Supplementary Material 3

Estimating fishing effort from highly resolved geospatial data: focusing on passive gears

The code presented here is a supplement to “Estimating fishing effort from highly resolved geospatial data: focusing on passive gears” by Tania Mendo, Gildas Glemarec, Jaime Mendo, Einar Hjorleifsson, Sophie Smout, Simon Northridge, Julien Rodriguez, Anna Mujal-Colilles and Mark James.

In this document we present a method for estimating gear soak time applie to a gillnet fisheries in Peru. This method is described in more detail in the paper. This fishery operates on a daily basis, and gear is set and hauled during the same fishing trip.

We provide a sample data set of vessel tracking data collected every 3 minutes from 3 trips by 3 different small scale fishing vessels.

The data can be found here, DOI: XXXXXXXXXXXX

#Install relevant libraries

First, call the libraries into R.

```
library(sf)
library(tidyverse)
library(sfheaders)
```

Estimating soak time

Input data: The columns needed to run the code below are: trip_id: a unique trip identifier time stamp: date and time x: longitude in UTM y: latitude in UTM dist: distance between consecutive observations rf_prediction: inferred vessel activity: hauling gear (fishing) or not hauling gear (not_fishing) activity_id: unique identifier for each hauling or deployment event.

#User input parameters

The following parameters need to be decided by the user, based on knowledge on each fishery.

```
crs_utm<-32717 #specify the UTM
dist_buffer<- 500 # distance around a hauling event which will overlap with a setting event
```

Estimate soak time

The following code will look at each trip, identify the hauling events, and allocate the most likely deployment event that can be matched to that hauling event.

```
df <- read.table("df.txt", sep=",") # 3 trips
df$date <- as.POSIXct((df$date), format = "%Y-%m-%d %H:%M:%S")

soak_time <- data.frame() ### create dataframe to store soak time estimates
```

```r
trips_list <- unique(df$trip_id) #create a list of trips

for (j in unique(trips_list)) {
 # j="A 2019-01-08"
  ### select a trip
  df_trip <- subset(df, df$trip_id == j)

  ### select only predicted fishing locations
  df_trip_fishing <- subset(df_trip, df_trip$rf_prediction == "fishing")

  ### transform to sf object, specify CRS
  df_trip_fishing <- st_as_sf(df_trip_fishing,
                              coords = c("x", "y"),
                              crs = crs_utm)

  ### create sf line objects, specify grouping by hauling event
  lines_hauling <- df_trip_fishing %>%
    dplyr::group_by(activity_id) %>%
    dplyr::summarise(do_union = FALSE) %>%
    st_cast("LINESTRING")

  ### Add a buffer around the hauling events to allow for potential distances
  ### between the net being hauled from a different position due to tide, or
  ### length of the rope connecting the net to the buoy
  lines_hauling_buff <- st_buffer(lines_hauling, dist = dist_buffer)

  ### select only predicted non fishing locations
  df_trip_not_fishing <- subset(df_trip, df_trip$rf_prediction == "not_fishing")

  ### transform to sf object, specify CRS
  df_trip_not_fishing <- st_as_sf(df_trip_not_fishing,
                                  coords = c("x", "y"), crs = crs_utm)

  ### Intersect the polygons on hauling activities with the lines for
  #non fishing activities
  Inter <- sf::st_intersection(lines_hauling_buff, df_trip_not_fishing)

  ### calculate distance of the intersection
  Inter$distance <- as.numeric(st_length(Inter))

  hauling_distance <- df_trip_fishing %>%
    dplyr::group_by(activity_id) %>%
    dplyr::summarise(distance_haul = sum(dist, na.rm = TRUE))

  ### add distance covered during hauling activities
  Inter <- left_join(Inter, as.data.frame(hauling_distance), by = "activity_id")

  ### For each non-fishing segment, select longest hauling event detected by model
  Inter <- Inter %>%
    group_by(activity_id.1) %>%
    top_n(1, distance) ### choose which hauling has the greatest distance of
  #intersection with hauling event
  ### Subset to keep only unique haul
  sel <- unique(Inter$activity_id)
  df_trip_fishing <- df_trip_fishing[df_trip_fishing$activity_id %in% sel,]

  ### Calculate the mean time stamp of the unique haul
  hauling_info <- df_trip_fishing %>%
    dplyr::group_by(activity_id, trip_id) %>%
    dplyr::summarise(mean_date_hauling = mean(date),
              activity_id = unique(activity_id))

   deployment_info <- as.data.frame(Inter) %>%
    dplyr::rename(hauling_id = activity_id.1,
                  setting_id = activity_id) %>%
    dplyr::group_by(setting_id) %>%
    dplyr::summarise(mean_date_setting = mean(date))

   hauling_info<-as.data.frame(hauling_info)

  soak <- hauling_info %>% left_join(deployment_info, by = c("activity_id" = "setting_id"))

  soak_time <- (rbind(soak_time, soak))

}

soak_time$soak_time <- soak_time$mean_date_hauling - soak_time$mean_date_setting

write.table(soak_time, "soak_time.txt", sep = ",")
```