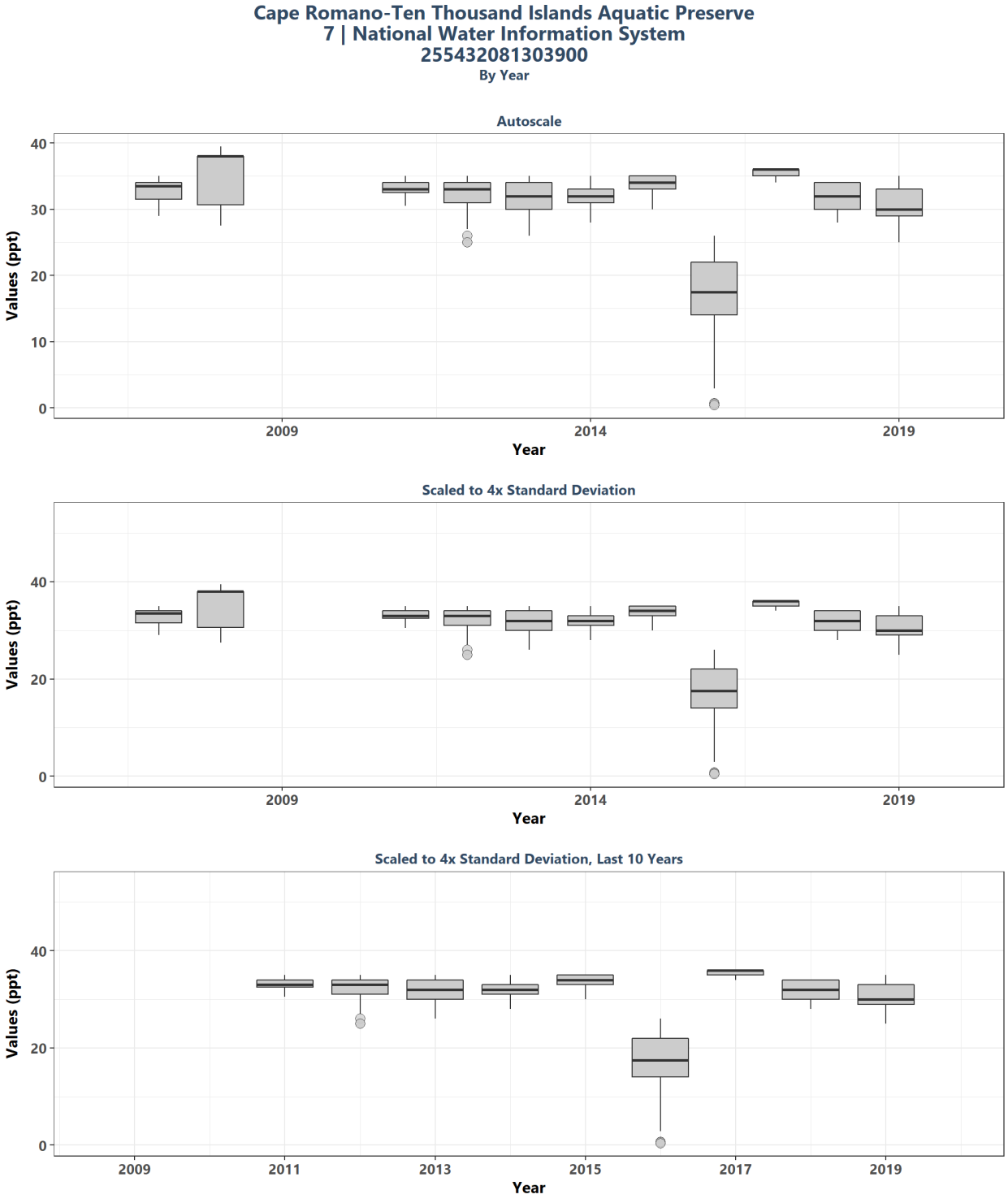
```
if(n==0){
   print("There are no monitoring locations that qualify.")
} else {
   for (i in 1:n) {
      year_lower <- min(data$Year[data$Use_In_Analysis==TRUE &
                                  data$MonitoringID==Mon_IDs[i]])
      year_upper <- max(data$Year[data$Use_In_Analysis==TRUE &
     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

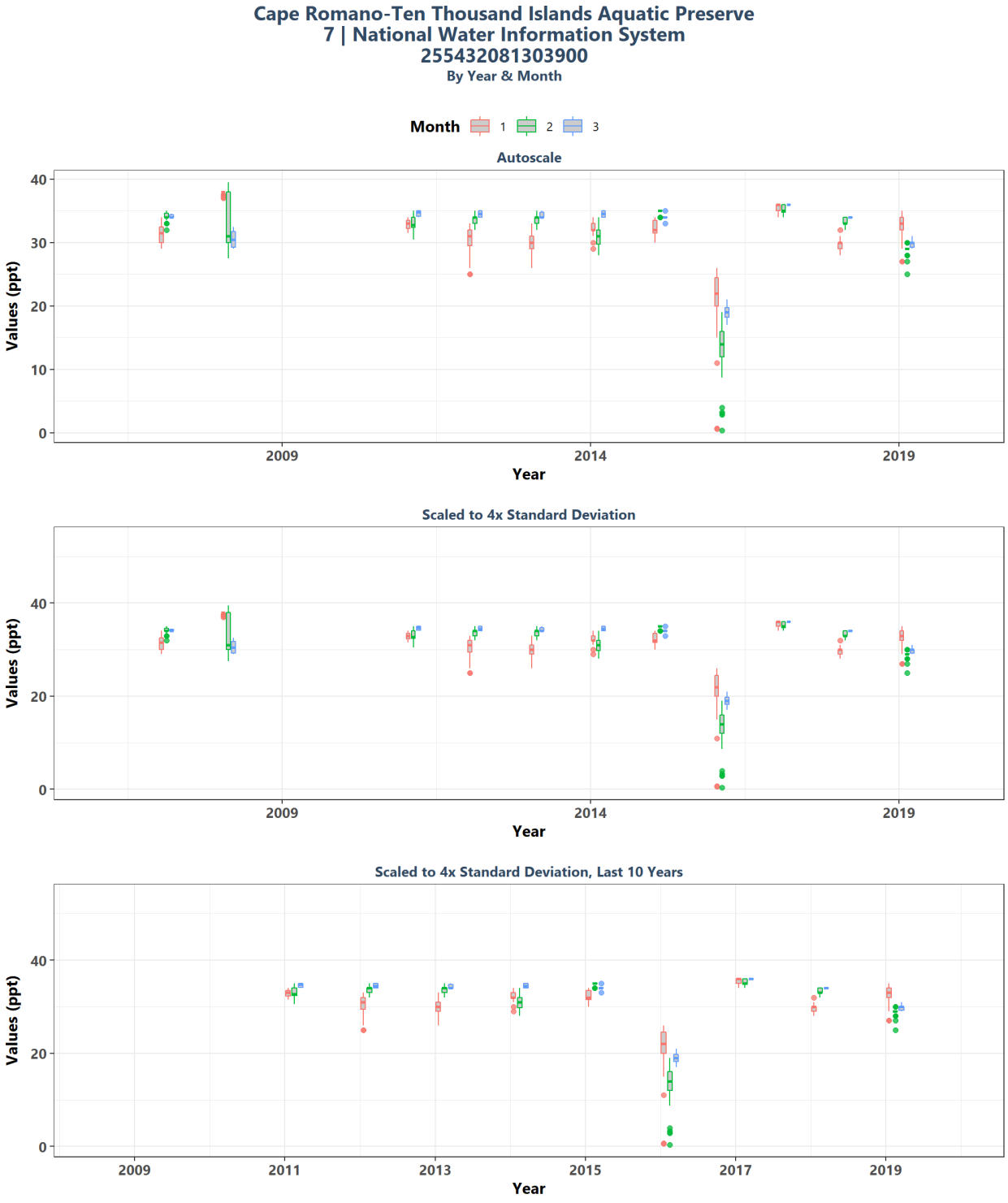
```
                aes(x=Year, y=ResultValue, group=Year)) +
    geom_boxplot(color="#333333", fill="#cccccc", outlier.shape=21,
                 outlier.size=3, outlier.color="#333333",
                 outlier.fill="#cccccc", outlier.alpha=0.75) +
    labs(subtitle="Autoscale",
         x="Year",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

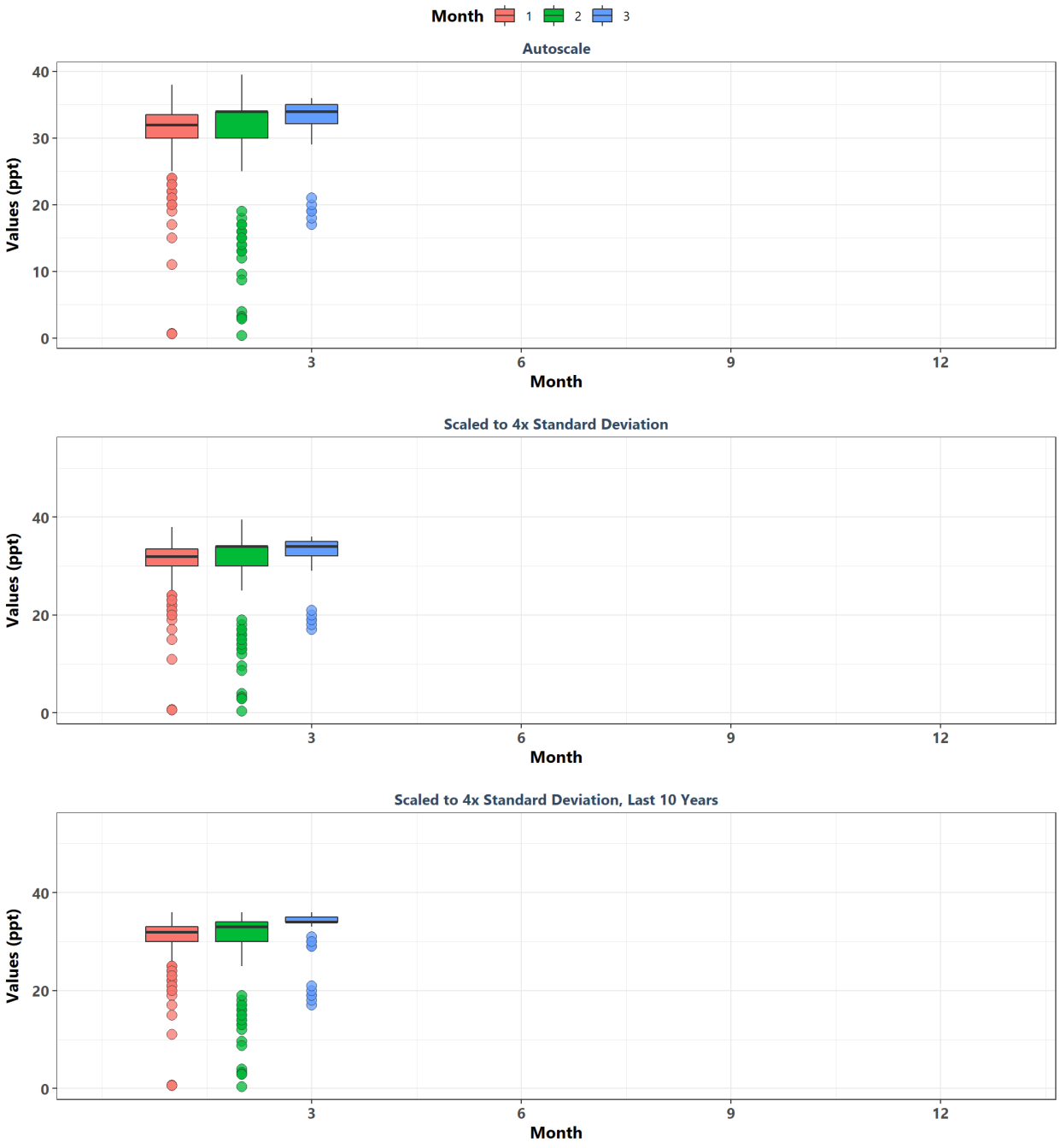
```
   geom_boxplot(color="#333333", outlier.shape=21, outlier.size=3,
                outlier.color="#333333", outlier.alpha=0.75) +
   labs(subtitle="Autoscale",
        x="Month", y=paste0("Values (", unit, ")"), fill="Month") +
   scale_x_continuous(limits=c(0, 13), breaks=seq(3, 12, 3)) +
   plot_theme +
   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)
```

```
    }
}
```

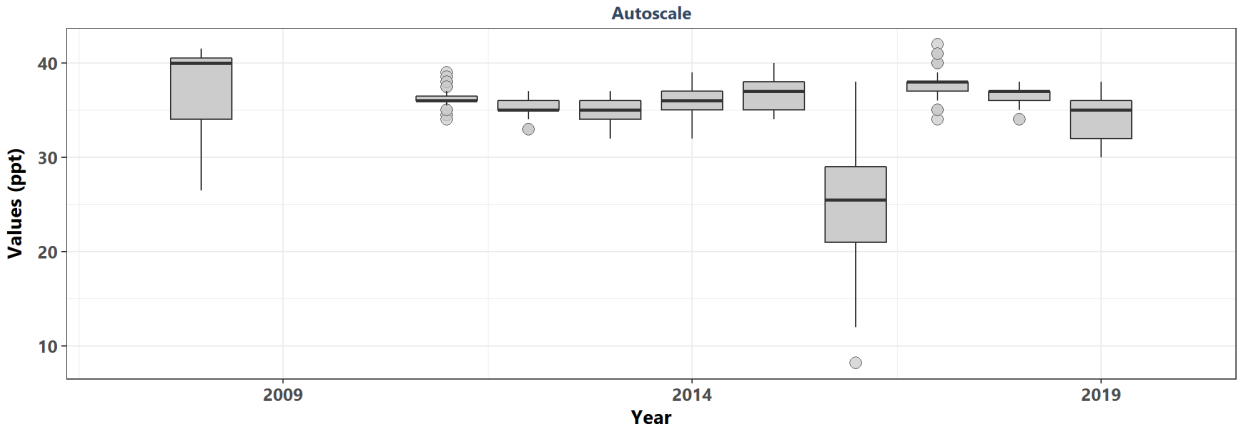

Cape Romano-Ten Thousand Islands Aquatic Preserve
7 | National Water Information System
255432081303900
By Year
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year

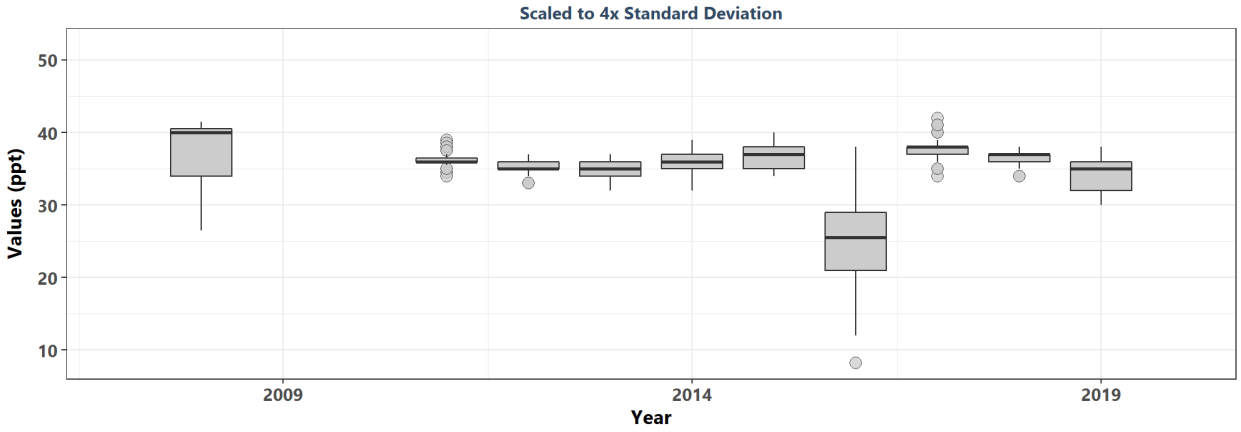

Cape Romano-Ten Thousand Islands Aquatic Preserve
7 | National Water Information System
255432081303900
By Year & Month
Month 1 2 3
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year

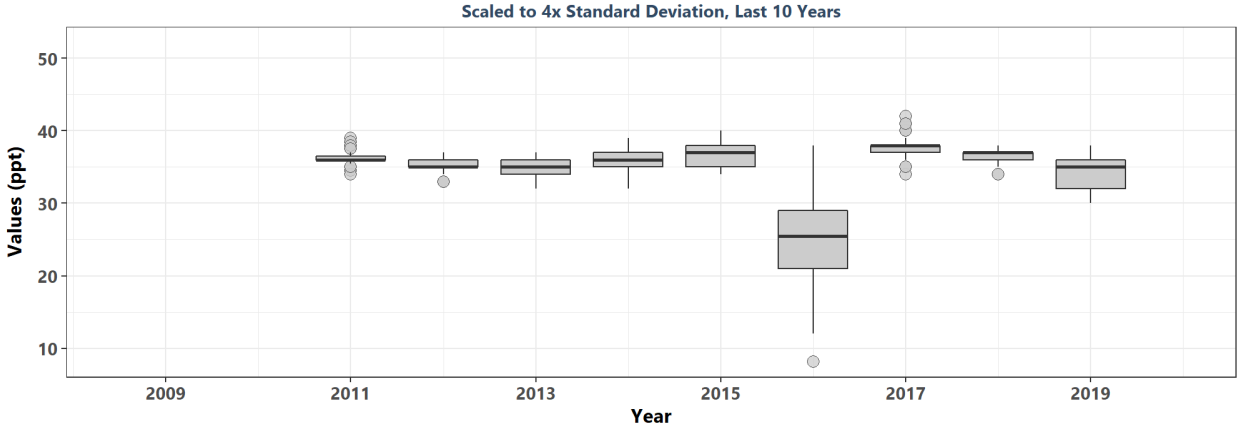
Cape Romano-Ten Thousand Islands Aquatic Preserve
7 | National Water Information System
255432081303900
By Month

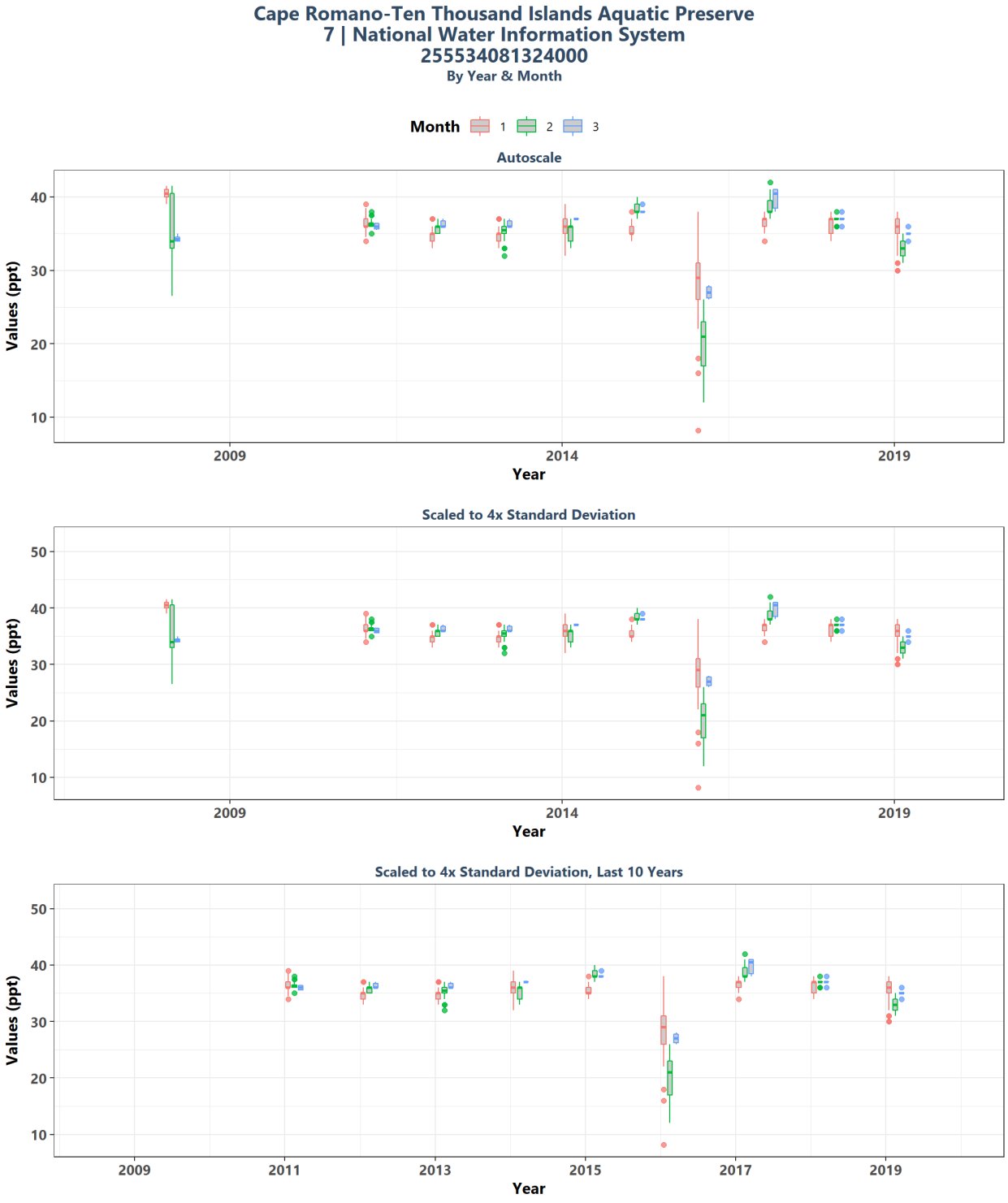


Cape Romano-Ten Thousand Islands Aquatic Preserve
7 | National Water Information System
255534081324000
By Year

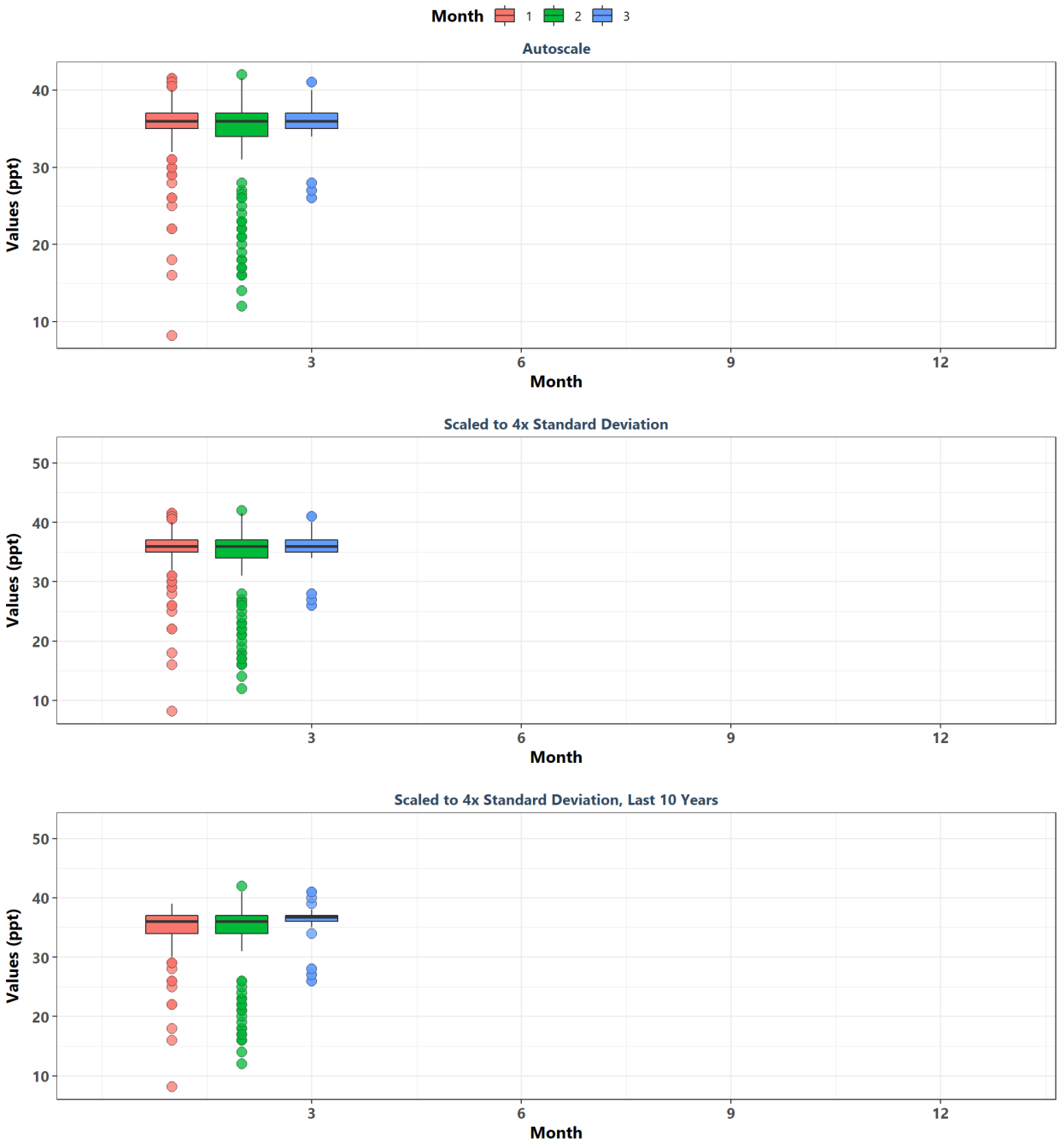




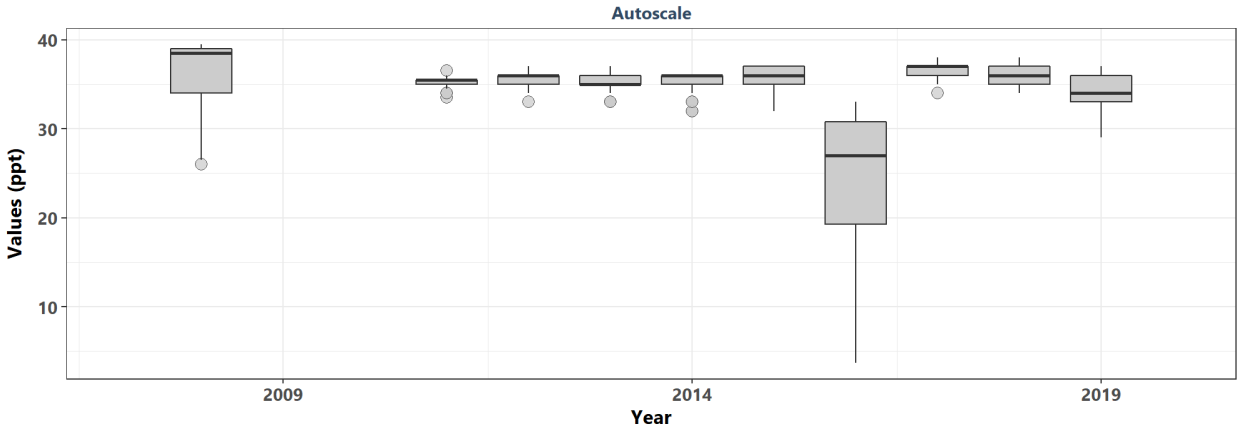


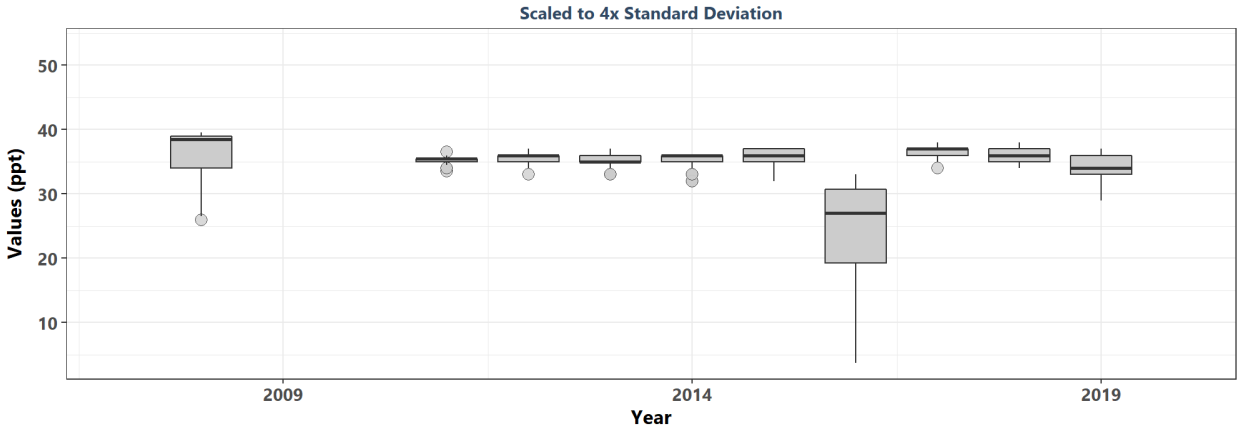

Cape Romano-Ten Thousand Islands Aquatic Preserve
7 | National Water Information System
255534081324000
By Year & Month
Month 1 2 3
Autoscale
Values (ppt)
Year
2009
2014
2019
Scaled to 4x Standard Deviation
Values (ppt)
Year
2009
2014
2019
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year
2009
2011
2013
2015
2017
2019

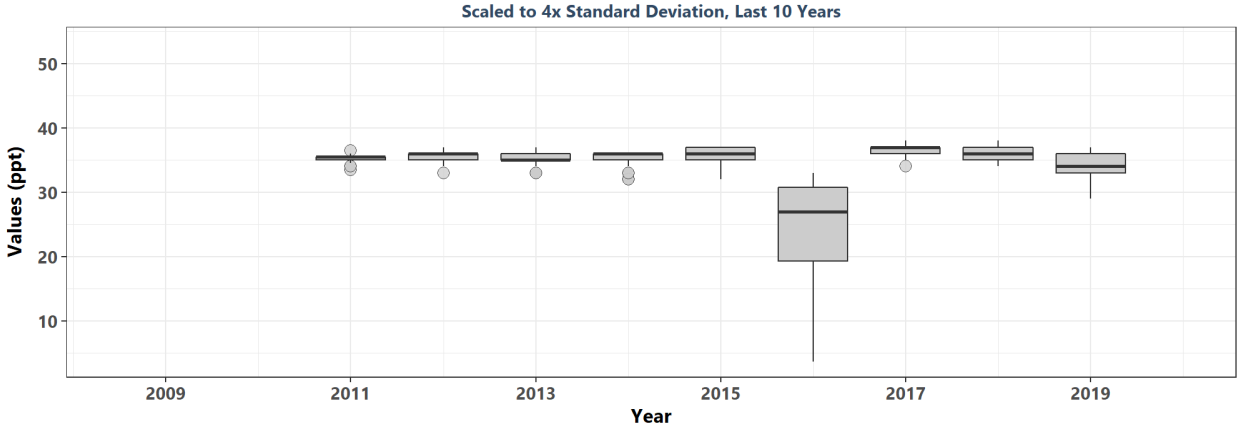
Cape Romano-Ten Thousand Islands Aquatic Preserve
7 | National Water Information System
255534081324000
By Month

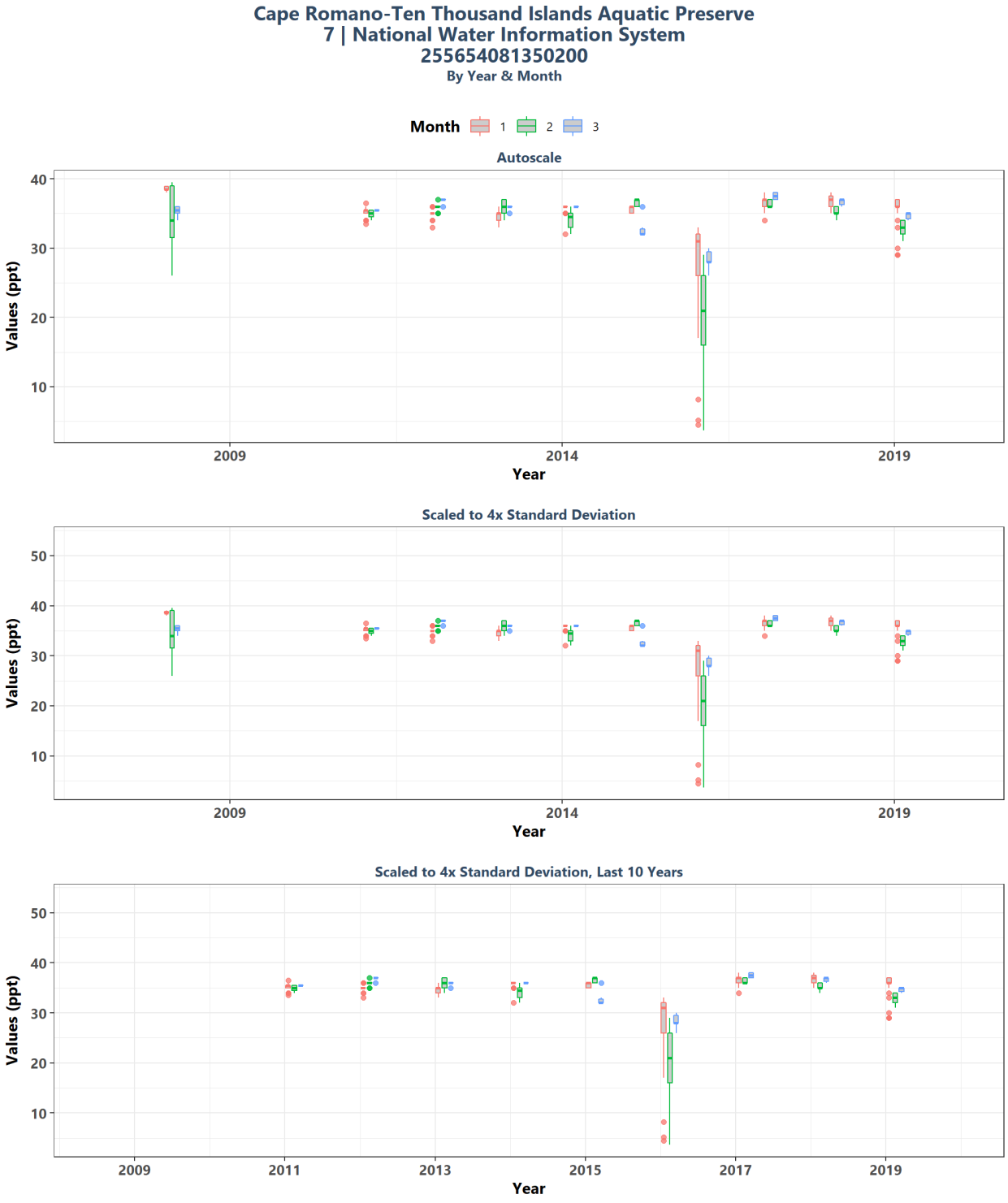


Cape Romano-Ten Thousand Islands Aquatic Preserve
7 | National Water Information System
255654081350200
By Year

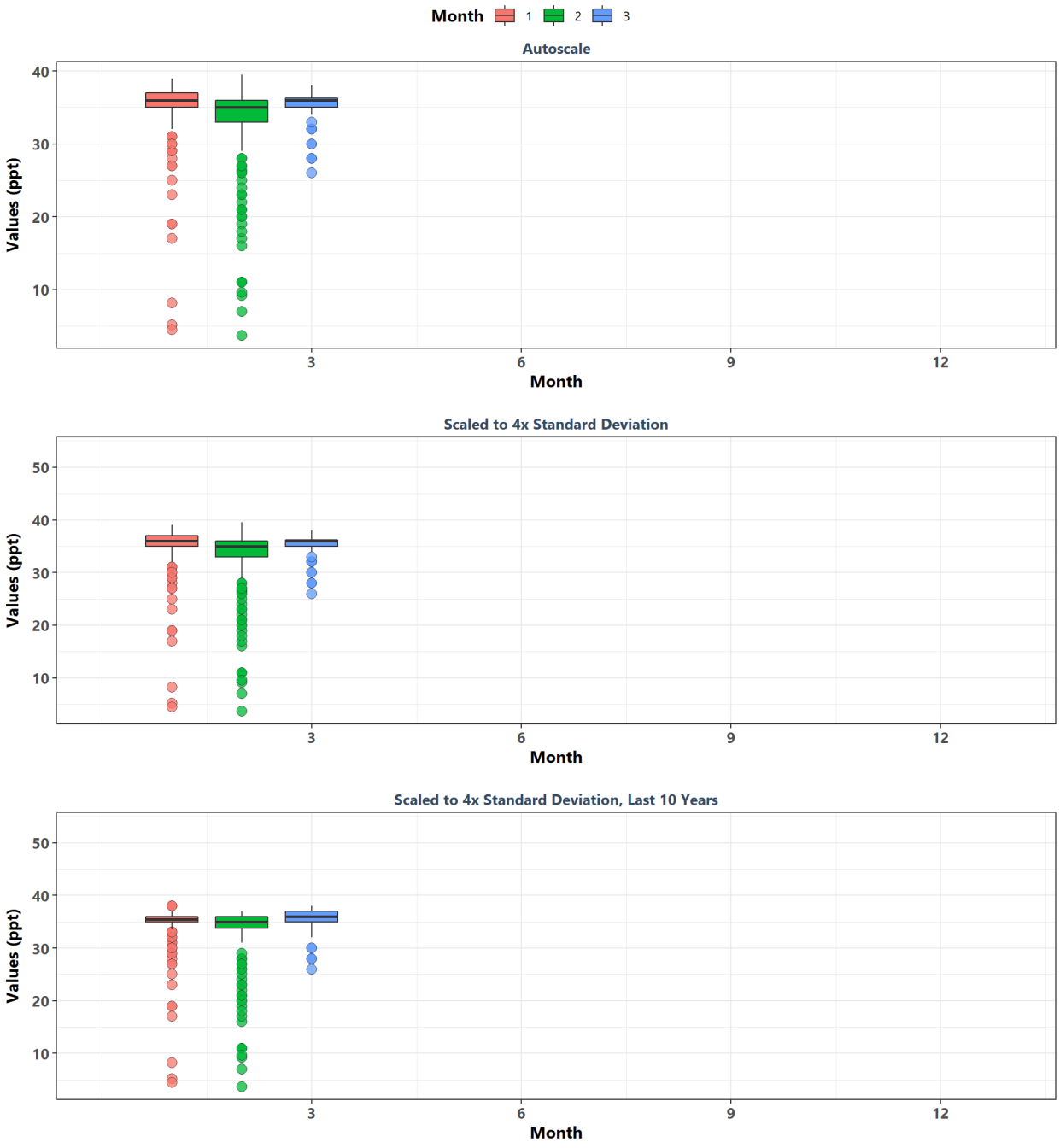




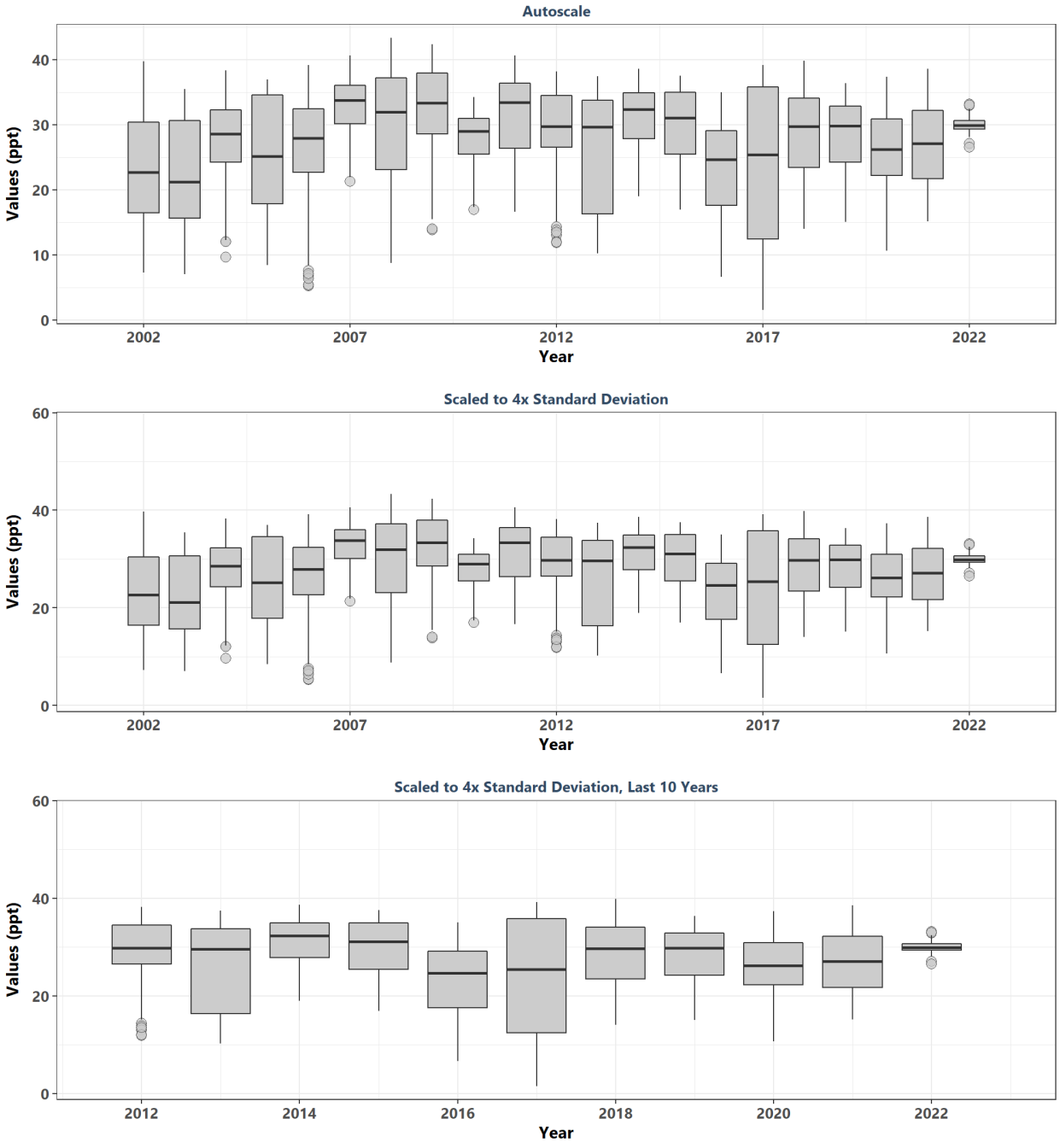



Cape Romano-Ten Thousand Islands Aquatic Preserve
7 | National Water Information System
255654081350200
By Year & Month
Month 1 2 3
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year

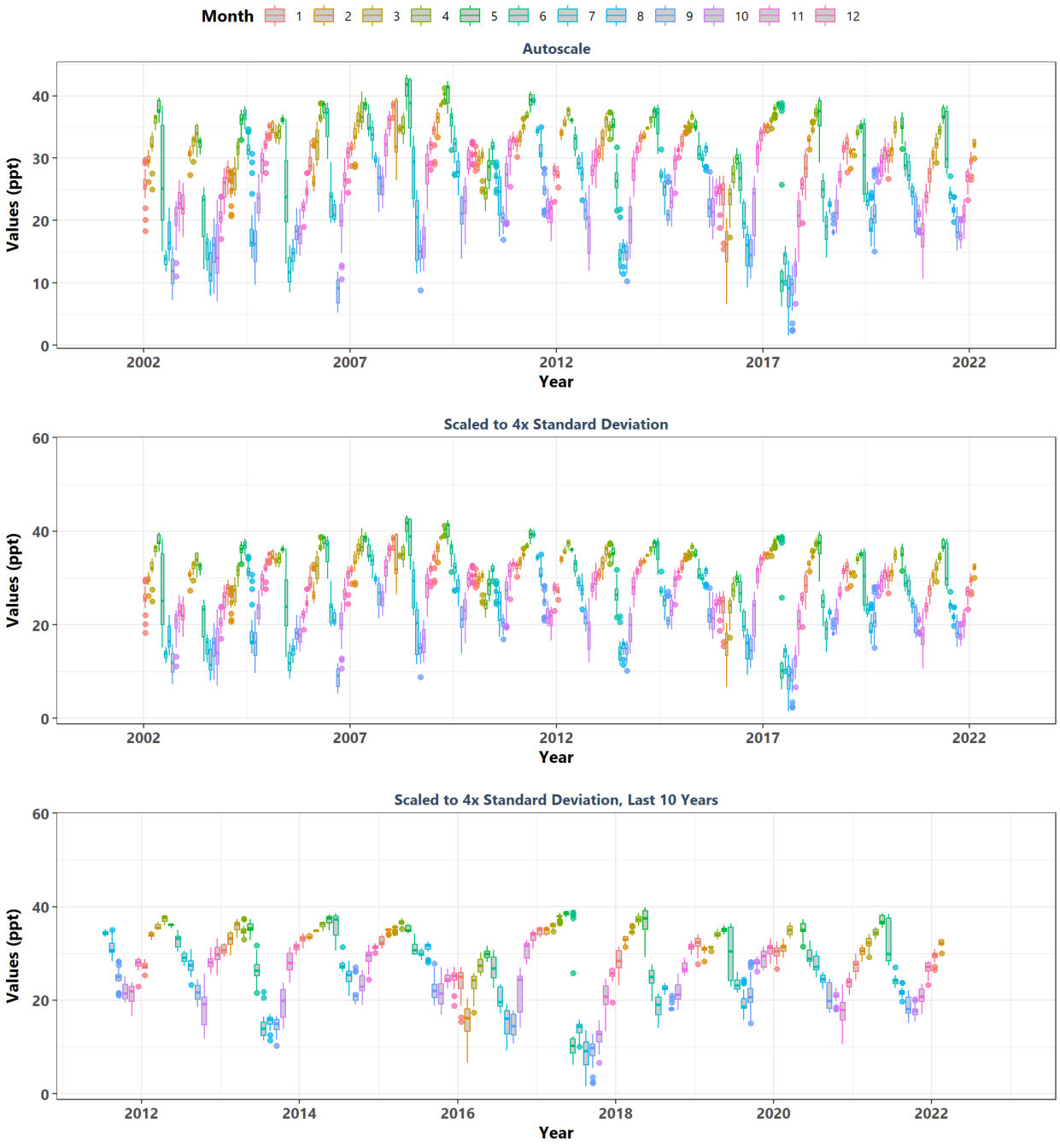
Cape Romano-Ten Thousand Islands Aquatic Preserve
7 | National Water Information System
255654081350200
By Month



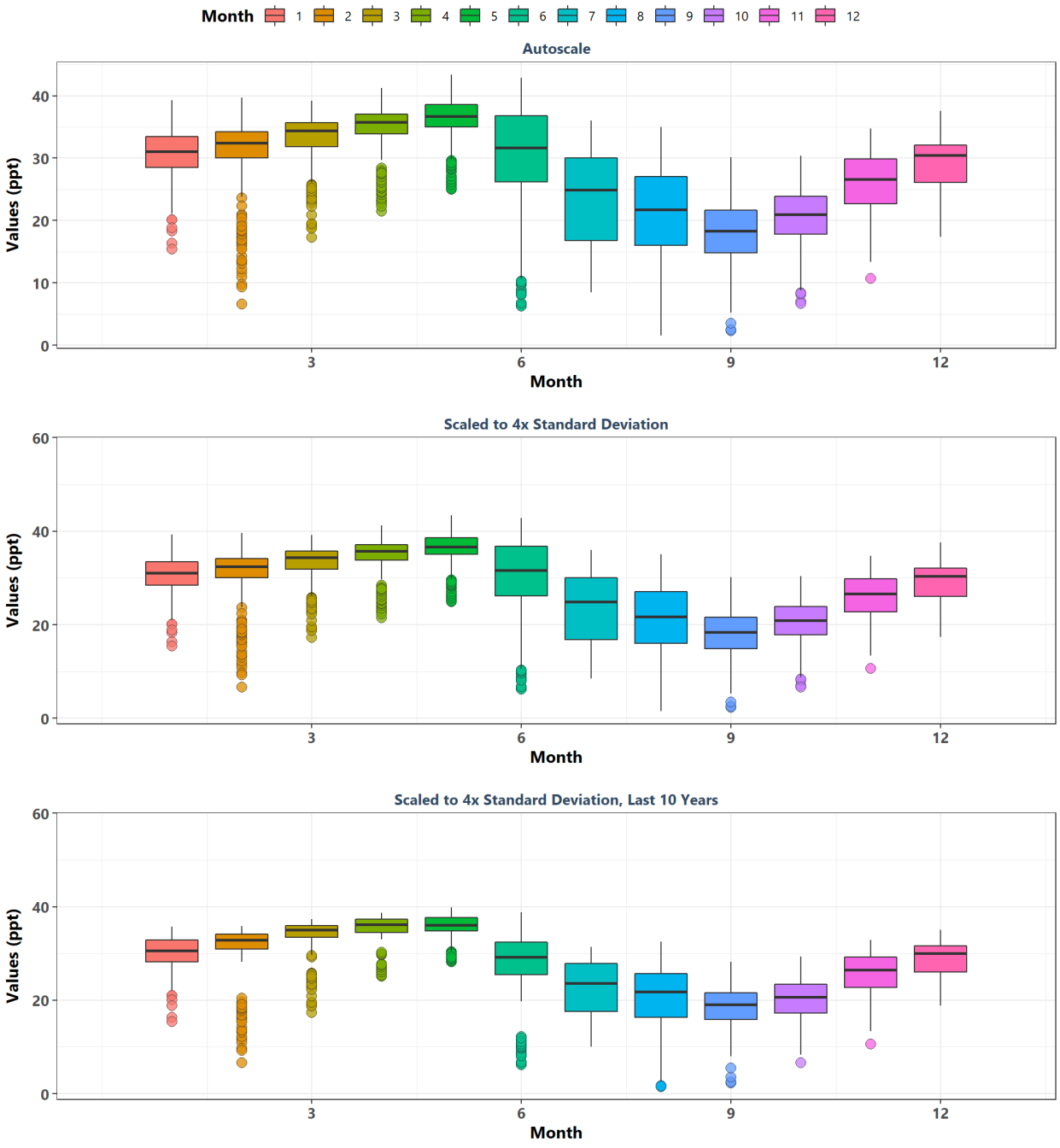
Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfbwq
By Year



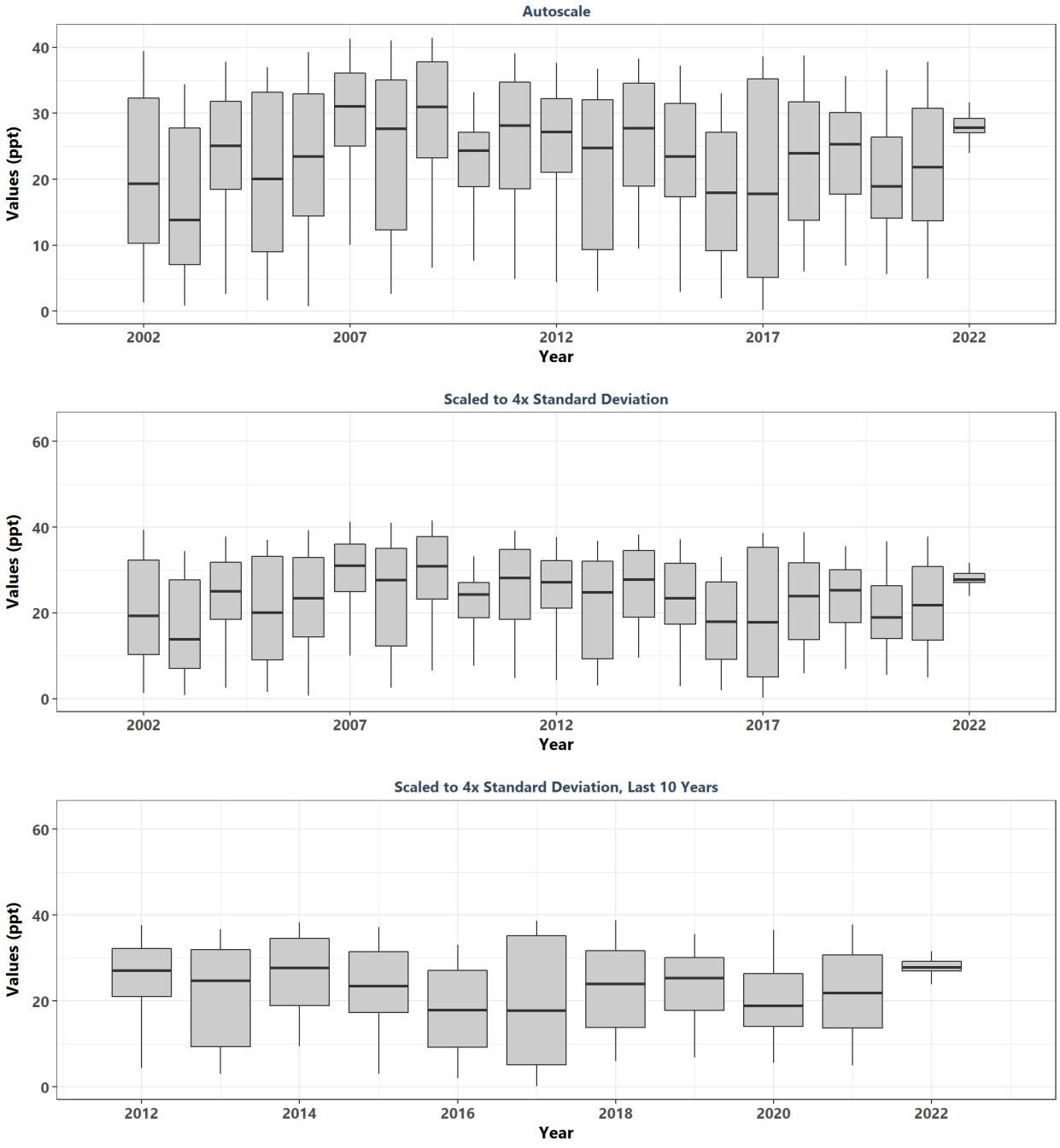
Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfbwq
By Year & Month

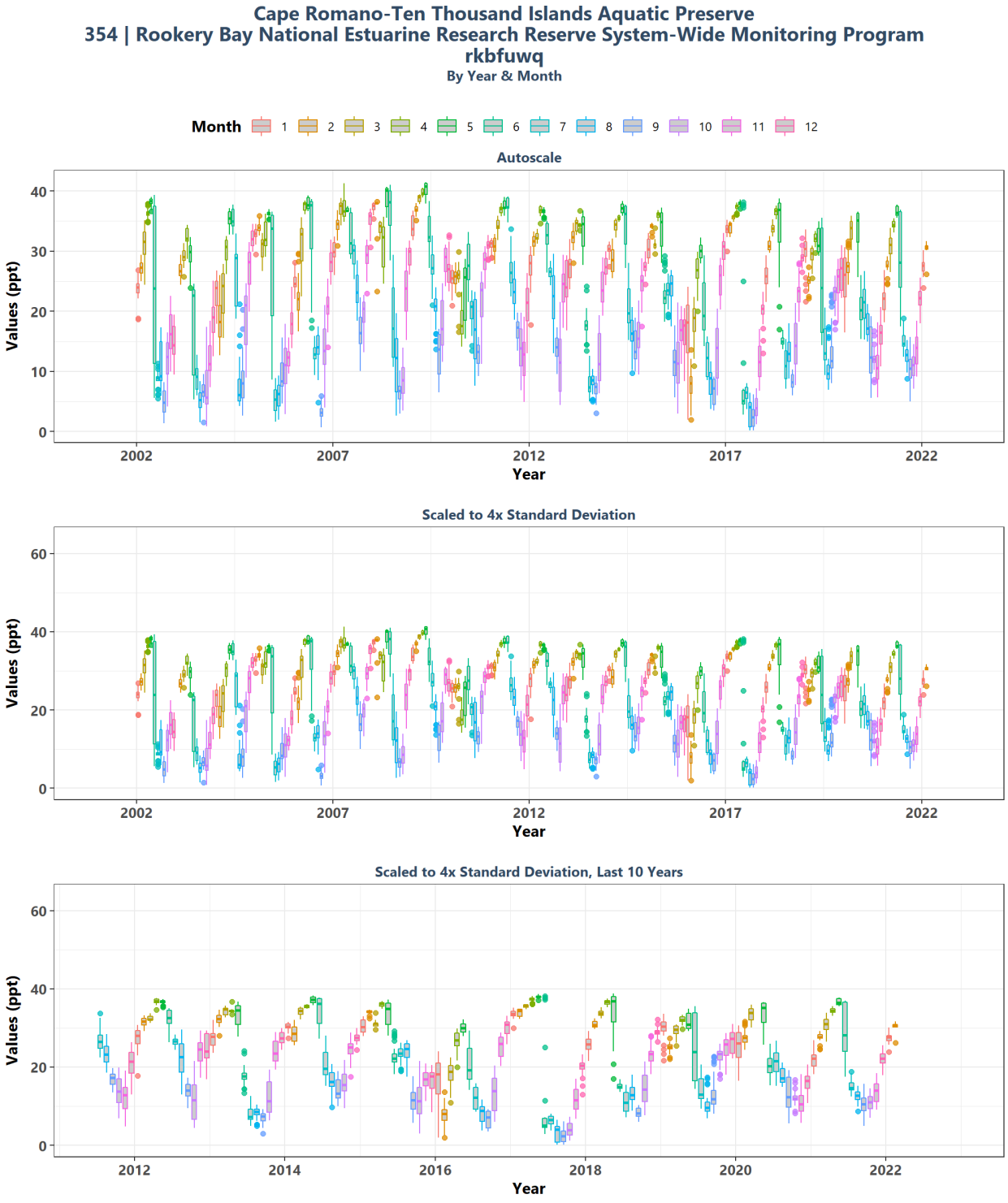


Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfbwq
By Month

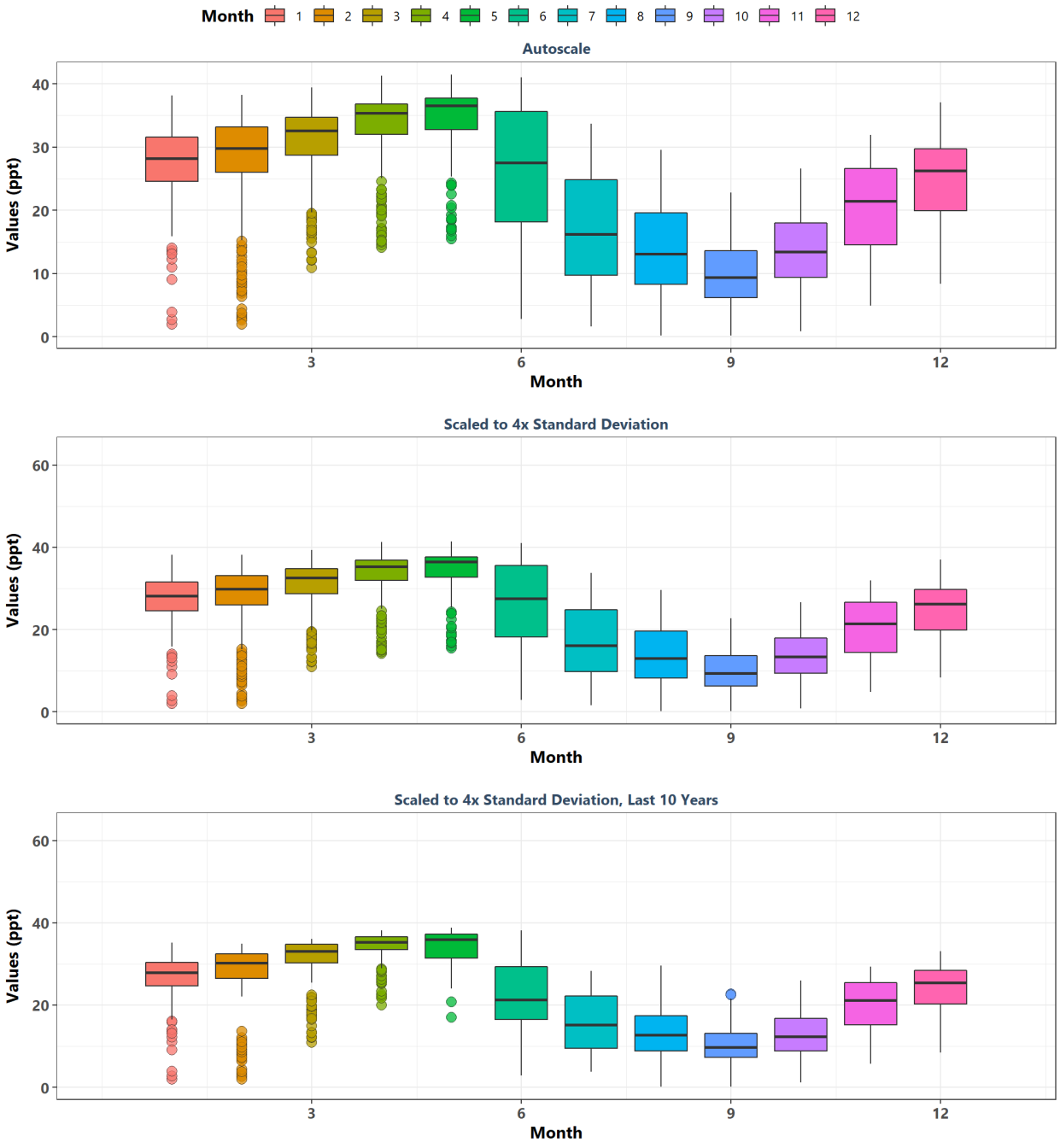


Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfuwq
By Year

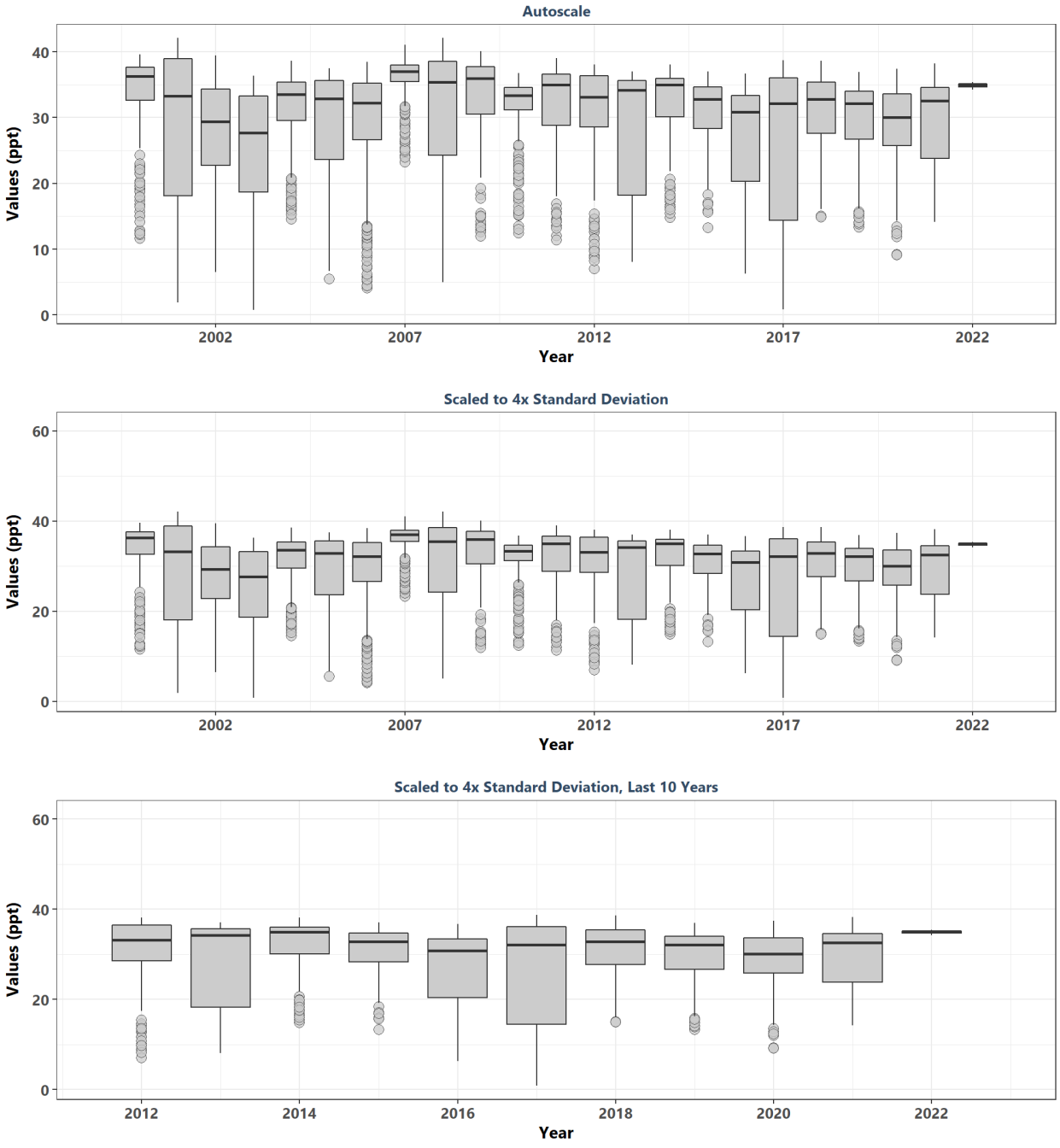



Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfuwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year

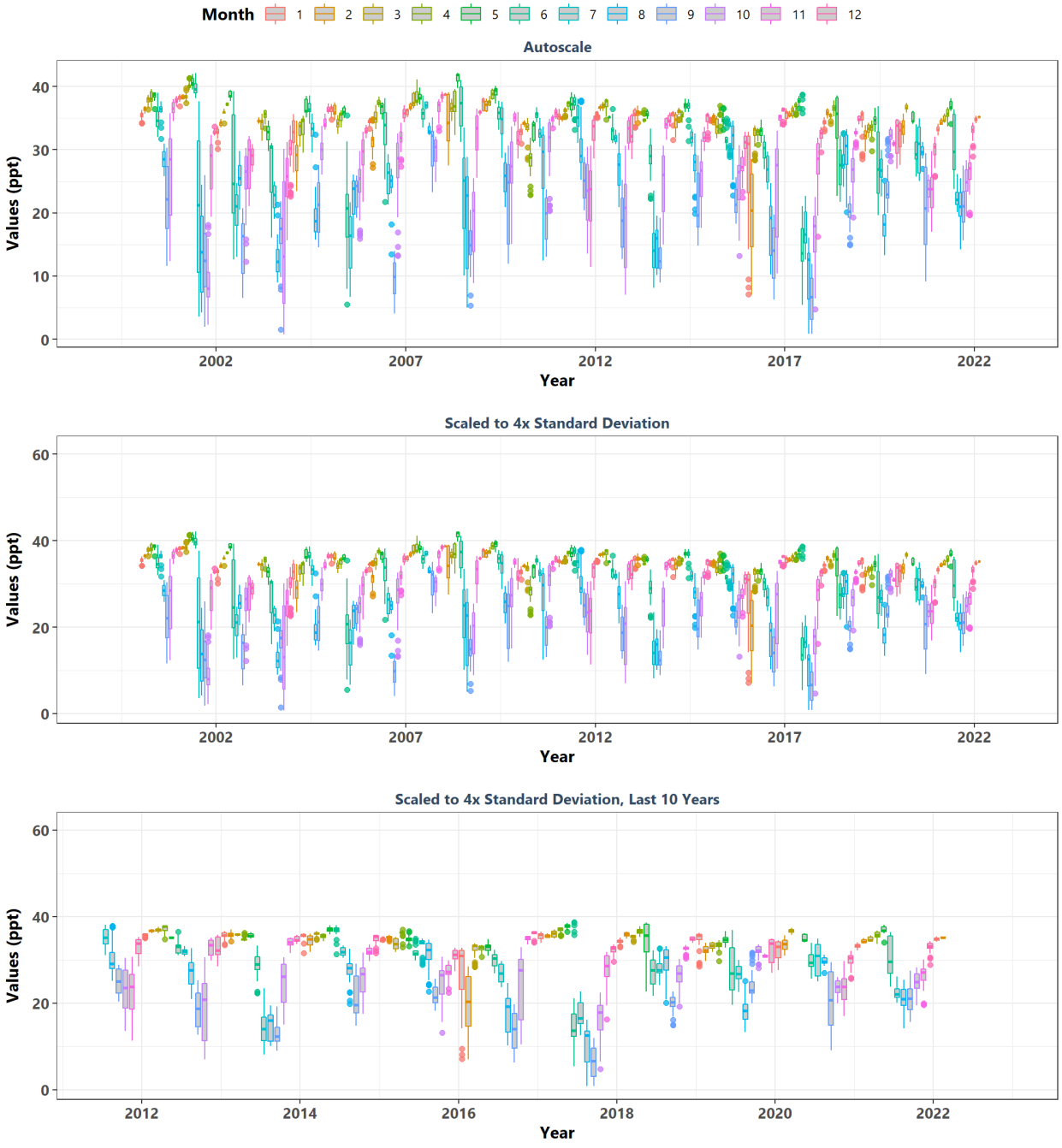
Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfuwq
By Month



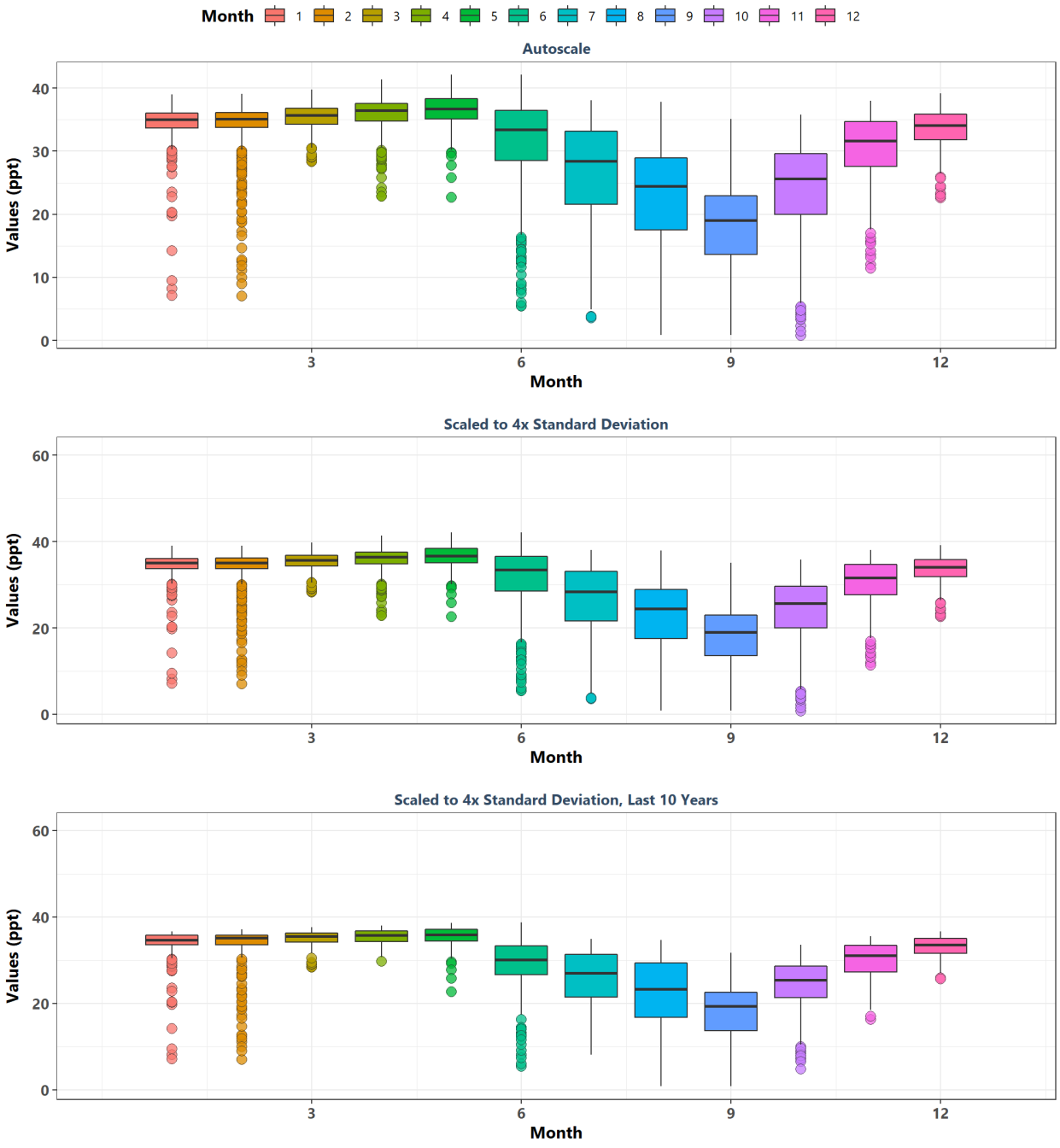
Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbmbwq
By Year



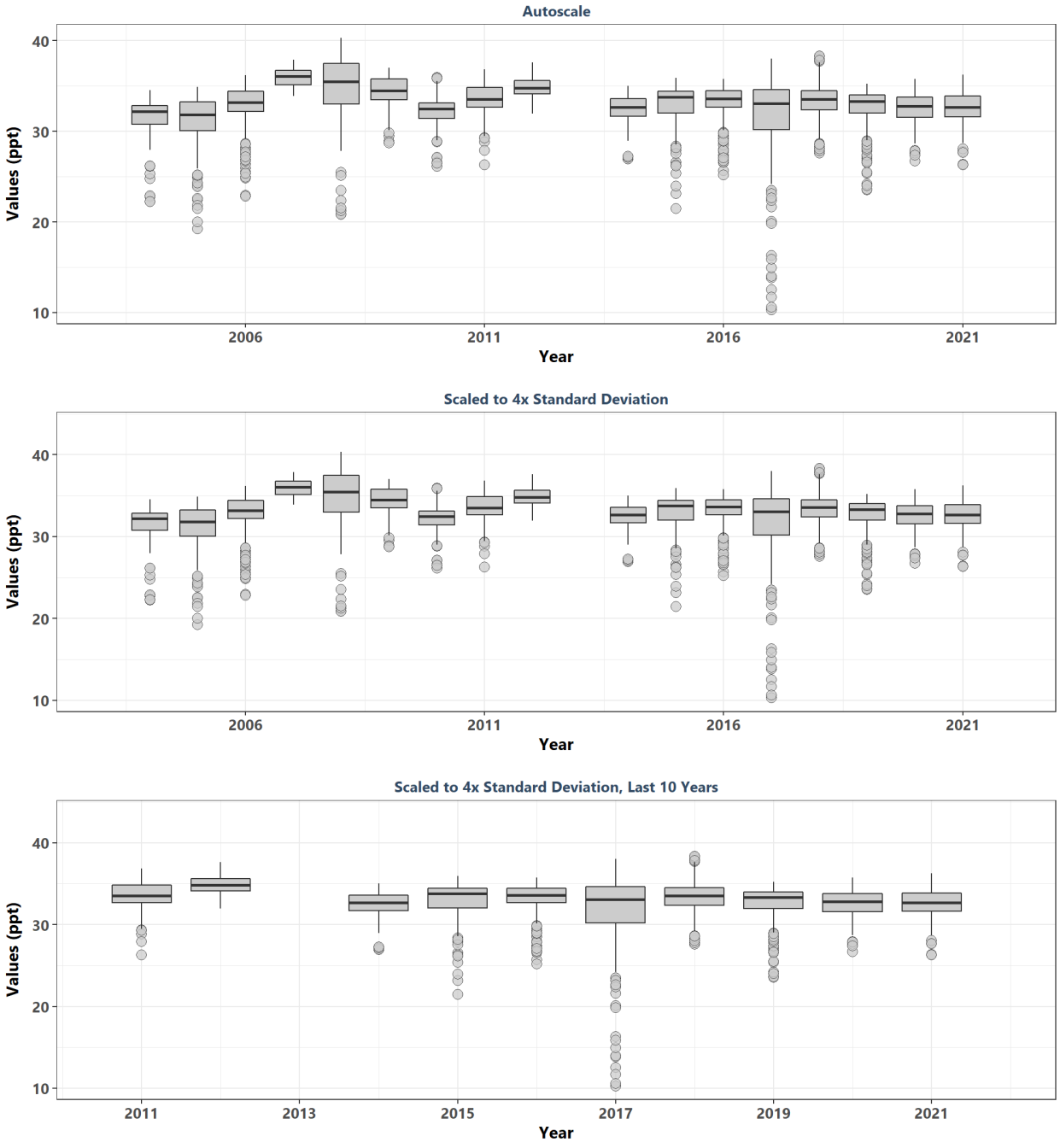
Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbmbwq
By Year & Month

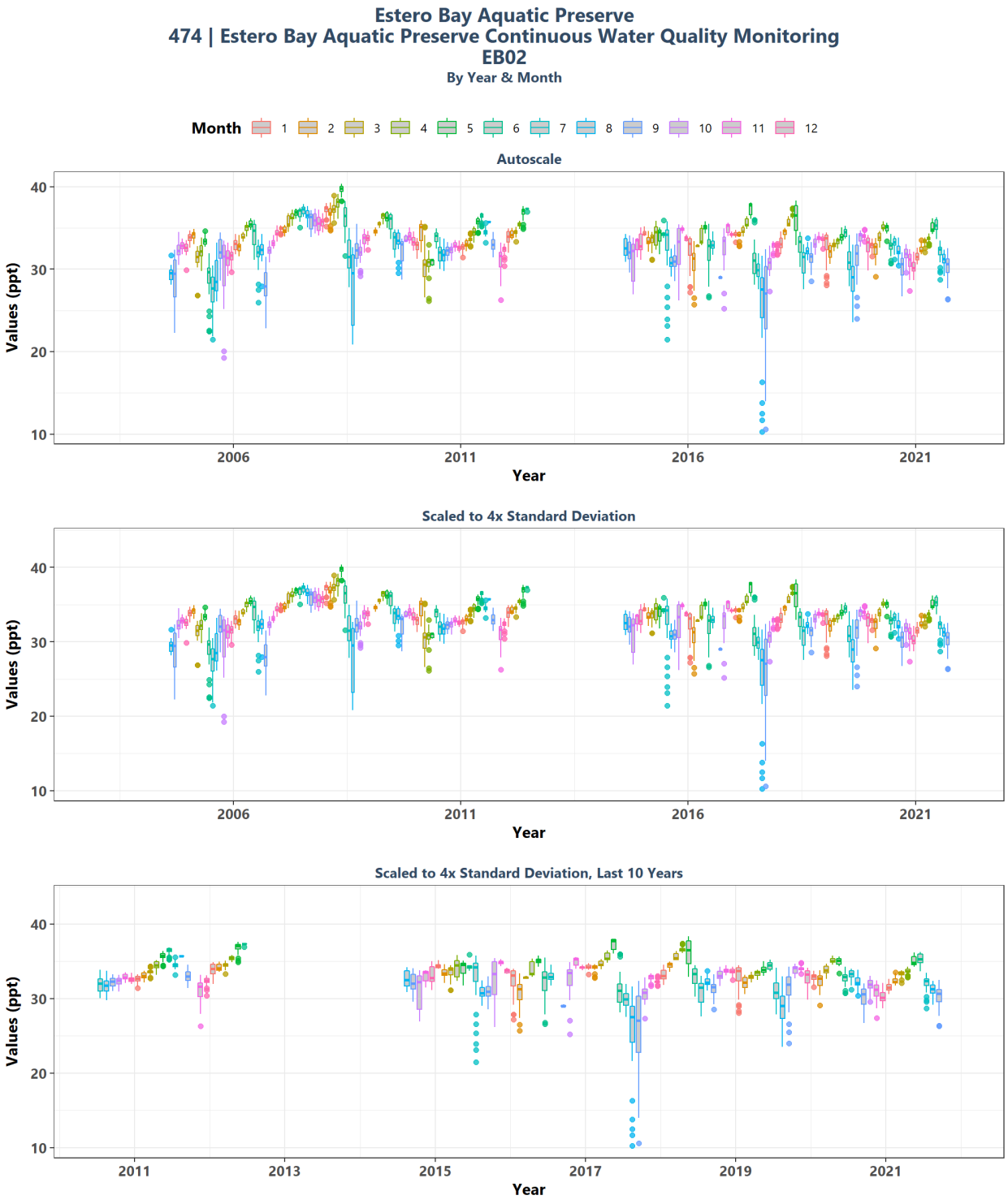


Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbmbwq
By Month

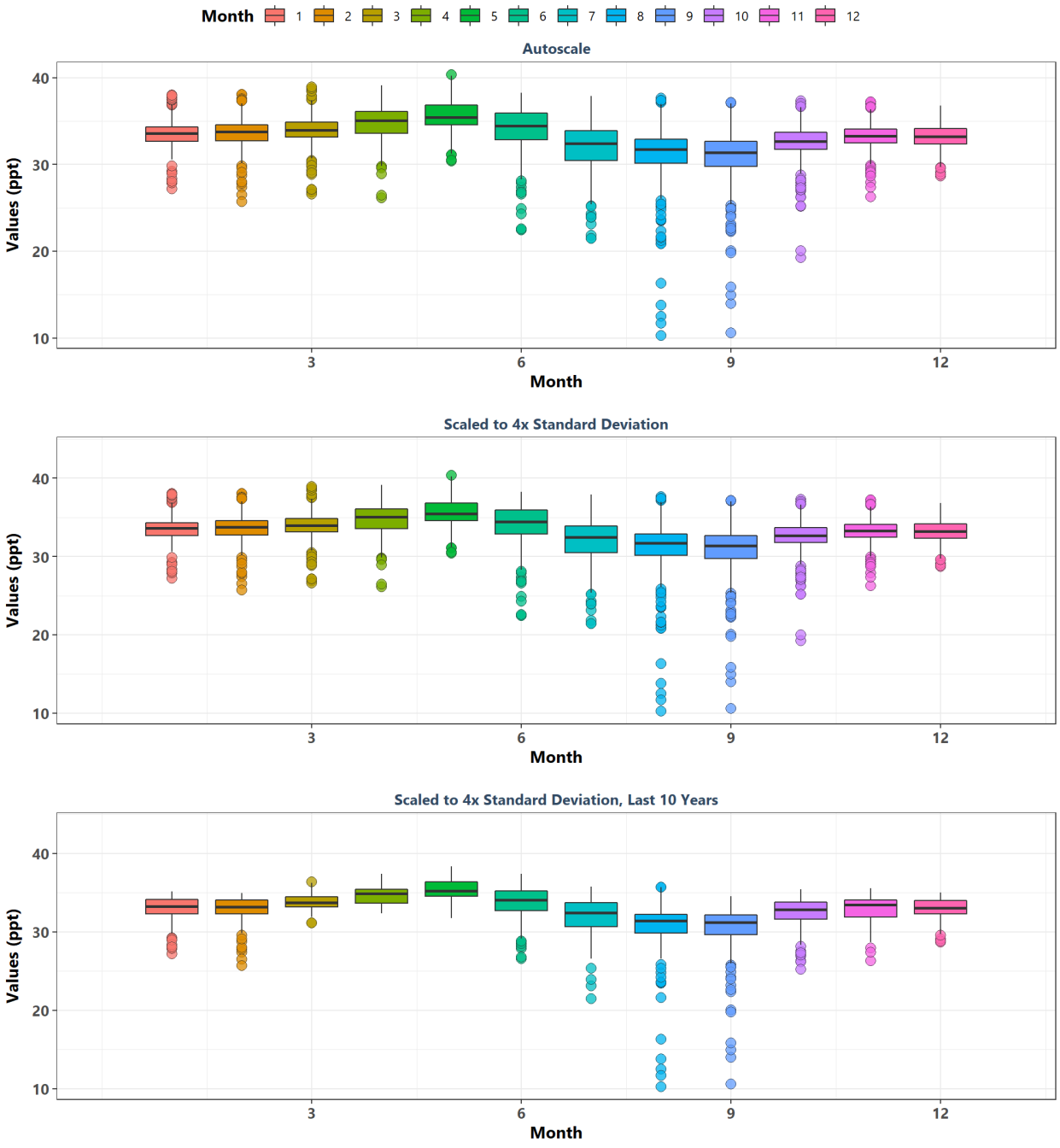


Estero Bay Aquatic Preserve
474 | Estero Bay Aquatic Preserve Continuous Water Quality Monitoring
EB02
By Year

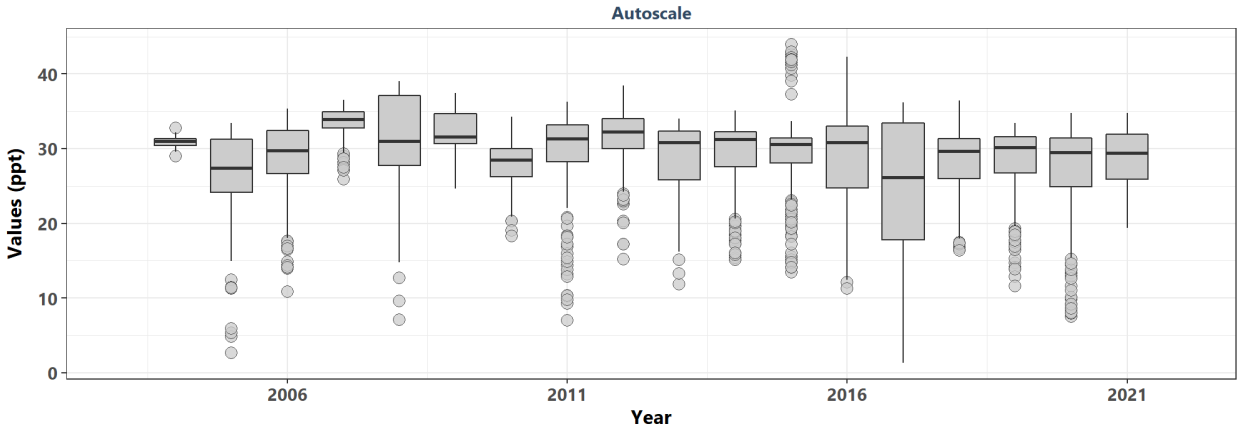


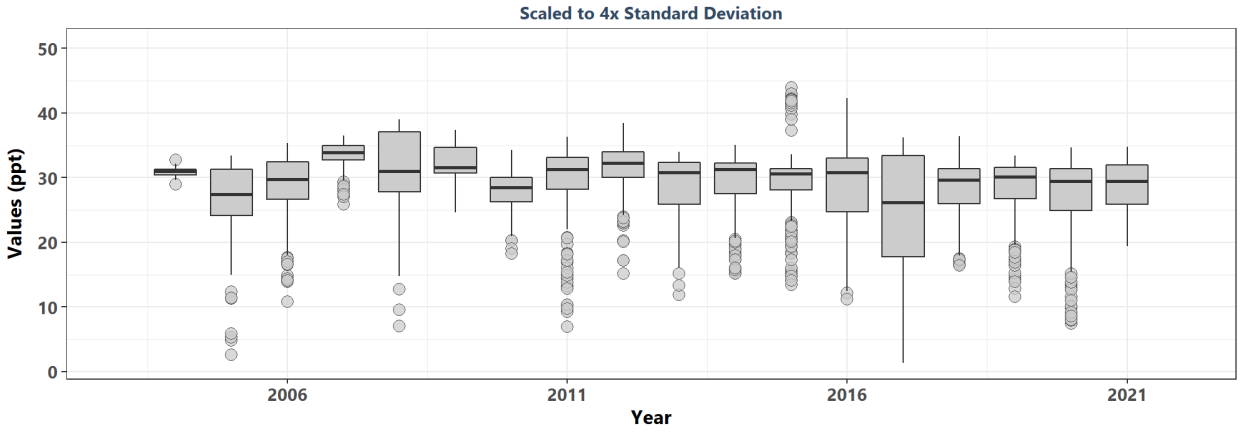

Estero Bay Aquatic Preserve
474 | Estero Bay Aquatic Preserve Continuous Water Quality Monitoring
EB02
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year

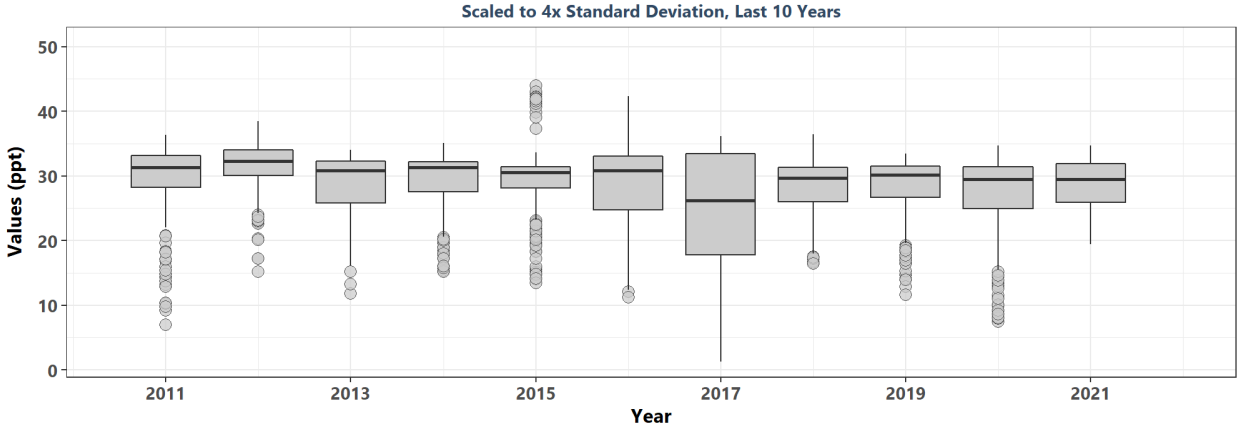
Estero Bay Aquatic Preserve
474 | Estero Bay Aquatic Preserve Continuous Water Quality Monitoring
EB02
By Month

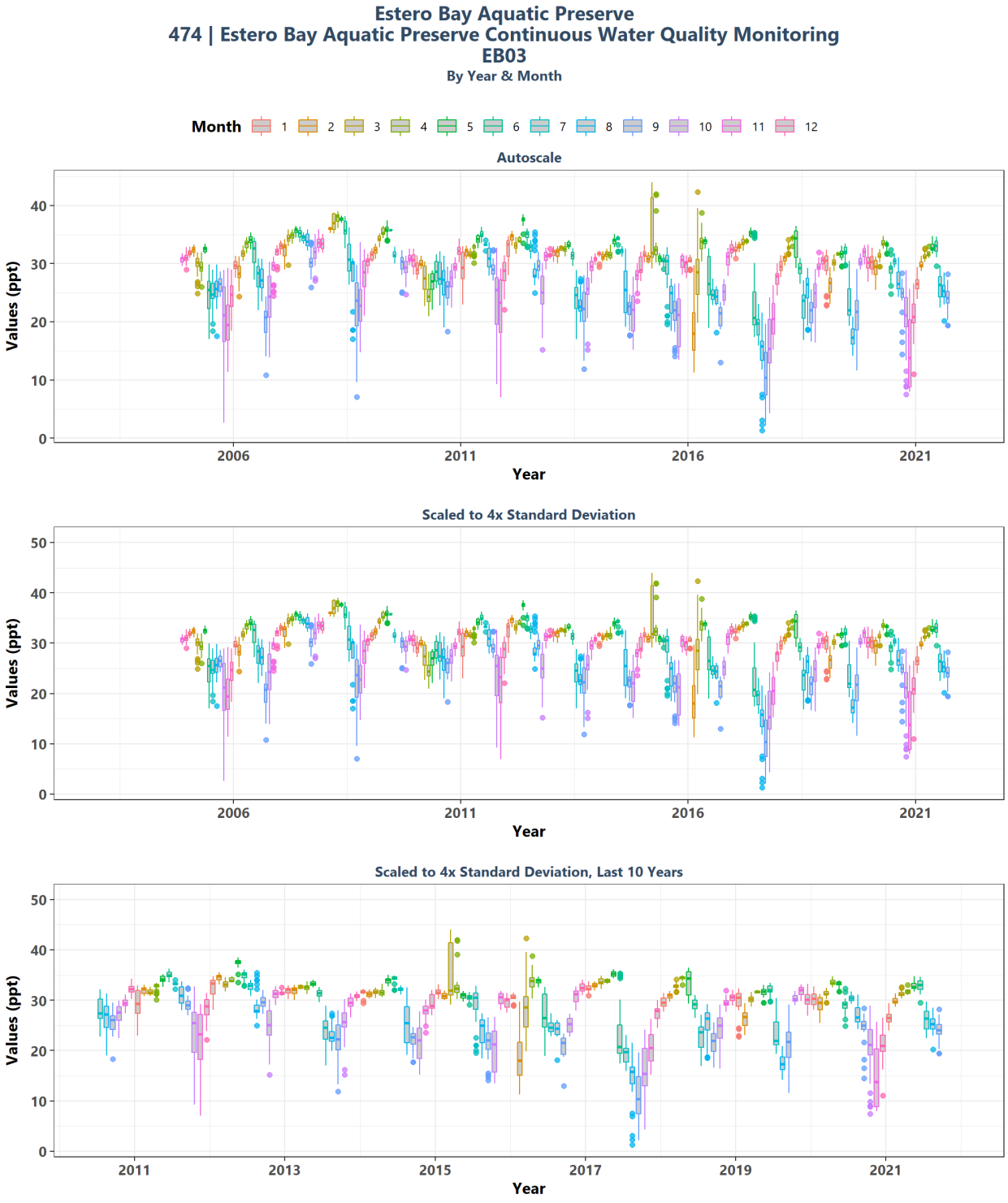


Estero Bay Aquatic Preserve
474 | Estero Bay Aquatic Preserve Continuous Water Quality Monitoring
EB03
By Year

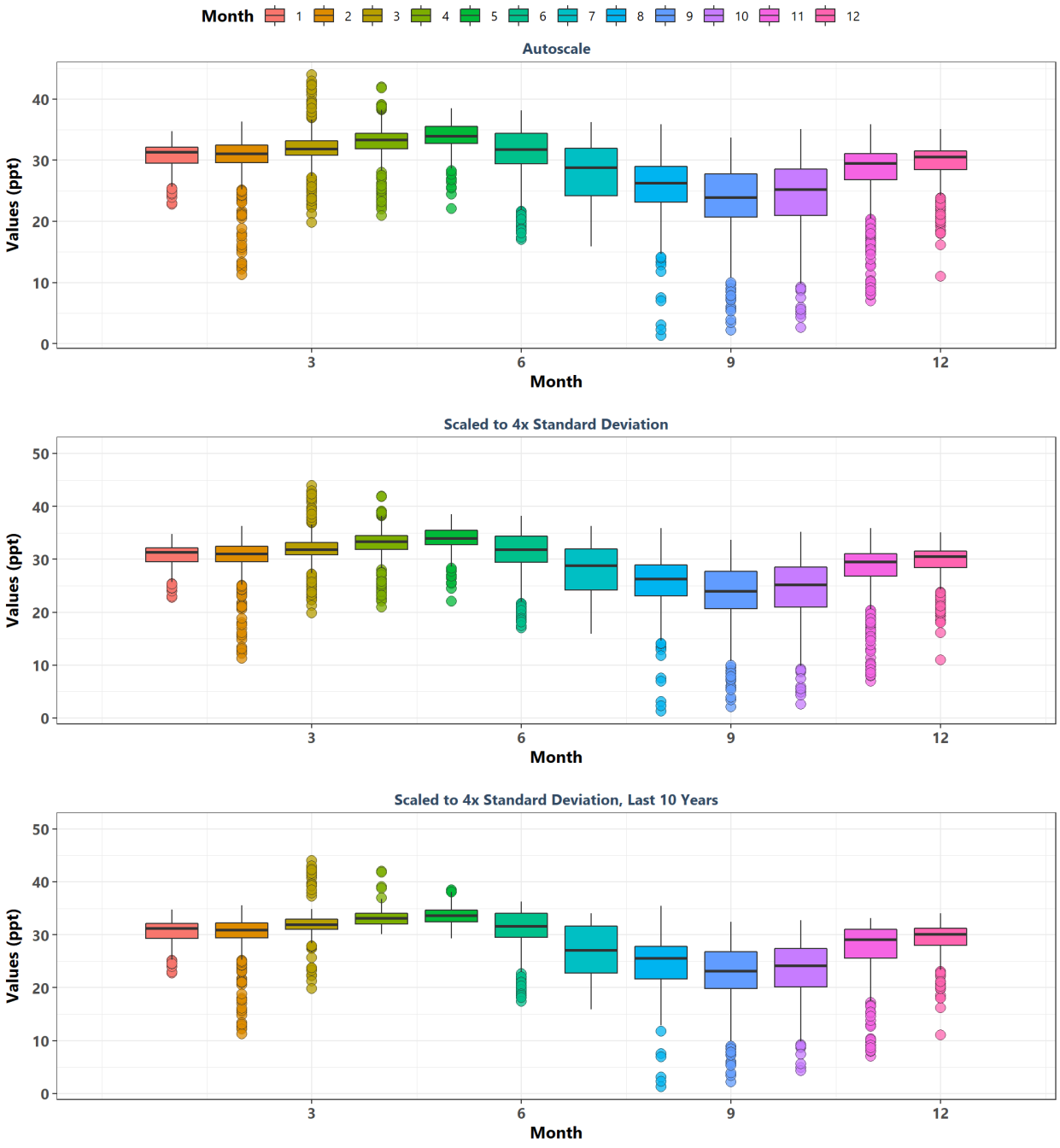




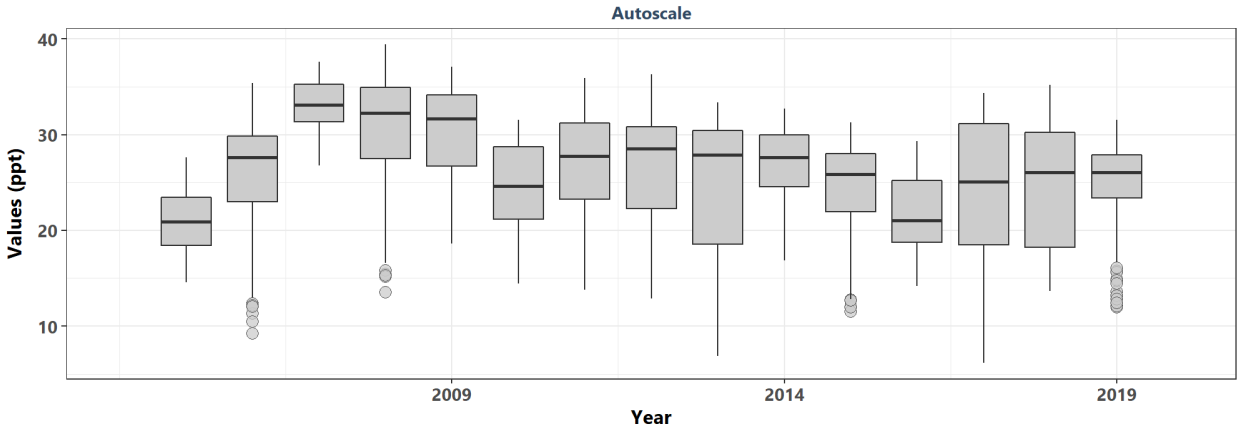


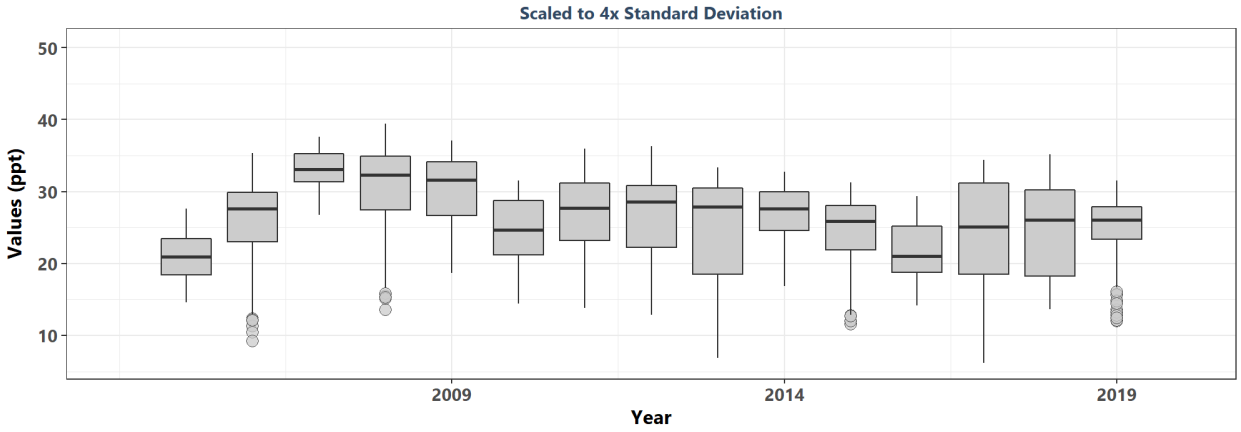

Estero Bay Aquatic Preserve
474 | Estero Bay Aquatic Preserve Continuous Water Quality Monitoring
EB03
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year

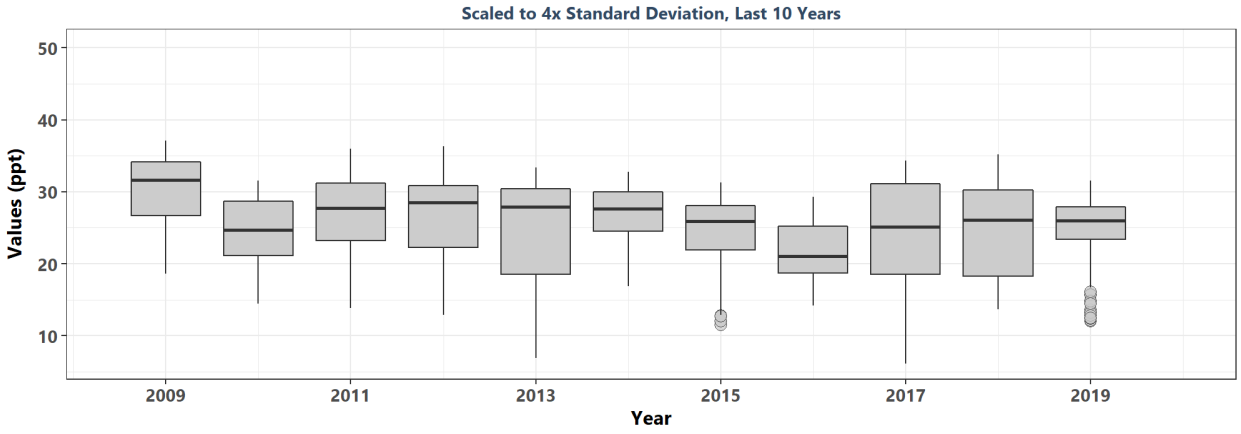
Estero Bay Aquatic Preserve
474 | Estero Bay Aquatic Preserve Continuous Water Quality Monitoring
EB03
By Month



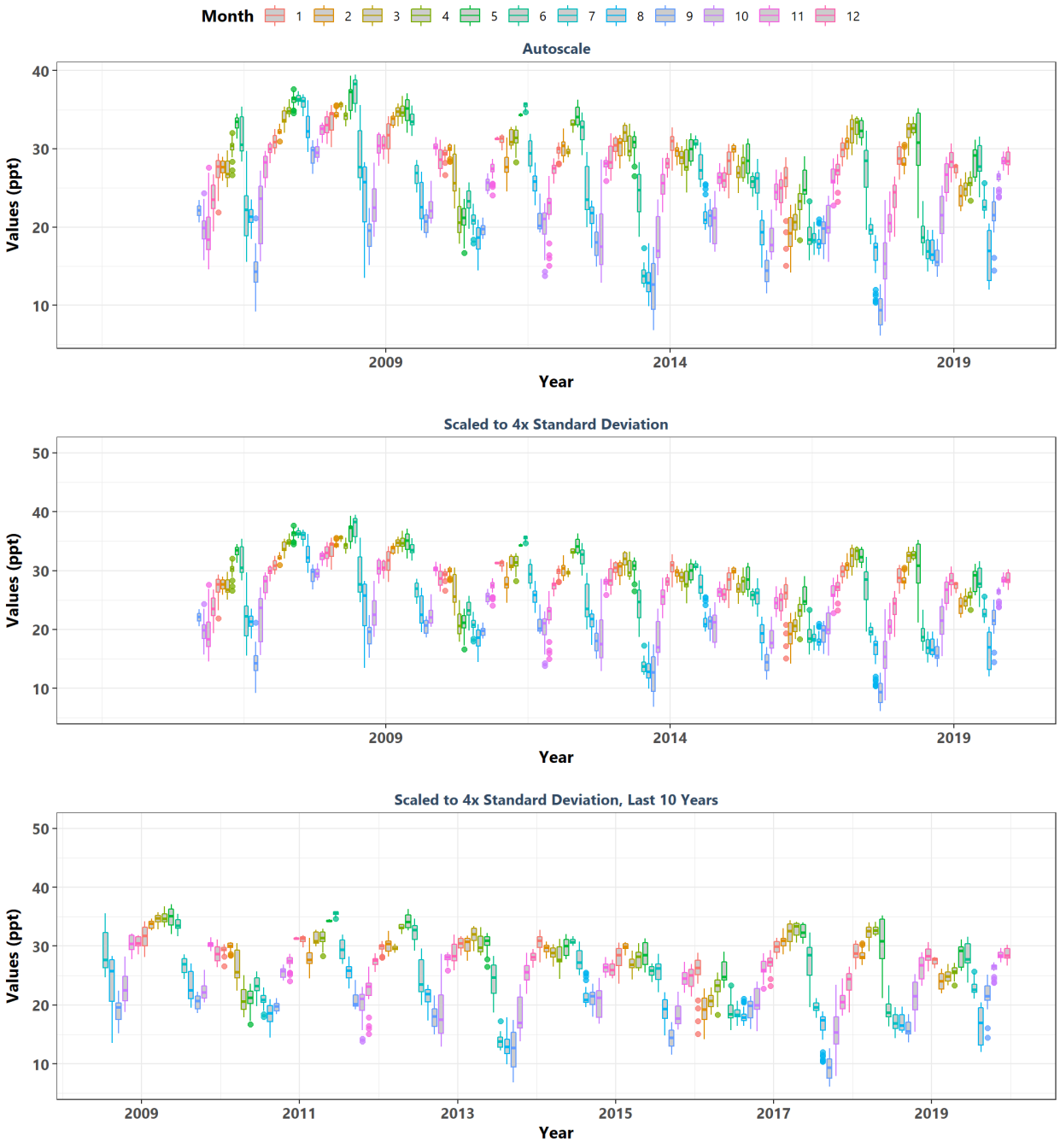
Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP1A
By Year



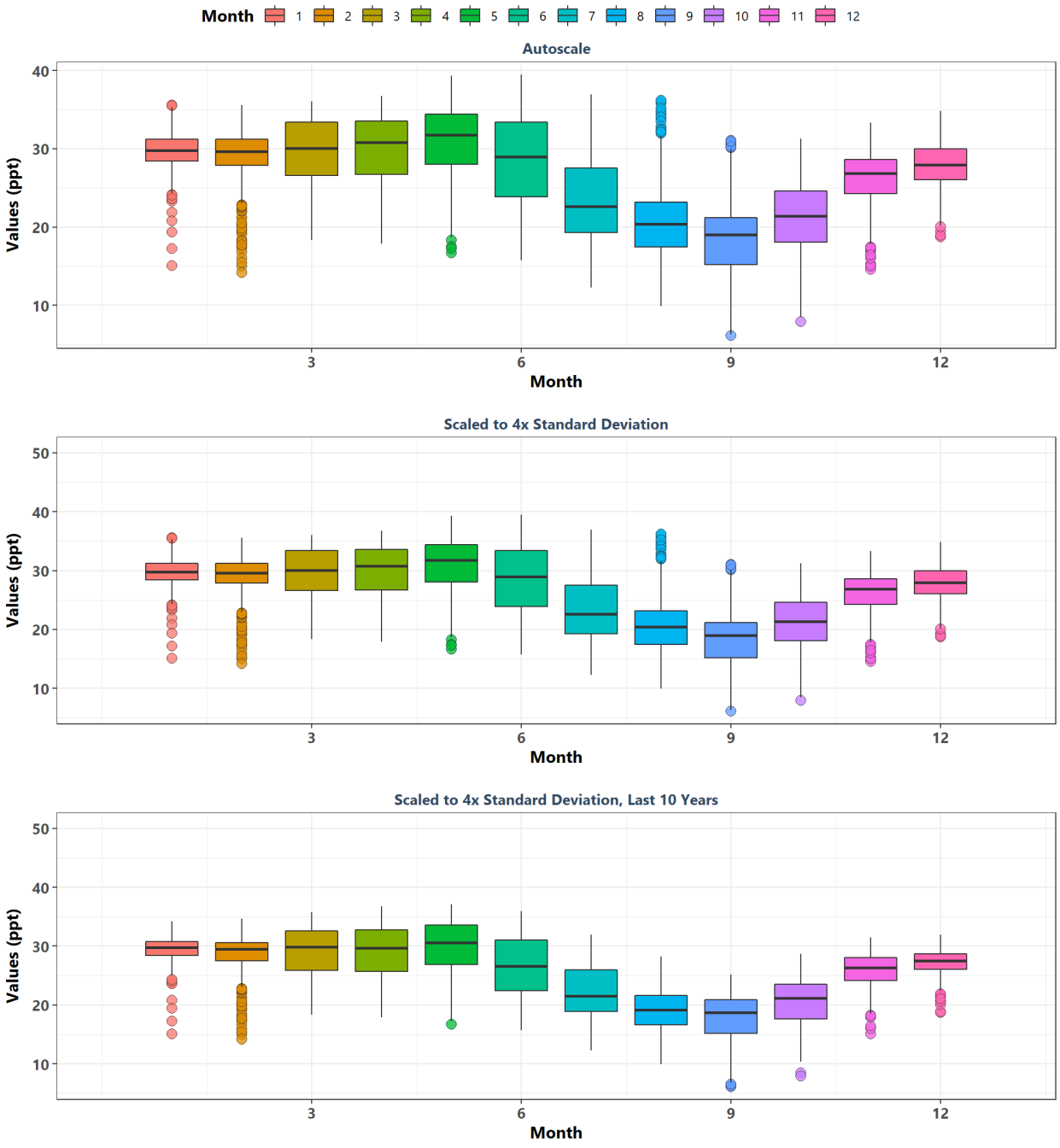




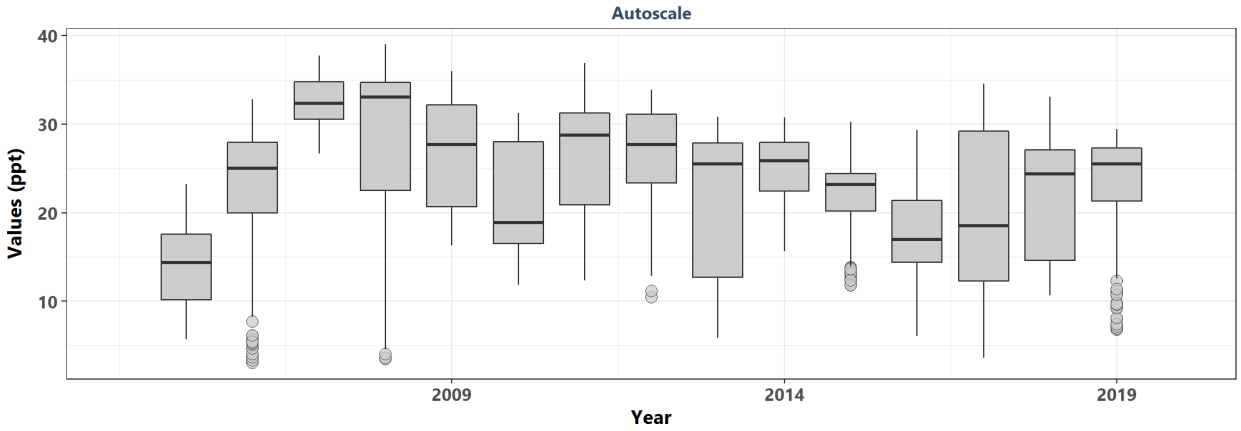
Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP1A
By Year & Month

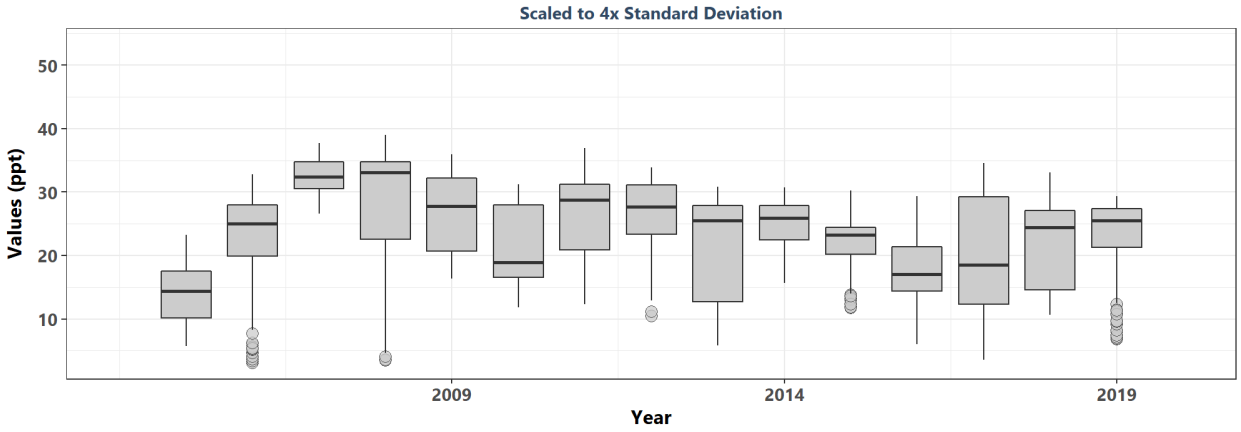


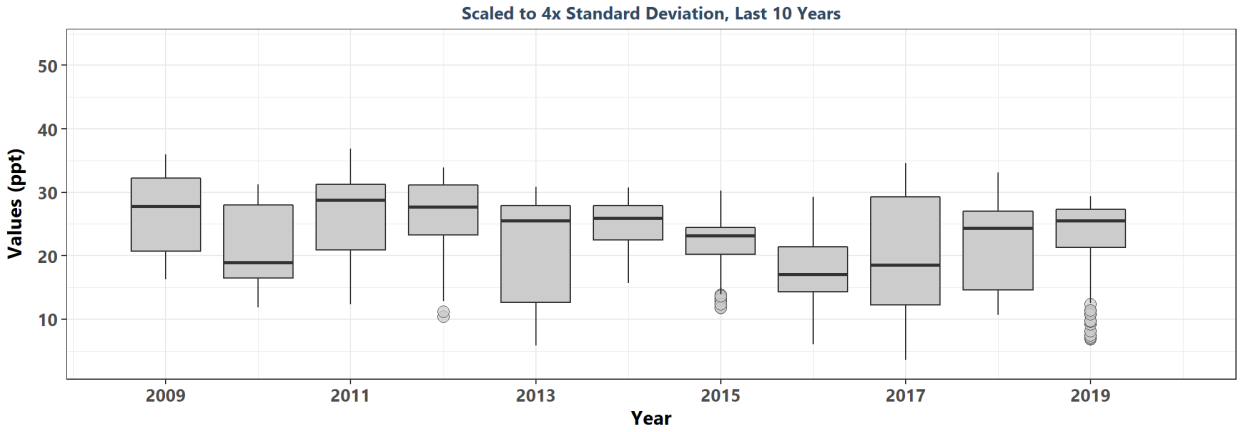
Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP1A
By Month



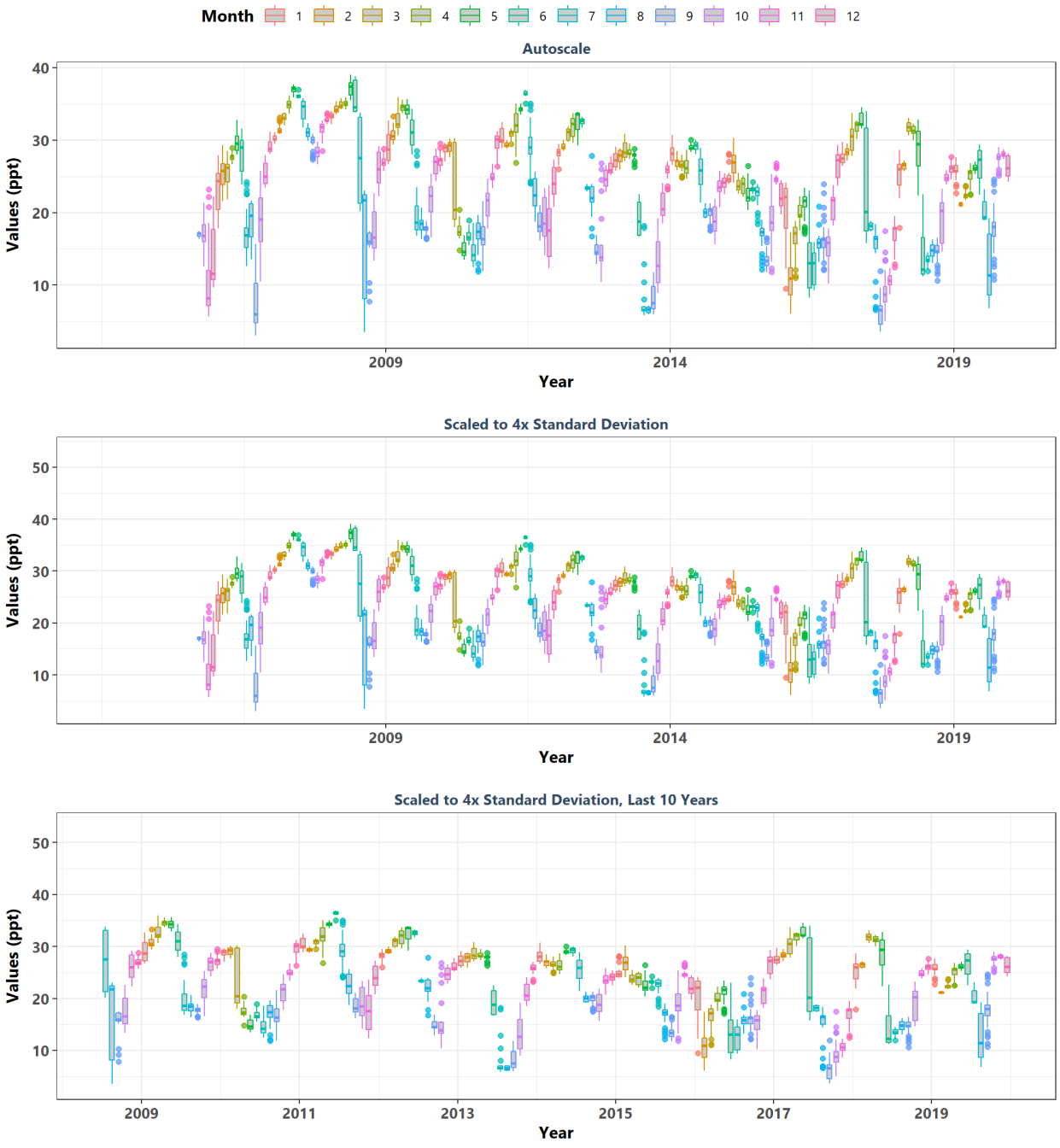
Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP2B
By Year



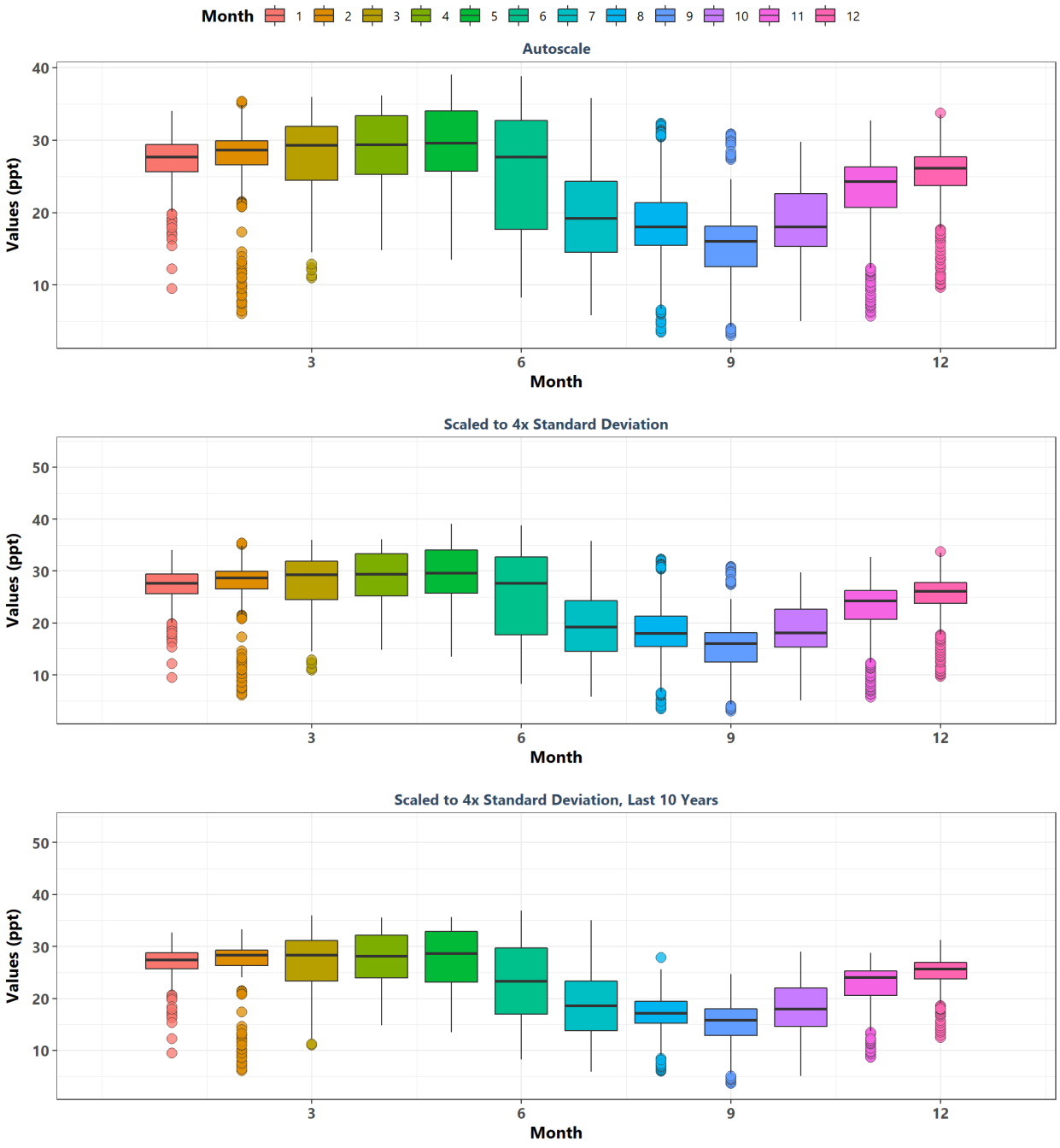




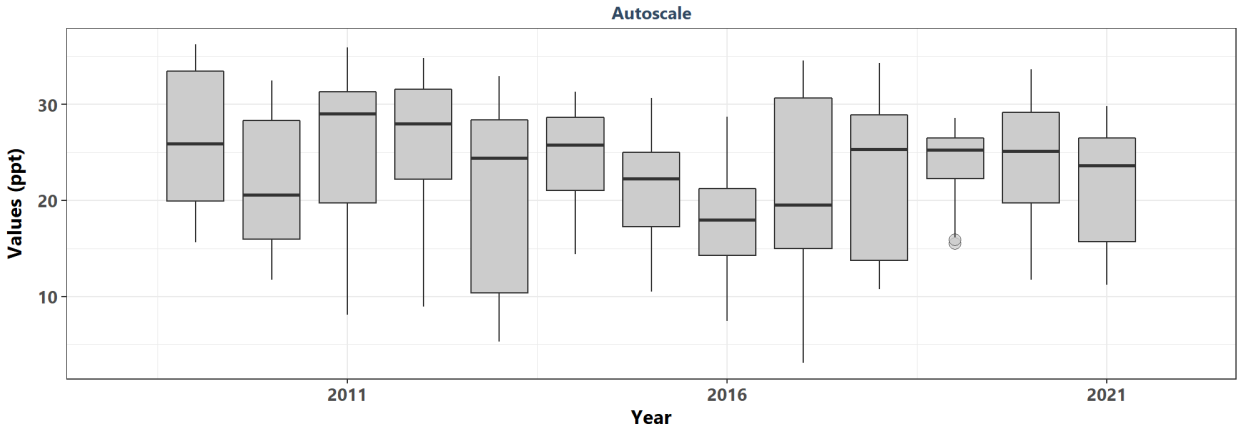
Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP2B
By Year & Month

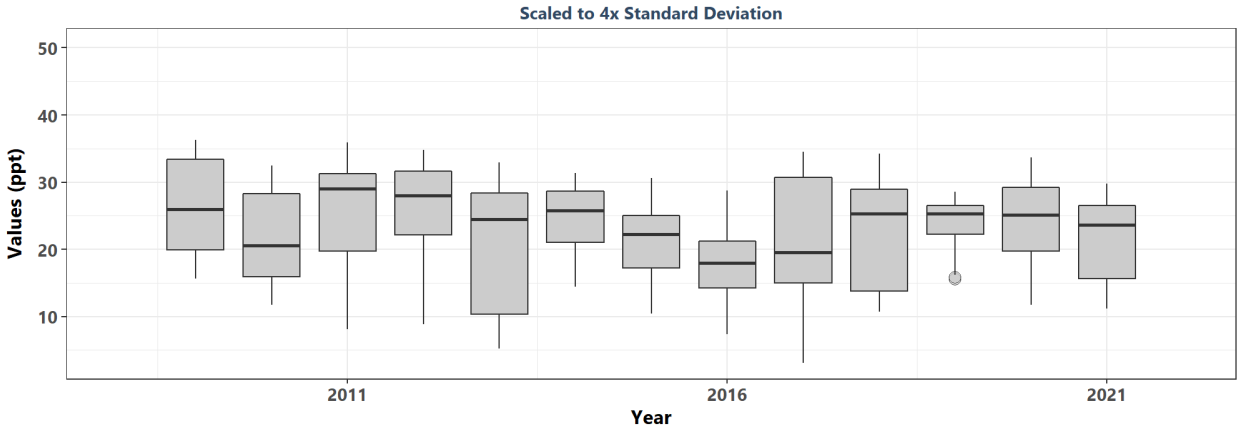


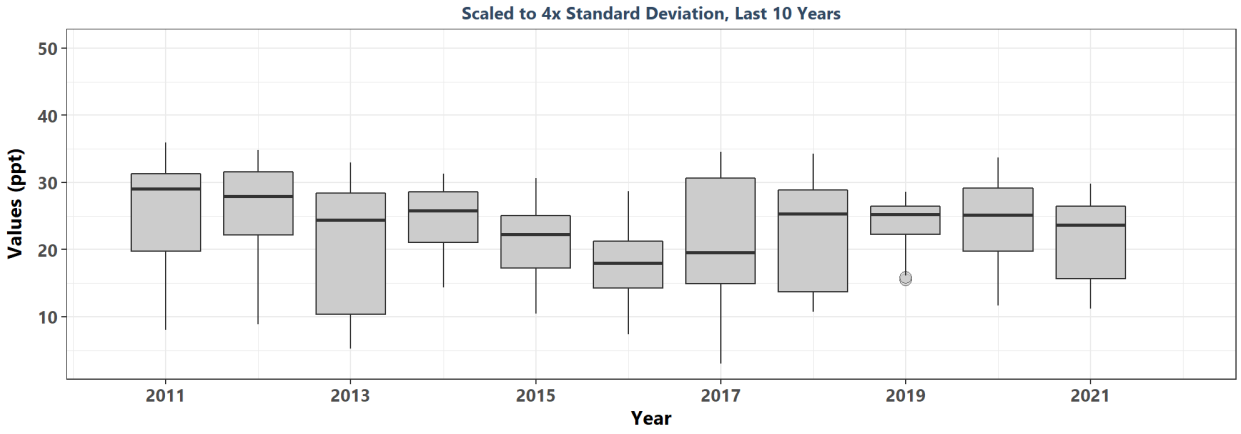
Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP2B
By Month



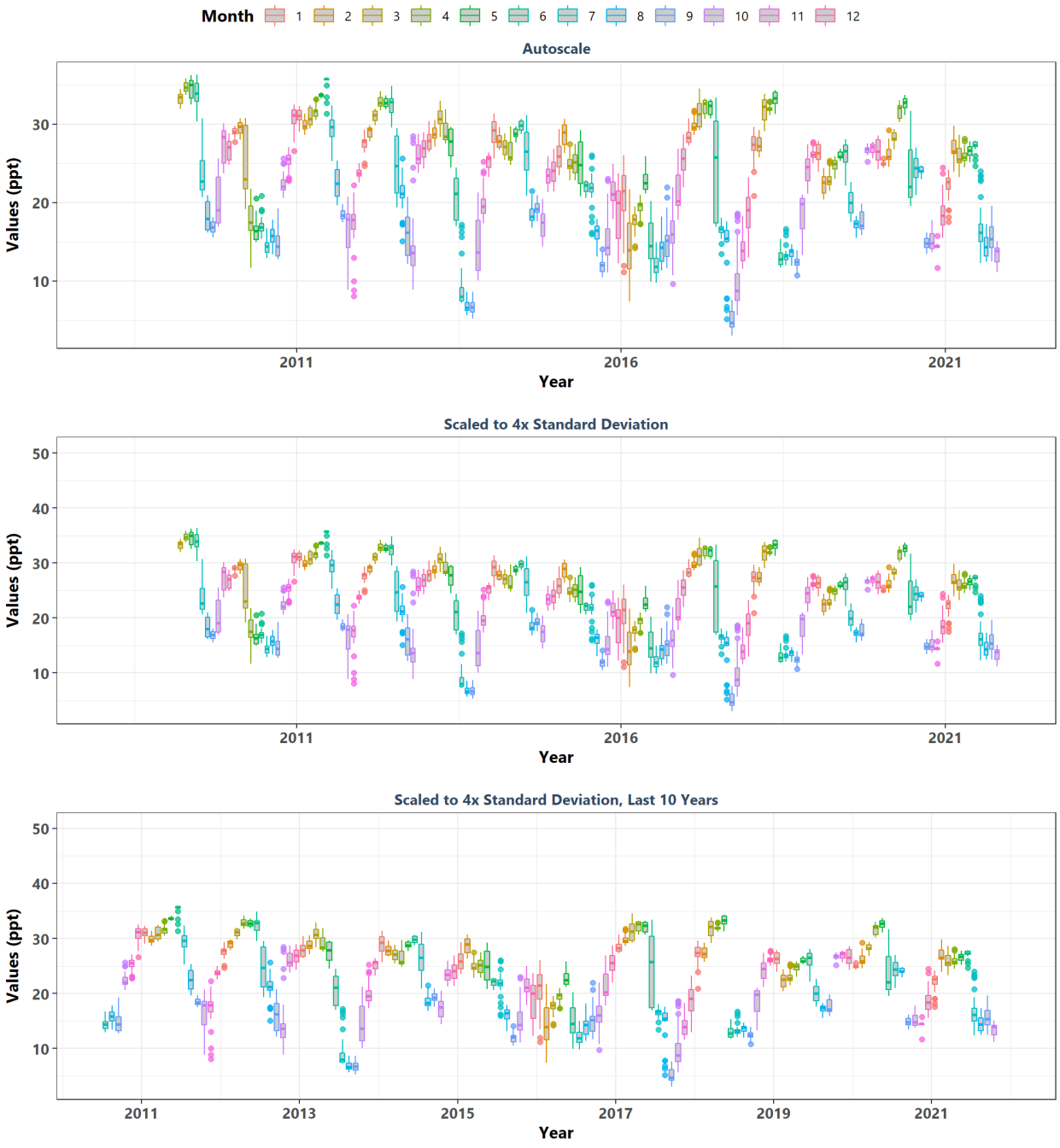
Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP3C
By Year



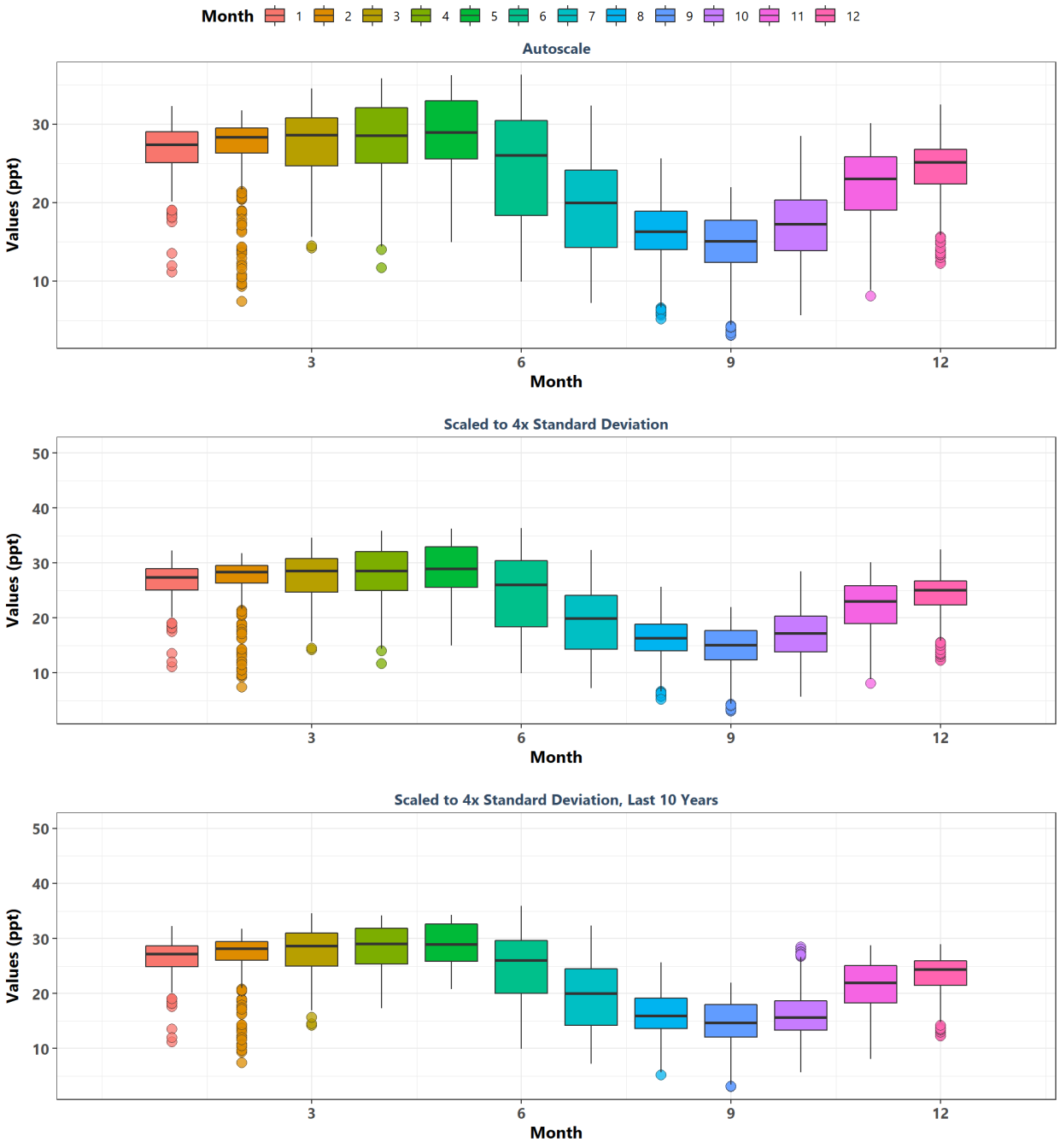




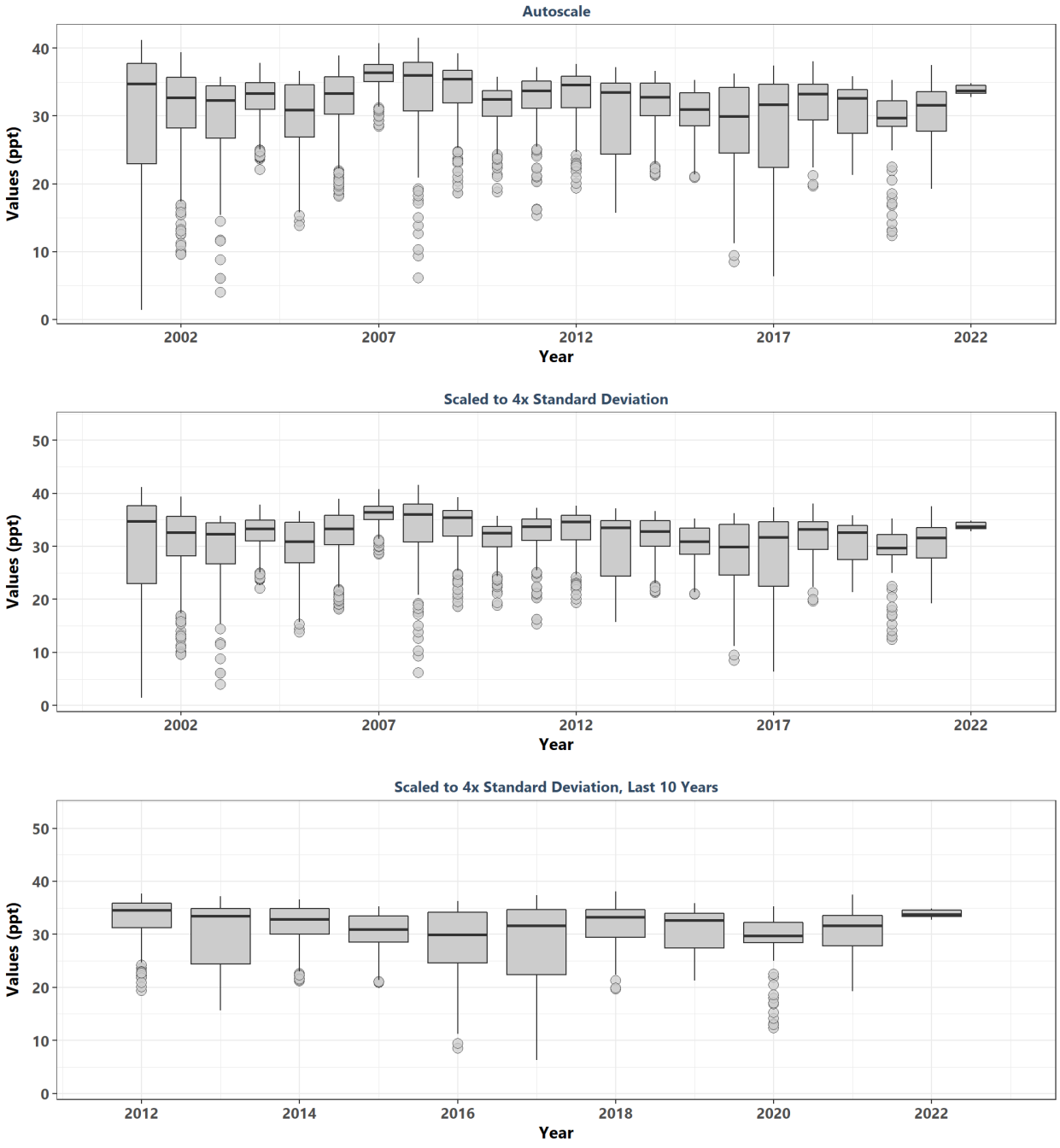
Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP3C
By Year & Month

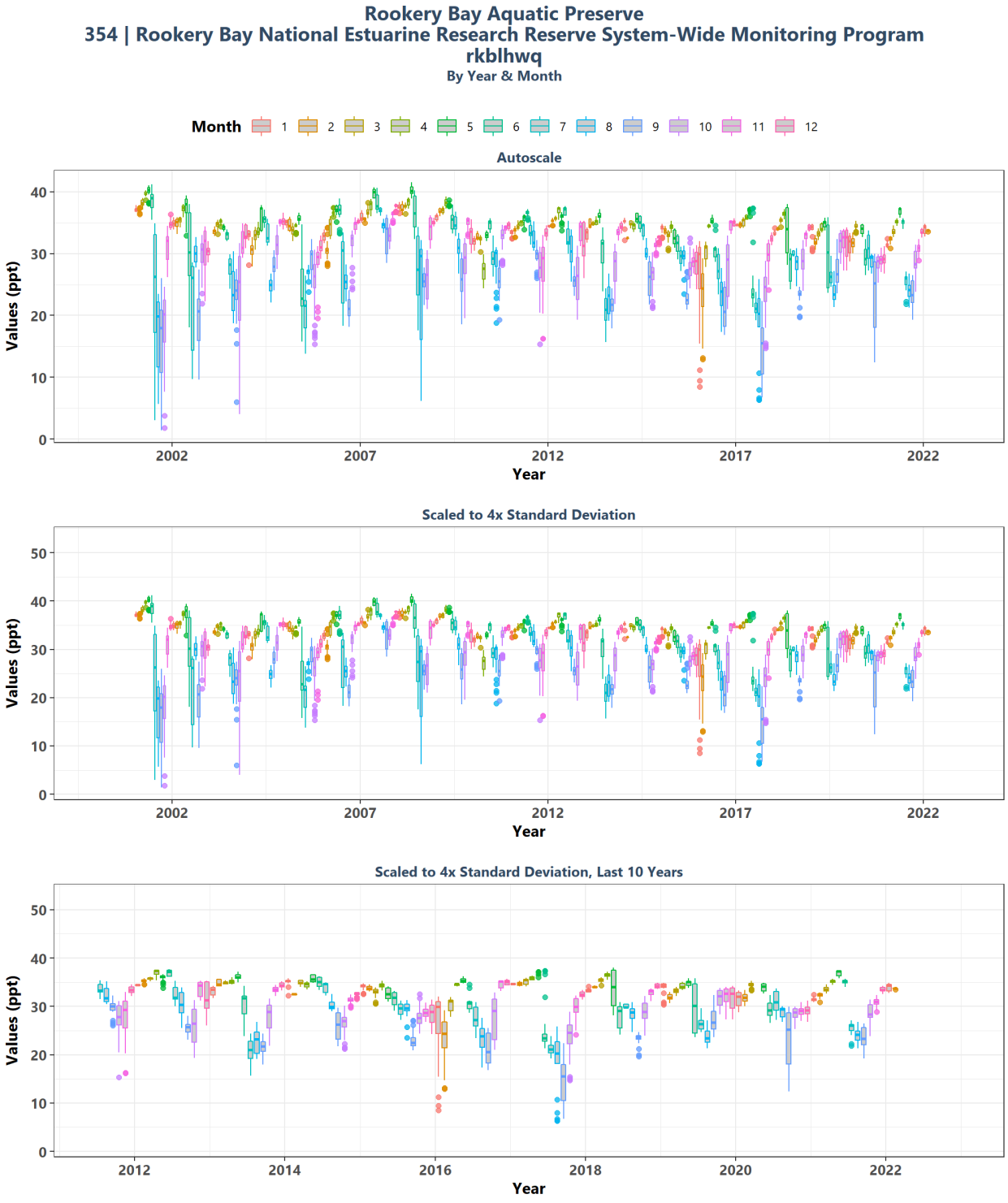


Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP3C
By Month

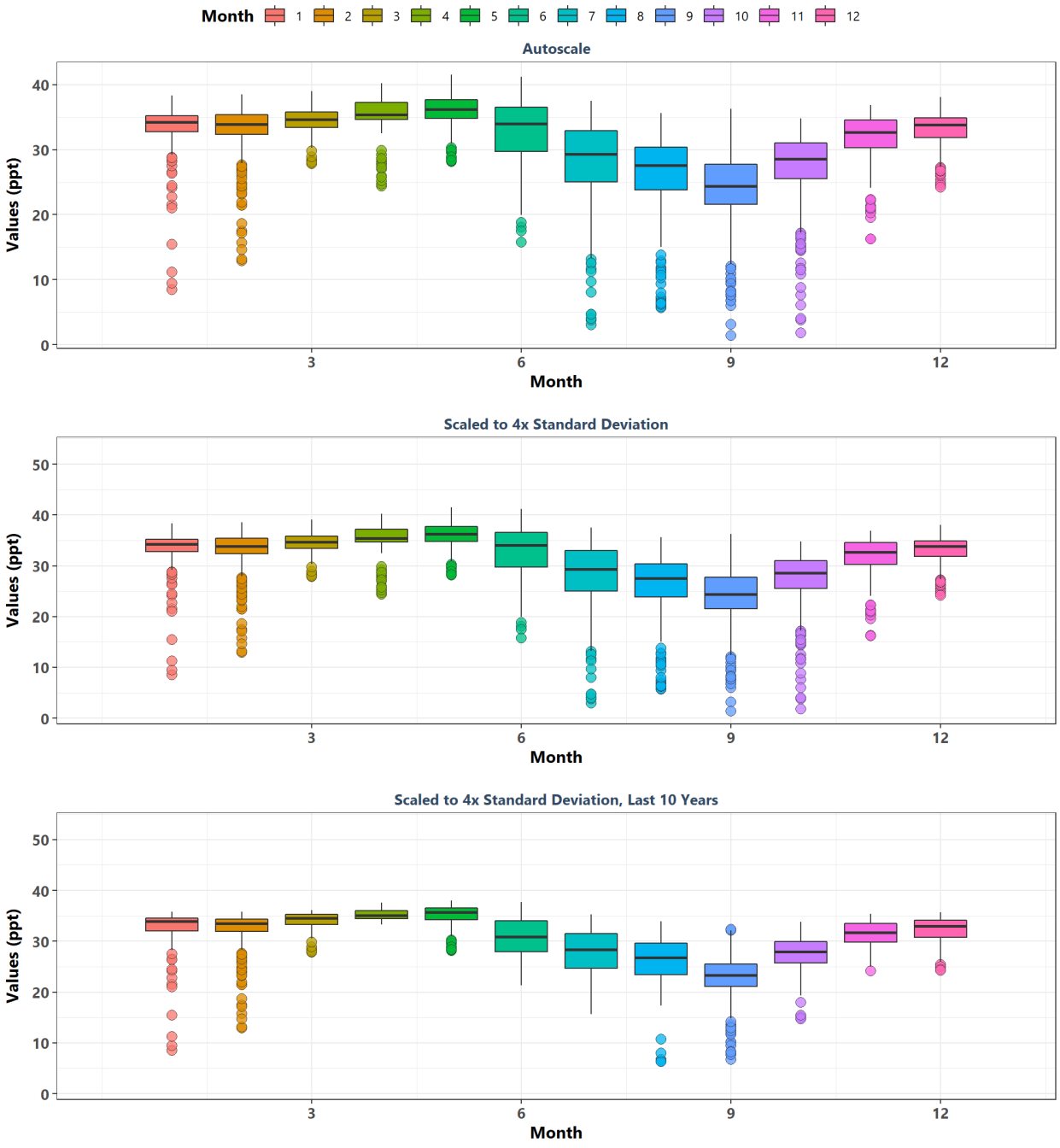


Rookery Bay Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkblhwq
By Year

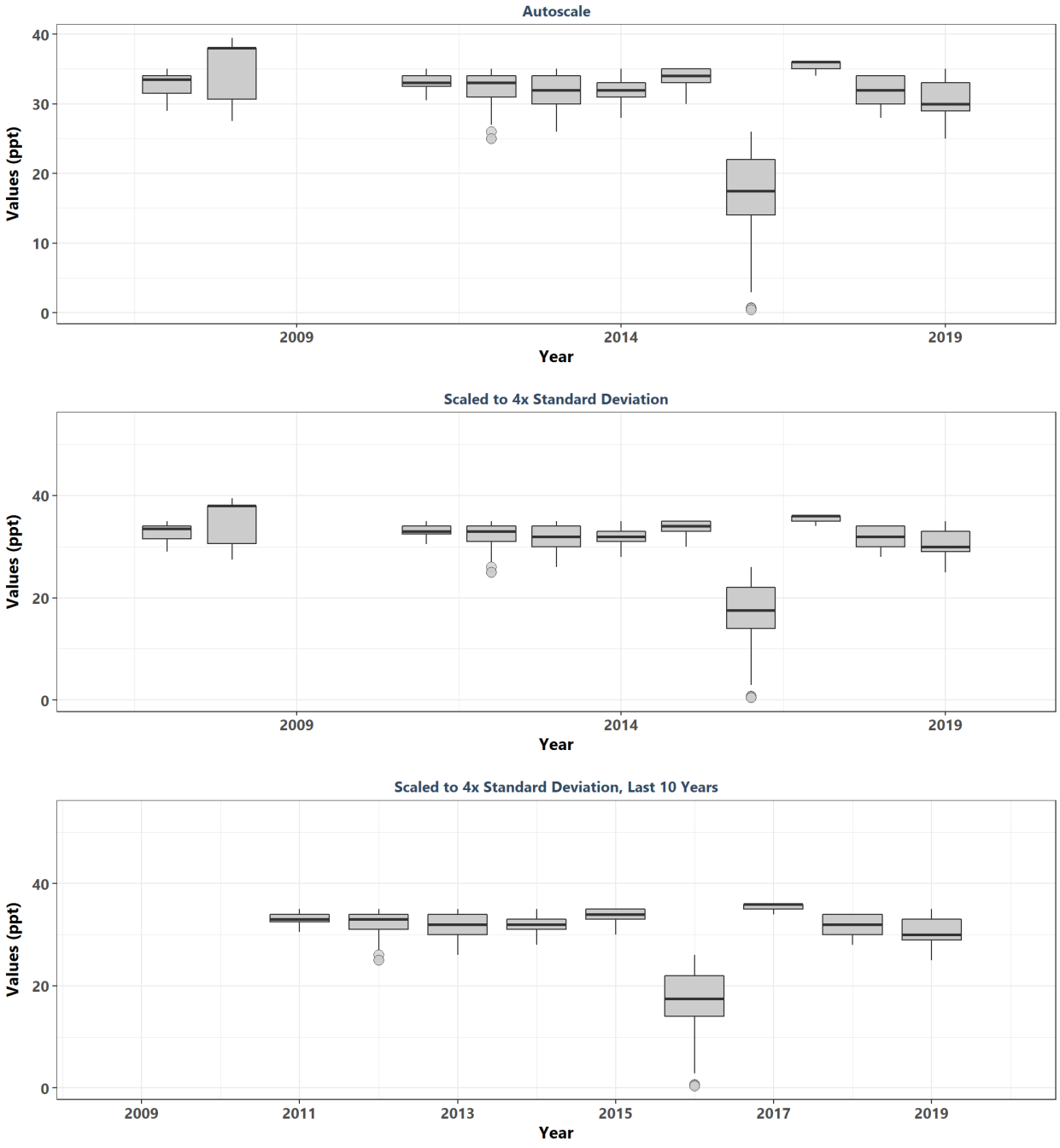


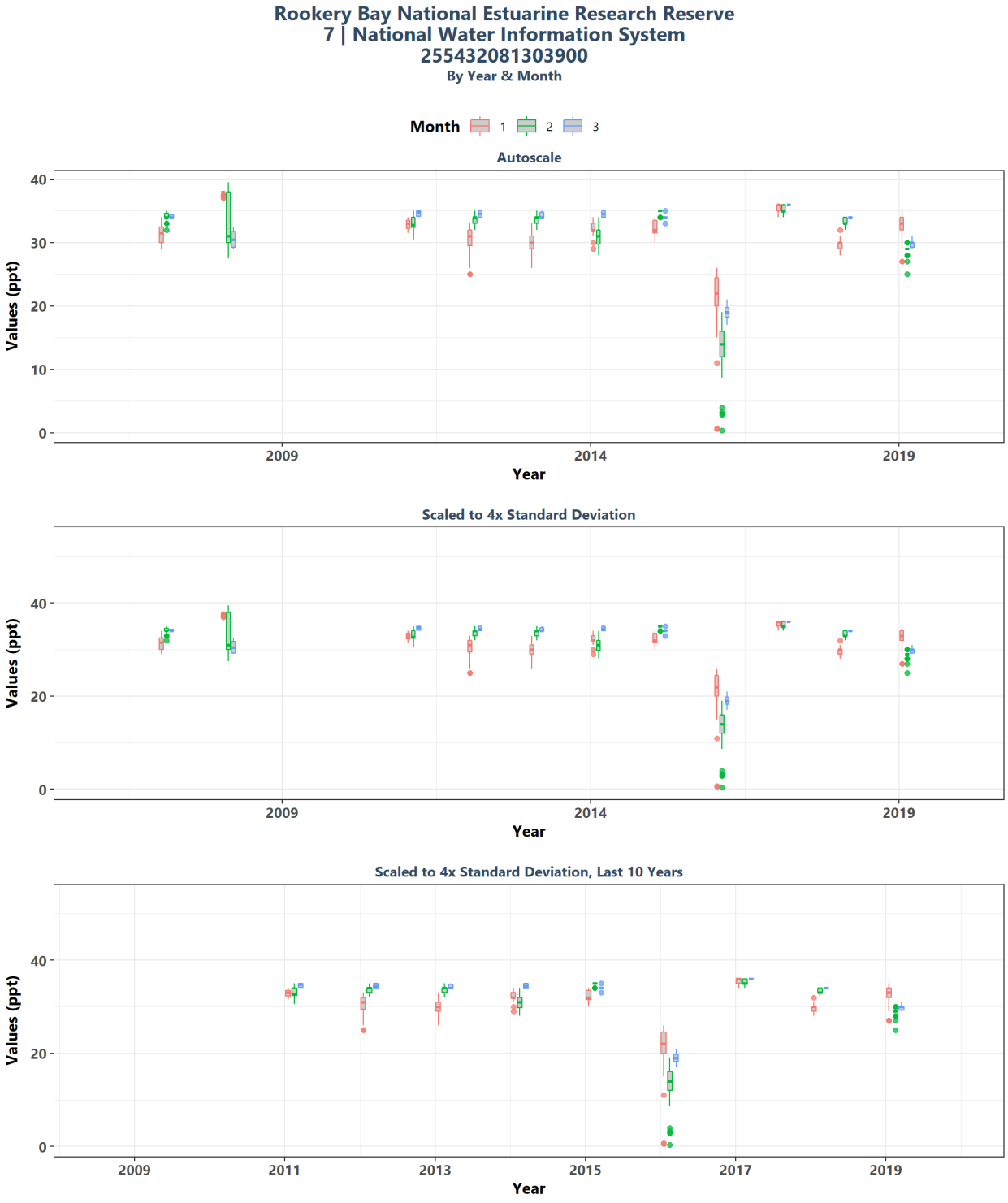

Rookery Bay Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkblhwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year

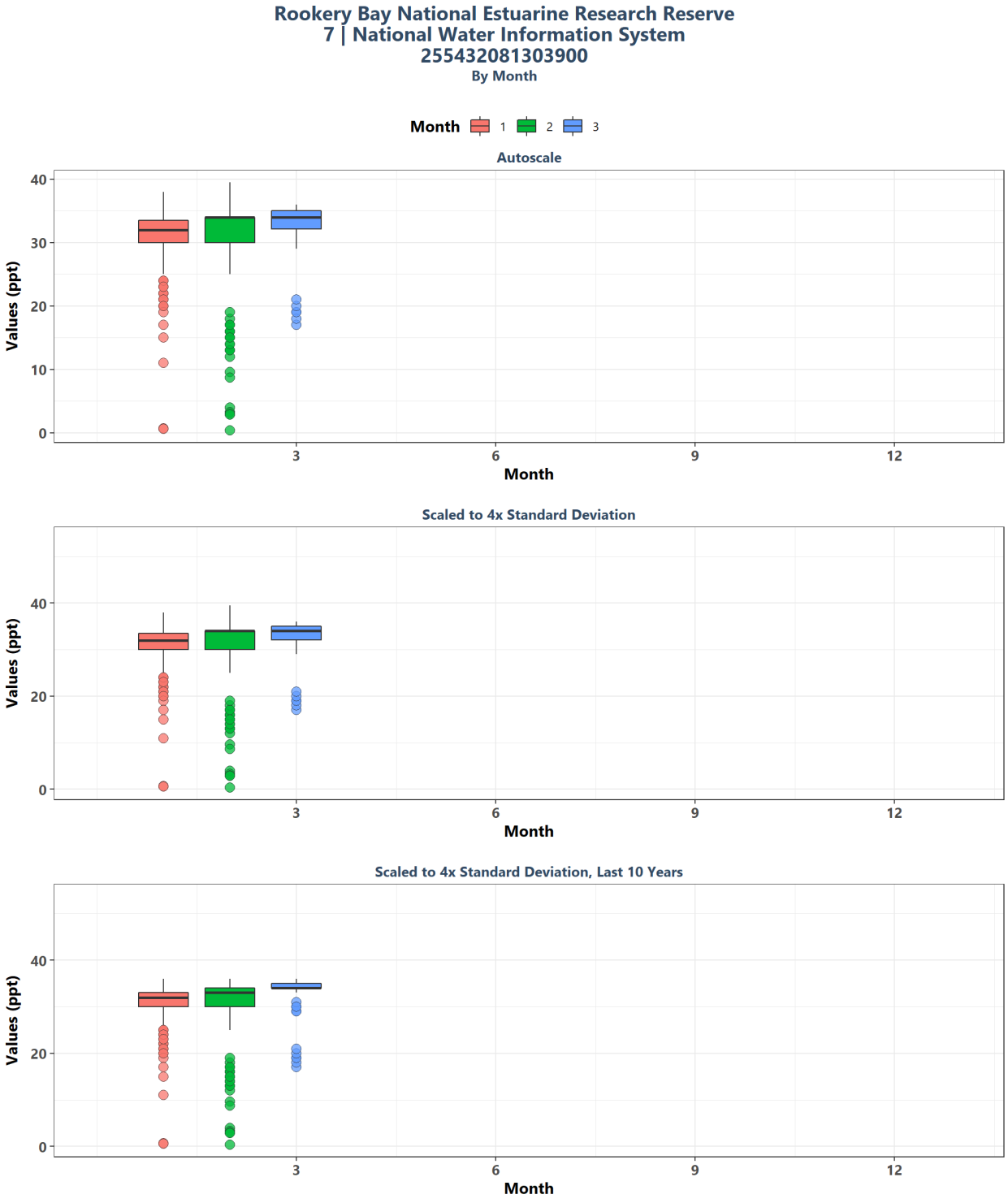
Rookery Bay Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkblhwq
By Month



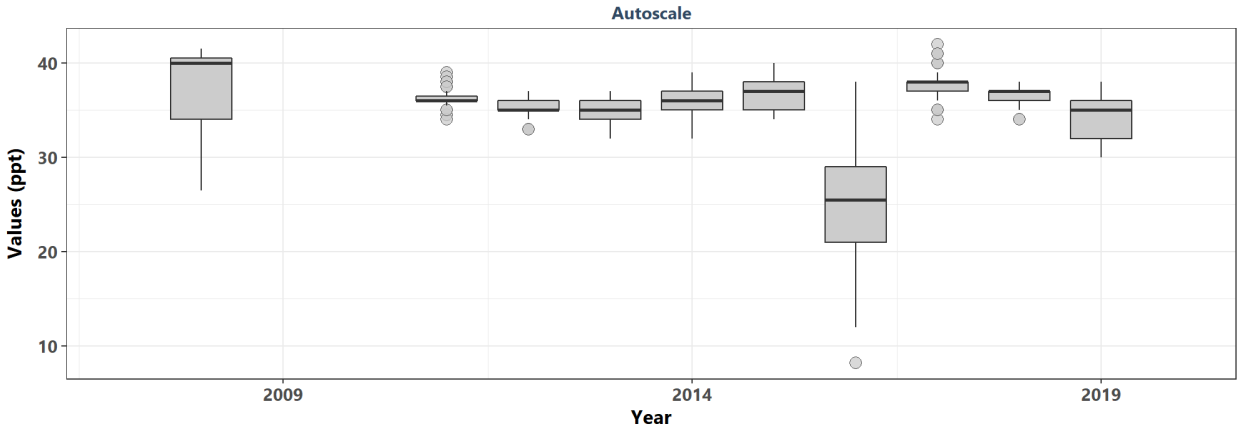
Rookery Bay National Estuarine Research Reserve
7 | National Water Information System
255432081303900
By Year

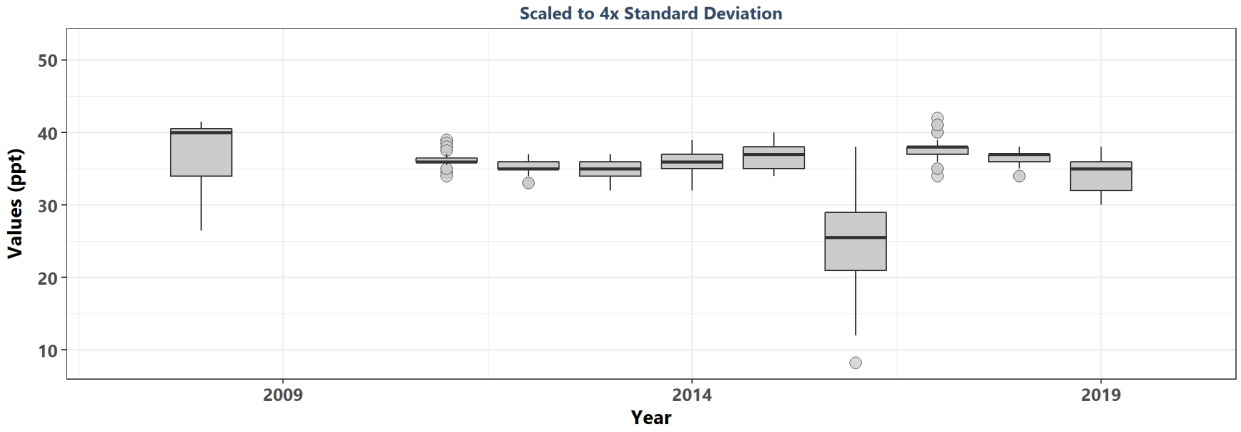


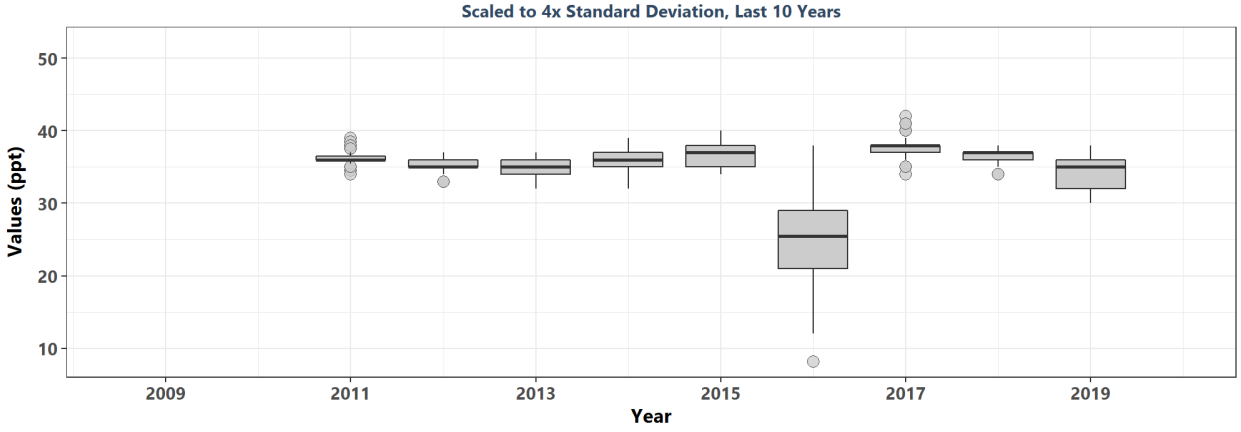

Rookery Bay National Estuarine Research Reserve
7 | National Water Information System
255432081303900
By Year & Month
Month 1 2 3
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year

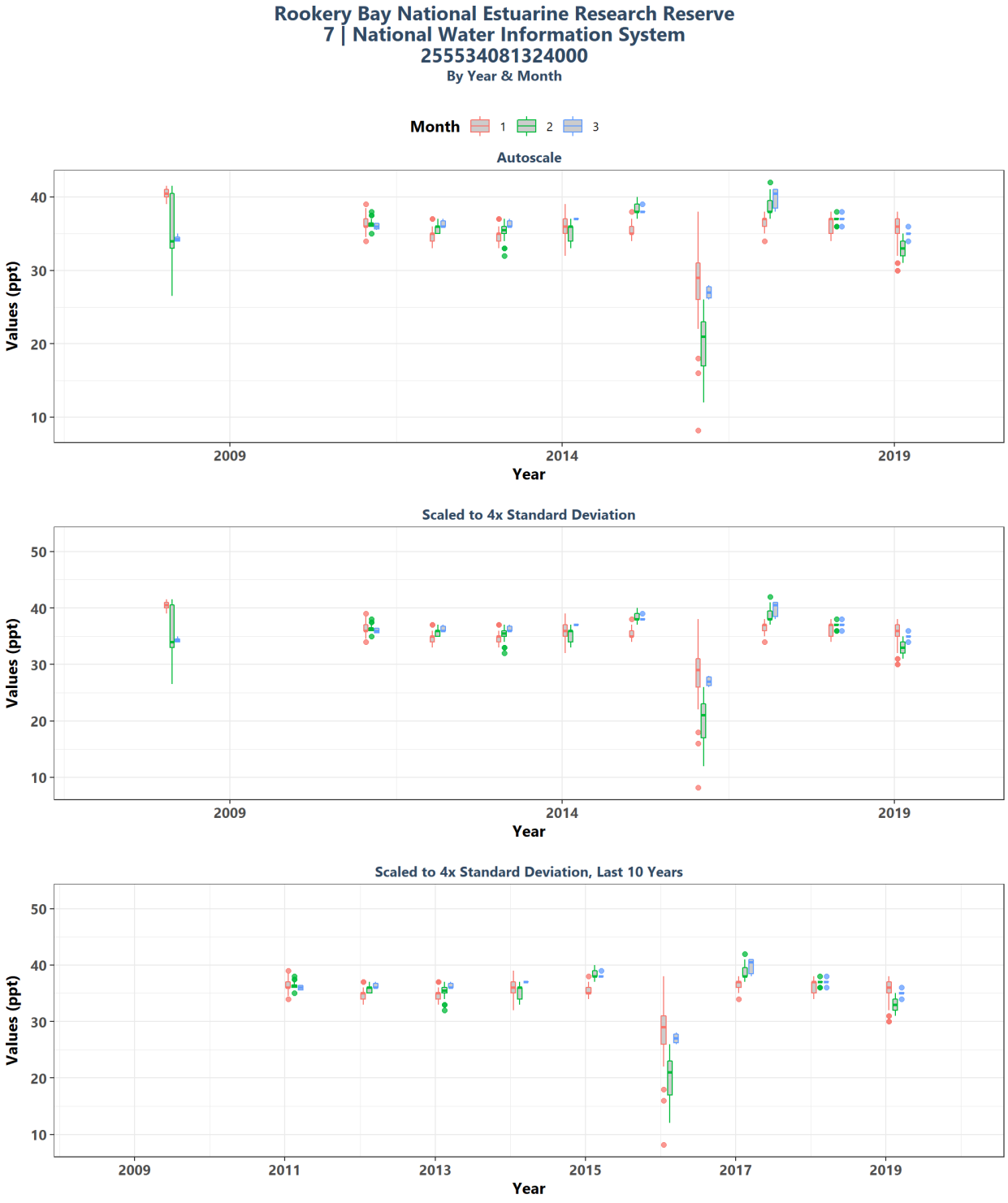

Rookery Bay National Estuarine Research Reserve
7 | National Water Information System
255432081303900
By Month
Month 1 2 3
Autoscale
Values (ppt)
Month
Scaled to 4x Standard Deviation
Values (ppt)
Month
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Month

Rookery Bay National Estuarine Research Reserve
7 | National Water Information System
255534081324000
By Year

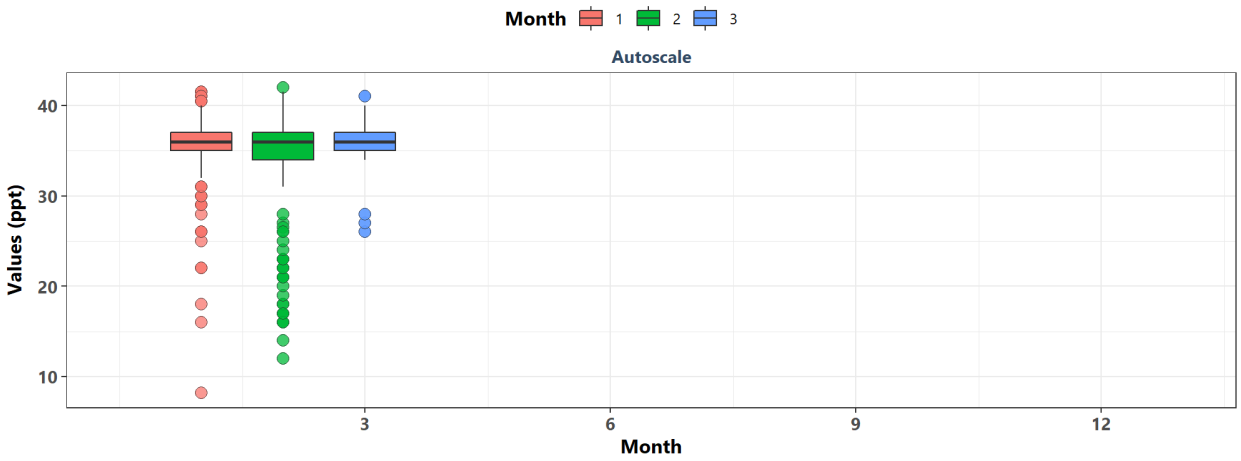


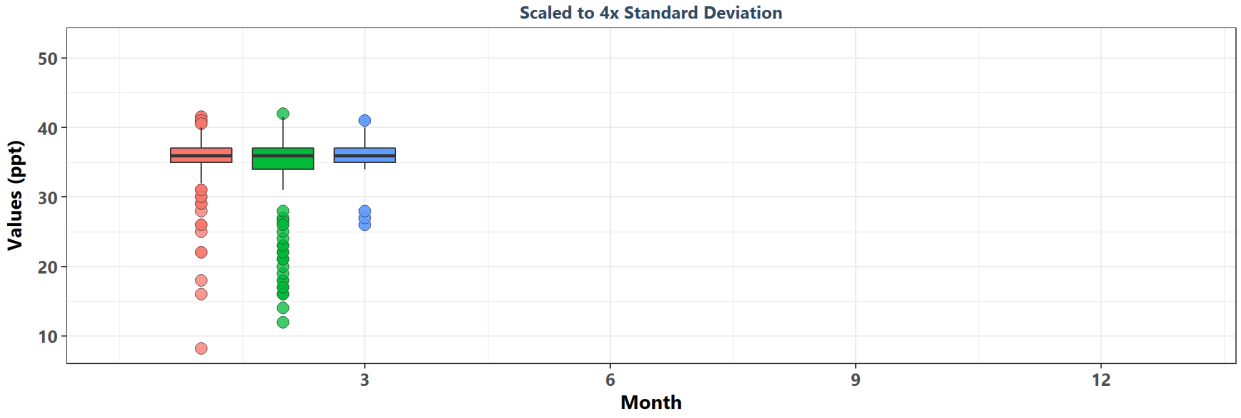


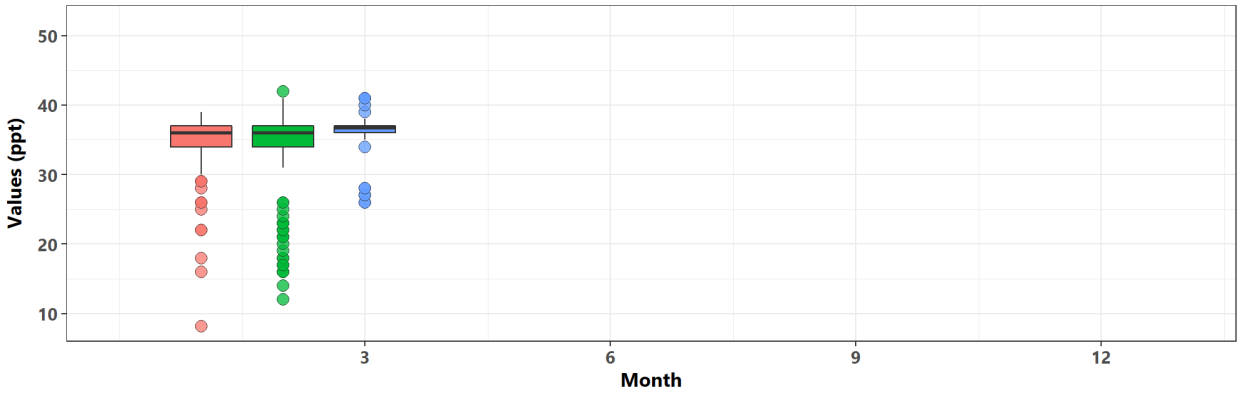



Rookery Bay National Estuarine Research Reserve
7 | National Water Information System
255534081324000
By Year & Month
Month 1 2 3
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year

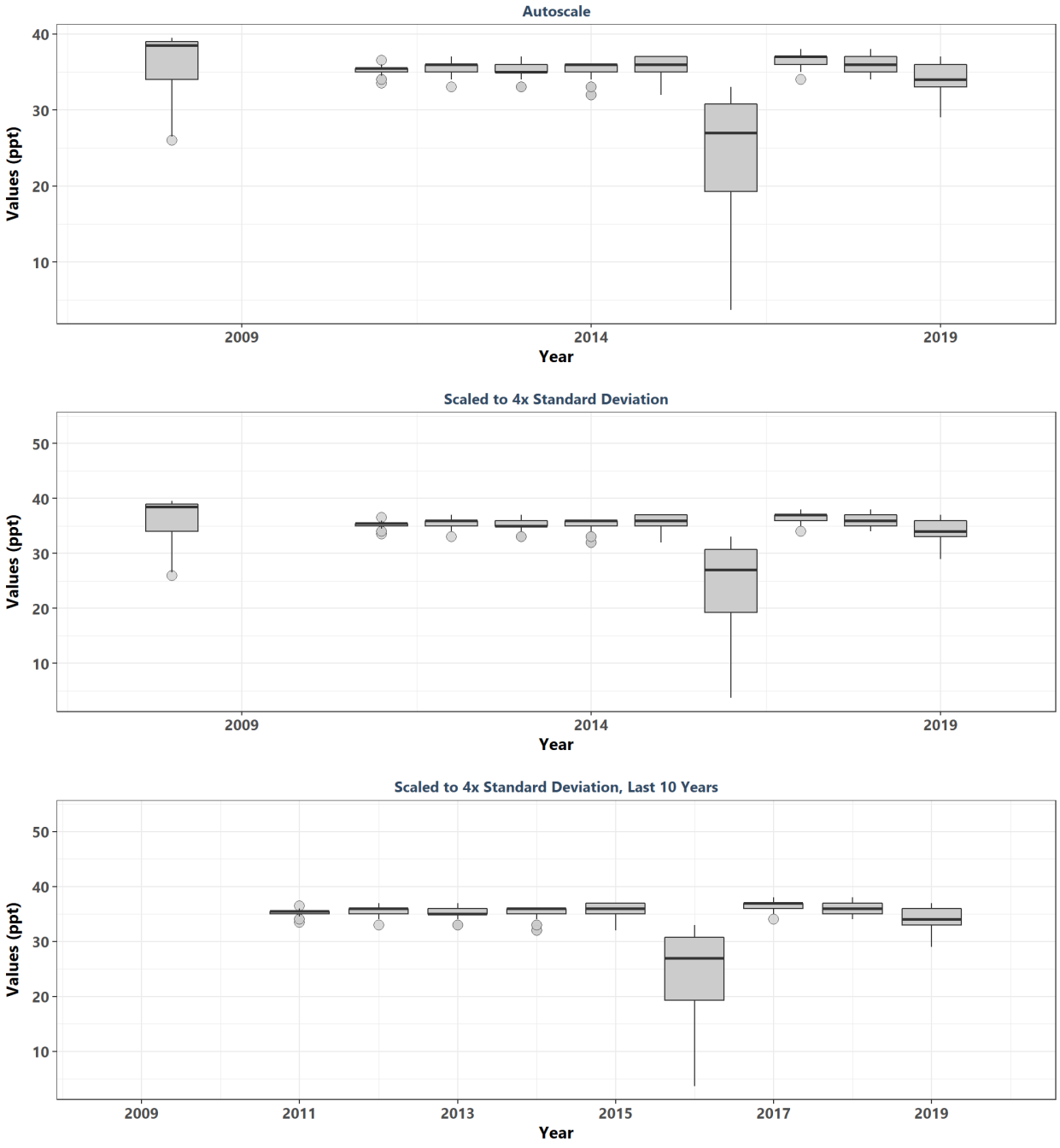
Rookery Bay National Estuarine Research Reserve
7 | National Water Information System
255534081324000
By Month

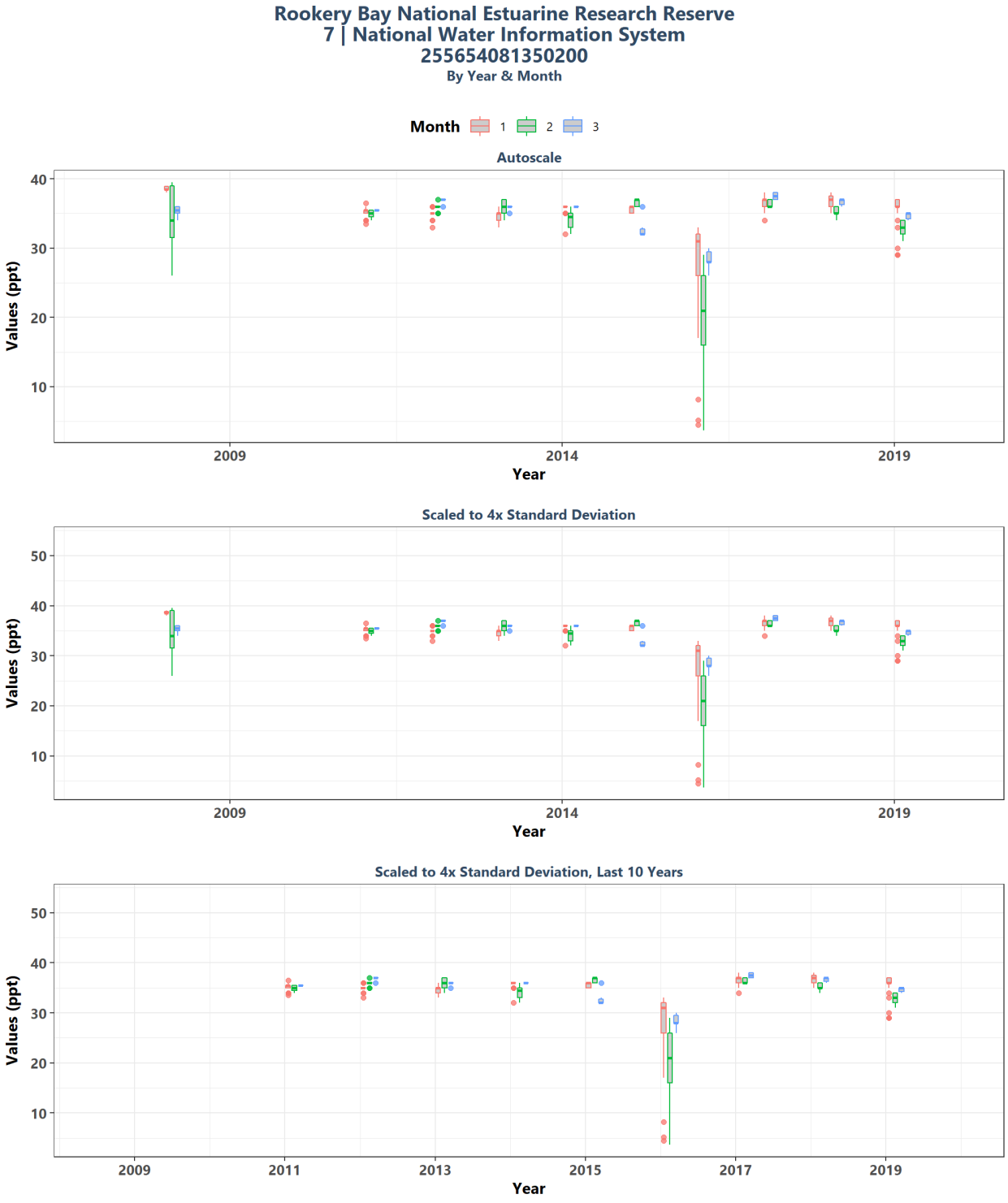




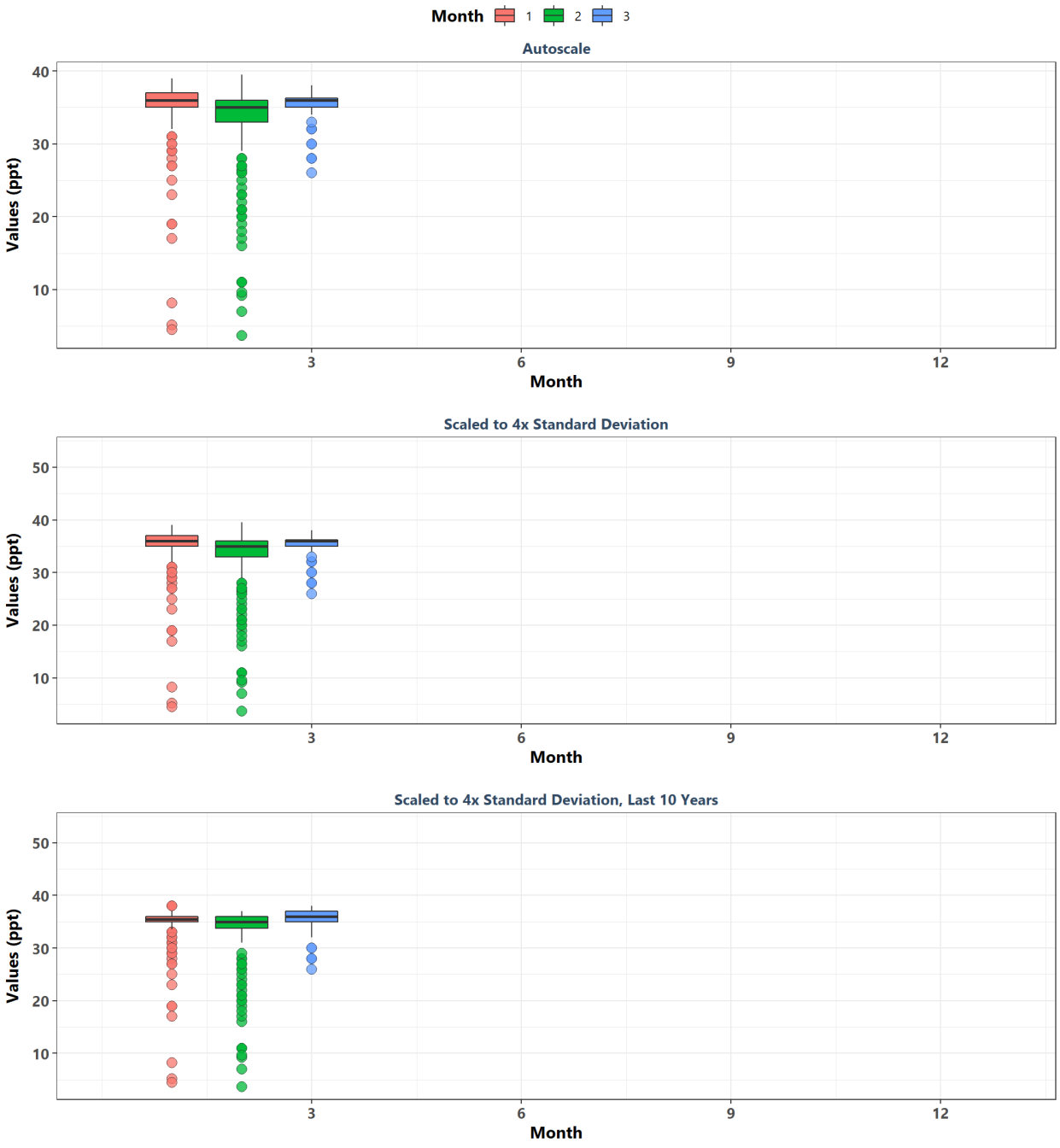


Rookery Bay National Estuarine Research Reserve
7 | National Water Information System
255654081350200
By Year

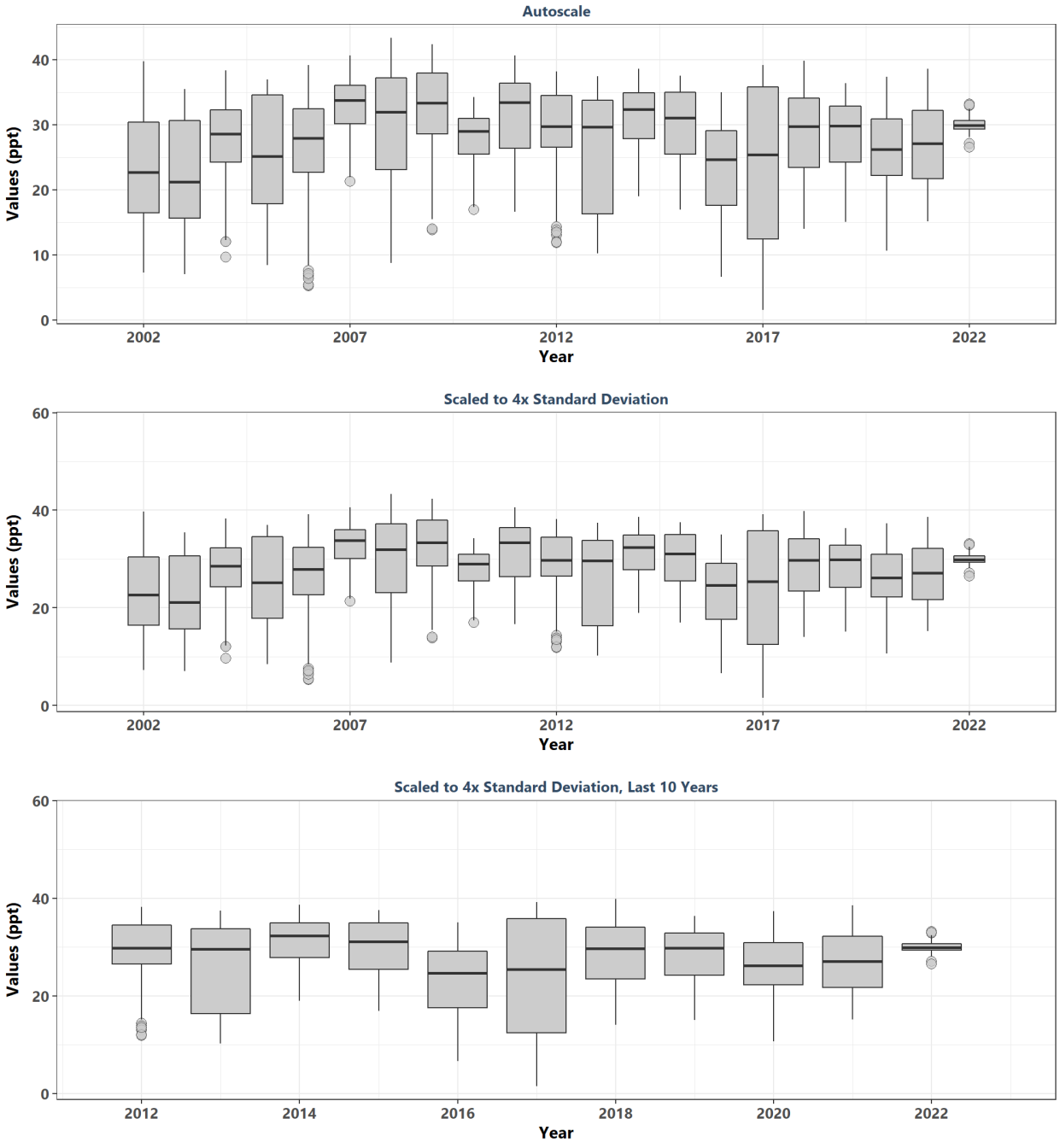



Rookery Bay National Estuarine Research Reserve
7 | National Water Information System
255654081350200
By Year & Month
Month 1 2 3
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year

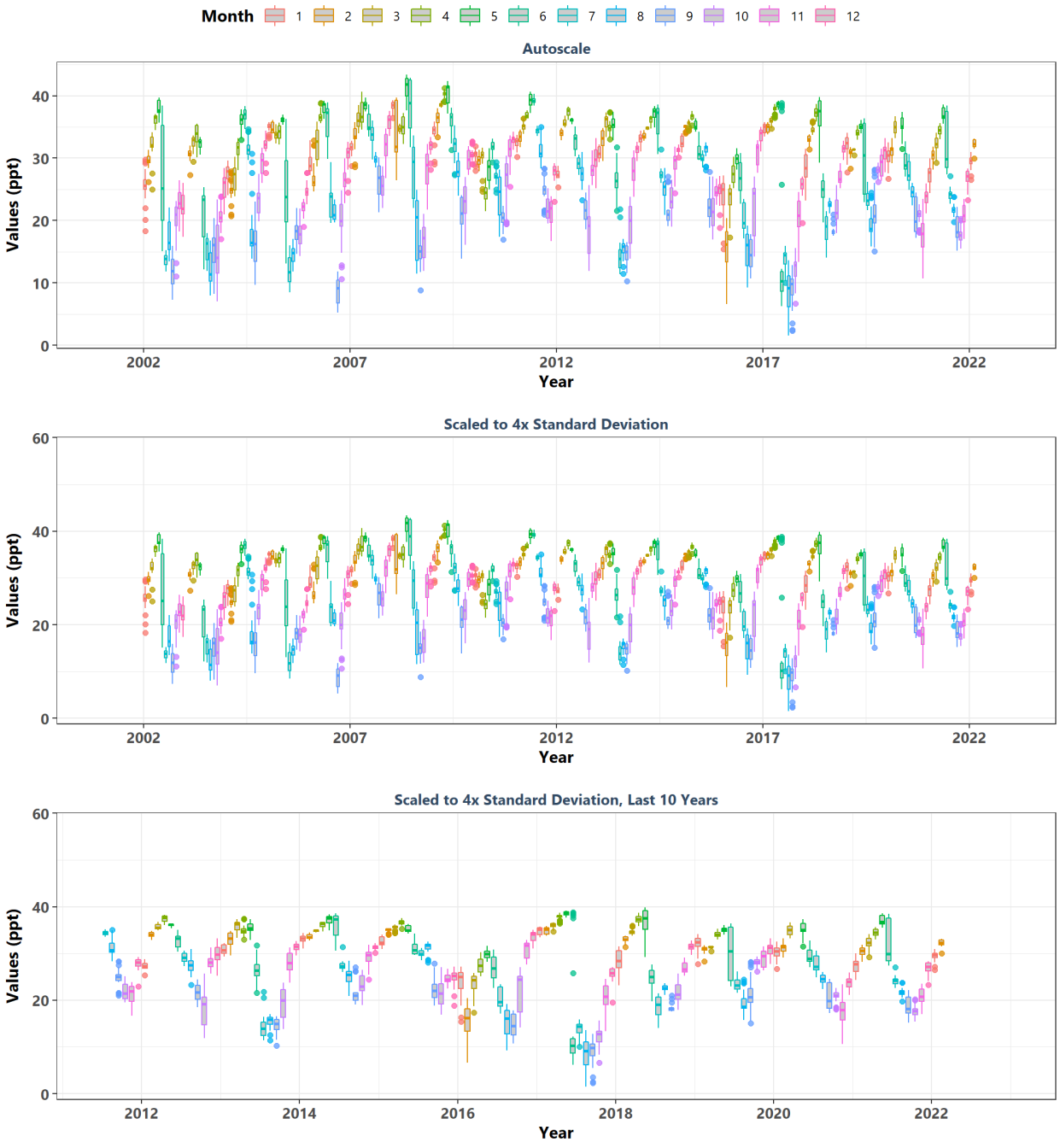
Rookery Bay National Estuarine Research Reserve
7 | National Water Information System
255654081350200
By Month



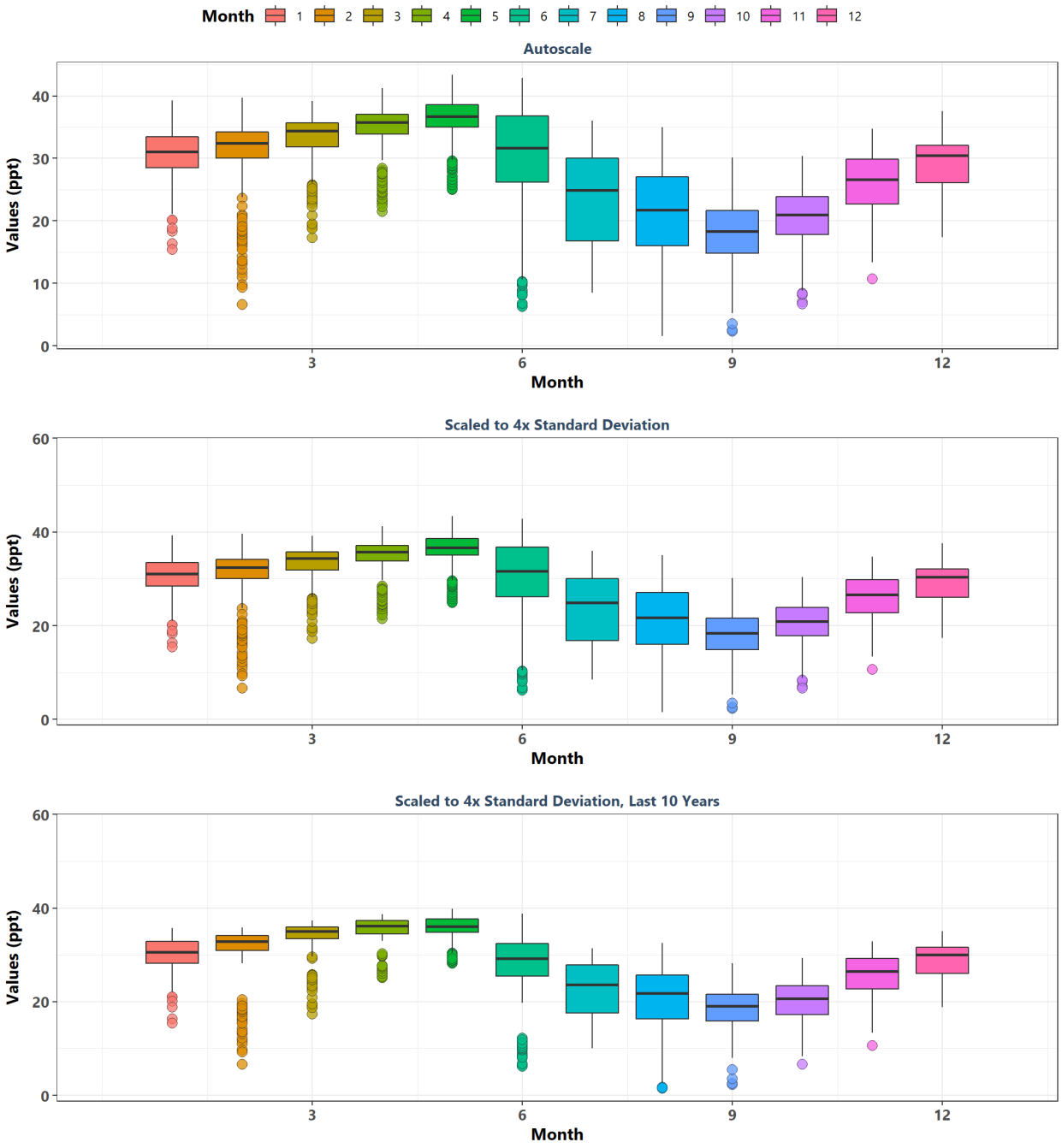
Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfbwq
By Year



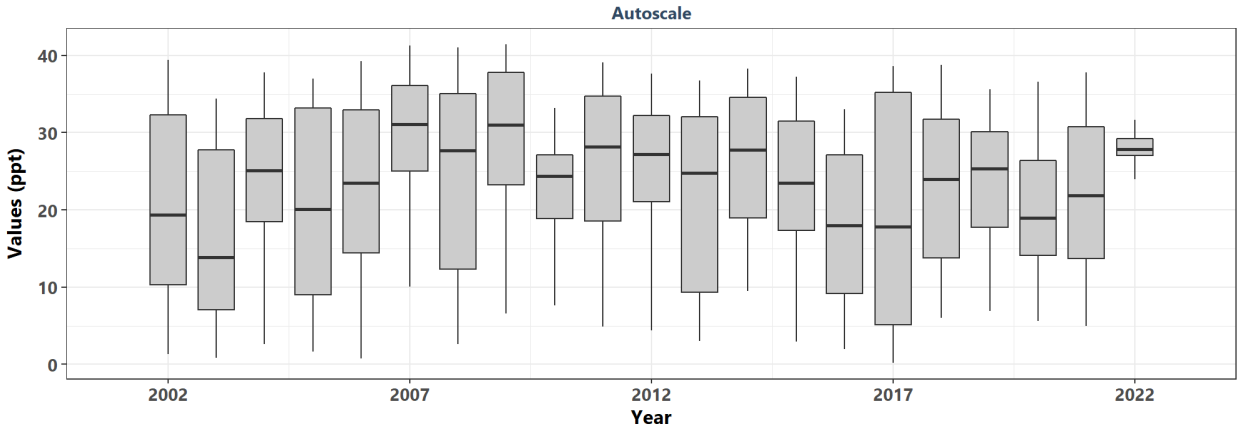
Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfbwq
By Year & Month

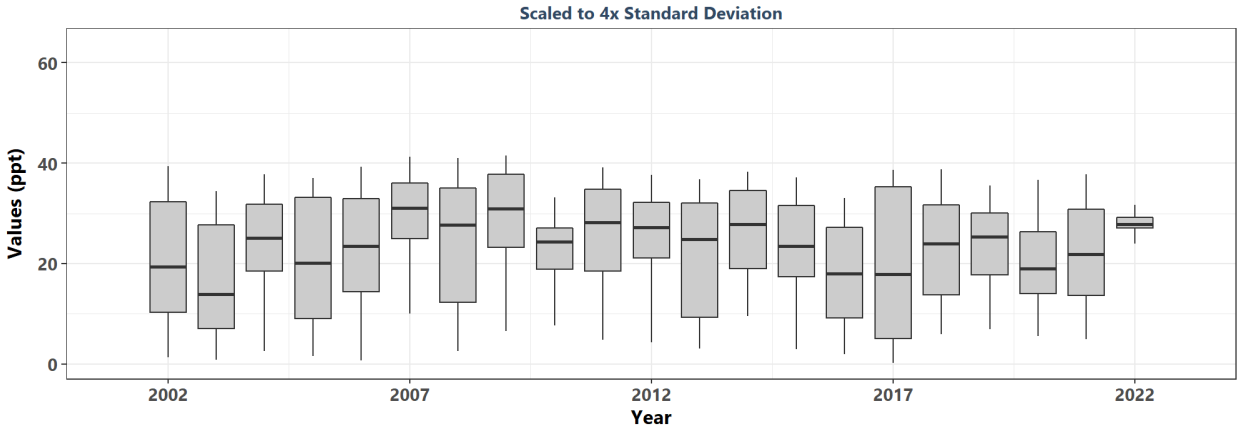


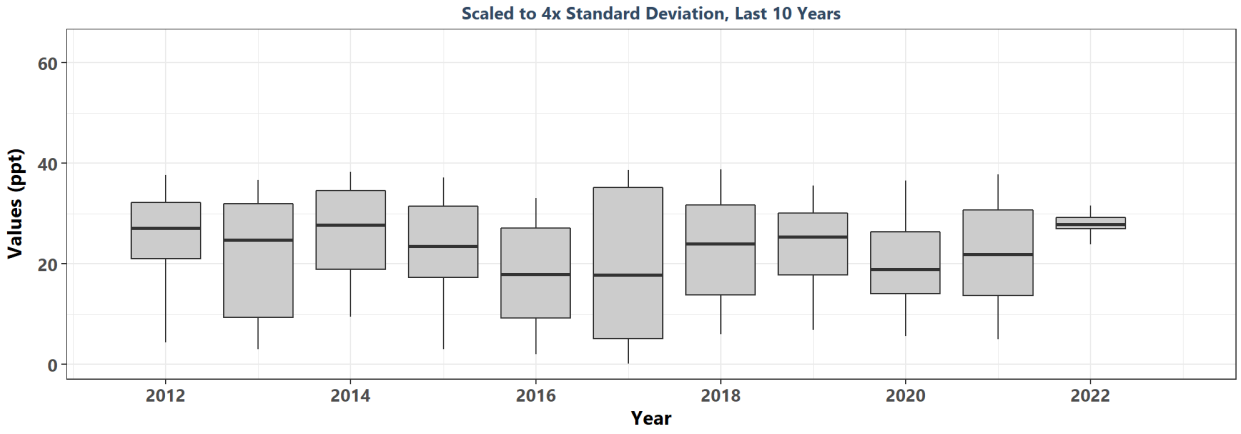
Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfbwq
By Month

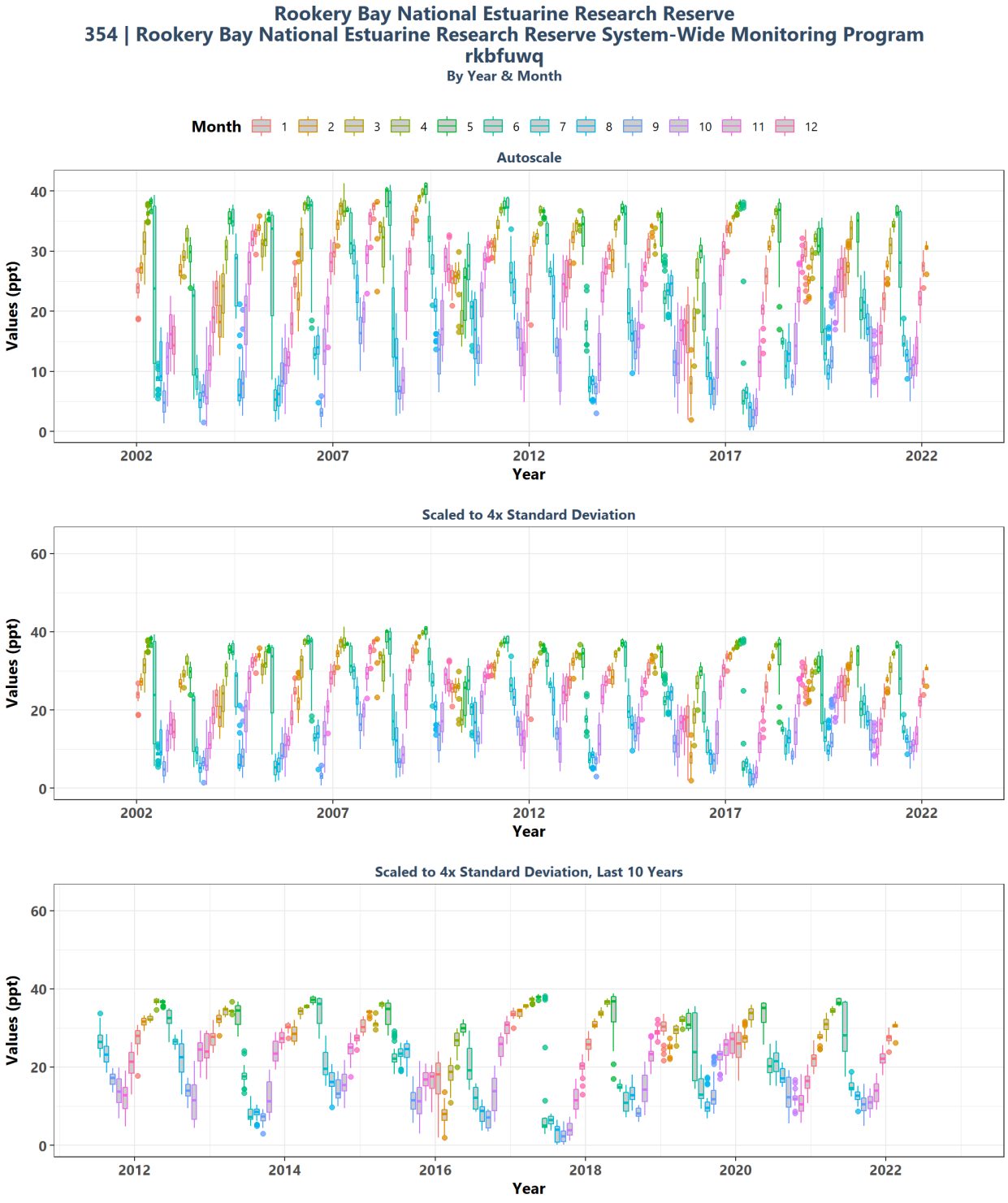


Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfuwq
By Year

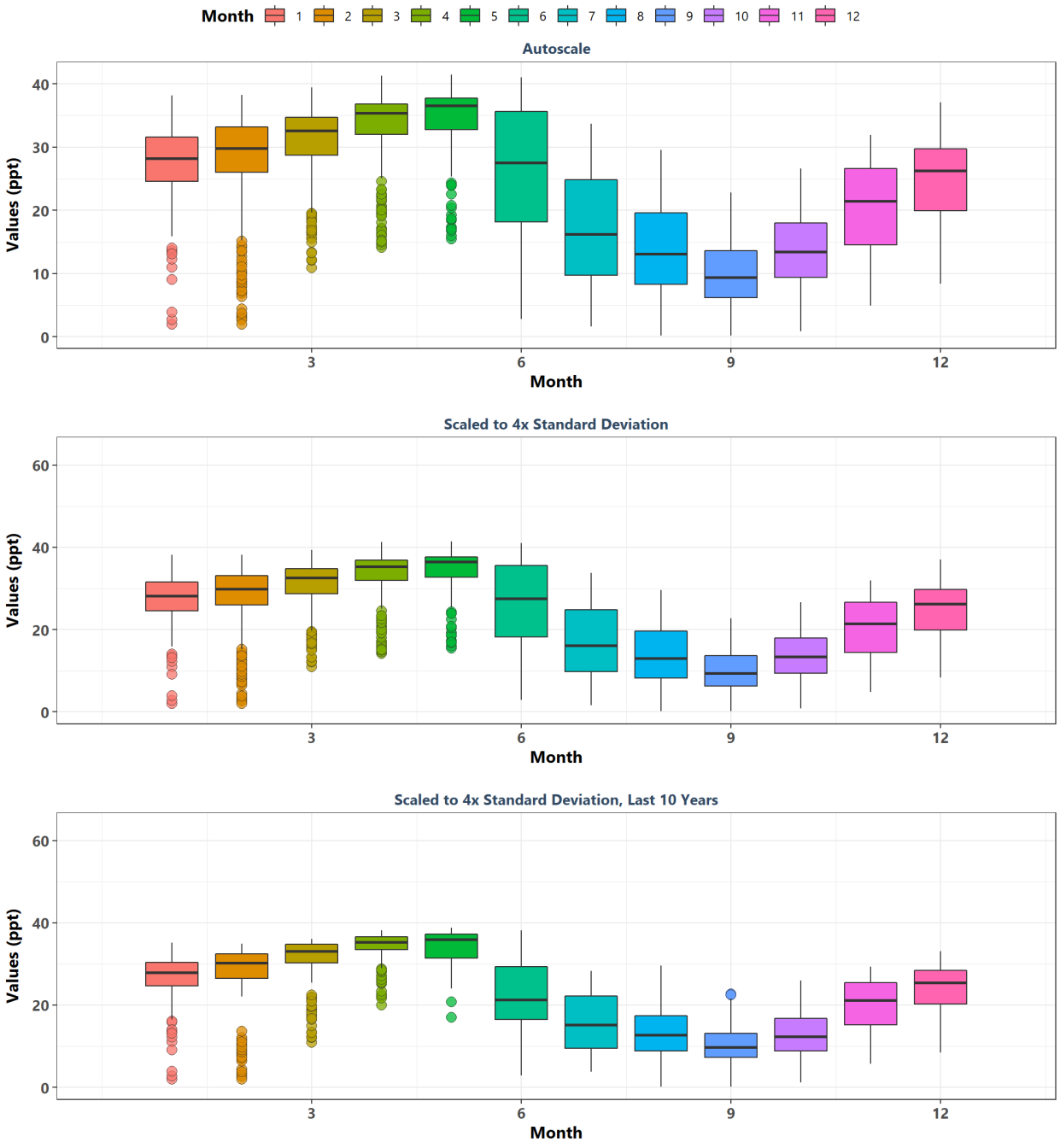




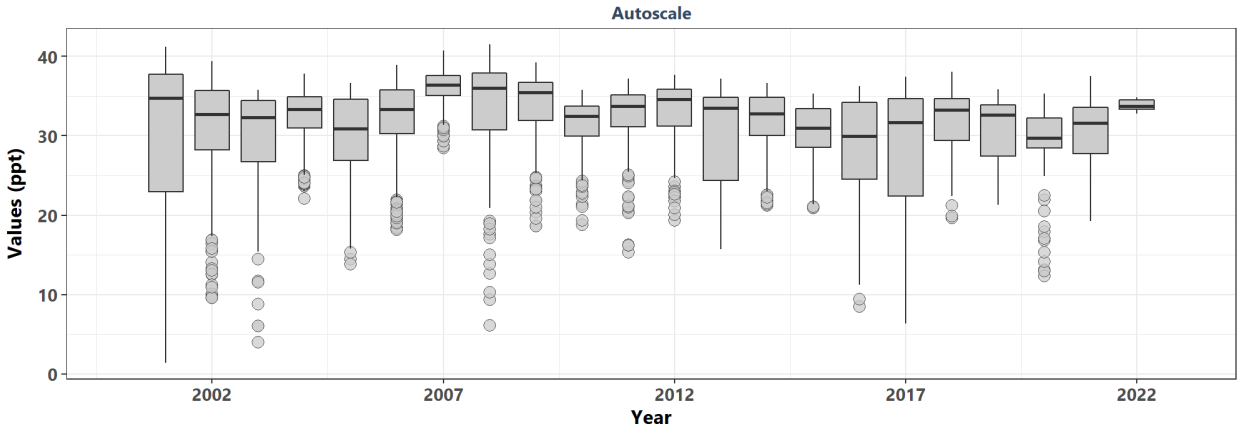


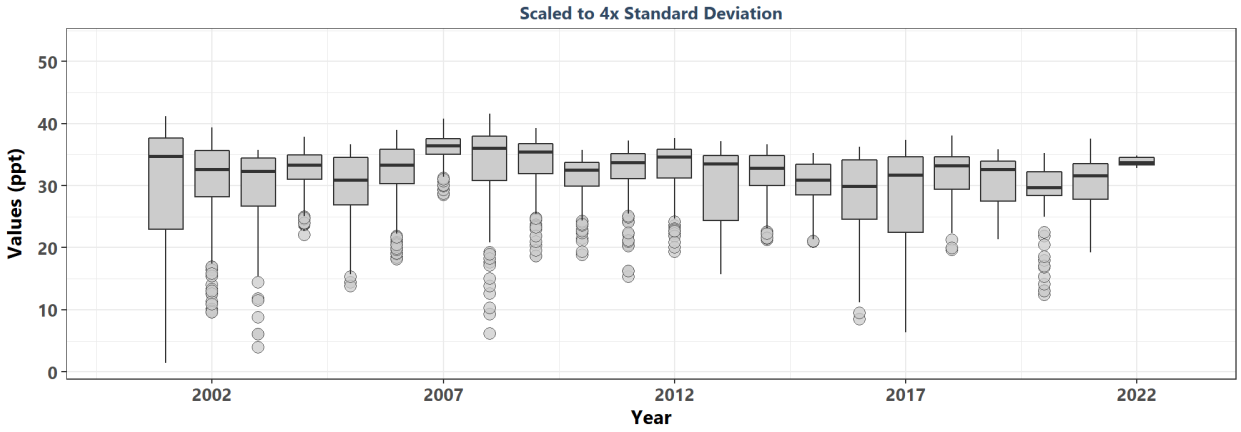

Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfuwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year

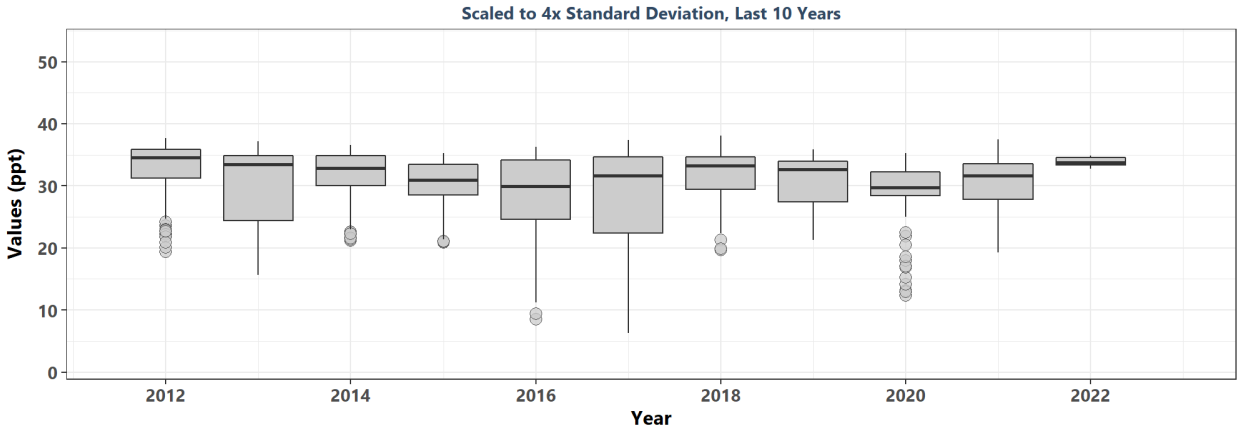
Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfuwq
By Month

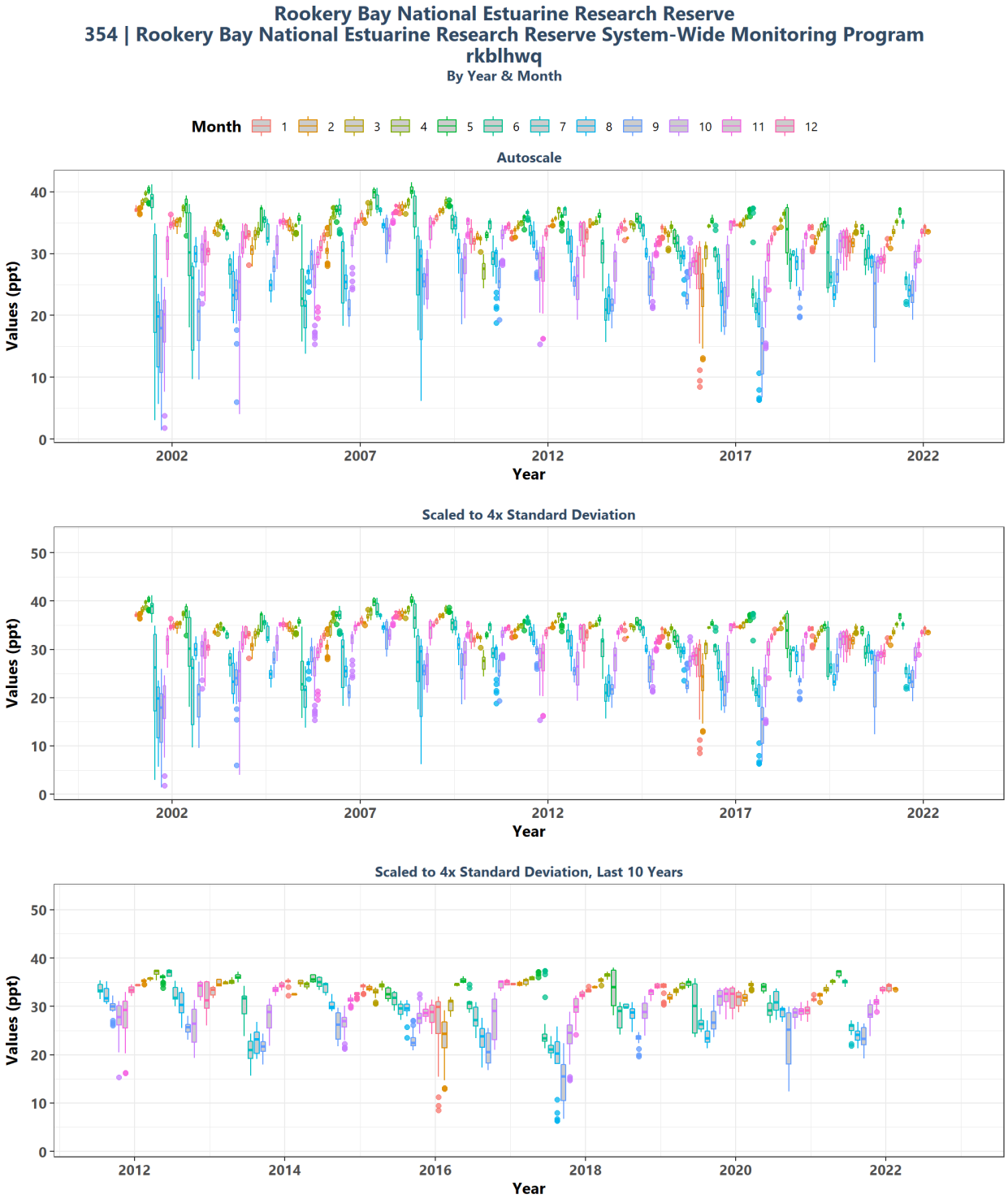


Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkblhwq
By Year

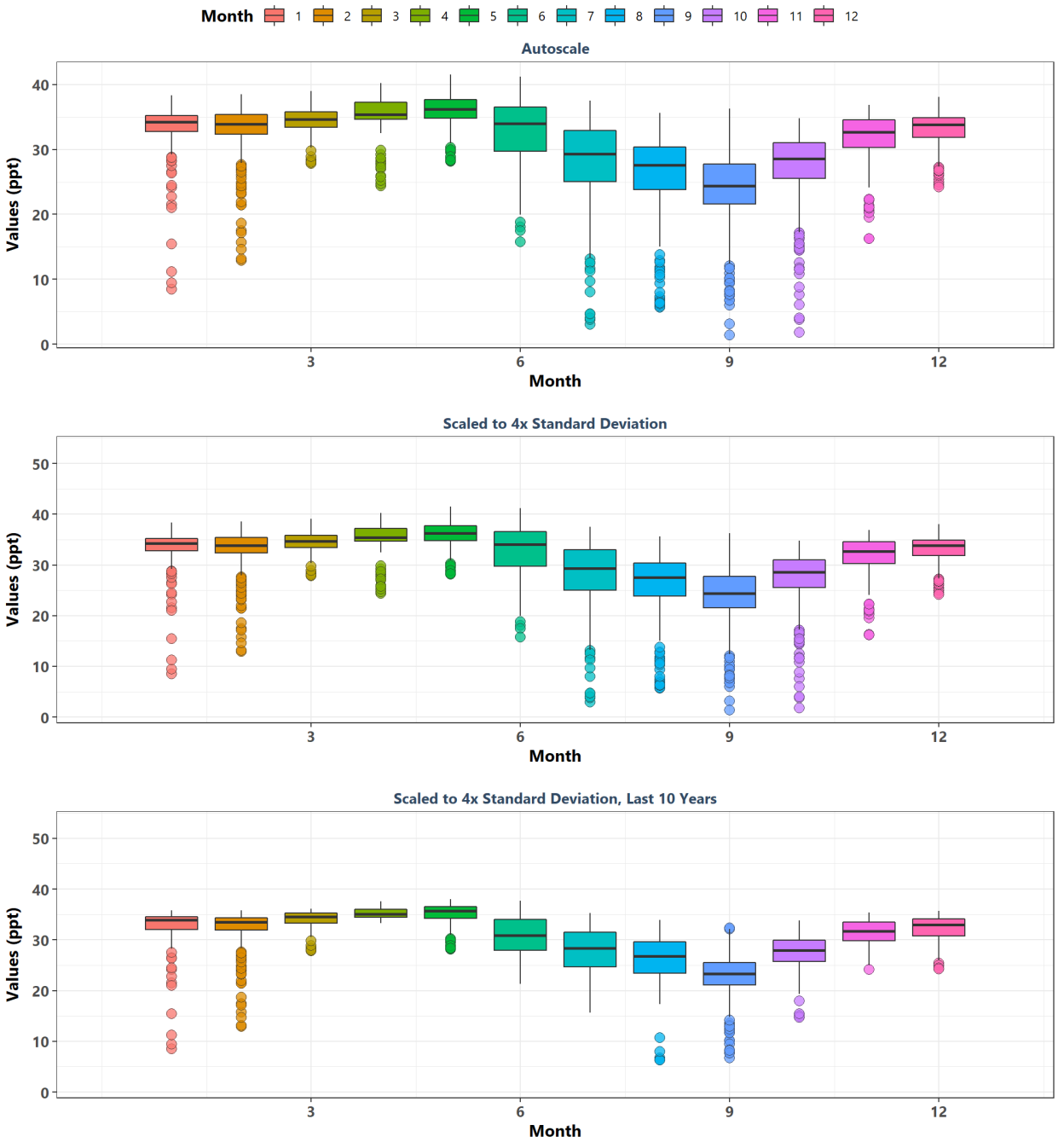




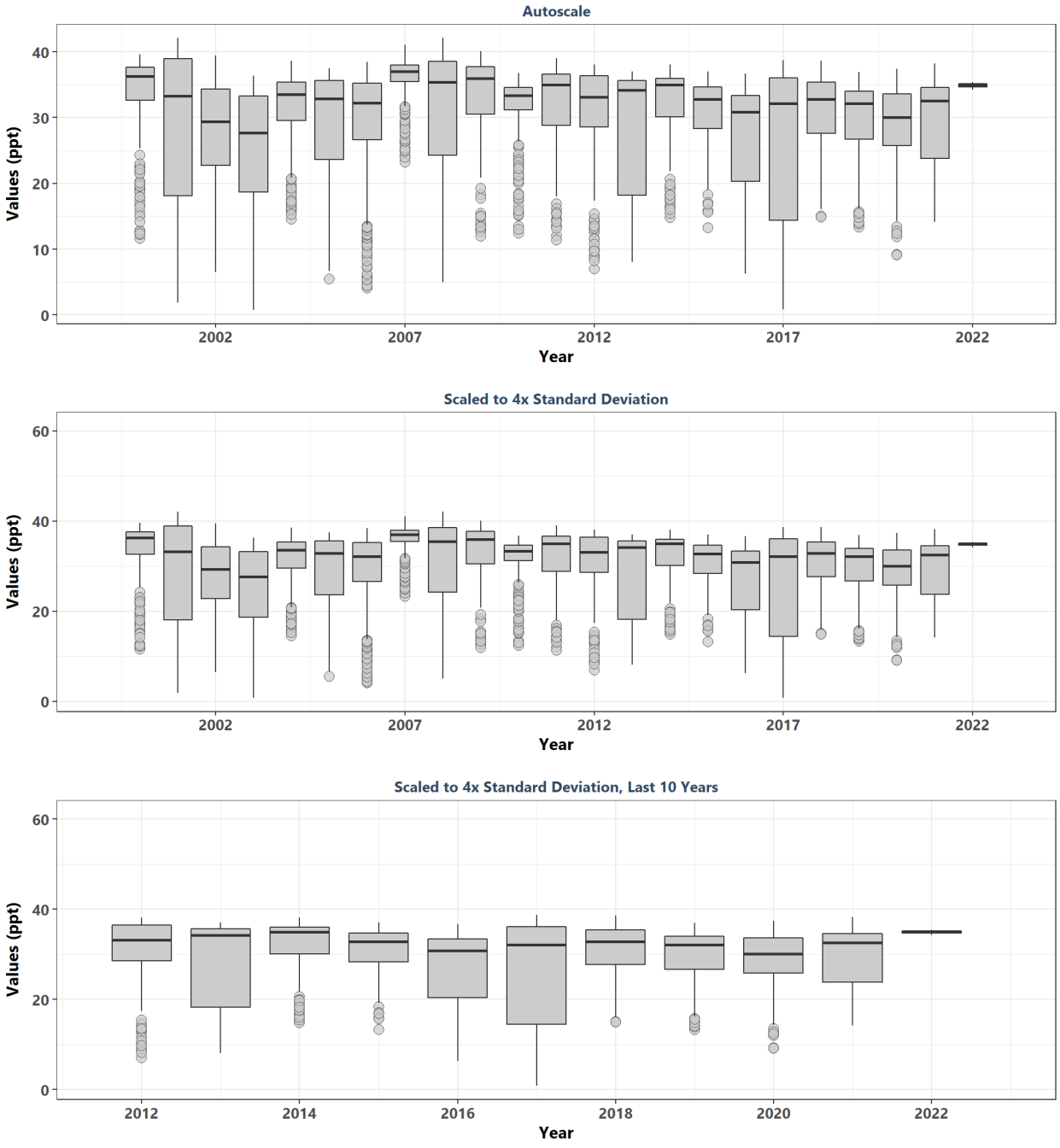


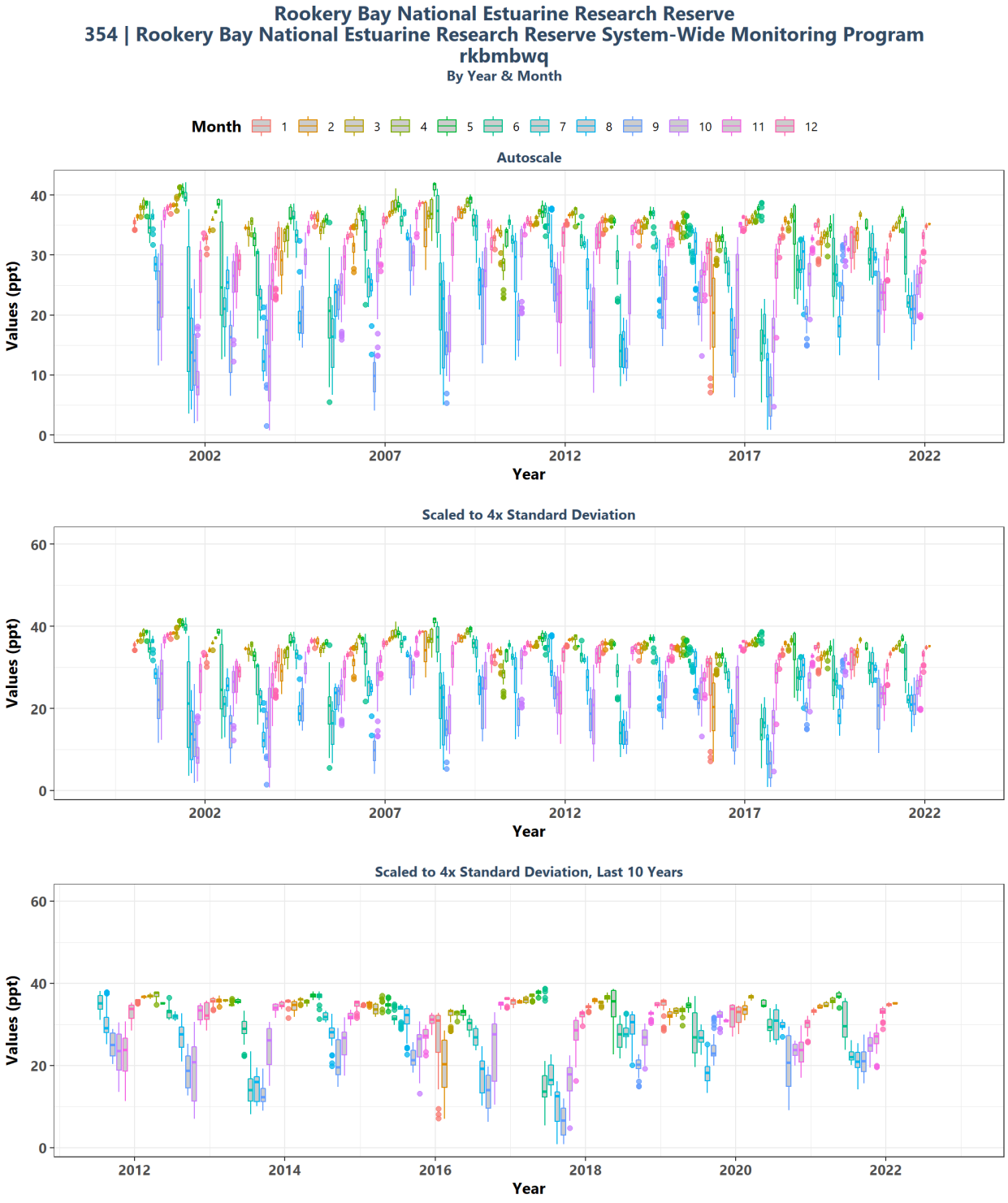

Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkblhwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year

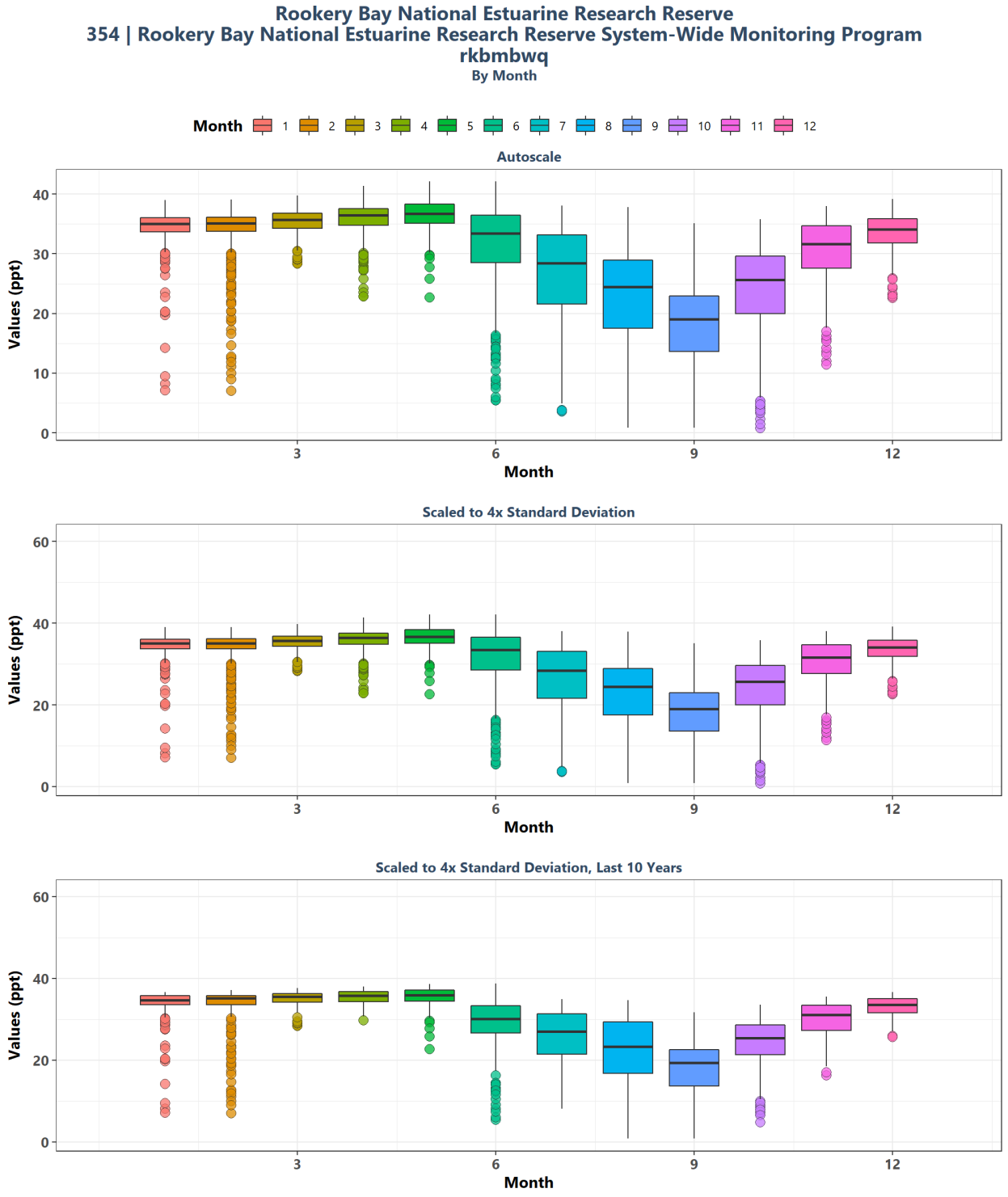
Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkblhwq
By Month



Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbmbwq
By Year




Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbmbwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year




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