Package: saltbush (via r-universe)

November 13, 2026

Title Spectral Diversity from Drone Images

Version 0.2.0

Description Processes drone data to calculate spectral diversity metrics. To facilitate testing of the spectral-biodiversity relationship, it also calculates on-the-ground diversity from on-the-ground, point-intercept surveys specifically from the AusPlots framework.

License GPL (>= 3)

URL https://github.com/traitecoevo/saltbush, https://traitecoevo.github.io/saltbush/

BugReports https://github.com/traitecoevo/saltbush/issues

Depends R (>= 4.0.0)

Imports dplyr, pROC, sf, tidyr, vegan, geometry, terra

Suggests ausplotsR, knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/Needs/website pkgdown

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

LazyData true

Config/roxygen2/version 7.3.3.9000

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev libicu-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev

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

Date/Publication 2026-11-07 22:56:58 UTC

RemoteUrl https://github.com/traitecoevo/saltbush

RemoteRef HEAD

RemoteSha 10094b22cbbd948366cab2e1b8ee529915e316c5

Contents

ausplots_NSABHC0009 . . . 2
calculate_field_diversity . . . 3
calculate_spectral_metrics . . . 4
cover_survey . . . 6
create_masked_raster . . . 7
create_multiband_image . . . 8
find_optimum_thresholds . . . 9
synthetic_survey . . . 10

Index **12**

`ausplots_NSABHC0009` *Cached AusPlots point-intercept data for site NSABHC0009*

Description

A cached subset of the point-intercept (`veg.PI`) table for AusPlots site `NSABHC0009` at Fowlers Gap, NSW. Saved as a package dataset so the worked example and tests do not require live access to the AusPlots database.

Usage

```
data(ausplots_NSABHC0009)
```

Format

A data frame with one row per point-intercept hit:

site_unique Character. Unique site/visit identifier.

site_location_name Character. Site identifier.

site_location_visit_id Integer. AusPlots visit identifier.

transect Character. Transect identifier within the visit.

point_number Integer. Point number along the transect.

standardised_name Character. Standardised species name.

Details

Only the columns used by saltbush plus a few useful identifiers are retained — taxonomy columns and free-text fields are dropped to keep the file small.

Source

Cached from `ausplotsR::get_ausplots()` with `my.Plot_IDs = "NSABHC0009"`. See `data-raw/ausplots_NSABHC0009.R`

`calculate_field_diversity` *Calculate field diversity*

Description

Takes species survey data and calculates field diversity at the grouping specified by `aoi_cols`. The defaults suit AusPlots point-intercept data from ausplotsR: grouping by `site_unique` (one row per survey visit), species in `standardised_name`, dead material excluded, and each row counted as one hit. Other survey data can be used by changing `species_col`, `exclude` and `abundance_col`.

Usage

```
calculate_field_diversity(
  survey_data,
  aoi_cols = "site_unique",
  species_col = "standardised_name",
  exclude = c("Dead grass", "Dead shrub"),
  abundance_col = NULL,
  group_by_cols = NULL
)
```

Arguments

`survey_data`	a data frame of survey records, e.g. AusPlots point-intercept data from ausplotsR, with one row per record.
`aoi_cols`	character vector of column names in `survey_data` that identify each area of interest (AOI), i.e. the unit diversity is calculated for. Defaults to `"site_unique"`, which gives one row per survey visit. Pass e.g. `c("site_unique", "subplot_id")` to compute diversity per subplot within each site, to match subplot AOIs in `calculate_spectral_metrics()`. (Previously `group_by_cols`.)
`species_col`	name of the column holding species names. Defaults to `"standardised_name"` (ausplotsR).
`exclude`	values of `species_col` to leave out, such as dead material. Defaults to `c("Dead grass", "Dead shrub")`; use `NULL` to keep everything.
`abundance_col`	optional name of a numeric column holding the abundance of each record (e.g. counts or percent cover). When `NULL` (default), each row counts as one hit, as in point-intercept data. Records with zero abundance are treated as absent; negative or missing values are an error.
`group_by_cols`	deprecated; use `aoi_cols`.

Value

a list, containing 'taxonomic_diversity' - values of species richness, shannon's index, simpsons index, exponential shannon's, inverse simpson, pielou's evenness, with one row per unique combination of `aoi_cols`, and 'community_matrices' - a list of per-group community matrices keyed by

the grouping values; each matrix has the `aoi_cols` as leading id columns followed by one column per species.

Examples

```
# Default: one row per survey visit, using the bundled synthetic data.
site_div <- calculate_field_diversity(synthetic_survey)
site_div$taxonomic_diversity

# Subplot-level diversity within each visit:
subplot_div <- calculate_field_diversity(
  synthetic_survey,
  aoi_cols = c("site_unique", "subplot_id")
)
subplot_div$taxonomic_diversity

# Cached real AusPlots point-intercept data for site NSABHC0009:
field_diversity_real <- calculate_field_diversity(ausplots_NSABHC0009)
field_diversity_real$taxonomic_diversity

# To pull fresh data from the AusPlots database (requires network):
## Not run:
ausplot.pi.data <- ausplotsR::get_ausplots(
  my.Plot_IDs = c("SATFLB0004", "QDAMGD0022", "NTASTU0002"),
  veg.PI = TRUE
)$veg.PI
calculate_field_diversity(ausplot.pi.data)

## End(Not run)
```

`calculate_spectral_metrics` *Calculate spectral metrics*

Description

Calculates CV, SV, and CHV for each area of interest (AOI) directly from multiband rasters, without first extracting every pixel into a data frame. Each raster is cropped and masked to each AOI polygon; CV and SV are calculated from per-band summary statistics (`terra::global()`), and CHV from the matrix of pixel values within the AOI.

Spectral variance (SV) is standardized by the number of spectral points, so it is not rarefied; `rarefaction` applies to CV and CHV only.

Usage

```
calculate_spectral_metrics(
  raster_files,
  aoi_files,
  wavelengths,
```

```
  aoi_id_col = NULL,
  masked = TRUE,
  rarefaction = FALSE,
  min_points = NULL,
  n = NULL,
  seed = NULL
)
```

Arguments

raster_files	Character vector of multiband raster file paths. The site name is taken from the start of each file name, up to the first _ or . (or the whole name, minus extension, if there is neither), so NSABHC0009_ortho.tif, NSABHC0009.ortho.tif and NSABHC0009.tif all give the site NSABHC0009.
aoi_files	Character vector of AOI file paths: shapefiles (.shp), GeoJSON (.geojson/.json) or GeoPackage (.gpkg), each containing one or more AOI polygons. Each raster is matched to the AOI file with the same site name, found the same way (e.g. NSABHC0009_aoi.shp). Files with other extensions (e.g. a shapefile's .dbf) are ignored. If an AOI's coordinate system differs from its raster's, the AOI is reprojected to match.
wavelengths	Character vector naming the wavelength of each raster layer, in layer order.
aoi_id_col	Optional character; name of a column in the AOI file to use as the aoi_id. When NULL (default), AOIs are numbered 1:n.
masked	Logical; labels the output image_type as "masked" or "unmasked". It does not mask the image.
rarefaction	Logical; if TRUE, CV and CHV are averaged over n random subsamples of min_points pixels per AOI, which standardizes uneven pixel numbers across AOIs and increases processing time.
min_points	Integer; pixels sampled per AOI when rarefaction = TRUE.
n	Integer; number of subsamples when rarefaction = TRUE.
seed	Integer or NULL; if provided, set.seed(seed) is called once at the start so the rarefaction sampling is reproducible.

Value

A data frame with one row per AOI per raster, and columns site, aoi_id, CV, SV, CHV and image_type.

Examples

```
aoi_files <- list.files(
  system.file("extdata/aoi", package = "saltbush"),
  pattern = "image_aoi\\.shp$", full.names = TRUE
)
raster_files <- list.files(
  system.file("extdata/multiband_image", package = "saltbush"),
  pattern = "\\.tif$", full.names = TRUE
```

```
)
metrics <- calculate_spectral_metrics(
  raster_files, aoi_files,
  wavelengths = c("blue", "green", "red", "red_edge", "nir"),
  masked = FALSE
)
metrics
```

`cover_survey` *Synthetic percent-cover survey data*

Description

A small, hand-crafted percent-cover survey for three plots, with species typical of chenopod shrubland at Fowlers Gap, NSW. Each row is one species' cover estimate in a plot (unlike synthetic_survey, where each row is a point-intercept hit), and non-plant categories are included. Used to show `calculate_field_diversity()` on survey data other than AusPlots, via `species_col`, `exclude` and `abundance_col`. The values are made up.

Usage

```
cover_survey
```

Format

A data frame with 15 rows and 3 columns:

plot_id Character. Plot identifier.

species Character. Species name, or a non-plant category (`"Bare ground"`, `"Litter"`).

cover_pct Numeric. Percent cover of the species in the plot.

Source

Hand-crafted for package examples. See `data-raw/cover_survey.R`.

Examples

```
calculate_field_diversity(cover_survey,
                          aoi_cols = "plot_id",
                          species_col = "species",
                          exclude = c("Bare ground", "Litter"),
                          abundance_col = "cover_pct")$taxonomic_diversity
```

`create_masked_raster` *Create masked raster from multiband image*

Description

This function creates a masked raster by removing non-vegetation pixels based on NDVI and NIR threshold values. It supports processing single files, multiple files in a directory, or a list of file paths. Threshold values can be specified as single values or as a data frame with site-specific thresholds. The function assumes that the input layers are stacked in wavelength order and saves the masked raster to the specified output directory.

Usage

```
create_masked_raster(
  input,
  output_dir = tempdir(),
  ndvi_threshold = NULL,
  nir_threshold = NULL,
  ndvi_threshold_df = NULL,
  nir_threshold_df = NULL,
  red_band_index = 3,
  nir_band_index = 5,
  make_plot = FALSE,
  return_raster = FALSE
)
```

Arguments

`input`	A directory containing multiple ENVI or TIF files, a single file, or a character vector of file paths.
`output_dir`	A directory where the masked raster(s) will be saved.
`ndvi_threshold`	Numeric. NDVI threshold for masking (pixels with NDVI values below this are masked as non-vegetation).
`nir_threshold`	Numeric. NIR threshold for masking (pixels with NIR values below this are masked as shadows).
`ndvi_threshold_df`	Optional. A data frame with two columns: `site` (site identifiers) and `threshold` (NDVI threshold values for each site). The `site` values must match the beginning of the input file names.
`nir_threshold_df`	Optional. A data frame with two columns: `site` (site identifiers) and `threshold` (NIR threshold values for each site). The `site` values must match the beginning of the input file names.
`red_band_index`	Integer. The index (layer number) of the red band in the input raster. Default is 3.

nir_band_index	Integer. The index (layer number) of the NIR band in the input raster. Default is 5.
make_plot	Logical. If TRUE, a plot of the masked raster is generated for verification. Default is FALSE.
return_raster	Logical. If TRUE, the masked raster object is returned. Default is FALSE.

Value

Saves masked raster(s) as TIF files in the specified output_dir. If return_raster = TRUE, the function returns the masked raster object as a terra raster.

Examples

```
input <- system.file("extdata/multiband_image", package = "saltbush")
output_dir <- tempdir()
create_masked_raster(input,
                     output_dir,
                     ndvi_threshold = 0.02,
                    nir_threshold = 0.04,
                      red_band_index = 3,
                      nir_band_index = 5)
```

---

create_multiband_image

Create a multiband image from single-band TIF files

Description

This function combines single reflectance band TIF files into a multiband image. It saves the combined multiband image to a file and, by default, also returns it and plots it for checking. Bands are ordered by the specified wavelength order. All bands in a folder must share the same extent, resolution and coordinate reference system (CRS); otherwise the function stops and lists the bands that differ.

Usage

```
create_multiband_image(
  input_dir,
  desired_band_order,
  output_dir = tempdir(),
  make_plot = TRUE,
  return_raster = TRUE
)
```

Arguments

input_dir	A directory containing folders with waveband TIF files to be combined, or a single folder with waveband TIF files to be combined.
desired_band_order	A character vector of band names, in wavelength order. Each band is matched to the TIF whose file name (excluding extension) is the band name or ends with it after a separator, so both blue.tif and Pix4D output such as NSABHC0009_transparent_reflectance_b match "blue". Matching ignores case, and treats spaces, hyphens and dots as underscores ("red edge" matches "red_edge"). Other TIFs in the folder are ignored. The output layers are named by these band names.
output_dir	A directory where the combined multiband TIF file will be saved.
make_plot	Logical. If TRUE, a plot is generated to visualize the output for verification. Default is TRUE.
return_raster	Logical. If TRUE, the combined raster object is returned. Default is TRUE.

Value

Saves a combined multiband TIF file in the specified output_dir. If return_raster = TRUE (default), the function also returns the combined multiband image as a terra raster, or a named list of rasters when input_dir holds several folders.

Examples

```
input_dir <- system.file("extdata/create_multiband_image", package = "saltbush")
output_dir <- tempdir()
create_multiband_image(input_dir, c('blue', 'green', 'red', 'red_edge', 'nir'), output_dir)
```

---

find_optimum_thresholds

Find optimum thresholds

Description

Takes input ground truth data to find optimum thresholds for image masking.

Usage

```
find_optimum_thresholds(
  df,
  class_col,
  band_or_index_col,
  site_col,
  class_value
)
```

Arguments

df	Input must be a df with columns for site name, pixel value, class
class_col	name of column where object class are stored
band_or_index_col	name of column where pixel values are stored - e.g. 'ndvi' or 'nir_values'
site_col	name of site column - which indicates where pixel values + class data are obtained from
class_value	value/s of class that should be masked - e.g. for ndvi, 'non-veg' or c('bare_ground','rocks'), for nir, 'shadow'

Value

a df with optimum thresholds for each site

Examples

```
ndvi_values <- data.frame(site = rep(c("site_one", "site_two"), each = 100),
point = rep(1:100, times = 2),
ndvi = runif(200, min = -1, max = 1),
class = sample(rep(c('veg', 'non-veg'), each = 100)))
ndvi_thresholds <- find_optimum_thresholds(ndvi_values,
class_col = 'class', band_or_index_col = 'ndvi',
site_col = 'site', class_value = 'non-veg')
```

---

synthetic_survey *Synthetic point-intercept survey data*

Description

A small, hand-crafted data frame that mimics the shape of the `veg.PI` table returned by `ausplotsR::get_ausplots()`. Two sites, each with two subplots and a handful of species hits. Used in examples, tests, and the package vignette so they can run without an internet connection.

Usage

```
data(synthetic_survey)
```

Format

A data frame with 12 rows and 4 columns:

site_unique Character. Unique site/visit identifier.

site_location_name Character. Site identifier (the prefix of `site_unique`).

subplot_id Character. Subplot identifier within the site.

standardised_name Character. Species name for the point hit.

Source

Hand-crafted for package examples. See `data-raw/synthetic_survey.R`.

Index

∗ **datasets**
ausplots_NSABHC0009, 2
cover_survey, 6
synthetic_survey, 10

ausplots_NSABHC0009, 2
ausplotsR::get_ausplots(), *2*, *10*

calculate_field_diversity, 3
calculate_field_diversity(), *6*
calculate_spectral_metrics, 4
calculate_spectral_metrics(), *3*
cover_survey, 6
create_masked_raster, 7
create_multiband_image, 8

find_optimum_thresholds, 9

synthetic_survey, *6*, 10