Empirical habitat association graphs and mapping

P. Solymos

2015-07-13

Contents

Goal

We use a cutoff based on number of detections per species to decide if we fit parametric/semiparametric models to their data. For the rest of the species (uncommon ones), we want to show basic information about their spatial distribution (detection maps) and habitat associations.

Function for calculating relative suitability

I called this function `wrsi` to refe to `w`eighted `r`elative `s`uitability `i`ndex:

```
wrsi <-
function(Y, X)
{
    Y <- as.integer(ifelse(Y > 0, 1L, 0L))
    X <- data.matrix(X)
    n <- length(Y)
    if (nrow(X) != n)
        stop("dimension mismatch: X and Y")
    K <- ncol(X)
    if (is.null(colnames(X)))
        colnames(X) <- paste0("V", seq_len(K))
    ## relative suitability index
    ## # of available units of type k / total # of available units (any type)
    Pavail <- colSums(X) / sum(X)
    ## # of used units of type k / total # of used units (any type)
    Xu <- X * Y
    ## sum(Xu) = sum(Y) except when rowsum != 1
    Pused <- colSums(Xu) / sum(Xu)
    ## crude weighted p-occ
    Pw <- colSums(Xu) / colSums(X)
    ## Weighted Relative Suitability Index
    WRSI <- Pused / Pavail
    #Var <- (1/colSums(Xu)) - (1/sum(Xu)) + (1/colSums(X)) - (1/sum(X))
    res <- data.frame(
        WRSI=WRSI,
        zWRSI=log(WRSI),
        rWRSI=(exp(2 * log(WRSI)) - 1)/(1 + exp(2 * log(WRSI))),
        Pused=Pused,
        Pavail=Pavail,
        Pw=Pw,
        u=colSums(Xu),
        a=colSums(X))
    rownames(res) <- colnames(X)
    class(res) <- c("wrsi", "data.frame")
    res
}
```

Example for mite species

Processing species and vegetation data

Load some R packages: mera4 for data processing, RColorBrewer for color palettes:

```
library(mefa4)
```

```
## Loading required package: Matrix
## mefa4 0.3-2   2015-05-13
```

```
library(RColorBrewer)
```

Load necessary data sets, you have to substitute an appropriate working directory. I set up my system so that home ~ points to `c:/Users/Peter/` on Windows (and not to the default `c:/Users/Peter/Documents`) and Mac iOS, and it points to `/home/peter/` on Linux by default. You can use the link to the lookup table (`tveg`) to download the file for offline work.

```
setwd("~/Dropbox/Public/")
load("OUT_mites_2015-05-22.Rdata")
load("veg-hf-clim-reg_abmi-onoff_fix-fire_fix-age0.Rdata")
tveg <- read.csv(paste0("https://raw.githubusercontent.com/psolymos/",
    "abmianalytics/master/lookup/lookup-veg-hf-age.csv"))
```

The site level detection/non-detection table for mites is called `m2`, the corresponding current vegetation table is `dd1ha$veg_current`. After extracting those, I am using a modified version of the `Type` column in the table `tveg` to reclass habitats:

```
yyy <- as.matrix(xtab(m2))
hhh <- as.matrix(dd1ha$veg_current)
iii <- intersect(rownames(yyy), rownames(hhh))
yyy <- yyy[iii,]
hhh <- hhh[iii,]
vvv <- as.character(tveg$Type)
vvv[is.na(vvv)] <- as.character(tveg$UseInAnalysis[is.na(vvv)])

vvv[vvv %in% c("WetGrassHerb","WetShrub")] <- "Wet"
vvv[vvv %in% c("GrassHerb","Shrub")] <- "Open"
vvv[vvv %in% c("Water","HWater","NonVeg","WetBare")] <- "EXCLUDE"
```

Now I use the vector `vvv` that matches the columns in `dd1ha$veg_current` to add up the cells in each row. Then I exclude the column `EXCLUDE` and starndardize by row sums, finally reorder the columns in the way I want them to show up in the figures:

```
hhh <- groupSums(hhh, 2, vvv)
hhh <- hhh[,colnames(hhh) != "EXCLUDE"]
hhh <- hhh / ifelse(rowSums(hhh) == 0, 1, rowSums(hhh))

hhh <- hhh[, c("Decid", "Mixwood", "Conif", "Pine",
    "BSpr", "Larch",
    "Open", "Wet",
    "Cult", "UrbInd", "HardLin", "SoftLin")]
```

I have to exclude rows where only those classes showed up that I have removed (e.g. water, etc.), and also remove species that happened to found only at these places:

```
keep <- rowSums(hhh) > 0
yyy <- yyy[keep,]
hhh <- hhh[keep,]
yyy <- yyy[,colSums(yyy>0) > 1]
```

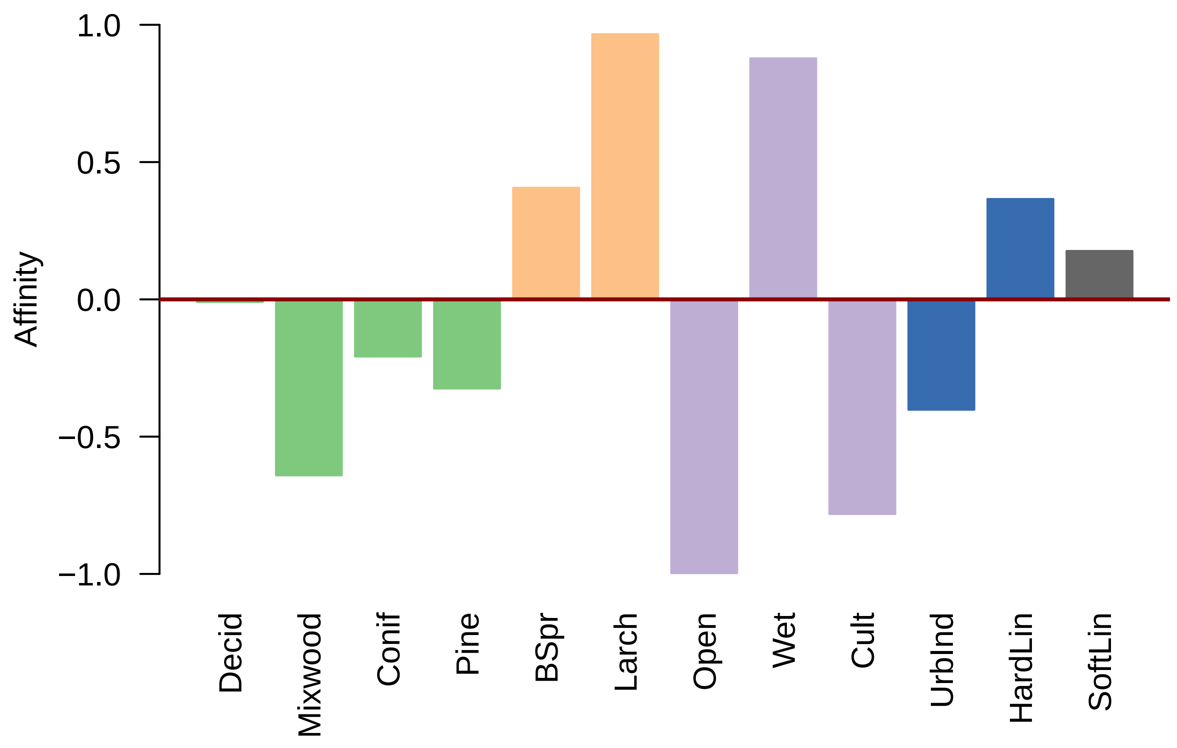
Plotting habitat associations

This plotting function takes two arguments: `Y` is a vector of detections/non-detections for a given species, mathching the rows in `X` that is the matrix with relative proportion of habitat classes (these are used as weights in the calculations):

```
plot_wrsi <-
function(Y, X, ...)
{
    ## calculate the index to plot
    w <- wrsi(Y, X)
    rw <- w$rWRSI
    names(rw) <- rownames(w)
    ## set up a color palette
    col <- brewer.pal(8, "Accent")[c(1,1,1,1, 3,3, 2,2,2, 5,5, 8,8)]
    ## adjust margins and text direction
    op <- par(mar=c(6,4,2,2)+0.1, las=2)
    ## produce a barplot without axes
    tmp <- barplot(rw, horiz=FALSE, ylab="Affinity",
        space=NULL, col=col, border=col, #width=sqrt(w$Pavail),
        ylim=c(-1,1), axes=FALSE, ...)
    ## add in the axis
    axis(2)
    ## a line
    abline(h=0, col="red4", lwd=2)
    ## reset par options
    par(op)
    ## return the index values invisibly
    invisible(w)
}
```

Here is how it looks line for a single species:

```
spp <- 1
plot_wrsi(yyy[,spp], hhh)
```



Here is the script to produce a pdf binder, or several png files. **Note**: naming does not follow the preferred naming convention (save as `SpeciesGenus.png` is preferred and the path would tell that it is a detection map).

```
output_type <- "pdf" # change this to png or pdf
if (output_type == "pdf") {
    pdf("mites-pw.pdf", onefile=TRUE)
    for (i in 1:ncol(yyy)) {
        cat(colnames(yyy)[i], "\n")
        flush.console()
        main <- paste0(as.character(taxa(m2)[colnames(yyy)[i], "SPECIES_OLD"]),
            " (", sum(yyy[,i] > 0), " detection", ifelse(sum(yyy[,i] > 0) > 2,
            "s)", ")"))
        plot_wrsi(yyy[,i], hhh, main=main)
    }
    dev.off()
}
if (output_type == "png") {
    for (i in 1:ncol(yyy)) {
        cat(colnames(yyy)[i], "\n")
        flush.console()
        png(paste0("mites-", colnames(yyy)[i], ".png"), width = 480, height = 480)
        main <- paste0(as.character(taxa(m2)[colnames(yyy)[i], "SPECIES_OLD"]),
            " (", sum(yyy[,i] > 0), " detection", ifelse(sum(yyy[,i] > 0) > 2,
            "s)", ")"))
        plot_wrsi(yyy[,i], hhh, main=main)
        dev.off()
    }
}
```

Mapping

Preparations (common objects for mapping)

Load some more packages:

```
library(raster)
```

```
## Loading required package: sp
```

```
library(sp)
library(rgdal)
```

```
## rgdal: version: 0.9-2, (SVN revision 526)
## Geospatial Data Abstraction Library extensions to R successfully loaded
## Loaded GDAL runtime: GDAL 1.11.2, released 2015/02/10
## Path to GDAL shared files: C:/R/R-3.2.0/library/rgdal/gdal
## GDAL does not use iconv for recoding strings.
## Loaded PROJ.4 runtime: Rel. 4.9.1, 04 March 2015, [PJ_VERSION: 491]
## Path to PROJ.4 shared files: C:/R/R-3.2.0/library/rgdal/proj
```

Define values for recovering path to current version of the data. If you copy the whole shebang from FTP as a single directory, all you need to do is to change `ROOT` because in that case everything should be mirrored:

```
ROOT <- "c:/p"
VER <- "AB_data_v2015"
```

Source some functions (you can use the link to download the file for offline work)

```r
source(paste0("https://raw.githubusercontent.com/psolymos/",
    "abmianalytics/master/R/maps_functions.R"))
```

Load the Rdata object that store the wall-to-wall vegetation/soil/HF information and the table with lat/long/climate etc:

```r
load(file.path(ROOT, VER, "out", "kgrid", "veg-hf_1kmgrid_fix-fire_fix-age0.Rdata"))
load(file.path(ROOT, VER, "out", "kgrid", "kgrid_table.Rdata"))
```

Lookup tables for vegetation and soil classes (combined with HF):

```r
tveg <- read.csv(paste0("https://raw.githubusercontent.com/psolymos/",
    "abmianalytics/master/lookup/lookup-veg-hf-age.csv"))
tsoil <- read.csv(paste0("https://raw.githubusercontent.com/psolymos/",
    "abmianalytics/master/lookup/lookup-soil-hf.csv"))
```

Creating raster templates (the file `AHM1k.asc` is also on FTP):

```r
## raster template to use
rt <- raster(file.path(ROOT, VER, "data", "kgrid", "AHM1k.asc"))
crs <- CRS(paste0("+proj=tmerc +lat_0=0 +lon_0=-115 +k=0.9992 +x_0=500000 ",
    "+y_0=0 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +units=m +no_defs"))
projection(rt) <- crs
mat0 <- as.matrix(rt)
```

Create raster layers that are going to be used in mapping water etc:

```r
r_water <- as_Raster(kgrid$Row, kgrid$Col, kgrid$pWater, rt)
r_water[r_water <= 0.99] <- NA
## Rockies to mask out
r_mask <- as_Raster(kgrid$Row, kgrid$Col,
    ifelse(kgrid$NRNAME == "Rocky Mountain" & kgrid$X < 800000, 1, 0), rt)
r_mask[r_mask < 1] <- NA
## combine the 2 (this is necessary due to a bug in raster plotting
## function: when >3 layers are shown there is a misterious mismatch...)
r_overlay <- r_water
r_overlay[!is.na(r_water)] <- 1
r_overlay[!is.na(r_mask)] <- 2
r_overlay <- as.factor(r_overlay)
rat <- levels(r_overlay)[[1]]
rat$levels <- c("Water", "Rockies")
rat$code <- 1:2
levels(r_overlay) <- rat
## natural regions
nr <- as.matrix(Xtab(as.integer(NRNAME) ~ Row + Col, kgrid))
nr[is.na(mat0)] <- NA
nr <- as.factor(raster(x=nr, template=rt))
rat <- levels(nr)[[1]]
rat$NaturalRegion <- levels(kgrid$NRNAME)
rat$code <- seq_len(nlevels(kgrid$NRNAME))
```

```r
levels(nr) <- rat
## city coordinates
city <-data.frame(x = -c(114,113,112,111,117,118)-c(5,30,49,23,8,48)/60,
    y = c(51,53,49,56,58,55)+c(3,33,42,44,31,10)/60)
rownames(city) <- c("Calgary","Edmonton","Lethbridge","Fort McMurray",
    "High Level","Grande Prairie")
coordinates(city) <- ~ x + y
proj4string(city) <- CRS(paste0("+proj=longlat +datum=WGS84 ",
    "+ellps=WGS84 +towgs84=0,0,0"))
city <- spTransform(city, CRS(paste0("+proj=tmerc +lat_0=0 +lon_0=-115 +k=0.9992 ",
    "+x_0=500000 +y_0=0 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +units=m +no_defs")))
```

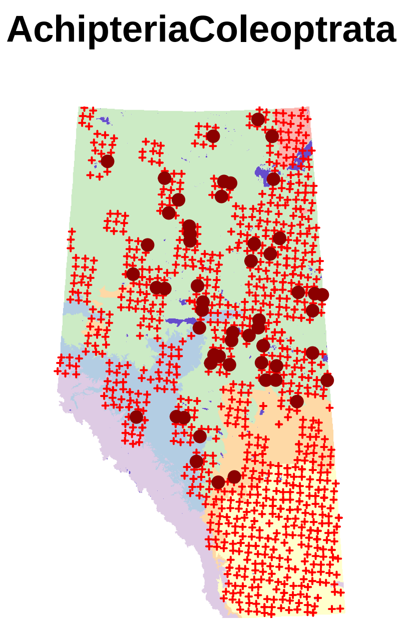
Mapping detections

Now we use the site level table (the ''Binomial" version but stored in the Rdata file), and select a species, plus the public lat/long columns:

```r
spp1 <- "AchipteriaColeoptrata" # the 1st species in the list
x <- res2
x <- x[!is.na(x$PUBLIC_X),]
y <- as.matrix(x[,which(colnames(x) == spp1):ncol(x)])
xy <- x[,c("PUBLIC_X","PUBLIC_Y")]
colnames(xy) <- c("POINT_X", "POINT_Y")
```

Here is how it looks like for the 1st species:

```r
xy_map(xy, y[,1], main=colnames(y)[1])
```



Saving the map as png. **Note**: naming does not follow the preferred naming convention (save as `SpeciesGenus.png` is preferred and the path would tell that it is a detection map). I played around with file resolution and `cex` values to get a reasonable looking map in the end:

```
for (i in 1:ncol(y)) {
    cat(i, "/", ncol(y), "\n");flush.console()
    #pdf(paste0("mites-pa-", spp1, ".pdf"), width=6, height=9)
    png(paste0("mites-pa-", spp1, ".png"), width=1200, height=1800)
    xy_map(xy, y[,i], main=colnames(y)[i], cex0=0.5*3, cex1=0.8*3)
    dev.off()
}
```

Plotting other stuff and predictive maps

There are different masks we want to use, e.g.

1. to mask the Rockies (we don't have enough data),
2. to mask places without soil info (for South soil based model predictions).

These options and the different color palettes are demonstrated here.

The function `map_fun` takes the following arguments:

- `x` is the vector (matching `kgrid`) to be plotted,

- `q` is the quantile that we want to use to censor the data (i.e. `x[x > quantile(x, q)] <- quantile(x, q))`,
- `main` is the title of the plot,
- `colScale` is the color scale with these pre-defined options: `"terrain"` is the default generic one, `"heat"`, `"topo"`, `"grey"` are also conventional ones, `"hf"` is for footprint, `"soil"` is for soil maps, `"abund"` is for relative abundance maps, `"diff"` is for the current-reference difference maps, it can also be a function returning a palette.
- `plotWater` whether to plot water or not,
- `maskRockies` whether to mask the Rockies or not,
- `plotCities` whether to plot cities or not,
- `legend` whether to produce a legend or not.

Here is the example of total HF: make sure that the vector matches up with the rows in the object `kgrid`:

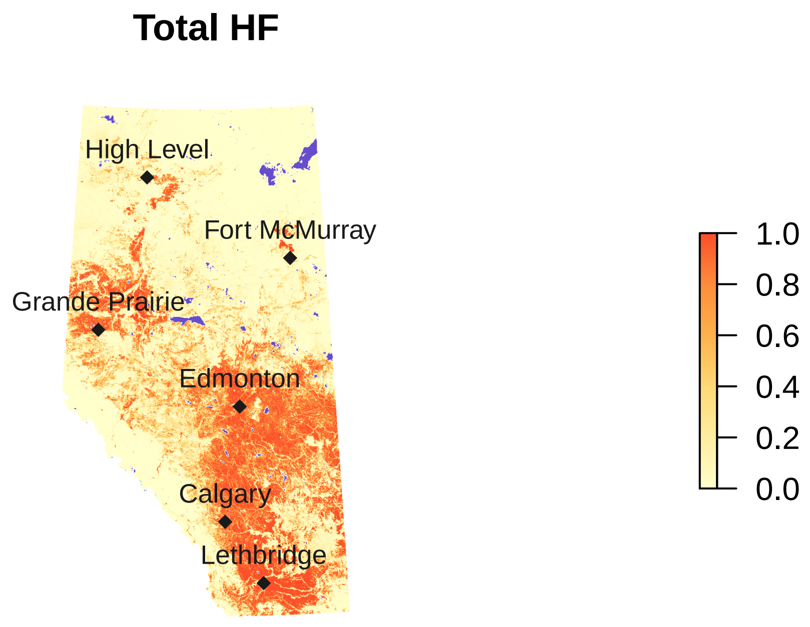
```
thf <- rowSums(dd1km_pred$veg_current[,!is.na(tveg$HF)]) /
    rowSums(dd1km_pred$veg_current)
length(thf)
```

```
## [1] 664767
```

```
dim(kgrid)
```

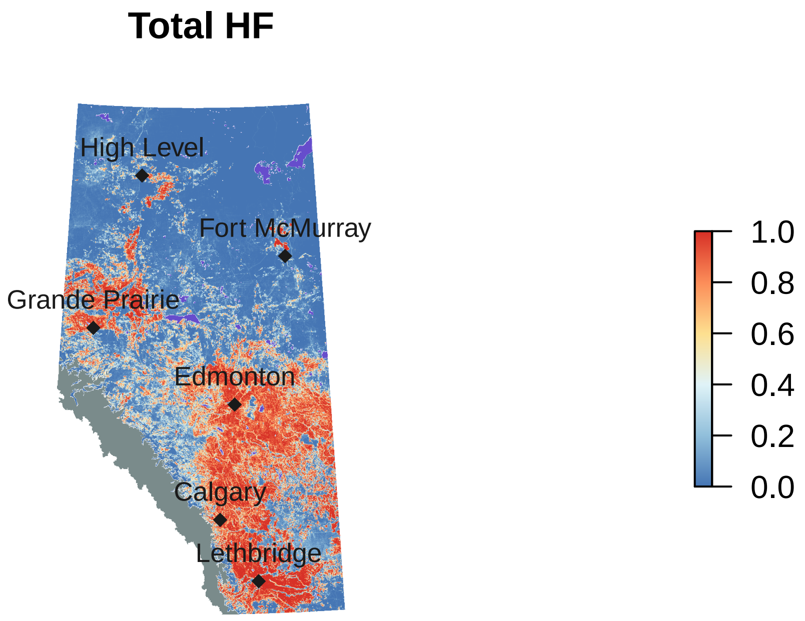
```
## [1] 664767     27
```

```
map_fun(thf, q=0.999, main="Total HF",
    colScale="hf", maskRockies=FALSE)
```



Now the same in ''abundance" style, masking the Rockies:

```r
map_fun(thf, q=0.999, main="Total HF",
    colScale="abund", maskRockies=TRUE)
```


Total HF
High Level
Fort McMurray
Grande Prairie
Edmonton
Calgary
Lethbridge
1.0
0.8
0.6
0.4
0.2
0.0