Package ‘isoreader’

February 16, 2021

Title  Read Stable Isotope Data Files

Description  Interface to the raw data file formats commonly encountered in scientific disciplines that make use of stable isotopes.

Version  1.3.0

URL  https://github.com/isoverse/isoreader

BugReports  https://github.com/isoverse/isoreader/issues

Depends  R (>= 4.0.0), stats

Imports  methods, R.utils, magrittr, rlang (>= 0.4.5), tidyselect (>= 1.0.0), vctrs (>= 0.3.4), tibble (>= 3.0.0), dplyr (>= 1.0.0), tidyr (>= 1.0.0), glue (>= 1.4.0), stringr (>= 1.4.0), purrr (>= 0.3.4), future (>= 1.18.0), lubridate (>= 1.7.9.2), readr (>= 1.4.0), progress (>= 1.2.2), UNF (>= 2.0.6)

Suggests  devtools, testthat, feather (>= 0.3.5), readxl (>= 1.3.1), openxlsx (>= 4.1.5), xml2 (>= 1.3.1), rhdf5 (>= 2.0.0), knitr, markdown, covr

biocViews

License  GPL (>= 2)

Encoding  UTF-8

LazyData  true

VignetteBuilder  knitr

RoxygenNote  7.1.1

NeedsCompilation  no

Author  Sebastian Kopf [aut, cre] <https://orcid.org/0000-0002-2044-0201>, Brett Davidheiser-Kroll [aut] <https://orcid.org/0000-0002-6153-7851>, Ilja Kocken [aut] <https://orcid.org/0000-0003-2196-8718>

Maintainer  Sebastian Kopf <sebastian.kopf@colorado.edu>

Repository  CRAN

Date/Publication  2021-02-16 06:20:07 UTC
### R topics documented:

<table>
<thead>
<tr>
<th>R Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>extract_data</td>
<td>3</td>
</tr>
<tr>
<td>extract_substring</td>
<td>4</td>
</tr>
<tr>
<td>extract_word</td>
<td>5</td>
</tr>
<tr>
<td>isoread</td>
<td>6</td>
</tr>
<tr>
<td>iso_add_file_info.iso_file_list</td>
<td>7</td>
</tr>
<tr>
<td>iso_caching</td>
<td>8</td>
</tr>
<tr>
<td>iso_calculate_ratios</td>
<td>9</td>
</tr>
<tr>
<td>iso_cleanup_reader_cache</td>
<td>9</td>
</tr>
<tr>
<td>iso_convert_signals</td>
<td>10</td>
</tr>
<tr>
<td>iso_convert_time</td>
<td>10</td>
</tr>
<tr>
<td>iso_debug_mode</td>
<td>10</td>
</tr>
<tr>
<td>iso_expand_paths</td>
<td>11</td>
</tr>
<tr>
<td>iso_export_to_excel</td>
<td>11</td>
</tr>
<tr>
<td>iso_export_to_feather</td>
<td>13</td>
</tr>
<tr>
<td>iso_filter_files</td>
<td>15</td>
</tr>
<tr>
<td>iso_filter_files_with_problems</td>
<td>16</td>
</tr>
<tr>
<td>iso_find_absolute_path_roots</td>
<td>16</td>
</tr>
<tr>
<td>iso_format</td>
<td>17</td>
</tr>
<tr>
<td>iso_get_all_data</td>
<td>18</td>
</tr>
<tr>
<td>iso_get_bgrd_data</td>
<td>20</td>
</tr>
<tr>
<td>iso_get_data</td>
<td>21</td>
</tr>
<tr>
<td>iso_get_data_summary</td>
<td>21</td>
</tr>
<tr>
<td>iso_get_default_reader_parameters</td>
<td>22</td>
</tr>
<tr>
<td>iso_get_file_info</td>
<td>22</td>
</tr>
<tr>
<td>iso_get_problems</td>
<td>23</td>
</tr>
<tr>
<td>iso_get_problems_summary</td>
<td>24</td>
</tr>
<tr>
<td>iso_get_raw_data</td>
<td>24</td>
</tr>
<tr>
<td>iso_get_reader_example</td>
<td>25</td>
</tr>
<tr>
<td>iso_get_resistors</td>
<td>26</td>
</tr>
<tr>
<td>iso_get_resistors_info</td>
<td>27</td>
</tr>
<tr>
<td>iso_get_standards</td>
<td>27</td>
</tr>
<tr>
<td>iso_get_standards_info</td>
<td>28</td>
</tr>
<tr>
<td>iso_get_supported_file_types</td>
<td>29</td>
</tr>
<tr>
<td>iso_get_units</td>
<td>29</td>
</tr>
<tr>
<td>iso_get_vendor_data_table</td>
<td>30</td>
</tr>
<tr>
<td>iso_has_problems</td>
<td>31</td>
</tr>
<tr>
<td>iso_info_messages</td>
<td>31</td>
</tr>
<tr>
<td>iso_is_double_with_units</td>
<td>32</td>
</tr>
<tr>
<td>iso_is_file</td>
<td>33</td>
</tr>
<tr>
<td>iso_make_units_explicit</td>
<td>34</td>
</tr>
<tr>
<td>iso_make_units_implicit</td>
<td>34</td>
</tr>
<tr>
<td>iso_mutate_file_info</td>
<td>35</td>
</tr>
<tr>
<td>iso_omit_files_with_problems</td>
<td>36</td>
</tr>
<tr>
<td>iso_parse_file_info</td>
<td>36</td>
</tr>
<tr>
<td>iso_plot_continuous_flow_data</td>
<td>37</td>
</tr>
<tr>
<td>iso_plot_dual_inlet_data</td>
<td>38</td>
</tr>
</tbody>
</table>
extract_data

Overview of text data extraction functions

Description

The following functions are intended to make it easy to extract relevant information from textual data. These functions are primarily intended for use in iso_mutate_file_info and inside the filtering conditions passed to iso_filter_files. However, they can of course also be used stand-alone and in regular mutate or filter calls on the data frames returned by the data retrieval functions (iso_get_raw_data, iso_get_file_info, iso_get_vendor_data_table, etc.). Not that all the parse_ functions are used in iso_parse_file_info for easy type conversions.

Details

For simultaneous extraction of pure text data into multiple columns, please see the extract function from the tidyr package.

- extract_substring is a generic convenience function to extract parts of textual data (based on regular expression matches). Can be used in combination with the parsing functions to turn extracted substrings into numerical or logical data.
extract_substring

• `extract_word` is a more specific convenience function to extract the 1st/2nd/3rd word from textual data.
• `parse_number` is a convenience function to extract a number even if it is surrounded by text (re-exported from the `readr` package).
• `parse_double` parses text that holds double (decimal) numerical values without any extraneous text around - use `parse_number` instead if this is not the case (re-exported from the `readr` package).
• `parse_integer` parses text that holds integer (whole number) numerical values without any extraneous text around - use `parse_number` instead if this is not the case (re-exported from the `readr` package).
• `parse_logical` parses text that holds logical (boolean, i.e. TRUE/FALSE) values (re-exported from the `readr` package).
• `parse_datetime` parses text that holds date and time information (re-exported from the `readr` package).

See Also

Other data extraction functions: `extract_substring()`, `extract_word()`

---

extract_substring

Extract a substring from text

Description

This is a convenience function to capture substrings from textual data. Uses `str_match_all` internally but instead of returning everything, always returns only one single part of the match, depending on parameters `capture_n` and `capture_group`.

Usage

```r
extract_substring(
  string,
  pattern,
  capture_n = 1,
  capture_bracket = 0,
  missing = NA_character_
)
```

Arguments

- `string` string to extract
- `pattern` regular expression pattern to search for
- `capture_n` within each string, which match of the pattern should be extracted? e.g. if the pattern searches for words, should the first, second or third word be captured?
extract_word

capture_bracket
for the captured match, which capture group should be extracted? i.e. which parentheses-enclosed segment of the pattern? by default captures the whole pattern (capture_bracket = 0).

missing
what to replace missing values with? Note that values can be missing because there are not enough captured matches or because the actual capture_bracket is empty.

Value
character vector of same length as string with the extracted substrings

See Also
Other data extraction functions: extract_data, extract_word()

extract_word Extract words from text

Description
This extracts words from text, by default looks for continuous sequences of numbers and/or letters. Can adjust whether characters such as ",", ",", ",", and "," should be counted as part of a word or separate them and whether numbers should be included.

Usage
extract_word(
  string,
  capture_n = 1,
  include_numbers = TRUE,
  include_underscore = FALSE,
  include_dash = FALSE,
  include_space = FALSE,
  include_colon = FALSE,
  missing = NA_character_
)

Arguments
  string string to extract
  capture_n which word to extract? 1st, 2nd, 3rd?
  include_numbers whether to include numbers (0-9) as part of the word (if FALSE, numbers will work as a word separator)
include_underscore
whether to include the underscore character (_) as part of a word (if FALSE, it will work as a word separator)

include_dash
whether to include the dash character (-) as part of a word (if FALSE, it will work as a word separator)

include_space
whether to include the space character ( ) as part of a word (if FALSE, it will work as a word separator)

include_colon
whether to include the colon character (:) as part of a word (if FALSE, it will work as a word separator)

missing
what to replace missing values with? Note that values can be missing because there are not enough captured matches or because the actual capture_bracket is empty.

See Also
Other data extraction functions: extract_data, extract_substring()

Examples
x_text <- extract_word(c("sample number16.2", "sample number7b"),
capture_n = 2, include_colon = TRUE)
# "number16.2" "number7b"
x_num <- parse_number(x_text)
# 16.2 7.0
Add additional file information

Description

This function makes it easy to add additional file info (iso_get_file_info) to isofile objects and data frames by a single left_join or multiple sequential left_join operations. The function provides a detailed summary of the information that was added unless quiet = TRUE. Note that one-to-many joins are not permitted (and will fail with an informative error) since this would lead to likely unintended data duplication in the isofiles. However, one-to-one and many-to-one joins are fully supported and should cover all needed use cases for this function. Also note that for each join, only the new_file_info rows that have defined non-NA, non-empty (""") values in all join_by columns will be considered for the join and that only new_file_info columns that do NOT already exist in ANY file information will be added. For changing the values of existing file information, please use iso_mutate_file_info instead.

Usage

## S3 method for class 'iso_file_list'
iso_add_file_info(iso_files, new_file_info, ..., quiet = default(quiet))

## S3 method for class 'data.frame'
iso_add_file_info(df, new_file_info, ..., quiet = default(quiet))

 iso_add_file_info(....)

Arguments

iso_files collection of iso_file objects
new_file_info data frame with new file information to add to the isofiles
... each parameter specifies a set of join_by column(s) to add the new_file_info to the existing file information. The provided parameters are applied sequentially. At least one must be specified.
quiet whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off

Details

Single left_join: this is the most common use of this function and basically a simple left join operation (with some additional safety checks). Specify a single join_by in the ..., such as e.g. c("file_id") to add additional file information joining by the file_id column.
Multiple sequential `left_join`: this use case is for applying a set of increasingly more specific `join_by` rules. For example, `... = c("Identifier 1","Identifier 2"),c("file_id")` would serve to first add one set of new file information for all iso files based on their `Identifier 1` and `Identifier 2` columns and then overwrite the new information with more specific details for a subset of iso files based on their `file_id` column, all based on a single overview `new_file_info` data frame. Basically, each set of `join_by` conditions specified in `...` must describe a valid `left_join` `join_by` parameter to merge the `new_file_info` with the existing file info. Each set of `new_file_info` data can overwrite the previous `join_by` matches such that the last set of `join_by` column(s) provided in `...` will overwrite all previous matches for which it applies, even if they have already been a match for a previous column.

**Value**

the original iso files or data frame with the new file info added in.

**See Also**

Other file info operations: `iso_filter_files()`, `iso_mutate_file_info()`, `iso_parse_file_info()`, `iso_rename_file_info()`, `iso_select_file_info()`, `iso_set_file_root()`

---

## iso_caching

**Turn caching on/off**

### Description

These functions turn caching of data files (and reading from cache) on/off in all subsequent isoread calls by changing the global settings for the `cache` parameter. Can be called stand alone or within a pipeline.

### Usage

```r
iso_turn_reader_caching_on(data = NULL)
iso_turn_reader_caching_off(data = NULL)
```

### Arguments

- `data` a data frame - returned invisibly as is if provided (e.g. in the middle of a pipeline)

### See Also

Other settings functions: `iso_get_default_reader_parameters()`, `iso_info_messages`, `iso_set_default_read_parameters()`
iso_calculate_ratios  moved to isoprocessor

Description
moved to isoprocessor

Usage
iso_calculate_ratios(…)

Arguments
...  deprecated

iso_cleanup_reader_cache

Cleanup cached files

Description
Removes all cached files.

Usage
iso_cleanup_reader_cache(all = FALSE)

Arguments
all  deprecated

iso_convert_signals  moved to isoprocessor

Description
moved to isoprocessor

Usage
iso_convert_signals(...)

Arguments
...  deprecated
Description

moved to isoprocessor

Usage

iso_convert_time(...)

Arguments

... deprecated

---

iso_debug_mode  Debugging functions

Description

For troubleshooting. Not exported.

Usage

iso_turn_debug_on(data = NULL, catch_errors = TRUE, cache = FALSE)

iso_turn_debug_off(data = NULL)

set_read_file_event_expr(event_expr = NULL)

set_finish_file_event_expr(event_expr = NULL)

Arguments

data  a data frame - returned invisibly as is if provided (e.g. in the middle of a pipeline)
catch_errors  whether to still catch errors in debug mode or whether to throw them
cache  whether to cache or read anything from cache
event_expr  an expression to evaluate in the context of reading individual iso files (evaluated in the local environment at the beginning of a file read)
iso_expand_paths

Expand file paths

Description

Helper function to expand the provided paths to find data files in folders and subfolders that match any of the specified extensions. Filepaths will be kept as is, only folders will be expanded. Note that this function is rarely called directly. It is used automatically by iso_read_dual_inlet and iso_read_continuous_flow to identify files of interest based on the file paths provided.

Usage

iso_expand_paths(path, extensions = c(), root = ".")

Arguments

- **path**: vector of file/folder paths, mixed relative and absolute paths are allowed.
- **extensions**: which extensions to look for? (with or without leading .) - this is typically one or more of the extensions listed by iso_get_supported_file_types
- **root**: root directory for the isofiles. Can be relative to the current working directory (e.g. "data") or an absolute path on the file system (e.g. "/Users/..." or "C:/Data/..”). The default is the current working directory ("."). Can be supplied as a vector of same length as the provided paths if the paths have different roots.

Value

data frame with columns root (root as provided) and path of all the found files.

See Also

Other file system functions: iso_find_absolute_path_roots(), iso_root_paths(), iso_shorten_relative_paths()

iso_export_to_excel

Export data to Excel

Description

This function exports the passed in iso_files to Excel. The different kinds of data (raw data, file info, methods info, etc.) are exported to separate tabs within the excel file. Use the various include... parameters to specify what information to include. Note that in rare instances where vectorized data columns exist in the file information (e.g. measurement_info), they are concatenated with ",," in the excel export. Note that the openxlsx package required for this export is not installed automatically as part of isoreader. Please install it manually if missing using install.packages("openxlsx").
iso_export_to_excel

Usage

```r
iso_export_to_excel(
  iso_files,
  filepath,
  include_file_info = everything(),
  include_raw_data = everything(),
  include_standards = !!enexpr(include_method_info),
  include_resistors = !!enquo(include_method_info),
  include_vendor_data_table = everything(),
  include_problems = everything(),
  with_explicit_units = FALSE,
  include_method_info = everything(),
  with_ratios = NULL,
  quiet = default(quiet)
)
```

Arguments

- **iso_files**: collection of `iso_file` objects
- **filepath**: the path (folder and filename) to the export file. The correct file extension is automatically added if not already in the filename, i.e. filename can be provided with or without extension.
- **include_file_info**: which file information to include (see `iso_get_file_info`). Use `c(...)` to select multiple, supports all `select` syntax including renaming columns.
- **include_raw_data**: which columns from the raw data to include. Use `c(...)` to select multiple columns, supports all `select` syntax including renaming columns. Includes all columns by default. Set to `NULL` to include no raw data.
- **include_standards**: which columns from the standards info to include. Use `c(...)` to select multiple columns, supports all `select` syntax including renaming columns. By default, everything is included (both standards and ratios). To omit the ratios, change to `select = file_id:reference`. Set to `NULL` to include no standards info.
- **include_resistors**: which columns from the resistors info to include. Use `c(...)` to select multiple columns, supports all `select` syntax including renaming columns. Includes all columns by default. Set to `NULL` to include no resistors info.
- **include_vendor_data_table**: which columns from the vendor data table to include. Use `c(...)` to select multiple columns, supports all `select` syntax including renaming columns. Includes all columns by default. Set to `NULL` to include no vendor data table.
- **include_problems**: which columns from problems to include. Use `c(...)` to select multiple columns, supports all `select` syntax including renaming columns. Includes none of the read
problems by default. Set to `include_problems = everything()` to include all columns.

`with_explicit_units`
whether to include units in the column headers of the returned data frame instead of the column data types (see `iso_double_with_units`). Note that any select conditions have to refer to the column names including the full units.

`include_method_info`
deprecated in favor of the more specific `include_standards` and `include_resistors`

`with_ratios`
deprecated, please use the `select` parameter to explicitly include or exclude ratio columns

`quiet`
whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to `iso_turn_info_messages_on` and `iso_turn_info_messages_off`

**Value**
returns the `iso_files` object invisibly for use in pipelines

**See Also**
Other export functions: `iso_export_to_feather()`, `iso_save()`

---

### iso_export_to_feather

**Export to feather**

This function exports the passed in `iso_files` to the Python and R shared feather file format. The different kinds of data (raw data, file info, methods info, etc.) are exported to separate feather files that are saved with the provided `filepath_prefix` as prefix. All are only exported if the corresponding `include_` parameter is set to `TRUE` and only for data types for which this type of data is available and was read (see `iso_read_dual_inlet`, `iso_read_continuous_flow` for details on read parameters). Note that in rare instances where vectorized data columns exist in the file information (e.g. `measurement_info`), they are concatenated with `,'` in feather output. Note that the feather package required for this export is not installed automatically as part of isoreader. Please install it manually if missing using `install.packages("feather")`.

**Usage**

```r
iso_export_to_feather(
  iso_files,
  filepath_prefix,
  include_file_info = everything(),
  include_raw_data = everything(),
  include_standards = !!enexpr(include_method_info),
  include_resistors = !!enquo(include_method_info),
)```
iso_export_to_feather

include_vendor_data_table = everything(),
include_problems = everything(),
with_explicit_units = FALSE,
include_method_info = everything(),
quiet = default(quiet)
)

Arguments

iso_files collection of iso_file objects
filepath_prefix what to use as the prefix for the feather file names (e.g. name of the data collection or current date)
include_file_info which file information to include (see iso_get_file_info). Use c(...) to select multiple, supports all select syntax including renaming columns.
include_raw_data which columns from the raw data to include. Use c(...) to select multiple columns, supports all select syntax including renaming columns. Includes all columns by default. Set to NULL to include no raw data.
include_standards which columns from the standards info to include. Use c(...) to select multiple columns, supports all select syntax including renaming columns. By default, everything is included (both standards and ratios). To omit the ratios, change to select = file_id:reference. Set to NULL to include no standards info.
include_resistors which columns from the resistors info to include. Use c(...) to select multiple columns, supports all select syntax including renaming columns. Includes all columns by default. Set to NULL to include no resistors info.
include_vendor_data_table which columns from the vendor data table to include. Use c(...) to select multiple columns, supports all select syntax including renaming columns. Includes all columns by default. Set parameter with_explicit_units = TRUE to make column units explicit (keep in mind that this will require specific include_vendor_data_table column selections to reflect the column names including the units). Set to NULL to include no vendor data table.
include_problems which columns from problems to include. Use c(...) to select multiple columns, supports all select syntax including renaming columns. Includes none of the read problems by default. Set to include_problems = everything() to include all columns.
with_explicit_units whether to include units in the column headers of the returned data frame instead of the column data types (see iso_double_with_units). Note that any select conditions have to refer to the column names including the full units.
include_method_info deprecated in favor of the more specific include_standards and include_resistors
iso_filter_files

quiet whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off

Value

returns the iso_files object invisibly for use in pipelines

See Also

Other export functions: iso_export_to_excel(), iso_save()

description

Filter for specific isofiles using file info columns (iso_get_file_info). Works just like dplyr’s filter except that it provides the user with some information on what has been filtered. Returns NULL if none of the isofiles’ file info matches the filter criteria. You can also use filter directly to filter collections of iso_file objects.

Usage

iso_filter_files(iso_files, ..., quiet = default(quiet))

Arguments

 iso_files collection of iso_file objects
... dplyr-style filter conditions applied based on each file’s file_info (see iso_get_file_info)
quiet whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off

See Also

Other file_info operations: iso_add_file_info.iso_file_list(), iso_mutate_file_info(), iso_parse_file_info(), iso_rename_file_info(), iso_select_file_info(), iso_set_file_root()
### iso_filter_files_with_problems

Filter out problematic files

**Description**

Use this function to filter out files that have encountered problems, either errors, warnings or both and returns the remaining iso_files. For additional functions available to check for and deal with problems, see the iso_problem_functions.

**Usage**

```r
iso_filter_files_with_problems(
  iso_files,
  remove_files_with_errors = TRUE,
  remove_files_with_warnings = FALSE,
  quiet = default(quiet)
)
```

**Arguments**

- `iso_files` : collection of iso_file objects
- `remove_files_with_errors` : whether to remove files with errors (default is TRUE)
- `remove_files_with_warnings` : whether to remove files with warnings (default is FALSE)
- `quiet` : whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to `iso_turn_info_messages_on` and `iso_turn_info_messages_off`

**See Also**

Other problem functions: `iso_get_problems_summary()`, `iso_get_problems()`, `iso_has_problems()`, `iso_problem_functions`

### iso_find_absolute_path_roots

Find roots for absolute paths

**Description**

Helper function to find the roots of absolute paths. Tries to put absolute paths into the context of the relative root. For those that this is not possible (because they are not in fact a sub-path of the relative roots), identifies the greatest common denominator for absolute paths as their root. Does not change relative paths but does check whether they do exist if `check_existence = TRUE` (the default). To modify relative paths, use `iso_shorten_relative_paths` prior to calling this function.
Usage

iso_find_absolute_path_roots(path, root = ".", check_existence = TRUE)

Arguments

- `path`: vector of file/folder paths, mixed relative and absolute paths are allowed.
- `root`: root directory for the isofiles. Can be relative to the current working directory (e.g. "data") or an absolute path on the file system (e.g. "/Users/..." or "C:/Data/.."). The default is the current working directory ("."). Can be supplied as a vector of same length as the provided paths if the paths have different roots.
- `check_existence`: whether to check for the existence of the paths

Value

a data frame with the root directories and paths relative to the root - order of input paths is preserved

See Also

Other file system functions: `iso_expand_paths()`, `iso_root_paths()`, `iso_shorten_relative_paths()`

---

**iso_format**  
*Format values*

Description

Convenience function to easily format and concatenate text and numeric values. Can be used with any text and number data. Automatically detects `iso_with_units` values and incorporates the units into the formatting.

Usage

```r
iso_format(
  ..., 
  signif = 3, 
  format_names = "%s: ", 
  format_units = "%s", 
  replace_permil = TRUE, 
  sep = "\n" 
)
```
iso_get_all_data

Aggregate all isofiles data

Description

This function aggregates all isofiles data and returns it in a large data frame with nested columns for each type of information (file_info, raw_data, etc.). For targeted retrieval of specific data iso_get_raw_data, iso_get_file_info, iso_get_vendor_data_table, etc. are much faster and easier to work with. This function is primarily useful for downstream processing pipelines that want to carry all information along. To unnest any of the specific data types (e.g. raw_data), make sure to filter first for the files that have this data type available (e.g. filter(has_raw_data)). Exclude specific types of information by setting its include... parameter to NULL (Note: for historical reasons, setting it to FALSE will also include the information).

Usage

iso_get_all_data(
  iso_files,
  include_file_info = everything(),
  include_raw_data = everything(),
  include_standards = everything(),
  include_resistors = everything(),
  include_vendor_data_table = everything(),
  include_problems = NULL,
  gather = FALSE,
  with_explicit_units = with_units,
  with_units = FALSE,
iso_get_all_data

with_ratios = NULL,
quiet = default(quiet)
)

Arguments

iso_files collection of iso_file objects

include_file_info
which file information to include (see iso_get_file_info). Use c(...) to
select multiple, supports all select syntax including renaming columns.

include_raw_data
which columns from the raw data to include. Use c(...) to select multiple
columns, supports all select syntax including renaming columns. Includes all
columns by default. Set to NULL to include no raw data.

include_standards
which columns from the standards info to include. Use c(...) to select multiple
columns, supports all select syntax including renaming columns. By default,
everything is included (both standards and ratios). To omit the ratios, change to
select = file_id:reference. Set to NULL to include no standards info.

include_resistors
which columns from the resistors info to include. Use c(...) to select multiple
columns, supports all select syntax including renaming columns. Includes all
columns by default. Set to NULL to include no resistors info.

include_vendor_data_table
which columns from the vendor data table to include. Use c(...) to select mul-
tiple columns, supports all select syntax including renaming columns. Includes
all columns by default. Set parameter with_explicit_units = TRUE to make
column units explicit (keep in mind that this will require specific include_vendor_data_table
column selections to reflect the column names including the units). Set to NULL
to include no vendor data table.

include_problems
which columns from problems to include. Use c(...) to select multiple columns,
supports all select syntax including renaming columns. Includes none of the read
problems by default. Set to include_problems = everything() to include all
columns.

gather
whether to gather raw data into long format (e.g. for ease of use in plotting).
Not that the select parameter applies to the data columns BEFORE gathering.

with_explicit_units
whether to include units in the column headers of the returned data frame instead
of the column data types (see iso_double_with_units). Note that any select
conditions have to refer to the column names including the full units.

with_units this parameter has been DEPRECATED with the introduction of unit-data types
(see iso_double_with_units) and will be removed in future versions of isoreader. Please use with_explicit_units instead if you really want columns
to have units explicitly in the column name. Alternatively, consider working
with the new implicit unit system and convert vendor data tables as needed with
iso_make_units_explicit and iso_make_units_implicit.
iso_get_bgrd_data

with_ratios deprecated, please use the select parameter to explicitly include or exclude ratio columns
quiet whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off

Value
data_frame with file_ids, file_types and nested data frames for each data type (file_info, raw_data, vendor_data_table, etc.)

See Also
Other data retrieval functions: iso_get_bgrd_data(), iso_get_file_info(), iso_get_raw_data(), iso_get_resistors(), iso_get_standards(), iso_get_vendor_data_table()

iso_get_bgrd_data Aggregate background data

Description
Aggregate the background data from the provided iso_files. Can aggregate either in a wide table (for easy overview) or a gathered long table (for plotting and further data processing). The background data is only available if the iso_files were read with parameter read_raw_data=TRUE.

Usage
iso_get_bgrd_data(
  iso_files,
  select = everything(),
  gather = FALSE,
  include_file_info = NULL,
  quiet = default(quiet)
)

Arguments
iso_files collection of iso_file objects
select which data columns to select - use c(...) to select multiple, supports all select syntax. By default, all columns are selected.
gather whether to gather raw data into long format (e.g. for ease of use in plotting). Not that the select parameter applies to the data columns BEFORE gathering.
include_file_info which file information to include (see iso_get_file_info). Use c(...) to select multiple, supports all select syntax including renaming columns.
quiet whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off
iso_get_data

See Also

Other data retrieval functions: iso_get_all_data(), iso_get_file_info(), iso_get_raw_data(), iso_get_resistors(), iso_get_standards(), iso_get_vendor_data_table()

iso_get_data  DEPRECATED

Description

Please use iso_get_all_data instead.

Usage

iso_get_data(...)

Arguments

... forwarded to iso_get_all_data

iso_get_data_summary  Get data summary

Description

Summarize the data information from one or multiple iso files.

Usage

iso_get_data_summary(iso_files, quiet = default(quiet))

Arguments

iso_files single iso file or collection of iso_file objects
quiet whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off

Value

a tibble that summarizes the data in the iso_files
iso_get_default_reader_parameters

*Get the current default parameters*

**Description**

Retrieve a table with all default function parameters for this package. To set read parameters, see *iso_set_default_read_parameters*. To set messaging and caching parameters see *iso_info_messages* and see *iso_caching*.

**Usage**

```r
iso_get_default_reader_parameters()
```

**See Also**

Other settings functions: *iso_caching*, *iso_info_messages*, *iso_set_default_read_parameters*()
**iso_get_problems**

Retrieve parsing problems

**Description**

This function retrieves parsing problems encountered during the reading of a set of iso files.

**Usage**

`iso_get_problems(iso_files, select = everything())`

**Arguments**

- `iso_files` collection of iso_file objects
- `select` which data columns to select - use `c(...)` to select multiple, supports all `select` syntax. By default, all columns are selected.

**See Also**

Other problem functions: `iso_filter_files_with_problems()`, `iso_get_problems_summary()`, `iso_has_problems()`, `iso_problem_functions`
iso_get_problems_summary

*Retrieve a summary of the problems*

**Description**

Returns a data frame listing how many errors and warnings were encountered for each file. For details on each error/warning, see `problems` and the `iso_problem_functions`.

**Usage**

```r
iso_get_problems_summary(
  iso_files,
  problem_files_only = TRUE,
  include_file_info = NULL
)
```

**Arguments**

- `iso_files`: collection of `iso_file` objects
- `problem_files_only`: whether to list only problem files or all files
- `include_file_info`: which file information to include (see `iso_get_file_info`). Use `c(...)` to select multiple, supports all `select` syntax including renaming columns.

**Value**

data frame with file_id and number of encountered errors and warnings

**See Also**

Other problem functions: `iso_filter_files_with_problems()`, `iso_get_problems()`, `iso_has_problems()`, `iso_problem_functions`

---

iso_get_raw_data

*Aggregate raw data*

**Description**

Aggregate the raw ion data from the provided `iso_files`. Can aggregate either in a wide table (for easy overview) or a gathered long table (for plotting and further data processing). The raw data is only available if the `iso_files` were read with parameter `read_raw_data=TRUE`. 

---
Usage

```r
iso_get_raw_data(
  iso_files,
  select = everything(),
  gather = FALSE,
  include_file_info = NULL,
  quiet = default(quiet)
)
```

Arguments

- `iso_files` collection of iso_file objects
- `select` which data columns to select - use `c(...)` to select multiple, supports all `select` syntax. By default, all columns are selected.
- `gather` whether to gather raw data into long format (e.g. for ease of use in plotting). Not that the select parameter applies to the data columns BEFORE gathering.
- `include_file_info` which file information to include (see `iso_get_file_info`). Use `c(...)` to select multiple, supports all `select` syntax including renaming columns.
- `quiet` whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to `iso_turn_info_messages_on` and `iso_turn_info_messages_off`

See Also

Other data retrieval functions: `iso_get_all_data()`, `iso_get_bgrd_data()`, `iso_get_file_info()`, `iso_get_resistors()`, `iso_get_standards()`, `iso_get_vendor_data_table()`

Example files

The isoreader package comes with a few example files to make it easy to illustrate the functionality.

Usage

```r
iso_get_reader_example(filename)
iso_get_reader_examples()
iso_get_reader_examples_folder()
```
iso_get_resistors

Arguments

filename the name of the example file for which to retrieve the system path

Details

iso_get_reader_example: retrieve the path to an isoreader example file
iso_get_reader_examples: list of all available isoreader example files
iso_get_reader_examples_folder: path to the location of the reader examples

Examples

iso_get_reader_examples()
iso_get_reader_examples_folder()

iso_get_resistors  Aggregate resistors from methods info

Description

Aggregates the resistor information recovered from the provided iso_files. This information is only available if the iso_files were read with parameter read_method_info=TRUE and only linked to specific masses if the iso_files were additionally read with parameter read_raw_data=TRUE.

Usage

iso_get_resistors(
  iso_files,
  select = everything(),
  include_file_info = NULL,
  quiet = default(quiet)
)

Arguments

iso_files collection of iso_file objects
select which data columns to select - use c(...) to select multiple, supports all select syntax. By default, all columns are selected.
include_file_info which file information to include (see iso_get_file_info). Use c(...) to select multiple, supports all select syntax including renaming columns.
quiet whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off
See Also

Other data retrieval functions: `iso_get_all_data()`, `iso_get_bgrd_data()`, `iso_get_file_info()`, `iso_get_raw_data()`, `iso_get_standards()`, `iso_get_vendor_data_table()`

---

### iso_get_resistors_info

*DEPRECATED*

**Description**

Please use `iso_get_resistors` instead.

**Usage**

```
iso_get_resistors_info(...)  
```

**Arguments**

... forwarded to `iso_get_resistors`

---

### iso_get_standards

*Aggregate standards from methods info*

**Description**

Aggregates the isotopic standard information recovered from the provided `iso_files`. Can aggregate just the standards’ delta values or combine the delta values with the recovered ratios (if any). Use parameter `select` to exclude/include the ratios. All standards info is only available if the `iso_files` were read with parameter `read_method_info=TRUE`.

**Usage**

```
iso_get_standards(
    iso_files,  
    select = everything(),  
    include_file_info = NULL,  
    with_ratios = NULL,  
    quiet = default(quiet)  
)  
```
iso_get_standards_info

Arguments

iso_files  collection of iso_file objects

select  which data columns to select - use c(...) to select multiple, supports all select syntax. By default, everything is included (both standards and ratios). To omit the ratios, change to select = file_id:reference.

include_file_info  which file information to include (see iso_get_file_info). Use c(...) to select multiple, supports all select syntax including renaming columns.

with_ratios  deprecated, please use the select parameter to explicitly include or exclude ratio columns

quiet  whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off

See Also

Other data retrieval functions: iso_get_all_data(), iso_get_bgrd_data(), iso_get_file_info(), iso_get_raw_data(), iso_get_resistors(), iso_get_vendor_data_table()

iso_get_standards_info

DEPRECATED

Description

Please use iso_get_standards instead.

Usage

iso_get_standards_info(...)
iso_get_supported_file_types

Supported file types

Description

Get an overview of all the file types currently supported by the isoreader package. To register additional file readers, use the iso_register_dual_inlet_file_reader and iso_register_continuous_flow_file_reader functions.

Usage

iso_get_supported_file_types()

See Also

Other file_types: iso_register_dual_inlet_file_reader()

iso_get_units

Retrieve number units

Description

This function returns the units of a numerical value generated by iso_double_with_units. It returns NA) for unitless variables. Returns a column-named vector of units if x is a data frame / tibble. Returns the direct units of x in all other cases.

Usage

iso_get_units(x)

Arguments

x variable to get the units for (vector or data frame)

See Also

Other functions for values with units: iso_is_double_with_units(), iso_make_units_explicit(), iso_make_units_implicit(), iso_strip_units(), iso_with_units()
iso_get_vendor_data_table

Aggregate vendor computed table data

Description

Aggregate data from the vendor-computed data table. This information is only available if the iso_files were read with parameter read_vendor_data_table=TRUE.

Usage

iso_get_vendor_data_table(
  iso_files,
  with_units = FALSE,
  select = everything(),
  include_file_info = NULL,
  with_explicit_units = with_units,
  quiet = default(quiet)
)

Arguments

iso_files  collection of iso_file objects
with_units  this parameter has been DEPRECATED with the introduction of unit-data types (see iso_double_with_units) and will be removed in future versions of isoreader. Please use with_explicit_units instead if you really want columns to have units explicitly in the column name. Alternatively, consider working with the new implicit unit system and convert vendor data tables as needed with iso_make_units_explicit and iso_make_units_implicit.
select  which data columns to select - use c(...) to select multiple, supports all select syntax. By default, all columns are selected.
include_file_info  which file information to include (see iso_get_file_info). Use c(...) to select multiple, supports all select syntax including renaming columns.
with_explicit_units  whether to include units in the column headers of the returned data frame instead of the column data types (see iso_double_with_units). Note that any select conditions have to refer to the column names including the full units.
quiet  whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off

See Also

Other data retrieval functions: iso_get_all_data(), iso_get_bgrd_data(), iso_get_file_info(), iso_get_raw_data(), iso_get_resistors(), iso_get_standards()
iso_has_problems

Check for parsing problems

Description
Check for parsing problems

Usage
iso_has_problems(iso_files)

Arguments
iso_files    collection of iso_file objects

Value
boolean

See Also
Other problem functions: iso_filter_files_with_problems(), iso_get_problems_summary(), iso_get_problems(), iso_problem_functions

iso_info_messages

Control information messages

Description
These functions control the global settings for information messages.

Usage
iso_turn_info_messages_on(data = NULL)
iso_turn_info_messages_off(data = NULL)
iso_turn_datetime_warnings_on(data = NULL)
iso_turn_datetime_warnings_off(data = NULL)

Arguments
data    a data frame - returned invisibly as is if provided (e.g. in the middle of a pipeline)
Details

iso_turn_info_messages_on() and iso_turn_info_messages_off() turn information messages on/off in all subsequent function calls by changing the global settings for the quiet parameter of most isoreader functions. These functions can be called stand alone or within a pipeline to turn messages on/off at a certain point during the pipeline.

iso_turn_datetime_warnings_on() and iso_turn_datetime_warnings_off() turn datetime warnings that occur on some platforms (mostly linux distributions) on/off for all subsequent isoreader functions. These warnings inform the user that file creation dates are not available from the operating system.

See Also

Other settings functions: iso_caching, iso_get_default_reader_parameters(), iso_set_default_read_parameters()

---

iso_is_double_with_units

*Check if a value has units*

Description

Check if a variable is a double with units. That is if it has been generated by `iso_double_with_units`.

Usage

iso_is_double_with_units(x)

Arguments

x vector to check for whether it is a double with units

See Also

Other functions for values with units: iso_get_units(), iso_make_units_explicit(), iso_make_units_implicit(), iso_strip_units(), iso_with_units()
**iso_is_file**

*Isoreader data structure functions*

---

**Description**

- **iso_is_file** tests if the object is an iso_file
- **iso_is_file_list** tests if the object is an iso_file list (collection of iso_files)
- **iso_is_object** test if the object is an iso-object (iso_file or iso_file list)
- **iso_is_dual_inlet** tests if an iso_file or iso_file list consists exclusively of dual inlet file objects
- **iso_is_continuous_flow** tests if an iso_file or iso_file list consists exclusively of continuous flow file objects
- **iso_is_scan** tests if an iso_file or iso_file list consists exclusively of scan file objects
- **iso_as_file_list** concatenates iso_file and iso_file list object(s) into one combined iso_file list (equivalent to calling `c(...)`, flattens all passed lists into one list structure, all individual objects and objects within iso_file lists have to be the same type of iso_file, issues warnings if there are duplicate file ids and summarizes all problems in the iso_file list. If duplicates are allowed (`discard_duplicates = FALSE`), their file IDs will append a #1, #2, #3, etc. to preserve unique file IDs (important for many data aggregation operations).

**Usage**

- **iso_is_file(x)**
- **iso_is_file_list(x)**
- **iso_is_object(x)**
- **iso_is_dual_inlet(x)**
- **iso_is_continuous_flow(x)**
- **iso_is_scan(x)**
- **iso_as_file_list(..., discard_duplicates = TRUE)**

**Arguments**

- **x** an object to test whether it has the specific class
- **...** iso_file and iso_file_list objects to concatenate
- **discard_duplicates** whether to automatically discard files with duplicate file IDs (i.e. duplicate file names). If TRUE (the default), only the first files are kept and any files with the same file ID are discarded. If FALSE, all duplicate files are kept but their file IDs are appended with suffix #1, #2, etc.
iso_make_units_explicit

Make units explicit

Description

This function is intended for data frames / tibbles only and makes the units of columns that have numbers with units explicit in the column name. It also strips the units attribute from those columns using iso_strip_units. The reverse function is iso_make_units_implicit.

Usage

iso_make_units_explicit(df, prefix = " ", suffix = "]")

Arguments

df: the data frame in which to make the units explicit
prefix: the prefix for the units
suffix: the suffix for the units

See Also

Other functions for values with units: iso_get_units(), iso_is_double_with_units(), iso_make_units_implicit(), iso_strip_units(), iso_with_units()

Examples

# a data frame with implicit units
df <- tibble(peak = 1:5, height = iso_double_with_units(1:5, "V"))
df

# show with explicit units
iso_make_units_explicit(df)

# show with explicit units (custom prefix & suffix)
iso_make_units_explicit(df, prefix = ".", suffix = "]")

iso_make_units_implicit

Make units implicit

Description

This function is intended for data frames / tibbles only and tries to figure out which numeric columns have units in the column names and makes those units implicit using iso_double_with_units. The reverse function is iso_make_units_explicit.
Usage

iso_make_units_implicit(df, prefix = "[", suffix = "]")

Arguments

df  the data frame in which to make the units implicit/explicit
prefix  the prefix for the units
suffix  the suffix for the units

See Also

Other functions for values with units: iso_get_units(), iso_is_double_with_units(), iso_make_units_explicit(), iso_strip_units(), iso_with_units()

Examples

# generate implicit units
df <- tibble(peak = 1:5, 'height [V]' = 1:5)
iso_make_units_implicit(df)

# convert back and forth
iso_make_units_implicit(df) %>% iso_make_units_explicit()

# implicit units from custom prefix & suffix
df <- tibble(peak = 1:5, height.V = 1:5)
iso_make_units_implicit(df, prefix = ".", suffix = ")

iso_mutate_file_info  Mutate file info

Description

Mutate the file info (iso_get_file_info) within isofile objects by changing existing columns or introducing new ones. Works just like dplyr's mutate. You can also use mutate directly but it will not provide summary information on the operation. Note that this will create missing columns that exist in some but not all of the passed in isofile objects in all isofile objects (filling them with NAs) the same way that iso_get_file_info does.

Usage

iso_mutate_file_info(iso_files, ..., quiet = default(quiet))

Arguments

iso_files  collection of isofile objects
...  dplyr-style mutate conditions applied to the combined file info (see iso_get_file_info)
quiet  whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off
See Also

Other file_info operations: iso_add_file_info, iso_file_list, iso_filter_files, iso_parse_file_info, iso_rename_file_info, iso_select_file_info, iso_set_file_root

iso_omit_files_with_problems

*Renamed to iso_filter_files_with_problems*

Description

This function has been renamed to iso_filter_files_with_problems for naming consistency.

Usage

iso_omit_files_with_problems(...)

Arguments

... deprecated

iso_parse_file_info  Parse file info

Description

Convenience function to batch parse file info (iso_get_file_info) columns in isofile objects for the most common parsing calls. Uses the parse_ functions exported from readr and described in extract_data. Note that for less common parsing calls or calls that require additional parameters to the parsing function, it is better to parse columns one-by-one using iso_mutate_file_info instead.

Usage

iso_parse_file_info(
    iso_files,
    number = c(),
    double = c(),
    integer = c(),
    logical = c(),
    datetime = c(),
    text = c(),
    quiet = default(quiet)
)
iso_plot_continuous_flow_data

**Arguments**

- `iso_files` collection of `iso_file` objects
- `number` dplyr-style `select` condition to choose columns that should be converted to a number using `parse_number`. Use `c(...)` to select multiple columns.
- `double` dplyr-style `select` condition to choose columns that should be converted to a double using `parse_double`. Use `c(...)` to select multiple columns.
- `integer` dplyr-style `select` condition to choose columns that should be converted to an integer using `parse_integer`. Use `c(...)` to select multiple columns.
- `logical` dplyr-style `select` condition to choose columns that should be converted to a boolean (TRUE/FALSE) using `parse_logical`. Use `c(...)` to select multiple columns.
- `datetime` dplyr-style `select` condition to choose columns that should be converted to a date-time using `parse_datetime`. Use `c(...)` to select multiple columns.
- `text` dplyr-style `select` condition to choose columns that should be converted to text using `as.character`. Use `c(...)` to select multiple columns.
- `quiet` whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to `iso_turn_info_messages_on` and `iso_turn_info_messages_off`

**See Also**

Other file info operations: `iso_add_file_info`, `iso_file_list`, `iso_filter_files`, `iso_mutate_file_info`, `iso_rename_file_info`, `iso_select_file_info`, `iso_set_file_root`

---

iso_plot_continuous_flow_data

*moved to isoprocessor*

**Description**

moved to isoprocessor

**Usage**

`iso_plot_continuous_flow_data(...)`

**Arguments**

... deprecated
iso_plot_dual_inlet_data

moved to isoprocessor

Description
moved to isoprocessor

Usage
iso_plot_dual_inlet_data(...)

Arguments
... deprecated

iso_plot_raw_data

moved to isoprocessor

Description
moved to isoprocessor

Usage
iso_plot_raw_data(...)

Arguments
... deprecated

iso_problem_functions

Problem Functions Overview

Description
The following functions to check for and deal with problems are available.

Details

- iso_get_problems is a re-export of problems
- iso_get_problems_summary
- iso_has_problems
- stop_for_problems
- iso_filter_files_with_problems
iso_read_continuous_flow

Load continuous flow data

Description

Load continuous flow data

Usage

iso_read_continuous_flow(
  ..., 
  root = ".",
  read_raw_data = default(read_raw_data),
  read_file_info = default(read_file_info),
  read_method_info = default(read_method_info),
  read_vendor_data_table = default(read_vendor_data_table),
  discard_duplicates = TRUE,
  parallel = FALSE,
  parallel_plan = future::multisession,
  parallel_cores = future::availableCores(),
  cache = default(cache),
  read_cache = default(cache),
  reread_outdated_cache = FALSE,
  quiet = default(quiet),
  cache_files_with_errors = TRUE
)

Arguments

... one or multiple file/folder paths. All files must have a supported file extension. All folders are expanded and searched for files with supported file extensions (which are then included in the read).
root root directory for the isofiles. Can be relative to the current working directory (e.g. "data") or an absolute path on the file system (e.g. "/Users/..." or "C:/Data/...".). The default is the current working directory (".".). Can be supplied as a vector of same length as the provided paths if the paths have different roots.
read_raw_data whether to read the raw mass/ion data from the file
read_file_info whether to read auxiliary file information (file id, sequence information, etc.)
read_method_info whether to read methods information (standards, processing info)

See Also

Other problem functions: iso_filter_files_with_problems(), iso_get_problems_summary(), iso_get_problems(), iso_has_problems()
read_vendor_data_table
whether to read the vendor computed data table

discard_duplicates
whether to automatically discard files with duplicate file IDs (i.e. duplicate file names). If TRUE (the default), only the first files are kept and any files with the same file ID are discarded. If FALSE, all duplicate files are kept but their file IDs are appended with suffix #1, #2, etc.

parallel
whether to process in parallel based on the number of available CPU cores. This may yield performance increases for files that are slow to parse such as continuous flow isodat files but usually provides little benefit for efficient data formats such as reading from R Data Archives.

parallel_plan which parallel processing strategy to use, see plan, typically future::multisession for compatibility with RStudio interactive mode. If supported by the operating system and running in detached mode (not interactively in RStudio) can also use future::multicore.

parallel_cores how many processor cores to use for parallel processing. By default the maximum available number of cores (availableCores), which will allow maximal processing speed but may slow other programs running on your machine. Choose a smaller number if you want some processing resources to remain available for other processes. Will issue a warning if too many cores are requested and reset to the maximum available.

cache whether to cache iso_files. Note that R Data Storage files (.rds, see iso_save) are never cached since they are already essentially in cached form.

read_cache whether to reload from cache if a cached version exists. Note that it will only read from cache if the raw data file has not been modified since. Files that have been modified on disc (e.g. edited in the vendor software) will always be read anew. To automatically reread cached files that were cached by an outdated version of the isoreader package, set the rread_outdated_cache flag.

rread_outdated_cache whether to re-read outdated cache files whenever they are encountered.

quiet whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off

cache_files_with_errors deprecated. Please use iso_reread_problem_files instead to selectively re-read all files in a collection of iso files that had been previously read with errors or warnings.

See Also

Other isoread functions for different types of IRMS data: iso_read_dual_inlet(), iso_read_scan()
iso_read_dual_inlet  

Load dual inlet data

Description

Load dual inlet data

Usage

iso_read_dual_inlet(
  ..., 
  root = ".", 
  read_raw_data = default(read_raw_data), 
  read_file_info = default(read_file_info), 
  read_method_info = default(read_method_info), 
  read_vendor_data_table = default(read_vendor_data_table), 
  nu_masses = c(), 
  discard_duplicates = TRUE, 
  parallel = FALSE, 
  parallel_plan = future::multisession, 
  parallel_cores = future::availableCores(), 
  cache = default(cache), 
  reread_outdated_cache = FALSE, 
  quiet = default(quiet), 
  cache_files_with_errors = TRUE
)

Arguments

...  one or multiple file/folder paths. All files must have a supported file extension. All folders are expanded and searched for files with supported file extensions (which are then included in the read).

root  root directory for the isofiles. Can be relative to the current working directory (e.g. "data") or an absolute path on the file system (e.g. "/Users/..." or "C:/Data/..."). The default is the current working directory ("."). Can be supplied as a vector of same length as the provided paths if the paths have different roots.

read_raw_data  whether to read the raw mass/ion data from the file

read_file_info  whether to read auxiliary file information (file id, sequence information, etc.)

read_method_info  whether to read methods information (standards, processing info)

read_vendor_data_table  whether to read the vendor computed data table
iso_read_dual_inlet

nu_masses  list of masses (e.g. `c("46","45","44")`) to map the collector channels (interpreted in order, i.e. the first channel will be linked to the first mass, the second channel to the second mass, etc.). This parameter is only used for reading Nu data files.

discard_duplicates  whether to automatically discard files with duplicate file IDs (i.e. duplicate file names). If `TRUE` (the default), only the first files are kept and any files with the same file ID are discarded. If `FALSE`, all duplicate files are kept but their file IDs are appended with suffix #1, #2, etc.

parallel  whether to process in parallel based on the number of available CPU cores. This may yield performance increases for files that are slow to parse such as continuous flow isodat files but usually provides little benefit for efficient data formats such as reading from R Data Archives.

parallel_plan  which parallel processing strategy to use, see `plan`, typically `future::multisession` for compatibility with RStudio interactive mode. If supported by the operating system and running in detached mode (not interactively in RStudio) can also use `future::multicore`.

parallel_cores  how many processor cores to use for parallel processing. By default the maximum available number of cores (`availableCores`), which will allow maximal processing speed but may slow other programs running on your machine. Choose a smaller number if you want some processing resources to remain available for other processes. Will issue a warning if too many cores are requested and reset to the maximum available.

cache  whether to cache iso_files. Note that R Data Storage files (.rds, see `iso_save`) are never cached since they are already essentially in cached form.

read_cache  whether to reload from cache if a cached version exists. Note that it will only read from cache if the raw data file has not been modified since. Files that have been modified on disc (e.g. edited in the vendor software) will always be read anew. To automatically reread cached files that were cached by an outdated version of the isoreader package, set the `reread_outdated_cache` flag.

reread_outdated_cache  whether to re-read outdated cache files whenever they are encountered.

quiet  whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to `iso_turn_info_messages_on` and `iso_turn_info_messages_off`.

cache_files_with_errors  deprecated. Please use `iso_reread_problem_files` instead to selectively re-read all files in a collection of iso files that had been previously read with errors or warnings.

See Also

Other isoread functions for different types of IRMS data: `iso_read_continuous_flow()`, `iso_read_scan()`
iso_read_files

Core function to read isotope data files

Description

This function takes care of extracting basic information about iso_files, dealing with problems and making sure only valid file formats are processed. This function is not typically called directly but indirectly by calling iso_read_dual_inlet, iso_read_continuous_flow and iso_read_scan. It is made available outside the package because it can be very useful for testing new file readers.

Usage

iso_read_files(
  paths,
  root,
  supported_extensions,
  data_structure,
  read_options = c(),
  reader_options = list(),
  discard_duplicates = TRUE,
  cache_files_with_errors = TRUE,
  parallel = FALSE,
  parallel_plan = future::multisession,
  parallel_cores = future::availableCores(),
  cache = default(cache),
  read_cache = default(cache),
  reread_outdated_cache = FALSE,
  quiet = default(quiet)
)

Arguments

paths one or multiple file/folder paths. All files must have a supported file extension. All folders are expanded and searched for files with supported file extensions (which are then included in the read). Paths can be absolute paths or relative to the provided file root (which is the current working directory by default). For absolute paths, a common root directory will be guessed using iso_find_absolute_path_roots. The root portion of paths will never be displayed in info messages.

root root directory for the isofiles. Can be relative to the current working directory (e.g. "data") or an absolute path on the file system (e.g. "/Users/..." or "C:/Data/..."). The default is the current working directory ("."). Can be supplied as a vector of same length as the provided paths if the paths have different roots.

supported_extensions data frame with supported extensions and corresponding reader functions (columns 'extension', 'func', 'cacheable')
**iso_read_files**

- **data_structure**: the basic data structure for the type of iso_file
- **read_options**: vector of read options to be stored in the data structure (e.g. `c(read_vendor_data_table = FALSE)`). The read_prefix is optional.
- **reader_options**: list of parameters to be passed on to the reader
- **discard_duplicates**: whether to automatically discard files with duplicate file IDs (i.e. duplicate file names). If TRUE (the default), only the first files are kept and any files with the same file ID are discarded. If FALSE, all duplicate files are kept but their file IDs are appended with suffix #1, #2, etc.
- **cache_files_with_errors**: deprecated. Please use `iso_reread_problem_files` instead to selectively re-read all files in a collection of iso files that had been previously read with errors or warnings.
- **parallel**: whether to process in parallel based on the number of available CPU cores. This may yield performance increases for files that are slow to parse such as continuous flow isodat files but usually provides little benefit for efficient data formats such as reading from R Data Archives.
- **parallel_plan**: which parallel processing strategy to use, see `plan`, typically `future::multisession` for compatibility with RStudio interactive mode. If supported by the operating system and running in detached mode (not interactively in RStudio) can also use `future::multicore`.
- **parallel_cores**: how many processor cores to use for parallel processing. By default the maximum available number of cores (`availableCores`), which will allow maximal processing speed but may slow other programs running on your machine. Choose a smaller number if you want some processing resources to remain available for other processes. Will issue a warning if too many cores are requested and reset to the maximum available.
- **cache**: whether to cache iso_files. Note that R Data Storage files (.rds, see `iso_save`) are never cached since they are already essentially in cached form.
- **read_cache**: whether to reload from cache if a cached version exists. Note that it will only read from cache if the raw data file has not been modified since. Files that have been modified on disc (e.g. edited in the vendor software) will always be read anew. To automatically reread cached files that were cached by an outdated version of the isoreader package, set the `reread_outdated_cache` flag.
- **reread_outdated_cache**: whether to re-read outdated cache files whenever they are encountered.
- **quiet**: whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to `iso_turn_info_messages_on` and `iso_turn_info_messages_off`.

**Value**

single iso_file object (if single file) or list of iso_files (iso_file_list)
iso_read_scan

Load scan data

Description

Load scan data

Usage

```r
iso_read_scan(
  ...,
  root = ".",
  read_raw_data = default(read_raw_data),
  read_file_info = default(read_file_info),
  read_method_info = default(read_method_info),
  discard_duplicates = TRUE,
  parallel = FALSE,
  parallel_plan = future::multisession,
  parallel_cores = future::availableCores(),
  cache = default(cache),
  read_cache = default(cache),
  reread_outdated_cache = FALSE,
  quiet = default(quiet),
  cache_files_with_errors = TRUE
)
```

Arguments

- `...` one or multiple file/folder paths. All files must have a supported file extension. All folders are expanded and searched for files with supported file extensions (which are then included in the read).
- `root` root directory for the isofiles. Can be relative to the current working directory (e.g. "data") or an absolute path on the file system (e.g. "/Users/..." or "C:/Data/..."). The default is the current working directory ("."). Can be supplied as a vector of same length as the provided paths if the paths have different roots.
- `read_raw_data` whether to read the raw mass/ion data from the file
- `read_file_info` whether to read auxiliary file information (file id, sequence information, etc.)
- `read_method_info` whether to read methods information (standards, processing info)
- `discard_duplicates` whether to automatically discard files with duplicate file IDs (i.e. duplicate file names). If TRUE (the default), only the first files are kept and any files with the same file ID are discarded. If FALSE, all duplicate files are kept but their file IDs are appended with suffix #1, #2, etc.
parallel whether to process in parallel based on the number of available CPU cores. This may yield performance increases for files that are slow to parse such as continuous flow isodat files but usually provides little benefit for efficient data formats such as reading from R Data Archives.

parallel_plan which parallel processing strategy to use, see plan, typically future::multisession for compatibility with RStudio interactive mode. If supported by the operating system and running in detached mode (not interactively in RStudio) can also use future::multicore.

parallel_cores how many processor cores to use for parallel processing. By default the maximum available number of cores (availableCores), which will allow maximal processing speed but may slow other programs running on your machine. Choose a smaller number if you want some processing resources to remain available for other processes. Will issue a warning if too many cores are requested and reset to the maximum available.

cache whether to cache iso_files. Note that R Data Storage files (.rds, see iso_save) are never cached since they are already essentially in cached form.

read_cache whether to reload from cache if a cached version exists. Note that it will only read from cache if the raw data file has not been modified since. Files that have been modified on disc (e.g. edited in the vendor software) will always be read anew. To automatically reread cached files that were cached by an outdated version of the isoreader package, set the rread_outdated_cache flag.

rread_outdated_cache whether to re-read outdated cache files whenever they are encountered.

quiet whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off.

See Also

Other isoread functions for different types of IRMS data: iso_read_continuous_flow(), iso_read_dual_inlet()

Description

Register file extensions and reader functions for different data files. Isoreader automatically registers all built-in file readers so this function is usually only needed when registering additional readers provided for testing purposes from outside of the isoreader package. Note that file extensions are case-insensitive, i.e. a reader for .ext will also recognize .Ext and .EXT.
iso_register_dual_inlet_file_reader

Usage

iso_register_dual_inlet_file_reader(
  extension,
  func,
  description = NA_character_,
  software = NA_character_,
  cacheable = TRUE,
  post_read_check = TRUE,
  overwrite = FALSE,
  env = find_func(func)
)

iso_register_continuous_flow_file_reader(
  extension,
  func,
  description = NA_character_,
  software = NA_character_,
  cacheable = TRUE,
  post_read_check = TRUE,
  overwrite = FALSE,
  env = find_func(func)
)

iso_register_scan_file_reader(
  extension,
  func,
  description = NA_character_,
  software = NA_character_,
  cacheable = TRUE,
  post_read_check = TRUE,
  overwrite = FALSE,
  env = find_func(func)
)

Arguments

extension the file extension (e.g. .dxf) of the data file. Must be unique otherwise different files can not automatically be matched with the appropriate file reader based on their extension.

func the name of the function that should be used a filter reader. All file reader functions must accept a data structure argument as the first argument and return the same data structure with added data.

description what is this file type about?

software what is the software program that creates this file type?

cacheable whether this file type is cacheable. If TRUE (the default), user requests to cache the file will be honored. If FALSE, this file type will never be cached no matter what the user requests.
post_read_check

whether isoreader should conduct a data integrity check after reading the file. Should always be TRUE unless there is independent data integrity checking already taking place inside the reader.

overwrite

whether to overwrite an existing file reader for the same extension

env

the environment where to find the function, by default this will be determined automatically and will throw an error if there is any ambiguity (e.g. the same function name in multiple packages) in which case it should be set manually

Details

iso_register_dual_inlet_file_reader: use this function to register file readers for dual inlet files.

iso_register_continuous_flow_file_reader: use this function to register file readers for continuous flow files.

iso_register_scan_file_reader: use this function to register file readers for scan files.

See Also

Other file_types: iso_get_supported_file_types()

iso_rename_file_info

Rename file info columns

Description

Rename file info columns (iso_get_file_info) within isofile objects. Works just like dplyr’s rename. You can also use rename directly but it will not provide summary information on the operation. To select specific columns to keep (discarding all others), use iso_select_file_info instead. Set file_specific = TRUE to rename different columns in different isofile files depending on what exists in each file. This is very useful when working with data from multiple instruments that may have the same information (e.g. sample name) stored in different columns.

Usage

iso_rename_file_info(
  iso_files,
  ...,
  file_specific = FALSE,
  quiet = default(quiet)
)
iso_reread_files

Arguments

iso_files collection of iso_file objects

... dplyr-style rename conditions applied based on each file’s file_info (see iso_get_file_info)

file_specific whether to run the select criteria (...) specifically within each individual file rather than on all files jointly. This is a lot slower but makes it possible to select different columns in different iso_files depending on what exists in each file and is mostly of use when working with data from multiple instruments.

quiet whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off

See Also

Other file_info operations: iso_add_file_info, iso_file_list(), iso_filter_files(), iso_mutate_file_info(), iso_parse_file_info(), iso_select_file_info(), iso_set_file_root()

iso_reread_files Re-read iso_files

Description

Sometimes it is useful to reload isotope files from their original data files (e.g. after modifying raw data files in vendor software, or after upgrading to a newer version of the isoreader package that provides new functionality). The functions described below are intended to make this very easy. However, re-reading files from disc is only possible if file paths still point to the original raw data files. If they have moved, please use iso_set_file_root first to change the root directory of your iso_files.

Usage

iso_reread_files(iso_files, ...)

iso_reread_all_files(
  iso_files,
  ...
  stop_if_missing = FALSE,
  quiet = default(quiet)
)

iso_reread_changed_files(
  iso_files,
  ...
  stop_if_missing = FALSE,
  quiet = default(quiet)
)
iso_reread_outdated_files(
  iso_files,
  ..., 
  stop_if_missing = FALSE, 
  quiet = default(quiet) 
)

iso_reread_problem_files(
  iso_files, 
  ..., 
  stop_if_missing = FALSE, 
  reread_files_with_errors = TRUE, 
  reread_files_with_warnings = FALSE, 
  quiet = default(quiet) 
)

iso_reread_storage(...)

iso_reread_archive(...)

Arguments

iso_files  collection of iso_files
...
additional read parameters that should be used for re-reading the iso_files, see iso_read_dual_inlet, iso_read_continuous_flow and iso_read_scan for details (except read_cache which is always set to FALSE to force re-reads).

stop_if_missing
whether to stop re-reading if any of the original data files are missing (if FALSE, will warn about the missing files adding a warning to them, but also re-read those that do exist)

quiet
whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off

reread_files_with_errors
whether to re-read files that had read in with errors the last time (default TRUE)

reread_files_with_warnings
whether to re-read files that had read in with warnings the last time (default TRUE)

Details

To re-read files that have been modified on disc, please use iso_reread_changed_files(). To re-read files because of an isoreader version upgrade, please use iso_reread_outdated_files(). To try re-reading files that previously had warnings and/or errors, please use iso_reread_problem_files().

iso_reread_all_files re-reads all files in the collection.

iso_reread_changed_files re-reads all files that have been modified (e.g. in the vendor software) since they were last read by isoreader.
iso_root_paths re-reads all files that were read with an outdated version of isoreader.

iso_reread_outdated_files re-reads all files that have had errors the last time they were read by isoreader (set reread_files_with_warnings = TRUE to also re-read those that have warnings).

iso_reread_problem_files is deprecated.

Examples

# example for re-reading a saved isofile collection
iso_turn_reader_caching_off()

saved_files_path <- "saved_isofile.scan.rds"

# create saved collection
iso_get_reader_examples_folder() %>%
  iso_read_scan() %>%
  iso_save(saved_files_path)

# load collection
iso_read_scan(saved_files_path) %>%
  # reread outdated files (alternatively ".all_" or ".changed_"
  iso_reread_outdated_files() %>%
  # re-save collection to its original location
  iso_save(saved_files_path)

# cleanup
unlink(saved_files_path)

iso_root_paths(path, root = ".", check_existence = TRUE)

Arguments

path vector of file/folder paths, mixed relative and absolute paths are allowed.
root

root directory for the isofiles. Can be relative to the current working directory (e.g. "data") or an absolute path on the file system (e.g. "/Users/..." or "C:/Data/..."). The default is the current working directory ("."). Can be supplied as a vector of same length as the provided paths if the paths have different roots.

check_existence

whether to check for the existence of the paths

Value

a data frame with the root directories and paths relative to the root - order of input paths is preserved

See Also

Other file system functions: iso_expand_paths(), iso_find_absolute_path_roots(), iso_shorten_relative_paths()

iso_save

Save data to R Data Storage (.rds)

Description

This function saves the passed in iso_files to an R Data Storage (.rds) file, which is an efficient compressed data storage format. Data exported this way can be easily read back into isoreader using the standard iso_read_continuous_flow and iso_read_dual_inlet functions.

Usage

iso_save(iso_files, filepath, quiet = default(quiet))

Arguments

iso_files collection of iso_file objects
filepath the path (folder and filename) to the export file. The correct file extension is automatically added if not already in the filename, i.e. filename can be provided with or without extension.
quiet whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off

Value

returns the iso_files object invisibly for use in pipelines

See Also

Other export functions: iso_export_to_excel(), iso_export_to_feather()
iso_select_file_info  Select file info columns

Description

Select which file info columns (iso_get_file_info) to keep within isofile objects. Works just like dplyr’s select and can rename columns on-the-fly. You can also use select directly but it will not provide summary information on the operation. To rename columns without removing all other information, use iso_rename_file_info instead. Set file_specific = TRUE to select different columns in different iso_files depending on what exists in each file. This is very useful when working with data from multiple instruments that may have the same information (e.g. sample name) stored in different columns.

Usage

iso_select_file_info(
  iso_files,
  ..., 
  file_specific = FALSE, 
  quiet = default(quiet)
)

Arguments

iso_files  collection of iso_file objects
...
  dplyr-style select conditions applied based on each file’s file_info (see iso_get_file_info). Note that the file_id column will always be kept, no matter the selection criteria, and cannot be renamed to protect from unexpected behavior.

file_specific  whether to run the select criteria (...) specifically within each individual file rather than on all files jointly. This is a lot slower but makes it possible to select different columns in different iso_files depending on what exists in each file and is mostly of use when working with data from multiple instruments.

quiet  whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off

See Also

Other file_info operations: iso_add_file_info(), iso_file_list(), iso_filter_files(), iso_mutate_file_info(), iso_parse_file_info(), iso_rename_file_info(), iso_set_file_root()
iso_set_default_read_parameters

Set default read options

Description

Set default read options

Usage

iso_set_default_read_parameters(
  data = NULL,
  read_raw_data,
  read_file_info,
  read_method_info,
  read_vendor_data_table,
  quiet = default(quiet)
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>a data frame - returned invisibly as is if provided (e.g. in the middle of a pipeline)</td>
</tr>
<tr>
<td>read_raw_data</td>
<td>if provided, set as the default for 'read_raw_data' parameters</td>
</tr>
<tr>
<td>read_file_info</td>
<td>if provided, set as the default for 'read_file_info' parameters</td>
</tr>
<tr>
<td>read_method_info</td>
<td>if provided, set as the default for 'read_method_info' parameters</td>
</tr>
<tr>
<td>read_vendor_data_table</td>
<td>if provided, set as the default for 'read_vendor_data_table' parameters</td>
</tr>
<tr>
<td>quiet</td>
<td>whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off</td>
</tr>
</tbody>
</table>

See Also

Other settings functions: iso_caching, iso_get_default_reader_parameters(), iso_info_messages
iso_set_file_root  Set iso file directory root

**Description**

Sets the root directory for a set of iso_files (property file_root in the file information), which is particularly useful for re-reading files (reread_iso_files) after they have changed location. Can optionally remove the previous root (remove_embedded_root) if it is still embedded in the isofiles' file_path instead of file_root. Will warn about any paths that cannot be simplified by removing the embedded root.

**Usage**

```r
iso_set_file_root(
  iso_files,
  root = ".",
  remove_embedded_root = NULL,
  quiet = default(quiet)
)
```

**Arguments**

- `iso_files` collection of iso_file objects
- `root` new root directory for the isofiles. Can be relative to the current working directory (e.g. "data") or an absolute path on the file system (e.g. "/Users/..." or "C:/Data/..."). Can be supplied as a vector of same length as the iso_files if the files have different roots. Use root = "." to set the root to the current working directory (the default).
- `remove_embedded_root` set this parameter to a root path that is embedded in the isofiles’ file_path. Will warn about any paths that cannot be simplified by removing the specified remove_embedded_root.
- `quiet` whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off

**See Also**

Other file_info operations: iso_add_file_info, iso_file_list(), iso_filter_files(), iso_mutate_file_info(), iso_parse_file_info(), iso_rename_file_info(), iso_select_file_info()
iso_shorten_relative_paths

Shorten relative paths

Description

Convenience function to shorten relative paths based on overlap with the provided root(s). Also simplifies current directory repeats (e.g. "././." becomes ".") for better legibility. Does not check whether the original or resulting paths point to valid files or folders. Relative paths that do not start with the supplied root default back to the current working directory (.

Usage

iso_shorten_relative_paths(path, root = ".")

Arguments

- path: vector of file/folder paths, mixed relative and absolute paths are allowed.
- root: root directory for the isofiles. Can be relative to the current working directory (e.g. "data") or an absolute path on the file system (e.g. "/Users/..." or "C:/Data/..."). The default is the current working directory ("."). Can be supplied as a vector of same length as the provided paths if the paths have different roots.

Value

a data frame with the root directories and paths relative to the root - order of input paths is preserved

See Also

Other file system functions: iso_expand_paths(), iso_find_absolute_path_roots(), iso_root_paths()

Examples

iso_shorten_relative_paths(file.path("A", "B", "C"), "A") # root = "A", path = B/C
iso_shorten_relative_paths(file.path("A", "B", "C"), file.path("A", "B")) # root = "A/B", path = "C"
iso_shorten_relative_paths(file.path("A", "C", "D"), file.path("A", "B")) # root = "A", path = "C/D"
iso_shorten_relative_paths(file.path("A", "B", "C"), "B") # root = ".", path stays "A/B/C"
iso_strip_units

Strip units from variables

Description

This function converts numbers with units back into unitless numbers both for single variables and
data frames / tibbles. For single variables, this is equivalent to the as.numeric function.

Usage

iso_strip_units(x)

Arguments

x variable to strip units from (vector or data frame)

See Also

Other functions for values with units: iso_get_units(), iso_is_double_with_units(), iso_make_units_explicit(),
iso_make_units_implicit(), iso_with_units()

iso_with_units

Generate values with units

Description

These functions generate values with units that work well within data frames and tibbles and im-
plement safety checks on operations that combine values with different units. To retrieve the value
without units, use iso_strip_units (works for single variables and data frames/tibbles). To re-
trieve the unit use iso_get_units. Note that to correctly combine data frames / tibbles that have
values with units in them, use vec_rbind instead of rbind or bind_rows. vec_rbind will combine
columns that have values with units if they have the same unit and otherwise convert back to plain
values without units with a warning. The other functions will either fail or reduce the unit values to
plain values with a cryptic warning message about not preserving attributes.

Usage

iso_with_units(x, units = "undefined units")

iso_double_with_units(x = double(), units = "undefined units")

Arguments

x the values (single value or vector)
units the units for the value, by default "undefined units" but this parameter should
always be supplied when working with real data that has units
Details

`iso_with_units` is the primary function to generate values with units. At present, only numeric values are supported so this function is just a shorter alias for the number-specific `iso_double_with_units`. It is not clear yet whether any non-numeric values with units make sense to be supported at a later point or whether integer and decimal numbers should be treated differently when they have units.

See Also

Other functions for values with units: `iso_get_units()`, `iso_is_double_with_units()`, `iso_make_units_explicit()`, `iso_make_units_implicit()`, `iso_strip_units()`

Other functions for values with units: `iso_get_units()`, `iso_is_double_with_units()`, `iso_make_units_explicit()`, `iso_make_units_implicit()`, `iso_strip_units()`

---

**map_binary_structure**

*Map isodat file binary structure.*

---

**Description**

Map out binary structure for easy visualization (used mostly for error messages and debugging). See the development vignette for details and example application.

**Usage**

```r
map_binary_structure(
  bfile,
  length = 100,
  start = bfile$pos,
  ctrl_blocks = get_ctrl_blocks_config()
)
```

**Arguments**

- `bfile` the binary file, stored in each `iso_file` under `$binary` if (and only if) the file was read with `iso_turn_debug_on` activated before.
- `length` how many bytes to map
- `start` at which byte position to start mapping (index 1 based)
- `ctrl_blocks` named list of block patterns with size, regexp and [optional] replace function
print.binary_structure_map

Print binary structure map

Description
Print binary structure map

Usage
## S3 method for class 'binary_structure_map'
print(
  x,
  ..., 
  data_as_raw = FALSE,
  line_break_blocks = c("cblock", "stx", "etx"),
  pos_info = TRUE
)

Arguments
x object to show.
... additional parameters passed to print.default
data_as_raw whether to show data as raw
line_break_blocks at which blocks to introduce a line break
pos_info whether to include position information

print.iso_file_list Isofile printing

Description
Print summary of individual iso_files (dual inlet or continuous flow) or collection of iso_files.

Usage
## S3 method for class 'iso_file_list'
print(x, ...)

## S3 method for class 'iso_file'
print(x, ..., show_problems = TRUE)

## S3 method for class 'dual_inlet'
read_iso_file

print(x, ..., show_problems = TRUE)

## S3 method for class 'continuous_flow'
print(x, ..., show_problems = TRUE)

## S3 method for class 'scan'
print(x, ..., show_problems = TRUE)

Arguments

x Object to show.
...
additional parameters passed to print.default
show_problems whether to show encountered problems

read_iso_file  Read individual iso file

Description

Low level read function for an individual iso file. Usually not called directly but available for methods development.

Usage

read_iso_file(
  ds,
  root,
  path,
  file_n,
  files_n,
  read_from_cache,
  read_from_old_cache,
  reread_outdated_cache,
  write_to_cache,
  cachepath,
  old_cachepath,
  post_read_check,
  ext,
  reader_fun,
  reader_options,
  reader_fun_env
)
reread_iso_files

Arguments

- **ds**: the basic data structure for the type of iso_file
- **root**: root directory for the isofiles. Can be relative to the current working directory (e.g. "data") or an absolute path on the file system (e.g. "/Users/..." or "C:/Data/..."). The default is the current working directory ("."). Can be supplied as a vector of same length as the provided paths if the paths have different roots.
- **path**: file path
- **file_n**: number of processed file for info messages
- **files_n**: total number of files for info messages
- **read_from_cache**: whether to read from cache
- **read_from_old_cache**: whether to read from old cache files (to be deprecated in isoreader 2.0)
- **reread_outdated_cache**: whether to reread outdated cache files
- **write_to_cache**: whether to write to cache
- **cachepath**: path for the cache file
- **old_cachepath**: path for the old cache files
- **post_read_check**: whether to run data integrity checks after a file read
- **ext**: file extension
- **reader_fun**: file reader function
- **reader_options**: list of parameters to be passed on to the reader
- **reader_fun_env**: where to find the reader function

---

reread_iso_files  
Re-read iso_files

Description

Actual multi-purpose file-reread function (not exported) that powers iso_reread_files.

Usage

```r
reread_iso_files(
  iso_files,
  ...,
  stop_if_missing = FALSE,
  reread_only_changed_files = FALSE,
  reread_only_outdated_files = FALSE,
  reread_files_without_problems = TRUE,
  reread_files_with_errors = TRUE,
  reread_files_with_warnings = TRUE,
  quiet = default(quiet)
)
```
Arguments

iso_files  collection of iso_files
...
... additional read parameters that should be used for re-reading the iso_files, see iso_read_dual_inlet, iso_read_continuous_flow and iso_read_scan for details (except read_cache which is always set to FALSE to force re-reads).

stop_if_missing  whether to stop re-reading if any of the original data files are missing (if FALSE, will warn about the missing files adding a warning to them, but also re-read those that do exist)

reread_only_changed_files  whether to re-read only files that have since been changed on disc (i.e. have no valid cache file), default FALSE i.e. re-read ALL files

reread_only_outdated_files  whether to re-read only files that were read by an outdated version of isoreader (default FALSE, i.e. re-read ALL files)

reread_files_without_problems  whether to re-read files that had read in without problems the last time (default TRUE)

reread_files_with_errors  whether to re-read files that had read in with errors the last time (default TRUE)

reread_files_with_warnings  whether to re-read files that had read in with warnings the last time (default TRUE)

quiet  whether to display (quiet=FALSE) or silence (quiet = TRUE) information messages. Set parameter to overwrite global defaults for this function or set global defaults with calls to iso_turn_info_messages_on and iso_turn_info_messages_off

set_temp  Set temporary option

Description

Set a temporary option for parallel processing in isoprocessor.

Usage

set_temp(name, value)

Arguments

name  name of the temporary option

value  value of the temporary option
vec_arith.iso_double_with_units

vec_arith for iso_double_with_units

Description

vec_arith for iso_double_with_units

Usage

## S3 method for class 'iso_double_with_units'
vec_arith(op, x, y, ...)

Arguments

- **op**: An arithmetic operator as a string
- **x**: A pair of vectors. For `!`, unary `+` and unary `-`, `y` will be a sentinel object of class `MISSING`, as created by `MISSING()`.
- **y**: A pair of vectors. For `!`, unary `+` and unary `-`, `y` will be a sentinel object of class `MISSING`, as created by `MISSING()`.
- **...**: These dots are for future extensions and must be empty.

vec_cast.iso_double_with_units

vec_cast for iso_double_with_units

Description

vec_cast for iso_double_with_units

Usage

## S3 method for class 'iso_double_with_units'
vec_cast(x, to, ...)

Arguments

- **x**: Vectors to cast.
- **to**: Type to cast to. If `NULL`, `x` will be returned as is.
- **...**: For `vec_cast_common()`, vectors to cast. For `vec_cast()`, `vec_cast_default()`, and `vec_restore()`, these dots are only for future extensions and should be empty.
Description

vec_ptype2 for iso_double_with_units

Usage

## S3 method for class 'iso_double_with_units'
vec_ptype2(x, y, ...)

Arguments

x  Vector types.
y  Vector types.
...  These dots are for future extensions and must be empty.
Index

* data extraction functions
  extract_data, 3
  extract_substring, 4
  extract_word, 5

* data retrieval functions
  iso_get_all_data, 18
  iso_get_bgrd_data, 20
  iso_get_file_info, 22
  iso_get_raw_data, 24
  iso_get_resistors, 26
  iso_get_standards, 27
  iso_get_vendor_data_table, 30

* export functions
  iso_export_to_excel, 11
  iso_export_to_feather, 13
  iso_save, 52

* file system functions
  iso_expand_paths, 11
  iso_find_absolute_path_roots, 16
  iso_root_paths, 51
  iso_shorten_relative_paths, 56

* file_info operations
  iso_add_file_info.iso_file_list, 7
  iso_filter_files, 15
  iso_mutate_file_info, 35
  iso_parse_file_info, 36
  iso_rename_file_info, 48
  iso_select_file_info, 53
  iso_set_file_root, 55

* file_types
  iso_get_supported_file_types, 29
  iso_register_dual_inlet_file_reader, 46

* functions for values with units
  iso_get_units, 29
  iso_is_double_with_units, 32
  iso_make_units_explicit, 34
  iso_make_units_implicit, 34
  iso_strip_units, 57
  iso_with_units, 57

* isoread functions for different types of IRMS data
  iso_read_continuous_flow, 39
  iso_read_dual_inlet, 41
  iso_read_scan, 45

* problem functions
  iso_filter_files_with_problems, 16
  iso_get_problems, 23
  iso_get_problems_summary, 24
  iso_has_problems, 31
  iso_problem_functions, 38

* settings functions
  iso_caching, 8
  iso_get_default_reader_parameters, 22
  iso_info_messages, 31
  iso_set_default_read_parameters, 54

as.character, 37
availableCores, 40, 42, 44, 46
bind_rows, 57
extract, 3
extract_data, 3, 4, 5, 6
extract_substring, 3, 4, 5, 6
extract_word, 4, 5, 5
filter, 3, 15

iso_add_file_info
  (iso_add_file_info.iso_file_list), 7
  iso_add_file_info.iso_file_list, 7, 15,
  36, 37, 49, 53, 55
iso_as_file_list(iso_is_file), 33
iso_caching, 8, 22, 32, 54
iso_calculate_ratios, 9
iso_cleanup_reader_cache, 9

65
iso_convert_signals, 9
iso_convert_time, 10
iso_debug_mode, 10
iso_double_with_units, 13, 14, 18, 19, 29, 30, 32, 34
iso_double_with_units(iso_with_units), 57
iso_expand_paths, 11, 17, 52, 56
iso_export_to_excel, 11, 15, 52
iso_export_to_feather, 13, 13, 52
iso_filter_files, 3, 8, 15, 36, 37, 49, 53, 55
iso_filter_files_with_problems, 16, 23, 24, 31, 36, 38, 39
iso_find_absolute_path_roots, 11, 16, 43, 51, 52, 56
iso_format, 17
iso_get_all_data, 18, 21, 23, 25, 27, 28, 30
iso_get_bgrd_data, 20, 20, 23, 25, 27, 28, 30
iso_get_data, 21
iso_get_data_summary, 21
iso_get_default_reader_parameters, 8, 22, 32, 54
iso_get_file_info, 3, 7, 12, 14, 15, 18–21, 22, 24–28, 30, 35, 36, 48, 49, 53
iso_get_problems, 16, 23, 24, 31, 39
iso_get_problems_summary, 16, 23, 24, 31, 38, 39
iso_get_raw_data, 3, 7, 18, 20, 21, 23, 24, 27, 28, 30
iso_get_reader_example, 25
iso_get_reader_examples (iso_get_reader_example), 25
iso_get_reader_examples_folder (iso_get_reader_example), 25
iso_get_resistors, 20, 21, 23, 25, 26, 27, 28, 30
iso_get_resistors_info, 27
iso_get_standards, 20, 21, 23, 25, 27, 28, 30
iso_get_standards_info, 29, 21, 25, 27, 28, 30
iso_get_supported_file_types, 11, 29, 48
iso_get_units, 29, 32, 34, 35, 57, 58
iso_get_vendor_data_table, 3, 18, 20, 21, 23, 25, 27, 28, 30
iso_has_problems, 16, 23, 24, 31, 38, 39
iso_info_messages, 8, 22, 31, 54
iso_is_continuous_flow(iso_is_file), 33
iso_is_double_with_units, 29, 32, 34, 35, 57, 58
iso_is_dual_inlet(iso_is_file), 33
iso_is_file, 33
iso_is_file_list(iso_is_file), 33
iso_is_object(iso_is_file), 33
iso_is_scan(iso_is_file), 33
iso_make_units_explicit, 19, 29, 30, 32, 34, 35, 57, 58
iso_make_unitsImplicit, 19, 29, 30, 32, 34, 35, 57, 58
iso_mutate_file_info, 3, 7, 8, 15, 35, 36, 37, 49, 53, 55
iso_omit_files_with_problems, 36
iso_parse_file_info, 3, 8, 15, 36, 37, 49, 53, 55
iso_plot_continuous_flow_data, 37
iso_plot_dual_inlet_data, 38
iso_plot_raw_data, 38
iso_problem_functions, 16, 23, 24, 31, 38
iso_read_continuous_flow, 6, 11, 13, 39, 42, 43, 46, 50, 52, 62
iso_read_dual_inlet, 6, 11, 13, 40, 41, 43, 46, 50, 52, 62
iso_read_files, 43
iso_read_scan, 6, 40, 42, 43, 45, 50, 62
iso_register_continuous_flow_file_reader, 29
iso_register_continuous_flow_file_reader (iso_register_dual_inlet_file_reader), 46
iso_register_dual_inlet_file_reader, 29, 46
iso_register_dual_inlet_file_reader (iso_register_scan_file_reader), 46
iso_register_scan_file_reader (iso_register_dual_inlet_file_reader), 46
iso_rename_file_info, 8, 15, 36, 37, 48, 53, 55
iso_reread_all_files (iso_reread_files), 49
iso_reread_archive(iso_reread_files), 49
iso_reread_changed_files (iso_reread_files), 49
iso_reread_files, 49, 61
iso_reread_outdated_files (iso_reread_files), 49
iso_reread_problem_files, 40, 42, 44, 46
iso_reread_problem_files