Assessed Coursework #1: Packages

2023-11-02

In this assessed coursework, I go over the package development package for an R package I have created called `hull2spatial`. The coursework primarily focuses on R package development, while also touching on use of and integration with `tidyverse` and different types of object-oriented programming models in R.

`hull2spatial`

Over the last few years, I have been developing an R package called `hull2spatial` that can be used to convert objects created by the `alphahull` package into spatial objects that are compatible with the `sp` R package. The `hull2spatial` package can be found here: https://github.com/babichmorrowc/hull2spatial.

```
# devtools::install_github("https://github.com/babichmorrowc/hull2spatial")
library(hull2spatial)
```

Package motivation

Example data and cleaning

When modeling the distribution of a given species, ecologists typically have data on given occurrence points where the species has been found, e.g. latitude-longitude coordinates, and want to use those points to infer the region over which the species is distributed. For example, the following data is occurrence points of various species in the infraorder Mygalomorphae (a group of spiders).

```
library(here) # package for filepaths
library(arrow) # package to read data from parquet file
library(sf) # spatial package
library(tidyverse)
myg_spiders <- read_parquet(here("data/Mygalomorphae_withassertions_2023-09-18_ALA.parquet"))
myg_spiders_cleaned <- myg_spiders %>%
  filter(!is.na(decimalLatitude) & !is.na(decimalLongitude))  %>%
  filter(!duplicated(decimalLatitude) & !duplicated(decimalLongitude))
head(myg_spiders_cleaned)
```

```
## # A tibble: 6 x 146
##   decimalLatitude decimalLongitude eventDate           scientificName
##             <dbl>            <dbl> <dttm>              <chr>
## 1           -51.3             123. 2020-06-22 00:00:00 Idiosoma clypeatum
## 2           -51.1             149. 1998-04-21 00:00:00 Namea
## 3           -49.6             142. 2020-06-22 00:00:00 Idiosoma formosum
## 4           -47.1             123. 2020-06-21 00:00:00 Idiosoma formosum
## 5           -46.1             168. 1991-03-07 00:00:00 Arbanitis
## 6           -45.9             170. 1951-03-18 00:00:00 Porrhothele antipodiana
```

```
## # i 142 more variables: taxonConceptID <chr>, recordID <chr>,
## #   dataResourceName <chr>, occurrenceStatus <chr>, phylum <chr>, class <chr>,
## #   order <chr>, family <chr>, genus <chr>, species <chr>, taxonRank <chr>,
## #   taxonID <chr>, raw_scientificName <chr>, raw_vernacularName <chr>,
## #   raw_geodeticDatum <chr>, stateProvince <chr>, locality <chr>,
## #   coordinatePrecision <dbl>, coordinateUncertaintyInMeters <dbl>,
## #   basisOfRecord <chr>, institutionCode <chr>, datasetName <chr>, ...
```

We use `dplyr` to identify the species with the most occurrences in our dataset:

```
myg_spiders_cleaned %>%
  filter(!is.na(species)) %>%
  group_by(species) %>%
  summarise(n_occs = n()) %>%
  arrange(desc(n_occs)) %>%
  head()
```

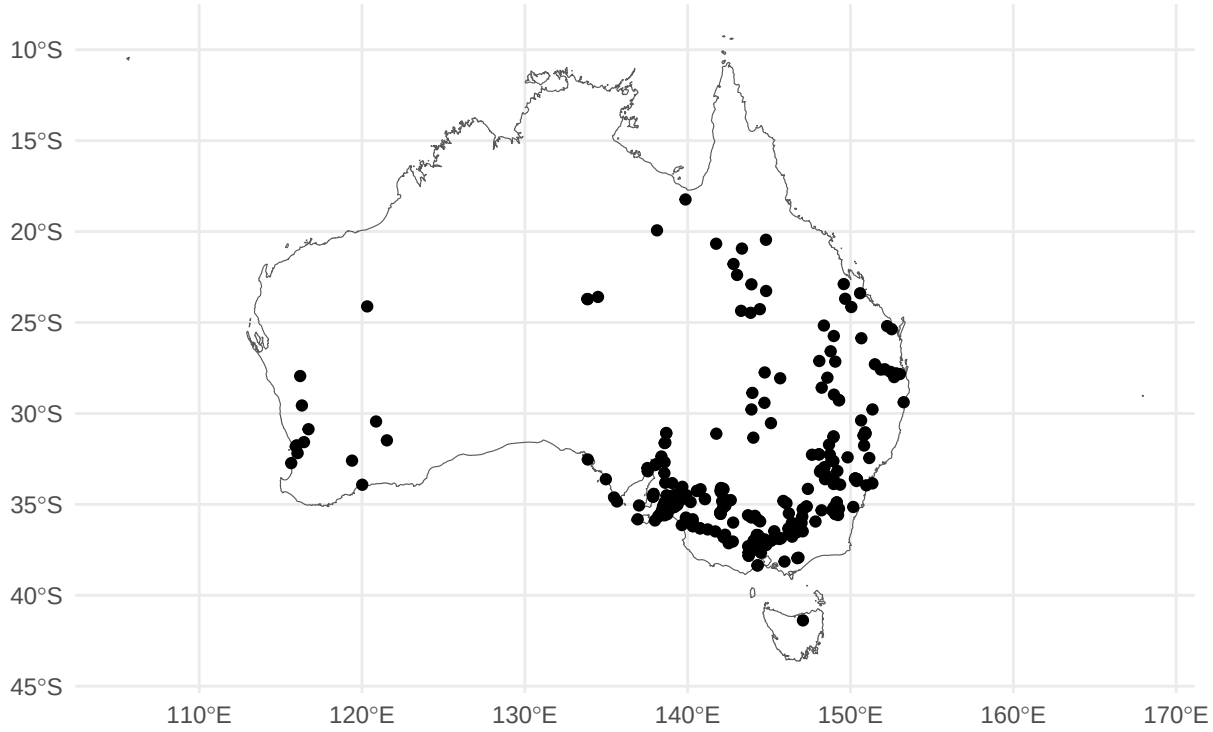
```
## # A tibble: 6 x 2
##   species             n_occs
##   <chr>                <int>
## 1 Missulena occatoria    418
## 2 Aname mellosa          177
## 3 Missulena bradleyi     176
## 4 Atrax robustus         171
## 5 Seqocrypta jakara       79
## 6 Hadronyche infensa      67
```

We will focus on the species *Missulena occatoria* and plot the occurrence points:

```
m_occatoria <- myg_spiders_cleaned %>%
  filter(species == "Missulena occatoria")

# Create map of Australia
# Transform projection
aus <- st_transform(ozmaps::ozmap_country, 4326)
# Base map
base_map <- ggplot() +
  geom_sf(data = aus, fill = NA) +
  theme_minimal()

base_map +
  geom_point(data = m_occatoria, aes(x = decimalLongitude, y = decimalLatitude)) +
  labs(x = "", y = "")
```



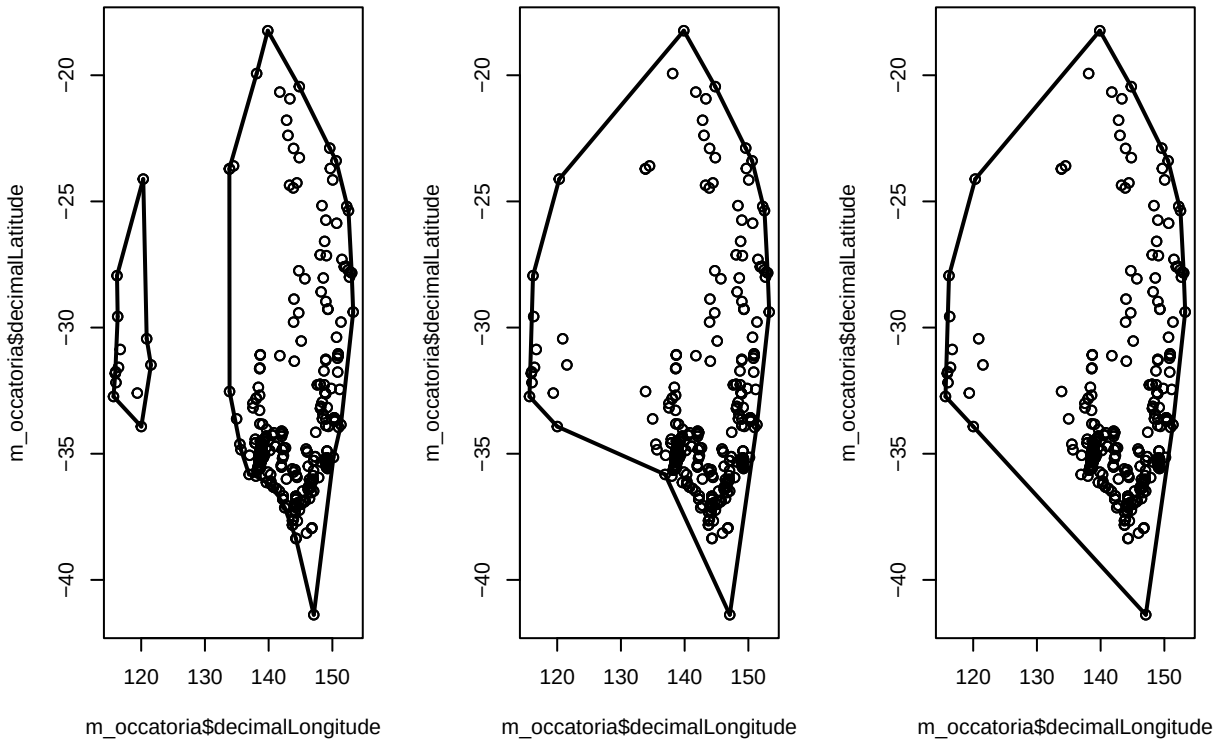
`alphahull` functions

α-shapes The `alphahull` R package provides functions for making two different types of shapes based on points: an α-shape and an α-hull. Both shapes are governed by a parameter α that determines how convex / concave the resulting shape is.

The following are some α-shapes based on the *M. occatoria* occurrence data:

```
library(alphahull)
alphashape_5 <- ashape(x = m_occatoria$decimalLongitude,
                       y = m_occatoria$decimalLatitude,
                       alpha = 5)
alphashape_20 <- ashape(x = m_occatoria$decimalLongitude,
                        y = m_occatoria$decimalLatitude,
                        alpha = 20)
alphashape_100 <- ashape(x = m_occatoria$decimalLongitude,
                         y = m_occatoria$decimalLatitude,
                         alpha = 100)
#  create a three-paneled figure
par(mfrow = c(1,3))
# Plot alpha = 5
plot(x = m_occatoria$decimalLongitude, y = m_occatoria$decimalLatitude)
plot(alphashape_5, add = T)
# Plot alpha = 20
plot(x = m_occatoria$decimalLongitude, y = m_occatoria$decimalLatitude)
```

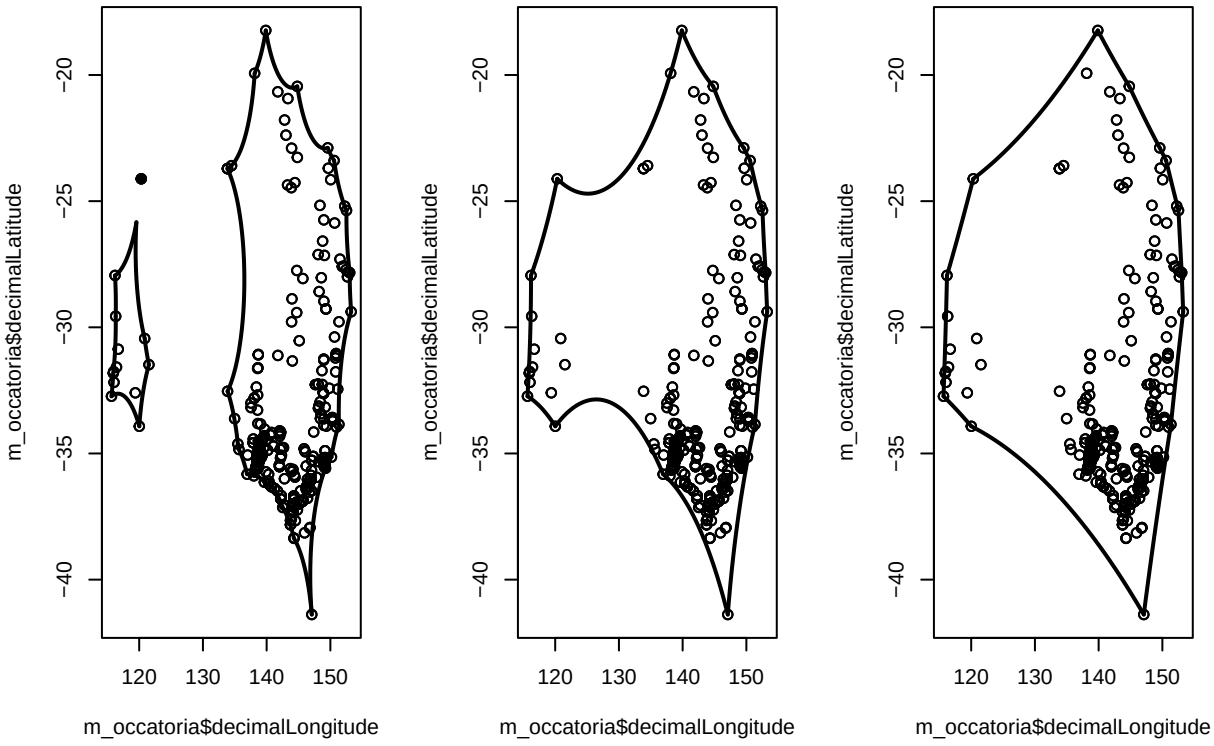
```
plot(alphashape_20, add = T)
# Plot alpha = 100
plot(x = m_occatoria$decimalLongitude, y = m_occatoria$decimalLatitude)
plot(alphashape_100, add = T)
```



α-hulls While α-shapes are comprised of straight lines, α-hulls consist of arcs. The following are α-hulls for the same values of α:

```
alphahull_5 <- ahull(x = m_occatoria$decimalLongitude,
                     y = m_occatoria$decimalLatitude,
                     alpha = 5)
alphahull_20 <- ahull(x = m_occatoria$decimalLongitude,
                      y = m_occatoria$decimalLatitude,
                      alpha = 20)
alphahull_100 <- ahull(x = m_occatoria$decimalLongitude,
                       y = m_occatoria$decimalLatitude,
                       alpha = 100)
#  create a three-paneled figure
par(mfrow = c(1,3))
# Plot alpha = 5
plot(x = m_occatoria$decimalLongitude, y = m_occatoria$decimalLatitude)
plot(alphahull_5, add = T)
# Plot alpha = 20
plot(x = m_occatoria$decimalLongitude, y = m_occatoria$decimalLatitude)
```

```
plot(alphahull_20, add = T)
# Plot alpha = 100
plot(x = m_occatoria$decimalLongitude, y = m_occatoria$decimalLatitude)
plot(alphahull_100, add = T)
```



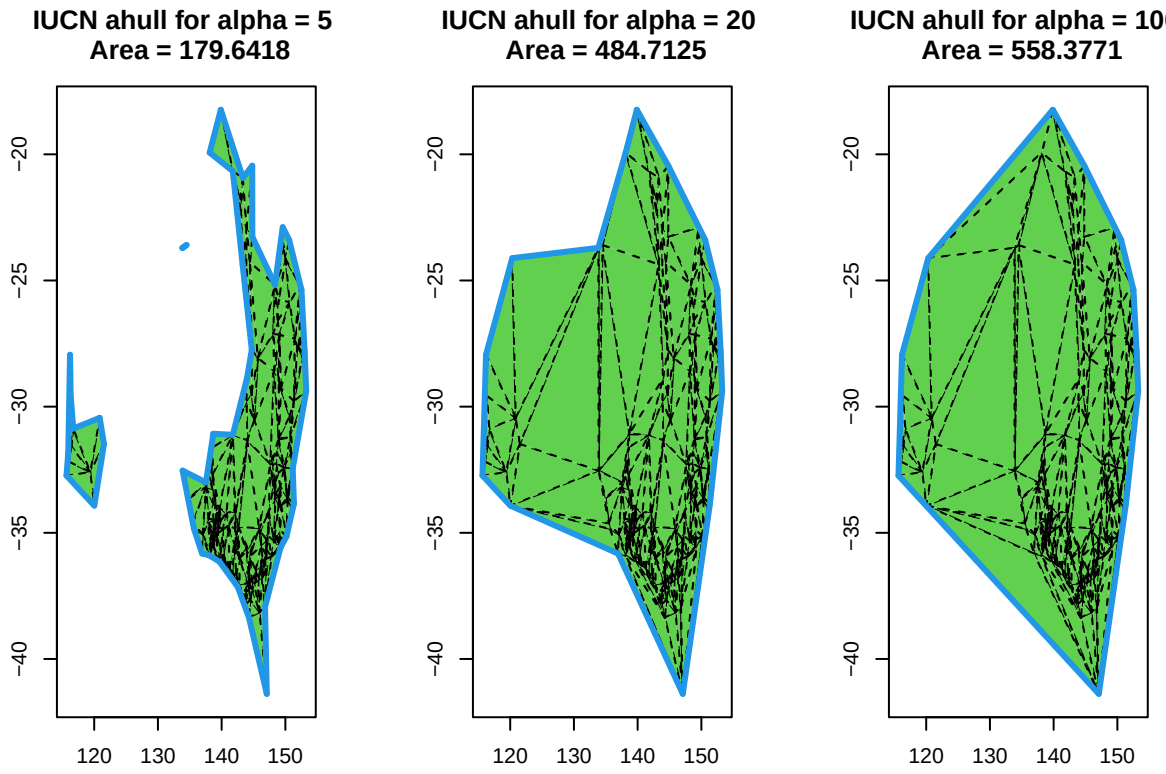
Recently, the developer of the `alphahull` package, Beatriz Pateiro, has been working on a new feature to create alpha-hull objects following the IUCN Red List method of creation. This new function is only available from the IUCN branch of the `alphahull` package:

```
# devtools::install_github("beatrizpateiro/alphahull", ref = "WIP-ahull.IUCN-feature")
library(alphahull)
```

The following plots show the IUCN α-hulls for 3 different values of α:

```
alphahull_iucn_5 <- ahull.IUCN(x = m_occatoria$decimalLongitude,
                               y = m_occatoria$decimalLatitude,
                               alpha = 5)
alphahull_iucn_20 <- ahull.IUCN(x = m_occatoria$decimalLongitude,
                                y = m_occatoria$decimalLatitude,
                                alpha = 20)
alphahull_iucn_100 <- ahull.IUCN(x = m_occatoria$decimalLongitude,
                                 y = m_occatoria$decimalLatitude,
                                 alpha = 100)
#  create a three-paneled figure
par(mfrow = c(1,3))
```

```
# Plot alpha = 5
plot(alphahull_iucn_5)
# Plot alpha = 20
plot(alphahull_iucn_20)
# Plot alpha = 100
plot(alphahull_iucn_100)
```



`hull2spatial` functions

When looking at the classes of objects created by `ashape`, `ahull`, and `ahull.IUCN`, note that these are not in a nice polygon form that integrates well with other R packages for spatial data:

```
class(alphashape_5)
```

```
## [1] "ashape"
```

```
class(alphahull_5)
```

```
## [1] "ahull"
```

```
class(alphahull_iucn_5)
```

```
## [1] "ahull.IUCN"
```

I wrote a set of functions that convert those objects into

ashape2poly

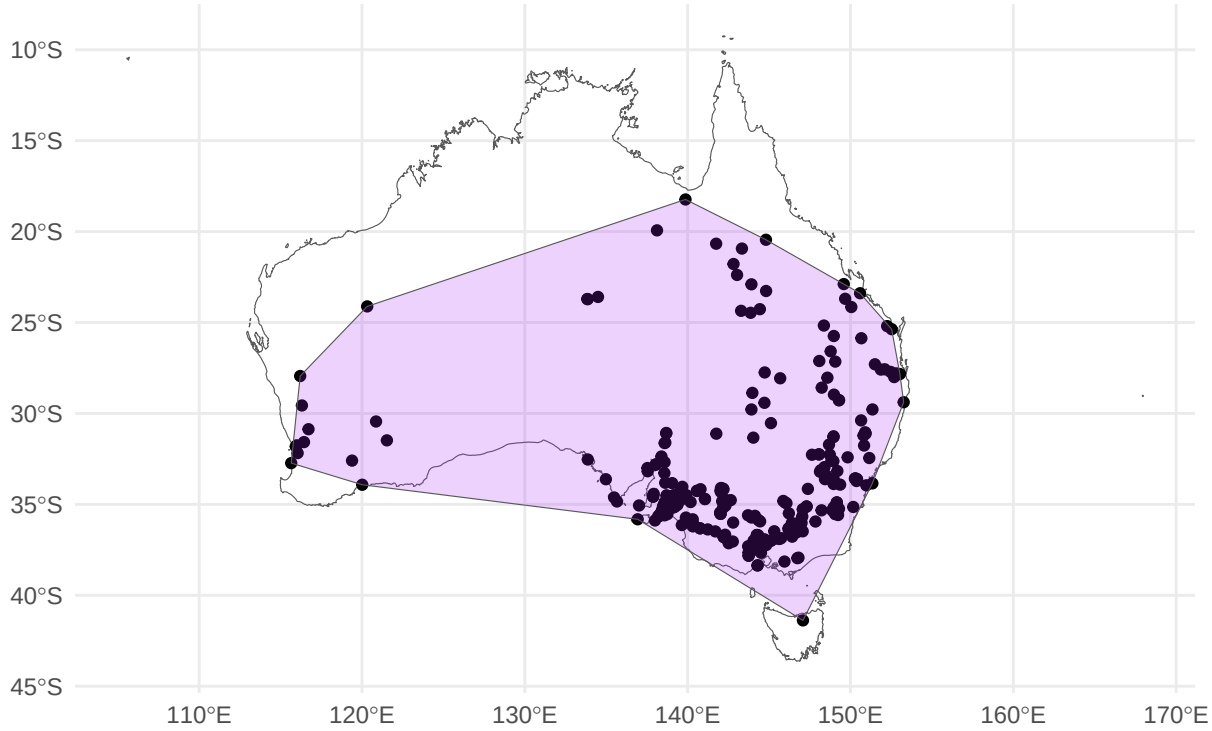
The `ashape2poly` function converts α-shape objects (of the class `ashape`) to `SpatialPolygons` objects:

```
alphashape_poly_20 <- ashape2poly(alphashape_20)
class(alphashape_poly_20)
```

```
## [1] "SpatialPolygons"
## attr(,"package")
## [1] "sp"
```

```
alphashape_poly_20_sf <- st_as_sf(alphashape_poly_20) %>%
  st_set_crs(4326)

base_map +
  geom_point(data = m_occatoria, aes(x = decimalLongitude, y = decimalLatitude)) +
  geom_sf(data = alphashape_poly_20_sf, fill = "purple", alpha = 0.2) +
  labs(x = "", y = "")
```



ahull2poly

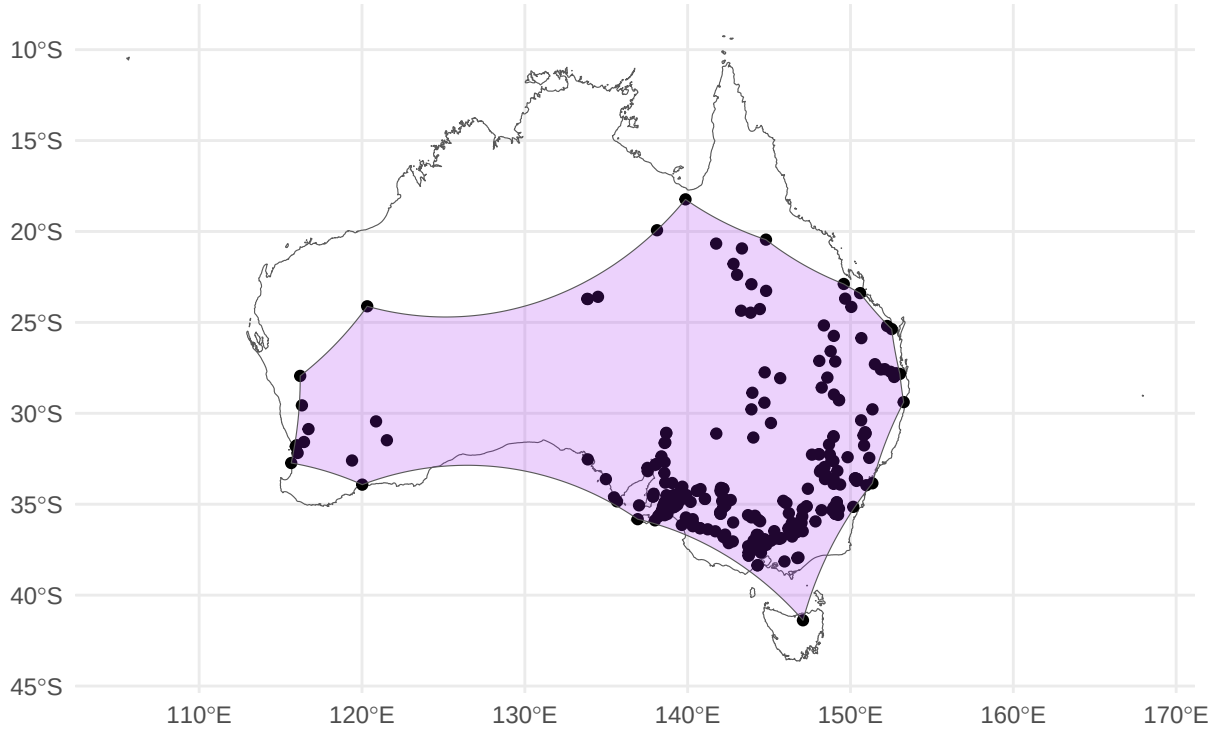
The `ahull2poly` function converts α-hull objects (of the class `ahull`) to `SpatialPolygons` objects:

```
alphahull_poly_20 <- ahull2poly(alphahull_20)
class(alphahull_poly_20)
```

```
## [1] "SpatialPolygons"
## attr(,"package")
## [1] "sp"
```

```
alphahull_poly_20_sf <- st_as_sf(alphahull_poly_20) %>%
  st_set_crs(4326)

base_map +
  geom_point(data = m_occatoria, aes(x = decimalLongitude, y = decimalLatitude)) +
  geom_sf(data = alphahull_poly_20_sf, fill = "purple", alpha = 0.2) +
  labs(x = "", y = "")
```



ahullIUCN2poly

In parallel with Pateiro's work on adding new functionality to the `alphahull` package to create IUCN α-hulls, I have created new functions to convert these objects to spatial-package-compatible objects as well. Over the course of the term, I have been working with an Australian scientist named Fonti Kar to collaborate on this new functionality for the package.

```
# devtools::install_github("babichmorrowc/hull2spatial", ref = "ahull_IUCN_wip")
library(hull2spatial)
```

I added two new functions to the package: `ahull.IUCN2lines` and `ahull.IUCN2poly`. The source code for these functions can be found here. These functions also give the flexibility to return objects compatible with either the `sp` package or the `sf` package. Both of these packages are commonly used for spatial analyses in R, with `sp` being the older package and `sf` being a more modern version.

To write these functions, I leveraged the structure of `ahull.IUCN` objects, which contain a two column matrix indicating which points in the occurrence dataset are connected by lines comprising the border of the resulting hull:

```
alphahull_iucn_20[["bd.ah.IUCN"]]
```

```
##       from  to
##  [1,]    1 309
##  [2,]   72   1
##  [3,]  309 374
##  [4,]  384 396
##  [5,]  374 384
##  [6,]  396 408
##  [7,]  408 416
##  [8,]  416 418
##  [9,]  417 405
## [10,]  418 417
## [11,]  306  72
## [12,]  403 405
## [13,]  403 383
## [14,]  306 333
## [15,]  383 333
```

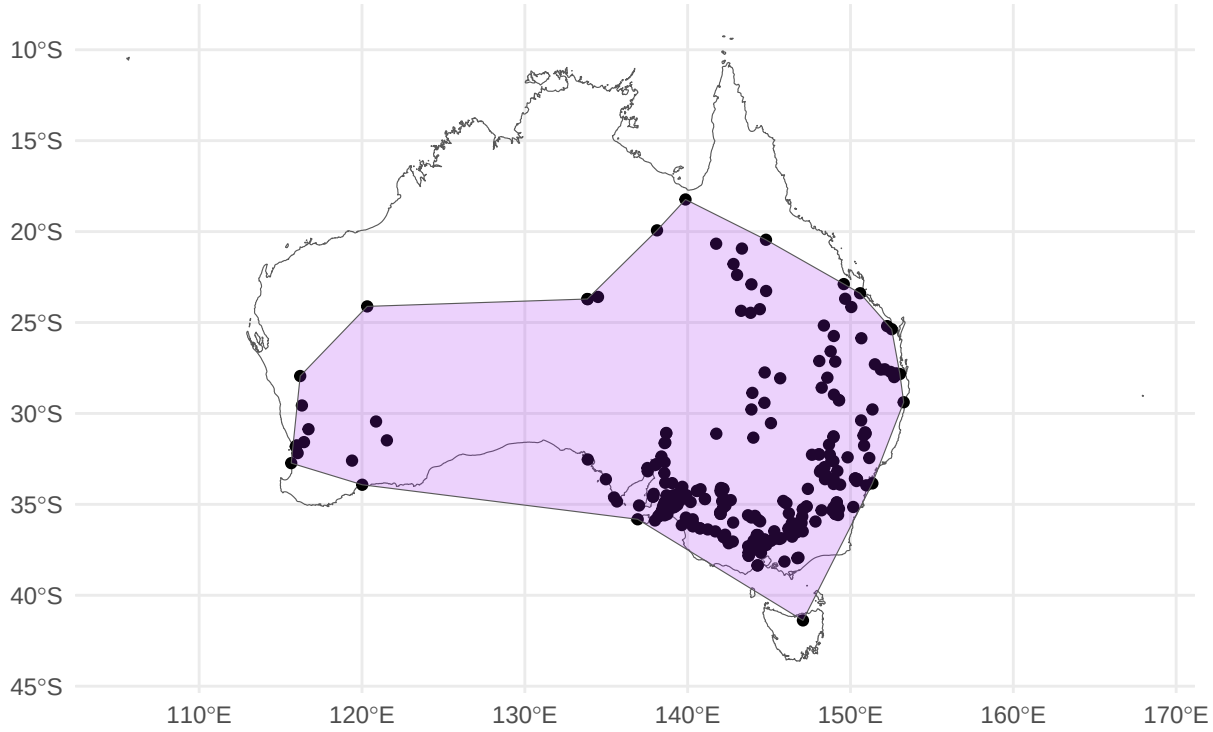
I used these coordinates to construct a set of lines around the border of the shape and then merged them using the `st_line_merge` function from the `sf` package (see code for `ahull.IUCN2lines`).

The following code demonstrates the use of `ahull.IUCN2poly` (which calls the `ahull.IUCN2lines` function):

```
alphahull_iucn_poly_20 <- ahull.IUCN2poly(alphahull_iucn_20, sp_or_sf = "sf") %>%
  st_set_crs(4326)
class(alphahull_iucn_poly_20)
```

```
## [1] "sf"         "data.frame"
```

```
base_map +
  geom_point(data = m_occatoria, aes(x = decimalLongitude, y = decimalLatitude)) +
  geom_sf(data = alphahull_iucn_poly_20, fill = "purple", alpha = 0.2) +
  labs(x = "", y = "")
```



Next steps

Next, I want to expand the functionality of the `ashape2poly` and `ahull2poly` functions to give compatibility with both the `sp` and `sf` packages as I did with the `ahull.IUCN2poly` function. The `sp` package came out in 2005 and is gradually being replaced by `sf` over time. `sp` uses S4 classes, whereas `sf` uses S3 classes. `sf` integrates well with the `tidyverse` (unlike `sp`), which is why I have primarily used `sf` objects to plot with `ggplot2` in the code above. In order to make `hull2spatial` more flexible and functional with the most up-to-date packages, I want to provide `sf` integration as a key functionality.

Additionally, I want to add more testing into `hull2spatial`. I have added several tests to the package using `testthat` (see tests here), but these primarily focus on testing the class of the output of the functions. I want to do more research into how to test functions with outputs that are spatial in nature.

Alongside these improvements, I will continue to work with Dr. Kar and Dr. Pateiro to align my work with the development being done on the `alphahull` package. My hope is that these functions will provide an easy way for ecologists to create a variety of range maps following the IUCN guidelins that can then be integrated with other commonly used R packages.