Using the sampbias R package

Alexander Zizka, Alexandre Antonelli, Daniele Silvestro

Installing sampbias

To install *sampbias*, you can use the `devtools` package:

```
require(devtools)
install_github(repo = "azizka/sampbias")
library(sampbias)
library(maptools)
library(rgdal)
library(raster)
```

Input data

Sampbias calculates spatial bias in a species occurrence data set based on two input files:

1. A table of species occurrence records
2. A set of geographical gazetteers

The example files for this tutorial are provided with the package, descriptions and help for all functions are accesible via `?Functionname`, for instance `?calculate_bias`.

1. Species occurrence records

Species occurrences can be provided as an `data.frame` including a minimum of three columns named "species", "decimalLongitude", and "decimalLatitude". *Sampbias* can also directly work with data downloaded from the Global Biodiversity Information Facility data portal (www.gbif.org). We will use the records of mammals from Borneo provided with the package (GBIF, 2016, doi.org/10.15468/dl.7fg4zx.) as example in this tutorial.

The input data can be easily loaded into R from a .txt. or .csv file, for instance using the `read.csv()` function.

```
#loading a text file to R
occ <-read.csv(system.file("extdata",
                           "mammals_borneo.csv",
                           package="sampbias"),
               sep = "\t")
```

Note: *Sampbias* evaluates the bias by comparing the observed records to a random sampling. This meaning that the tool is designed for multi-species data sets and **not** single species data sets where the distribution of records might reflect ecological preferences. In general, the more species and the more records, the more reliable the results will be.

2. Geographic gazetteers

Sampbias evaluates the distribution of the sampled species occurrences in relation to geographic features that might bias sampling effort. These are usually related to accessibility or means of travel. *Sampbias* includes a

set of default gazetteers for cities, airports, roads and rivers (from https://www.naturalearthdata.com/), which can give an estimate of bias for large and medium scale analyses. For higher resolutions custom gazetteers will yield better results, and can be provided via the `gaz` option of `calculate_bias` function as a list of `SpatialPointsDataFrame` and `SpatialLinesDataFrame` objects. These can easily be loaded into R from standard shape files using the `rgdal` package:

```
cit <- readOGR(dsn = system.file("extdata", package="sampbias"),
               layer = "Borneo_major_cities", verbose = FALSE)
roa <- readOGR(dsn = system.file("extdata", package="sampbias"),
               layer = "Borneo_major_roads", verbose = FALSE)

gazetteers <- list(cities = cit,
                   roads = roa)
```

See https://cran.r-project.org/web/packages/sp/vignettes/intro_sp.pdf and `?SpatialPointsDataFrame` or `?SpatialLinesDataFrame` on how to create a these object from a table of coordinates.

Running an analysis

A sampbias analyses is run in one line of code via the `calculate_bias` function:

```
bias.out <- calculate_bias(x = occ, gaz = gazetteers)
```

In addition to the input from above, `calculate_bias` offers options to customize analyses, of which the most important ones are shown in Table 1. See `?calculate_bias` for a detailed description of all options, and the section "Changing default settings" of this tutorial for a description and examples for the "restricted sample" and "inp_raster" options. **It is of special importance to adapt the raster resoltuion to the extent of the study area via the `res` or `inp_raster` options.**

Option	Description
res	the raster resolution for the distance calculation to the geographic features and the data visualization in decimal degrees. The default is to one degree, but higher resolution will be desirable for most analyses. `res` together with the extent of the study area determine computation time and memory requirements.
restrict_sample	a `SpatialPolygons` or `SpatialPolygonsDataFrame` object. If provided the area for the bias test will be restricted to raster cells within these polygons (and the extent of the sampled points in x). Make sure to use adequate values for `res`. Default = NULL.
terrestrial	logical. If TRUE, the empirical distribution (and the output maps) are restricted to terrestrial areas. Default = TRUE.
inp_raster	a raster object. This option enables to directly provide the grid for the bias calculation, instead of specifying a resolution. This can be used to work with equal area grids or other coordinate references systems.

The *sampbias* model uses a Markov chain Monte Carlo approach to estimate the model parameters. The default MCMC runs for 100,000 generations with a 20% burn-in, which has proven sufficient for most analyses.

We suggest to verify that the effective sample size of the posterior estimates is large enough (e.g. > 200) using the `ESS` function of the `LaplacesDemon` package (`LaplacesDemon::ESS(bias.out$bias_estimate)`).

Output

The output of `calculate_bias` is a list of different R objects.

Object	Description
summa	A list of summary statistics, including the total number of occurrence points in x, the total number of species in x, the extent of the output rasters as well as the settings for res, binsize, and restrict_area used in the analyses.
occurrences	a raster indicating occurrence records per grid cell, with resolution `res`.
bias_estimate	A `data.frame` with the posterior estimates of the model parameters, including bias weights
distance_raster	A raster or RasterStack, with the calculated distances for all bias factors (e.g. cities, roads, etc.) as used in the model

The package includes summary and plot methods for an easy exploration of the results. The plot method generates a boxplot of the posterior estimates of the weights for each biasing factor (Fig. 1).

```
summary(bias.out)
```

```
## Number of occurences:  6262
## Raster resolution:  1
## Convexhull:
## Geographic extent:
## class      : Extent
## xmin       : 108
## xmax       : 120
## ymin       : -5
## ymax       : 7
## Bias weights:
##           bias_weight      std_dev
## w_cities 6.827455e-03 1.719944e-04
## w_roads  1.109673e-05 9.482415e-06
```

```
plot(bias.out)
```

As the last step, it is possible to project the bias effects into space and map them to identify areas with particular high bias, for instance, to design future field campaigns (Fig. 2).

```
proj <- project_bias(bias.out)
map_bias(proj)
```

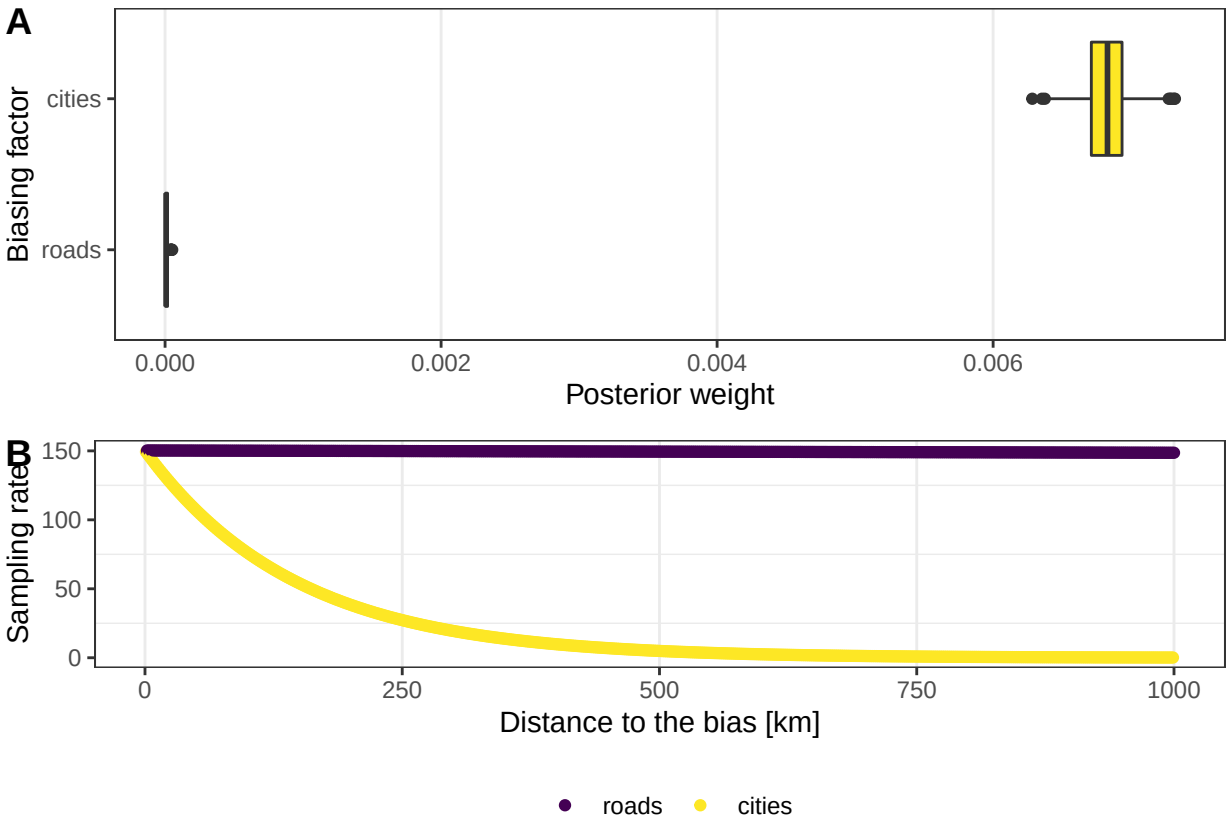

Figure 1: Visulaization of the output of a sampbias analysis. A) Posterior estimates of the bias weights for cities and roads, B) the decay of the sampling rate with increasing distaince from cities and roads.

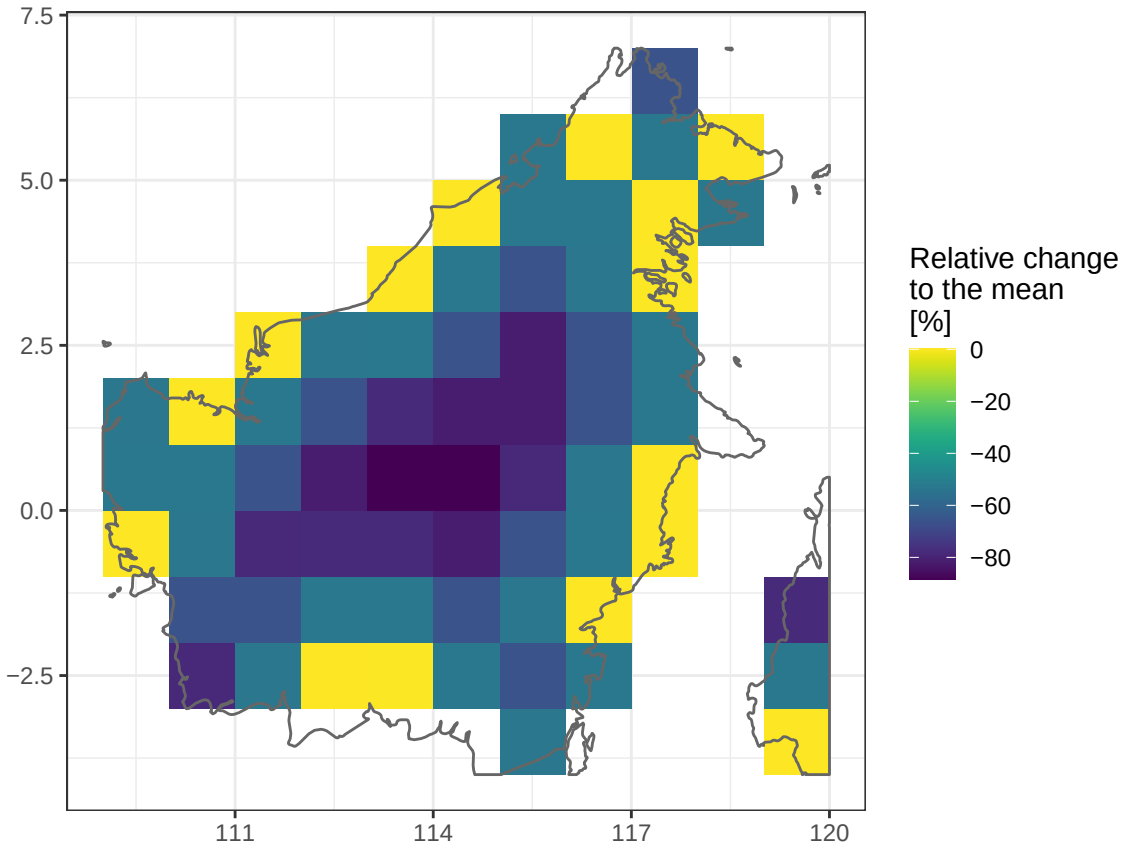

Figure 2: Sampling bias effect projected in space. Note the coarse grid resolution which should be increased.

Changing default settings

The `calculate_bias` function provides various options to customize analyses (see above). For two important options—providing a custom study area and using a equal area grid—we provide examples below.

Custom study area

By default *sampbias* uses a rectangle defined by the minimum and maximum longitude and latitude values in the input point records to define the study area. It may be desirable to change this, for instance if this rectangle comprises a set of different habitats such as rainforest and desert, which may differ in species richness and therefore the number of records expected. *Sampbias* enables user-defined study areas via the `restrict_sample` option of the `calculate_bias` function, which uses objects of the class `SpatialPolygonsDataFrame` or `SpatialPolygons`. This could be simple custom polygons as in the `area_example` dataset provided with *sampbias*:

```
data(area_example)
borneo <- crop(sampbias::landmass, extent(108, 120, -5, 7))

plot(borneo)
plot(area_example, col = "red", add = TRUE)
```

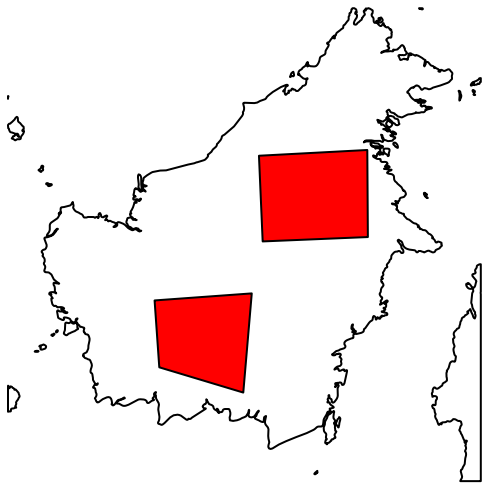

Figure 3: Example of a customized study area which can be provided to the claculate_bias function via the restrict_sample option. Only grid cells within the red area are included in the sampling bias calculation.

```
bias.out <- calculate_bias(x = occ,
                           gaz = gazetteers,
                           restrict_sample = area_example)
```

More complex restrictions are possible, for instance limiting the study area to the ecoregion "Borneo montane rain forests" (Olson, et al. 2001). For diagnostics the sample raster can be plotted using the `plot_raster` argument.

```
data(ecoregion_example)

plot(borneo)
plot(ecoregion_example, col = "red", add = TRUE)
```

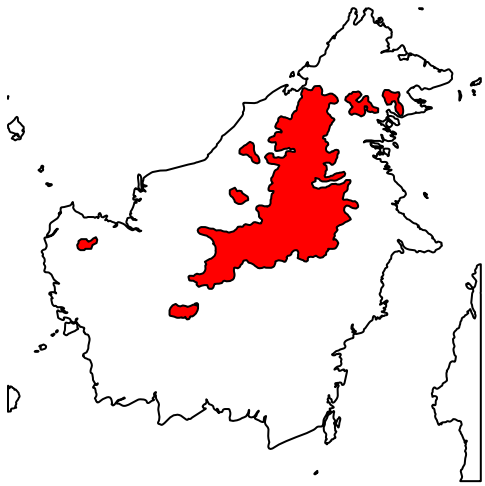

Figure 4: Example of a more complex customized study area representing the Borneo montane rain forests ecoregion from Olson et al. (2001). Only grid cells within red area are included in the sampling bias calculation.

```
bias.out <- calculate_bias(x = occ,
                           gaz = gazetteers,
                           restrict_sample = ecoregion_example)
```

Equal area grid

By default *sampbias* uses a lat/lon grid. This is a reasonable approximation for local and regional scales. However, since the area of the cells in such a grid differs with latitude, an equal area grid—for instance using a Behrmann projection—is a better choice for large-scale studies. When using an equal area grid, it is necessary for the input occurrence records as well as the gazetteer files to share the same projection and coordinate reference system.

```r
#projection
wgs84 <- CRS("+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs")

# an example for an equal area raster
data(ea_raster)

# reproject the occurrence coordinates
## select coordinates from the occ data.frame and create spatial object
ea_occ <- SpatialPoints(occ[,c(3,2)],
                        proj4string = wgs84)
## transform to the same CRS as the equal area grid
ea_occ <- spTransform(ea_occ, CRSobj = proj4string(ea_raster))
## retransform into a data.frame
ea_occ <- data.frame(species = occ[,1], coordinates(ea_occ))

# reproject gazetteers
## set the CRS in case it is not defined. Make sure to know the correct CRS.
proj4string(gazetteers[[1]]) <-
  proj4string(gazetteers[[2]]) <-
  wgs84

#transform to the new CRS
ea_gaz <- lapply(gazetteers, "spTransform", CRSobj = proj4string(ea_raster))

# run sampbias
ea_bias <- calculate_bias(x = ea_occ,
                          gaz = ea_gaz,
                          inp_raster = ea_raster)
```