SEACAR Nekton Analysis: Species Richness

Last compiled on 09 June, 2023

Table of Contents

# Important Notes

The purpose of this script is to determine species richness by gear type and size, create managed area statistics, generate plots, and create reports in pdf and Word document form for Nekton data.

These scripts were created by [J.E. Panzik](mailto:jepanzik@usf.edu) ([jepanzik@usf.edu](mailto:jepanzik@usf.edu)) for SEACAR.

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 [SEACAR\_Nekton\_SpeciesRichness\_ReportRender.R](https://github.com/FloridaSEACAR/SEACAR_Trend_Analyses/blob/main/Nekton/SEACAR_Nekton_SpeciesRichness_ReportRender.R) (<https://github.com/FloridaSEACAR/SEACAR_Trend_Analyses/blob/main/Nekton/SEACAR_Nekton_SpeciesRichness_ReportRender.R>).

Details on the determination of catch per unit effort can be found in the document [SEACAR Nekton catch per unit effort.pdf](https://github.com/FloridaSEACAR/SEACAR_Trend_Analyses/blob/main/Nekton/SEACAR%20Nekton%20catch%20per%20unit%20effort.pdf) (<https://github.com/FloridaSEACAR/SEACAR_Trend_Analyses/blob/main/Nekton/SEACAR%20Nekton%20catch%20per%20unit%20effort.pdf>).

# 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(ggpubr)  
library(scales)  
options(scipen=999)  
knitr::opts\_chunk$set(  
 warning=FALSE,  
 message=FALSE,  
 dpi=200  
 )

# File Import

Imports file that is determined in the SEACAR\_Nekton\_SpeciesRichness\_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 Nekton data is available at: <https://usf.box.com/s/35sn0n0lrrxi9dtkik030nozbvnj9dyj>

The file being used for the analysis is: **All\_NEKTON\_Parameters-2023-Jun-05.txt**

#Import data from nekton file  
data <- fread(file\_in, sep="|", header=TRUE, stringsAsFactors=FALSE,  
 na.strings=c("NULL","","NA"))  
  
cat(paste("The data file used is:", file\_short, sep="\n"))

## The data file used is:  
## All\_NEKTON\_Parameters-2023-Jun-05.txt

# Data Filtering

Documentation on database filtering is provided here: [SEACAR Documentation- Analysis Filters and Calculations.pdf](https://github.com/FloridaSEACAR/SEACAR_Trend_Analyses/blob/main/SEACAR%20Documentation%20-%20Analysis%20Filters%20and%20Calculations.pdf) (<https://github.com/FloridaSEACAR/SEACAR_Trend_Analyses/blob/main/SEACAR%20Documentation%20-%20Analysis%20Filters%20and%20Calculations.pdf>).

Imported data is initially filtered to only contain the parameter of interest.

The other filtering performed by the script at this point removes rows that are missing values for ResultValue and EffortCorrection\_100m2, and removes any EffortCorrection\_100m2 that is 0 because it will cause an infinite number when determining Species Richness.

A group of unique ManagedAreaName, ProgramID, ProgramName, ProgramLocationID, SampleDate, and GearSize\_m are being considered a “reference” for measurement. For each “reference”, the number of observed species is summed and then divided by the EffortCorrection\_100m2to determine the Species Richness per 100 square meters.

The ManagedAreaName values from the data are actually shortened versions, and are merged with the full versions. The species richness data is then written to a file. And the list of Managed Areas with observations is stored.

# Filter data for the desired parameter  
data <- data[data$ParameterName==param\_name,]  
  
if (param\_name=="Presence"){  
 parameter <- "Species Richness"  
 }  
# Makes sure EffortCorrection is numeric value  
data$EffortCorrection\_100m2 <- as.numeric(data$EffortCorrection\_100m2)  
  
# Remove any data with missing EffortCorrection values  
data <- data[!is.na(data$EffortCorrection\_100m2),]  
  
# Only keep data that has non-zero EffortCorrection values  
data <- data[data$EffortCorrection\_100m2!=0,]  
  
# Remove any data with missing ResultValue entries  
data <- data[!is.na(data$ResultValue),]  
  
# Create Species Richness values for groups of unique combinations of  
# ManagedAreaName, ProgramID, ProgramName, ProgramLocationID, SampleDate,  
# GearType, and GearSize\_m.  
data <- data %>%  
 group\_by(ManagedAreaName, ProgramID, ProgramName, ProgramLocationID,  
 SampleDate, GearType, GearSize\_m) %>%  
 summarise(ParameterName=parameter,  
 Year=unique(Year), Month=unique(Month),  
 N\_Species=sum(ResultValue),  
 EffortCorrection\_100m2=as.numeric(unique(EffortCorrection\_100m2)),  
 SpeciesRichness=N\_Species/unique(EffortCorrection\_100m2))  
  
# Adds AreaID for each managed area by combining the MA\_All datatable to the  
# data based on ManagedAreaName  
data <- merge.data.frame(MA\_All[,c("AreaID", "ManagedAreaName")],  
 data, by="ManagedAreaName", all=TRUE)  
  
# Writes this data that is used by the rest of the script to a text file  
fwrite(data, paste0(out\_dir,"/Nekton\_", param\_file, "\_UsedData.txt"), sep="|")  
  
# Makes sure SampleDate is being stored as a Date object  
data$SampleDate <- as.Date(data$SampleDate)  
  
# Creates a variable with the names of all the managed areas that contain  
# species observations  
MA\_Include <- unique(data$ManagedAreaName[!is.na(data$N\_Species)])  
  
# Puts the managed areas in alphabetical order  
MA\_Include <- MA\_Include[order(MA\_Include)]  
  
# Determines the number of managed areas used  
n <- length(MA\_Include)

# 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, Month, GearType, and GearSize\_m.
   * Second summary statistics do not use the Month grouping and are only for ManagedAreaName, Year, GearType, and GearSize\_m.
   * Third summary statistics do not use Year grouping and are only for ManagedAreaName, Month, GearType, and GearSize\_m
   * Fourth summary statistics are only grouped based on ManagedAreaName, GearType, and GearSize\_m
     + Determines the years that the minimum and maximum species richness occurred
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 Program IDs 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
   * [Nekton Output Files in SEACAR GitHub](https://github.com/FloridaSEACAR/SEACAR_Trend_Analyses/tree/main/Nekton/output) (<https://github.com/FloridaSEACAR/SEACAR_Trend_Analyses/tree/main/Nekton/output>)

# Create summary statistics for each managed area based on Year and Month  
# intervals, and each gear type and size.  
MA\_YM\_Stats <- data %>%  
 group\_by(AreaID, ManagedAreaName, Year, Month, GearType, GearSize\_m) %>%  
 summarize(ParameterName=parameter,  
 N\_Data=length(na.omit(SpeciesRichness)),  
 Min=min(SpeciesRichness),  
 Max=max(SpeciesRichness),  
 Median=median(SpeciesRichness),  
 Mean=mean(SpeciesRichness),  
 StandardDeviation=sd(SpeciesRichness),  
 Programs=paste(sort(unique(ProgramName), decreasing=FALSE),  
 collapse=', '),  
 ProgramIDs=paste(sort(unique(ProgramID), decreasing=FALSE),  
 collapse=', '))  
# Puts the data in order based on ManagedAreaName, Year, Month, then GearSize  
MA\_YM\_Stats <- as.data.table(MA\_YM\_Stats[order(MA\_YM\_Stats$ManagedAreaName,  
 MA\_YM\_Stats$Year,  
 MA\_YM\_Stats$Month,  
 MA\_YM\_Stats$GearSize\_m), ])  
# Writes summary statistics to file  
fwrite(MA\_YM\_Stats, paste0(out\_dir,"/Nekton\_", 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,  
# and each gear type and size.  
MA\_Y\_Stats <- data %>%  
 group\_by(AreaID, ManagedAreaName, Year, GearType, GearSize\_m) %>%  
 summarize(ParameterName=parameter,  
 N\_Data=length(na.omit(SpeciesRichness)),  
 Min=min(SpeciesRichness),  
 Max=max(SpeciesRichness),  
 Median=median(SpeciesRichness),  
 Mean=mean(SpeciesRichness),  
 StandardDeviation=sd(SpeciesRichness),  
 Programs=paste(sort(unique(ProgramName), decreasing=FALSE),  
 collapse=', '),  
 ProgramIDs=paste(sort(unique(ProgramID), decreasing=FALSE),  
 collapse=', '))  
# Puts the data in order based on ManagedAreaName, Year, then GearSize  
MA\_Y\_Stats <- as.data.table(MA\_Y\_Stats[order(MA\_Y\_Stats$ManagedAreaName,  
 MA\_Y\_Stats$Year,  
 MA\_Y\_Stats$GearSize\_m), ])  
# Writes summary statistics to file  
fwrite(MA\_Y\_Stats, paste0(out\_dir,"/Nekton\_", param\_file,  
 "\_MA\_Yr\_Stats.txt"), sep="|")  
  
# Create summary statistics for each managed area based on Month intervals,  
# and each gear type and size.  
MA\_M\_Stats <- data %>%  
 group\_by(AreaID, ManagedAreaName, Month, GearType, GearSize\_m) %>%  
 summarize(ParameterName=parameter,  
 N\_Data=length(na.omit(SpeciesRichness)),  
 Min=min(SpeciesRichness),  
 Max=max(SpeciesRichness),  
 Median=median(SpeciesRichness),  
 Mean=mean(SpeciesRichness),  
 StandardDeviation=sd(SpeciesRichness),  
 Programs=paste(sort(unique(ProgramName), decreasing=FALSE),  
 collapse=', '),  
 ProgramIDs=paste(sort(unique(ProgramID), decreasing=FALSE),  
 collapse=', '))  
# Puts the data in order based on ManagedAreaName, Month, then GearSize  
MA\_M\_Stats <- as.data.table(MA\_M\_Stats[order(MA\_M\_Stats$ManagedAreaName,  
 MA\_M\_Stats$Month,  
 MA\_M\_Stats$GearSize\_m), ])  
# Writes summary statistics to file  
fwrite(MA\_M\_Stats, paste0(out\_dir,"/Nekton\_", 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 based each gear type  
# and size.  
MA\_Ov\_Stats <- data %>%  
 group\_by(AreaID, ManagedAreaName, GearType, GearSize\_m) %>%  
 summarize(ParameterName=parameter,  
 N\_Years=length(unique(na.omit(Year))),  
 EarliestYear=min(Year),  
 LatestYear=max(Year),  
 N\_Data=length(na.omit(SpeciesRichness)),  
 Min=min(SpeciesRichness),  
 Max=max(SpeciesRichness),  
 Median=median(SpeciesRichness),  
 Mean=mean(SpeciesRichness),  
 StandardDeviation=sd(SpeciesRichness),  
 Programs=paste(sort(unique(ProgramName), decreasing=FALSE),  
 collapse=', '),  
 ProgramIDs=paste(sort(unique(ProgramID), decreasing=FALSE),  
 collapse=', '))  
# Puts the data in order based on ManagedAreaName then GearSize  
MA\_Ov\_Stats <- as.data.table(MA\_Ov\_Stats[order(MA\_Ov\_Stats$ManagedAreaName,  
 MA\_Ov\_Stats$GearSize\_m), ])  
# Creates Year\_MinRichness and Year\_MaxRichness columns  
MA\_Ov\_Stats$Year\_MinRichness <- NA  
MA\_Ov\_Stats$Year\_MaxRichness <- NA  
  
# Loops through each ManagedAreaName, GearType, and GearSize\_m.  
# determines what year the minimum and maximum species richness occurred  
for(m in 1:nrow(MA\_Ov\_Stats)){  
 # Stores ManagedAreaName, GearType, and GearSize\_m for this row  
 ma <- MA\_Ov\_Stats$ManagedAreaName[m]  
 gear <- MA\_Ov\_Stats$GearType[m]  
 size <- MA\_Ov\_Stats$GearSize\_m[m]  
 # Skips to next row if there are no data for this combination  
 if(MA\_Ov\_Stats$N\_Data[m]==0){  
 next  
 }  
 # Gets subset of data from MA\_Y\_Stats (yearly summary stats) with this  
 # combination of ManagedAreaName, GearType, and GearSize\_m  
 ds <- MA\_Y\_Stats[MA\_Y\_Stats$ManagedAreaName==ma &  
 MA\_Y\_Stats$GearType==gear &  
 MA\_Y\_Stats$GearSize\_m==size,]  
 # Gets the minimum and maximum Mean (yearly averages)  
 min <- min(ds$Mean)  
 max <- max(ds$Mean)  
 #Determines what years those minimum and maximum values occured  
 year\_min <- ds$Year[ds$Mean==min]  
 year\_max <- ds$Year[ds$Mean==max]  
 # Stores the occurrence years of the minimum and maximum into the overall  
 # stats for this row  
 MA\_Ov\_Stats$Year\_MinRichness[m] <- year\_min  
 MA\_Ov\_Stats$Year\_MaxRichness[m] <- year\_max  
}  
# Replaces blank ProgramIDs with NA (missing values)  
MA\_Ov\_Stats$ProgramIDs <- replace(MA\_Ov\_Stats$ProgramIDs,  
 MA\_Ov\_Stats$ProgramIDs=="", NA)  
MA\_Ov\_Stats$Programs <- replace(MA\_Ov\_Stats$Programs,  
 MA\_Ov\_Stats$Programs=="", NA)  
# Write overall statistics to file  
fwrite(MA\_Ov\_Stats, paste0(out\_dir,"/Nekton\_", param\_file,  
 "\_MA\_Overall\_Stats.txt"), sep="|")  
# Removes entries from the overall statistics that do not have data.  
# Based on presence or absence of EarliestYear  
MA\_Ov\_Stats <- MA\_Ov\_Stats[!is.na(MA\_Ov\_Stats$EarliestYear), ]

# Appendix I: Managed Area Species Richness

The plots shown here are the species richness for each managed area with a yearly average, separated by gear size.

1. Set common plot theme.
2. Determine the earliest and latest year of the data to create x-axis scale and intervals
3. Determine the upper and lower limit of the plot for better y-axis labels
4. Determines what gear types are present and adjusts legend entries
5. Add the plot line
6. Set the plot type as a point plot with the size of the points
7. Create the title, x-axis, y-axis, and color fill labels
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

1. Create file name to save figure
2. Save figure as png file

* [Nekton Figures in SEACAR GitHub](https://github.com/FloridaSEACAR/SEACAR_Trend_Analyses/tree/main/Nekton/output/Figures) (<https://github.com/FloridaSEACAR/SEACAR_Trend_Analyses/tree/main/Nekton/output/Figures>)

# 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  
# 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.align = 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))  
  
# Color palette for SEACAR  
color\_palette <- c("#005396", "#0088B1", "#00ADAE", "#65CCB3", "#AEE4C1", "#FDEBA8", "#F8CD6D", "#F5A800", "#F17B00")  
  
# Defines and sets variable with standardized gear colors for plots  
gear\_colors <- c("Trawl (4.8 m)"=color\_palette[1],  
 "Trawl (6.1 m)"=color\_palette[2],  
 "Seine (183 m)"=color\_palette[3])  
  
# Defines and sets variable with standardized gear shapes for plots  
gear\_shapes <- c("Trawl (4.8 m)"=21,  
 "Trawl (6.1 m)"=22,  
 "Seine (183 m)"=24)  
  
# Loop that cycles through each managed area with data  
if(n==0){  
 # Prints a statement if there are no managed areas with approproiate data  
 print("There are no monitoring locations that qualify.")  
} else {  
 for (i in 1:n) {  
 # Gets data for target managed area  
 plot\_data <- MA\_Y\_Stats[MA\_Y\_Stats$ManagedAreaName==MA\_Include[i]]  
 # Gets the gear type(s) present for the managed area.  
 # Combine type and size into one label for plots  
 plot\_data$GearType\_Plot <- paste0(plot\_data$GearType, " (",  
 plot\_data$GearSize\_m, " m)")  
 # Determines most recent year with available data for managed area  
 t\_max <- max(MA\_Ov\_Stats$LatestYear[MA\_Ov\_Stats$ManagedAreaName==  
 MA\_Include[i]])  
 # Determines earliest recent year with available data for managed area  
 t\_min <- min(MA\_Ov\_Stats$EarliestYear[MA\_Ov\_Stats$ManagedAreaName==  
 MA\_Include[i]])  
 # 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){  
 # Set breaks to every year if less than 4 years of data  
 brk <- -1  
 }  
 # Determine range of data values for the managed area  
 y\_range <- max(plot\_data$Mean) - min(plot\_data$Mean)  
   
 # Determines lower bound of y-axis based on data range. Set based on  
 # relation of data range to minimum value. Designed to set lower boundary  
 # to be 10% of the data range below the minimum value  
 y\_min <- if(min(plot\_data$Mean)-(0.1\*y\_range)<0){  
 # If 10% of the data range below the minimum value is less than 0,  
 # set as 0  
 y\_min <- 0  
 } else {  
 # Otherwise set minimum bound as 10% data range below minimum value  
 y\_min <- min(plot\_data$Mean)-(0.1\*y\_range)  
 }  
   
 # Sets upper bound of y-axis to be 10% of the data range above the  
 # maximum value.  
 y\_max <- max(plot\_data$Mean)+(0.1\*y\_range)  
   
 # Determines what combination of gear are present for managed area  
 # and subsets color and shape scheme to be used by plots.  
 # Used so only gear combinations present for managed area appear in  
 # the legend.  
 gear\_colors\_plot <- gear\_colors[unique(plot\_data$GearType\_Plot)]  
 gear\_shapes\_plot <- gear\_shapes[unique(plot\_data$GearType\_Plot)]  
   
 # Creates plot object using plot\_data and grouping by the plot gear types.  
 # Data is plotted as symbols with connected lines.  
 p1 <- ggplot(data=plot\_data, group=as.factor(GearType\_Plot)) +  
 geom\_line(aes(x=Year, y=Mean, color=as.factor(GearType\_Plot)),  
 size=0.75, alpha=1) +  
 geom\_point(aes(x=Year, y=Mean, fill=as.factor(GearType\_Plot),  
 shape=as.factor(GearType\_Plot)), size=2,  
 color="#333333", alpha=1) +  
 labs(title="Nekton Species Richness",  
 subtitle=MA\_Include[i],  
 x="Year", y=bquote('Richness (species/100'\*~m^{2}\*')'),  
 fill="Gear type", color="Gear type", shape="Gear type") +  
 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)) +  
 scale\_fill\_manual(values=gear\_colors\_plot) +  
 scale\_color\_manual(values=gear\_colors\_plot) +  
 scale\_shape\_manual(values=gear\_shapes\_plot) +  
 plot\_theme  
 # Sets file name of plot created  
 outname <- paste0("Nekton\_", param\_file, "\_",  
 gsub(" ", "", MA\_Include[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 <-  
 MA\_Ov\_Stats[MA\_Ov\_Stats$ManagedAreaName==MA\_Include[i],]  
 # Removes location, gear, and parameter information because it is in plot  
 # labels  
 ResultTable <- ResultTable[,-c("AreaID", "ManagedAreaName",  
 "ProgramIDs", "Programs", "GearType\_Plot",  
 "ParameterName")]  
 # Renames StandardDeviation to StDev to save horizontal space  
 ResultTable <- ResultTable %>%  
 rename("StDev"="StandardDeviation")  
 # Converts all non-integer values to 2 decimal places for space  
 ResultTable$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)  
 # Stores as plot table object  
 t1 <- ggtexttable(ResultTable, rows = NULL,  
 theme=ttheme(base\_size=7))  
 # 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")   
 }  
 }  
}





















 