Supplementary Material

Spatially adaptive background delimitation strategies for species distribution models with unevenly surveyed occurrences

A. Márcia Barbosa

Spatial Biology Lab, Porto University, Portugal

This is a quick illustrated guide to using the techniques and reproducing the outputs shown in the *Biodiversity Informatics* article of the same title. The following commands (install and) load the required packages. The installation commands are commented-out, to avoid reinstalling every time the script is rerun, but users should un-comment and run them the first time, to install updated versions of the packages with all the necessary functionality:

```
# install.packages("fuzzySim")
# install.packages("terra")
# install.packages("geodata")
# install.packages("concaveman")

library("fuzzySim")
library("terra")
library("geodata")
library("concaveman")
```

The following commands set the working directory to the folder that contains this script; create a folder for the output files; and create the colour palette that we will use for the background polygons ahead:

```
# setwd(dirname(rstudioapi::getSourceEditorContext()$path))

dir.create("outputs/figures", recursive = TRUE, showWarnings = FALSE)
```

```
yellows <- colorRampPalette(c("#FFF9C4", "#FFF59D", "#FFEE58",
                              "#FDD835", "#FBC02D", "#F9A825"))
```

The commands below import a table with some species occurrence data from GBIF, perform a quick but safe data cleaning procedure, and save the table to the outputs folder. They are commented out because they only need to be run once for each analysis, and they may take long if there are many occurrence points. Mind that GBIF data should be properly cited - see e.g. the `?sp_occurrence` help file.

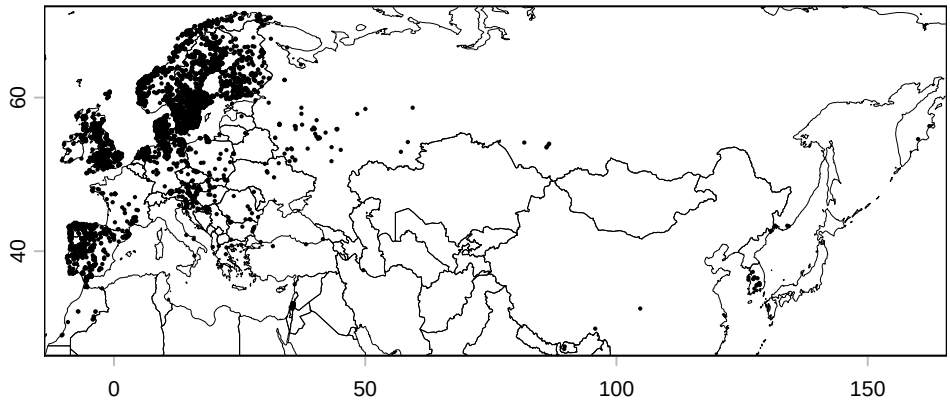
```
# occs <- geodata::sp_occurrence("Lutra", "lutra", args = c("year=2025"))
# # restricted year to limit download time for example dataset
#
# write.csv(occs, paste0("outputs/occs_raw.csv"), row.names = FALSE)
#
# occs <- fuzzySim::cleanCoords(occs, coord.cols = c("lon", "lat"),
# uncert.col = "coordinateUncertaintyInMeters", uncert.limit = 30000,
# abs.col = "occurrenceStatus", plot = TRUE)
#
# write.csv(occs, paste0("outputs/occs_clean.csv"), row.names = FALSE)
```

After running the above commands once, we can just import the data that were saved to disk, and convert them to a vector map of presence points:

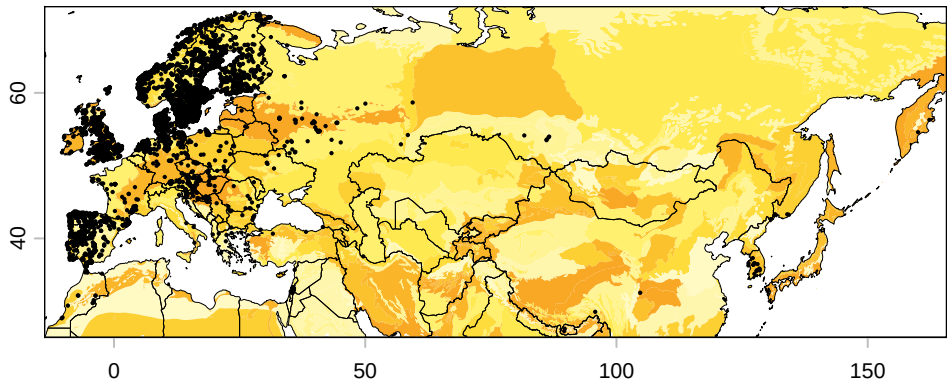
```
occs <- read.csv("outputs/occs_clean.csv")

occs <- terra::vect(occs, crs = "EPSG:4326")
```

The next commands import and plot some additional maps that we'll need:

```
## COUNTRIES (for geographical context) ----

cntry <- geodata::world(path = "outputs")


plot(occs, cex = 0.3)

lines(cntry, lwd = 0.2)
```

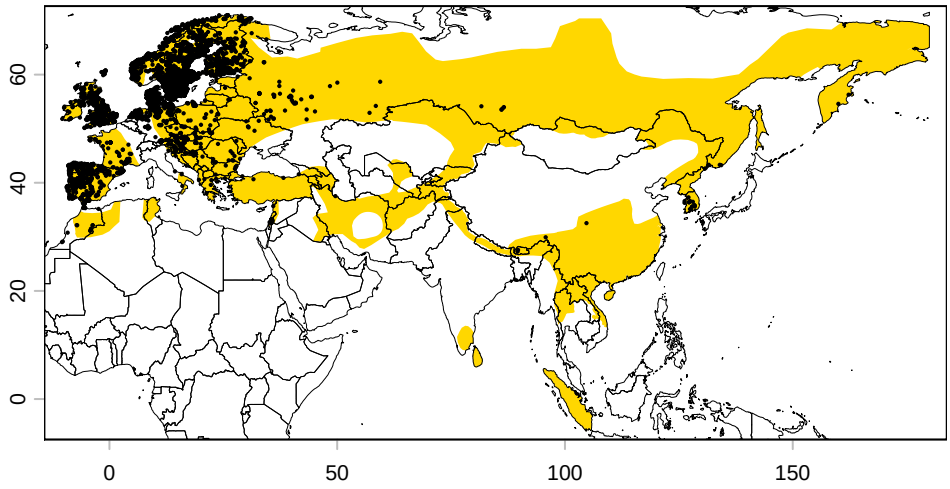



```
## ECOREGIONS ----

ecoreg <- terra::vect("/vsizip/inputs/ecoregions.zip")
# from https://doi.org/10.6084/m9.figshare.20972071
# which in turn came from
# https://doi.org/10.1641/0006-3568(2001)051[0933:TEOTWA]2.0.CO;2

plot(ecoreg, "ECO_NAME", col = yellows(10), ext = occs, border = NA,
     legend = FALSE)

points(occs, cex = 0.3)

lines(cntry, lwd = 0.2)
```




```
## RANGE MAP ----

rangemap <- terra::vect("/vsizip/inputs/Lutra_lutra.zip")
# from IUCN version 2025-2, available upon login at
# https://www.iucnredlist.org/resources/spatial-data-download
```

```r
plot(rangemap, col = "gold", border = NA)

points(occs, cex = 0.3)

lines(cntry, lwd = 0.2)
```

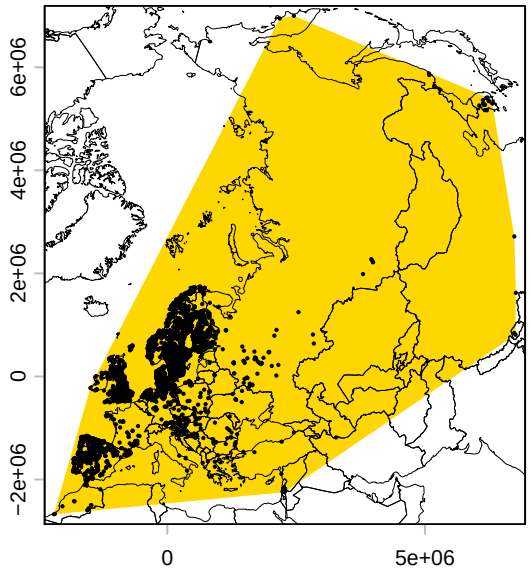


Next, we will project the maps to an equidistant CRS centered at the species occurrence region. This will allow a more accurate visualization of the distances within the maps:

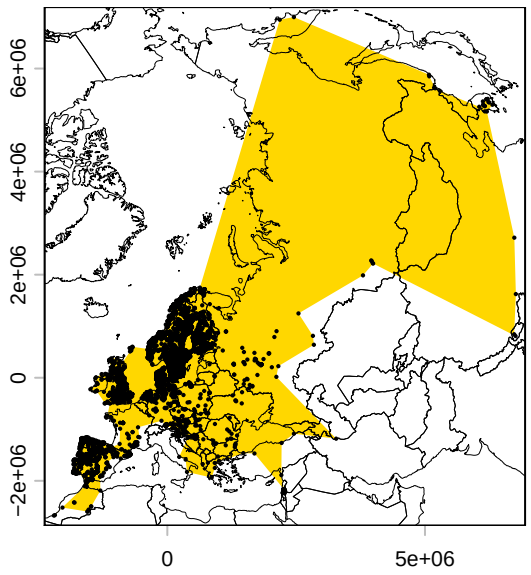
```r
centr <- terra::crds(terra::centroids(terra::aggregate(occs)))

prj <- paste("+proj=aeqd",
             paste0("+lat_0=", centr[2]),
             paste0("+lon_0=", centr[1]))

cntry_prj <- terra::project(cntry, prj)

occs_prj <- terra::project(occs, prj)

ecoreg_prj <- terra::project(ecoreg, prj)

rangemap_prj <- terra::project(rangemap, prj)
```

Now we will compute and plot some traditional background maps, using previously available techniques:

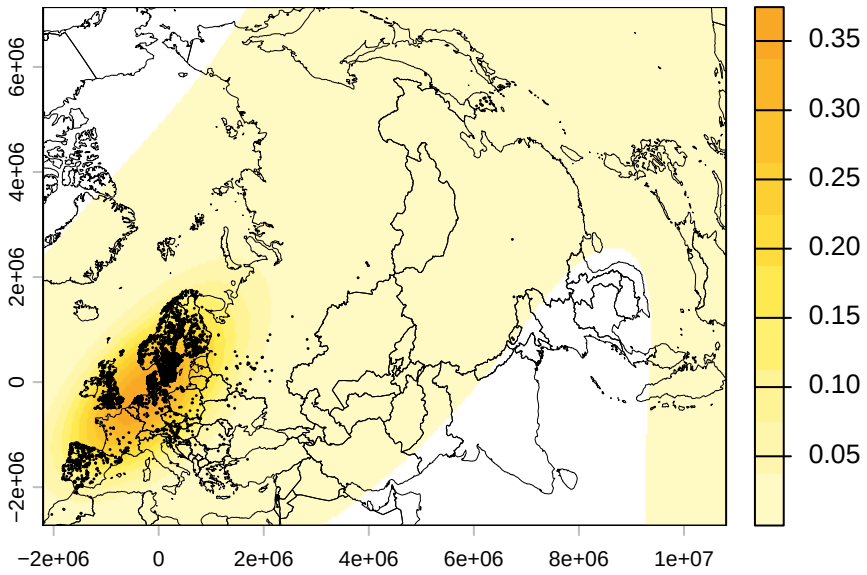
```
## MINIMUM CONVEX POLYGON (CONVEX HULL) ----

mcp <- terra::hull(occs_prj)

plot(mcp, col = "gold", border = NA)

points(occs_prj, cex = 0.3)

lines(cntry_prj, lwd = 0.2)
```



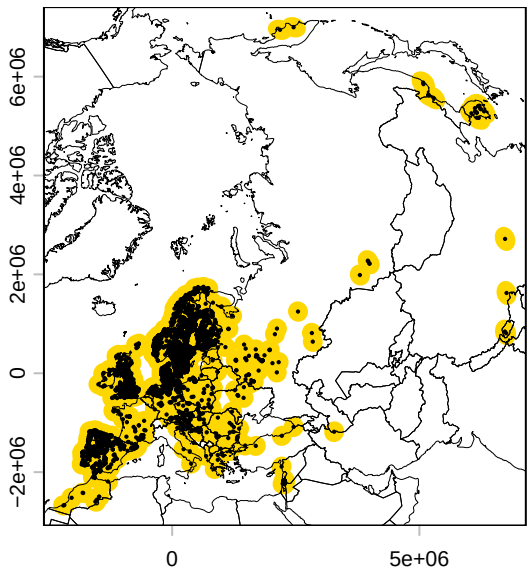

```
## CONCAVE HULL ----

conc <- terra::vect(concaveman::concaveman(terra::crds(occs_prj)),
                    type = "polygons", crs = prj)

plot(conc, col = "gold", border = NA)

points(occs_prj, cex = 0.3)

lines(cntry_prj, lwd = 0.2)
```




```
## TREND SURFACE ----

r <- terra::rast(extent = terra::ext(mcp + rangemap_prj) + 5)

terra::values(r) <- 1:ncell(r)

tsa <- fuzzySim::multTSA(r, sp.cols = terra::crds(occs_prj),
                         degree = 3, step = FALSE)

tsa_pres <- terra::extract(tsa, occs_prj, ID = FALSE)

# qt <- quantile(tsa_pres[,1], prob = 0.05)

tsa <- terra::ifel(tsa >= min(tsa_pres), tsa, NA)

# tsa <- terra::ifel(tsa >= qt, tsa, NA)

plot(tsa, col = yellows(10))

points(occs_prj, cex = 0.2)

lines(cntry_prj, lwd = 0.2)
```



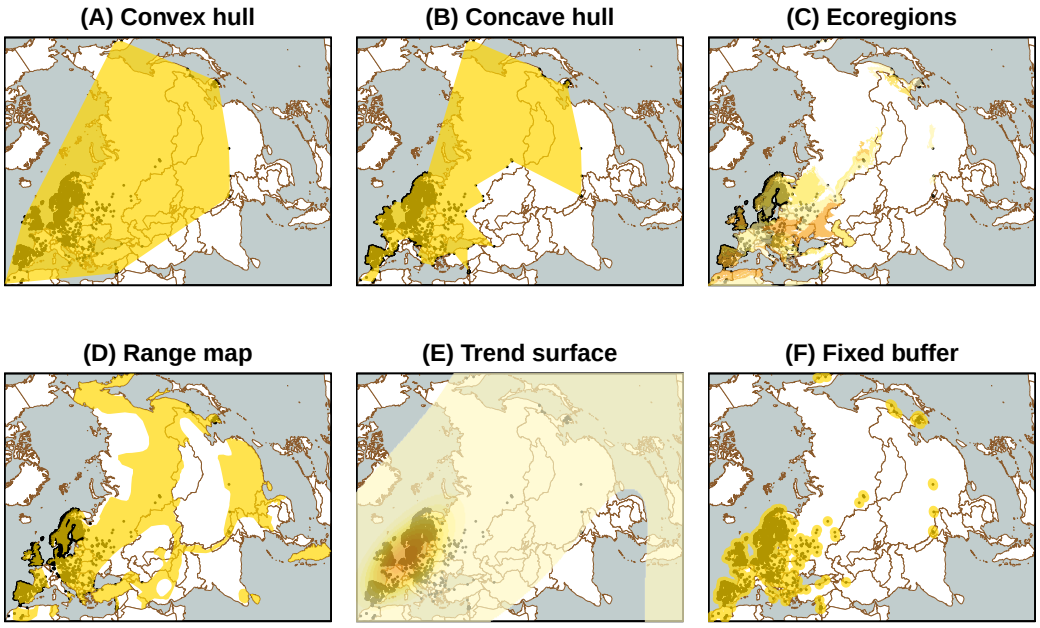

```
## FIXED-RADIUS BUFFER ----

buff <- terra::aggregate(terra::buffer(occs, width = 200 * 1000))  # 200 km
# note the buffer is more accurate when computed on unprojected coordinates
# to account for the Earth's curvature (see ?terra::buffer)
# we project it after computation, for a better visual depiction of distances

buff_prj <- terra::project(buff, prj)

plot(buff_prj, col = "gold", border = NA)

points(occs_prj, cex = 0.3)

lines(cntry_prj, lwd = 0.2)
```



Next, we will reproduce the first figure in the article:

```
base_maps <- function(main = "", cex.main = 1.1) {

  plot(cntry_prj, ext = mcp + rangemap_prj, col = "white", border = "tan4",
       background = "azure3", lwd = 0.3, main = main, cex.main = cex.main,
       buffer = FALSE, axes = FALSE, box = TRUE, mar = c(0.5, 0.5, 1.5, 0.5))

  points(occs_prj, cex = 0.3)
}

# pdf("outputs/figures/Fig1.pdf", width = 6, height = 3.5)
# png("outputs/figures/Fig1.png", width = 900, height = 600)

par(mfrow = c(2, 3))

base_maps(main = "(A) Convex hull")

# plot(mcp_buff, border = NA, col = "gold", alpha = 0.4, add = TRUE)
plot(mcp, border = NA, col = "gold", alpha = 0.7, add = TRUE)

base_maps(main = "(B) Concave hull")

plot(conc, border = NA, col = "gold", alpha = 0.7, add = TRUE)

base_maps(main = "(C) Ecoregions")
```

```
plot(subset(ecoreg_prj, occs_prj), "ECO_NAME", border = NA, col = yellows(10),
     legend = FALSE, alpha = 0.7, add = TRUE)

base_maps(main = "(D) Range map")

plot(rangemap_prj, border = NA, col = "gold", alpha = 0.7, add = TRUE)

base_maps(main = "(E) Trend surface")

plot(tsa, border = NA, col = yellows(10), alpha = 0.7, add = TRUE,
     legend = FALSE)

base_maps(main = "(F) Fixed buffer")

plot(buff_prj, border = NA, col = "gold", alpha = 0.7, add = TRUE)

# dev.off()
```

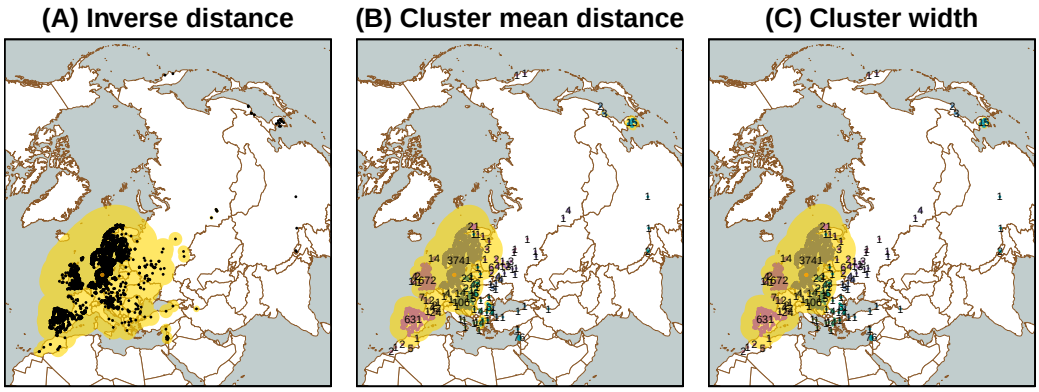


Now we compute and plot the adaptive background regions proposed in the article, reproducing the second figure:

```
e <- terra::ext(occs_prj) + 1200 * 1000

cexmain <- 1.2  # pdf output
```

```r
# cexmain <- 1.4  # png output

# pdf("outputs/figures/Fig2.pdf", width = 6, height = 2)
# png("outputs/figures/Fig2.png", width = 900, height = 400, pointsize = 24)

par(mfrow = c(1, 3))


## INVERSE DISTANCE ----

plot(cntry_prj, col = "white", border = "tan4", background = "azure3",
     lwd = 0.3, axes = FALSE, buffer = FALSE, mar = c(0.5, 0.5, 1.5, 0.5),
     ext = e, box = TRUE, main = "(A) Inverse distance", cex.main = cexmain)

reg_invdist <- fuzzySim::getRegion(occs, type = "inv_dist", alpha = 0.6,
                                   add = TRUE, prj = prj)

lines(reg_invdist, col = "orange", lwd = 2, alpha = 0.5)


## CLUSTER MEAN DISTANCE ----

plot(cntry_prj, col = "white", border = "tan4", background = "azure3",
     lwd = 0.3, axes = FALSE, buffer = FALSE, mar = c(0.5, 0.5, 1.5, 0.5),
     ext = e, box = TRUE, main = "(B) Cluster mean distance", cex.main = cexmain)

reg_clustmeandist <- fuzzySim::getRegion(occs, type = "clust_mean_dist",
                                         alpha = 0.6, add = TRUE, prj = prj)

lines(reg_clustmeandist, col = "orange", lwd = 2, alpha = 0.5)


## CLUSTER WIDTH ----

plot(cntry_prj, col = "white", border = "tan4", background = "azure3",
     lwd = 0.3, axes = FALSE, buffer = FALSE, mar = c(0.5, 0.5, 1.5, 0.5),
     ext = e, box = TRUE, main = "(C) Cluster width", cex.main = cexmain)

reg_clustwidth <- fuzzySim::getRegion(occs, type = "clust_width",
                                      alpha = 0.6, add = TRUE, prj = prj)

lines(reg_clustwidth, col = "orange", lwd = 2, alpha = 0.5)
```

```
# dev.off()
```

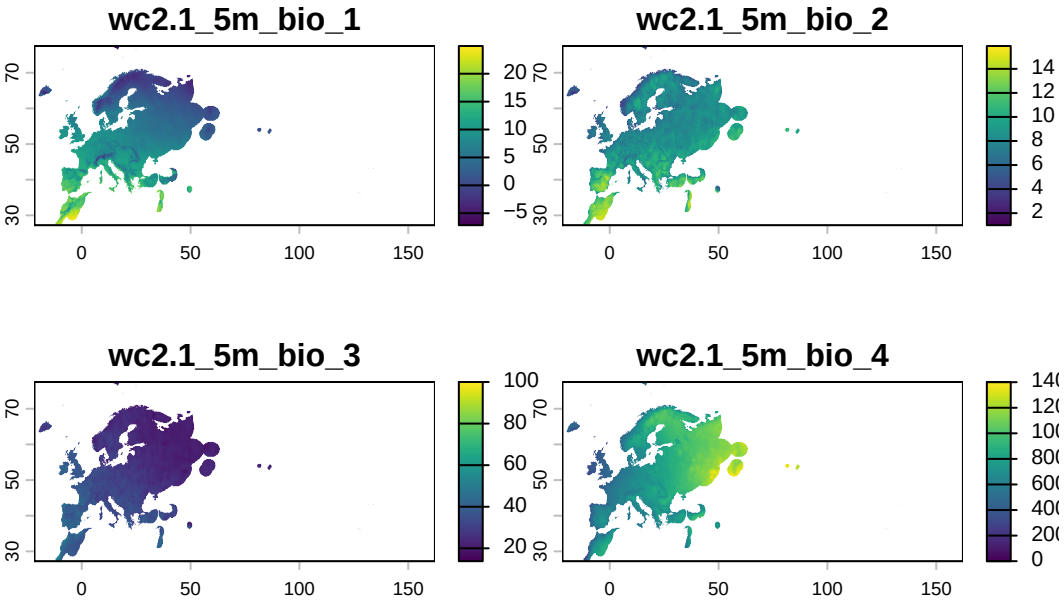


We can save the adaptive background polygons e.g. to GeoPackage files (which can be read with any modern GIS software), in case we want to re-use them later:

```
terra::writeVector(reg_invdist, "outputs/reg_invdist.gpkg",
                   overwrite = TRUE)

terra::writeVector(reg_clustmeandist, "outputs/reg_clustmeandist.gpkg",
                   overwrite = TRUE)

terra::writeVector(reg_clustwidth, "outputs/reg_clustwidth.gpkg",
                   overwrite = TRUE)
```

Now that we have our polygons, we can use them to crop and mask the environmental layers on which we want to do our variable selection and modelling. For example, if we want to use WorldClim 5-arc-min resolution bioclimatic layers, and the polygons generated by the `inv_dist` method:

```
layers <- geodata::worldclim_global(var = "bio", res = 5,
                                    path = "outputs/variables")

layers_buff <- terra::crop(layers, reg_invdist, mask = TRUE)

plot(layers_buff[[1:4]])  # show the first 4 cropped layers
```



Note that the maps in the two figures before this one were in a different cartographic projection for correctly visualizing distances, while here we use the original coordinate reference system of the layers, to avoid any distortion caused by raster projection.

The next command performs a thinning of the occurrence records to the spatial resolution of the environmental layers (keeping only one presence per pixel), and it produces equally thinned pseudoabsences (or unoccupied background points) outside the presence pixels but within the polygons defined above. It outputs the resulting presence and pseudo-absence data in a table ready for modeling. We could additionally use `fuzzySim::selectAbsences()` to select a subsample of the pseudoabsences, optionally with a bias layer (see examples in `?selectAbsences`) to further minimize survey bias within the modelling region.

```
occs_gridded <- fuzzySim::gridRecords(layers_buff, occs)

head(occs_gridded)
```

```
  presence        x        y cells wc2.1_5m_bio_1 wc2.1_5m_bio_2 wc2.1_5m_bio_3
1        0 22.62500 77.45833   533      -5.275000       3.350000       16.97635
2        0 22.70833 77.45833   534      -5.348026       3.336404       16.63823
3        0 22.79167 77.45833   535      -5.707028       3.223293       16.57682
4        0 22.87500 77.45833   536      -6.129083       3.134167       16.35104
5        0 22.95833 77.45833   537      -6.640208       2.980917       16.04282
6        0 23.04167 77.45833   538      -6.525792       2.970417       16.04763
  wc2.1_5m_bio_4 wc2.1_5m_bio_5 wc2.1_5m_bio_6 wc2.1_5m_bio_7 wc2.1_5m_bio_8
1       628.3532       4.483334      -15.25000       19.73333      -7.158333
```

```
2       638.7188       4.678947      -15.37368       20.05263      -7.348246
3       625.9432       4.050602      -15.39398       19.44458      -7.687149
4       620.6061       3.516000      -15.65200       19.16800      -8.112666
5       609.5587       2.700000      -15.88100       18.58100      -8.581000
6       604.5749       2.840000      -15.67000       18.51000      -8.431334
  wc2.1_5m_bio_9 wc2.1_5m_bio_10 wc2.1_5m_bio_11 wc2.1_5m_bio_12
1    -0.02499998        2.730556       -12.25000             480
2     0.07280701        2.775439       -12.40702             484
3    -0.42550200        2.282129       -12.55562             503
4    -0.92300004        1.803500       -12.89083             521
5    -1.53049994        1.165333       -13.24333             545
6    -1.47983336        1.236333       -13.05650             550
  wc2.1_5m_bio_13 wc2.1_5m_bio_14 wc2.1_5m_bio_15 wc2.1_5m_bio_16
1              58              24        28.51962             156
2              59              25        28.37101             158
3              61              26        28.19773             164
4              63              27        27.73161             169
5              65              29        27.27380             176
6              67              29        27.71816             179
  wc2.1_5m_bio_17 wc2.1_5m_bio_18 wc2.1_5m_bio_19
1              78             110             133
2              80             110             133
3              84             115             138
4              87             119             143
5              92             125             150
6              92             125             151
```

We will now exclude redundant (highly correlated) variables within this dataset, and compute a generalized linear model on the resulting data (though any other modelling method can be used, if additional required packages are installed):

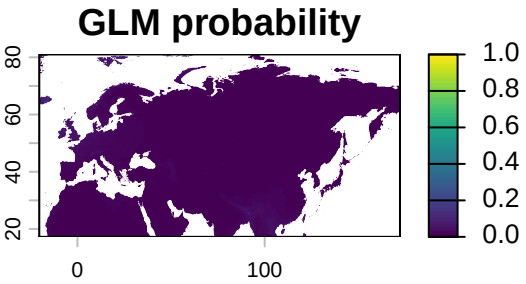
```
vars_sel <- fuzzySim::corSelect(occs_gridded, sp.cols = "presence",
                                var.cols = names(layers_buff), simplif = TRUE)

glm_form <- reformulate(termlabels = vars_sel, response = "presence")

glm_mod <- glm(formula = glm_form, family = binomial, data = occs_gridded)

summary(glm_mod)
```

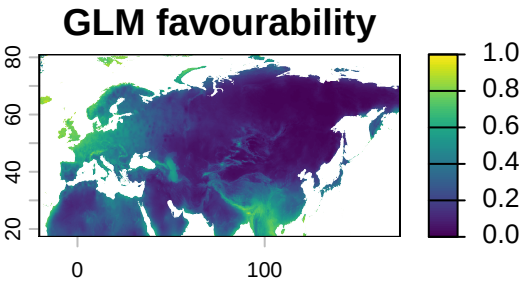
```
Call:
```

```
glm(formula = glm_form, family = binomial, data = occs_gridded)

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)      0.4238640  0.1425338   2.974 0.002942 **
wc2.1_5m_bio_2   0.0035765  0.0193987   0.184 0.853725
wc2.1_5m_bio_6  -0.0293858  0.0084448  -3.480 0.000502 ***
wc2.1_5m_bio_7  -0.1498816  0.0097956 -15.301  < 2e-16 ***
wc2.1_5m_bio_8  -0.0018477  0.0057566  -0.321 0.748236
wc2.1_5m_bio_15 -0.0057280  0.0015939  -3.594 0.000326 ***
wc2.1_5m_bio_18  0.0002809  0.0003685   0.762 0.445891
wc2.1_5m_bio_19 -0.0013447  0.0002656  -5.063 4.12e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 35715  on 205704  degrees of freedom
Residual deviance: 33260  on 205697  degrees of freedom
AIC: 33276

Number of Fisher Scoring iterations: 7
```

We can now compute the predictions of this model, and also remove the effect of modelled prevalence (common to all presence-absence modelling methods) by using the favourability function. Note that prediction need not be limited to the fragmented buffers on which the model was computed – we can predict on any region of interest:

```r
layers_cut <- terra::crop(layers, terra::ext(occs) + 10)

glm_pred <- predict(layers_cut, glm_mod, type = "response")

glm_fav <- fuzzySim::Fav(pred = glm_pred,
                         sample.preval = prevalence(model = glm_mod))


par(mfrow = c(2, 1))

plot(glm_pred, range = c(0, 1), main = "GLM probability")

plot(glm_fav, range = c(0, 1), main = "GLM favourability")
```





Note that predictions outside the modelled areas may be outside the range of environmental values analysed by the models, so they need to be taken with extra caution. Functions like `mess` in the `predicts` package, or `mop` in the `mop` package, can help assess where and how much extrapolation is taking place.