Tutorial for using the speciesgeocodeR package

Alexander Zizka, Antonelli, etc., February 17, 2014

Abstract The purpose of this manual is to provide easy-to-use instructions for the speciesgeocodeR package for R beginners, and to explain the functionality of the package to more advanced users. The specisgeocodeR package is a collection of R functions designed to link geographical point coordinates with geographical polygons. It is a tool especially suited for biogeographical and ecological questions, i.e the distribution of specimen and species in geographic regions. Based on a list of point coordinates (specimen occurences) and a list of points building the polygons (geographical areas) speciesgeocodeR handles the classification geographic points to geographical areas (point in polygon test) and runs multiple statistical analyses to summarize the results. This includes a nexus file to use the results as direct input for biogeographic analysis linked with phylogenetic trees, as well as summary tables, -graphs and geographical maps. Chapter one of this manual specifies the input-file format, chapter two provides a 3-step tutorial to run a standard analysis, chapter three describes how to use the WWF ecoregions as input polygons and chapter four describes the standard outputfiles. Chapter five is a step-by-step commented script for the use of single functions, chapter six contains a discription of all function in the package and chapter seven shows results of benchmarking tests to give an idea of the computational time needed depending on the size of the dataset. SpeciesgeocodeR is the R implication of the python speciesgeocoder program (**?**).

Contents

1 Input files

All results produced with speciesgeocodeR are based on two input files that can easily be generated using freely available textformatting and GIS software: a list of sample coordinates (e.g. GPS-coordinates of specimen, herafter called “speciesdata”) and a list of polygon points (e.g biomes or national park borders, herafter called “polygondata”). The package is designed and tested for small-scale (e.g. radiocolar data in combination with national park borders) and large scale (e.g global distribuiton of all birds to biomes) analyses. As an additional feature speciesgeocodeR allows the use of the WWF- Realm/Biome/Eoregion classification as input polygons (see chapter 3).

Input-files: The two input files are required to be tab-delimited .txt files with three columns each. For both files, column names must be: identifier, XCOOR, YCOOR. In case of the speciesdata the first column contains the species name (or individual name), the second column contains the longitude coordinates (as decimal degrees, e.g. 59.867 for East, or -59.867 for West) and the third column contains latitude coordinates (as decimal degrees, e.g. 59.867 for North, or -59.867 for South). The polygondata should be in the same format: each row represents the coordinates of one polygon corner. The first column contains the polygon identifier, the second one the longitude coordinates, the third one the latitude coordinates. It is important that the coordinates of the first and the last point of each polygon are identical. You can easily export such a file from any GIS program (e.g. QGIS: `http://www.qgis.org/en/site/`). If you use polygondata exported from QGIS or formated for the python version of speciesgeocoder you can use the ConvertPoly() function to convert the format.

When preparing the input files please make sure:

1. that the order of the columns and the column headers are correct (“identifier”, “XCOOR”, “YCOOR”)

2. that your coordinates conform with the limits of the coordinate system (XCOOR between -180 and 180; YCOOR between -90 and 90)

3. that for each polygon in the polygon data the first and last point have identical coordinates.

2 All-in-one: the quick and easy way

You can run a standard speciesgeocoder analysis with only three commands. The following paragraph will give a simple tutorial for R beginners. Two example scripts (hereafter called #1 and #2) are given at the end of the chapter. The example files are also provided in a .txt file with the package, so if you are unfamiliar with the R environment, you can just copy the Example code from there to the R consol (do not forget to change the paths given in capitals). If you are interested in a step by step process or in the use of single functions refer to chapters five and six. The #numbers refer to codelines in the example scripts:

1. Create a new folder in your home directory (the working directory). Copy the speciesgeocoder package (speciesgeocodeR.R) and your two input files into the folder and give them characteristic names (here we will use "point_coordinates.txt" for our speciesdata and "polygon_coordinates.txt" for the polygondata). If you are insecure about the format of your input files or an error occurs right away, check chapter one of this manual or take look at the example files delivered with the package.

2. Start R

3. Tell R where to find the input files and save the outputfiles (#1.1). You must put the exact path of the folder in the quotation marks. If you use mac (or linux) use / instead of \\.

4. Load the functions of the speciesgeocoder package into your R session (#1.2)

5. Execute the SpeciesGeoCoder() function with the names of your two input files as arguments (#1.3). Depending on the size of your dataset (especially the number of polygons) this might take a while. The graphs argument controls if graphical output is produced (default = T), the"coex"' argument defines if a coexistence matrix for all species in each polygon is produced (default = F, this is only recommended for datasets with small to medium size species number, as it is computationally very expensive)

6. Done! Close R, the outputfiles are in your working directory. Summary tables are saved as tab-delimited .txt files, graphs and maps as .pdf files. If problems occur, check your input-files or try using one of the example datasets delivered with the package. If this does not help, you might want to use the step-by-step procedure described in chapter five.

7. If your polygondata input file is exported from QGIS and/or is formated for use with the python version of speciesgeocodeR, you can use the ConvertPoly() function to convert it to a format suitable for use with speciegeocodeR (#2)

Example #1

Line	Code	Explanation
1.1	*setwd("YOUR WOKINGDIRECTORY PATH")*	Define where to load and save files, e.g. "C:\\Desktop\\data"
1.2	*source("speciesgeocodeR.R")*	load the speciesgeocodeR package
1.3	*SpeciesGeoCoder("SPECIESDATA", "POLYGONDATA", graphs = T, coex = F)*	run SpeciesGeoCoder(), e.g "point_coordinates.txt", "polygon_coordinates.txt"

Example #2

Line	Code	Explanation
2.1	*setwd("YOUR WOKINGDIRECTORY PATH")*	Define where to load and save files, e.g. "C:\\Desktop\\data"
2.2	*source("speciesgeocodeR.R")*	load the speciesgeocodeR package
2.3	*poly <- ConvertPoly("POLYGOND INPUTFILE")*	load and convert your polygon file
2.4	*write.table(poly, "polygon_coordinates.txt", row.names = F, sep = "\t")*	write the new input-file
2.5	*SpeciesGeoCoder("SPECIESDATA", "POLYGONDATA", graphs = T, coex = F)*	run SpeciesGeoCoder(), e.g "point_coordinates.txt", "polygon_coordinates.txt"

3 Using the WWF Realms/Biomes/Ecoregions as input polygons

You can use speciesgeocodeR to produce the input polygons for your analysis directly from the global WWF definition of terrestrial Ecoregions (and Realms and Biomes, **?**). To do so, download the source files (e.g. from `https://worldwildlife.org/publications/terrestrial-ecoregions-of-t` and extract them to your working directory. After you loaded them into R (#3.3), you can use the WWFpick() function to create the polygon input object for SpeciesGeoCoder() (#3.4). To use it run SpeciesGeoCoder() with a normal input table for the pointdata and with the object containing result of WWFpick() as polygondata (#3.5). The WWFpick() function needs three input objects (#3.4). When you run SpeciesGeoCodeR() with WWF input polygons you must specify to additional arguments; “wwf = T” and scale =“FOCUS SCALE”. The scale arguments defines the scale of clustering of the results (this must be one of“ECOREGION”, “BIOME”, “REALM”, #3.5):

1. The WWF shape file stored in an R object, “wwf” in the example

2. The name of the region of interest. Depending on the scale this for example could be “Neotropics”, “Temperate Conifer Forests” or “Ethiopian montane moorlands”. It must be a character string included in the scale specified in “scale”. You can use the WWFnam() function on the shape file object to display all names available (#3.6). Vectors of names are also possible, e.g. c(“Neotropis”, “Afrotropis”) or ”all” (giving all polygons for the specified scale)

3. The scale of interest. This must be one of the following “REALM”, “BIOME”, “ECOREGION”.

Example # 3 WWF terrestrial ecoregions as input polygon

Line	Code	Explanation
3.1	*setwd("YOUR WOKINGDIRECTORY PATH")*	e.g. C:\\User\\Desktop
3.2	*source("speciesgeocodeR.R")*	load the speciegeocodeR package
3.3	*wwf <- readShapeSpatial("NAME OF THE SHAPE FILE")*	load the WWF shape, name e.g "wwf_terr_ecos.shp"
3.6	*WWFnam(wwf)*	Display the area names available for WWFpick()
3.4	*poly <- WWFpick(wwf, name = "FOCUS REGION", scale = "FOCUS SCALE")*	pick the polygons of interest, e.g name = "Neotropics", scale = "REALM"
3.5	*SpeciesGeoCoder("SPECIESDATA", poly, graphs = T, coex = F, wwf = T, scale = "FOCUS SCALE FOR CLUSTERING")*	run SpeciesGeoCoder(), e.g "point_coordinates.txt", "polygon_coordinates.txt"

4 Output-file description

With the “graphs” and “coex” options swiched on, SpeciesGeoCoder() produces 13 outputfiles (1 nexus file, 5 summary tables, 3 barcharts, 3 maps and 1 heatplot). By default all files are saved to the working directory defined before the analysis (#1.1). The summary tables are tab delimited .txt files and all graphics and maps are put out as .pdf files. Depending on the size of you dataset, some of the graphical representations might not be adequate.

Barchart_per_polygon.pdf

This file contains one barchart for each input polygon showing the total number of occurences for each species in this polygon.

Barchart_per_species.pdf

This file contains one barchart per species, showing the relative number of occurences in each input polygon.

Heatplot_coexistence.pdf

This file contains a heat plot showing the coexistence pattern of all species in the analysis. This output is turned of by default. To turn it on use “coex = T” as argument of the SpeciesGeoCoder() function.

Map_samples_overview.pdf

This file contains an overview map showing all input points and polygons.

Map_samples_per_polygon.pdf

This file contains one map for each input polygon, showing all occurence points included in the polygon colour-coded for species.

Map_samples_per_species.pdf

This file contains one map for each input species showing all polygons and all occurrence points of this species. Point samples classified to any polygon are shown in blue, unclassified samples are shown in red.

Numer_of_species_per_polygon.pdf

This file contains a bargraph indicating the number of species per polygon.

Species_classification.nex

This is a nexus file containing the area coding for each species (1 = occurence, 0 = absence) for biogeographic analysis in combination with phylogenetic data.

Sample_classification_to_polygon.txt

This is a summary table showing the classification for each sample point.

Species_occurrence_per_polygon.txt

This is a summary table showing the presence or absence of every species in the input polygons.

Species_number_per_polygon.txt

This is a table summarizing the number of species for each polygon.

Unclassified_samples.txt

This file contains a map showing all samples that could not be clssified to any of the input polygons.

5 Step-by-step: Specific output files

This is to be written if necessary

6 Function description

BarChartPoly(x, plotout, . . .) *Input:* x = a object of class spgeoOUT, plotout = logical indicating if the function will be viewed in the R graphics device or passed to another output function, default is false *Function:* this function gives out one bar chart per input polygon showing the number of occurrences for each species in this polygon. *Output:* a series of bar charts

BarChartSpec(x, plotout,)

Input: x = a object of class spgeoOUT, plotout = logical (T/F)

Function: this function gives out one bar chart per input species (or identifier) showing the number of occurrences in each polygon. Plotout indicates if the function will be viewed in the R graphics device or passed to another output function, default = FALSE.

Output: a series of bar charts

CoExClass(x)

Input: x = class spgeoOUT

Function: this applies CoExClassH() to an object of class spgeoOUT, to add a species coexistence matrix to the object. This considers only species classified to polygons!

Output: additional slot of the spgeoOUT object containing the coexistence matrix

CoExClassH(x)

Input: x = data.frame of species occurrence numbers per polygon; ncol = number of polygons, nrow = number of species

Function: calculates a coexistence matrix of species. The number indicates the percentage of the species in the row co-occurring with the species in the column species in the row. This is the percentage of species classified to polygons!

Output: data.frame, ncol= number of species + 1, nrow = number of species. The numbers indicate the percentage of the species in the row co-occurring with the species in the column.

ConvertPoly(x)

Input: a character vector indicating the path to a .txt file with polygon information exported from QGIS

Function: This function re-formats the dataframe to be suitable as input for the SpeciesGeoCoder() function.

Output: a data.frame that can be exported for used as polygon input file for the SpeciesGeocoder() function

Cord2Polygon(x)

Input: x = a tab delimited text file in the working directory OR a data.frame containing the Polygon information. Three columns: identifier, XCOOR, YCOOR.

Function: creates spatial polygon objects out of point lists; Projection: latlon, wgs84.

Output: SpatialPolygons object (=list of polygons).

CropPointCountry(x,y)

Input: x = a data.frame with coordinates of points of interest, 3 columns: identifier, XCOOR, YCOOR; y = a string or vector of strings with the countries of interest.

Function: crops the point in x to the country borders of y, taken from the wrld-simpl data from the maptools package.

Output: a data.frame, with points within the country borders

CropPointPolygon(points, polygon, outside)

Input: points = data.frame, 3 columns: identifier XCOOR, YCOOR, polygon = polygon of interest as SpatialPolygons object, outside = logical value, T = points inside the polygons, F = points outside the polygons.

Function: crops a data.frame with point coordinates to a polygon. Depending on outside it returns either a data.frame with the points lying inside or outside the polygon.

Output: data.frame, with point coordinates

***GetPythonIn(coordinates, polygon, sampletable, speciestable)**

Input: coordinates =table of point coordinates, polygon = table of polygon coordinates, sampletable = results of species classification to polygons, speciestable = table of summary per species.

Function: this is a helper function to pass the objects from the python code of the species-geocoder program.

Output: object of class spgeoOUT

HeatPlotCoEx(x,)

Input: x = a object of class spgeoOUT OR a data.frame with ncol = identifier, number of

species/identifiers; nrow = number of species/identifiers

Function: creates a heatplot from the coexistence matrix.

Output: a single heat/level plot.

MapAll(x,polyg,moreborders)

Input: x = an object of class spgeoOUT, than polyg is not necessary, OR a matrix/data.frame of point coordinates 2 columns: XCOOR, YCOOR; polyg a SpatialPolygons object, with polygons of interest, moreborder = logical (T/F).

Function: if method spgeoOUT than it maps all points and all polygons in the object. If method = matrix plots the points together with the polygons. The moreborders option adds borders of countries younger than 1990, default = F.

Output: a plotted map

MapPerPoly(x, plotout)

Input: x = an object of class spgeoOUT; plotout = logical (T/F)

Function: creates a series of maps, showing each polygon and its close environment, with all samples classified to this polygon. Species are color-coded. Plotout indicates if the function will be viewed in the R graphics device or passed to another output function, default = FALSE.

Output: a series of map plots.

MapPerSpecies(x, moreborders, plotout)

Input: x = an object of class spgeoOUT; moreborederss = logical (T/F); plotout = logical (T/F).

Function: creates one map per species/identifier, showing all occurrence points related and all polygons. Plotout indicates if the function will be viewed in the R graphics device or passed to another output function, default = F. The moreborders option, add borders of countries younger than 1990, default = F.

Output: a series of plotted maps

MapUnclassified(x, moreborders,)

Input: x = object of class spgeoOUT, moreborederss = logical, should additional borders be added?

Function: creates a map of all input samples that could not be classified to one of the input polygons. The moreborders option, add borders of countries younger than 1990, default = F.

Output: a plotted map

NexusOut(x, verbose)

Input: x = an object of the class spgeoOUT, verbose = logical(T/F)

Function: this function creates a table of the species classification to each area in nexus format for the further use in combination with phylogenetic trees. If verbose = T (default = F) the number of occurences per area are included in the file.

Output: a .nex file in the working directory.

***OutBarChartPoly(x,)**

Input: x = object of class spgeoOUT

Function: produces a .pdf with a bar chart for each polygon showing the number of occurrences for all input species in this specific polygon. This is a helper function producing the speciesgeocoder standard pdf from BarChartPoly().

Output: barchart_per_polygon.pdf in the working directory

***OutBarChartSpec(x, . . .)**

Input: x = object of class spgeoOUT

Function: produces a .pdf with a bar chart for each species showing the occurrence in all input polygons. . This is a helper function producing the speciesgeocoder standard pdf from BarChartSpec().

Output: barchart_per_species.pdf in the working directory

***OutHeatCoEx(x,)**

Input: x = object of class spgeoOUT

Function: produces a .pdf showing a heat plot, representing the coexistence pattern between the input species. This is a helper function producing the speciesgeocoder standard pdf from HeatPlotCoEx().

Output: heatplot_coexistence.pdf in the working directory

***OutMapAll(x,)**

Input: x = object of class spgeoOUT

Function: produces a .pdf showing a map of all points and a map of all points not classified to any polygon. This is a helper function producing the speciesgeocoder standard pdf from MapAll() and MapUnclassified()

Output: map_samples_overview.pdf in the working directory

***OutMapPerPoly(x,)**

Input: x = object of class spgeoOUT

Function: produces a .pdf showing a map of each polygon and its close environment with, all points that were classified to this polygon, color-coded by their identifier. This is a helper function producing the speciesgeocoder standard pdf from MapPerPoly().

Output: map_samples_per_polygon.pdf in the working directory

***OutMapPerSpecies(x,)**

Input: x = object of class spgeoOUT

Function: produces a .pdf showing a map for each species with all polygons individuals of this species where classified to. This is a helper function producing the speciesgeocoder standard pdf from MapPerSpecies().

Output: map_ per_species.pdf in the working directory

***OutPlotSpPoly(x,)**

Input: x = object of class spgeoOUT

Function: produces a .pdf showing a bar chart of the species number per polygon. This is a helper function producing the speciesgeocoder standard pdf from PlotSpPoly.

Output: number_of_species_per_polygon.pdf in the working directory

PipSamp(x)

Input: class spgeoIN

Function: calculates the homepolygon for each gives out a data.frame with the home polygon for each sample

Output: data frame; 2 columns: sample Id, containing polygon

PlotOutSpGeo(x,)

Input: x = an object of the class spgeoOUT.

Function: This function calls all output functions, producing the standard package of output plots for speciesgeocoder.

Output: .pdf files in the working directory

PlotSpPoly(x,)

Input: x = object of class spgeoOUT.

Function: creates a bar chart on species numbers per polygon.

Output: a bar chart.

PointInPolygon(x,y)

Input: x = data.frame with point coordinates; 3 columns: identifier, XCOOR, YCOOR ; y = data.frame with polygon points, 3 columns: identifier,

Function: performs a point in polygon test, and produces a table indicating the name of home-polygon of each sample as character string.

Output: data.frame, 2 columns: identifier, homepolygon.

ReadPoints(x,y)

Input: 2 text files in the working directory: x = coordinates of the points of interest, y = the coordinates of the polygons. 3 Columns: identifier, XCOOR, YCOOR.

Function: creates an object of the class spgoeIN (= list of pointidentifiers, point coordinate, spatial polygons

Output: object of class spgeoIN

SpeciesGeoCoder(x, y, coex, graphs)

Input: 2 text files in the working directory: x = coordinates of the points of interest, y = the coordinates of the polygons. 3 Columns: identifier, XCOOR, YCOOR.

Function: the main function of the package. This performs the whole species geocoder and produces the standard package of output files. Produces a number of standard output-tables containing: 1. classification of all samples to a polygon, 2. Summary of species (identifier) occurrence per polygon, 3. A table of samples that could not be classified to any of the input polygons, 4. a Nexus-file, including the species classification, 5. A coexistence matrix, showing to which percentage species to co-occurr in the input polygons, 5. A table giving species numbers per polygon. Furthermore produces a set of .pdf files in the output directory: 1. A barchart showing the number of species per polygon, 2. a barchart summarizing numbers of each species for each polygon, 3. A bar chart summarizing the occurrences in each polygon per species, 4. A map of all polygons with the points classified to them, colored for species (identifier), 5. a map of the occurrences of all species, A map showing all points used and all unclassified points in the geographic context, 6. A heat plot showing the co-occurrence pat-

terns of all species in the polygons.
Output: multiple tab delimited tables and graphics in pdf format.

SpGeoCod(x, y)
Input: x = .txt file containing the information of species occurrence, 3 columns: identifier,XCOOR,YCOOR; y = points comprising the polygons, 3 columns: identifier, XCOOR, YCOOR.
Function: this function combines ReadPoints() and SpGeoCodH() to produce an object of the class spgeoOUT from input text files.
Output: object of class spgeoOUT

SpGeoCodH(x)
Input: object of class spgeoIN.
Function: this is one of the core functions. It uses an object of the class spgeoIN and performs appoint in polygon test classifying each species to a polygon, summarizes the information per species, summarizes the number of species per polygon, produces a table showing the samples that could not be classified and calculates a coexistence matrix. These objects are then put together with the input information to an object of the class spgeoOUT.
Output: spgeoOUT

spPerPol(x)
Input: class spgeoIN
Function: combines PipSamp(), SpSumH and SpPerPolH;
Output: a named vector with species number per

***spPerPolH(x)**
Input: data.frame, ncol = number of polygons, nrow = number of species. This is a helper function, which takes input directly from SpSumH().
Function: calculates the number of species per polygon
Output: a named vector, with species number per polygon.

spSum(x)
Input: x = object of class spgeoIN
Function: calculates a summary of the number ofspecies occurences per polygon.
Output: a data.frame summarizing the occurrence number of species per polygon; ncol = num-

ber of polygons, nrow = number of species

***spSumH(x)**

Input: x = class spgeodataframe, this is a helper function to take output from pipSamp

Function: calculates a summary of species occurrences from a 2 column table: identifier, hom-polygon

Output: A data.frame summarizing the occurrence number of species per polygon.

Uses output of pipSamp to give out a dataframe of number of species occurrences per polygon, ncol = number of polygons, nrow = number of species

WriteTablesSpGeo(x,)

Input: object of class spgeoOUT

Function: creates 5 output - tables as tab delimited text files in the working directory: 1. classification of all samples to a polygon, 2. Summary of species (identifier) occurrence per polygon, 3. A table of samples that could not be classified to any of the input polygons, 4. A coexistence matrix, showing to whichpercentage species to co-occurr in the input polygons, 5. A table giving species numbers per polygon

Output: tab delimited .txt files in the working directory.

WWFconvert(x)

Input: x = SpatialPolygonsDataframe object of the biomes and ecoregions of the WWF or any subset of it. Available from http://maps.tnc.org/gis

data.html, and loaded to R with readShapeSpatial() from the raster package

Function: converts the polygons in the object to a list of points

Output: data.frame, 3 columns, identifier,XCOOR,YCOOR

WWFnam(x)

Input: x = a SpatialpolygonsDataFrame of the biomes and ecoregions of the WWF or any subset of it. Available from http://maps.tnc.org/gis

data.html, and loaded to R with readShapeSpatial() from the raster package

Function: shows all available subset categories for the of the dataset for WWFpick() and WWFpoly2point().

Output: A list of strings

WWFpick(x, name, scale)

Input: x = a SpatialpolygonsDataFrame of the biomes and ecoregions of the WWF or any subset of it. Available from http://maps.tnc.org/gis
data.html, and loaded to R with readShapeSpatial() from the raster package; name = name of the realm, biome or ecoregion of interest, scale = character string indicating the scale of interest (one of: REALM, BIOME, ECOREGION).

Function: creates a subset of the WWF ecoregion Polygons

Output: class SpatialPolygonsDataFrame

WWFpoly2point(x, name, scale)

Input: x = a SpatialpolygonsDataFrame of the biomes and ecoregions of the WWF or any subset of it. Available from http://maps.tnc.org/gis
data.html, and loaded to R with readShapeSpatial() from the raster package; name = name of the realm, biome or ecoregion of interest, scale = character string indicating the scale of interest (one of: REALM, BIOME, ECOREGION).

Function: combines WWFpick() and WWFconvert(). Produces a list of points defining the realm, biome or ecoregion defined in the input.

Output: data.frame, 3 columns, identifier,XCOOR,YCOOR

7 Benchmarking