density_envelope

Description

This function performs kernel density estimation. The result can either be used for plotting (`output = "short"`) or for the global envelope test (`output = "long"`). If `output = "both"` the result will be a list of the short and the long output.
The function can perform both, log-transformed density estimation and classical density estimation. The log-transformed density estimation comes from the package `logKDE`. For the log-transformed density estimation the same kernels can be used as for the classical density estimation and in addition, a logistic kernel (`Kernel = "logistic"`) is provided.
The parameters `R`, `point_density_index`, `path`, `name` are only used to construct the correct file names for loading the simulation results.

Needed packages:

- `dplyr`
- `logKDE`
- `landscapemetrics`
- `spatstat`
- `raster`

Parameters:

	Description	Value
`metric`	A landscape metric to use (only patch-level metrics)	e.g. `lsm_p_area`
`R`	(*), Radius of the Boolean model which should be used	e.g. `3`
`point_density_index`	(*), For which Intensity per hectare should the estimation be done, given as their index (usually Intensity/50). Values from, to in a vector	e.g. `c(3, 14)`
`path`	(*), If the simulation results are not stored in the current working directory, specifies the folder. Do not forget the slash at the end.	e.g. `"C:/Documents/"`
`name`	(*), Name of the `.Rdata` files containing the simulation results	e.g. `"boolean_simu"`
`iterations`	How many iteration have been used during the simulation?	e.g. `2500`
`log`	Use log-density instead of classical density?	`TRUE`, `FALSE`
`verbose`	Print some output while computing	`TRUE`, `FALSE`
`bandwidth`	Bandwidth for kernel density estimation	e.g. `1.5`
`area`	Size of the simulated areas (in pixels, not meters)	e.g. `c(161, 235)`
`res`	Resolution	e.g. `2.5`
`xlim`	Range of values for which the kernel density estimation will be performed	e.g. `c(0, 12)`
`Kernel`	Kernel to be used for the estimation	e.g. `"gaussian"`
`output`	See above	`"short"`, `"long"`, `"both"`

(*) only used for constriction the correct paths to load the `.Rdata` files containing the simulation results

Result/Value:

If `output = "short"`:

A list of length `point_density_index[2]` where each entry is a list with 3 entries: `R`, `point_density_index` and a matrix called `plot_mat`. This matrix has four columns giving the value for the mean-curve, the lower and upper bound of the envelope and the value at which the curve is evaluated.

If `output = "long"`:

A list of length `point_density_index[2]` where each entry is a list with 3 entries: `R`, `point_density_index` and a matrix called `dens_mat`. This matrix contains the raw simulations. Each column gives the value at which the curve is simulated (the value is given in the row names) and each row is one simulation. Therefore, the number of rows equals `iterations`.

If `output = "both"`:

A list containing the two lists described above.

Notice that list entries whose index is smaller than `point_density_index[1]` will be empty.

Example:

```
m_test <- density_envelope(metric = lsm_p_area,
                           R = 3,
                           Log = TRUE,
                           point_density_index = c(3, 14),
                           bandwidth = 2/sqrt(5),
                           area_size = c(161 * 2.5, 235 * 2.5),
                           xlim = c(0.0001, 0.2),
                           Kernel = "logistic",
                           output = "long")
```

plot_CI

Description

This function plots confidence intervals for a given scalar landscape metric. It does not perform any calculations but only the plotting.

Needed packages:

- dplyr

Parameters:

	Description	Value
mean_var	Variable containing the values for the mean of the simulations	e.g. m$mean_lsm_l_np
q1_var	Variable containing the lower bound of the confidence interval (e.g. the 0.025 quantile)	e.g. m$q1_lsm_l_np
q2_var	Variable containing the upper bound of the confidence interval (e.g. the 0.975 quantile)	e.g. m$q2_lsm_l_np
n_var	Variable containing the values for n (Intensity of the Boolean model) or n per hectare	e.g. m$n
R_var	Variable containing the values for R (Radius of the Boolean model)	e.g. m$R
value_2015	Value of the landscape metric which was observed in 2015 (scalar)	e.g. lsm_l_np(data15)
value_2006	Value of the landscape metric which was observed in 2006 (scalar)	e.g. lsm_l_np(data06)
n_lim	Minimum and maximum value of n as vector	e.g. c(50,700)
header	Header to be printed. If NULL (the default) no header is printed.	e.g. "Number of Gaps"
XLAB	Label for x-axis, default is ""	e.g. ""
Ylab	Label for y-axis, default is ""	e.g. ""
Ylim	Limit of the y-axis. If NULL (the default) it is determined automatically.	e.g. c(5,20)
set_par	Should par(mfrow = …) be set automatically?	TRUE, FALSE
v_lines	Should vertical lines be printed? They are ugly but make it easier to analyse the plot.	TRUE, FALSE
plot_value	Should the observed values be plotted or not? Should only be FALSE if plot_dashed is TRUE.	TRUE, FALSE
plot_dashed	If both observed values are extremely close to each other and it is impossible to plot both lines separately, one can set plot_dashed to TRUE, This leads to one single dashed linie located at the mean of both observed values instead. If doing so, one should set plot_value = FALSE.	TRUE, FALSE
plot_legend	Should the plot have a legend? Default is TRUE.	TRUE, FALSE
PDF_file	Can be used to specify a file name used to save the plot as PDF. If NULL (the default) no PDF is generated.	e.g. "PLAND_Loc1"
pdf_size	Specifies the size of the PDF containing the plot in inches. Is ignored if PDF_file = NULL.	e.g. c(6, 6)

MAR	Specifies the margin of the plot saved as PDF. If NULL (the default), global setting is used. Is ignored if PDF_file = NULL.	e.g. c(2.1, 3.9, 1.6, 1.6)

Result/Value:
None return value, just a plot.

Example:

```
load("metrics.Rdata")

m <- metrics %>% group_by(R, density) %>% summarise(
  mean_lsm_l_np = mean(lsm_l_np),
  q1_lsm_l_np = quantile(lsm_l_np, 0.025),
  q2_lsm_l_np = quantile(lsm_l_np, 0.975)
)

plot_CI(mean_var = metrics3$mean_lsm_l_np,
        q1_var = m$q1_lsm_l_np,
        q2_var = m$q2_lsm_l_np,
        n_var = m$density,
        R_var = m$R,
        value_2015 = lsm_l_np(data_2015)$value,
        value_2006 = lsm_l_np(data_2006)$value,
        header = "Number of Gaps",
        n_lim = c(50,700)
)
```

plot_envelope

Description

- This function plots the result of `density_envelope`. To do so, it performs the kernel density estimation for the study years, too.

Needed packages:

- `dplyr`
- `logKDE`
- `landscapemetrics`
- `spatstat`
- `raster`

Parameters:

	Description	Value
`envelope_data`	The result of `density_envelope` with `output = "short"`	e.g. `enn$short`
`metric`	The function for the metric	e.g. `lsm_p_enn`
`R`	Radius of the Boolean model	e.g. `3`
`intensity_index`	For which Intensity (per hectare) should the estimation be done, given as their index (usually Intensity/50)? Values listed in a vector (no from-to-form, i.e. not c(6,20) but 6:20.	e.g. `6:20` or `c(15,17,20)`
`log = FALSE`	Use log-density instead of classical density?	`TRUE, FLASE`
`bandwidth`	Bandwidth for kernel density estimation. Should be the same as used to calculate `envelope_data.`	e.g. `0.2`
`ylim`	Limit of the y-axis. If `NULL` (the default) it is determined automatically.	e.g. `c(0,0.3)`
`xlim`	Range of values for which the kernel density estimation will be performed. Involved in both kernel density estimation and plotting. Should be the same as used to calculate `envelope_data.`	e.g. `c(0, 12)`
`xlim_plot`	Limit of the x-axis. Default is `xlim_plot = xlim`. Can be used if not all of the estimated curve should be plotted.	e.g. `xlim` or `c(0, 10)`
`xlab`	Label for x-axis, default is `""`	e.g. `"ENN"`
`data_2006`	Gap map of 2006	e.g. `data__Loc1_2006`
`data_2015`	Gap map of 2015	e.g. `data__Loc1_2015`
`mfrow`	Can be used to arrange several plots in one. Sets `par(mfrow = …)`	e.g. `c(3,5)`
`Kernel`	Kernel to be used for the estimation	e.g. `"gaussian"`
`plot_legend`	Should the plot have a legend? Default is `TRUE`.	`TRUE, FALSE`
`PDF_file`	Can be used to specify a file name used to save the plot as PDF. If `NULL` (the default) no PDF is generated.	e.g. `"PLAND_Loc1"`
`pdf_size`	Specifies the size of the PDF containing the plot in inches. Is ignored if `PDF_file = NULL.`	e.g. `c(6, 6)`

MAR	Specifies the margin of the plot saved as PDF. If NULL (the default), global setting is used. Is ignored if PDF_file = NULL.	e.g. c(2.1, 3.9, 1.6, 1.6)

Result/Value:
None return value, just a plot.

Example:

```
load(file = "enn_Loc1_1.Rdata")

plot_envelope(envelope_data = enn$short,
              metric = lsm_p_enn,
              R = 3,
              intensity_index = c(15, 18),
              log = TRUE,
              bandwidth = 0.2,
              xlim = c(0, 12),
              ylim = c(0,0.3),
              xlab = "",
              data_2006 = data__Loc1_2006,
              data_2015 = data__Loc1_2015,
              mfrow = c(1,1),
              Kernel = "gaussian",
              plot_legend = FALSE,
              PDF_file = "ENN_Loc1_R3",
              pdf_size = c(6,6),
              MAR = c(2.1, 3.9, 1.6, 1.6))
```

simulate_boolean

Description
This function performs the simulation of a Boolean Model.
nd is an alternative parametrisation of n, therefore n and nd cannot be specified at the same time.

Needed packages:

- spatstat

Parameters:

	Description	Value
x	width of the result (in meters, not pixels)	e.g. 161*2.5
y	height of the result (in meters, not pixels)	e.g. 235*2.5
n	Intensity of the underlying Poisson process	e.g. 1000
nd	Intensity per hectare, alternative parametrisation of n	e.g. 300
R	Radius of the Boolean model	e.g. 3
res	Resolution, size of one pixel edge in meter	e.g. 2.5

Result/Value:
An owin-object as defined by the package spatstat. See example below on hoe to transform it to a RasterLayer object from the package raster.

Example:

```
b <- simulate_boolean(x = (161 * 2.5),
                      y = (235 * 2.5),
                      nd = 300,
                      R = 3,
                      res = 2.5)

r <- raster(b$m,
            xmn = 1,
            xmx = (161 * 2.5),
            ymn = 1,
            ymx = (235 * 2.5))
```

GET

Description

This function performs the global envelope test from the package GET. To do so, it creates a curve set in a first step. There is no difference between using this function and using the functions from the GET package directly. Therefore, its many objective is to make the code more readable by combining several steps in one function.

Needed packages:

- GET

Parameters:

	Description	Value
dens_mat	A dens_mat, which is part of the output of density_envelope if output = "long"	e.g. m[[3]]$dens_mat
obs	The observed curve, corresponds to the obs argument of create_curve_set of the GET package	See example below
GET_type	Type of GET to perform, corresponds to the type argument of global_envelope_test of the GET package	e.g. "erl"
plot	Plot the result?	TRUE, FALSE

Result/Value:

The same as when executing the function global_envelope_test of the GET package.

Example:

```
m <- density_envelope(metric = lsm_p_area,
                      Kernel = "logistic",
                      log = TRUE,
                      R = 3,
                      point_density_index = c(3, 14),
                      bandwidth = 2/sqrt(5),
                      area_size = c(161 * 2.5, 235 * 2.5),
                      xlim = c(0.0001, 0.2),
                      output = "long")

obs <- data_2006 %>%
  lsm_p_area() %>%
  filter(class == 0) %>%
  .$value %>%
  round(5) %>%
  logdensity(.,
             kernel = "logistic",
             bw = 2/sqrt(5),
             from =0.0001,
             to =0.2
             ) %>%
  .$y

GET(dens_mat = m[[3]]$dens_mat, obs = obs)
```

p_matrix_GET

Description

This function performs the global envelope test from the package GET for several theoretical models (i.e. several combinations of R and n) and returns the p values in a matrix. It uses the function GET for the calculation of each of the p values.

Needed packages:

- GET

Parameters:

	Description	Value
Rs	A vector of the values for R used.	e.g. c(3,4)
ns	A vector of the values for n used.	e.g. seq(50, 700, 50)
outputs	A list of outputs of density_envelope (with output = "long") sorted ascending by R	See example below
obs	The observed curve, corresponds to the obs argument of create_curve_set of the GET package	See example below
GET_type	Type of GET to perform, corresponds to the type argument of global_envelope_test of the GET package	e.g. "erl"

Result/Value:

A matrix of p values of several combinations of R and n. The values for R are given by the column names and the values for n are given by the rownames.

Example:

```
m1 <- density_envelope(metric = lsm_p_area,
                       Kernel = "logistic",
                       log = TRUE,
                       R = 3,
                       point_density_index = c(3, 14),
                       bandwidth = 2/sqrt(5),
                       area_size = c(161 * 2.5, 235 * 2.5),
                       xlim = c(0.0001, 0.2),
                       output = "long")

m2 <- density_envelope(metric = lsm_p_area,
                       Kernel = "logistic",
                       log = TRUE,
                       R = 4,
                       point_density_index = c(3, 14),
                       bandwidth = 2/sqrt(5),
                       area_size = c(161 * 2.5, 235 * 2.5),
                       xlim = c(0.0001, 0.2),
                       output = "long")
```

```
obs <- data_2006 %>%
  lsm_p_area() %>%
  filter(class == 0) %>%
  .$value %>%
  round(5) %>%
  logdensity(.,
             kernel = "logistic",
             bw = 2/sqrt(5),
             from =0.0001,
             to =0.2
             ) %>%
  .$y

p_matrix_GET(Rs = c(3,4),
             ns = seq(150, 700, 50),
             outputs = list(m1, m2),
             obs = obs,
             GET_type = "erl")
```