Initial Analysis

Ben Lorentz

2023-09-30

1. Descriptive statistics
2. Read in Data

source(".installpackages.R")

## Package: plyr was already installed.  
## Package: dplyr was already installed.  
## Package: ggplot2 was already installed.  
## Package: gganimate was already installed.  
## Package: tidyverse was already installed.  
## Package: lubridate was already installed.  
## Package: knitr was already installed.  
## Package: png was already installed.  
## Package: grid was already installed.  
## Package: amt was already installed.

## The legacy packages maptools, rgdal, and rgeos, underpinning the sp package,  
## which was just loaded, will retire in October 2023.  
## Please refer to R-spatial evolution reports for details, especially  
## https://r-spatial.org/r/2023/05/15/evolution4.html.  
## It may be desirable to make the sf package available;  
## package maintainers should consider adding sf to Suggests:.  
## The sp package is now running under evolution status 2  
## (status 2 uses the sf package in place of rgdal)

## Package: ctmm was already installed.

## Registered S3 methods overwritten by 'adehabitatMA':  
## method from  
## print.SpatialPixelsDataFrame sp   
## print.SpatialPixels sp

## Package: adehabitatLT was already installed.  
## Package: circular was already installed.  
## Package: move2 was already installed.  
## Package: KernSmooth was already installed.  
## Package: sf was already installed.  
## Package: raster was already installed.  
## Package: terra was already installed.  
## Package: sp was already installed.

## Please note that rgdal will be retired during October 2023,  
## plan transition to sf/stars/terra functions using GDAL and PROJ  
## at your earliest convenience.  
## See https://r-spatial.org/r/2023/05/15/evolution4.html and https://github.com/r-spatial/evolution  
## rgdal: version: 1.6-7, (SVN revision 1203)  
## Geospatial Data Abstraction Library extensions to R successfully loaded  
## Loaded GDAL runtime: GDAL 3.4.2, released 2022/03/08  
## Path to GDAL shared files: /Users/benlorentz/Library/R/x86\_64/4.2/library/rgdal/gdal  
## GDAL binary built with GEOS: FALSE   
## Loaded PROJ runtime: Rel. 8.2.1, January 1st, 2022, [PJ\_VERSION: 821]  
## Path to PROJ shared files: /Users/benlorentz/Library/R/x86\_64/4.2/library/rgdal/proj  
## PROJ CDN enabled: FALSE  
## Linking to sp version:1.6-1  
## To mute warnings of possible GDAL/OSR exportToProj4() degradation,  
## use options("rgdal\_show\_exportToProj4\_warnings"="none") before loading sp or rgdal.

## Package: rgdal was already installed.  
## Package: pracma was already installed.  
## Package: adehabitatHR was already installed.  
## Package: igraph was already installed.  
## Package: ergm was already installed.  
## Package: network was already installed.  
## Package: vroom was already installed.

library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.3 ✔ readr 2.1.4  
## ✔ forcats 1.0.0 ✔ stringr 1.5.0  
## ✔ ggplot2 3.4.3 ✔ tibble 3.2.1  
## ✔ lubridate 1.9.2 ✔ tidyr 1.3.0  
## ✔ purrr 1.0.2

## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

library(amt)

##   
## Attaching package: 'amt'  
##   
## The following object is masked from 'package:stats':  
##   
## filter

library(sf)

## Linking to GEOS 3.10.2, GDAL 3.4.2, PROJ 8.2.1; sf\_use\_s2() is TRUE

library(lubridate)   
library(ggplot2)  
library(terra)

## terra 1.7.46  
##   
## Attaching package: 'terra'  
##   
## The following object is masked from 'package:tidyr':  
##   
## extract

library(png)  
  
  
vulture\_metadata <- read.csv("../data/Black-Vultures-and-Turkey-Vultures-Southeastern-USA-reference-data.csv")  
print(vulture\_metadata)

## tag.id animal.id animal.taxon deploy.on.date  
## 1 171 6 Cathartes aura 2013-06-17 13:19:20.000  
## 2 167 8 Coragyps atratus 2013-06-18 23:23:34.000  
## 3 162 12 Coragyps atratus 2013-06-20 16:40:40.000  
## 4 165 22 Coragyps atratus 2013-06-20 17:36:23.000  
## 5 163 1 Cathartes aura 2013-06-26 14:07:33.000  
## 6 173 3 Cathartes aura 2013-06-26 14:38:58.000  
## 7 180 13 Cathartes aura 2013-06-27 18:39:55.000  
## 8 161 47 Coragyps atratus 2013-07-03 16:06:48.000  
## 9 177 48 Coragyps atratus 2013-07-03 18:51:03.000  
## 10 172 60 Cathartes aura 2013-07-05 12:52:58.000  
## 11 178 57 Coragyps atratus 2013-07-05 16:39:00.000  
## 12 176 75 Cathartes aura 2013-07-09 15:31:45.000  
## 13 166 90 Cathartes aura 2013-07-30 14:34:17.000  
## 14 179 91 Cathartes aura 2013-07-31 15:52:51.000  
## 15 175 92 Coragyps atratus 2013-08-02 12:47:47.000  
## 16 168 108 Coragyps atratus 2014-04-21 18:40:25.000  
## 17 178 123 Cathartes aura 2014-04-23 19:44:16.000  
## 18 174 126 Coragyps atratus 2014-05-01 13:38:20.000  
## deploy.off.date animal.life.stage animal.sex attachment.type  
## 1 2015-09-01 04:37:25.000 adult m harness  
## 2 2014-06-18 22:02:41.000 adult m harness  
## 3 2015-09-01 09:52:41.000 adult m harness  
## 4 2015-09-01 03:07:07.000 adult f harness  
## 5 2015-09-01 04:28:52.000 adult f harness  
## 6 2015-09-01 04:52:35.000 adult f harness  
## 7 2013-12-21 15:16:23.000 adult f harness  
## 8 2015-09-01 04:31:54.000 adult f harness  
## 9 2014-09-22 16:35:20.000 adult m harness  
## 10 2015-09-01 04:58:46.000 adult m harness  
## 11 2014-03-21 19:44:56.000 adult f harness  
## 12 2015-09-01 04:46:12.000 adult m harness  
## 13 2015-09-01 02:56:53.000 adult m harness  
## 14 2015-09-01 04:55:41.000 adult m harness  
## 15 2015-09-01 04:21:36.000 adult f harness  
## 16 2015-09-01 04:20:15.000 adult m harness  
## 17 2015-09-01 04:48:56.000 adult m harness  
## 18 2015-09-01 04:54:57.000 adult m harness  
## deployment.end.comments  
## 1 Active at end of study  
## 2 Transmission ceased; bird fate unknown.  
## 3 Active at end of study  
## 4 Vulture deceased  
## 5 Active at end of study  
## 6 Active at end of study  
## 7 Transmission ceased; bird fate unknown  
## 8 Active at end of study  
## 9 Transmitter dropped; bird fate unknown; transmitter redeployed  
## 10 Active at end of study  
## 11 Deceased; transmitter redeployed.  
## 12 Active at end of study  
## 13 Active at end of study  
## 14 Active at end of study  
## 15 Active at end of study  
## 16 Active at end of study  
## 17 Active at end of study  
## 18 Active at end of study  
## deployment.end.type deployment.id manipulation.type tag.manufacturer.name  
## 1 other 171-6 none Microwave  
## 2 unknown 167-8 none Microwave  
## 3 other 162-12 none Microwave  
## 4 dead 165-22 none Microwave  
## 5 other 163-1 none Microwave  
## 6 other 173-3 none Microwave  
## 7 unknown 180-13 none Microwave  
## 8 other 161-47 none Microwave  
## 9 fall-off 177-48 none Microwave  
## 10 other 172-60 none Microwave  
## 11 dead 178-57 none Microwave  
## 12 other 176-75 none Microwave  
## 13 other 166-90 none Microwave  
## 14 other 179-91 none Microwave  
## 15 other 175-92 none Microwave  
## 16 other 168-108 none Microwave  
## 17 other 178-123 none Microwave  
## 18 other 174-126 none Microwave  
## tag.mass tag.readout.method  
## 1 70 phone-network  
## 2 70 phone-network  
## 3 70 phone-network  
## 4 70 phone-network  
## 5 70 phone-network  
## 6 70 phone-network  
## 7 70 phone-network  
## 8 70 phone-network  
## 9 70 phone-network  
## 10 70 phone-network  
## 11 70 phone-network  
## 12 70 phone-network  
## 13 70 phone-network  
## 14 70 phone-network  
## 15 70 phone-network  
## 16 70 phone-network  
## 17 70 phone-network  
## 18 70 phone-network

vulture\_dat <- read.csv("../data/Black-Vultures-and-Turkey-Vultures-Southeastern-USA.csv")  
head(vulture\_dat)

## event.id visible timestamp location.long location.lat  
## 1 3378871917 true 2013-06-18 23:26:49.000 -81.64039 33.15934  
## 2 3378871918 true 2013-06-18 23:50:02.000 -81.64057 33.15920  
## 3 3378871919 true 2013-06-19 00:14:17.000 -81.63882 33.15654  
## 4 3378871920 true 2013-06-19 00:38:31.000 -81.63679 33.15425  
## 5 3378871921 true 2013-06-19 01:02:46.000 -81.63669 33.15413  
## 6 3378871922 true 2013-06-19 01:29:01.000 -81.63675 33.15424  
## gps.hdop gps.satellite.count gps.vdop ground.speed heading  
## 1 3.0 5 2.2 0 0  
## 2 0.9 9 1.4 0 0  
## 3 0.9 9 1.3 0 317  
## 4 1.0 8 1.3 0 0  
## 5 1.4 6 1.8 0 0  
## 6 1.2 6 1.6 0 0  
## height.above.ellipsoid manually.marked.outlier sensor.type  
## 1 80 NA gps  
## 2 79 NA gps  
## 3 73 NA gps  
## 4 72 NA gps  
## 5 76 NA gps  
## 6 48 NA gps  
## individual.taxon.canonical.name tag.local.identifier  
## 1 Coragyps atratus 167  
## 2 Coragyps atratus 167  
## 3 Coragyps atratus 167  
## 4 Coragyps atratus 167  
## 5 Coragyps atratus 167  
## 6 Coragyps atratus 167  
## individual.local.identifier  
## 1 8  
## 2 8  
## 3 8  
## 4 8  
## 5 8  
## 6 8  
## study.name  
## 1 Black Vultures and Turkey Vultures Southeastern USA  
## 2 Black Vultures and Turkey Vultures Southeastern USA  
## 3 Black Vultures and Turkey Vultures Southeastern USA  
## 4 Black Vultures and Turkey Vultures Southeastern USA  
## 5 Black Vultures and Turkey Vultures Southeastern USA  
## 6 Black Vultures and Turkey Vultures Southeastern USA

str(vulture\_dat)

## 'data.frame': 2605997 obs. of 17 variables:  
## $ event.id : num 3.38e+09 3.38e+09 3.38e+09 3.38e+09 3.38e+09 ...  
## $ visible : chr "true" "true" "true" "true" ...  
## $ timestamp : chr "2013-06-18 23:26:49.000" "2013-06-18 23:50:02.000" "2013-06-19 00:14:17.000" "2013-06-19 00:38:31.000" ...  
## $ location.long : num -81.6 -81.6 -81.6 -81.6 -81.6 ...  
## $ location.lat : num 33.2 33.2 33.2 33.2 33.2 ...  
## $ gps.hdop : num 3 0.9 0.9 1 1.4 1.2 1.2 1.5 1.5 2.2 ...  
## $ gps.satellite.count : int 5 9 9 8 6 6 7 7 7 5 ...  
## $ gps.vdop : num 2.2 1.4 1.3 1.3 1.8 1.6 1.7 2.3 1.7 2.2 ...  
## $ ground.speed : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ heading : num 0 0 317 0 0 0 0 158 0 0 ...  
## $ height.above.ellipsoid : num 80 79 73 72 76 48 83 59 66 72 ...  
## $ manually.marked.outlier : logi NA NA NA NA NA NA ...  
## $ sensor.type : chr "gps" "gps" "gps" "gps" ...  
## $ individual.taxon.canonical.name: chr "Coragyps atratus" "Coragyps atratus" "Coragyps atratus" "Coragyps atratus" ...  
## $ tag.local.identifier : int 167 167 167 167 167 167 167 167 167 167 ...  
## $ individual.local.identifier : int 8 8 8 8 8 8 8 8 8 8 ...  
## $ study.name : chr "Black Vultures and Turkey Vultures Southeastern USA" "Black Vultures and Turkey Vultures Southeastern USA" "Black Vultures and Turkey Vultures Southeastern USA" "Black Vultures and Turkey Vultures Southeastern USA" ...

nrow(vulture\_dat)

## [1] 2605997

#vulture\_dat

1. how many individuals examined

# tag id's 17 tags   
  
sort(unique(vulture\_dat$tag.local.identifier))

## [1] 161 162 163 165 166 167 168 171 172 173 174 175 176 177 178 179 180

length(unique(vulture\_dat$tag.local.identifier))

## [1] 17

# individual bird id's 18 birds  
  
sort(unique(vulture\_dat$individual.local.identifier))

## [1] 1 3 6 8 12 13 22 47 48 57 60 75 90 91 92 108 123 126

length(unique(vulture\_dat$individual.local.identifier))

## [1] 18

1. How many datapoints are present for each individual in the timeperiod

bird\_ids <- unique(vulture\_dat$individual.local.identifier)  
  
record\_table <- data.frame()  
  
for(i in 1:length(bird\_ids)){  
 current\_bird <- bird\_ids[i]  
 number\_of\_records <- sum(vulture\_dat$individual.local.identifier == current\_bird)  
 new\_row <- c(current\_bird,number\_of\_records)  
 #colnames(new\_row) <- c("bird id","number of records")  
 record\_table <- rbind(record\_table, new\_row)  
}  
  
colnames(record\_table) <- c("bird id","number of records")  
  
(record\_table)

## bird id number of records  
## 1 8 97255  
## 2 12 168457  
## 3 22 112667  
## 4 47 113321  
## 5 48 89945  
## 6 57 29717  
## 7 123 135554  
## 8 92 172766  
## 9 108 87646  
## 10 126 123437  
## 11 1 237897  
## 12 3 243371  
## 13 6 198840  
## 14 13 38560  
## 15 60 210787  
## 16 75 188019  
## 17 90 170442  
## 18 91 187316

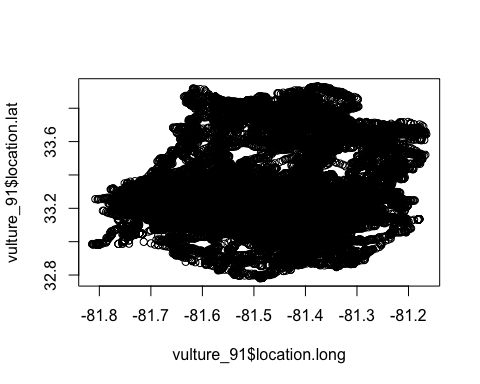
1. When does each record start and stop?

bird\_ids <- unique(vulture\_dat$individual.local.identifier)  
  
start\_stop\_table <- data.frame()  
  
for(i in 1:length(bird\_ids)){  
 current\_bird <- bird\_ids[i]  
 curr\_records <- vulture\_dat[vulture\_dat$individual.local.identifier == current\_bird,]  
 start\_rec <- head(curr\_records, n=1)$timestamp  
 stop\_rec <- tail(curr\_records, n=1)$timestamp  
 n\_days <- round(as.numeric(difftime(ymd\_hms(stop\_rec), ymd\_hms(start\_rec),units = "days")),3)  
 new\_row <- c(current\_bird,start\_rec, stop\_rec, n\_days)  
   
 start\_stop\_table <- rbind(start\_stop\_table, new\_row)  
}  
  
colnames(start\_stop\_table) <- c("bird id","start","stop","n days")  
  
(start\_stop\_table)

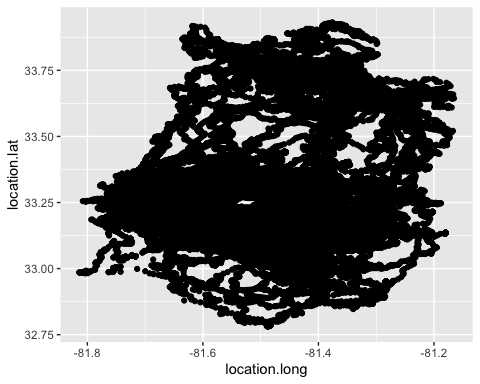
## bird id start stop n days  
## 1 8 2013-06-18 23:26:49.000 2014-06-18 22:02:41.000 364.942  
## 2 12 2013-06-20 16:49:51.000 2015-09-01 04:52:41.000 802.502  
## 3 22 2013-06-20 17:37:38.000 2015-09-01 03:07:07.000 802.395  
## 4 47 2013-07-03 16:06:48.000 2015-09-01 04:31:54.000 789.517  
## 5 48 2013-07-03 19:10:03.000 2014-09-22 16:35:20.000 445.893  
## 6 57 2013-07-05 16:43:23.000 2014-03-21 19:44:56.000 259.126  
## 7 123 2014-04-23 19:44:16.000 2015-09-01 04:48:56.000 495.378  
## 8 92 2013-08-02 12:47:47.000 2015-09-01 04:21:36.000 759.648  
## 9 108 2014-04-21 18:40:25.000 2015-09-01 04:20:15.000 497.403  
## 10 126 2014-05-01 13:38:20.000 2015-09-01 04:54:57.000 487.637  
## 11 1 2013-06-26 14:07:33.000 2015-09-01 04:28:52.000 796.598  
## 12 3 2013-06-26 14:38:58.000 2015-09-01 04:52:35.000 796.593  
## 13 6 2013-06-17 13:19:20.000 2015-09-01 04:37:25.000 805.638  
## 14 13 2013-06-27 18:49:55.000 2013-12-21 15:16:23.000 176.852  
## 15 60 2013-07-05 12:52:58.000 2015-09-01 04:58:46.000 787.671  
## 16 75 2013-07-09 15:31:45.000 2015-09-01 04:46:12.000 783.552  
## 17 90 2013-07-30 14:34:17.000 2015-09-01 02:56:53.000 762.516  
## 18 91 2013-07-31 15:52:51.000 2015-09-01 04:55:41.000 761.544

1. Simple plot of the data points for one individual

#select bird 91  
  
id <- 91  
  
#subset whole dataset  
  
vulture\_91 <- vulture\_dat[vulture\_dat$individual.local.identifier == 91,]  
  
plot(vulture\_91$location.long, vulture\_91$location.lat)



ggplot(vulture\_91, aes(x=location.long, y=location.lat))+ geom\_point()



1. Estimate home range for one individual using three methods of your own choice.
2. Choose Individual and generate Tracks

vulture\_91 <- vulture\_dat %>%  
 filter(individual.local.identifier == 91)  
  
head(vulture\_91)

## event.id visible timestamp location.long location.lat  
## 1 3383407128 true 2013-07-31 15:52:51.000 -81.64166 33.16277  
## 2 3383365434 true 2013-07-31 15:54:06.000 -81.64163 33.16284  
## 3 3383282583 true 2013-07-31 15:55:20.000 -81.64163 33.16282  
## 4 3383407129 true 2013-07-31 15:56:28.000 -81.64168 33.16279  
## 5 3383282584 true 2013-07-31 15:58:21.000 -81.64163 33.16279  
## 6 3383282585 true 2013-07-31 16:00:19.000 -81.64157 33.16281  
## gps.hdop gps.satellite.count gps.vdop ground.speed heading  
## 1 0.9 9 1.4 0 0  
## 2 0.9 9 1.4 0 0  
## 3 0.9 9 1.4 0 0  
## 4 0.9 9 1.4 0 58  
## 5 0.9 9 1.4 0 0  
## 6 0.9 9 1.4 0 236  
## height.above.ellipsoid manually.marked.outlier sensor.type  
## 1 53 NA gps  
## 2 66 NA gps  
## 3 59 NA gps  
## 4 74 NA gps  
## 5 77 NA gps  
## 6 75 NA gps  
## individual.taxon.canonical.name tag.local.identifier  
## 1 Cathartes aura 179  
## 2 Cathartes aura 179  
## 3 Cathartes aura 179  
## 4 Cathartes aura 179  
## 5 Cathartes aura 179  
## 6 Cathartes aura 179  
## individual.local.identifier  
## 1 91  
## 2 91  
## 3 91  
## 4 91  
## 5 91  
## 6 91  
## study.name  
## 1 Black Vultures and Turkey Vultures Southeastern USA  
## 2 Black Vultures and Turkey Vultures Southeastern USA  
## 3 Black Vultures and Turkey Vultures Southeastern USA  
## 4 Black Vultures and Turkey Vultures Southeastern USA  
## 5 Black Vultures and Turkey Vultures Southeastern USA  
## 6 Black Vultures and Turkey Vultures Southeastern USA

# check timestamp  
class(vulture\_91$timestamp)

## [1] "character"

# convert timestamp to posixct format  
vulture\_91$timestamp <- ymd\_hms(vulture\_91$timestamp, tz = "UTC")  
head(vulture\_91$timestamp)

## [1] "2013-07-31 15:52:51 UTC" "2013-07-31 15:54:06 UTC"  
## [3] "2013-07-31 15:55:20 UTC" "2013-07-31 15:56:28 UTC"  
## [5] "2013-07-31 15:58:21 UTC" "2013-07-31 16:00:19 UTC"

str(vulture\_91$timestamp)

## POSIXct[1:187316], format: "2013-07-31 15:52:51" "2013-07-31 15:54:06" "2013-07-31 15:55:20" ...

# make track for vulture 91  
trk\_91 <- make\_track(vulture\_91, location.long,location.lat, timestamp, id = individual.local.identifier, crs = 4326)  
  
# save this file to disk  
saveRDS(trk\_91, file = "../output/vulture\_91\_gps\_data\_track.rds")  
  
# check sampling rate for bird 91  
summarize\_sampling\_rate(trk\_91)

## # A tibble: 1 × 9  
## min q1 median mean q3 max sd n unit   
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <int> <chr>  
## 1 0.55 1 1.03 5.85 1.98 1492. 14.4 187315 min

# This suggests that the median sampling rate is 2h, but varying up to  
# 12h. We can now resample the whole track to 2h interval (with tolerance of  
# 10 min), so that if there are more than 2h between relocations, they will   
# be divided into different bursts.  
  
trk\_91\_resamp <- track\_resample(trk\_91, rate = minutes(1), tolerance = minutes(1490))  
  
# add step length as a new col  
trk\_91\_sl <- trk\_91\_resamp %>% mutate(sl = step\_lengths(.))   
  
# calculate steps by burst  
trk\_91\_sbb <- trk\_91\_resamp %>% steps\_by\_burst()

1. MCP Home Range

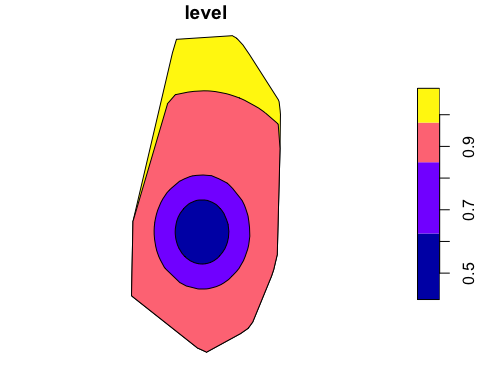
mcps <- hr\_mcp(trk\_91, levels = c(0.5, 0.75, 0.95, 1))  
mcps

## $mcp  
## Simple feature collection with 4 features and 3 fields  
## Geometry type: POLYGON  
## Dimension: XY  
## Bounding box: xmin: -81.81408 ymin: 32.77979 xmax: -81.16542 ymax: 33.9309  
## Geodetic CRS: WGS 84  
## level what area geometry  
## 1 0.50 estimate 440882403 [m^2] POLYGON ((-81.39618 33.1804...  
## 2 0.75 estimate 1401770831 [m^2] POLYGON ((-81.29887 33.2017...  
## 3 0.95 estimate 5006729740 [m^2] POLYGON ((-81.17934 33.1347...  
## 4 1.00 estimate 5983974085 [m^2] POLYGON ((-81.17934 33.1347...  
##   
## $levels  
## [1] 0.50 0.75 0.95 1.00  
##   
## $estimator  
## [1] "mcp"  
##   
## $crs  
## Coordinate Reference System:  
## User input: EPSG:4326   
## wkt:  
## GEOGCRS["WGS 84",  
## ENSEMBLE["World Geodetic System 1984 ensemble",  
## MEMBER["World Geodetic System 1984 (Transit)"],  
## MEMBER["World Geodetic System 1984 (G730)"],  
## MEMBER["World Geodetic System 1984 (G873)"],  
## MEMBER["World Geodetic System 1984 (G1150)"],  
## MEMBER["World Geodetic System 1984 (G1674)"],  
## MEMBER["World Geodetic System 1984 (G1762)"],  
## MEMBER["World Geodetic System 1984 (G2139)"],  
## ELLIPSOID["WGS 84",6378137,298.257223563,  
## LENGTHUNIT["metre",1]],  
## ENSEMBLEACCURACY[2.0]],  
## PRIMEM["Greenwich",0,  
## ANGLEUNIT["degree",0.0174532925199433]],  
## CS[ellipsoidal,2],  
## AXIS["geodetic latitude (Lat)",north,  
## ORDER[1],  
## ANGLEUNIT["degree",0.0174532925199433]],  
## AXIS["geodetic longitude (Lon)",east,  
## ORDER[2],  
## ANGLEUNIT["degree",0.0174532925199433]],  
## USAGE[  
## SCOPE["Horizontal component of 3D system."],  
## AREA["World."],  
## BBOX[-90,-180,90,180]],  
## ID["EPSG",4326]]  
##   
## $data  
## # A tibble: 187,316 × 4  
## x\_ y\_ t\_ id  
## \* <dbl> <dbl> <dttm> <int>  
## 1 -81.6 33.2 2013-07-31 15:52:51 91  
## 2 -81.6 33.2 2013-07-31 15:54:06 91  
## 3 -81.6 33.2 2013-07-31 15:55:20 91  
## 4 -81.6 33.2 2013-07-31 15:56:28 91  
## 5 -81.6 33.2 2013-07-31 15:58:21 91  
## 6 -81.6 33.2 2013-07-31 16:00:19 91  
## 7 -81.6 33.2 2013-07-31 16:02:13 91  
## 8 -81.6 33.2 2013-07-31 16:04:07 91  
## 9 -81.6 33.2 2013-07-31 16:06:50 91  
## 10 -81.6 33.2 2013-07-31 16:08:45 91  
## # ℹ 187,306 more rows  
##   
## attr(,"class")  
## [1] "mcp" "hr\_geom" "hr"

# Get area in km^2  
hr\_area(mcps, units = TRUE) %>%   
 mutate(area = units::set\_units(area, "km^2"))

## # A tibble: 4 × 3  
## level what area  
## <dbl> <chr> [km^2]  
## 1 1 estimate 5984.  
## 2 0.95 estimate 5007.  
## 3 0.75 estimate 1402.  
## 4 0.5 estimate 441.

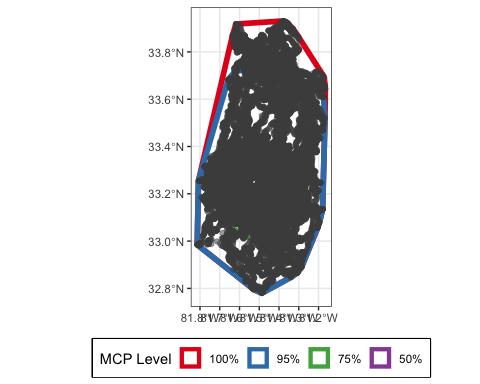
# plot the polygon isopleths by themeself  
plot(hr\_isopleths(mcps)[1])



# Plot the isopleth with data on top  
plot(mcps)



# Custom plot with ggplot2 (thanks to Brian Smith for this code chunk)  
  
hr\_isopleths(mcps) %>%   
 # Make level a factor for discrete color scales  
 # Can control order and labels here  
 mutate(level = factor(level,   
 levels = c("1", "0.95", "0.75", "0.5"),  
 labels = c("100%", "95%", "75%", "50%"))) %>%   
 ggplot() +  
 geom\_sf(aes(color = level),   
 fill = NA, linewidth = 2) +  
 geom\_point(data = mcps$data, aes(x = x\_, y = y\_),  
 color = "gray30", alpha = 0.5) +  
 xlab(NULL) +  
 ylab(NULL) +  
 scale\_color\_brewer(name = "MCP Level",  
 palette = "Set1") +  
 theme\_bw() +  
 theme(legend.position = "bottom",   
 legend.box.background = element\_rect(colour = "black", linewidth = 0.8))

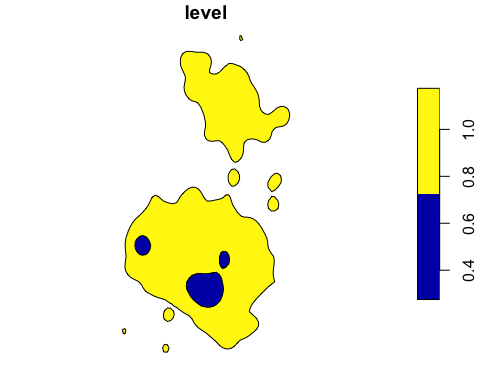


1. Kernel density estimation of Home Range

kdes <- hr\_kde(trk\_91, levels = c(0.5, 0.95),)  
  
# Get area in km^2  
hr\_area(kdes, units = TRUE) %>%   
 mutate(area = units::set\_units(area, "km^2"))

## # A tibble: 2 × 3  
## level what area  
## <dbl> <chr> [km^2]  
## 1 0.95 estimate 2689.  
## 2 0.5 estimate 169.

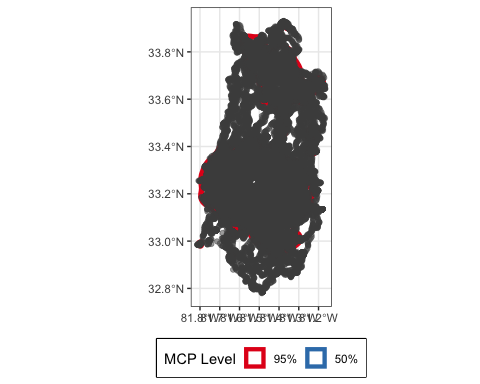
# plot the polygon isopleths by themeself  
plot(hr\_isopleths(kdes)[1])



# Plot the isopleth with data on top  
plot(kdes)



# Custom plot with ggplot2 (thanks to Brian Smith for this code chunk)  
  
hr\_isopleths(kdes) %>%   
 # Make level a factor for discrete color scales  
 # Can control order and labels here  
 mutate(level = factor(level,   
 levels = c("1", "0.95", "0.75", "0.5"),  
 labels = c("100%", "95%", "75%", "50%"))) %>%   
 ggplot() +  
 geom\_sf(aes(color = level),   
 fill = NA, linewidth = 2) +  
 geom\_point(data = mcps$data, aes(x = x\_, y = y\_),  
 color = "gray30", alpha = 0.5) +  
 xlab(NULL) +  
 ylab(NULL) +  
 scale\_color\_brewer(name = "MCP Level",  
 palette = "Set1") +  
 theme\_bw() +  
 theme(legend.position = "bottom",   
 legend.box.background = element\_rect(colour = "black", linewidth = 0.8))



hr\_overlap(mcps,kdes)

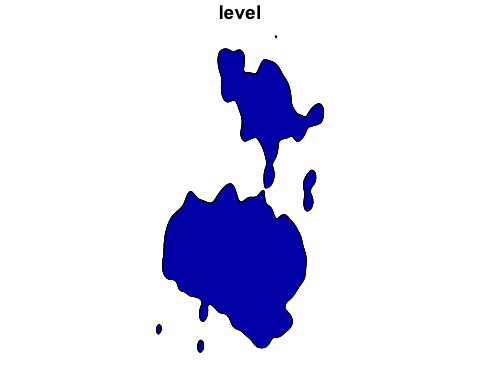
## # A tibble: 4 × 2  
## levels overlap  
## <dbl> <dbl>  
## 1 1 0.449   
## 2 0.95 0.0337  
## 3 0.75 0   
## 4 0.5 0

1. Autocorrelated Kernel Density estimator of home range

# Without going into detail on the different CTMMs, we'll demonstrate fitting  
# an aKDE with an Ornstein-Uhlenbeck model.  
akdes <- hr\_akde(trk\_91, model = fit\_ctmm(trk\_91, "iid"), levels = 0.95)  
  
# Get area in km^2  
hr\_area(akdes, units = TRUE) %>%   
 mutate(area = units::set\_units(area, "km^2"))

## # A tibble: 3 × 3  
## level what area  
## <dbl> <chr> [km^2]  
## 1 0.95 lci (0.95) 2640.  
## 2 0.95 estimate 2651.  
## 3 0.95 uci (0.95) 2662.

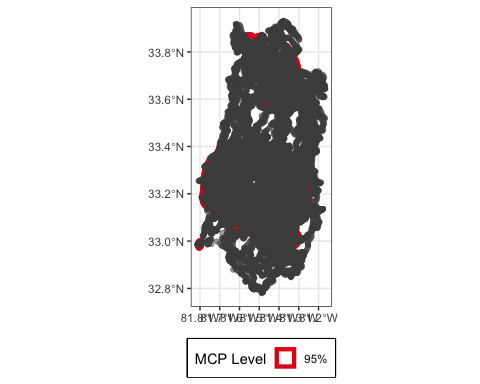
# plot the polygon isopleths by themeself  
plot(hr\_isopleths(akdes)[1])



# Plot the isopleth with data on top  
plot(akdes)



# Custom plot with ggplot2 (thanks to Brian Smith for this code chunk)  
  
hr\_isopleths(akdes) %>%   
 # Make level a factor for discrete color scales  
 # Can control order and labels here  
 mutate(level = factor(level,   
 levels = c("1", "0.95", "0.75", "0.5"),  
 labels = c("100%", "95%", "75%", "50%"))) %>%   
 ggplot() +  
 geom\_sf(aes(color = level),   
 fill = NA, linewidth = 2) +  
 geom\_point(data = mcps$data, aes(x = x\_, y = y\_),  
 color = "gray30", alpha = 0.5) +  
 xlab(NULL) +  
 ylab(NULL) +  
 scale\_color\_brewer(name = "MCP Level",  
 palette = "Set1") +  
 theme\_bw() +  
 theme(legend.position = "bottom",   
 legend.box.background = element\_rect(colour = "black", linewidth = 0.8))



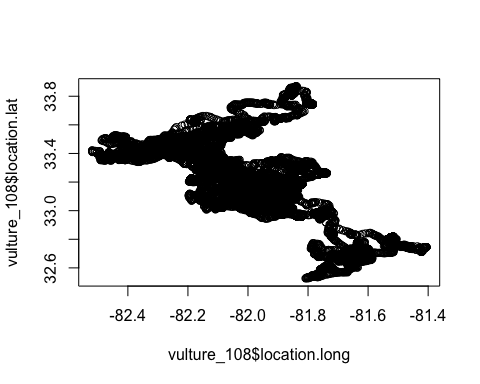
hr\_overlap(kdes,akdes)

## # A tibble: 2 × 2  
## levels overlap  
## <dbl> <dbl>  
## 1 0.95 0.959  
## 2 0.5 1

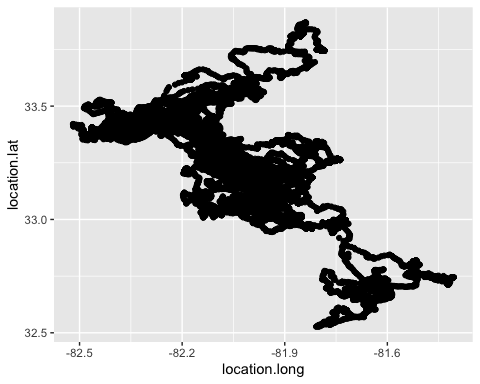
### Examine Bird 108

1. Simple plot of the data points for one individual

#select bird 108  
  
id <- 108  
  
#subset whole dataset  
  
vulture\_108 <- vulture\_dat[vulture\_dat$individual.local.identifier == 108,]  
  
plot(vulture\_108$location.long, vulture\_108$location.lat)



ggplot(vulture\_108, aes(x=location.long, y=location.lat))+ geom\_point()



1. Estimate home range for one individual using three methods of your own choice.
2. Choose Individual and generate Tracks

vulture\_108 <- vulture\_dat %>%  
 filter(individual.local.identifier == 108)  
  
head(vulture\_108)

## event.id visible timestamp location.long location.lat  
## 1 3381246783 true 2014-04-21 18:40:25.000 -81.73844 33.26194  
## 2 3381246784 true 2014-04-21 18:52:55.000 -81.73833 33.26218  
## 3 3381246785 true 2014-04-21 19:05:17.000 -81.73848 33.26196  
## 4 3381246786 true 2014-04-21 19:18:03.000 -81.73826 33.26196  
## 5 3381246787 true 2014-04-21 19:31:25.000 -81.73846 33.26190  
## 6 3381246788 true 2014-04-21 19:44:51.000 -81.73858 33.26191  
## gps.hdop gps.satellite.count gps.vdop ground.speed heading  
## 1 1.7 6 2.4 0 0  
## 2 1.9 5 2.4 0 263  
## 3 1.6 6 2.2 0 0  
## 4 2.1 6 3.3 0 0  
## 5 2.9 6 4.9 0 0  
## 6 5.6 6 9.6 0 59  
## height.above.ellipsoid manually.marked.outlier sensor.type  
## 1 29 NA gps  
## 2 60 NA gps  
## 3 63 NA gps  
## 4 61 NA gps  
## 5 77 NA gps  
## 6 77 NA gps  
## individual.taxon.canonical.name tag.local.identifier  
## 1 Coragyps atratus 168  
## 2 Coragyps atratus 168  
## 3 Coragyps atratus 168  
## 4 Coragyps atratus 168  
## 5 Coragyps atratus 168  
## 6 Coragyps atratus 168  
## individual.local.identifier  
## 1 108  
## 2 108  
## 3 108  
## 4 108  
## 5 108  
## 6 108  
## study.name  
## 1 Black Vultures and Turkey Vultures Southeastern USA  
## 2 Black Vultures and Turkey Vultures Southeastern USA  
## 3 Black Vultures and Turkey Vultures Southeastern USA  
## 4 Black Vultures and Turkey Vultures Southeastern USA  
## 5 Black Vultures and Turkey Vultures Southeastern USA  
## 6 Black Vultures and Turkey Vultures Southeastern USA

# check timestamp  
class(vulture\_108$timestamp)

## [1] "character"

# convert timestamp to posixct format  
vulture\_108$timestamp <- ymd\_hms(vulture\_108$timestamp, tz = "UTC")  
head(vulture\_108$timestamp)

## [1] "2014-04-21 18:40:25 UTC" "2014-04-21 18:52:55 UTC"  
## [3] "2014-04-21 19:05:17 UTC" "2014-04-21 19:18:03 UTC"  
## [5] "2014-04-21 19:31:25 UTC" "2014-04-21 19:44:51 UTC"

str(vulture\_108$timestamp)

## POSIXct[1:87646], format: "2014-04-21 18:40:25" "2014-04-21 18:52:55" "2014-04-21 19:05:17" ...

# make track for vulture 91  
trk\_108 <- make\_track(vulture\_108, location.long,location.lat, timestamp, id = individual.local.identifier, crs = 4326)  
  
# save this file to disk  
saveRDS(trk\_108, file = "../output/vulture\_108\_gps\_data\_track.rds")  
  
# check sampling rate for bird 108  
summarize\_sampling\_rate(trk\_108)

## # A tibble: 1 × 9  
## min q1 median mean q3 max sd n unit   
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <int> <chr>  
## 1 0.667 1.02 1.77 8.17 6.28 300. 15.8 87645 min

# This suggests that the median sampling rate is 2h, but varying up to  
# 12h. We can now resample the whole track to 2h interval (with tolerance of  
# 10 min), so that if there are more than 2h between relocations, they will   
# be divided into different bursts.  
  
trk\_108\_resamp <- track\_resample(trk\_108, rate = minutes(2), tolerance = minutes(300))  
  
# add step length as a new col  
trk\_108\_sl <- trk\_108\_resamp %>% mutate(sl = step\_lengths(.))   
  
# calculate steps by burst  
trk\_108\_sbb <- trk\_108\_resamp %>% steps\_by\_burst()

1. MCP Home Range

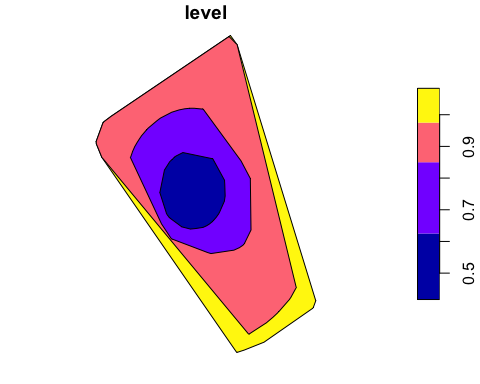
mcps <- hr\_mcp(trk\_108, levels = c(0.5, 0.75, 0.95, 1))  
mcps

## $mcp  
## Simple feature collection with 4 features and 3 fields  
## Geometry type: POLYGON  
## Dimension: XY  
## Bounding box: xmin: -82.51917 ymin: 32.52645 xmax: -81.40612 ymax: 33.86881  
## Geodetic CRS: WGS 84  
## level what area geometry  
## 1 0.50 estimate 858545605 [m^2] POLYGON ((-81.8695 33.16665...  
## 2 0.75 estimate 2743379417 [m^2] POLYGON ((-81.73823 33.2636...  
## 3 0.95 estimate 6871757454 [m^2] POLYGON ((-81.50507 32.8004...  
## 4 1.00 estimate 8166896342 [m^2] POLYGON ((-81.40612 32.7443...  
##   
## $levels  
## [1] 0.50 0.75 0.95 1.00  
##   
## $estimator  
## [1] "mcp"  
##   
## $crs  
## Coordinate Reference System:  
## User input: EPSG:4326   
## wkt:  
## GEOGCRS["WGS 84",  
## ENSEMBLE["World Geodetic System 1984 ensemble",  
## MEMBER["World Geodetic System 1984 (Transit)"],  
## MEMBER["World Geodetic System 1984 (G730)"],  
## MEMBER["World Geodetic System 1984 (G873)"],  
## MEMBER["World Geodetic System 1984 (G1150)"],  
## MEMBER["World Geodetic System 1984 (G1674)"],  
## MEMBER["World Geodetic System 1984 (G1762)"],  
## MEMBER["World Geodetic System 1984 (G2139)"],  
## ELLIPSOID["WGS 84",6378137,298.257223563,  
## LENGTHUNIT["metre",1]],  
## ENSEMBLEACCURACY[2.0]],  
## PRIMEM["Greenwich",0,  
## ANGLEUNIT["degree",0.0174532925199433]],  
## CS[ellipsoidal,2],  
## AXIS["geodetic latitude (Lat)",north,  
## ORDER[1],  
## ANGLEUNIT["degree",0.0174532925199433]],  
## AXIS["geodetic longitude (Lon)",east,  
## ORDER[2],  
## ANGLEUNIT["degree",0.0174532925199433]],  
## USAGE[  
## SCOPE["Horizontal component of 3D system."],  
## AREA["World."],  
## BBOX[-90,-180,90,180]],  
## ID["EPSG",4326]]  
##   
## $data  
## # A tibble: 87,646 × 4  
## x\_ y\_ t\_ id  
## \* <dbl> <dbl> <dttm> <int>  
## 1 -81.7 33.3 2014-04-21 18:40:25.000000 108  
## 2 -81.7 33.3 2014-04-21 18:52:55.000000 108  
## 3 -81.7 33.3 2014-04-21 19:05:17.000000 108  
## 4 -81.7 33.3 2014-04-21 19:18:03.000000 108  
## 5 -81.7 33.3 2014-04-21 19:31:25.000000 108  
## 6 -81.7 33.3 2014-04-21 19:44:51.000000 108  
## 7 -81.7 33.3 2014-04-21 19:53:19.000000 108  
## 8 -81.7 33.3 2014-04-21 19:58:48.000000 108  
## 9 -81.7 33.3 2014-04-21 20:03:08.000000 108  
## 10 -81.7 33.3 2014-04-21 20:09:16.000000 108  
## # ℹ 87,636 more rows  
##   
## attr(,"class")  
## [1] "mcp" "hr\_geom" "hr"

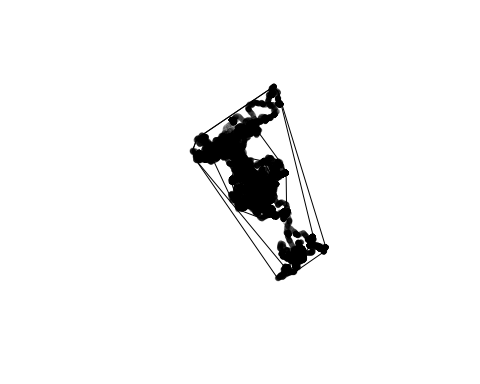
# Get area in km^2  
hr\_area(mcps, units = TRUE) %>%   
 mutate(area = units::set\_units(area, "km^2"))

## # A tibble: 4 × 3  
## level what area  
## <dbl> <chr> [km^2]  
## 1 1 estimate 8167.  
## 2 0.95 estimate 6872.  
## 3 0.75 estimate 2743.  
## 4 0.5 estimate 859.

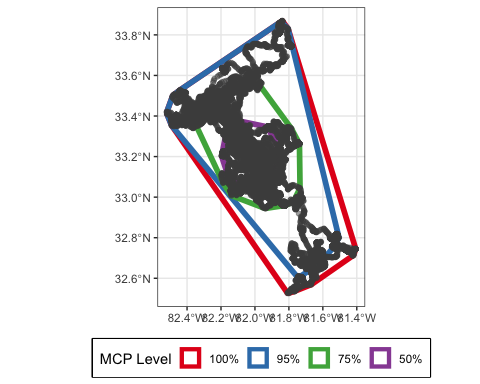
# plot the polygon isopleths by themeself  
plot(hr\_isopleths(mcps)[1])



# Plot the isopleth with data on top  
plot(mcps)



# Custom plot with ggplot2 (thanks to Brian Smith for this code chunk)  
  
hr\_isopleths(mcps) %>%   
 # Make level a factor for discrete color scales  
 # Can control order and labels here  
 mutate(level = factor(level,   
 levels = c("1", "0.95", "0.75", "0.5"),  
 labels = c("100%", "95%", "75%", "50%"))) %>%   
 ggplot() +  
 geom\_sf(aes(color = level),   
 fill = NA, linewidth = 2) +  
 geom\_point(data = mcps$data, aes(x = x\_, y = y\_),  
 color = "gray30", alpha = 0.5) +  
 xlab(NULL) +  
 ylab(NULL) +  
 scale\_color\_brewer(name = "MCP Level",  
 palette = "Set1") +  
 theme\_bw() +  
 theme(legend.position = "bottom",   
 legend.box.background = element\_rect(colour = "black", linewidth = 0.8))

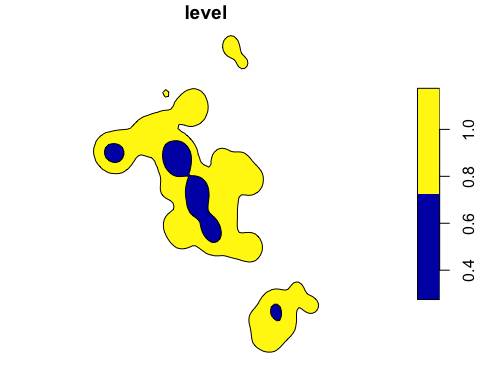


1. Kernel density estimation of Home Range

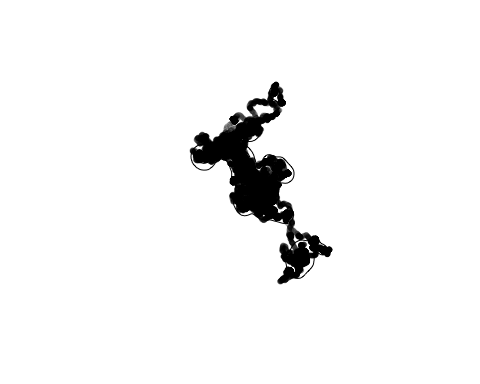
kdes <- hr\_kde(trk\_108, levels = c(0.5, 0.95),)  
  
# Get area in km^2  
hr\_area(kdes, units = TRUE) %>%   
 mutate(area = units::set\_units(area, "km^2"))

## # A tibble: 2 × 3  
## level what area  
## <dbl> <chr> [km^2]  
## 1 0.95 estimate 3506.  
## 2 0.5 estimate 537.

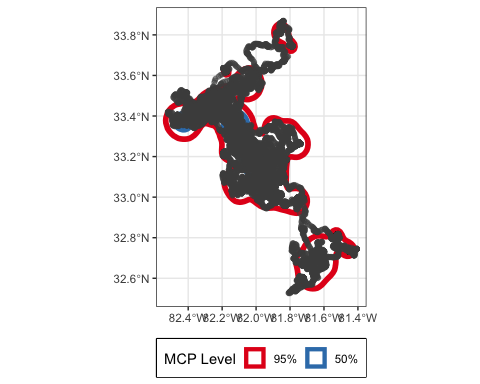
# plot the polygon isopleths by themeself  
plot(hr\_isopleths(kdes)[1])



# Plot the isopleth with data on top  
plot(kdes)



# Custom plot with ggplot2 (thanks to Brian Smith for this code chunk)  
  
hr\_isopleths(kdes) %>%   
 # Make level a factor for discrete color scales  
 # Can control order and labels here  
 mutate(level = factor(level,   
 levels = c("1", "0.95", "0.75", "0.5"),  
 labels = c("100%", "95%", "75%", "50%"))) %>%   
 ggplot() +  
 geom\_sf(aes(color = level),   
 fill = NA, linewidth = 2) +  
 geom\_point(data = mcps$data, aes(x = x\_, y = y\_),  
 color = "gray30", alpha = 0.5) +  
 xlab(NULL) +  
 ylab(NULL) +  
 scale\_color\_brewer(name = "MCP Level",  
 palette = "Set1") +  
 theme\_bw() +  
 theme(legend.position = "bottom",   
 legend.box.background = element\_rect(colour = "black", linewidth = 0.8))



hr\_overlap(mcps,kdes)

## # A tibble: 4 × 2  
## levels overlap  
## <dbl> <dbl>  
## 1 1 0.419   
## 2 0.95 0.0782  
## 3 0.75 0   
## 4 0.5 0

1. Autocorrelated Kernel Density estimator of home range

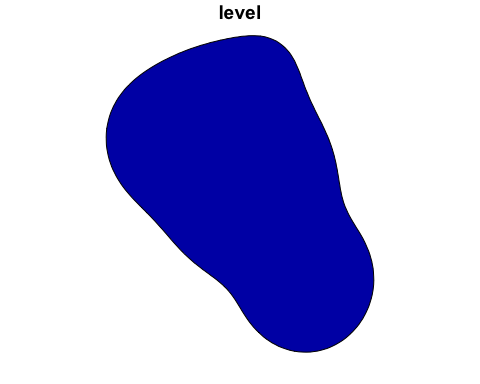
# Without going into detail on the different CTMMs, we'll demonstrate fitting  
# an aKDE with an Ornstein-Uhlenbeck model.  
akdes <- hr\_akde(trk\_108, model = fit\_ctmm(trk\_108, "ou"), levels = 0.95)

## Default grid size of 35.3333333333333 seconds chosen for bandwidth(...,fast=TRUE).

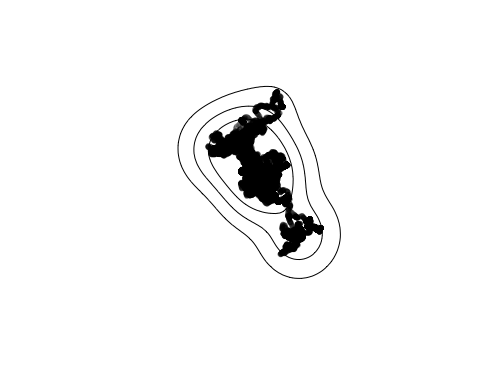
# Get area in km^2  
hr\_area(akdes, units = TRUE) %>%   
 mutate(area = units::set\_units(area, "km^2"))

## # A tibble: 3 × 3  
## level what area  
## <dbl> <chr> [km^2]  
## 1 0.95 lci (0.95) 4861.  
## 2 0.95 estimate 9858.  
## 3 0.95 uci (0.95) 16588.

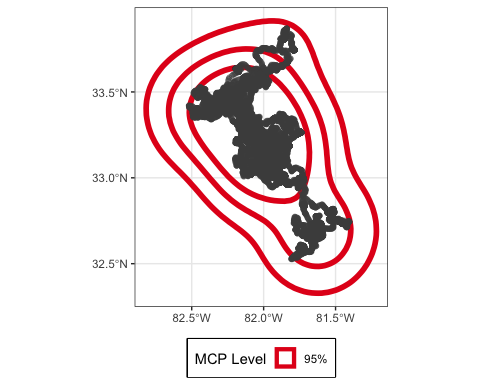
# plot the polygon isopleths by themeself  
plot(hr\_isopleths(akdes)[1])



# Plot the isopleth with data on top  
plot(akdes)



# Custom plot with ggplot2 (thanks to Brian Smith for this code chunk)  
  
hr\_isopleths(akdes) %>%   
 # Make level a factor for discrete color scales  
 # Can control order and labels here  
 mutate(level = factor(level,   
 levels = c("1", "0.95", "0.75", "0.5"),  
 labels = c("100%", "95%", "75%", "50%"))) %>%   
 ggplot() +  
 geom\_sf(aes(color = level),   
 fill = NA, linewidth = 2) +  
 geom\_point(data = mcps$data, aes(x = x\_, y = y\_),  
 color = "gray30", alpha = 0.5) +  
 xlab(NULL) +  
 ylab(NULL) +  
 scale\_color\_brewer(name = "MCP Level",  
 palette = "Set1") +  
 theme\_bw() +  
 theme(legend.position = "bottom",   
 legend.box.background = element\_rect(colour = "black", linewidth = 0.8))



hr\_overlap(kdes, akdes)

## # A tibble: 2 × 2  
## levels overlap  
## <dbl> <dbl>  
## 1 0.95 0.795  
## 2 0.5 1.00

1. Explore the movement of all animals in the data set extracting a continuous covariate (for example elevation or distance to roads) to the data. Extract the covariate both at the end points and along the steps. Explore the differences between the two ways of extracting the data, for example fitting a linear regression with step length as response variable and the extracted variable as explanatory variable.
2. Fit a habitat selection function of your own choice (resource selection function at one scale or an (integrated) step-selection function) to the data using one covariate.