R code for Southern Lakes grizzly analysis (updated 2017-03-15)

Murray Efford and John Boulanger

2017-03-15

Contents

Introduction

This RMarkdown vignette contains the R code for the main analyses. It uses a pre-release version of the R package **secr**, labelled here **secr3**. The package is expected to be published on the CRAN repository in about April 2017. The R packages **rgdal**, **rgeos** and **raster** were used for particular geographic data manipulations (shapefile import, projections, polygon union, cropping). The package **gdata** is used to import Excel spreadsheets.

Code to generate figures in the report appears at the end of this document. A list is also provided of files containing intermediate results. Some times-consuming code chunks here have the option 'eval = FALSE'; the intermediate files allow this document to be 'knitted' without setting eval = TRUE.

Data sources

Data were provided by the Yukon Department of Environment in Excel spreadsheets and ESRI shape files.

Hair snare data

The main data file was 'SLGB_Snare and DNA data_Nov2016_JG_AM.xls'. This contained separate sheets for the snare locations ('Snare_Sites') and records of individual bears ('Individual_GBs').

Telemetry data

Records of GPS collaring were provided in two spreadsheets, 'SL_Collar_Summary_Sept2016_JG.xls' and 'SL_ALL_COLLARS_for JBME_updated19Sept2016.xls'

Study area

The study area was defined in an ESRI polygon shapefile 'SL_StudyArea_by_ecozone.shp'. This comprised three ecozone polygons for each of the ecozones 'Yukon Southern Lakes' (4696 km^2), 'Yukon-Stikine Highlands' (3098 km^2) and 'Boreal Mountains and Plateaus' (65 km^2). A single study area polygon was formed from their union. A further polygon shapefile '2012DNAGrid3.shp' defined the sampling grid (unused marginal grid cells were omitted in Fig. 1 of the main text).

Elevation data were obtained from the digital elevation model of Natural Resources Canada (https://www.nrcan.gc.ca/earth-sciences/geography/topographic-information/free-data-geogratis/11042). Two overlapping downloads were needed to cover the buffered study area.

Ecoregion and Ecodistrict polygons were downloaded as shapefiles from http://sis.agr.gc.ca/cansis/nsdb/ecostrat/gis_data.html.

Preliminaries

First, load required packages.

```
library(secr3)
library(rgdal)    # read shapefiles, transform coordinates
library(rgeos)    # polygon area
library(gdata)    # import Excel spreadsheets
library(raster)   # crop
library(chron)    # telemetry datetime
```

Next, assign folder names. These will depend on where data have been stored.

```
datadir <- 'd:/bears/yukon/southern lakes'                # hair snare & GPS collar spreadsheets
# SLGB_Snare and DNA data_Nov2016_JG_AM.xls
# SL_ALL_COLLARS_for JBME_updated19Sept2016.xls
# SL_Collar_Summary_Sept2016_JG.xls
GISdir <- paste0(datadir, '/SL - GIS')                    # study area & related GIS files
# SL_StudyArea_by_ecozone.shp  Yukon Dept Environment
# 2012DNAGrid3.shp             Yukon Dept Environment
# AC_1M_Waterbodies.shp
# ecodistricts.shp
# ecoregions.shp
# SLdem1.tif                   GeoGratis digital elevation model
# SLdem2.tif                   GeoGratis digital elevation model
workdir <- 'd:/bears/yukon/southern lakes/work'           # folder for files generated
perlexe <- 'c:/perl64/bin/perl'   # used by gdata::read.xls
```

Name the spreadsheets to be used.

```
xlsname <- paste0(datadir, '/SLGB_Snare and DNA data_Nov2016_JG_AM.xls')
capturesheetname <- 'Individual_GBs'
snaresheetname <- 'Snare_Sites'
```

Import snare site spreadsheet

```
# skip=1 corrects for extra header line
sitedf <- read.xls (xls = xlsname, sheet = snaresheetname, header = TRUE,
                perl = perlexe, stringsAsFactors = FALSE,
                 na.strings = c("NA", "#DIV/0!", "", " ", "#VALUE!"))

message('Column names as imported')
```

```
## Column names as imported
```

```
names(sitedf)
message('Column class as imported')
```

```
## Column class as imported
```

```
sapply(sitedf, class)
message('Number of sites')
```

```
## Number of sites
```

```
nrow(sitedf)

# drop two missing sites 2013/14, 2013/59
sitedf <- sitedf[!is.na(sitedf$Latitude_DecDeg.),]
```

```
# drop duplicate site in 2012
sitedf <- sitedf[!(sitedf$Station_ID==43 & sitedf$Waypt_No==546),]

# unprojected site coordinates
sites <- sitedf[,c("Longitude_DecDeg.","Latitude_DecDeg.")]
# use Station_ID as rownames
rownames(sites) <- paste(sitedf$Year,sitedf$Station_ID, sep = '.')
tmp <- SpatialPointsDataFrame(sites, data = sitedf)
proj4string(tmp) <- CRS("+init=epsg:4326")       # initially long-lat WGS84
# write file for Google Earth (all snare sites)
kmlname <- paste0(workdir,'/YukonSL.kml')
if (file.exists(kmlname)) file.remove(kmlname)
writeOGR(tmp, kmlname, 'YukonSL', driver = 'KML')
tmp <- spTransform(tmp, CRS("+init=epsg:3578"))  # Yukon Albers
sitexy <- data.frame(coordinates(tmp))
names(sitexy) <- c('x','y')
sitexy <- split(sitexy, substring(rownames(sitexy),1,4))
siteobj <- lapply(sitexy, read.traps, file = NULL, detector = 'proximity')
class(siteobj) <- c('list','traps') # restore correct class
```

```
##  [1] "Year"                "Station_ID"          "Set_Date"
##  [4] "St_Time"             "End_Date"            "Session_1"
##  [7] "St_Time.1"           "End_Date.1"          "Session_2"
## [10] "St_Time.2"           "End_Date.2"          "Session_3"
## [13] "St_Time.3"           "End_Date.3"          "Session_4"
## [16] "Set_up_crew."        "Waypt_No"            "Latitude_DecDeg."
## [19] "Longitude_DecDeg."   "Elevation_m."        "Bear_Sign"
## [22] "Bear_Foods"          "Veg_Community"       "Area_Description"
## [25] "Station_Type."       "Camera_Left_At_Site" "Set_Up_Comments."
## [28] "Other_Comments.."    "X"                   "X.1"
## [31] "X.2"                 "X.3"                 "X.4"
## [34] "X.5"
##                Year          Station_ID            Set_Date
##           "integer"           "integer"         "character"
##             St_Time            End_Date            Session_1
##         "character"         "character"           "integer"
##           St_Time.1          End_Date.1            Session_2
##         "character"         "character"           "integer"
##           St_Time.2          End_Date.2            Session_3
##         "character"         "character"         "character"
##           St_Time.3          End_Date.3            Session_4
##         "character"         "character"           "integer"
##        Set_up_crew.            Waypt_No    Latitude_DecDeg.
##         "character"           "integer"           "numeric"
##   Longitude_DecDeg.        Elevation_m.            Bear_Sign
##           "numeric"         "character"         "character"
##          Bear_Foods       Veg_Community    Area_Description
##         "character"         "character"         "character"
##       Station_Type. Camera_Left_At_Site    Set_Up_Comments.
##         "character"         "character"         "character"
##    Other_Comments..                   X                 X.1
##         "character"           "logical"           "logical"
##                 X.2                 X.3                 X.4
##           "logical"           "logical"           "logical"
```

```
##                 X.5
##           "logical"
## [1] 341
## [1] TRUE
```

Import capture spreadsheet

```
# skip=1 avoids extra header line
df <- read.xls (xls = xlsname, sheet = capturesheetname, header = TRUE, perl = perlexe,
                stringsAsFactors = FALSE, skip = 1,
                na.strings = c("NA", "#DIV/0!", "", " ", "#VALUE!"))

names(df) <- c('Individual','Sample','empty', 'Collection','Site','Session','Date',
               'Origin','Age', 'empty2','Species','NumLoci','Sex','SEX')
```

```
df$Date <- as.Date(df$Date, '%m/%d/%Y')
df$Year <- as.POSIXlt(df$Date)$year + 1900
# form integer occasions ('sessions')
table(df$Year, df$Session, exclude = NULL)
```

```
##
##          1   2   3   4 capture/incidental n/a <NA>
##   2012  70 119 161 104                   0   0    1
##   2013  49  92  81  58                   0   3    0
##   <NA>   0   0   0   0                  13  20    8
```

```
# deliberately turn non-numeric values into NA
df$Session <- as.numeric(df$Session)
```

```
## Warning: NAs introduced by coercion
```

```
# drop undated incidental records
df <- df[!is.na(df$Date),]
df <- df[!is.na(df$Session),]
# drop rub tree records (including one with no Site)
table(df$Origin)
```

```
##
## broken sign post at station       incidental to station
##                           1                           3
##         rub tree at station         rub tree near Stn 4
##                          13                           1
##             sign at station                     station
##                           2                         714
```

```
df <- df[df$Origin == 'station',]
table(df$Year, df$Session, exclude = NULL)
```

```
##
##          1   2   3   4 <NA>
##   2012  67 111 158 104    0
##   2013  47  90  79  58    0
##   <NA>   0   0   0   0    0
```

```
df$Site <- paste(df$Year, df$Site, sep='.')
```

Build capthist object

Here we form a multi-session (i.e. multi-year) secr capthist object from the DNA data and the previously prepared site locations in 'siteobj'.

```
SLCH <- make.capthist(df[,c('Year','Individual','Session', 'Site','SEX')], traps = siteobj)
# drop duplicate detections of an individual at the same snare on one occasion
for (sess in 1:2) SLCH[[sess]][SLCH[[sess]]>1] <- 1
save(SLCH, siteobj, file = paste0(workdir, '/SLCH.RData'))
summary(SLCH, terse = TRUE)
```

```
##             2012 2013
## Occasions      4    4
## Detections   188  129
## Animals       75   65
## Detectors    169  167
```

Investigating long moves

```
M1 <- moves(SLCH[[1]])
M2 <- moves(SLCH[[2]])
M1[sapply(M1, max)>40000]  # 2012
```

```
## $`7469`
## [1] 14114.48 75474.52 75474.52
```

```
M2[sapply(M2, max)>40000]  # 2013
```

```
## $`6053`
## [1]  7721.191 60175.671 61058.929
##
## $SL031
## [1] 61058.93
```

```
df[df$Individual=='7469',]
```

```
##     Individual Sample empty Collection     Site Session       Date  Origin
## 26        7469   7468    NA         15  2012.31       1 2012-06-18 station
## 27        7469   7369    NA         15  2012.31       1 2012-06-18 station
## 28        7469   7381    NA         15  2012.31       1 2012-06-18 station
## 29        7469   7470    NA         15  2012.31       1 2012-06-18 station
## 30        7469   7469    NA         15  2012.31       1 2012-06-18 station
## 31        7469   7366    NA         15  2012.31       1 2012-06-18 station
## 166       7469   6553    NA        106  2012.55       2 2012-06-28 station
## 167       7469   6584    NA        106  2012.55       2 2012-06-28 station
## 168       7469   6583    NA        106  2012.55       2 2012-06-28 station
## 192       7469   7426    NA        136 2012.147       2 2012-07-01 station
## 226       7469   6487    NA        158  2012.55       3 2012-07-08 station
## 528       7469   9862    NA        685 2013.115       4 2013-07-18 station
##      Age empty2 Species NumLoci    Sex SEX Year
```

```
## 26  <NA>     NA gb G10J       8 204.25   M 2012
## 27  <NA>     NA gb G10J      21 204.25   M 2012
## 28  <NA>     NA gb G10J       8 204.25   M 2012
## 29  <NA>     NA gb G10J       8 204.25   M 2012
## 30  <NA>     NA gb G10J      21 204.25   M 2012
## 31  <NA>     NA gb G10J       8 204.25   M 2012
## 166 <NA>     NA gb G10J       8 204.25   M 2012
## 167 <NA>     NA gb G10J       8 204.25   M 2012
## 168 <NA>     NA gb G10J       8 204.25   M 2012
## 192 <NA>     NA gb G10J      10 204.25   M 2012
## 226 <NA>     NA gb G10J       8 204.25   M 2012
## 528  unk     NA gb G10J       8 204.25   M 2013
```

```
df[df$Individual=='SL031',]
```

```
##     Individual Sample empty Collection     Site Session       Date  Origin
## 681      SL031  10636    NA        604   2013.5       3 2013-07-08 station
## 723      SL031   9686    NA        570 2013.102       3 2013-07-10 station
##     Age empty2 Species NumLoci    Sex SEX Year
## 681 unk     NA gb G10J       8 250.25   F 2013
## 723 unk     NA gb G10J       8 250.25   F 2013
```

```
df[df$Individual=='6053',]
```

```
##     Individual Sample empty Collection     Site Session       Date  Origin
## 42        6053   6053    NA         33   2012.3       1 2012-06-17 station
## 622       6053   9266    NA        475  2013.83       2 2013-07-01 station
## 625       6053   9274    NA        476 2013.101       2 2013-07-01 station
## 682       6053  10643    NA        604   2013.5       3 2013-07-08 station
## 721       6053   9682    NA        570 2013.102       3 2013-07-10 station
## 722       6053   9684    NA        570 2013.102       3 2013-07-10 station
##      Age empty2 Species NumLoci    Sex SEX Year
## 42  <NA>     NA gb G10J      21 204.25   M 2012
## 622  unk     NA gb G10J       8 204.25   M 2013
## 625  unk     NA gb G10J       8 204.25   M 2013
## 682  unk     NA gb G10J       8 204.25   M 2013
## 721  unk     NA gb G10J       8 204.25   M 2013
## 722  unk     NA gb G10J       8 204.25   M 2013
```

Female SL031 was detected at sites 5 and 102 61 km apart in the same sampling interval (3) in 2013 (she was ultimately shot in September that year "in defense of life or property" DLP). Telemetry data confirm a long-distance movement. Male 6053 was also detected at both these stations in the same interval. As SL031 was mature and not translocated with cubs, it appears likely that a consort male accompanied her on a homing movement.

We have no particular reason to doubt the long-distance movement of 7469 (75 km 2012).

GIS work

Import study area shapefiles

```
Studyareafile <- paste0(GISdir, "/SL_StudyArea_by_ecozone.shp")
Studyarea <- readOGR(dsn = Studyareafile, layer = 'SL_StudyArea_by_ecozone')
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "d:/bears/yukon/southern lakes/SL - GIS/SL_StudyArea_by_ecozone.shp", layer: "SL_StudyArea_by
## with 3 features
## It has 15 fields
```

```
Studyarea1 <- gUnaryUnion(Studyarea)
gArea(Studyarea1)/1e6
```

```
## [1] 7859.314
```

The grid shapefile comprised 185 7 x 7 km cells. Not all of these were sampled, and the sample differed slightly between years.

```
Gridfile <- paste0(GISdir, "/2012DNAGrid3.shp")
Grid <- readOGR(dsn = Gridfile, layer = '2012DNAGrid3')
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "d:/bears/yukon/southern lakes/SL - GIS/2012DNAGrid3.shp", layer: "2012DNAGrid3"
## with 185 features
## It has 5 fields
```

```
Grid$Id <- 1:185
gridcellsused <- vector('list',2)
names(gridcellsused) <- 2012:2013
for (sess in 1:2) {
    spts <-SpatialPoints(as.matrix(siteobj[[sess]]))
    crs(spts) <- crs(Grid)
    gridcellsused[[sess]] <- over(spts, Grid)$Id
    OK <- gridcellsused[[sess]]
    cat ('Hair snares outside any grid cell', sum(is.na(OK)), '\n')
}
```

```
## Hair snares outside any grid cell 1
## Hair snares outside any grid cell 0
```

```
# number per grid cell
lapply(gridcellsused, function (x) table(table(x)))
```

```
## $`2012`
##
##   1   2
## 160   4
##
## $`2013`
##
##   1   2
## 151   8
```

```
OK <- unique(do.call(c,gridcellsused))
trueGrid <- Grid[OK[!is.na(OK)],]
par(mfrow=c(1,2))
for (sess in 1:2) {
    plot(Studyarea1, col='peachpuff')
    plot(Grid[OK[!is.na(OK)],], add = TRUE)
    plot(siteobj[[sess]], add=T)
    mtext(side=3, (2012:2013)[sess])
}
```


```
cat ("Number of grid cells used in either year", length(OK))
```

```
## Number of grid cells used in either year 176
```

Sites checked by helicopter

Sites not checked by helicopter were identified from field maps (see Notes sheet in the snare site spreadsheet) and shaded pink in `snaresheetname`. They total 33 in 2012 and 48 in 2013.

Ecoregions and ecodistricts

```
Ecoregionfile<- paste(GISdir, "ecoregions.shp", sep = "/")
Ecoregions <- readOGR(dsn = Ecoregionfile, layer = 'ecoregions')
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "d:/bears/yukon/southern lakes/SL - GIS/ecoregions.shp", layer: "ecoregions"
## with 218 features
## It has 8 fields
```

```
Ecoregions <- spTransform(Ecoregions, CRS("+init=epsg:3578"))  # Yukon Albers

Ecodistrictfile<- paste(GISdir, "ecodistricts.shp", sep = "/")
Ecodistricts <- readOGR(dsn = Ecodistrictfile, layer = 'ecodistricts')
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "d:/bears/yukon/southern lakes/SL - GIS/ecodistricts.shp", layer: "ecodistricts"
## with 1025 features
## It has 7 fields
```

```
Ecodistricts <- spTransform(Ecodistricts, CRS("+init=epsg:3578"))  # Yukon Albers
```

Towns

Generate Yukon Albers coordinates for Whitehorse and Carcross.

```
towns <- matrix(c(-134.707, 60.169, -135.062, 60.721), nrow=2, byrow=T)
towns <- SpatialPoints(towns)
proj4string(towns) <- CRS("+proj=longlat +ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +no_defs")
towns <- spTransform(towns, CRS('+init=epsg:3578'))
```

Water bodies

```
waterfile <- paste0(GISdir, "/AC_1M_Waterbodies.shp")
water <- readOGR(dsn = waterfile, layer = 'AC_1M_Waterbodies')
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "d:/bears/yukon/southern lakes/SL - GIS/AC_1M_Waterbodies.shp", layer: "AC_1M_Waterbodies"
## with 128576 features
## It has 9 fields
```

```
water <- spTransform(water, CRS("+init=epsg:3578"))   # Yukon Albers
tmpmask <- make.mask(do.call(rbind,siteobj), buffer=50000) # buffer around all hair snares
```

```
## Warning in rbind(deparse.level, ...): inputs differ in detector type; using
## first
```

```
SLwater <- crop(water,extent(x=attr(tmpmask, 'boundingbox')[c(1,3),]))
save(SLwater, file= paste0(workdir, '/SLwater.RData'))
```

Elevation data

First, a utility function to return SpatialPoints with new projection.

```
reproject <- function (xy, proj1 = '+init=epsg:3578', # Yukon Albers
        proj2 = "+proj=longlat +ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +no_defs") {
    # make spatialpoints object
    xy1 <- SpatialPoints(as.matrix(xy))
    # assign initial projection
    proj4string(xy1) <- CRS(proj1)
    # transform and return
    spTransform(xy1, CRS(proj2))
}
```

Read the elevation data from GeoTiff files.

The first GeoTiff file from GeoGratis was renamed from cdem_dem_160809_110634.tif to SLdem1.tif. The second GeoTiff file from GeoGratis was renamed from cdem_dem_170208_150233.tif to SLdem2.tif.

```
dem <- readGDAL (fname= paste0(GISdir, '/SLdem1.tif'))
```

```
## d:/bears/yukon/southern lakes/SL - GIS/SLdem1.tif has GDAL driver GTiff
## and has 2508 rows and 5028 columns
```

```
# a sliver east of Marsh Lake was missed in the window extracted earlier
# add this bit 2017-02-09
demsuppl <- readGDAL (fname = paste0(GISdir,'/SLdem2.tif'))
```

```
## d:/bears/yukon/southern lakes/SL - GIS/SLdem2.tif has GDAL driver GTiff
## and has 7201 rows and 3841 columns
```

Define a function to extract the elevation of each point on a mask and store it as a covariate.

```
getelevation <- function (mask) {
    if (ms(mask)) {
        out <- lapply(mask, getelevation)
        names(out) <- names(mask)
        class (out) <- class(mask)
        out
    }
    else {
        covariates(mask) <- data.frame(
            elevation = over(reproject(mask), dem)$band1,
            elevation2 = over(reproject(mask), demsuppl)$band1    # supplementary DEM
        )
        miss <- is.na(covariates(mask)$elevation)
        covariates(mask)$elevation[miss] <- covariates(mask)$elevation2[miss]
        covariates(mask)$elevation2 <- NULL   # drop this incomplete layer
        mask
    }
}
```

Habitat masks

Create basic mask for each year buffering 50 km around the hair snare sites in 'siteobj'. We exclude cells centred in a lake and attach the elevation using the function defined previously.

```
load(file= paste0(workdir, '/SLwater.RData'))
mask2000 <- make.mask(siteobj, buffer = 50000, spacing = 2000, type = 'trapbuffer',
                      poly = SLwater, poly.habitat = FALSE, keep.poly = FALSE)
```

```
## Warning in FUN(X[[i]], ...): some traps are inside non-habitat polygon
```

```
mask2000 <- getelevation(mask2000)
```

Now develop the dry-path distance matrix (userd) and restricted masks allowing for lakes as barriers (mask2000b). This requires a much finer raster representation of the landscape - we use 250-m x 250-m grid cells. The calculation is SLOW.

```
# 250-m cells (about 500000!)
mask250 <- make.mask(rbind(siteobj), buffer = 50000, spacing = 250, type = 'trapbuffer',
                     poly = SLwater, poly.habitat = F, keep.poly = F)
```

```
## Warning in make.mask(rbind(siteobj), buffer = 50000, spacing = 250, type =
## "trapbuffer", : some traps are inside non-habitat polygon
```

```
userd <- vector('list')
mask2000b <- vector('list')
for (sess in 1:2) {
    userd[[sess]] <- nedist(siteobj[[sess]], mask2000[[sess]], mask250)
    OK <- apply(userd[[sess]], 2, min) < 50000
    mask2000b[[sess]] <- subset(mask2000[[sess]], OK)
    userd[[sess]] <- userd[[sess]][,OK]
    userd[[sess]][!is.finite(userd[[sess]])] <- 1e10   # can't get there from here
}
```

```
## Loading required namespace: gdistance
```

```
class(mask2000b) <- class(mask2000)
```

For later plotting we store a restricted version of the 250-m mask (study area only), and extract the 1250-m contour as many horizontal or vertical line segments using plotMaskEdge updated in secr 3.0.

```
mask250 <- getelevation(mask250)
maskstudy250 <- subset(mask250, pointsInPolygon(mask250, Studyarea1))
median(covariates(maskstudy250)$elevation, na.rm = TRUE)
```

```
## [1] 1254.5
```

```
maskstudyhigh <- subset(maskstudy250, covariates(maskstudy250)$elevation>1250)
c1250 <- plotMaskEdge(maskstudyhigh, plt = FALSE)
```

The contour is more useful if we can use it to plot polygons. The output from contourLines is almost right for this, but in our case one contour screws up (some lines are detached). We fix this manually.

```
tmp <-rectangularMask(maskstudy250)
consave <- contourLines (unique(tmp$x), unique(tmp$y), matrix(covariates(tmp)$elevation,
                         nrow=length(unique(tmp$x))), levels=1250)
# fix contour problem
xy <- consave[c(31,29,28,27)]
xy <- list(x=unlist(sapply(xy,'[[','x')), y=unlist(sapply(xy, '[[','y')))
consave[[25]]$x <- c(consave[[25]]$x, xy$x)
consave[[25]]$y <- c(consave[[25]]$y, xy$y)
```

Next we attach an elevation covariate to the barrier masks. We also include a binary covariate 'highground' that is TRUE for elevation above 1250 m. A few elevations are missing and we assign 'highground' at random for these pixels.

```
mask2000b <- getelevation(mask2000b)   # attach elevations
# fill missing data
for (sess in 1:2) {
    covariates(mask2000b[[sess]])$highground <- covariates(mask2000b[[sess]])$elevation>1250
    elev <- covariates(mask2000b[[sess]])$elevation
    covariates(mask2000b[[sess]])$highground[elev==0] <- (runif(sum(elev==0))>0.5)
}
```

Analyse ecoregion composition of mask and study area.

```
mask250 <- addCovariates(mask250, Ecoregions, 'ECOREGION')
maskstudy250 <- addCovariates(maskstudy250, Ecoregions, 'ECOREGION')
tab250 <- as.numeric(table(covariates(mask250)$ECOREGION))
tabstudy <- c(0,0,table(covariates(maskstudy250)$ECOREGION),0)
sumtab <- data.frame(Ecoregion = c(174, 175, 177, 179, 180, 185),
           REGION_NAM = Ecoregions$REGION_NAM[match( c(174, 175, 177, 179, 180, 185), Ecoregions$ECOREG
           area.mask = round(tab250 * attr(mask250,'area') / 100),
           area.study = round(tabstudy * attr(maskstudy250,'area') / 100))
sumtab$pct.mask <- round(sumtab$area.mask / sum(sumtab$area.mask) * 100,1)
sumtab$pct.study <- round(sumtab$area.study / sum(sumtab$area.study) * 100,1)
sumtab
```

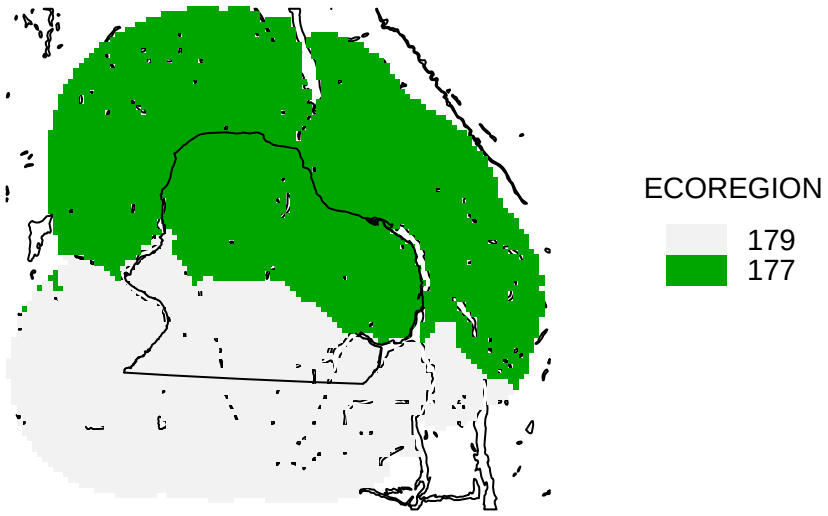
```
##   Ecoregion                    REGION_NAM area.mask area.study pct.mask
## 1       174                   Ruby Ranges      2125          0      7.0
## 2       175         Yukon Plateau-Central       148          0      0.5
## 3       177          Yukon Southern Lakes     15759       4683     51.8
## 4       179       Yukon-Stikine Highlands      9133       2911     30.0
## 5       180 Boreal Mountains and Plateaus      2505         56      8.2
```

```
## 6       185    Northern Coastal Mountains       760          0      2.5
##   pct.study
## 1       0.0
## 2       0.0
## 3      61.2
## 4      38.1
## 5       0.7
## 6       0.0
```

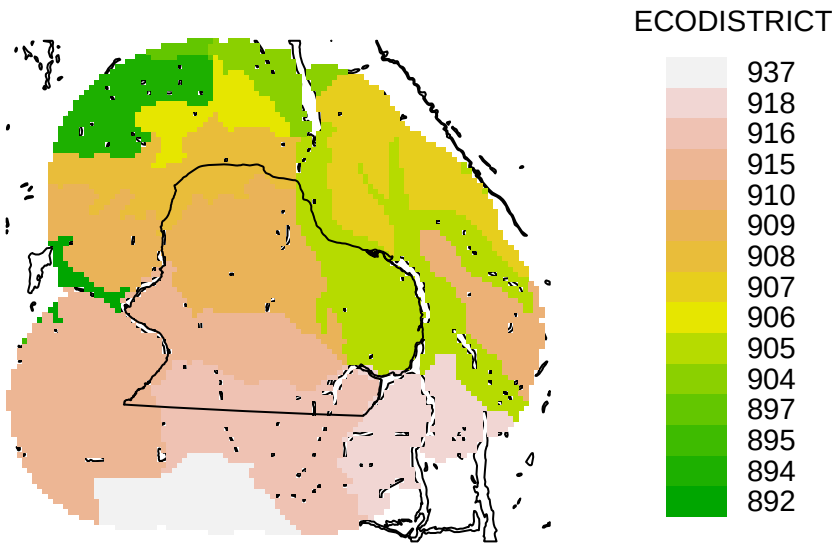
Add ecoregion and ecodistrict as mask covariates.

```
load(file = paste(workdir, 'mask.RData', sep='/'))
# replace covariates if already present
mask2000b <- addCovariates(mask2000b, Ecoregions, 'ECOREGION', replace = TRUE)
mask2000b <- addCovariates(mask2000b, Ecodistricts, 'ECODISTRIC', replace = TRUE)
for (sess in 1:2) {
    ecoreg <- covariates(mask2000b[[sess]])$ECOREGION
    ecoreg[is.na(ecoreg)] <- 179
    covariates(mask2000b[[sess]])$ECOREGION <- as.factor(ecoreg)
    levels(covariates(mask2000b[[sess]])$ECOREGION) <- c('177','177','177','179','179','179')
    ecodis <- covariates(mask2000b[[sess]])$ECODISTRIC
    ecodis[is.na(ecodis)] <- 937 # on basis of longest boundary
    covariates(mask2000b[[sess]])$ECODISTRICT <- as.factor(ecodis)
}
save(mask2000, mask2000b, mask250, maskstudy250, c1250, userd, Studyarea1, trueGrid, consave,
     file = paste0(workdir, '/mask.RData'))
```

```
par(mfrow=c(1,1))
plot(mask2000b[[1]], dots=F, cov='ECOREGION')
plot(Studyarea1, add=T)
```




```
plot(mask2000b[[1]], dots=F, cov='ECODISTRICT')
plot(Studyarea1, add=T)
```



The resulting masks for 2012 and 2013 have `sapply(mask2000b, nrow)` rows.

SECR model fitting

Now we use the session-specific masks and distance matrices to fit a bundle of SECR models. We use a 'hybrid' mixture model in which the covariate specified in the hcov argument (SEX) determines class membership if it is not missing (Efford 2015). Sex was known for all bears, so this is equivalent to treating SEX as a grouping variable. The advantage is that the model is parameterized in terms of total density (D) and sex ratio (pmix), rather than separate male and female densities. The predictor 'h2' in this case refers to known class membership (sex) rather than stochastic class membership as usual in finite mixture models.

Multiple models are fitted in parallel with the function 'par.secr.fit'. The number of processors is set with 'ncores'. We assume up to 8 cores are available, as is the case on a quadcore Windows PC.

Results are saved in RData files. Model fitting chunks are self-contained: they load the required data and masks, and save their out put to RData files. With eval=FALSE, model-fitting chunks are not executed; we load previously fitted results to continue with the summary in this vignette.

```
load(file = paste0(workdir, '/SLCH.RData'))
load(file = paste0(workdir, '/mask.RData'))

# bears without barriers
detailslist1 <- list(distribution = 'binomial')
baseargs1 <- list (capthist = 'SLCH', mask = 'mask2000', trace = FALSE, hcov='SEX',
        details = 'detailslist1', detectfn = 'HEX')
fit2args <- c(baseargs1, list(model = list(lambda0 ~ h2, sigma ~ h2)))
```

```
fit3args <- c(baseargs1, list(model = list(lambda0 ~ h2+bk, sigma ~ h2)))
fit4args <- c(baseargs1, list(model = list(lambda0 ~ h2+session, sigma ~ h2+session)))
fit5args <- c(baseargs1, list(model = list(lambda0 ~ h2+bk+session, sigma ~ h2+session)))

# bears with barriers
detailslist2 <- list(distribution = 'binomial', userdist = userd)
baseargs2 <- list (capthist = 'SLCH', mask = 'mask2000b', trace = FALSE, hcov='SEX',
        details = 'detailslist2', detectfn = 'HEX')
fit6args <- c(baseargs2, list(model = list(lambda0 ~ h2, sigma ~ h2)))
fit7args <- c(baseargs2, list(model = list(lambda0 ~ h2+bk, sigma ~ h2)))
fit8args <- c(baseargs2, list(model = list(lambda0 ~ h2+session, sigma ~ h2+session)))
fit9args <- c(baseargs2, list(model = list(lambda0 ~ h2+bk+session, sigma ~ h2+session)))

fits2 <- par.secr.fit(list(fit2 = fit2args, fit3 = fit3args, fit4 = fit4args,
                           fit5 = fit5args, fit6 = fit6args, fit7 = fit7args,
                           fit8 = fit8args, fit9 = fit9args), ncores = 8)

save(fits2, file = paste0(workdir, '/fits2.RData'))
```

```
load(file = paste0(workdir, '/fits2.RData'))
AIC(fits2[1:4])
```

```
##                                                          model
## fit.fit5 D~1 lambda0~h2 + bk + session sigma~h2 + session pmix~h2
## fit.fit3                      D~1 lambda0~h2 + bk sigma~h2 pmix~h2
## fit.fit2                           D~1 lambda0~h2 sigma~h2 pmix~h2
## fit.fit4      D~1 lambda0~h2 + session sigma~h2 + session pmix~h2
##                     detectfn npar    logLik      AIC     AICc  dAICc AICcwt
## fit.fit5 hazard exponential    9 -1345.059 2708.118 2709.503  0.000 0.6096
## fit.fit3 hazard exponential    7 -1347.773 2709.546 2710.394  0.891 0.3904
## fit.fit2 hazard exponential    6 -1381.787 2775.573 2776.205 66.702 0.0000
## fit.fit4 hazard exponential    8 -1379.581 2775.162 2776.261 66.758 0.0000
```

```
AIC(fits2[5:8])
```

```
##                                                          model
## fit.fit9 D~1 lambda0~h2 + bk + session sigma~h2 + session pmix~h2
## fit.fit7                      D~1 lambda0~h2 + bk sigma~h2 pmix~h2
## fit.fit8      D~1 lambda0~h2 + session sigma~h2 + session pmix~h2
## fit.fit6                           D~1 lambda0~h2 sigma~h2 pmix~h2
##                     detectfn npar    logLik      AIC     AICc  dAICc AICcwt
## fit.fit9 hazard exponential    9 -1767.191 3552.383 3553.767  0.000 0.7509
## fit.fit7 hazard exponential    7 -1770.563 3555.125 3555.974  2.207 0.2491
## fit.fit8 hazard exponential    8 -1798.477 3612.955 3614.054 60.287 0.0000
## fit.fit6 hazard exponential    6 -1801.570 3615.140 3615.772 62.005 0.0000
```

```
extract <- function (fit, parm='D', scale=1e5) {
    pred <- predict(fit)
    est <- pred[[1]][parm,'estimate']
    SE.est <- pred[[1]][parm,'SE.estimate']
    lcl <- pred[[1]][parm,'lcl']
    ucl <- pred[[1]][parm,'ucl']
    c (est=est*scale, SE.est=SE.est*scale, lcl=lcl*scale, ucl=ucl*scale, RSE=SE.est/est)
}
t(sapply(fits2, extract))
```

```
##                est    SE.est      lcl      ucl        RSE
## fit.fit2  8.845484 0.7178493 7.546686 10.36781 0.08115432
## fit.fit3  9.782517 0.8948805 8.179881 11.69915 0.09147752
## fit.fit4  8.941114 0.7324875 7.616840 10.49563 0.08192351
## fit.fit5  9.899779 0.9134171 8.265223 11.85759 0.09226641
## fit.fit6  9.530268 0.7637431 8.147044 11.14834 0.08013868
## fit.fit7 10.576485 0.9489259 8.874099 12.60545 0.08972035
## fit.fit8  9.648558 0.7817070 8.234031 11.30609 0.08101801
## fit.fit9 10.711142 0.9692625 8.973599 12.78512 0.09049105
```

```
t(sapply(fits2, extract, parm='pmix', scale=1))
```

```
##                est     SE.est       lcl       ucl        RSE
## fit.fit2 0.6193165 0.04281931 0.5326236 0.6990159 0.06913962
## fit.fit3 0.6407192 0.04432104 0.5501129 0.7222896 0.06917388
## fit.fit4 0.6190798 0.04292410 0.5321781 0.6989700 0.06933533
## fit.fit5 0.6402429 0.04444584 0.5493922 0.7220437 0.06942028
## fit.fit6 0.6067069 0.04301408 0.5200329 0.6871453 0.07089763
## fit.fit7 0.6244191 0.04452707 0.5340024 0.7069209 0.07130958
## fit.fit8 0.6063838 0.04313699 0.5194684 0.6870493 0.07113810
## fit.fit9 0.6235228 0.04467683 0.5328281 0.7063151 0.07165228
```

Effect of missing usage data

So far we have assumed that all detectors were visited on each occasion. What was the effect of not visiting some detectors in July 2012? If stations were visited and hair samples were lost then that station was effectively 'unused'. If hair samples were not collected but rather accumulated until the final check in early August then the August sample...

```
load(file = paste0(workdir, '/SLCH.RData'))
load(file = paste0(workdir, '/mask.RData'))
SLCH1a <- SLCH1b <- SLCH[[1]]
usage(traps(SLCH1a)) <- matrix(1, nrow = nrow(traps(SLCH1a)), ncol = 4)
usage(traps(SLCH1b)) <- matrix(1, nrow = nrow(traps(SLCH1b)), ncol = 4)
usage(traps(SLCH1b))[, 3:4] <- !is.na(sitedf[sitedf$Year==2012,c(11,14)])
detailslist <- list(distribution = 'binomial', userdist = userd[[1]])
mask <- mask2000b[[1]]
baseargs <- list (mask = 'mask', trace = FALSE, hcov='SEX',
                  model = list(D ~ 1, lambda0 ~ h2+bk, sigma ~ h2),
                  details = 'detailslist', detectfn = 'HEX')
fit1a.args <- c(baseargs, list(capthist = 'SLCH1a'))
fit1b.args <- c(baseargs, list(capthist = 'SLCH1b'))
fits1ab <- par.secr.fit(list(fit1a = fit1a.args, fit1b = fit1b.args), ncores = 2)
save(fits1ab, file = paste0(workdir, '/fits1ab.RData'))
```

```
load(file = paste0(workdir, '/fits1ab.RData'))
collate(fits1ab)[1,,,1] * 1e5
```

```
##           estimate SE.estimate      lcl      ucl
## fit.fit1a 10.75474    1.238187 8.588627 13.46715
## fit.fit1b 10.94346    1.267119 8.728186 13.72099
```

Telemetry

Data preparation

Input telemetry data from consolidated spreadsheet of 19 Sept 2016.

```
AllCollarsname <- paste0(datadir, "/SL_ALL_COLLARS_for JBME_updated19Sept2016.xls")
AllCollars <- read.xls(AllCollarsname, sheet = 1, header = TRUE, perl = perlexe,
       stringsAsFactors = FALSE, na.strings = c("NA", "#DIV/0!", "#VALUE!"))
names(AllCollars)[1] <- 'id'
names(AllCollars)[2] <- 'age.sex'
AllCollars$date <- as.Date(AllCollars$GPS_Fix_Time, "%Y.%m.%d")  # first 10 ch
dx <- as.Date(AllCollars$GPS_Fix_Time, "%Y %b %d") # first 11 ch
AllCollars$date <- ifelse(is.na(AllCollars$date), dx, AllCollars$date)
class(AllCollars$date) <- "Date" # restore lost class
AllCollars$time <- chron(times. = substring(AllCollars$GPS_Fix_Time,12,19))
dt <- chron(times. = paste0( sprintf('%02.0f',AllCollars$Hour), ':00:00'))
AllCollars$time <- ifelse(is.na(AllCollars$time), dt, AllCollars$time)
class(AllCollars$time) <- 'times'
AllCollars$year <- as.numeric(substring(AllCollars$GPS_Fix_Time,1,4))
AllCollars$mon <- as.POSIXlt(AllCollars$date)$mon+1
```

Next we append columns with projected x and y coordinates. We also add a column 'inSA' that is TRUE only if fix is within the nominal study area.

```
xy <- SpatialPoints(cbind(AllCollars$GPS_Longitude, AllCollars$GPS_Latitude),
                        proj4string= CRS("+init=epsg:4326") )
xy <- spTransform(xy, CRS("+init=epsg:3578"))   # Yukon Albers
xy <- coordinates(xy)
AllCollars <- cbind(AllCollars, x=xy[,1], y=xy[,2])
AllCollars$inSA <- pointsInPolygon(AllCollars[,c('x','y')], Studyarea1)
```

Acquire collar summary table and save all data.

```
AC.split.locs <- split(AllCollars, list(AllCollars$id, AllCollars$year))
collarsummaryname <- paste0(datadir, "/SL_Collar_Summary_Sept2016_JG.xls")
collarsummary <- read.xls(collarsummaryname, sheet = 2, header = TRUE, perl = perlexe,
       stringsAsFactors = FALSE, na.strings = c("NA", "#DIV/0!", "#VALUE!"))
collarsummary$Date <- as.Date(collarsummary$Date_Collared, format="%d-%b-%y")
Albersxy <- function(xy) {
    xy <- matrix(as.numeric(unlist(xy)), ncol = 2)
    OK <- apply(xy,1,function (x) !any(is.na(x)))
    xysp <- SpatialPoints(xy[OK,2:1], proj4string= CRS("+init=epsg:4326") )
    xytr <- spTransform(xysp, CRS("+init=epsg:3578"))   # Yukon Albers
    xy[OK,] <- coordinates(xytr)
    colnames(xy) <- c('x','y')
    xy
}
capturexy <- Albersxy(collarsummary[,12:13])
```

```
## Warning in matrix(as.numeric(unlist(xy)), ncol = 2): NAs introduced by
## coercion
```

```
releasexy <- Albersxy(collarsummary[,14:15])
```

```
## Warning in matrix(as.numeric(unlist(xy)), ncol = 2): NAs introduced by
```

```
## coercion
```

```
collarsummary <- cbind(collarsummary,
                       capture.x = capturexy[,'x'], capture.y = capturexy[,'y'],
                       release.x = releasexy[,'x'], release.y = releasexy[,'y'])

save(AC.split.locs, AllCollars, collarsummary, file = paste0(workdir, '/AllCollars.Rdata'))
```

Now we are ready to build capthist objects that include the telemetry data for selected bears in 2012 and 2013.

```
load(file = paste0(workdir, '/AllCollars.Rdata'))
load(file= paste0(workdir, '/SLwater.RData'))
SLWdf <- as.data.frame(SLwater)
# select telemetry fixes concurrent with DNA hair snares,
# drop two bears telemetered for only 2-3 days in interval,
# and limit to one fix per day
 # drop bear SL028 with only 3d & relocated SL031,SL032
OK <- AllCollars$year %in% c(2012,2013) &        # DNA hair snare years
      AllCollars$mon %in% 6:8 &                  # June-Aug
      !(AllCollars$id %in% c('SL028')) &
      !duplicated(AllCollars[,c('id', 'date')])  # one per day

# shape fixes into dataframe for input as capthist
AllCollars$sex <- substring(AllCollars$age.sex,2,2)
cols <- c('year','id','mon','x','y','sex')
df <- AllCollars[OK, cols]
df$mon <- df$mon-5   # as 'occasion' numbered from 1

# build capthist of telemetry data
CHall <- read.telemetry(data = df, covnames = 'SEX')   # standalone, with SL013, SL032
```

```
## No errors found :-)
```

```
df2 <- df[!(df$id %in% c('SL031', 'SL032')),]
CH <- read.telemetry(data = df2, covnames = 'SEX')   # standalone
```

```
## No errors found :-)
```

```
ch <- addTelemetry(SLCH, CH, type='concurrent')     # combined telemetry, hair snares
```

```
## No errors found :-)
```

```
summary(ch)
```

```
## $`2012`
## Object class      capthist
## Detector type     proximity (4), telemetry
## Telemetry type    concurrent
## Detector number   169
## Average spacing   4828.281 m
## x-range           291307.8 401088.2 m
## y-range           615166.9 709697.5 m
##
##  Usage range by occasion
##     1 2 3 4 5
## min 0 0 0 0 0
## max 1 1 1 1 1
```

```
##
## Counts by occasion
##                      1   2   3   4   5 Total
## n                   24  31  44  41   8   148
## u                   24  23  20   8   4    79
## f                   32  27  18   2   0    79
## M(t+1)              24  47  67  75  79    79
## losses               0   0   0   0   0     0
## detections          29  42  61  56 649   837
## detectors visited   21  27  39  42   0   129
## detectors used     169 169 169 169   0   676
##
## Empty histories :  4
## 8 telemetered animals, 4 detected
## 57-92 locations per animal, mean =  81.12, sd = 15.6
##
## $`2013`
## Object class       capthist
## Detector type      proximity (4), telemetry
## Telemetry type     concurrent
## Detector number    167
## Average spacing    4830.718 m
## x-range            291660.1 401466.2 m
## y-range            614636 708866.6 m
##
##  Usage range by occasion
##     1 2 3 4 5
## min 0 0 0 0 0
## max 1 1 1 1 1
##
## Counts by occasion
##                      1   2   3   4   5 Total
## n                   20  33  30  22   5   110
## u                   20  24  13   8   3    68
## f                   35  26   5   2   0    68
## M(t+1)              20  44  57  65  68    68
## losses               0   0   0   0   0     0
## detections          25  40  37  27 214   343
## detectors visited   18  30  27  20   0    95
## detectors used     167 167 167 167   0   668
##
## Empty histories :  3
## 5 telemetered animals, 2 detected
## 22-92 locations per animal, mean =  42.8, sd = 28.93
```

```
save(ch, CH, CHall, file = paste0(workdir, '/ch.RData'))
```

Telemetry model fitting

We fit telemetry models using Euclidean distances (i.e. without specifying userdist = userd).

```
load(file = paste0(workdir, '/ch.RData'))
baseargs <- list (mask = 'mask2000b', trace = FALSE, hcov = 'SEX',
                  model = list(D ~ 1, lambda0 ~ h2+bk, sigma ~ h2),
```

```
                  details = 'detailslist', detectfn = 'HEX', start = fitsTh2CL)
detailslist <- list(distribution = 'binomial', telemetryscale=1e9)
fitsTh2 <- par.secr.fit(list(fit0h2 = c(baseargs, list(capthist = 'SLCH')), # hair only
                        fitTh2 = c(baseargs, list(capthist = 'ch'))),  # hair + telemetry
                        ncores = 2)
# Save fitted models
save(fitsTh2, file = paste0(workdir, '/fitsT.RData'))
```

```
load(file = paste0(workdir, '/fitsT.RData'))

# Compare relative SE
collate(fitsTh2)[1,,2,1] /  collate(fitsTh2)[1,,1,1]
```

```
## fit.fit0h2 fit.fitTh2
## 0.09077973 0.09357798
```

```
# Compare relative length of 95% CI
apply(collate(fitsTh2)[1,,3:4,1],1,diff) / collate(fitsTh2)[1,,1,1]
```

```
## fit.fit0h2 fit.fitTh2
##  0.3569891  0.3680663
```

```
# compare estimates
predict(fitsTh2, newdata=data.frame(h2=c('F','M'), bk=0))
```

```
## $fit.fit0h2
## $fit.fit0h2$`h2 = F, bk = 0`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 9.819924e-05 8.914501e-06 8.222328e-05 1.172793e-04
## lambda0   log 1.832577e-01 3.724971e-02 1.235354e-01 2.718521e-01
## sigma     log 3.717469e+03 3.116190e+02 3.155148e+03 4.380008e+03
## pmix    logit 6.406583e-01 4.431346e-02 5.500703e-01 7.222180e-01
##
## $fit.fit0h2$`h2 = M, bk = 0`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 9.819924e-05 8.914501e-06 8.222328e-05 1.172793e-04
## lambda0   log 1.216671e-01 2.305171e-02 8.420146e-02 1.758031e-01
## sigma     log 6.007779e+03 5.243541e+02 5.064804e+03 7.126318e+03
## pmix    logit 3.593417e-01 4.431346e-02 2.777820e-01 4.499297e-01
##
##
## $fit.fitTh2
## $fit.fitTh2$`h2 = F, bk = 0`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 1.010071e-04 9.452039e-06 8.411466e-05 1.212920e-04
## lambda0   log 2.501493e-01 3.979668e-02 1.834955e-01 3.410147e-01
## sigma     log 3.060918e+03 1.461773e+02 2.787564e+03 3.361078e+03
## pmix    logit 6.657781e-01 4.235880e-02 5.783612e-01 7.431208e-01
##
## $fit.fitTh2$`h2 = M, bk = 0`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 1.010071e-04 9.452039e-06 8.411466e-05 1.212920e-04
## lambda0   log 9.240733e-02 1.177454e-02 7.205809e-02 1.185032e-01
## sigma     log 6.990835e+03 2.522938e+02 6.513581e+03 7.503057e+03
## pmix    logit 3.342219e-01 4.235880e-02 2.568792e-01 4.216388e-01
```

Density models

Here we are looking at elevation, ecoregion and year effects.

```
load(file = paste0(workdir, '/SLCH.RData'))
load(file = paste0(workdir, '/mask.RData'))
detailslist <- list(distribution = 'binomial', userdist = userd)
baseargs <- list (capthist = 'SLCH', mask = 'mask2000b', trace = FALSE, hcov='SEX',
        details = 'detailslist', detectfn = 'HEX')
detailslist <- list(distribution = 'binomial', userdist = userd)
fit9args  <- c(baseargs, list(model = list(D ~ 1, lambda0 ~ h2+bk, sigma ~ h2)))
fit11args <- c(baseargs, list(model = list(D ~ highground, lambda0 ~ h2 + bk, sigma ~ h2)))
fit12args <- c(baseargs, list(model = list(D ~ session + highground,
                                           lambda0 ~ h2 + bk, sigma ~ h2)))
fit13args <- c(baseargs, list(model = list(D ~ session, lambda0 ~ h2 + bk, sigma ~ h2)))
fit27args <- c(baseargs, list(model = list(D ~ ECOREGION, lambda0 ~ h2 + bk, sigma ~ h2)))
fit28args <- c(baseargs, list(model = list(D ~ ECOREGION+highground, lambda0 ~ h2 + bk, sigma ~ h2)))
fit29args <- c(baseargs, list(model = list(D ~ ECOREGION*highground, lambda0 ~ h2 + bk, sigma ~ h2)))

densityfits <- par.secr.fit(list(fit9=fit9args, fit11 = fit11args, fit12 = fit12args,
                                 fit13=fit13args, fit27 = fit27args, fit28 = fit28args,
                                 fit29 = fit29args), ncores = 7)
save(densityfits, file=paste0(workdir,"/densityfits.Rdata"))
```

Now consider regression spline using elevation as a continuous predictor.

```
load(file = paste0(workdir, '/SLCH.RData'))
load(file = paste0(workdir, '/mask.RData'))
detailslist <- list(distribution = 'binomial', userdist = userd)
baseargs <- list (capthist = 'SLCH', mask = 'mask2000b', trace = FALSE, hcov='SEX',
                  details = 'detailslist', detectfn = 'HEX')
detailslist <- list(distribution = 'binomial', userdist = userd, debug=0)
fit11args <- c(baseargs, list(model = list(D ~ highground, lambda0 ~ h2 + bk, sigma ~ h2)))
fit23args  <- c(baseargs, list(model = list(D ~ s(elevation, k=3, fx=TRUE),
                                            lambda0 ~ h2+bk, sigma ~ h2)))
fit24args  <- c(baseargs, list(model = list(D ~ s(elevation, k=5, fx=TRUE),
                                            lambda0 ~ h2+bk, sigma ~ h2)))
densityfits2 <- par.secr.fit(list(fit11 = fit11args, fit23 = fit23args, fit24=fit24args),
                             ncores = 3)
save(densityfits2, file=paste0(workdir,"/densityfits2.Rdata"))
```

```
load(file = paste0(workdir,"/densityfits.Rdata"))
load(file = paste0(workdir,"/densityfits2.Rdata"))
AIC(densityfits[c(1,2,5,6,7)])
```

```
##                                                                  model
## fit.fit28 D~ECOREGION + highground lambda0~h2 + bk sigma~h2 pmix~h2
## fit.fit29 D~ECOREGION * highground lambda0~h2 + bk sigma~h2 pmix~h2
## fit.fit11             D~highground lambda0~h2 + bk sigma~h2 pmix~h2
## fit.fit9                      D~1 lambda0~h2 + bk sigma~h2 pmix~h2
## fit.fit27              D~ECOREGION lambda0~h2 + bk sigma~h2 pmix~h2
##                      detectfn npar    logLik      AIC     AICc  dAICc
## fit.fit28 hazard exponential    9 -1759.110 3536.220 3537.605  0.000
## fit.fit29 hazard exponential   10 -1758.966 3537.933 3539.638  2.033
## fit.fit11 hazard exponential    8 -1761.843 3539.687 3540.786  3.181
```

```
## fit.fit9  hazard exponential    7 -1770.563 3555.125 3555.974 18.369
## fit.fit27 hazard exponential    8 -1770.474 3556.948 3558.047 20.442
##           AICcwt
## fit.fit28 0.6387
## fit.fit29 0.2311
## fit.fit11 0.1302
## fit.fit9  0.0000
## fit.fit27 0.0000
```

```
AIC(c(densityfits2, densityfits[1])) # include constant model in set
```

```
##                                                                      model
## fit.fit23 D~s(elevation, k = 3, fx = TRUE) lambda0~h2 + bk sigma~h2 pmix~h2
## fit.fit24 D~s(elevation, k = 5, fx = TRUE) lambda0~h2 + bk sigma~h2 pmix~h2
## fit.fit11                     D~highground lambda0~h2 + bk sigma~h2 pmix~h2
## fit.fit9                             D~1 lambda0~h2 + bk sigma~h2 pmix~h2
##                      detectfn npar    logLik      AIC     AICc  dAICc
## fit.fit23 hazard exponential    9 -1758.142 3534.285 3535.669  0.000
## fit.fit24 hazard exponential   11 -1757.914 3537.828 3539.891  4.222
## fit.fit11 hazard exponential    8 -1761.843 3539.687 3540.786  5.117
## fit.fit9  hazard exponential    7 -1770.563 3555.125 3555.974 20.305
##           AICcwt
## fit.fit23 0.8344
## fit.fit24 0.1011
## fit.fit11 0.0646
## fit.fit9  0.0000
```

```
predict(densityfits$fit.fit11)
```

```
## $`session = 2012, h2 = F, bk = 0, highground = FALSE`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 3.395104e-05 1.610882e-05 1.403784e-05 8.211187e-05
## lambda0   log 1.992343e-01 3.998698e-02 1.349612e-01 2.941163e-01
## sigma     log 3.696604e+03 3.011468e+02 3.151909e+03 4.335429e+03
## pmix    logit 6.191265e-01 4.431176e-02 5.293507e-01 7.014363e-01
##
## $`session = 2013, h2 = F, bk = 0, highground = FALSE`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 3.395104e-05 1.610882e-05 1.403784e-05 8.211187e-05
## lambda0   log 1.992343e-01 3.998698e-02 1.349612e-01 2.941163e-01
## sigma     log 3.696604e+03 3.011468e+02 3.151909e+03 4.335429e+03
## pmix    logit 6.191265e-01 4.431176e-02 5.293507e-01 7.014363e-01
##
## $`session = 2012, h2 = M, bk = 0, highground = FALSE`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 3.395104e-05 1.610882e-05 1.403784e-05 8.211187e-05
## lambda0   log 1.403906e-01 2.724254e-02 9.631369e-02 2.046389e-01
## sigma     log 5.550558e+03 4.772499e+02 4.691191e+03 6.567351e+03
## pmix    logit 3.808735e-01 4.431176e-02 2.985637e-01 4.706493e-01
##
## $`session = 2013, h2 = M, bk = 0, highground = FALSE`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 3.395104e-05 1.610882e-05 1.403784e-05 8.211187e-05
## lambda0   log 1.403906e-01 2.724254e-02 9.631369e-02 2.046389e-01
## sigma     log 5.550558e+03 4.772499e+02 4.691191e+03 6.567351e+03
```

```
## pmix    logit 3.808735e-01 4.431176e-02 2.985637e-01 4.706493e-01
```

```
# oops - there is a software glitch in the reporting of pmix; not to worry
predict(densityfits$fit.fit11, newdata=expand.grid(h2=c('F','M'), bk=0:1, highground=c(F,T)))
```

```
## $`h2 = F, bk = 0, highground = FALSE`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 3.395104e-05 1.610882e-05 1.403784e-05 8.211187e-05
## lambda0   log 1.992343e-01 3.998698e-02 1.349612e-01 2.941163e-01
## sigma     log 3.696604e+03 3.011468e+02 3.151909e+03 4.335429e+03
## pmix    logit 5.000000e-01 4.697847e-02 4.089508e-01 5.910492e-01
##
## $`h2 = M, bk = 0, highground = FALSE`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 3.395104e-05 1.610882e-05 1.403784e-05 8.211187e-05
## lambda0   log 1.403906e-01 2.724254e-02 9.631369e-02 2.046389e-01
## sigma     log 5.550558e+03 4.772499e+02 4.691191e+03 6.567351e+03
## pmix    logit 6.191265e-01 4.431176e-02 5.293507e-01 7.014363e-01
##
## $`h2 = F, bk = 1, highground = FALSE`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 3.395104e-05 1.610882e-05 1.403784e-05 8.211187e-05
## lambda0   log 9.378344e-01 1.752079e-01 6.523267e-01 1.348302e+00
## sigma     log 3.696604e+03 3.011468e+02 3.151909e+03 4.335429e+03
## pmix    logit 5.000000e-01 4.697847e-02 4.089508e-01 5.910492e-01
##
## $`h2 = M, bk = 1, highground = FALSE`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 3.395104e-05 1.610882e-05 1.403784e-05 8.211187e-05
## lambda0   log 6.608460e-01 1.272261e-01 4.546902e-01 9.604725e-01
## sigma     log 5.550558e+03 4.772499e+02 4.691191e+03 6.567351e+03
## pmix    logit 6.191265e-01 4.431176e-02 5.293507e-01 7.014363e-01
##
## $`h2 = F, bk = 0, highground = TRUE`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 1.790592e-04 2.195963e-05 1.409274e-04 2.275085e-04
## lambda0   log 1.992343e-01 3.998698e-02 1.349612e-01 2.941163e-01
## sigma     log 3.696604e+03 3.011468e+02 3.151909e+03 4.335429e+03
## pmix    logit 5.000000e-01 4.697847e-02 4.089508e-01 5.910492e-01
##
## $`h2 = M, bk = 0, highground = TRUE`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 1.790592e-04 2.195963e-05 1.409274e-04 2.275085e-04
## lambda0   log 1.403906e-01 2.724254e-02 9.631369e-02 2.046389e-01
## sigma     log 5.550558e+03 4.772499e+02 4.691191e+03 6.567351e+03
## pmix    logit 3.808735e-01 4.431176e-02 2.985637e-01 4.706493e-01
##
## $`h2 = F, bk = 1, highground = TRUE`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 1.790592e-04 2.195963e-05 1.409274e-04 2.275085e-04
## lambda0   log 9.378344e-01 1.752079e-01 6.523267e-01 1.348302e+00
## sigma     log 3.696604e+03 3.011468e+02 3.151909e+03 4.335429e+03
## pmix    logit 5.000000e-01 4.697847e-02 4.089508e-01 5.910492e-01
##
## $`h2 = M, bk = 1, highground = TRUE`
```

```
##          link     estimate  SE.estimate          lcl          ucl
## D         log 1.790592e-04 2.195963e-05 1.409274e-04 2.275085e-04
## lambda0   log 6.608460e-01 1.272261e-01 4.546902e-01 9.604725e-01
## sigma     log 5.550558e+03 4.772499e+02 4.691191e+03 6.567351e+03
## pmix    logit 3.808735e-01 4.431176e-02 2.985637e-01 4.706493e-01
```

```
# highground
LR.test(densityfits$fit.fit11, densityfits$fit.fit9)
```

```
##
##  Likelihood ratio test for two models
##
## data:  densityfits$fit.fit11 vs densityfits$fit.fit9
## X-square = 17.438, df = 1, p-value = 2.968e-05
```

```
# session
LR.test(densityfits$fit.fit13, densityfits$fit.fit9)
```

```
##
##  Likelihood ratio test for two models
##
## data:  densityfits$fit.fit13 vs densityfits$fit.fit9
## X-square = 1.0824, df = 1, p-value = 0.2982
```

```
predict(densityfits$fit.fit13)
```

```
## $`session = 2012, h2 = F, bk = 0`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 1.138742e-04 1.294964e-05 9.118821e-05 1.422042e-04
## lambda0   log 1.864223e-01 3.764031e-02 1.259930e-01 2.758347e-01
## sigma     log 3.791640e+03 3.156259e+02 3.221760e+03 4.462324e+03
## pmix    logit 6.242987e-01 4.451932e-02 5.339025e-01 7.067912e-01
##
## $`session = 2013, h2 = F, bk = 0`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 9.754116e-05 1.172821e-05 7.712723e-05 1.233582e-04
## lambda0   log 1.864223e-01 3.764031e-02 1.259930e-01 2.758347e-01
## sigma     log 3.791640e+03 3.156259e+02 3.221760e+03 4.462324e+03
## pmix    logit 6.242987e-01 4.451932e-02 5.339025e-01 7.067912e-01
##
## $`session = 2012, h2 = M, bk = 0`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 1.138742e-04 1.294964e-05 9.118821e-05 1.422042e-04
## lambda0   log 1.359483e-01 2.623222e-02 9.346029e-02 1.977519e-01
## sigma     log 5.656509e+03 4.873073e+02 4.779177e+03 6.694896e+03
## pmix    logit 3.757013e-01 4.451932e-02 2.932088e-01 4.660975e-01
##
## $`session = 2013, h2 = M, bk = 0`
##          link     estimate  SE.estimate          lcl          ucl
## D         log 9.754116e-05 1.172821e-05 7.712723e-05 1.233582e-04
## lambda0   log 1.359483e-01 2.623222e-02 9.346029e-02 1.977519e-01
## sigma     log 5.656509e+03 4.873073e+02 4.779177e+03 6.694896e+03
## pmix    logit 3.757013e-01 4.451932e-02 2.932088e-01 4.660975e-01
```

```
# ecoregion
LR.test(densityfits$fit.fit28, densityfits$fit.fit11)
```

```
##
##  Likelihood ratio test for two models
##
## data:  densityfits$fit.fit28 vs densityfits$fit.fit11
## X-square = 5.4667, df = 1, p-value = 0.01938
```

Heterogeneity check and final model fits

This code addresses the question of whether there is significant unmodelled heterogeneity, over and above sex differences. For this it is necessary to use the 'group' (g) construct in secr.fit.

The model fit22args is the final model and the basis for Table 6.

```
load(file = paste0(workdir, '/SLCH.RData'))
load(file = paste0(workdir, '/mask.RData'))
detailslist <- list(distribution = 'binomial', userdist = userd)
baseargs <- list (capthist = 'SLCH', mask = 'mask2000b', trace = FALSE,
        details = 'detailslist', detectfn = 'HEX', groups = 'SEX')
fit14args  <- c(baseargs, list(model = list(D ~ g, lambda0 ~ bk, sigma ~ 1)))
fit15args  <- c(baseargs, list(model = list(D ~ g, lambda0 ~ g+bk, sigma ~ g)))
fit16args  <- c(baseargs, list(model = list(D ~ g, lambda0 ~ g+bk+h2, sigma ~ g+h2)))
fit19args  <- c(baseargs, list(model = list(D ~ g+highground,
                                            lambda0 ~ g+bk+session, sigma ~ g+session)))
fit21args  <- c(baseargs, list(model = list(D ~ g+highground,
                                            lambda0 ~ g+bk+h2, sigma ~ g+h2)))
fit22args  <- c(baseargs, list(model = list(D ~ g+highground,
                                            lambda0 ~ g+bk+h2+session, sigma ~ g+h2+session)))
groupfits <- par.secr.fit(list(fit14=fit14args, fit15=fit15args, fit16=fit16args,
                               fit19=fit19args, fit21=fit21args, fit22=fit22args),
                          ncores = 6)
save(groupfits, file=paste0(workdir, "/groupfits.Rdata"))
```

```
load(file = paste0(workdir,"/groupfits.Rdata"))
AIC(groupfits)
```

```
##                                                                                  model
## fit.fit22 D~g + highground lambda0~g + bk + h2 + session sigma~g + h2 + session pmix~h2
## fit.fit21                     D~g + highground lambda0~g + bk + h2 sigma~g + h2 pmix~h2
## fit.fit19                   D~g + highground lambda0~g + bk + session sigma~g + session
## fit.fit20                         D~g + highground a0~bk sigma~g + h2 + session pmix~h2
## fit.fit16                                 D~g lambda0~g + bk + h2 sigma~g + h2 pmix~h2
## fit.fit15                                                    D~g lambda0~g + bk sigma~g
## fit.fit14                                                       D~g lambda0~bk sigma~1
##                      detectfn npar    logLik      AIC     AICc  dAICc
## fit.fit22 hazard exponential   13 -1756.879 3539.758 3542.647  0.000
## fit.fit21 hazard exponential   11 -1759.689 3541.377 3543.440  0.793
## fit.fit19 hazard exponential   10 -1761.673 3543.346 3545.051  2.404
## fit.fit20 hazard exponential   10 -1762.349 3544.697 3546.403  3.756
## fit.fit16 hazard exponential   10 -1763.273 3546.547 3548.252  5.605
## fit.fit15 hazard exponential    7 -1768.986 3551.973 3552.821 10.174
## fit.fit14 hazard exponential    5 -1777.740 3565.480 3565.928 23.281
##           AICcwt
## fit.fit22 0.4573
## fit.fit21 0.3076
```

```
## fit.fit19 0.1375
## fit.fit20 0.0699
## fit.fit16 0.0277
## fit.fit15 0.0000
## fit.fit14 0.0000
```

```
extractD <- function (fit) {
    pred <- predict(fit)
    est <- pred[[1]]['D','estimate'] + pred[[3]]['D','estimate']
    SE.est <- (pred[[1]]['D','SE.estimate']^2 + pred[[3]]['D','SE.estimate']^2)^0.5
    tmp <- data.frame (estimate=est*1e5, SE.estimate=SE.est*1e5)
    tmp <- secr3:::add.cl(tmp, 0.05, TRUE)
    tmp$RSE <- SE.est/est
    tmp
}
t(sapply(groupfits, extractD))
```

```
##           estimate SE.estimate lcl      ucl      RSE
## fit.fit14 10.3731  0.8413575   8.850771 12.15727 0.08110956
## fit.fit15 10.55851 0.9417286   8.868153 12.57107 0.08919141
## fit.fit16 11.9862  1.399351    9.542071 15.05638 0.1167468
## fit.fit19 16.84049 2.139336    13.14179 21.58017 0.1270353
## fit.fit20 23.72163 12.67615    8.882463 63.35131 0.5343708
## fit.fit21 18.24841 2.666449    13.72474 24.26308 0.1461195
## fit.fit22 18.49095 2.68132     13.93704 24.53285 0.1450071
```

```
load(file=paste0(workdir,"/groupfits.Rdata"))

# elaborate code to average detection parameters across mixture classes for Table 6
betabar <- function(sex = 'F', bk = 0, session = 1, thetaname = 'lambda0',
                    object = groupfits[[7]]){
    newdata <- expand.grid(g=factor(sex, levels=c('F','M')), bk = bk, h2 = factor(1:2),
                           session = session)
    bmat <- secr3:::secr.lpredictor (formula = object$model[[thetaname]],
                newdata = newdata, indx = object$parindx[[thetaname]],
                beta = object$fit$par, field = thetaname, beta.vcv = object$beta.vcv)
    pmix <- secr3:::secr.lpredictor (formula = object$model[['pmix']], newdata = newdata,
                indx = object$parindx[['pmix']], beta = object$fit$par,
                field = 'pmix',
                beta.vcv = object$beta.vcv)
    pmix <- invlogit(pmix[,'estimate'])
    est <- sum(bmat[,'estimate'] * pmix)
    SEest <- sum( pmix * sqrt(bmat[,'se']^2 + (bmat[,'estimate']-est)^2) )
    exp(c(est=est, lcl=est-1.96*SEest, ucl=est+1.96*SEest))
}

# lambda0
lev <- data.frame(sex=rep(c('F','M'), c(4,4)), bk = rep(rep(0:1, each=2),2), session=rep(1:2,4))
cbind(lev, t(mapply(betabar, sex=lev$sex, bk=lev$bk, session=lev$session )))
```

```
##   sex bk session       est       lcl       ucl
## 1   F  0       1 0.15503280 0.08330451 0.2885218
## 2   F  0       2 0.08997077 0.03437065 0.2355131
## 3   F  1       1 0.65024187 0.36350674 1.1631545
## 4   F  1       2 0.37735732 0.14806987 0.9616984
```

```
## 5   M  0       1 0.12018820 0.06352297 0.2274013
## 6   M  0       2 0.06974927 0.02666300 0.1824611
## 7   M  1       1 0.50409592 0.27928710 0.9098619
## 8   M  1       2 0.29254389 0.11556965 0.7405225
```

```
# sigma
lev <- data.frame(sex = rep(c('F','M'), each=2), session = rep(1:2,2))
cbind(lev, t(mapply(betabar, sex = lev$sex, session = lev$session,
                    MoreArgs = list(bk = 0, thetaname = 'sigma'))))
```

```
##   sex session      est      lcl      ucl
## 1   F       1 3550.573 2282.619 5522.852
## 2   F       2 4109.865 2343.002 7209.123
## 3   M       1 4895.071 3183.772 7526.205
## 4   M       2 5666.151 3255.744 9861.115
```

Population size

```
load(file=paste0(workdir,"/groupfits.Rdata"))
load(file=paste0(workdir,"/densityfits.Rdata"))
load(file=paste0(workdir,"/densityfits2.Rdata"))
load(file = paste0(workdir, '/mask.RData'))

# create a region mask corresponding to
# 2-km x 2-km mask cells centred in the study area
OK <- pointsInPolygon(mask2000b[[1]], Studyarea1)
OK2 <- pointsInPolygon(mask2000b[[2]], Studyarea1)
maskstudy <- list(subset(mask2000b[[1]], OK), subset(mask2000b[[2]],OK2))
class(maskstudy) <- class(mask2000b)

# manually drop unwanted columns from userd non-Euclidean distance matrix
# we want only columns matching region mask 'maskstudy'
Nfn <- function (object, groups=FALSE) {
    OK <- pointsInPolygon(mask2000b[[1]], Studyarea1)
    OK2 <- pointsInPolygon(mask2000b[[2]], Studyarea1)
    maskstudy <- list(subset(mask2000b[[1]], OK), subset(mask2000b[[2]],OK2))
    object$details$userdist[[1]] <- userd[[1]][,OK]
    object$details$userdist[[2]] <- userd[[2]][,OK2]
    if (groups) {
        out <- rbind(region.N(object, region = maskstudy, group=1)[[1]][1,],
              region.N(object, region = maskstudy, group=2)[[1]][1,])
        out <- rbind(out, apply(out,2,sum))
        out[3,'SE.estimate'] <- sqrt(sum(out[1:2,'SE.estimate']^2))
        out <- secr3:::add.cl(out, 0.05,T)
        rownames(out) <- c('Female','Male','Total')
        out
    }
    else
        rbind(region.N(object, region = maskstudy)[[1]][1,])


}
Nfn(densityfits[['fit.fit27']])  # D~ecoregion
```

```
##     estimate SE.estimate      lcl      ucl  n
## E.N 80.97856    7.259935 67.95326 96.50054 75
```

```
Nfn(densityfits[['fit.fit28']])  # D~highground + ecoregion
```

```
##     estimate SE.estimate      lcl      ucl  n
## E.N 81.88355    7.007926 69.25965 96.80841 75
```

```
Nfn(densityfits2[[1]])
```

```
##     estimate SE.estimate      lcl      ucl  n
## E.N 81.86393    7.129951 69.03941 97.07068 75
```

```
Nfn(densityfits2[[2]])
```

```
##     estimate SE.estimate      lcl      ucl  n
## E.N 82.42606    7.143298 69.57204 97.65497 75
```

```
Nfn(densityfits2[[3]])
```

```
##     estimate SE.estimate      lcl      ucl  n
## E.N 82.84156     7.01303 70.19685 97.76399 75
```

```
Nfn(groupfits[[4]], T)
```

```
##        estimate SE.estimate      lcl      ucl  n
## Female 52.10469    6.294563 41.15464 65.96824 40
## Male   32.05899    3.903548 25.27479 40.66421 35
## Total  84.16369    7.406700 70.85336 99.97446 75
```

```
Nfn(groupfits[[6]], T)
```

```
##        estimate SE.estimate      lcl       ucl  n
## Female 57.52581    9.368123 41.89383  78.99062 40
## Male   36.26818    5.094220 27.57717  47.69819 35
## Total  93.79399   10.663621 75.11226 117.12220 75
```

```
Nfn(groupfits[[7]], T)
```

```
##        estimate SE.estimate      lcl       ucl  n
## Female 58.19327    9.380014 42.51564  79.65201 40
## Male   36.56471    5.141421 27.79449  48.10227 35
## Total  94.75798   10.696676 76.00333 118.14054 75
```

Compare detection functions HEX, HHN

```
load(file = paste0(workdir, '/SLCH.RData'))
load(file = paste0(workdir, '/mask.RData'))
baseargs <- list(capthist = 'SLCH', mask = 'mask2000b', trace = FALSE,
        details = list(distribution = 'binomial', userdist = userd), groups = 'SEX',
        model = list(D ~ g+highground, lambda0 ~ g+h2+bk+session, sigma ~ g+h2+session))
fit22args <- c(baseargs,  detectfn = 'HEX')
fit18args <- c(baseargs,  detectfn = 'HHN')
hhnfits <- par.secr.fit(list(fit17=fit22args, fit18=fit18args), ncores = 2)
save(hhnfits, file=paste0(workdir,"/hhnfits.Rdata"))
```

```
load(file=paste0(workdir,"/hhnfits.Rdata"))
AIC(hhnfits)
```

```
##                                                                                 model
## fit.fit17 D~g + highground lambda0~g + h2 + bk + session sigma~g + h2 + session pmix~h2
## fit.fit18 D~g + highground lambda0~g + h2 + bk + session sigma~g + h2 + session pmix~h2
##                      detectfn npar    logLik      AIC     AICc dAICc AICcwt
## fit.fit17 hazard exponential   13 -1756.879 3539.759 3542.647 0.000 0.7772
## fit.fit18  hazard halfnormal   13 -1758.129 3542.257 3545.146 2.499 0.2228
```

```
lev <- data.frame(sex=rep(c('F','M'), c(4,4)), bk = rep(rep(0:1, each=2),2), session=rep(1:2,4))
#lambda0 HEX
cbind(lev, t(mapply(betabar, sex=lev$sex, bk=lev$bk, session=lev$session,
                    MoreArgs=list(thetaname='lambda0', object=hhnfits[[1]]) )))
```

```
##   sex bk session       est       lcl       ucl
## 1   F  0       1 0.15502615 0.08330590 0.2884923
## 2   F  0       2 0.08996240 0.03436945 0.2354775
## 3   F  1       1 0.65021771 0.36350511 1.1630733
## 4   F  1       2 0.37732439 0.14806176 0.9615832
## 5   M  0       1 0.12018443 0.06352177 0.2273913
## 6   M  0       2 0.06974359 0.02666121 0.1824436
## 7   M  1       1 0.50408300 0.27927984 0.9098389
## 8   M  1       2 0.29252173 0.11556068 0.7404678
```

```
#lambda0 HHN
cbind(lev, t(mapply(betabar, sex=lev$sex, bk=lev$bk, session=lev$session,
                    MoreArgs=list(thetaname='lambda0', object=hhnfits[[2]]) )))
```

```
##   sex bk session       est       lcl       ucl
## 1   F  0       1 0.06021718 0.03707099 0.09781527
## 2   F  0       2 0.03377491 0.01530994 0.07451003
## 3   F  1       1 0.29167216 0.18075523 0.47065110
## 4   F  1       2 0.16359451 0.07416297 0.36086964
## 5   M  0       1 0.05315677 0.03050305 0.09263475
## 6   M  0       2 0.02981483 0.01305402 0.06809584
## 7   M  1       1 0.25747387 0.15377023 0.43111591
## 8   M  1       2 0.14441321 0.06468000 0.32243625
```

```
lev <- data.frame(sex=rep(c('F','M'), each=2), session=rep(1:2,2))
#sigma HEX
cbind(lev, t(mapply(betabar, sex=lev$sex, session=lev$session,
                    MoreArgs=list(bk=0, thetaname='sigma', object=hhnfits[[1]]))))
```

```
##   sex session      est      lcl      ucl
## 1   F       1 3550.673 2282.669 5523.042
## 2   F       2 4110.107 2343.114 7209.626
## 3   M       1 4895.133 3183.768 7526.406
## 4   M       2 5666.397 3255.795 9861.818
```

```
#sigma HHN
cbind(lev, t(mapply(betabar, sex=lev$sex, session=lev$session,
                    MoreArgs=list(bk=0, thetaname='sigma', object=hhnfits[[2]]))))
```

```
##   sex session      est      lcl       ucl
## 1   F       1 6172.543 4352.470  8753.717
## 2   F       2 7429.007 4590.294 12023.227
```

```
## 3   M       1 7023.620 4705.671 10483.358
## 4   M       2 8453.327 5045.889 14161.771
```

Supplementary models using Ecoregion as a predictor of density

```
load(file = paste0(workdir, '/SLCH.RData'))
load(file = paste0(workdir, '/mask.RData'))
detailslist <- list(distribution = 'binomial', userdist = userd)
baseargs <- list (capthist = 'SLCH', mask = 'mask2000b', trace = FALSE,
        details = 'detailslist', detectfn = 'HEX', groups = 'SEX')
fit21args  <- c(baseargs, list(model = list(D ~ g+highground,
                                            lambda0 ~ g+bk+h2, sigma ~ g+h2)))
fit25args  <- c(baseargs, list(model = list(D ~ g+ECOREGION,
                                            lambda0 ~ g+bk+h2, sigma ~ g+h2)))
fit26args  <- c(baseargs, list(model = list(D ~ g+highground+ECOREGION,
                                            lambda0 ~ g+bk+h2, sigma ~ g+h2)))
fit27args  <- c(baseargs, list(model = list(D ~ g+highground*ECOREGION,
                                            lambda0 ~ g+bk+h2, sigma ~ g+h2)))
groupfitsSuppl <- par.secr.fit(list(fit21=fit21args, fit25=fit25args,
                                    fit26=fit26args, fit27=fit27args),
                           ncores = 4)
save(groupfitsSuppl, file=paste0(workdir, "/groupfitsSuppl.Rdata"))
```

```
load(file=paste0(workdir, "/groupfitsSuppl.Rdata"))
# adding ecoregion does not significantly improve fit
LR.test(groupfitsSuppl$fit.fit26, groupfitsSuppl$fit.fit21)
```

```
##
##  Likelihood ratio test for two models
##
## data:  groupfitsSuppl$fit.fit26 vs groupfitsSuppl$fit.fit21
## X-square = 2.5374, df = 1, p-value = 0.1112
```

```
AIC(groupfitsSuppl)[,-2]
```

```
##                                                                        model
## fit.fit26 D~g + highground + ECOREGION lambda0~g + bk + h2 sigma~g + h2 pmix~h2
## fit.fit21             D~g + highground lambda0~g + bk + h2 sigma~g + h2 pmix~h2
## fit.fit27 D~g + highground * ECOREGION lambda0~g + bk + h2 sigma~g + h2 pmix~h2
## fit.fit25              D~g + ECOREGION lambda0~g + bk + h2 sigma~g + h2 pmix~h2
##           npar    logLik      AIC     AICc dAICc AICcwt
## fit.fit26   12 -1758.421 3540.843 3543.299 0.000 0.3781
## fit.fit21   11 -1759.690 3541.380 3543.443 0.144 0.3518
## fit.fit27   13 -1757.580 3541.159 3544.048 0.749 0.2600
## fit.fit25   11 -1763.240 3548.481 3550.543 7.244 0.0101
```

```
Nfn(groupfitsSuppl[['fit.fit26']],T)  # D~highground+ecoregion
```

```
##        estimate SE.estimate      lcl       ucl  n
## Female 56.86634    9.163102 41.55045  77.82781 40
## Male   36.72166    5.112239 27.98915  48.17867 35
## Total  93.58799   10.492733 75.17692 116.50800 75
```

Intermediate results

Intermediate results from the computations in this vignette are saved in the following R binary files for future reference, and to streamline the process of 'knitting' this document:

File	Contents
SLwater.RData	water bodies SpatialPolygonsDataFrame
SLCH.RData	hair snare capthist object
mask.RData	habitat masks, syudy area polygon and related objects
AllCollars.Rdata	GPS collar data
ch.RData	capthist objects for GPS data, standalone (CH) or with hair snares (ch)
fits2.RData	SECR model fits: barriers vs no barriers
fits1ab.RData	SECR model fits: missing usage data
fitsT.RData	SECR model fits: telemetry
densityfits.Rdata	SECR model fits: missing usage data
densityfits2.Rdata	SECR model fits: year effects and elevation
groupfits.Rdata	SECR model fits: spline elevation models
groupfitsSuppl.Rdata	SECR model fits: ecoregion
hhnfits.Rdata	SECR model fits: compare halfnormal and negative exponential detection function

Figures

The code here may be used to plot the figures in the report. First retrieve the data:

```
load(file = paste0(workdir, '/mask.RData'))  # mask2000b, maskstudy250, Studyarea1, trueGrid)
load(file = paste0(workdir, '/SLwater.RData')) # SLwater
load(file = paste0(workdir, '/ch.RData'))      # CH
load(file = paste0(workdir, '/SLCH.RData'))    # SLCH, siteobj
load(file = paste0(workdir, '/AllCollars.Rdata')) # collarsummary, AllCollars, AC.split.locs
```

Define two useful functions. Also need getelevation(), reproject, dem and demsuppl from above.

```
getfixes <- function(df, id, y, mn) {
    df[df$id==id & df$year==y & df$mon %in% mn,]
}
myplot <- function(df, mcol = c('orange','forestgreen','blue'),
                   y = 2012, mn = 6:8, minpts = 10, add = FALSE, ...) {
    # mn 6:8 corresp June, July, Aug
    df <- df[order(df$date, df$time),]
    OK <- (df$year == y) & (df$mon %in% mn)
    if (sum(OK)>= minpts) {
        if (!add) plot(Studyarea1)
        lines(df[OK, c('x','y')], col = 'black', xpd = TRUE)
        points(df[OK, c('x','y')], col = mcol[df$mon[OK]-5], xpd = TRUE,
               type = 'o', ...)
        mtext(side = 3, df$id[1], cex = 0.8, adj=0.05, line=-1)
        TRUE
    }
    else
        FALSE
```

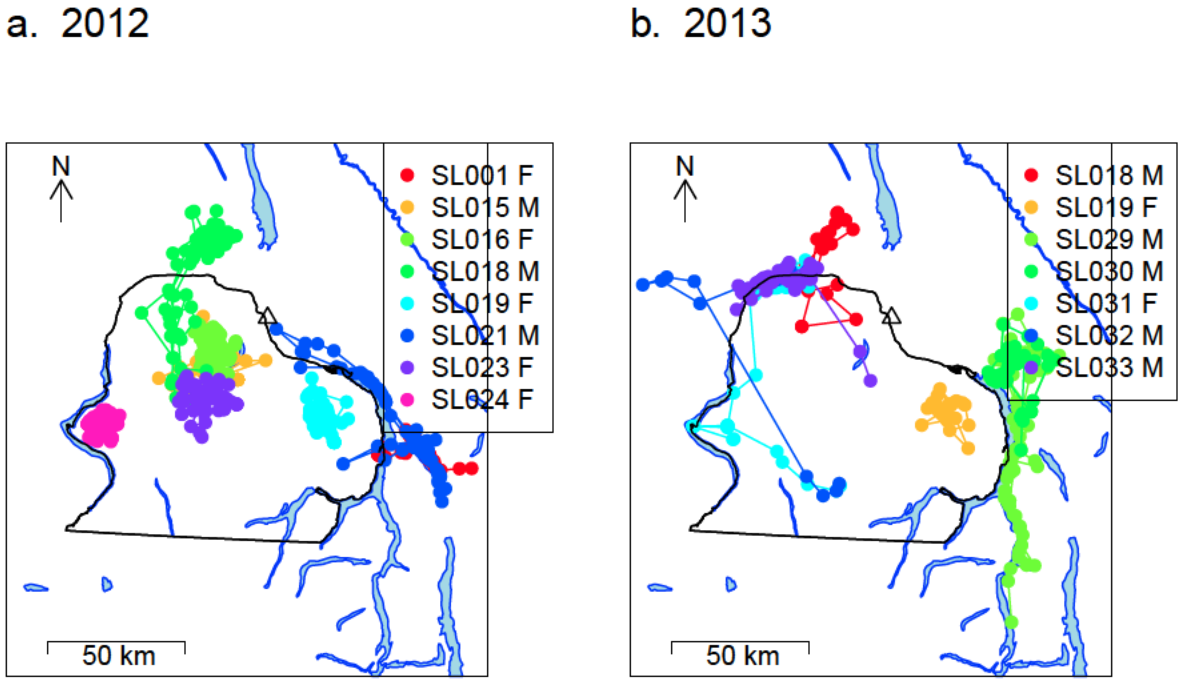
```
}
```

Main text Figure 2: Comparative distribution of elevation. (maskstudy250, siteobj)

```
hab <- ecdf(covariates(maskstudy250)$elevation)  # median 1254
siteobj <- addCovariates(siteobj, maskstudy250)  # attach elevation covariate to snare sites
trp <- ecdf(covariates(siteobj[[1]])$elevation)
par(mfrow=c(1,1), mar=c(5,6,5,4))
plot(hab, xlim=c(500,2200), xlab='Elevation  (m)', ylab='Cumulative proportion', main='')
plot(trp, add=T, verticals=T, col='red', cex=0)
abline(h=0.5, lty=2)
legend ('bottomright', lty=1, col=c('red','black'), legend=c('Hair snares','Habitat'))
```

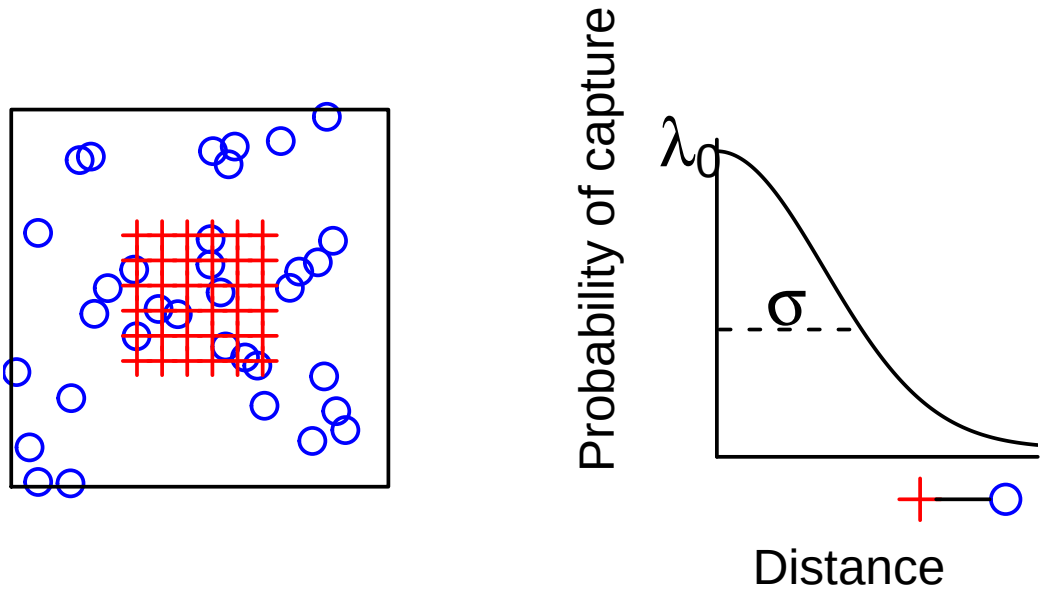
Main text Fig. 3: Telemetry. (SLwater, CH)

```
SLWdf <- as.data.frame(SLwater)
par(mfrow=c(1,2), mar = c(0,2,2,2), xpd = F, xaxs='i',yaxs='i')
icolours <- rainbow(8)
for (i in 1:2) {
    plot(Studyarea1, xlim=c(265000, 440000), ylim=c(564400, 758600))
    plot(SLwater[SLWdf[,'SHAPE_Area']>1e7,], col = 'lightblue',
         border = 'blue', add = TRUE, xpd = FALSE)
    plot(CHall[[i]], icolours = icolours, gridsp=10000, type='telemetry',
         tracks = TRUE, add = TRUE, cappar=list(pch=16, cex=1),
         title='',subtitle='')
    mtext(side=3, adj=0, c('a.  2012', 'b.  2013')[i], col = 'black', cex=1.3)
    plot(Studyarea1, add = TRUE)
    rect(265000, 564750, 440000, 758600)
    opar <- palette(icolours)
    sl <- rownames(CHall[[i]])
    sex <- collarsummary$Sex[match(sl, collarsummary$BEAR_ID)]
    legend(402000, 758500, legend= paste(sl,sex), col=1:nrow(CHall[[i]]),
           pch=16, cex=0.9, pt.cex=1, xpd=TRUE, text.col='black')
    palette(opar)
    par(col='black')
    lines(c(280000, 280000, 330000, 330000), c(573000, 577000, 577000, 573000))
    text(306000, 572000, '50 km', cex=0.9)
    points(360000, 695000, pch=2)  # whitehorse
    arrows(285000, 730000, 285000, 745000, length=0.12)
    text(285000, 750000, 'N', cex=0.9)
}
```



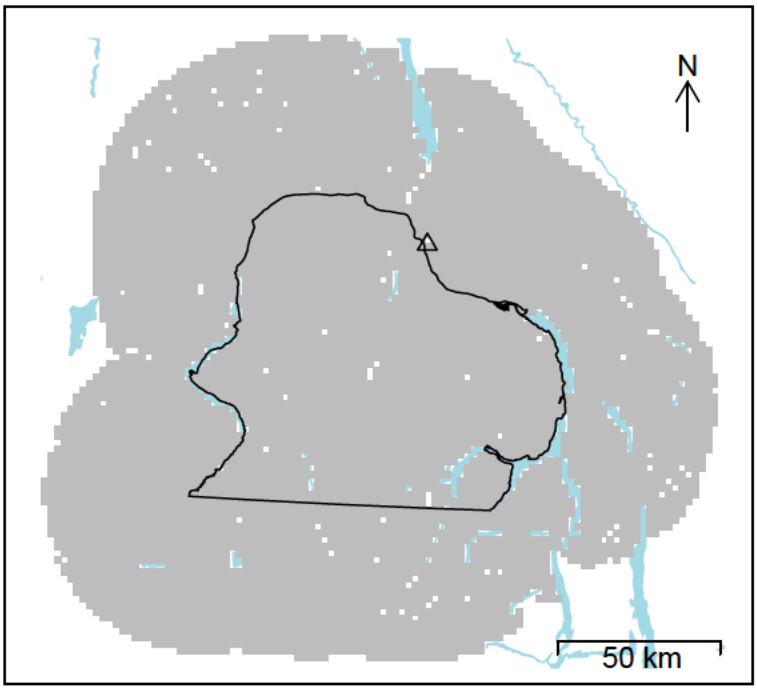
Main text Figure 4. Schematic of SECR submodels.

```
tr <- make.grid()
pop <- sim.popn(tr, D=4)
par(lwd=2, mfrow=c(1,2))

par(cex=1.3, mar = c(2,2,2,3))
plot(pop, frame=T, pch=1, col='blue', cex=1.5)
plot(tr, add=T, detpar=list(cex=1.5, col='red'), gridl = F)

par(cex=1.7, mar=c(4,3,4,2), mgp = c(1.5,1,0), xpd=F)
xv <- seq(0,60,0.5)
plot(xv, 0.7 * exp(-xv^2/2/20^2), type='l', ylab = 'Probability of capture',
     xlab='', axes=F, bty='l', xaxs='i', adj=0.1)
mtext(side=1, line=1.3, adj=0.28, 'Distance', cex=1.7)
axis(1, at=c(-100,100), lwd=2)
axis(2, at=c(-100,100), lwd=2)
v <- -0.12 # adjust as needed
points(38, v, pch=3, col='red', cex=1.3, xpd = TRUE)
points(54, v, pch=1, col='blue', cex=1.3, xpd = TRUE)
segments(41, v,51,v, xpd=T)
text(-5, 0.71, expression(lambda[0]), cex=1.3, xpd=T)
text(13, 0.33, expression(sigma), cex=1.3)
segments(0.5, 0.28, 25.5, 0.28, xpd=T, lty=2)
```



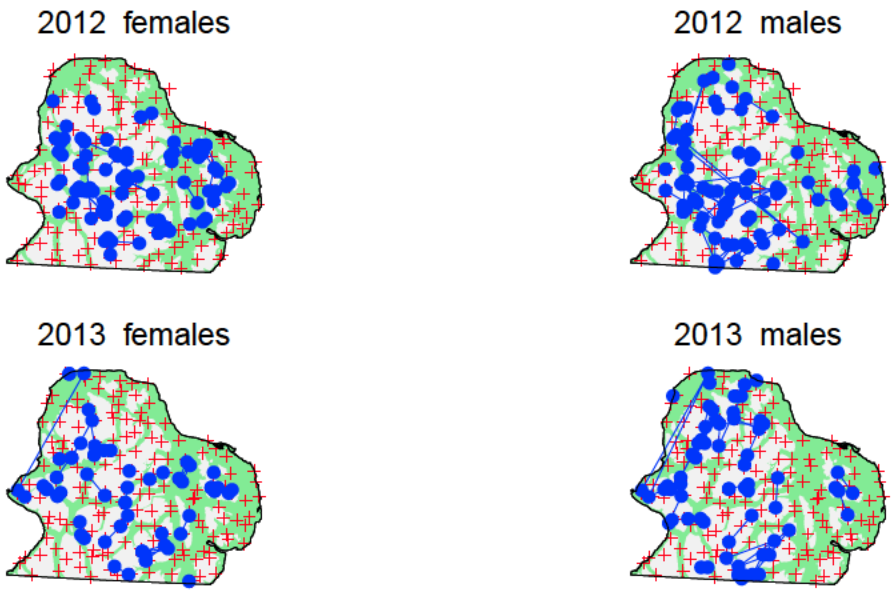
Main text Figure 5. (SLwater, Studyarea1, mask2000b)

```
par(mfrow=c(1,1), mar = c(2,2,2,4))
SLWdf <- as.data.frame(SLwater)
plot(mask2000b[[1]], ppoly=FALSE, dots=FALSE, border=10000)
plot(Studyarea1, add = TRUE)
plot(SLwater[SLWdf[,'SHAPE_Area']>1e7,], col = 'lightblue',
     border = NA, add = TRUE)

plot(mask2000b[[1]], ppoly=F, dots=FALSE, add=TRUE, bty='o')
plot(Studyarea1, add = TRUE)
rect(230000, 560000, 460000, 768000)
par(col='black')

lines(c(400000, 400000, 450000, 450000), c(569000, 573000, 573000, 569000))
text(426000, 568000, '50 km', cex=0.9)

points(360000, 695000, pch=2)  # whitehorse
arrows(440000, 730000, 440000, 745000, length=0.12)
text(440000, 750000, 'N', cex=0.9)
```



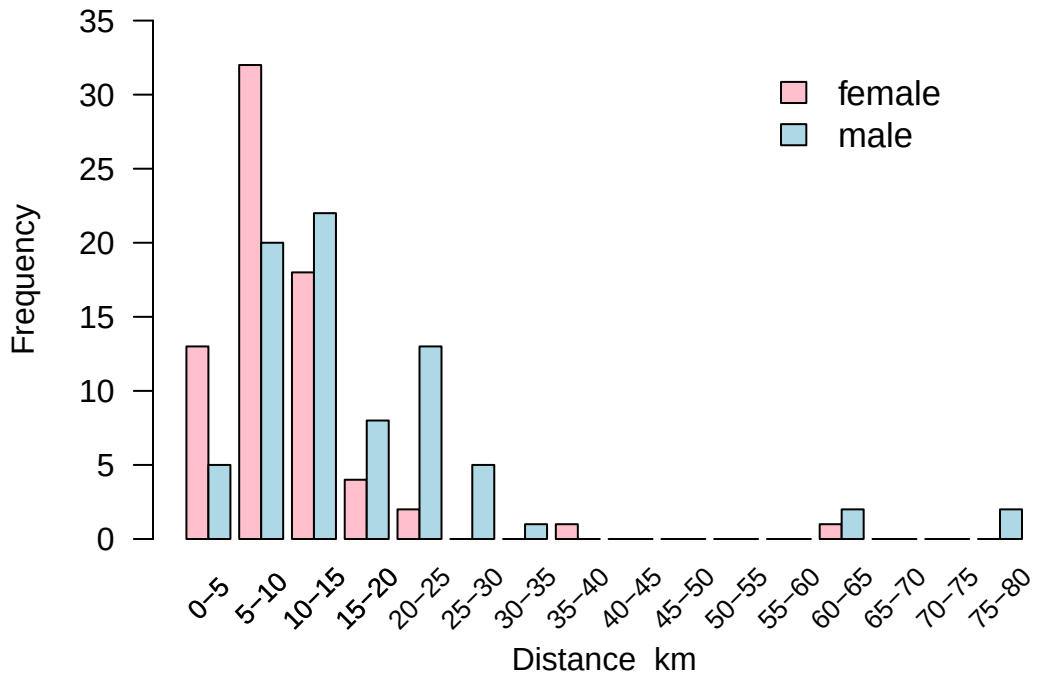
Main text Figure 6. (trueGrid, SLCH, Studyarea1)

```
SLCHFM <- mapply(split, SLCH, lapply(SLCH, function(x) covariates(x)$SEX), SIMPLIFY=FALSE)
par(mfrow = c(2,2), mar=c(1,1,2.5,1))
for (i in 1:2) {
    for (j in 1:2) {
        plot(Studyarea1, col = 'lightgreen')
        lapply(consave, polygon, col=grey(0.95), border=NA)
        plot(traps(SLCH[[i]]), add=TRUE)
        plot(SLCHFM[[i]][[j]], add = TRUE, rad=1500, tracks = TRUE, varycol = FALSE, border = 5000,
             gridl = FALSE, bty='o', title = '', subtitle='')
        mtext(side=3,line=0.5, cex=1.3, paste(i+2011, c('females','males')[j], sep='  '))
        plot(Studyarea1, add=T)
    }
}
```



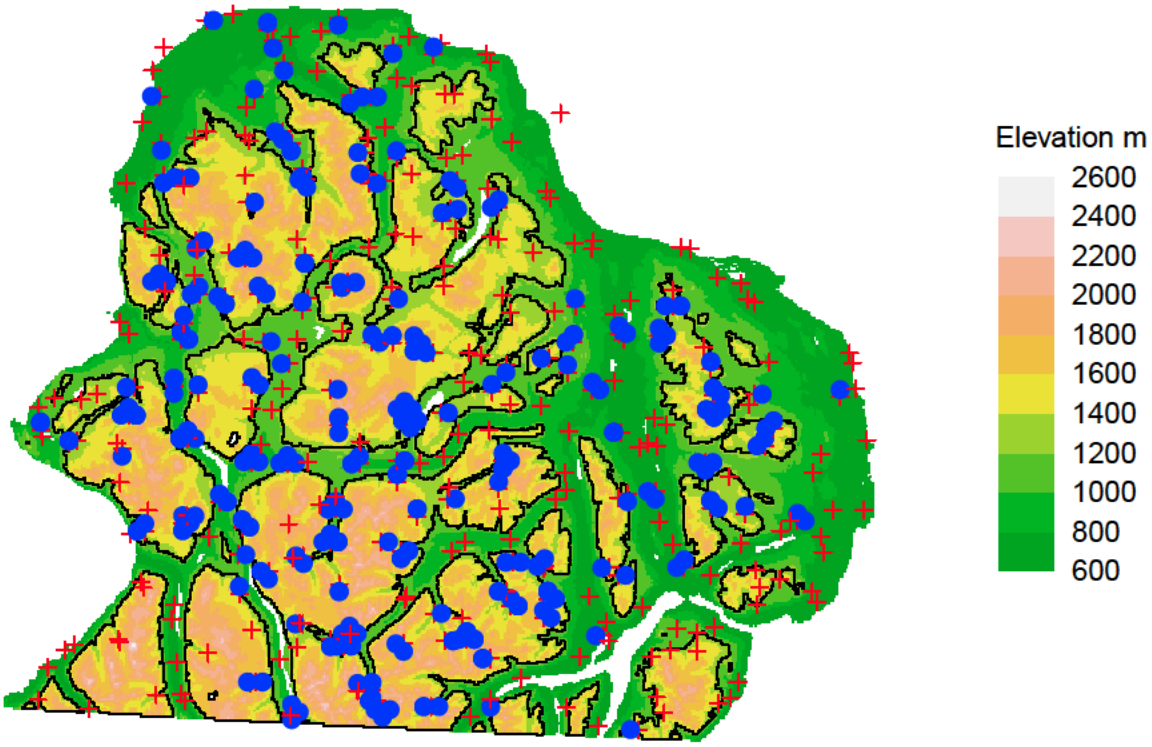
Main text Figure 7. Distribution of snare-revealed movements. (SLCH)

```
SLCHFM <- mapply(split, SLCH, lapply(SLCH, function(x) covariates(x)$SEX), SIMPLIFY=FALSE)
f2012 <- unlist(moves(SLCHFM[[1]][[1]]))
m2012 <- unlist(moves(SLCHFM[[1]][[2]]))
f2013 <- unlist(moves(SLCHFM[[2]][[1]]))
m2013 <- unlist(moves(SLCHFM[[2]][[2]]))
fF <- table(cut(c(f2012,f2013), breaks=seq(0,80000,5000)))
fM <- table(cut(c(m2012,m2013), breaks=seq(0,80000,5000)))
par(mfrow=c(1,1), mar=c(5,5,4,4), cex=1)
pos <- barplot(height=t(cbind(fF,fM)), beside=TRUE, axisnames=F, las=1, ylim=c(0,35), space=c(0,0.4),
        col=c('pink','lightblue'))
xmid <- apply(pos,2,mean)
text(xmid, rep(-4,20), paste(seq(0,75,5), seq(5,80,5), sep='-'), cex=0.85, srt=45, xpd=T)
mtext(side=1, line=2.6, 'Distance  km', cex=1)
mtext (side=2, line = 2.9, 'Frequency', cex = 1)
par(srt=0)
legend (26, 33, legend=c('female','male'), fill=c('pink','lightblue'), bty='n', cex=1.1)
```



Main text Figure 8. (maskstudy250, SLCH). Computation is SLOW.

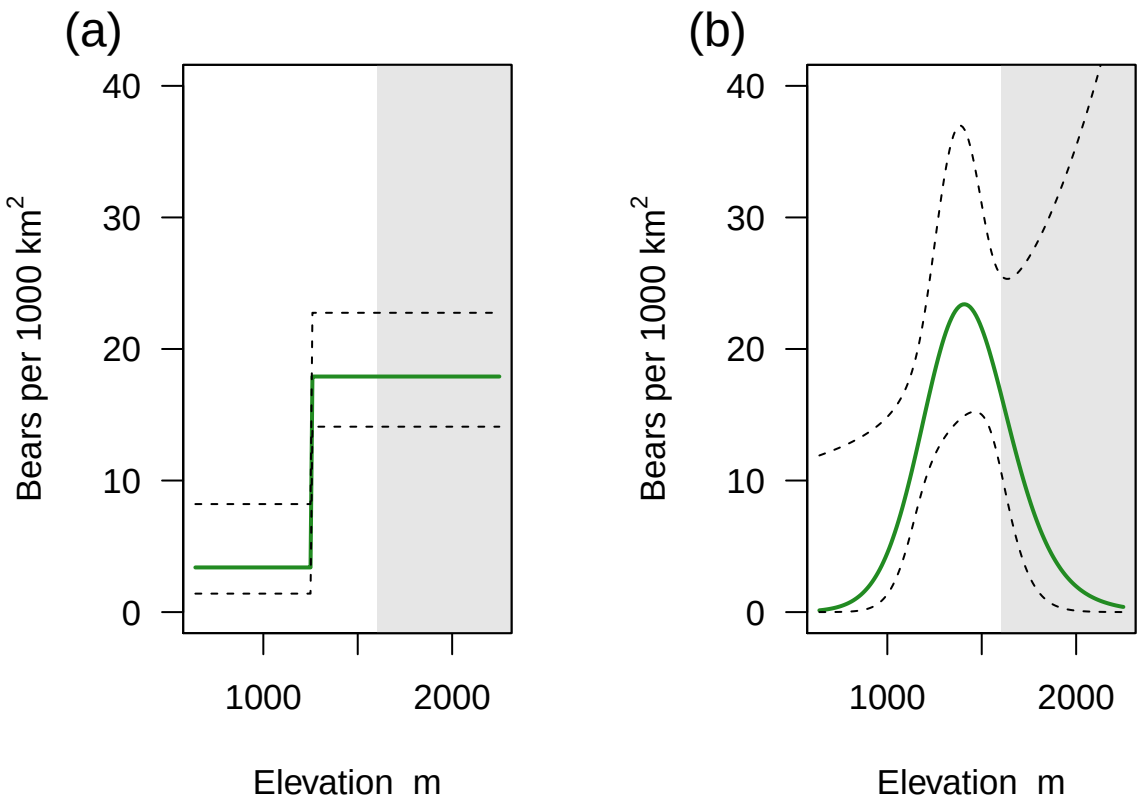
```
load(file = paste0(workdir, '/mask.RData'))
par(mfrow = c(1,1), mar = c(1,1,1,8))
plot(maskstudy250, dots = FALSE, cov = 'elevation', title = 'Elevation m')
segments (c1250[1,], c1250[2,], c1250[3,], c1250[4,], col = 'black')
for (sess in 1:2) {
    plot(traps(SLCH)[[sess]], add=T)
    plot(SLCH[[sess]], add=T, title='', subtitle='', varycol=FALSE, rad=1000)
}
```



Main text Figure 9. Density vs elevation. These are 'hcov' models so the same total density is reported for F and M.

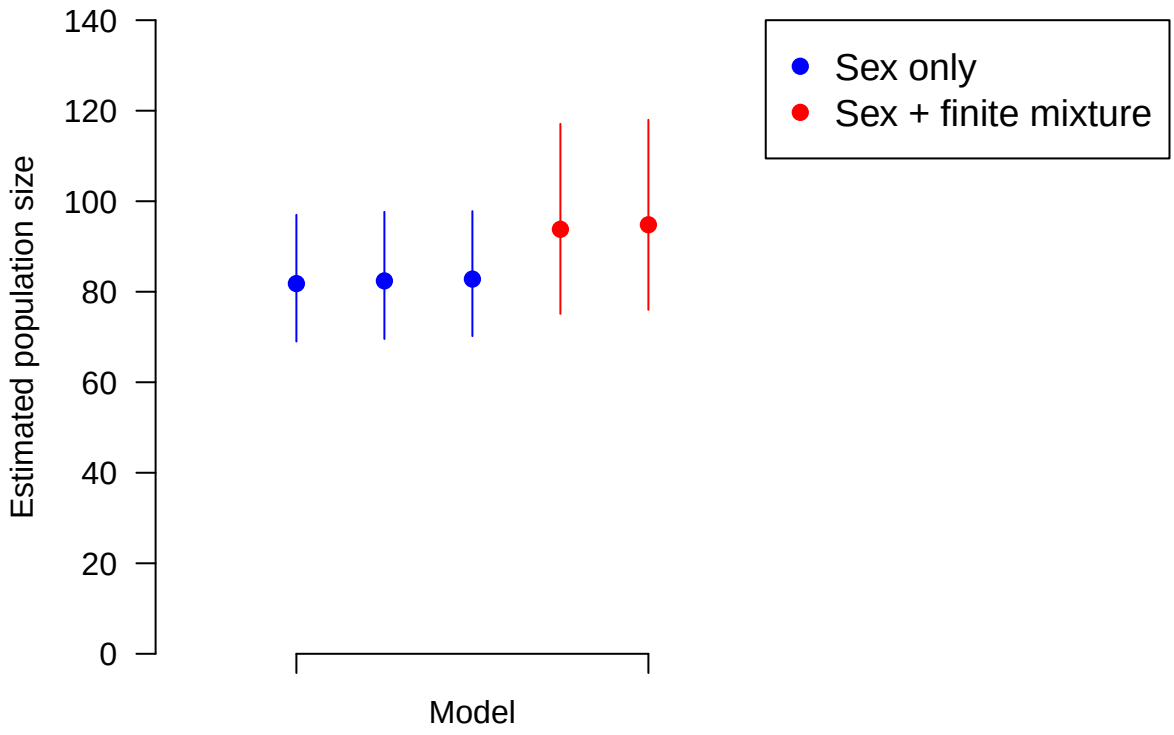
```
load(file=paste0(workdir,"/densityfits2.Rdata"))

xv <- seq(640,2250,10)
nxv <- length(xv)
newdat <- expand.grid(elevation=xv, bk=0, h2=c('F','M'))
newdat$highground <- newdat$elevation>1250
predhg <- predict(densityfits2[[1]], newdata=newdat)
pred3 <- predict(densityfits2[[2]], newdata=newdat)
pred5 <- predict(densityfits2[[3]], newdata=newdat)

par(mfrow=c(1,2), mar=c(5,5,2,2), cex=1.1)

plot(xv, rep(0,nxv), ylim=c(0,40), xlab='Elevation  m',
     ylab = expression(paste('Bears per 1000 ', km^2)), type='n', las=1)
polygon(c(1600,1600,2500,2500),c(-2,42,42,-2), col=grey(0.9), xpd=F, border=NA)
lines(xv, 1e5 * sapply(predhg[1:nxv],'[[',1,2), col='forestgreen', lwd=2)
lines(xv, 1e5 * sapply(predhg[1:nxv],'[[',1,4),lty=2)
lines(xv, 1e5 * sapply(predhg[1:nxv],'[[',1,5),lty=2)
text(100,44,'(a)',cex=1.4, xpd=T)
box()

plot(xv, rep(0,nxv), ylim=c(0,40), xlab='Elevation  m',
     ylab = expression(paste('Bears per 1000 ', km^2)), type='n', las=1)
polygon(c(1600,1600,2500,2500),c(-2,42,42,-2), col=grey(0.9), xpd=F, border=NA)
```

```
lines(xv, 1e5 * sapply(pred3[1:nxv],'[[',1,2), col='forestgreen', lwd=2)
lines(xv, 1e5 * sapply(pred3[1:nxv],'[[',1,4),lty=2)
lines(xv, 1e5 * sapply(pred3[1:nxv],'[[',1,5),lty=2)
text(100,44,'(b)',cex=1.4, xpd=T)
box()
```



A supplementary plot to visualise effect of including finite mixture h2 in models (not used 2017-02-16).

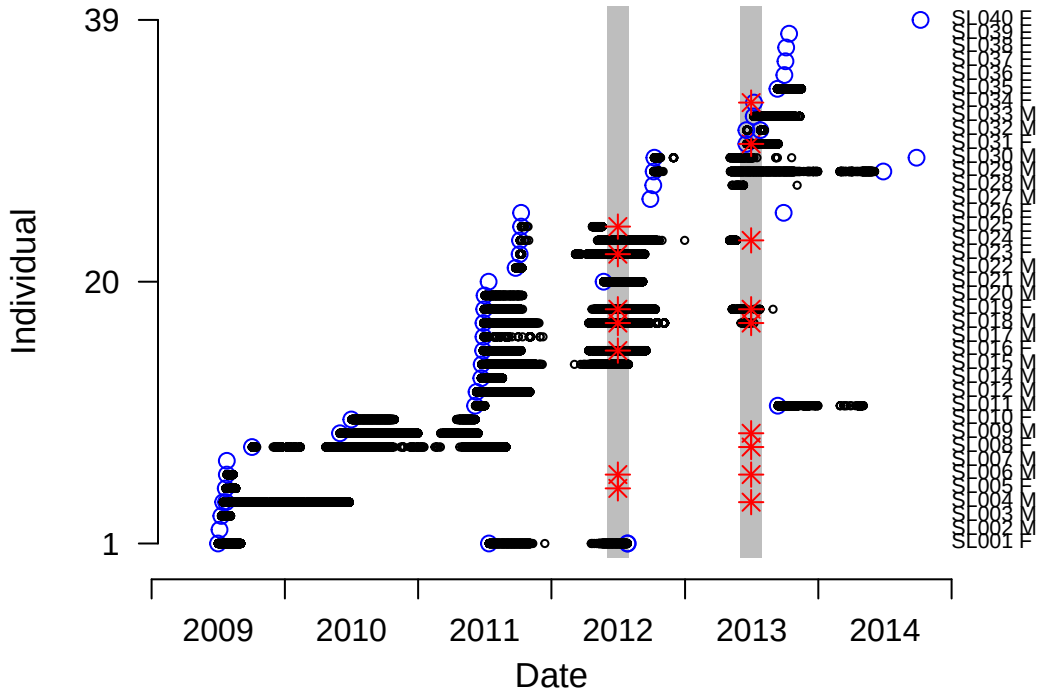
```
par(mfrow=c(1,1), pty='m', yaxs='i', mar=c(3,4,3,12))
plot(0,0,type='n', xlab = '', ylab = 'Estimated population size', xlim = c(0,1),
     ylim = c(0,140), axes = FALSE)
mtext(side=1, line=1, 'Model')
xv <- seq(0.2,0.8,0.15)
cols <- rep(c('blue','red'), c(3,2))
points (xv, c(81.8,82.4,82.8, 93.8, 94.8), pch=16, cex = 1.2, col=cols)
segments (xv, c(69,69.57,70.2, 75.1, 76), xv, c(97,97.65,97.8, 117.1, 118), col=cols)
axis(2, las=1)
axis(1, at = c(0.2,0.8), label=F)
legend (1,140,pch=16,cex=1.2, col=c('blue','red'), xpd = TRUE,
        legend=c('Sex only', 'Sex + finite mixture'))
```



Appendix 1 Figure: histories of GPS-collared bears. (collarsummary, AllCollars, SLCH)

```
collarsplit <- split(collarsummary, collarsummary$BEAR_ID)
telemsplit <- split(AllCollars, AllCollars$id)
one <- function(x) {
    idno <<- idno + 1
    points (x$Date, rep(idno,nrow(x)), cex=1.1, col='blue')
}
onetelem <- function(x) {
    if (!is.null(x))
        points (x$date, rep(idno,nrow(x)), cex=0.5)
}

par(mfrow=c(1,1), mar=c(4,4,3,6), xpd=T)
idno <- 0
plot(collarsummary$Date,rep(0,nrow(collarsummary)), axes = FALSE,
     type = 'n', xlab = '',ylab = '',
     xlim = as.Date(c('2009-01-01', '2015-01-01')), ylim = c(0,40))

axis(1, at = seq.Date(as.Date('2009-01-01'), as.Date('2015-01-01'), by = 'year'),
     lab = FALSE)
axis(2, las=1, at = seq(1,39,19), line=-1)
mtext(side=1, at = seq.Date(as.Date('2009-01-01'), as.Date('2014-01-01'),
                            by = 'year')+365/2, 2009:2014, line = 0.8)
mtext(side=1, 'Date', line=2, cex=1.1)
mtext(side=2, 'Individual', line=2, cex=1.1)
polygon(x=as.Date(c('01-06-2012', '01-06-2012', '31-07-2012', '31-07-2012','01-06-2012'),
```

```
                  format = "%d-%m-%Y"),
        y=c(0,40,40,0,0), col='grey', border=NA)
polygon(x=as.Date(c('01-06-2013', '01-06-2013', '31-07-2013', '31-07-2013','01-06-2013'),
                  format = "%d-%m-%Y"),
        y=c(0,40,40,0,0), col='grey', border=NA)
for (i in 1:39) {
    sl <- names(collarsplit)[i]
    sex <- collarsummary$Sex[match(sl, collarsummary$BEAR_ID)]
    text(as.Date('2015-01-01'), i+0.2, paste(sl,sex), adj=0, cex=0.7)
    one(collarsplit[[sl]])
    if (sl %in% names(telemsplit))
        onetelem(telemsplit[[sl]])
    if (sl %in% rownames(SLCH[[1]]))
        points(as.Date('2012-07-01'), i, cex=1.2, pch=8, col='red')
    if (sl %in% rownames(SLCH[[2]]))
        points(as.Date('2013-07-01'), i, cex=1.2, pch=8, col='red')
}
```



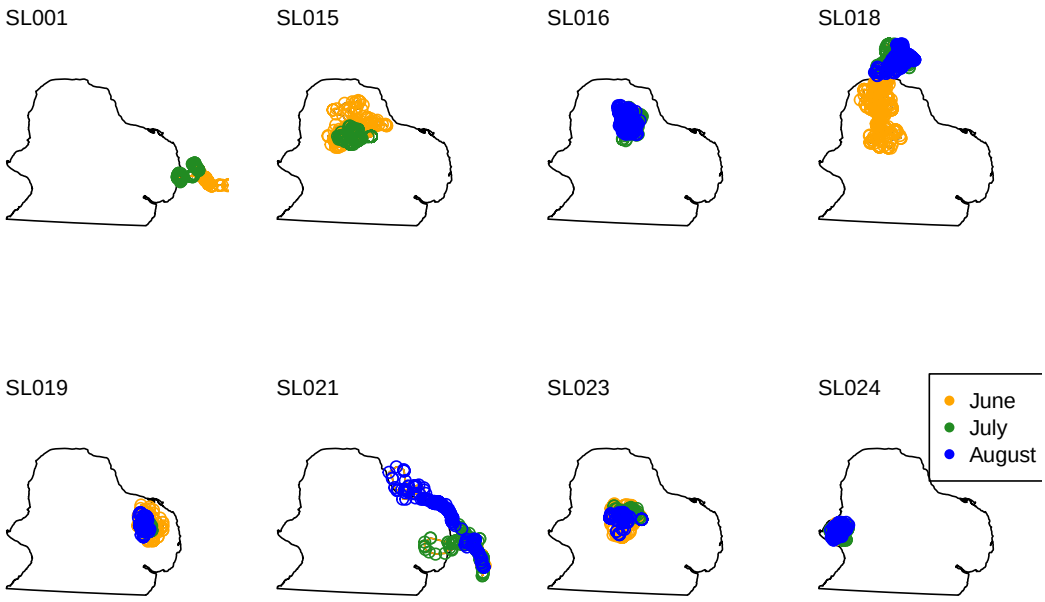
Appendix figure: Monthly stratified telemetry. (AC.split.locs, collarsummary)

```
par(mfrow=c(2,4), mar=c(2,2,2,2), oma=c(0,2,2,6))
for (i in 1:length(AC.split.locs)) {
    if (grepl('2012',names(AC.split.locs)[i])) {
        nonzero <- myplot(AC.split.locs[[i]], y=2012, cex=1.5)
        if (nonzero) print(i)
    }
```

```
}
```

```
## [1] 88
## [1] 99
## [1] 100
## [1] 102
## [1] 103
## [1] 105
## [1] 107
## [1] 108
```

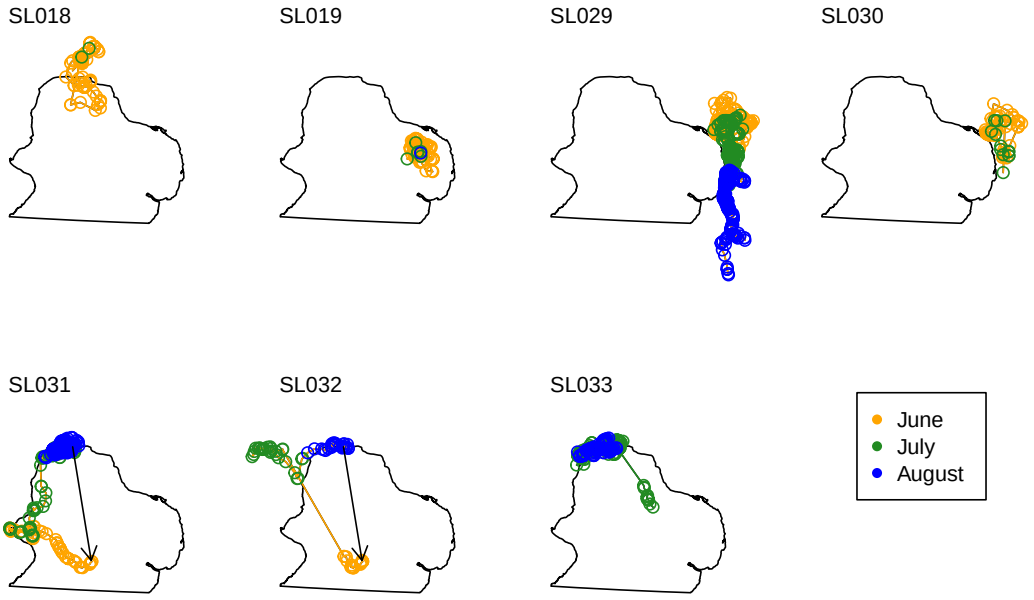
```
legend(360000, 760000, legend=c('June','July','August'),
       col=c('orange','forestgreen','blue'), pch = 16, xpd = TRUE, cex = 1.3)
```




```
par(mfrow = c(2,4), mar = c(2,2,2,2))
for (i in 1:length(AC.split.locs)) {
    if (grepl('2013',names(AC.split.locs)[i])) {
        nonzero <- myplot(AC.split.locs[[i]], y=2013, cex=1.5)
        id <- substring(names(AC.split.locs)[i], 1,5)
        if (id %in% c('SL031', 'SL032')) {
            xy <- collarsummary[match(id, collarsummary$BEAR_ID),]
            arrows(xy$capture.x, xy$capture.y, xy$release.x, xy$release.y, length=0.1)
        }
        if (nonzero) print(i)
    }
}
```

```
## [1] 131
## [1] 132
## [1] 140
## [1] 141
## [1] 142
## [1] 143
## [1] 144
```

```
plot(Studyarea1, border='white')
legend(310000, 745900, legend=c('June','July','August  '),
       col=c('orange','forestgreen','blue'), pch=16, xpd=TRUE, cex=1.3)
```



This code shows the ecoregions near the Southern Lakes study area (across a buffered 50 km area, using 250-m pixels). The plot is slow to display and wasn't finally used.

```
plot(mask250, cov='ECOREGION', dots=FALSE, xy='topright')
plot(Studyarea1, add=T)
lines(c(270000, 270000, 320000, 320000), c(553000, 557000, 557000, 553000), xpd=TRUE)
text(296000, 552000, '50 km', cex=1, xpd=TRUE)
arrows(245000, 730000, 245000, 745000, length=0.12)
text(245000, 750000, 'N', cex=1)
```