Package 'lightr'

April 1, 2020

Title Read Spectrometric Data and Metadata

Version 1.1

Description Parse various reflectance/transmittance/absorbance spectra file formats to extract spectral data and metadata, as described in Gruson, White & Maia (2019) <doi:10.21105/joss.01857>. Among other formats, it can import files from 'Avantes' <https://www.avantes.com/>, 'CRAIC' <http://www.microspectra.com/>, and 'OceanInsight' (formerly 'OceanOptics') <https://www.oceaninsight.com/> brands.
This package has been peer-reviewed by rOpenSci (v. 0.1).

Depends R (>= 3.5.0)

Imports future.apply, progressr, xml2 (>= 1.0.0)

Suggests knitr, pavo, rmarkdown, spelling, testthat (>= 2.0.0)

URL https://docs.ropensci.org/lightr, https://github.com/ropensci/lightr

BugReports https://github.com/ropensci/lightr/issues

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

Language en-GB

VignetteBuilder knitr

NeedsCompilation no

Author Hugo Gruson [cre, aut] (<https://orcid.org/0000-0002-4094-1476>), Rafael Maia [aut] (<https://orcid.org/0000-0002-7563-9795>), Thomas White [aut] (<https://orcid.org/0000-0002-3976-1734>), Kotya Karapetyan [cph] (Author of the MATLAB script to read AvaSoft7 binary files (CC-BY))

Maintainer Hugo Gruson <hugo.gruson+R@normalesup.org>

Repository CRAN

Date/Publication 2020-04-01 14:10:06 UTC

R topics documented:

`compute_processed` *Compute processed spectral data*

Description

Compute processed spectral data, from the raw count/scope data, counts from a dark reference, and from a white reference.

Usage

```
compute_processed(spdata)
```

Arguments

`spdata`	data.frame containing the spectral data with the columns 'scope', 'dark', and 'white'

`lr_convert_tocsv` *Convert spectral data files to csv files*

Description

Convert spectral data files to csv files

Usage

```
lr_convert_tocsv(
  where = NULL,
  ext = "txt",
  decimal = ".",
  sep = NULL,
  subdir = FALSE,
  cores = NULL,
  ignore.case = TRUE,
  overwrite = FALSE
)
```

Arguments

where	Folder in which files are located (defaults to current working directory).
ext	File extension to be searched for, without the "." (defaults to txt). You can also use a character vector to specify multiple file extensions.
decimal	Character to be used to identify decimal plates (defaults to .).
sep	Column delimiting characters to be considered in addition to the default (which are: tab, space, and ";")
subdir	Should subdirectories within the where folder be included in the search? (defaults to FALSE).
cores	deprecated. See future::plan() for more details on how to customise your parallelisation strategy.
ignore.case	Should the extension search be case insensitive? (defaults to TRUE)
overwrite	logical. Should the function overwrite existing files with the same name? (defaults to FALSE).

Details

You can customise the type of parallel processing used by this function with the future::plan() function. This works on all operating systems, as well as high performance computing (HPC) environment. Similarly, you can customise the way progress is shown with the progressr::handlers() functions (progress bar, acoustic feedback, nothing, etc.)

Value

Convert input files to csv and invisibly return the list of created file paths

Warning

This step loses all metadata associated to the spectra. This metadata is critical to ensure reproducibility. We recommended you use lr_get_metadata() to extract this information from your raw data.

`lr_get_metadata` *Extract metadata from spectra files*

Description

Finds and imports metadata from spectra files in a given location.

Usage

```
lr_get_metadata(
  where = getwd(),
  ext = "ProcSpec",
  sep = NULL,
  subdir = FALSE,
  subdir.names = FALSE,
  cores = NULL,
  ignore.case = TRUE
)
```

Arguments

`where`	Folder in which files are located (defaults to current working directory).
`ext`	File extension to be searched for, without the "." (defaults to `txt`). You can also use a character vector to specify multiple file extensions.
`sep`	Column delimiting characters to be considered in addition to the default (which are: tab, space, and ";")
`subdir`	Should subdirectories within the `where` folder be included in the search? (defaults to `FALSE`).
`subdir.names`	Should subdirectory path be included in the name of the spectra? (defaults to `FALSE`).
`cores`	deprecated. See `future::plan()` for more details on how to customise your parallelisation strategy.
`ignore.case`	Should the extension search be case insensitive? (defaults to `TRUE`)

Details

You can customise the type of parallel processing used by this function with the `future::plan()` function. This works on all operating systems, as well as high performance computing (HPC) environment. Similarly, you can customise the way progress is shown with the `progressr::handlers()` functions (progress bar, acoustic feedback, nothing, etc.)

Value

A data.frame containing one file per row and the following columns:

- `name`: File name (without the extension)
- `user`: Name of the spectrometer operator
- `date`: Timestamp of the recording (ISO 8601 format)
- `spec_model`: Model of the spectrometer
- `spec_ID`: Unique ID of the spectrometer
- `white_inttime`: Integration time of the white reference (in ms)
- `dark_inttime`: Integration time of the dark reference (in ms)
- `sample_inttime`: Integration time of the sample (in ms)
- `white_avgs`: Number of averaged measurements for the white reference
- `dark_avgs`: Number of averaged measurements for the dark reference
- `sample_avgs`: Number of averaged measurements for the sample
- `white_boxcar`: Boxcar width for the white reference
- `dark_boxcar`: Boxcar width for the dark reference
- `sample_boxcar`: Boxcar width for the sample reference

Warning

`white_inttime`, `dark_inttime` and `sample_inttime` should be equal. The normalised data may be inaccurate otherwise.

References

White TE, Dalrymple RL, Noble DWA, O'Hanlon JC, Zurek DB, Umbers KDL. Reproducible research in the study of biological coloration. Animal Behaviour. 2015 Aug 1;106:51-7 (doi: 10.1016/j.anbehav.2015.05.007).

Examples

```
lr_get_metadata(system.file("testdata", "procspec_files",
                            package = "lightr"),
                ext = "ProcSpec")
```

`lr_get_spec` *Extract spectral data from spectra files*

Description

Finds and imports reflectance/transmittance/absorbance data from spectra files in a given location.

Usage

```
lr_get_spec(
  where = getwd(),
  ext = "txt",
  lim = c(300, 700),
  decimal = ".",
  sep = NULL,
  subdir = FALSE,
  subdir.names = FALSE,
  cores = NULL,
  ignore.case = TRUE,
  interpolate = TRUE
)
```

Arguments

`where`	Folder in which files are located (defaults to current working directory).
`ext`	File extension to be searched for, without the "." (defaults to `txt`). You can also use a character vector to specify multiple file extensions.
`lim`	A vector with two numbers determining the wavelength limits to be considered (defaults to `c(300,700)`).
`decimal`	Character to be used to identify decimal plates (defaults to `.`).
`sep`	Column delimiting characters to be considered in addition to the default (which are: tab, space, and ";")
`subdir`	Should subdirectories within the `where` folder be included in the search? (defaults to `FALSE`).
`subdir.names`	Should subdirectory path be included in the name of the spectra? (defaults to `FALSE`).
`cores`	deprecated. See `future::plan()` for more details on how to customise your parallelisation strategy.
`ignore.case`	Should the extension search be case insensitive? (defaults to `TRUE`)
`interpolate`	Boolean indicated whether spectral data should be interpolated and pruned at every nanometre. Note that this option can only work if all input data samples the same wavelengths. Defaults to `TRUE`.

Details

You can customise the type of parallel processing used by this function with the `future::plan()` function. This works on all operating systems, as well as high performance computing (HPC) environment. Similarly, you can customise the way progress is shown with the `progressr::handlers()` functions (progress bar, acoustic feedback, nothing, etc.)

Value

A data.frame, containing the wavelengths in the first column and individual imported spectral files in the subsequent columns.

See Also

`pavo::getspec()`

Examples

```
lr_get_spec(system.file("testdata", package = "lightr"), ext = "jdx")
```

`lr_parse_generic` *Generic function to parse spectra files that don't have a specific parser*

Description

Generic function to parse spectra files that don't have a specific parser

Usage

```
lr_parse_generic(filename, decimal = ".", sep = NULL)
```

Arguments

`filename`	Path of the file to parse
`decimal`	Character to be used to identify decimal plates (defaults to `.`).
`sep`	Column delimiting characters to be considered in addition to the default (which are: tab, space, and ";")

Details

'processed' column computed by official software and provided as is.

Value

A list of two elements:

- a dataframe with columns "wl", "dark", "white", "scope" and "processed", in that order
- a character vector with metadata including:
  - `user`: Name of the spectrometer operator
  - `date`: Timestamp of the recording (ISO 8601 format)
  - `spec_model`: Model of the spectrometer
  - `spec_ID`: Unique ID of the spectrometer
  - `white_inttime`: Integration time of the white reference (in ms)
  - `dark_inttime`: Integration time of the dark reference (in ms)
  - `sample_inttime`: Integration time of the sample (in ms)
  - `white_avgs`: Number of averaged measurements for the white reference
  - `dark_avgs`: Number of averaged measurements for the dark reference
  - `sample_avgs`: Number of averaged measurements for the sample
  - `white_boxcar`: Boxcar width for the white reference
  - `dark_boxcar`: Boxcar width for the dark reference
  - `sample_boxcar`: Boxcar width for the sample reference

Examples

```
lr_parse_generic(system.file("testdata", "spec.csv", package = "lightr"),
                 sep = ",")
lr_parse_generic(system.file("testdata", "CRAIC_export.txt",
                             package = "lightr"))
```

---

lr_parse_jaz *Parse OceanInsight converted file*

Description

Parse OceanInsight (formerly OceanOptics) converted file. https://www.oceaninsight.com/

Usage

```
lr_parse_jaz(filename)

lr_parse_jazirrad(filename)
```

Arguments

`filename`	Path of the file to parse

Details

'processed' column computed by official software and provided as is.

Value

A list of two elements:

- a dataframe with columns "wl", "dark", "white", "scope" and "processed", in that order
- a character vector with metadata including:
  - `user`: Name of the spectrometer operator
  - `date`: Timestamp of the recording (ISO 8601 format)
  - `spec_model`: Model of the spectrometer
  - `spec_ID`: Unique ID of the spectrometer
  - `white_inttime`: Integration time of the white reference (in ms)
  - `dark_inttime`: Integration time of the dark reference (in ms)
  - `sample_inttime`: Integration time of the sample (in ms)
  - `white_avgs`: Number of averaged measurements for the white reference
  - `dark_avgs`: Number of averaged measurements for the dark reference
  - `sample_avgs`: Number of averaged measurements for the sample
  - `white_boxcar`: Boxcar width for the white reference
  - `dark_boxcar`: Boxcar width for the dark reference
  - `sample_boxcar`: Boxcar width for the sample reference

Examples

```
lr_parse_jaz(system.file("testdata", "jazspec.jaz", package = "lightr"))
lr_parse_jazirrad(system.file("testdata", "irrad.JazIrrad",
                  package = "lightr"))
```

`lr_parse_jdx` *Parse OceanInsight JCAMP-DX (.jdx) file*

Description

Parse OceanInsight (formerly OceanOptics) JCAMP-DX (.jdx) file. https://www.oceaninsight.com/

Usage

```
lr_parse_jdx(filename)
```

Arguments

`filename`	Path of the file to parse

Details

'processed' column computed by lightr with the function `compute_processed()`.

Value

A list of two elements:

- a dataframe with columns "wl", "dark", "white", "scope" and "processed", in that order
- a character vector with metadata including:
  - `user`: Name of the spectrometer operator
  - `date`: Timestamp of the recording (ISO 8601 format)
  - `spec_model`: Model of the spectrometer
  - `spec_ID`: Unique ID of the spectrometer
  - `white_inttime`: Integration time of the white reference (in ms)
  - `dark_inttime`: Integration time of the dark reference (in ms)
  - `sample_inttime`: Integration time of the sample (in ms)
  - `white_avgs`: Number of averaged measurements for the white reference
  - `dark_avgs`: Number of averaged measurements for the dark reference
  - `sample_avgs`: Number of averaged measurements for the sample
  - `white_boxcar`: Boxcar width for the white reference
  - `dark_boxcar`: Boxcar width for the dark reference
  - `sample_boxcar`: Boxcar width for the sample reference

References

McDonald RS, Wilks PA. JCAMP-DX: A Standard Form for Exchange of Infrared Spectra in Computer Readable Form. Applied Spectroscopy. 1988;42(1):151-62.

Examples

```
lr_parse_jdx(system.file("testdata", "OceanOptics_period.jdx",
                         package = "lightr"))
```

`lr_parse_procspec` *Parse OceanInsight ProcSpec file*

Description

Parse OceanInsight (formerly OceanOptics) ProcSpec file. https://www.oceaninsight.com/

Usage

```
lr_parse_procspec(filename)
```

Arguments

`filename`	Path of the file to parse

Details

'processed' column computed by official software and provided as is.

Value

A list of two elements:

- a dataframe with columns "wl", "dark", "white", "scope" and "processed", in that order
- a character vector with metadata including:
  - `user`: Name of the spectrometer operator
  - `date`: Timestamp of the recording (ISO 8601 format)
  - `spec_model`: Model of the spectrometer
  - `spec_ID`: Unique ID of the spectrometer
  - `white_inttime`: Integration time of the white reference (in ms)
  - `dark_inttime`: Integration time of the dark reference (in ms)
  - `sample_inttime`: Integration time of the sample (in ms)
  - `white_avgs`: Number of averaged measurements for the white reference
  - `dark_avgs`: Number of averaged measurements for the dark reference
  - `sample_avgs`: Number of averaged measurements for the sample
  - `white_boxcar`: Boxcar width for the white reference
  - `dark_boxcar`: Boxcar width for the dark reference
  - `sample_boxcar`: Boxcar width for the sample reference

References

https://www.oceaninsight.com/support/faqs/software/

Examples

```
lr_parse_procspec(system.file("testdata", "procspec_files",
                              "OceanOptics_Linux.ProcSpec",
                              package = "lightr"))
```

`lr_parse_spc` *Parse SPC binary file*

Description

Parse SPC binary file. (Used by CRAIC http://www.microspectra.com and OceanInsight https://www.oceaninsight.com/)

Usage

```
lr_parse_spc(filename)
```

Arguments

`filename`	Path of the file to parse

Details

'processed' column computed by official software and provided as is.

Value

A list of two elements:

- a dataframe with columns "wl", "dark", "white", "scope" and "processed", in that order
- a character vector with metadata including:
  - `user`: Name of the spectrometer operator
  - `date`: Timestamp of the recording (ISO 8601 format)
  - `spec_model`: Model of the spectrometer
  - `spec_ID`: Unique ID of the spectrometer
  - `white_inttime`: Integration time of the white reference (in ms)
  - `dark_inttime`: Integration time of the dark reference (in ms)
  - `sample_inttime`: Integration time of the sample (in ms)
  - `white_avgs`: Number of averaged measurements for the white reference
  - `dark_avgs`: Number of averaged measurements for the dark reference
  - `sample_avgs`: Number of averaged measurements for the sample
  - `white_boxcar`: Boxcar width for the white reference
  - `dark_boxcar`: Boxcar width for the dark reference
  - `sample_boxcar`: Boxcar width for the sample reference

In development

Metadata parsing has not yet been implemented for this file format.

Examples

```
lr_parse_spc(system.file("testdata", "compare", "CRAIC", "CRAIC.spc",
                         package = "lightr"))
```

`lr_parse_trm` *Parse Avantes binary file*

Description

Parse Avantes binary file (TRM, ABS, ROH, DRK, REF file extensions). https://www.avantes.com/products/spectrometers

Usage

```
lr_parse_trm(filename)

lr_parse_abs(filename)

lr_parse_roh(filename)

lr_parse_rfl8(filename, specnum = 1L)

lr_parse_raw8(filename, specnum = 1L)
```

Arguments

`filename`	Path of the file to parse
`specnum`	Integer representing the position of the spectrum to read in the file. This option only makes sense for AvaSoft8 files and is ignored in the other cases.

Details

'processed' column computed by lightr with the function `compute_processed()`.

Value

A list of two elements:

- a dataframe with columns "wl", "dark", "white", "scope" and "processed", in that order
- a character vector with metadata including:
  - `user`: Name of the spectrometer operator
  - `date`: Timestamp of the recording (ISO 8601 format)
  - `spec_model`: Model of the spectrometer
  - `spec_ID`: Unique ID of the spectrometer
  - `white_inttime`: Integration time of the white reference (in ms)

- dark_inttime: Integration time of the dark reference (in ms)
- sample_inttime: Integration time of the sample (in ms)
- white_avgs: Number of averaged measurements for the white reference
- dark_avgs: Number of averaged measurements for the dark reference
- sample_avgs: Number of averaged measurements for the sample
- white_boxcar: Boxcar width for the white reference
- dark_boxcar: Boxcar width for the dark reference
- sample_boxcar: Boxcar width for the sample reference

Examples

```
lr_parse_trm(system.file("testdata", "avantes_trans.TRM",
                         package = "lightr"))
lr_parse_roh(system.file("testdata", "avantes_reflect.ROH",
                         package = "lightr"))

lr_parse_rfl8(system.file("testdata", "compare", "Avantes",
                          "feather.RFL8",
                          package = "lightr"),
              specnum = 1)
lr_parse_rfl8(system.file("testdata", "compare", "Avantes",
                          "feather.RFL8",
                          package = "lightr"),
              specnum = 2)
```

---

lr_parse_ttt *Parse Avantes converted file*

Description

Parse Avantes converted file. https://www.avantes.com/products/spectrometers

Usage

```
lr_parse_ttt(filename)

lr_parse_trt(filename)
```

Arguments

filename	Path of the file to parse

Details

'processed' column computed by official software and provided as is.

Value

A list of two elements:

- a dataframe with columns "wl", "dark", "white", "scope" and "processed", in that order
- a character vector with metadata including:
  - `user`: Name of the spectrometer operator
  - `date`: Timestamp of the recording (ISO 8601 format)
  - `spec_model`: Model of the spectrometer
  - `spec_ID`: Unique ID of the spectrometer
  - `white_inttime`: Integration time of the white reference (in ms)
  - `dark_inttime`: Integration time of the dark reference (in ms)
  - `sample_inttime`: Integration time of the sample (in ms)
  - `white_avgs`: Number of averaged measurements for the white reference
  - `dark_avgs`: Number of averaged measurements for the dark reference
  - `sample_avgs`: Number of averaged measurements for the sample
  - `white_boxcar`: Boxcar width for the white reference
  - `dark_boxcar`: Boxcar width for the dark reference
  - `sample_boxcar`: Boxcar width for the sample reference

Examples

```
lr_parse_ttt(system.file("testdata", "avantes_export.ttt",
                         package = "lightr"))
lr_parse_trt(system.file("testdata", "avantes_export2.trt",
                         package = "lightr"))
```

Index