# Package ‘habitat’

August 14, 2025

**Title** An R Package for Analyzing and Comparing Habitat Changes

**Date** 2025-08-14

**Version** 1.5-10

**Author** Saman Ghasemian Sorboni [aut, cre],
Mehrdad Hadipour [aut]

**Maintainer** Saman Ghasemian Sorboni <samangh.edu@gmail.com>

**Description**
An R package for analyzing and comparing habitat and species distributions changes. Ideal for students, ecologists, environmental researchers, and conservation planners.

**Imports** terra,
raster,
ggplot2,
sf,
ggspatial,
sdm,
tidyr

**Suggests** sp

**License** GPL-3

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.2

## Contents

---

`frequency` *Raster Value Frequency Table*

---

## Description

This function generates a frequency table of raster values. It calculates the frequency of each unique value in the raster, providing a summary of the distribution of values.

## Usage

```
frequency(raster)
```

## Arguments

| | |
|---|---|
| `raster` | A SpatRaster object. This represents the raster dataset from which the frequency table will be generated. |

## Details

The function is designed to take a raster dataset and create a frequency table that summarizes the distribution of values within the raster. This is useful for understanding the composition and variability of the raster data.

## Value

A data frame containing the frequency of each raster value. The data frame includes two columns: `Value`, which represents the unique values in the raster, and `Frequency`, which indicates the number of times each value occurs.

## Examples

```
# Example usage with a SpatRaster object
library(terra)

# Create a sample raster dataset with random values between 1 and 5
raster <- rast(nrows=10, ncols=10, vals=sample(1:5, 100, replace=TRUE))

# Generate the frequency table for the raster values
freq_table <- frequency(raster)

# Display the frequency table
print(freq_table)
```

---

## `hb_agg_md_raster` *Aggregate Multidimensional Raster*

---

### Description

Summarizes the multidimensional dataset across a specific dimension (e.g., calculating the average habitat suitability over time). This is the second step in a sequence of functions to analyze habitat trends.

### Usage

```
hb_agg_md_raster(multidimensional_raster, fun = mean)
```

### Arguments

| | |
|---|---|
| `multidimensional_raster` | A `SpatRaster` object representing the multidimensional dataset. |
| `fun` | A function to apply for aggregation (e.g., mean, sum). |

### Value

A `SpatRaster` object representing the aggregated raster.

### Examples

```
# List of raster files
raster_files <- c("path/to/raster1.tif", rast("path/to/raster2.tif"))
# Corresponding time points
time_points <- c(2022, 2023)
# Create multidimensional raster layer
multidimensional_raster <- hb_multiD_raster(raster_files, time_points)

 # Aggregate multidimensional raster to compute the mean
aggregated_raster <- hb_agg_md_raster(multidimensional_raster, fun = mean)

# Analyze changes using statistical summaries
trends <- hb_analyze_changes(multidimensional_raster, fun = trend_analysis)

# Plot the time series data (output_path is optional)
hb_plot_timesrs(trends, output_path = "path/to/timesrs_plot.tif")
```

## hb_analyze_changes *Analyze Changes Using Statistical Summaries*

### Description

Applies statistical functions to identify trends, anomalies, or patterns in habitat changes. This is the third step in a sequence of functions to analyze habitat trends.

### Usage

```
hb_analyze_changes(multidimensional_raster, fun = trend)
```

### Arguments

| | |
|---|---|
| multidimensional_raster | A SpatRaster object representing the multidimensional dataset. |
| fun | A function to compute the statistical summary (e.g., trend, anomaly detection). |

### Value

A data frame containing the statistical summaries.

### Examples

```
# List of raster files
raster_files <- c("path/to/raster1.tif", rast("path/to/raster2.tif"))
# Corresponding time points
time_points <- c(2022, 2023)
# Create multidimensional raster layer
multidimensional_raster <- hb_multiD_raster(raster_files, time_points)

 # Aggregate multidimensional raster to compute the mean
aggregated_raster <- hb_agg_md_raster(multidimensional_raster, fun = mean)

# Analyze changes using statistical summaries
# NOTE: Depending on your dataset, it is normal for calculations to take some time.
trends <- hb_analyze_changes(multidimensional_raster, fun = trend_analysis)

# Plot the time series data (output_path is optional)
hb_plot_timesrs(trends, output_path = "path/to/timesrs_plot.tif")
```

## hb_an_habitat *Analyze Suitable Habitat*

### Description

Analyzes the range of suitable habitat based on a given threshold and provides statistics and a histogram plot.

### Usage

```
hb_an_habitat(x1, x2, threshold = 0.5)
```

### Arguments

| | |
|---|---|
| `x1` | Path to the input raster file (e.g., Elevation, NDVI). |
| `x2` | Path to the habitat suitability raster file (e.g., ensemble map). |
| `threshold` | Numeric value for the presence probability threshold. Default is 0.5. |

### Details

Calculates the range of suitable habitat areas using pre-aligned raster files.

### Value

A list containing the range statistics and the histogram plot.

### Examples

```
# Example usage
x1 <- "path/to/DEM.tif"
x2 <- "path/to/habitat.tif"
result <- hb_an_habitat(x1, x2, threshold = 0.5)
print(result$statistics)
print(result$plot)
```

---

## `hb_binary` *Convert Raster to Binary Map*

---

### Description

Converts a continuous raster dataset into a binary map based on a specified threshold. Values in the raster that are greater than the threshold are converted to `TRUE`, while values less than or equal to the threshold are converted to `FALSE`.

### Usage

```
hb_binary(x, th)
```

### Arguments

| | |
|---|---|
| `x` | A RasterLayer or SpatRaster object representing the continuous raster dataset to be converted. |
| `th` | A numeric threshold value between 0 and 1, determining the cutoff point for converting values to `TRUE` or `FALSE`. |

### Details

This function is designed to convert a continuous raster dataset into a binary map. This is particularly useful for applications requiring binary classification of data, such as presence/absence mapping or suitability analysis.

### Value

A binary RasterLayer or SpatRaster where values greater than the specified threshold are `TRUE`, and values less than or equal to the threshold are `FALSE`.

## Examples

```
# Example usage with a SpatRaster object


# Create a sample raster dataset with random values
r <- rast(nrows = 10, ncols = 10, vals = runif(100))

# Convert the raster to a binary map using a threshold of 0.5
binary_map <- hb_binary(r, th = 0.5)

# Plot the resulting binary map
plot(binary_map, main = "Binary Map (Threshold = 0.5)")
```

---

`hb_cal_area` *Calculate Suitable and Unsuitable Areas*

---

## Description

Calculates the suitable and unsuitable areas in hectares, square kilometers, and square meters from a binary raster.

## Usage

```
hb_cal_area(binary_raster)
```

## Arguments

| | |
|---|---|
| `binary_raster` | A SpatRaster object representing the binary raster with suitable (1/TRUE) and unsuitable (0/FALSE) areas. |

## Details

This function calculates the total area of suitable and unsuitable regions in a binary raster and prints the results in hectares, square kilometers, and square meters.

## Value

None. The results are printed in the console.

## Examples

```
# Example usage with a binary SpatRaster object


# Create a sample binary raster
binary_raster <- rast(nrows = 10, ncols = 10, vals = sample(c(0, 1), 100, replace = TRUE))

# Calculate and print the suitable and unsuitable areas
hb_cal_area(binary_raster)
```

## hb_changes — *Calculate Habitat Changes Between Two Rasters*

### Description

Compares two binary raster files and generates a new raster highlighting areas of gain, loss, stability, and absence. This function is specifically designed for comparing habitat changes between two time points or scenarios.

### Usage

```
hb_changes(raster1, raster2)
```

### Arguments

| | |
|---|---|
| `raster1` | A SpatRaster or RasterLayer object representing the first habitat map. |
| `raster2` | A SpatRaster or RasterLayer object representing the second habitat map. |

### Details

hb_changes function ensures both input rasters are in the same CRS and extent, computes the differences, and generates a new raster with categories for gain (green), loss (red), stability (yellow), and absence (gray). It is useful for ecological and environmental studies to visualize how habitats have changed over time or under different conditions.

### Value

A SpatRaster object representing the habitat changes.

### Examples

```
# Load sample rasters
raster1 <- rast("path/to/first_raster.tif")
raster2 <- rast("path/to/second_raster.tif")

# Convert to binary
raster1_binary <- hb_binary(raster1, th = 0.5)
raster2_binary <- hb_binary(raster2, th = 0.5)

# Calculate changes
changes_raster <- hb_changes(raster1_binary, raster2_binary)

# Plot changes
hb_plot_changes(changes_raster)
```

## hb_env_at_presence *Analyze environmental values at species presence*

### Description

Computes descriptive statistics and plots a histogram of an environmental raster (e.g., elevation, slope) at species presence locations defined by a binary/suitability raster and a threshold. This wraps the common workflow of masking the environmental layer by presence, extracting values, summarizing them, and visualizing their distribution.

### Usage

```
hb_env_at_presence(
  env_raster,
  suit_raster,
  threshold = 0.5,
  mask_geom = NULL,
  plot = TRUE,
  breaks = NULL,
  col = "lightgray",
  border = NA,
  main = "Environmental values at presence locations",
  xlab = "Environmental value",
  ylab = "Frequency",
  na_rm = TRUE
)
```

### Arguments

| | |
|---|---|
| env_raster | A SpatRaster (preferred) or RasterLayer of the environmental variable (e.g., elevation in meters, slope in degrees). |
| suit_raster | A SpatRaster or RasterLayer representing species suitability or presence probability. If already binary (0/1), set `threshold = NULL`. |
| threshold | Numeric or NULL. Threshold applied to `suit_raster` to derive presence (1) vs absence (0). If `NULL`, values > 0 are treated as presence. |
| mask_geom | Optional SpatVector or SpatRaster mask/crop geometry (e.g., a border). If provided, `env_raster` is cropped and masked to this geometry before analysis. |
| plot | Logical. If TRUE, a histogram is plotted immediately. |
| breaks | Integer or vector. Breaks passed to `hist()` for the histogram. Defaults to "Sturges" via `breaks = "Sturges"` if left NULL. |
| col | Character. Fill color for the histogram bars. Default `"lightgray"`. |
| border | Character. Border color for the histogram bars. Default `NA`. |
| main | Character. Plot title. Default `"Environmental values at presence locations"`. |
| xlab | Character. X-axis label. Default `"Environmental value"`. |
| ylab | Character. Y-axis label. Default `"Frequency"`. |
| na_rm | Logical. Whether to remove NA values in summaries. Default `TRUE`. |

## Details

The function aligns CRS and geometry between `env_raster` and `suit_raster` and uses nearest-neighbor methods for reprojection/resampling to preserve binary presence masks derived from the suitability raster. Presence cells are defined as:

- if `threshold` is numeric: `suit_raster >= threshold`
- if `threshold` is NULL: `suit_raster > 0`

The histogram is drawn with base R `hist()` for speed and simplicity. Returned statistics include Min, 1st Qu., Median, Mean, 3rd Qu., Max, SD, and N.

## Value

A named list with:

- `summary`: a data.frame with Min, Q1, Median, Mean, Q3, Max, SD, N
- `values`: a numeric vector of extracted environmental values at presence cells
- `presence_mask`: the SpatRaster (0/1) presence mask used for extraction

## Examples

```
## Not run:
env <- rast(system.file("ex/elev.tif", package = "terra"))
suit <- env / max(values(env), na.rm = TRUE)  # fake suitability 0..1

# Analyze elevation values where suitability >= 0.5 and plot histogram
res <- hb_env_at_presence(
  env_raster = env,
  suit_raster = suit,
  threshold = 0.5,
  plot = TRUE,
  col = "lightblue",
  main = "Elevation at presence (>= 0.5)",
  xlab = "Elevation (m)"
)
res$summary

## End(Not run)
```

---

`hb_exclude_value` *Exclude Specific Value from Binarized Raster or Vector Data*

---

## Description

Removes a specified value from binarized raster or vector data.

## Usage

```
hb_exclude_value(data, exclude_value)
```

### Arguments

| | |
|---|---|
| `data` | A binarized raster (SpatRaster) or vector (SpatVector) object. |
| `exclude_value` | The value to exclude (e.g., FALSE or 0). |

### Value

A filtered raster or vector object with the specified value removed.

### Examples

```
# Load sample raster data
raster_data <- rast("path/to/binarized_raster.tif")

# Exclude the specified value (e.g., 0)
filtered_data <- hb_exclude_value(raster_data, exclude_value = 0)

# If you have a SpatVector and want to filter it:
polygon_data <- vect("path/to/polygon.shp")
filtered_polygon <- hb_exclude_value(polygon_data, exclude_value = 0)

# Plot the results
plot(filtered_data)
plot(filtered_polygon)
```

---

## `hb_exp_csv` *Export Habitat Results to CSV*

---

### Description

Exports habitat change metrics to a CSV file. The habitat change metrics are extracted from the provided result list and saved as a CSV file at the specified file path.

### Usage

```
hb_exp_csv(result, file_path)
```

### Arguments

| | |
|---|---|
| `result` | A list containing the habitat change metrics. This list should include a component named `Compt.By.Models`, which holds the data to be exported. |
| `file_path` | The file path where the CSV file will be saved. The path should include the desired file name and the `.csv` extension. |

### Details

Designed to facilitate the export of habitat change metrics to a CSV file, making it easier to save analysis results in a format that can be readily shared and imported into other software for further analysis.

### Value

None. This function is used for its side effect of writing the data to a CSV file.

## Examples

```
# Example usage

# Create a sample result list with habitat change metrics
result <- list(Compt.By.Models = data.frame(Model = c("Model1", "Model2"),
Metric1 = c(0.1, 0.2), Metric2 = c(0.3, 0.4)))

# Export the habitat change metrics to a CSV file
hb_exp_csv(result, "habitat_results.csv")
```

---

`hb_exp_raster` *Export Raster to File*

---

## Description

Exports a raster to a specified file format (.tif, .png, or .jpg).

## Usage

```
hb_exp_raster(raster_data, file_path)
```

## Arguments

| | |
|---|---|
| `raster_data` | A raster (SpatRaster) object to be exported. |
| `file_path` | The file path where the raster will be saved. The extension should be .tif, .png, or .jpg. |

## Value

None. This function is used for its side effect of writing the raster to a file.

## Examples

```
# Load sample raster data
raster_data <- rast("path/to/raster.tif")

# Export the raster to a .tif file
hb_exp_raster(raster_data, "output_raster.tif")

# Export the raster to a .png file
hb_exp_raster(raster_data, "output_raster.png")

# Export the raster to a .jpg file
hb_exp_raster(raster_data, "output_raster.jpg")
```

## hb_exp_txt *Export Habitat Results to TXT*

### Description

Exports habitat change metrics to a TXT file. The habitat change metrics are extracted from the provided result list and saved as a TXT file at the specified file path.

### Usage

```
hb_exp_txt(result, file_path)
```

### Arguments

| | |
|---|---|
| `result` | A list containing the habitat change metrics. This list should include a component named `Compt.By.Models`, which holds the data to be exported. |
| `file_path` | The file path where the TXT file will be saved. The path should include the desired file name and the `.txt` extension. |

### Details

Designed to facilitate the export of habitat change metrics to a TXT file, making it easier to save analysis results in a text format that can be readily shared and imported into other software for further analysis.

### Value

None. This function is used for its side effect of writing the data to a TXT file.

### Examples

```
# Example usage

# Create a sample result list with habitat change metrics
result <- list(Compt.By.Models = data.frame(Model = c("Model1", "Model2"),
Metric1 = c(0.1, 0.2), Metric2 = c(0.3, 0.4)))

# Export the habitat change metrics to a TXT file
hb_exp_txt(result, "habitat_results.txt")
```

## hb_ext_eval *Extract Evaluation Metrics from sdmModels Object*

### Description

Extracts evaluation metrics including threshold, sensitivity, specificity, TSS, MCC, F1, Kappa, NMI, PHI, PPV, NPV, CCR, MCR, Omission, Commission, and Prevalence from a trained sdm model.

### Usage

```
hb_ext_eval(sdm_model)
```

### Arguments

| | |
|---|---|
| `sdm_model` | An sdmModels object generated by the sdm function. |

### Value

A data frame with evaluation metrics.

### Examples

```
# Assume that the user has already trained their model and wants to extract the-
 # threshold value based on metrics such as sensitivity (sp), specificity (se), max(sp + se),
  # kappa, positive predictive value (ppv), negative predictive value (npv),
  # normalized mutual information (nmi), correct classification rate (ccr), prevalence,
  # and specific percentages (P10, P5, P1, P0).

  library(sdm)
# sdm_model <- trained_model_object

# Extract evaluation metrics
eval_metrics <- hb_ext_eval(sdm_model)
print(eval_metrics)
```

---

## `hb_frequency` *Raster Value Frequency Table*

---

### Description

Generates a frequency table of raster values, calculating the frequency of each unique value in the raster and providing a summary of the distribution of values.

### Usage

```
hb_frequency(raster)
```

### Arguments

| | |
|---|---|
| `raster` | A SpatRaster object representing the raster dataset from which the frequency table will be generated. |

### Details

Designed to create a frequency table that summarizes the distribution of values within a raster dataset, this function is useful for understanding the composition and variability of the raster data.

### Value

A data frame containing the frequency of each raster value. The data frame includes two columns: `Value`, representing the unique values in the raster, and `Frequency`, indicating the number of times each value occurs.

### Examples

```
# Example usage with a SpatRaster object

# Create a sample raster dataset with random values between 1 and 5
raster <- rast(nrows = 10, ncols = 10, vals = sample(1:5, 100, replace = TRUE))

# Generate the frequency table for the raster values
freq_table <- hb_frequency(raster)

# Display the frequency table
print(freq_table)
```

## `hb_load_shp` *Load and Prepare Shapefile*

### Description

Loads a shapefile and ensures it is ready for compatibility with raster operations, such as CRS, extent, clipping, masking, and resolution modifications.

### Usage

```
hb_load_shp(file_path)
```

### Arguments

|  |  |
|---|---|
| `file_path` | A character string specifying the path to the shapefile. |

### Details

Reads a shapefile from the specified path and prepares it for compatibility with raster operations by aligning its CRS and extent.

### Value

An sf object containing the shapefile data.

### Examples

```
# Example usage

# Load a sample shapefile
shapefile_path <- "path/to/shapefile.shp"

# Load and prepare the shapefile
shapefile_data <- hb_load_shp(shapefile_path)

# Check the CRS and extent
print(st_crs(shapefile_data))
print(st_bbox(shapefile_data))
```

## `hb_merge` *Merge Rasters*

### Description

Merges two or more rasters into a single raster file. If you encounter any issues with `RasterLayer` objects, use the `spat_to_raster()` function to convert them.

### Usage

```
hb_merge(raster_list, output_path = NULL)
```

### Arguments

| | |
|---|---|
| `raster_list` | A list of `RasterLayer` objects to be merged. |
| `output_path` | A character string representing the file path to save the merged raster. If `NULL`, the raster will not be saved to a file. |

### Value

A `RasterLayer` object representing the merged raster.

### Examples

```
# Load sample rasters
raster1 <- raster("path/to/raster1.tif")
raster2 <- raster("path/to/raster2.tif")

# Merge rasters (two or more), and save it in '.tif' format (optional).
merged_raster <- hb_merge(list(raster1, raster2), "path/to/merged_raster.tif")

# NOTE: 'RasterLayer' issues? Use the spat_to_raster() function
raster1 <- spat_to_raster(rast("path/to/raster1.tif"))
raster2 <- spat_to_raster(rast("path/to/raster2.tif"))

# Plot the merged raster
plot(merged_raster)
```

## `hb_modify_raster` *Modify Raster Parameters*

### Description

Modifies the CRS, extent, resolution, and applies crop and mask operations to a raster, providing a flexible method for adjusting the spatial parameters of a raster dataset.

## Usage

```
hb_modify_raster(
  raster,
  crs = NULL,
  extent = NULL,
  resolution = NULL,
  crop_extent = NULL,
  mask = NULL
)
```

## Arguments

| | |
|---|---|
| `raster` | A SpatRaster object to be modified. Represents the raster dataset undergoing modifications. |
| `crs` | Optional. A character string specifying the new Coordinate Reference System (CRS) (e.g., "EPSG:4326"). If provided, the raster will be reprojected to this CRS. |
| `extent` | Optional. A numeric vector of four values specifying the new extent (xmin, xmax, ymin, ymax). If provided, the extent of the raster will be modified accordingly. |
| `resolution` | Optional. A numeric value or a vector of two numeric values specifying the new resolution. If provided, the raster will be resampled to this resolution. |
| `crop_extent` | Optional. A SpatExtent object or numeric vector specifying the extent to crop to. If provided, the raster will be cropped to this extent. |
| `mask` | Optional. A SpatRaster object to be used as a mask. If provided, the raster will be masked by this raster. |

## Details

Designed to provide a comprehensive set of modifications to a raster dataset, including changing the CRS, adjusting the extent, resampling the resolution, cropping to a specified extent, and applying a mask. These modifications are useful for preparing raster data for analysis, visualization, or integration with other spatial datasets.

## Value

A modified SpatRaster object with the specified modifications applied.

## Examples

```
# Example usage with SpatRaster objects

# Create sample raster datasets
r1 <- rast(nrows = 10, ncols = 10, vals = runif(100))
r2 <- rast(nrows = 11, ncols = 11, vals = runif(100))

# Modify the CRS and extent of the first raster
modified_r1 <- hb_modify_raster(r1, crs = "EPSG:4326", extent = c(0, 1000, 0, 1000))
plot(modified_r1, main = "Modified Raster (CRS and Extent)")

# Modify the CRS, extent, and crop the second raster
modified_r2 <- hb_modify_raster(r2, crs = "EPSG:4326", extent = c(0, 1000, 0, 1000), crop_extent = modified_r1)
```

```
plot(modified_r2, main = "Modified Raster (CRS, Extent, and Crop)")

# Apply a mask to the first raster
mask <- rast(nrows = 10, ncols = 10, vals = sample(c(0, 1), 100, replace = TRUE))
masked_r1 <- hb_modify_raster(r1, mask = mask)
plot(masked_r1, main = "Masked Raster")
```

## hb_modify_shp *Modify Shapefile Parameters*

### Description

Modifies the CRS, applies cropping, masking, and extent adjustments to a shapefile based on specified parameters or derived from another shapefile.

### Usage

```
hb_modify_shp(shapefile, crs = NULL, extent = NULL, crop = NULL, mask = NULL)
```

### Arguments

| | |
|---|---|
| shapefile | An sf object to be modified. This represents the shapefile dataset that will undergo modifications. |
| crs | Optional. A character string specifying the new Coordinate Reference System (CRS) (e.g., "EPSG:4326") or an sf object to derive the CRS from. If provided, the shapefile will be reprojected to this CRS. |
| extent | Optional. An sf object specifying the new extent. If provided, the shapefile will be modified to match this extent. |
| crop | Optional. An sf object specifying the extent to crop to. If provided, the shapefile will be cropped to this extent. |
| mask | Optional. An sf object to be used as a mask. If provided, the shapefile will be masked by this shapefile. |

### Details

Designed to provide a comprehensive set of modifications to a shapefile dataset. This includes changing the CRS, adjusting the extent, cropping, and masking using other shapefiles.

### Value

A modified sf object with the specified modifications applied.

### Examples

```
# Example usage with sf objects

# Load a sample shapefile
shapefile_path <- "path/to/shapefile.shp"
shapefile <- st_read(shapefile_path)

# Load another sample shapefile to use for operations
shapefile2_path <- "path/to/shapefile2.shp"
```

```
shapefile2 <- st_read(shapefile2_path)

# Modify the CRS using another shapefile's CRS
modified_shapefile <- hb_modify_shp(shapefile, crs = shapefile2)
plot(modified_shapefile, main = "Modified Shapefile (CRS)")

# Modify the extent using another shapefile's extent
modified_shapefile <- hb_modify_shp(shapefile, extent = shapefile2)
plot(modified_shapefile, main = "Modified Shapefile (Extent)")

# Crop the shapefile using another shapefile
modified_shapefile <- hb_modify_shp(shapefile, crop = shapefile2)
plot(modified_shapefile, main = "Cropped Shapefile")

# Apply a mask to the shapefile using another shapefile
modified_shapefile <- hb_modify_shp(shapefile, mask = shapefile2)
plot(modified_shapefile, main = "Masked Shapefile")
```

---

## `hb_multiD_raster` *Create Multidimensional Raster Layer*

---

### Description

Converts a series of raster files or loaded raster objects into a multidimensional dataset for time analysis. This is the first step in a sequence of functions to analyze habitat trends.

### Usage

```
hb_multiD_raster(raster_files, time_points)
```

### Arguments

| | |
|---|---|
| `raster_files` | A character vector of file paths to raster files or a list of `SpatRaster` objects. |
| `time_points` | A numeric vector of time points corresponding to the raster files (e.g., 2022, 2023). |

### Value

A `SpatRaster` object representing the multidimensional dataset.

### Examples

```
# List of raster files
raster_files <- c("path/to/raster1.tif", rast("path/to/raster2.tif"))
# Corresponding time points
time_points <- c(2022, 2023)
# Create multidimensional raster layer
multidimensional_raster <- hb_multiD_raster(raster_files, time_points)

 # Aggregate multidimensional raster to compute the mean
aggregated_raster <- hb_agg_md_raster(multidimensional_raster, fun = mean)

# Analyze changes using statistical summaries
trends <- hb_analyze_changes(multidimensional_raster, fun = trend_analysis)
```

```
# Plot the time series data (output_path is optional)
hb_plot_timesrs(trends, output_path = "path/to/timesrs_plot.tif")
```

---

`hb_opt_th` *Calculate Optimal Thresholds from Evaluation Metrics*

---

## Description

Calculates various optimal thresholds based on different criteria such as max(sp + se), max(kappa), and others.

## Usage

```
hb_opt_th(eval_metrics)
```

## Arguments

| | |
|---|---|
| `eval_metrics` | A data frame with evaluation metrics. |

## Value

A list with optimal thresholds for different criteria.

## Examples

```
Not run:
Assuming the user has already extracted evaluation metrics
example_function(metrics)

# Calculate optimal thresholds
thresholds <- hb_opt_th(eval_metrics)
print(thresholds)

# Example usage of the extracted threshold with existing functions:
# max_sp_se_threshold <- thresholds$max_sp_se
# binary_map <- hb_binary(r, th = max_sp_se_threshold)
# habitat_analysis <- hb_an_habitat(x1, x2, threshold = max_sp_se_threshold)
# range_analysis <- hb_range(x, y, th = max_sp_se_threshold)

End(Not run)
```

`hb_plot` *Plot Habitat Map*

## Description

Plots a habitat map with enhanced visualization options using ggplot2.

## Usage

```
hb_plot(
  raster,
  main = "Habitat Map",
  lonlat = TRUE,
  add_north_arrow = FALSE,
  background_color = "white",
  habitat_palette = "viridis",
  add_legend = TRUE
)
```

## Arguments

| | |
|---|---|
| `raster` | A SpatRaster object representing the habitat map. |
| `main` | Title for the plot. |
| `lonlat` | Logical. If TRUE, plots the habitat map on a longitude and latitude scale. |
| `add_north_arrow` | Logical. If TRUE, adds a North arrow to the plot. |
| `background_color` | Character string specifying the background color of the map. Default is "white". |
| `habitat_palette` | Character string specifying the palette for habitat areas. |
| `add_legend` | Logical. If TRUE, includes a legend in the plot. |

## Details

This function plots a habitat map with various optional parameters for enhanced visualization.

## Examples

```
# Example usage with a SpatRaster object

# Create a sample raster
raster <- rast(nrows = 10, ncols = 10, vals = runif(100))

# Plot habitat map
hb_plot(raster, main = "Sample Habitat Map", lonlat = TRUE, add_north_arrow = TRUE, background_color = "lightbl
```

## hb_plot_changes *Plot Habitat Changes (terra)*

### Description

Plots a SpatRaster highlighting areas of habitat gain, loss, stability, and absence using terra's native plotting.

### Usage

```
hb_plot_changes(
  changes_raster,
  title = "Changes",
  xlab = "Longitude",
  ylab = "Latitude",
  bg_color = "white"
)
```

### Arguments

| | |
|---|---|
| `changes_raster` | A SpatRaster object generated by `hb_changes`. |
| `title` | Optional. Title of the plot. Default is "Changes". |
| `xlab` | Optional. Label for the x-axis. Default is "Longitude". |
| `ylab` | Optional. Label for the y-axis. Default is "Latitude". |
| `bg_color` | Optional. Background color of the plot region and NA cells. Default is "white". |

### Details

Uses terra::plot() for fast, seam-free visualization of rasters.

### Examples

```
# changes_raster <- hb_changes(raster1_binary, raster2_binary)
# hb_plot_changes(changes_raster, title = "Habitat Changes")
```

## hb_plot_timesrs *Plot Time Series Data*

### Description

Visualizes the trends in habitat suitability. This is the final step in a sequence of functions to analyze and visualize habitat trends.

## Usage

```
hb_plot_timesrs(
  trends,
  lon_min = NULL,
  lon_max = NULL,
  lat_min = NULL,
  lat_max = NULL,
  low_color = "darkred",
  mid_color = "gray90",
  high_color = "darkblue",
  x_label = "Longitude",
  y_label = "Latitude",
  plot_title = "Habitat Suitability Trends",
  bg_color = "white",
  output_path = NULL
)
```

## Arguments

| | |
|---|---|
| `trends` | A `SpatRaster` object representing the trend data. |
| `lon_min` | Minimum longitude for plotting the map. |
| `lon_max` | Maximum longitude for plotting the map. |
| `lat_min` | Minimum latitude for plotting the map. |
| `lat_max` | Maximum latitude for plotting the map. |
| `low_color` | Color for the low end of the gradient. |
| `mid_color` | Color for the midpoint of the gradient. |
| `high_color` | Color for the high end of the gradient. |
| `x_label` | Custom x-axis label. |
| `y_label` | Custom y-axis label. |
| `plot_title` | Custom plot title. |
| `bg_color` | Background color for the plot. |
| `output_path` | A character string representing the file path to save the plot. If `NULL`, the plot will not be saved to a file. |

## Value

A ggplot object representing the time series plot.

## Examples

```
# Plot the time series data (output_path is optional)
hb_plot_timesrs(trends, lon_min = -10, lon_max = 10, lat_min = 35, lat_max = 45,
              low_color = "blue", mid_color = "white", high_color = "red",
            x_label = "Longitude", y_label = "Latitude", plot_title = "My Custom Title",
              bg_color = "black", output_path = "path/to/timesrs_plot.tif")
```

`hb_range` *Analyze Habitat Changes*

## Description

Computes metrics such as gain, loss, stable areas, and total changes between two binary raster maps. Provides a detailed analysis of habitat changes over time.

## Usage

```
hb_range(x, y, th)
```

## Arguments

| | |
|---|---|
| `x` | A SpatRaster object representing the current habitat (binary). This raster should contain binary values indicating habitat presence and absence. |
| `y` | A SpatRaster object representing the future habitat (binary). This raster should also contain binary values indicating habitat presence and absence. |
| `th` | A numeric threshold value between 0 and 1, used to convert continuous data into a binary format before analysis. |

## Details

Designed to compare two binary raster maps representing habitat data at different time points. Calculates various metrics to summarize the changes between the two maps, which can be used to assess the impact of environmental changes or conservation efforts.

## Value

A list containing:

- `Compt.By.Models`: A data frame with detailed metrics, including loss, gain, stable areas, and percentage changes.
- `Diff.By.Pixel`: A SpatRaster showing pixel-wise differences between the current and future habitat maps.

## Examples

```
# Example usage with SpatRaster objects

# Create sample binary raster datasets
r1 <- rast(nrows = 10, ncols = 10, vals = sample(c(0, 1), 100, replace = TRUE))
r2 <- rast(nrows = 10, ncols = 10, vals = sample(c(0, 1), 100, replace = TRUE))

# Analyze habitat changes
result <- hb_range(r1, r2, th = 0.5)

# Display the computed metrics
print(result$Compt.By.Models)

# Plot the habitat changes
hb_plot(result$Compt.By.Models)
```

## `hb_range_plot` *Plot Habitat Changes*

### Description

Plots habitat changes as a bar chart. The bar chart visualizes the percentage of loss, gain, and overall species range change.

### Usage

```
hb_range_plot(data)
```

### Arguments

| | |
|---|---|
| `data` | A data frame containing the habitat change metrics. This data frame should include the percentage metrics such as `PercLoss`, `PercGain`, and `SpeciesRangeChange`. |

### Details

Designed to take the habitat change metrics computed by the `hb_habitat_range` function and visualize them in a bar chart. This helps in understanding the extent of habitat changes visually.

### Value

None. This function is used for its side effect of creating and displaying the plot.

### Examples

```
# Example usage with the results from hb_habitat_range function

# Assume result is obtained from hb_habitat_range function
# result <- hb_habitat_range(r1, r2, th = 0.5)

# Plot the habitat changes
hb_range_plot(result$Compt.By.Models)
```

## `hb_ras_to_pol` *Convert Raster to Polygon*

### Description

Converts a binary or continuous SpatRaster object to a polygon.

### Usage

```
hb_ras_to_pol(raster, binary = FALSE)
```

### Arguments

| | |
|---|---|
| `raster` | A SpatRaster object to be converted. |
| `binary` | Logical. If `TRUE`, the raster will be treated as binary. Default is `FALSE`. |

## Details

The function is designed to convert a raster dataset into a polygon, which is useful for visualizing raster data as vector data and performing vector-based spatial analyses.

## Value

An sf object representing the raster converted to polygons.

## Examples

```
# Example usage with SpatRaster objects

# Create a sample binary raster
binary_raster <- rast(nrows = 10, ncols = 10, vals = sample(c(0, 1), 100, replace = TRUE))

# Convert the binary raster to polygons
binary_polygons <- hb_raster_to_polygon(binary_raster, binary = TRUE)

# Plot the resulting polygons
plot(binary_polygons)

# Create a sample continuous raster
continuous_raster <- rast(nrows = 10, ncols = 10, vals = runif(100))

# Convert the continuous raster to polygons
continuous_polygons <- hb_ras_to_pol(continuous_raster)

# Plot the resulting polygons
plot(continuous_polygons)
```

---

`hb_reclass` *Reclassify Raster Values*

---

## Description

Reclassifies raster values based on specified bins. The raster values are grouped into specified bins, and each bin is assigned a new value according to the provided values vector.

## Usage

```
hb_reclass(raster, bins, values)
```

## Arguments

| | |
|---|---|
| `raster` | A SpatRaster object to be reclassified. Represents the raster dataset whose values are to be reclassified. |
| `bins` | A numeric vector defining the breakpoints for reclassification. These breakpoints specify the intervals for reclassification. |
| `values` | A numeric vector defining the new values for each bin. Each element in this vector corresponds to a bin defined by the `bins` vector. |

## Details

Designed to take a continuous or categorical raster dataset and reclassify its values based on specified breakpoints (bins). This is useful for simplifying or categorizing raster data for further analysis, visualization, or modeling.

## Value

A reclassified SpatRaster object with values reclassified according to the specified bins and new values.

## Examples

```
# Example usage with a SpatRaster object

# Create a sample raster dataset with random values
raster <- rast(nrows = 10, ncols = 10, vals = runif(100))

# Reclassify the raster values using specified bins and new values
reclassified_raster <- hb_reclass(raster, bins = c(0, 0.25, 0.5, 0.75, 1), values = c(1, 2, 3, 4))

# Plot the resulting reclassified raster
plot(reclassified_raster, main = "Reclassified Raster Values")
```

---

`hb_res_check` *Validate Raster Inputs*

---

## Description

Checks whether two raster datasets have identical extent, CRS (Coordinate Reference System), dimensions, and resolution, ensuring compatibility for further analysis.

## Usage

```
hb_res_check(x, y)
```

## Arguments

| | |
|---|---|
| x | A RasterLayer or SpatRaster object representing the first dataset. This is the initial raster that will be compared. |
| y | A RasterLayer or SpatRaster object representing the second dataset. This is the raster that will be compared against the first raster. |

## Details

Designed to validate the compatibility of two raster datasets by checking their extent, CRS, dimensions, and resolution. This is crucial for ensuring that subsequent spatial analyses can be performed accurately without encountering alignment issues.

## Value

Returns `TRUE` if all checks pass, otherwise throws an error indicating which check failed.

### Examples

```
# Example usage with SpatRaster objects

# Create sample raster datasets
r1 <- rast(nrows = 10, ncols = 10, vals = runif(100))
r2 <- rast(nrows = 10, ncols = 10, vals = runif(100))

# Validate that the raster datasets have identical properties
hb_res_check(r1, r2) # Should return TRUE if all checks pass
```

## `hb_sstat` *Summary Statistics for Raster Data*

### Description

Provides summary statistics for a given SpatRaster object, calculating key statistical measures to summarize the distribution of values within the raster dataset.

### Usage

```
hb_sstat(raster)
```

### Arguments

| | |
|---|---|
| `raster` | A SpatRaster object representing the raster dataset for which summary statistics will be generated. |

### Details

Designed to take a raster dataset and compute summary statistics that provide insights into the data's distribution and variability. These statistics are useful for understanding the overall characteristics of the raster data.

### Value

A list containing summary statistics of the raster values, including the mean, median, standard deviation, minimum, and maximum of the raster values.

### Examples

```
# Example usage with a SpatRaster object

# Create a sample raster dataset with random values
raster <- rast(nrows = 10, ncols = 10, vals = runif(100))

# Generate the summary statistics for the raster
raster_summary <- hb_sstat(raster)

# Display the summary statistics
print(raster_summary)
```

## hb_stack_dfs *Stack Data Frames*

### Description

Stacks multiple data frames into a single data frame.

### Usage

```
hb_stack_dfs(dfs)
```

### Arguments

| | |
|---|---|
| `dfs` | A list of data frames to be stacked. |

### Details

This function takes a list of data frames and stacks them into a single data frame by row binding. It is useful for combining multiple data frames for further analysis.

### Value

A single data frame that combines all input data frames by row binding.

### Examples

```
# Example usage with data frames

# Create sample data frames
df1 <- data.frame(a = 1:5, b = letters[1:5])
df2 <- data.frame(a = 6:10, b = letters[6:10])

# Stack the data frames
stacked_df <- hb_stack_dfs(list(df1, df2))
print(stacked_df)
```

## hb_stack_rasters *Stack Raster Layers*

### Description

Stacks multiple SpatRaster objects into a single SpatRaster object.

### Usage

```
hb_stack_rasters(rasters)
```

### Arguments

| | |
|---|---|
| `rasters` | A list of SpatRaster objects to be stacked. |

## Details

This function takes a list of SpatRaster objects and stacks them into a single SpatRaster object. It is useful for combining multiple raster layers for further analysis.

## Value

A SpatRaster object that combines all input raster layers.

## Examples

```
# Example usage with SpatRaster objects

# Create sample raster datasets
raster1 <- rast(nrows = 10, ncols = 10, vals = runif(100))
raster2 <- rast(nrows = 10, ncols = 10, vals = runif(100))

# Stack the rasters
stacked_rasters <- hb_stack_rasters(list(raster1, raster2))
plot(stacked_rasters)
```

---

`raster_to_spat` *Convert RasterLayer to SpatRaster*

---

## Description

Converts a RasterLayer object to a SpatRaster object.

## Usage

```
raster_to_spat(raster_layer)
```

## Arguments

| | |
|---|---|
| `raster_layer` | A RasterLayer object. |

## Value

A SpatRaster object.

---

`spat_to_raster` *Convert SpatRaster to RasterLayer*

---

## Description

Converts a SpatRaster object to a RasterLayer object.

## Usage

```
spat_to_raster(spat_raster)
```

## Arguments

| | |
|---|---|
| `spat_raster` | A SpatRaster object. |

## Value

A RasterLayer object.

# Index