Data Filtering Process, Condensed

Anna Steel

August 12, 2016

Filtering of VEMCO post-processed VPS positions

This code runs from the datafiles provided by VEMCO, and builds final datasets for analysis Also includes reference numbers to track how filtering affects the dataset

metrics considered include number of positions, number of fish in dataset, and positions per fish

- filters by HPE: see 'Exploration_DataFiltering' for details
- filters by speed:
- filters for likely predators:
- filters out sparse tracks:
- breaks apart tracks with large gaps (for subsequent smoothing/rediscretization)

Read in Data & Clean for proper dates and TagIDs

- Open script from Fremont16.Rproj in GitHub to ensure directories are correct
- add UTM coordinates
- filter out tags in the 65xxx series
- filter any tags detected outside of period of complete array (none this year)
- tabulate total fish, total positions, and total positions per fish

```
load("Maestros/alldf.RData")
options("digits.secs"=6)
alldf$Time <- as.POSIXct(as.character(alldf$Time), format="%Y-%m-%d %H:%M:%OS", tz = "GMT")

# remove fish tags in the 65xxx series (5 tags)
alldf <- alldf[alldf$Id<65000,]  # matches VEMCO

# incomplete array
alldfg <- alldf[alldf$Time > as.POSIXct("2016-02-109 14:00:00", tz="GMT"),]
 print(paste0(nrow(alldf) - nrow(alldfg)," positions removed due to incomplete VPS array"))
```

```
## [1] "0 positions removed due to incomplete VPS array"
```

```
# convert the Lat Long into UTMs using 'sp'
 alldf.sp <- SpatialPointsDataFrame(coords = alldf[,c("Longitude","Latitude")],
             data = alldf,
             proj4string=CRS("+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs"))
       # confirmed string with VEMCO; the XY coords in azimuthal equal area
 options(digits=10)
 alldf.utm <- spTransform(alldf.sp,
                          CRS("+proj=utm +zone=10 +datum=WGS84 +units=m +no_defs +ellps=WGS84 +towgs84=0
 alldf.utm@data[,c("east","north")] <- alldf.utm@coords
```

```
 alldf.utm = alldf.utm@data

# N fish & total N positions
print(paste0("Prior to filtering, ",nrow(alldf)," total positions in dataset"))
```

```
## [1] "Prior to filtering, 122971 total positions in dataset"
```

```
print(paste0("Prior to filtering, ",length(unique(alldf$Id))," individual fish positioned"))
```

```
## [1] "Prior to filtering, 433 individual fish positioned"
```

```
# N pos per fish
 all.npf = alldfg %>%
  group_by(Id) %>%
  summarize(npos.all=n())%>%
  data.frame()

 print("Summary of N positions per fish, after reducing to applicable data")
```

```
## [1] "Summary of N positions per fish, after reducing to applicable data"
```

```
 summary(all.npf$npos.all)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
##   37.0000  185.0000  238.0000  283.9977  317.0000 5577.0000
```

Primary HPE filter: <1.0 HPEs

After filtering @ HPEs < 1, dataset retained:

```
reddf = alldfg[alldfg$Hpes < 1,]

 # N fish & total N positions
  print(paste0(nrow(reddf)," positions")) # 110,211
```

```
## [1] "106529 positions"
```

```
   print(paste0("   ",round(nrow(reddf)/nrow(alldfg)*100,2), "% of fish tag positions"))  # 84.35%
```

```
## [1] "   86.63% of fish tag positions"
```

```
  print(paste0(length(unique(reddf$Id))," individual fish"))  # 438
```

```
## [1] "433 individual fish"
```

```
    print(paste0("   ",round((length(unique(reddf$Id))/length(unique(alldf$Id)))*100,1),"% of individual
```

```
## [1] "   100% of individual fish retained "
```

```
  # N pos per fish after HPE filtering
  red.npf = reddf %>%
   group_by(Id) %>%
   summarize(npos.red= n()) %>%
   data.frame()

  print("Summary of N positions per fish, after filtering at HPE<1")
```

```
## [1] "Summary of N positions per fish, after filtering at HPE<1"
```

```
  summary(red.npf$npos.red)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
##   28.0000  165.0000  212.0000  246.0254  283.0000 2015.0000
```

```
  save(reddf, file="Maestros/AllFishPrimaryFilt.RData")
```

Write out HPE filtered dataset

End of Primary Filtering Process

Beginning of Secondary Filtering Process

Excessive Speeds Filtering

```
  load("Maestros/AllFishPrimaryFilt.RData") # object called 'reddf'

  # Critical to order this properly for the ordered step analysis below (TypeI)
  reddf = reddf[order(reddf$Id, reddf$Time),]
  reddf$IDcol = 1:nrow(reddf)

  # convert the Lat Long into UTMs using 'sp'
  reddf.sp <- SpatialPointsDataFrame(coords = reddf[,c("Longitude","Latitude")],
              data = reddf,
              proj4string=CRS("+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs"))
        # confirmed string with VEMCO; the XY coords in azimuthal equal area
  options(digits=10)
  reddf.utm <- spTransform(reddf.sp,
                           CRS("+proj=utm +zone=10 +datum=WGS84 +units=m +no_defs +ellps=WGS84 +towgs84=0
```

```
reddf.utm@data[,c("east","north")] <- reddf.utm@coords

reddf2 = reddf.utm@data

# calculate speed and distance with adehabitatLT
red.ltraj = as.ltraj(xy=reddf2[,c("east","north")], date=reddf2$Time,
                     id=reddf2$Id, infolocs = reddf2[,c("Id","Hpes","east","north")])

red2 = ld(red.ltraj)
red2$spd_mps = red2$dist / red2$dt

# identify 'bad' positions (99%ile of step-speeds is ~7.7mps
red2$prevspd = lag(red2$dist)/lag(red2$dt)

red2$badpos <- 0
 red2$badpos[red2$spd_mps>7.7 & red2$prevspd>7.7] <- 1

# filter out bad positions
red3 = red2[red2$badpos==0,]

# recalculate speed and distance
red3.ltraj = as.ltraj(xy=red3[,c("east","north")], date=red3$date,
                     id=red3$Id, infolocs = red3[,c("Id","Hpes","east","north")])

red4 = ld(red3.ltraj)
red4$spd_mps = red4$dist / red4$dt

save(red4, file="Maestros/AllFish_FiltSec1Speed.RData")
```

Departed from and returned again to array

- Remove select bursts for fish returning to array after some time absent

```
dt_threshold = 6*3600  # 6 hours

dtcut = function(dt) { return (dt > dt_threshold) }

red4.ltraj = dl(red4)
red5.ltraj <- cutltraj(red4.ltraj, "dtcut(dt)", nextr=TRUE)

red5 = ld(red5.ltraj)

red5 = red5[!(red5$burst %in% c(36472.3, 36472.4, 36612.2, 36612.3)),]

save(red5, file="Maestros/AllFish_FiltSec2Pred.RData")
```

Suspect holding behavior

Should revisit this - these might be milling smolts, but might also be a predator - see "Exploration_DataFilteringPreds" for more on these three tracks

```
red6 = red5[!(red5$Id %in% c(36379,36483,36675)),]
save(red6, file="Maestros/AllFish_FiltSec3Hold.RData")
```

Cut tracks into sub-bursts when gaps are > a selected threshold to avoid interpolating between them

- for now I'll use 50m until I make more time to get into this deeper.
- see final few chunks in "Exploration_TrackGapBias" for more details on how to select this threshold.

```
dist_threshold = 50
gapcut = function(dist) { return (dist > dist_threshold) }

red6.ltraj = as.ltraj(xy=red6[,c("east","north")], date = red6$date, id = red6$Id, infolocs=red6[,c("Hp
red7.ltraj <- cutltraj(red6.ltraj, "gapcut(dist)", nextr=TRUE)
```

```
## Warning in cutltraj(red6.ltraj, "gapcut(dist)", nextr = TRUE): At least 3 relocations are needed for
##  103 relocations have been deleted
```

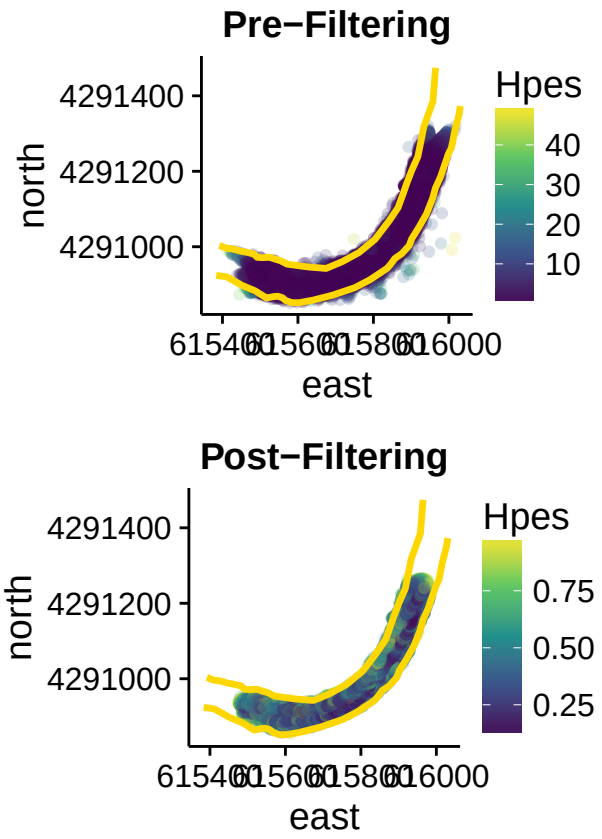
```
red7 = ld(red7.ltraj)
  red7$spd_mps = red7$dist / red7$dt

save(red7, file="Maestros/AllFish_FiltSec4Bursts.RData")
```

Plot of pre- & post-filtering positions

```
## OGR data source with driver: ESRI Shapefile
## Source: "C:/Users/Anna/Documents/GitHub/Fremont16/GIS/2004_channel", layer: "2004_channel_freTightcl
## with 2 features
## It has 1 fields
```

```
## Warning: Removed 5 rows containing missing values (geom_point).
```


Pre−Filtering
4291400
4291200
4291000
north
Hpes
40
30
20
10
615400
615600
615800
616000
east
Post−Filtering
4291400
4291200
4291000
north
Hpes
0.75
0.50
0.25
615400
615600
615800
616000
east