## Bre NOHA 30.1 with new suisun raster

library(move)

#bring in file from Movebank

Bre\_move <- move(x = "~/Desktop/R\_Forever/Dissertation/noha-move-hab/Data/NOHA 30.1.csv")

Bre\_bursted <- move::burst(Bre\_move, c('normal','long')[1+(timeLag(Bre\_move, units='mins')>35)])

Bre\_bursted\_trans <- spTransform(x = Bre\_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(Bre\_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=-122.0374075 +lat\_0=38.2021575"

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

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

proj4string(nlcd\_new\_Bre)

proj4string(Bre\_bursted\_trans)

Bre\_dbbmm <- brownian.bridge.dyn(Bre\_bursted\_trans, burstType = 'normal', raster = nlcd\_new\_Bre, location.error = 10, ext = .3, time.step = 60, margin = 3, window.size = 7)

## below are the UDs calculated from the dbbmm

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

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

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

Bre\_cont95 <- Bre\_cont95<=.95

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

area95

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

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

Bre\_cont5 <- Bre\_cont5<=.5

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

area5

#dbbmm dataframe- keep this!

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

#save UD raster

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

#save contours

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

writeOGR(cont\_new, dsn = '.', layer = 'Bre\_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(Bre.dbbmm.df[,1:2], proj4string = crs(nlcd\_new\_Bre))

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(Bre\_dbbmm\_UD)

str(nlcd\_new\_Bre)

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

plot(Bre\_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(Bre.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 = "Bre\_landcover\_probs\_final\_new.csv")