SEACAR Continuous Water Quality Analysis: SW Region for Dissolved Oxygen Saturation

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.

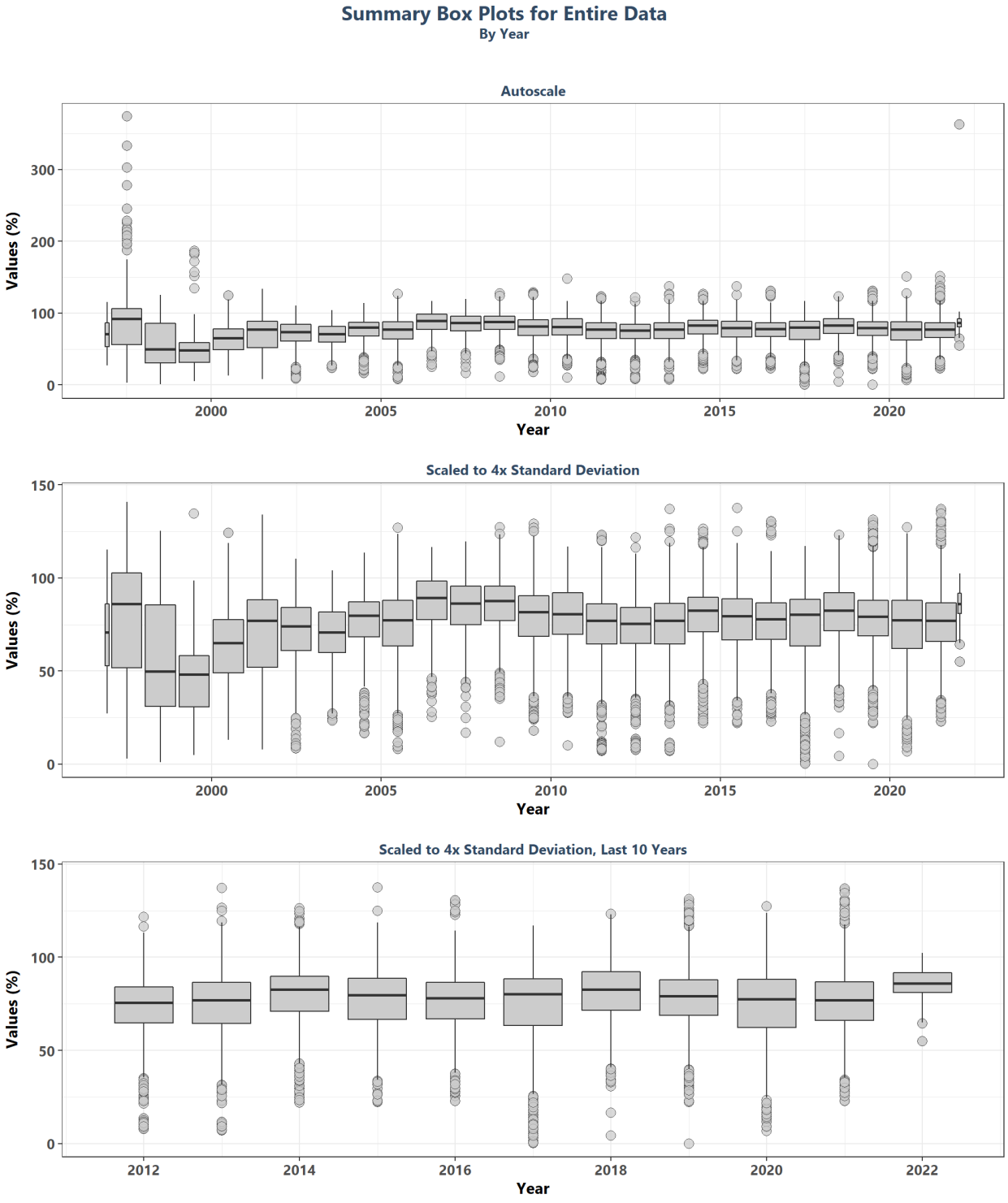
```
p1 <- ggplot(data=data[data$Include==TRUE, ],
             aes(x=YearMonthDec, y=ResultValue,
                 group=YearMonth, color=as.factor(Month))) +
   geom_boxplot(fill="#cccccc", outlier.size=1.5, outlier.alpha=0.75) +
   labs(subtitle="Autoscale", x="Year",
        y=paste0("Values (", unit,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

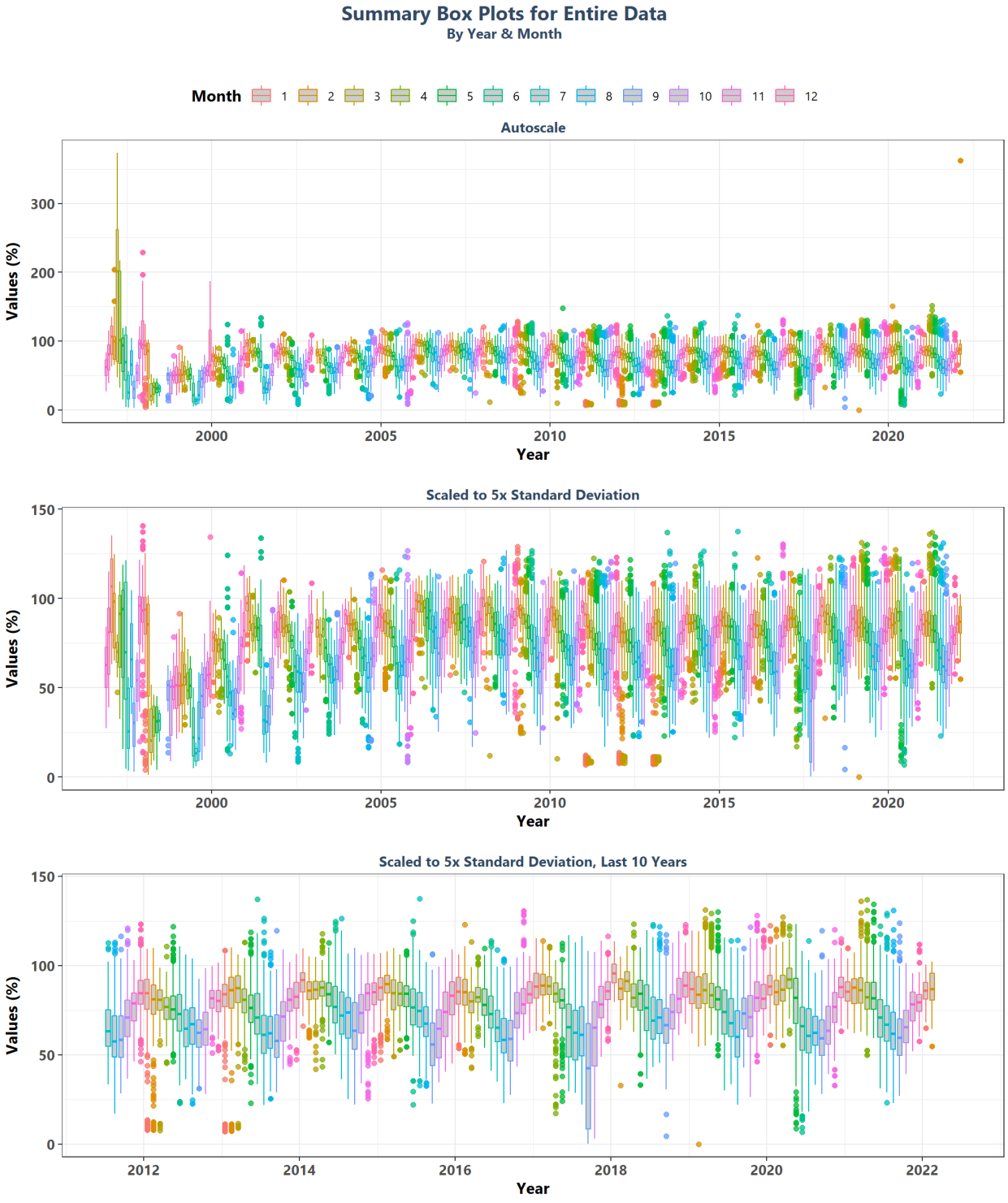
```
p3 <- ggplot(data=data[data$Include==TRUE, ],
             aes(x=YearMonthDec, y=ResultValue,
                 group=YearMonth, color=as.factor(Month))) +
   geom_boxplot(fill="#cccccc", outlier.size=1.5, outlier.alpha=0.75) +
   labs(subtitle="Scaled to 5x Standard Deviation, Last 10 Years",
  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

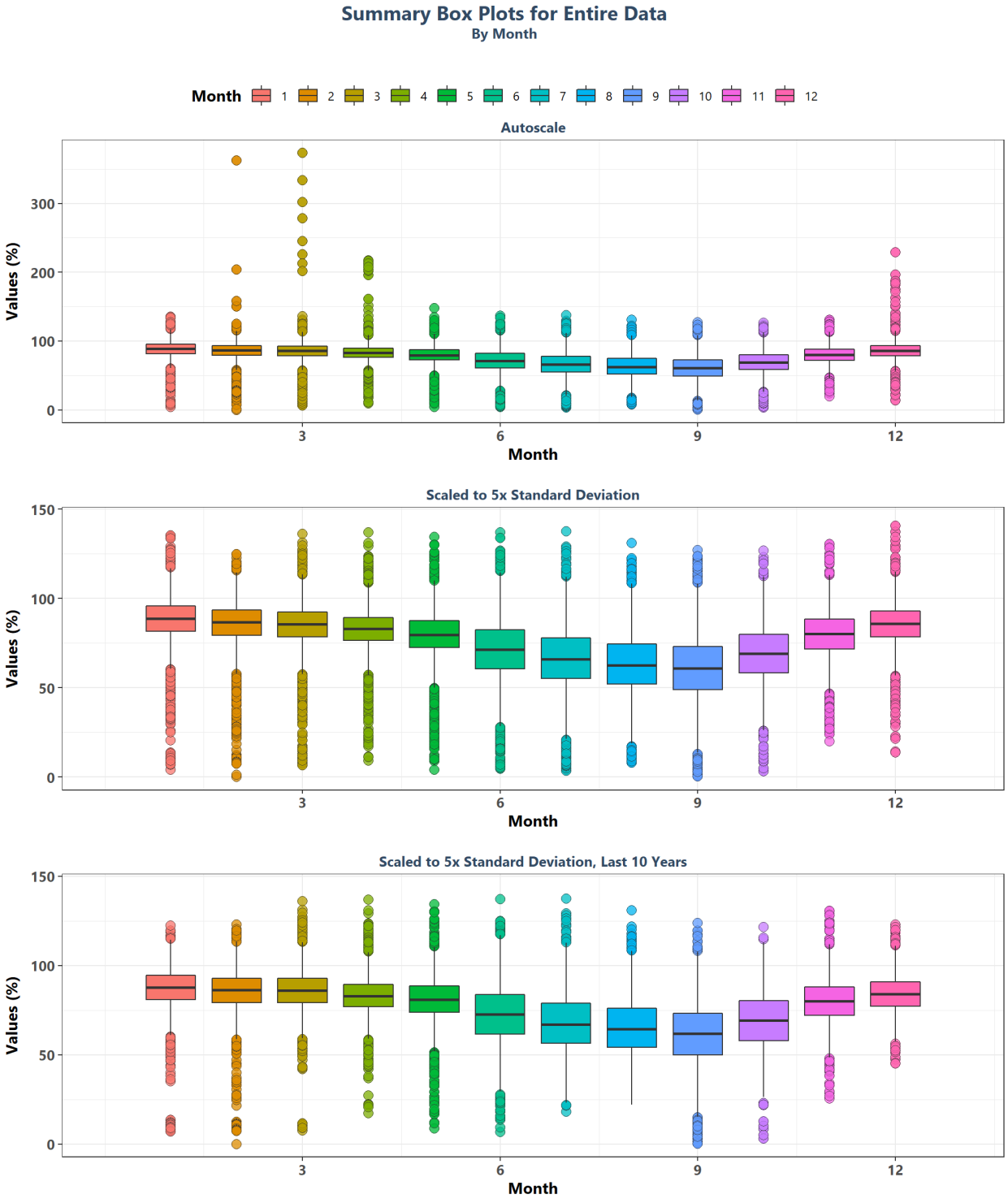
The following box plots are grouped by month with fill color being related to the month. This is designed to view potential seasonal trends.

```
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 (%)
300
200
100
0
2000
2005
2010
2015
2020
Year
Scaled to 4x Standard Deviation
Values (%)
150
100
50
0
2000
2005
2010
2015
2020
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (%)
150
100
50
0
2012
2014
2016
2018
2020
2022
Year

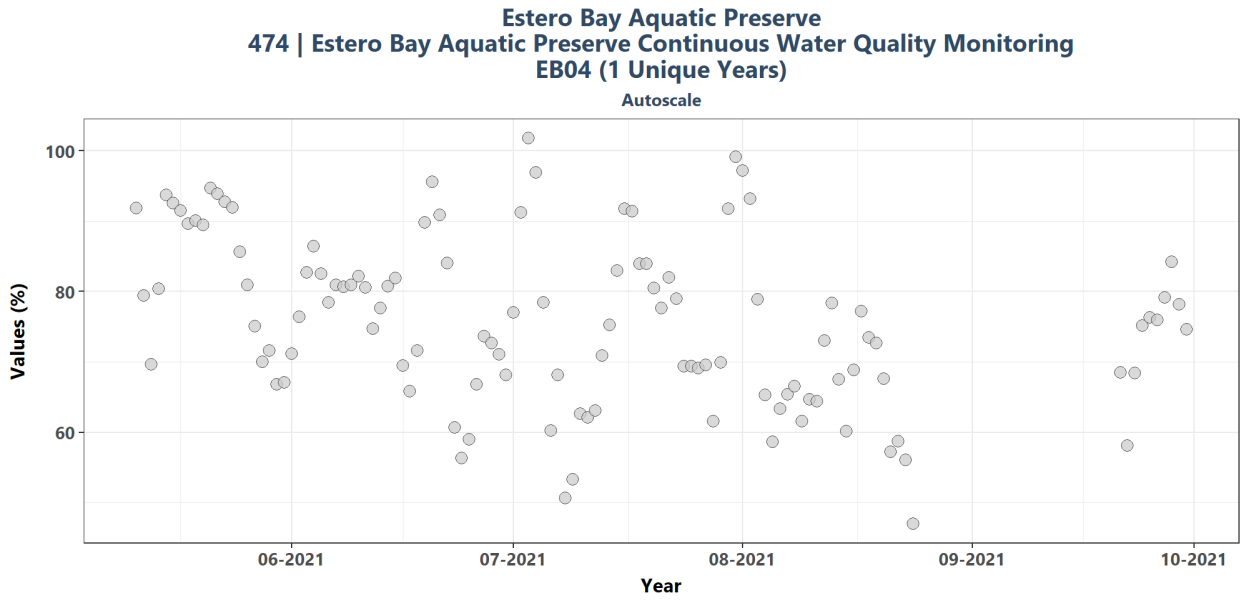

Summary Box Plots for Entire Data
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
300
200
100
0
2000
2005
2010
2015
2020
Year
Scaled to 5x Standard Deviation
Values (%)
150
100
50
0
2000
2005
2010
2015
2020
Year
Scaled to 5x Standard Deviation, Last 10 Years
Values (%)
150
100
50
0
2012
2014
2016
2018
2020
2022
Year

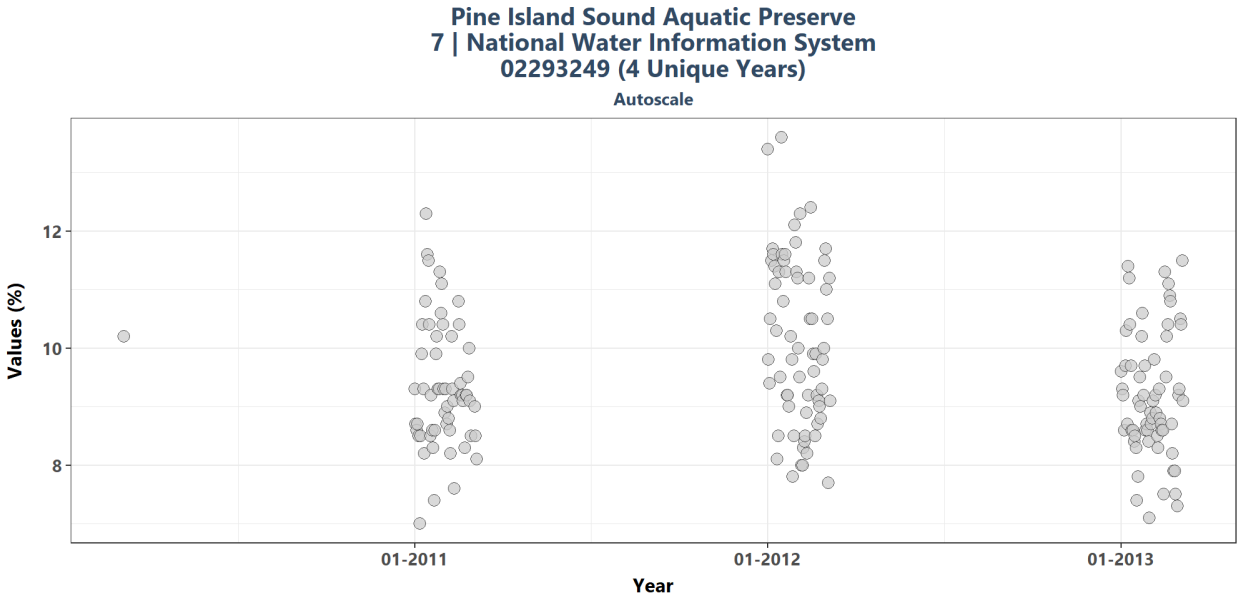


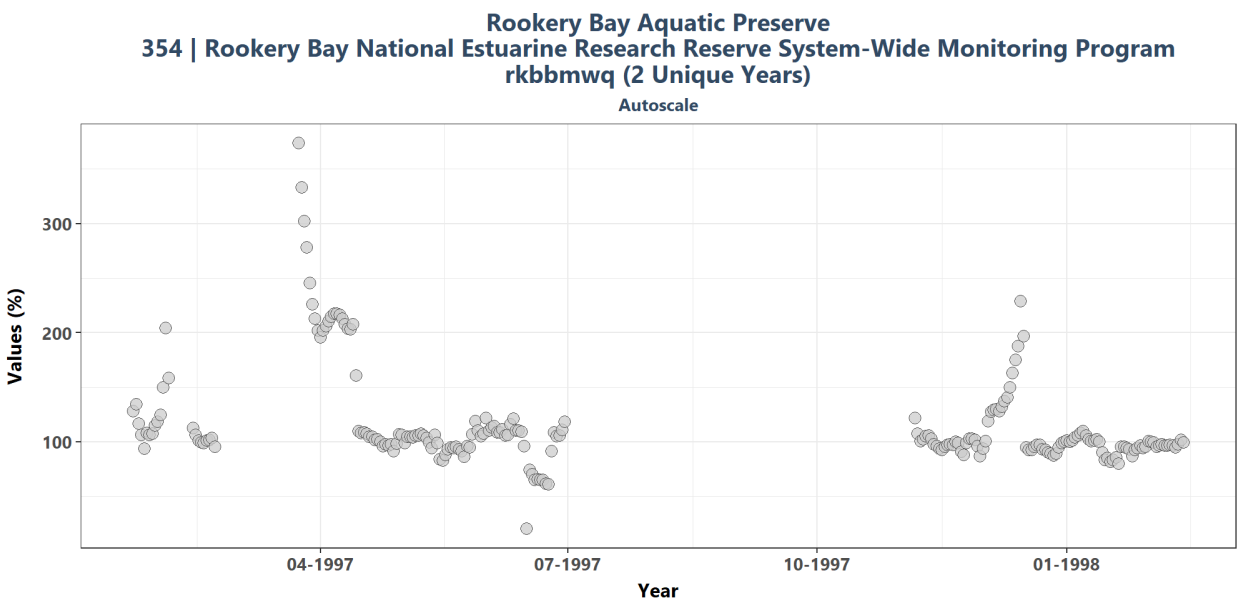
Appendix II: Excluded Monitoring Locations

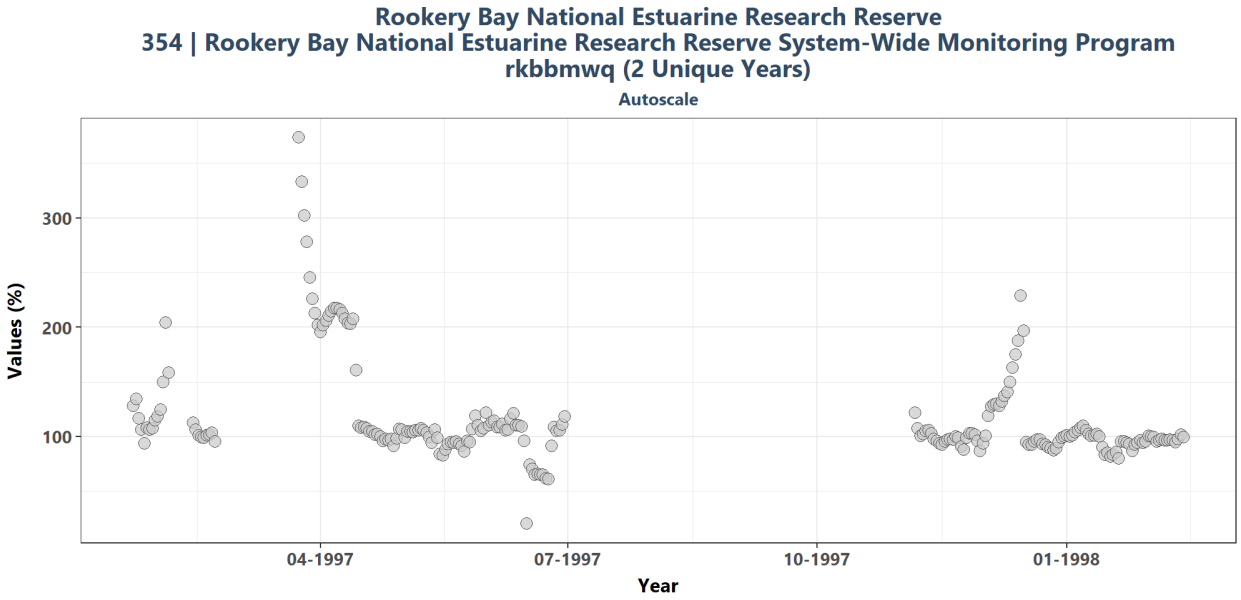
Scatter plots of data values are created for monitoring locations that have fewer than 5 separate years of data entries.

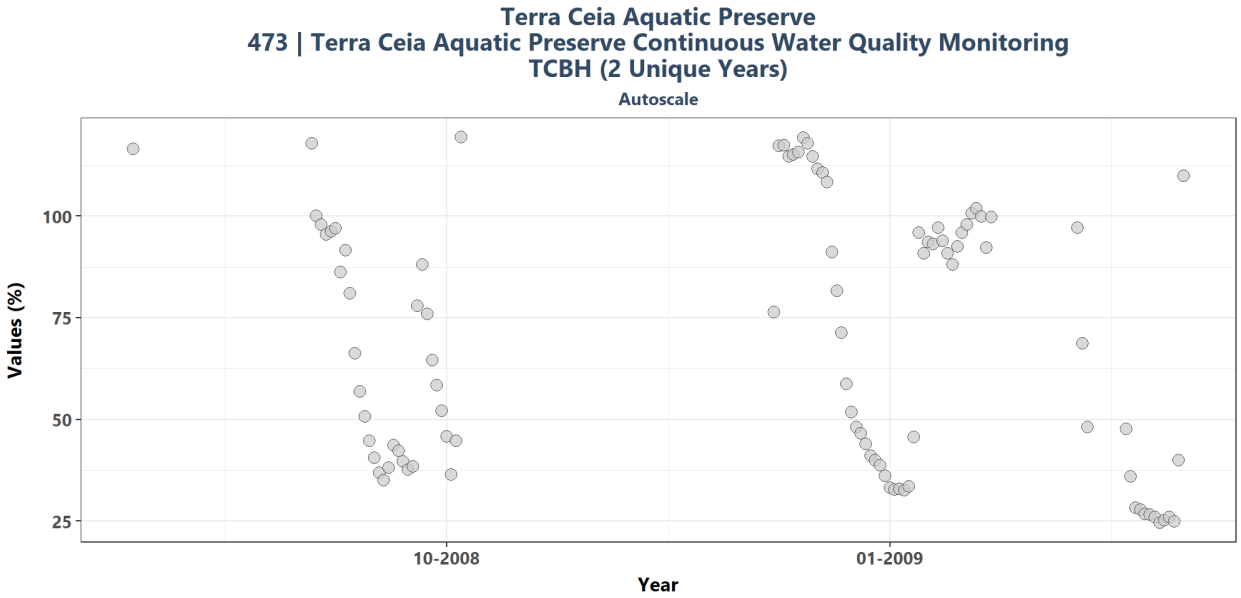
```
Mon_Exclude <- Mon_Summ[Mon_Summ$N_Years<5 & Mon_Summ$N_Years>0,]
Mon_Exclude <- Mon_Exclude[order(Mon_Exclude$MonitoringID),]
z=nrow(Mon_Exclude)

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

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




Pine Island Sound Aquatic Preserve
7 | National Water Information System
02293249 (4 Unique Years)
Autoscale
Values (%)
12
10
8
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 (%)
300
200
100
04-1997
07-1997
10-1997
01-1998
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

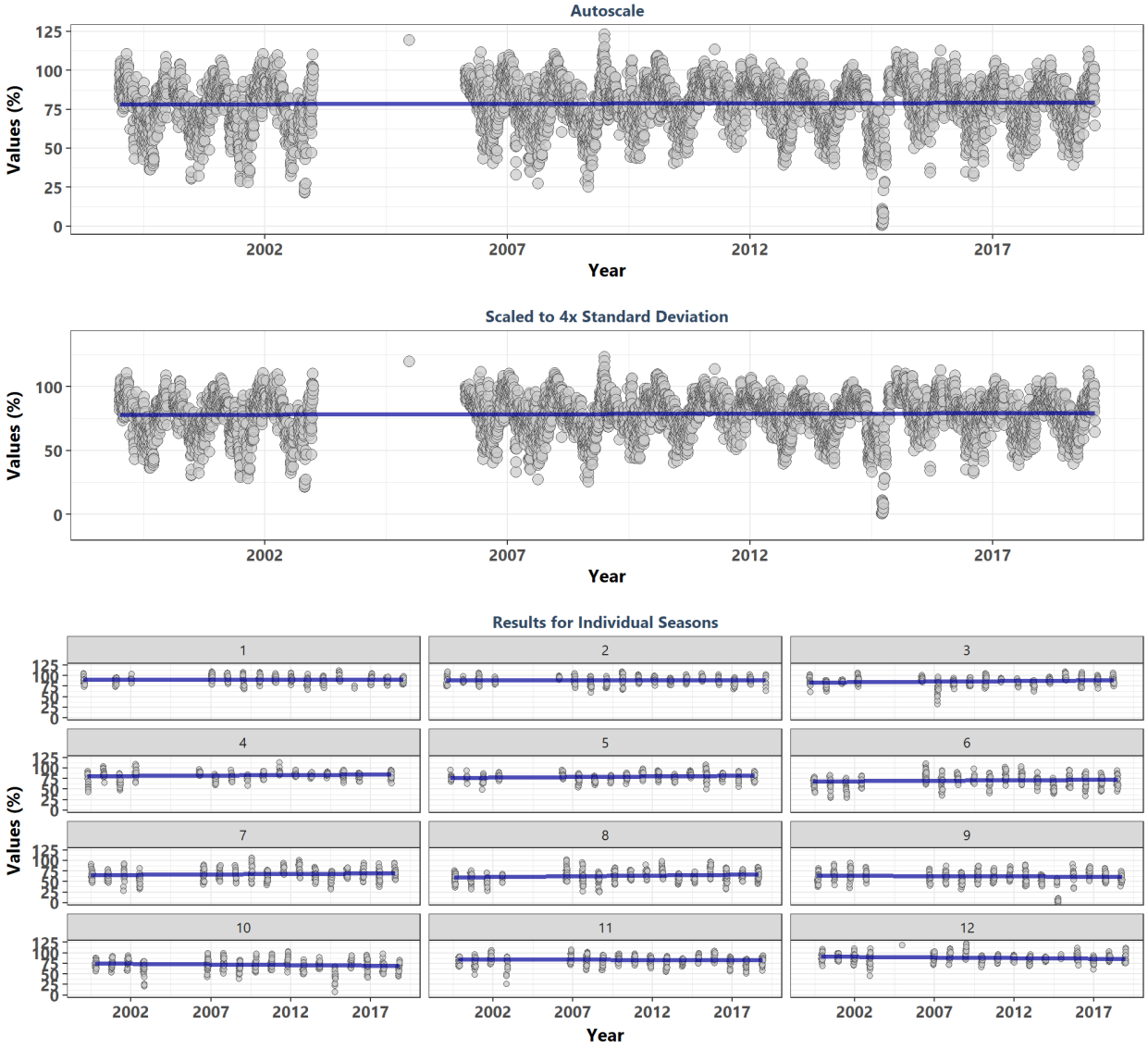
5. Set the plot type as a point plot with the size of the points
6. Add the linear trend
7. Create the title, x-axis, y-axis, and color fill labels
8. Set the y and x limits
9. Make the axis labels bold
10. Plot the arrangement as a set of panels

```
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) +
```

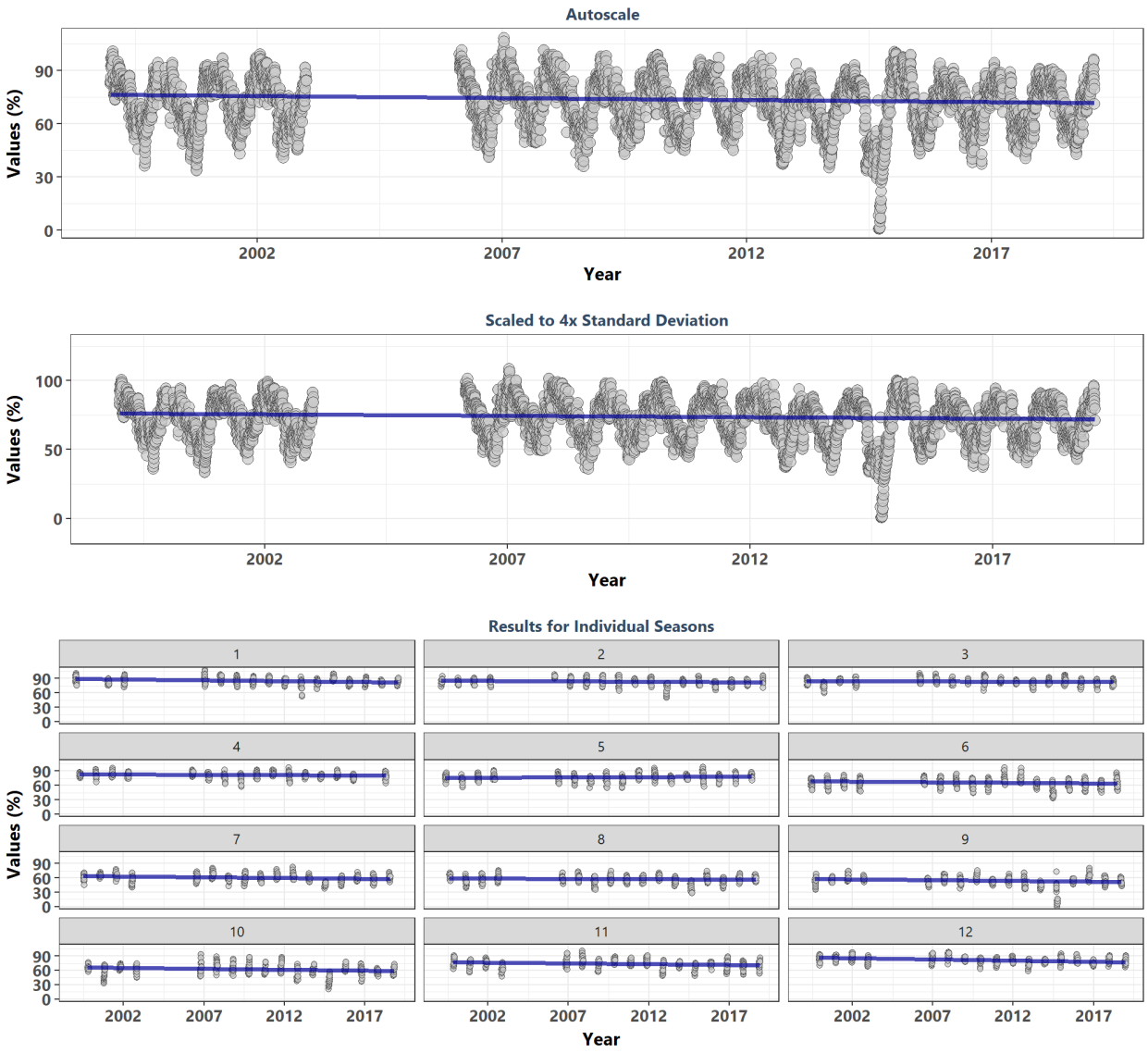
```
            scale_x_continuous(breaks = xbrks,
                               labels = xlabs) +
            labs(y = paste0("Values (", unit, ")"), x = "Year", subtitle = "Results for Individual Seas
            facet_wrap(~Season, ncol = 3) +
            plot_theme

      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)
   }
}
```

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



Season	N	Median	tau	Slope	Int.	z	p__z	chi__sq	p__chi__sq	Trend
All	5605	79.79	0.0250	0.0658	77.9273	2.7	0.0075	94.8	0	1
1	444	90.31	-0.0279	-0.0613	91.1713	-0.9	0.3793	NA	NA	-1
2	437	89.19	-0.0065	-0.0131	89.3762	-0.2	0.8391	NA	NA	-1
3	477	86.58	0.1437	0.3249	82.3529	4.7	0.0000	NA	NA	1
4	431	83.49	0.0953	0.2328	80.4628	3.0	0.0030	NA	NA	1
5	469	79.42	0.1433	0.3269	75.1676	4.6	0.0000	NA	NA	1
6	473	70.56	0.0578	0.2171	67.5181	1.9	0.0597	NA	NA	1
7	483	67.63	0.0725	0.2518	64.3565	2.4	0.0170	NA	NA	1
8	465	63.88	0.1017	0.3250	59.6521	3.3	0.0010	NA	NA	1
9	446	62.86	-0.0285	-0.0974	64.1271	-0.9	0.3677	NA	NA	-1
10	496	71.80	-0.0850	-0.2761	75.3918	-2.8	0.0046	NA	NA	-1
11	486	83.37	-0.0179	-0.0500	84.0188	-0.6	0.5558	NA	NA	-1
12	498	88.43	-0.1354	-0.3163	92.5419	-4.5	0.0000	NA	NA	-1

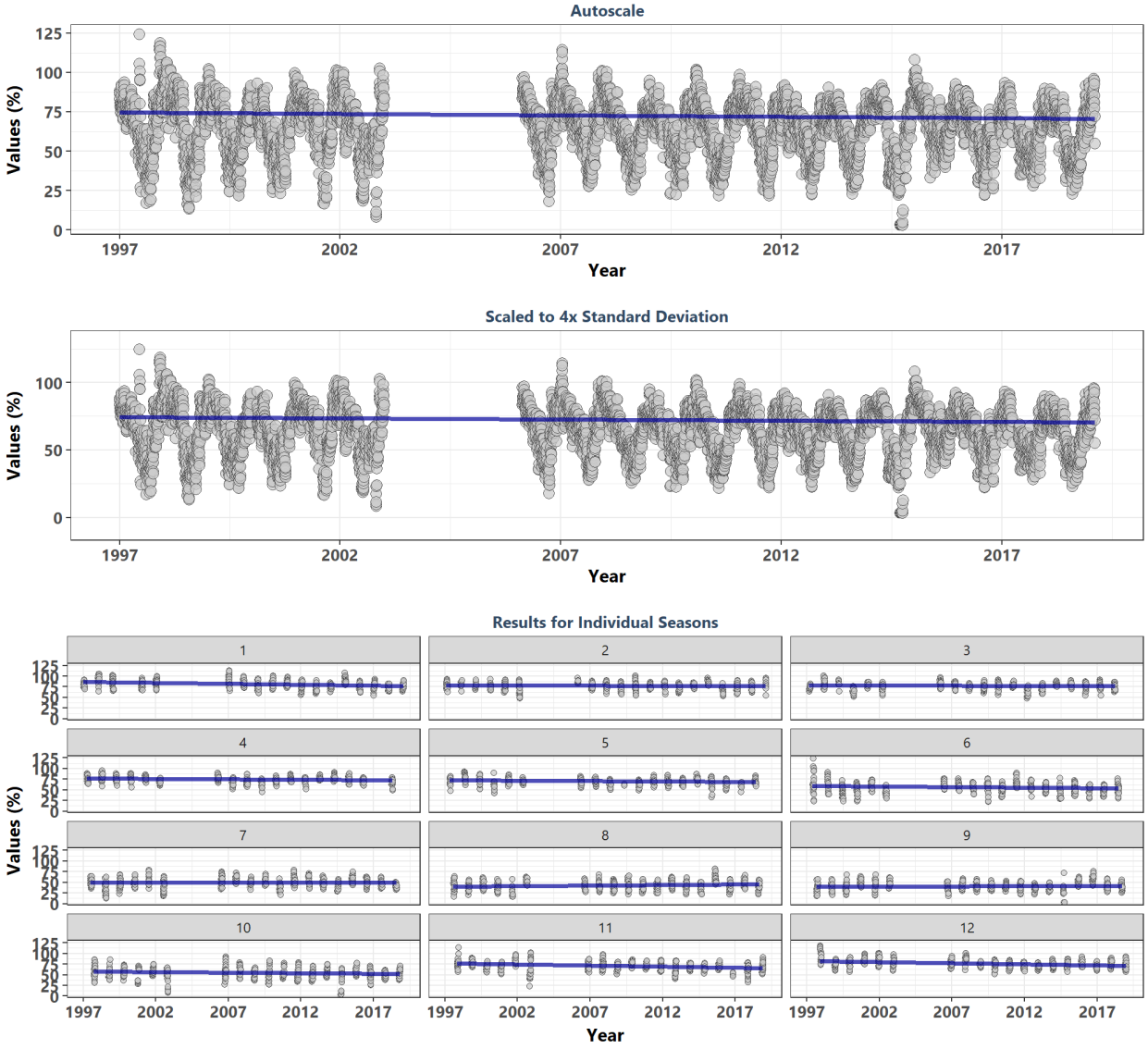
[a] $p < 0.00005$ appear as 0 due to rounding



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	5933	73.94	-0.1084	-0.2163	76.6448	-12.5	0.0000	84.2	0	-1
1	488	85.07	-0.1504	-0.3007	89.2816	-5.0	0.0000	NA	NA	-1
2	445	83.42	-0.1363	-0.2364	86.7270	-4.3	0.0000	NA	NA	-1
3	486	83.60	-0.0385	-0.0701	84.5108	-1.3	0.2033	NA	NA	-1
4	462	81.69	-0.1085	-0.1499	83.6365	-3.5	0.0005	NA	NA	-1
5	502	77.21	0.0970	0.1675	75.0274	3.3	0.0011	NA	NA	1
6	499	66.01	-0.1013	-0.2472	69.2281	-3.4	0.0007	NA	NA	-1
7	508	60.58	-0.1323	-0.2735	64.1399	-4.5	0.0000	NA	NA	-1
8	512	57.46	-0.0672	-0.1310	59.1629	-2.3	0.0227	NA	NA	-1
9	501	54.36	-0.1305	-0.2767	57.9538	-4.4	0.0000	NA	NA	-1
10	518	61.69	-0.1508	-0.3853	66.7027	-5.1	0.0000	NA	NA	-1
11	502	74.11	-0.1327	-0.3194	78.2622	-4.5	0.0000	NA	NA	-1
12	510	81.42	-0.2487	-0.4705	87.5411	-8.4	0.0000	NA	NA	-1

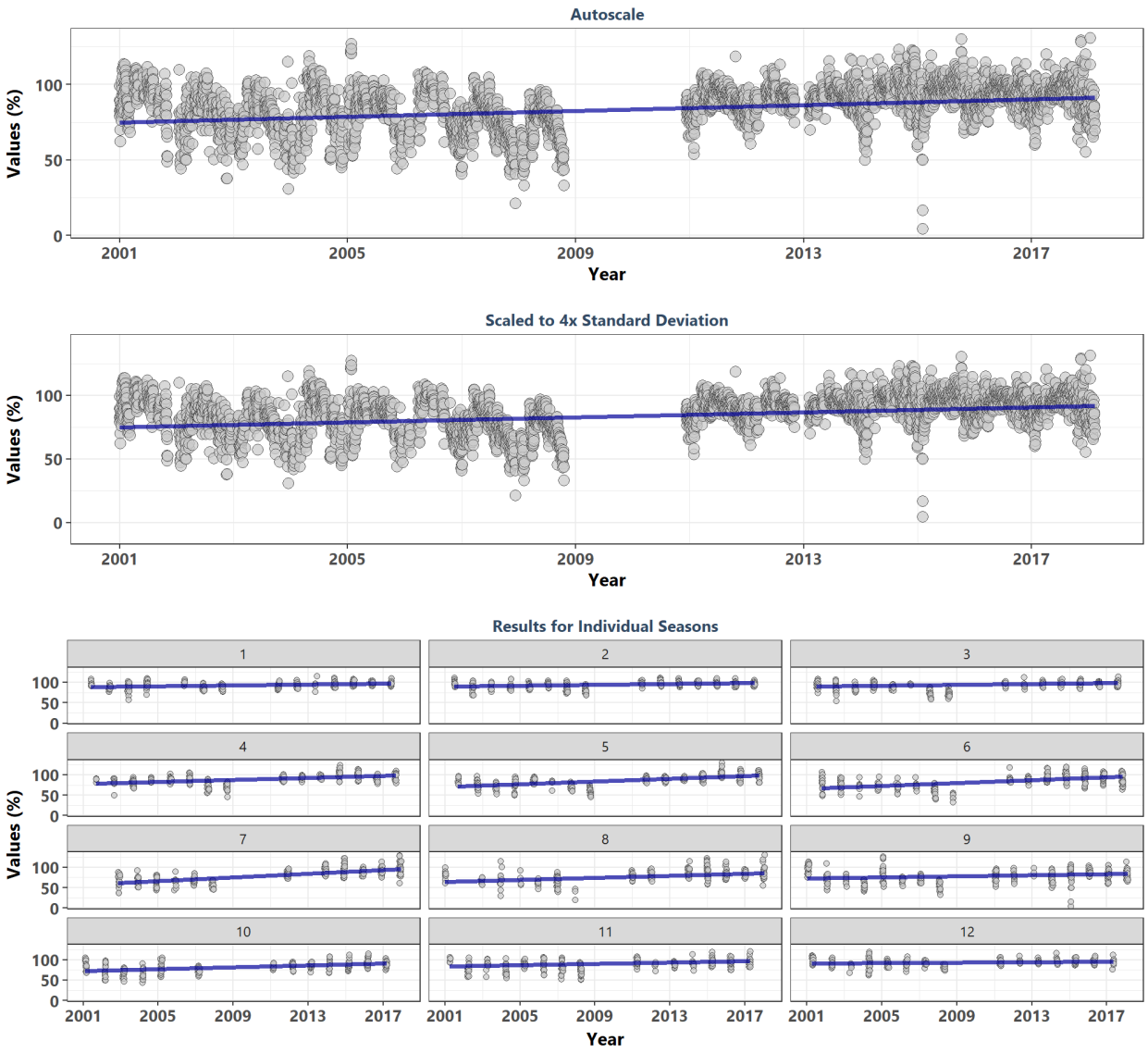
[a] $p < 0.00005$ appear as 0 due to rounding

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	6472	67.13	-0.0768	-0.1837	74.5536	-9.2	0.0000	124	0	-1
1	554	80.27	-0.1950	-0.4520	86.3671	-6.9	0.0000	NA	NA	-1
2	494	77.61	-0.0412	-0.0920	78.8039	-1.4	0.1709	NA	NA	-1
3	568	77.55	-0.0152	-0.0295	77.9077	-0.5	0.5880	NA	NA	-1
4	514	74.36	-0.0882	-0.1605	76.2817	-3.0	0.0028	NA	NA	-1
5	507	70.35	-0.0923	-0.1907	72.8254	-3.1	0.0019	NA	NA	-1
6	539	56.27	-0.1011	-0.3060	59.9471	-3.5	0.0004	NA	NA	-1
7	563	49.55	0.0093	0.0260	49.2354	0.3	0.7419	NA	NA	1
8	546	42.89	0.0869	0.2326	40.1005	3.0	0.0024	NA	NA	1
9	544	41.12	0.0237	0.0653	40.3406	0.8	0.4088	NA	NA	1
10	560	54.55	-0.0893	-0.2530	57.5877	-3.2	0.0015	NA	NA	-1
11	523	70.83	-0.1946	-0.4817	76.6104	-6.7	0.0000	NA	NA	-1
12	560	76.00	-0.2275	-0.5151	82.1780	-8.1	0.0000	NA	NA	-1

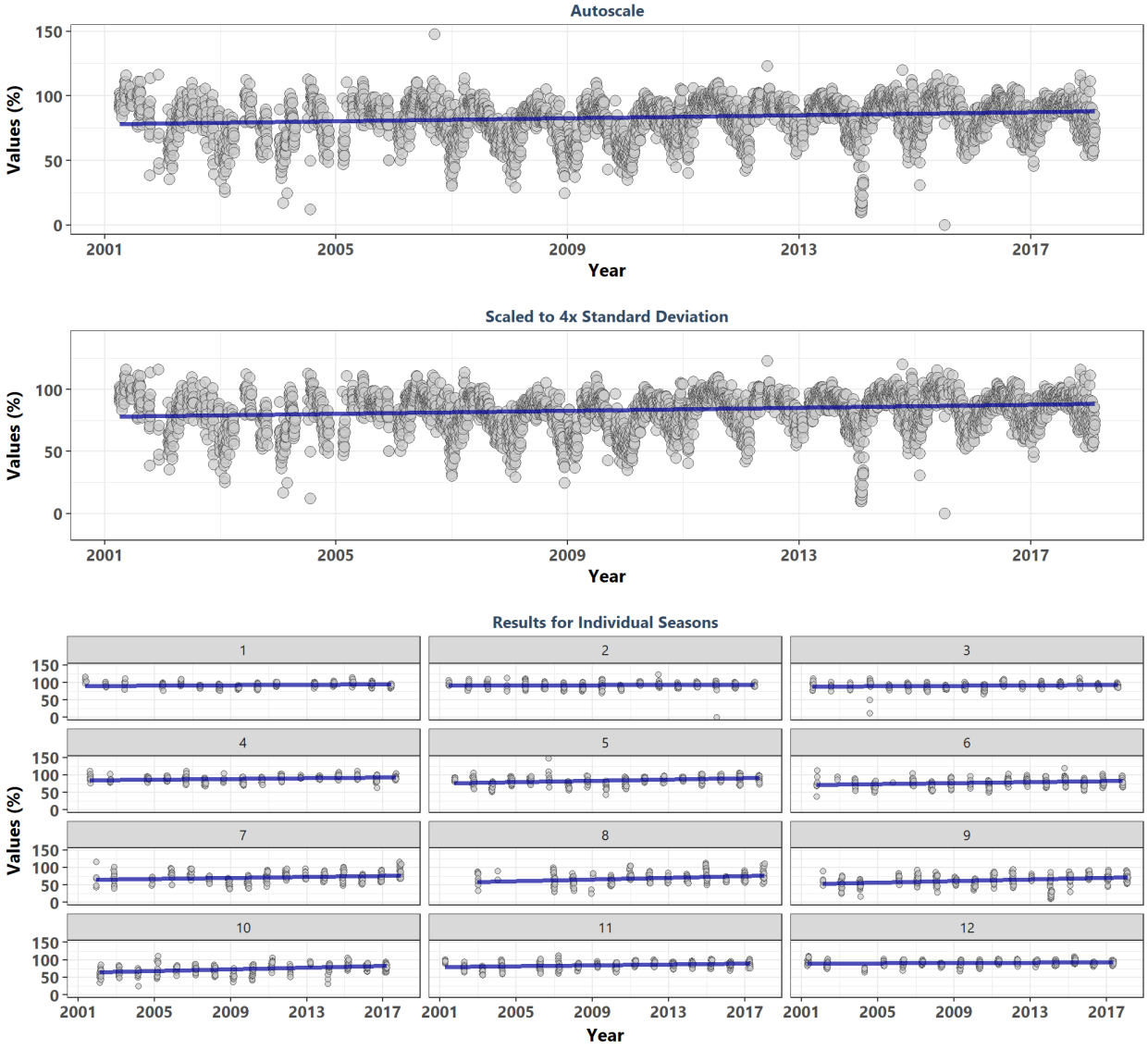
[a] $p < 0.00005$ appear as 0 due to rounding



Season	N	Median	tau	Slope	Int.	z	p__z	chi__sq	p__chi__sq	Trend
All	4169	89.13	0.2828	0.9682	74.9318	27.3	0.0000	130.2	0	1
1	363	94.74	0.2255	0.5440	88.7607	6.4	0.0000	NA	NA	1
2	363	95.24	0.2049	0.5015	89.7221	5.8	0.0000	NA	NA	1
3	361	93.83	0.2158	0.5801	89.1862	6.1	0.0000	NA	NA	1
4	376	90.85	0.3919	1.2376	77.2397	11.4	0.0000	NA	NA	1
5	402	88.62	0.4618	1.6093	70.9217	13.9	0.0000	NA	NA	1
6	328	86.51	0.3710	1.7804	65.1419	10.1	0.0000	NA	NA	1
7	316	82.50	0.4630	2.2805	57.4136	12.3	0.0000	NA	NA	1
8	298	80.73	0.2849	1.2740	64.1719	7.4	0.0000	NA	NA	1
9	372	78.93	0.1327	0.6519	72.4128	3.8	0.0001	NA	NA	1
10	323	83.77	0.3278	1.1143	72.6238	8.8	0.0000	NA	NA	1
11	325	88.98	0.2139	0.7978	83.3979	5.8	0.0000	NA	NA	1
12	342	92.63	0.0993	0.2375	90.9701	2.8	0.0060	NA	NA	1

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

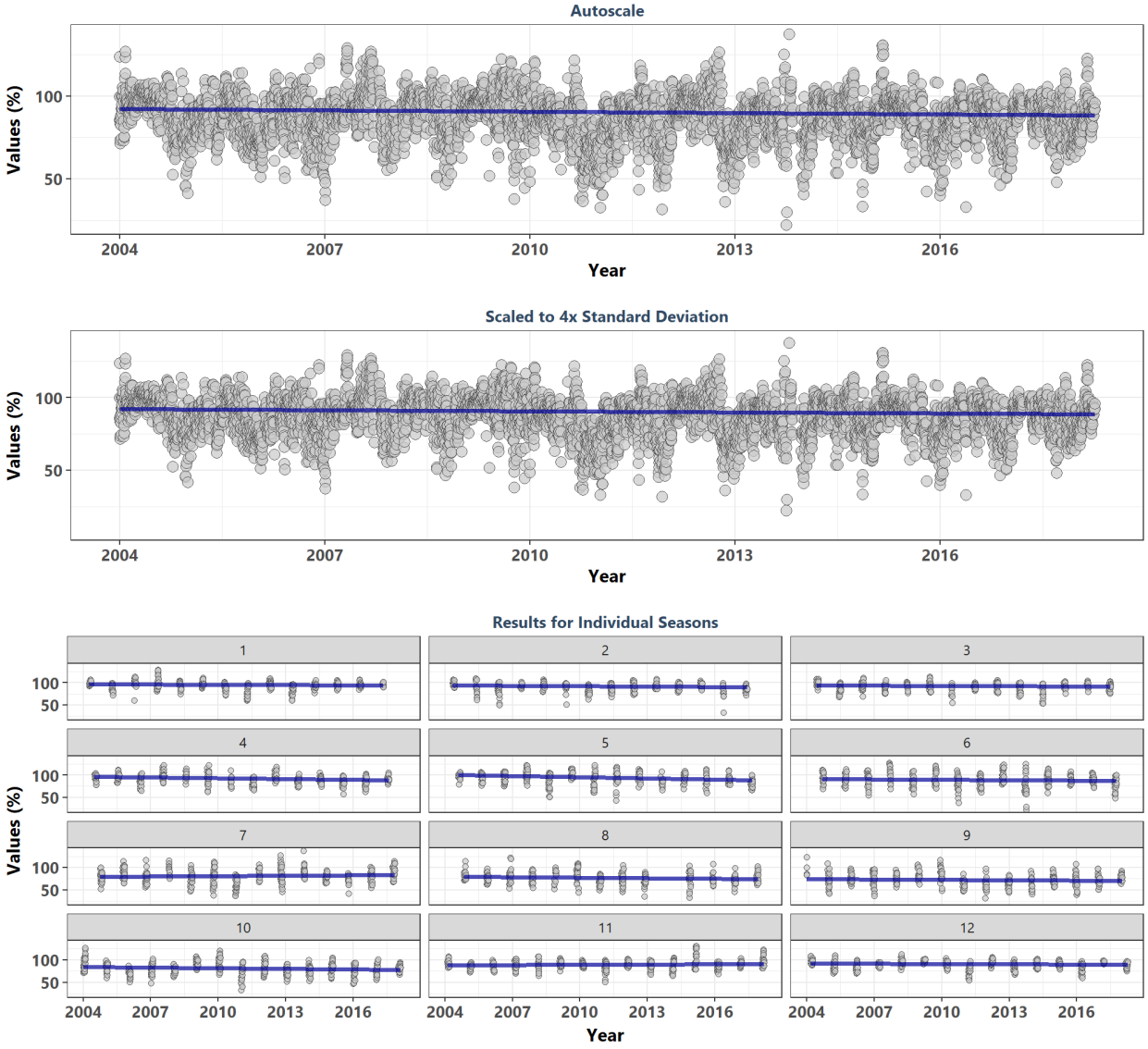
Estero Bay Aquatic Preserve
474 | Estero Bay Aquatic Preserve Continuous Water Quality Monitoring
EB03



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	4558	85.16	0.1786	0.5955	78.0096	18.0	0.0000	66.3	0	1
1	342	93.84	0.1056	0.2752	90.8091	2.9	0.0034	NA	NA	1
2	391	92.05	0.0398	0.1040	91.0132	1.2	0.2385	NA	NA	1
3	408	91.77	0.1183	0.3199	88.5678	3.6	0.0003	NA	NA	1
4	330	89.78	0.2195	0.5619	84.1575	6.0	0.0000	NA	NA	1
5	443	85.73	0.3037	0.9492	76.2393	9.6	0.0000	NA	NA	1
6	360	78.38	0.1807	0.7205	70.4522	5.1	0.0000	NA	NA	1
7	373	72.31	0.1293	0.7077	64.5207	3.7	0.0002	NA	NA	1
8	299	68.84	0.2247	1.1881	55.7737	5.8	0.0000	NA	NA	1
9	396	63.60	0.2435	1.1986	51.6159	7.3	0.0000	NA	NA	1
10	392	74.19	0.2873	1.1127	64.1771	8.5	0.0000	NA	NA	1
11	396	86.21	0.1914	0.6426	79.7800	5.7	0.0000	NA	NA	1
12	428	91.69	0.0981	0.2408	89.5213	3.0	0.0024	NA	NA	1

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

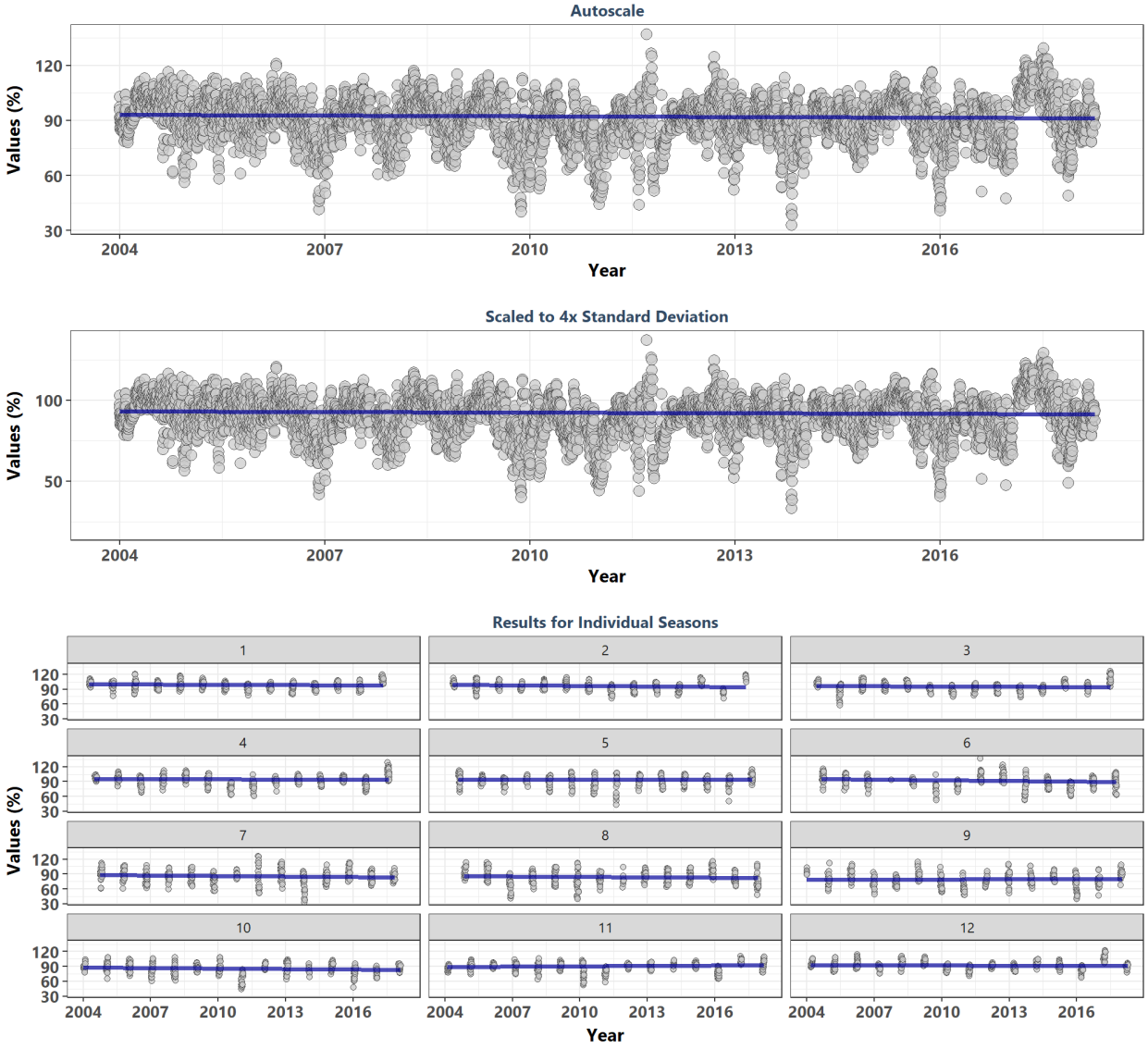
Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP1A



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	4706	88.22	-0.0661	-0.2641	92.1473	-6.7	0.0000	42.3	0	-1
1	399	95.38	-0.0632	-0.1906	96.9010	-1.9	0.0588	NA	NA	-1
2	347	91.99	-0.0796	-0.2742	94.1823	-2.2	0.0264	NA	NA	-1
3	366	92.46	-0.0413	-0.1253	93.3974	-1.2	0.2373	NA	NA	-1
4	391	91.36	-0.1649	-0.6138	96.2738	-4.9	0.0000	NA	NA	-1
5	394	94.40	-0.1731	-0.8693	100.4807	-5.1	0.0000	NA	NA	-1
6	383	89.12	-0.0569	-0.3410	91.8499	-1.7	0.0956	NA	NA	-1
7	375	81.21	0.0451	0.2656	79.3531	1.3	0.1906	NA	NA	1
8	354	76.86	-0.0895	-0.4202	79.8062	-2.5	0.0117	NA	NA	-1
9	388	72.66	-0.0574	-0.2938	74.7163	-1.7	0.0906	NA	NA	-1
10	452	81.14	-0.0891	-0.4294	84.1492	-2.8	0.0045	NA	NA	-1
11	430	89.20	0.0433	0.1465	88.1789	1.3	0.1793	NA	NA	1
12	427	90.94	-0.0706	-0.2146	92.4448	-2.2	0.0289	NA	NA	-1

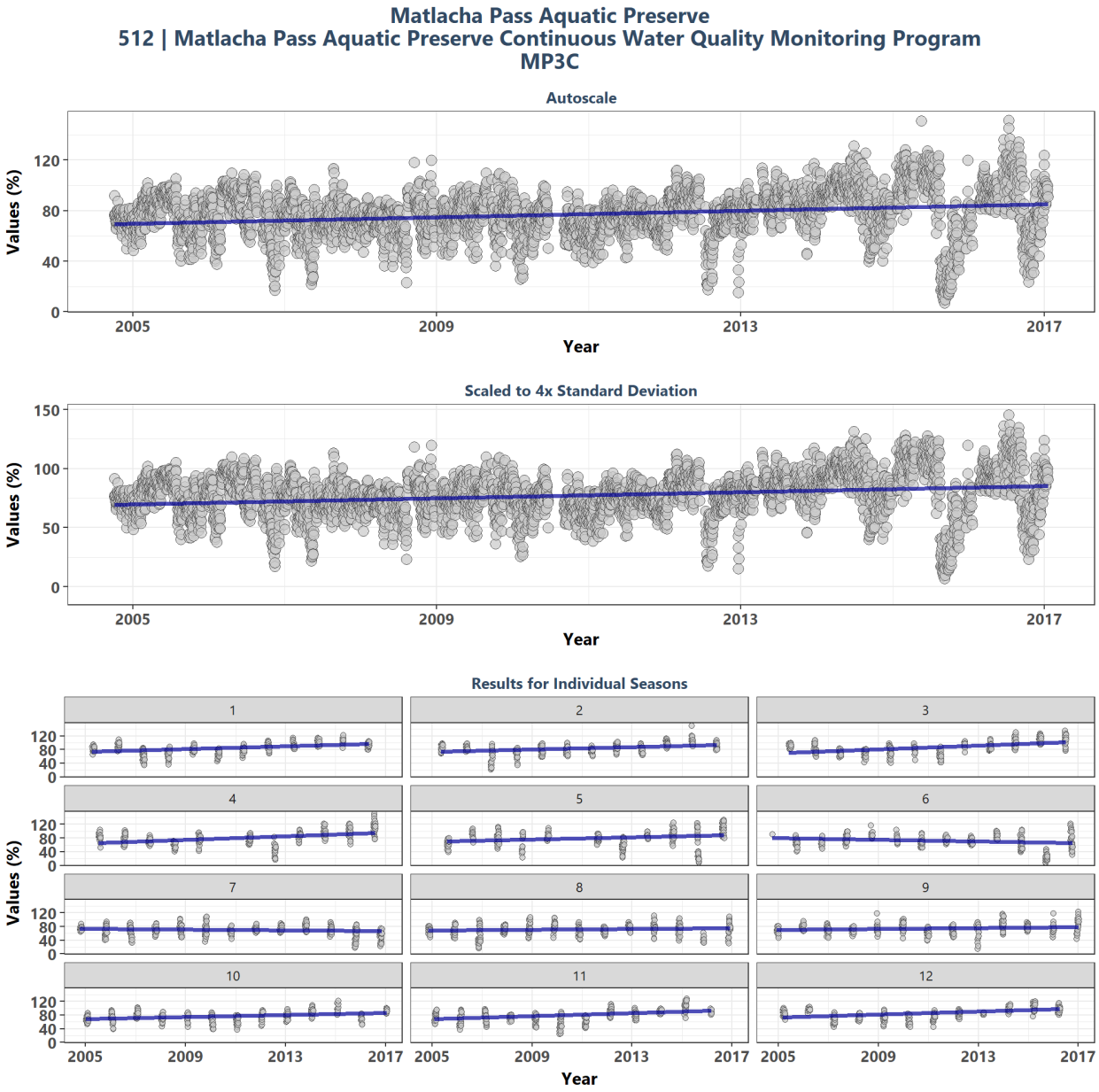
[a] $p < 0.00005$ appear as 0 due to rounding

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



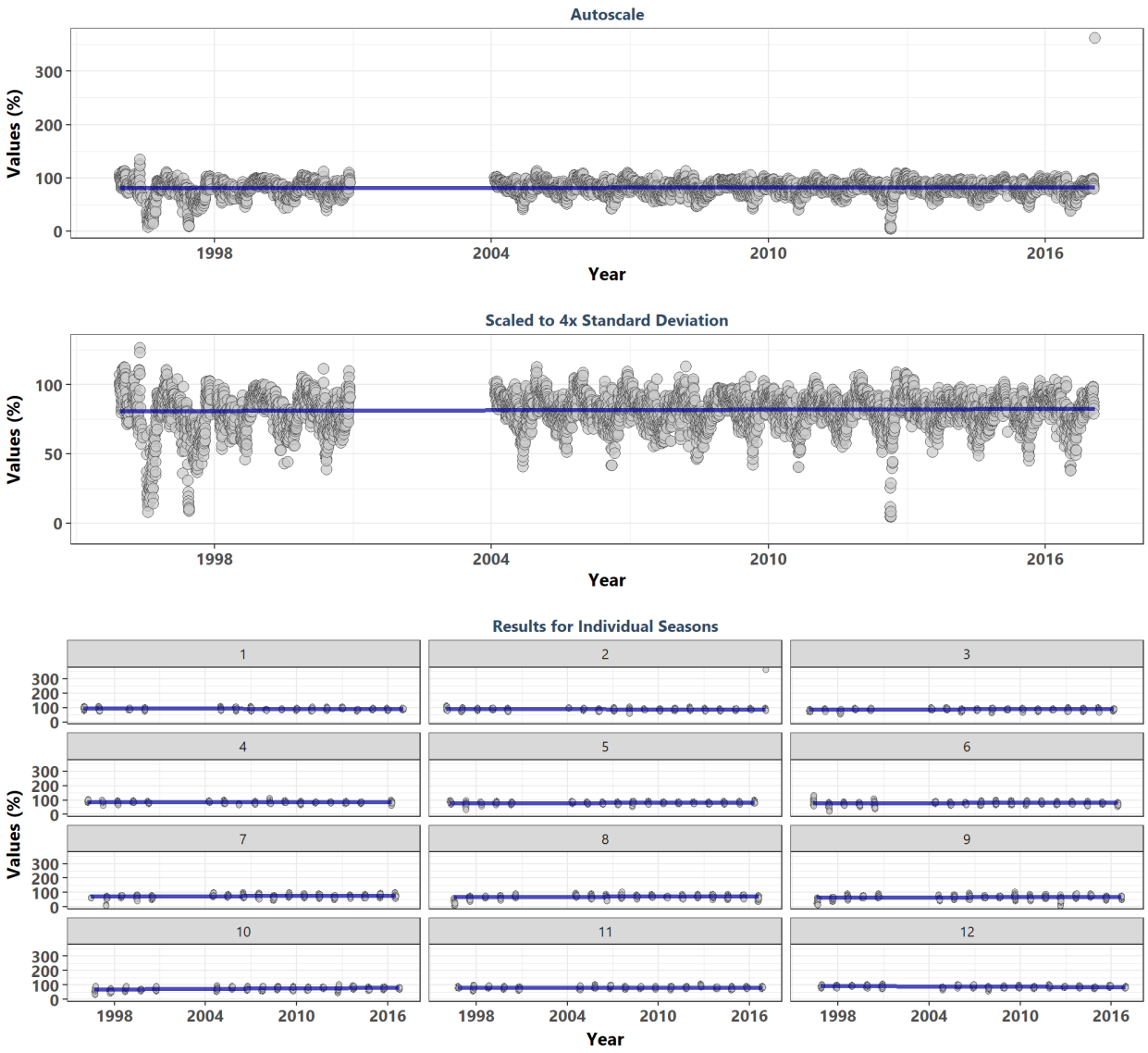
Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	4915	91.39	-0.0351	-0.1250	93.1884	-3.5	0.0004	29.1	0.0022	-1
1	433	98.93	-0.0515	-0.1500	100.1330	-1.6	0.1083	NA	NA	-1
2	364	96.36	-0.0901	-0.3500	98.8056	-2.6	0.0101	NA	NA	-1
3	400	95.33	-0.0298	-0.1052	96.0702	-0.9	0.3724	NA	NA	-1
4	416	94.24	-0.0210	-0.0836	94.8274	-0.6	0.5204	NA	NA	-1
5	413	94.07	0.0007	0.0036	94.0447	0.0	0.9837	NA	NA	1
6	352	91.09	-0.1074	-0.4770	95.3866	-3.0	0.0026	NA	NA	-1
7	410	85.75	-0.0741	-0.3682	88.6990	-2.2	0.0246	NA	NA	-1
8	415	84.13	-0.0529	-0.2816	86.1025	-1.6	0.1067	NA	NA	-1
9	412	79.47	0.0074	0.0406	79.1443	0.2	0.8222	NA	NA	1
10	415	86.38	-0.0977	-0.3431	88.4365	-3.0	0.0029	NA	NA	-1
11	420	90.56	0.0864	0.2556	88.7755	2.7	0.0080	NA	NA	1
12	465	92.08	-0.0123	-0.0357	92.3320	-0.4	0.6925	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	4094	77.91	0.1798	1.2744	64.6925	16.8	0.0000	199.5	0	1
1	347	85.40	0.3070	1.9585	65.8121	8.6	0.0000	NA	NA	1
2	319	83.72	0.2646	1.8153	65.5642	7.1	0.0000	NA	NA	1
3	333	85.92	0.3220	2.7611	58.3108	8.8	0.0000	NA	NA	1
4	320	82.22	0.3177	2.6608	52.9483	8.5	0.0000	NA	NA	1
5	294	81.42	0.1691	1.6152	63.6554	4.3	0.0000	NA	NA	1
6	311	71.76	-0.1275	-1.1422	84.3281	-3.4	0.0008	NA	NA	-1
7	403	70.10	-0.0770	-0.4570	74.6681	-2.3	0.0207	NA	NA	-1
8	383	72.08	0.1008	0.6074	66.0008	3.0	0.0031	NA	NA	1
9	349	74.21	0.0975	0.5883	68.3229	2.7	0.0064	NA	NA	1
10	338	76.42	0.2439	1.5040	62.8891	6.7	0.0000	NA	NA	1
11	344	79.01	0.2931	2.2816	58.4724	8.1	0.0000	NA	NA	1
12	353	83.80	0.2874	2.2196	63.8208	8.1	0.0000	NA	NA	1

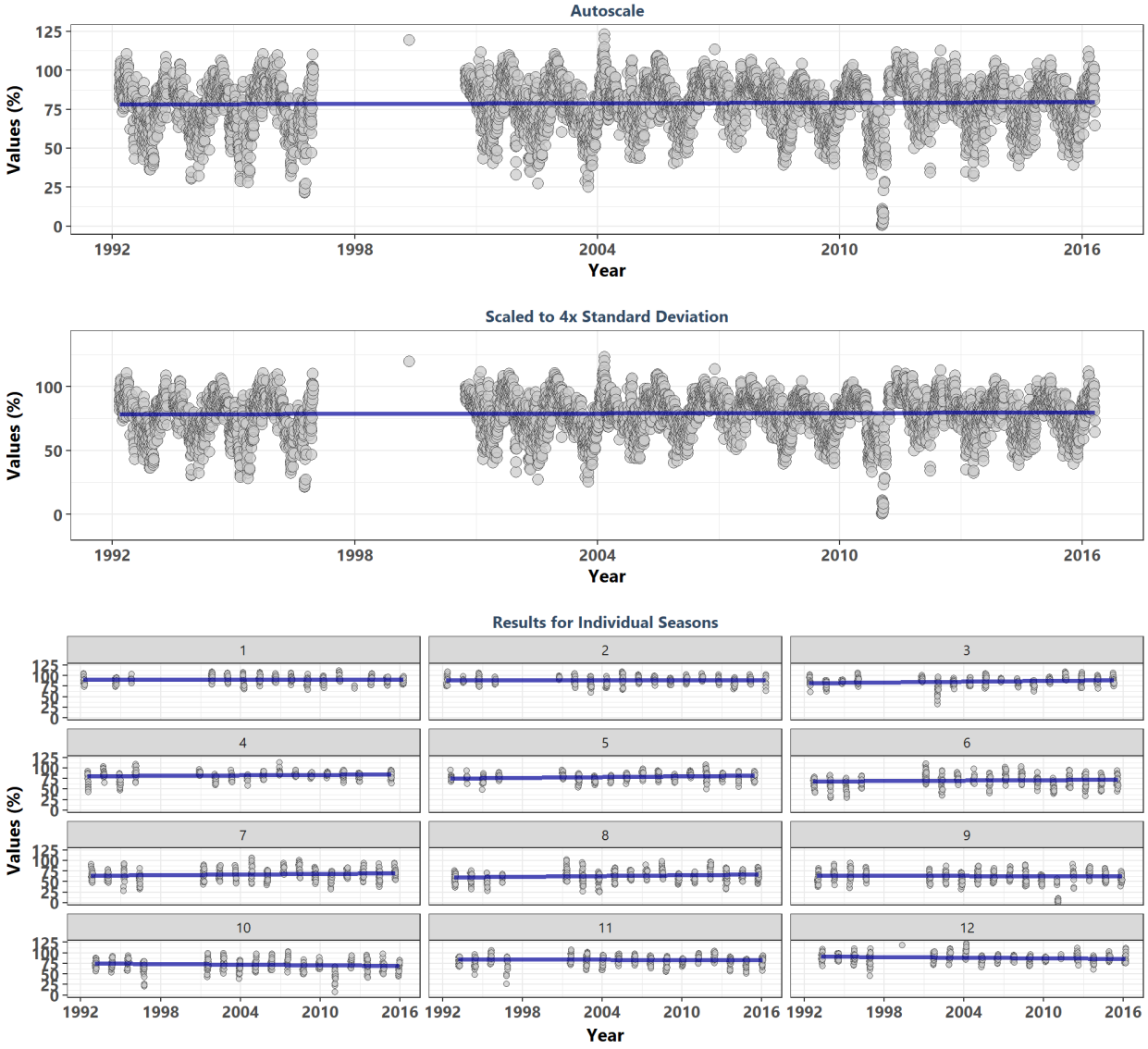
[a] $p < 0.00005$ appear as 0 due to rounding



Season	N	Median	tau	Slope	Int.	z	p__z	chi__sq	p__chi__sq	Trend
All	6020	82.44	0.0322	0.0761	80.6578	4.0	0.0001	195.6	0	1
1	500	90.98	-0.1183	-0.2260	94.8266	-4.0	0.0001	NA	NA	-1
2	488	89.22	-0.0668	-0.1218	91.1708	-2.2	0.0271	NA	NA	-1
3	529	87.71	0.0986	0.1819	84.8004	3.4	0.0007	NA	NA	1
4	452	85.52	-0.1028	-0.1471	87.8727	-3.3	0.0011	NA	NA	-1
5	508	81.75	0.1263	0.2250	78.1510	4.3	0.0000	NA	NA	1
6	495	79.39	0.0914	0.2461	75.6938	3.0	0.0023	NA	NA	1
7	455	75.74	0.0681	0.2015	72.5177	2.2	0.0296	NA	NA	1
8	486	71.59	0.0829	0.2518	67.5626	2.7	0.0062	NA	NA	1
9	525	68.42	0.0986	0.3313	63.4539	3.4	0.0007	NA	NA	1
10	552	74.83	0.2540	0.6598	64.2788	8.9	0.0000	NA	NA	1
11	500	83.06	-0.0038	-0.0069	83.1646	-0.1	0.8981	NA	NA	-1
12	530	88.72	-0.1709	-0.3216	93.8681	-5.9	0.0000	NA	NA	-1

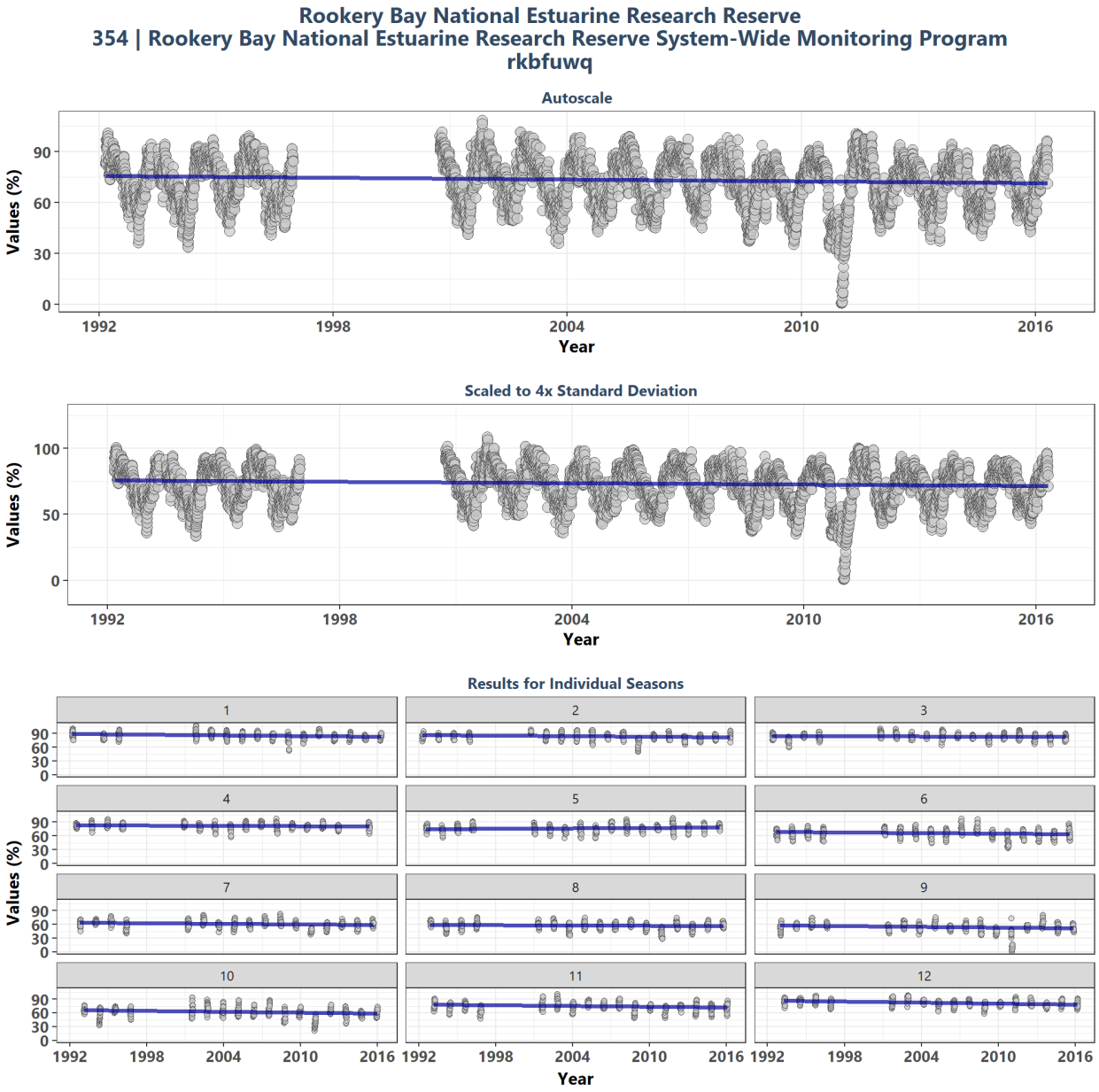
[a] $p < 0.00005$ appear as 0 due to rounding

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



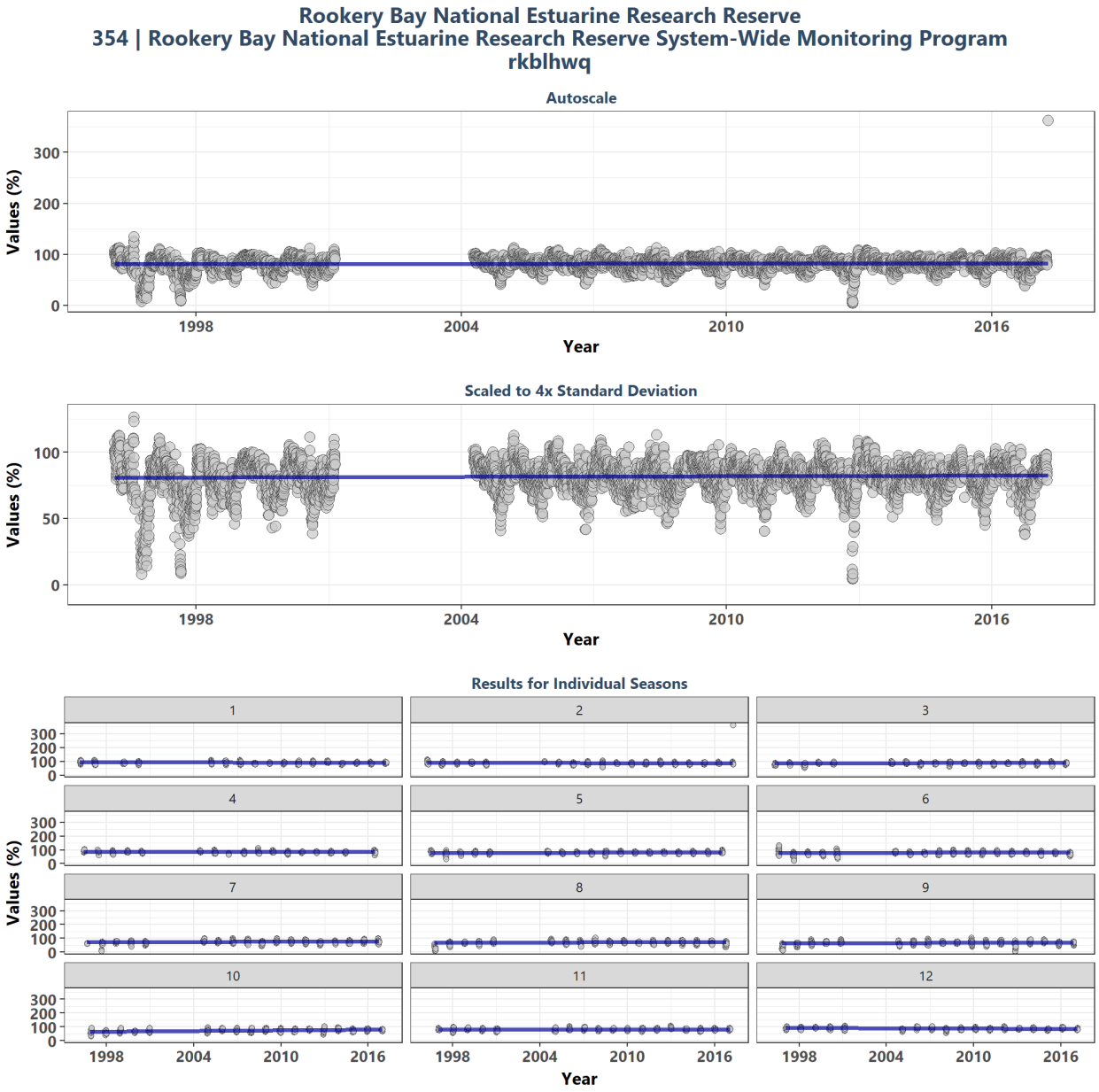
Season	N	Median	tau	Slope	Int.	z	p__z	chi__sq	p__chi__sq	Trend
All	5605	79.79	0.0250	0.0658	78.0139	2.7	0.0075	94.8	0	1
1	444	90.31	-0.0279	-0.0613	91.4164	-0.9	0.3793	NA	NA	-1
2	437	89.19	-0.0065	-0.0131	89.4286	-0.2	0.8391	NA	NA	-1
3	477	86.58	0.1437	0.3249	81.0532	4.7	0.0000	NA	NA	1
4	431	83.49	0.0953	0.2328	79.5314	3.0	0.0030	NA	NA	1
5	469	79.42	0.1433	0.3269	73.8602	4.6	0.0000	NA	NA	1
6	473	70.56	0.0578	0.2171	66.6498	1.9	0.0597	NA	NA	1
7	483	67.63	0.0725	0.2518	63.3492	2.4	0.0170	NA	NA	1
8	465	63.88	0.1017	0.3250	58.3521	3.3	0.0010	NA	NA	1
9	446	62.86	-0.0285	-0.0974	64.5168	-0.9	0.3677	NA	NA	-1
10	496	71.80	-0.0850	-0.2761	76.4963	-2.8	0.0046	NA	NA	-1
11	486	83.37	-0.0179	-0.0500	84.2188	-0.6	0.5558	NA	NA	-1
12	498	88.43	-0.1354	-0.3163	93.8072	-4.5	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	5933	73.94	-0.1084	-0.2163	76.9486	-12.5	0.0000	84.2	0	-1
1	488	85.07	-0.1504	-0.3007	90.4844	-5.0	0.0000	NA	NA	-1
2	445	83.42	-0.1363	-0.2364	87.6726	-4.3	0.0000	NA	NA	-1
3	486	83.60	-0.0385	-0.0701	84.7913	-1.3	0.2033	NA	NA	-1
4	462	81.69	-0.1085	-0.1499	84.2362	-3.5	0.0005	NA	NA	-1
5	502	77.21	0.0970	0.1675	74.3573	3.3	0.0011	NA	NA	1
6	499	66.01	-0.1013	-0.2472	70.2168	-3.4	0.0007	NA	NA	-1
7	508	60.58	-0.1323	-0.2735	65.2339	-4.5	0.0000	NA	NA	-1
8	512	57.46	-0.0672	-0.1310	59.6868	-2.3	0.0227	NA	NA	-1
9	501	54.36	-0.1305	-0.2767	59.0608	-4.4	0.0000	NA	NA	-1
10	518	61.69	-0.1508	-0.3853	68.2440	-5.1	0.0000	NA	NA	-1
11	502	74.11	-0.1327	-0.3194	79.5399	-4.5	0.0000	NA	NA	-1
12	510	81.42	-0.2487	-0.4705	89.4233	-8.4	0.0000	NA	NA	-1

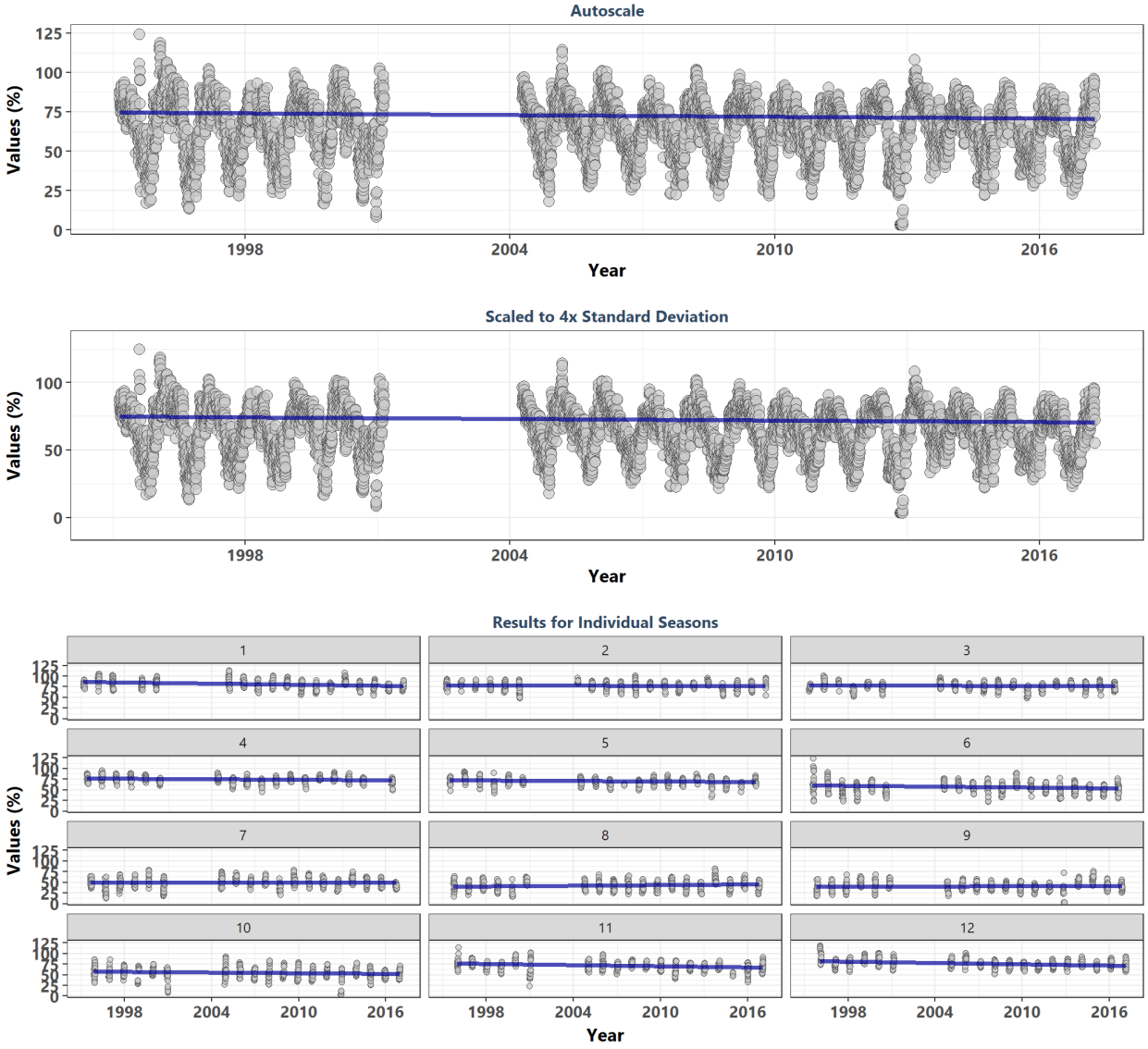
[a] $p < 0.00005$ appear as 0 due to rounding



Season	N	Median	tau	Slope	Int.	z	p_z	chi_sq	p_chi_sq	Trend
All	6020	82.44	0.0322	0.0761	80.5488	4.0	0.0001	195.6	0	1
1	500	90.98	-0.1183	-0.2260	95.0526	-4.0	0.0001	NA	NA	-1
2	488	89.22	-0.0668	-0.1218	91.2926	-2.2	0.0271	NA	NA	-1
3	529	87.71	0.0986	0.1819	84.6185	3.4	0.0007	NA	NA	1
4	452	85.52	-0.1028	-0.1471	88.0197	-3.3	0.0011	NA	NA	-1
5	508	81.75	0.1263	0.2250	77.9260	4.3	0.0000	NA	NA	1
6	495	79.39	0.0914	0.2461	75.4477	3.0	0.0023	NA	NA	1
7	455	75.74	0.0681	0.2015	72.3162	2.2	0.0296	NA	NA	1
8	486	71.59	0.0829	0.2518	67.3108	2.7	0.0062	NA	NA	1
9	525	68.42	0.0986	0.3313	63.1226	3.4	0.0007	NA	NA	1
10	552	74.83	0.2540	0.6598	63.6191	8.9	0.0000	NA	NA	1
11	500	83.06	-0.0038	-0.0069	83.1715	-0.1	0.8981	NA	NA	-1
12	530	88.72	-0.1709	-0.3216	94.1897	-5.9	0.0000	NA	NA	-1

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

Rookery Bay National Estuarine Research Reserve
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	6472	67.13	-0.0768	-0.1837	75.2559	-9.2	0.0000	124	0	-1
1	554	80.27	-0.1950	-0.4520	88.1750	-6.9	0.0000	NA	NA	-1
2	494	77.61	-0.0412	-0.0920	79.1717	-1.4	0.1709	NA	NA	-1
3	568	77.55	-0.0152	-0.0295	78.0257	-0.5	0.5880	NA	NA	-1
4	514	74.36	-0.0882	-0.1605	76.9237	-3.0	0.0028	NA	NA	-1
5	507	70.35	-0.0923	-0.1907	73.5881	-3.1	0.0019	NA	NA	-1
6	539	56.27	-0.1011	-0.3060	61.1711	-3.5	0.0004	NA	NA	-1
7	563	49.55	0.0093	0.0260	49.1312	0.3	0.7419	NA	NA	1
8	546	42.89	0.0869	0.2326	39.1703	3.0	0.0024	NA	NA	1
9	544	41.12	0.0237	0.0653	40.0792	0.8	0.4088	NA	NA	1
10	560	54.55	-0.0893	-0.2530	58.5998	-3.2	0.0015	NA	NA	-1
11	523	70.83	-0.1946	-0.4817	78.5371	-6.7	0.0000	NA	NA	-1
12	560	76.00	-0.2275	-0.5151	84.2384	-8.1	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

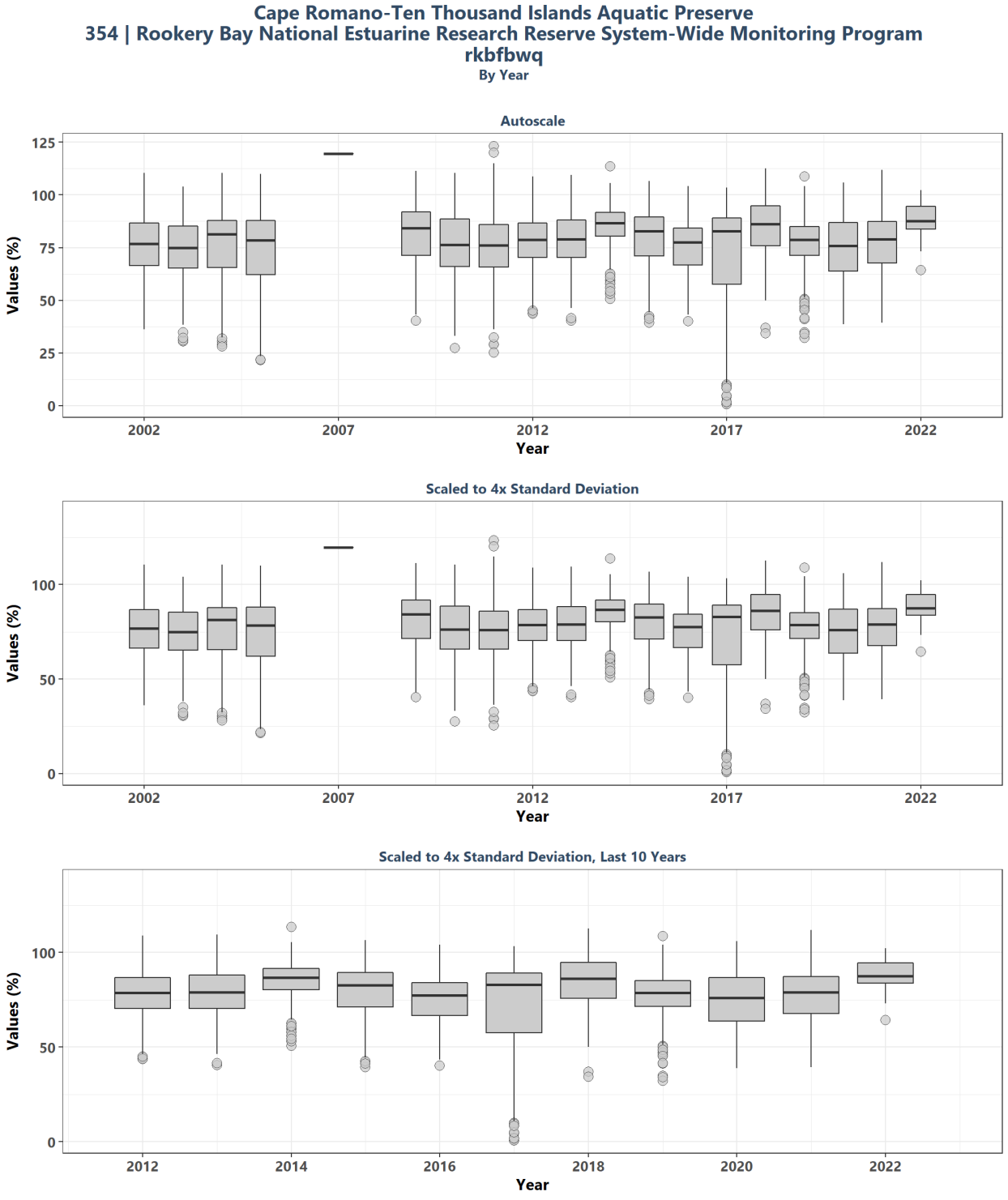
```
if(n==0){
   print("There are no monitoring locations that qualify.")
} else {
   for (i in 1:n) {
      year_lower <- min(data$Year[data$Use_In_Analysis==TRUE &
                                  data$MonitoringID==Mon_IDs[i]])
      year_upper <- max(data$Year[data$Use_In_Analysis==TRUE &
     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

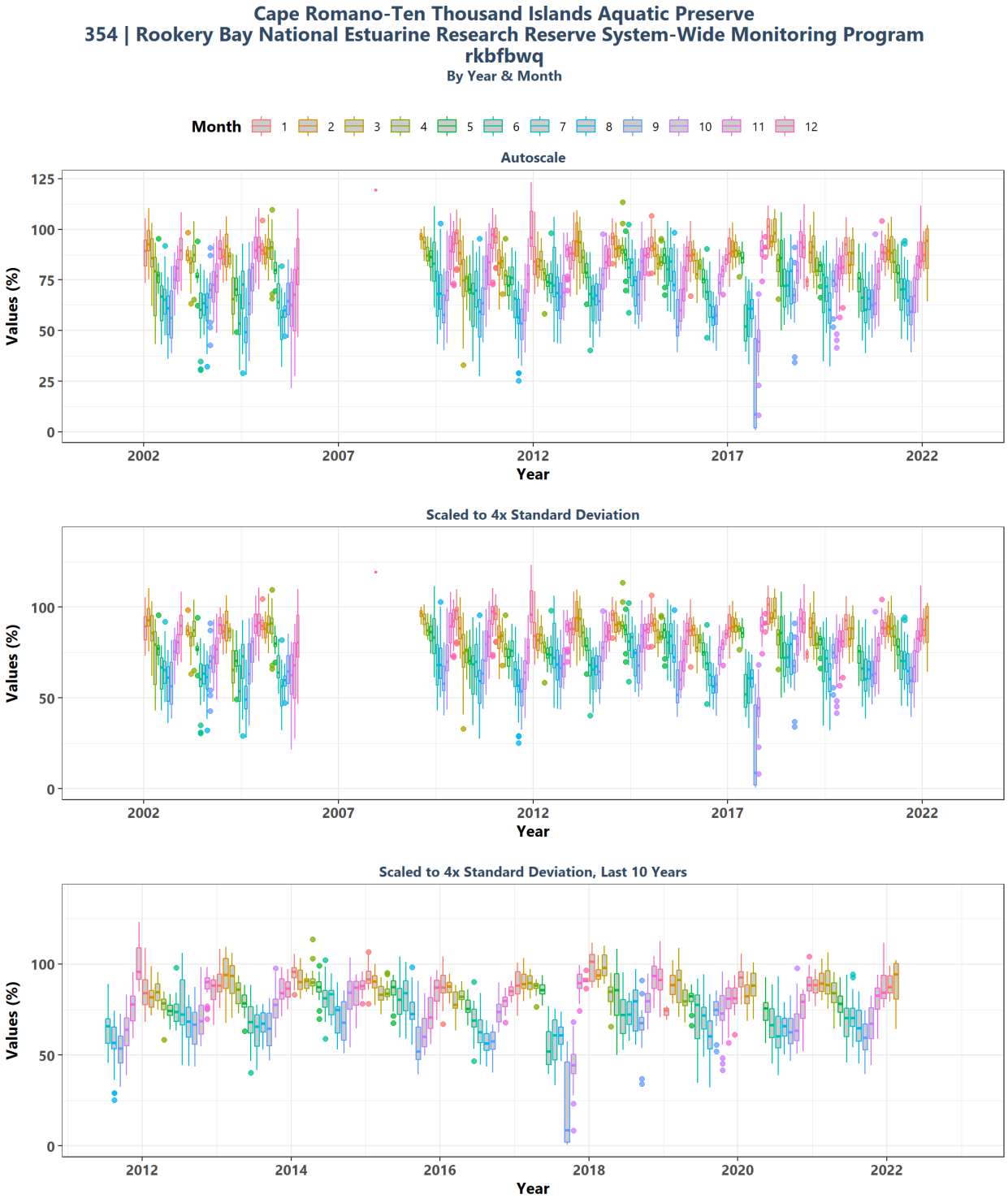
```
                aes(x=Year, y=ResultValue, group=Year)) +
    geom_boxplot(color="#333333", fill="#cccccc", outlier.shape=21,
                 outlier.size=3, outlier.color="#333333",
                 outlier.fill="#cccccc", outlier.alpha=0.75) +
    labs(subtitle="Autoscale",
         x="Year",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

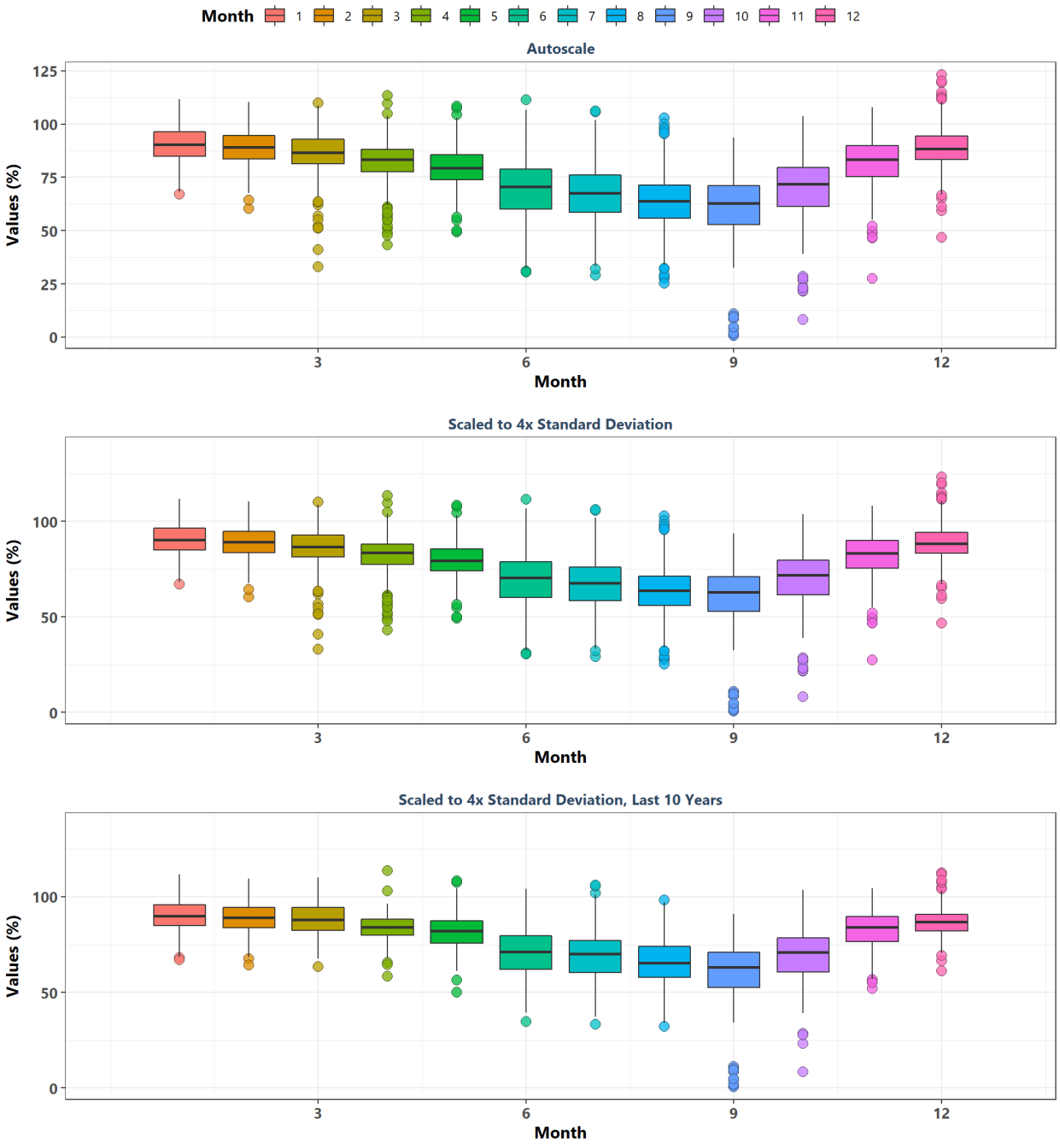
```
   geom_boxplot(color="#333333", outlier.shape=21, outlier.size=3,
                outlier.color="#333333", outlier.alpha=0.75) +
   labs(subtitle="Autoscale",
        x="Month", y=paste0("Values (", unit, ")"), fill="Month") +
   scale_x_continuous(limits=c(0, 13), breaks=seq(3, 12, 3)) +
   plot_theme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

```
    }
}
```

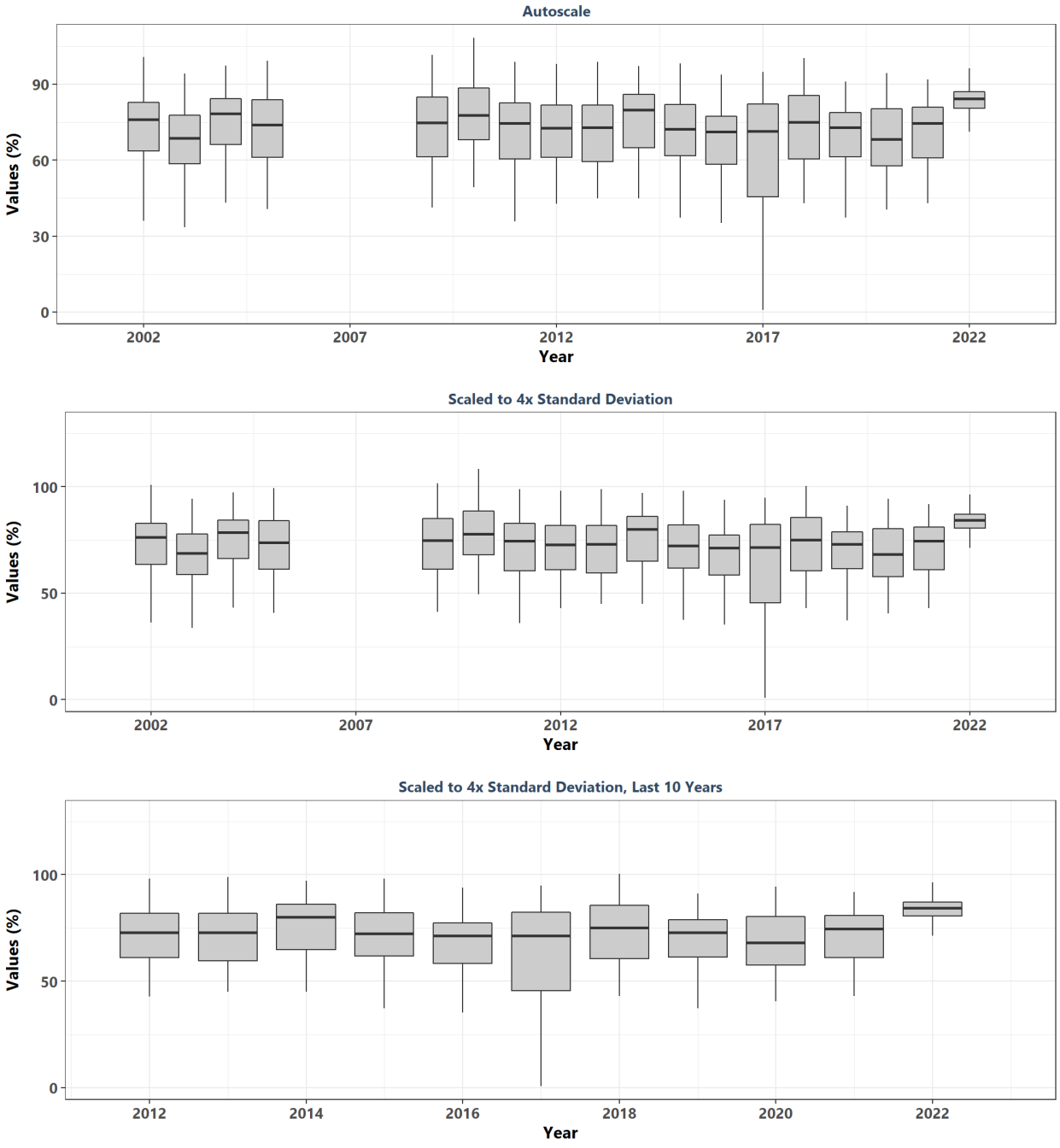

Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfbwq
By Year
Autoscale
Values (%)
125
100
75
50
25
0
2002
2007
2012
2017
2022
Year
Scaled to 4x Standard Deviation
Values (%)
100
50
0
2002
2007
2012
2017
2022
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (%)
100
50
0
2012
2014
2016
2018
2020
2022
Year

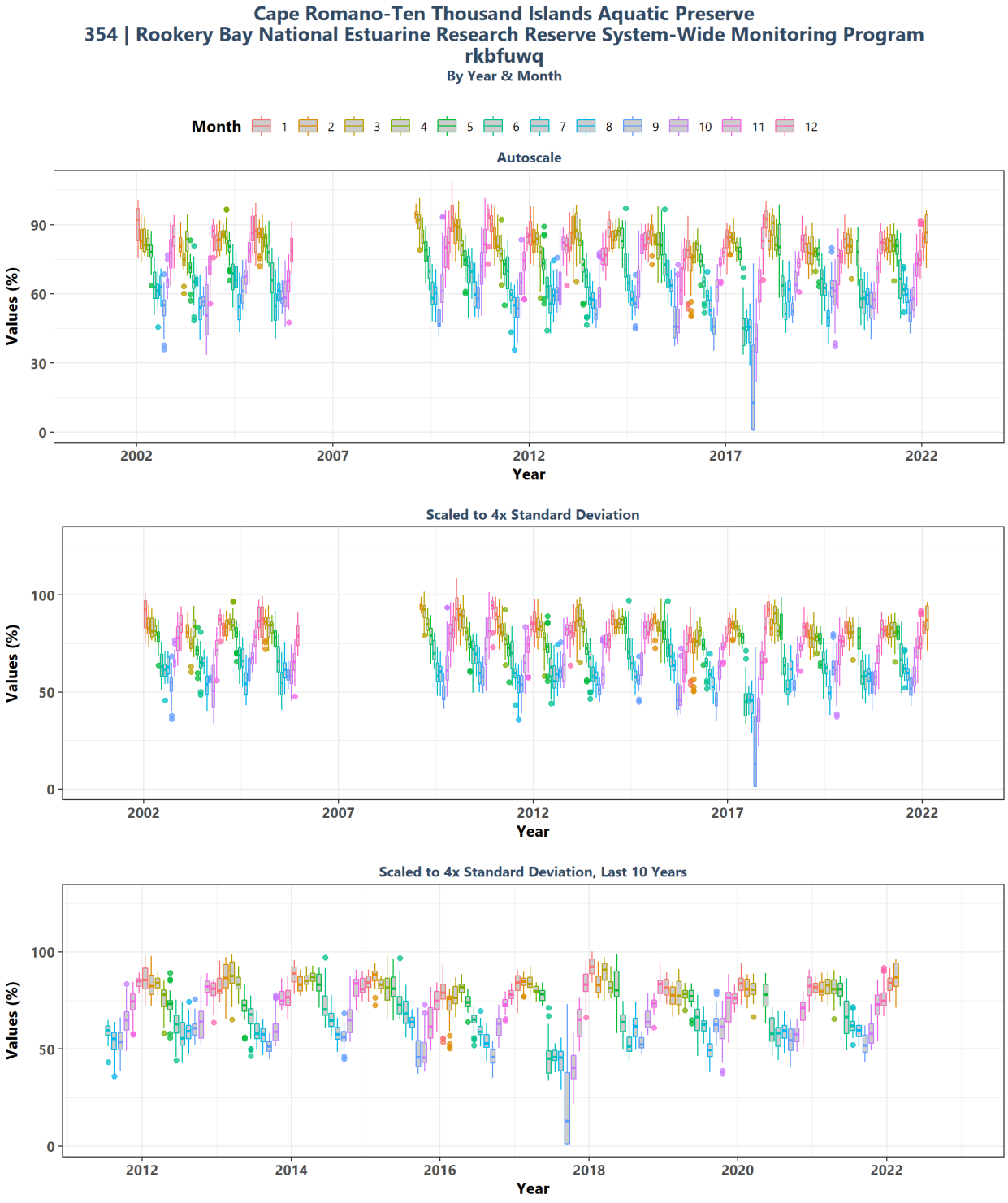

Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfbwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
Year
Scaled to 4x Standard Deviation
Values (%)
Year
Scaled to 4x Standard Deviation, Last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

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

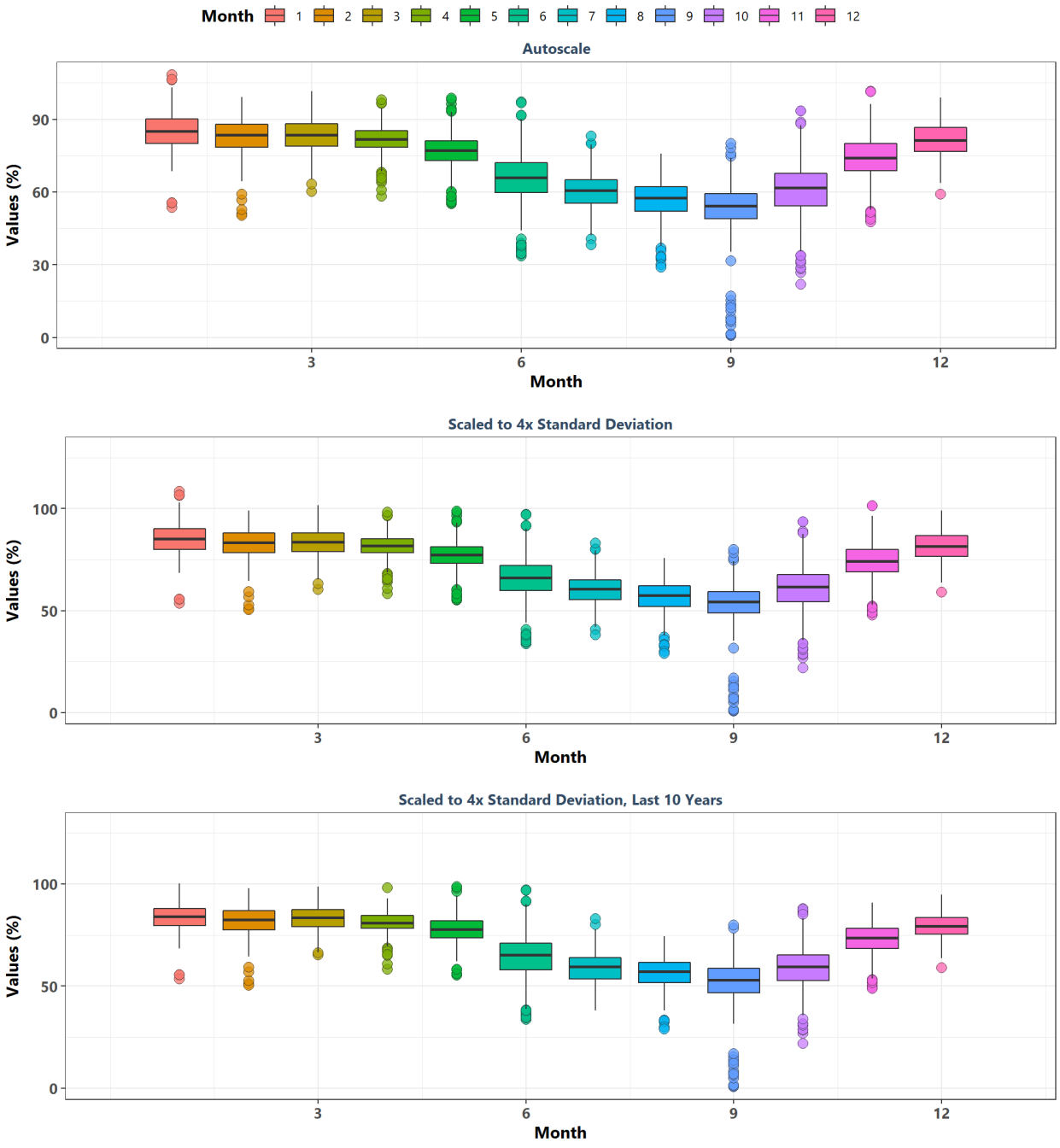


Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfuwq
By Year

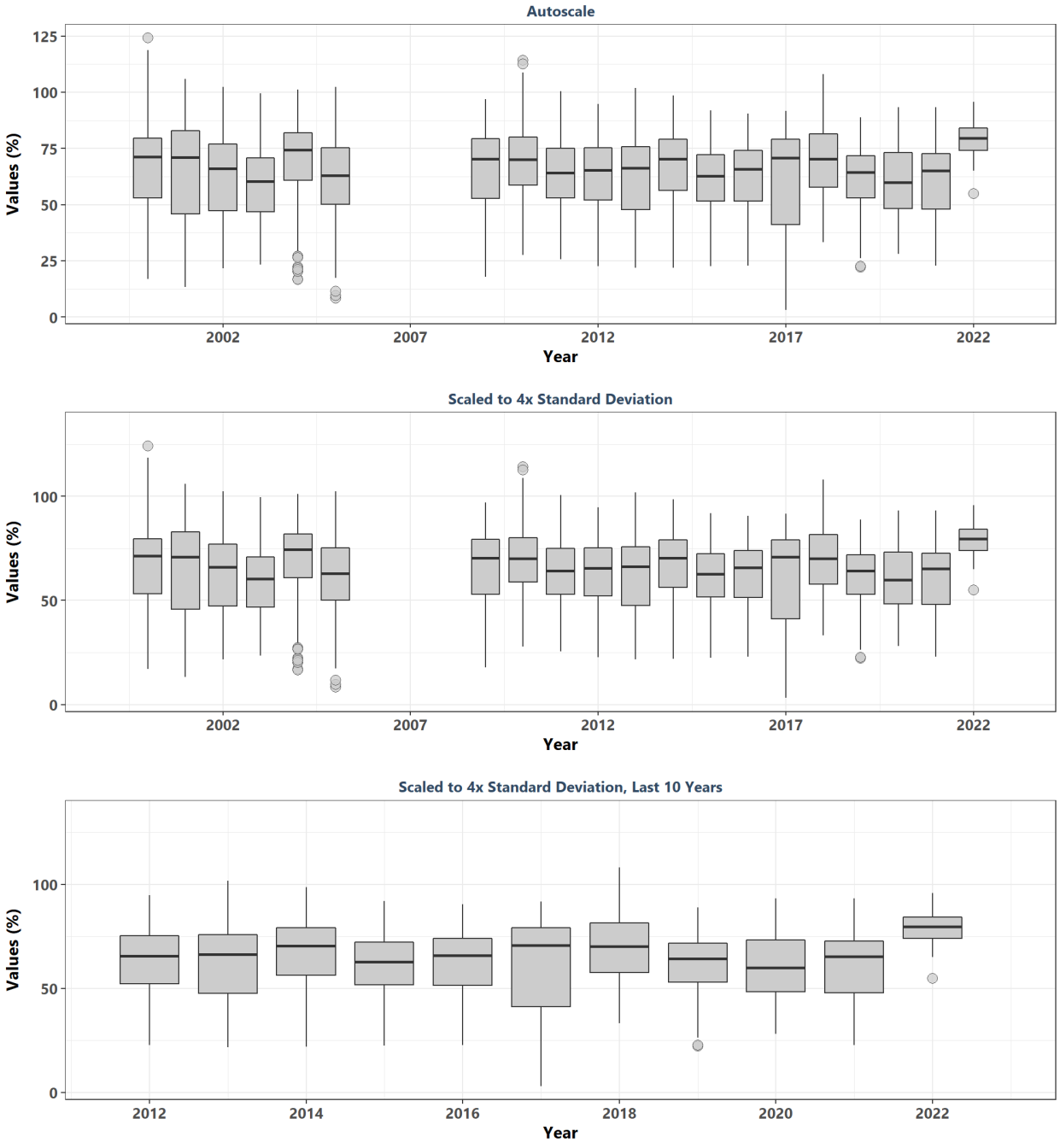


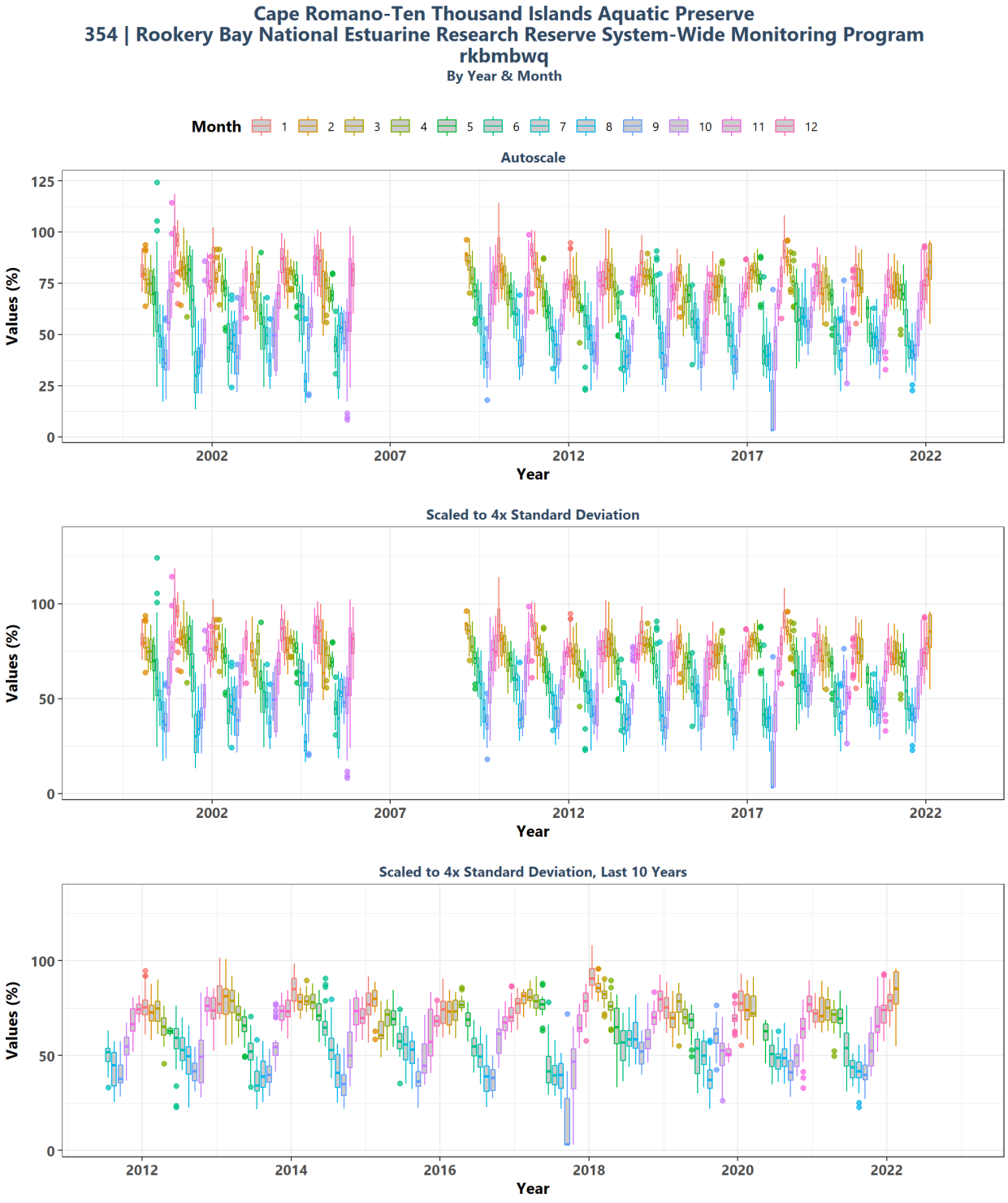

Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfuwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
Year
Scaled to 4x Standard Deviation
Values (%)
Year
Scaled to 4x Standard Deviation, Last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

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

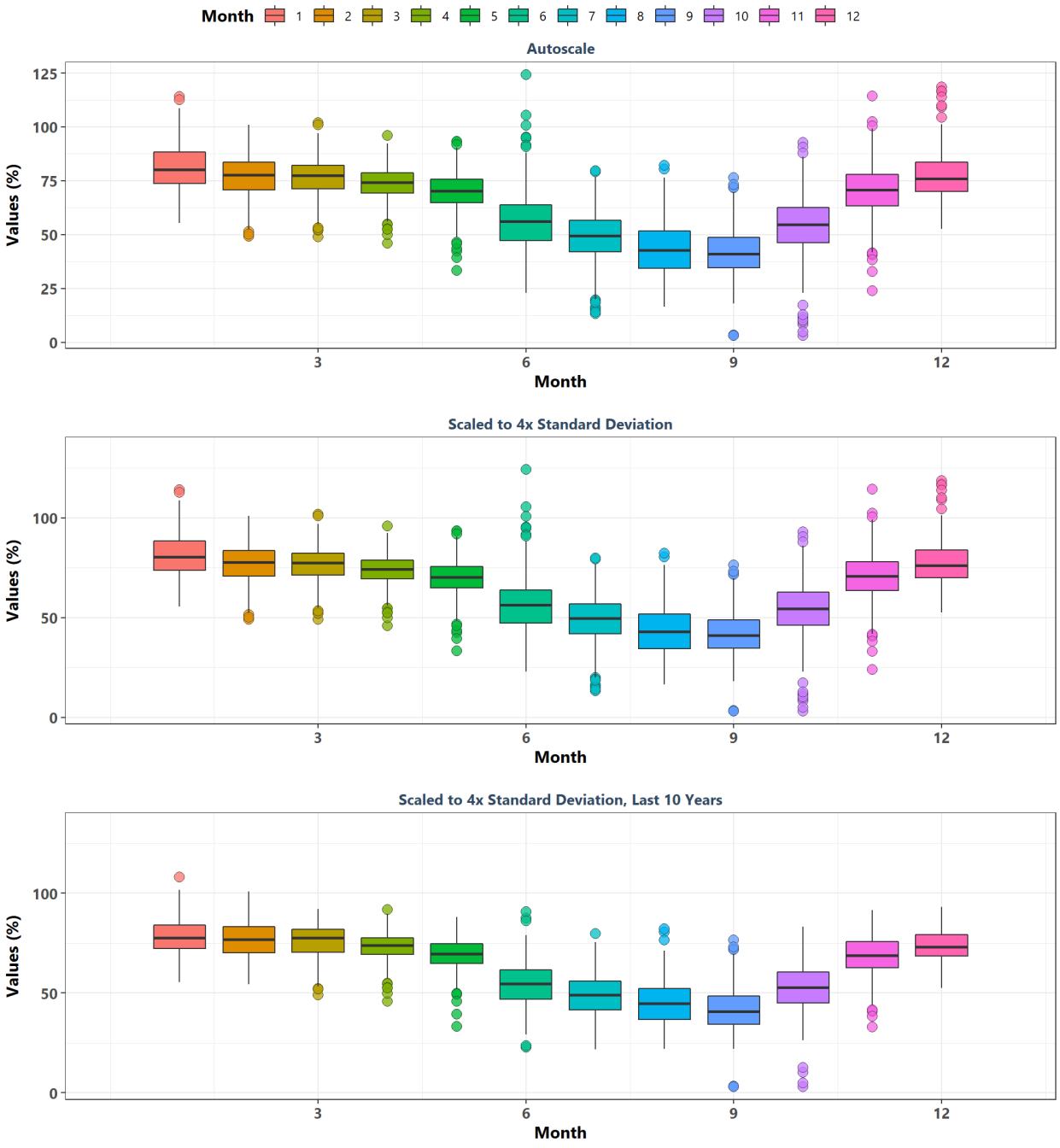


Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbmbwq
By Year

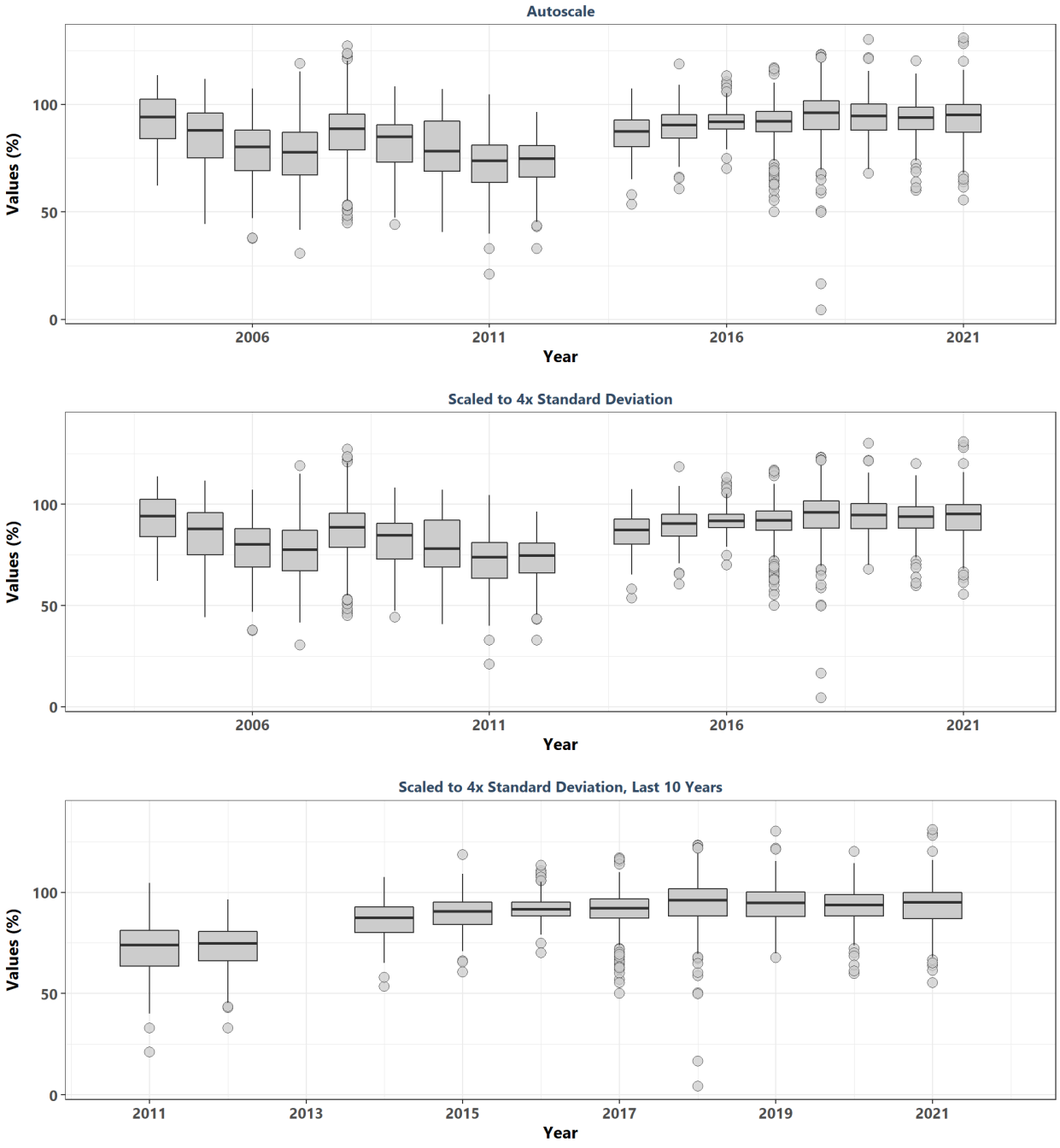


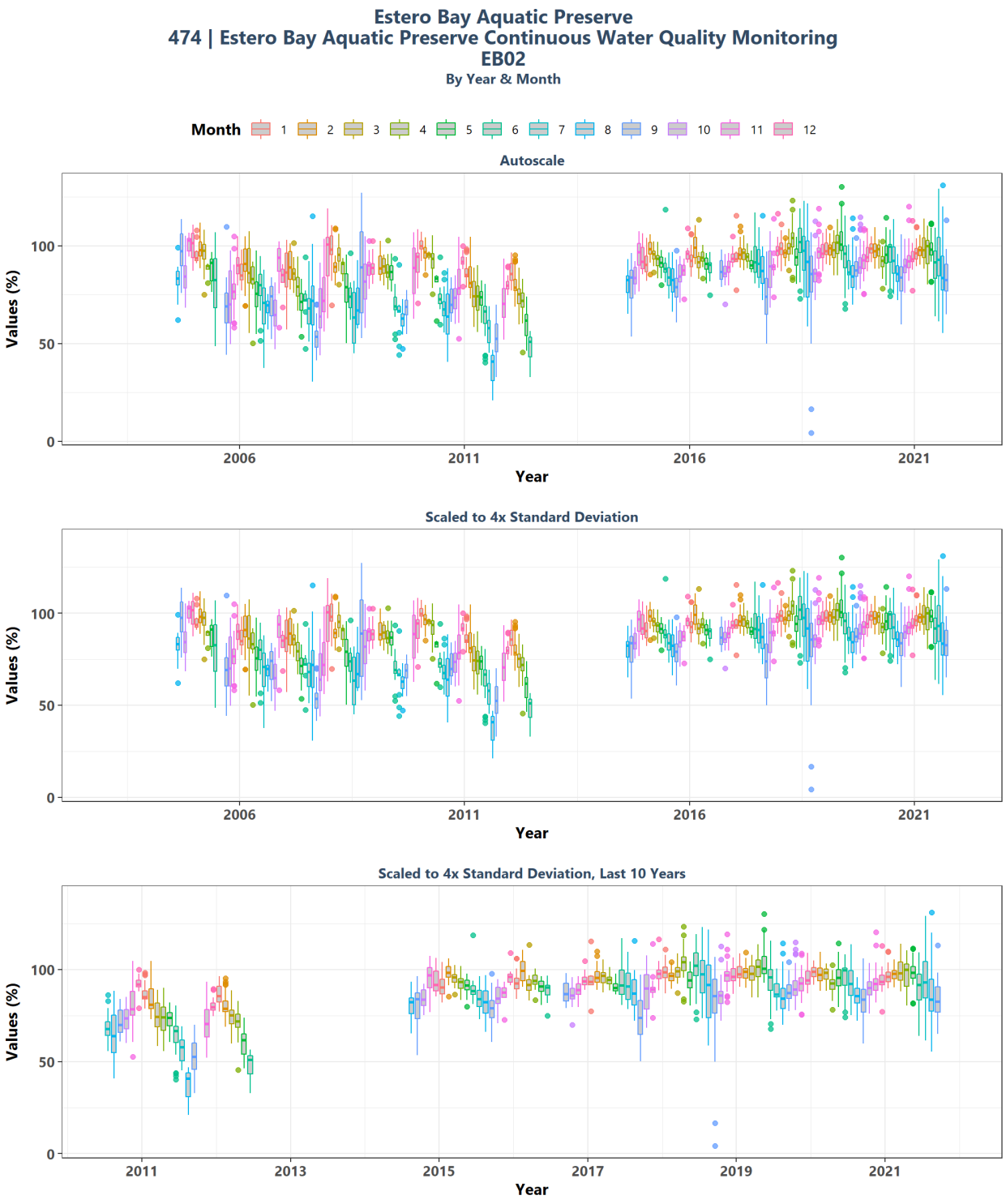

Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbmbwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
Year
Scaled to 4x Standard Deviation
Values (%)
Year
Scaled to 4x Standard Deviation, Last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

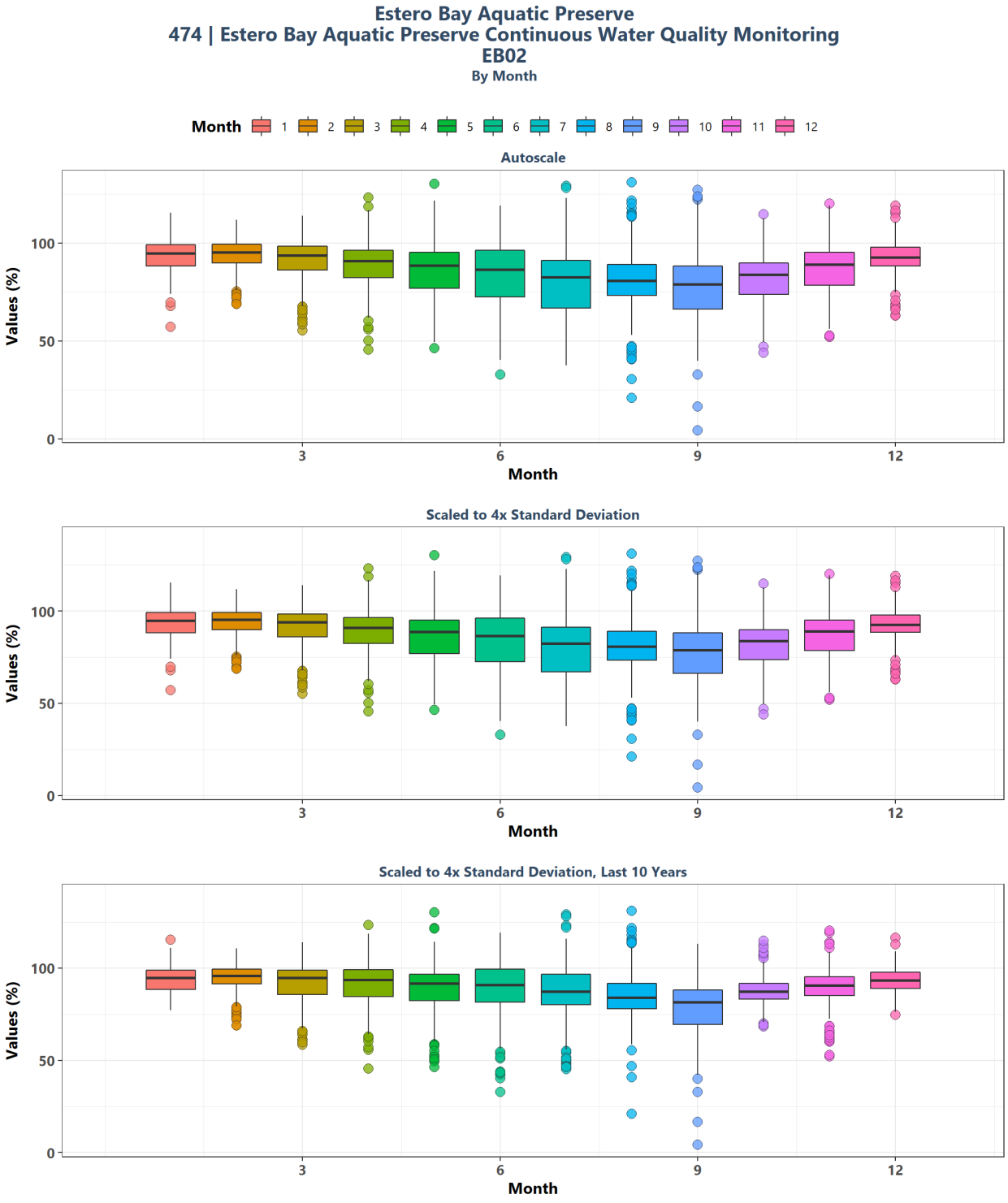
Cape Romano-Ten Thousand Islands Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbmbwq
By Month



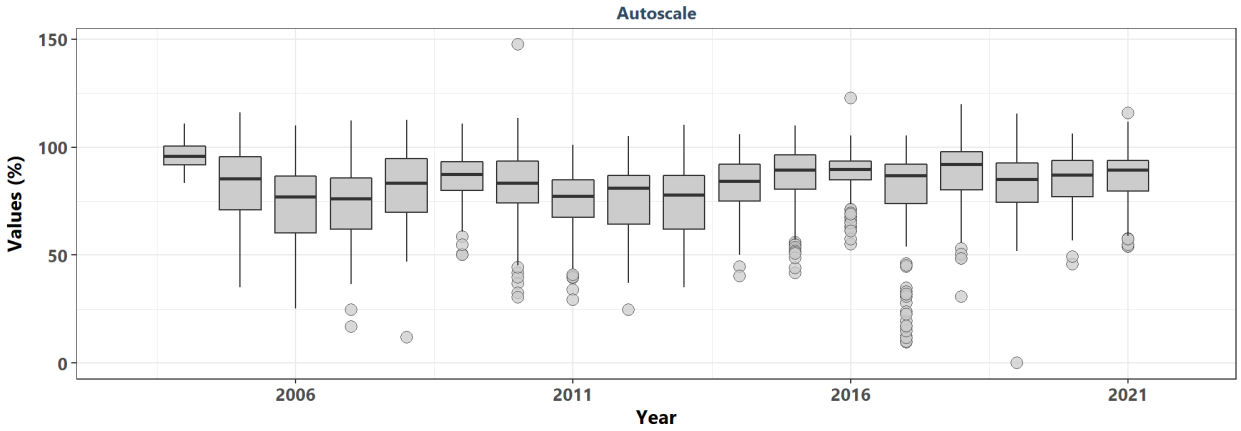
Estero Bay Aquatic Preserve
474 | Estero Bay Aquatic Preserve Continuous Water Quality Monitoring
EB02
By Year

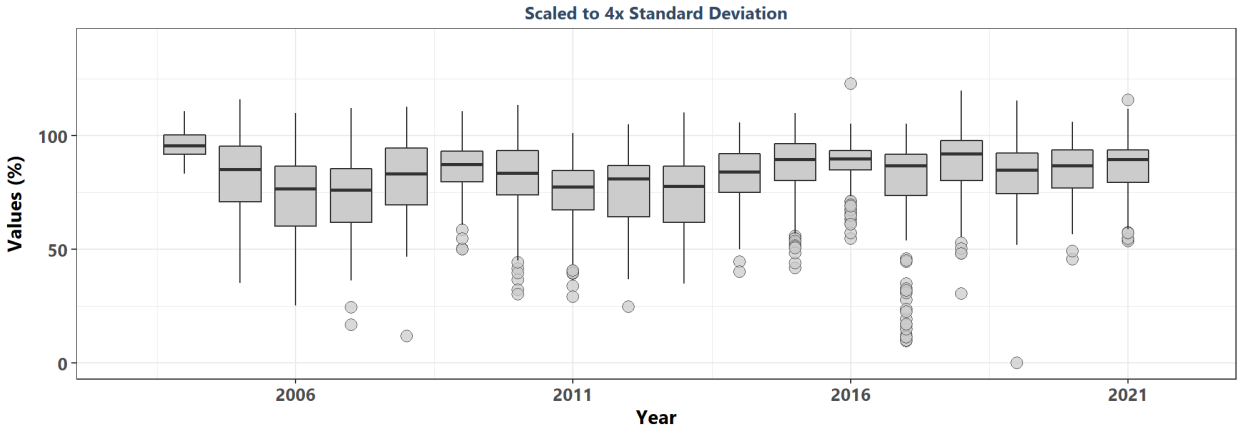


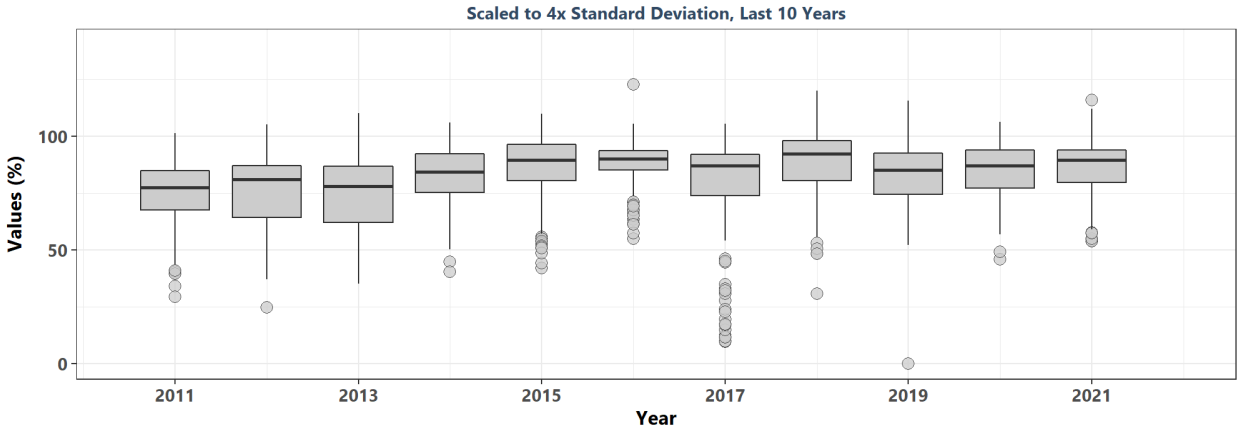

Estero Bay Aquatic Preserve
474 | Estero Bay Aquatic Preserve Continuous Water Quality Monitoring
EB02
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
Year
Scaled to 4x Standard Deviation
Values (%)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (%)
Year

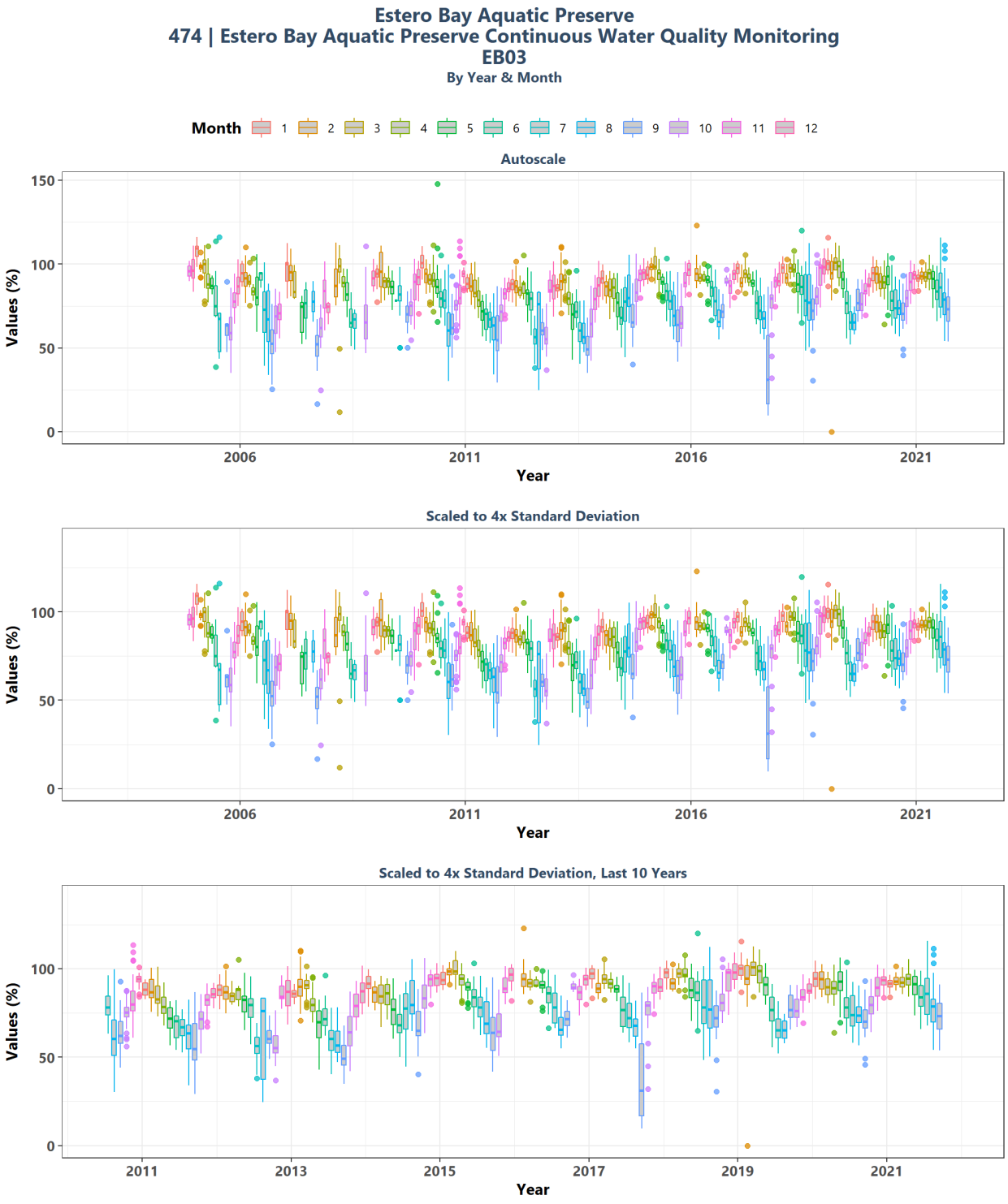

Estero Bay Aquatic Preserve
474 | Estero Bay Aquatic Preserve Continuous Water Quality Monitoring
EB02
By Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
100
50
0
3
6
9
12
Month
Scaled to 4x Standard Deviation
Values (%)
100
50
0
3
6
9
12
Month
Scaled to 4x Standard Deviation, Last 10 Years
Values (%)
100
50
0
3
6
9
12
Month

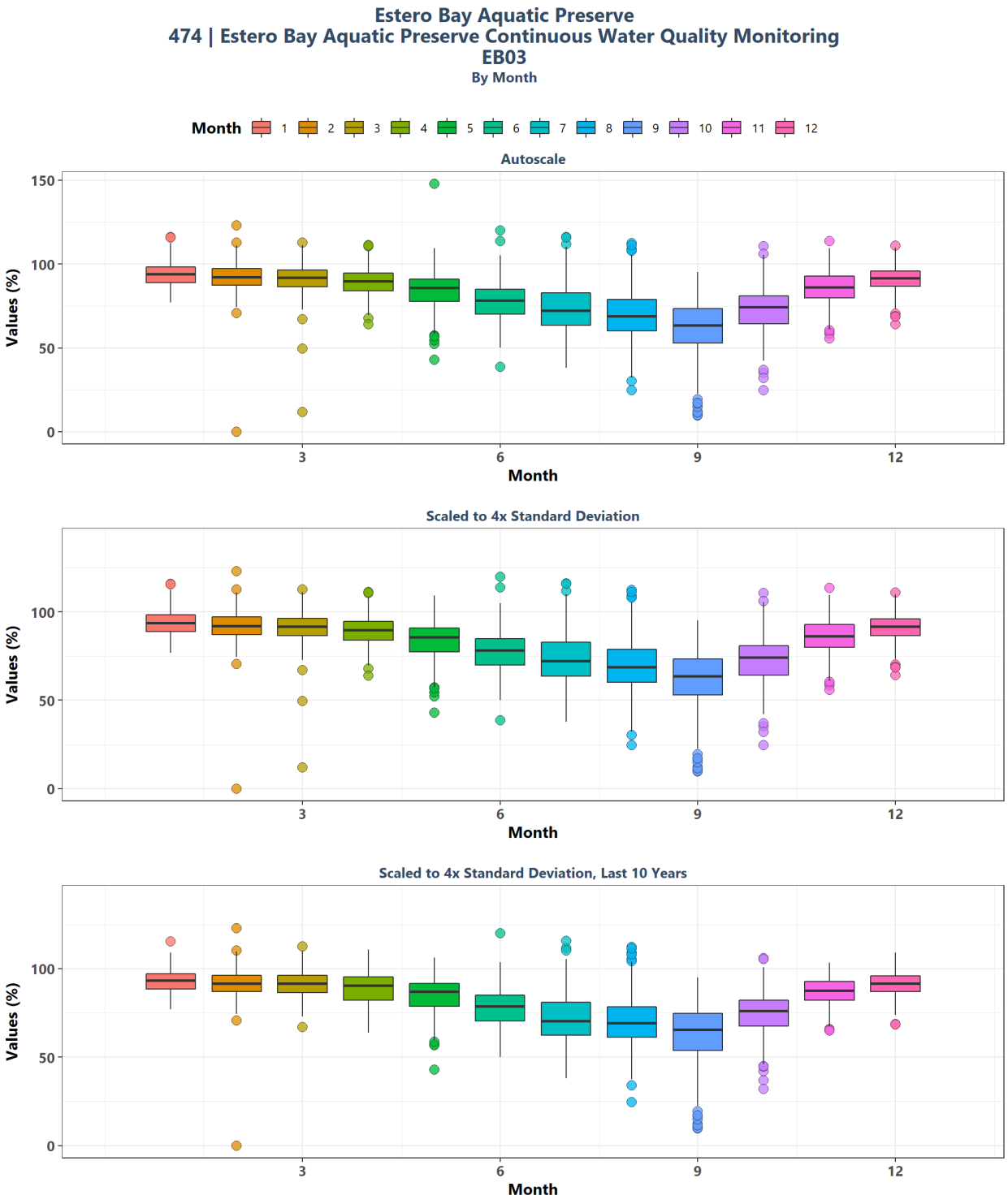
Estero Bay Aquatic Preserve
474 | Estero Bay Aquatic Preserve Continuous Water Quality Monitoring
EB03
By Year



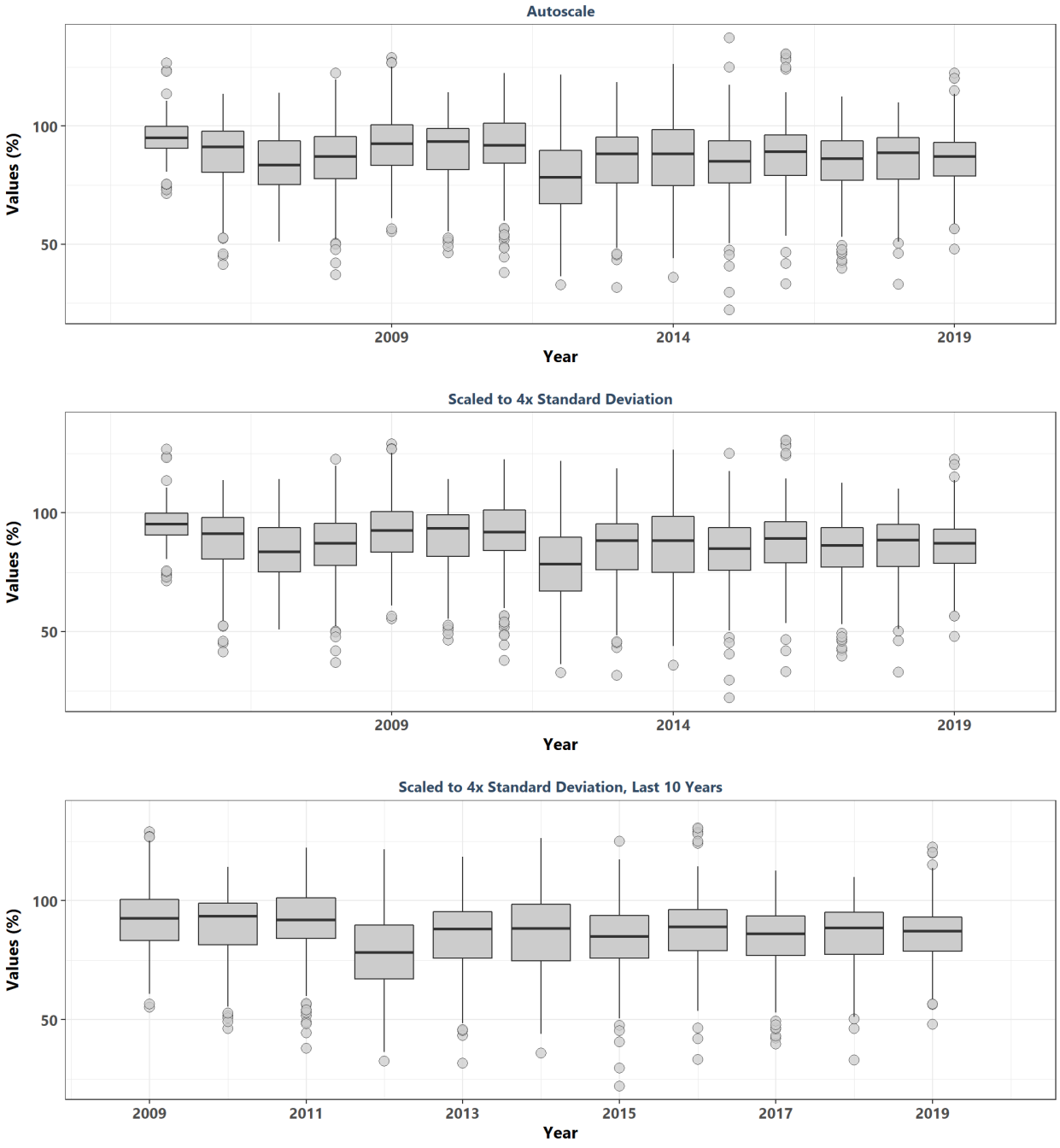


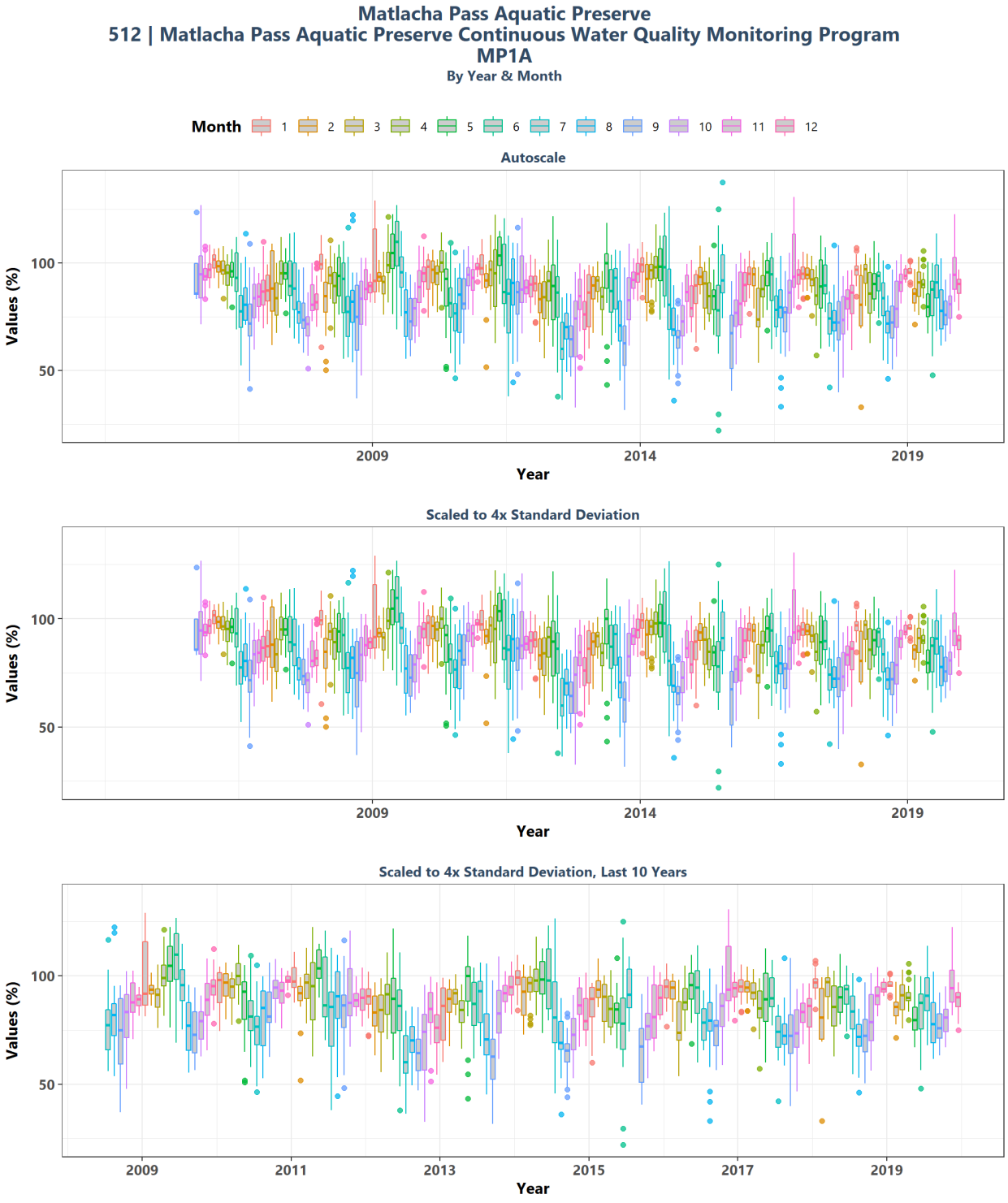



Estero Bay Aquatic Preserve
474 | Estero Bay Aquatic Preserve Continuous Water Quality Monitoring
EB03
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
Year
Scaled to 4x Standard Deviation
Values (%)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (%)
Year

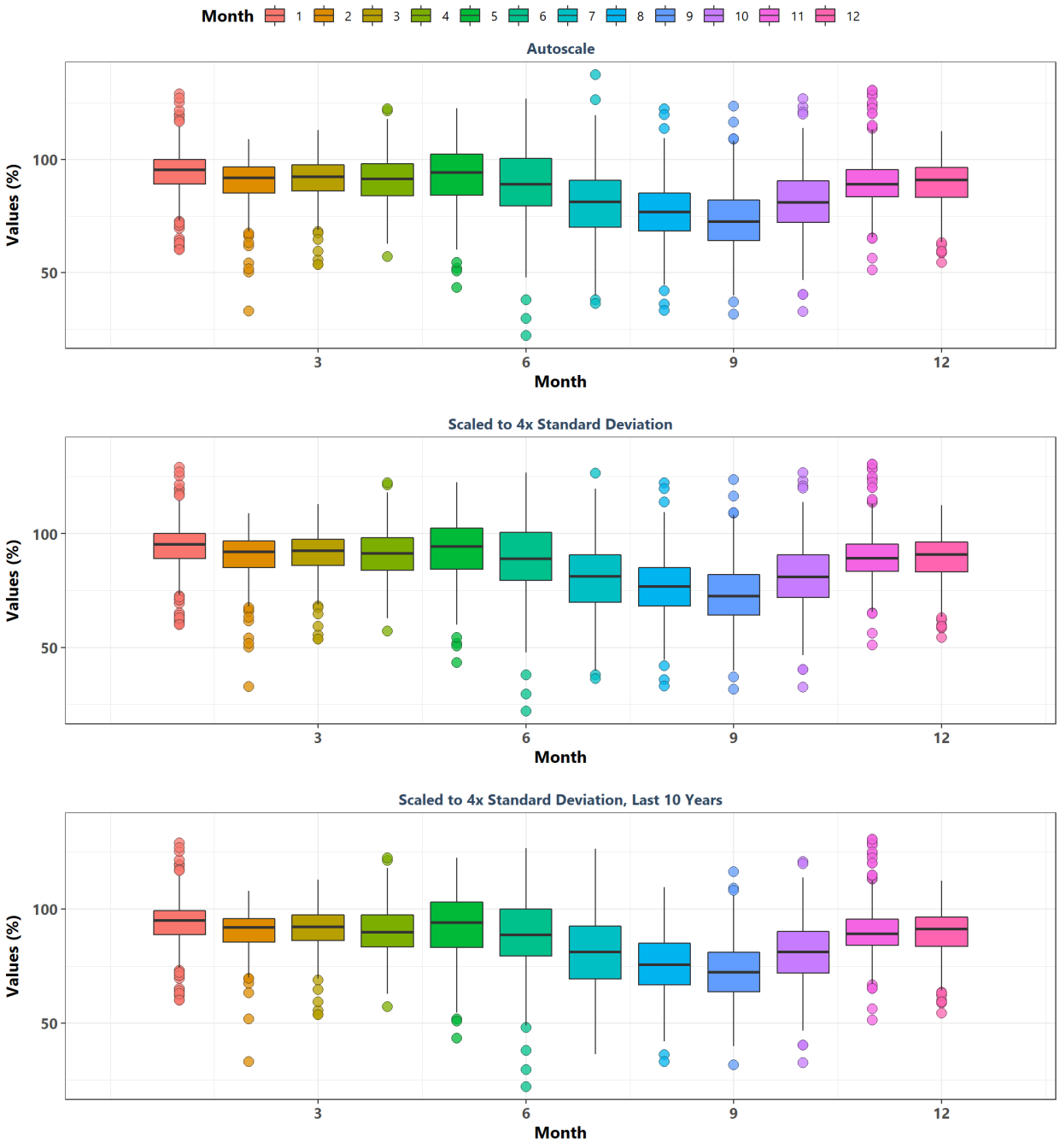

Estero Bay Aquatic Preserve
474 | Estero Bay Aquatic Preserve Continuous Water Quality Monitoring
EB03
By Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
Month
Scaled to 4x Standard Deviation
Values (%)
Month
Scaled to 4x Standard Deviation, Last 10 Years
Values (%)
Month

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

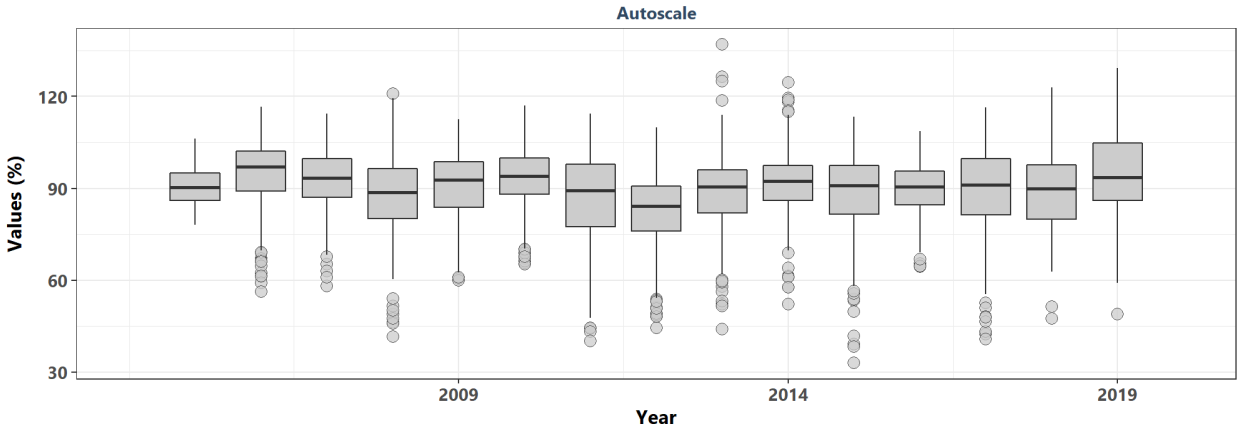


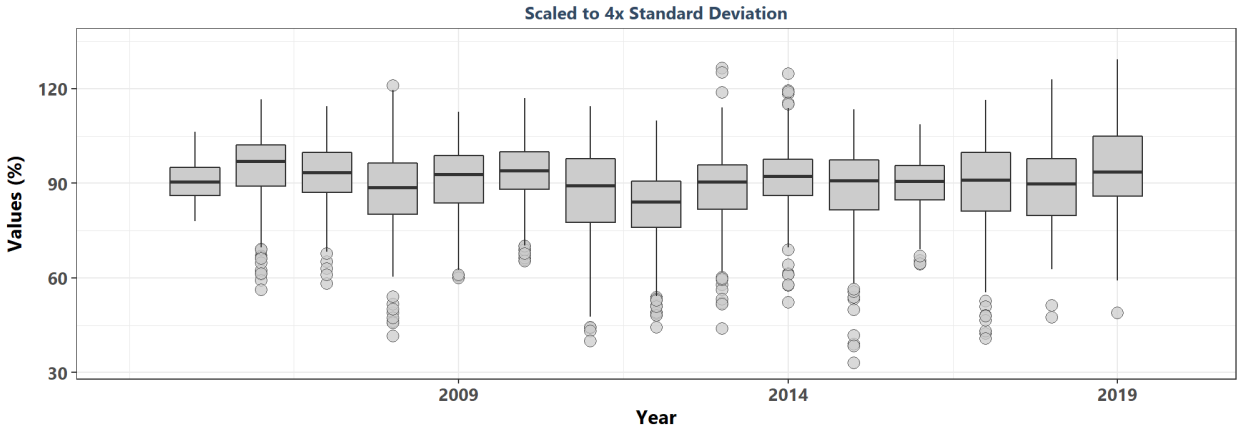

Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP1A
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
100
50
2009
2014
2019
Year
Scaled to 4x Standard Deviation
Values (%)
100
50
2009
2014
2019
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (%)
100
50
2009
2011
2013
2015
2017
2019
Year

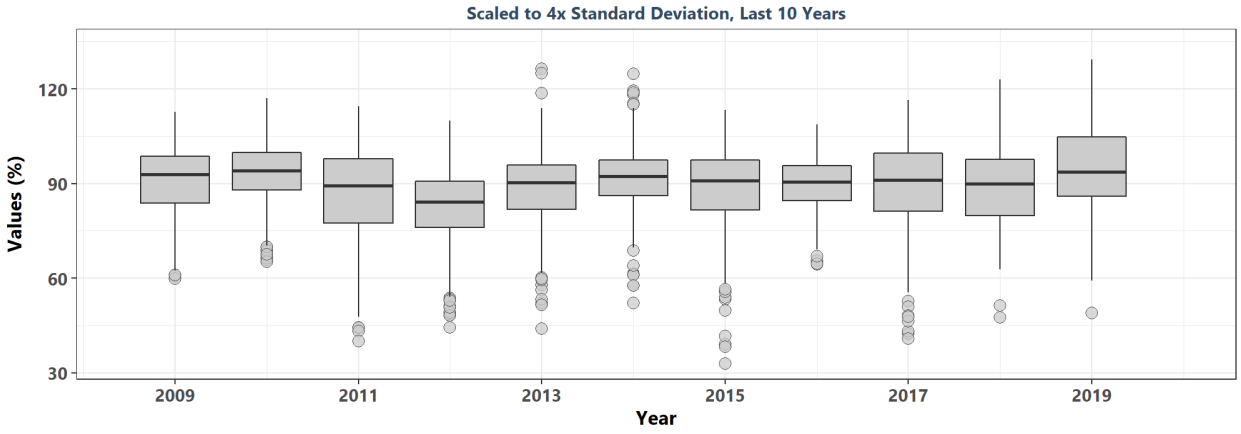
Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP1A
By Month

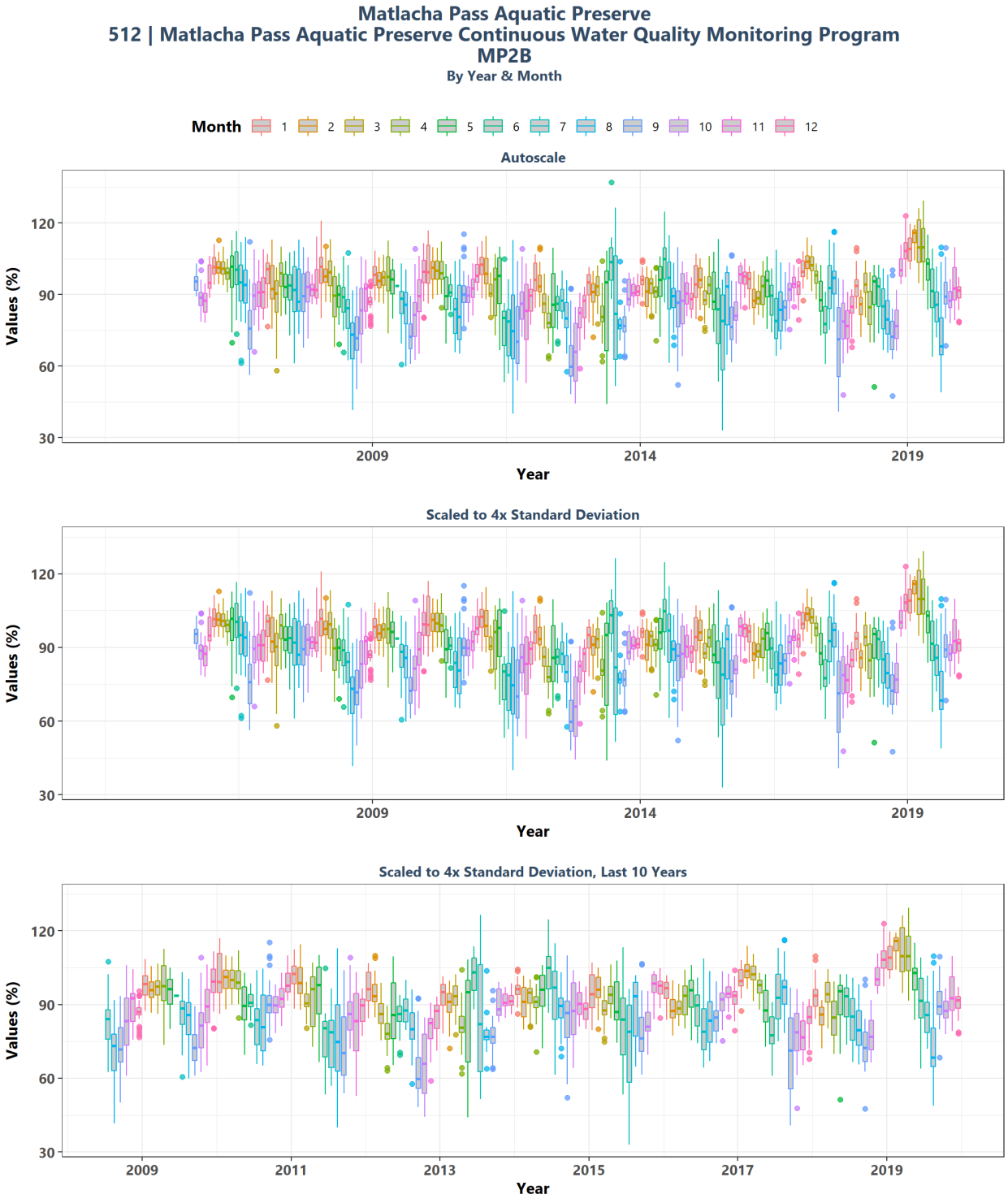


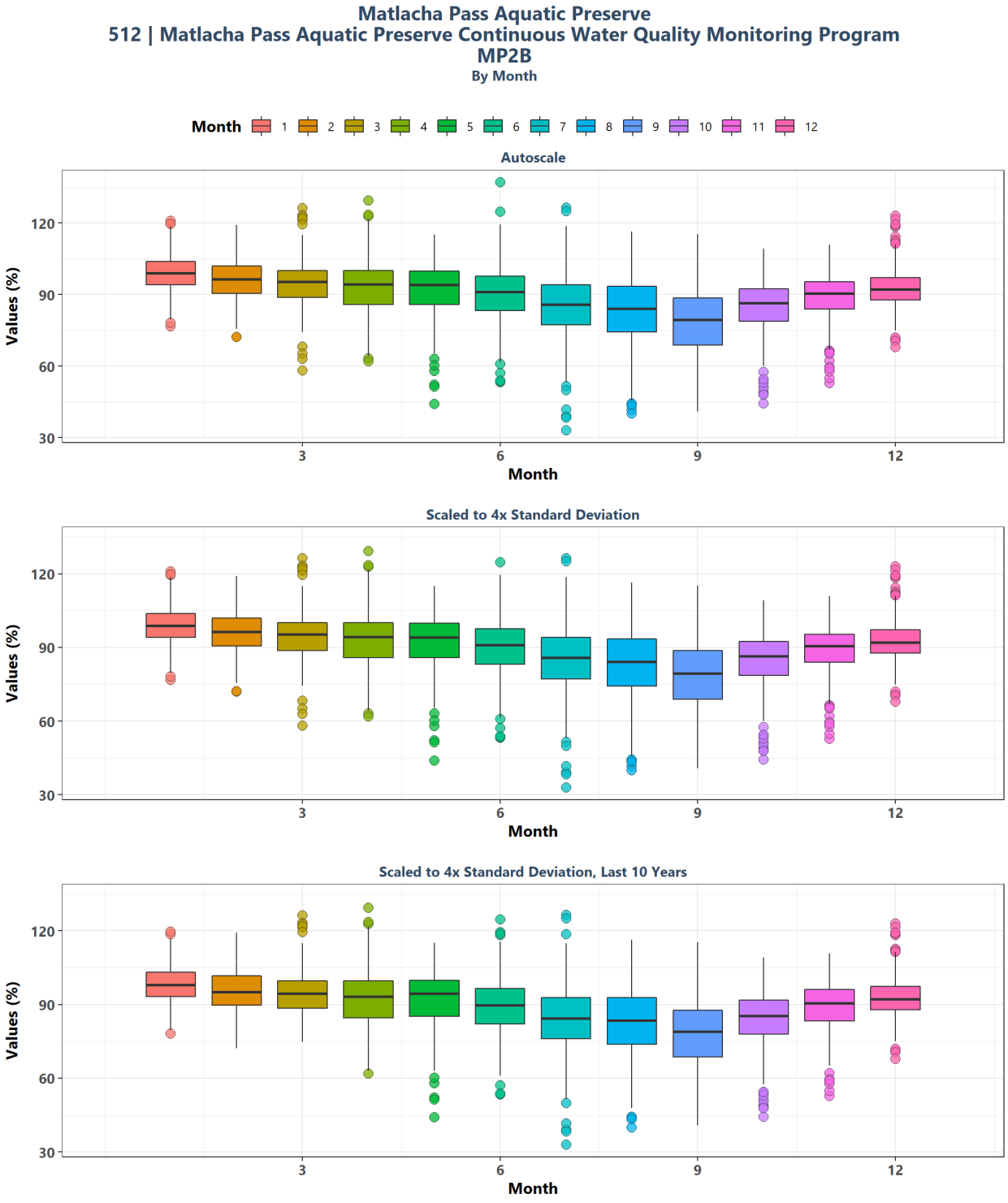
Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP2B
By Year



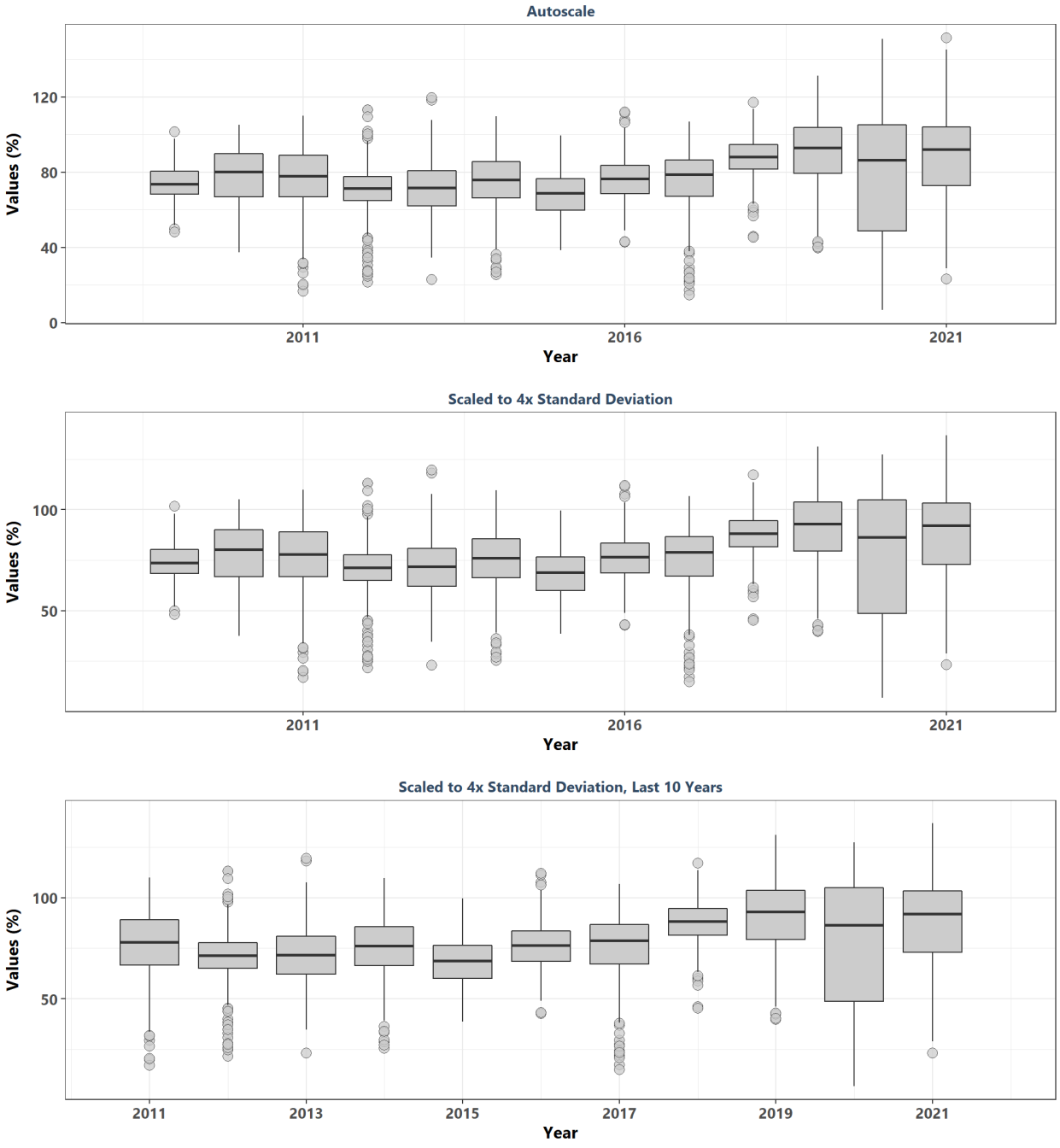


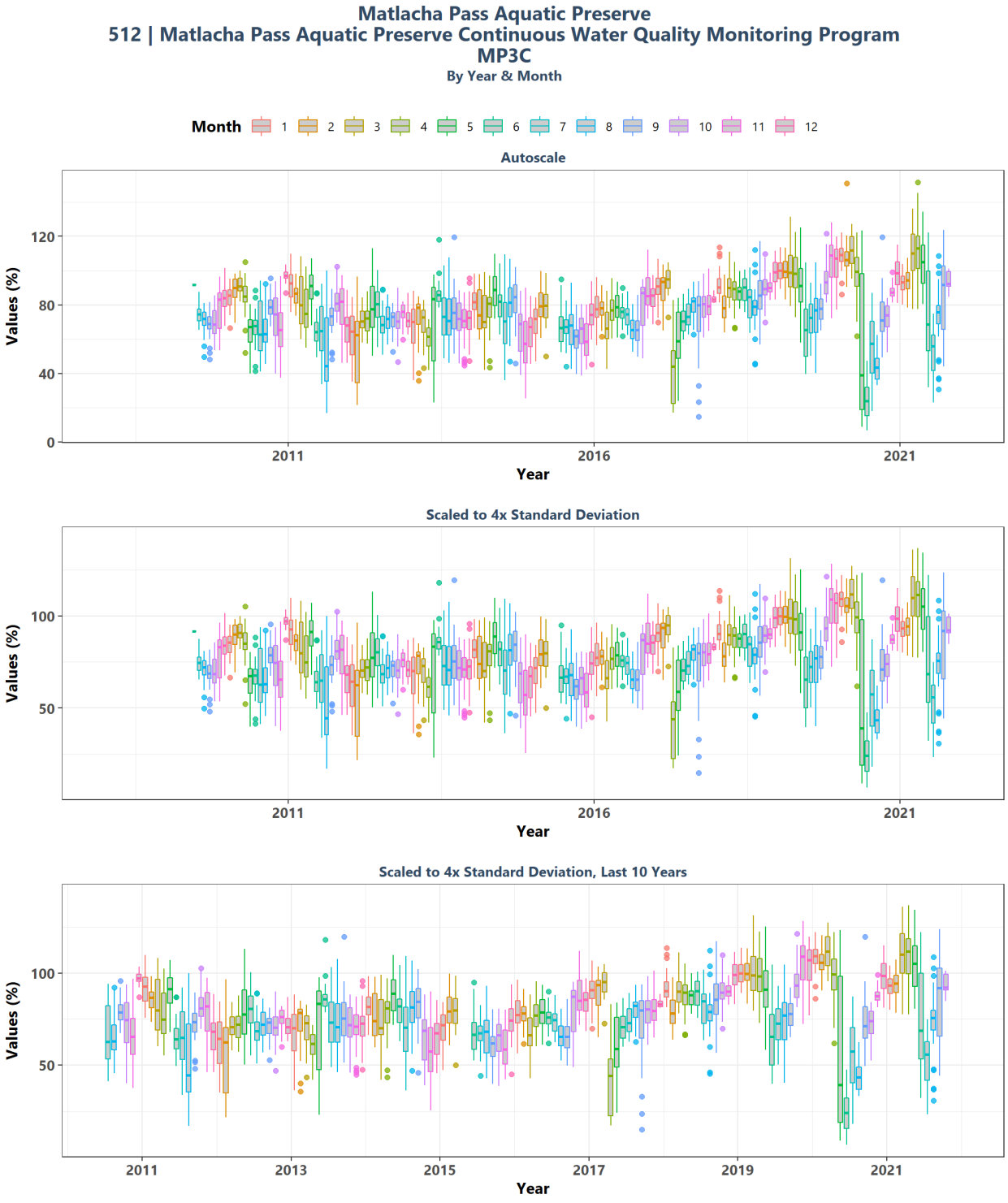



Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP2B
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
120
90
60
30
2009
2014
2019
Year
Scaled to 4x Standard Deviation
Values (%)
120
90
60
30
2009
2014
2019
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (%)
120
90
60
30
2009
2011
2013
2015
2017
2019
Year

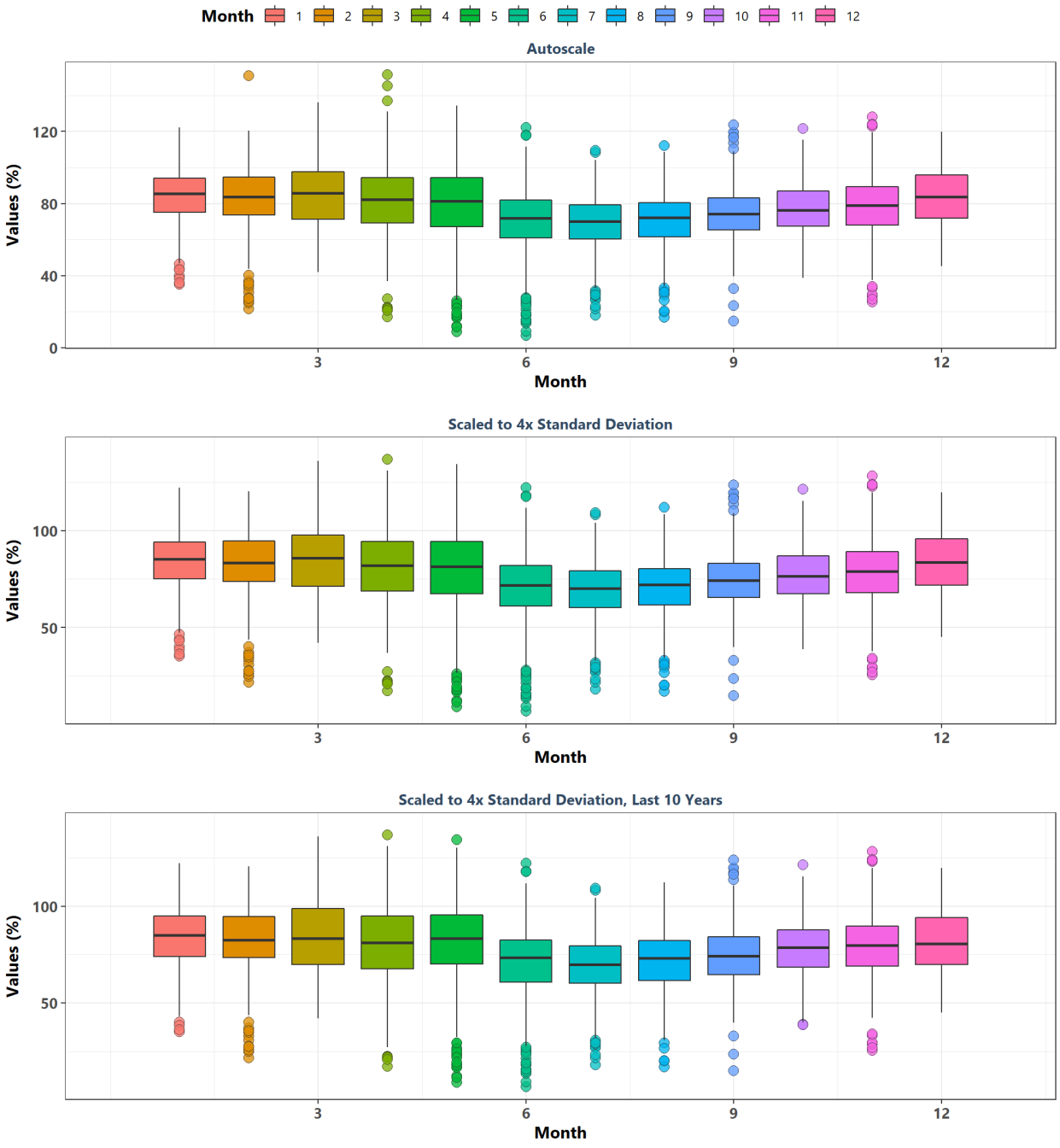

Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP2B
By Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
120
90
60
30
3
6
9
12
Month
Scaled to 4x Standard Deviation
Values (%)
120
90
60
30
3
6
9
12
Month
Scaled to 4x Standard Deviation, Last 10 Years
Values (%)
120
90
60
30
3
6
9
12
Month

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

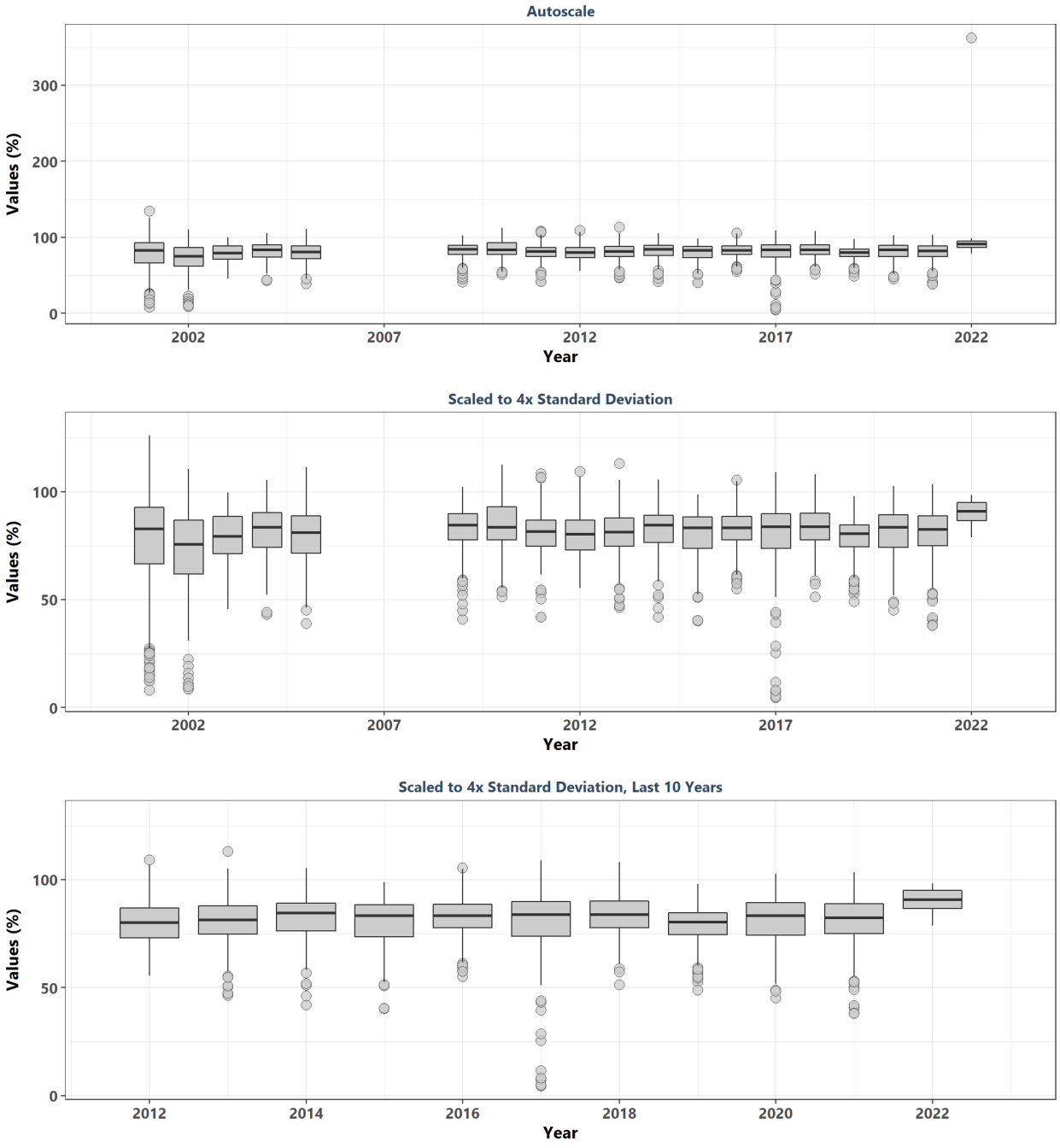


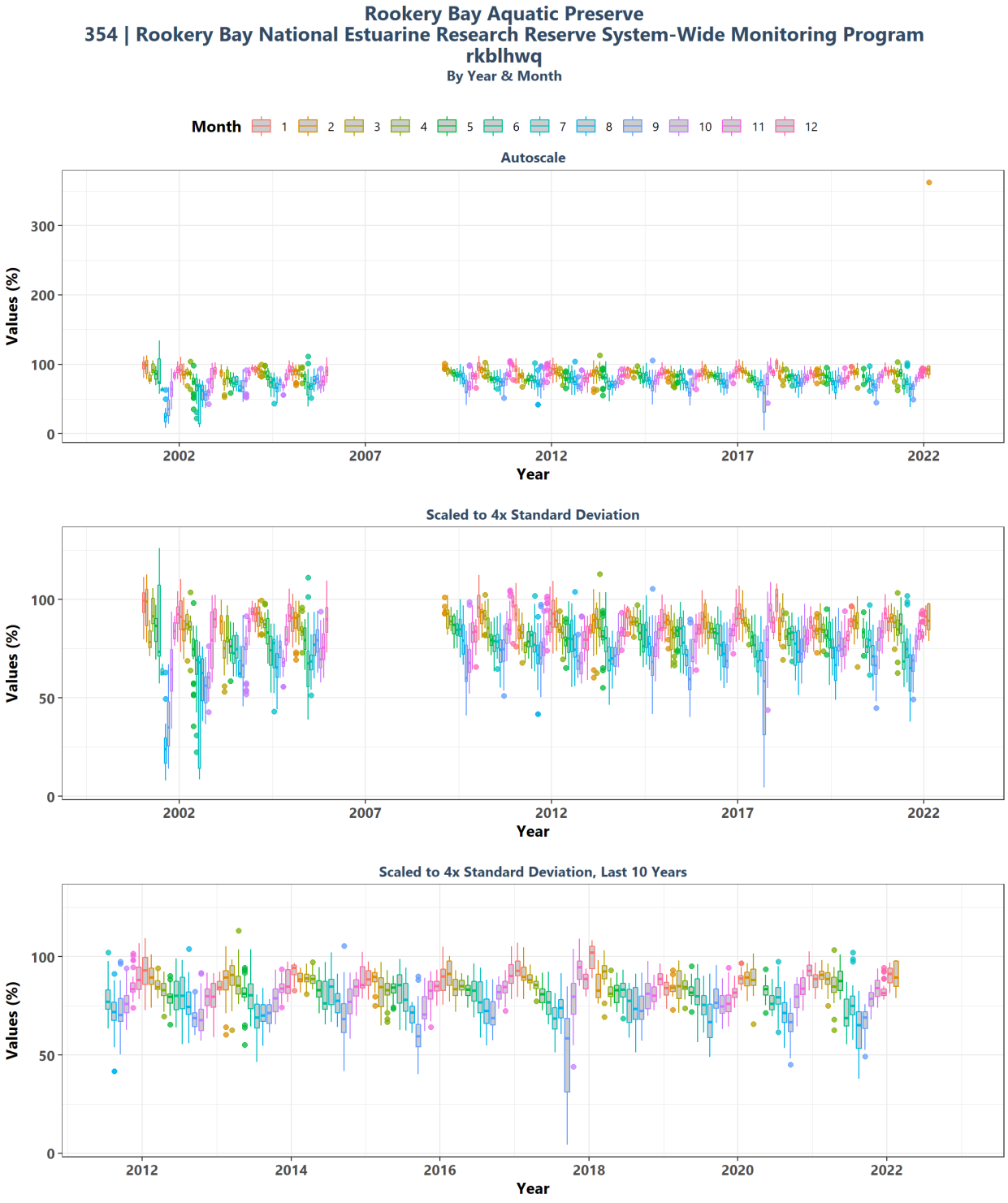

Matlacha Pass Aquatic Preserve
512 | Matlacha Pass Aquatic Preserve Continuous Water Quality Monitoring Program
MP3C
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
Year
Scaled to 4x Standard Deviation
Values (%)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (%)
Year

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

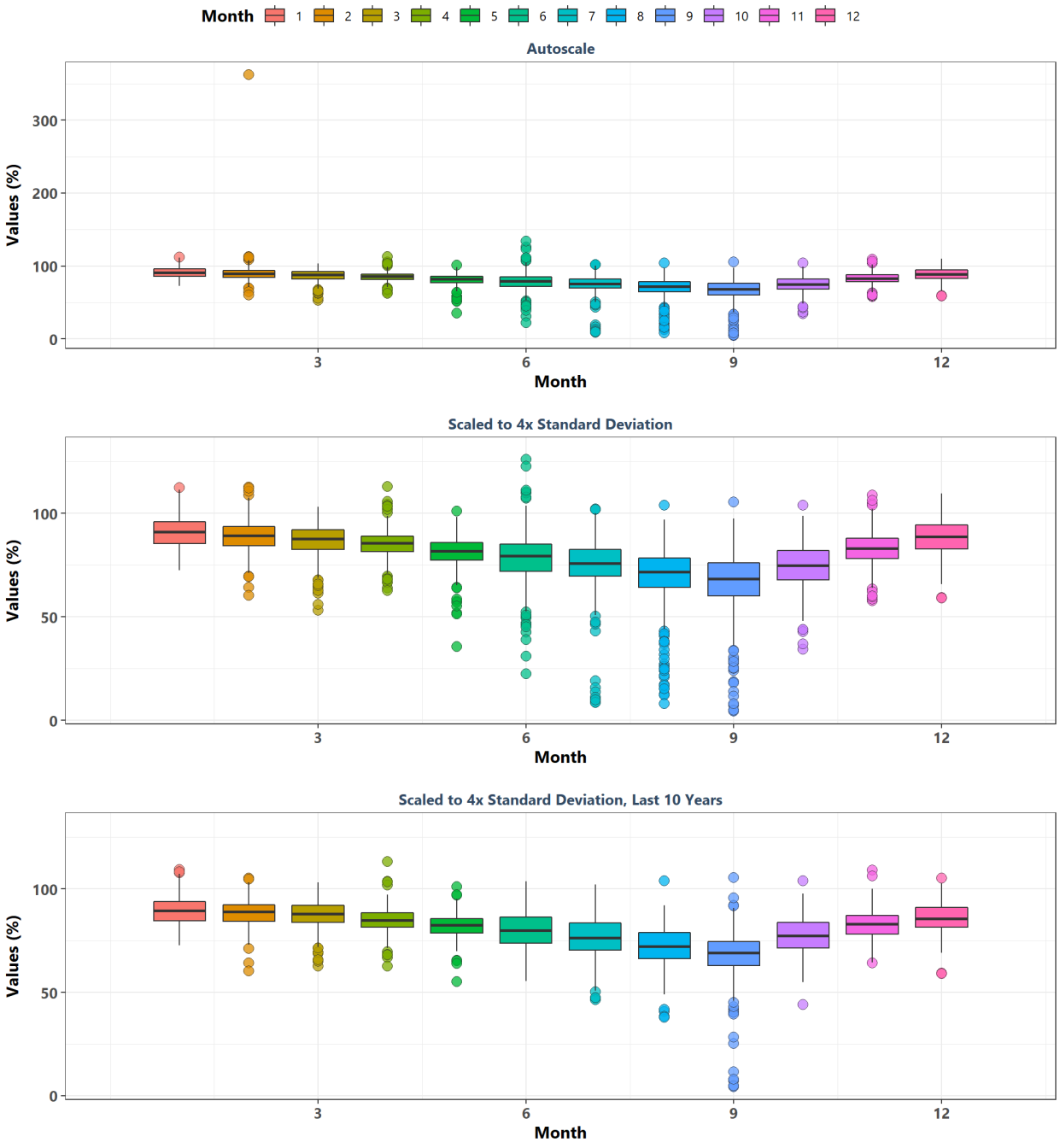


Rookery Bay Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkblhwq
By Year

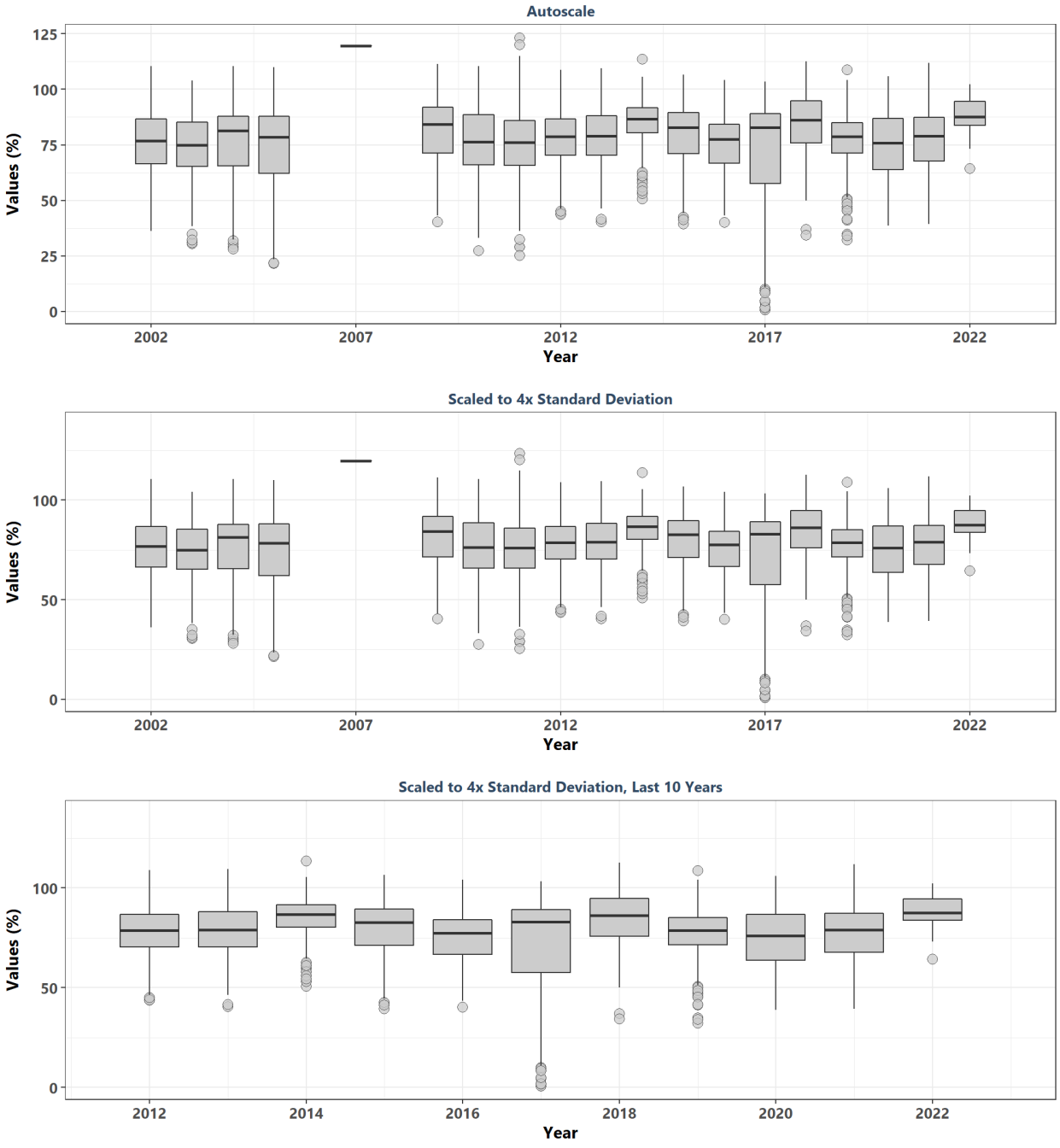


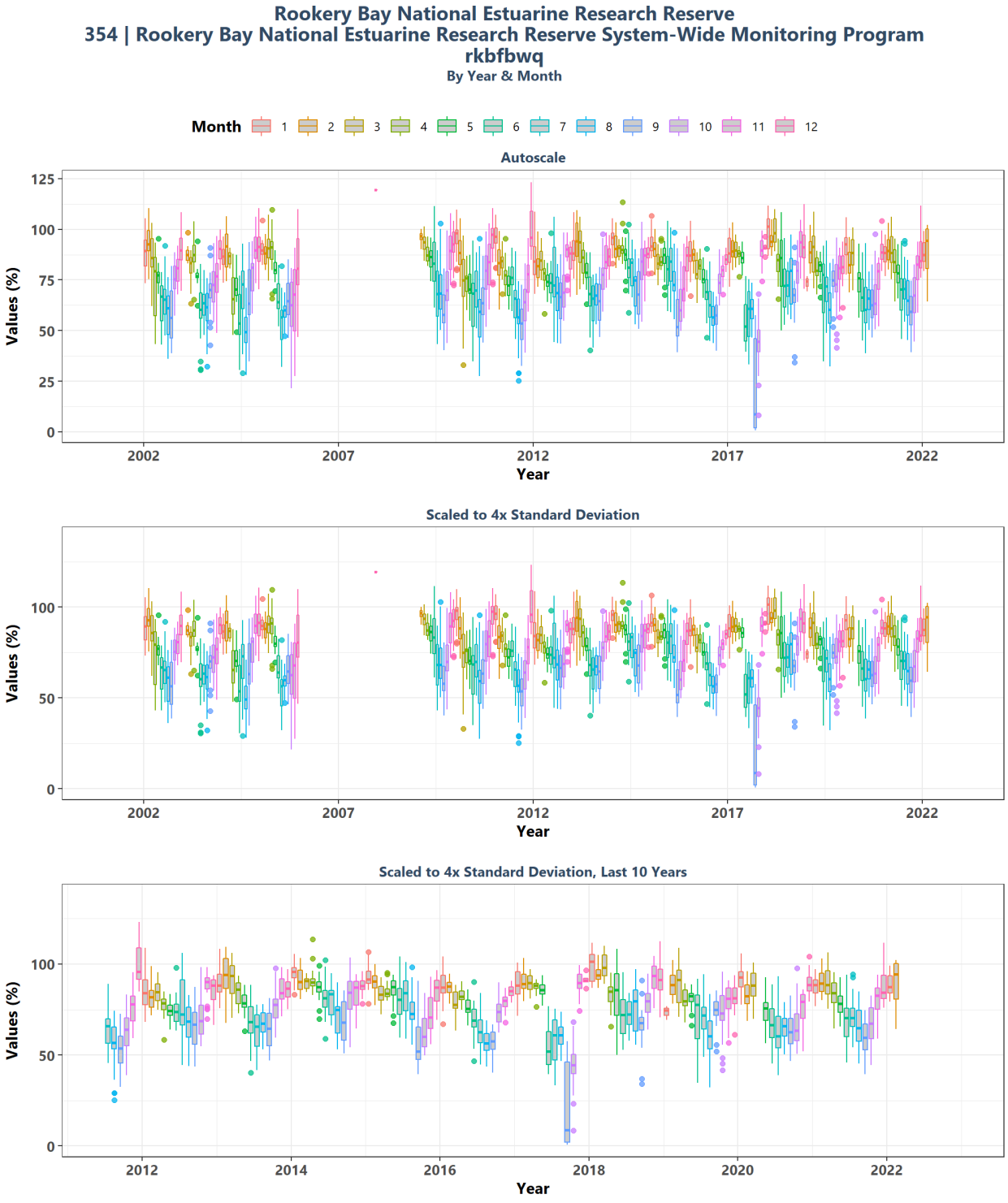

Rookery Bay Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkblhwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
Year
Scaled to 4x Standard Deviation
Values (%)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (%)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

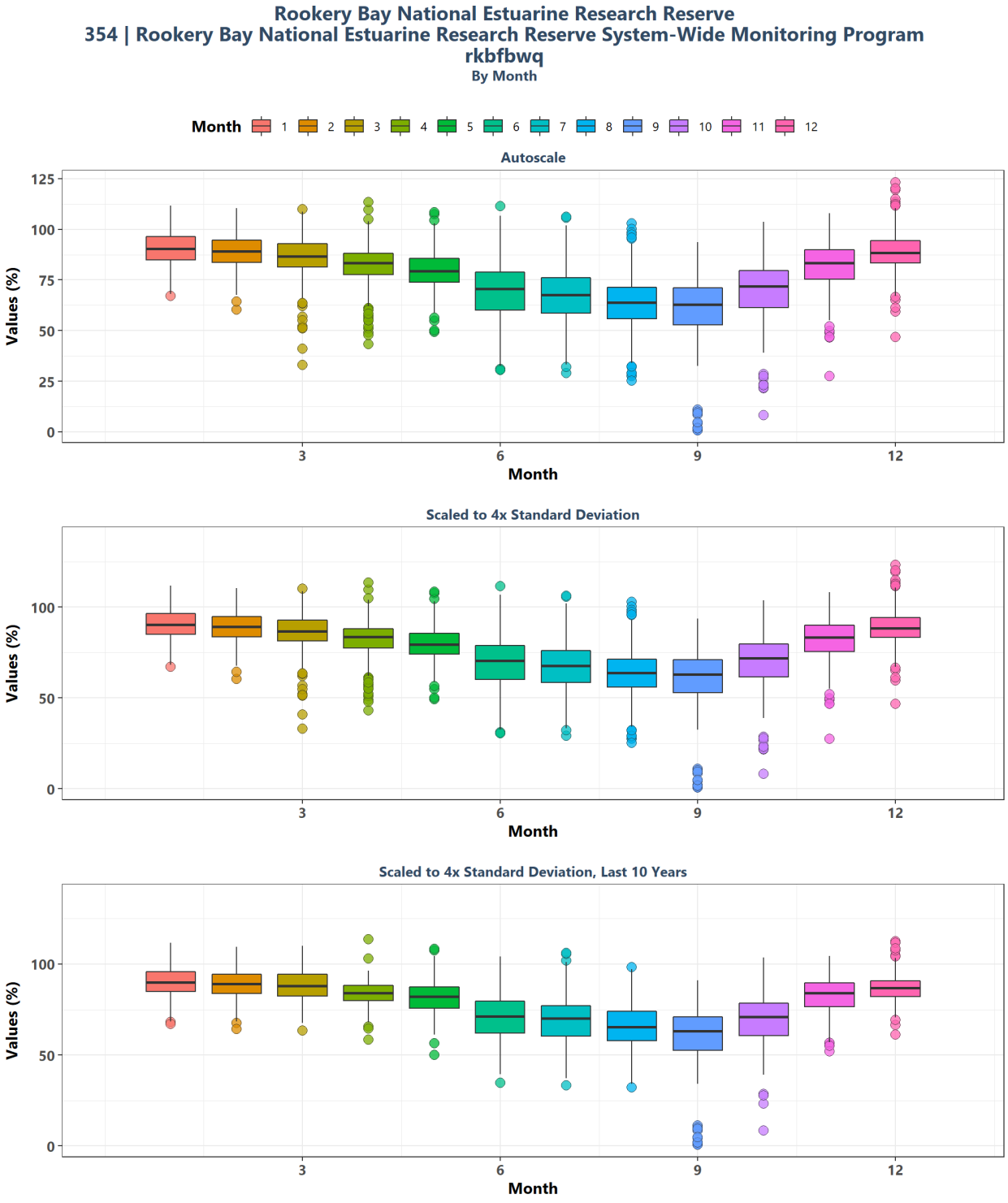
Rookery Bay Aquatic Preserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkblhwq
By Month



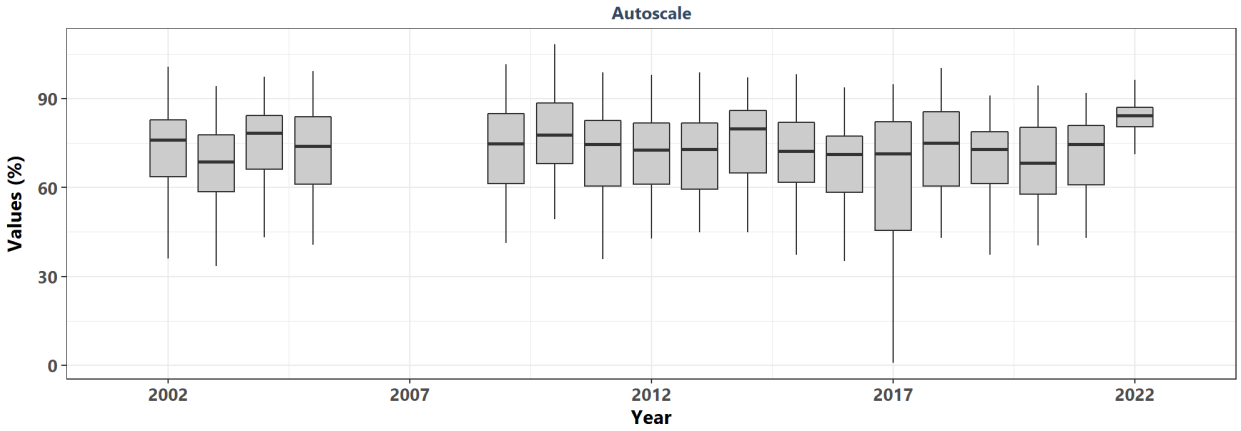
Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfbwq
By Year

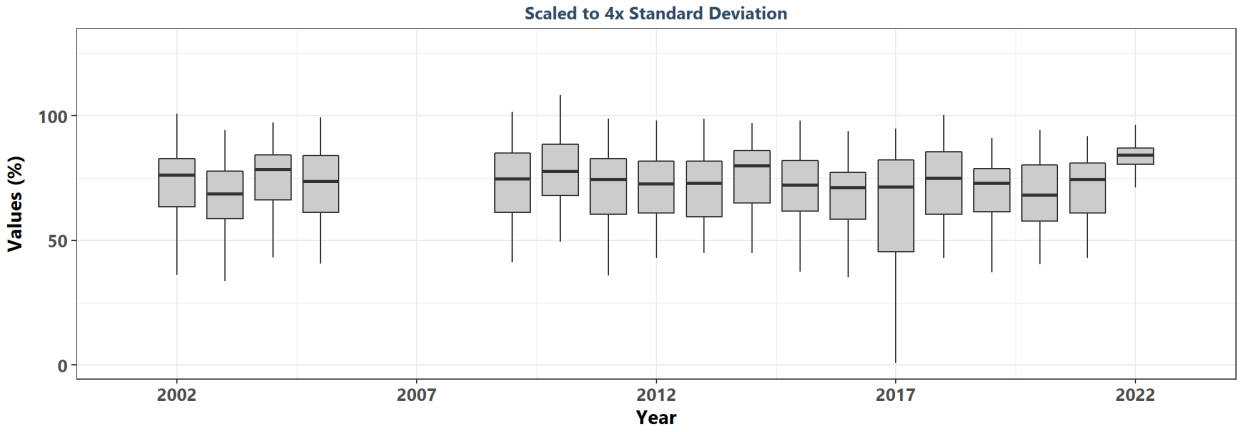


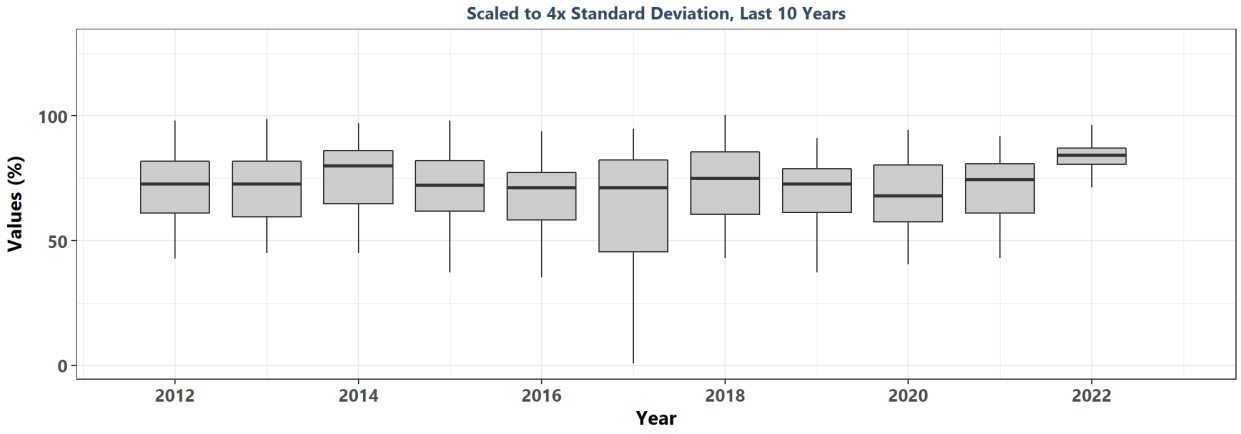

Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfbwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
Year
Scaled to 4x Standard Deviation
Values (%)
Year
Scaled to 4x Standard Deviation, Last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

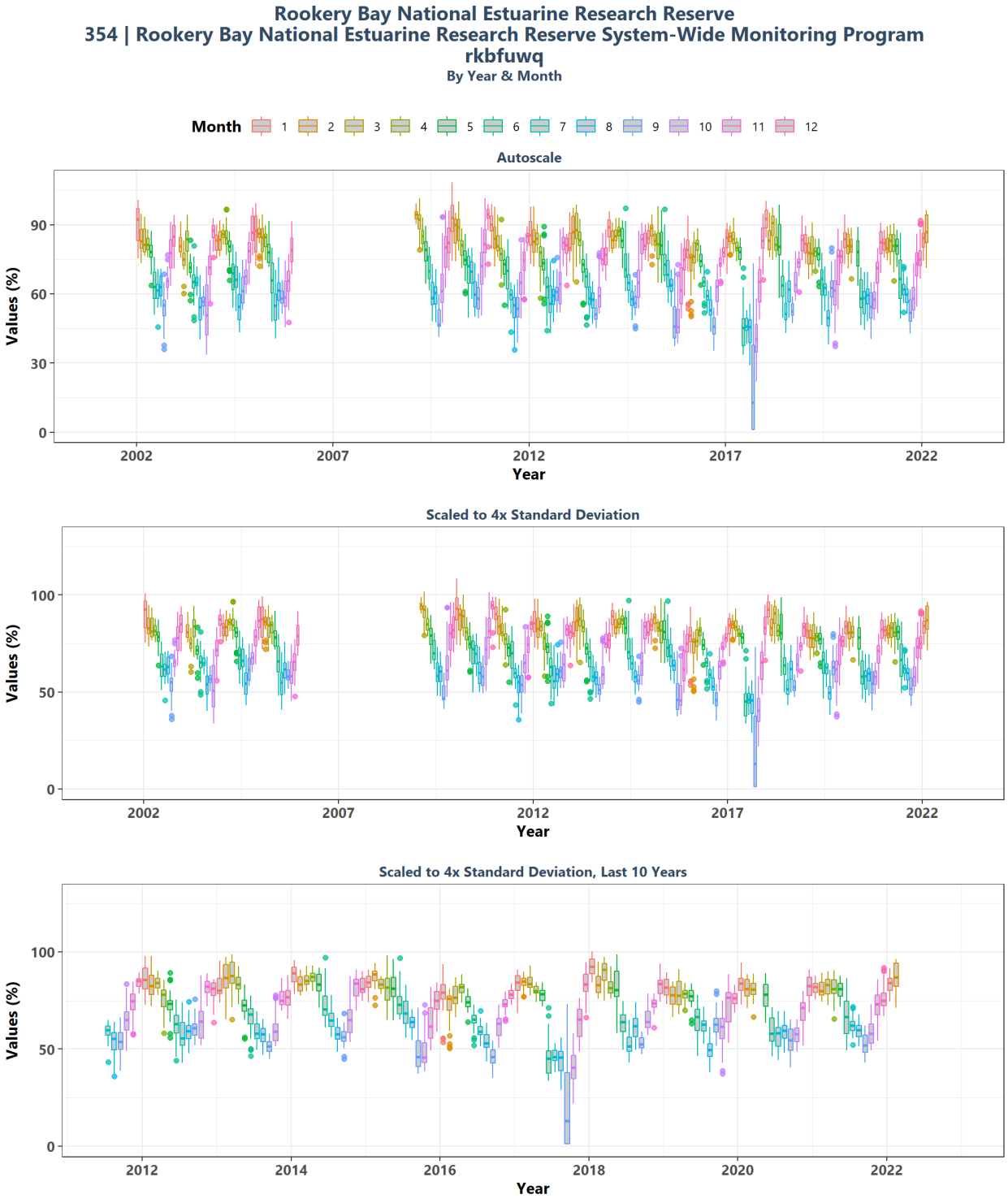

Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfbwq
By Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
125
100
75
50
25
0
3
6
9
12
Month
Scaled to 4x Standard Deviation
Values (%)
100
50
0
3
6
9
12
Month
Scaled to 4x Standard Deviation, Last 10 Years
Values (%)
100
50
0
3
6
9
12
Month

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

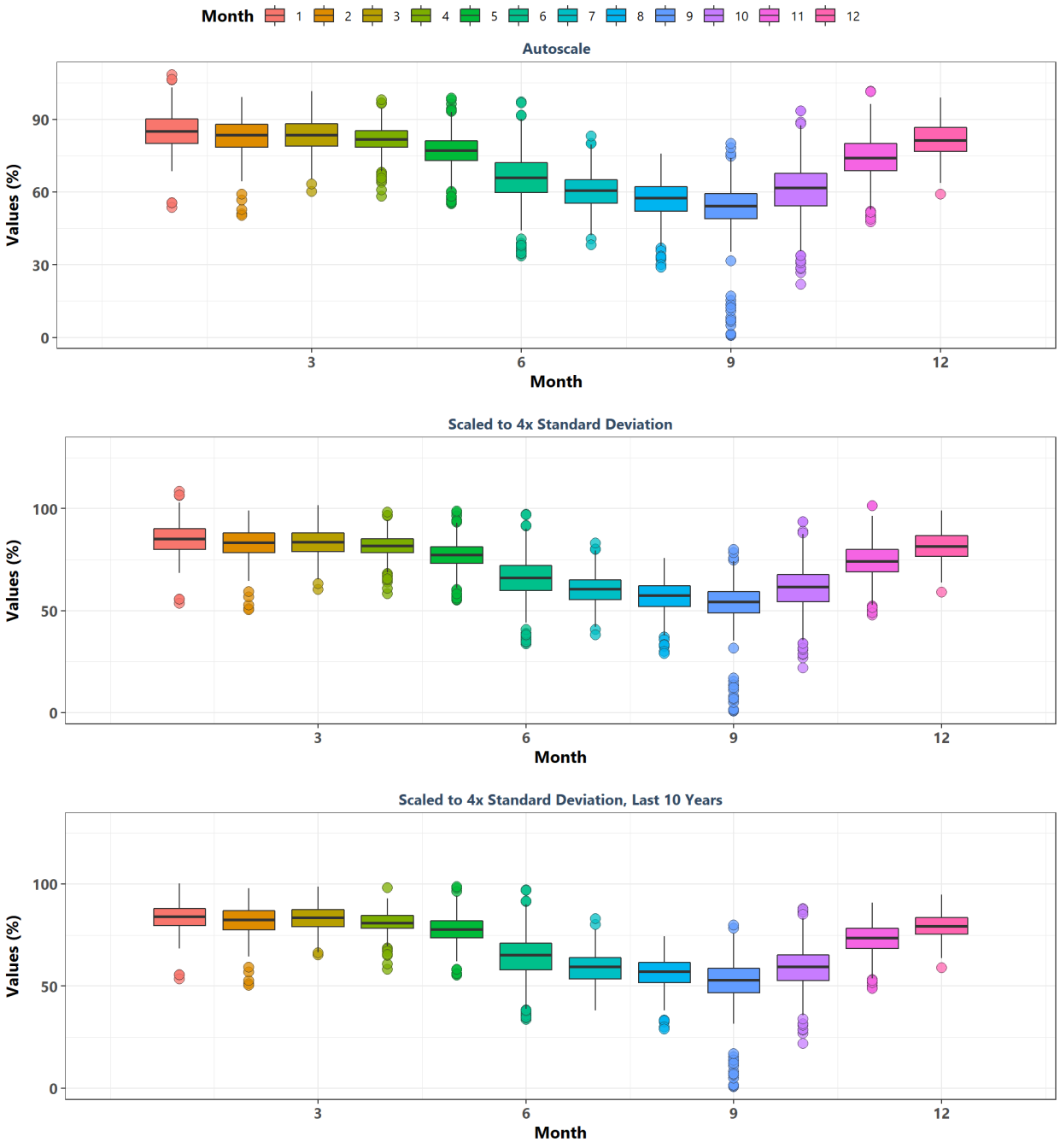




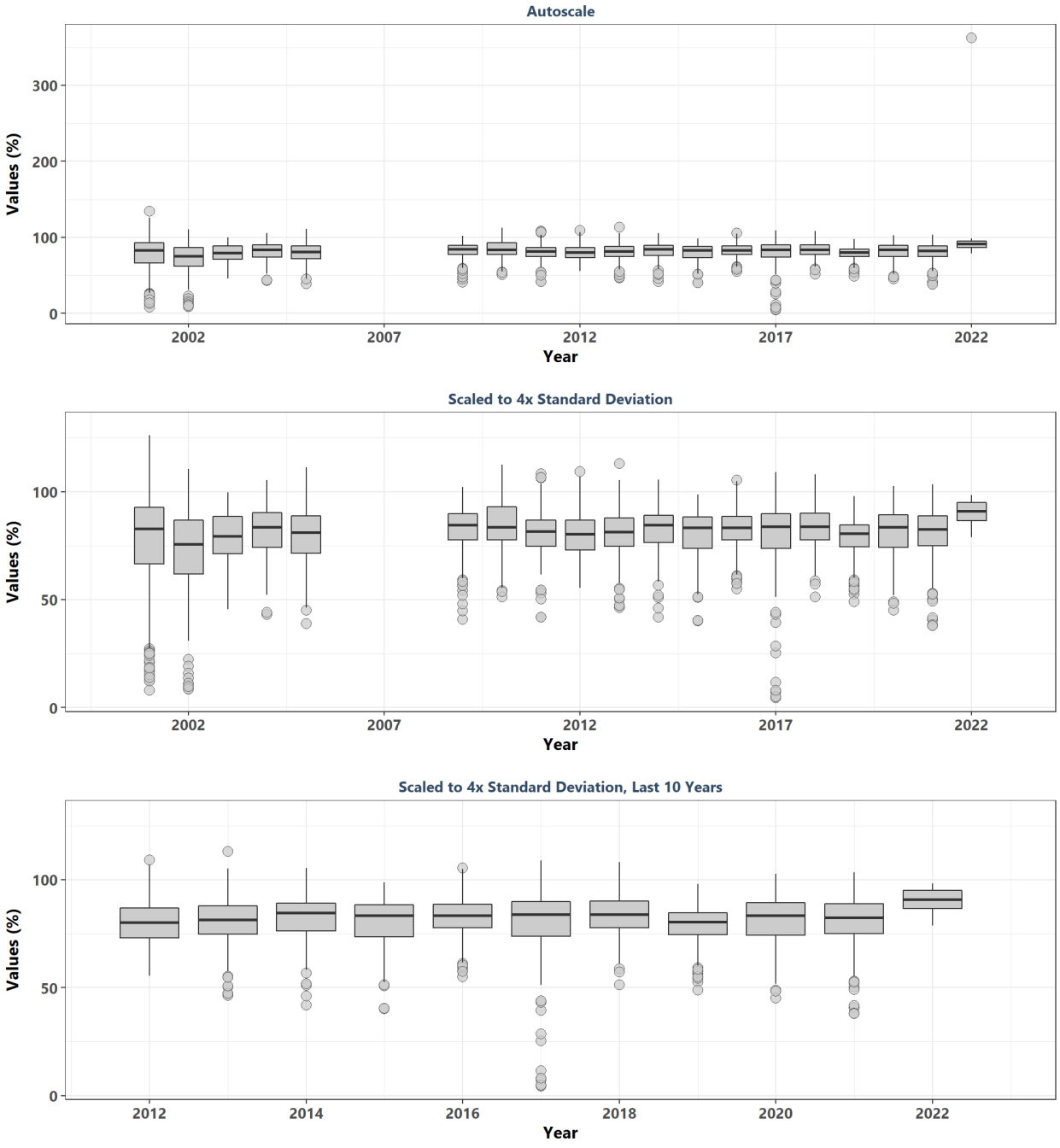


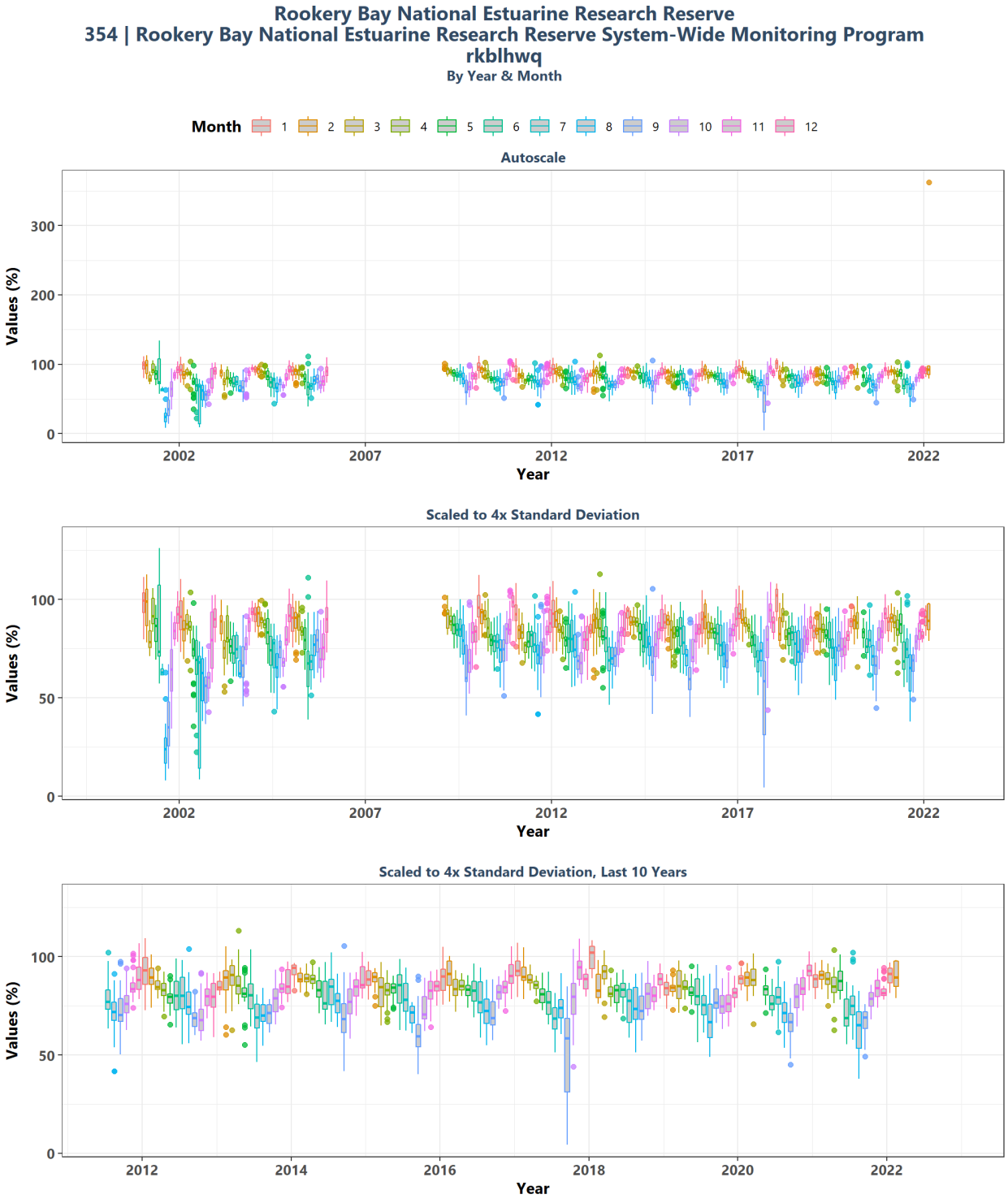

Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkbfuwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
Year
Scaled to 4x Standard Deviation
Values (%)
Year
Scaled to 4x Standard Deviation, Last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

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

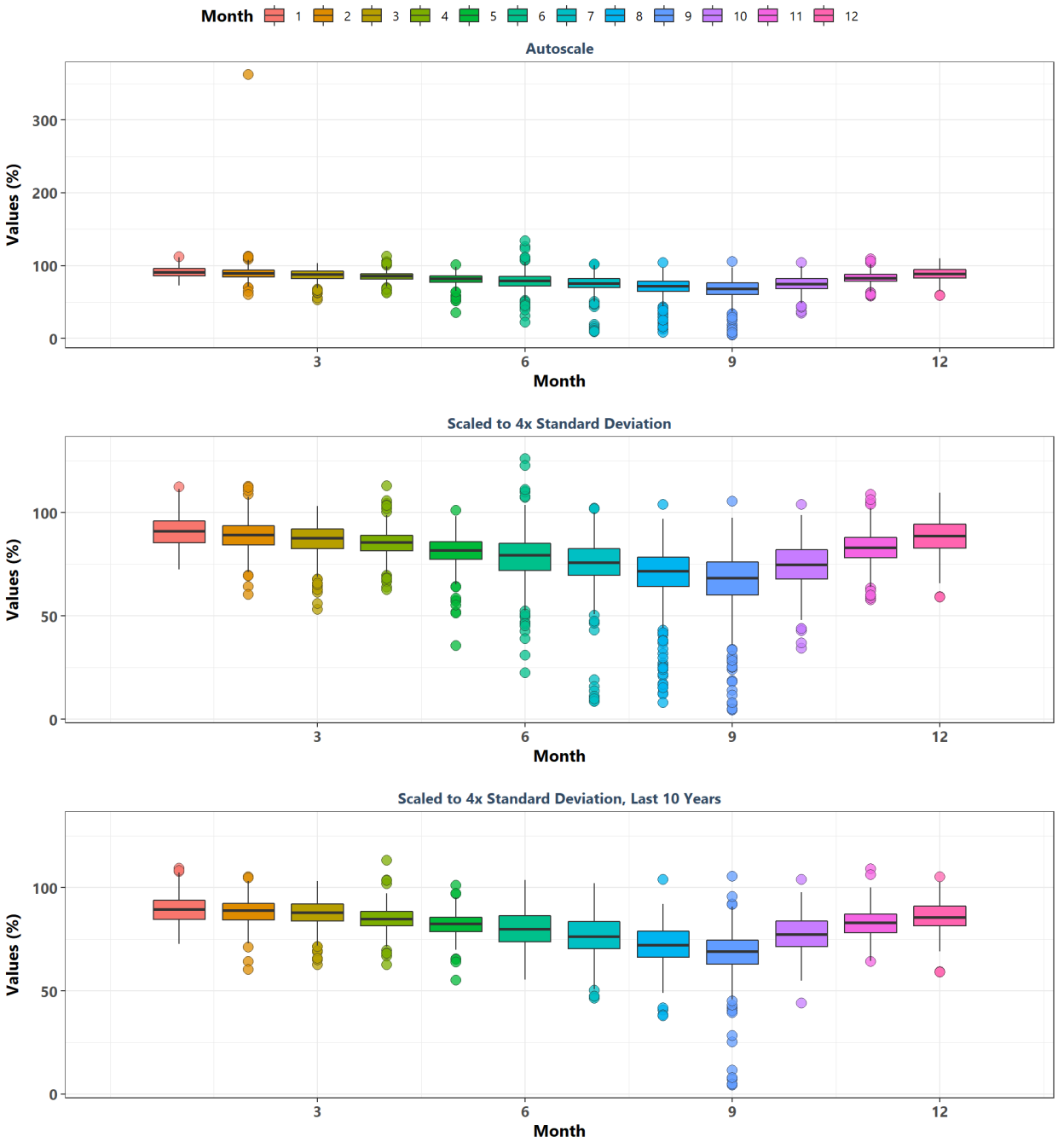


Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkblhwq
By Year

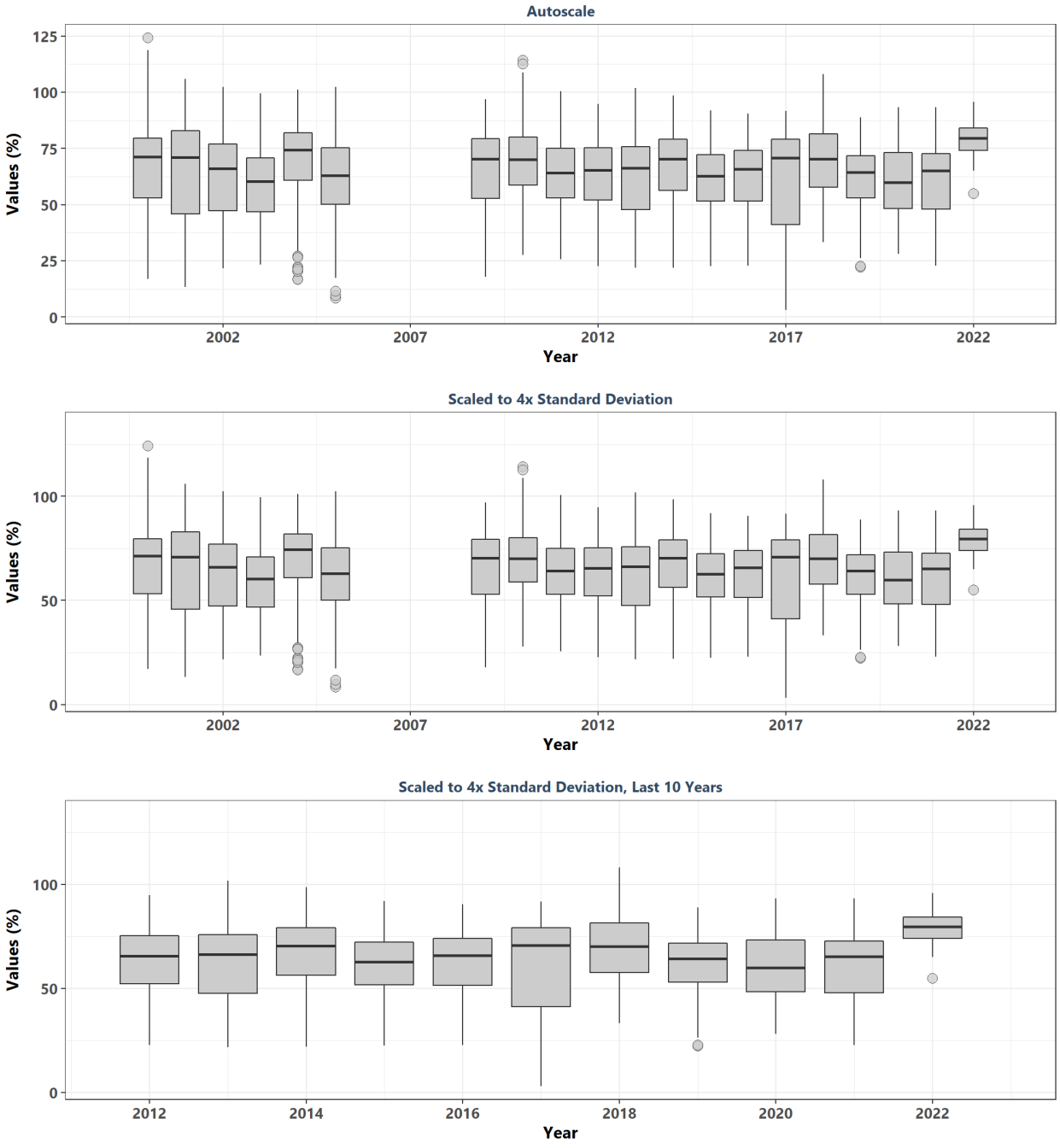


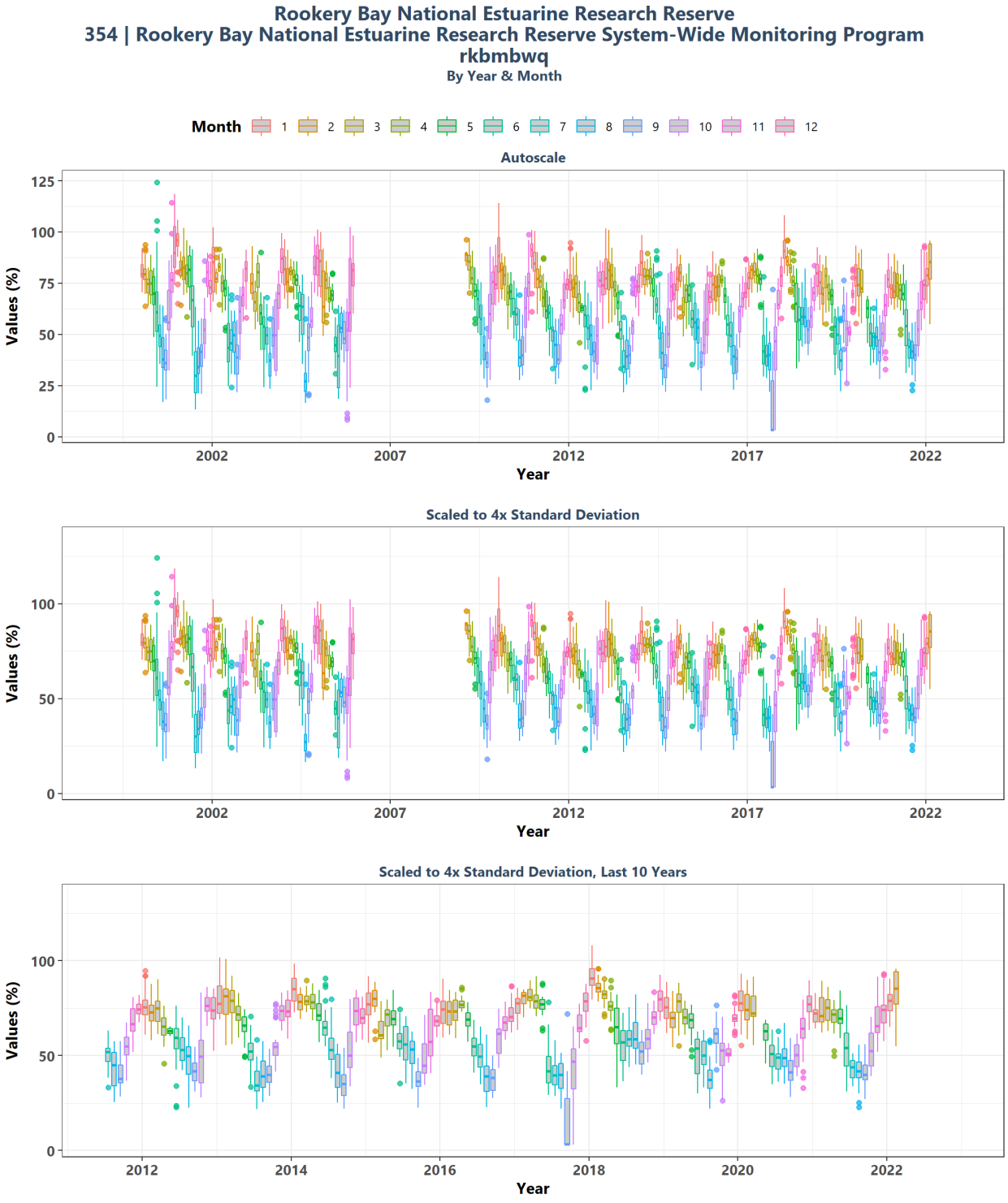

Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkblhwq
By Year & Month
Month 1 2 3 4 5 6 7 8 9 10 11 12
Autoscale
Values (%)
Year
Scaled to 4x Standard Deviation
Values (%)
Year
Scaled to 4x Standard Deviation, Last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

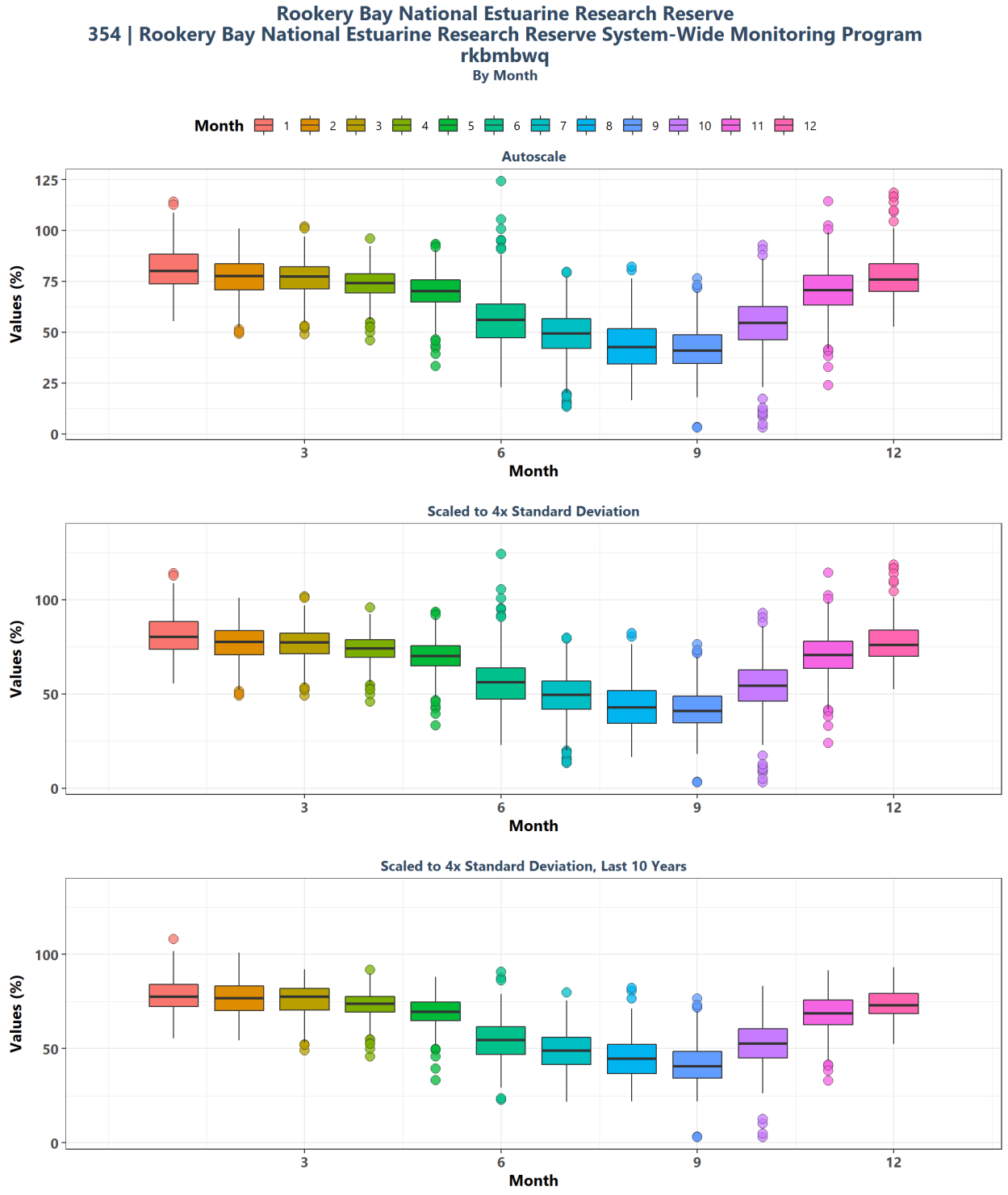
Rookery Bay National Estuarine Research Reserve
354 | Rookery Bay National Estuarine Research Reserve System-Wide Monitoring Program
rkblhwq
By Month



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 (%)
Year
Scaled to 4x Standard Deviation
Values (%)
Year
Scaled to 4x Standard Deviation, Last 10 Years
Values (%)
Year




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