Package ‘OpenSpecy’

October 12, 2022

Type Package
Title Analyze, Process, Identify, and Share Raman and (FT)IR Spectra
Version 0.9.5
Date 2022-07-06
Description Raman and (FT)IR spectral analysis tool for plastic particles and other environmental samples (Cowger et al. 2021, <doi:10.1021/acs.analchem.1c00123>). Supported features include reading spectral data files (.asp, .csv, .jdx, .spc, .spa, .0), Savitzky-Golay smoothing of spectral intensities with smooth_intens(), correcting background noise with subtr_bg() in accordance with Zhao et al. (2007) <doi:10.1366/000370207782597003>, and identifying spectra using an onboard reference library (Cowger et al. 2020, <doi:10.1177/0003702820929064>). Analyzed spectra can be shared with the Open Specy community. A Shiny app is available via run_app() or online at <https://openanalysis.org/openspecy/>.

URL https://github.com/wincowgerDEV/OpenSpecy-package/
BugReports https://github.com/wincowgerDEV/OpenSpecy-package/issues/
License CC BY 4.0
Encoding UTF-8
LazyLoad true
LazyData true
VignetteBuilder knitr
Depends R (>= 4.0.0)
Imports dplyr, rlang, osfr, hyperSpec, hexView, digest, signal, shiny
Suggests knitr, rmarkdown, testthat (>= 3.0.0), config, shinyjs, shinythemes, shinyBS, shinyWidgets, ggplot2, plotly, data.table, DT, curl, rdrop2, mongolite, loggit
RoxygenNote 7.2.0
Config/testthat/edition 3
NeedsCompilation no
adj_intens

Description

Converts reflectance or transmittance intensity units to absorbance units.

Usage

adj_intens(x, ...)

## S3 method for class 'formula'
adj_intens(formula, data = NULL, ...)
adj_intens

## S3 method for class 'data.frame'
adj_intens(x, ...)

## Default S3 method:
adj_intens(x, y, type = "none", make_rel = TRUE, ...)

Arguments

- **x**: a numeric vector containing the spectral wavenumbers; alternatively a data frame containing spectral data as "wavenumber" and "intensity" can be supplied.
- **formula**: an object of class 'formula' of the form intensity ~ wavenumber.
- **data**: a data frame containing the variables in formula.
- **y**: a numeric vector containing the spectral intensities.
- **type**: a character string specifying whether the input spectrum is in absorbance units ("none", default) or needs additional conversion from "reflectance" or "transmittance" data.
- **make_rel**: logical; if TRUE spectra are automatically normalized with make_rel().
- **...**: further arguments passed to the submethods.

Details

Many of the Open Specy functions will assume that the spectrum is in absorbance units. For example, see match_spec() and subtr_bg(). To run those functions properly, you will need to first convert any spectra from transmittance or reflectance to absorbance using this function. The transmittance adjustment uses the \( \log_{10}(1/T) \) calculation which does not correct for system and particle characteristics. The reflectance adjustment uses the Kubelka-Munk equation \((1 - R)^2/2R\). We assume that the reflectance intensity is a percent from 1-100 and first correct the intensity by dividing by 100 so that it fits the form expected by the equation.

Value

adj_intens() returns a data frame containing two columns named "wavenumber" and "intensity".

Author(s)

Win Cowger, Zacharias Steinmetz

See Also

subtr_bg() for spectral background correction; match_spec() matches spectra with the Open Specy or other reference libraries

Examples

data("raman_hdpe")
adj_intens(raman_hdpe)
adj_neg  

Normalization of spectral data

Description

adj_neg() converts numeric values \( x < 1 \) into values \( \geq 1 \), keeping absolute differences between values by shifting intensity values with the value of the smallest number. make_rel() converts values \( x \) into relative values between 0 and 1 using the standard normalization equation. If na.rm is TRUE, missing values are removed before the computation proceeds.

Usage

adj_neg(x, na.rm = FALSE)

make_rel(x, na.rm = FALSE)

Arguments

x a numeric vector or an R object which is coercible to one by as.vector(x, "numeric"); \( x \) should be intensity data.

na.rm logical. Should missing values be removed?

Details

adj_neg() is used in Open Specy to avoid errors that could arise from log transforming spectra when using adj_intens() and other functions. make_rel() is used in Open Specy to retain the relative height proportions between spectra while avoiding the large numbers that can result from some spectral instruments.

Value

adj_neg() and make_rel() return numeric vectors with the normalized data.

Author(s)

Win Cowger, Zacharias Steinmetz

See Also

min() for the calculation of minima; adj_intens() for log transformation functions

Examples

adj_neg(c(-1000, -1, 0, 1, 10))

make_rel(c(-1000, -1, 0, 1, 10))
Description

These functions will import the spectral libraries from Open Specy if they were not already downloaded. The CRAN does not allow for deployment of large datasets so this was a workaround that we are using to make sure everyone can easily get Open Specy functionality running on their desktop.

Usage

```r
check_lib(
    which = c("ftir", "raman"),
    types = c("metadata", "library", "peaks"),
    path = "system",
    condition = "warning"
)

get_lib(
    which = c("ftir", "raman"),
    types = c("metadata", "library", "peaks"),
    path = "system",
    node = "x7dpz",
    conflicts = "overwrite",
    ...
)

load_lib(
    which = c("ftir", "raman"),
    types = c("metadata", "library", "peaks"),
    path = "system"
)
```

Arguments

- **which**: a character string specifying which library to use, "raman" or "ftir".
- **types**: library types to check/retrieve; defaults to c("metadata", "library", "peaks").
- **path**: where to save or look for local library files; defaults to "system" pointing to system.file("extdata", package = "OpenSpecy").
- **condition**: determines if check_lib() should warn ("warning", the default) or throw and error ("error").
- **node**: the OSF node to be retrieved; should be "x7dpz".
- **conflicts**: determines what happens when a file with the same name exists at the specified destination. Can be one of the following (see osf_download() for details):
check_lib

- "error" throw an error and abort the file transfer operation.
- "skip" skip the conflicting file(s) and continue transferring the remaining files.
- "overwrite" (default) replace the existing file with the transferred copy.

... further arguments passed to osf_download().

Details

check_lib() checks to see if the Open Specy reference library already exists on the users computer. get_lib() downloads the Open Specy library from OSF (doi:10.17605/OSF.IO/X7DPZ). load_lib() will load the library into the global environment for use with the Open Specy functions.

Value

check_lib() and get_lib() return messages only; load_lib() returns a list object containing the respective spectral reference library.

Author(s)

Zacharias Steinmetz

References


See Also

match_spec()

Examples

```r
## Not run:
check_lib(which = c("ftir", "raman"))
get_lib(which = c("ftir", "raman"))

spec_lib <- load_lib(which = c("ftir", "raman"))

## End(Not run)
```
human_ts

Create human readable timestamps

Description
This helper function creates human readable timestamps in the form of \%Y\%m\%d-\%H\%M\%S at the current time.

Usage
human_ts()

Details
Human readable timestamps are appended to file names and fields when metadata are shared with the Open Specy community.

Value
human_ts() returns a character value with the respective timestamp.

Author(s)
Win Cowger

See Also
format.Date for date conversion functions

Examples
human_ts()
Usage

match_spec(x, ...)

## S3 method for class 'formula'
match_spec(formula, data = NULL, ...)

## S3 method for class 'data.frame'
match_spec(x, ...)

## Default S3 method:
match_spec(
  x,
  y,
  library,
  which = NULL,
  type = "full",
  range = seq(0, 6000, 0.1),
  top_n = 100,
  ...
)

find_spec(
  subset,
  library,
  which = NULL,
  type = "metadata",
  ...
)

Arguments

x a numeric vector containing the spectral wavenumbers; alternatively a data frame containing spectral data as "wavenumber" and "intensity" can be supplied.

formula an object of class 'formula' of the form intensity ~ wavenumber.

data a data frame containing the variables in formula.

y a numeric vector containing the spectral intensities.

library reference library you want to compare against.

which a character string specifying which library to match, "raman" or "ftir".
**match_spec**

- **type**: a character string specifying whether the "full" spectrum should be matched or spectrum "peaks" only. "metadata" is needed to browser spectra with `find_spec()`.
- **range**: this should be all possible wavenumber values from your spectral library.
- **top_n**: number of top matches that you want to be returned.
- **subset**: logical expression indicating elements or rows to search for; see `subset()` for details.
- **cols**: columns to retrieve from the Open Specy reference library; columns containing no or missing values are automatically removed.
- **...**: further arguments passed to the submethods.

### Details

This routine will match the spectrum you want to identify to the wavenumbers present in the spectral library. Once the spectra are aligned, it computes the Pearson correlation coefficient between the spectrum you want to identify and all spectra in the library (see `cor`). The function returns a table with the Pearson correlation coefficient values and all metadata for the top spectral matches. If using the Open Specy library, all intensity values are in absorbance, so your spectra should also be in absorbance units. If you need to convert your spectrum, use `adj_intens()`.

### Value

`match_spec()` returns a data frame with the top_n material matches, their Pearson's r value, and the organization they were provided by. `find_spec()` returns a data frame with the spectral raw data or metadata of a specific reference spectrum.

### Author(s)

Win Cowger, Zacharias Steinmetz

### See Also

- `adj_intens()` converts spectra; `get_lib()` retrieves the Open Specy reference library; `load_lib()` loads the Open Specy reference library into an R object of choice.

### Examples

```r
## Not run:
data("raman_hdpe")

get_lib("raman")
spec_lib <- load_lib("raman")

match_spec(raman_proc, library = spec_lib, which = "raman")

find_spec(sample_name == 5381, library = spec_lib, which = "raman")
```

## End(Not run)
Sample Raman spectrum

Description

Raman spectrum of high-density polyethylene (HDPE).

Format

A data table containing 964 rows and 2 columns:

- wavenumber: spectral wavenumber [1/cm]
- intensity: absorbance values [-]

Author(s)

Win Cowger

References


Examples

data(“raman_hdpe”)

Read spectral data

Description

Functions for reading spectral data types including .asp, .jdx, .spc, .spa, .0, and .csv.

Usage

read_text(
  file = ".",
  cols = NULL,
  method = “read.csv”,
  share = NULL,
  id = paste(digest(Sys.info()), digest(sessionInfo()), sep = “/”),
)
...  
  
read_asp(
    file = ".",
    share = NULL,
    id = paste(digest(Sys.info()), digest(sessionInfo()), sep = "/"),
    ...  
)

read_spa(
    file = ".",
    share = NULL,
    id = paste(digest(Sys.info()), digest(sessionInfo()), sep = "/"),
    ...  
)

read_jdx(
    file = ".",
    share = NULL,
    id = paste(digest(Sys.info()), digest(sessionInfo()), sep = "/"),
    ...  
)

read_spc(
    file = ".",
    share = NULL,
    id = paste(digest(Sys.info()), digest(sessionInfo()), sep = "/"),
    ...  
)

read_0(
    file = ".",
    share = NULL,
    id = paste(digest(Sys.info()), digest(sessionInfo()), sep = "/"),
    ...  
)

read_exdata(file = NULL)

Arguments

file file to be read from.

cols character vector of length = 2 indicating the column names for the wavenumber and intensity; if NULL columns are guessed.

method submethod to be used for reading text files; defaults to read.csv but fread works as well.
run_app

run_app(app_dir = "system", path = "system", log = TRUE, ...)

Description

This wrapper function starts the graphical user interface of Open Specy.

Usage

run_app(app_dir = "system", path = "system", log = TRUE, ...)

Details

read_spc() and read_jdx() are just a wrapper around the functions provided by the hyperSpec package. Other functions have been adapted various online sources. All functions convert datasets to a 2 column table with one column labeled "wavenumber" and the other "intensity". There are many unique iterations of spectral file formats so there may be bugs in the file conversion. Please contact us if you identify any.

Value

All read_*() functions return data frames containing two columns named "wavenumber" and "intensity".

Author(s)

Zacharias Steinmetz, Win Cowger

See Also

read.jdx(); read.spc(); readRaw(); share_spec()

Examples

read_text(read_extdata("raman_hdpe.csv"))
read_asp(read_extdata("ftir_ldpe_soil.asp"))
read_0(read_extdata("ftir_ps.0"))
Arguments

app_dir  the app to run; defaults to "system" pointing to system.file("shiny", package = "OpenSpecy").
path     where to look for the local library files; defaults to "system" pointing to system.file("extdata", package = "OpenSpecy").
log      logical; enables/disables logging to tempdir()
...      arguments passed to runApp().

Details

After running this function the Open Specy GUI should open in a separate window or in your computer browser.

Value

This function normally does not return any value, see runApp().

Author(s)

Zacharias Steinmetz

See Also

runApp()

Examples

## Not run:
run_app()

## End(Not run)

---

**share_spec**

Share data with the Open Specy community

Description

This helper function shares spectral data and metadata with the Open Specy community.

Please note that share_spec() only provides basic sharing functionality if used interactively. This means that files are only formatted and saved for sharing but are not send automatically. This only works with hosted instances of Open Specy.
share_spec

Usage

share_spec(data, ...)

## Default S3 method:
share_spec(data, ...)

## S3 method for class 'data.frame'
share_spec(
  data,
  metadata = c(user_name = "", contact_info = "", organization = "", citation = ",
  spectrum_type = "", spectrum_identity = "", material_form = "", material_phase = "",
  material_producer = "", material_purity = "", material_quality = "", material_color =
  "", material_other = "", cas_number = "", instrument_used = "",
  instrument_accessories = "", instrument_mode = "", spectral_resolution = "",
  laser_light_used = "", number_of_accumulations = "", total_acquisition_time_s = "",
  data_processing_procedure = "", level_of_confidence_in_identification = "",
  other_info = "", license = "CC BY-NC"),
  file = NULL,
  share = "system",
  id = paste(digest(Sys.info()), digest(sessionInfo()), sep = "/"),
  ...
)

Arguments

data a data frame containing the spectral data; columns should be named "wavenumber" and "intensity".

metadata a named vector of the metadata to share; see details below.

file file to share (optional).

share accepts any local directory to save the spectrum for later sharing via e-mail to <wincowger@gmail.com>; "system" (default) uses the Open Specy package directory at system.file("extdata", package = "OpenSpecy"); if a correct API token exists, "dropbox" shares the spectrum with the cloud.

id a unique user and/or session ID; defaults to paste(digest(Sys.info()), digest(sessionInfo()), sep = "/").

Arguments passed to the submethods.

Details

The metadata argument may contain a named vector with the following details (* = mandatory):

user_name* User name, e.g. "Win Cowger"
contact_info Contact information, e.g. "1-513-673-8956, wincowger@gmail.com"
organization Affiliation, e.g. "University of California, Riverside"
citation Data citation, e.g. "Primpke, S., Wirth, M., Lorenz, C., & Gerdts, G. (2018). Analytical and Bioanalytical Chemistry. doi:10.1007/s002160181156x"
spectrum_type* Raman or FTIR
share_spec

spectrum_identity*: Material/polymer analyzed, e.g. "Polystyrene"
matterial_form: Form of the material analyzed, e.g. textile fiber, rubber band, sphere, granule
material_phase: Phase of the material analyzed (liquid, gas, solid)
materiaheel_producer: Producer of the material analyzed, e.g. Dow
material_purity: Purity of the material analyzed, e.g. 99.98%
materiaheel_quality: Quality of the material analyzed, e.g. consumer product, manufacturer material
material_color: Color of the material analyzed, e.g. blue, #0000ff, (0, 0, 255)
materiaheel_other: Other material description, e.g. 5 µm diameter fibers, 1 mm spherical particles
cas_number: CAS number, e.g. 9003-53-6
instrument_used: Instrument used, e.g. Horiba LabRam
instrument_accessories: Instrument accessories, e.g. Focal Plane Array, CCD
instrument_mode: Instrument modes/settings, e.g. transmission, reflectance
spectral_resolution: Spectral resolution, e.g. 4/cm
laser_light_used: Wavelength of the laser/light used, e.g. 785 nm
number_of_accumulations: Number of accumulations, e.g 5
total_acquisition_time_s: Total acquisition time (s), e.g. 10 s
data_processing_procedure: Data processing procedure, e.g. spikefilter, baseline correction, none
level_of_confidence_in_identification: Level of confidence in identification, e.g. 99%
other_info: Other information
license: The license of the shared spectrum; defaults to "CC BY-NC" (see https://creativecommons.org/licenses/by-nc/4.0/ for details). Any other creative commons license is allowed, for example, CC0 or CC BY

Value

share_spec() returns only messages/warnings.

Author(s)

Zacharias Steinmetz, Win Cowger

See Also

read_text(); digest(); sessionInfo()

Examples

## Not run:
data("raman_hdpe")
share_spec(raman_hdpe,
  metadata = c(user_name = "Win Cowger",
               spectrum_type = "FTIR",
               spectrum_identity = "PE",
               license = "CC BY-NC"),
  share = tempdir())

## End(Not run)
smooth_intens  Smooth spectral intensities

Description
This smoother can enhance the signal to noise ratio of the data and uses a Savitzky-Golay filter with a running window of data points and the polynomial specified.

Usage
smooth_intens(x, ...)

## S3 method for class 'formula'
smooth_intens(formula, data = NULL, ...)

## S3 method for class 'data.frame'
smooth_intens(x, ...)

## Default S3 method:
smooth_intens(x, y, p = 3, n = 11, make_rel = TRUE, ...)

Arguments

x  a numeric vector containing the spectral wavenumbers; alternatively a data frame containing spectral data as "wavenumber" and "intensity" can be supplied.

formula  an object of class 'formula' of the form intensity ~ wavenumber.

data  a data frame containing the variables in formula.

y  a numeric vector containing the spectral intensities.

p  polynomial order for the filter

n  number of data points in the window, filter length (must be odd).

make_rel  logical; if TRUE spectra are automatically normalized with make_rel().

...  further arguments passed to sgolay().

Details
This is a wrapper around the filter function in the signal package to improve integration with other Open Specy functions. A typical good smooth can be achieved with 11 data point window and a 3rd or 4th order polynomial.

Value
smooth_intens() returns a data frame containing two columns named "wavenumber" and "intensity".

Author(s)
Win Cowger, Zacharias Steinmetz
spec_res

References

See Also
s golay()

Examples
data("raman_hdpe")
smooth_intens(raman_hdpe)

<table>
<thead>
<tr>
<th>spec_res</th>
<th>Spectral resolution</th>
</tr>
</thead>
</table>

Description
Helper function for calculating the spectral resolution from wavenumber data.

Usage
spec_res(x)

Arguments
x a numeric vector or an R object which is coercible to one by as.vector(x, "numeric"); x should be wavenumber data.

Details
The spectral resolution is the minimum wavenumber, wavelength, or frequency difference between two lines in a spectrum that can still be distinguished.

Value
spec_res() returns a single numeric value.

Author(s)
Win Cowger, Zacharias Steinmetz

Examples
data("raman_hdpe")
spec_res(raman_hdpe$wavenumber)
Automated background subtraction for spectral data

Description

This baseline correction routine iteratively finds the baseline of a spectrum using a polynomial fitting.

Usage

`subtr_bg(x, ...)`

## S3 method for class 'formula'
`subtr_bg(formula, data = NULL, ...)`

## S3 method for class 'data.frame'
`subtr_bg(x, ...)`

## Default S3 method:
`subtr_bg(x, y, degree = 8, raw = FALSE, make_rel = TRUE, ...)`

Arguments

- `x` a numeric vector containing the spectral wavenumbers; alternatively a data frame containing spectral data as "wavenumber" and "intensity" can be supplied.
- `formula` an object of class 'formula' of the form intensity ~ wavenumber.
- `data` a data frame containing the variables in formula.
- `y` a numeric vector containing the spectral intensities.
- `degree` the degree of the polynomial. Must be less than the number of unique points when raw is FALSE. Typically a good fit can be found with a 8th order polynomial.
- `raw` if TRUE, use raw and not orthogonal polynomials.
- `make_rel` logical; if TRUE spectra are automatically normalized with `make_rel()`.
- `...` further arguments passed to `poly()`.

Details

This is a translation of Michael Stephen Chen’s MATLAB code written for the `imodpolyfit` routine from Zhao et al. 2007.

Value

`subtr_bg()` returns a data frame containing two columns named "wavenumber" and "intensity".
test_lib

Author(s)
Win Cowger, Zacharias Steinmetz

References

See Also
poly(); smooth_intens()

Examples
data("raman_hdpe")
subtr_bg(raman_hdpe)

test_lib Test reference library

Description
Reference library of 34 Raman spectra used for internal testing.

Format
A list named "test" with two elements:

- metadata: metadata of 34 Raman spectra
- library: all reference spectra, sample_name serves as identifier

Author(s)
Jennifer Lynch

Examples
data("test_lib")
Index

* data
  - raman_hdpe, 10
  - test_lib, 19

adj_intens, 2, 4, 9
adj_neg, 4

check_lib, 5
cor, 9

digest, 15

find_spec (match_spec), 7
format.Date, 7
formula, 3, 8, 16, 18
fread, 11

get_lib, 9
get_lib (check_lib), 5

human_ts, 7
hyperSpec, 12

load_lib, 9
load_lib (check_lib), 5

make_rel, 3, 16, 18
make_rel (adj_neg), 4
match_spec, 3, 6, 7
min, 4

osf_download, 5, 6

poly, 18, 19

raman_hdpe, 10
read.csv, 11
read.jdx, 12
read.spc, 12
read_0 (read_text), 10
read_asp (read_text), 10
read_extdata (read_text), 10
read_jdx (read_text), 10
read_spa (read_text), 10
read_spc (read_text), 10
read_text, 10, 15
readRaw, 12
run_app, 12
runApp, 13

sessionInfo, 15
gsogay, 16, 17
share_spec, 12, 13
smooth_intens, 16, 19
spec_res, 17
subset, 9
subtr_bg, 3, 18
tempdir, 13
test_lib, 19