bbsRDM: A Package for Calculating Regime Detection Measures on the N. American Breeding Bird Survey Data

2019-07-11

Contents

1 Setup and Overview

This document briefly reviews the functionality of **bbsRDM**. Although this package can be used to calculate and visualize BBS data using time series, the example at hand presents an application to large spatial transects across the continental United States.

First, install the bbsRDM and regimeDetectionMeasures packages from GitHub.

```
devtools::install_github("trashbirdecology/bbsRDM",
                         force = FALSE, dependencies = FALSE)
devtools::install_github("trashbirdecology/regimedetectionmeasures",
                         force = FALSE, dependencies = FALSE)
```

```
library(bbsRDM)
library(regimeDetectionMeasures)
library(sp)
library(raster)
library(feather)
library(here)
library(ggplot2)
```

```
ggplot2::theme_set(theme_bw())
```

Next, create diretories for storing the BBS and results locally (if directories exist, this BBS data will **not** be downloaded (unless you specify 'downloadBBSData=TRUE'), however, the **results will be overwritten**!).

```
# a. Create a directory to store and/or load the BBS data as feathers
bbsDir <- here::here("bbs_raw_data")
# If bbsDir is NOT empty then we will NOT download any data.
if (length(list.files(bbsDir, pattern = "*.feather")) > 0) {
downloadBBSData = FALSE
} else
(
{
dir.create(bbsDir)
downloadBBSData = TRUE
}
) # If this returns a warning, proceed with caution as directory already exists,
     # results WILL be written over
# b. Create a directory to store and/or load the BBS data as feathers
resultsDir <- here::here("myResults")
dir.create(resultsDir)
# c. Create directory for storing early warning signal results
dir.create(here::here("myResults/ews"))
# d. Create directory for storing distance travelled results
dir.create( here::here("myResults/distances"))
```

2 Retrieve the BBS Data and Save Locally

If necessary, download all the state data. Downloading the entire BBS data takes a while, so only run if you have not recently downloaded the BBS data. If you only need a subset of the data, please see the documentation for `bbsAssistant::get_regions` and `bbsRDM::getDataBBS`.

```
# a. Load the regional .txt file from Patuxent's FTP server
# (you must be connected to the internet to perform this step)
regions <- bbsAssistant::get_regions()

# b. Create a series or one filenames for states, regions
regionFileName <- regions$zipFileName %>% na.omit()

# c.  Download and unzip the BBS data.
if(downloadBBSData==TRUE){
for(i in 1:length(regionFileName)){
        bbsData <-  importDataBBS(
            # arguments for getDataBBS()
            file = regionFileName[i],
            dir =  "ftp://ftpext.usgs.gov/pub/er/md/laurel/BBS/DataFiles/States/",
            year = NULL,
            aou = NULL,
            countrynum = NULL,
            states = NULL,
            #  arguments for getRouteInfo():
            routesFile = "routes.zip",
            routesDir =  "ftp://ftpext.usgs.gov/pub/er/md/laurel/BBS/DataFiles/",
```

```
            RouteTypeID = 1,
            # one or more of c(1,2,3)
            Stratum = NULL,
            BCR = NULL
        )


# d. Save the unzipped files to disk.
birdsToFeathers(dataIn  = bbsData,
                newDir  = bbsDir,
                filename = regionFileName[i])
# e. Clear large object from memory
rm(bbsData)}}
```

3 Create a Spatial Sampling Grid Across North America

Next we build a spatial sampling grid for aligning BBS data to regularly spaced cells. This is important for spatial interpretation of the regime detection metric results.

```
# Define the grid's cell size (lat, long; unit:degrees)
        ## 1 deg latitude ~= 69 miles
        ## 1 deg longitude ~= 55 miles
cs <-
    c(0.5, 0.5)  # default is cell size 0.5 deg lat x 0.5 deg long

# Create the grid
routes_gridList <- createSamplingGrid(cs = cs)
```

Now we load in the BBS data from the feathers we created and align with the sampling grid. This requires a bit of memory, proceed with caution.

4 Subset the BBS data (optional)

4.1 By species (using AOU numbers)

Although subsetting the species is optional, this package contains features for subsetting by AOU code, functional groups, or by spatial location (e.g. remove all Montana observaitons).

Subset species according to AOU species codes (i.e. by family, genera, etc..)
For this example we will remove shorebirds, wading birds, and waterfowl (i.e., AOU species' codes 0000:2880).
*See `R/subsetByAOU.R` source code or documentation for options (see: `subset.by`)

```
# Subset the species
feathers <- subsetByAOU(myData = feathers, subset.by= 'remove.shoreWaderFowl')
```

4.2 Subset species according to functional traits (or body mass).

Note: eval = FALSE... change to true if you wish to evaluate in rmd knitting.

```
# Create a single list of mass and functional traits.
funMass <-
    funcMass(dataWD = here::here("data/"),
             fxn = TRUE, # get functional trait data?
             mass = TRUE) # get body mass data?
# Combine the functional trqits and/or body mass
```

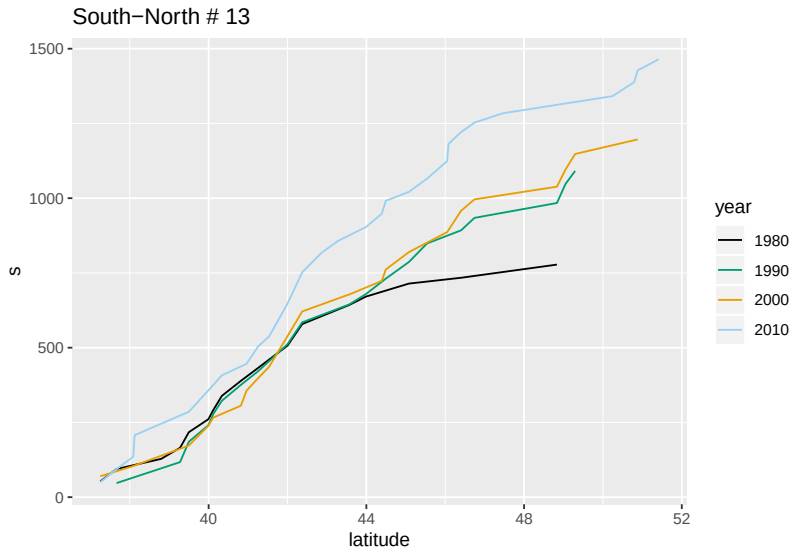

Figure 1: Distance traveled (s) by the bird community in a single, North-South-running transect.

```
bbsData <-
    mergeFunMassBBS(bbsData = feathers[1:1e3,], funMass = funMass)
```

5 Calculate regime detection metrics across space or time

5.1 First, define the parameters required to calcualte the metrics.

5.2 Define the years we want to analyze

For this (spatial) example, we will analyze only every tenth year

5.3 Calculate Regime Detection Measures Across Spatial Transects

This section will loop through `years.use` and `dir.use`, running each BBS route (temporal analysis) or spatial transect by year (spatial analysis) at a time. Results are saved in directories created in section below. **Please note**: depending on the # of years and # spatial transects, this could take a while.

6 Import and munge the results to prepare for visualization

First, use the function 'importResults' to import and combine the results as created in the previous code chunk. The following chunk chunk will import the EWS results and the distance results separately, combining each into their own data frames.

```
## [1] "I am importing 420 files. Does this sound right?!"
## [1] "I am importing 426 files. Does this sound right?!"
```

Next, get the results to align with our sampling grid for visualizing results across space.

7 Visualize Results: Temporally

First, specify plotting parameters (below). We can visualize either the distance results (`results_dist`) or the early-warning signal results (`results_ews`). For this example we will visualize the **distance** results.

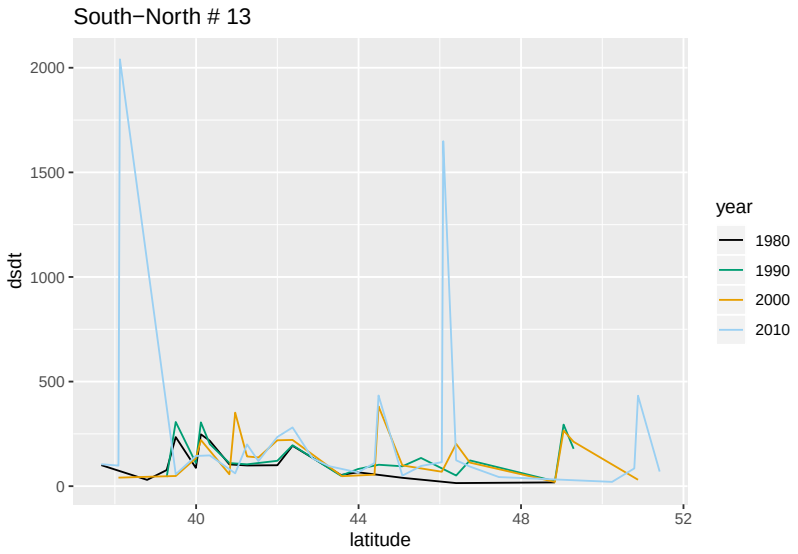

Figure 2: Velocity (rate of change, dsdt) of the distance travelled by the bird community in a single, North-South-running transect.

Plot the distance travelled, s for transect # 13 (Figure 1). This measure indicates how quickly the multivariable system is moving through phase space. Notice that the absolute value of s increases over time. This is unsurprising given that the number of species in BBS censuses has increased over time. The importance of this phenomenon should be evaluated at the local scale.

Plot the velocity of the distance travelled, *dsdt* for transect # 13 (Figure 2). Rapid changes in the velocity of the distance travelled indicates rapid species turnover, or large changes in the abundances of a few species. The ecological significance of rapid change in community abundances, or among the state variables used to calculate distance travelled and velocity, should be evaluated at local and sub-regional scales.

8 Visualize Results: Spatially

Plot the distance travelled across the entire range (Figure 3).

Plot the velocity of the distance travelled across the entire range (Figure 4).

9 Military Bases

Use the function `getMilBases()` to retrieve MIRTA military base point shapefiles:

```
milBases <- getMilBases() # default arguments will download the MIRTA FY 2018 points
## OGR data source with driver: ESRI Shapefile
## Source: "C:\Users\jburnett8\AppData\Local\Temp\RtmpU7RhOS", layer: "FY18_MIRTA_Points"
## with 810 features
## It has 7 fields
## Integer64 fields read as strings:  OBJECTID

sp::proj4string(milBases) <-
    sp::CRS("+proj=longlat +datum=WGS84")# Set projection to WGS84 lat long friendly
```

Next, we need to assign the row and column ID numbers from our sampling grid to each base location, such that we can visualize the results of the regime detection measures within our spatial sampling grid.

Figure 3: Distance traveled (s) by the bird community across all spatial transects in the sampling grid area.

Figure 4: Rate of change (dsdt) of the distance traveled metric (s) visualized for all spatial transects in our sampling area in the continental U.S.

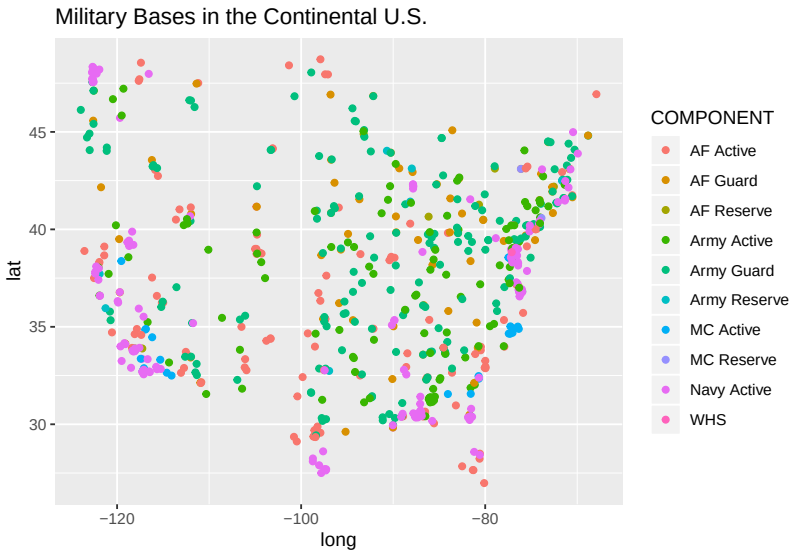

Figure 5: Military bases in the sampling area. Points are based on MIRTA FY 2018 military base locations.

```
# Extract the CELL ID in which each military base falls.
milBases.df <- milBases %>% as.data.frame() %>%
    rename(long = coords.x1, lat = coords.x2)
milBases.df$cellID <-  milBases %over%  routes_gridList$sp_grd

# remove bases outside our sampling grid
milBases.df <- milBases.df %>% filter(!is.na(cellID))
```

Plot the location of military bases (Figure 5).