The following supplement accompanies the article

Discriminating trophic niches of carnivorous benthic macroinvertebrates with gut contents, stable isotopes, and fatty acids

Christopher A. North*, James R. Lovvorn, Jason M. Kolts, Lee W. Cooper, Jacqueline M. Grebmeier

*Corresponding author: cnorth@uwyo.edu

Marine Ecology Progress Series 631: 49–66 (2019)

```
setwd("/Users/cnorth/Dropbox/Predators MS/Preds R code/SISUS/")

# Set the working directory in which input files are found
# and output files are saved

library(sp)

# Loads "sp" R package that provides "point.in.polygon" function
# May need to be installed before first use

rm(list=ls())

#resets each run



####### DESCRIPTION - TDF estimation

# This program estimates trophic discrimination factors (TDFs)
# from field data mean isotope ratio values (d13C & d15N)
# for sources that define a polygon in bivariate isotope space
# and isotope values for individual mixtures (e.g. consumers).
# The goal is to find all combinations of TDFs (Delta13C &
# Delta15N) that minimize the number of individual mixtures
# that fall outside the polygon (as written, with 0 mixtures
# falling outside, though in some case this may need to be
# modified).  Mean, standard deviation, minimum, and maximum
# TDFs for each isotope ratio are extracted from these minimizing
# combinations and returned.
```

```
####### INPUTS

TDF.C.min <- -0.5
TDF.C.max <- 3.0
TDF.N.min <- 0
TDF.N.max <- 4.0

# Range (minimum and maximum for both C and N)of potential
# TDF values to evaluate. Can be changed.

mixgroup <- "NC"

# Identifier for the group of mixtures (e.g. consumers) to
# evaluate. Can be changed.

SI_data <- read.csv("input_file_name.csv", header=TRUE)

# File (.csv) containing stable isotope data for sources (means)
# and mixtures (individuals) with 4 needed columns labeled as
# follows (may contain other columns to keep track of data):
#   - point: contains either "bound" (no quotes) for mean
#          source values that bound the polygon, or some
#          identifier for groups of mixtures (e.g. "NC"
#          for individual whelks of the species Neptunea
#            communis in this example).  Multiple groups
#            may be contained in the same file, and can
#            are evaluated separately by replacing the
#            mixture group identifier entered above as input.
#   - b_order: numbers for "bound" points (source means)
#          indicating their place around the polygon defined
#          by sources in clockwise order, and these points
#          must be listed in order (i.e. 1, 2, 3, etc.)
#            in the file - the "point.in.polygon" function
#            does not work properly otherwise.
#   - d13C & d15N: stable isotope data (means for sources,
#                  individual values for mixtures)



####### PROCEDURE - TDF estimation

# by = interval by which TDFs are evaluated (can be changed)

TDF.C <- seq(from = TDF.C.min, to = TDF.C.max, by = 0.05)
TDF.N <- seq(from = TDF.N.min, to = TDF.N.max, by = 0.05)

TDFn <- as.vector(NULL)
TDFc <- as.vector(NULL)

# Create matrix to hold the number of mixtures outside the
```

```
# source polygon for each combination of TDFs evaluated

TDFmatrix <- matrix(0L, nrow = length(TDF.C), ncol = length(TDF.N))
colnames(TDFmatrix) <- TDF.N
rownames(TDFmatrix) <- TDF.C

# Read in mean source isotope values bounding the polygon

bounds.C <- subset(SI_data$d13C, SI_data$point == "bound")
bounds.N <- subset(SI_data$d15N, SI_data$point == "bound")

# Read in isotope values for individual mixtures

mix.C <- subset(SI_data$d13C, SI_data$point == mixgroup)
mix.N <- subset(SI_data$d15N, SI_data$point == mixgroup)

# Evaluate every TDF combination and records combinations
# for which no ("outside == 0") mixtures lie outside the
# source polygon.
# ***Note: For some data sets, there may be no combinations
# that encompass all mixtures (i.e. some mixtures may
# always lie outside the source polygon). In these try
# numbers greater than zero (i.e. the minimum number of sources
# outside the polygon may be 1 or more). Examine "TDFmatrix"
# and revise ("outside == x") accordingly

for (i in 1:length(TDF.C)){

        for (j in 1:length(TDF.N)){

                within <- point.in.polygon((mix.C-TDF.C[i]),
                        (mix.N-TDF.N[j]),
                        bounds.C,
                        bounds.N)

                outside <- length(mix.N) - sum(within)
                TDFmatrix[i,j] <- TDFmatrix[i,j] + outside

                if (outside == 0){
                        TDFn <- c(TDFn,TDF.N[j])
                }

                if (outside == 0){
                        TDFc <- c(TDFc,TDF.C[i])
                }

        }
}

# Return estimated TDF means, standard deviation, minimum and maximums
```

```
mean(TDFc)
sd(TDFc)
max(TDFc)
min(TDFc)

mean(TDFn)
sd(TDFn)
max(TDFn)
min(TDFn)



####### PRINT DATA FILES (optional)

# To do so, enter desired file name and remove # symbol before running

# write.table(TDFc, file = "output_file_name_1.csv", quote = F, sep = ",", row.names = F)
# write.table(TDFn, file = "output_file_name_2.csv", quote = F, sep = ",", row.names = F)
# write.table(TDFmatrix, file = "output_file_name_3.csv", quote = F, sep = ",", row.names =
F)
```