Bolas\_WFC198\_Hwk4

Ellie Bolas

April 30, 2018

setwd("C:/Users/ebola/Google Drive/Git/WFC198\_Git/WFC198/Lab4\_Hwk4\_DistanceSampling")

## Distance Sampling

-data set of rabbits, sampled over 4 transects, 2 in good habitat and two in poor habitat -The data has 4 rows, corresponding to the four transects. The first 28 columns (column names X1 through X28) of the data are the number of rabbit detections in 28 5-m distance intervals. Column number 29, named “hab”, holds a binary variable coding habitat quality; 1 stands for good habitat, 0 stands for poor habitat. The last column, named “tlength”, provides the length of each transect.

## Task 1: Data summary statistics (3 Pts)

1. Read the data into R (1 Pt)

library(unmarked)

## Warning: package 'unmarked' was built under R version 3.3.3

## Loading required package: reshape

## Warning: package 'reshape' was built under R version 3.3.3

## Loading required package: lattice

## Loading required package: parallel

## Loading required package: Rcpp

getwd()

## [1] "C:/Users/ebola/Google Drive/Git/WFC198\_Git/WFC198/Lab4\_Hwk4\_DistanceSampling"

rabbit<- read.csv("RabbitDSData.csv")

1. Calculate the total number of rabbit detections across all distances and transects (1 Pt)

#remember for subsetting its [rows,columns]  
  
allrabbits<-sum(rabbit[,c(1:28)]) #means sum all rows and columns 1-28  
allrabbits

## [1] 110

1. Calculate the number of rabbits detected in good habitat and in poor habitat (1 Pts)

#good habitat is the first two rows with 1, poor is the second two rows with 0  
#good habitat  
goodrabbits<-sum(rabbit[c(1:2), c(1:28)])  
goodrabbits

## [1] 61

#poor habitat  
poorrabbits <-allrabbits-goodrabbits  
poorrabbits

## [1] 49

## Task 2: Creating the unmarkedFrame (6 Pts)

1. For the next step (2b) you need the distance breaks, so set up a numeric vector that holds the 5-m distance breaks for the 28 distance categories, starting at 0m and going out to 140m. (2 Pts)

#doing this bc for unmarked, distances have to be binned into discrete distance classes  
distr <- seq(0, 140, 5)  
length(distr) #returns 29, which is actually necessary for the unmarkedFrame

## [1] 29

1. Use the unmarkedFrameDS() function to get the data into the right format for analysis with the distsamp() function. Use the R help and the R script from the Distance sampling lab for a reminder of how to specify the function arguments. Remember: this is line transect data, and distances are measured in meters. Also, we have a categorical covariate (good vs bad habitat) here, which should not be scaled. We need to specify one additional argument (compared to what we did in lab), and that is transect length. The argument to do that within the unmarkedFrameDS() function is “tlength”. Transect length has to be provided in the form of a vector; lengths of the four transects are stored in the column “tlength”, so in the unamrkedFrameDS() functions, set tlength=DataObject$tlength [note: use the name of the object holding the rabbit data in place of “DataObject”]. (4 Pts)

umfr <- unmarkedFrameDS(y = as.matrix(rabbit[,1:28]), #y is detection data, makes an RxJ matrix. R is number of sites/transects (all rows bc each row is one transect), J is number of distance classes, so that's the first 28 columns  
 siteCovs = data.frame(rabbit$hab), #this is the site covariates which is just poor/good habitat, so all rows and one column. However, when I put it in like rabbit[,29] it was hard to pull out this covariate later, so now I'm just telling it this column  
 dist.breaks = distr, #distance breaks is a vector J+1, so it's ok that what we made previously had 29 slots  
 tlength= rabbit$tlength,   
 survey = "line",  
 unitsIn = "m"  
 )  
  
summary(umfr)

## unmarkedFrameDS Object  
##   
## line-transect survey design  
## Distance class cutpoints (m): 0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 105 110 115 120 125 130 135 140   
##   
## 4 sites  
## Maximum number of distance classes per site: 28   
## Mean number of distance classes per site: 28   
## Sites with at least one detection: 4   
##   
## Tabulation of y observations:  
## 0 1 2 3 4 5 6 8 <NA>   
## 66 19 9 9 3 4 1 1 0   
##   
## Site-level covariates:  
## rabbit.hab   
## Min. :0.0   
## 1st Qu.:0.0   
## Median :0.5   
## Mean :0.5   
## 3rd Qu.:1.0   
## Max. :1.0

#I think the following code is identifcal to above, in terms of getting the habitat variable in, but using the below just in case  
  
hab <- data.frame(hab = rabbit[,29])  
  
umfrb <- unmarkedFrameDS(y = as.matrix(rabbit[,1:28]), #y is detection data, makes an RxJ matrix. R is number of sites/transects (all rows bc each row is one transect), J is number of distance classes, so that's the first 28 columns  
 siteCovs = data.frame(hab), #this is the site covariates which is just poor/good habitat, so all rows and one column. However, when I put it in like rabbit[,29] it was hard to pull out this covariate later, so now I'm just telling it this column  
 dist.breaks = distr, #distance breaks is a vector J+1, so it's ok that what we made previously had 29 slots  
 tlength= rabbit$tlength,   
 survey = "line",  
 unitsIn = "m"  
 )  
  
summary(umfrb)

## unmarkedFrameDS Object  
##   
## line-transect survey design  
## Distance class cutpoints (m): 0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 105 110 115 120 125 130 135 140   
##   
## 4 sites  
## Maximum number of distance classes per site: 28   
## Mean number of distance classes per site: 28   
## Sites with at least one detection: 4   
##   
## Tabulation of y observations:  
## 0 1 2 3 4 5 6 8 <NA>   
## 66 19 9 9 3 4 1 1 0   
##   
## Site-level covariates:  
## hab   
## Min. :0.0   
## 1st Qu.:0.0   
## Median :0.5   
## Mean :0.5   
## 3rd Qu.:1.0   
## Max. :1.0

## Task 3: Fitting distance sampling models (8 Pts)

1. Fit the following six models: (i) no covariates on detection or abundance, with a half-normal detection function; (ii) no covariates on detection or abundance, with a hazard detection function; (iii) habitat as a covariate on abundance, with a half-normal detection function; (iv) habitat as a covariate on abundance, with a hazard detection function; (v) habitat as a covariate on detection, with a half-normal detection function; (vi) habitat as a covariate on detection, with a hazard detection function (6 pts)

11: no covariates on detection or abundance X1: covariate X on detection, none on abundance 1X: no covariate on detection, X on abundance

#i) half normal, no covariates  
hn <- distsamp(~1~1, umfrb, keyfun = "halfnorm", output = "density", units = "ha")  
summary(hn) #returns info on log scale

##   
## Call:  
## distsamp(formula = ~1 ~ 1, data = umfrb, keyfun = "halfnorm",   
## output = "density", unitsOut = "ha")  
##   
## Density (log-scale):  
## Estimate SE z P(>|z|)  
## 0.179 0.117 1.53 0.127  
##   
## Detection (log-scale):  
## Estimate SE z P(>|z|)  
## 3.54 0.0678 52.3 0  
##   
## AIC: 232.4596   
## Number of sites: 4  
## optim convergence code: 0  
## optim iterations: 31   
## Bootstrap iterations: 0   
##   
## Survey design: line-transect  
## Detection function: halfnorm  
## UnitsIn: m  
## UnitsOut: ha

#ii) hazard, no covariates  
hz <- distsamp(~1~1, umfrb, keyfun = "hazard", output = "density", units = "ha")  
summary(hz)

##   
## Call:  
## distsamp(formula = ~1 ~ 1, data = umfrb, keyfun = "hazard", output = "density",   
## unitsOut = "ha")  
##   
## Density (log-scale):  
## Estimate SE z P(>|z|)  
## 0.153 0.123 1.25 0.212  
##   
## Detection (log-scale):  
## Estimate SE z P(>|z|)  
## 3.56 0.103 34.6 7.03e-262  
##   
## Hazard-rate(scale) (log-scale):  
## Estimate SE z P(>|z|)  
## 1.23 0.158 7.79 6.52e-15  
##   
## AIC: 227.6153   
## Number of sites: 4  
## optim convergence code: 0  
## optim iterations: 38   
## Bootstrap iterations: 0   
##   
## Survey design: line-transect  
## Detection function: hazard  
## UnitsIn: m  
## UnitsOut: ha

#iii) half normal, habitat covariate on abundance  
hn\_hab\_abun <- distsamp(~1~hab, umfrb, keyfun = "halfnorm", output = "density", units = "ha")   
summary(hn\_hab\_abun)

##   
## Call:  
## distsamp(formula = ~1 ~ hab, data = umfrb, keyfun = "halfnorm",   
## output = "density", unitsOut = "ha")  
##   
## Density (log-scale):  
## Estimate SE z P(>|z|)  
## (Intercept) -0.215 0.158 -1.36 1.74e-01  
## hab 0.884 0.192 4.61 4.06e-06  
##   
## Detection (log-scale):  
## Estimate SE z P(>|z|)  
## 3.54 0.0678 52.3 0  
##   
## AIC: 213.2255   
## Number of sites: 4  
## optim convergence code: 0  
## optim iterations: 41   
## Bootstrap iterations: 0   
##   
## Survey design: line-transect  
## Detection function: halfnorm  
## UnitsIn: m  
## UnitsOut: ha

#iv) hazard, habitat covariate on abundance  
hz\_hab\_abun <- distsamp(~1~hab, umfrb, keyfun = "hazard", output = "density", units = "ha")  
summary(hz\_hab\_abun)

##   
## Call:  
## distsamp(formula = ~1 ~ hab, data = umfrb, keyfun = "hazard",   
## output = "density", unitsOut = "ha")  
##   
## Density (log-scale):  
## Estimate SE z P(>|z|)  
## (Intercept) -0.240 0.163 -1.48 1.39e-01  
## hab 0.884 0.192 4.61 4.06e-06  
##   
## Detection (log-scale):  
## Estimate SE z P(>|z|)  
## 3.56 0.103 34.6 7.04e-262  
##   
## Hazard-rate(scale) (log-scale):  
## Estimate SE z P(>|z|)  
## 1.23 0.158 7.79 6.52e-15  
##   
## AIC: 208.3813   
## Number of sites: 4  
## optim convergence code: 0  
## optim iterations: 57   
## Bootstrap iterations: 0   
##   
## Survey design: line-transect  
## Detection function: hazard  
## UnitsIn: m  
## UnitsOut: ha

#v) half normal, habitat on detection  
hn\_hab\_det<- distsamp(~hab~1, umfrb, keyfun = "halfnorm", output = "density", units = "ha")  
  
#vi) hazard, habitat on detection  
hz\_hab\_det <- distsamp(~hab~1, umfrb, keyfun = "hazard", output = "density", units = "ha")

1. Using AIC, determine the best model out of the six models fit under a (make sure your knitted report shows the model selection table). (2 Pts)

#fit list to compare models, just list all models by name  
rabbitfl <- fitList(hn, hz, hn\_hab\_abun, hz\_hab\_abun, hn\_hab\_det, hz\_hab\_det)

## Warning in fitList(hn, hz, hn\_hab\_abun, hz\_hab\_abun, hn\_hab\_det,  
## hz\_hab\_det): Your list was unnamed, so model names were added as object  
## names

rabbit\_modsel<-modSel(rabbitfl)  
rabbit\_modsel

## nPars AIC delta AICwt cumltvWt  
## hz\_hab\_abun 4 208.38 0.00 9.2e-01 0.92  
## hn\_hab\_abun 3 213.23 4.84 8.1e-02 1.00  
## hz 3 227.62 19.23 6.1e-05 1.00  
## hz\_hab\_det 4 227.78 19.40 5.6e-05 1.00  
## hn 2 232.46 24.08 5.4e-06 1.00  
## hn\_hab\_det 3 234.28 25.90 2.2e-06 1.00

#best model is habitat as a covariate on abundance as a hazard function

## Task 4: Results interpretation (10 Pts)

1. Produce a model summary for the best model as selected under 3b (1 Pt)

summary(hz\_hab\_abun)

##   
## Call:  
## distsamp(formula = ~1 ~ hab, data = umfrb, keyfun = "hazard",   
## output = "density", unitsOut = "ha")  
##   
## Density (log-scale):  
## Estimate SE z P(>|z|)  
## (Intercept) -0.240 0.163 -1.48 1.39e-01  
## hab 0.884 0.192 4.61 4.06e-06  
##   
## Detection (log-scale):  
## Estimate SE z P(>|z|)  
## 3.56 0.103 34.6 7.04e-262  
##   
## Hazard-rate(scale) (log-scale):  
## Estimate SE z P(>|z|)  
## 1.23 0.158 7.79 6.52e-15  
##   
## AIC: 208.3813   
## Number of sites: 4  
## optim convergence code: 0  
## optim iterations: 57   
## Bootstrap iterations: 0   
##   
## Survey design: line-transect  
## Detection function: hazard  
## UnitsIn: m  
## UnitsOut: ha

1. For that same model, and using the appropriate R command, calculate rabbit density in good and in poor habitat. Note: calculate density on the real scale, not the link scale! Make sure your knitted report shows the code as well as the numeric result of the calculation (2 Pts)

#use parameter estimates to calculate expected log(density) then backTransform, predict does both for you  
#log(lambda) = beta0 + beta1\*x  
#log(lambda) = -0.240 + 0.884\*x is good habitat  
#log(lambda) = -0.240 is poor habitat  
  
pred.hab <- predict(hz\_hab\_abun, "state") #state means transform to natural state  
pred.hab #this returns a prediction for each row, so both the good and poor habitat density predictions (second two rows are poor habitat in original df)

## Predicted SE lower upper  
## 1 1.9034424 0.2850415 1.419292 2.552746  
## 2 1.9034424 0.2850415 1.419292 2.552746  
## 3 0.7863408 0.1278618 0.571745 1.081482  
## 4 0.7863408 0.1278618 0.571745 1.081482

1. What is the ecological interpretation of these quantities? Write a short answer into your R script that contains the estimates (with standard errors) produced under 4b – remember to comment out anything that is not a command! (2Pts)

A: Habitat has a positive relationship with abundance that is statistically significant. In a unit of good habitat, there is an effect size of 1.9 increase in rabbits, with a standard error of 0.29. In a unit of poor habitat, there is only a 0.79 increase in rabbits, with a standard error of 0.13.

1. Plot the detection function for the best model over a range of distances from 0 to 140 m. Tip: We developed code to plot a detection function early on in the Distance sampling lab. In that code, you need to specify the parameters of the detection function as they were estimated by the best model. Remember to back-transform model estimates to the real scale. Specify distances at 1-m intervals. Label your axes. (5 Pts)

# Hazard fxn formula  
#p = g(x) = 1-exp(-(x/sigma)^-b)  
#need scale paramter sigma and shape parameter b to plot detection. summary(hz\_hn\_abun) returns those parameters on the log scale, so we need to back transform  
  
names(hz\_hab\_abun)

## [1] "state" "det" "scale"

#need to back transform all the detection function info to get paramters for plotting  
  
shape.b <- backTransform(hz\_hab\_abun, type = "det") # gives shape parameter b  
shape.b #35

## Backtransformed linear combination(s) of Detection estimate(s)  
##   
## Estimate SE LinComb shape(Intercept)  
## 35 3.6 3.56 1  
##   
## Transformation: exp

scale.sigma <- backTransform(hz\_hab\_abun, type = "scale") #gives scale parameter sigma  
scale.sigma #3.43

## Backtransformed linear combination(s) of Hazard-rate(scale) estimate(s)  
##   
## Estimate SE LinComb scale  
## 3.43 0.543 1.23 1  
##   
## Transformation: exp

b.r<- 35  
sigmahaz <- 3.43  
distz <- seq(1,140) #made new distance on 1 m scale  
  
#formula for   
dethaz <-1- exp(-(distz/sigmahaz)^-b.r) #formula for detection probability on real scale  
head(dethaz)

## [1] 1.000000e+00 1.000000e+00 1.000000e+00 4.594042e-03 1.867857e-06  
## [6] 3.162281e-09

plot(distz, dethaz, type = "l", xlab = "Distance", ylab = "Probability of Detection", ylim= c(0,1))

