Package: readdst (via r-universe)

October 22, 2024

Title Convert Distance for Windows projects to R analyses

Version 0.0.6.9003

Description Take projects built using Distance for Windows and create R scripts which duplicate the analysis. Optionally build a test suite that checks analysis results from Distance with the equivalent R results.

Depends R (>= 3.2.0)

License GPL

LazyData true

Imports Hmisc, RODBC, stringr, magrittr, mrds, testthat, plyr

Suggests knitr, testthat, rmarkdown

VignetteBuilder knitr

RoxygenNote 7.1.1

Repository https://distancedevelopment.r-universe.dev

RemoteUrl https://github.com/distanceDevelopment/readdst

RemoteRef HEAD

RemoteSha 12e8a5bfe5c21d5ea05acc01a1d592159db0ff46

Contents

`readdst-package` *Convert Distance for Windows analyses to R code*

Description

This package read data and model definitions from a Distance for Windows project (`.dst` and `.dat` files) and converts models to run in the R package `mrds`.

Details

Usually, a workflow will look something like that below, centred around the functions `convert_project` and `run_analysis` . See also the vignette shipped with the package for example output.

Examples

```
## Not run:
library(readdst)
# load the golftees sample project and convert it
project <- system.file("Golftees-example", package="readdst")
project <- paste0(project,"/Golftees")
converted <- convert_project(project)

# run the first analysis in the project and look at model summary
```

```
analysis_1 <- run_analysis(converted[[1]], debug=TRUE)
summary(analysis_1)

## End(Not run)
```

`build_layer_hierarchy` *Work out the layer hierarchy in the Distance database*

Description

Use the DataLayers table to work out the hierarchy of the tables and layers in the database held by Distance.

Usage

```
build_layer_hierarchy(data_file)
```

Arguments

`data_file`	a data file to load the database from

Author(s)

David L Miller

`converted_distance_analyses` *Converted analysis objects*

Description

Once `convert_project` has been run on a project, two types of object are created: first an object of class `converted_distance_analyses`, which is just a list of `converted_distance_analysis` objects.

Details

`converted_distance_analysis` contain all the information necessary to run a Distance for Windows model in R. Each object has the following elements:

- `call` string with the call to `ddf` to build and run the model
- `aic.select` maximum number of terms to select by AIC if AIC term selection has been enabled (for key plus adjustment terms models only)
- `status` what the status of this model was in Distance for Windows (see "Status" below)

- `env` an `environment` that contains data needed to run the model (`data` containing entire dataset in flatfile form, `obs.table` containing the observation table, `sample.table` is the sample table, `reg.table` is the region table and `units` is a matrix describing conversion factor of distance measures (effort and detection distance) to areal measurements (for density))
- `filter` string used to subset the data to get the same filter as in Distance for Windows
- `group_size` describes how size bias adjustment is conducted, and the level of hierarchy at which E(s) is computed
- `detection_by` level of design hierarchy at which detection function is computed (e.g. pooled across strata)
- `gof_intervals` if binning is done for GOF testing, cutpoints are provided here
- `estimation` what sort of weighted average is used to compute region-level density estimate
- `name` the name for this analysis, as used in Distance for Windows
- `ID` the ID number for this analysis, as used in Distance for Windows

Status

The `status` code is taken from Distance for Windows to indicate whether the analysis has been run yet and what the outcome was. Status codes are as follows:

- `0` analyses has not been run in Distance for Windows yet
- `1` analysis ran without errors or warnings
- `2` analysis ran with warnings
- `3` analysis ran with errors

Note that an analysis that runs with error in Distance for Windows may run fine in R and an analysis that runs fine in Distance for Windows may not work in R. In the latter case, please consider submitting this a a bug to `github.com/distancedevelopment/distance-bugs`.

`converted_distance_data` *Converted distance data*

Description

If `convert_project` has been run on a project, but there are no analyses present in the project, then a list of the data will be returned. The list has one element for each data filter which was present in the project. Each element of the list has the following tables in it:

`convert_project` *Convert a Distance for Windows project to be run in R*

Description

Take each analysis in a Distance for Windows project and convert the model definition to an `mrds` model, data and data filters are also extracted and associated with the relevant models.

Usage

```
convert_project(project)
```

Arguments

`project`	a path to a project (path to the `dst` file with `".dst"` removed from the end of the path)

Value

an object of class `converted_distance_analyses` (if there are analyses defined), an object of class `converted_distance_data` (if no analyses are present in the project). Either way an attribute called `"flatfile"` is also returned with a flat version of the data.

Details

Only CDS/MCDS/MRDS analyses are supported.

Model names are as they are in Distance for Windows (so if you have nonsensical names in Distance for Windows they will be the same in R).

Author(s)

David L Miller

See Also

converted_distance_analyses readdst-package

create_bins *Create bins from a set of binned distances and a set of cutpoints.*

Description

This is an internal routine and shouldn't be necessary in normal analyses.

Usage

```
create_bins(data, cutpoints)
```

Arguments

data	data.frame with at least the column distance.
cutpoints	vector of cutpoints for the bins

Value

data data with two extra columns distbegin and distend.

Author(s)

David L. Miller

db_get *Get data from the Distance project database*

Description

This function is a wrapper around either calls to RODBC (on Windows) or mdb.get (on Unix-a-like systems). Given a database file name it will return either the contents of the table (as data.frame), if table=NULL then it will return all tables and if table=TRUE then it will return a character vector of table names.

Usage

```
db_get(file, table = NULL)
```

Arguments

file	the path to the database file to access
table	the table to extract (if NULL all tables are extracted, if TRUE a list of tables names are extracted)

Value

a `data.frame` with the contents of a database table

Note

Currently not implemented on Windows systems.

Author(s)

David L Miller

`filter_data` *Filter the data*

Description

Take the "data filters" applied by Distance for Windows to the data and use them to subset the data.

Usage

```
filter_data(data, data_filter)
```

Arguments

`data`	the data to be filtered
`data_filter`	a data filter to be parsed (output from `parse_definition.data_filter`)

Value

a list with two elements, the data and the filter string

Author(s)

David L Miller

get_data *Extract data from a Distance database*

Description

Extracts the relevant tables from the Distance for Windows database to build data that can be used with `mrds` or `Distance`.

Usage

```
get_data(data_file)
```

Arguments

`data_file`	the path to a `DistData.mdb` file.

Value

a "flatfile" compatible `data.frame` containing all of the information necessary to make a stratified abundance/density estimate.

Author(s)

David L Miller

get_definitions *Extract definition information from tables*

Description

Takes the "Definition" column from a table, converts it to `character` and puts each row in a list with names as the corresponding ID of that row.

Usage

```
get_definitions(file, table)
```

Arguments

`file`	the name of the `.mdb` file
`table`	which table to extract the "Description" column from

Value

a `list` of definitions, each element of which is a character vector.

Author(s)

David L Miller

`get_stats` *Extract saved statistics for analyses*

Description

At the moment only extracts the AIC and likelihood

Usage

```
get_stats(project_file, stats_table)
```

Arguments

`project_file`	path to project file
`stats_table`	a `data.frame` containing possible statistics

Details

Codes used to determine the meanings of statistics are given at https://github.com/DistanceDevelopment/readdst/wiki/distance-results-codes.

Author(s)

David L Miller

`get_unit_conversion` *Get the unit conversions for the data*

Description

Obtain a list of conversion to SI units from the units that the measurements in a Distance for Windows project are in.

Usage

```
get_unit_conversion(data_file)
```

Arguments

`data_file`	Distance for Windows project data file

Value

a data.frame with columns Variable, Units and Conversion, giving the variable name, the units it is measured in and the conversation factor to SI units.

Author(s)

David L Miller

---

group_size_est *Estimate group size*

Description

Distance for Windows includes a few different methods for accounting for group (or cluster) size (at the adundance/density estimation stage). These include using the mean group size for all observations or using a regression of size against distance or log size against distance.

Usage

```
group_size_est(data, group_size, model)
```

Arguments

data	the data for this analysis
group_size	the group_size element of an analysis object
model	a fitted model

Value

estimated cluster sizes (numeric vector of length nrow(data)), or NULL if there were no instructions on how to estimate group/cluster size

Author(s)

David L Miller

make_analysis *Make an analysis*

Description

This function calls make_model to create the call to ddf it also creates an environment with the data necessary to perform the call.

Usage

```
make_analysis(this_analysis, model_definitions, data_filters, data, transect)
```

Arguments

this_analysis	an analysis from Distance
model_definitions	a list of model definitions
data_filters	a list of data filters
data	the data to use with the model (see get_data and unflatfile)
transect	the transect type

Value

a list with the following elements: a character string specifying a call to ddf, an environment to run it in, the name of the analysis and it's ID.

Author(s)

David L Miller

make_control *Make the control element of a call to* ddf

Description

Build the control options for a ddf call.

Usage

```
make_control(md)
```

Arguments

md	model definition data to parse

Value

character string describing the `control list`

---

make_dsmodel *Build a dsmodel call*

Description

From a model definition build the `dsmodel` part of the model.

Usage

```
make_dsmodel(md)
```

Arguments

`md`	a model definition

Value

a character string starting with "`dsmodel=`" or `NULL` if no `dsmodel` component in this model

Author(s)

David L Miller

---

make_formula *Build a dsmodel or mrmodel formula*

Description

Build a formula, ensuring that the correct terms are factors

Usage

```
make_formula(md_formula, md_factors)
```

Arguments

`md_formula`	"Formula" data from a model definition
`md_factors`	"Factors" data from a model definition

Value

a character string specifying a formula, starting with "`formula=~`"

Author(s)

David L Miller

make_meta.data *Build the model meta.data*

Description

From a model definition build the dsmodel part of the model.

Usage

```
make_meta.data(df, transect, data)
```

Arguments

df	a data filter object
transect	type of transect
data	the data used in the model

Value

a character string starting with meta.data=

Author(s)

David L Miller

make_model *Build a distance sampling analysis*

Description

Reproduce the corresponding call to ddf to reproduce an analysis from Distance for Windows.

Usage

```
make_model(this_analysis, model_definitions, data_filters, transect, data)
```

Arguments

this_analysis	an analysis from Distance
model_definitions	a list of model definitions
data_filters	a list of data filters
transect	the transect type
data	the data

Value

a character string specifying a call to ddf

Author(s)

David L Miller

---

make_mrmodel *Build a mrmodel call*

Description

From a model definition build the mrmodel part of the model.

Usage

```
make_mrmodel(md)
```

Arguments

md	a model definition

Value

a character string starting with "mrmodel=" or NULL if there is no mrmodel component in this model.

Author(s)

David L Miller

---

merge_results *Merge results from stratified analyses*

Description

In Distance for Windows, one can choose to estimate the detection function by stratum. In this case more than one detection function is returned when run_analysis is used to run the analysis. In order to test the statistics stored in the Distance for Windows project, one must first combine the resulting models (and their corresponding abundance and density estimates). This function performs these operations.

Usage

```
merge_results(models, analysis)
```

Arguments

models	a list of model (ddf/ds objects)
analysis	an analysis specification (to inform us on the stratification to be used.

Value

a list including the "combined" model, summary and density/abundance estimates (dht output). Note that these are almost definitely not valid objects for their respective classes, they are only to be used to test statistics.

Author(s)

David L Miller

---

model_description *Make a user-readable model description string*

Description

This takes a fitted mrds model object and returns a string that describes the detection function fitted and the fitted model's AIC.

Usage

```
model_description(model)
```

Arguments

model	a fitted model

Value

a string describing the model

Author(s)

David L Miller

model_selection *Model selection for key plus adjustment models*

Description

Run model selection for a given analysis. The returned object is exactly as if the model has been run using ddf, so anything that can normally be done with a ddf object can be done with the return.

Usage

```
model_selection(analysis, debug = FALSE)
```

Arguments

analysis	a converted analysis
debug	display the call and name of the model before it is run, print AIC selection details

Details

Model selection is performed via AIC.

Value

fitted ddf object

Author(s)

David L Miller

parse_definition.data_filter *Parse a Definition of a data filter*

Description

Given a data filter "Definition", pre-processed by get_definitions, extract the useful information from it.

Usage

```
parse_definition.data_filter(df)
```

Arguments

df	a definition

Value

named list of defintions

Details

A definition consists either of a key=value pair or a name then key=value pairs separated by \ and terminated with ;.

Note that this function should be called for a single definition, usually using `lapply`.

Author(s)

David L Miller

`parse_definition.model`

Parse a Definition

Description

Given data from a "Definition", pre-processed by `get_definitions`, extract the useful information from it.

Usage

```
parse_definition.model(df)
```

Arguments

`df`	a definition (vector of character strings)

Value

a `list` of `lists`

Details

See the "MCDS Command Language" section of the Distance manual for more information.

Note that this function should be called for a single definition, usually using `lapply`.

Author(s)

David L Miller

`print.converted_distance_analyses`
Converted distance analyses table

Description

Prints a table of the analyses that have been converted and their status from Distance for Windows.

Usage

```
## S3 method for class 'converted_distance_analyses'
print(x, ...)
```

Arguments

`x`	converted distance analyses
`...`	unused additional args for S3 compatibility

`print.converted_distance_analysis`
Print a converted distance analysis

Description

Prints details of an analyses that has been converted.

Usage

```
## S3 method for class 'converted_distance_analysis'
print(x, ...)
```

Arguments

`x`	converted distance analyses
`...`	unused additional args for S3 compatibility

print.distance_stats_table *Print tested statistics*

Description

This is simply a `print` method to nicely ouput the results of `test_stats`.

Usage

```
## S3 method for class 'distance_stats_table'
print(x, ..., digits = NULL)
```

Arguments

`x`	the result of a call to `test_stats`

Value

just prints the results

Author(s)

David L Miller

run_analysis *Run a converted distance sampling analysis*

Description

Take a single converted analysis and run the model contained therein.

Usage

```
run_analysis(analysis, debug = FALSE)
```

Arguments

`analysis`	a converted analysis
`debug`	display the call and name of the model before it is run, print AIC selection details

Details

A previous call to `convert_project` will return a list of projects. Only one analysis at a time can be run with `run_analysis`. If you wish to run all the analyses in the project, see the code below using `lapply`.

If an analysis needs to select the number of adjustment terms (for key plus adjustment detection functions) by AIC, then that selection is done at this stage.

Value

fitted ddf object

Author(s)

David L Miller

Examples

```
## Not run:
library(readdst)

# load and convert the golftees project
project <- system.file("Golftees-example", package="readdst")
project <- paste0(project, "/Golftees")
converted <- convert_project(project)

# run the first analysis
analysis_1 <- run_analysis(converted[[1]], debug=TRUE)

# look at the resulting model output
summary(analysis_1)

# run all the analyses in a project
all_analyses_run <- lapply(converted, run_analysis)

## End(Not run)
```

---

set_covar_names *Set column names in data to be as in formulae*

Description

Set column names in data to be as in formulae

Usage

```
set_covar_names(data, covnames)
```

Arguments

data	a data.frame of the data to be modelled
covnames	the covariates that are factors

stats_table *Generate table of possible statistics to test*

Description

To use get_stats we need a set of statistics to test. We also require their codes (to look up in the Distance for Windows database) and their equivalent values in mrds (or how to calculate those values). This function provides such a table.

Usage

```
stats_table(engine = "CDS")
```

Arguments

engine	which engine do we need to compute stats for?

Value

a data.frame with statistics Distance for Windows collects that have equivalents in mrds. The data.frame has three columns: Code, the numeric code for the statistic (as used in the Distance for Windows database); Name, the short name for this statistic; MRDS, the operation required to obtain the equivalent statistic in mrds; Description, a short description of the statistic.

Details

Data for this table (numeric code and descriptions) is from the DistIni.mdb which is shipped with Distance for Windows. See also https://github.com/distancedevelopment/readdst/wiki/distance-results-codes.

Additional notes

Note that the Cramer-von Mises p-value as recorded in Distance for Windows is only recorded to the nearest 0.1.

Author(s)

David L Miller

test_stats *Test to see if Distance for Windows and R get the same results*

Description

Tests the results stored in the Distance for Windows project file against those generated from running the same analysis in R.

Usage

```
test_stats(analysis, statuses = 1, tolerance = 0.01)
```

Arguments

analysis	a converted (but not run) analysis
statuses	for which statuses should tests be run? See "Status", below (Defaults to 1: analysis that ran without error or warning in Distance for Windows).
tolerance	the tolerance of the test (default 0.01)

Details

A previous call to convert_project will return a list of projects. Only one analysis at a time can be run with test_stats. If you wish to run all the analyses in the project, you can use lapply.

Value

a data.frame with five columns: Statistic, a description of the tested statistic; Distance_value the value of the statistic stored by Distance for Windows; mrds_value the value of the statistic calculated by mrds; Difference the proportional difference between the previous two columns (computed using all.equal); Pass a series of ticks, indicating that the value in the Difference column is less than tolerance.

Status

The status code is taken from Distance for Windows to indicate whether the analysis has been run yet and what the outcome was. Status codes are as follows:

- 0 analyses has not been run in Distance for Windows yet
- 1 analysis ran without errors or warnings
- 2 analysis ran with warnings
- 3 analysis ran with errors

If an analysis has a status of 0 or 3 there will usually not be any statistics attached to the analysis, so no tests will be run.

Note that an analysis that runs with error in Distance for Windows may run fine in R and an analysis that runs fine in Distance for Windows may not work in R. In the latter case, please consider submitting this a a bug to github.com/distancedevelopment/distance-bugs.

Note

Tests all available statistics.

Examples

```
## Not run:
library(readdst)
# load the golftees sample project and convert it
project <- system.file("Golftees-example", package="readdst")
project <- paste0(project,"/Golftees")
converted <- convert_project(project)

# run tests for analysis 1
test_stats(converted[[1]])

## End(Not run)
```

`unflatfile` *Take a flatfile data.frame and make dht-compatible data.frames*

Description

Given distance sampling survey data in flatfile format, convert it to the four tables required by `dht`.

Usage

```
unflatfile(data)
```

Value

list of the four `data.frames` described in "Details".

Details

- region.table `data.frame` with two columns: `Region.Label`, label for the region; `Area`, area of the region. `region.table` has one row for each stratum. If there is no stratification then `region.table` has one entry with `Area` corresponding to the total survey area.
- sample.table `data.frame` mapping the regions to the samples (i.e. transects). There are three columns: `Sample.Label`, label for the sample; `Region.Label`, label for the region that the sample belongs to.; `Effort`, the effort expended in that sample (e.g. transect length).
- obs.table `data.frame` mapping the individual observations (objects) to regions and samples. There should be three columns: `object`, the observation ID; `Region.Label`, label for the region that the sample belongs to; `Sample.Label`, label for the sample.
- data a `data.frame` containing at least a column called `distance`. NOTE! If there is a column called `size` in the data then it will be interpreted as group/cluster size.

Note

Based on `checkdata` from package `Distance`.

Author(s)

David L. Miller

`units_table` *Generate table of unit conversions*

Description

Returns a table of conversions between the units used in Distance for Windows. This is extracted from the `DistIni.mdb` default database.

Usage

```
units_table()
```

Author(s)

David L Miller

Index