**Introduction**

The purpose of segmentation is to associate spatial point observations with a given period of survey effort. There are two major steps that make up this process:

1. Split effort track lines into distances equal to the target segment length
2. Pair observations to the segments by finding the line segment with the shortest distance from each observation (using projected coordinates)

Each observation is then associated with the midpoint of a segment (either geographic midpoint or distance traveled, which can be specified in the function). The result is a data frame with segments as rows and species names as columns, containing the abundance of each species on a particular segment. The primary key is a combination of survey, transect, and segment number, as segments are calculated within transects within surveys. Each segment represents the distance observed by a single individual. For transects where the total distance in not evenly divisible by the target segment length, the remaining distance is dealt with as follows:

1. If the extra distance is greater than or equal to half the target segment length, the extra distance is classified as its own segment with location on the transect randomly placed relative to other full length segments
2. If the extra distance is less than half of the target segment length, the extra distance is added randomly to an existing full length segment on the transect

Setting the cutoff threshold to half of the target segment length ensures the expected segment distance will be equal to the target segment distance. Note that in both cases above, exactly one segment on each transect will have a length that differs from the target length.

Other cases of note:

1. If data in the database are already in segmented form, the segments will be rejoined to recreate the initial track line. These track lines will then be segmented from scratch by the function. This is particularly helpful in cases when the target segment length differs from the target length used to originally segment the data.
2. Breaks in effort on a transect are segmented individually as separate pieces (i.e., midpoints cannot fall in off-effort periods).

DTS data:

DTS data cannot be truly “segmented” following the procedures above, as track information does not exist. However, it is still possible to obtain estimates of effort from the data that is available.

For consistency, only DTS transects with 10 and 15 minute survey durations are included. The on-effort distance is calculated by multiplying the vessel speed by the survey duration. When the vessel speed is unavailable, a speed of 10 knots is assumed. When DTS transect records consist of only a single point, the segment midpoint is calculated using the vessel heading and the total distance traveled. When the transect record contains two or more points, the segment midpoint is defined as the geographic midpoint of those points.

**NWASC Segmentation Instructions**

07/25/2016

This document contains instructions for segmenting data in the Northwest Atlantic Seabird Catalog.

Latest R version: 3.3.1

Required packages and latest versions: **CTS and DTS**;CTS only

broom: 0.4.1

**dplyr: 0.5.0**

**geosphere: 1.5-5**

**lubridate: 1.6.0**

**maptools: 0.8-39**

**rgeos: 0.3-20**

tibble: 1.2

**tidyr: 0.6.0**

zoo: 1.7-13

Required files and locations:

1. CTS
2. Q:/Kyle\_Working\_Folder/Segmentation/Data/database\_extract\_cts\_obs.RData

* Contains observation table and transect table in form of R objects ‘cts.dat’ and ‘obs.dat’
* Extracted from NWASC on 06/21/16 with the following script: Q:/Kyle\_Working\_Folder/Segmentation/From\_Arliss/Export\_R\_workspace\_for\_Kyle.R
* Required variables:
  1. cts.dat: **start\_dt, end\_dt, survey\_method\_cd, survey\_type\_cd, dataset\_id, source\_dataset\_id, transect\_id, segmented\_transect\_id, start\_tm, end\_tm, transect\_width\_nb, time\_from\_midnight\_start, time\_from\_midnight\_stop**
  2. obs.dat: **obs\_dt, obs\_start\_tm, dataset\_id, transect\_id, st\_astext, spp\_cd obs\_count\_intrans\_nb, time\_from\_midnight**

Note: st\_astext column is not necessary in ‘cts.dat’. Spatial information is taken from file in step 2 below. Only columns necessary for segmentation are included in tables.

1. Q:/Kyle\_Working\_Folder/ Segmentation/Data/transect\_shp\_files/line

* Contains effort shapefile with all multipart spatial line geometries
* Extracted from NWASC on 03/07/16

Note: only variables necessary in data attribute are ‘**transect\_i**’ and ‘**dataset\_id**’

1. R scripts:
   1. Pre-segmentation: Q:/Kyle\_Working\_Folder/ Segmentation/pre\_seg\_new.R
   2. Segmentation: Q:/Kyle\_Working\_Folder/ Segmentation/seg\_new.R
2. DTS
3. Q:/Kyle\_Working\_Folder/Segmentation/Data/database\_extract\_dts\_obs.RData

* Contains observation table and transect table in form of R objects ‘dts.dat’ and ‘obs.dat’
* Extracted from NWASC on 06/21/16 with the following script: Q:/Kyle\_Working\_Folder/Segmentation/From\_Arliss/Export\_R\_workspace\_for\_Kyle.R
* Required variables:
  1. dts.dat: **start\_dt, end\_dt, survey\_method\_cd, survey\_type\_cd, dataset\_id, source\_dataset\_id, transect\_id, segmented\_transect\_id, start\_tm, end\_tm, transect\_time\_min\_nb, traversal\_speed\_nb, transect\_width\_nb, heading\_tx, st\_astext, time\_from\_midnight\_start, time\_from\_midnight\_stop**
  2. obs.dat: Same as in CTS above

1. R script: Q:/Kyle\_Working\_Folder/Segmentation/process\_dts.R

Instructions:

1. CTS
2. Open pre-segmentation script and change path names for items 1 and 2 above as necessary.
3. Source pre-segmentation script (approx. run time: 10 min)

e.g., source(“Q:/Kyle\_Working\_Folder/ Segmentation/pre\_seg\_new.R”)

* This creates objects ‘**obs.pre**’ (variable names: **long, lat, transect\_id, spp\_cd, count**) and ‘**shp.pre**’ (variable names: **long, lat, piece, order, dataset\_id, transect\_id**) in the global environment to be used by the segmentation function.

1. Open segmentation script and run function (approx. run time: 20 min)

e.g., seg.dat.cts = segmentCTS(obs.pre, shp.pre, cts.dat)

* See comments in script header for instructions on optional arguments

This produces a wide-form data frame with column names: **source\_dataset\_id, segmented\_transect\_id, transect\_id, seg\_num, start\_dt, seg\_dist, transect\_width\_nb, mid\_long, mid\_lat, survey\_type\_cd, survey\_method\_cd**, plus a column for each species

1. DTS
2. Open script and change path name for item 1 above as necessary.
3. Run function (approx. run time: under 1 min)

e.g., seg.dat.dts = segmentDTS(obs.dat, dts.dat)

* See comments in script header for instructions on optional arguments

This produces a wide-form data frame with column names: **source\_dataset\_id, segmented\_transect\_id, transect\_id, start\_dt, seg\_dist, transect\_width\_nb, mid\_long, mid\_lat, survey\_type\_cd, survey\_method\_cd**, plus a column for each species

1. Objects seg.dat.cts and seg.dat.dts can be combined into a single data frame with code at end of “process\_dts.R” script.