SEACAR Coral Analysis: Percent Cover

Last compiled on 09 July, 2025

Contents

Important Notes

The purpose of this script is to create managed area statistics, perform linear mixed effect analysis, generate summary plots, and create reports in pdf and Word document form for Coral percent cover.

These scripts were created by J.E. Panzik (jepanzik@usf.edu) for SEACAR. Updated by T.G. Hill (Tyler.Hill@FloridaDEP.gov).

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

https://github.com/FloridaSEACAR/SEACAR_Trend_Analyses

This markdown file is designed to be compiled by Coral_PC_ReportRender.R (https://github.com/FloridaSEACAR/SEACAR_Trend_Analyses/blob/main/Coral/Coral_PC_ReportRender.R).

Coral Percent Cover

Biscayne Bay-Cape Florida to Monroe County Line Aquatic Preserve

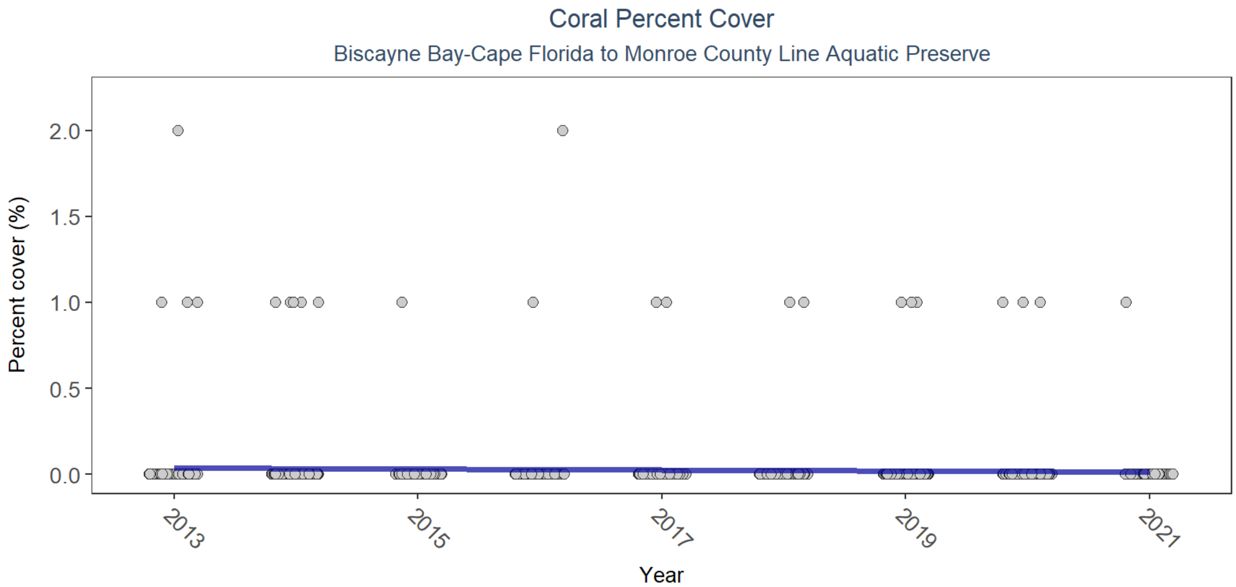
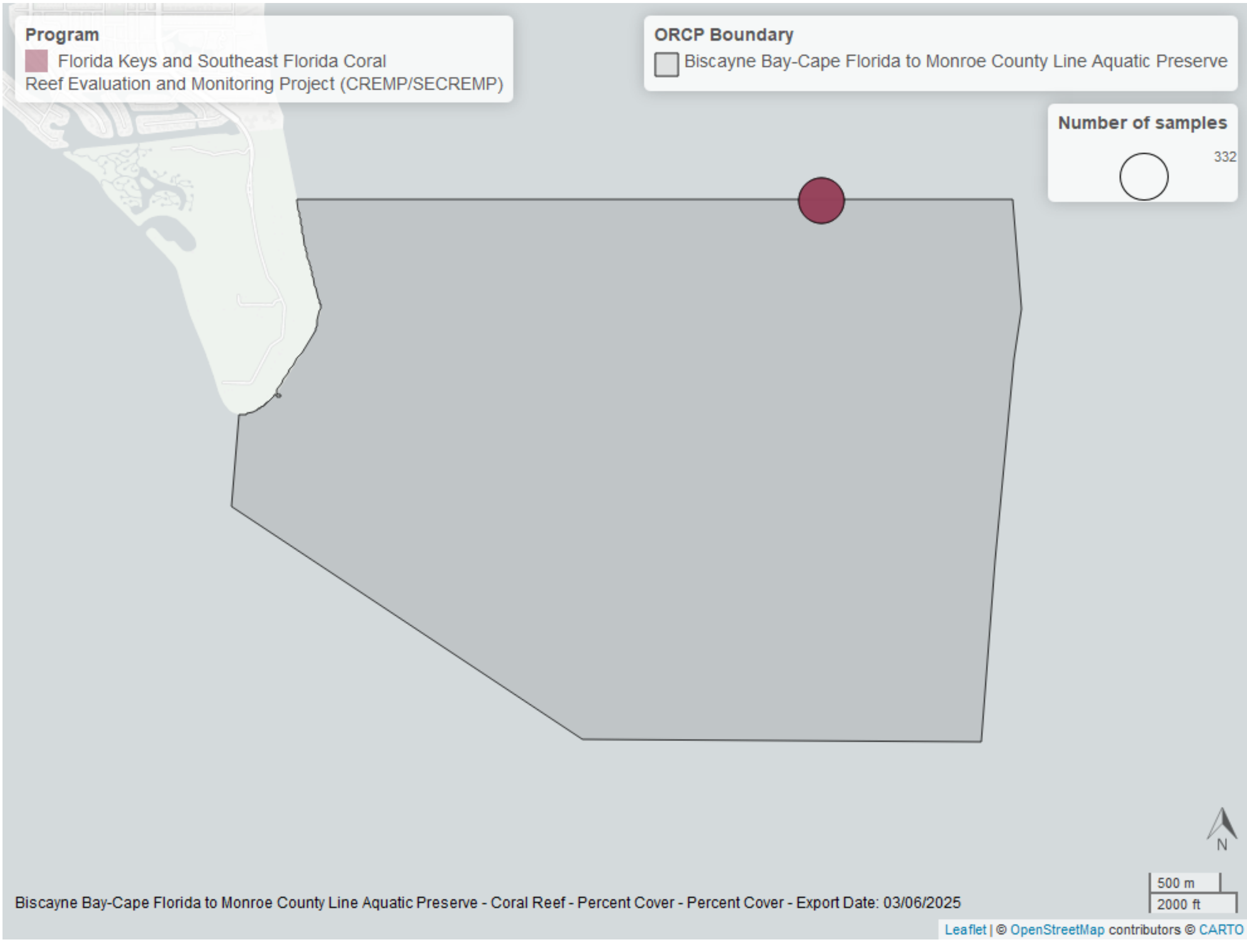


Figure 1: Scatter plot of live coral coverage over time as a percent of reef surface. Species groups include octocorals, milleporans, and scleractinians. If the time series included five or more years of observations, a significant (blue) or non-significant (magenta) trend line is also shown. Data points are jittered horizontally to reduce overlap.

Table 1: Coral Percent Cover - Biscayne Bay-Cape Florida to Monroe County Line Aquatic Preserve

Statistical Trend	Period of Record	LME Intercept	LME Slope	p
No significant trend	2013 - 2021	6.17	0	0.16

Percent cover showed no detectable trend between 2013 and 2021.



Florida Keys National Marine Sanctuary

Table 2: Coral Percent Cover - Florida Keys National Marine Sanctuary

Statistical Trend	Period of Record	LME Intercept	LME Slope	p
Significantly decreasing trend	1996 - 2023	10.78	0	0

Percent cover showed no detectable trend between 1996 and 2021.

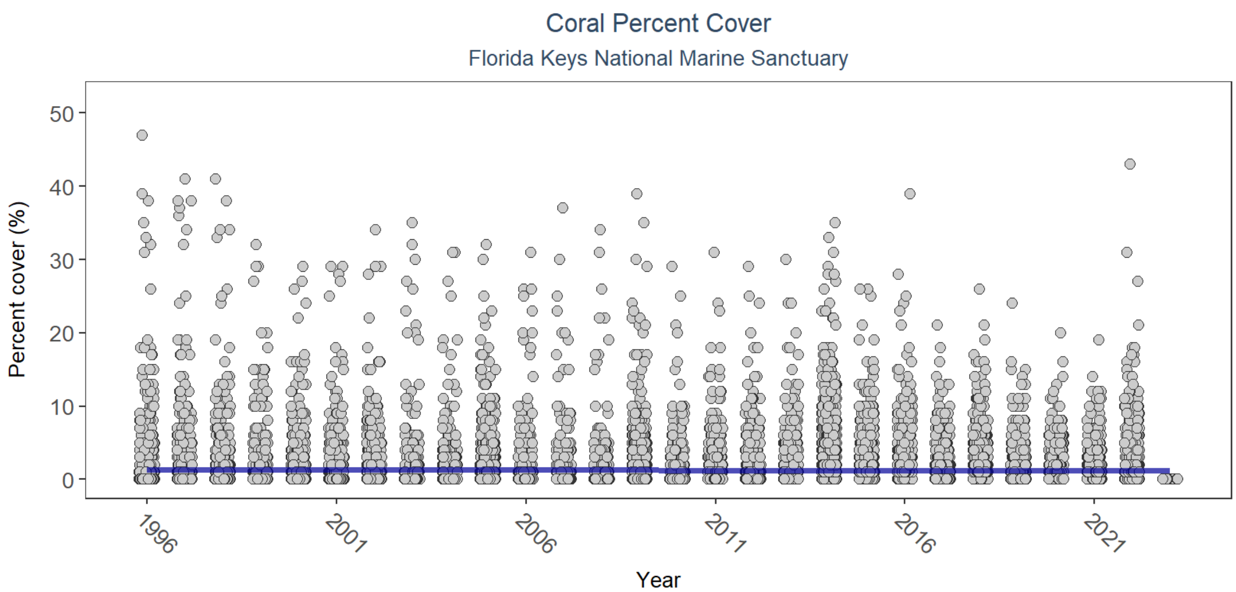
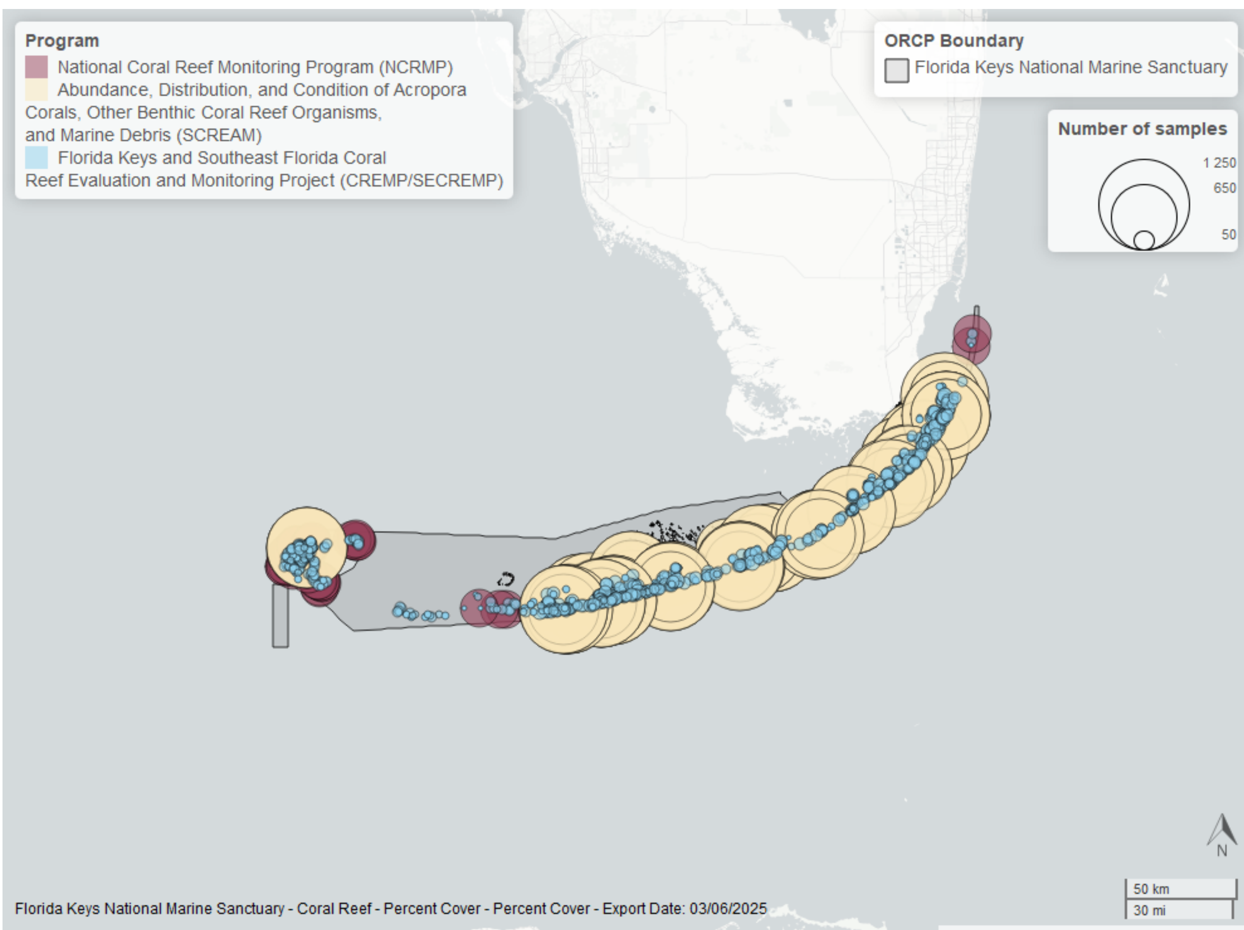


Figure 2: Scatter plot of live coral coverage over time as a percent of reef surface. Species groups include octocorals, milleporans, and scleractinians. If the time series included five or more years of observations, a significant (blue) or non-significant (magenta) trend line is also shown. Data points are jittered horizontally to reduce overlap.



Kristin Jacobs Coral Aquatic Preserve

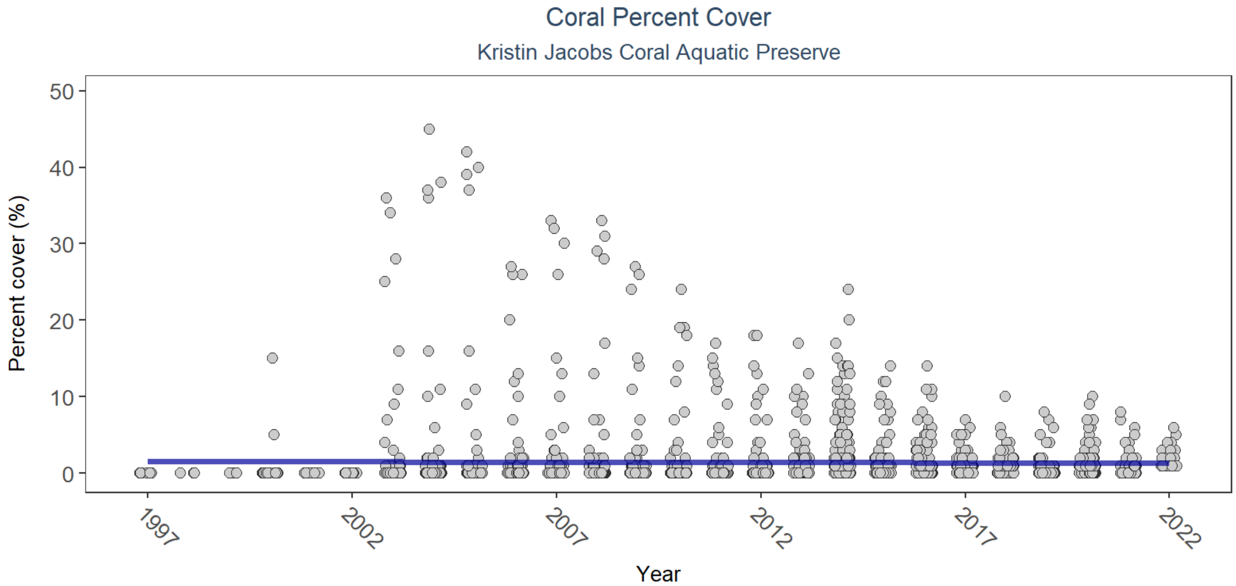
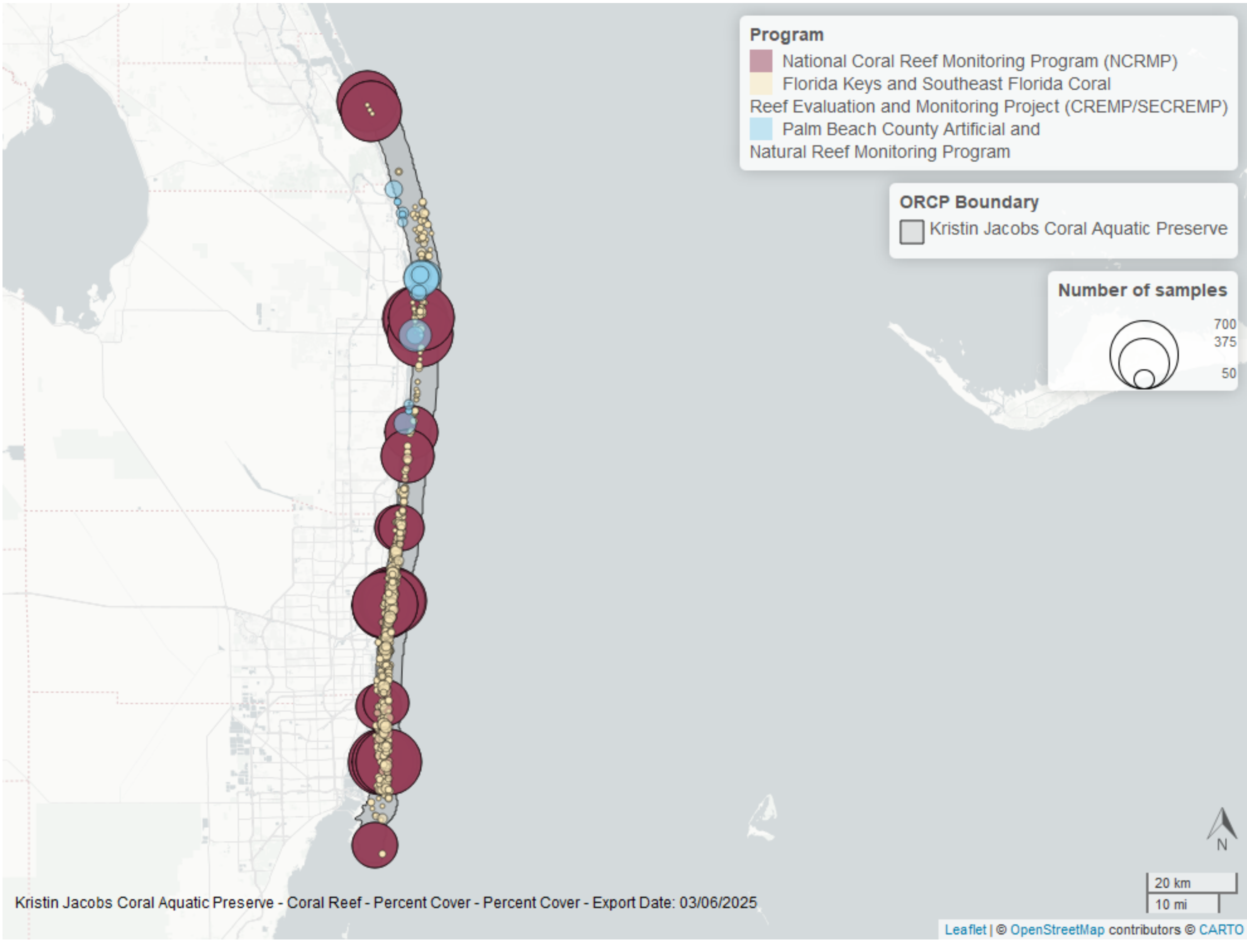


Figure 3: Scatter plot of live coral coverage over time as a percent of reef surface. Species groups include octocorals, milleporans, and scleractinians. If the time series included five or more years of observations, a significant (blue) or non-significant (magenta) trend line is also shown. Data points are jittered horizontally to reduce overlap.

Table 3: Coral Percent Cover - Kristin Jacobs Coral Aquatic Preserve

Statistical Trend	Period of Record	LME Intercept	LME Slope	p
Significantly decreasing trend	1997 - 2022	18.27	-0.01	0

Annual average percent cover increased by less than 0.01%.


Program
National Coral Reef Monitoring Program (NCRMP)
Florida Keys and Southeast Florida Coral Reef Evaluation and Monitoring Project (CREMP/SECREMP)
Palm Beach County Artificial and Natural Reef Monitoring Program
ORCP Boundary
Kristin Jacobs Coral Aquatic Preserve
Number of samples
700
375
50
N
20 km
10 mi
Kristin Jacobs Coral Aquatic Preserve - Coral Reef - Percent Cover - Percent Cover - Export Date: 03/06/2025
Leaflet | © OpenStreetMap contributors © CARTO

Libraries and Settings

Loads libraries used in the script. The inclusion of `scipen` option limits how frequently R defaults to scientific notation. Sets default settings for displaying warning and messages in created document, and sets figure dpi.

```
library(knitr)
library(data.table)
library(dplyr)
library(lubridate)
library(ggplot2)
library(scales)
library(tidyr)
library(gridExtra)
#library(tidyverse)
library(hrbrthemes)
library(nlme)
library(ggpubr)
options(scipen=999)
knitr::opts_chunk$set(
   warning=FALSE,
   message=FALSE,
   dpi=200
)
seed <- 42
```

File Import

Imports file that is determined in the Coral_PC_ReportRender.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.

The latest version of Coral data is available at: https://usf.box.com/s/8hyj2ur5arothlifg1isnq2gxisjzbdg

The file(s) being used for the analysis: **All_CORAL_Parameters-2025-Jun-06.txt**

```
data <- fread(file_in, sep="|", header=TRUE, stringsAsFactors=FALSE,
              na.strings=c("NULL","","NA"))

cat(paste("The data file(s) used:", file_short, sep="\n"))
```

Data Filtering

The processing and filtering that is done to the data is as follows:

1. Only take data rows that are Percent Cover measurements
2. Shorten parameter names to `Percent Cover`
3. Sets units
4. Removes any data that is not coral

- Only looks for `Octocoral`, `Milleporans`, or `Scleractinian`

5. Removes rows that contains NA values in `ManagedAreaName`, `GenusName`, `SpeciesName`, `Month`, `Year`, `SpeciesGroup1`, `ResultValue`, and `SampleDate`
6. Removes duplicates (`MADup`==1)

7. Combines genus and species names
8. Corrects some managed area names to match what is being used with other habitats

```r
# Only keep data for Percent Cover
# Formerly "Percent Cover - Species Composition"
data <- data[ParameterName=="Percent Cover", ]

# Sets units for percent cover
unit <- "%"
data$ParameterUnits <- unit

# Remove any rows that are not corals
data <- data[SpeciesGroup1 %in% c("Octocoral","Milleporans","Scleractinian"), ]
# Remove rows with missing GenusName
data <- data[!is.na(data$GenusName),]
# Remove rows with missing SpeciesName
data <- data[!is.na(data$SpeciesName),]
# Remove rows with missing Months
data <- data[!is.na(data$Month),]
# Remove rows with missing Years
data <- data[!is.na(data$Year),]
# Remove rows with missing SpeciesGroup1
data <- data[!is.na(data$SpeciesGroup1),]
# Remove rows with missing ResultValue
data <- data[!is.na(data$ResultValue),]
# Remove rows with missing SampleDate
data <- data[!is.na(data$SampleDate),]
# Remove duplicate rows
data <- data[data$MADup==1,]
# Create variable that combines the genus and species name
data$gensp <- paste(data$GenusName, data$SpeciesName, sep=" ")
```

Managed Area Statistics

Gets summary statistics for each managed area. Uses piping from dplyr package to feed into subsequent steps. The following steps are performed:

1. Group data that have the same `ManagedAreaName`, `Year`, and `Month`.
   - Second summary statistics do not use the `Month` grouping and are only for `ManagedAreaName` and `Year`.
   - Third summary statistics do not use `Year` grouping and are only for `ManagedAreaName` and `Month`
   - Fourth summary statistics are only grouped based on `ManagedAreaName`
2. For each group, provide the following information: Parameter Name (ParameterName), Number of Entries (N_Data), Lowest Value (Min), Largest Value (Max), Median, Mean, Standard Deviation, and a list of all Programs included in these measurements.
3. Sort the data in ascending (A to Z and 0 to 9) order based on `ManagedAreaName` then `Year` then `Month`
4. Write summary stats to a pipe-delimited .txt file in the output directory
   - Coral Percent Cover Output Files in SEACAR GitHub (https://github.com/FloridaSEACAR/SEACAR_Trend_Analyses/tree/main/Coral/output/PercentCover)
5. Determines if there is sufficient data to be analyzed based on having more than 5 years of records.
6. Gets list of managed areas to be analyzed.

```r
# Create summary statistics for each managed area based on Year and Month
# intervals.
MA_YM_Stats <- data %>%
```

```r
  group_by(AreaID, ManagedAreaName, Year, Month) %>%
  summarize(ParameterName=parameter,
            N_Data=length(na.omit(ResultValue)),
            Min=min(ResultValue),
            Max=max(ResultValue),
            Median=median(ResultValue),
            Mean=mean(ResultValue),
            StandardDeviation=sd(ResultValue),
            Programs=paste(sort(unique(ProgramName), decreasing=FALSE),
                           collapse=', '),
            ProgramIDs=paste(sort(unique(ProgramID), decreasing=FALSE),
                             collapse=', '),
            .groups = "keep")
# Puts the data in order based on ManagedAreaName, Year, then Month
MA_YM_Stats <- as.data.table(MA_YM_Stats[order(MA_YM_Stats$ManagedAreaName,
                                               MA_YM_Stats$Year,
                                               MA_YM_Stats$Month), ])
# Writes summary statistics to file
fwrite(MA_YM_Stats, paste0(out_dir,"/Coral_", param_file,
                           "_MA_MMYY_Stats.txt"), sep="|")
# Removes variable storing data to improve computer memory
rm(MA_YM_Stats)

# Create summary statistics for each managed area based on Year intervals
MA_Y_Stats <- data %>%
  group_by(AreaID, ManagedAreaName, Year) %>%
  summarize(ParameterName=parameter,
            N_Data=length(na.omit(ResultValue)),
            Min=min(ResultValue),
            Max=max(ResultValue),
            Median=median(ResultValue),
            Mean=mean(ResultValue),
            StandardDeviation=sd(ResultValue),
            Programs=paste(sort(unique(ProgramName), decreasing=FALSE),
                           collapse=', '),
            ProgramIDs=paste(sort(unique(ProgramID), decreasing=FALSE),
                             collapse=', '),
            .groups = "keep")
# Puts the data in order based on ManagedAreaName then Year
MA_Y_Stats <- as.data.table(MA_Y_Stats[order(MA_Y_Stats$ManagedAreaName,
                                             MA_Y_Stats$Year), ])
# Writes summary statistics to file
fwrite(MA_Y_Stats, paste0(out_dir,"/Coral_", param_file,
                          "_MA_Yr_Stats.txt"), sep="|")

# Create summary statistics for each managed area based on Month intervals.
MA_M_Stats <- data %>%
  group_by(AreaID, ManagedAreaName, Month) %>%
  summarize(ParameterName=parameter,
            N_Data=length(na.omit(ResultValue)),
            Min=min(ResultValue),
            Max=max(ResultValue),
            Median=median(ResultValue),
```

```r
            Mean=mean(ResultValue),
            StandardDeviation=sd(ResultValue),
            Programs=paste(sort(unique(ProgramName), decreasing=FALSE),
                           collapse=', '),
            ProgramIDs=paste(sort(unique(ProgramID), decreasing=FALSE),
                             collapse=', '),
            .groups = "keep")
# Puts the data in order based on ManagedAreaName then Month
MA_M_Stats <- as.data.table(MA_M_Stats[order(MA_M_Stats$ManagedAreaName,
                                             MA_M_Stats$Month), ])
# Writes summary statistics to file
fwrite(MA_M_Stats, paste0(out_dir,"/Coral_", param_file,
                          "_MA_Mo_Stats.txt"), sep="|")
# Removes variable storing data to improve computer memory
rm(MA_M_Stats)

# Create summary overall statistics for each managed area.
MA_Ov_Stats <- data %>%
  group_by(AreaID, ManagedAreaName) %>%
  summarize(ParameterName=parameter,
            N_Years=length(unique(na.omit(Year))),
            SufficientData=ifelse(N_Years>=5, TRUE, FALSE),
            EarliestYear=min(Year),
            LatestYear=max(Year),
            N_Data=length(na.omit(ResultValue)),
            Min=min(ResultValue),
            Max=max(ResultValue),
            Median=median(ResultValue),
            Mean=mean(ResultValue),
            StandardDeviation=sd(ResultValue),
            Programs=paste(sort(unique(ProgramName), decreasing=FALSE),
                           collapse=', '),
            ProgramIDs=paste(sort(unique(ProgramID), decreasing=FALSE),
                             collapse=', '),
            .groups = "keep")
# Puts the data in order based on ManagedAreaName
MA_Ov_Stats <- as.data.table(MA_Ov_Stats[order(MA_Ov_Stats$ManagedAreaName), ])

# Replaces blank ProgramIDs with NA (missing values)
MA_Ov_Stats$ProgramIDs <- replace(MA_Ov_Stats$ProgramIDs,
                                  MA_Ov_Stats$ProgramIDs=="", NA)
MA_Ov_Stats$Programs <- replace(MA_Ov_Stats$Programs,
                                MA_Ov_Stats$Programs=="", NA)
# Write overall statistics to file
fwrite(MA_Ov_Stats, paste0(out_dir,"/Coral_", param_file,
                           "_MA_Overall_Stats.txt"), sep="|")
# Creates a variable with the names of all the managed areas that contain
# species observations
coral_pc_MA_Include <- MA_Ov_Stats[!is.na(Mean) & SufficientData==TRUE, unique(ManagedAreaName)]

# Puts the managed areas in alphabetical order
coral_pc_MA_Include <- coral_pc_MA_Include[order(coral_pc_MA_Include)]
```

```r
# Determines the number of managed areas used
n <- length(coral_pc_MA_Include)
```

Linear Mixed Effects Models

Performs a linear mixed effects (LME) model on each managed area between using a relationship between percent cover and year.

The following steps are performed:

1. Create a blank data frame to store results
2. Sets the column names for the data to be stored from LME model
3. Starts a loop for each managed area included in the analysis
4. Gets data for the current managed area
5. Performs LME on current managed area
6. Stores information and fits into lme_stats data frame for current managed area
7. Merges lme_stats with MA_Ov_Stats to create a data frame with gegneral statistics and the LME parameters
8. Puts lme_stats in alphabetical order by managed area name
9. Write lme_stats to a pipe-delimited .txt file in the output directory
    - Coral Percent Cover Output Files in SEACAR GitHub (https://github.com/FloridaSEACAR/SEACAR_Trend_Analyses/tree/main/Coral/output/PercentCover)
10. Gets the start and endpoints for LME fit to be used in plots.

```r
# Creates blank data frame with number of rows defined by how many managed areas
# are going to be analyzed
lme_stats <- data.frame(matrix(ncol = 5, nrow = n))
# Sets column names for blank data frame
colnames(lme_stats) <- c("AreaID", "ManagedAreaName", "LME_Intercept",
                         "LME_Slope", "LME_p")

# Begins to loop through each managed area for analysis
for(i in 1:n){
  ma_i <- coral_pc_MA_Include[i]
  # Gets data for current managegd area
  lme_data <- data[ManagedAreaName==ma_i,]
  # Perform LME for relation between ResultValue and Year for current managed area
  AnyCoral<-lme(ResultValue ~ Year,
              random =~1|ProgramLocationID,
              na.action = na.omit,
              data = lme_data)
  # Store information and model fits in appropriate row of data frame
  lme_stats$AreaID[i] <- unique(lme_data$AreaID)
  lme_stats$ManagedAreaName[i] <- ma_i
  lme_stats$LME_Intercept[i] <- AnyCoral$coefficients$fixed[1]
  lme_stats$LME_Slope[i] <- AnyCoral$coefficients$fixed[2]
  lme_stats$LME_p[i] <- anova(AnyCoral)$p[2]

  # Clears temporary variables for memory
  rm(lme_data)
  (AnyCoral)
}

# Merges LME stats with overall stats to complete stats for each managed area
```

```r
lme_stats <- merge.data.frame(MA_Ov_Stats[,-c("Programs", "ProgramIDs")],
                              lme_stats, by=c("AreaID", "ManagedAreaName"), all=TRUE)

# Puts the data in order based on ManagedAreaName
lme_stats <- as.data.frame(lme_stats[order(lme_stats$ManagedAreaName), ])

# Write lme statistics to file
fwrite(lme_stats, paste0(out_dir,"/Coral_", param_file,
                         "_LME_Stats.txt"), sep="|")

# Gets lower x and y values based on LME fit to use in plot
lme_plot <- lme_stats %>%
  group_by(AreaID, ManagedAreaName) %>%
  summarize(x=EarliestYear,
            y=LME_Slope*x+LME_Intercept, .groups = "keep")
# Gets upper x and y values based on LME fit to use in plot
lme_plot2 <- lme_stats %>%
  group_by(AreaID, ManagedAreaName) %>%
  summarize(x=LatestYear,
            y=LME_Slope*x+LME_Intercept, .groups = "keep")
# Merges LME fit values for plot into one data frame
lme_plot <- bind_rows(lme_plot, lme_plot2)
rm(lme_plot2)
# Puts LME plot data fram in alphabetical order by managed area
lme_plot <- as.data.frame(lme_plot[order(lme_plot$ManagedAreaName), ])
lme_plot <- lme_plot[!is.na(lme_plot$y),]
```

Appendix I: Plots

The plots shown here are the percent cover for each managed area by year with the LME trendline.

1. Set common plot theme.
2. Starts a loops that creates plots for each managed area analyzed
3. Determine the earliest and latest year of the data to create x-axis scale and intervals
4. Determine the upper and lower limit of the plot for better y-axis labels
5. Set the plot type as a jitter plot with the size of the points to show concentration of data
6. Add LME trendline
7. Create the title, x-axis, y-axis
8. Set the y and x limits
9. Apply common plot theme
10. Add table with summary statistics below each figure
    - Numerical non-integer values are rounded to 2 decimal places
    - StandardDeviation is renamed StDev for space reasons
11. Create file name to save figure
12. Save figure as png file

- Coral Percent Cover Figures in SEACAR GitHub (https://github.com/FloridaSEACAR/SEACAR_Trend_Analyses/tree/main/Coral/output/PercentCover/Figures)

```r
# Defines standard plot theme: black and white, no major or minor grid lines,
# Arial font. Title is centered, size 12, and blue (hex coded). Subtitle is
# centered, size 10, and blue (hex coded). Legend title is size 10 and the
# legend is left-justified. X-axis title is size 10 and the margins are padded
# at the top and bottom to give more space for angled axis labels. Y-axis title
```

```r
# is size 10 and margins are padded on the right side to give more space for
# axis labels. Axis labels are size 10 and the x-axis labels are rotated -45
# degrees with a horizontal justification that aligns them with the tick mark
plot_theme <- theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        text=element_text(family="Arial"),
        plot.title=element_text(hjust=0.5, size=12, color="#314963"),
        plot.subtitle=element_text(hjust=0.5, size=10, color="#314963"),
        legend.title=element_text(size=10),
        legend.text = element_text(hjust=0),
        axis.title.x = element_text(size=10, margin = margin(t = 5, r = 0,
                                                             b = 10, l = 0)),
        axis.title.y = element_text(size=10, margin = margin(t = 0, r = 10,
                                                             b = 0, l = 0)),
        axis.text=element_text(size=10),
        axis.text.x=element_text(angle = -45, hjust = 0))
# Create jitter object that sets the height and width
# Sets seed to be reproducible
plot_jitter <- position_jitter(width = 0.2, height = 0.2, seed=seed)

# Loop that cycles through each managed area with data
if(n==0){
  # Prints a statement if there are no managed areas with appropriate data
  print("There are no locations that qualify.")
} else {
  for (i in 1:n) {
    ma_i <- coral_pc_MA_Include[i]
    # Gets data for target managed area
    plot_data <- data[ManagedAreaName==ma_i,]

    lme_plot_data <- lme_plot[ManagedAreaName==ma_i,]
    # Determines most recent year with available data for managed area
    t_max <- max(MA_Ov_Stats[ManagedAreaName==ma_i, LatestYear])
    # Determines earliest recent year with available data for managed area
    t_min <- min(MA_Ov_Stats[ManagedAreaName==ma_i, EarliestYear])
    # Determines how many years of data are present
    t <- t_max-t_min

    # Creates break intervals for plots based on number of years of data
    if(t>=30){
      # Set breaks to every 10 years if more than 30 years of data
      brk <- -10
    }else if(t<30 & t>=10){
      # Set breaks to every 5 years if between 30 and 10 years of data
      brk <- -5
    }else if(t<10 & t>=4){
      # Set breaks to every 2 years if between 10 and 4 years of data
      brk <- -2
    }else if(t<4 & t>=2){
      # Set breaks to every year if between 4 and 2 years of data
      brk <- -1
    }else if(t<2){
```

```
    # Set breaks to every year if less than 2 years of data
    brk <- -1
    # Sets t_max to be 1 year greater and t_min to be 1 year lower
    # Forces graph to have at least 3 tick marks
    t_max <- t_max+1
    t_min <- t_min-1
  }
  # Determine range of data values for the managed area
  y_range <- max(plot_data$ResultValue) - min(plot_data$ResultValue)

  # Sets y_min to be -1
  y_min <- -1

  # Sets upper bound of y-axis to be 10% of the data range above the
  # maximum value.
  y_max <- max(plot_data$ResultValue)+(0.1*y_range)

  # Creates plot object using plot_data.
  # Data is plotted as a point pot with jitter to show concentrations
  # that overlap. LME fit is plotted as a line
  p1 <- ggplot(data=plot_data) +
    geom_point(aes(x=Year, y=ResultValue),
               position=plot_jitter, shape=21, size=2,
               color="#333333", fill="#cccccc", alpha=1) +
    geom_line(data=lme_plot_data, aes(x=x, y=y),
              color="#000099", size=2, alpha=0.8) +
    labs(title="Coral Percent Cover",
         subtitle=ma_i,
         x="Year", y="Percent cover (%)") +
    scale_x_continuous(limits=c(t_min-0.25, t_max+0.25),
                       breaks=seq(t_max, t_min, brk)) +
    scale_y_continuous(limits=c(y_min, y_max),
                       breaks=pretty_breaks(n=5)) +
    plot_theme
  # Sets file name of plot created
  outname <- paste0("Coral_", param_file, "_", gsub(" ", "", ma_i),
                    ".png")
  # Saves plot as a png image
  png(paste0(out_dir, "/Figures/", outname),
      width = 8,
      height = 4,
      units = "in",
      res = 200)
  print(p1)
  dev.off()

  # Creates a data table object to be shown underneath plots in report
  ResultTable <- lme_stats[ManagedAreaName==ma_i,]
  # Removes location, and parameter information because it is in plot
  # labels
  ResultTable <- select(ResultTable, -c("AreaID", "ManagedAreaName",
                                        "ParameterName"))
  # Renames StandardDeviation to StDev to save horizontal space
```

```
    ResultTable <- ResultTable %>%
      rename("StDev"="StandardDeviation")
    # Converts all non-integer values to 2 decimal places for space
    ResultTable$Min <- round(ResultTable$Min, digits=2)
    ResultTable$Max <- round(ResultTable$Max, digits=2)
    ResultTable$Median <- round(ResultTable$Median, digits=2)
    ResultTable$Mean <- round(ResultTable$Mean, digits=2)
    ResultTable$StDev <- round(ResultTable$StDev, digits=2)
    ResultTable$LME_Intercept <- round(ResultTable$LME_Intercept, digits=2)
    ResultTable$LME_Slope <- round(ResultTable$LME_Slope, digits=2)
    ResultTable$LME_p <- round(ResultTable$LME_p, digits=4)
    # Stores as plot table object
    t1 <- ggtexttable(ResultTable, rows = NULL,
                      theme=ttheme(base_size=7)) %>%
      tab_add_footnote(text="LME_p < 0.00005 appear as 0 due to rounding.",
                       size=10, face="italic")
    # Combines plot and table into one figure
    print(ggarrange(p1, t1, ncol=1, heights=c(0.85, 0.15)))

    # Add extra space at the end to prevent the next figure from being too
    # close. Does not add space after last plot
    if(i!=n){
      cat("\n \n \n \n")
    }
  }
}
```