Package ‘rSPACE’

October 3, 2014

Type Package

Title Spatially-explicit Power Analysis for Conservation and Ecology

Version 0.0.2

Date 2014-10-01

Author Martha Ellis, Jake Ivan, Mike Schwartz

Maintainer Martha Ellis <martha.ellis@gmail.com>

Description This package provides subroutines to run a spatially-
explicit power analysis for detecting population trends through occupancy-based modeling.

License GPL (>=2)

Imports raster, RMark, ggplot2, tcltk2, sp, grid, plyr, tcltk

R topics documented:

`rSPACE-package` *Spatially-explicit Power Analysis for Conservation and Ecology*

Description

This package provides subroutines necessary to run a simple, spatially-explicit population simulation and then conduct a power analysis for detecting population trends through occupancy-based modeling.

Details

Package:	rSPACE
Type:	Package
Version:	1.0
Date:	2013-10-15
License:	GPL (>=2)

There are many possible options for how a simulation and power analysis could be conducted. We set up this package to provide options for how to conduct both the simulations and the data analysis. We would like to help make this framework accessible, but we also acknowledge that it may not always work in situations for which the software was not originally designed. We will be working to make more of the functionality accessible as we go along and will be interested to hear how things work (or fail to work) for anyone else who is interested in using our program.

For now, we have only made the basics of the package available. We originally designed our population simulation for territorial carnivores. Given an underlying landscape, we distribute individuals according to a number of spacing rules (e.g. home range size, percent overlap, etc). We use bivariate normal homeranges, adjusted based on the underlying habitat/landscape layer, to build a probability of use layer for each individual. We then combine the individual probabilities to create a layer describing the probability of use for at least one individual for the entire landscape. The landscape is then gridded into cells and used to create an encounter history at each possible grid cell.

There are a large number of parameters needed to set up this stage of the analysis. The parameters are stored as a list, which can either be entered separately or we have set up a dialog box to create it (for details see `enter.parameters`). The function, `encounter.history`, takes the list of parameters and a raster map of the landscape to produce a single encounter history file. Or you can use `create.landscapes` to create replicated encounter histories. The encounter histories will be created as text files (.txt) that can either be used as inputs for Program MARK, etc or read back in later in the power analysis side of our simulations.

Just as there are many options for simulating a population on a landscape, there are many ways to potentially analyze each encounter history. The way our analysis is set up, we have one wrapper function `test_samples` set up to subset the encounter history files to simulate varying sampling effort. `test_samples` takes as an argument a function name that will define the analysis that is run on each potential encounter history. This function should take the encounter history and some information about that file as arguments and return a data frame with the simulation results. We have provided the original test file that we used to analyze simulated wolverine occupancy in Ellis et al. (2013) in `wolverine_analysis`. The final output of the simulations at this stage is a text file storing simulation results, based on the output of `wolverine_analysis`.

There are many options at each of these steps, but the basic process is:

1. Enter parameters `enter.parameters`
2. Build encounter history files `create.landscapes`
3. Analyze encounter histories `test_samples`

The help files will focus on these three steps, and explain additional details needed at each step.

Author(s)

Martha Ellis, Jake Ivan, Mike Schwartz

Maintainer: Martha Ellis <martha.ellis@gmail.com>

References

ELLIS, MARTHA M., JACOB S. IVAN, and MICHAEL K. SCHWARTZ. "Spatially Explicit Power Analyses for Occupancy-Based Monitoring of Wolverine in the US Rocky Mountains." Conservation Biology (2013).

Examples

```
#Parameters<-enter.parameters()
  #create.landscapes(10, map=map, Parameters=Parameters)     # Will need to have a map to do this.
   #test_samples(main_wd,folder,Parameters,function_name="test_file")   ##Impossible!
```

`create.landscapes` *Wrapper function to create replicate simulated landscapes*

Description

Creates a series of files with simulated encounter histories for using in a mark-recapture analysis based on population parameters.

Usage

```
create.landscapes(n_runs, map, Paramters, ... )
```

Arguments

`n_runs`	number of replicate encounter histories to produce
`map`	a raster layer with underlying habitat information
`Parameters`	list of parameter values for simulation (see `enter.parameters`)
`...`	additional options

Details

This function will produce `n_run` text files with simulated encounter histories. Each individual file does not contain identifying information about the parameters used to create it, so it will be good practice to create a folder structure to keep separate simulations separate. Default is to create a folder for the runs (./rSPACE_X) and name all encounter histories ./rSPACE_X/rSPACEx??.txt where ?? indicates the run number. If `Paramters` is not specified, `enter.parameters` is called by default.

`map` will define the landscape for the population simulation. Values of `map` are used as weights for both selecting home range centers and scaling bivariate normal home ranges.

There are additional arguments that can be included to specify other options, but some are more sketchy than others.

Value

returns list with directory location and output file names

Author(s)

Martha Ellis

References

ELLIS, MARTHA M., JACOB S. IVAN, and MICHAEL K. SCHWARTZ. "Spatially Explicit Power Analyses for Occupancy-Based Monitoring of Wolverine in the US Rocky Mountains." Conservation Biology (2013).

`encounter.history`	*Produce a single encounter history file based on population parameters.*

Description

Given an underlying landscape, populates the landscape with simulated individuals based on spacing rules in `Parameters`. Calculates probability of at least one individual at every location on the landscape and accumulates probability over grid cells to determine presence/absence.

Usage

```
encounter.history(Parameters, map, grid_layer, n.cells)
```

Arguments

`Parameters`	list of parameter values for simulation
`map`	a raster layer with underlying habitat information
`grid_layer`	a vector the length of the number of pixels in `map` indicating a grid cell index for each pixel
`n.cells`	total number of grid cells in sampling frame

Details

Called through `create.landscapes`, but can be used as a standalone function to produce a single encounter history for testing.

Value

a matrix with `n.cells` rows and `Parameters$n_yrs*Parameters$n_visits` columns with 1/0 indicating presence/absence of at least one individual in each cell at each visit.

See Also

`create.landscapes`

`enter.parameters` *Enter parameters*

Description

Dialogue box for help setting up a parameter list

Usage

```
enter.parameters(Parameters = NULL)
```

Arguments

`Parameters`	Optional parameters to use as defaults in clicky box.

Details

Creates a minimal set of parameters necessay to run simulations and analysis. Additional arguments can be added to the outputted list as needed (see Examples).

Parameter definitions

Population parameters	————-
Initial population size	Population size at the start of the simulation
Population growth rate	Annual population multiplication rate (Lambda=N_t/N_0)
Number of groups	Types of individuals to treat separately in the population simulation
Proportion of population	Proportion of population in each group (e.g. sex ratio)
Movement parameters	————-
Buffer distance	Minimum distance between movement centers for each type of individual
Minimum habitat value	Cutoff value for where individual centers should be located
Movement radius	With percent of time in radius, used to define bivariate normal use distribution
Percent of time in radius	With movement radius, used to define bivariate normal use distribution
Truncation	Limits to long-distance movements
Sampling parameters	————-
Number of years	Maximum number of years for the simulation
Number of visits	Maximum number of visits to each grid cell per year
Cell size	Area of grid cell
Percent of habitat	Cutoff for home much habitat must be included in cell to sample

Value

A parameter list with the following items

N Initial population size

lmda Population growth rate

n_yrs Maximum number of years in the simulation

n_visits Maximum number of visits per year

grid_size Area of each grid cell

HRcenter.cutoff Cutoff for value of `map` in `create.landscapes` where individual movement centers should be located.

sample.cutoff Cutoff sum of `map` values required in a grid cell for that cell to be included.

buffer Distance between individual movement centers for each type of individual

howmuch Percent of time within movement radius for each type of individual

howfar Movement radius for each type of individual

trunk Truncation value (SDs from movement center to truncate)

MFratio Proportion of population by type of individual

detP Initial detection probability (By default, encounter histories are created with perfect detection (detP=1), then sampled with imperfect detection via `test_samples`)

Note

Please direct all complaints about the damn clicky box to Mike Schwartz

Author(s)

Martha Ellis, Mike Schwartz

References

ELLIS, MARTHA M., JACOB S. IVAN, and MICHAEL K. SCHWARTZ. "Spatially Explicit Power Analyses for Occupancy-Based Monitoring of Wolverine in the US Rocky Mountains." Conservation Biology (2013).

`find.power` *Predict power from an rSPACE simulation and analysis*

Description

Uses a loess smoother to predict either the number of grid cells needed to achieve a given level of power or the expected power from a specific sample size based on rSPACE output.

Usage

```
find.power(folder, pwr = 0.8, CI = 0.95, n_grid = NULL)
```

Arguments

`folder`	Full path for rSPACE folder containing both simulation and analysis output
`pwr`	If specified, desired power
`CI`	Significance level
`n_grid`	If specified, number of grid cells used

Details

Depending on whether `pwr` or `n_grid` are specified, `find.power` will return the opposite. If both `pwr` and `n_grid`, `n_grid` will be ignored

Value

Either the number of cells required to reach a given power or the power expected from a sample of a given number of cells.

Author(s)

Martha Ellis

`plot.results` *Plot power curves*

Description

Plot power curves for representing each subset in a specified rSPACE folder

Usage

```
plot.results(folder, CI = 0.95, returnData = T)
```

Arguments

`folder`	Full path for rSPACE folder containing both simulation and analysis output
`CI`	Significance value for two-tailed test
`returnData`	Logical: return contents of results file as a dataframe?

Value

If returnData is TRUE, return contents of results file as a data.frame. Otherwise, just creates figure.

Author(s)

Martha Ellis

`test_samples` *Analyze a set of encounter histories*

Description

For a folder containing simulated encounter histories, reads in encounter histories, subsets data if applicable, and runs a function on each encounter history. Results are accumulated into a data.frame and written into a results output file.

Usage

```
test_samples(main_wd, folder, Parameters, function_name = "wolverine_analysis", ... )
```

Arguments

`folder`	Folder containing encounter history files (full path).
`Parameters`	list of parameter values
`function_name`	name of function specifying analysis on each subsetted encounter history
`...`	additional arguments

Details

The main role of `test_samples` is to read in encounter histories, subset them based on varying sampling effort, and then run an analysis on each "observed" encounter history (e.g. subsetted file). The actual analysis that gets applied will be determined by the function specified with `function_name`. The default function, `wolverine_analysis`, includes code from the original analysis used in Ellis et al. (2013).

Options for subsetting the encounter histories can be specified via `Parameters`. For example, the number of visits per year to test can be specified by `Parameters$n_visit_test`, the percent of the grid included in the sample `Parameters$grid_sample`, the detection probability per visit `Parameters$detP_test`, and the sampling scheme by year `Parameters$sample_yrs`. If these objects are not in the `Parameters` list, the default values are:

- `Parameters$n_visit_test = 2:Parameters$n_visits`
- `Parameters$detP_test    = c(1,0.8,0.2)`
- `Parameters$grid_sample  = c(0.05, 0.15, 0.25, 0.35, 0.45, 0.55, 0.75, 0.95)`
- `Parameters$sample_yrs   = c(0,1)`

For `Parameters$sample_yrs`, codes will need to match analyses in the function specified by `function_name`. Using `wolverine_analysis`, `Parameters$sample_yrs = 0` indicates continuous sampling and `Parameters$sample_yrs = 1` indicates alternate year sampling.

Value

Stores simulation results based on output specified in the analysis to `sim_results.txt`. The only output to the console is the time used for the function run.

Author(s)

Martha Ellis, Jake Ivan

References

ELLIS, MARTHA M., JACOB S. IVAN, and MICHAEL K. SCHWARTZ. "Spatially Explicit Power Analyses for Occupancy-Based Monitoring of Wolverine in the US Rocky Mountains." Conservation Biology (2013).

See Also

`wolverine_analysis`

`wolverine_analysis` *Default function for analyzing an individual encounter history*

Description

Original function designed for analyzing wolverine encounter histories in Ellis et al. 2013.

Usage

```
wolverine_analysis(n_yrs, ch = NULL, n_visit = NULL, gap_yr = 0, FPC = 1, ... )
```

Arguments

`n_yrs`	Number of years in the encounter history. Run with all else NULL to produce an initial data.frame for setting up further results.
`ch`	Character vector with encounters
`n_visit`	Number of visits per year
`gap_yr`	Code for possible alternate model formulations
`FPC`	Finite population correction
`...`	additional arguments

Details

This function provides an example of an analysis function, which will be called by `test_samples` on each individual encounter history. Should include appropriate model options for every subset of encounter histories. Returns a dataframe with results from current simulation, which will be compiled into a single output file by `test_samples`. Can include multiple model runs on the same encounter history by returning a data.frame with multiple lines.

The current function runs a robust design occupancy model through RMark with options for continuous or alternate year sampling. Trend is tested using variance.components procedure.

Value

data.frame with model results

Note

The analysis function you give `test_samples` via `function_name` will get called in two places in `test_samples`. First, it will get called by `RunAnalysis(n_yrs)` to set up a data.frame header. Then it will get called in each step of the loop over all the encounter history files and subsets using `RunAnalysis(n_yrs, ch, n_visit, gap_yr,fpc , ... )`. This is clunky, but for now, if you want to change `wolverine_analysis`, you need to make sure the function you specify will work with those two function calls.

Author(s)

Jake Ivan, Martha Ellis

References

ELLIS, MARTHA M., JACOB S. IVAN, and MICHAEL K. SCHWARTZ. "Spatially Explicit Power Analyses for Occupancy-Based Monitoring of Wolverine in the US Rocky Mountains." Conservation Biology (2013).

See Also

`test_samples`

Index