**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.

\_

**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 = 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: 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 speciesgeocoder 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, add borders of countries younger than 1990. Default is 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; moreboreders = 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 = FALSE. The “moreborders” option, add borders of countries younger than 1990, default = F.

Output = a series of plotted maps

\_

**Mapping(x, pdataf, mode, pointmode, scale, name, xmin, xmax, ymin, ymax)**

Input:x = object of class spgeoOUT (mode “spgeoOUT”) or a SpatialPolygons object (mode = “dataset”), in the second case pdataf must be a data.frame with point coordinates and the following column names: identifier, XCOOR, YCOOR. Pointmode = character string (“all” or “classified”) for mode = spgeoOUT and defines if all samples or only classified samples should be shown. Scale = character string (“auto\_extend”, “extend”, “world”, “country“); default = auto\_extend. Name = character string or vector of characters for scale = “ountry”, defining the countries to be plottet; xmin,xmax,ymin,ymax = for scale = extent, defining the limits of the maps.

Function: This function offer multiple options to produces maps from point data on a backgroundmap of political borders.

Output: a plotted map

\_

**MapUnclassified(x, moreborders, …)**

Input: x = object of class spgeoOUT, moreboreders = 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

This function remains to be defined

\_

**\*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\_samples\_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 homepolygon 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. NO HEADERS!

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

Output: Object of class spgeoIN

\_

**SpeciesGeoCoder(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. NO HEADERS!

Function: 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 coexistence matrix, showing to whi 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 patterns 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 = number 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, hompolygon

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 whi percentage 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 WWFpcik() 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