Home range analysis of kākā at Orokonui Ecosanctuary

Scott Forrest

2025-05-19

In this script we are fitting home range models to GPS data of kākā (Nestor meridionalis) at Orokonui Ecosanctuary, New Zealand. We are fitting data across the whole tracking period to estimate their long-term space use.

Load packages and set working directory.

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.2     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
packages <- c("move", "lattice", "purrr", "here", "raster", "sf", "ggpubr",
              "patchwork", "jtools", "MuMIn", "statmod", "ctmm", "terra")

walk(packages, require, character.only = T)
```

```
## Loading required package: move
## Loading required package: geosphere
## Loading required package: sp
## Loading required package: raster
##
## Attaching package: 'raster'
##
## The following object is masked from 'package:dplyr':
##
##     select
##
## Loading required package: lattice
## Loading required package: here
## here() starts at /Users/scottforrest/Library/CloudStorage/OneDrive-QueenslandUniversityofTechnology/
## Loading required package: sf
## Linking to GEOS 3.13.0, GDAL 3.8.5, PROJ 9.5.1; sf_use_s2() is TRUE
## Loading required package: ggpubr
##
## Attaching package: 'ggpubr'
```

```
##
## The following object is masked from 'package:raster':
##
##     rotate
##
## Loading required package: patchwork
##
## Attaching package: 'patchwork'
##
## The following object is masked from 'package:raster':
##
##     area
##
## Loading required package: jtools
## Loading required package: MuMIn
## Registered S3 methods overwritten by 'MuMIn':
##   method        from
##   nobs.multinom broom
##   nobs.fitdistr broom
## Loading required package: statmod
## Loading required package: ctmm
##
## Attaching package: 'ctmm'
##
## The following objects are masked from 'package:move':
##
##     distance, plot, speed
##
## The following objects are masked from 'package:raster':
##
##     distance, head, plot, predict, tail
##
## The following object is masked from 'package:sp':
##
##     plot
##
## The following object is masked from 'package:ggplot2':
##
##     annotate
##
## Loading required package: terra
## terra 1.8.42
##
## Attaching package: 'terra'
##
## The following objects are masked from 'package:ctmm':
##
##     distance, meta
##
## The following object is masked from 'package:patchwork':
##
##     area
##
## The following object is masked from 'package:ggpubr':
```

```
##
##     rotate
##
## The following object is masked from 'package:tidyr':
##
##     extract
##
## The following object is masked from 'package:knitr':
##
##     spin
```

Import data.

```r
# create vector of GPS date filenames
GPSfiles <- list.files("CSV input data - dd_speed_6")

# create a vector for the tag numbers indicating different kākā
id <- 45505:45514

# import data
all_tags_list <- vector(mode = "list", length = length(GPSfiles))

for(i in 1:length(GPSfiles)){
  all_tags_list[[i]] <-  read.csv(paste("CSV input data - dd_speed_6/",
                                        GPSfiles[[i]],
                                        sep = ""))
  all_tags_list[[i]]$id <- id[i]
}
```

Create an extent raster to be used for each individual. It is important to keep these the same size for the dynamic space use incremental analysis, so it is important that the raster covers the extent of each individual's locations.

```r
ext_object <- as(extent(1390000, 1435000, 4910000, 4950000), 'SpatialPolygons')
crs(ext_object) <- "EPSG:2193"
ext_raster <- raster::raster(ext_object, res = 25) # resolution in m
```

Fit home range models

Using the ctmm package to fit continuous-time movement models to the data.

We are mostly following the step-by-step approach presented by Silva et al. (2022), Supplementary Materials 2.

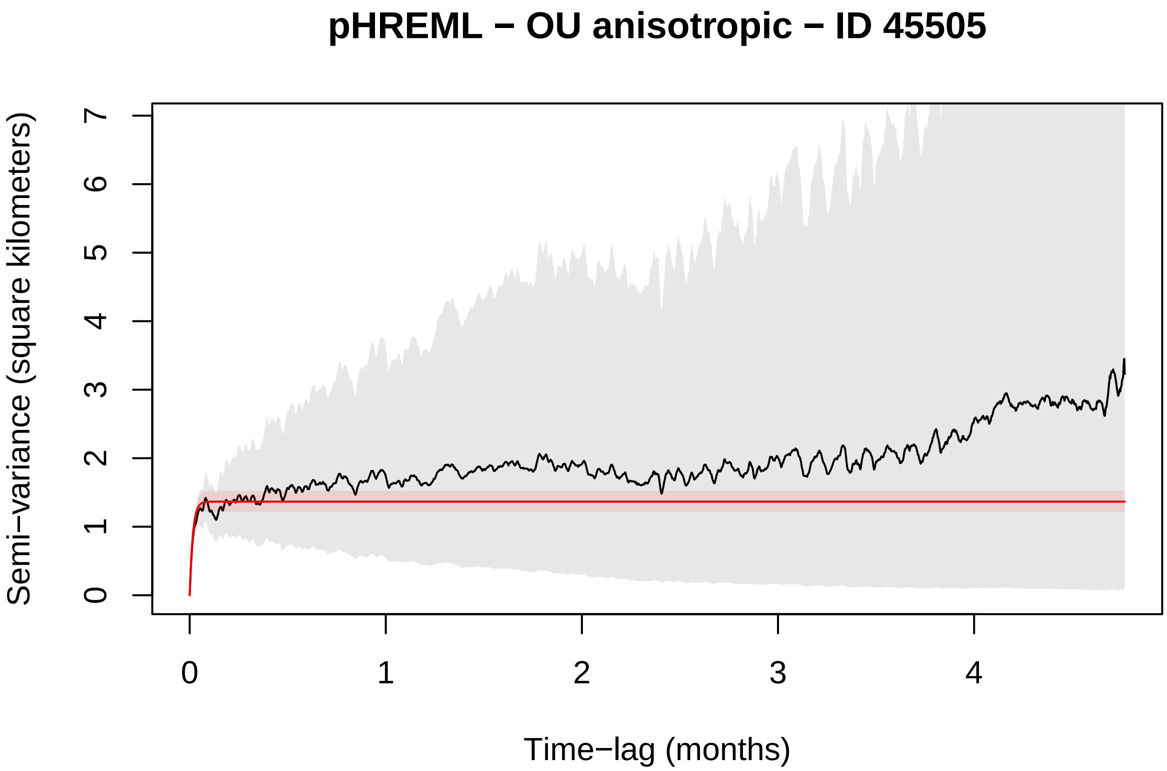
To determine the best model for **each individual**, we start by using the ctmm.guess function. This gets starting values for the ctmm.select function, which is used for model selection. We then fit the best model for each individual using the `akde` function. We do this for each individual by looping over each one. We leave commented out code here to show the diagnostic and model checking steps that we took to ensure that the model fits were suitable for each individual's data, such as trying to fit models with the **pHREML** method, and using the **ML** method. We also check the variogram of the data for each individual.

```r
tag_ctmm_list <- vector(mode = "list", length = 10)
# FIT1_ML_list <- vector(mode = "list", length = 10)
FIT1_pHREML_list <- vector(mode = "list", length = 10)
# UD1w_ML_list <- vector(mode = "list", length = 10)
```

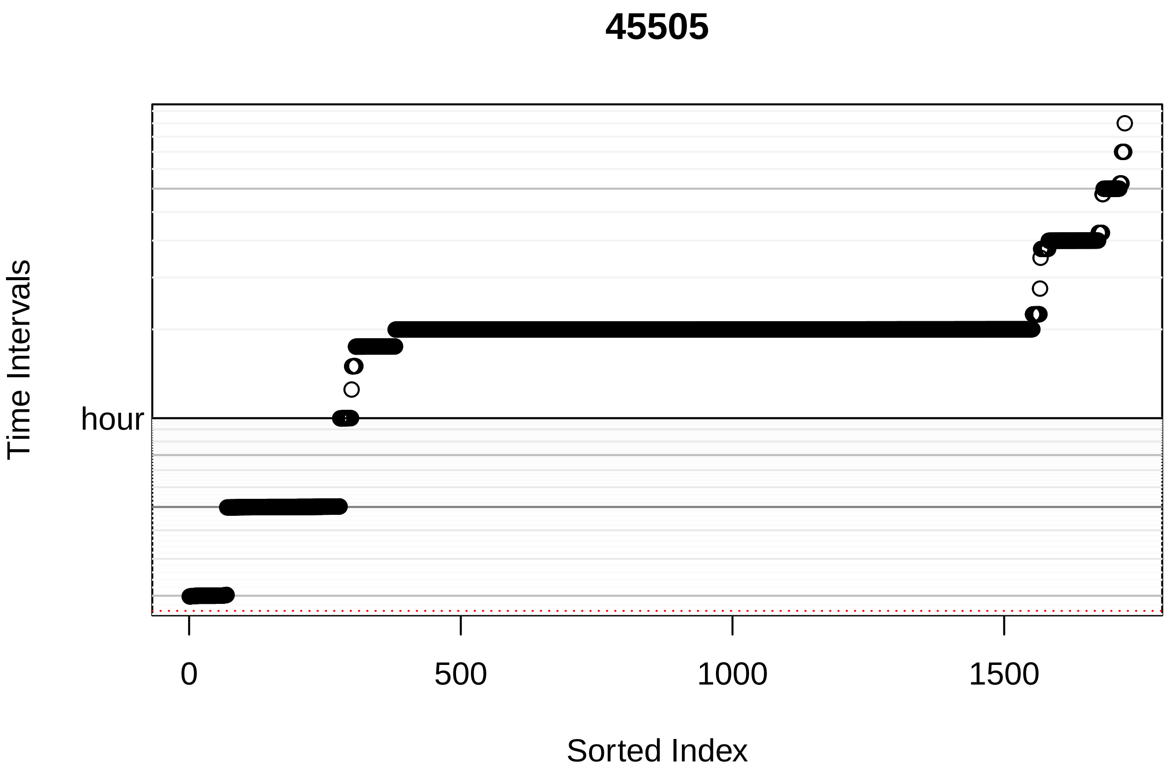
```r
UD1w_pHREML_list <- vector(mode = "list", length = 10)

for(i in 1:10) {

  # create data frames of only necessary information
tag <- all_tags_list[[i]] %>% mutate(ID = id,
                                     timestamp = DateTime,
                                     longitude = lon,
                                     latitude = lat,
                                     .keep = "none")

# reproject
tag_ctmm_list[[i]] <- as.telemetry(tag, timezone = "GMT",
                        projection = paste0("+proj=tmerc +lat_0=0 +lon_0=173 ",
                                            "+k=0.9996 +x_0=1600000 +y_0=10000000 ",
                                            "+ellps=GRS80 +towgs84=0,0,0,0,0,0,0 ",
                                            "+units=m +no_defs +type=crs"))
# plot(tag_ctmm_list[[i]])

level <- 0.95 # we want to display 95% confidence intervals
SVF <- variogram(tag_ctmm_list[[i]])
# plot(SVF, fraction = 1, level = level,
#      main = paste0("id ", tag_ctmm_list[[i]]@info$identity))

# Estimating the ctmm that is most appropriate for the data

# Calculate an automated model guesstimate:
# ctmm.guess(tag_ctmm_list[[i]], interactive = TRUE)
# using interactive starting estimates
# FIT1_ML_list[[i]] <- ctmm.select(tag_ctmm_list[[i]], GUESS,
#                         method = 'ML', IC = "AIC",
#                         verbose = TRUE)


# Calculate an automated model guesstimate:
GUESS1 <- ctmm.guess(tag_ctmm_list[[i]], interactive = FALSE)

# Automated model selection, starting from GUESS:
## reminder: it will default to pHREML if no method is specified.
# FIT1_ML_list[[i]] <- ctmm.select(tag_ctmm_list[[i]], GUESS1,
#                         method = 'ML', IC = "AIC",
#                         verbose = TRUE)

# print(summary(FIT1_ML_list[[i]]))

# plot(SVF, CTMM = FIT1_ML_list[[i]][[1]],
#      main = paste0("ML - ", rownames(summary(FIT1_ML_list[[i]]))[1],
#                    " - ID ", tag_ctmm_list[[i]]@info$identity))
#
# plot(SVF, CTMM = FIT1_ML_list[[i]][[1]], fraction = 0.02,
#      main = paste0("ML - ", rownames(summary(FIT1_ML_list[[i]]))[1],
#                    " - ID ", tag_ctmm_list[[i]]@info$identity))
```

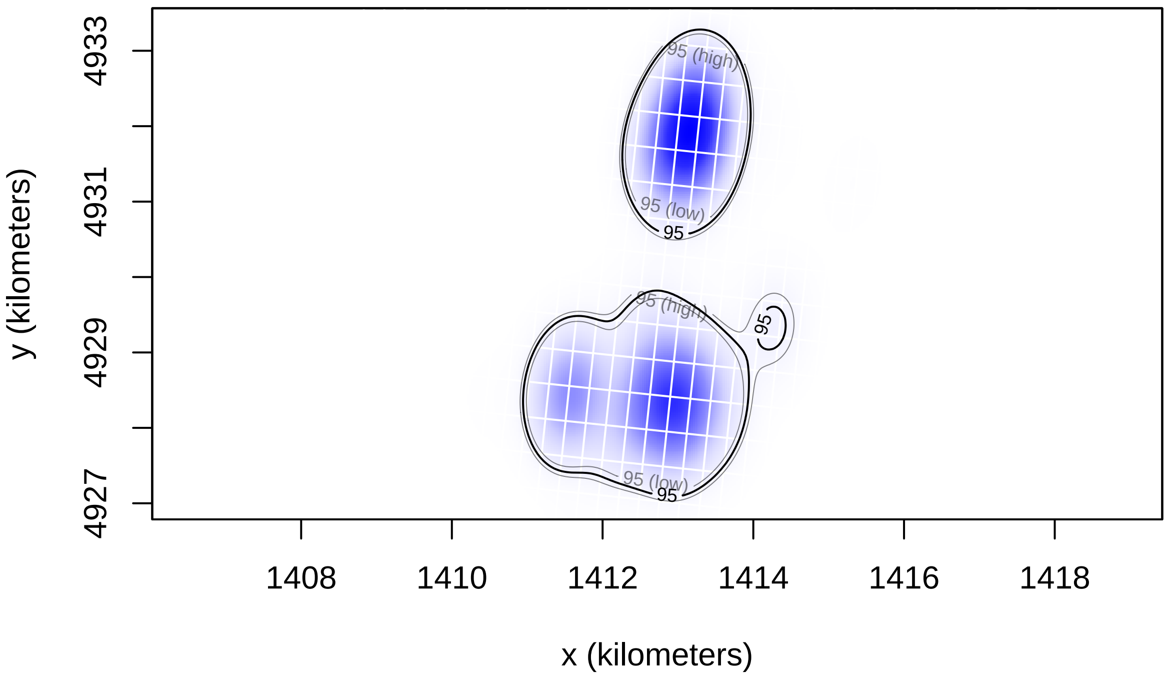
```r
# using pHREML
FIT1_pHREML_list[[i]] <- ctmm.select(tag_ctmm_list[[i]], GUESS1,
                           method = 'pHREML', IC = "AIC",
                           verbose = TRUE)

print(summary(FIT1_pHREML_list[[i]]))
# OUa_pHREML <- FIT1_pHREML_list[[i]][[1]]

plot(SVF, CTMM = FIT1_pHREML_list[[i]][[1]], fraction = 1,
     main = paste0("pHREML - ", rownames(summary(FIT1_pHREML_list[[i]]))[1],
                   " - ID ", tag_ctmm_list[[i]]@info$identity))

# to view zoomed in plot to assess fit
# plot(SVF, CTMM = FIT1_pHREML_list[[i]][[1]], fraction = 0.02,
#      main = paste0("pHREML - ", rownames(summary(FIT1_pHREML_list[[i]]))[1],
#                    " - ID ", tag_ctmm_list[[i]]@info$identity))

# Fitting home range model

# Run an area-corrected AKDE with weights:
# UD1w_ML_list[[i]] <- akde(tag_ctmm_list[[i]], FIT1_ML_list[[i]],
#                            weights = TRUE, grid = ext_raster)
# summary(UD1w_ML_list[[i]])
# summary(UD1_ML)$CI # home range area estimation
# plot(UD1w_ML_list[[i]])
# print(summary(UD1w_ML_list[[i]])$DOF["area"]) # effective sample size of animal1
# print(nrow(tag_ctmm_list[[i]])) # absolute sample size

# Run an area-corrected AKDE with weights using the pHREML estimated model:
UD1w_pHREML_list[[i]] <- akde(tag_ctmm_list[[i]], FIT1_pHREML_list[[i]],
                              weights = TRUE, grid = ext_raster)

summary(UD1w_pHREML_list[[i]])
plot(UD1w_pHREML_list[[i]])
print(summary(UD1w_pHREML_list[[i]])$DOF["area"]) # effective sample size of animal1
print(nrow(tag_ctmm_list[[i]])) # absolute sample size

}
```

```
## Minimum sampling interval of 14.9 minutes in 45505

##                        ΔAIC ΔRMSPE (m) DOF[area]
## OU anisotropic     0.000000   107.2540  286.8548
## OUF anisotropic    1.986637   102.4139  293.9135
## OU               465.438629   331.8688  227.5025
## OUF              467.422086   324.4040  234.9283
## OUf anisotropic 1290.708487     0.0000  978.8073
## IID anisotropic 3711.231921   303.6194 1722.0000
```



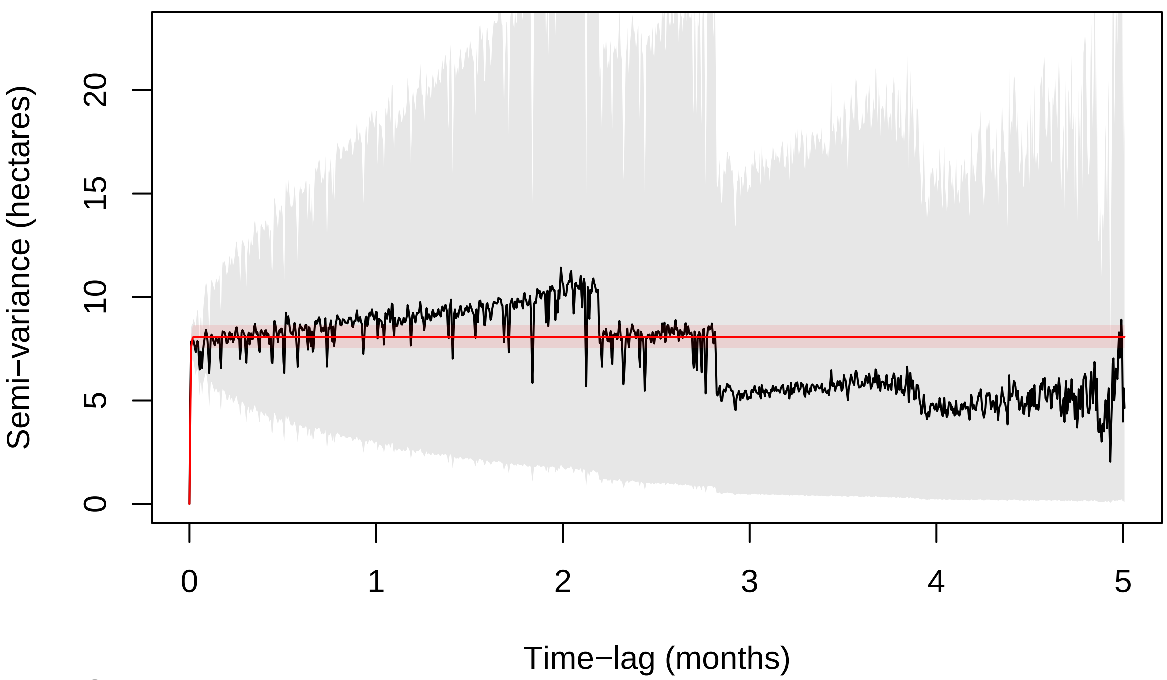

```
## Default grid size of 13.3314814814815 minutes chosen for bandwidth(...,fast=TRUE).
```

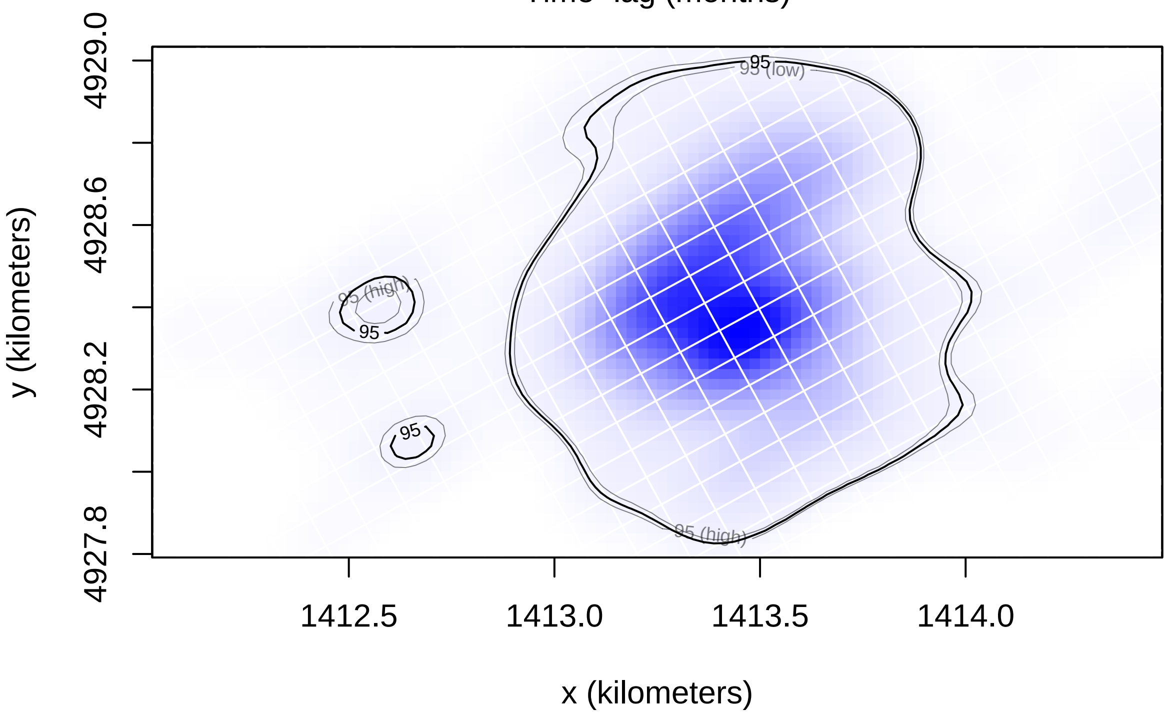





```
##     area
## 286.8548
## [1] 1723

## Minimum sampling interval of 59.9 minutes in 45506

##                         ΔAIC ΔRMSPE (m) DOF[area]
## OU anisotropic    0.0000000  0.0000000  846.9103
## OUF anisotropic   0.8208396  0.4442276  842.8596
## OUf anisotropic  23.4689080  1.1613180  891.1219
## OUF             120.7719872  4.7694035  827.8475
## OU              122.6437241  3.6203542  830.1620
## IID anisotropic 150.4756940  0.3683857  985.0000
```

pHREML − OU anisotropic − ID 45506



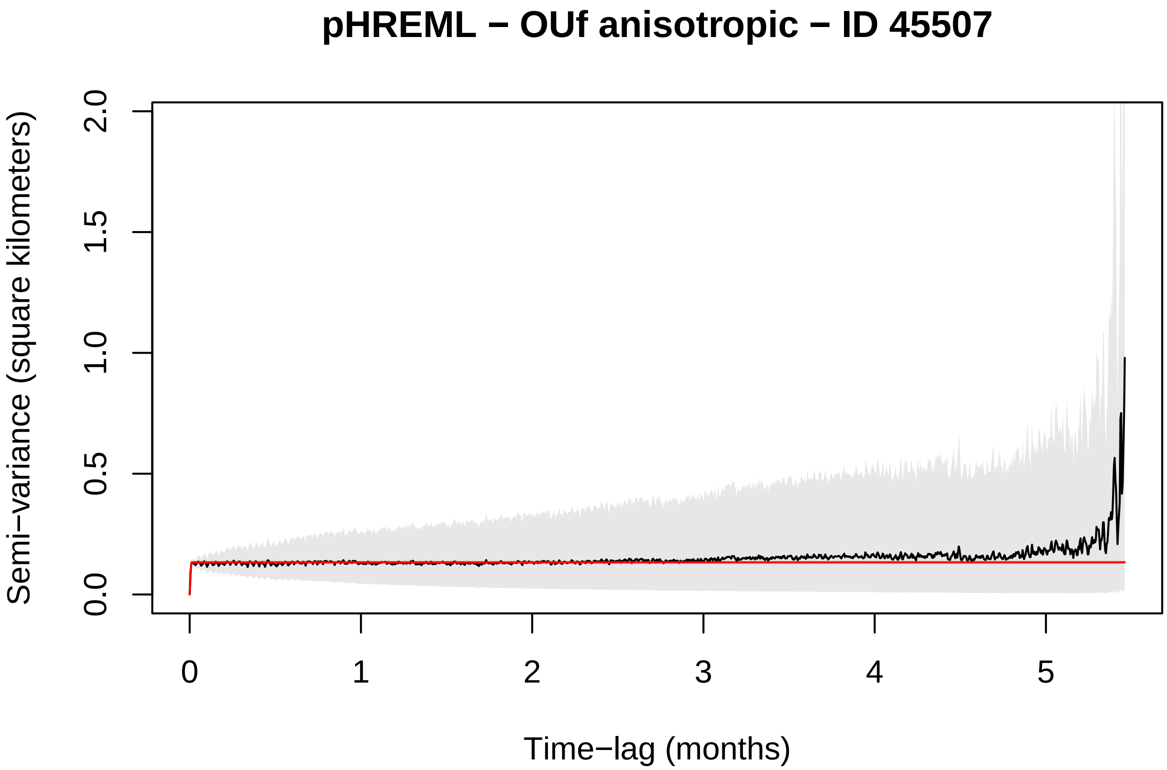



```
##     area
## 846.9103
## [1] 986
```

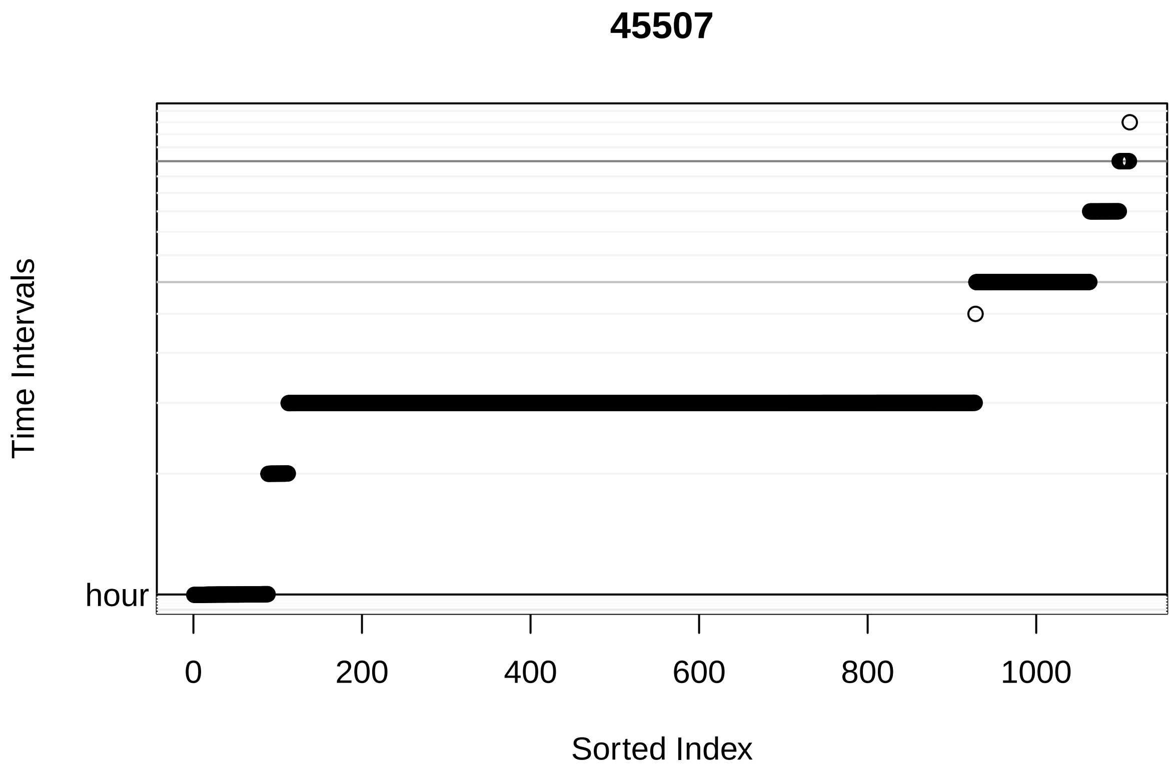
```
## Minimum sampling interval of 59.9 minutes in 45507
```

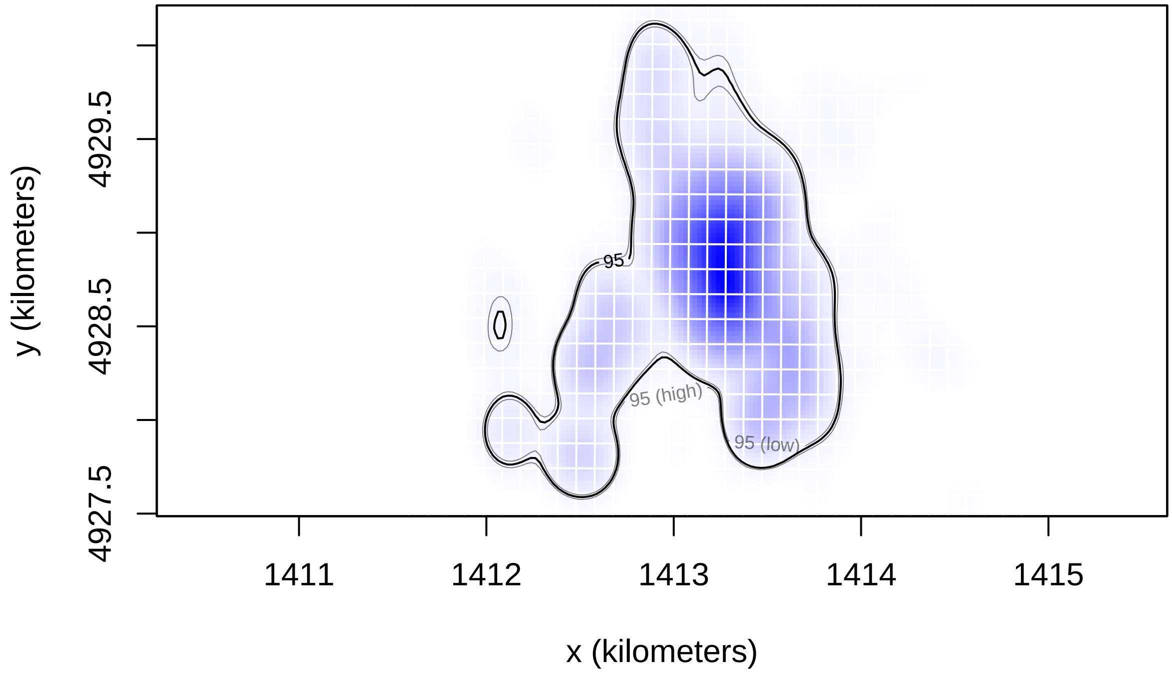
```
##                       ΔAIC ΔRMSPE (m) DOF[area]
## OUf anisotropic  0.00000   8.900758  1011.298
```

```
## OUΩ anisotropic  2.00000   8.900758  1000.172
## OUF anisotropic  2.00000   8.900758  1011.298
## OU anisotropic  24.04701   5.590181  1015.459
## IID anisotropic 86.69121   0.000000  1111.000
## OUf             92.34493   8.332524  1015.071
## OUF             94.34493   8.332524  1015.071
```




```
## Default grid size of 45 minutes chosen for bandwidth(...,fast=TRUE).
```





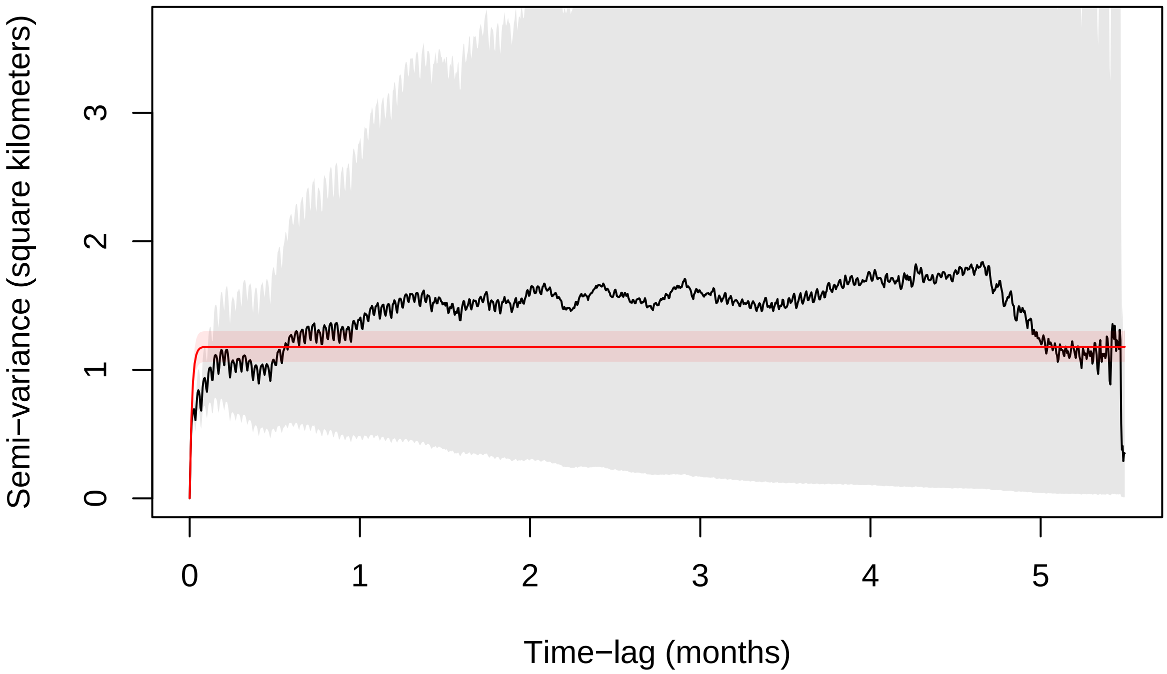

```
##     area
## 1011.298
## [1] 1112
```

```
## Minimum sampling interval of 59.9 minutes in 45508
```

```
##                          ΔAIC ΔRMSPE (m) DOF[area]
## OUF anisotropic   0.0000000    70.7448  369.4021
## OU anisotropic    0.4458837    71.0737  357.2924
## OUf anisotropic 315.8271248     0.0000  695.0203
```

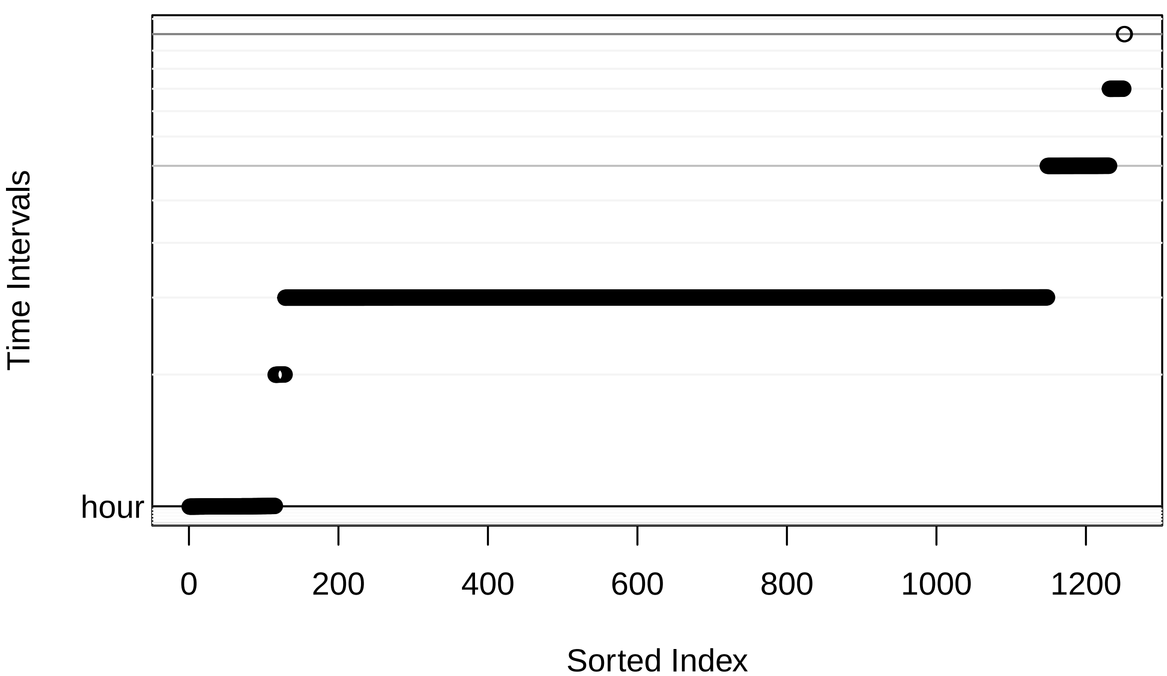
```
## OUF              534.8997393   225.9840  321.7635
```

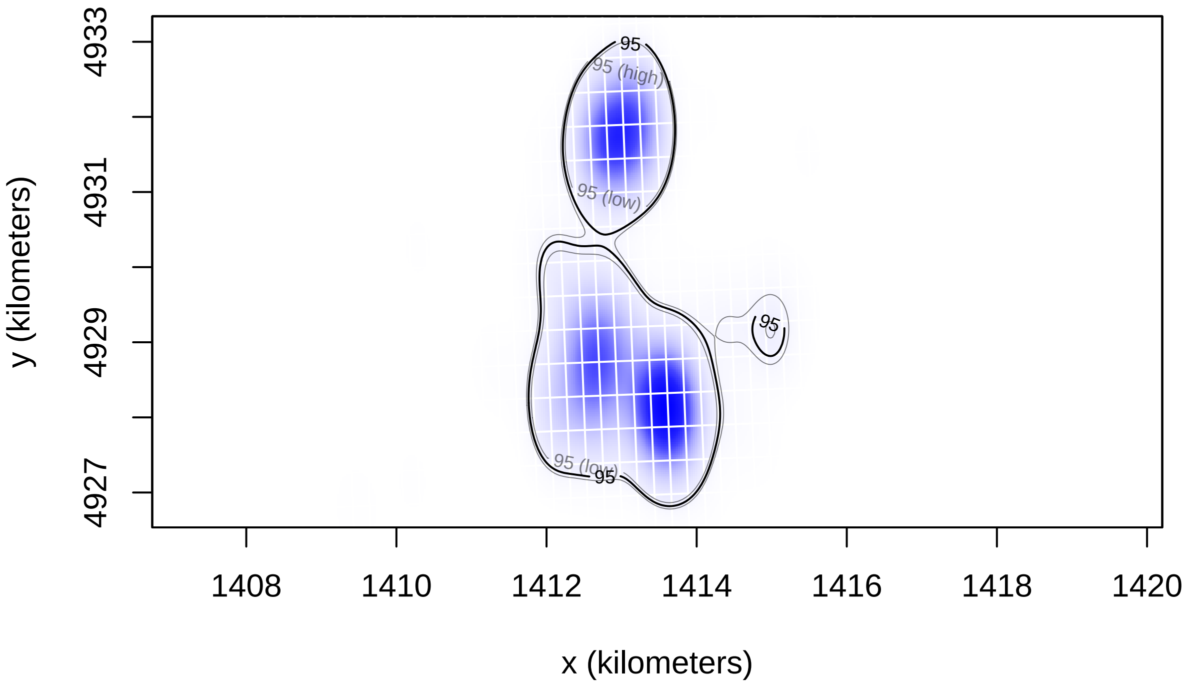
pHREML − OUF anisotropic − ID 45508



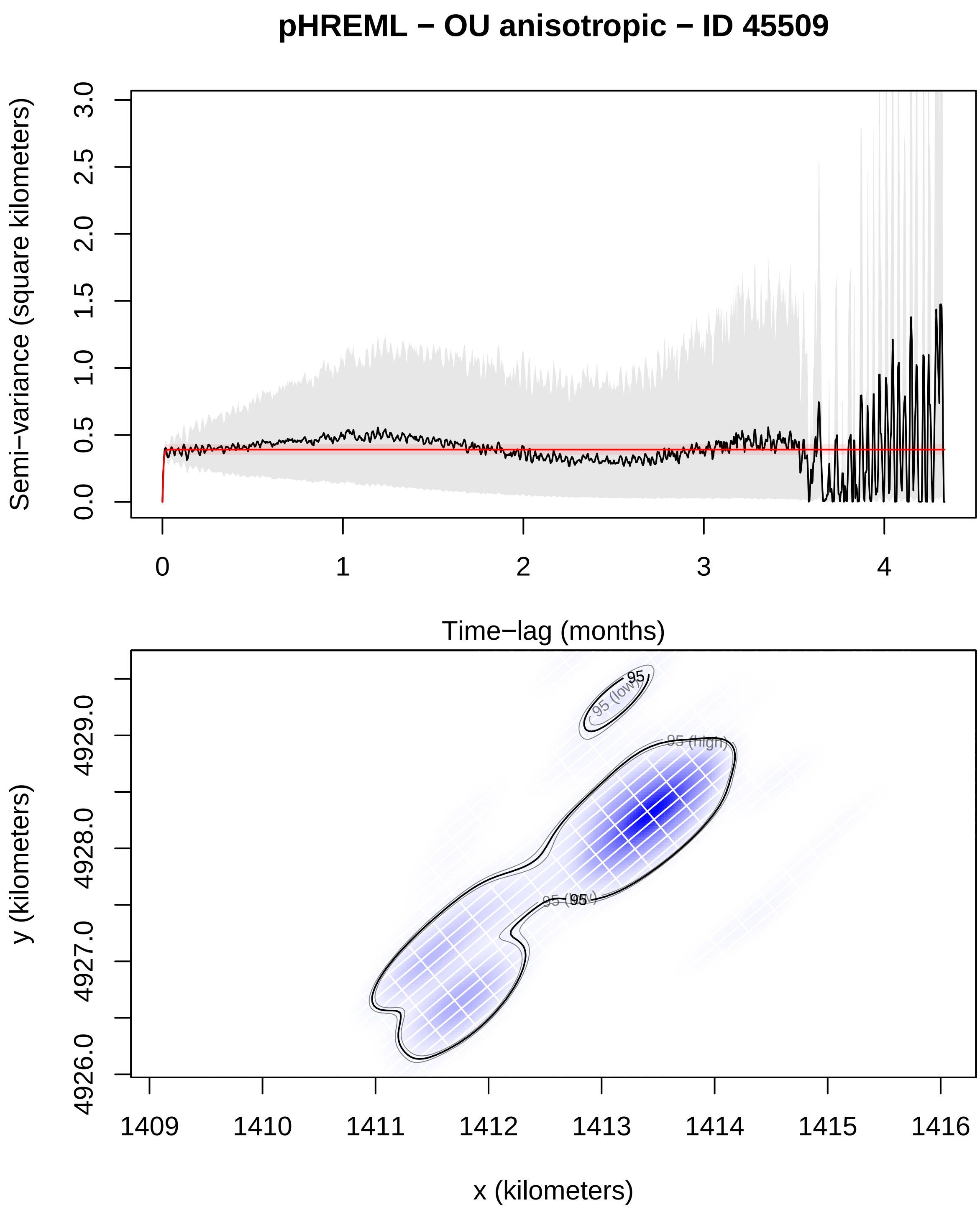

```
## Default grid size of 45 minutes chosen for bandwidth(...,fast=TRUE).
```

45508






```
##     area
## 369.4021
## [1] 1253

## Minimum sampling interval of 59.9 minutes in 45509

##                        ΔAIC ΔRMSPE (m) DOF[area]
## OU anisotropic    0.000000   9.153379  513.3899
## OUF anisotropic   1.950689   9.088046  510.7531
## OUf anisotropic  67.582592   0.000000  611.0610
## IID anisotropic 324.424055  33.217677  720.0000
## OUF             770.090095  55.208573  483.0277
## OU              797.168288  48.119598  457.0585
```

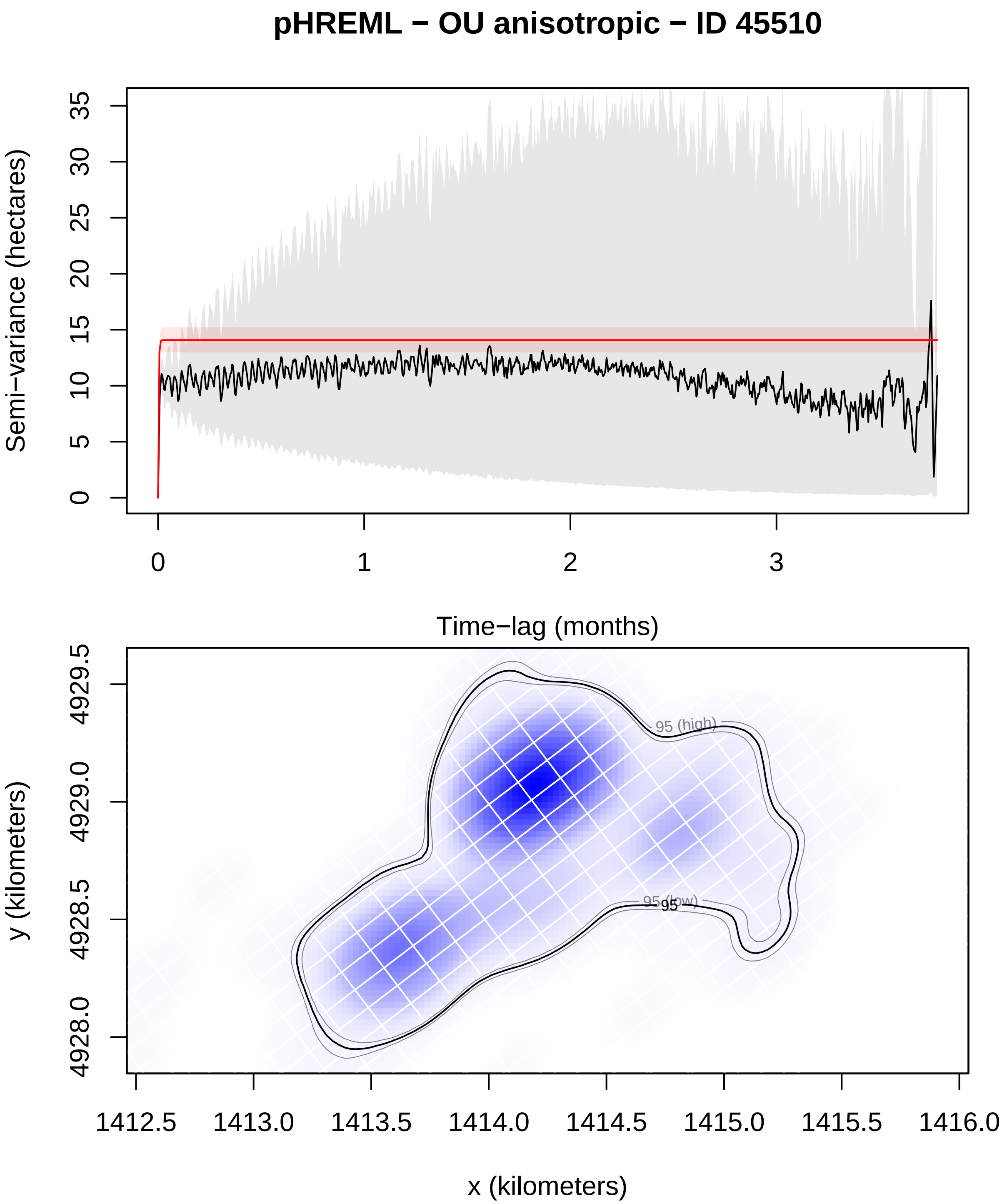



```
##     area
## 513.3899
## [1] 721
```

```
## Minimum sampling interval of 59.9 minutes in 45510
```

```
##                         ΔAIC ΔRMSPE (m) DOF[area]
## OU anisotropic      0.000000  0.1841355  685.6219
## OUF anisotropic     1.996786  0.0000000  687.7744
```

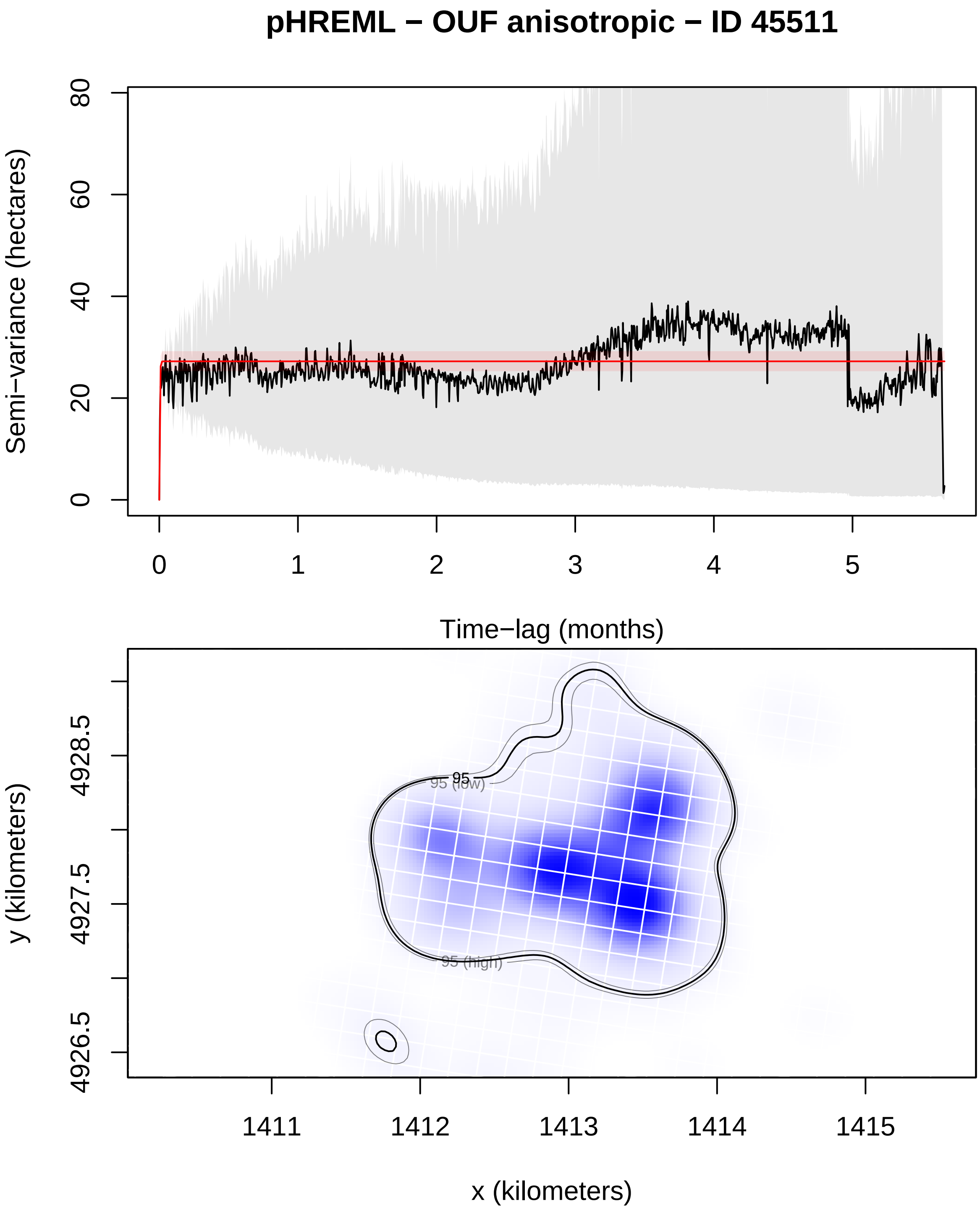
```
## OUf anisotropic  41.376353  2.0059506  724.7845
## IID anisotropic 170.746204  7.4402083  817.0000
## OU              182.431389  9.7269138  655.5609
## OUF             184.427858  9.4481709  658.0421
```




```
##     area
## 685.6219
## [1] 818
```

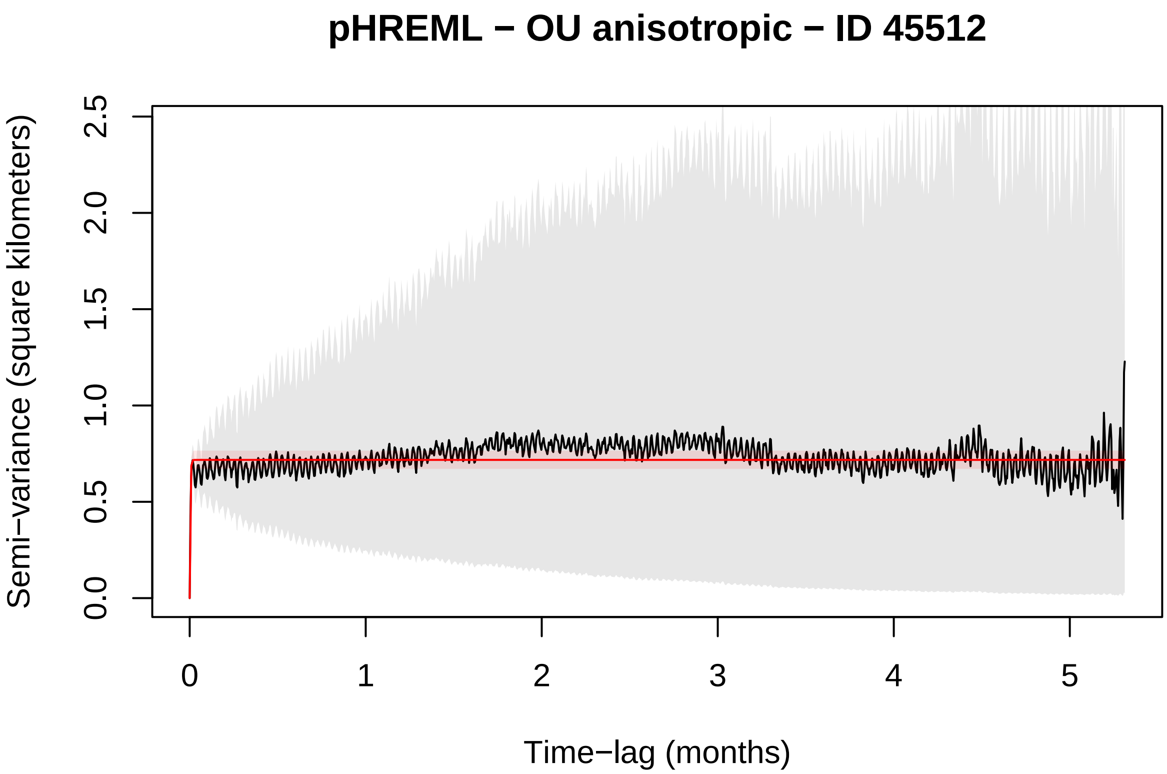
```
## Minimum sampling interval of 59.9 minutes in 45511
```

```
##                        ΔAIC ΔRMSPE (m) DOF[area]
## OUF anisotropic  0.000000   1.967446  733.7471
## OU anisotropic   7.389306   0.000000  717.7578
## OUf anisotropic 13.300706   0.661853  794.1486
## OUF             19.242447   6.925795  720.5158
```

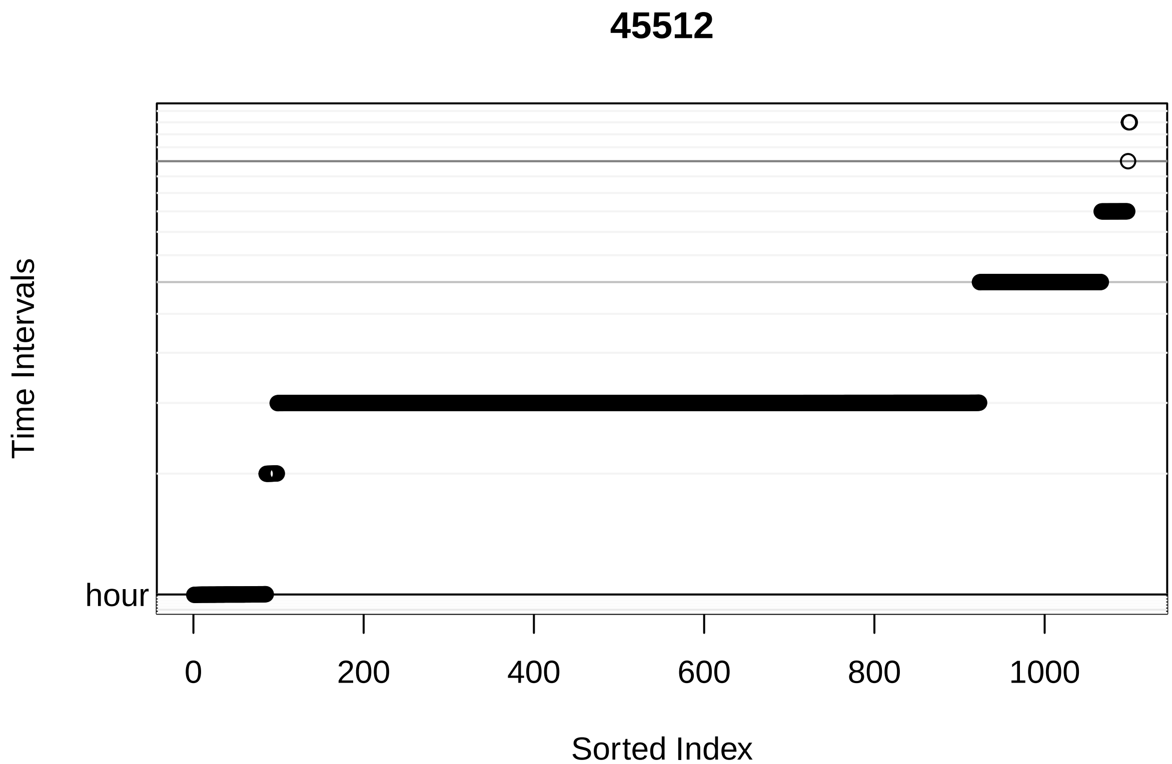


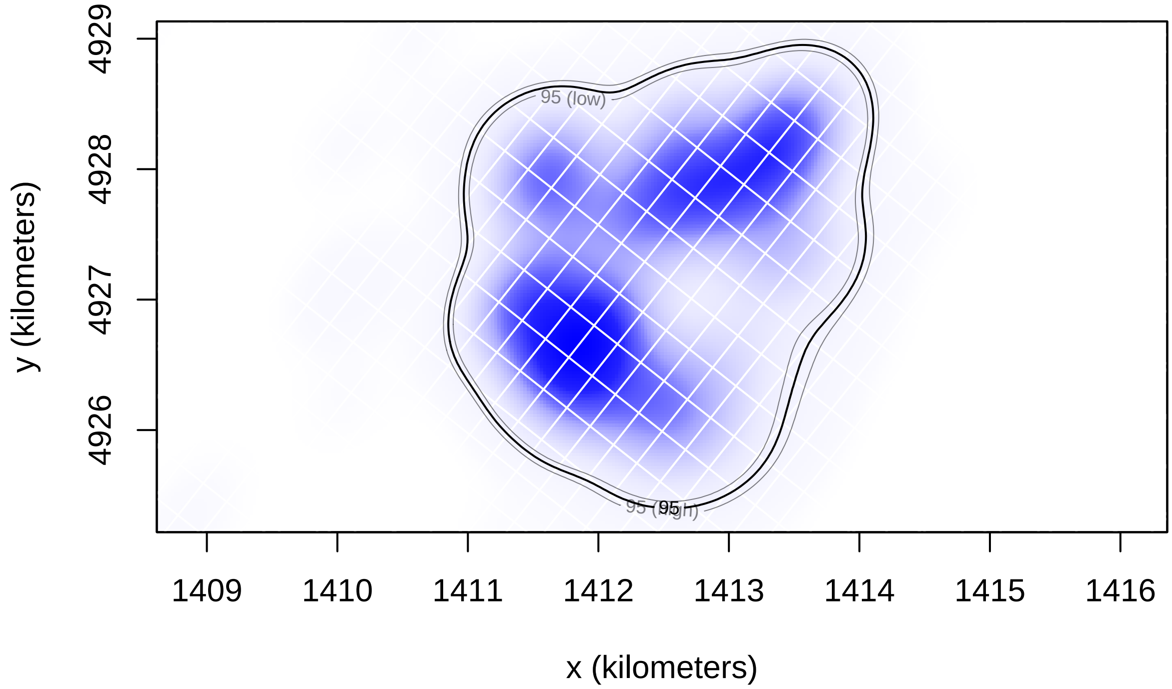

```
##     area
## 733.7471
```

```
## [1] 912

## Minimum sampling interval of 59.9 minutes in 45512

##                        ΔAIC ΔRMSPE (m) DOF[area]
## OU anisotropic    0.000000  11.301191  941.5755
## OUF anisotropic   1.997032  10.911905  944.1140
## OUf anisotropic  57.401865   7.375213 1041.0132
## OU              102.398751  14.997668  927.6605
## OUF             104.395637  14.561947  930.3576
## IID anisotropic 143.321028   0.000000 1100.0000
```




```
## Default grid size of 45 minutes chosen for bandwidth(...,fast=TRUE).
```



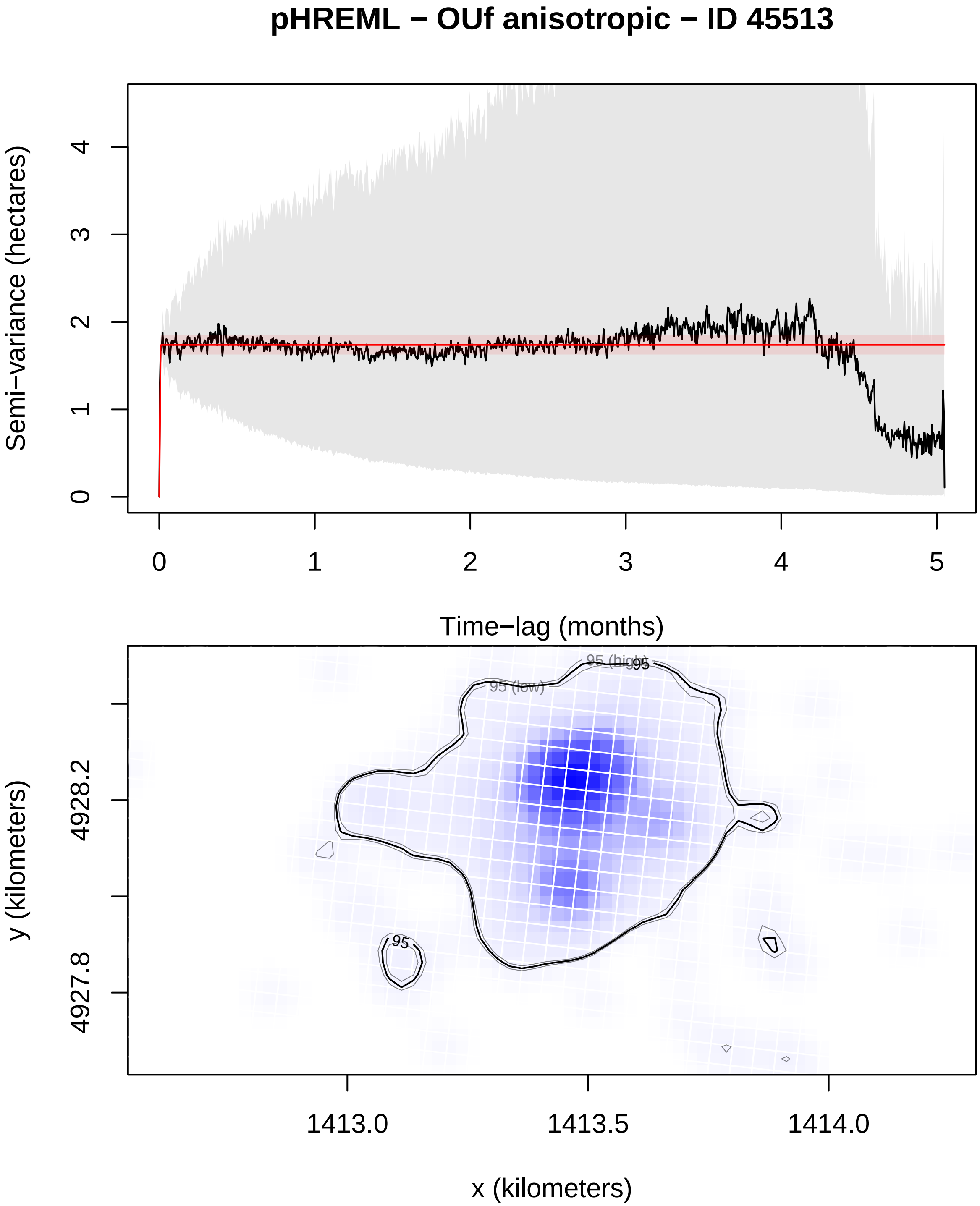



```
##      area
## 941.5755
## [1] 1101
```

```
## Minimum sampling interval of 59.9 minutes in 45513
```

```
##                        ΔAIC ΔRMSPE (m) DOF[area]
## OUf anisotropic  0.000000   1.907064  936.4161
## OUΩ anisotropic  2.000000   1.907064  906.3812
```

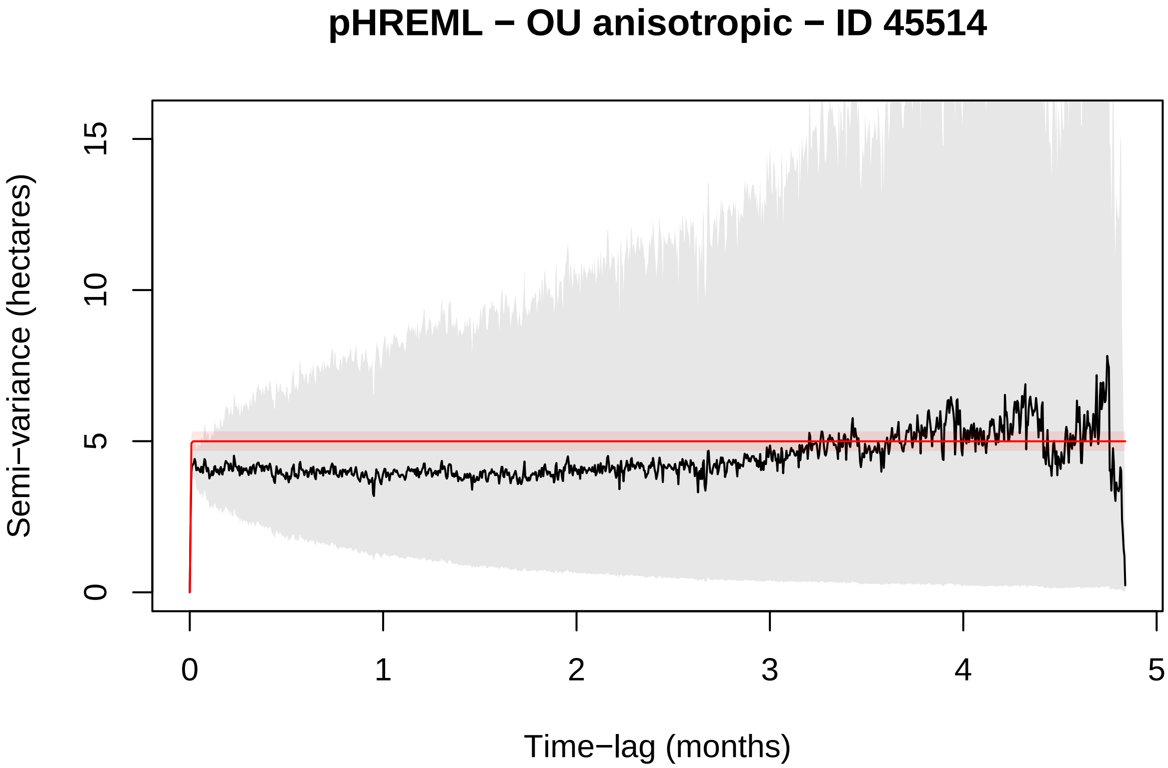
```
## OUF anisotropic  2.000000   1.907064  936.4161
## OUf              5.692126   1.997073  934.4467
## OUF              7.692126   1.997073  934.4467
## OU anisotropic  15.527871   1.068795  944.9249
## IID anisotropic 54.242845   0.000000  999.0000
```



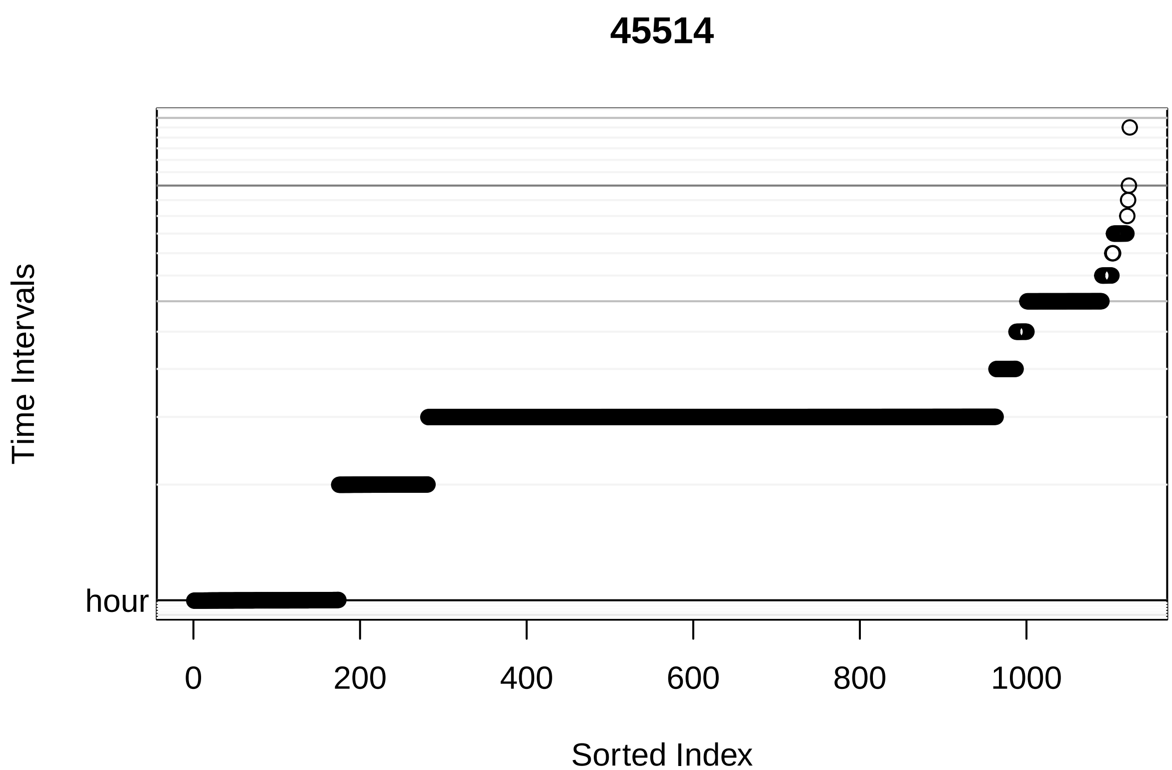

```
##     area
## 936.4161
## [1] 1000
```

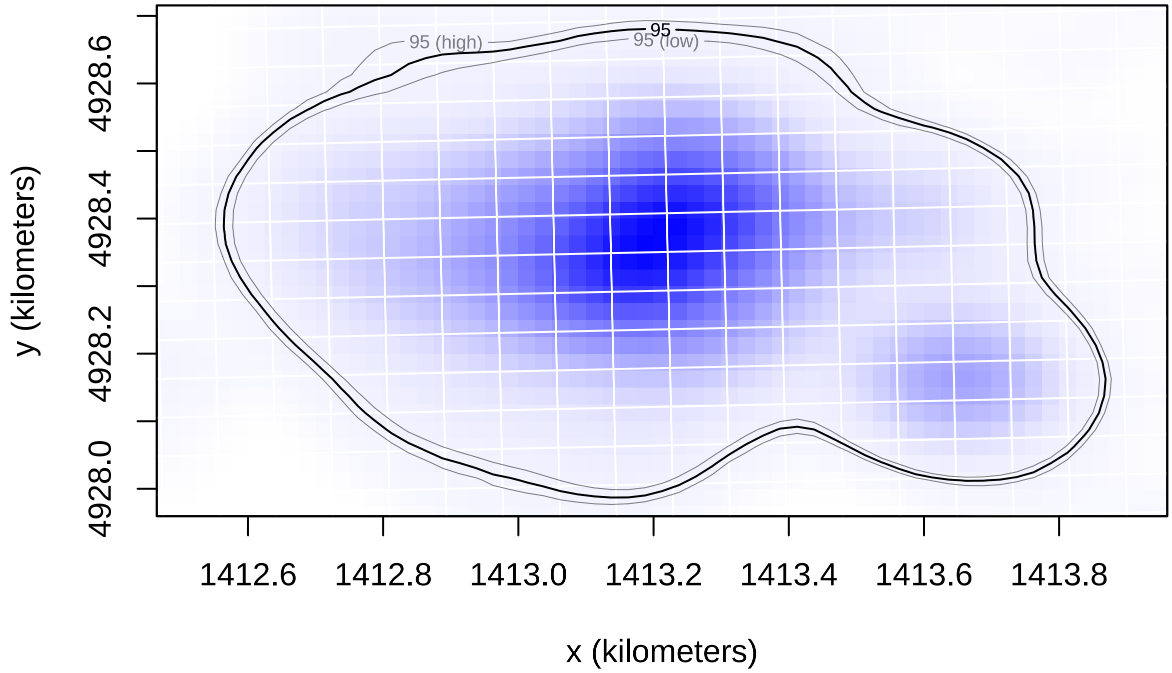
```
## Minimum sampling interval of 59.9 minutes in 45514
```

```
##                        ΔAIC ΔRMSPE (m) DOF[area]
## OU anisotropic    0.000000   4.027020  1001.722
## OUF anisotropic   1.997307   3.942493  1003.638
## OUf anisotropic  15.378231   4.118174  1013.689
## IID anisotropic  55.374160   0.000000  1124.000
## OU              156.194320   4.234378  1002.376
## OUF             158.191681   4.149972  1004.292
```




```
## Default grid size of 44.9958333333333 minutes chosen for bandwidth(...,fast=TRUE).
```






```
##     area
## 1001.722
## [1] 1125
```

To print any specific outputs.

These are commnented out but uncomment to view results (if the objects have been created).

```
for(i in 1:10) {
# print(summary(UD1w_pHREML_list[[i]]))
```

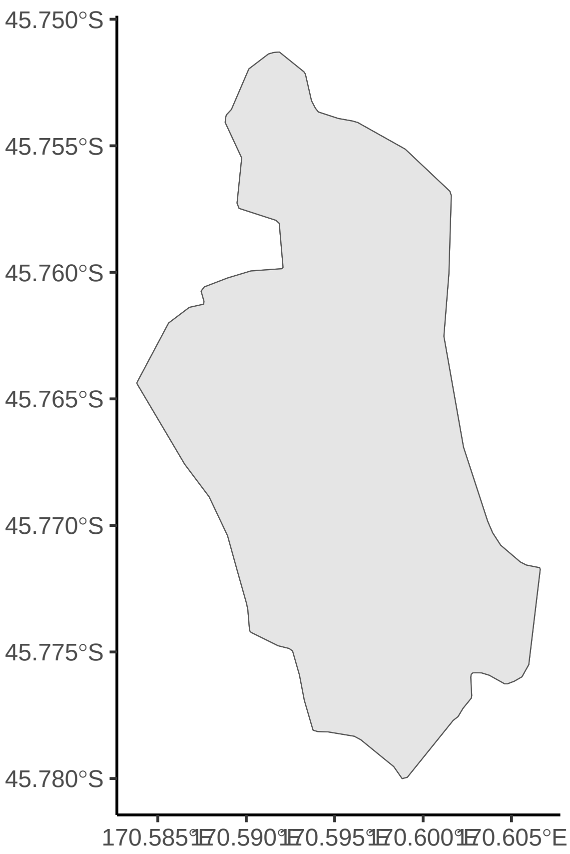
```
# print(summary(UD1w_pHREML_list[[i]])$CI)
# plot(UD1w_pHREML_list[[i]], level.UD = c(0.5, 0.95))
# plot(UD1w_pHREML_list[[i]], level.UD = c(0.5, 0.95))
}
```

Import a spatial object of the Orokonui Ecosanctuary fence.

```
OrokonuiFence <- st_read("mapping/OrokonuiFence.shp")
```

```
## Reading layer `OrokonuiFence' from data source
##   `/Users/scottforrest/Library/CloudStorage/OneDrive-QueenslandUniversityofTechnology/MSc - Scott Fo
##   using driver `ESRI Shapefile'
## Simple feature collection with 2 features and 1 field
## Geometry type: POLYGON
## Dimension:     XY
## Bounding box:  xmin: 1412176 ymin: 4927554 xmax: 1413950 ymax: 4930746
## Projected CRS: NZGD2000 / New Zealand Transverse Mercator 2000
```

```
# check map
ggplot(OrokonuiFence) +
  geom_sf() +
  theme_classic()
```



Extract contours for further analysis.

```
# use the ML estimated UDs
# UD_ctmm_contour_list <- map(UD1w_ML_list, SpatialPolygonsDataFrame.UD,
#                             level.UD = c(0.5, 0.95),
#                             level = 0.95)
```

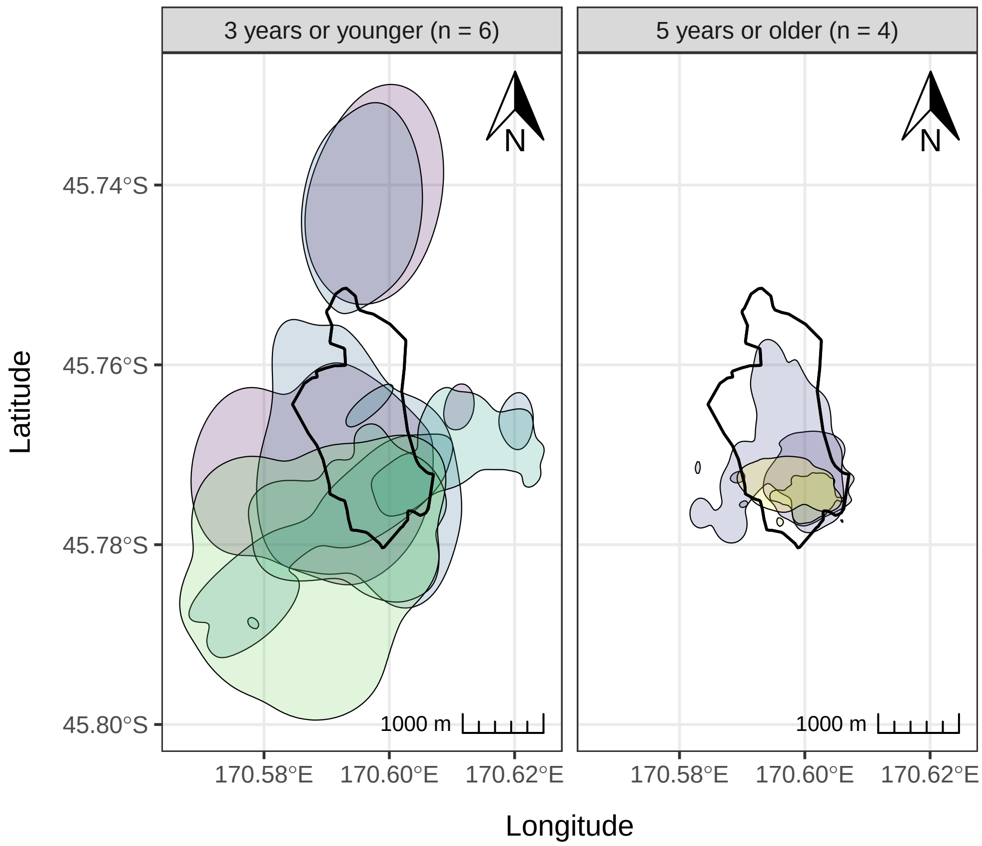
```r
# use the perturbative Hybrid REML (pHREML) estimated UDs
UD_ctmm_contour_list <- map(UD1w_pHREML_list, SpatialPolygonsDataFrame.UD,
                            level.UD = c(0.5, 0.95),
                            level = 0.95)

# create a df with the spatial objects
UD_ctmm_sf <- map(UD_ctmm_contour_list, st_as_sf, crs = 4326) %>%
  map(., st_transform, crs = 4326) %>%
  bind_rows() %>% mutate(id = substr(name, start = 1, stop = 5),
                         age = rep(c(1, 10, 5, 1, 3, 2, 2, 3, 10, 8), each = 6),
                         contour = substr(name, start = 7, stop = 8),
                         interval = substr(name, start = 11, stop = 13),
                         age_group = ifelse(age < 4,
                                            "3 years or younger (n = 6)",
                                            "5 years or older (n = 4)"))

UD_ctmm_sf
```

```
## Simple feature collection with 60 features and 6 fields
## Geometry type: GEOMETRY
## Dimension:     XY
## Bounding box:  xmin: 170.5661 ymin: -45.79998 xmax: 170.6251 ymax: -45.72836
## Geodetic CRS:  WGS 84
## First 10 features:
##                           name                       geometry    id age contour
## 45505 50% low   45505 50% low MULTIPOLYGON (((170.5884 -4... 45505   1      50
## 45505 50% est   45505 50% est MULTIPOLYGON (((170.5878 -4... 45505   1      50
## 45505 50% high 45505 50% high MULTIPOLYGON (((170.5874 -4... 45505   1      50
## 45505 95% low   45505 95% low MULTIPOLYGON (((170.5691 -4... 45505   1      95
## 45505 95% est   45505 95% est MULTIPOLYGON (((170.5685 -4... 45505   1      95
## 45505 95% high 45505 95% high MULTIPOLYGON (((170.5679 -4... 45505   1      95
## 45506 50% low   45506 50% low MULTIPOLYGON (((170.5972 -4... 45506  10      50
## 45506 50% est   45506 50% est MULTIPOLYGON (((170.5969 -4... 45506  10      50
## 45506 50% high 45506 50% high MULTIPOLYGON (((170.5969 -4... 45506  10      50
## 45506 95% low   45506 95% low MULTIPOLYGON (((170.5886 -4... 45506  10      95
##                interval                  age_group
## 45505 50% low       low 3 years or younger (n = 6)
## 45505 50% est       est 3 years or younger (n = 6)
## 45505 50% high      hig 3 years or younger (n = 6)
## 45505 95% low       low 3 years or younger (n = 6)
## 45505 95% est       est 3 years or younger (n = 6)
## 45505 95% high      hig 3 years or younger (n = 6)
## 45506 50% low       low   5 years or older (n = 4)
## 45506 50% est       est   5 years or older (n = 4)
## 45506 50% high      hig   5 years or older (n = 4)
## 45506 95% low       low   5 years or older (n = 4)
```

Plotting for manuscript

```r
ggplot() +
  geom_sf(data = UD_ctmm_sf %>% dplyr::filter(contour == 95 & interval == "est"),
          aes(fill = factor(id)),
          alpha = 0.2,
```

```r
            colour = "black",
            size = 0.25) +
# to also add the 50% contours - makes it a bit messy in this case
# geom_sf(data = UD_ctmm_sf %>% dplyr::filter(contour == 50 & interval == "est"),
#         # aes(fill = factor(id)),
#         alpha = 0,
#         colour = "black",
#         size = 0.25,
#         linetype = "dashed") +
geom_sf(data = OrokonuiFence, colour = "black", fill = NA, lwd = 0.5) +
coord_sf() +
scale_y_continuous("Latitude", breaks = seq(-45.74, -45.80, by = -0.02)) +
scale_x_continuous("Longitude", breaks = seq(170.56, 170.62, by = 0.02)) +
scale_fill_viridis_d(name = "ID") +
facet_wrap(vars(age_group)) +
theme_bw() +
theme(legend.position = "none",
      axis.title.y = element_text(margin = margin(t = 0, r = 10, b = 0, l = 0)),
      axis.title.x = element_text(margin = margin(t = 10, r = 0, b = 0, l = 0))) +
ggspatial::annotation_north_arrow(location = "tr",
                                  which_north = "true",
                                  height = unit(1, "cm"),
                                  width = unit(.75, "cm")) +
ggspatial::annotation_scale(location = "br",
                            style = "ticks",
                            bar_cols = c("grey60", "white"),
                            height = unit(0.25, "cm"))
```




```
filetypes <- c("pdf", "tiff", "jpeg", "png")

for(i in 1:length(filetypes)) {
ggsave(filename = paste0("Graphical outputs/",
                        "all_home_ranges_wAKDE_young_old_", Sys.Date(), ".",
                        filetypes[i]),
       width = 180,
       height = 150,
       units = "mm",
       dpi = 800)
}
```

Determine the area contained within the isopleths

```
UD_ctmm_area_50 <- vector(mode = "numeric", length = 10)
UD_ctmm_area_95 <- vector(mode = "numeric", length = 10)

for(i in 1:10) {
  # estimated 50% contour
  UD_ctmm_area_50[[i]] <- UD_ctmm_contour_list[[i]]@polygons[[2]]@area
  # estimated 95% contour
  UD_ctmm_area_95[[i]] <- UD_ctmm_contour_list[[i]]@polygons[[5]]@area
}
```

Add additional information about each kākā and create data frame

Familiarity is the number of years that a kākā had been in Orokonui Ecosanctuary, as we were trying to determine whether the kākā's age was leading to small home ranges, or whether it was how long each kākā had been in Orokonui, as we thought that individuals which had been recently released in Orokonui might have been more exploratory, even though they were older. For most individuals their age is the same as their familiarity, as they fledged in Orokonui or were released as juveniles, but there is one 10 year old individual that was released as an adult. This individual had a very small home range, which gave more weight to the 'age' hypothesis rather than 'familiarity'.

```
id <- 45505:45514
age <-          c(1, 10, 5, 1, 3, 2, 2, 3, 10, 8)
familiarity <-  c(1, 10, 5, 1, 3, 2, 2, 3, 1,  8)
sex <- c("M", "M", "M", "F", "F", "F", "F", "M", "M", "F")
origin <- c("Orokonui", "Orokonui", "Captive", "Orokonui", "Orokonui", "Orokonui",
            "Orokonui",  "Captive", "Captive", "Orokonui")

all_contours_95_df <- data.frame("ID" = id,
                                 "Age" = age,
                                 "Familiarity" = familiarity,
                                 "Sex" = sex,
                                 "Origin" = origin,
                                 "UD50_area" = UD_ctmm_area_50,
                                 "UD95_area" = UD_ctmm_area_95,
                                 "UD50_area_km2" = UD_ctmm_area_50/1e6,
                                 "UD95_area_km2" = UD_ctmm_area_95/1e6) %>%
  mutate(age_group = ifelse(Age < 4, "3 years or younger (n = 6)",
                            "5 years or older (n = 4)"))

head(all_contours_95_df)
```

```
##      ID Age Familiarity Sex   Origin UD50_area UD95_area UD50_area_km2
## 1 45505   1           1   M Orokonui 1860836.7 9924325.7     1.8608367
## 2 45506  10          10   M Orokonui  145453.7  987249.6     0.1454537
## 3 45507   5           5   M  Captive  251523.8 2377871.7     0.2515238
## 4 45508   1           1   F Orokonui 1902419.0 9705908.9     1.9024190
## 5 45509   3           3   F Orokonui  241209.0 3044136.8     0.2412090
## 6 45510   2           2   F Orokonui  190743.3 1823074.1     0.1907433
##   UD95_area_km2                  age_group
## 1     9.9243257 3 years or younger (n = 6)
## 2     0.9872496   5 years or older (n = 4)
## 3     2.3778717   5 years or older (n = 4)
## 4     9.7059089 3 years or younger (n = 6)
## 5     3.0441368 3 years or younger (n = 6)
## 6     1.8230741 3 years or younger (n = 6)
```

```
max(all_contours_95_df$UD95_area_km2) / min(all_contours_95_df$UD95_area_km2)
```

```
## [1] 29.16754
```

```
# size difference between largest and smallest UDs = 29 fold-difference
```

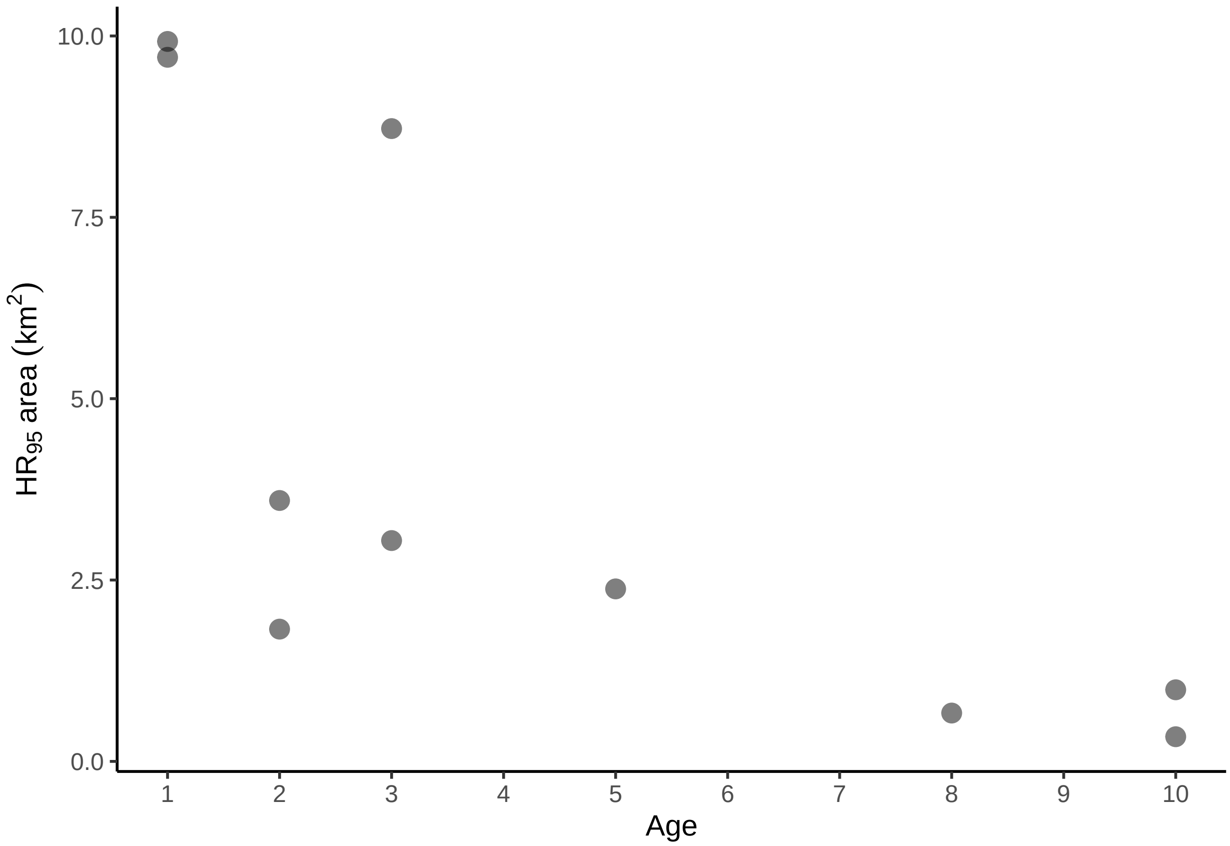
Convert to utilisation distributions (continuous probability surfaces).

Calculating the area contained within UD isopleths, which will be in m^2.

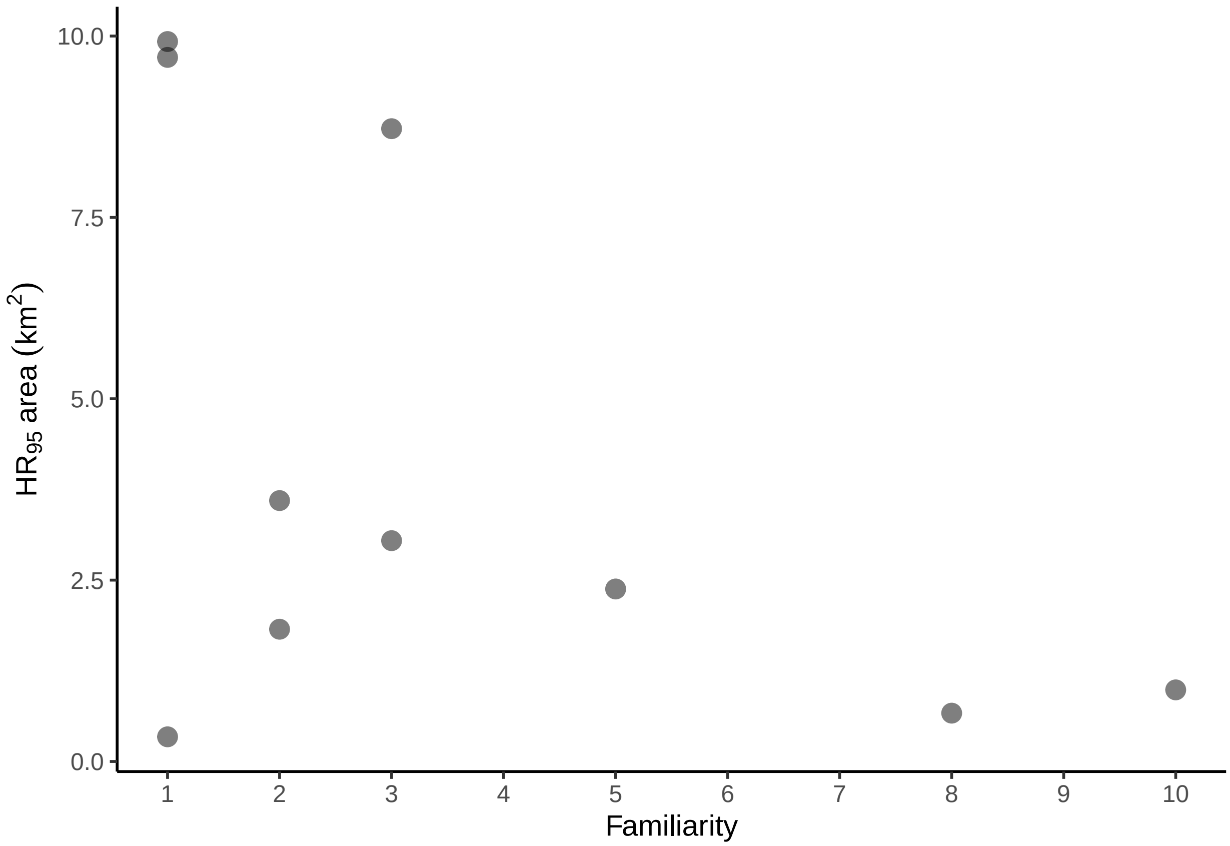
Add in individual-level covariates and create a data frame.

Plot

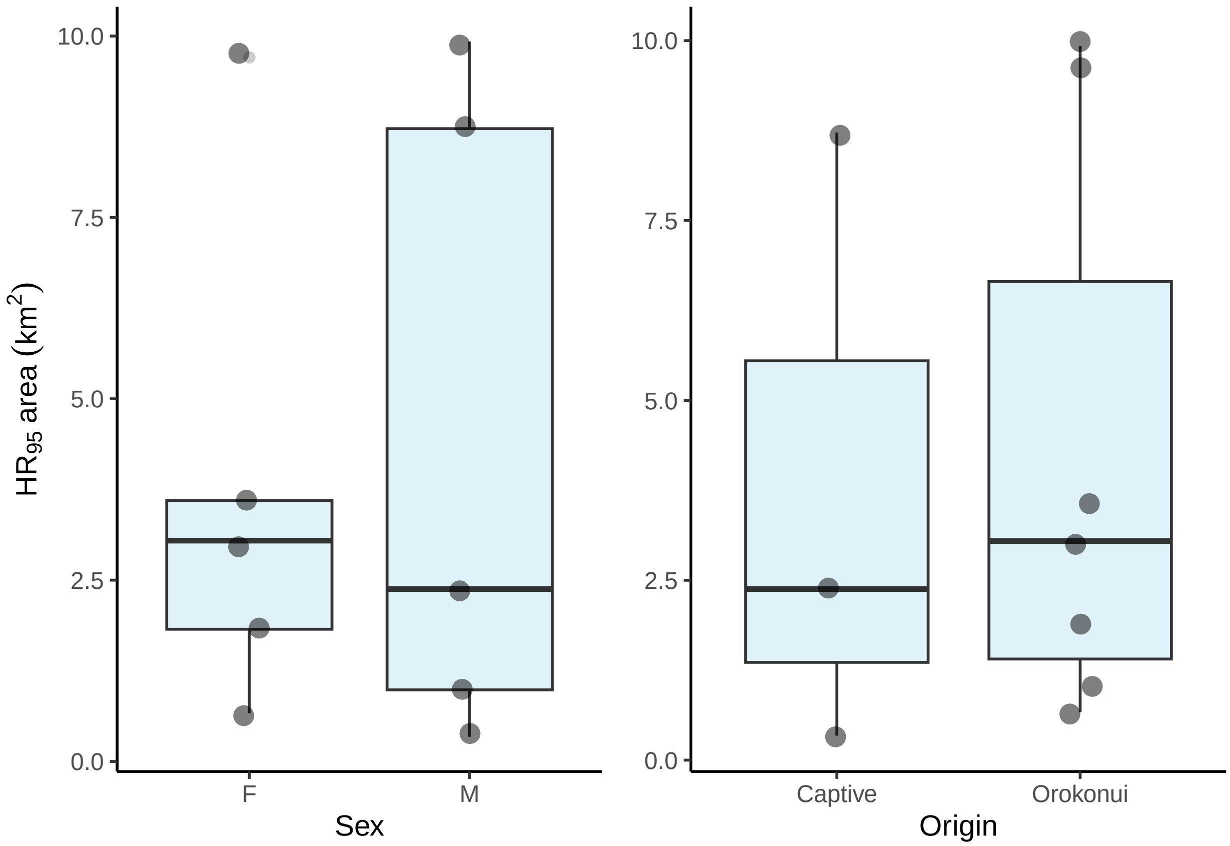
```
all_contours_95_df %>% ggplot(aes(Age, UD95_area_km2)) +
  geom_point(alpha = 0.5, size = 3) +
  scale_x_continuous(breaks = c(1:10)) +
  labs(y = expression(HR[95]~area~(km^2))) +
  theme_classic()
```




```
all_contours_95_df %>% ggplot(aes(Familiarity, UD95_area_km2)) +
  geom_point(alpha = 0.5, size = 3) +
  scale_x_continuous(breaks = c(1:10)) +
  labs(y = expression(HR[95]~area~(km^2))) +
  theme_classic()
```




```
sexplot <- all_contours_95_df %>% ggplot(aes(Sex, UD95_area_km2)) +
  geom_boxplot(alpha = 0.25, fill = "skyblue") +
  # geom_violin(alpha = 0.25, fill = "skyblue") +
  geom_jitter(width = 0.05, size = 3, alpha = 0.5) +
  labs(y = expression(HR[95]~area~(km^2))) +
  theme_classic()

originplot <- all_contours_95_df %>% ggplot(aes(Origin, UD95_area_km2)) +
  geom_boxplot(alpha = 0.25, fill = "skyblue") +
  # geom_violin(alpha = 0.25, fill = "skyblue") +
  geom_jitter(width = 0.05, size = 3, alpha = 0.5) +
  theme_classic()

ggarrange(sexplot, originplot + rremove("ylab"))
```




```
# save objects in multiple filetypes
for(i in 1:length(filetypes)){
ggsave(filename = paste0("Graphical outputs/",
                        "sex_age_plots_", Sys.Date(), ".",
                        filetypes[i]),
       width = 160,
       height = 100,
       units = "mm",
       dpi = 800)
}
```

Statistical analysis of home range area

Now that we have extracted the area from the contours, we can use those values in statistical models such as GLMs, to assess whether there are any significant differences in home range area due to factors such as age and sex.

Fit the models

```
# four models used for model selection
area_age_glm <- glm(UD95_area_km2 ~ Age,
                    data = all_contours_95_df,
                    family = Gamma(link = "log"))

area_familiarity_glm <- glm(UD95_area_km2 ~ Familiarity,
                            data = all_contours_95_df,
```

```
                            family = Gamma(link = "log"))

area_sex_glm <- glm(UD95_area_km2 ~ Sex,
                    data = all_contours_95_df,
                    family = Gamma(link = "log"))

area_origin_glm <- glm(UD95_area_km2 ~ Origin,
                       data = all_contours_95_df,
                       family = Gamma(link = "log"))

area_null_glm <- glm(UD95_area_km2 ~ 1,
                     data = all_contours_95_df,
                     family = Gamma(link = "log"))
```

Model summaries of age model

```
# model with age
summary(area_age_glm)
```

```
##
## Call:
## glm(formula = UD95_area_km2 ~ Age, family = Gamma(link = "log"),
##     data = all_contours_95_df)
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.33979    0.30026   7.792 5.27e-05 ***
## Age         -0.28734    0.05333  -5.388 0.000655 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 0.3256456)
##
##     Null deviance: 10.1179  on 9  degrees of freedom
## Residual deviance:  2.7417  on 8  degrees of freedom
## AIC: 40.088
##
## Number of Fisher Scoring iterations: 5
```

```
AIC(area_age_glm)
```

```
## [1] 40.08817
```

```
anova(area_age_glm,test="F")
```

```
## Analysis of Deviance Table
##
## Model: Gamma, link: log
##
## Response: UD95_area_km2
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev      F   Pr(>F)
```

```
## NULL                     9    10.1179
## Age   1   7.3762         8     2.7417 22.651 0.001428 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
r.squaredLR(area_age_glm)
```

```
## [1] 0.7592754
## attr(,"adj.r.squared")
## [1] 0.7653702
```

```r
confint(area_age_glm)
```

```
## Waiting for profiling to be done...
```

```
##                   2.5 %     97.5 %
## (Intercept)  1.7917699  2.9596986
## Age         -0.3867466 -0.1798822
```

Model summaries of familiarity model

```r
# model with age
summary(area_familiarity_glm)
```

```
##
## Call:
## glm(formula = UD95_area_km2 ~ Familiarity, family = Gamma(link = "log"),
##     data = all_contours_95_df)
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.06945    0.34236   6.045 0.000308 ***
## Familiarity -0.23166    0.07333  -3.159 0.013408 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 0.4752928)
##
##     Null deviance: 10.1179  on 9  degrees of freedom
## Residual deviance:  6.5156  on 8  degrees of freedom
## AIC: 49.361
##
## Number of Fisher Scoring iterations: 5
```

```r
AIC(area_familiarity_glm)
```

```
## [1] 49.3608
```

```r
anova(area_familiarity_glm,test="F")
```

```
## Analysis of Deviance Table
##
## Model: Gamma, link: log
##
## Response: UD95_area_km2
##
## Terms added sequentially (first to last)
```

```
##
##
##             Df Deviance Resid. Df Resid. Dev     F  Pr(>F)
## NULL                            9    10.1179
## Familiarity  1   3.6023         8     6.5156 7.579 0.02494 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
r.squaredLR(area_familiarity_glm)
```

```
## [1] 0.3915488
## attr(,"adj.r.squared")
## [1] 0.3946919
```

```
confint(area_familiarity_glm)
```

```
## Waiting for profiling to be done...
```

```
##                  2.5 %      97.5 %
## (Intercept)  1.4365759  2.78387860
## Familiarity -0.3647995 -0.07404951
```

Model summaries of sex model

```
# model with sex
summary(area_sex_glm)
```

```
##
## Call:
## glm(formula = UD95_area_km2 ~ Sex, family = Gamma(link = "log"),
##     data = all_contours_95_df)
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.3263     0.4342   3.055   0.0157 *
## SexM          0.1712     0.6140   0.279   0.7875
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 0.9425835)
##
##     Null deviance: 10.118  on 9  degrees of freedom
## Residual deviance: 10.045  on 8  degrees of freedom
## AIC: 54.245
##
## Number of Fisher Scoring iterations: 6
```

```
AIC(area_sex_glm)
```

```
## [1] 54.24532
```

```
anova(area_sex_glm,test="F")
```

```
## Analysis of Deviance Table
##
## Model: Gamma, link: log
##
```

```
## Response: UD95_area_km2
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev      F Pr(>F)
## NULL                      9     10.118
## Sex   1 0.073168          8     10.045 0.0776 0.7876
```

```r
r.squaredLR(area_sex_glm)
```

```
## [1] 0.008351581
## attr(,"adj.r.squared")
## [1] 0.008418621
```

```r
confint(area_sex_glm)
```

```
## Waiting for profiling to be done...
```

```
##                  2.5 %   97.5 %
## (Intercept)  0.5810799 2.317249
## SexM        -1.0689425 1.411305
```

Model summaries of origin model

```r
# model with kākā origin
summary(area_origin_glm)
```

```
##
## Call:
## glm(formula = UD95_area_km2 ~ Origin, family = Gamma(link = "log"),
##     data = all_contours_95_df)
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.3386     0.5697   2.350   0.0467 *
## OriginOrokonui    0.1082     0.6810   0.159   0.8777
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 0.9737961)
##
##     Null deviance: 10.118  on 9  degrees of freedom
## Residual deviance: 10.094  on 8  degrees of freedom
## AIC: 54.301
##
## Number of Fisher Scoring iterations: 6
```

```r
AIC(area_origin_glm)
```

```
## [1] 54.30147
```

```r
anova(area_origin_glm,test="F")
```

```
## Analysis of Deviance Table
##
## Model: Gamma, link: log
```

```
##
## Response: UD95_area_km2
##
## Terms added sequentially (first to last)
##
##
##        Df Deviance Resid. Df Resid. Dev      F Pr(>F)
## NULL                       9     10.118
## Origin  1 0.024235         8     10.094 0.0249 0.8786
```

```r
r.squaredLR(area_origin_glm)
```

```
## [1] 0.002768081
## attr(,"adj.r.squared")
## [1] 0.002790301
```

```r
confint(area_origin_glm)
```

```
## Waiting for profiling to be done...
```

```
##                      2.5 %   97.5 %
## (Intercept)      0.3970024 2.708181
## OriginOrokonui -1.4032936 1.371822
```

Model summaries of null model

```r
# null model (intercept only)
summary(area_null_glm)
```

```
##
## Call:
## glm(formula = UD95_area_km2 ~ 1, family = Gamma(link = "log"),
##     data = all_contours_95_df)
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.4156     0.2939   4.817 0.000951 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 0.8635406)
##
##     Null deviance: 10.118  on 9  degrees of freedom
## Residual deviance: 10.118  on 9  degrees of freedom
## AIC: 52.329
##
## Number of Fisher Scoring iterations: 5
```

```r
AIC(area_null_glm)
```

```
## [1] 52.32919
```

```r
anova(area_null_glm,test="F")
```

```
## Analysis of Deviance Table
##
## Model: Gamma, link: log
```

```
##
## Response: UD95_area_km2
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev F Pr(>F)
## NULL                     9     10.118
```

```
r.squaredLR(area_null_glm)
```

```
## [1] 0
## attr(,"adj.r.squared")
## [1] 0
```

```
confint(area_null_glm)
```

```
## Waiting for profiling to be done...
```

```
##     2.5 %    97.5 %
## 0.8901078 2.0525576
```

Model diagnostics

Only the age and familiarity models were significant, but as they were competing hypotheses we chose one, which was the age model, as it explains the data better (the older individual that was released into Orokonui had a small home range), and has a higher R-squared. We therefore check model diagnostics and produce a plot with fitted model.

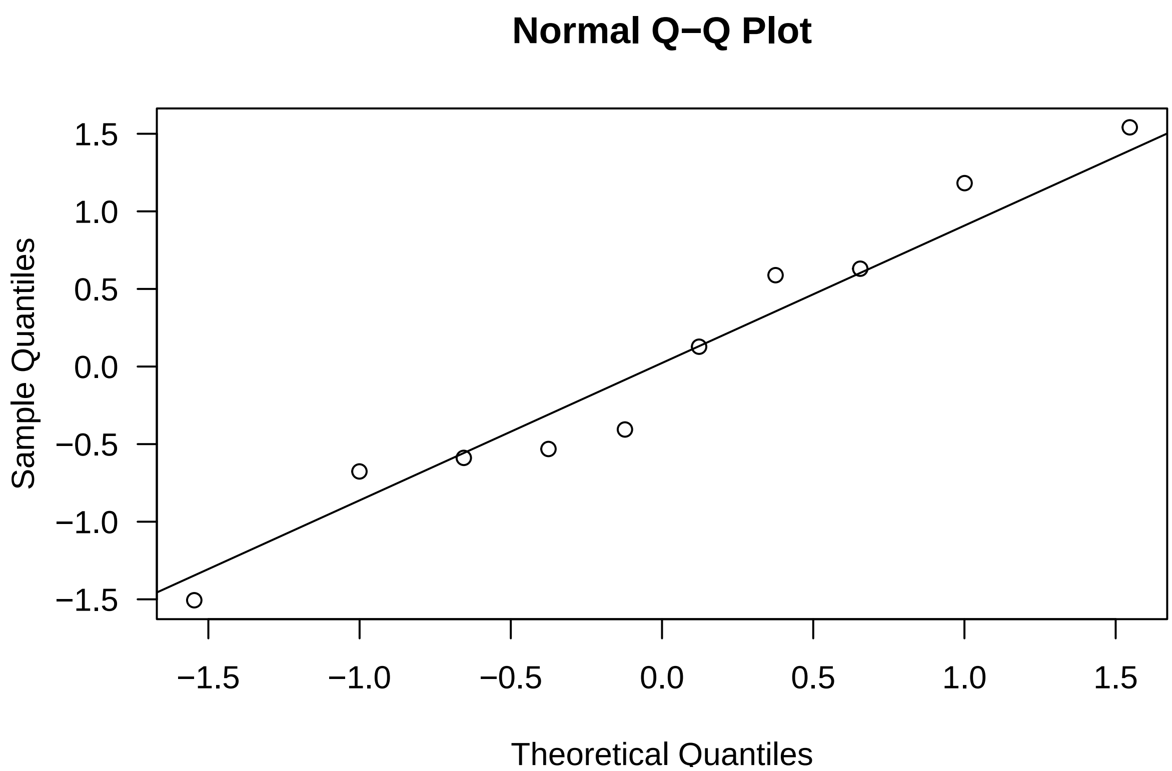
```
dfun <- function(object) {
  with(object,sum((weights * residuals^2)[weights > 0])/df.residual)
}

dfun(area_age_glm)
```

```
## [1] 0.3256456
```

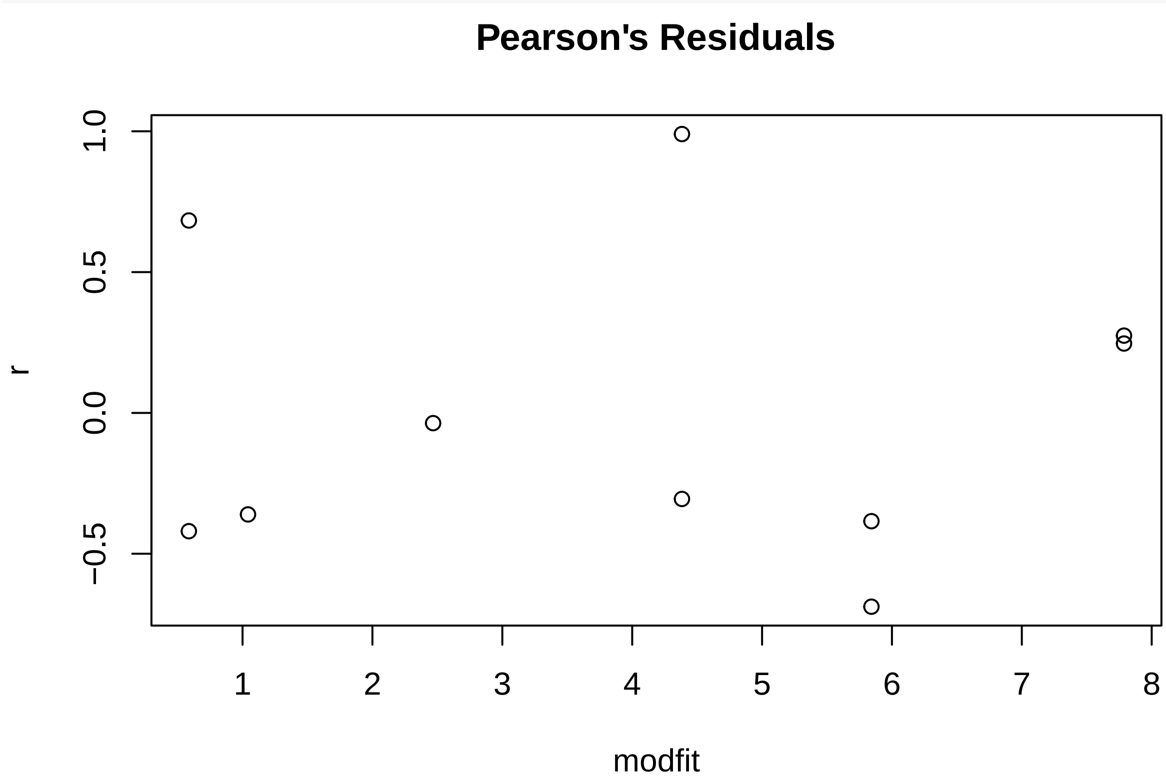
```
pseudoR2 <- 1 - (area_age_glm$deviance / area_age_glm$null.deviance)
pseudoR2
```

```
## [1] 0.7290285
```

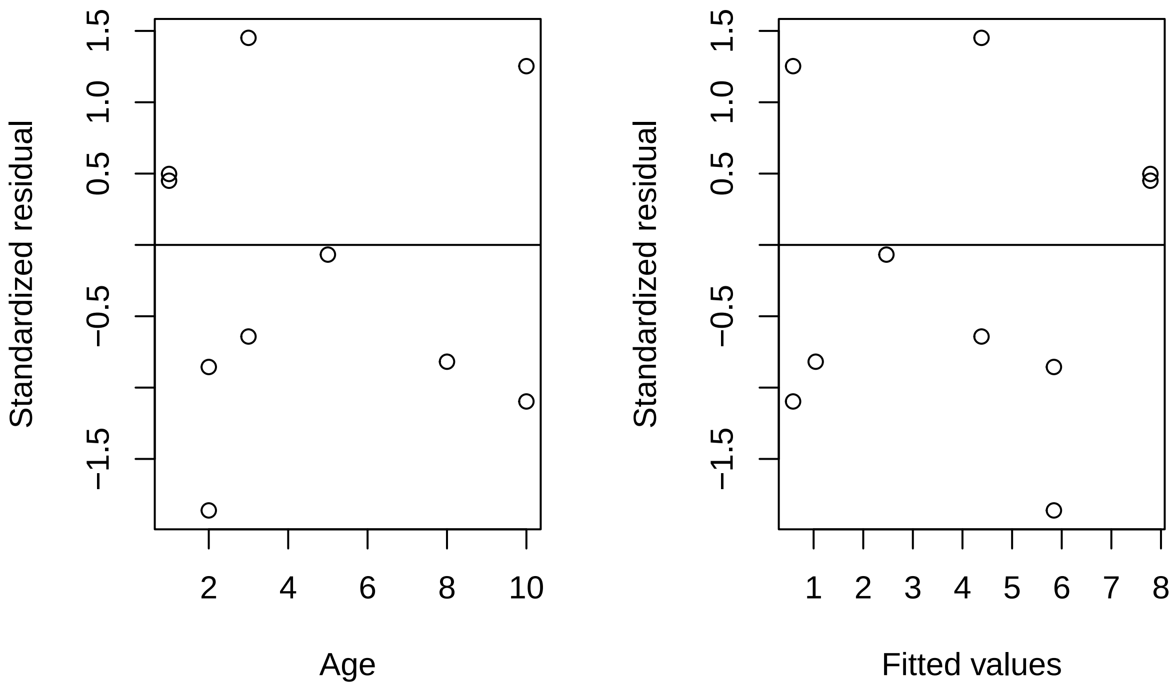
```
qr.area_age_glm <- qresid(area_age_glm)
qqnorm(qr.area_age_glm, las = 1)
qqline(qr.area_age_glm)
```



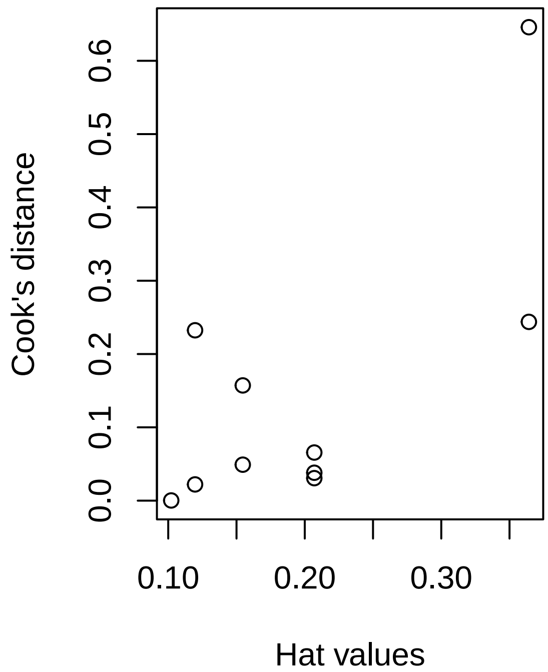

```r
r<-residuals(area_age_glm,type="pearson")
modfit<-fitted(area_age_glm)

plot(r~modfit, main = "Pearson's Residuals")
```



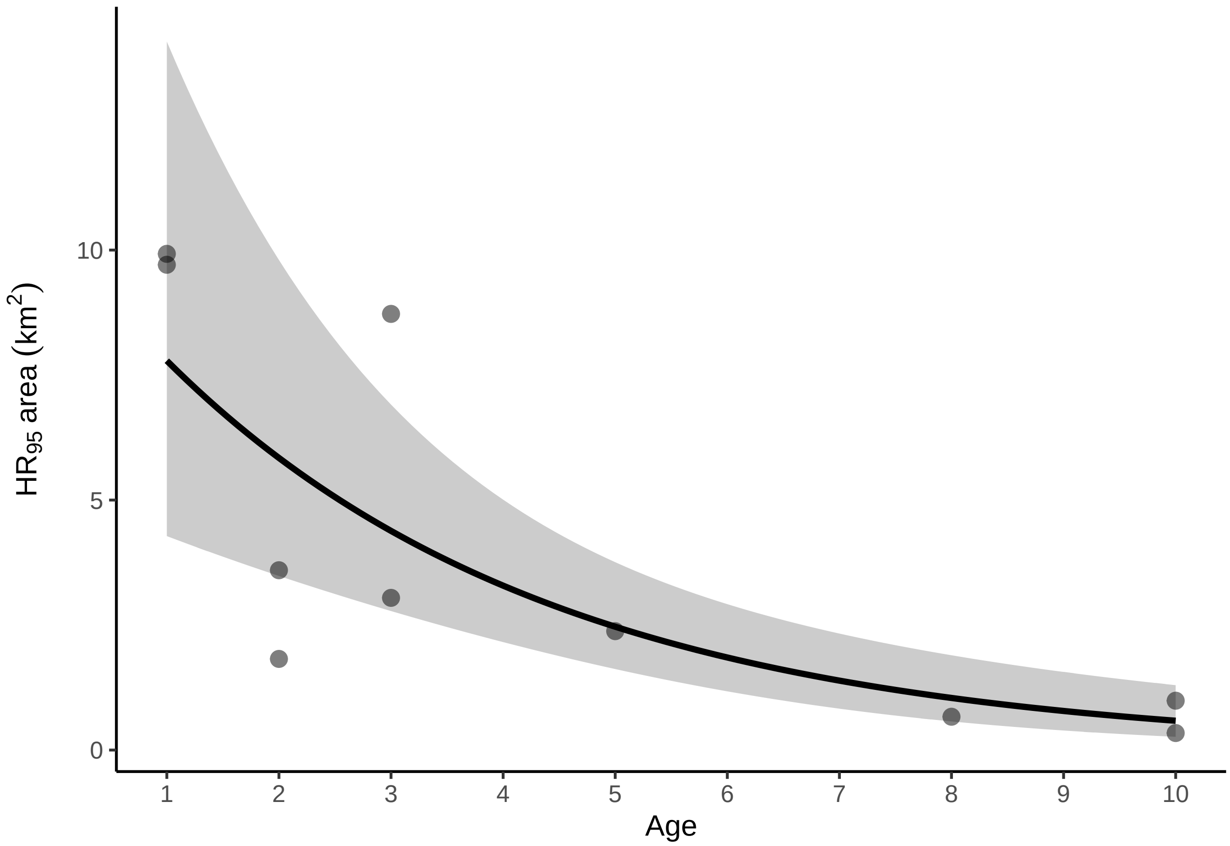
```r
par(mfrow=c(1,2)) # change plot window to accommodate 2 plots

r = rstandard(area_age_glm)
lf <- fitted(area_age_glm)
plot(all_contours_95_df$Age, r,xlab="Age",
     ylab="Standardized residual")
abline(h=0)

plot(lf, r,xlab="Fitted values",ylab="Standardized residual")
abline(h=0)
```




```r
h <- hatvalues(area_age_glm)
cd <- cooks.distance(area_age_glm)
plot(h,cd,xlab="Hat values",ylab="Cook's distance")

par(mfrow=c(1,1)) # return to single plotting
```




```
area_age_glm_plot <- effect_plot(area_age_glm,
                                 pred = Age,
                                 plot.points = T,
                                 interval = T,
                                 point.size = 2.5,
                                 point.alpha = 0.5) +
  scale_x_continuous("Age", breaks = c(1:10)) +
  ylab(expression(HR[95]~area~(km^2))) +
  theme_classic() +
  theme(axis.title.y = element_text(margin = margin(r = 10)))

area_age_glm_plot
```




```
for(i in 1:length(filetypes)){
ggsave(filename = paste0("Graphical outputs/",
                         "age_area_glm_", Sys.Date(), ".",
                         filetypes[i]),
       width = 160,
       height = 100,
       units = "mm",
       dpi = 800)
}
```

We also check the diagnostics and create a plot for the familiarity model.

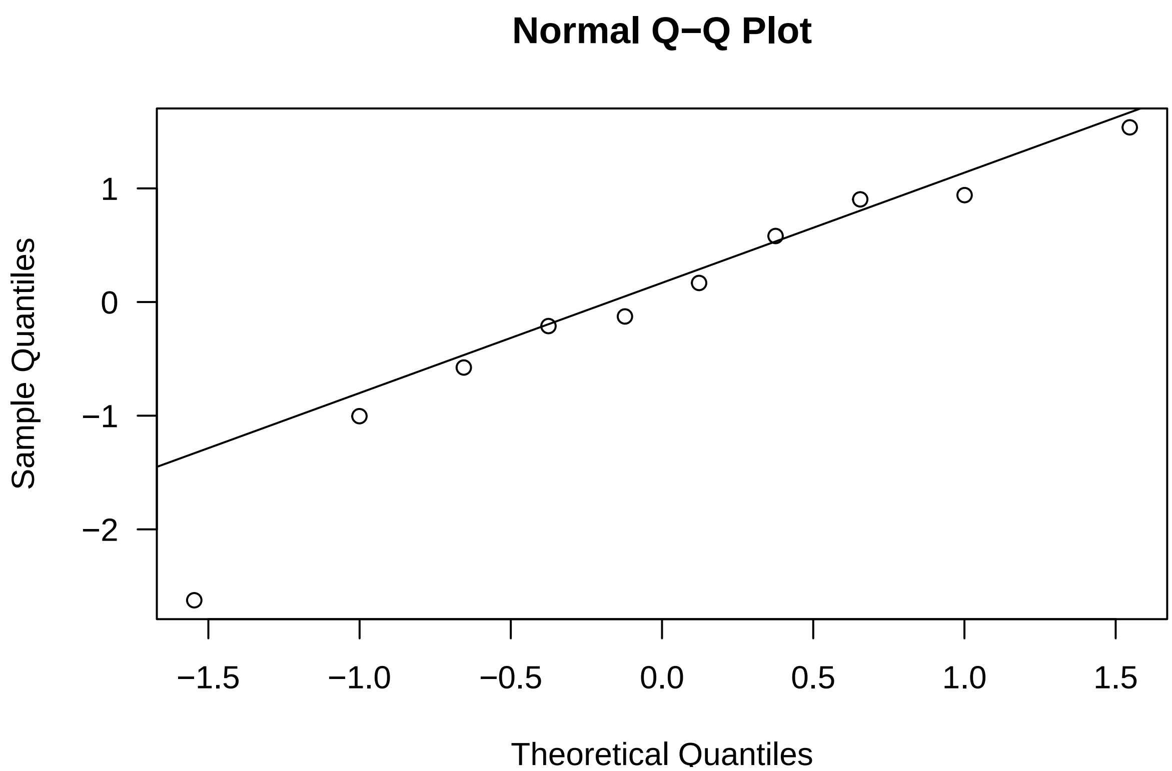
```
dfun <- function(object) {
  with(object,sum((weights * residuals^2)[weights > 0])/df.residual)
}

dfun(area_familiarity_glm)
```

```
## [1] 0.4752928
```

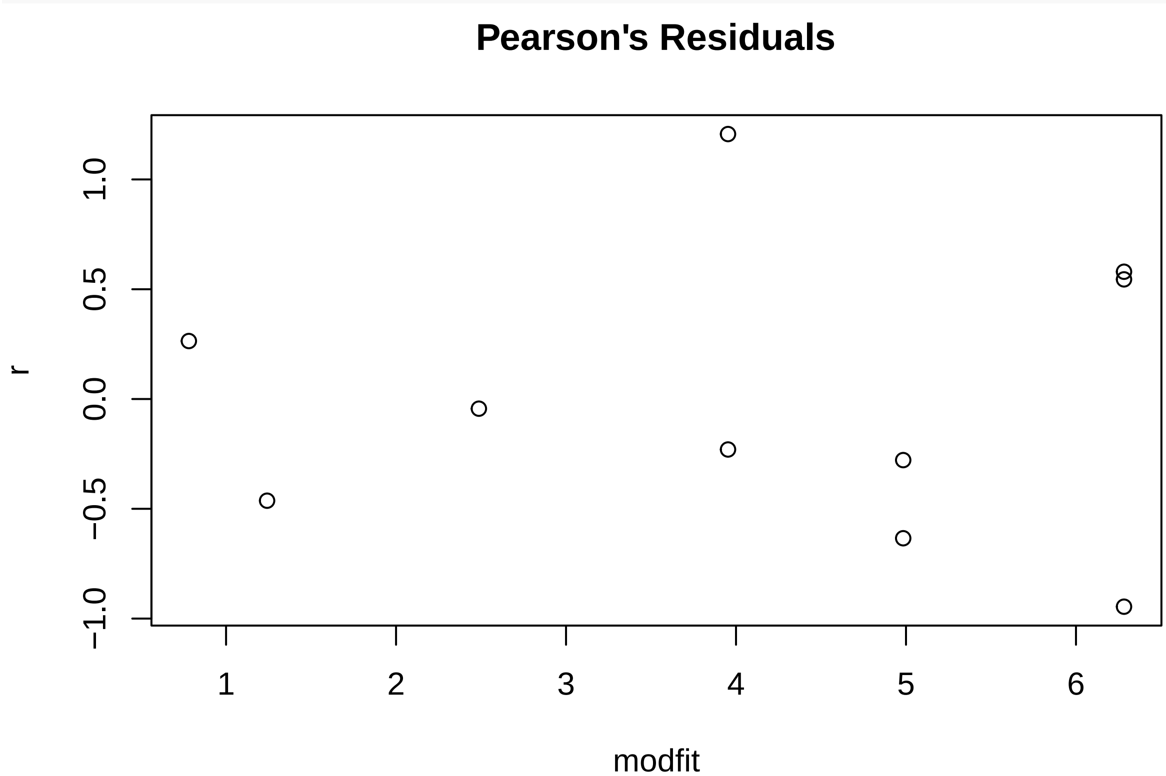
```
pseudoR2 <- 1 - (area_familiarity_glm$deviance / area_familiarity_glm$null.deviance)
pseudoR2
```

```
## [1] 0.3560283
```

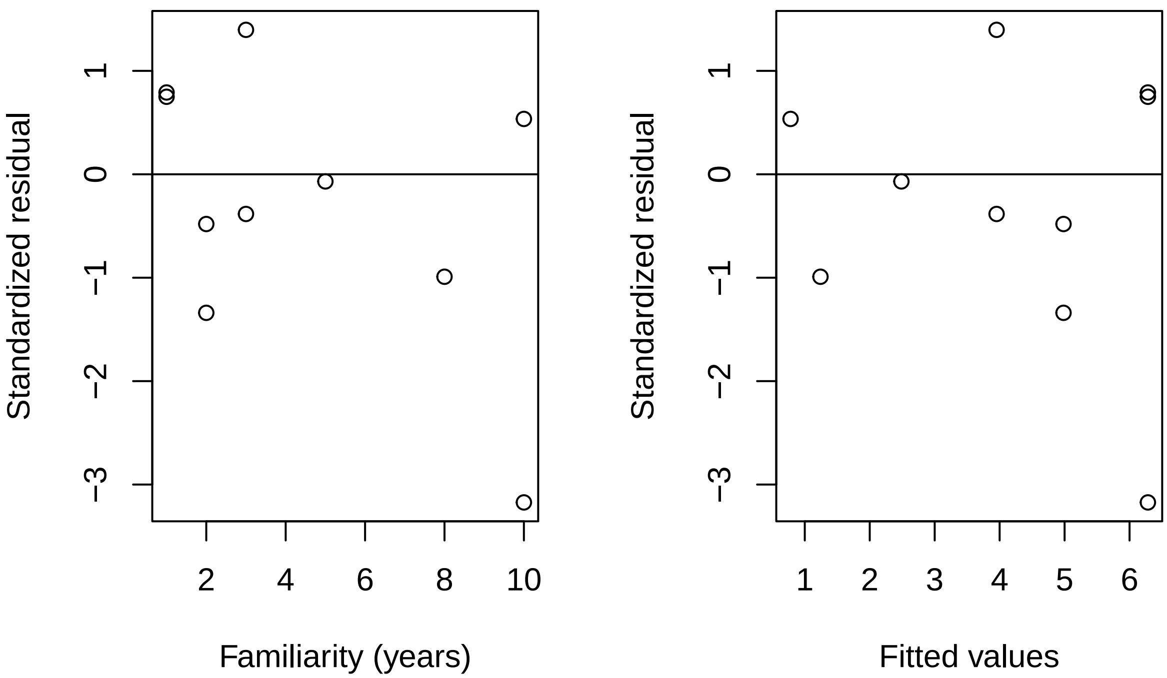
```
qr.area_familiarity_glm <- qresid(area_familiarity_glm)
qqnorm(qr.area_familiarity_glm, las = 1)
qqline(qr.area_familiarity_glm)
```



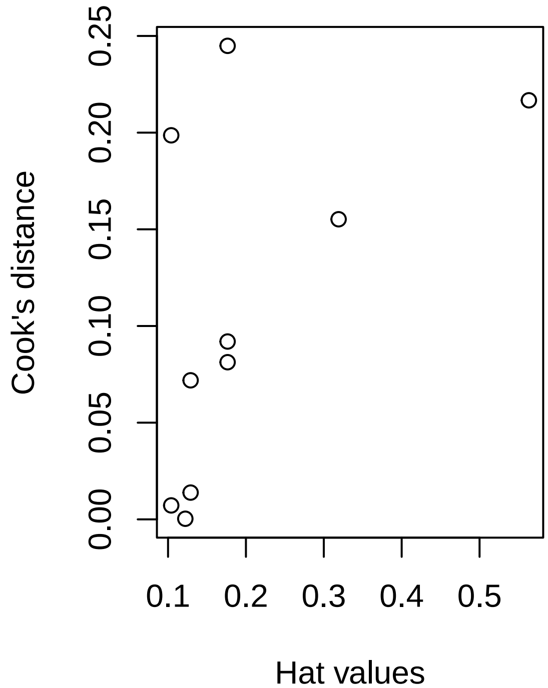

```
r<-residuals(area_familiarity_glm,type="pearson")
modfit<-fitted(area_familiarity_glm)

plot(r~modfit, main = "Pearson's Residuals")
```



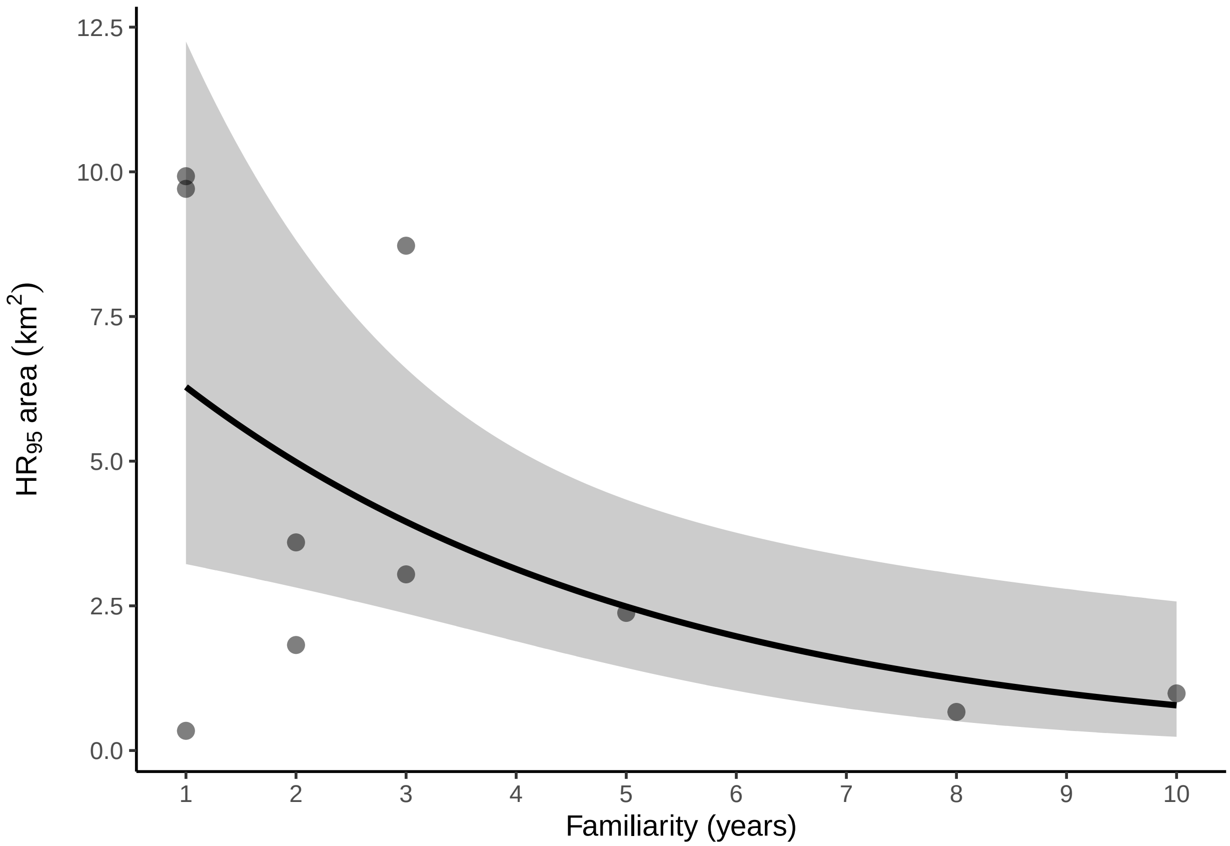
```r
par(mfrow=c(1,2)) # change plot window to accommodate 2 plots

r = rstandard(area_familiarity_glm)
lf <- fitted(area_familiarity_glm)
plot(all_contours_95_df$Age, r,xlab="Familiarity (years)",
     ylab="Standardized residual")
abline(h=0)

plot(lf, r,xlab="Fitted values",ylab="Standardized residual")
abline(h=0)
```




```r
h <- hatvalues(area_familiarity_glm)
cd <- cooks.distance(area_familiarity_glm)
plot(h,cd,xlab="Hat values",ylab="Cook's distance")

par(mfrow=c(1,1)) # return to single plotting
```




```
area_familiarity_glm_plot <- effect_plot(area_familiarity_glm,
                                 pred = Familiarity,
                                 plot.points = T,
                                 interval = T,
                                 point.size = 2.5,
                                 point.alpha = 0.5) +
  scale_x_continuous("Familiarity (years)", breaks = c(1:10)) +
  ylab(expression(HR[95]~area~(km^2))) +
  theme_classic() +
  theme(axis.title.y = element_text(margin = margin(r = 10)))

area_familiarity_glm_plot
```

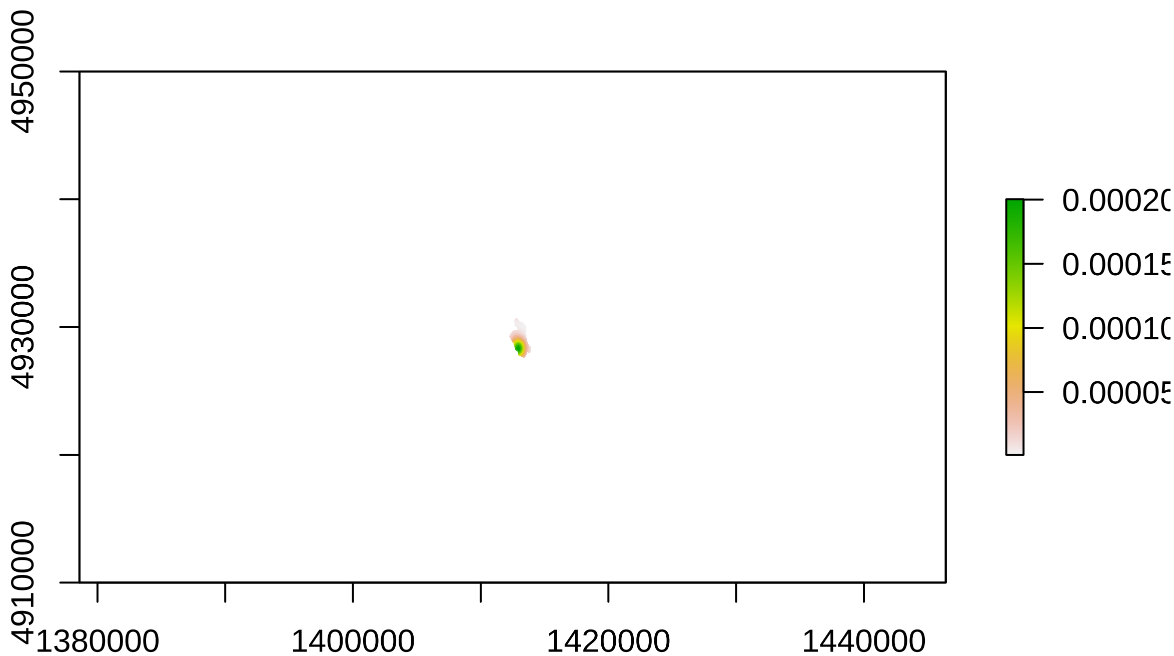



```
for(i in 1:length(filetypes)){
ggsave(filename = paste0("Graphical outputs/",
                         "age_familiarity_glm_", Sys.Date(), ".",
                         filetypes[i]),
       width = 160,
       height = 100,
       units = "mm",
       dpi = 800)
}
```

Calculate the overlap between the UD and the Orokonui Ecosanctuary fence

For checking the overlap between the fence of Orokonui and each UD. This calculation is a *summation of the values inside the cells* (i.e. the probability values), rather than the number of cells, as we want to approximate the time each kākā spent inside the sanctuary.

```
raster_UD_list <- map(UD1w_pHREML_list, raster, DF = "PMF")
# for(i in 1:10) raster::plot(raster_UD_list[[i]])

# checking that the cell values add up to 1
# for(i in 1:10) print(cellStats(raster_UD_list[[i]], sum))

# testing for a single UD
plot(mask(raster_UD_list[[1]], OrokonuiFence))
```




```
cellStats(mask(raster_UD_list[[1]], OrokonuiFence), sum)
```

```
## [1] 0.2506134
```

```
inside_overlap <- vector(mode = "numeric", length = 10)

for (i in 1:10){
inside_overlap[[i]] <- cellStats(mask(raster_UD_list[[i]], OrokonuiFence), sum)
print(inside_overlap[[i]])
}
```

```
## [1] 0.2506134
## [1] 0.9458736
## [1] 0.9070769
## [1] 0.4071398
## [1] 0.7316322
## [1] 0.1868212
## [1] 0.3452734
## [1] 0.1680315
## [1] 0.977431
## [1] 0.969272
```

```
# adding to data frame with inside and outside overlap
all_contours_95_df <- all_contours_95_df %>%
  mutate(inside = inside_overlap, outside = 1 - inside_overlap)

all_contours_95_df
```

```
##       ID Age Familiarity Sex   Origin  UD50_area UD95_area UD50_area_km2
## 1  45505   1           1   M Orokonui 1860836.70 9924325.7    1.86083670
## 2  45506  10          10   M Orokonui  145453.66  987249.6    0.14545366
## 3  45507   5           5   M  Captive  251523.82 2377871.7    0.25152382
## 4  45508   1           1   F Orokonui 1902418.95 9705908.9    1.90241895
## 5  45509   3           3   F Orokonui  241208.96 3044136.8    0.24120896
## 6  45510   2           2   F Orokonui  190743.30 1823074.1    0.19074330
```

```
## 7  45511   2          2   F Orokonui  701001.13 3596511.3    0.70100113
## 8  45512   3          3   M  Captive 2209902.46 8723160.6    2.20990246
## 9  45513  10          1   M  Captive   31690.85  340252.4    0.03169085
## 10 45514   8          8   F Orokonui   79866.68  666430.1    0.07986668
##    UD95_area_km2                  age_group    inside    outside
## 1      9.9243257 3 years or younger (n = 6) 0.2506134 0.74938663
## 2      0.9872496   5 years or older (n = 4) 0.9458736 0.05412637
## 3      2.3778717   5 years or older (n = 4) 0.9070769 0.09292314
## 4      9.7059089 3 years or younger (n = 6) 0.4071398 0.59286017
## 5      3.0441368 3 years or younger (n = 6) 0.7316322 0.26836781
## 6      1.8230741 3 years or younger (n = 6) 0.1868212 0.81317883
## 7      3.5965113 3 years or younger (n = 6) 0.3452734 0.65472657
## 8      8.7231606 3 years or younger (n = 6) 0.1680315 0.83196848
## 9      0.3402524   5 years or older (n = 4) 0.9774310 0.02256905
## 10     0.6664301   5 years or older (n = 4) 0.9692720 0.03072801
```

```r
max(all_contours_95_df$UD95_area_km2) / min(all_contours_95_df$UD95_area_km2)
```

```
## [1] 29.16754
```

```r
all_contours_95_df %>% group_by(age_group) %>%
  summarise(mean_UD50km2 = mean(UD50_area_km2),
            mean_UD95km2 = mean(UD95_area_km2),
            meanUD_outside = mean(outside),
            sd_UD50km2 = sd(UD50_area_km2),
            sd_UD95km2 = sd(UD95_area_km2),
            sdUD_outside = sd(outside))
```

```
## # A tibble: 2 x 7
##   age_group       mean_UD50km2 mean_UD95km2 meanUD_outside sd_UD50km2 sd_UD95km2
##   <chr>                  <dbl>        <dbl>          <dbl>      <dbl>      <dbl>
## 1 3 years or you~        1.18          6.14         0.652      0.909       3.70
## 2 5 years or old~        0.127         1.09         0.0501     0.0951      0.896
## # i 1 more variable: sdUD_outside <dbl>
```

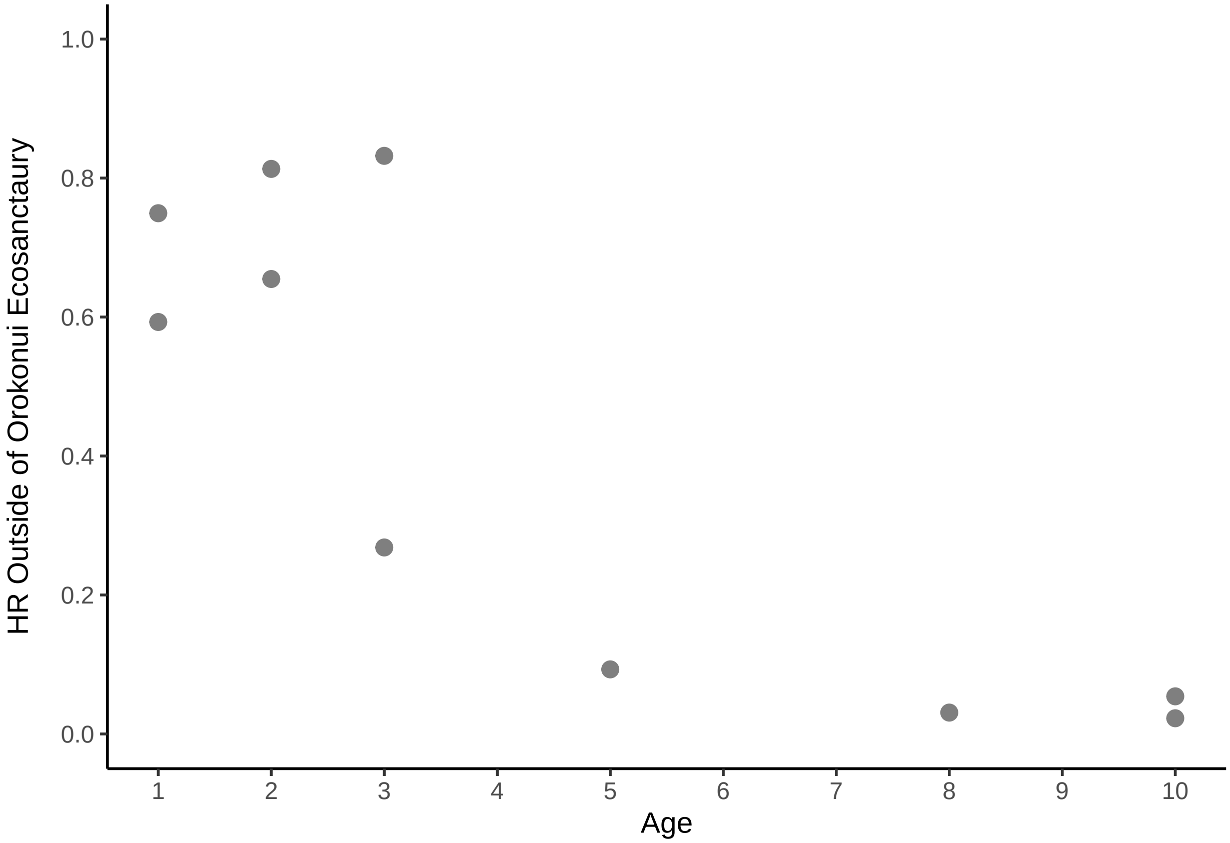
```r
all_contours_95_df %>% summarise(mean_UD50km2 = mean(UD50_area_km2),
                                 mean_UD95km2 = mean(UD95_area_km2),
                                 meanUD_inside = mean(inside),
                                 meanUD_outside = mean(outside),
                                 sd_UD50km2 = sd(UD50_area_km2),
                                 sd_UD95km2 = sd(UD95_area_km2),
                                 sdUD_inside = sd(inside),
                                 sdUD_outside = sd(outside))
```

```
##   mean_UD50km2 mean_UD95km2 meanUD_inside meanUD_outside sd_UD50km2 sd_UD95km2
## 1    0.7614647     4.118892     0.5889165      0.4110835  0.8721026   3.827558
##   sdUD_inside sdUD_outside
## 1   0.3480588    0.3480588
```

Plotting home range area, individual-level covariates and overlap with Orokonui Ecosanctuary

```r
all_contours_95_df %>% ggplot(aes(x = age, y = outside)) +
  geom_point(size = 2.5, alpha = 0.5) +
  scale_x_continuous(breaks = c(1:10), "Age") +
  scale_y_continuous(breaks = seq(0,1,0.2), limits = c(0,1),
                     "HR Outside of Orokonui Ecosanctaury") +
```

```r
  theme_classic() +
  theme(axis.title.y = element_text(margin = margin(r = 10)))
```



Statistical analysis using the same methodology as above, for the *proportion of area that lies outside of the Orokonui Ecosanctuary fence.*

```r
outsideglm <- glm(outside ~ age,
                  data = all_contours_95_df,
                  family = Gamma(link = "log"))

summary(outsideglm)
```

```
##
## Call:
## glm(formula = outside ~ age, family = Gamma(link = "log"), data = all_contours_95_df)
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.23668    0.29713   0.797  0.44869
## age         -0.37045    0.05277  -7.020  0.00011 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 0.3188882)
##
##     Null deviance: 13.2352  on 9  degrees of freedom
## Residual deviance:  2.4075  on 8  degrees of freedom
```

```
## AIC: -10.475
##
## Number of Fisher Scoring iterations: 5
```

```r
AIC(outsideglm)
```

```
## [1] -10.47522
```

```r
anova(outsideglm,test="F")
```

```
## Analysis of Deviance Table
##
## Model: Gamma, link: log
##
## Response: outside
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev      F    Pr(>F)
## NULL                     9    13.2352
## age   1   10.828         8     2.4075 33.955 0.0003929 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
r.squaredLR(outsideglm)
```

```
## [1] 0.8466956
## attr(,"adj.r.squared")
## [1] 4.156123
```

```r
confint(outsideglm)
```

```
## Waiting for profiling to be done...
```

```
##                  2.5 %     97.5 %
## (Intercept) -0.3063422  0.8467735
## age         -0.4679829 -0.2638832
```

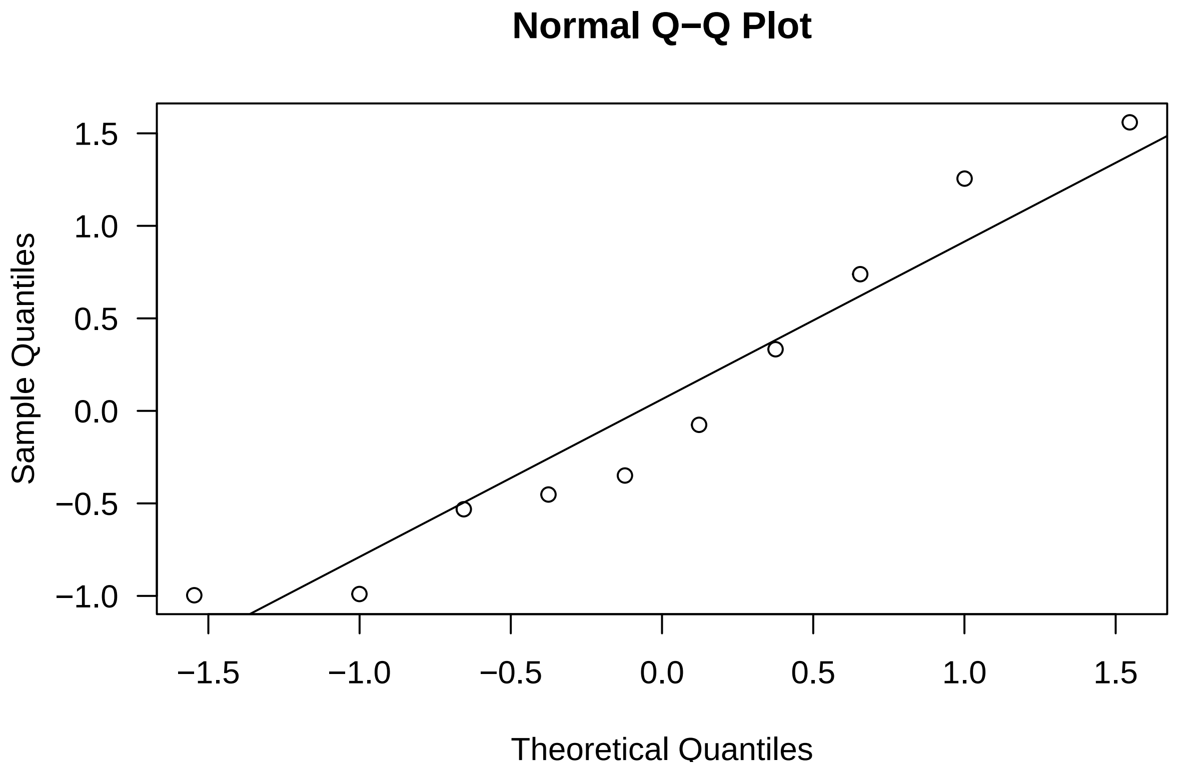
```r
dfun <- function(object) {
  with(object,sum((weights * residuals^2)[weights > 0])/df.residual)
}

dfun(outsideglm)
```

```
## [1] 0.3188882
```

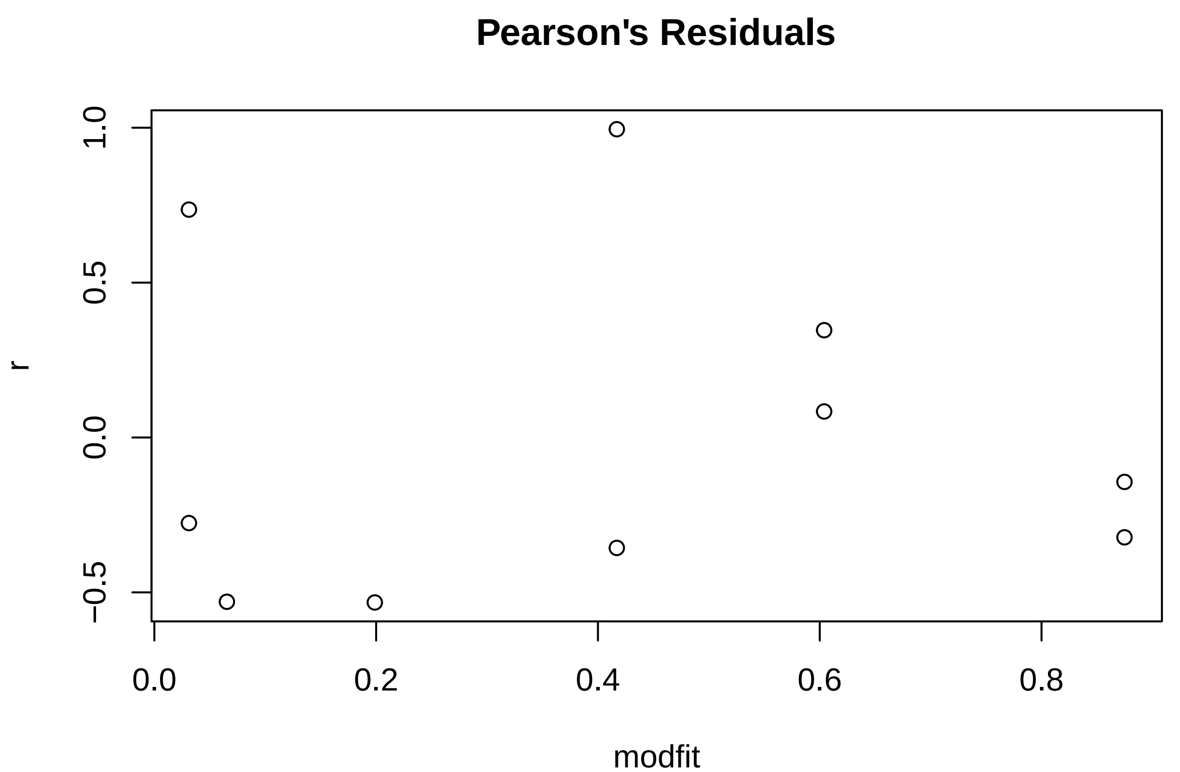
```r
pseudoR2 <- 1 - (outsideglm$deviance / outsideglm$null.deviance)
pseudoR2
```

```
## [1] 0.8181013
```

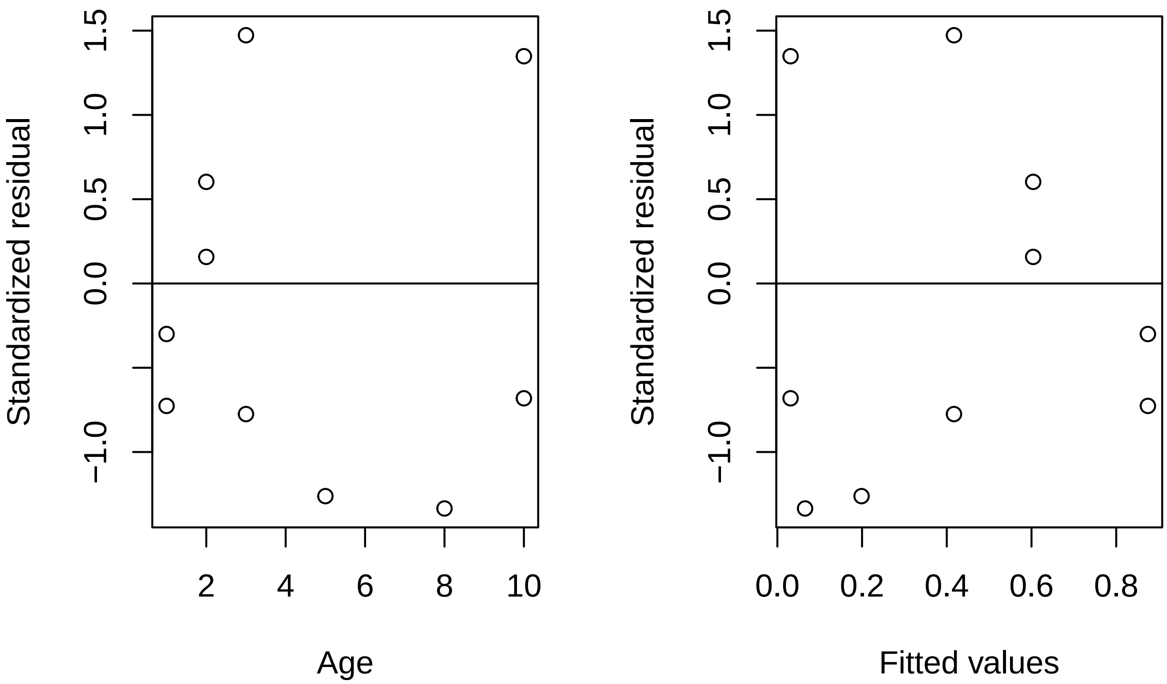
```r
qr.outsideglm <- qresid(outsideglm)
qqnorm(qr.outsideglm, las = 1)
qqline(qr.outsideglm)
```



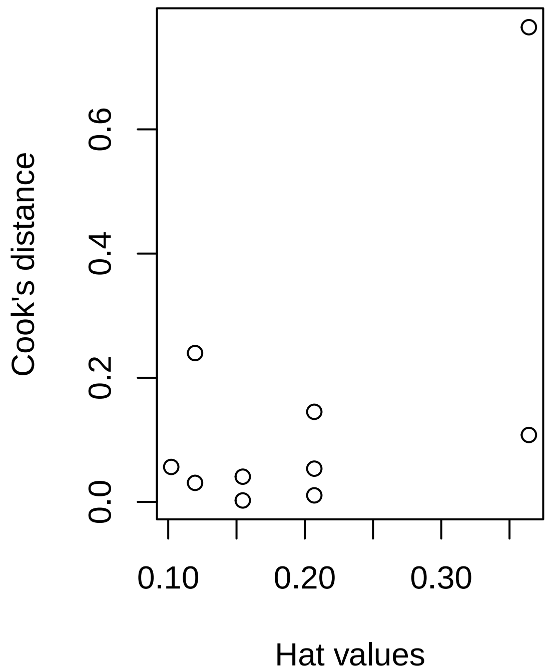

```r
r<-residuals(outsideglm,type="pearson")
modfit<-fitted(outsideglm)

plot(r~modfit, main = "Pearson's Residuals")
```



```r
par(mfrow=c(1,2)) # change plot window to accommodate 2 plots

r = rstandard(outsideglm)
lf <- fitted(outsideglm)
plot(all_contours_95_df$Age, r,xlab="Age",
     ylab="Standardized residual")
abline(h=0)

plot(lf, r,xlab="Fitted values",ylab="Standardized residual")
abline(h=0)
```

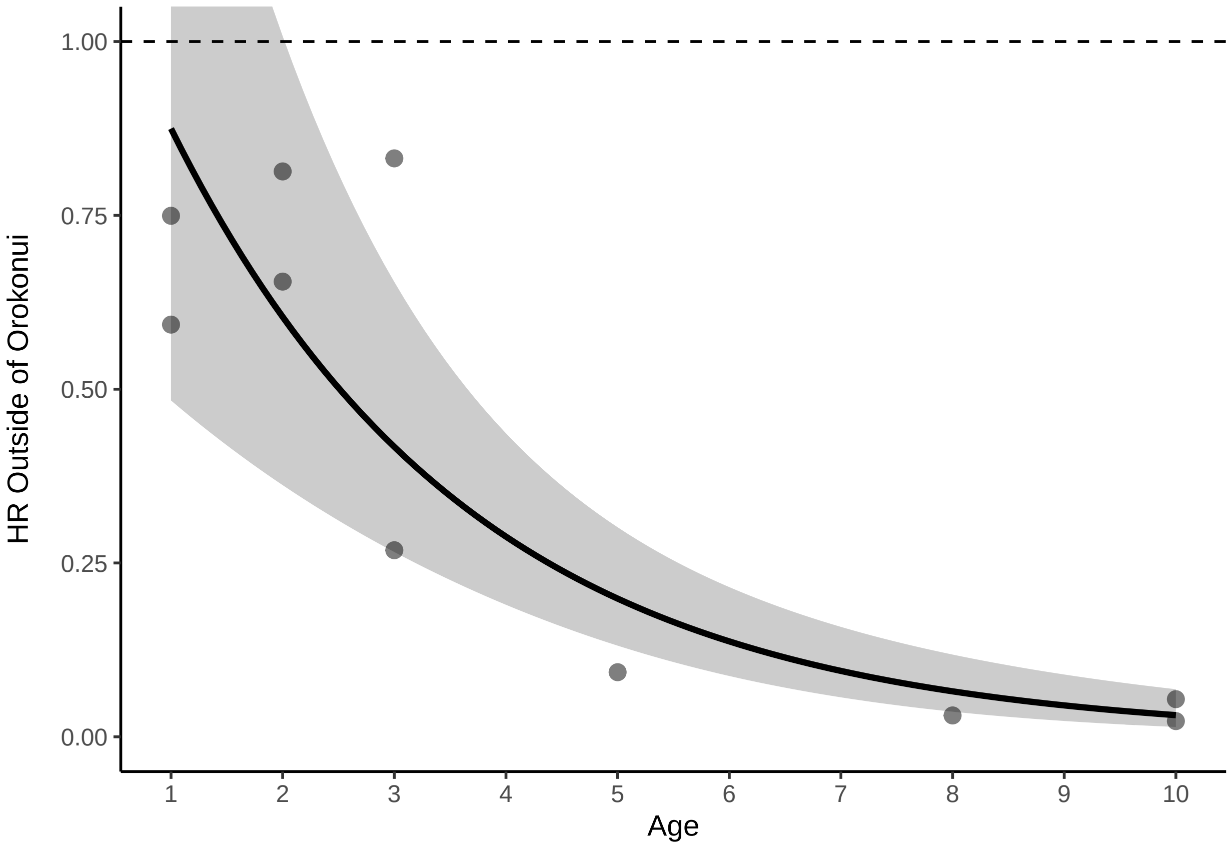



```r
h <- hatvalues(outsideglm)
cd <- cooks.distance(outsideglm)
plot(h,cd,xlab="Hat values",ylab="Cook's distance")
```

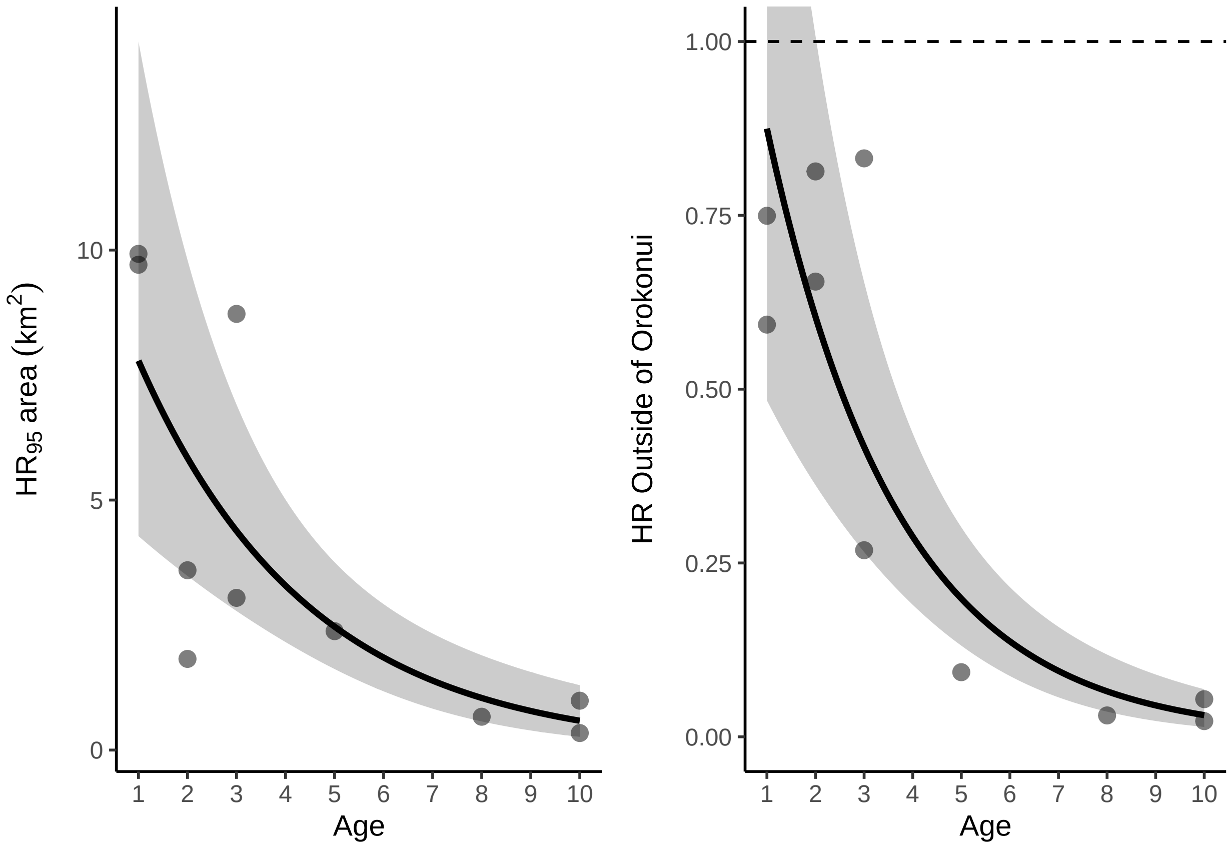


Plot the age model results and overlap model results together

```
par(mfrow=c(1,1)) # return to single plotting

outsideglmplot <- effect_plot(outsideglm,
                              pred = age,
                              plot.points = T,
                              interval = T,
                              point.size = 2.5,
                              point.alpha = 0.5) +
  scale_x_continuous(breaks = c(1:10)) +
  # scale_y_continuous(limits = c(0,1)) +
  coord_cartesian(ylim = c(0, 1), xlim = c(1,10)) +
  labs(x = "Age", y = "HR Outside of Orokonui") +
  geom_hline(yintercept = 1, linetype = "dashed") +
  theme_classic() +
  theme(axis.title.y = element_text(margin = margin(r = 10)))

outsideglmplot
```



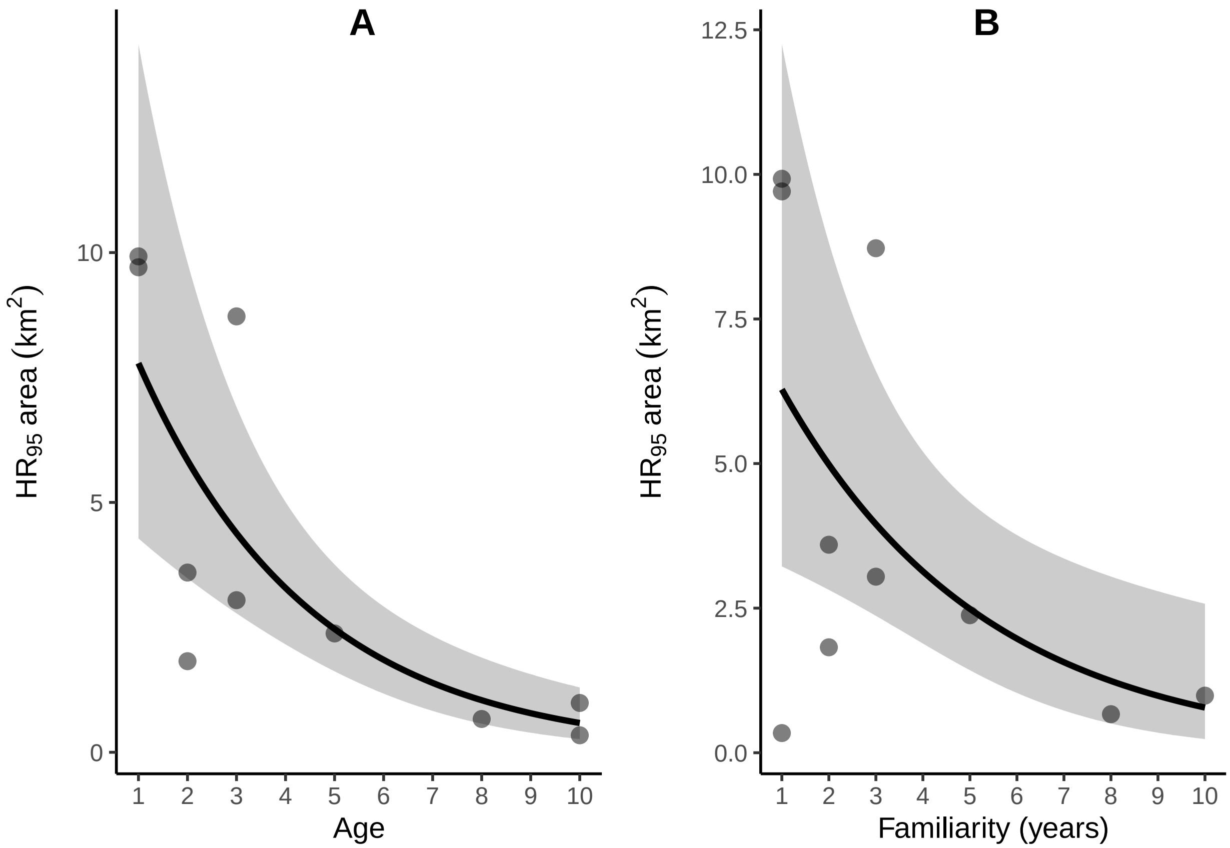

```
ggarrange(area_age_glm_plot, outsideglmplot) # , labels = "AUTO", label.x = 0.55
```




```
for(i in 1:length(filetypes)) {
ggsave(filename = paste0("Graphical outputs/",
                         "age_area_overlap_glms_", Sys.Date(), ".",
                         filetypes[i]),
       width = 180,
       height = 50,
       units = "mm",
       dpi = 800)
}
```

Compare the age model results and familiarity model results together

```
ggarrange(area_age_glm_plot, area_familiarity_glm_plot,
          labels = "AUTO", label.x = 0.55)
```




```
for(i in 1:length(filetypes)) {
ggsave(filename = paste0("Graphical outputs/",
                        "age_area_familiarity_glms_", Sys.Date(), ".",
                        filetypes[i]),
       width = 160,
       height = 100,
       units = "mm",
       dpi = 800)
}
```

References

Silva, Inês, Christen H Fleming, Michael J Noonan, Jesse Alston, Cody Folta, William F Fagan, and Justin M Calabrese. 2022. "Autocorrelation-Informed Home Range Estimation: A Review and Practical Guide." *Methods in Ecology and Evolution / British Ecological Society* 13 (3): 534–44. https://doi.org/10.1111/2041-210x.13786.

Session info

```
sessionInfo()
```

```
## R version 4.5.0 (2025-04-11)
## Platform: aarch64-apple-darwin20
## Running under: macOS Sequoia 15.4.1
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libRblas.0.dylib
```

```
## LAPACK: /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libRlapack.dylib;  LAPACK ve
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: Australia/Brisbane
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] terra_1.8-42     ctmm_1.2.0       statmod_1.5.0    MuMIn_1.48.11
##  [5] jtools_2.3.0     patchwork_1.3.0  ggpubr_0.6.0     sf_1.0-20
##  [9] here_1.0.1       lattice_0.22-6   move_4.2.6       raster_3.6-32
## [13] sp_2.2-0         geosphere_1.5-20 lubridate_1.9.4  forcats_1.0.0
## [17] stringr_1.5.1    dplyr_1.1.4      purrr_1.0.4      readr_2.1.5
## [21] tidyr_1.3.1      tibble_3.2.1     ggplot2_3.5.2    tidyverse_2.0.0
## [25] knitr_1.50
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.1    viridisLite_0.4.2   farver_2.1.2
##  [4] fastmap_1.2.0       digest_0.6.37       timechange_0.3.0
##  [7] lifecycle_1.0.4     ggspatial_1.1.9     magrittr_2.0.3
## [10] compiler_4.5.0      rlang_1.1.6         tools_4.5.0
## [13] utf8_1.2.5          yaml_2.3.10         ggsignif_0.6.4
## [16] labeling_0.4.3      classInt_0.4-11     xml2_1.3.8
## [19] RColorBrewer_1.1-3  abind_1.4-8         KernSmooth_2.23-26
## [22] expm_1.0-0          numDeriv_2016.8-1.1 withr_3.0.2
## [25] grid_4.5.0          stats4_4.5.0        e1071_1.7-16
## [28] future_1.49.0       globals_0.18.0      scales_1.4.0
## [31] tinytex_0.57        cli_3.6.5           rmarkdown_2.29
## [34] ragg_1.4.0          generics_0.1.3      parsedate_1.3.2
## [37] rstudioapi_0.17.1   httr_1.4.7          tzdb_0.5.0
## [40] DBI_1.2.3           cachem_1.1.0        proxy_0.4-27
## [43] pander_0.6.6        splines_4.5.0       parallel_4.5.0
## [46] vctrs_0.6.5         Matrix_1.7-3        carData_3.0-5
## [49] car_3.1-3           hms_1.1.3           rstatix_0.7.2
## [52] Formula_1.2-5       listenv_0.9.1       systemfonts_1.2.3
## [55] units_0.8-7         glue_1.8.0          parallelly_1.44.0
## [58] codetools_0.2-20    cowplot_1.1.3       stringi_1.8.7
## [61] gtable_0.3.6        broom.mixed_0.2.9.6 pillar_1.10.2
## [64] furrr_0.3.1         htmltools_0.5.8.1   R6_2.6.1
## [67] textshaping_1.0.1   rprojroot_2.0.4     evaluate_1.0.3
## [70] backports_1.5.0     memoise_2.0.1       broom_1.0.8
## [73] class_7.3-23        Rcpp_1.0.14         nlme_3.1-168
## [76] xfun_0.52           pkgconfig_2.0.3
```