SEACAR Continuous Water Quality Analysis: NW 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.

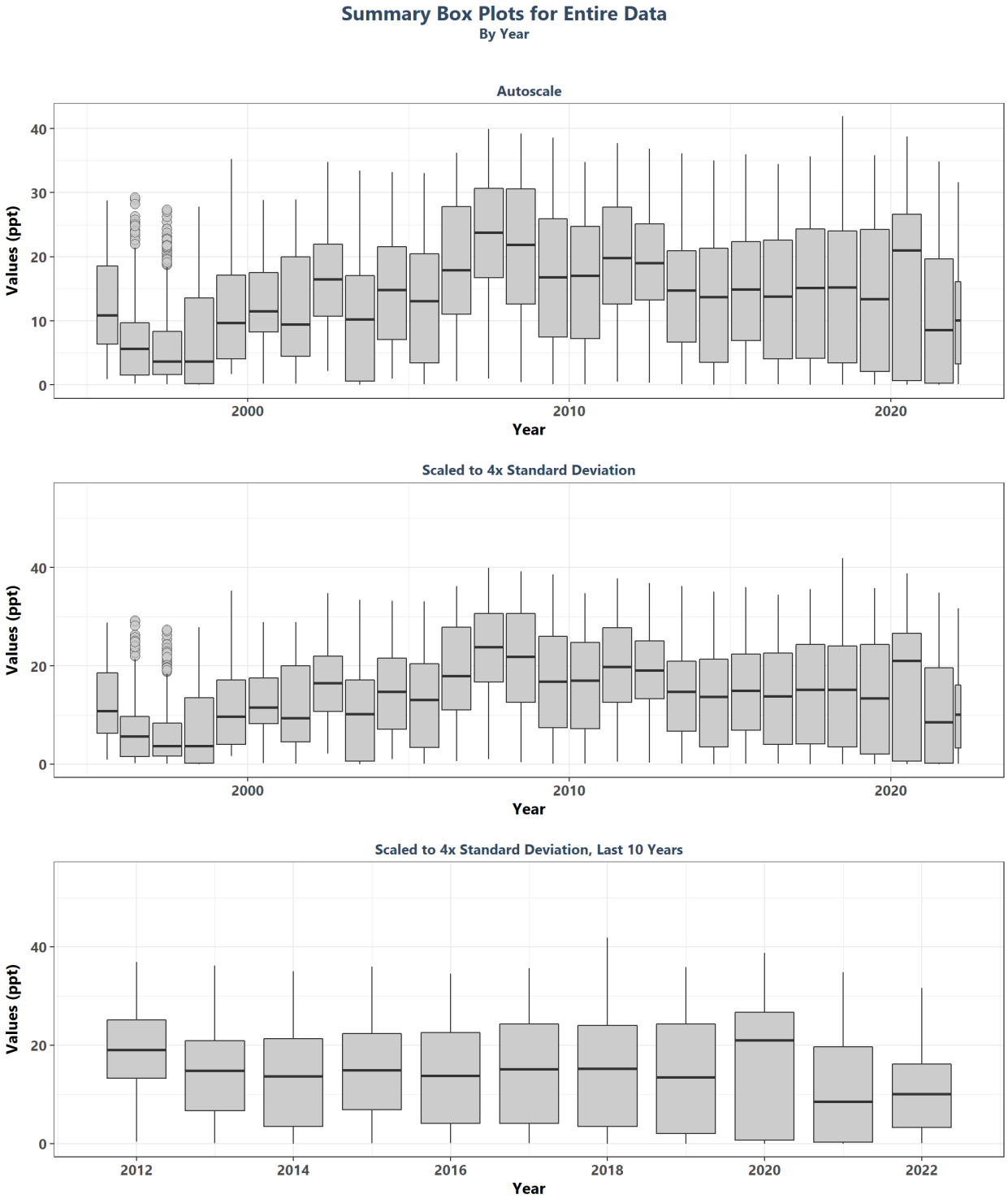
```
p1 <- ggplot(data=data[data$Include==TRUE, ],
             aes(x=YearMonthDec, y=ResultValue,
                 group=YearMonth, color=as.factor(Month))) +
   geom_boxplot(fill="#cccccc", outlier.size=1.5, outlier.alpha=0.75) +
   labs(subtitle="Autoscale", x="Year",
        y=paste0("Values (", unit,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

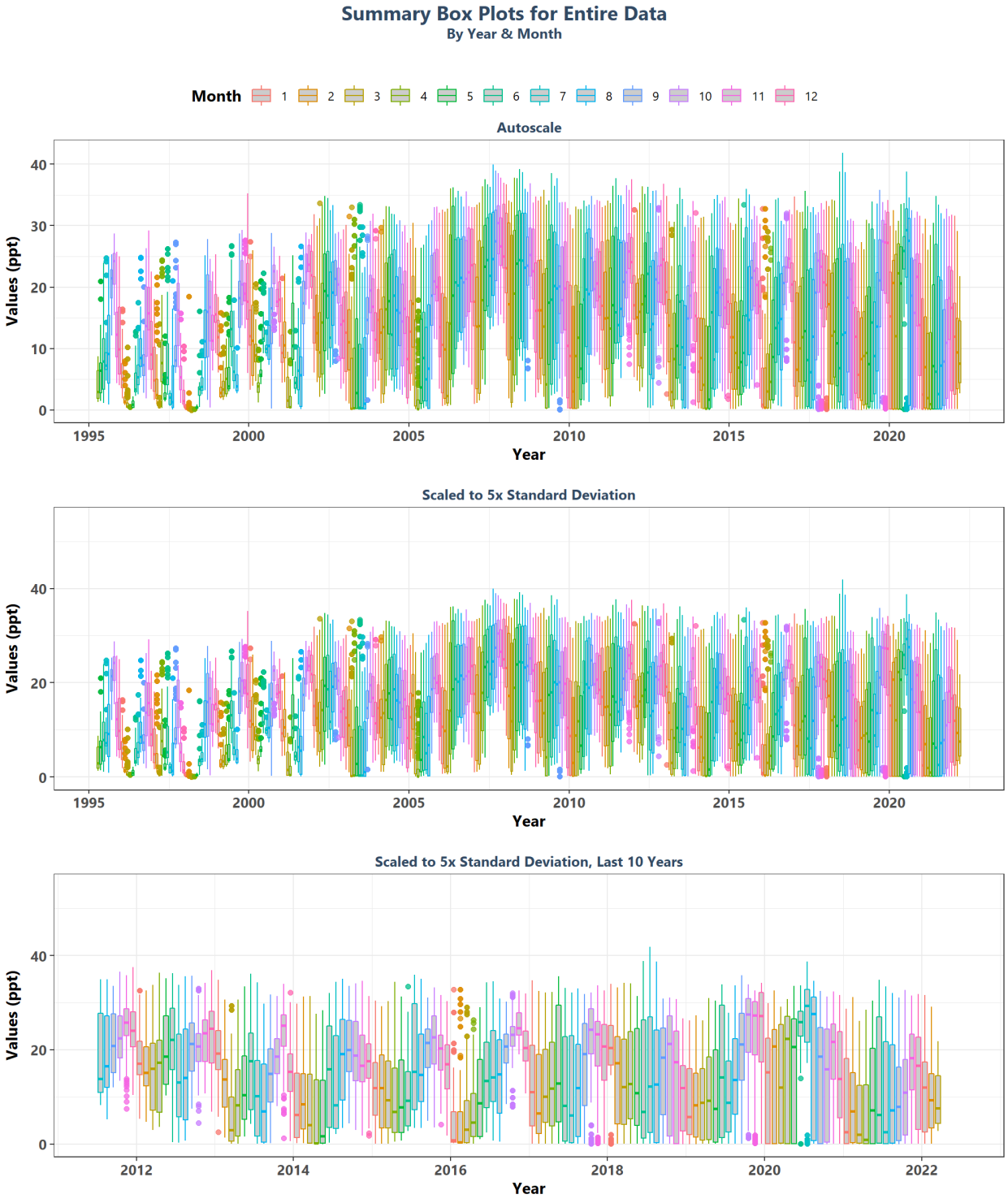
```
p3 <- ggplot(data=data[data$Include==TRUE, ],
             aes(x=YearMonthDec, y=ResultValue,
                 group=YearMonth, color=as.factor(Month))) +
   geom_boxplot(fill="#cccccc", outlier.size=1.5, outlier.alpha=0.75) +
   labs(subtitle="Scaled to 5x Standard Deviation, Last 10 Years",
  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

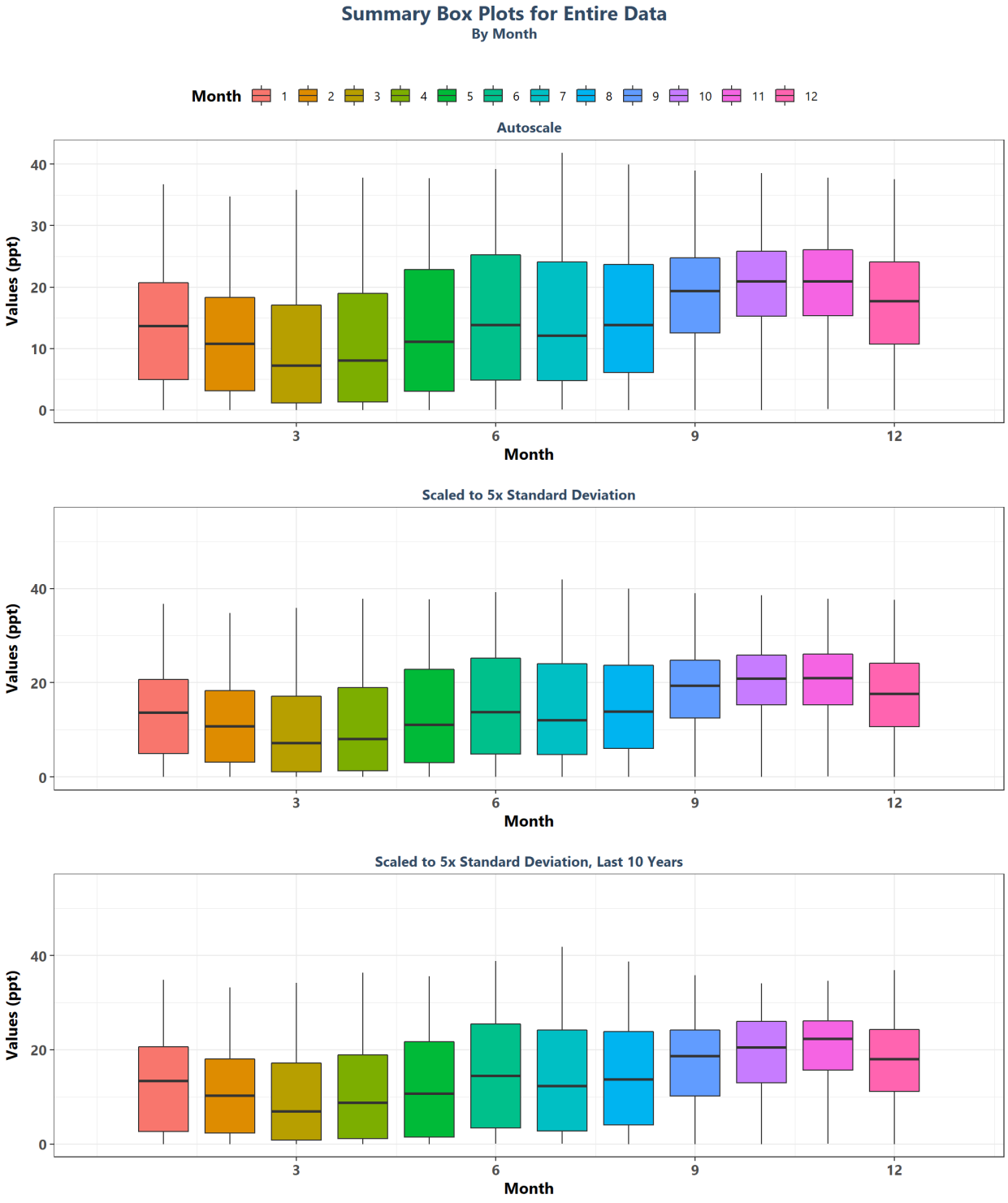
The following box plots are grouped by month with fill color being related to the month. This is designed to view potential seasonal trends.

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

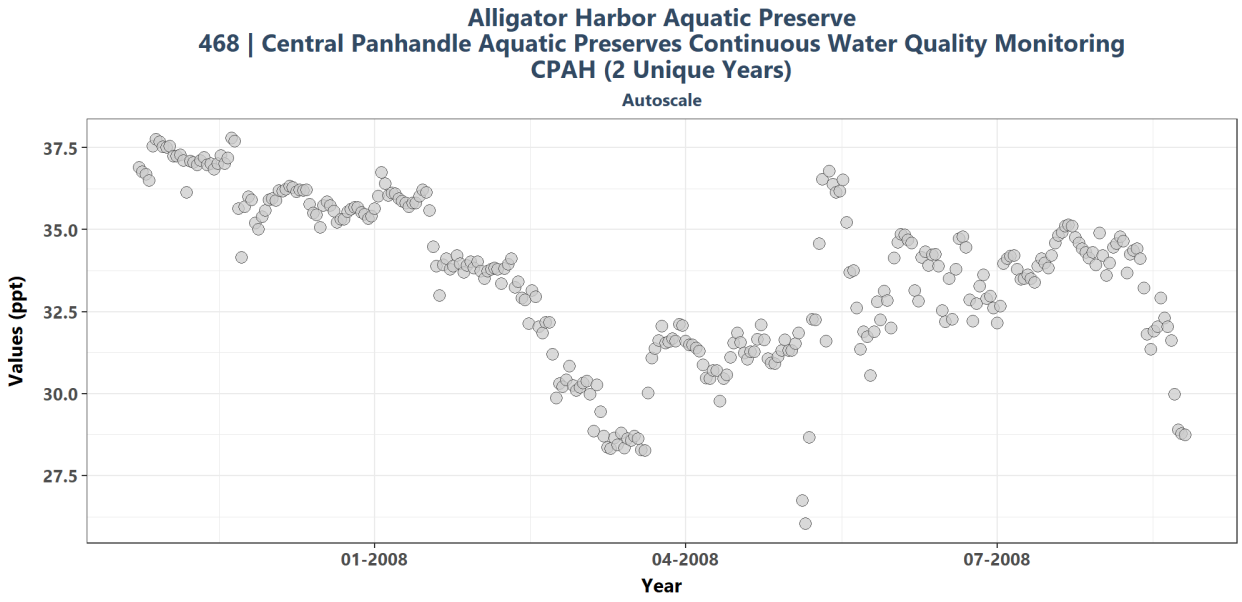

Summary Box Plots for Entire Data
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 5x Standard Deviation
Values (ppt)
Year
Scaled to 5x Standard Deviation, Last 10 Years
Values (ppt)
Year

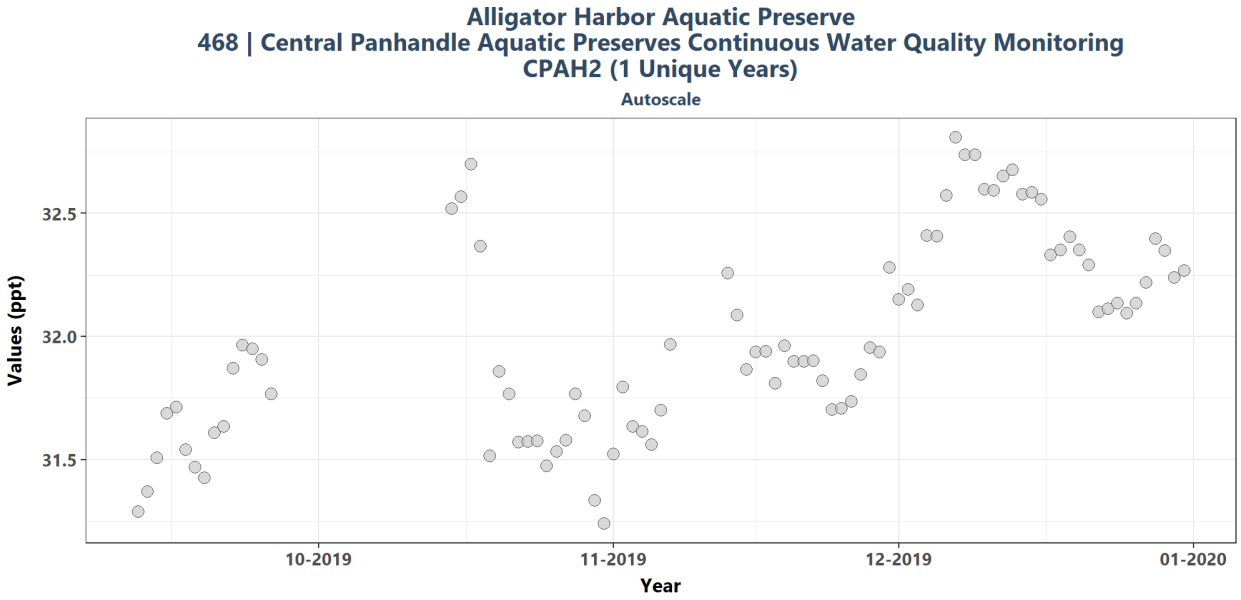


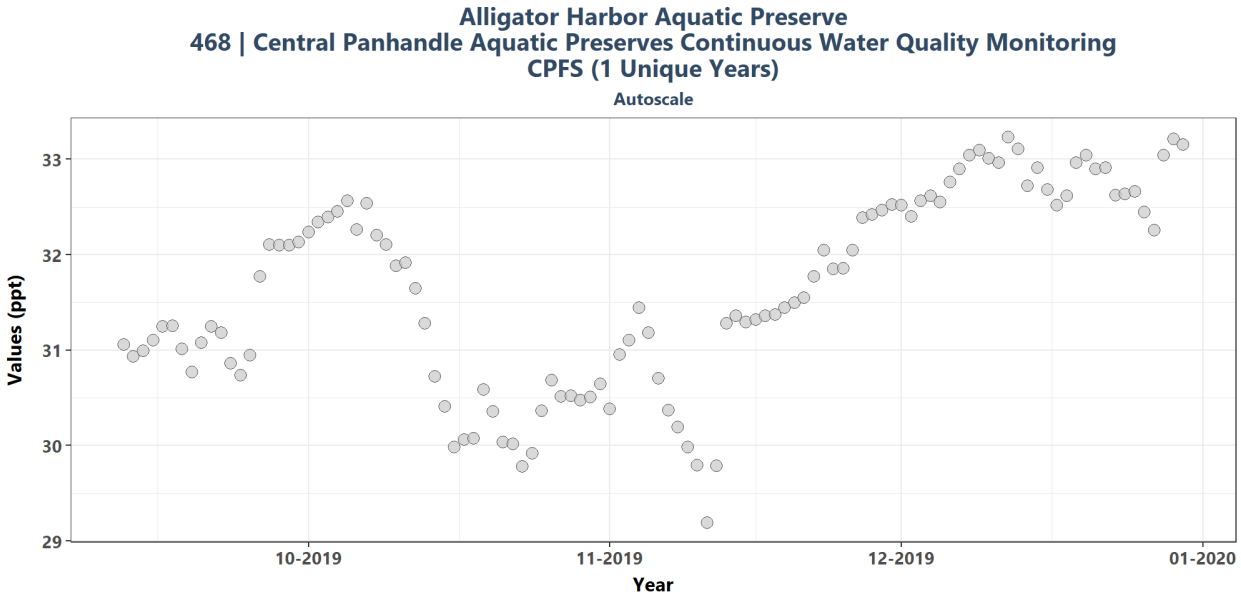
Appendix II: Excluded Monitoring Locations

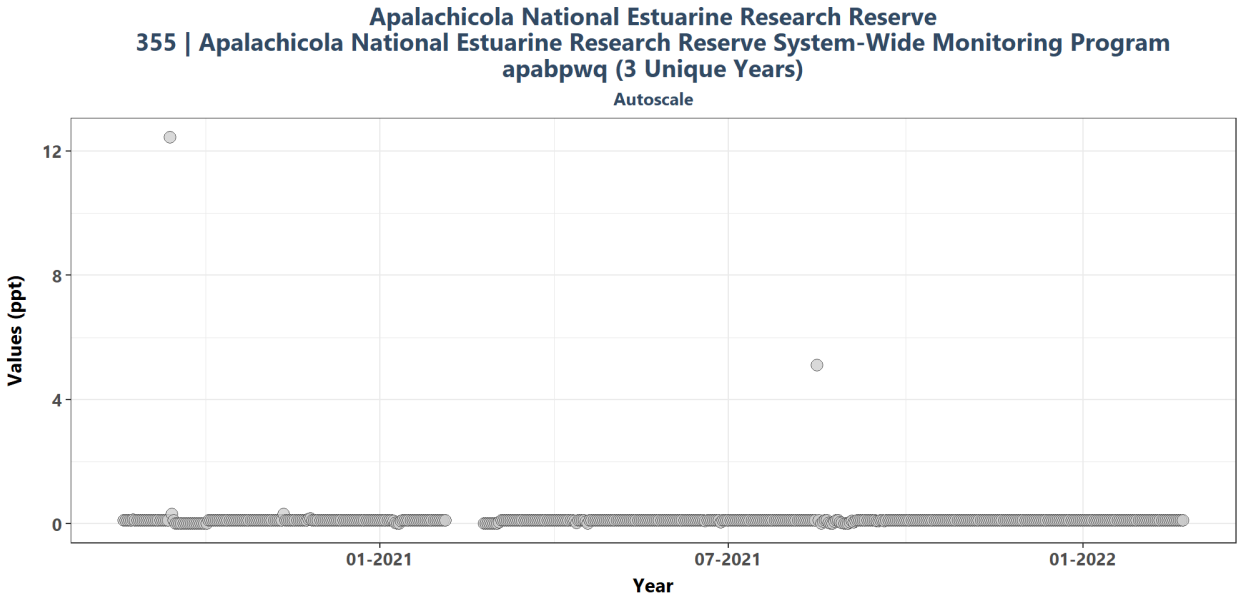
Scatter plots of data values are created for monitoring locations that have fewer than 5 separate years of data entries.

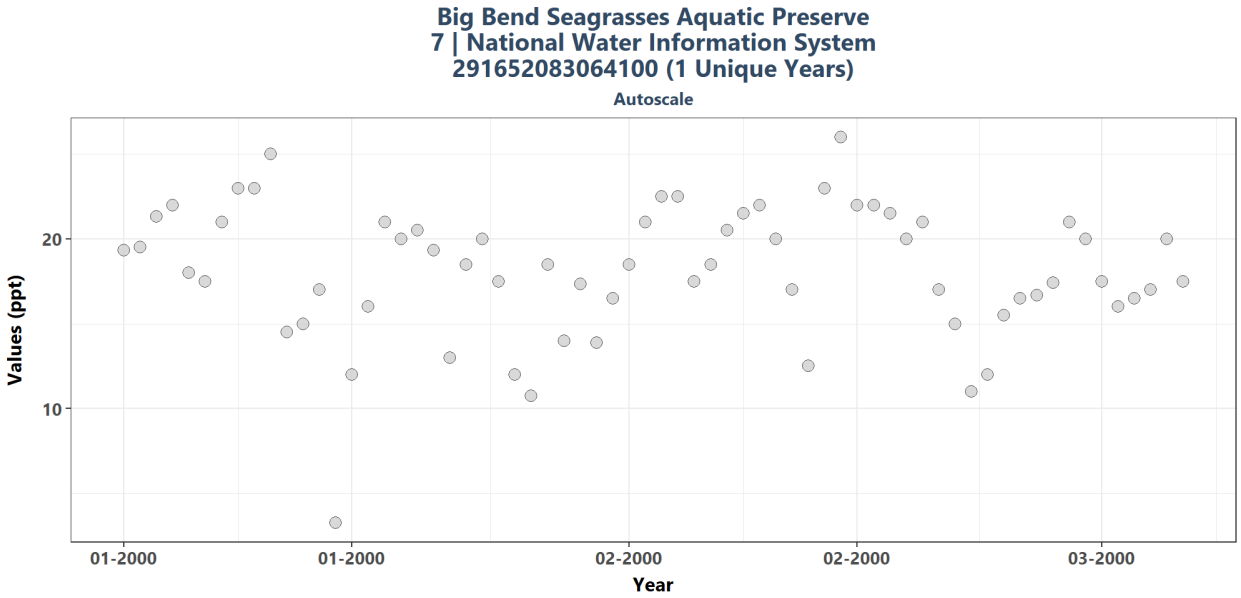
```
Mon_Exclude <- Mon_Summ[Mon_Summ$N_Years<5 & Mon_Summ$N_Years>0,]
Mon_Exclude <- Mon_Exclude[order(Mon_Exclude$MonitoringID),]
z=nrow(Mon_Exclude)

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)
   }
}
```

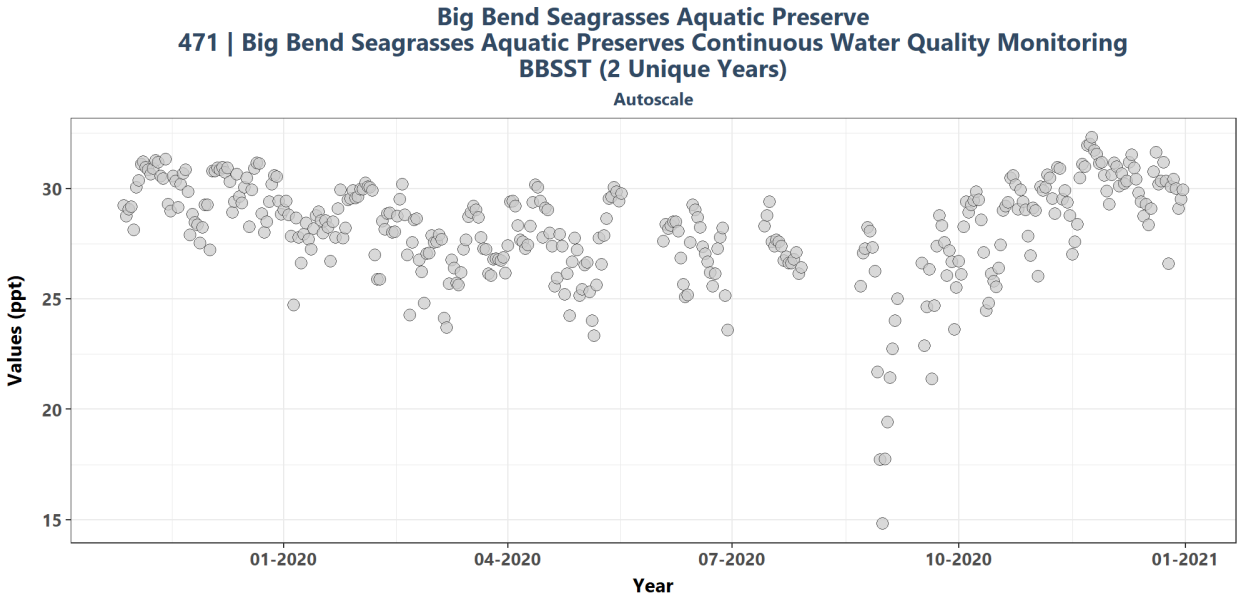


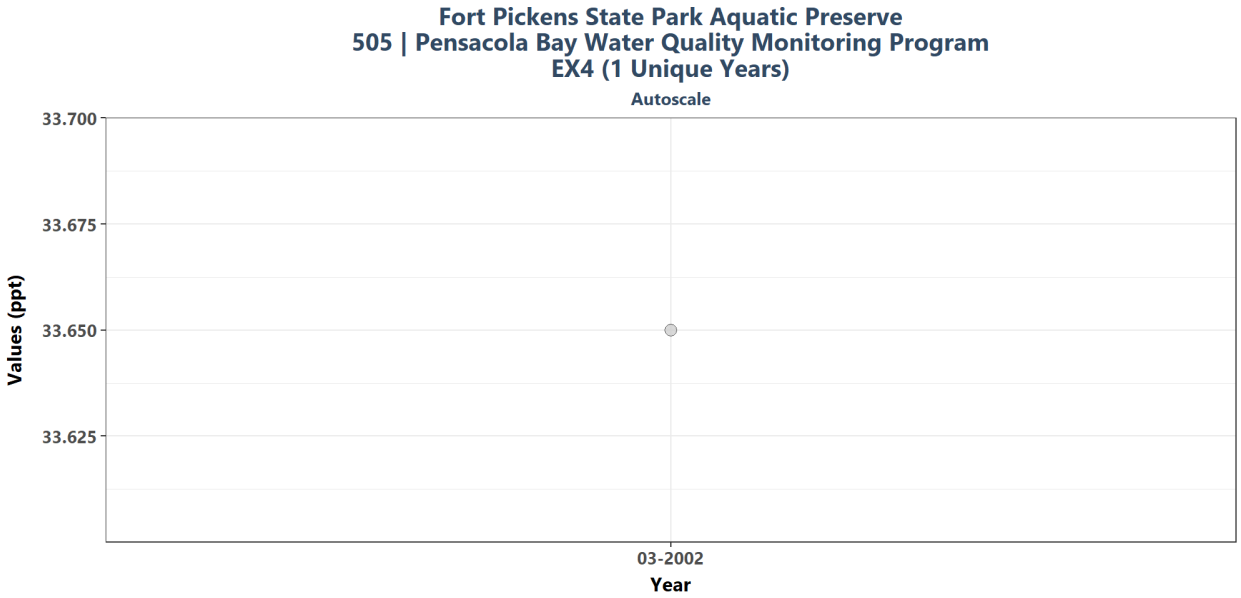

Alligator Harbor Aquatic Preserve
468 | Central Panhandle Aquatic Preserves Continuous Water Quality Monitoring
CPAH2 (1 Unique Years)
Autoscale
Values (ppt)
32.5
32.0
31.5
10-2019
11-2019
12-2019
01-2020
Year

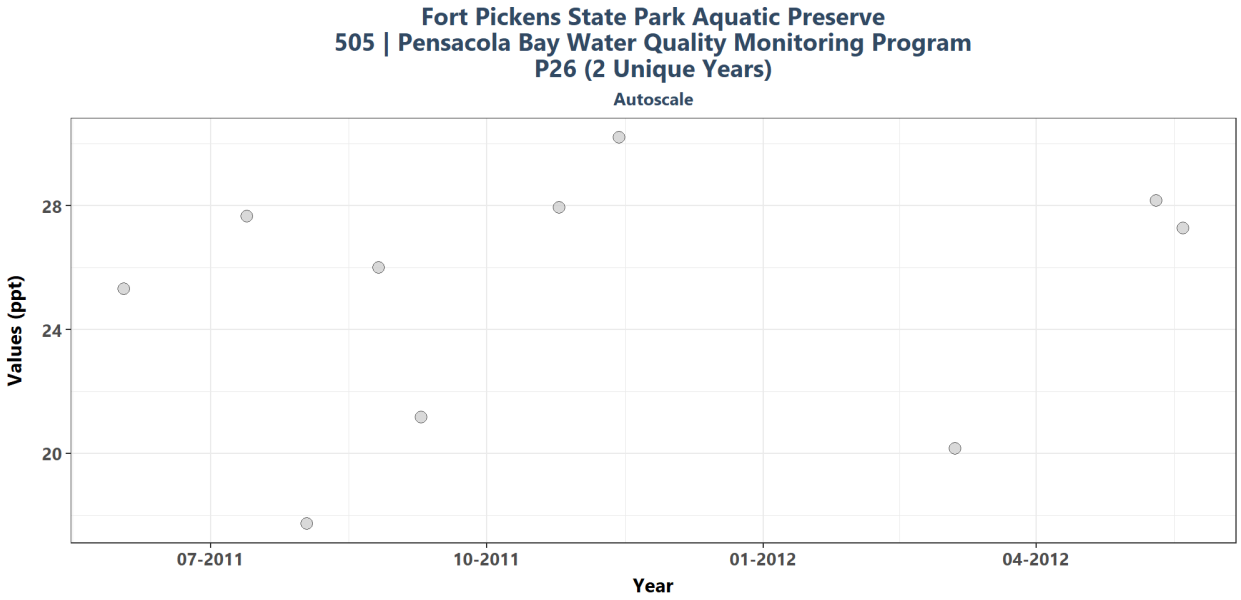

Alligator Harbor Aquatic Preserve
468 | Central Panhandle Aquatic Preserves Continuous Water Quality Monitoring
CPFS (1 Unique Years)
Autoscale
Values (ppt)
33
32
31
30
29
10-2019
11-2019
12-2019
01-2020
Year

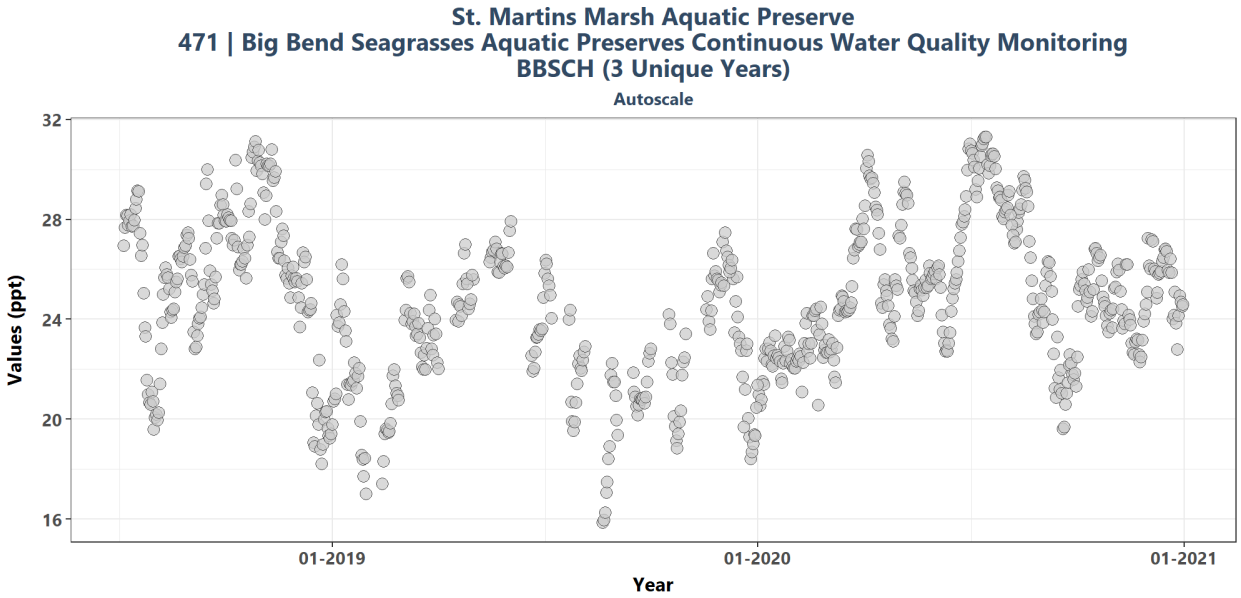

Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apabpwq (3 Unique Years)
Autoscale
Values (ppt)
12
8
4
0
01-2021
07-2021
01-2022
Year


Big Bend Seagrasses Aquatic Preserve
7 | National Water Information System
291652083064100 (1 Unique Years)
Autoscale
Values (ppt)
20
10
01-2000
01-2000
02-2000
02-2000
03-2000
Year


Big Bend Seagrasses Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring
BBSST (2 Unique Years)
Autoscale
Values (ppt)
30
25
20
15
01-2020
04-2020
07-2020
10-2020
01-2021
Year


Fort Pickens State Park Aquatic Preserve
505 | Pensacola Bay Water Quality Monitoring Program
EX4 (1 Unique Years)
Autoscale
Values (ppt)
33.700
33.675
33.650
33.625
03-2002
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

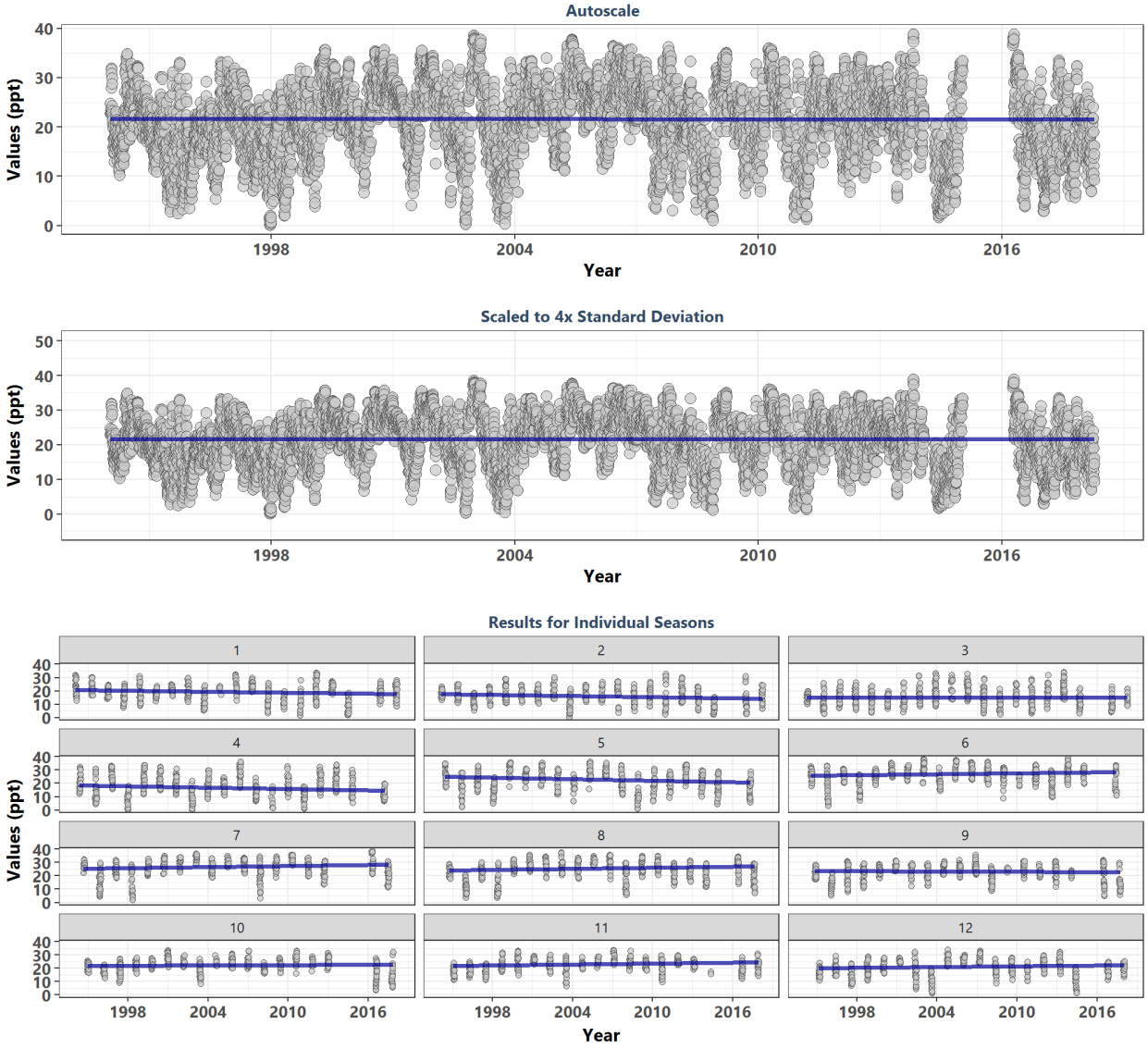
5. Set the plot type as a point plot with the size of the points
6. Add the linear trend
7. Create the title, x-axis, y-axis, and color fill labels
8. Set the y and x limits
9. Make the axis labels bold
10. Plot the arrangement as a set of panels

```
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[Moni
KT.Stats[, season := NULL]


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


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


p2 <- ggplot(data=plot_data,
             aes(x=relyear_dd, y=ResultValue)) +
   geom_point(shape=21, size=3, color="#333333", fill="#cccccc",
              alpha=0.75) +
   # geom_abline(aes(slope=s_slope, intercept=s_int),
   #             color="#000099", size=1.2, alpha=0.7) +
   geom_segment(data = KT.Stats[MonitoringID == Mon_IDs[i] & Season == "All", ], aes(x = relyear_
                                                                                     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) +
```

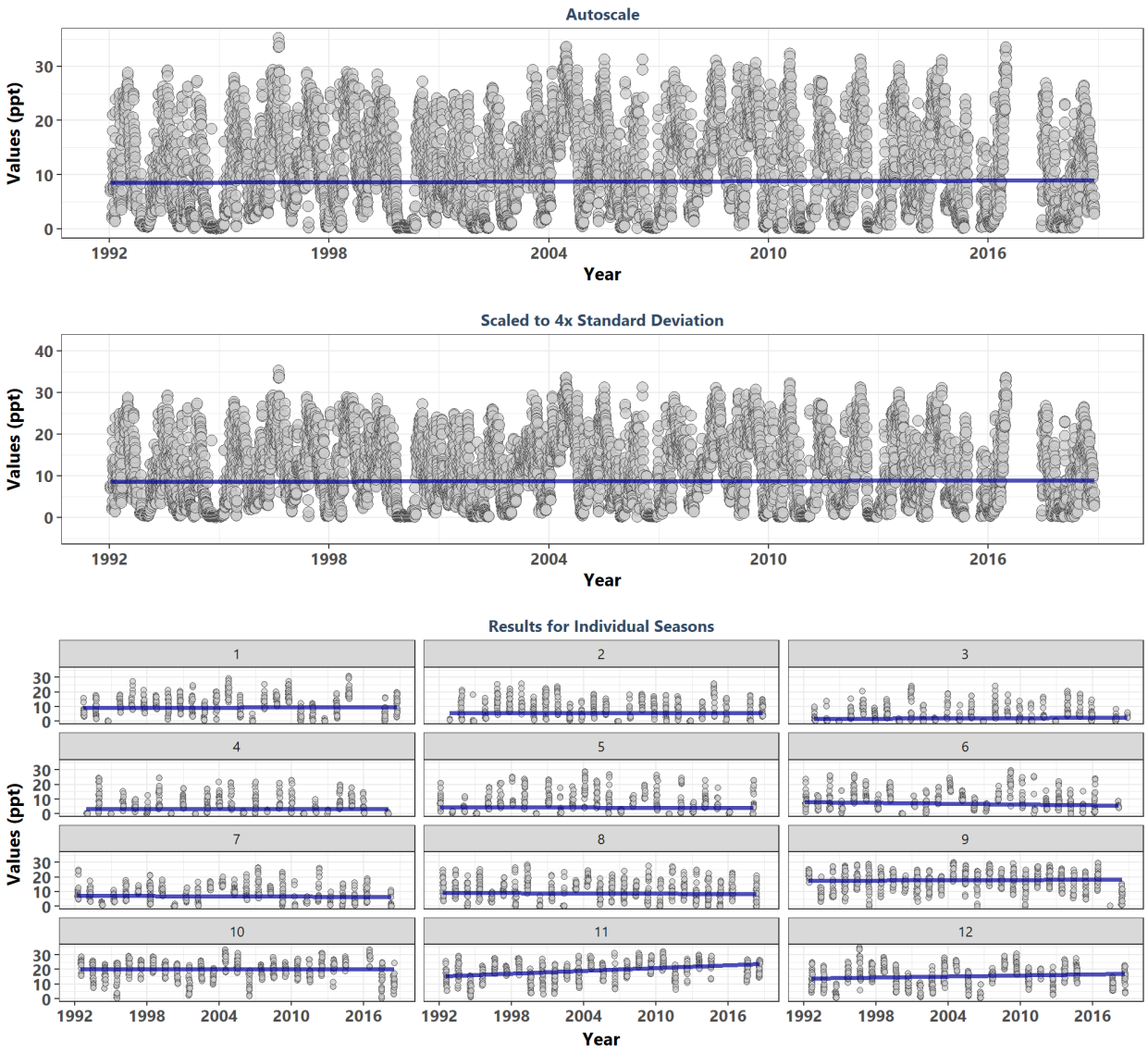
```
            scale_x_continuous(breaks = xbrks,
                               labels = xlabs) +
            labs(y = paste0("Values (", unit, ")"), x = "Year", subtitle = "Results for Individual Seas
            facet_wrap(~Season, ncol = 3) +
            plot_theme

      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)
   }
}
```

Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apadbwq



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	6367	21.64	-0.0007	-0.0022	21.6388	-0.1	0.8841	87.8	0	0
1	550	19.29	-0.0945	-0.1607	21.7832	-3.3	0.0009	NA	NA	0
2	509	16.22	-0.1167	-0.1797	19.0919	-3.9	0.0001	NA	NA	0
3	541	15.02	0.0037	0.0068	14.9118	0.1	0.8981	NA	NA	0
4	549	16.76	-0.0907	-0.2118	20.1519	-3.2	0.0015	NA	NA	0
5	553	22.96	-0.1009	-0.2303	26.6463	-3.6	0.0004	NA	NA	0
6	537	27.10	0.0710	0.1303	25.0168	2.5	0.0137	NA	NA	0
7	496	26.74	0.0856	0.1631	24.1288	2.9	0.0043	NA	NA	0
8	563	25.42	0.0720	0.1493	23.0361	2.6	0.0106	NA	NA	0
9	500	23.13	-0.0250	-0.0452	23.8047	-0.8	0.4033	NA	NA	0
10	499	22.33	0.0348	0.0522	21.4944	1.2	0.2451	NA	NA	0
11	509	23.02	0.0952	0.1318	20.9078	3.2	0.0013	NA	NA	0
12	561	20.94	0.0648	0.1141	19.1188	2.3	0.0215	NA	NA	0

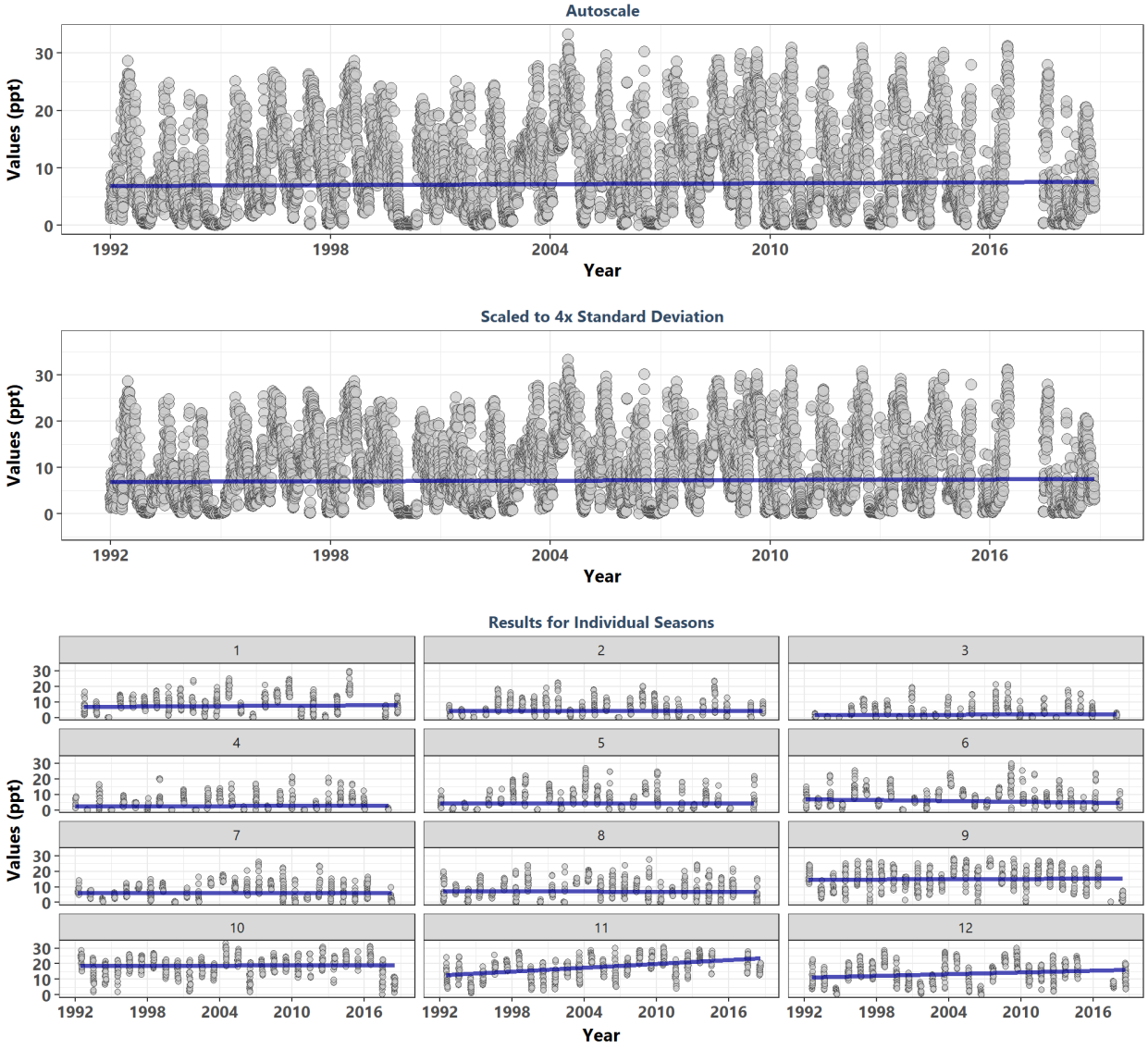
[a] $p < 0.00005$ appear as 0 due to rounding



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	8768	9.50	0.0211	0.0138	8.5424	2.9	0.0035	121.4	0	1
1	716	9.40	0.0204	0.0159	9.1927	0.8	0.4146	NA	NA	1
2	690	5.62	0.0027	0.0023	5.5925	0.1	0.9159	NA	NA	1
3	760	2.07	0.0803	0.0215	1.7886	3.3	0.0009	NA	NA	1
4	684	3.29	0.0025	0.0003	3.2899	0.1	0.9235	NA	NA	1
5	735	4.31	-0.0300	-0.0148	4.5009	-1.2	0.2229	NA	NA	-1
6	711	6.89	-0.0868	-0.0921	7.9009	-3.5	0.0005	NA	NA	-1
7	748	6.85	-0.0312	-0.0340	7.2539	-1.3	0.2018	NA	NA	-1
8	767	8.75	-0.0302	-0.0336	9.1839	-1.3	0.2102	NA	NA	-1
9	751	17.76	0.0198	0.0286	17.3866	0.8	0.4155	NA	NA	1
10	763	20.19	-0.0083	-0.0108	20.3236	-0.3	0.7329	NA	NA	-1
11	711	18.73	0.2385	0.3111	15.3122	9.5	0.0000	NA	NA	1
12	732	15.22	0.0798	0.1251	13.8482	3.2	0.0012	NA	NA	1

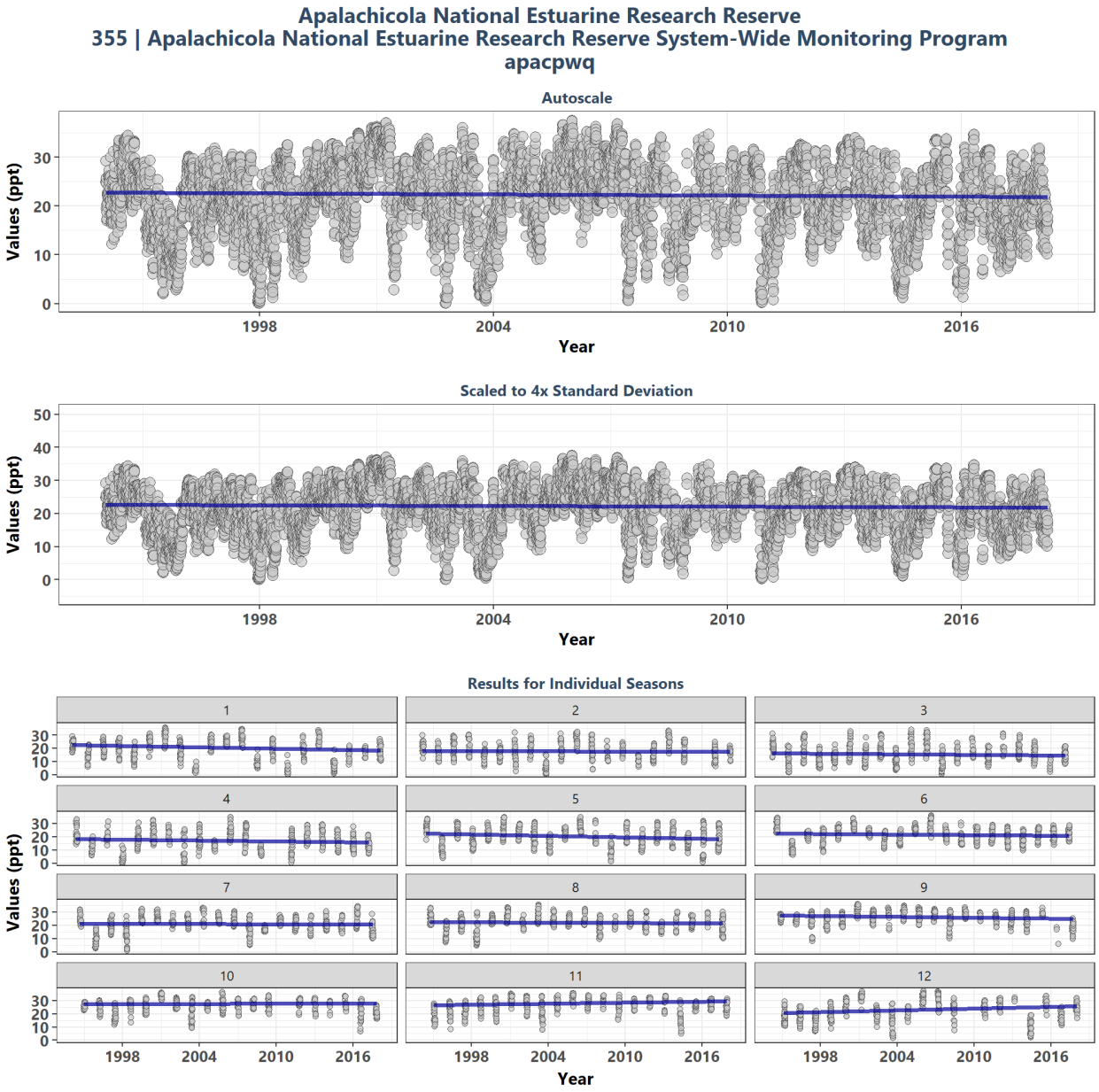
[a] $p < 0.00005$ appear as 0 due to rounding

Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apaeswq



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	8956	7.55	0.0360	0.0252	6.8851	5.1	0.0000	174.7	0	1
1	757	7.38	0.0467	0.0393	6.8704	1.9	0.0543	NA	NA	1
2	716	4.30	0.0023	0.0015	4.2779	0.1	0.9260	NA	NA	1
3	741	2.08	0.0393	0.0117	1.9112	1.6	0.1089	NA	NA	1
4	739	2.55	0.0299	0.0063	2.4729	1.2	0.2237	NA	NA	1
5	761	4.28	-0.0212	-0.0104	4.4106	-0.9	0.3804	NA	NA	-1
6	746	5.78	-0.1000	-0.0863	6.8997	-4.1	0.0000	NA	NA	-1
7	734	6.16	-0.0066	-0.0057	6.2312	-0.3	0.7881	NA	NA	-1
8	772	7.10	-0.0145	-0.0126	7.2494	-0.6	0.5475	NA	NA	-1
9	736	15.01	0.0229	0.0354	14.5823	0.9	0.3527	NA	NA	1
10	749	18.81	0.0187	0.0249	18.5149	0.8	0.4433	NA	NA	1
11	738	17.46	0.3025	0.4118	12.5182	12.3	0.0000	NA	NA	1
12	767	13.49	0.1137	0.1805	11.3198	4.7	0.0000	NA	NA	1

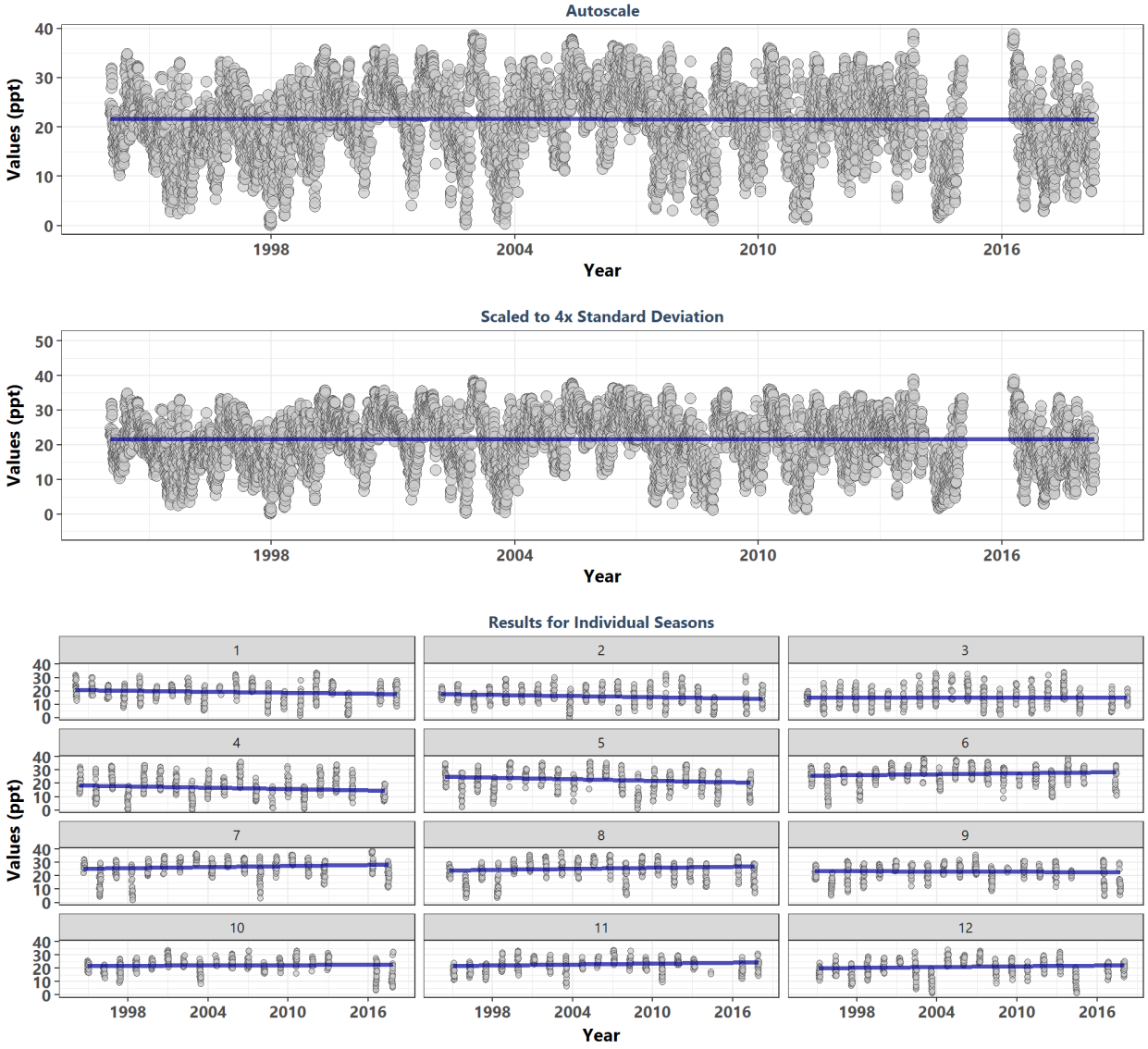
[a] $p < 0.00005$ appear as 0 due to rounding



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	6473	21.91	-0.0233	-0.0405	22.9785	-2.8	0.0046	71.7	0	-1
1	537	20.53	-0.1059	-0.2216	24.0732	-3.7	0.0002	NA	NA	-1
2	501	17.79	-0.0186	-0.0330	18.3166	-0.6	0.5332	NA	NA	-1
3	557	15.51	-0.0495	-0.0923	16.9908	-1.8	0.0800	NA	NA	-1
4	516	17.03	-0.0559	-0.1157	18.8761	-1.9	0.0573	NA	NA	-1
5	569	20.52	-0.1065	-0.2226	24.0775	-3.8	0.0001	NA	NA	-1
6	586	21.64	-0.0509	-0.0814	22.9392	-1.8	0.0650	NA	NA	-1
7	563	21.11	-0.0132	-0.0233	21.4838	-0.5	0.6394	NA	NA	-1
8	538	22.03	-0.0419	-0.0619	23.0177	-1.5	0.1455	NA	NA	-1
9	498	26.39	-0.0810	-0.1131	28.0835	-2.7	0.0068	NA	NA	-1
10	533	27.43	0.0213	0.0294	26.9623	0.7	0.4618	NA	NA	1
11	552	27.76	0.1104	0.1629	25.1485	3.9	0.0001	NA	NA	1
12	523	23.04	0.1157	0.2641	18.8115	4.0	0.0001	NA	NA	1

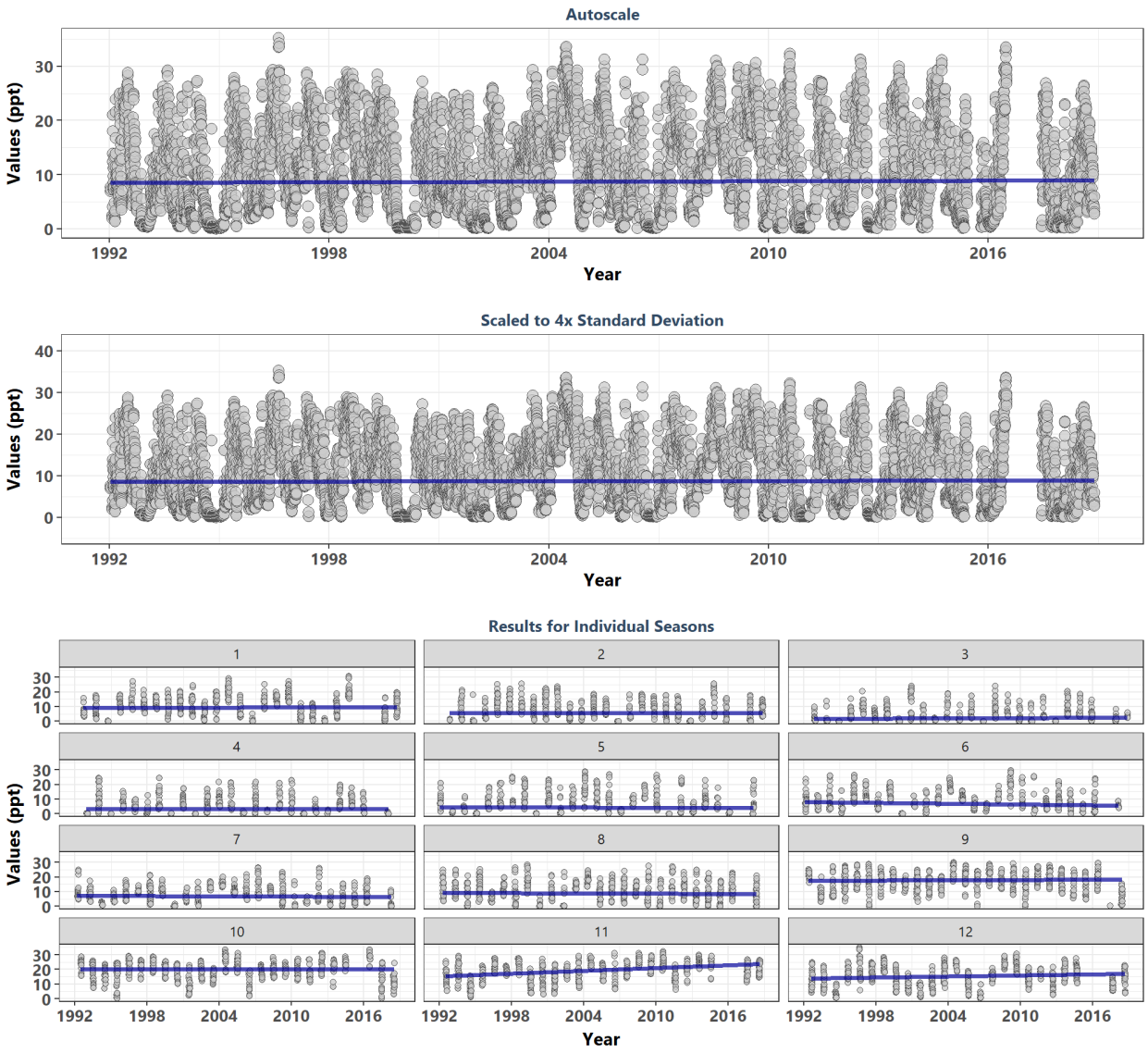
[a] $p < 0.00005$ appear as 0 due to rounding

Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apadbwq



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	6367	21.64	-0.0007	-0.0022	21.6388	-0.1	0.8841	87.8	0	0
1	550	19.29	-0.0945	-0.1607	21.7832	-3.3	0.0009	NA	NA	0
2	509	16.22	-0.1167	-0.1797	19.0919	-3.9	0.0001	NA	NA	0
3	541	15.02	0.0037	0.0068	14.9118	0.1	0.8981	NA	NA	0
4	549	16.76	-0.0907	-0.2118	20.1519	-3.2	0.0015	NA	NA	0
5	553	22.96	-0.1009	-0.2303	26.6463	-3.6	0.0004	NA	NA	0
6	537	27.10	0.0710	0.1303	25.0168	2.5	0.0137	NA	NA	0
7	496	26.74	0.0856	0.1631	24.1288	2.9	0.0043	NA	NA	0
8	563	25.42	0.0720	0.1493	23.0361	2.6	0.0106	NA	NA	0
9	500	23.13	-0.0250	-0.0452	23.8047	-0.8	0.4033	NA	NA	0
10	499	22.33	0.0348	0.0522	21.4944	1.2	0.2451	NA	NA	0
11	509	23.02	0.0952	0.1318	20.9078	3.2	0.0013	NA	NA	0
12	561	20.94	0.0648	0.1141	19.1188	2.3	0.0215	NA	NA	0

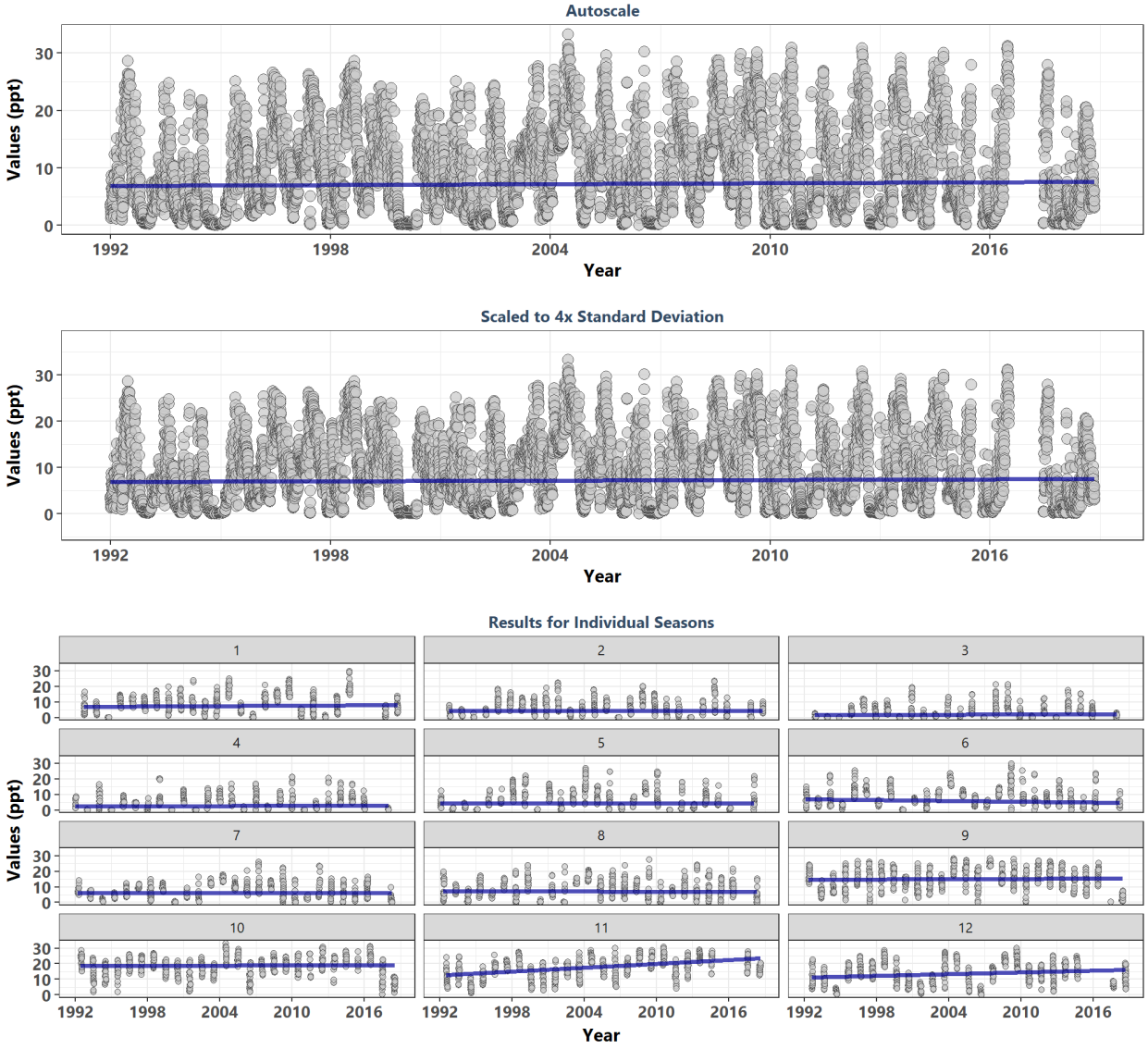
[a] $p < 0.00005$ appear as 0 due to rounding



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	8768	9.50	0.0211	0.0138	8.5424	2.9	0.0035	121.4	0	1
1	716	9.40	0.0204	0.0159	9.1927	0.8	0.4146	NA	NA	1
2	690	5.62	0.0027	0.0023	5.5925	0.1	0.9159	NA	NA	1
3	760	2.07	0.0803	0.0215	1.7886	3.3	0.0009	NA	NA	1
4	684	3.29	0.0025	0.0003	3.2899	0.1	0.9235	NA	NA	1
5	735	4.31	-0.0300	-0.0148	4.5009	-1.2	0.2229	NA	NA	-1
6	711	6.89	-0.0868	-0.0921	7.9009	-3.5	0.0005	NA	NA	-1
7	748	6.85	-0.0312	-0.0340	7.2539	-1.3	0.2018	NA	NA	-1
8	767	8.75	-0.0302	-0.0336	9.1839	-1.3	0.2102	NA	NA	-1
9	751	17.76	0.0198	0.0286	17.3866	0.8	0.4155	NA	NA	1
10	763	20.19	-0.0083	-0.0108	20.3236	-0.3	0.7329	NA	NA	-1
11	711	18.73	0.2385	0.3111	15.3122	9.5	0.0000	NA	NA	1
12	732	15.22	0.0798	0.1251	13.8482	3.2	0.0012	NA	NA	1

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

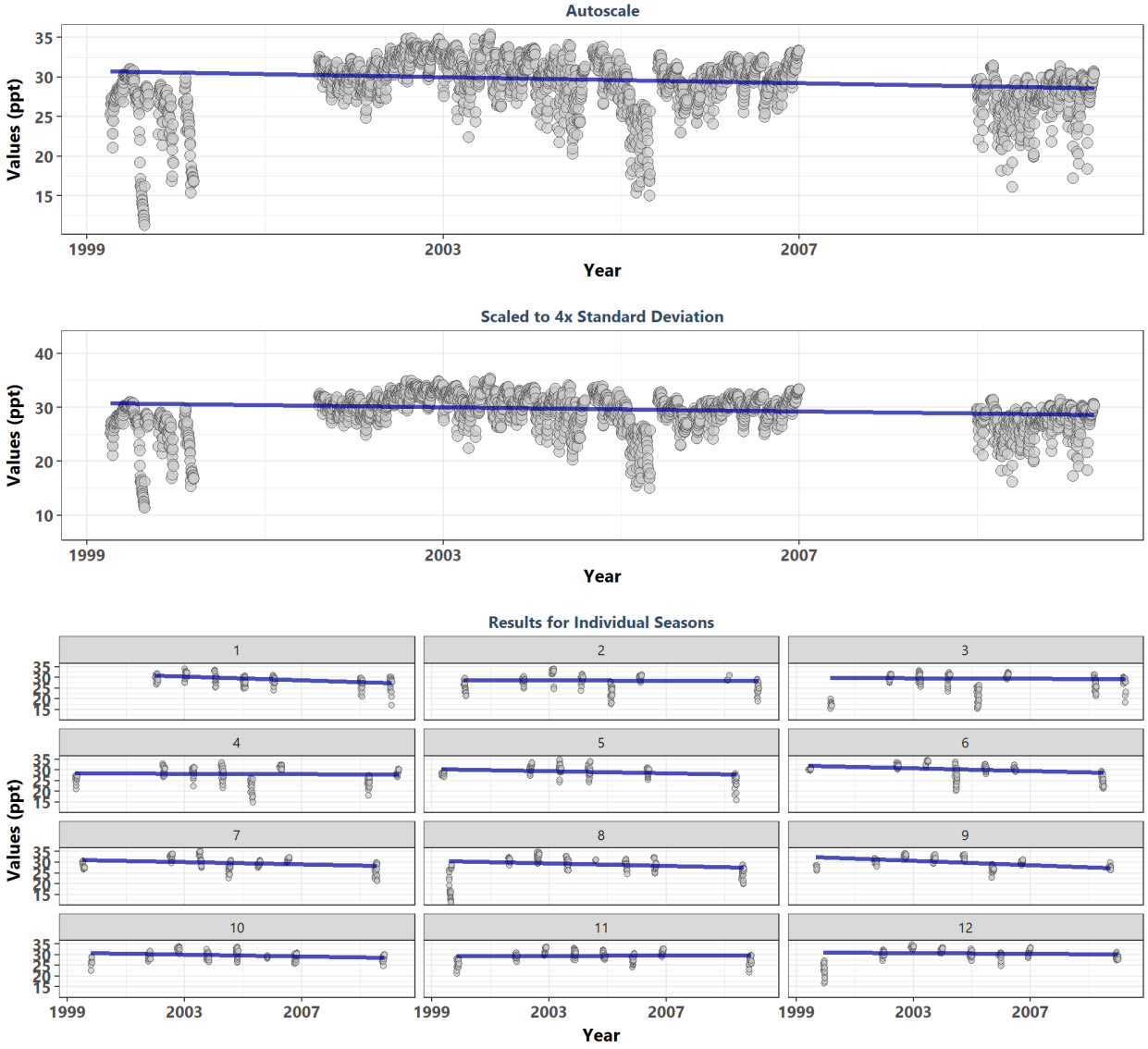
Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apaeswq



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	8956	7.55	0.0360	0.0252	6.8851	5.1	0.0000	174.7	0	1
1	757	7.38	0.0467	0.0393	6.8704	1.9	0.0543	NA	NA	1
2	716	4.30	0.0023	0.0015	4.2779	0.1	0.9260	NA	NA	1
3	741	2.08	0.0393	0.0117	1.9112	1.6	0.1089	NA	NA	1
4	739	2.55	0.0299	0.0063	2.4729	1.2	0.2237	NA	NA	1
5	761	4.28	-0.0212	-0.0104	4.4106	-0.9	0.3804	NA	NA	-1
6	746	5.78	-0.1000	-0.0863	6.8997	-4.1	0.0000	NA	NA	-1
7	734	6.16	-0.0066	-0.0057	6.2312	-0.3	0.7881	NA	NA	-1
8	772	7.10	-0.0145	-0.0126	7.2494	-0.6	0.5475	NA	NA	-1
9	736	15.01	0.0229	0.0354	14.5823	0.9	0.3527	NA	NA	1
10	749	18.81	0.0187	0.0249	18.5149	0.8	0.4433	NA	NA	1
11	738	17.46	0.3025	0.4118	12.5182	12.3	0.0000	NA	NA	1
12	767	13.49	0.1137	0.1805	11.3198	4.7	0.0000	NA	NA	1

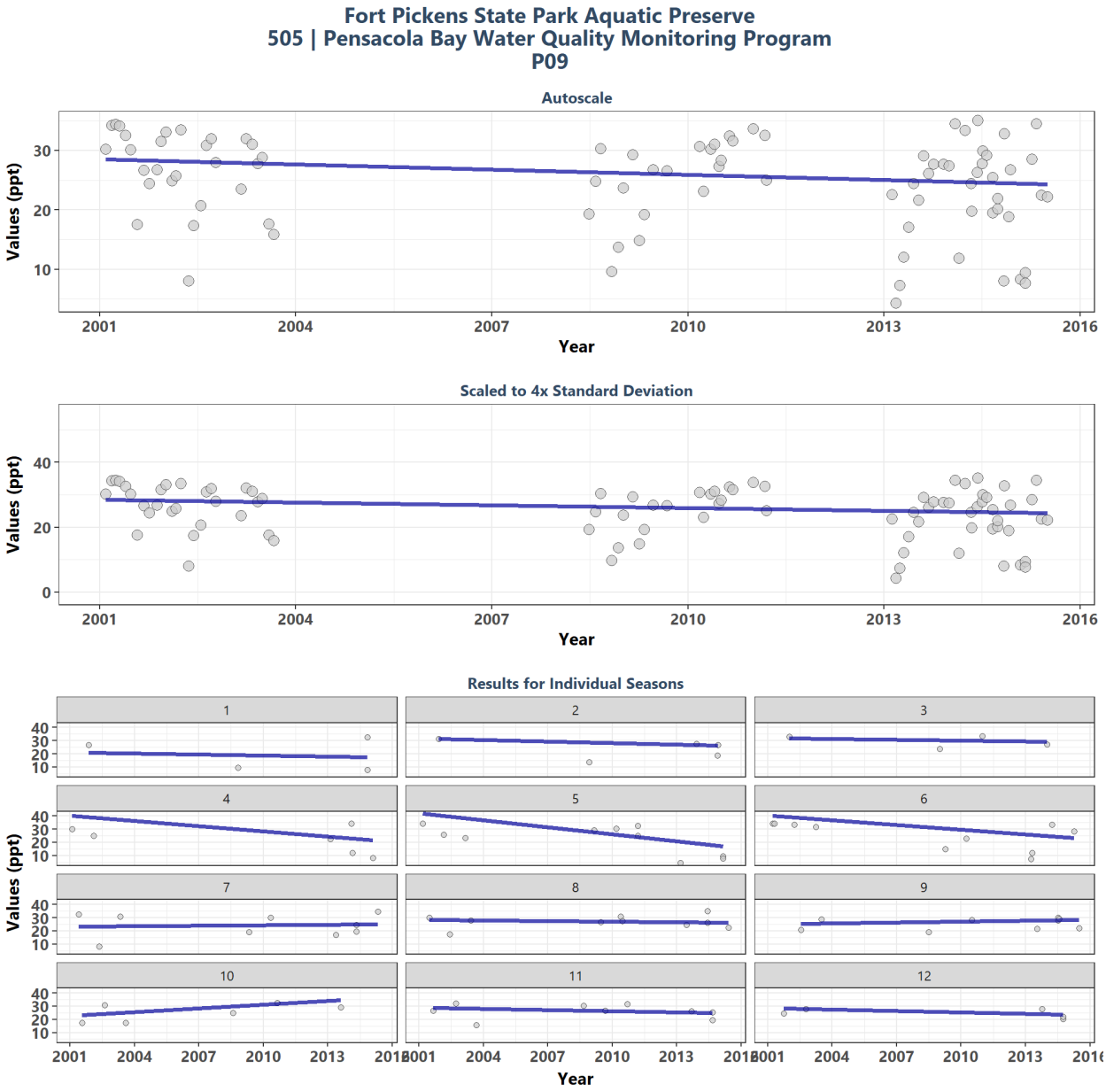
[a] $p < 0.00005$ appear as 0 due to rounding

Big Bend Seagrasses Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring
BBSSK



Season	N	Median	tau	Slope	Int.	z	p__z	chi__sq	p__chi__sq	Trend
All	2565	29.70	-0.1301	-0.1954	31.5629	-9.5	0.0000	67.5	0	-1
1	213	29.54	-0.3266	-0.4127	33.6663	-7.2	0.0000	NA	NA	-1
2	205	28.69	-0.0158	-0.0340	28.9939	-0.3	0.7344	NA	NA	-1
3	232	29.42	-0.0349	-0.0537	29.9552	-0.8	0.4252	NA	NA	-1
4	230	28.11	-0.0204	-0.0379	28.4924	-0.5	0.6431	NA	NA	-1
5	186	29.29	-0.1925	-0.2566	31.4701	-4.0	0.0001	NA	NA	-1
6	195	30.48	-0.2370	-0.3390	33.5346	-5.0	0.0000	NA	NA	-1
7	212	29.77	-0.1534	-0.2743	32.2407	-3.4	0.0008	NA	NA	-1
8	213	29.44	-0.1317	-0.2891	31.7563	-2.9	0.0039	NA	NA	-1
9	213	30.23	-0.2912	-0.4958	34.6927	-6.4	0.0000	NA	NA	-1
10	204	30.03	-0.1738	-0.1999	31.6274	-3.7	0.0002	NA	NA	-1
11	240	29.54	0.0198	0.0293	29.2951	0.5	0.6451	NA	NA	1
12	222	30.85	-0.0576	-0.0807	31.4983	-1.3	0.1975	NA	NA	-1

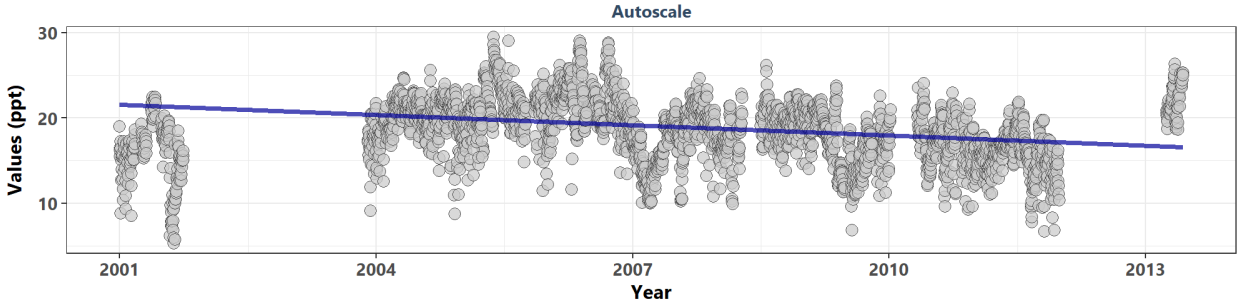
[a] $p < 0.00005$ appear as 0 due to rounding

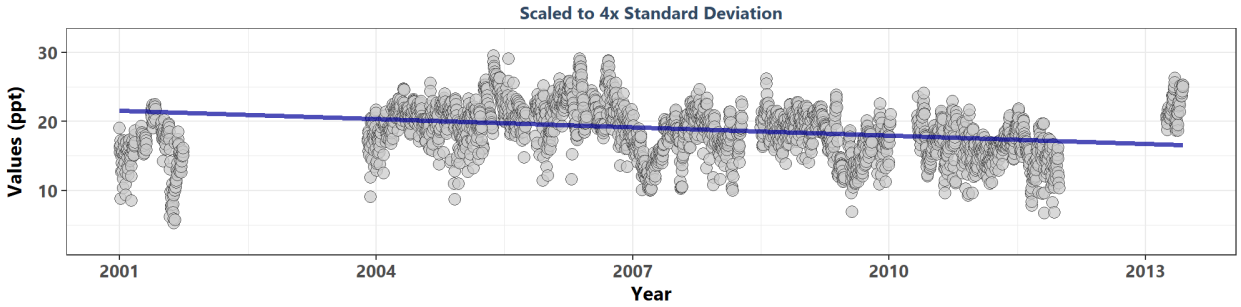


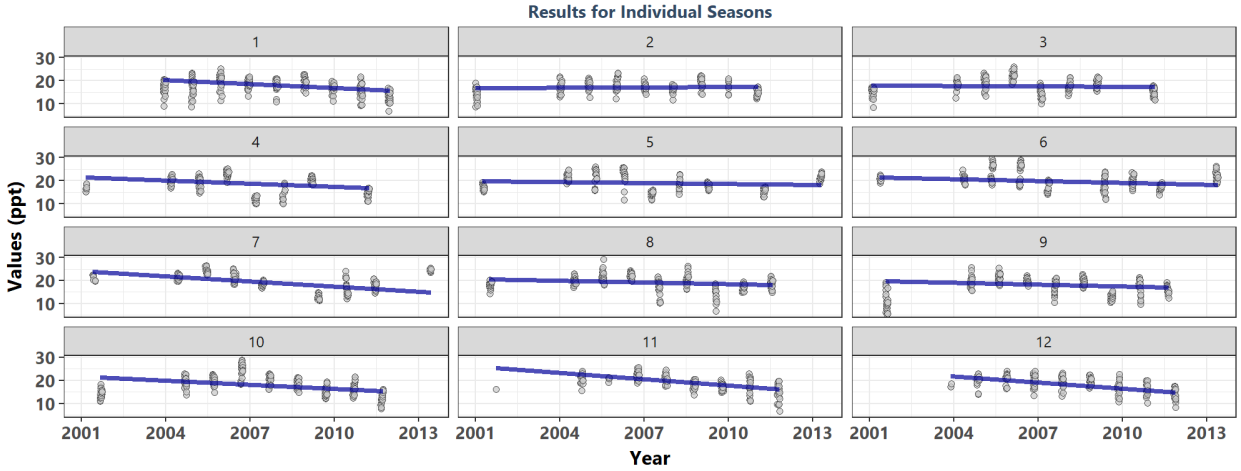
Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	87	26.63	-0.1812	-0.2883	28.5400	-2.1	0.0365	12.7	0.3143	-1
1	4	18.21	-0.1667	-0.2720	21.1969	0.0	1.0000	NA	NA	-1
2	5	26.69	-0.3000	-0.3725	31.5316	-0.5	0.6134	NA	NA	-1
3	4	30.21	0.0000	-0.2028	32.0346	0.0	1.0000	NA	NA	-1
4	6	23.72	-0.5333	-1.3406	40.4768	-1.3	0.1806	NA	NA	-1
5	10	25.34	-0.4222	-1.7706	42.1615	-1.6	0.1046	NA	NA	-1
6	10	30.22	-0.5111	-1.2074	40.4807	-2.0	0.0473	NA	NA	-1
7	9	24.45	0.0833	0.1388	23.1959	0.2	0.8339	NA	NA	1
8	10	27.03	-0.1111	-0.1342	28.2340	-0.4	0.7183	NA	NA	-1
9	9	27.78	0.2500	0.2084	25.2740	0.9	0.3947	NA	NA	1
10	6	26.94	0.4667	0.9639	22.6000	1.1	0.2597	NA	NA	1
11	9	26.56	-0.3611	-0.2671	28.6992	-1.3	0.2084	NA	NA	-1
12	5	24.44	-0.5000	-0.3288	28.3809	-1.0	0.3122	NA	NA	-1

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

Nature Coast Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring
BBSHS







Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	2922	18.61	-0.2029	-0.3984	21.5217	-16.1	0.0000	146.1	0	-1
1	269	17.98	-0.2710	-0.5467	21.8026	-6.7	0.0000	NA	NA	-1
2	231	17.05	0.0364	0.0499	16.7485	0.8	0.4076	NA	NA	1
3	244	17.69	-0.0262	-0.0524	18.0069	-0.6	0.5398	NA	NA	-1
4	203	18.72	-0.1646	-0.4737	21.5632	-3.5	0.0004	NA	NA	-1
5	254	18.99	-0.0859	-0.1558	19.9256	-2.1	0.0400	NA	NA	-1
6	248	19.79	-0.1740	-0.2812	21.4802	-4.1	0.0000	NA	NA	-1
7	229	19.68	-0.3170	-0.7398	24.1179	-7.2	0.0000	NA	NA	-1
8	260	19.14	-0.1693	-0.2441	20.6073	-4.1	0.0000	NA	NA	-1
9	270	18.28	-0.1064	-0.2592	19.8331	-2.6	0.0087	NA	NA	-1
10	279	18.15	-0.2402	-0.5999	21.7492	-6.0	0.0000	NA	NA	-1
11	204	19.57	-0.5072	-0.9131	25.9575	-10.9	0.0000	NA	NA	-1
12	231	19.05	-0.4668	-0.8744	24.2983	-10.6	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

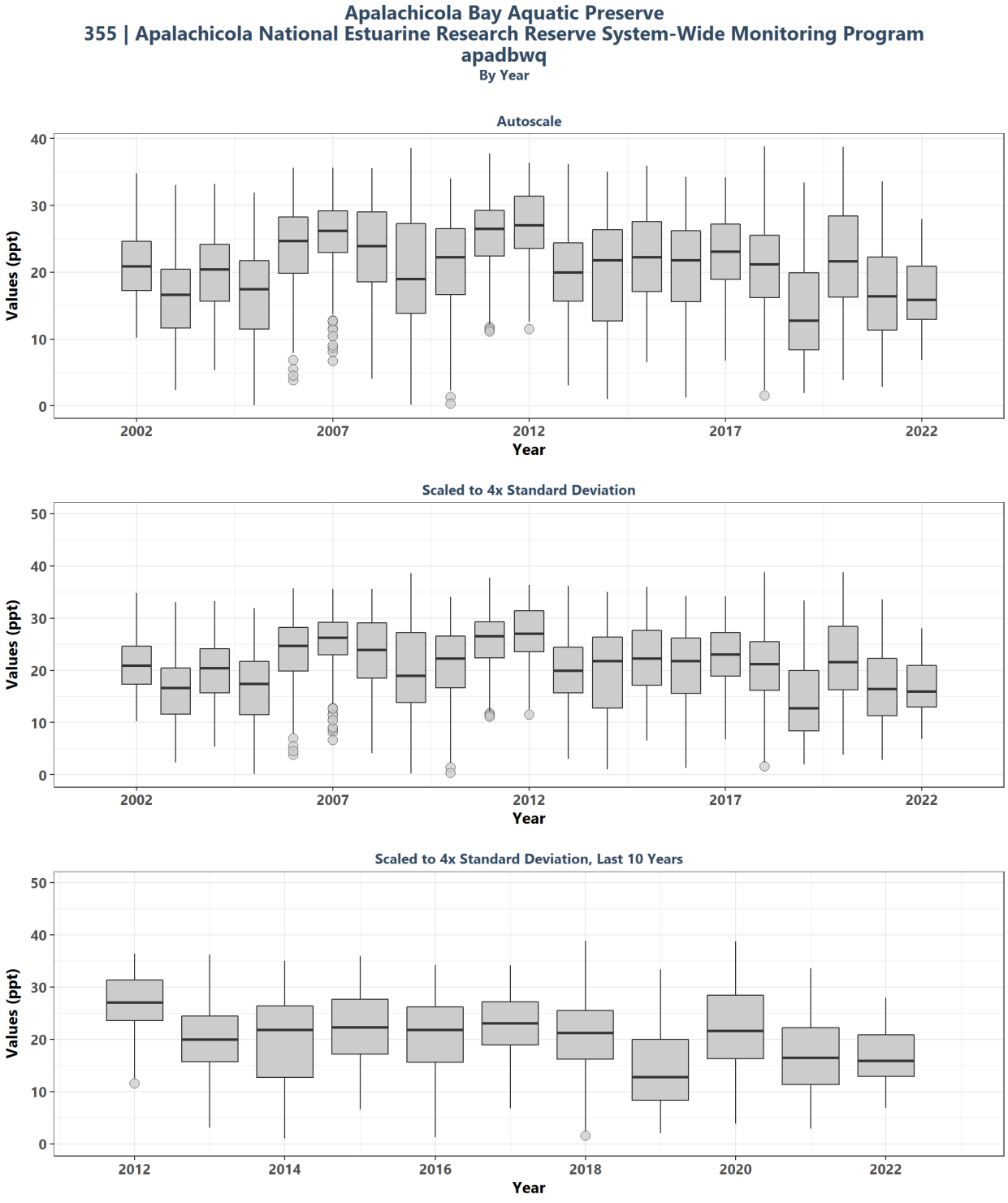
```
if(n==0){
   print("There are no monitoring locations that qualify.")
} else {
   for (i in 1:n) {
      year_lower <- min(data$Year[data$Use_In_Analysis==TRUE &
                                  data$MonitoringID==Mon_IDs[i]])
      year_upper <- max(data$Year[data$Use_In_Analysis==TRUE &
     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

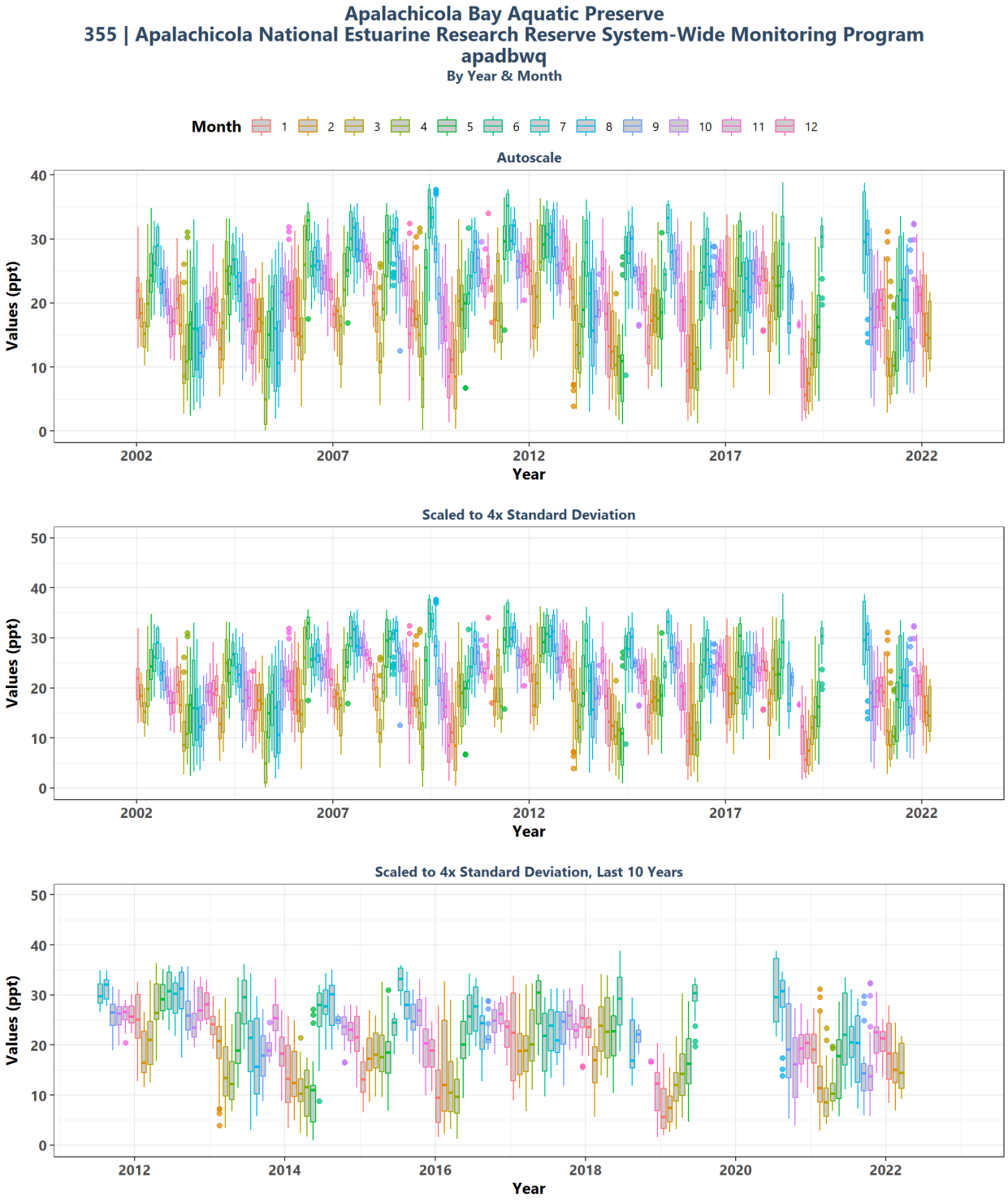
```
            aes(x=Year, y=ResultValue, group=Year)) +
   geom_boxplot(color="#333333", fill="#cccccc", outlier.shape=21,
                outlier.size=3, outlier.color="#333333",
                outlier.fill="#cccccc", outlier.alpha=0.75) +
   labs(subtitle="Autoscale",
        x="Year", y=paste0("Values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

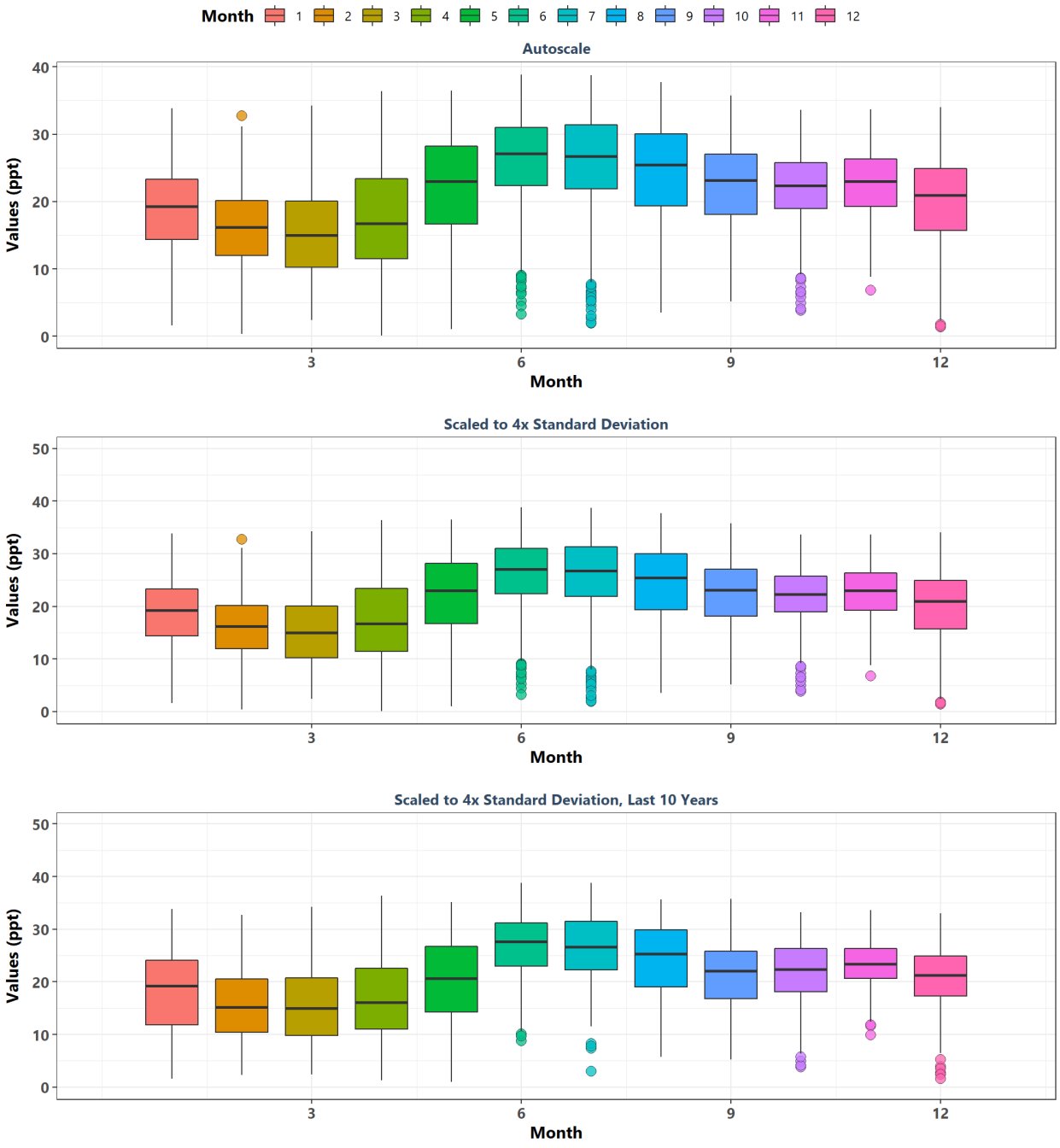
```
   geom_boxplot(color="#333333", outlier.shape=21, outlier.size=3,
                outlier.color="#333333", outlier.alpha=0.75) +
   labs(subtitle="Autoscale",
        x="Month", y=paste0("Values (", unit, ")"), fill="Month") +
   scale_x_continuous(limits=c(0, 13), breaks=seq(3, 12, 3)) +
   plot_theme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

```
    }
}
```

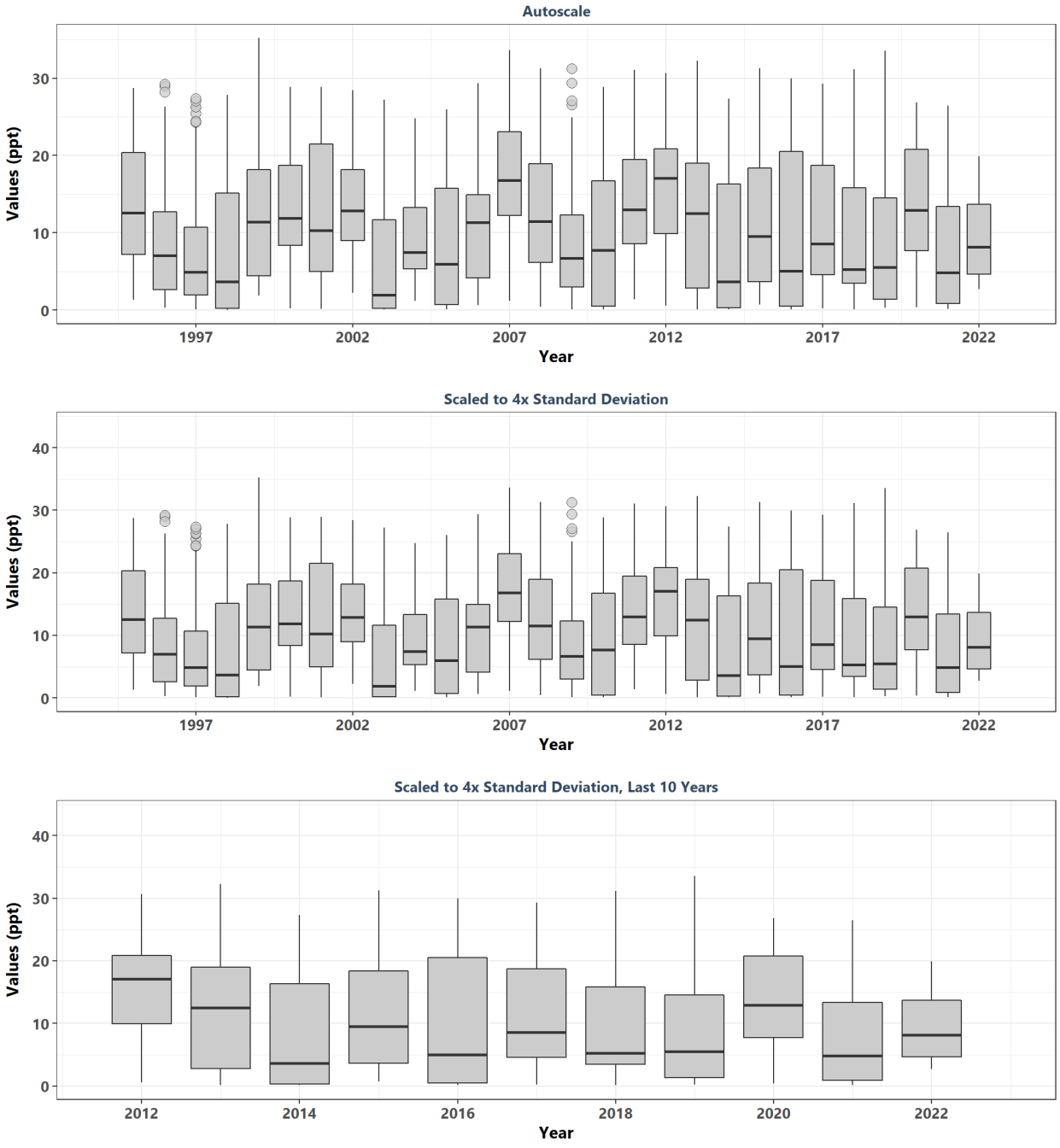

Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apadbwq
By Year
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year

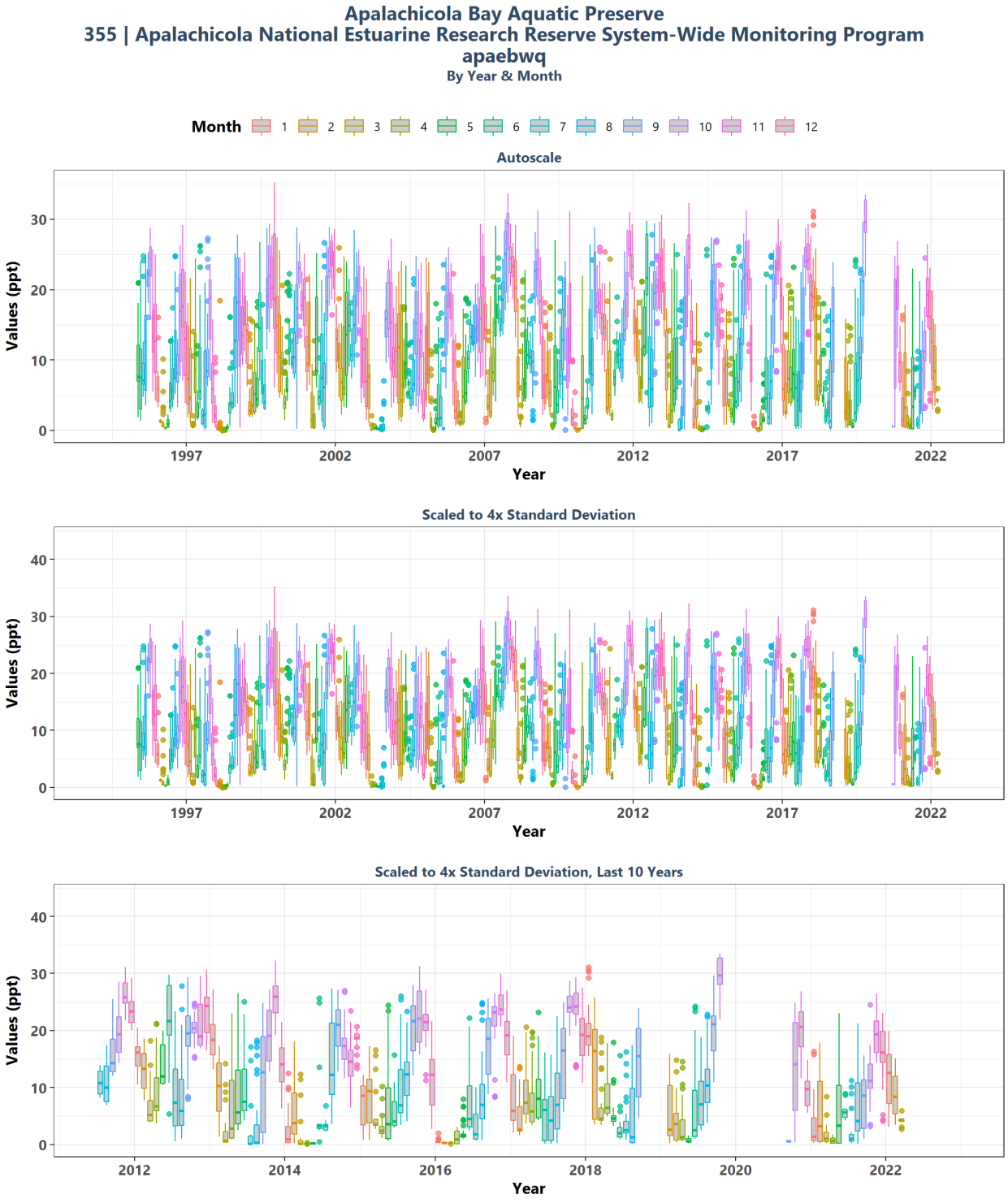

Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apadbwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years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

Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apadbwq
By Month

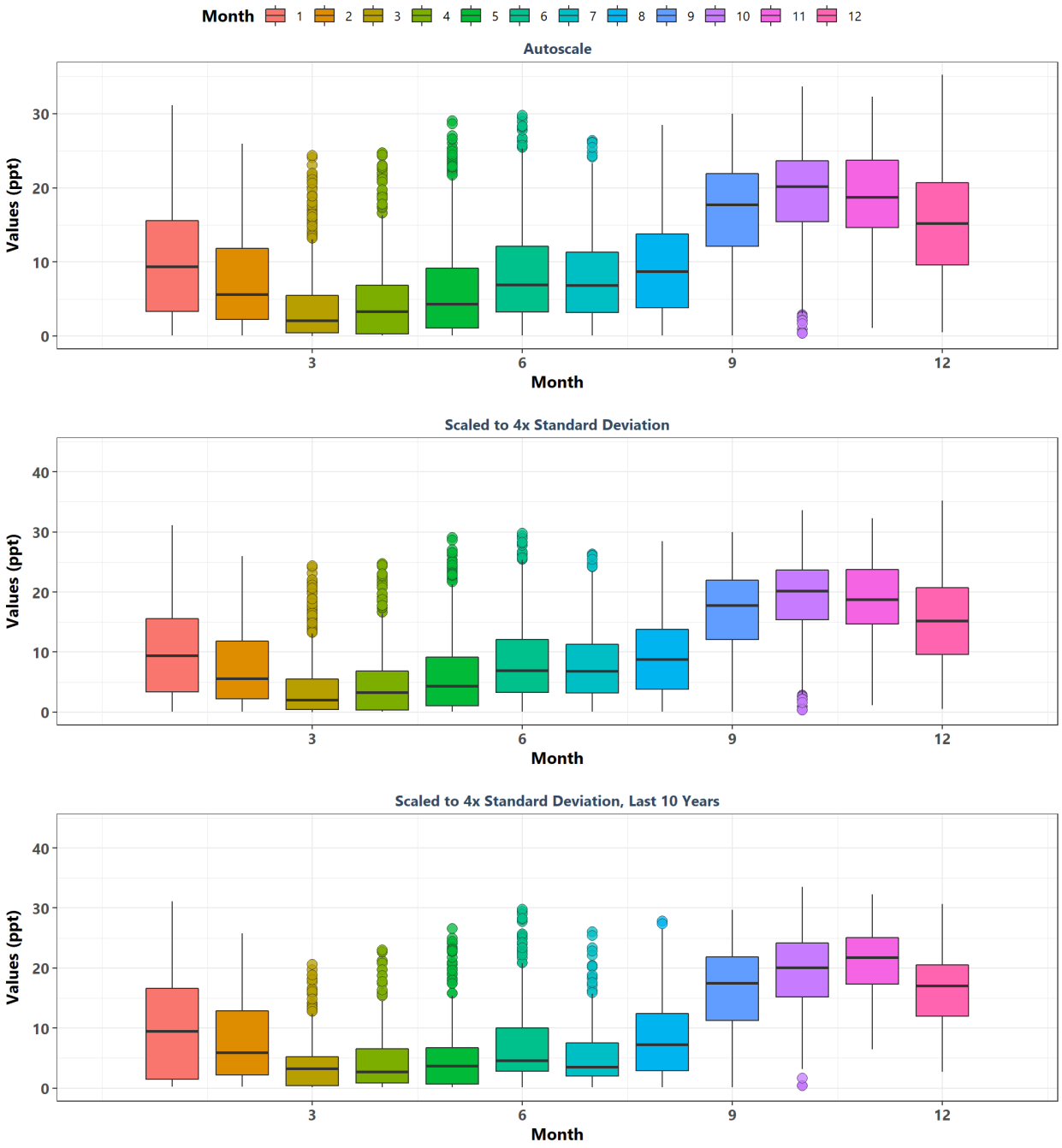


Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apaebwq
By Year

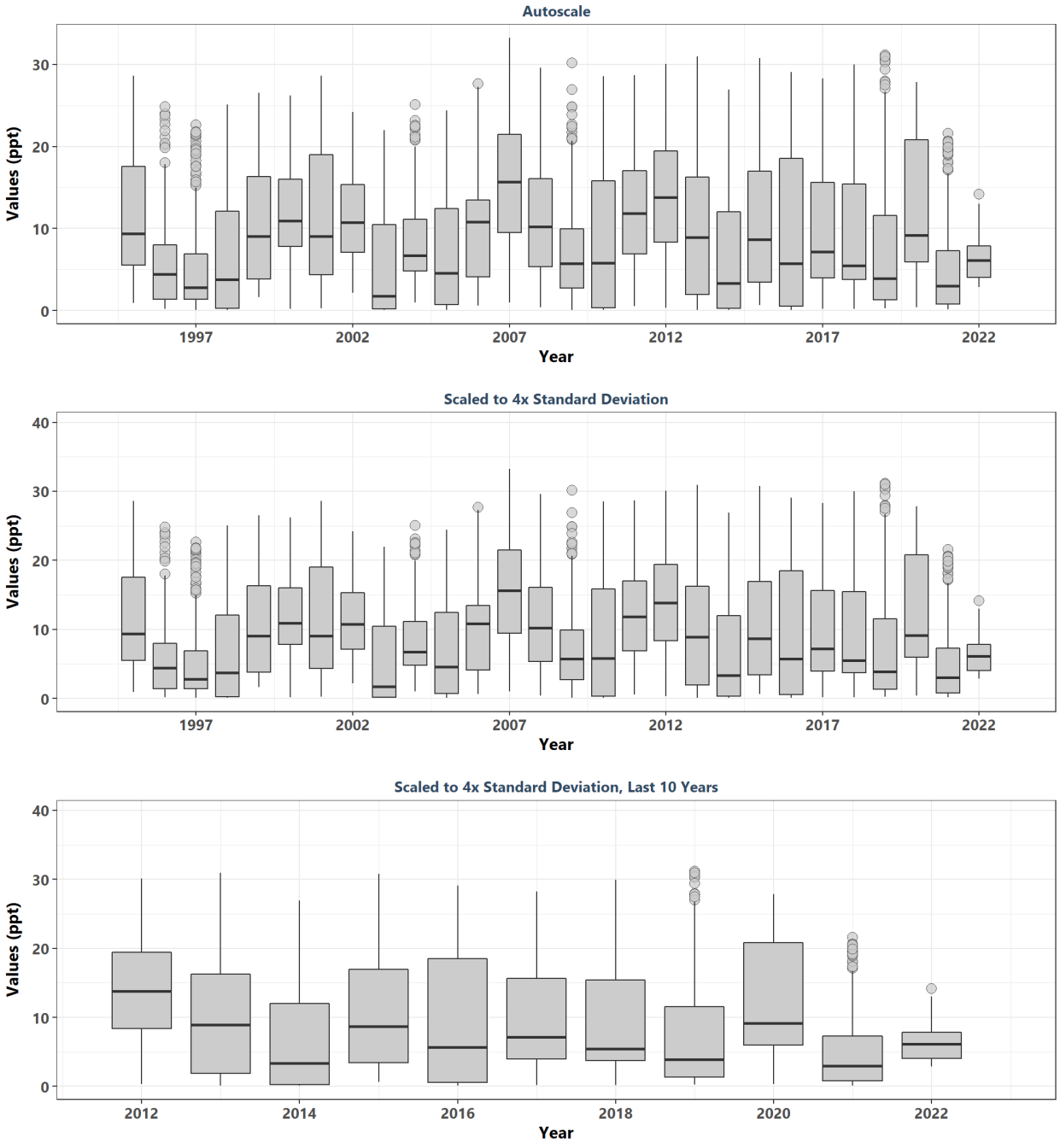


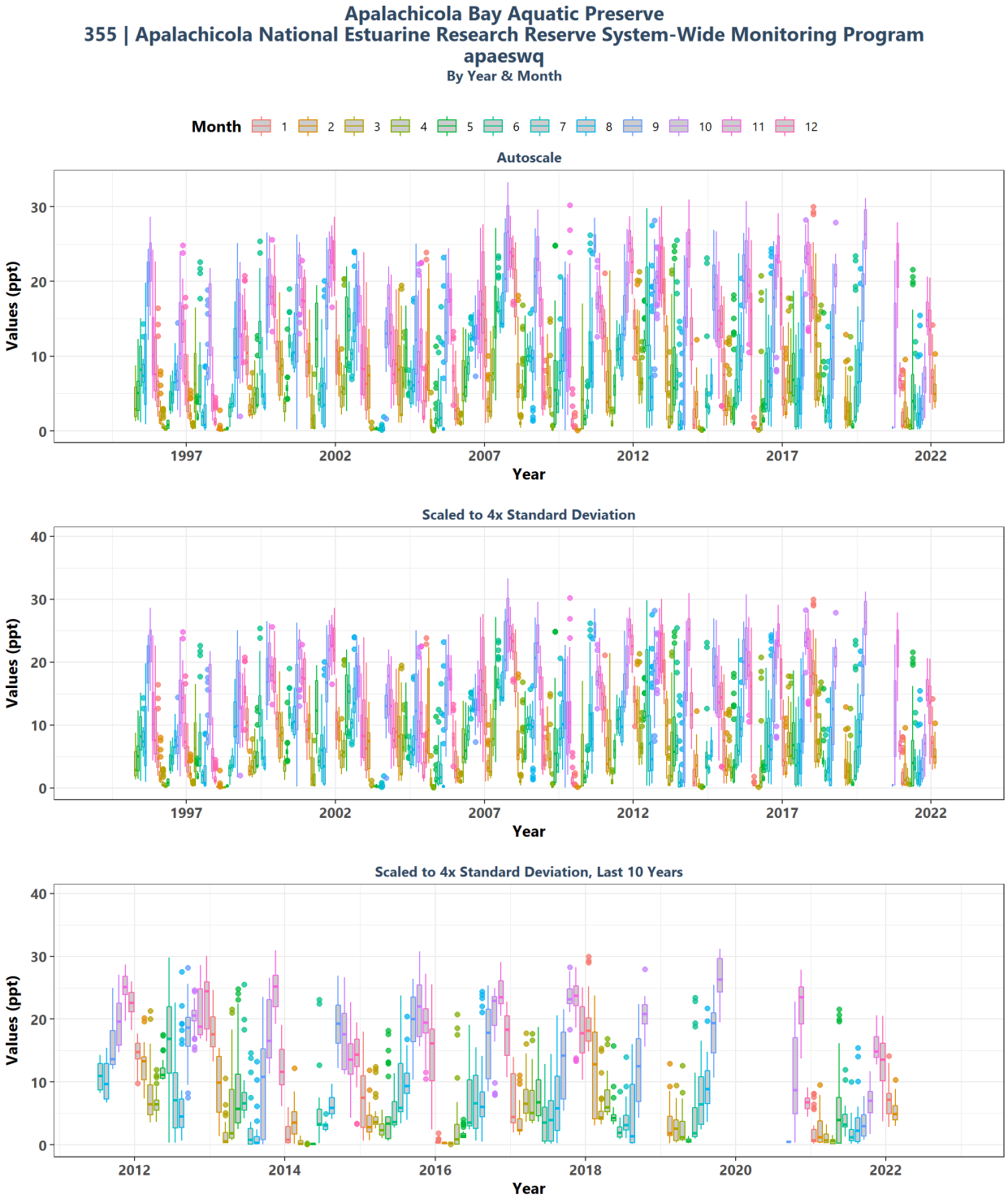

Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apaebwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years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

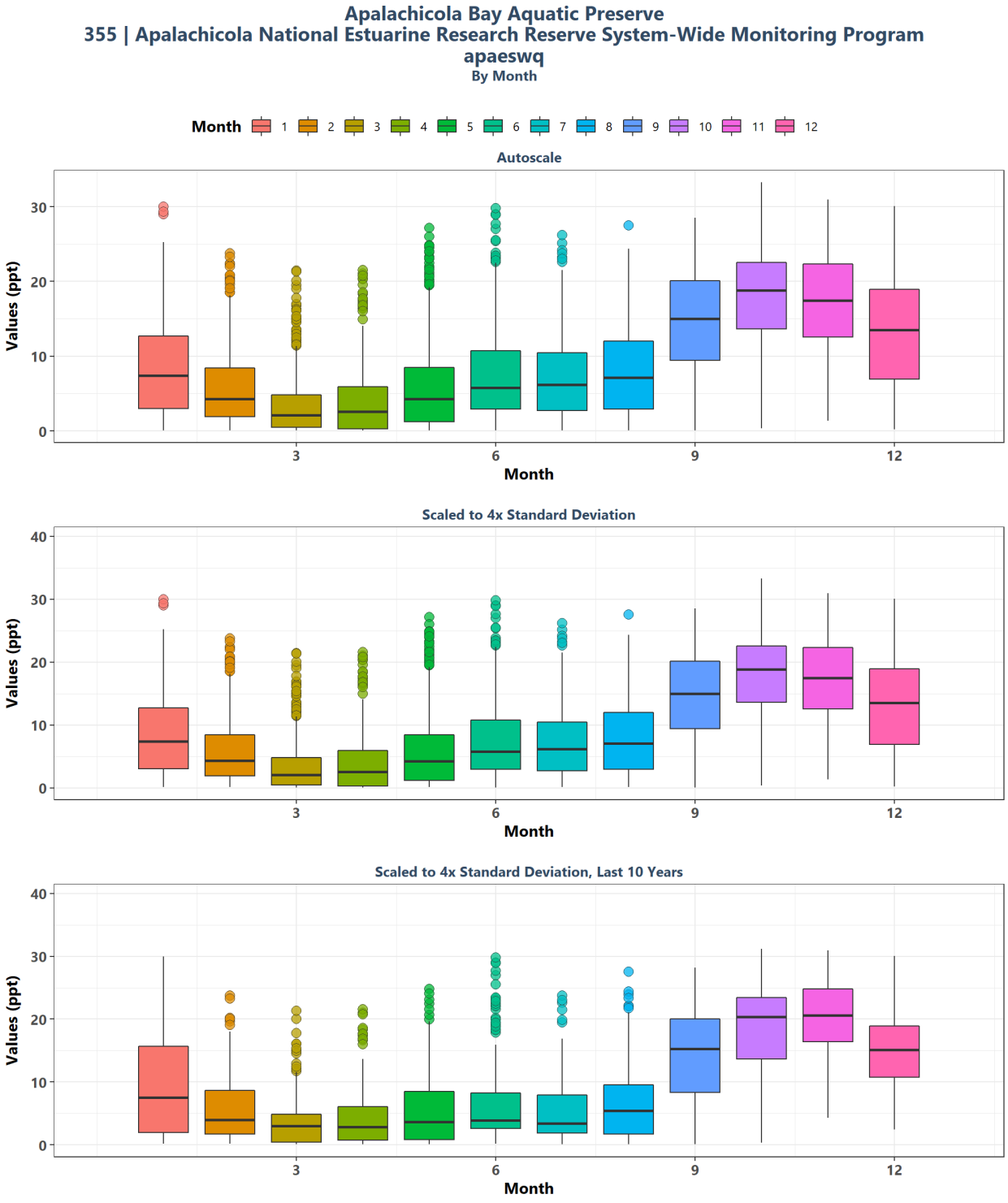
Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apaebwq
By Month



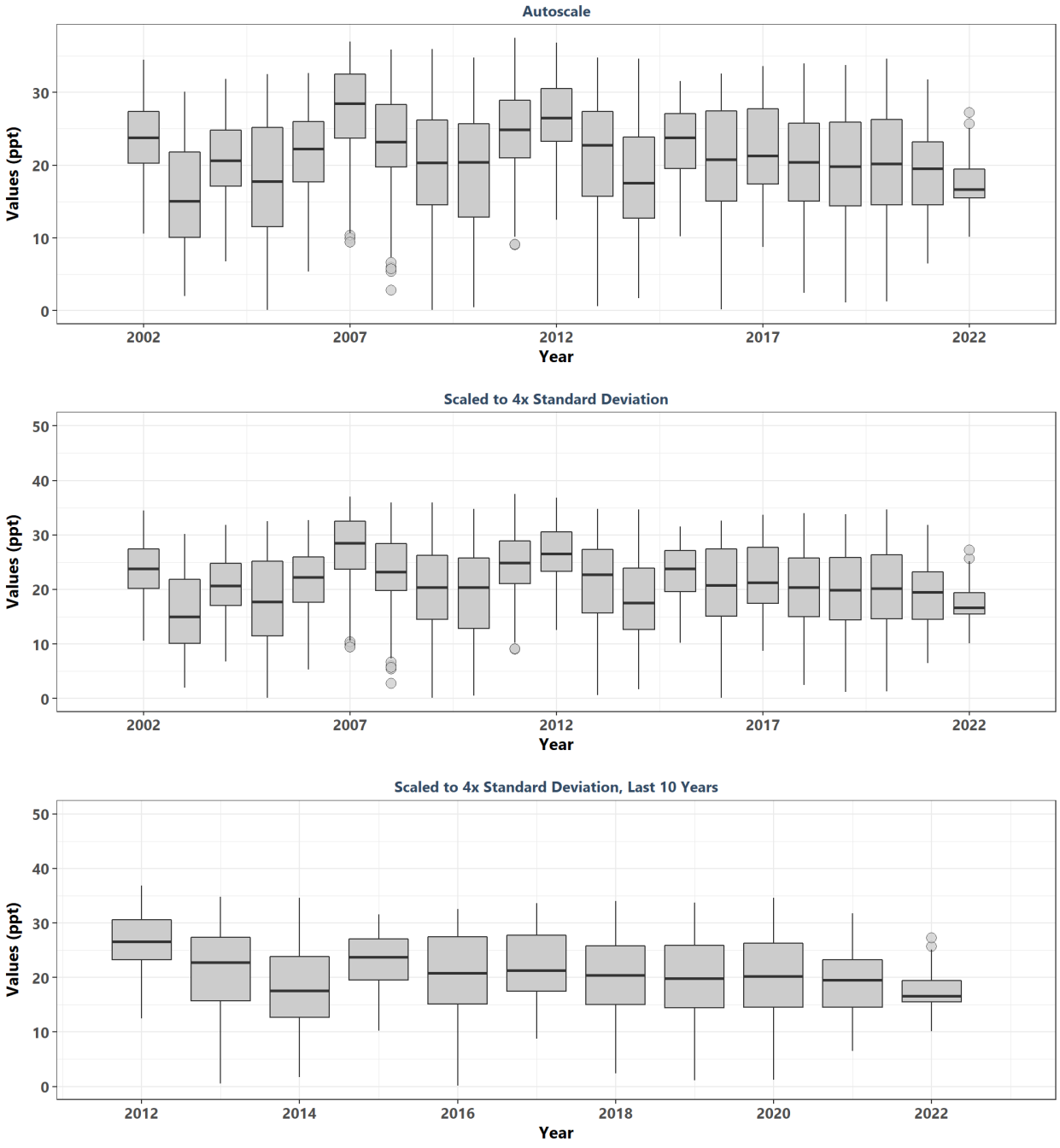
Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apaeswq
By Year

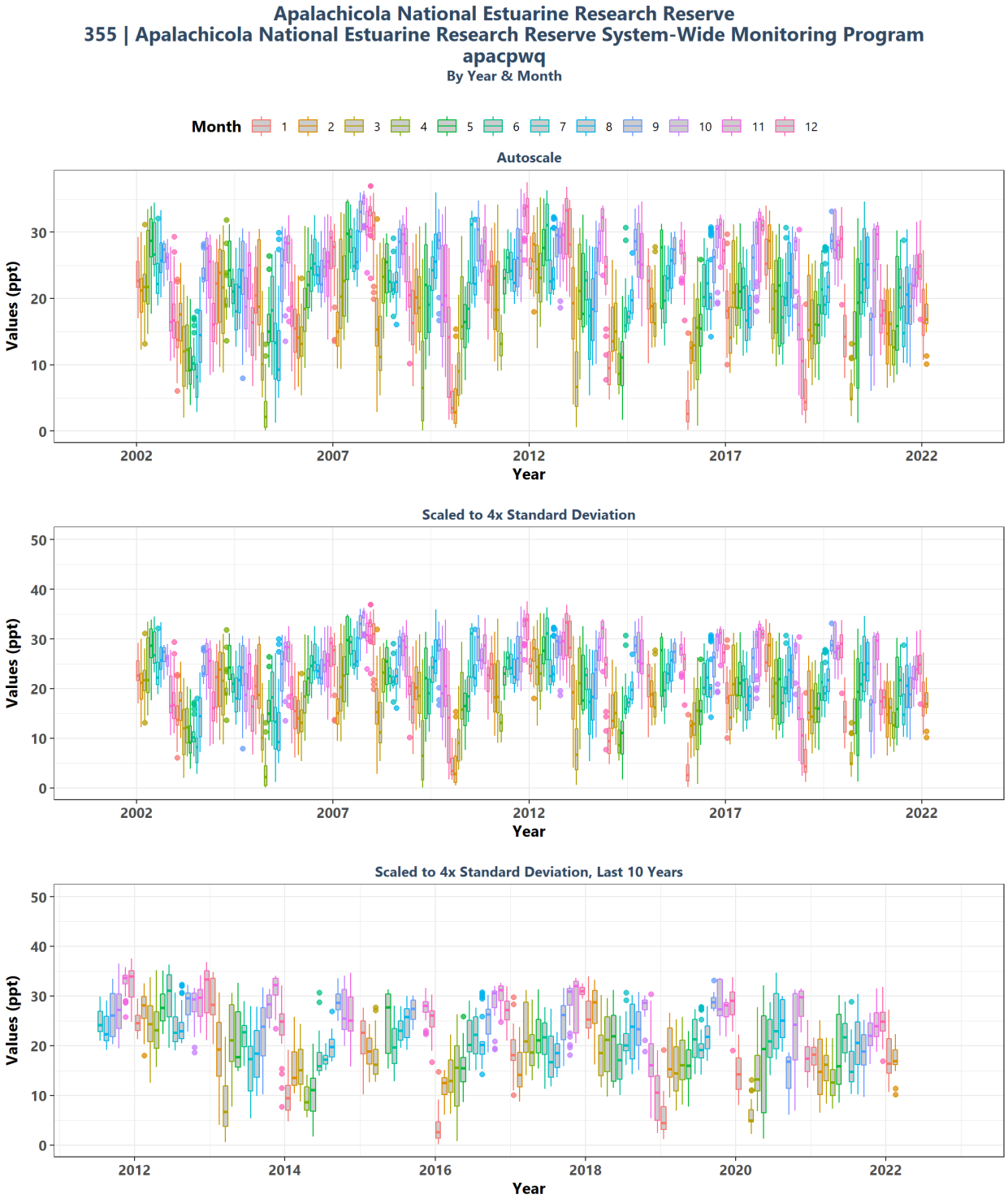



Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apaeswq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years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

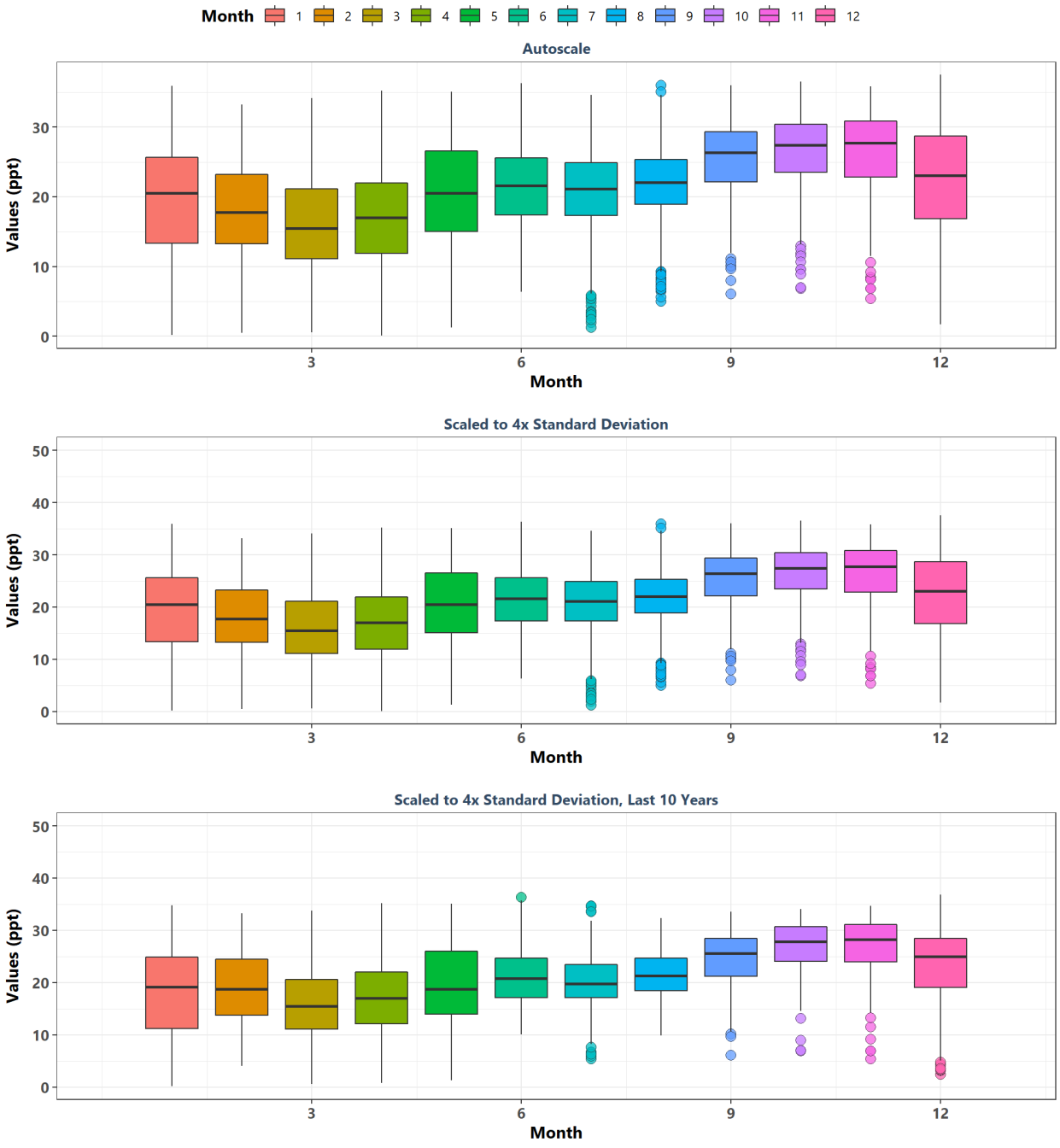

Apalachicola Bay Aquatic Preserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apaeswq
By Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Month
Scaled to 4x Standard Deviation
Values (ppt)
Month
Scaled to 4x Standard Deviation, Last 10 Years
Values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

Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apacpwq
By Year

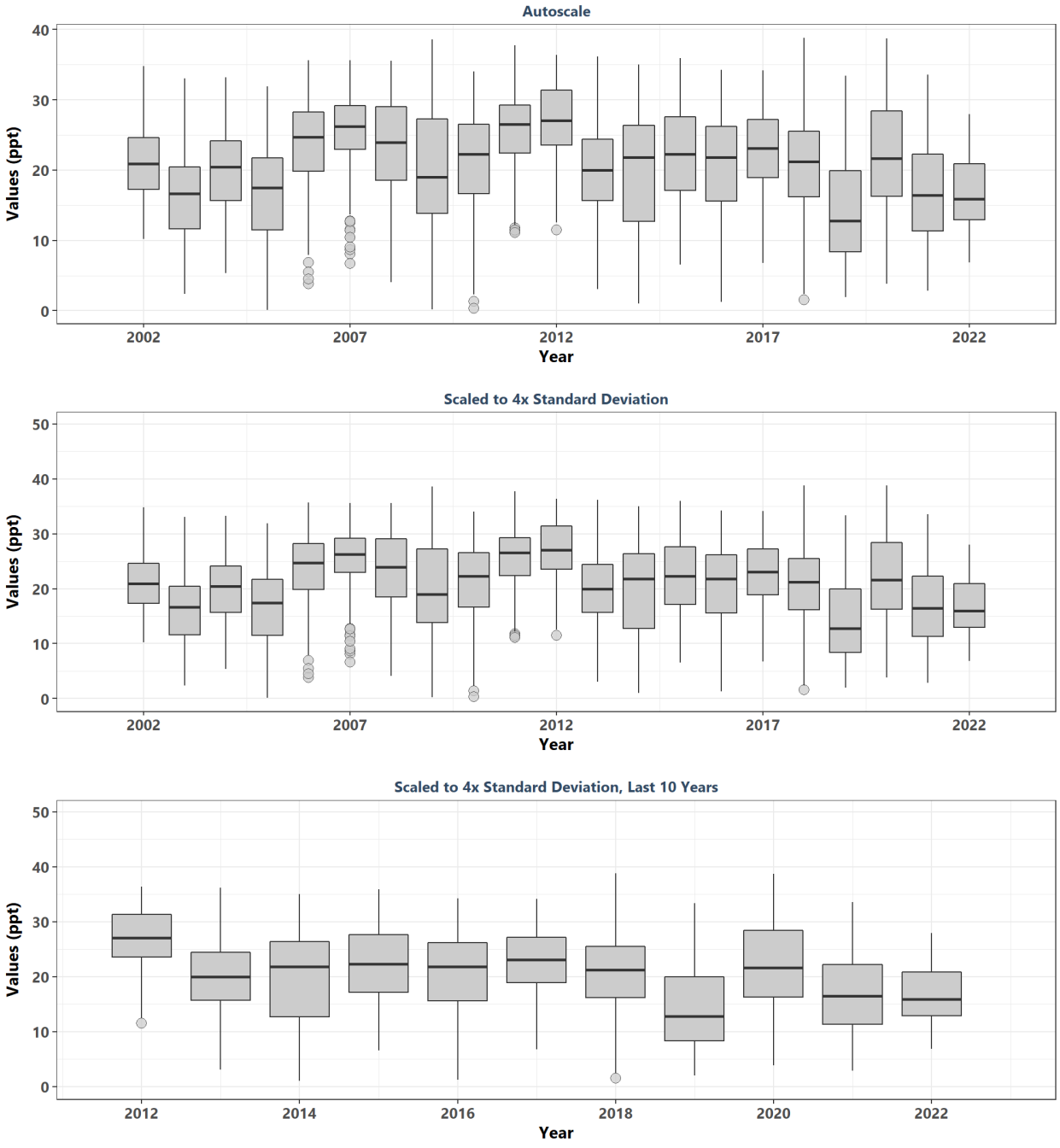


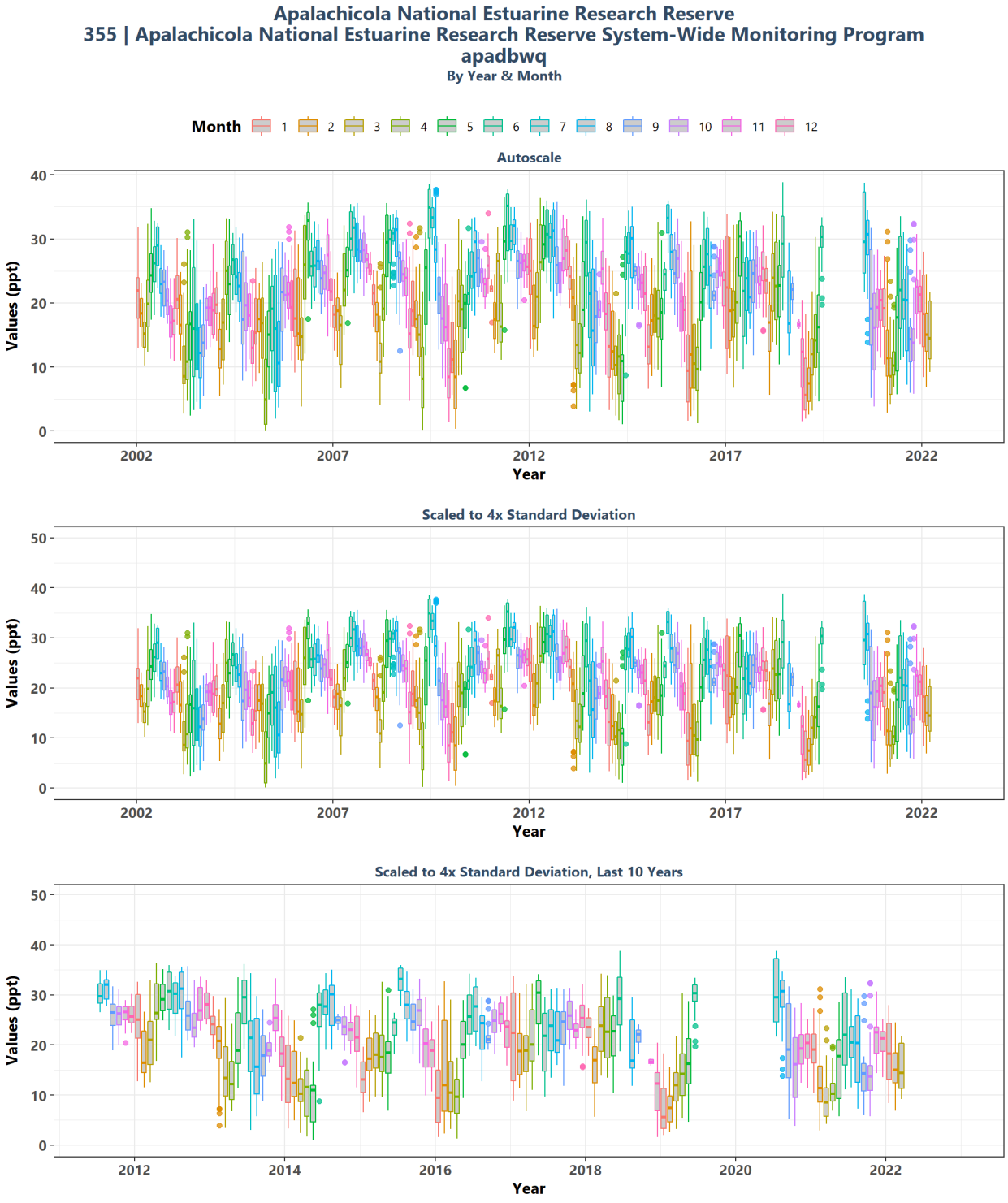

Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apacpwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation,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

Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apacpwq
By Month

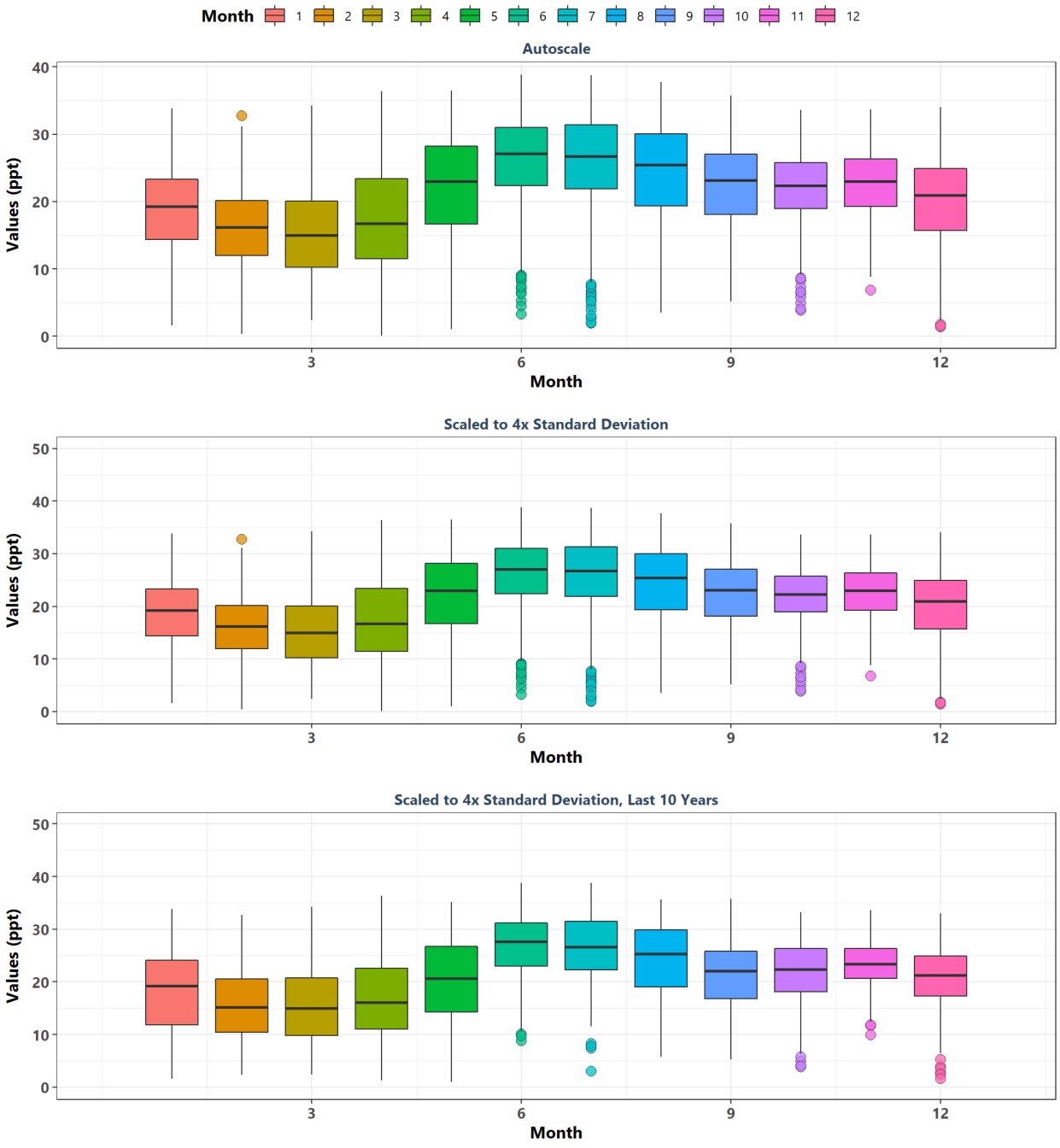


Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apadbwq
By Year

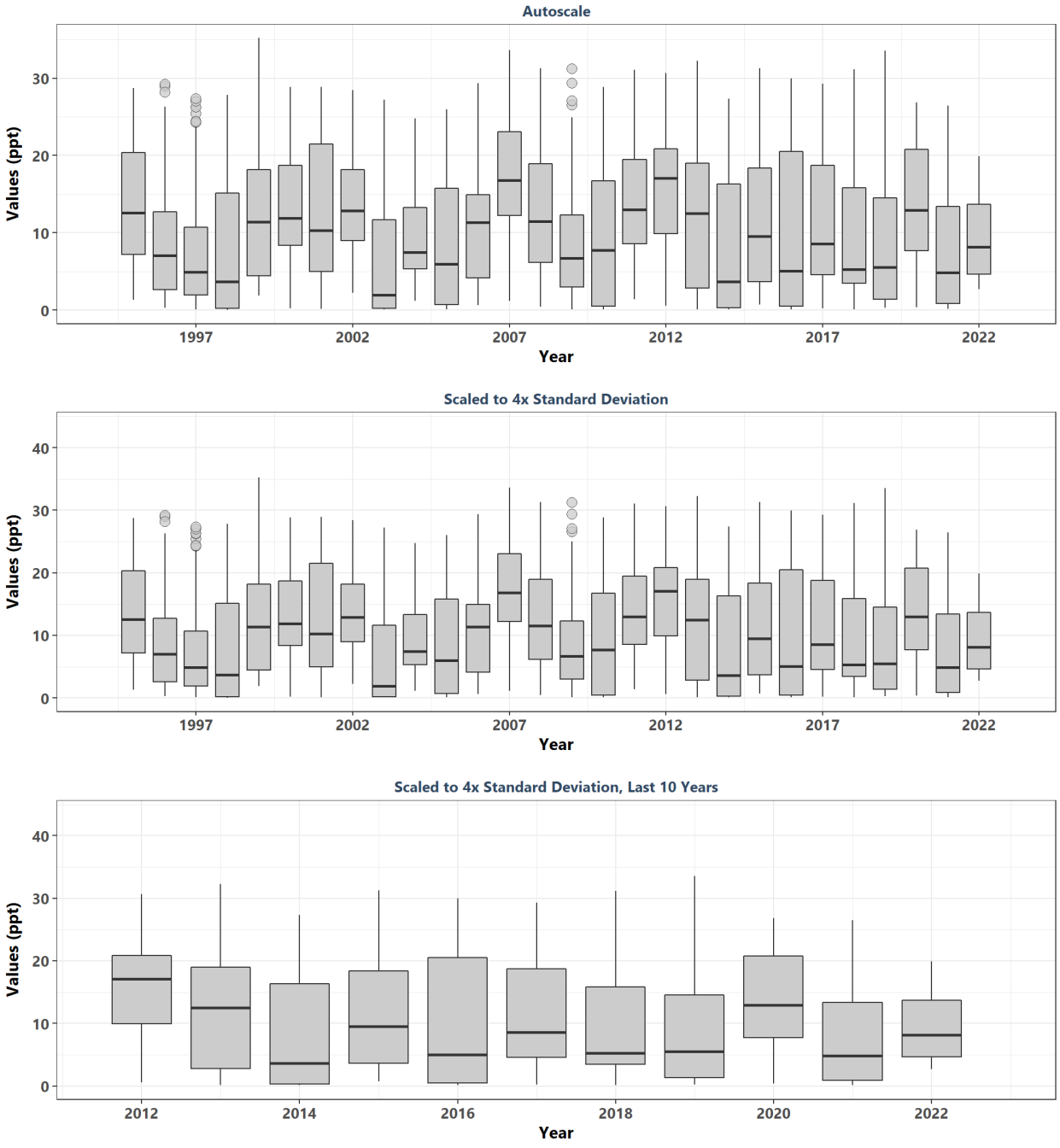


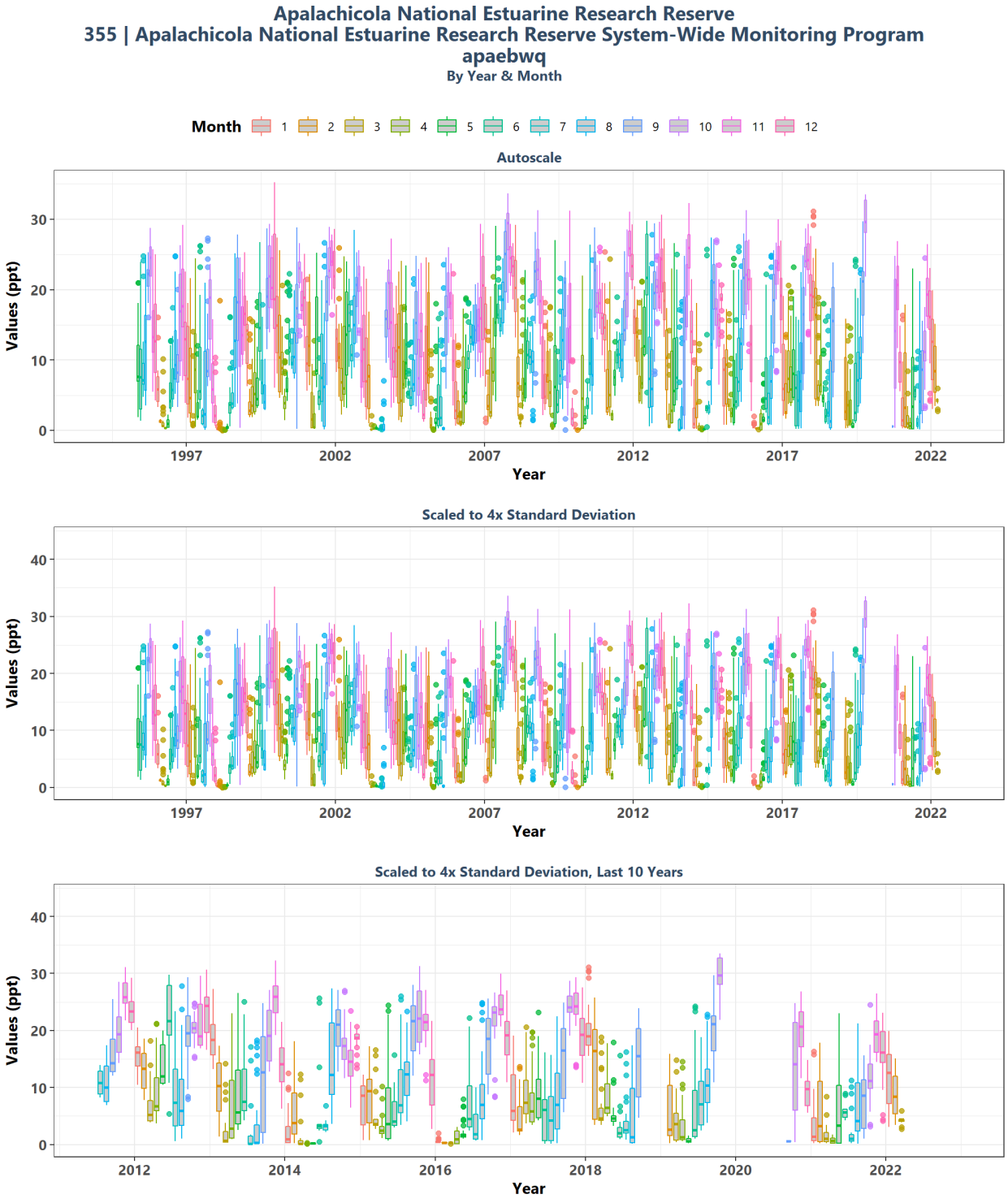

Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apadbwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation,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

Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apadbwq
By Month

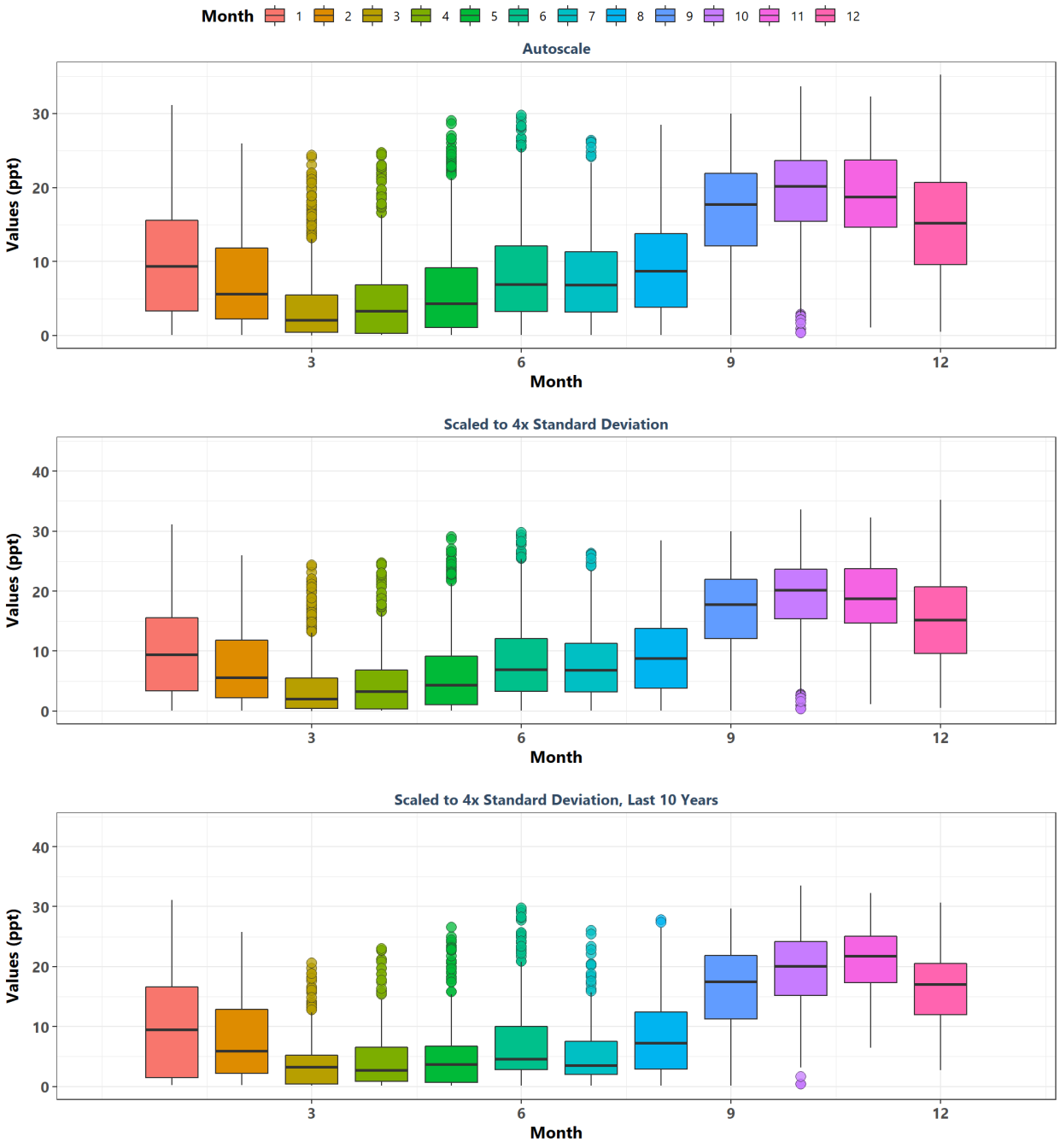


Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apaebwq
By Year

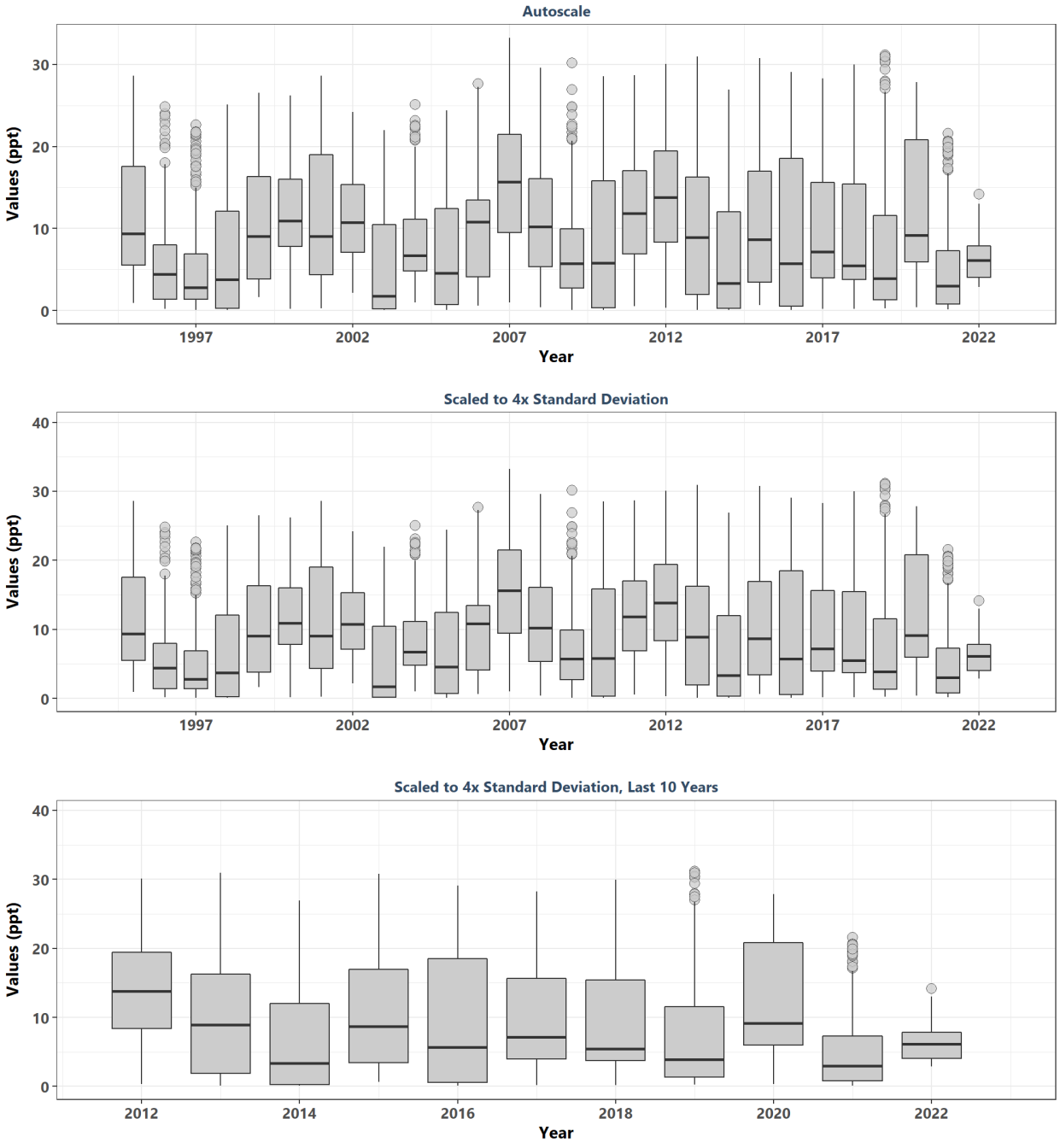


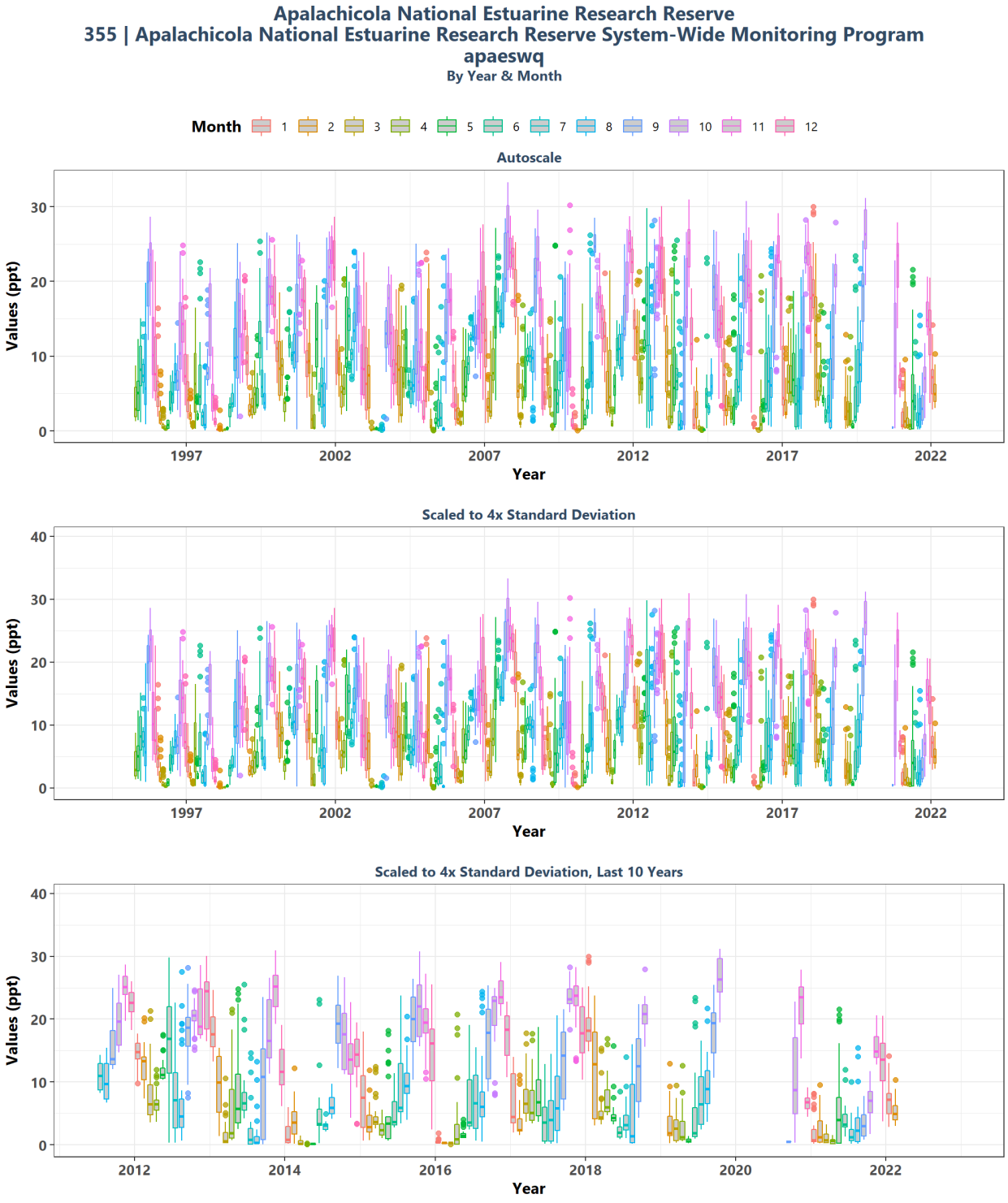

Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apaebwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation,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

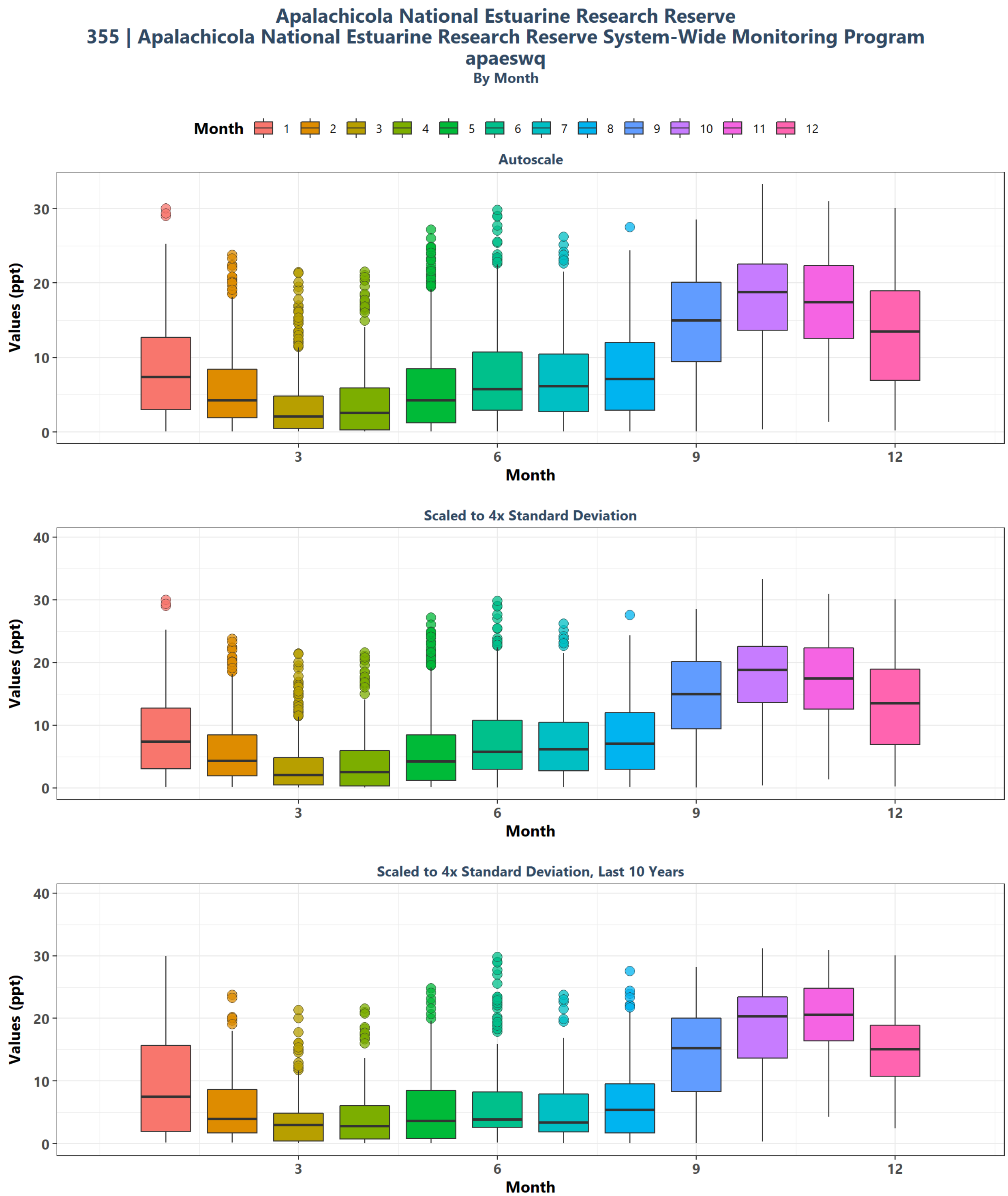
Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apaebwq
By Month



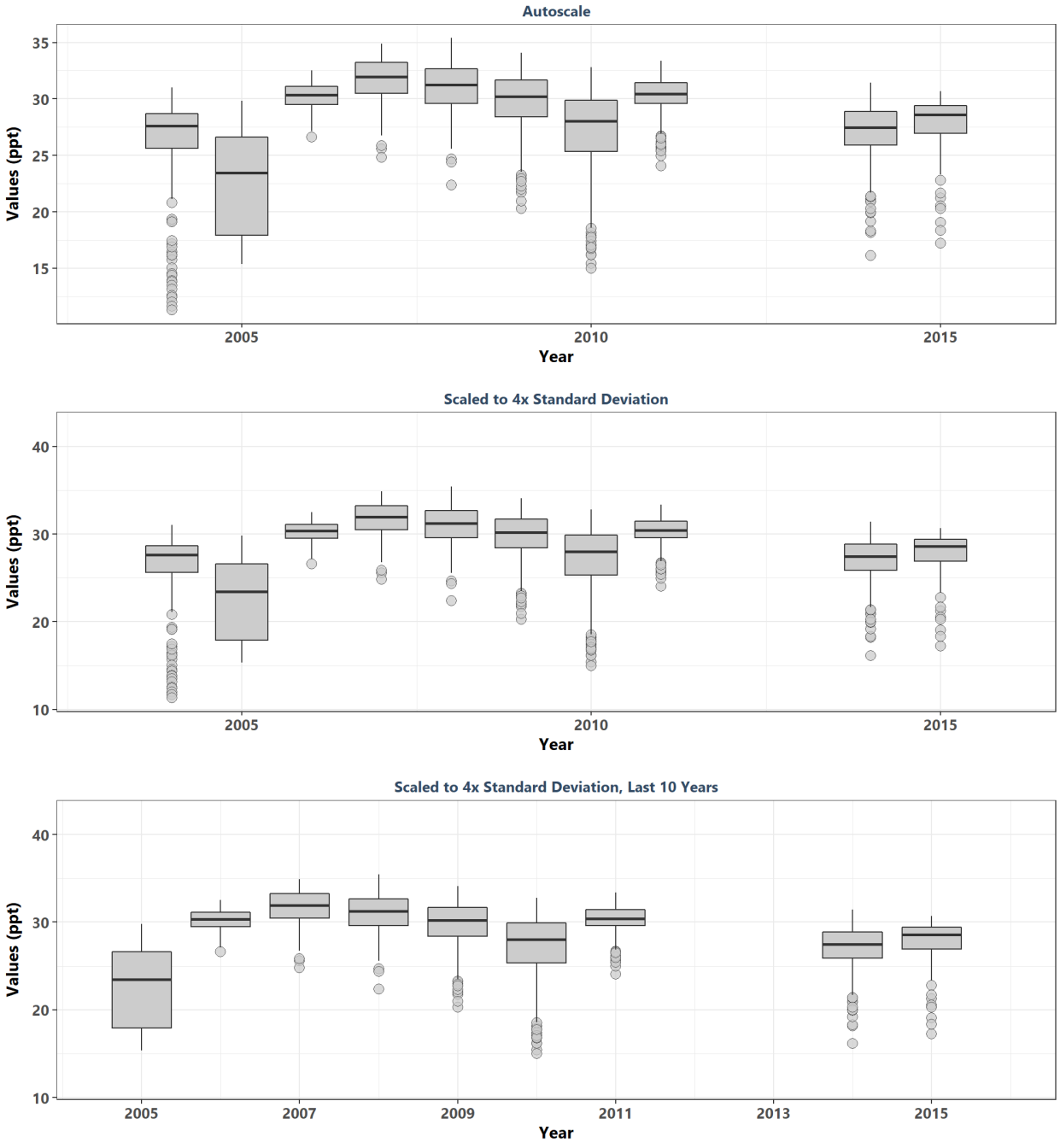
Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apaeswq
By Year

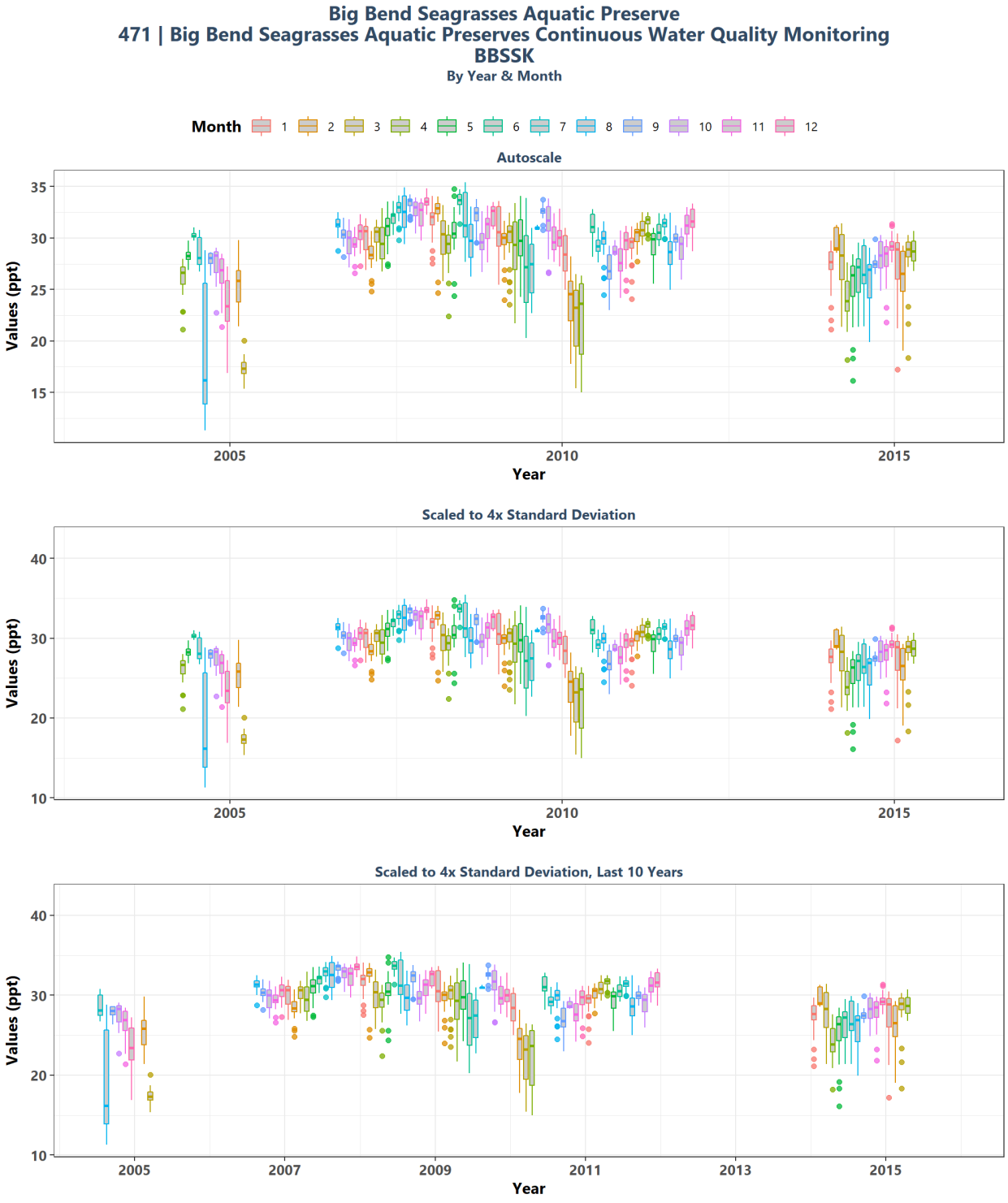


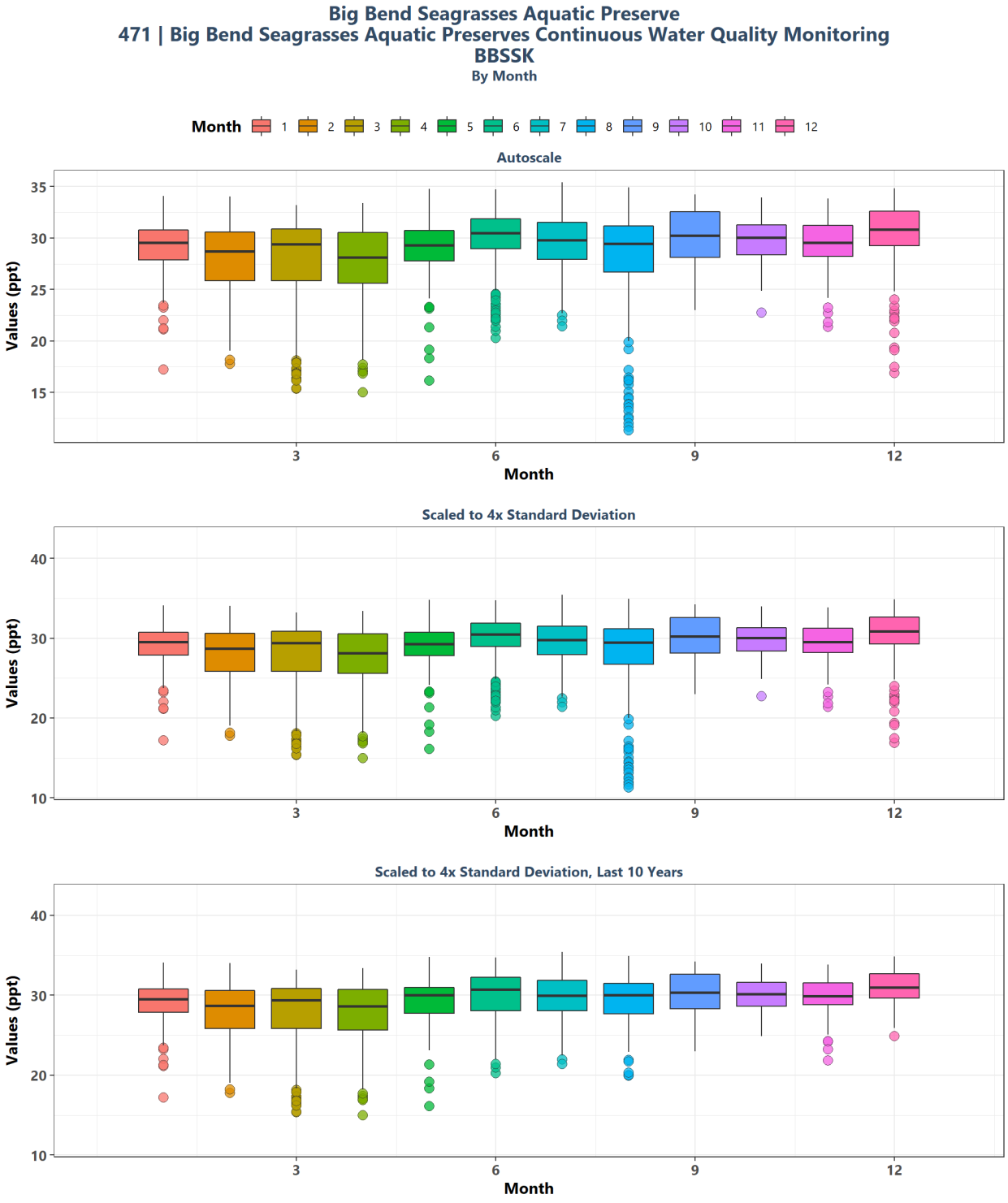

Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apaeswq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation,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


Apalachicola National Estuarine Research Reserve
355 | Apalachicola National Estuarine Research Reserve System-Wide Monitoring Program
apaeswq
By Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
0
10
20
30
3
6
9
12
Month
Scaled to 4x Standard Deviation
Values (ppt)
0
10
20
30
40
3
6
9
12
Month
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
0
10
20
30
40
3
6
9
12
Month

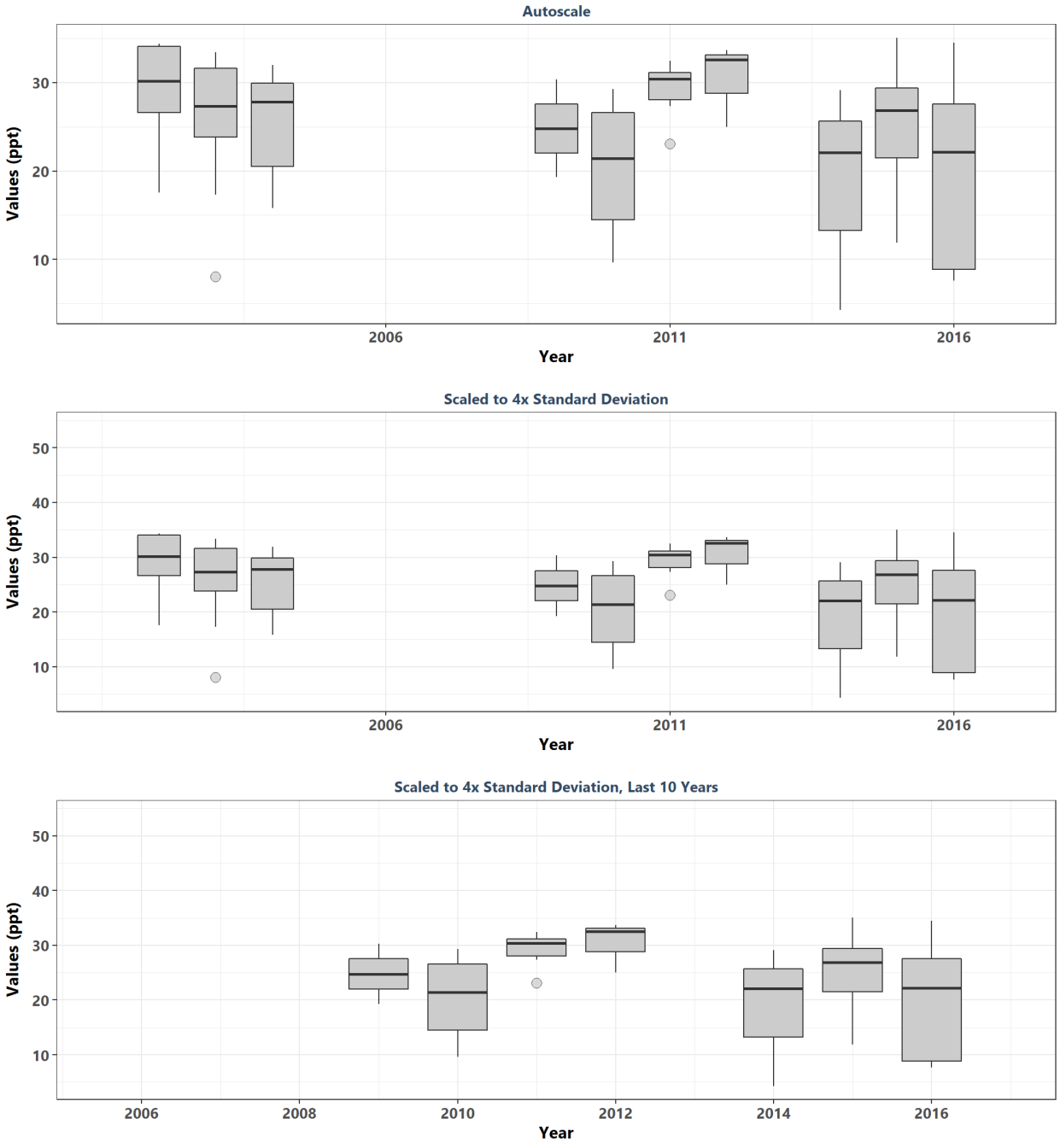
Big Bend Seagrasses Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring
BBSSK
By Year

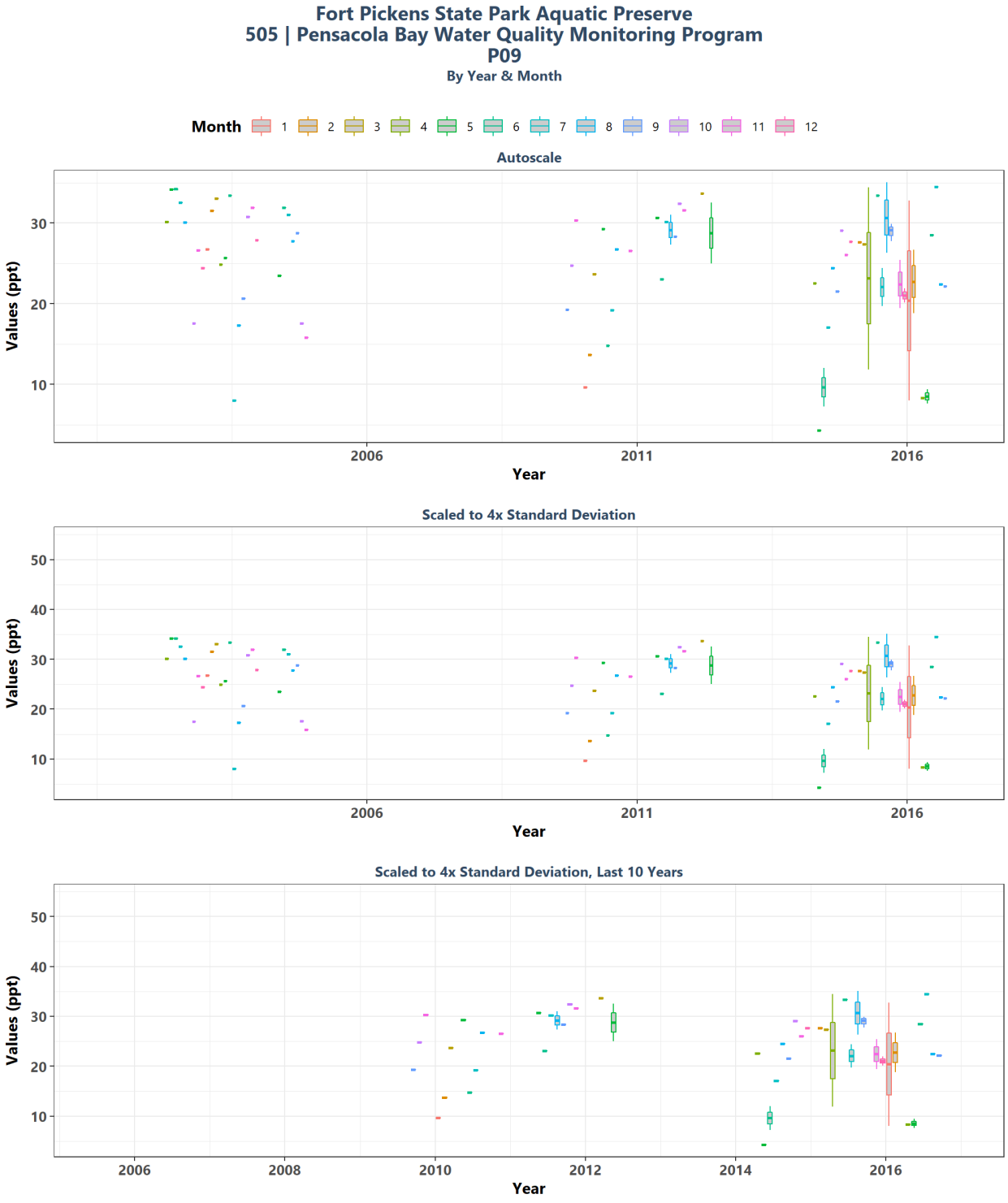



Big Bend Seagrasses Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring
BBSSK
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
Scaled to 4x Standard Deviation
Values (ppt)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year

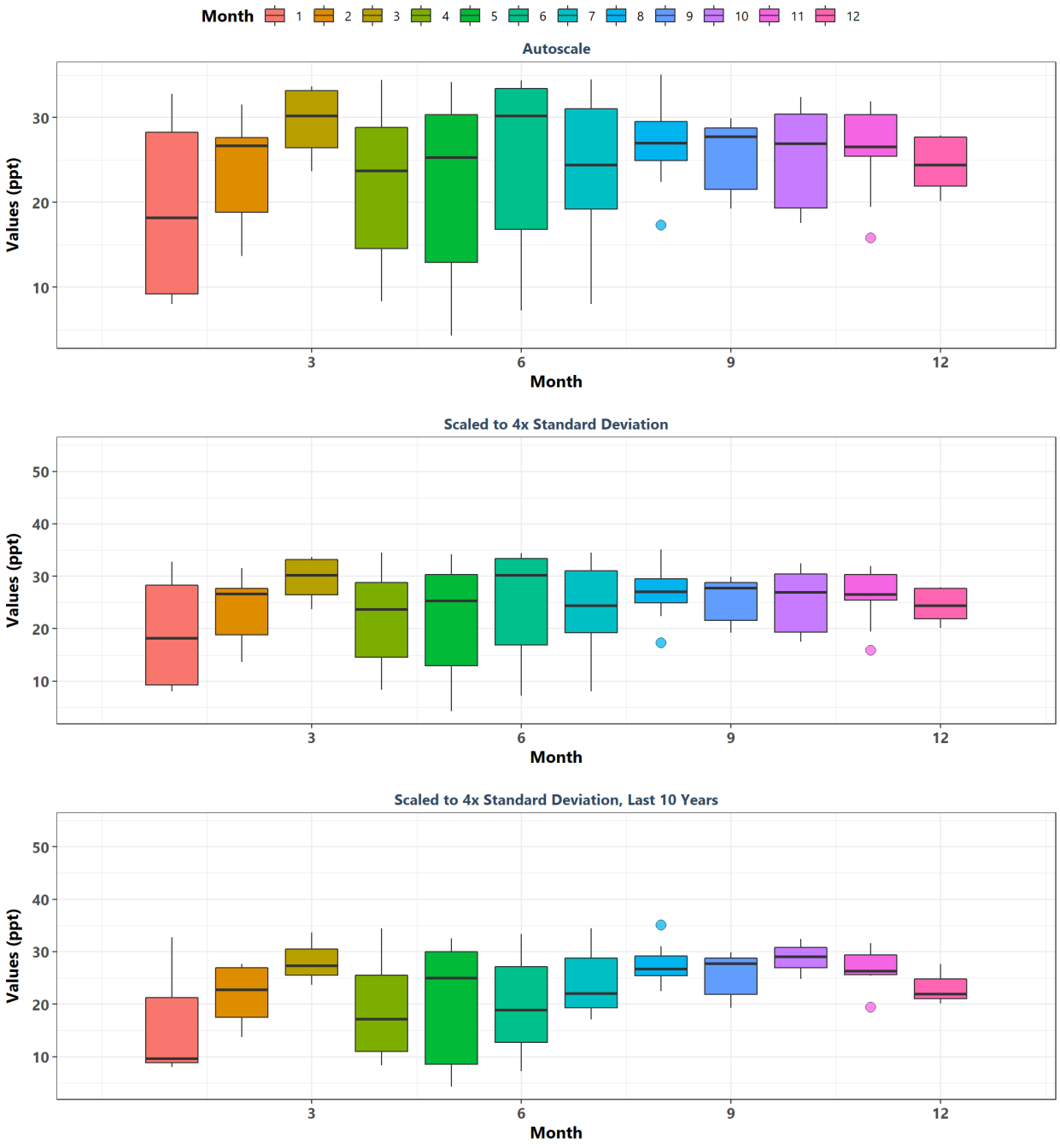

Big Bend Seagrasses Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring
BBSSK
By Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
35
30
25
20
15
3
6
9
12
Month
Scaled to 4x Standard Deviation
Values (ppt)
40
30
20
10
3
6
9
12
Month
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
40
30
20
10
3
6
9
12
Month

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

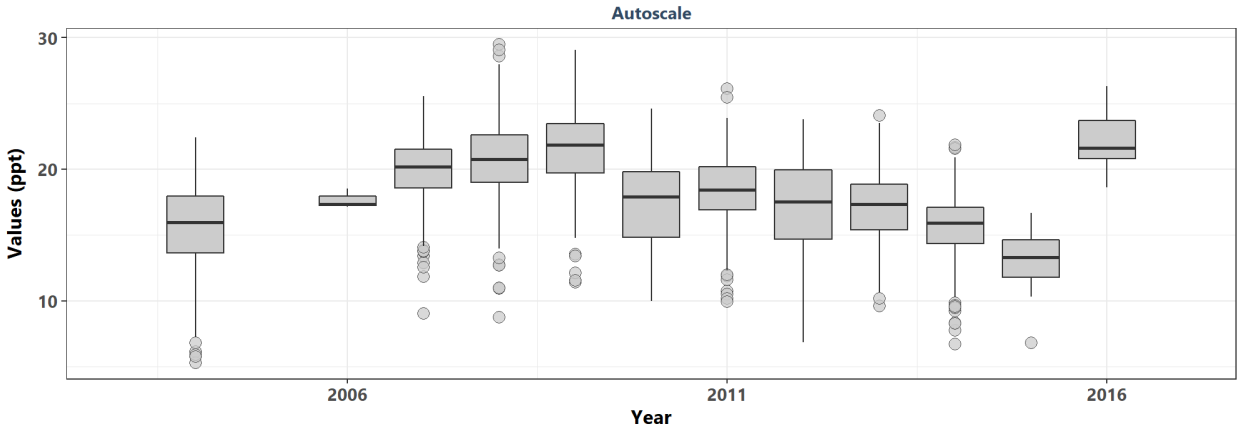


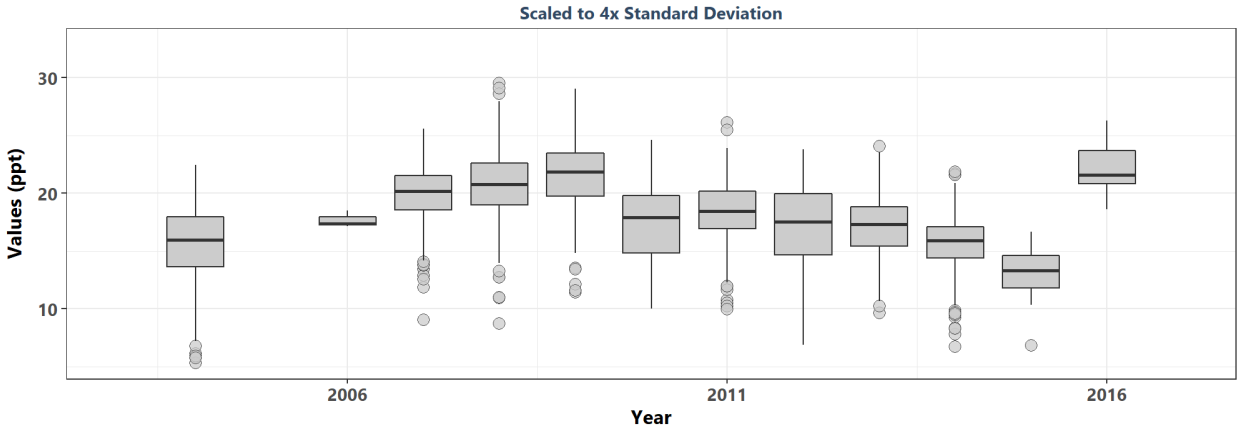

Fort Pickens State Park Aquatic Preserve
505 | Pensacola Bay Water Quality Monitoring Program
P09
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
Year
2006
2011
2016
Scaled to 4x Standard Deviation
Values (ppt)
Year
2006
2011
2016
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
Year
2006
2008
2010
2012
2014
2016

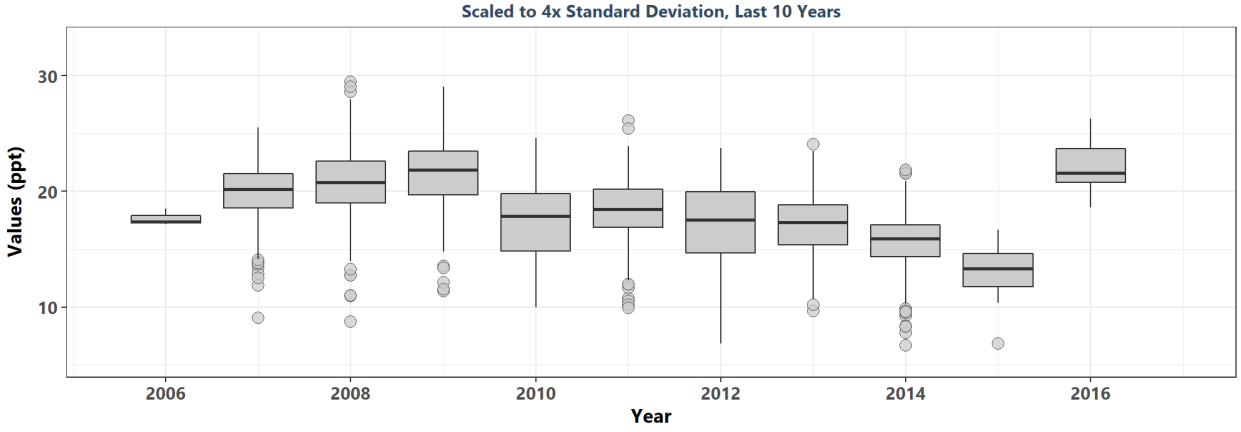
Fort Pickens State Park Aquatic Preserve
505 | Pensacola Bay Water Quality Monitoring Program
P09
By Month

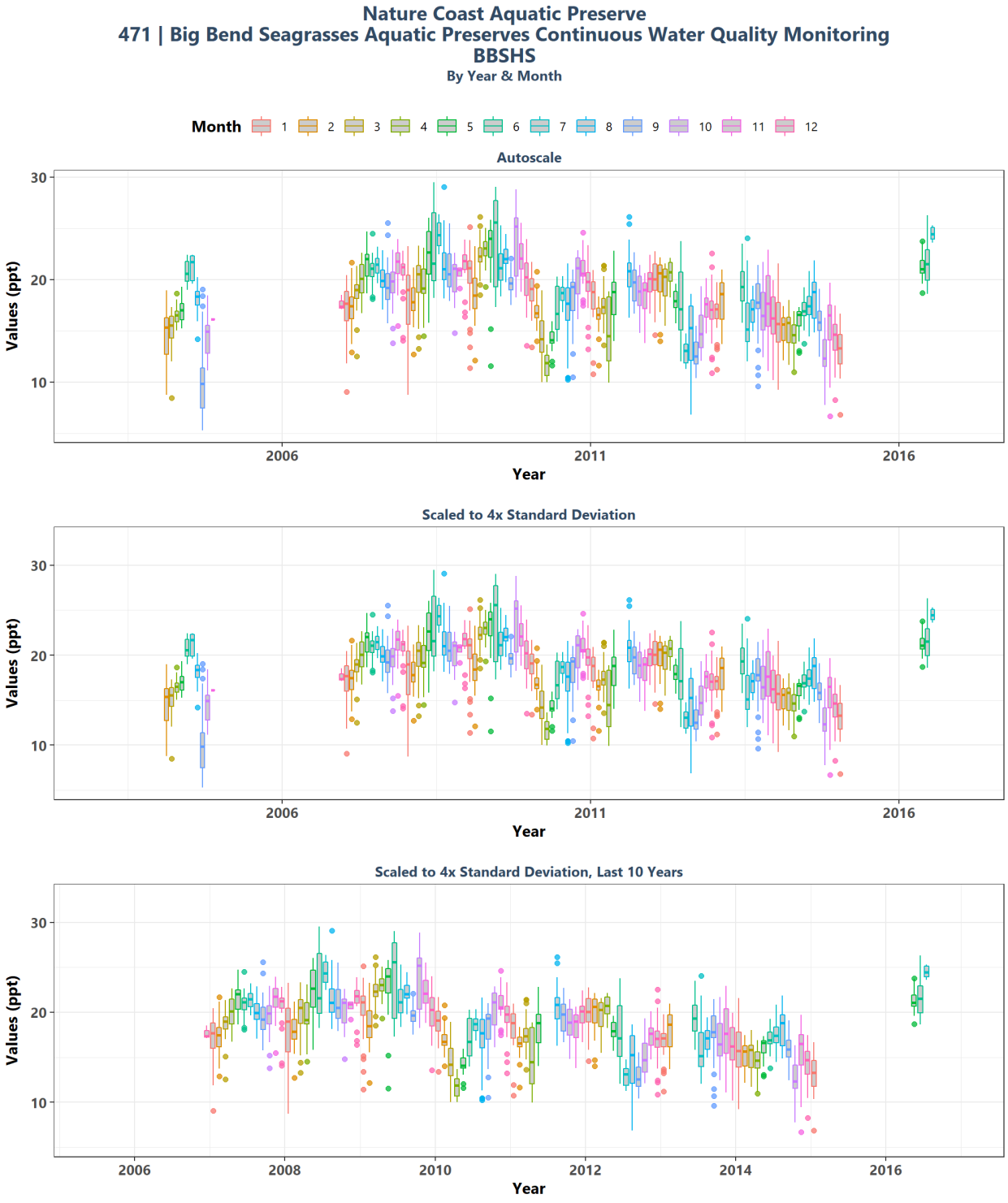


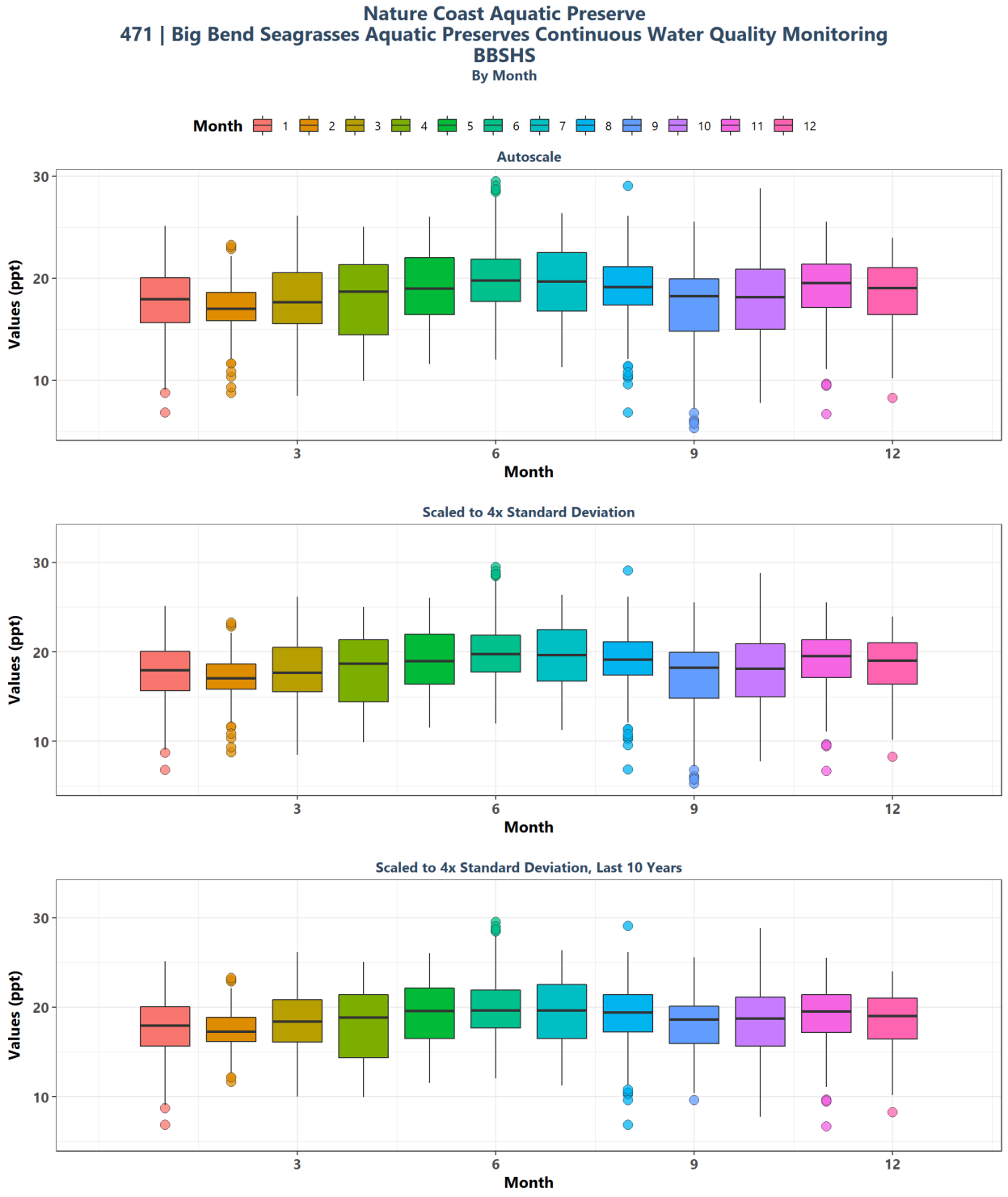
Nature Coast Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring
BBSHS
By Year








Nature Coast Aquatic Preserve
471 | Big Bend Seagrasses Aquatic Preserves Continuous Water Quality Monitoring
BBSHS
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (ppt)
30
20
10
2006
2011
2016
Year
Scaled to 4x Standard Deviation
Values (ppt)
30
20
10
2006
2011
2016
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (ppt)
30
20
10
2006
2008
2010
2012
2014
2016
Year




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