

Package: camtrapDensity (via r-universe)

October 25, 2026

Type Package

Title Density Estimation Functions for camtrapDP Data

Version 0.1.18

Author Marcus Rowcliffe

Maintainer Marcus Rowcliffe <marcus.rowcliffe@ioz.ac.uk>

Description Currently provides functions for working with camtrapDP format data exported from Agouti - checking deployment calibration diagnostic plots, filtering deployments, correcting mis-specified date-time data at specific deployments, running REM density analysis.

License GPL-3

Encoding UTF-8

Imports activity, camtrapr, Distance, dplyr, ggplot2, gtools, jpeg, jsonlite, leaflet, lubridate, sbd, plotly, rlang, tibble, tidyr

Suggests testthat (>= 3.1.7)

LazyData true

Config/roxygen2/version 8.0.0

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev make libicu-dev libjpeg-dev libjq-dev libpng-dev libuv1-dev libxml2-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev libnode-dev libx11-dev

Repository <https://camera-traps.r-universe.dev>

Date/Publication 2026-09-13 20:42:24 UTC

RemoteUrl <https://github.com/MarcusRowcliffe/camtrapDensity>

RemoteRef HEAD

RemoteSha c853dc4a5ab311d0d04e84d38864de8e2885792f

Contents

.with_seed	2
check_deployment_models	3

convert_units	3
correct_time	4
downversion_camtrapDP	5
fit_actmodel	6
fit_detmodel	7
fit_speedmodel	8
get_agouti_url	10
get_multiplier	10
get_parameter_table	11
get_trap_rate	13
get_traprate_data	14
lnorm_confint	15
map_deployments	15
map_traprates	16
merge_camtrapDP	16
pkg	17
plot_deployment_schedule	18
read_camtrap_dp_csv	18
read_camtrapDP	19
recalc_events	20
rem	21
rem_estimate	22
select_species	23
slice_camtrap_dp	24
subset_deployments	25
write_rem_csv	26
Index	27

.with_seed	<i>Evaluate expression with a temporary RNG seed</i>
------------	--

Description

Sets the random seed for the duration of evaluating ‘expr’, then restores the previous RNG state. If ‘seed’ is ‘NULL’ the expression is evaluated without changing the RNG state.

Usage

.with_seed(seed, expr)

Arguments

seed	Integer seed to set, or NULL to skip setting seed
expr	An expression to evaluate

Value

The result of evaluating ‘expr’

 check_deployment_models

Check deployment calibration model diagnostic plots

Description

Displays deployment calibration model diagnostic plots and allows users to record interactively whether each deployment is reliable.

Usage

```
check_deployment_models(package)
```

Arguments

package Camera trap data package object, as returned by [read_camtrap_dp](#).

Value

The original package with logical column useDeployment‘ added to deployments and observations data.

Examples

```
## Not run:
pkg <- read_camtrapDP("./datapackage/datapackage.json")
pkg_checked <- check_deployment_models(pkg)

## End(Not run)
```

 convert_units

Change the units of an REM parameter table

Description

Changes the units of parameters from their current setting to new user-defined units.

Usage

```
convert_units(
  param,
  radius_unit = c("km", "m", "cm"),
  angle_unit = c("radian", "degree"),
  active_speed_unit = c("km/day", "km/hour", "m/hour", "m/second"),
  overall_speed_unit = c("km/day", "km/hour", "m/hour", "m/second"),
  trap_rate_unit = c("n/day", "n/100day", "n/hour", "n/minute", "n/second"),
  density_unit = c("n/km2", "n/ha", "n/100km2")
)
```

Arguments

param	An REM parameter dataframe (see details).
radius_unit	A character string giving the output unit of radius.
angle_unit	A character string giving the output unit of angle.
active_speed_unit	A character string giving the output unit of speed while active.
overall_speed_unit	A character string giving the output unit of day range.
trap_rate_unit	A character string giving the output unit of trap rate.
density_unit	A character string giving the output unit of density.

Details

Input dataframe param must contain field unit, and at least one field among estimate, se, lc195, and uc195. Row names must be among radius, angle, activity_level, active_speed, overall_speed. Input is typically created with function [get_parameter_table](#).

Value

A replica of input dataframe param with estimate, se and confidence limit values converted to output units.

Examples

```
## Not run:
pkg <- read_camtrapDP("./datapackage/datapackage.json")

## End(Not run)
data(pkg)
sp <- "Vulpes vulpes"
trdata <- get_traprate_data(pkg, species=sp)
radmod <- fit_detmodel(radius~1, pkg, species=sp, order=0)
angmod <- fit_detmodel(angle~1, pkg, species=sp, unit="radian", order=0)
spdmod <- fit_speedmodel(pkg, species=sp)
actmod <- fit_actmodel(pkg, species=sp, reps=100)
param <- get_parameter_table(trdata, radmod, angmod, spdmod, actmod)
convert_units(param, radius_unit="m", angle_unit="degree", active_speed_unit="m/second")
```

correct_time

Correct times for a given deployment in a camera trap datapackage

Description

When a camera trap starts with the wrong time stamp, times in all datapackage tables can be corrected given a reference time recorded by the camera, the correct time for this reference time and the ID of the deployment to correct.

Usage

```
correct_time(package, depID = NULL, locName = NULL, wrongTime, rightTime)
```

Arguments

package	Camera trap data package object, as returned by read_camtrap_dp .
depID	A character value giving the deployment ID, to be matched in package\$data\$deployments\$deploymentID.
locName	A character value giving the location name, to be matched in package\$data\$deployments\$locationName.
wrongTime	A character or POSIX reference date-time recorded wrongly by the camera.
rightTime	A character or POSIX value giving the correct date-time when the reference time was recorded.

Details

One, but not both, of depID and locName must be provided, as single text values. If locName is provided, the deployment associated with this in pkg\$data\$deployments is corrected, but if locName is associated with more than one deployment the function does not run.

Value

As for [read_camtrap_dp](#), with all date-times corrected by the difference between rightTime and wrongTime.

Examples

```
## Not run:
pkg <- read_camtrapDP("./datapackage/datapackage.json")

## End(Not run)
data(pkg)
pkg_corrected <- correct_time(pkg,
                             locName = "S01",
                             wrongTime = "2017-10-02 08:06:43",
                             rightTime = "2017-09-01 10:36:00")
```

downversion_camtrapDP *Downversion Camtrap DP datapackage*

Description

Downversions a Cmatrap DP datapackage to V0 by duplicating dataframe fields whose names differ between versions with V0 fields created in addition to existing V1 fields.

Usage

```
downversion_camtrapDP(pkg)
```

Arguments

pkg Camera trap data package list, as returned by [read_camtrapDP](#).

Value

A duplicate of the original package with the addition of duplicate of fields with V0 fieldnames to data.

Examples

```
## Not run: pkgV0 <- downversion_camtrapDP(pkgV1)
```

fit_actmodel	<i>Fit an activity model</i>
--------------	------------------------------

Description

Fits an activity model to data package data and estimates activity level (proportion of time spent active).

Usage

```
fit_actmodel(
  package,
  species = NULL,
  reps = 999,
  obsdef = c("individual", "sequence"),
  seed = 42,
  ...
)
```

Arguments

package	Camera trap data package object, as returned by read_camtrap_dp .
species	A character string indicating species subset to analyse; if NULL runs select_species to get user input.
reps	Number of bootstrap replicates to run.
obsdef	Observation definition, either individual or sequence.
seed	Integer seed for reproducible fitting; 'NULL' to skip setting a seed. Default 42.
...	Arguments passed to fitact .

Value

An 'actmod' list.

See Also[fitact](#)**Examples**

```
## Not run:
pkg <- read_camtrapDP("./datapackage/datapackage.json")

## End(Not run)
data(pkg)
## Not run:
# With interactive species definition
act_model <- fit_actmodel(pkg)

## End(Not run)
# With species predefined, reps reduced for speed
act_model <- fit_actmodel(pkg, species="Vulpes vulpes", reps=100)
act_model@act
```

fit_detmodel

*Fit a detection function model***Description**

Fits a detection function to a data package and estimates effective detection distance (EDD).

Usage

```
fit_detmodel(
  formula,
  package,
  species = NULL,
  newdata = NULL,
  unit = c("m", "km", "cm", "degree", "radian"),
  ...
)
```

Arguments

formula	A two sided formula relating radius or angle data to covariates.
package	Camera trap data package object, as returned by read_camtrap_dp .
species	A character string indicating species subset to analyse; if NULL runs select_species to get user input; if "all" all data are used.
newdata	A dataframe of covariate values at which to predict detection distance.
unit	The units in which to return the result.
...	Arguments passed to ds .

Details

The type of detection function (line or point) is determined by the `unit` argument.

Value

A ddf detection function model list, with additional elements: `edd`, a vector with estimated and standard error effective detection distance, or the `newdata` dataframe with EDD estimate and `se` columns added; `proportion_used`, the proportion of the observations used to fit the detection function in the case of truncation.

See Also

[ds](#)

Examples

```
## Not run:
pkg <- read_camtrapDP("../datapackage/datapackage.json")

## End(Not run)
data(pkg)
## Not run:
# With interactive species definition
radius_model <- fit_detmodel(radius~1, pkg, order=0)

## End(Not run)
# With species predefined
sp <- "Vulpes vulpes"
radius_model <- fit_detmodel(radius~1, pkg, species=sp, order=0)
angle_model <- fit_detmodel(angle~1, pkg, species=sp, order=0, unit="degree")
radius_model$edd
angle_model$edd
plot(radius_model, pdf=TRUE)
plot(angle_model)
```

fit_speedmodel	<i>Estimate average animal speed</i>
----------------	--------------------------------------

Description

Calculates harmonic mean and standard error of animal speed while active from a data package.

Usage

```
fit_speedmodel(
  package,
  species = NULL,
  formula = speed ~ 1,
```

```

newdata = NULL,
reps = 1000,
distUnit = c("m", "km", "cm"),
timeUnit = c("second", "minute", "hour", "day"),
seed = 42,
...
)

```

Arguments

package	Camera trap data package object, as returned by read_camtrap_dp .
species	A character string indicating species subset to analyse; if NULL runs select_species to get user input; if all uses all data.
formula	A formula with speed on the left and covariates on the right.
newdata	A data frame of covariate values at which to predict speeds.
reps	Number of random draws to use for standard calculation.
distUnit	A character string indicating distance unit of speed observations.
timeUnit	A character string indicating time unit of speed observations.
seed	Integer seed for reproducible speed uncertainty estimates; 'NULL' to skip setting a seed. Default 42.
...	Other parameters passed to sbm for covariate modelling (see details).

Details

If a formula is provided, the model is fitted using the sbm (size biased model) function, which can be installed using: `devtools::source_url("https://raw.githubusercontent.com/MarcusRowcliffe/sbd/master/sbd")`.

Value

List with elements:

- speed: a dataframe containing columns estimate (mean) and se (standard error) speed while active.
- data: a numeric vector of the data from which the estimate is derived.

Examples

```

## Not run:
pkg <- read_camtrapDP("./datapackage/datapackage.json")

## End(Not run)
data(pkg)
## Not run:
# With interactive species definition
speed_model <- fit_speedmodel(pkg)

## End(Not run)
# With species predefined

```

```
speed_model <- fit_speedmodel(pkg, species="Vulpes vulpes")
speed_model$estimate
hist(speed_model)
```

get_agouti_url	<i>Gets a set Agouti sequences URLs.</i>
----------------	--

Description

Obtains web addresses for sequences of selected observations, based on criteria defined using fields in the observations table.

Usage

```
get_agouti_url(package, obsChoice)
```

Arguments

package	Camera trap data package object, as returned by read_camtrap_dp .
obsChoice	A logical expression using column names from the observations table defining which observations you want to inspect.

Value

A dataframe of Agouti URLs.

Examples

```
data(pkg)
get_agouti_url(pkg, speed>1)
```

get_multiplier	<i>Get a unit multiplier</i>
----------------	------------------------------

Description

Returns a multiplier to convert values from one unit to another, in one of four categories: distance, time, angle, count.

Usage

```
get_multiplier(unitIN, unitOUT)
```

Arguments

unitIN	A character vector giving the units of input.
unitOUT	A character vector giving the units of output, the same length as unitIN,

Details

Possible unitIN and unitOUT values are "cm", "m", "km" for distances; "second", "minute", "hour", "day", "100day" for times; "radian", "degree" for angles; "n" for count; "none" for no units. Unit ratios are allowed for rates or densities. In this case, units should be separated with a forward slash (e.g. "n/day", "km/hour", "n/km2"). Input and output types must match.

Value

A vector of numbers giving the amount by which to multiply input values to arrive at unit-converted values.

Examples

```
get_multiplier(c("m", "m/second"), c("km", "km/day"))
```

get_parameter_table	<i>Create a parameter table from a set of models</i>
---------------------	--

Description

Creates a table of REM parameters taken from models for detection radius, detection angle, speed and activity level.

Usage

```
get_parameter_table(
  traprate_data,
  radius_model,
  angle_model,
  speed_model,
  activity_model,
  strata = NULL,
  reps = 999,
  seed = 42
)
```

Arguments

traprate_data	A dataframe of trap rate data (counts and effort), as returned by get_traprate_data .
radius_model	A detection radius model fitted using fit_detmodel
angle_model	A detection angle model fitted using fit_detmodel

<code>speed_model</code>	A speed model fitted using <code>fit_speedmodel</code>
<code>activity_model</code>	An activity model fitted using <code>fit_actmodel</code>
<code>strata</code>	A dataframe of stratum information passed to <code>get_trap_rate</code>
<code>reps</code>	Number of bootstrap replicates for estimating trap rate error (see <code>get_trap_rate</code>)
<code>seed</code>	Integer seed passed to <code>get_trap_rate</code> for reproducible bootstrapping; 'NULL' to skip setting a seed. Default 42.

Value

A dataframe of unit-harmonised parameter estimates with rows:

- `radius`: detection radius
- `angle`: detection angle
- `active_speed`: speed while active
- `activity_level`: proportion of time spent active
- `overall_speed`: long-term average speed (day range) - the product of `active_speed` and `activity_level`
- `trap_rate`: number of camera trap records per unit time

and columns

- `estimate`: parameter estimates
- `se`: standard error
- `cv`: proportional coefficient of variation
- `lc195`, `uc195`: lower and upper 95% confidence limits
- `n`: sample size
- `unit`: the unit of the estimate

Examples

```
## Not run:
pkg <- read_camtrapDP("./datapackage/datapackage.json")

## End(Not run)
data(pkg)
sp <- "Vulpes vulpes"
trdata <- get_traprate_data(pkg, species=sp)
radmod <- fit_detmodel(radius~1, pkg, species=sp, order=0)
angmod <- fit_detmodel(angle~1, pkg, species=sp, unit="radian", order=0)
spdmod <- fit_speedmodel(pkg, species=sp)
actmod <- fit_actmodel(pkg, species=sp, reps=100)
get_parameter_table(trdata, radmod, angmod, spdmod, actmod)
```

get_trap_rate	<i>Get average trap rate from REM data</i>
---------------	--

Description

Calculates average trap rate and its bootstrapped error from a table of per-location observation counts and camera time.

Usage

```
get_trap_rate(traprate_data, strata = NULL, reps = 999, seed = 42)
```

Arguments

traprate_data	A dataframe containing (at least) columns <code>n</code> and <code>effort</code> , as returned by get_traprate_data ; if <code>strata</code> supplied for stratified calculation, must also have column <code>stratumID</code> .
strata	A dataframe with one row per stratum, and columns <code>stratumID</code> and <code>area</code> .
reps	The number of bootstrap replicates to run.
seed	Integer seed for reproducible bootstrap; 'NULL' to skip setting a seed. Default 42.

Value

A dataframe with columns: - `estimate`: average trap rate - `se`: standard error - `cv`: proportional coefficient of variation - `lcl95`, `ucl95`: lower and upper 95 - `n`: sample size (number of locations) - `unit`: the unit of the estimate

Examples

```
## Not run:
pkg <- read_camtrapDP("../datapackage/datapackage.json")

## End(Not run)
data(pkg)
trdata <- get_traprate_data(pkg, species="Vulpes vulpes")
get_trap_rate(trdata)
```

get_traprate_data	<i>Get REM data from a camtrap-dp datapackage</i>
-------------------	---

Description

Extracts a data table of observation counts and effort for each camera location in a camtrap-dp data package.

Usage

```
get_traprate_data(
  package,
  species = NULL,
  unit = c("day", "hour", "minute", "second")
)
```

Arguments

package	Camera trap data package object, as returned by read_camtrap_dp .
species	A character string indicating species subset to extract data for; if NULL runs select_species to get user input.
unit	The time unit in which to return camera effort.

Value

A tibble with columns:

- locationName: name of the camera location
- effort: the camera time for the location
- unit: the effort time unit
- scientificName: the scientific name of the species data extracted
- n: the observation counts
- stratumID: stratum identifier (only if this column is present in package\$data\$deployments)

Examples

```
## Not run:
pkg <- read_camtrapDP("../datapackage/datapackage.json")

## End(Not run)
data(pkg)
## Not run:
# With interactive species definition
trdata <- get_traprate_data(pkg)

## End(Not run)
# With species predefined
trdata <- get_traprate_data(pkg, species="Vulpes vulpes")
```

lnorm_confint	<i>Log-normal confidence interval</i>
---------------	---------------------------------------

Description

Calculates approximate log-normal confidence intervals given estimates and their standard errors.

Usage

```
lnorm_confint(estimate, se, percent = 95)
```

Arguments

estimate	Numeric estimate value(s)
se	Standard error(s) of the estimate
percent	Percentage confidence level

Value

A dataframe with a row per estimate input, and columns lcl and ucl (lower and upper confidence limits).

Examples

```
lnorm_confint(10.13, 3.57)
```

map_deployments	<i>Plot a map of deployments</i>
-----------------	----------------------------------

Description

Creates an OpenStreetMap street or satellite map over-plotted with deployment locations.

Usage

```
map_deployments(pkg, basemap = c("street", "satellite"), ...)
```

Arguments

pkg	Camera trap data package object, as returned by read_camtrap_dp .
basemap	Basemap to plot, street (default) or satellite
...	Additional arguments passed to addCircleMarkers

Examples

```
## Not run: pkg <- read_camtrapDP("../data/datapackage.json")
data(pkg)
map_deployments(pkg)
```

map_traprates	<i>Plot a map of deployment trap rates</i>
---------------	--

Description

Creates an OpenStreetMap street or satellite map over-plotted with deployment locations, with points sized in proportion to trap rate for a given species.

Usage

```
map_traprates(
  pkg,
  species = NULL,
  basemap = c("street", "satellite"),
  maxSize = 25,
  minSize = 3
)
```

Arguments

pkg	Camera trap data package object, as returned by read_camtrap_dp .
species	A character string indicating species subset to analyse. Use scientific names. If NULL runs select_species to get user input; if all uses all data.
basemap	Basemap to plot, street (default) or satellite
maxSize	Maximum point size to plot.
minSize	Minimum point size to plot.

Examples

```
## Not run: pkg <- read_camtrapDP("./data/datapackage.json")
data(pkg)
map_traprates(pkg, species="Vulpes vulpes")
```

merge_camtrapDP	<i>Merge Camtrap DP datapackages</i>
-----------------	--------------------------------------

Description

Merges a list of several Camtrap DP datapackages into a single datapackage.

Usage

```
merge_camtrapDP(pkgs)
```

Arguments

pkgs A list of camera trap data packages, as returned by [read_camtrapDP](#).

Details

Original datapackage metadata are stored in named top-level components of the output list, along with a single data component with the usual dataframes merged across datapackages, and additional packageName fields indicating the datapackage from which data row originates.

Value

A single datapackage with component dataframes merged.

Examples

```
## Not run: bigpkg <- merge_camtrapDP(list(pkg1, pkg2))
```

pkg	<i>Data and metadata from an example study exported from the Agouti camera trap data management platform in camtrap-DP format. Metadata includes study name, authors, location and other details. Data is held in element data, itself a list holding dataframes deployments, media and observations. See https://tdwg.github.io/camtrap-dp for details.</i>
-----	--

Description

Data and metadata from an example study exported from the Agouti camera trap data management platform in camtrap-DP format. Metadata includes study name, authors, location and other details. Data is held in element data, itself a list holding dataframes deployments, media and observations. See <https://tdwg.github.io/camtrap-dp> for details.

Format

A list holding study data and metadata.

plot_deployment_schedule

Plot a deployment Gantt chart

Description

Plots an interactive Gantt chart illustrating deployment times (black lines) and the occurrence of observations within those deployments (orange bars). Useful for checking errors in specification of deployment start and end dates, and visualising spatiotemporal distribution of observations.

Usage

```
plot_deployment_schedule(package)
```

Arguments

package Camera trap data package object, as returned by [read_camtrap_dp](#).

Examples

```
## Not run: pkg <- read_camtrapDP("../data/datapackage.json")
data(pkg)
plot_deployment_schedule(pkg)
```

read_camtrap_dp_csv

Create a datapackage from csv files

Description

Reads csv data from a folder into a list that functions like a camtrap-dp object for the purposes of density estimation.

Usage

```
read_camtrap_dp_csv(
  folder,
  tryFormats = c("%Y-%m-%d %H:%M:%OS", "%Y/%m/%d %H:%M:%OS",
    "%Y:%m:%d %H:%M:%OS")
)
```

Arguments

folder A character string giving the folder in which the csv files sit.

tryFormats A character string defining date-time format, passed to as.POSIXct.

Details

The folder must contain three csv files: deployments.csv, media.csv and observations.csv, each of which must contain at least the following fields:

- deployments: deploymentID, locationID, locationName, longitude, latitude, start, end
- media: mediaID, deploymentID, sequenceID, timestamp, filePath, fileName
- observations: observationID, deploymentID, sequenceID, mediaID, timestamp, scientificName, count, speed, radius, angle

Value

As for [read_camtrap_dp](#) but with reduced package metadata.

Examples

```
## Not run: pkg <- read_camtrap_dp_csv("../data")
```

read_camtrapDP	<i>Read a Camtrap DP datapackage</i>
----------------	--------------------------------------

Description

Reads the metadata and csv files from a Camtrap DP version 1 datapackage as exported from Agouti.

Usage

```
read_camtrapDP(
  file,
  media = TRUE,
  recalc = TRUE,
  addV0 = TRUE,
  dtFormat = "YmdHMSz"
)
```

Arguments

file	Path to a datapackage.json file.
media	Logical defining whether to read media table.
recalc	Logical defining whether to recalculate event; if TRUE, calls recalc_events .
addV0	Logical defining whether to add camtrapDP version 0 fields, for compatibility with functions using this data model.
dtFormat	Character format used to read date-time fields, passed to parse_date_time

Value

A list of data package metadata components and a data component with dataframes: deployments, media, observations (containing event observations) and positions (containing media observations). If the media argument is FALSE, the media component is NULL. The positions may also be NULL if there are no media observations in the input.

Examples

```
## Not run: pkg <- read_camtrapDP("./data/datapackage.json")
```

recalc_events	<i>Recalculate event observations</i>
---------------	---------------------------------------

Description

Recalculates event observation values (individualPositionRadius, individualPositionAngle, individualSpeed) from media observations.

Usage

```
recalc_events(pkg)
```

Arguments

pkg Camera trap data package list, as returned by [read_camtrapDP](#).

Details

The process requires media data and fails if this is not available in the datapackage. The datapackage should also contain positions data, but the original package is simply returned unmodified if the positions data table contains no data. In recalculating speeds, observations with zero time difference (hence infinite speed) impute time elapsed from average frame rate.

Value

A duplicate of the original package with event observation values recalculated.

Examples

```
## Not run: pkg2 <- recalc_events(pkg2)
```

rem

Fit a random encounter model

Description

Estimates REM density given a dataframe of parameters and their errors.

Usage

```
rem(parameters)
```

Arguments

parameters A dataframe containing REM parameter estimates with (at least) rows:

- radius: effective detection radius
- angle: effective detection angle
- overall_speed: average animal speed (day range)
- trap_rate: animal observations per unit time

and columns:

- estimate: numeric parameter estimate
- se: numeric parameter standard error
- unit: character parameter units (see [convert_units](#) for allowable values)

Value

A dataframe with the input parameters plus estimated density and its errors.

Examples

```
## Not run:
pkg <- read_camtrapDP("./datapackage/datapackage.json")

## End(Not run)
data(pkg)
sp <- "Vulpes vulpes"
trdata <- get_traprate_data(pkg, species=sp)
radmod <- fit_detmodel(radius~1, pkg, species=sp, order=0)
angmod <- fit_detmodel(angle~1, pkg, species=sp, unit="radian", order=0)
spdmod <- fit_speedmodel(pkg, species=sp)
actmod <- fit_actmodel(pkg, species=sp, reps=100)
param <- get_parameter_table(trdata, radmod, angmod, spdmod, actmod)
rem(param)
```

rem_estimate

*Integrated random encounter model density estimate***Description**

Estimates animal density for a given species given a camtrapDP datapackage. Models for detection radius and angle, speed and/or activity level can be fitted externally and provided as arguments, or are fitted internally if not provided (NULL default). Input units are assumed to be distance in m and time in seconds.

Usage

```
rem_estimate(
  package,
  check_deployments = TRUE,
  species = NULL,
  radius_model = NULL,
  angle_model = NULL,
  speed_model = NULL,
  activity_model = NULL,
  strata = NULL,
  reps = 999,
  seed = 42
)
```

Arguments

package	Camera trap data package object, as returned by read_camtrap_dp .
check_deployments	Logical indicating whether to check deployment calibration model diagnostic plots. If TRUE (default) runs <code>check_deployment_models</code> ; radius, angle and speed data from any excluded deployments are then dropped from analysis. If FALSE all data are used.
species	A character string indicating species subset to analyse; if NULL runs select_species to get user input.
radius_model	A detection function model for radii fitted using fit_detmodel .
angle_model	A detection function model for angles fitted using fit_detmodel with unit argument "radian" or "degree".
speed_model	A speed model fitted using fit_speedmodel .
activity_model	An activity model fitted using fitact or fit_actmodel .
strata	A dataframe of stratum areas, passed to get_trap_rate .
reps	Number of bootstrap replicates for error estimation.
seed	Integer seed for reproducible speed and activity model fitting and bootstrapping; 'NULL' to skip setting a seed. Default 42.

Value

A dataframe containing estimates and their errors for density and all contributing parameters.

Examples

```
# Load data
## Not run:
pkg <- read_camtrapDP("./datapackage/datapackage.json")

## End(Not run)
data(pkg)
# Sense check deployment schedules
plot_deployment_schedule(pkg)
## Not run:
# Sense check deployment calibration model diagnostic plots
pkg_checked <- check_deployment_models(pkg)
# Fully automated analysis (interactive species definition by default;
# reps reduced to limit run time).
res <- rem_estimate(pkg_checked, check_deployments=FALSE, reps=100)

## End(Not run)
# Automated analysis with species predefined and no deployment checking
sp <- "Vulpes vulpes"
res <- rem_estimate(pkg, species=sp, check_deployments=FALSE, reps=100)
# Inspect results
res$estimates

# Analysis with radius model fitted separately
radmod <- fit_detmodel(radius~1, pkg, species=sp, truncation=15, order=0)
res <- rem_estimate(pkg, check_deployments = FALSE, species = sp,
                    radius_model = radmod, reps=100)
res$estimates
```

select_species	<i>Select a species name</i>
----------------	------------------------------

Description

Presents a table of species names with observation count for each and allows the user to interactively select one.

Usage

```
select_species(package, species = NULL)
```

Arguments

package	Camera trap data package object, as returned by <code>read_camtrap_dp</code> .
species	NULL (default) to select interactively, or character vector giving either one or more valid species names found within <code>package\$data\$observations\$scientificName</code> , or the string "all" to return all available species names.

Value

A character vector of one or more scientific species names.

Examples

```
## Not run:
pkg <- read_camtrapDP("./data/datapackage.json")
select_species(pkg)
# If provided, a valid species name is simply passed through
select_species(pkg, "Vulpes vulpes")
# Providing a species name that isn't found in the data throws an error
select_species(pkg, "Vulpes vuppes")

## End(Not run)
```

slice_camtrap_dp	<i>Take a time slice of a data package</i>
------------------	--

Description

Discards any observations, media, or deployments that fall wholly outside the time range defined by start and end. When start or end are not specified, no slicing is applied to start or end points respectively. Slicing applies to all deployments by default, or can be applied to only a subset of deployments specified by depChoice. The fully default behaviour for this function is therefore to do nothing (the datapackage is returned unchanged).

Usage

```
slice_camtrap_dp(
  package,
  start = NULL,
  end = NULL,
  depChoice = NULL,
  suffix = ""
)
```

Arguments

package	Camera trap data package object, as returned by read_camtrap_dp .
start, end	Single character or POSIXct values defining the time range within which to slice the package.
depChoice	A logical expression using column names from the deployments table defining which deployments to slice.
suffix	A character value to be added to the package name.

Value

As for [read_camtrap_dp](#), with all data tables reduced according to the choice criteria.

Examples

```
## Not run:
pkg <- read_camtrapDP("./datapackage/datapackage.json")

## End(Not run)
data(pkg)
# Slicing the whole package to mid October 2017
subpkg <- slice_camtrap_dp(pkg,
                           start = "2017/10/10",
                           end = "2017/10/20")
# Slicing only deployments at location "S03" to a specific start time/date
subpkg <- slice_camtrap_dp(pkg,
                           start = "2017/10/15 14:30:00",
                           depChoice = locationName=="S03")
```

subset_deployments	<i>Subset a camera trap datapackage deployments</i>
--------------------	---

Description

Select a subset of deployments from a datapackage defined by a choice based on columns in the deployments table.

Usage

```
subset_deployments(package, choice, suffix = "")
```

Arguments

package	Camera trap data package object, as returned by read_camtrap_dp .
choice	A logical expression using column names from the deployments table.
suffix	A character value to be added to the package name.

Value

As for [read_camtrap_dp](#), with all data tables reduced according to the choice criteria at the deployment level.

Examples

```
# subset excluding a location and including only October 2017
## Not run:
pkg <- camtrapr::read_camtrapDP("./datapackage/datapackage.json")

## End(Not run)
data(pkg)
subpkg <- subset_deployments(pkg,
                             locationName != "S01" &
                             start >= as.POSIXct("2017-10-01", tz="UTC") &
                             end <= as.POSIXct("2017-10-31", tz="UTC"))
```

write_rem_csv

Write REM results to csv file

Description

Writes one or more REM estimate tables to a single csv file, with identifying columns added for project, datapackage, sampling design, sampling effort, project location, project dates and species. Input must be REM analysis object(s) created using [rem_estimate](#). The resulting file name is taken from the project and current date, and the file is saved to the working directory.

Usage

```
write_rem_csv(...)
```

Arguments

```
...           One or more REM analysis objects, separated by commas.
```

Value

None - creates a csv file.

Examples

```
## Not run:
foxREM <- rem_estimate(pkg_checked, check_deployments=FALSE, species="Vulpes vulpes")
hhogREM <- rem_estimate(pkg_checked, check_deployments=FALSE, species="Erinaceus europaeus")
write_rem_csv(foxREM, hhogREM)

## End(Not run)
```

Index

.with_seed, 2

addCircleMarkers, 15

check_deployment_models, 3

convert_units, 3, 21

correct_time, 4

downversion_camtrapDP, 5

ds, 7, 8

fit_actmodel, 6, 12, 22

fit_detmodel, 7, 11, 22

fit_speedmodel, 8, 12, 22

fitact, 6, 7, 22

get_agouti_url, 10

get_multiplier, 10

get_parameter_table, 4, 11

get_trap_rate, 12, 13, 22

get_traprate_data, 11, 13, 14

lnorm_confint, 15

map_deployments, 15

map_traprates, 16

merge_camtrapDP, 16

parse_date_time, 19

pkg, 17

plot_deployment_schedule, 18

read_camtrap_dp, 3, 5–7, 9, 10, 14–16, 18, 19, 22, 24–26

read_camtrap_dp_csv, 18

read_camtrapDP, 6, 17, 19, 20

recalc_events, 19, 20

rem, 21

rem_estimate, 22, 26

select_species, 22, 23

slice_camtrap_dp, 24

subset_deployments, 25

write_rem_csv, 26