Package: simplerspec (via r-universe)

February 18, 2025

Type Package

Version 0.2.1

Date 2023-09-27

Title Soil and plant spectroscopic model building and prediction

Description Functions that cover reading of spectral data, outlier removal, spectral preprocessing, calibration sampling, PLS regression using caret, and model diagnostic statistics and plots.

URL https://github.com/philipp-baumann/simplerspec

BugReports https://github.com/philipp-baumann/simplerspec

Depends R (>= 3.0), foreach

Imports caret, cowplot, data.table, dplyr (>= 0.7.0), e1071, ggplot2 (>= 2.0.0), hexView, magrittr, mvoutlier, modelr, pls, plyr, prospectr, purrr, reshape2, rlang (>= 0.2.0), stringr, tibble, tidyselect, tidyr, utils, glue, broom

License GPL-2

LazyData true

RoxygenNote 7.2.3

Roxygen list(markdown = TRUE)

Encoding UTF-8

Config/pak/sysreqs libicu-dev

Repository https://philipp-baumann.r-universe.dev

RemoteUrl https://github.com/philipp-baumann/simplerspec

RemoteRef HEAD

RemoteSha 60e120130ce83cfab34115bbfa369ea200b366b4

Contents

`assess_multimodels` *Assess multiple pairs of measured and predicted values*

Description

Return performance metrics for test set predictions and measured values, e.g. for different model outcome variables.

Usage

```
assess_multimodels(
  data,
  ...,
  .metrics = c("simplerspec", "yardstick"),
  .model_name = "model"
)
```

Arguments

data	Data frame with all measured (observed) and predicted variables.
...	Multiple arguments with observed (measured)-predicted pairs, specified with dplyr::vars(o = <column_name>, p = <column_name>). Column names can strings or symbols. The arguments in ... need to be named.
.metrics	Character vector with package used for metrics calculation. Default is "simplerspec", which uses simplerspec::evaluate_model().
.model_name	String with name for the new column that specifies the model or the outcome variable. Default is "model".

Value

Data frame with with summary statistics for measured values and performance metrics for the pairs of measured and predicted values.

---

average_spc *Average spectra in list-column by entries in grouping column*

Description

Average spectra in list-column of spectra tibble (spc_tbl) by groups given in group column.

Usage

```
average_spc(spc_tbl, by = "sample_id", column_in = "spc_rs")
```

Arguments

spc_tbl	Tibble data frame containing at least the grouping column given in argument by and input spectra given in list-column column_in.
by	Character vector of length 1L or name/symbol that specifies the column by which groups of spectra are averaged. Default is "sample_id".
column_in	Character vector of length 1L or or name/symbol that specifies the list-column that contains the inputs spectra to be averaged. Default is "spc_rs", which are resampled spectra (i.e., resulting after preceding resample_spc() step).

Details

For memory efficiency and subsequent modeling, consider slicing the extra row copies of `spc_mean` resulting from `average_spc()` for example by

- `split(x = spc_tbl, f = spc_tbl$<by>) %>% lapply(., function(x) x x[1, ]) %>% do.call(., rbind)`
- `dplyr::group_by(spc_tbl, <by>) %>% dplyr::slice(1L)`

Value

Spectra tibble data frame (class `"tbl_df"`, `"tbl"`, `"data.frame"`) with a new list-column of column name `"spc_mean"` at the last position, containing mean spectra with identical row replicates within the same by-group.

---

bind_lcols_dts *Bind list-columns within a tibble into a list of data.tables*

Description

Bind one to many list-columns in spectral tibble into a list of data.tables.

Usage

```
bind_lcols_dts(spc_tbl, lcols, spc_id = "unique_id", group_id = "sample_id")
```

Arguments

`spc_tbl`	Spectral data in a tibble data frame (classes "tibble_df", "tbl" and "data.frame").
`lcols`	Character vector of column names of list-columns to be bound into a list of data.tables
`spc_id`	Character vector denoting column name for a unique spectrum ID. Default is `"unique_id"`.
`group_id`	Character vector denoting column name for the spectrum group ID. Default is `"sample_id"`. The group ID can later be used for plotting spectra by group (e.g. by using different colors or panels).

Value

A list of data.tables. Elements contain data from list-columns specified in `lcols` argument as data.tables. All data.tables contain in addition `spc_id` and `group_id` columns.

`create_vip_rects` *Create a data frame containing start and end positions (wavenumbers) where variable importance in projection (VIP) is > 1*

Description

Given a data frame with VIP outputs (wavenumber and vip columns), start and end values denoting spectral regions where VIP > 1 are returned as data frame. The functions can be used as helper function for plotting VIP.

Usage

```
create_vip_rects(df_vip)
```

Arguments

`df_vip`	Data frame containing `wavenumber` and `vip` columns (numeric)

Value

Data.frame containing vectors `start` (numeric; wavenumber), `end` (numeric; wavenumber) and group (integer; values are `1:length(start)`).

`evaluate_model` *Calculate model evaluation metrics*

Description

Calculate summary statistics of observed values and model evaluation statistics for assessing agreement between observed (`obs`) and predicted (`pred`) values.

Usage

```
evaluate_model(data, obs, pred)

summary_df(df, x, y)
```

Arguments

`data`	`data.frame` with predicted and observed data in columns.
`obs`	Column that contains observed values, `symbol`/`name` or `character` (wrapped in "").
`pred`	Column that contains predicted values, `symbol`/`name` or `character` (wrapped in "").
`df`	`data.frame` with predicted and observed data in columns.

x	Column that contains observed values, symbol/name or character (wrapped in "").
y	Column that contains predicted values, symbol/name or character (wrapped in "").

`extract_lcols2dts` *Extract multiple tibble list-columns and return data as list of data.tables*

Description

Extract multiple tibble list columns, row bind them separately into single data tables and return a list of data.tables.

Usage

```
extract_lcols2dts(spc_tbl, lcols)
```

Arguments

spc_tbl	Spectral tibble (data frame) with spectral data contained in list-columns
lcols	Character vector containing names of list-columns to be extracted into a list of data.tables

Value

List of data.tables. Each element is a data.table derivied from a list-column specified in the `lcols` argument.

`extract_pls_vip` *Extract VIPs (variable importance in the projection) for a PLS regression model output returned from model fitting with* `simplerspec::fit_pls()`

Description

VIPs are extracted based on the `finalModel` sublist in the `caret::train` output contained in the `model` element of the `simplerspec::fit_pls()` model output list. The VIPs for derived number of PLS components in the `finalModel` are computed.

Usage

```
extract_pls_vip(mout)
```

Arguments

mout	Model output list returned from simplerspec::fit_pls().

Value

A tibble data frame with columns wavenumber and correponding VIP values in the column vip for the finally chosen PLS regression model at the final number of PLS components.

---

fit_pls *Calibration sampling, model tuning, and PLS regression*

Description

Perform calibration sampling and use selected calibration set for model tuning

Usage

```
fit_pls(
  spec_chem,
  response,
  variable = NULL,
  center = TRUE,
  scale = TRUE,
  evaluation_method = "test_set",
  validation = TRUE,
  split_method = "ken_stone",
  ratio_val = 1/3,
  ken_sto_pc = 2,
  pc,
  invert = TRUE,
  tuning_method = "resampling",
  resampling_method = "kfold_cv",
  cv = NULL,
  resampling_seed = 123,
  pls_ncomp_max = 20,
  ncomp_fixed = 5,
  print = TRUE,
  env = parent.frame()
)

pls_ken_stone(
  spec_chem,
  response,
  variable = NULL,
  center = TRUE,
  scale = TRUE,
  evaluation_method = "test_set",
```

```
  validation = TRUE,
  split_method = "ken_stone",
  ratio_val = 1/3,
  ken_sto_pc = 2,
  pc,
  invert = TRUE,
  tuning_method = "resampling",
  resampling_method = "kfold_cv",
  cv = NULL,
  resampling_seed = 123,
  pls_ncomp_max = 20,
  ncomp_fixed = 5,
  print = TRUE,
  env = parent.frame()
)
```

Arguments

spec_chem	Tibble that contains spectra, metadata and chemical reference as list-columns. The tibble to be supplied to spec_chem can be generated by the join_chem_spc() function
response	Response variable as symbol or name (without quotes, no character string). The provided response symbol needs to be a column name in the spec_chem tibble.
variable	Depreciated and replaced by response
center	Logical whether to perform mean centering of each spectrum column (e.g. wavenumber or wavelength) after common spectrum preprocessing. Default is center = TRUE
scale	Logical whether to perform standard deviation scaling of each spectrum column (e.g. wavenumber or wavelength) after common spectrum preprocessing. Default is scale = TRUE
evaluation_method	Character string stating evaluation method. Either "test_set" (default) or "resampling". "test_set" will split the data into a calibration (training) and validation (test) set, and evaluate the final model by predicting on the validation set. If "resampling", the finally selected model will be evaluated based on the cross-validation hold-out predictions.
validation	Depreciated and replaced by evaluation_method. Default is TRUE.
split_method	Method how to to split the data into a independent test set. Default is "ken_sto", which will select samples for calibration based on Kennard-Stone sampling algorithm of preprocessed spectra. The proportion of validation to the total number of samples can be specified in the argument ratio_val. split_method = "random" will create a single random split.
ratio_val	Ratio of validation (test) samples to total number of samples (calibration (training) and validation (test)).
ken_sto_pc	Number of component used for calculating mahalanobsis distance on PCA scores for computing Kennard-Stone algorithm. Default is ken_sto_pc = 2, which will use the first two PCA components.

pc	Depreciated; renamed argument is ken_sto_pc.
invert	Logical
tuning_method	Character specifying tuning method. Tuning method affects how caret selects a final tuning value set from a list of candidate values. Possible values are "resampling", which will use a specified resampling method such as repeated k-fold cross-validation (see argument resampling_method) and the generated performance profile based on the hold-out predictions to decide on the final tuning values that lead to optimal model performance. The value "none" will force caret to compute a final model for a predefined canditate PLS tuning parameter number of PLS components. In this case, the value supplied by ncomp_fixed‘ is used to set model complexity at a fixed number of components.
resampling_method	Character specifying resampling method. Currently, "kfold_cv" (default, performs 10-fold cross-validation), "rep_kfold_cv" (performs 5-times repeated 10-fold cross-validation), "loocv" (performs leave-one-out cross-validation), and "none" (if resampling_method = "none") are supported.
cv	Depreciated. Use resampling_method instead.
resampling_seed	Random seed (integer) that will be used for generating resampling indices, which will be supplied to caret::trainControl. This makes sure that modeling results are constant when re-fitting. Default is resampling_seed = 123.
pls_ncomp_max	Maximum number of PLS components that are evaluated by caret::train. Caret will aggregate a performance profile using resampling for an integer sequence from 1 to pls_ncomp_max
ncomp_fixed	Integer of fixed number of PLS components. Will only be used when tuning_method = "none" and resampling_method = "none" are used.
print	Logical expression whether model evaluation graphs shall be printed
env	Environment where function is evaluated. Default is parent.frame.

`fit_rf` *Calibration sampling, and random forest model tuning and evaluation*

Description

Perform calibration sampling and use selected calibration set for model tuning

Usage

```
fit_rf(
  spec_chem,
  response,
  variable = NULL,
  evaluation_method = "test_set",
  validation = NULL,
```

```
  split_method = "ken_stone",
  ratio_val,
  ken_sto_pc = 2,
  pc = NULL,
  invert = TRUE,
  tuning_method = "resampling",
  resampling_seed = 123,
  cv = NULL,
  ntree_max = 500,
  print = TRUE,
  env = parent.frame()
)
```

Arguments

spec_chem	Tibble that contains spectra, metadata and chemical reference as list-columns. The tibble to be supplied to spec_chem can be generated by the join_chem_spc() function
response	Response variable as symbol or name (without quotes, no character string). The provided response symbol needs to be a column name in the spec_chem tibble.
variable	Depreciated and replaced by response
evaluation_method	Character string stating evaluation method. Either "test_set" (default) or "resampling". "test_set" will split the data into a calibration (training) and validation (test) set, and evaluate the final model by predicting on the validation set. If "resampling", the finally selected model will be evaluated based on the cross-validation hold-out predictions.
validation	Depreciated and replaced by evaluation_method. Default is TRUE.
split_method	Method how to to split the data into a independent test set. Default is "ken_sto", which will select samples for calibration based on Kennard-Stone sampling algorithm of preprocessed spectra. The proportion of validation to the total number of samples can be specified in the argument ratio_val. split_method = "random" will create a single random split.
ratio_val	Ratio of validation (test) samples to total number of samples (calibration (training) and validation (test)).
ken_sto_pc	Number of component used for calculating mahalanobsis distance on PCA scores for computing Kennard-Stone algorithm. Default is ken_sto_pc = 2, which will use the first two PCA components.
pc	Depreciated; renamed argument is ken_sto_pc.
invert	Logical
tuning_method	Character specifying tuning method. Tuning method affects how caret selects a final tuning value set from a list of candidate values. Possible values are "resampling", which will use a specified resampling method such as repeated k-fold cross-validation (see argument resampling_method) and the generated performance profile based on the hold-out predictions to decide on the final tuning values that lead to optimal model performance. The value "none" will force caret to compute a final model for a predefined canditate PLS tuning parameter

number of PLS components. In this case, the value supplied by `ncomp_fixed`‘ is used to set model complexity at a fixed number of components.

`resampling_seed`
: Random seed (integer) that will be used for generating resampling indices, which will be supplied to `caret::trainControl`. This makes sure that modeling results are constant when re-fitting. Default is `resampling_seed = 123`.

`cv`
: Depreciated. Use `resampling_method` instead.

`ntree_max`
: Maximum random forest trees by caret::train. Caret will aggregate a performance profile using resampling for an integer sequence from 1 to `ntree_max` trees.

`print`
: Logical expression whether model evaluation graphs shall be printed

`env`
: Environment where function is evaluated. Default is `parent.frame`.

`gather_spc` *Gather measurements of different spectra types, corresponding x-axis values and metadata from nested list.*

Description

Gather spectra, corresponding x-axis values, and device and measurement metadata from a nested list into a spectra tibble, so that one row represents one spectral measurement. Spectra, x-axis values and metadata are mapped from the individual list elements (named after file name including the extension) and transformed into (list-)columns of a spectra tibble, which is an extended data frame. For each measurement, spectral data and metadata are combined into one row of the tidy data frame. In addition, the ID columns `unique_id`, `file_id`, and `sample_id` are extracted from `"metadata"` (data frame) list entries and returned as identifier columns of the spectra tibble. List-columns facilitate keeping related data together in a rectangular data structure. They can be manipulated easily during subsequent transformations, for example using the standardized functions of the simplerspec data processing pipeline.

Usage

```
gather_spc(data, spc_types = "spc")
```

Arguments

`data`
: Recursive list named with filename (`file_id`) at first level entries, where each element containing a sample measurement has nested metadata (`"metadata"`), spectra types (see `spc_types`), corresponding x-axis values (see section *"Details on spectra data checks and matching"*). The `data` list is a structural convention to organize spectra and their metadata. It follows for example the list structure returned from the Bruker OPUS binary reader `simplerspec::read_opus_univ()`.

`spc_types`
: Character vector with the spectra types to be extracted from `data` list and gathered into list-columns. The spectra type names need to exactly follow the naming conventions, and the element names and contents need to be present at the second list hierarchy of `data`. These values are allowed:

- "spc" (default): final raw spectra after atmospheric compensation, if performed (named AB in Bruker OPUS software; results from referencing sample to reference single channel reflectance and transforming to absorbance).
- "spc_nocomp": raw spectra without atmospheric correction
- "sc_sm": Single channel reflectance spectra of the samples
- "sc_rf": Single channel reflectance spectra of the reference (background spectra)
- "ig_sm": Interferograms of the sample spectra (currently only spectra without x-axis list-columns are matched and returned)
- "ig_rf": Interferograms of the reference spectra (currently only spectra without x-axis list-columns are matched and returned)

Value

Spectra tibble (spc_tbl with classes "tbl_df", "tbl", and "data.frame") with the following (list-)columns:

- "unique_id": Character vector with unique measurement identifier, likely a string with file names in combination with date and time (extracted from each "metadata" data frame column).
- "file_id" : Character vector with file name including the extension (extracted from each "metadata" data frame column).
- "sample_id": Character vector with sample identifier. For Bruker OPUS binary files, this corresponds to the file name without the file extension in integer increments of sample replicate measurements.
- One or multiple of "spc", "spc_nocomp", "sc_sm", or "sc_rf": List(s) of data.table's containing spectra type(s).
- One or multiple of "wavenumbers", "wavelengths", "x_values", "wavenumbers_sc_sm", "wavelengths_sc_sm", "x_values_sc_sm", "wavenumbers_sc_rf", "wavelengths_sc_rf", or "x_values_sc_rf": List(s) of numeric vectors with matched x-axis values (see *"Details on spectra data checks and matching"* below).

Details on spectra data checks and matching

gather_spc() checks whether these conditions are met for each measurement in the list data:

1. Make sure that the first level data elements are named (assumed to be the file name the data originate from), and remove missing measurements with an informative message.
2. Remove any duplicated file names and raise a message if there are name duplicates at first level.
3. Check whether spc_types inputs are supported (see argument spc_types) and present at the second level of the data list. If not, remove all data elements for incomplete spectral measurements.
4. Match spectra types and possible corresponding x-axis types from a lookup list. For each selected spectrum type (left), at least one of the element names of the x-axis type (right) needs to be present for each measurement in the list data:
   - "spc" : "wavenumbers", "wavelengths", or "x_values"

- "spc_nocomp" : "wavenumbers", "wavelengths", or "x_values"
- "sc_sm" : "wavenumbers_sc_sm", "wavelengths_sc_sm", or "x_values_sc_sm"
- "sc_rf" : "wavenumbers_sc_rf", "wavelengths_sc_rf", or "x_values_sc_rf"

5. Check if "metadata" elements are present and remove data elements for measurements with missing or incorrectly named metadata elements (message).

`join_chem_spec` *Join chemical and spectral data frames*

Description

Combines spectral data (data.frame) and chemical data (data.frame).

Usage

```
join_chem_spec(dat_chem, dat_spec, by = "sample_ID")
```

Arguments

dat_chem	data.frame that contains chemical values of the sample
dat_spec	List that contains spectral data
by	character of column name that defines sample_ID

Value

List: xxx

`join_spc_chem` *Join spectra data and chemical data tibbles*

Description

Combines spectral data (tibble class) and chemical data (tibble class).

Usage

```
join_spc_chem(spc_tbl, chem_tbl, by = "sample_id")
```

Arguments

spc_tbl	Tibble that contains spectral data
chem_tbl	Tibble that contains chemical reference values of the samples
by	character of column name that defines sample_ID

Value

Tibble joined by sample_id

`merge_dts` *Merge list-columns of spectra, x-axis values, metadata and additional measured variables into a single long form data.table*

Description

Helper function that merges all spectra and related data into a single long form data.table than can subsequently be used for plotting.

Usage

```
merge_dts(
  spc_tbl,
  lcols_spc = c("spc", "spc_pre"),
  lcol_measure = NULL,
  spc_id = "unique_id",
  group_id = "sample_id"
)
```

Arguments

`spc_tbl`	Tibble data frame containing spectra, x-axis values, metadata and eventual measured variables as list-columns.
`lcols_spc`	Character vector of spectral list-columns to be extracted. Default is `c("spc", "spc_pre")` (raw and preprocessed spectra).
`lcol_measure`	Character vector of length 1 denoting the column name of the measure columns. This argument is optional. Default is `NULL`, which does not extract an additional measure column.
`spc_id`	Character vector of column that contains a unique spectral identifier for all spectra. Default is `"unique_id"`.
`group_id`	Character vector of columns that is used assigning spectra into groups. Default is `"sample_id"`. The `group_id` can be used for later plotting and thereby visually separating spectral groups into using different colors or panels.

Value

A single data.table containing long form aggregated data of spectra, x-axis values, metadata and an additionally measured variable.

`merge_dts_l` *Wrapper function around* `merge_dts()` *for list of tibbles to aggregate data for plotting.*

Description

Instead of a single spectral tibble (data frame) multiple spectral tibbles can be merged into a long-form data.table for plotting spectra and related data. For details, see `merge_dts`.

Usage

```
merge_dts_l(
  spc_tbl_l,
  lcols_spc = c("spc", "spc_pre"),
  lcol_measure = NULL,
  spc_id = "unique_id",
  group_id = "sample_id"
)
```

Arguments

`spc_tbl_l`	List of spectral tibbles (data frames).
`lcols_spc`	Character vector of spectral list-columns to be extracted. Default is `c("spc", "spc_pre")` (raw and preprocessed spectra).
`lcol_measure`	Character vector of length 1 denoting the column name of the measure columns. This argument is optional. Default is `NULL`, which does not extract an additional measure column.
`spc_id`	Character vector of column that contains a unique spectral identifier for all spectra. Default is `"unique_id"`.
`group_id`	Character vector of columns that is used assigning spectra into groups. Default is `"sample_id"`. The `group_id` can be used for later plotting and thereby visually separating spectral groups into using different colors or panels.

Value

A single data.table containing long form aggregated data of spectra, x-axis values, metadata and an additionally measured variable. An additional column called `group_id_tbl` is appended. It denotes the name of the spectral tibble supplied with the list `spc_tbl_l`.

plot_pls_vip *Plot stacked ggplot2 graphs with the Variable Importance for the Projection (VIP) scores, mean replicate spectra (absorbance) per sample_id, and the preprocessed spectra.*

Description

Plot stacked ggplot2 graphs of VIP for the final PLS regression model output of the calibration (training) data set for the final number of components, raw (replicate mean) spectra, and preprocessed spectra. Regions with VIP > 1 are highlighted across the stacked graphs in beige colour rectangles. VIP calculation is implemented as described in Chong, I.-G., and Jun, C.-H. (2005). Performance of some variable selection methods when multicollinearity is present. Chemometrics and Intelligent Laboratory Systems, 78(1–2), 103–112. https://doi.org/10.1016/j.chemolab.2004.12.011

Usage

```
plot_pls_vip(mout, y1 = "spc_mean", y2 = "spc_pre",
  by = "sample_id",
  xlab = expression(paste("Wavenumber [", cm^-1, "]")),
  ylab1 = "Absorbance", ylab2 = "Preprocessed Abs.",
  alpha = 0.2)
```

Arguments

mout	Model output list that is returned from `simplerspec::fit_pls()`. This object contains a nested list with the `caret::train()` object (class `train`), based on which VIPs at finally selected number of PLS components are computed.
y1	Character vector of list-column name in `mout$data$calibration`, where spectra for bottom graph are extracted. Default is `"spc_mean"`, which plots the mean calibration spectra after resampling.
y2	Character string of list-column name in `mout$data$calibration`, where spectra for bottom graph are extracted. Default is `"spc_pre"`, which plots the preprocessed calibration spectra after resampling.
by	Character string that is used to assign spectra to the same group and therefore ensures that all spectra are plotted with the same colour. Default is `"sample_id"`
xlab	Character string of X axis title for shared x axis of stacked graphs. Default is `expression(paste("Wavenumber [", cm^-1, "]"))`
ylab1	Y axis title of bottom spectrum. Default is `"Absorbance"`.
ylab2	Y axis title of bottom spectrum. Default is `"Preprocessed Abs."`.
alpha	Double between 0 and 1 that defines transparency of spectra lines in returned graph (ggplot plot object).

plot_spc *Plot tibble spectra*

Description

Plot spectra from tibble spectra objects.

Usage

```
plot_spc(spc_tbl, spc_tbl_2 = NULL,
  x_unit = "wavenumber",
  y = "spc", by = "unique_id",
  graph_id_1 = "Set 1", graph_id_2 = "Set 2",
  graph_id_1_col = "black", graph_id_2_col = "red",
  xlab = expression(paste("Wavenumber [", cm^-1, "]")),
  ylab = "Absorbance",
  alpha = 0.2, legend = TRUE)
```

Arguments

spc_tbl	Tibble that contains the first set of spectra to plot as list-column
spc_tbl_2	Tibble that contains the second set of spectra (optional) to plot as list-column.
x_unit	Character string describing the x axis unit. Default is "wavenumber", which will produce a graph with wavenumbers on the x axis with reversed number. If x_unit = "wavelength", the axis will be in regular order (lower wavelengths in nm on the left and higher on the right side of the axis).
y	Character string of list-column name in tibble where spectra of desired type are extracted to plot.
by	Character string of column that is used to group the spectra. Default is "unique_id". If replica spectra are present in the file and processed spectra resulting after averaging need to be plotted, it is recommend to use "sample_id" as argument to group according the sample_id column in the tibble(s) containing the spectra (spc_tbl and spc_tbl_2).
graph_id_1	Character string used for grouping the first spectra set (spc_tbl) and producing the label text accordingly. Default is "Set 1".
graph_id_2	Character string used for grouping the second spectra set (spc_tbl_2) and producing the label text accordingly. Default is "Set 2"
graph_id_1_col	Character string for the colour of the first spectra set. Default is "black".
graph_id_2_col	Character string for the colour of the first spectra set. Default is "red".
xlab	Character string or mathematical expression (use expression) for the x axis title. Default is expression(paste("Wavenumber [", cm^-1, "]")).
ylab	Character string or mathematical expression (use expression) for the y axis title. Default is "absorbance".
alpha	Double in between 0 and 1. Sets the transparency for the plotted spectra lines.

legend	Logical whether to plot a legend for the spectra describing its name selected in arguments graph_id_1 and graph_id_2. Default is TRUE.

---

plot_spc_ext *ggplot2 wrapper for extended spectra plotting*

Description

plot_spc_ext is a custom plotting function developed within the simplerspec framework. Returns plots based on ggplot2 (class "ggplot"). Different spectra types such as raw or preprocessed spectra and groups can be differentiated by different colors or by using panels (so called facets). Additionally, spectra can be colored based on an additional measure variable, e.g. determined by chemical reference analysis.

Usage

```
plot_spc_ext(
  spc_tbl,
  spc_tbl_l = NULL,
  lcols_spc = "spc",
  lcol_measure = NULL,
  lcol_measure_col_palette = "Spectral",
  lcol_measure_col_direction = -1,
  spc_id = "unique_id",
  group_id = "sample_id",
  group_id_order = TRUE,
  group_color = TRUE,
  group_color_palette = NULL,
  group_panel = TRUE,
  group_legend = FALSE,
  ncol = NULL,
  relabel_spc = TRUE,
  ylab = "Spectrum value",
  alpha = 0.5,
  line_width = 0.2,
  ...
)
```

Arguments

spc_tbl	Tibble data frame containing spectra, x-axis values, metadata and eventual measured variables as list-columns.
spc_tbl_l	List of spectral tibbles (data frames). Default is NULL (argument is not used).
lcols_spc	Character vector of spectral list-columns to be extracted. Default is "spc" (raw spectra).

`lcol_measure` Character vector of length 1 denoting the column name of the measure columns. This argument is optional. Default is `NULL`, which does not extract an additional measure column.

`lcol_measure_col_palette` Palette value supplied to `ggplot::scale_colour_brewer()`. Default is `"Spectral"`, but you can set it to the default argument `1` (will use `scale_colour_brewer(..., palette = 1)`).

`lcol_measure_col_direction` Sets the the order of colours in the scale that is based on a measure column. Default is `-1` which reverses the scale. Argument is passed on to the function `ggplot2::sclae_colour_brewer()` as argument `direction`.

`spc_id` Character vector denoting column name for a unique spectrum ID. Default is `"unique_id"`.

`group_id` Character vector denoting column name for the spectrum group ID. Default is `"sample_id"`. The group ID is used for plotting spectra by group (e.g. by using different colors or panels).

`group_id_order` Logical that specifies whether the panel names derived from a numeric `group_id` column are reordered using ascending numbers. Default is `TRUE`.

`group_color` Logical defining whether spectra are colored by the column specified by `group_id`.

`group_color_palette` Character (1L) defining the diverging colour scales from colorbrewer.org; see `?scale_colour_brewer` for supported diverging colur types (`palette` argument).

`group_panel` Logical defining whether spectra are arranged into panels by groups specified in `group_id`. Default is `TRUE`.

`group_legend` Logical defining whether a legend for the `group_id` is plotted. Default is `FALSE`.

`ncol` Integer vector of length 1. Defines number of columns when plotting panels (facets). Default is `NULL` (argument not used).

`relabel_spc` Logical defining whether panels are relabeled with custom names for spectra types. Default is TRUE. When `TRUE`, arguments from `relabel_spc_types` can be passed to `plot_spc_ext` (supported via the `...` (ellipsis) argument)

`ylab` Character vector or vector of type `"expression"` created by mathematical expression created by `expression`. Custom annotation for y-axis of spectra

`alpha` Integer of length 1, from 0 to 1. Defines transparency of spectral lines. Default is `0.5` (0 is completely transparent and 1 is no transparency).

`line_width` Numeric vector of length 1 specifying the width of the spectral lines. Default is `0.2`.

`...` Further arguments to be passed to `plot_spc_ext`. Currently, arguments of `relabel_spc_types` are supported.

Value

Object of class `"ggplot"` (ggplot2 graph).

predict_from_spc *Predict soil properties of new spectra based on a list of calibration models*

Description

Append predictions for a set of responses specified by a list of calibration models and a tibble containing preprocessed spectra as list-columns.

Usage

```
predict_from_spc(model_list, spc_tbl, slice = TRUE)
```

Arguments

model_list	List of model output generated from calibration step (pls_ken_stone()
spc_tbl	Tibble of spectra after preprocessing (preprocess_spc())
slice	Logical expression wheather only one row per sample_id returned.

Value

tibble with new columns model, and predicted values with column names of model list.

preprocess_spc *Preprocess spectra*

Description

Preprocesses spectra in tibble column by sample_id after averaging spectra by simplerspec::average_spc().

Usage

```
preprocess_spc(spc_tbl, select, column_in = "spc_mean", custom_function = NULL)
```

Arguments

spc_tbl	Tibble that contains spectra to be preprocessed within a list-column.
select	Character vector of predefined preprocessing options to be applied to the spectra list-column specified in column_in. Common prefined values are stated as abbreviated preprocessing methods and options such as "sg_1_w21", where "sg" stands for Savitzky-Golay and 1 for first derivative and "w21" for a window size of 21 points.
column_in	Character vector of single list-column in spc_tbl that contain list of spectra (1 row matrix) to be processed by function supplied in select.

custom_function

A character string of a custom processing function that is later parsed (produces expression in a list) and evaluated within the function preprocess_spc. The character vector argument of custom_function needs to contain "spc_raw", which is the single data table of spectra that results from binding a list of data.tables (spectra to preprocess) from the spectra list-column specified in column_in. An example for a value is "prospectr::savitzkyGolay(X = spc_raw, m = 0, p = 3, w = 9)". Optional argument. Default is NULL.

---

read_asd *Read ASD fieldspec spectrometer data export into into simplerspec spectra tibble.*

Description

Read tab delimited text (.txt) files exported from ASD field spectrometer into simplerspec spectra tibble. ASD Fieldspec data files are expected in .txt tab delimited file format. The first row should contain the name 'Wavelength' for the first column and the file names for the remaining columns.

Usage

```
read_asd(file)
```

Arguments

file	Tab delmited file from ASD software export where the first column called Wavelength contais wavelengths in nanometer and the remaining columns are sample spectra referred by an ID name provided in the first row of these columns.

Value

Spectra data in tibble data frame (class tbl_df) that contains columns sample_id (derived from 2nd and following column names of tab delimited ASD exported text file), spc (list-column of spectral matrices) and wavelengths (list-column containing wavelength vectors).

---

read_asd_bin *Read ASD binary files and gather spectra and metadata in tibble data frame.*

Description

Read multiple ASD binary files and gather spectra and metadata into a simplerspec spectral tibble (data frame). The resulting spectral tibble is compatible with the simplerspec spectra processing and modeling framework.

Usage

```
read_asd_bin(fnames)
```

Arguments

fnames	Character vector containing full paths of ASD binary files to be read

Value

A spectral tibble (data frame) containing the follwing columns:

unique_id	Character vector. Unique identifier containing file name pasted with date and time.
file_id	Character vector containing file names and exension
sample_id	Character vector containing files names without extension
metadata	List-column. List of data frames containing spectral metadata
wavelengths	List-column. List of wavelengths vectors (numeric).
spc_radiance	List-column. List of data.tables containing radiance sample spectra.
spc_reference	List-column. List of data.tables containing reference reflectance spectra.
spc	List-column. List of data.tables containing final reflectance spectra.

---

read_opus_bin_univ *Read a Bruker OPUS spectrum binary file*

Description

Read single binary file acquired with an Bruker Vertex FTIR Instrument

Usage

```
read_opus_bin_univ(file_path, extract = c("spc"),
print_progress = TRUE, atm_comp_minus4offset = FALSE)
```

Arguments

file_path	Character vector with path to file
extract	Character vector of spectra types to extract from OPUS binary file. Default is c("spc"), which will extract the final spectra, e.g. expressed in absorbance (named AB in Bruker OPUS programs). Possible additional values for the character vector supplied to extract are "ScSm" (single channel spectrum of the sample measurement), \ code"ScRf" (single channel spectrum of the reference measurment), "IgSm" (interferogram of the sample measurment) and "IgRf" (interferogram of the reference measurement).
print_progress	Logical (default TRUE) whether a message is printed when an OPUS binary file is parsed into an R list entry.

atm_comp_minus4offset
: Logical whether spectra after atmospheric compensation are read with an offset of -4 bites from Bruker OPUS files. Default is FALSE.

---

read_opus_univ *Read a list of Bruker OPUS spectrum binary files.*

Description

Read multiple spectral files measured with a Bruker FTIR Instrument. Files containing spectra are in OPUS binary format. read_opus_univ is a wrapper for read_opus_bin_univ())

Usage

```
read_opus_univ(fnames, extract = c("spc"), parallel = FALSE,
atm_comp_minus4offset = FALSE)
```

Arguments

fnames
: List of character vectors containing full path names of spectra

extract
: Character vector of spectra types to extract from file. Possible values are: "spc" (AB block in Bruker Opus software), "spc_nocomp" (Spectra before final atmospheric compensation; only present if background correction has been set in Opus), "ScSm" (Single channel spectrum of the sample), "ScRf" (Single channel spectrum of the sample), "IgSm" (Interferogram of the sample), "IgRf" (Interferogram of the reference). Default is extract = c("spc").

parallel
: Logical (TRUE or FALSE indicating whether files are read in parallel (multiple processors or multiple cores)). Default is parallel = FALSE. If TRUE a parallel backend needs to be registered, e.g. by using the doParallel package.

atm_comp_minus4offset
: Logical whether spectra after atmospheric compensation are read with an offset of -4 bites from Bruker OPUS files. Default is FALSE.

Value

out List spectra and metadata (parameters) extracted from Bruker OPUS spectrometer files. List names are the names of the OPUS files whose spectral data were extracted.

remove_outliers *Remove outlier spectra*

Description

Remove outlier spectra based on the `pcout()` function of the `mvoutlier` package.

Usage

```
remove_outliers(list_spectra, remove = TRUE)
```

Arguments

`list_spectra`	List that contains averaged spectral information in list element `MIR_mean` (data.frame) and metadata in `data_meta` (data.frame).
`remove`	logical expression (`TRUE` or `FALSE`) that specifies weather spectra shall be removed. If `rm = FALSE`, there will be no outlier removal

Details

This is an optional function if one wants to remove outliers.

Value

Returns list `spectra_out` that contains:

- `MIR_mean`: Outlier removed MIR spectra as data.frame object. If `remove = FALSE`, the function will return almost identical list identical to `list_spectra`, except that the first `indices` column of the spectral data frame `MIR_mean` is removed (This is done for both options `remove = TRUE` and `remove = FALSE`).
- `data_meta`: metadata data.frame, identical as in the `list_spectra` input list.
- `plot_out`: (optional) ggplot2 graph that shows all spectra (absorbance on x-axis and wavenumber on y-axis) with outlier marked, if `remove = TRUE`.

resample_spc *Resample spectra in list-column to new x-axis interval*

Description

Resamples (interpolates) different spectra types with corresponding x-axis values that are both stored in list-columns of a spectra tibble. A spectra tibble hosts spectra, x-axis vectors, metadata, and further linked data with standardized naming conventions. Data input for resampling can for example be generated with `simplerspec::gather_spc()`. Resampling is a key harmonizing step to process and later model spectra measured at different resolutions and spectral ranges (i.e., different spectrometer devices and/or measurement settings).

Usage

```
resample_spc(
  spc_tbl,
  column_in = "spc",
  x_unit = c("wavenumber", "wavelength"),
  wn_lower = 500,
  wn_upper = 4000,
  wn_interval = 2,
  wl_lower = 350,
  wl_upper = 2500,
  wl_interval = 1,
  interpol_method = c("linear", "spline")
)
```

Arguments

spc_tbl	Spectra data embedded in a tibble object (classes "tbl_df", "tbl", "data.frame"). The spectra tibble needs to contain at least of one of the the spectra columns spc, spc_rs, spc_mean, spc_nocomp, sc_sm, sc_rf, or spc_pre (list-columns with spectral data.tables), and wavenumbers or wavelengths (list-column with vectors of x-axis values corresponding to each spectrum). The help section *"Matching spectrum type and corresponding x-axis type"* describes the spectra types and corresponding x-axis types.
column_in	Character vector of length 1L or symbol/name specifying the name of list-column that contains the spectra to be resampled.
x_unit	Character vector of length 1L specifying the measurement unit of the x-axis values (list-column) of the input spectra in spc_tbl. Possible values are "wavenumber" (default) or "wavelength". Wavenumber is a convenient unit of frequency in the mid-infrared spectral range, where wavelength is often used as spatial period for the visible and near-infrared range.
wn_lower	Numeric value of lowest wavenumber. This argument will only be used if x_unit = "wavenumber". The value serves as starting value for the new wavenumber sequence that the spectra will be resampled upon. Default value is 500 (i.e., in reciprocal centimeters).
wn_upper	Numeric value of highest wavenumber. This argument will only be used if x_unit = "wavenumber. The value will be used as last value of the new wavenumber sequence that the spectra will be resampled upon. Default value is 4000 (i.e., in reciprocal centimeters).
wn_interval	Numeric value of the wavenumber increment for the new wavenumber sequence that the spectra will be resampled upon. Default value is 2 (i.e., in reciprocal centimeters).
wl_lower	Numeric value of lowest wavelength. This argument will only be used if x_unit = "wavelength". The value serves as starting value of the new wavenumber sequence that the spectra will be resampled upon. Default value is 350 (i.e. in nanometers).

wl_upper — Numeric value of highest wavelength. This argument will only be used if x_unit = "wavelength". The value will be used as last value of the new wavenumber sequence that the spectra will be resampled upon. Default value is 2500 (i.e., in nanometers).

wl_interval — Numeric value of the wavelength increment for the new wavenumber sequence that the spectra will be resampled upon. This argument will only be used if x_unit = "wavelength". Default value is 1 (i.e., in nanometers).

interpol_method — Character of "linear" (default) or "spline" with the interpolation method. "spline" uses a cubic spline to interpolate the input spectra at given x-axis values to new equispaced x-axis intervals.

Value

A spectra tibble (spc_tbl) containing two added list-columns:

- spc_rs: Resampled spectra as list of data.tables
- wavenumbers_rs or wavelengths_rs: Resampled x-axis values as list of numeric vectors

Matching spectrum type and corresponding x-axis type

The combinations of input spectrum types (column_in) and corresponding x-axis types are generated from a simple lookup list. The following key-value(s) pairs can be matched at given key, which is the column name from column_in containing the spectra.

- "spc" : "wavenumbers" or "wavelengths" (raw spectra)
- "spc_rs" : "wavenumbers_rs" or "wavelengths_rs") (resampled spectra)
- "spc_mean" : "wavenumbers_rs" or "wavelengths_rs" (mean spectra)
- "spc_nocomp" "wavenumbers" or "wavelengths" (spectra prior atmospheric compensation)
- "sc_sm" : c("wavenumbers_sc_sm", "wavelengths_sc_sm") (single channel sample spectra)
- "sc_rf" : c("wavenumbers_sc_rf", "wavelengths_sc_rf") (single channel reference spectra)
- "spc_pre" : "xvalues_pre" (preprocessed spectra)

---

select_ref_spc — *Select a set of reference spectra to be measured by reference analysis methods*

Description

Select a set of calibration spectra to develop spectral models. Samples in this list will be analyzed using laboratory reference methods.

Usage

```
select_ref_spc(spc_tbl, ratio_ref, pc, print = TRUE)
```

Arguments

spc_tbl	Spectra as tibble objects that contain preprocessed spectra
ratio_ref	Ratio of desired reference samples to total sample number
pc	Number of principal components (numeric). If pc < 1, the number of principal components kept corresponds to the number of components explaining at least (pc * 100) percent of the total variance.
print	logical expression whether a plot (ggplot2) of sample selection for reference analysis is shown in PCA space (TRUE or FALSE).

---

select_spc_vars *Select every n-th spectral variable for all spectra and x-values in spectral tibble (*spc_tbl*)*

Description

Select every n-th spectral variable for all spectra and x-values in spectral tibble (spc_tbl)

Usage

```
select_spc_vars(
  spc_tbl,
  lcol_spc = "spc_pre",
  lcol_xvalues = "xvalues_pre",
  every = NULL
)
```

Arguments

spc_tbl	Tibble data.frame containing spectra in list-column
lcol_spc	List-column containing spectra, specified with column name as symbols or 1L character vector.
lcol_xvalues	List-column containing x-values, specified with column name as symbols or 1L character vector.
every	Every n-th spectral positions to keep as 1L integer vector.

Value

a spectral tibble

slice_xvalues *Slice spectra into defined x-axis ranges*

Description

Slice spectra contained in list-column of spectral tibble (data frame). A list of x-axis value ranges can be specified. Spectra are cut based on these ranges.

Usage

```
slice_xvalues(
  spc_tbl,
  xunit_lcol = "wavenumbers",
  spc_lcol = "spc",
  xvalues_cut = NULL
)
```

Arguments

spc_tbl	Spectral data in a tibble object (classes "tibble_df", "tbl" and "data.frame"). The spectra tibble is expected to contain at least the column spc (list-column with spectral matrices stored in a list) and wavenumbers or wavelengths (list-column that contains list of x-axis values).
xunit_lcol	Character vector that specifies column name where x-axis axis units are stored within spc_tbl. Default is "wavenumber".
spc_lcol	Character vector that specifies which column (list-column) contains spectra to be sliced. Default is "spc".
xvalues_cut	List of numeric vectors that contains upper and lower bounds of respective regions to keep in spectra. The spectral regions outside the xvalues_cut intervals will be cut out in the output spectra.

Value

Spectral tibble (data frame with list-columns) with sliced x-axis column and spectral column. Both the x-axis list-column and the spectral tibble list-column only contain data specified within the xvalues_cut argument (list of numeric vectors).

soilspec_yamsys *Soil spectra and laboratory reference data from Baumann et al. (2021)*

Description

Data from "Estimation of soil properties with mid-infrared soil spectroscopy across yam production landscapes in West Africa".

Usage

```
soilspec_yamsys
```

Format

`soilspec_yamsys`:
A tibble data frame with 284 rows and 40 columns. The spectra are in the `spc` list-column.

Source

https://soil.copernicus.org/articles/7/717/2021/

`split_df2l` *Split a tibble data frame into a list of tibbles by a group column*

Description

Helper function that calls `split` on a tibble using a grouping column within tibble.

Usage

```
split_df2l(tbl_df, group)
```

Arguments

`tbl_df`	Tibble data frame
`group`	Character vector with name of column based on which tibble is split into a list of tibbles

Value

List of tibbles. Each tibble contains data split by a group column within `tbl_df`.

Index