library(move)

#bring in file from Movebank

laureen\_move <- move(x = "~/Desktop/R\_Forever/Dissertation/noha-move-hab/Data/SIMP 01.csv")

laureen\_bursted <- move::burst(laureen\_move, c('normal','long')[1+(timeLag(laureen\_move, units='mins')>72)])

laureen\_bursted\_trans <- spTransform(x = laureen\_bursted, CRSobj = '+proj=utm +zone=10 +datum=NAD83 +units=m', center = T)

proj4string(Suisun\_NLCD\_new) #this raster is incorrect

proj4string(suisun\_polygon\_new) # this polygon is incorrect

proj4string(laureen\_bursted\_trans) # need all layers to match this projection

#matching projections below

library(sf)

library(raster)

r <- raster(suisun\_polygon\_new)

#r <- setValues(r, 1:ncell(r))

newproj <- "+proj=utm +zone=10 +datum=NAD83 +units=m +ellps=GRS80 +towgs84=0,0,0 +lon\_0=-121.910674 +lat\_0=38.115666"

nlcd\_new <- projectRaster(Suisun\_NLCD\_new, crs = newproj)

#now they match, but still need the Suisun\_nlcd\_trans\_laureen layer to match, too

proj4string(nlcd\_new)

proj4string(laureen\_bursted\_trans)

#current projection - this is not working and we don't need this layer below anyway because the nlcd\_new and bursted\_trans files above match already

#Suisun\_nlcd\_trans\_laureen <- raster(Suisun\_NLCD\_new)

#crs(Suisun\_nlcd\_trans\_laureen) <- "+proj=utm +zone=10 +datum=NAD83 +units=m +ellps=GRS80 +towgs84=0,0,0 +lon\_0=-121.910674 +lat\_0=38.115666 "

#plot(Suisun\_nlcd\_trans\_laureen)

#now these match, too

#proj4string(Suisun\_nlcd\_trans\_laureen)

#proj4string(laureen\_bursted\_trans)

laureen\_dbbmm <- brownian.bridge.dyn(laureen\_bursted\_trans, burstType = 'normal', raster = nlcd\_new, location.error = 10, ext = .3, time.step = 60, margin = 3, window.size = 7) #location error is 10 m as per the transmitter specifications, extent is 30% of raster extent, time step is 60 mins becasue locations were approximately every hour, margin is 3 which is the minimum number of locations needed to calculate breakpoints a a leave-one-out approach, and window size is 7 because this is equivalent to 7 locations, which equals 7 hours and may be able to detect behavioral changes within this relatively short window.

## below are the UDs calculated from the dbbmm

laureen\_dbbmm\_UD<-new(".UD",calc(laureen\_dbbmm, sum)) ## it works!!!

#get the area of the 95% UD - i think these areas are in meters

laureen\_cont95 <- getVolumeUD(laureen\_dbbmm\_UD)

laureen\_cont95 <- laureen\_cont95<=.95

area95 <- sum(values(laureen\_cont95))

area95

#get the area of the 50% UD - i think these areas are in meters

laureen\_cont5 <- getVolumeUD(laureen\_dbbmm\_UD)

laureen\_cont5 <- laureen\_cont5<=.5

area5 <- sum(values(laureen\_cont5))

area5

#dbbmm dataframe- keep this!

dbbmm.df <- as.data.frame(laureen\_dbbmm\_UD, xy = TRUE)

#save UD raster

writeRaster(laureen\_dbbmm\_UD, "~/Desktop/R\_Forever/Dissertation/noha-move-hab/Output/laureen\_ud\_raster4.tif", overwrite = TRUE)

#save contours

cont\_new <-raster2contour(laureen\_dbbmm\_UD, level=c(.5,.95))

writeOGR(cont\_new, dsn = '.', layer = 'laureen\_contour\_new', driver = "ESRI Shapefile", overwrite\_layer = TRUE)

#using the nlcd\_new raster lines up with the correct number of rows and columns from our dbbmm dataframe, and we can extract the landcover values (finally!)

nlcd\_sp <- SpatialPoints(dbbmm.df[,1:2], proj4string = crs(nlcd\_new))

nlcd\_extract <- extract(Suisun\_NLCD\_new, nlcd\_sp)

head(nlcd\_extract)

nlcd\_extract[which(!is.na(nlcd\_extract))]

#check that the columns and rows match - they do

str(laureen\_dbbmm\_UD)

str(nlcd\_new)

# test to make sure it works - it does (red square represents the nlcd raster layer)

plot(laureen\_dbbmm\_UD)

library(scales)

plot(nlcd\_new, col = alpha("red", .5), add = TRUE)

# combine the raster cell probabilities with their coord pairs with landcover grid cells

final <- cbind.data.frame(dbbmm.df, nlcd\_extract)

head(final)

#above works, but returns all columns, including empty grid cells with NA and 0 values

# below code removes NA in the 4th column (the landcover column) and returns only columns with landcover values

final <- final[which(!is.na(final[,4])),]

head(final)

# for loop to calculate probabilities of use within each landcover types using the UDs

prob.vec <- rep(NA, length(unique(final[,4])))

unique.vec <- unique(final[,4])

tot.prob <- sum(final[,3])

for (i in 1:length(prob.vec)){

prob.vec[i] <- sum(final[which(final[,4] == unique.vec[i]),3])/tot.prob

}

#check that the for loop worked and the probabilities sum to 1 - they do

sum(prob.vec)

#save the probability table for each landcover class - it works!

probs.cover.tables <- cbind(prob.vec, unique.vec)

#view the entire table

probs.cover.tables

write.csv(probs.cover.tables, file = "laureen\_landcover\_probs\_final.csv")