Package 'chromConverter'

May 31, 2026

Title Chromatographic File Converter

Version 0.9.0

Maintainer Ethan Bass <ethanbass@gmail.com>

Description Reads chromatograms from binary formats into R objects. Currently supports conversion of 'Agilent ChemStation', 'Agilent MassHunter', 'Agilent OpenLab', 'Shimadzu LabSolutions', 'ThermoRaw', 'Varian Workstation', and 'Waters Empower' files as well as various other formats. In addition to its internal parsers, chromConverter contains bindings to parsers in external libraries, such as 'Aston' <https://github.com/bovee/aston>, 'Entab' <https://github.com/bovee/entab>, 'rainbow' <https://rainbow-api.readthedocs.io/>, and 'ThermoRawFileParser' <https://github.com/compomics/ThermoRawFileParser>.

License GPL (>= 3)

URL https://ethanbass.github.io/chromConverter/, https://github.com/ethanbass/chromConverter/

BugReports https://github.com/ethanbass/chromConverter/issues/

Depends R (>= 4.1.0)

Imports bitops, fs, purrr, readxl, reticulate (>= 1.41.0), stringr, tidyr, utils, RaMS, tibble, xml2, bit64, data.table, base64enc, jsonlite, digest

Suggests entab, ncdf4, pbapply, testthat (>= 3.0.0), mzR, chromConverterExtraTests

Encoding UTF-8

Language en-US

Additional_repositories https://ethanbass.github.io/drat/, https://ethanbass.r-universe.dev/

Config/testthat/edition 3

Config/Needs/website rmarkdown, ggplot2, dplyr

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Ethan Bass [aut, cre] (ORCID: <https://orcid.org/0000-0002-6175-6739>),
James Dillon [ctb, cph] (Author and copyright holder of source code adapted from the 'Chromatography Toolbox' for parsing 'Agilent' FID files.),
Evan Shi [ctb, cph] (Author and copyright holder of source code adapted from 'rainbow' for parsing 'Agilent' UV files.)

Repository CRAN

Date/Publication 2026-05-31 15:00:31 UTC

Contents

`call_entab` *Call Entab*

Description

Converts chromatography date files using entab parsers.

Usage

```
call_entab(
  path,
  data_format = c("wide", "long"),
  format_out = c("matrix", "data.frame", "data.table"),
  format_in = "",
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw")
)
```

Arguments

`path`	Path to file.
`data_format`	Whether to return data in `wide` (default) or `long` format.
`format_out`	Class of output. Either `matrix`, `data.frame`, or `data.table`.
`format_in`	Format of input.
`read_metadata`	Logical. Whether to attach metadata. Defaults to `TRUE`.
`metadata_format`	Format to output metadata. Either `chromconverter` or `raw`.

Value

A chromatogram in the format specified by the `format_out` and `data_format` arguments.

See Also

Other external parsers: `call_openchrom()`, `call_rainbow()`, `read_thermoraw()`, `sp_converter()`, `uv_converter()`

`call_openchrom` *Parse files with OpenChrom*

Description

Writes `xml` batch-files and calls OpenChrom file parsers using a system call to the command-line interface. Unfortunately, the command-line interface is no longer supported in newer versions of OpenChrom (starting with version 1.5.0) and older versions of OpenChrom that do support the command line interface are no longer available from Lablicate. Thus, this function is deprecated since it will only work if you happen to have access to OpenChrom version 1.4.0, which has been scrubbed from the internet.

Usage

```
call_openchrom(
  files,
  path_out = NULL,
  format_in,
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  export_format = c("mzml", "csv", "cdf", "animl"),
  return_paths = FALSE,
  verbose = getOption("verbose")
)
```

Arguments

`files`	Path to files.
`path_out`	Directory to export converted files.
`format_in`	Either `msd` for mass spectrometry data, `csd` for flame ionization data, or `wsd` for DAD/UV data.
`format_out`	Class of output. Either `matrix`, `data.frame`, or `data.table`.
`data_format`	Whether to return data in `wide` (default) or `long` format.
`export_format`	Either `mzml`, `csv`, `cdf`, `animl`. Defaults to `mzml`.
`return_paths`	Logical. If `TRUE`, the function will return a character vector of paths to the newly created files.
`verbose`	Logical. Whether to print output from OpenChrom to the console.

Details

The `call_openchrom` function works by creating an xml batchfile and feeding it to the OpenChrom command-line interface. OpenChrom batchfiles consist of `InputEntries` (specifying the files you want to convert) and `ProcessEntries` (specifying what you want to do to the files). The parsers are organized into broad categories by detector-type and output format. The detector-types are `msd` (mass selective detectors), `csd` (current selective detectors, e.g., FID, ECD, NPD), and `wsd` (wavelength selective detectors, e.g., DAD, and UV/VIS). Thus, when calling the OpenChrom parsers, one of these three options must be specified using the `format_in` argument.

Value

If `return_paths` is `FALSE`, the function will return a list of chromatograms (if an appropriate parser is available to import the files into R). The chromatograms will be returned in `matrix` or `data.frame` format according to the value of `format_out`. If `return_paths` is `TRUE`, the function will return a character vector of paths to the newly created files.

Side effects

Chromatograms will be exported in the format specified by `export_format` in the folder specified by `path_out`.

Note

Activating the OpenChrom command-line will deactivate the graphical user interface (GUI). Thus, if you wish to continue using the OpenChrom GUI, it is recommended to create a separate command-line version of OpenChrom to call from R.

Author(s)

Ethan Bass

References

Wenig, Philip and Odermatt, Juergen. OpenChrom: A Cross-Platform Open Source Software for the Mass Spectrometric Analysis of Chromatographic Data. *BMC Bioinformatics* **11**, no. 1 (July 30, 2010): 405. doi:10.1186/1471210511405.

See Also

Other external parsers: `call_entab()`, `call_rainbow()`, `read_thermoraw()`, `sp_converter()`, `uv_converter()`

`call_rainbow` *Call 'rainbow' parsers Parse 'Agilent' or 'Waters' files with rainbow parsers*

Description

Uses rainbow parsers to read in Agilent (`.D`) and Waters (`.raw`) files. If `format_in` is `"agilent_d"` or `"waters_raw"`, a directory of the appropriate format (`.D` or `.raw`) should be provided to the `path` argument. If `format_in` is `"chemstation_uv"` a `.uv` file should be provided. Data can be filtered by detector type using the `what` argument.

Usage

```
call_rainbow(
  path,
  format_in = c("agilent_d", "waters_raw", "masshunter", "chemstation", "chemstation_uv",
    "chemstation_fid", "chemstation_ms"),
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  by = c("detector", "name"),
  what = NULL,
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  collapse = TRUE,
  precision = 1,
  sparse = TRUE
)
```

Arguments

path	Path to file.
format_in	Format of the supplied files. Either agilent_d, waters_raw, or chemstation.
format_out	Class of output. Either matrix, data.frame, or data.table.
data_format	Whether to return data in wide (default) or long format.
by	How to order the list that is returned. Either detector (default) or name.
what	What types of data to return (e.g. MS, UV, CAD, ELSD). This argument only applies if by == "detector".
read_metadata	Logical. Whether to attach metadata. Defaults to TRUE.
metadata_format	Format to output metadata. Either chromconverter or raw.
collapse	Logical. Whether to collapse lists that only contain a single element. Defaults to TRUE.
precision	Number of decimals to round mz values. Defaults to 1.
sparse	Logical. Whether to return MS data in sparse format (excluding zeros). Defaults to TRUE. Applies only when data are requested in long format.

Value

Returns a (nested) list of matrices or data.frames according to the value of format_out. Data is ordered according to the value of by.

Author(s)

Ethan Bass

See Also

Other external parsers: call_entab(), call_openchrom(), read_thermoraw(), sp_converter(), uv_converter()

configure_openchrom *Configure 'OpenChrom' parser*

Description

Configures OpenChrom to use command-line interface. Requires OpenChrom version prior to 0.5.0.

Usage

```
configure_openchrom(cli = c("null", "true", "false", "status"), path = NULL)
```

Arguments

cli	Defaults to NULL. If "true", R will rewrite openchrom ini file to enable CLI. If "false", R will disable CLI. If NULL, R will not modify the ini file.
path	Path to 'OpenChrom' executable (Optional). The supplied path will overwrite the current path.

Value

If `cli` is set to `"status"`, returns a Boolean value indicating whether 'OpenChrom' is configured correctly. Otherwise, returns the path to OpenChrom command-line application.

Author(s)

Ethan Bass

See Also

`call_openchrom`

extract_metadata *Extract metadata*

Description

Extract metadata as a `data.frame`, `data.table` or `tibble` from a list of chromatograms.

Usage

```
extract_metadata(
  chrom_list,
  what = c("instrument", "detector", "detector_id", "software", "method", "batch",
    "operator", "run_datetime", "sample_name", "sample_id", "injection_volume",
   "time_range", "time_interval", "time_unit", "detector_range", "detector_y_unit",
    "detector_x_unit", "intensity_multiplier", "scaled", "source_file",
    "source_file_format", "source_sha1", "data_format", "parser", "format_out"),
  format_out = c("data.frame", "data.table", "tibble")
)
```

Arguments

chrom_list	A list of chromatograms with attached metadata (as returned by read_chroms with read_metadata = TRUE).
what	A character vector specifying the metadata elements to extract.
format_out	Format of object. Either data.frame, data.table or tibble.

Value

A data.frame, tibble, or data.table (according to the value of format_out), with samples as rows and the specified metadata elements as columns.

---

print.chrom_list *Print a chrom_list object*

Description

Prints a summary of a chrom_list without displaying the underlying chromatographic data. Attributes that are constant across all chromatograms are collapsed into a single header line, while varying attributes are shown as a table truncated to the first n rows.

Usage

```
## S3 method for class 'chrom_list'
print(
  x,
  n = 5,
  cols = c("sample_name", "run_datetime", "method", "detector"),
  ...
)
```

Arguments

x	A chrom_list object.
n	Integer. Maximum number of chromatograms to show in the table. Defaults to 10.
cols	Character vector of attribute names to extract and display. Defaults to c("sample_name", "run_datetime", "method", "detector").
...	Additional arguments (currently ignored).

Value

Invisibly returns x.

See Also

extract_metadata

---

read_acaml *Read 'Agilent' ACAML files from directory.*

Description

Extracts injection metadata from 'Agilent Common Analytical Markup Language' (ACAML) files into an R object.

Usage

```
read_acaml(
  path,
  find_files,
  format_out = c("data.frame", "data.table", "tibble"),
  progress_bar = TRUE,
  cl = 1
)
```

Arguments

path	Path(s) to ACAML files or to folders that contain the files.
find_files	Logical. Set to TRUE (default) if you are providing the function with a folder or vector of folders containing the files. Otherwise, set to FALSE.
format_out	Class of output. Either matrix, data.frame, or data.table.
progress_bar	Logical. Whether to show progress bar. Defaults to TRUE if pbapply is installed.
cl	Argument to pbapply specifying the number of clusters to use or a cluster object created by makeCluster. Defaults to 1.

Details

ACAML is an XML-based format used by Agilent OpenLab to store sequence and sample metadata. This function extracts information from the `InjectionMetaData` nodes embedded in the `InjectionMetaDataItems` custom field files, which do not seem to be readily accessible through other means.

Value

A `data.frame`, `data.table` or `tibble` (according to the value of `format_out`) containing sample metadata derived from the supplied ACAML files.

Examples

```
## Not run:
read_acaml(path)

## End(Not run)
```

---

read_agilent_amx *Read Agilent AMX method file*

Description

Parses an Agilent `.amx` method archive, extracting instrument parameters from one or more of its driver sub-files.

Usage

```
read_agilent_amx(
  path,
  what = c("dad", "pump", "comp", "sampler"),
  path_out = NULL,
  format_out = c("data.frame", "tibble", "data.table"),
  gradient_format = c("wide", "long")
)
```

Arguments

`path`	Path to the `.amx` file.
`what`	One or more instrument modules to parse. Any combination of `"dad"`, `"pump"`, `"comp"`, and `"sampler"`. Defaults to all four.
`path_out`	Directory into which the archive is extracted. If `NULL` (default), a temporary directory is used and cleaned up on exit.
`format_out`	Class of output (for tables). Either `"data.frame"`, `"tibble"` or `"data.table"`.
`gradient_format`	Whether to return the gradient in `"wide"` (default) or `"long"` format.

Value

A named list with one element per parsed module, plus "metadata". Elements present depend on what; see below for the structure of each.

metadata — a list with scalar elements:

method_name Original method name.

version Method version string.

status Approval state.

created Creation timestamp (POSIXct, UTC).

created_by Username of creator.

modified Last-modified timestamp (POSIXct, UTC).

modified_by Username of last modifier.

pump — a list with scalar elements flow_mL_min, stop_time_min, post_time_min, pressure_low_bar, pressure_high_bar, plus:

solvents A data.frame of active solvent channels: channel, percentage, solvent.

gradient A data.frame of timetable entries. Wide format (default): time_min plus one pct_<channel> column per active channel. Long format: time_min, channel, percent.

dad — a list with scalar elements peakwidth_nm, slitwidth_nm, uv_lamp_required, vis_lamp_required, spectra_from_nm, spectra_to_nm, spectra_step_nm, plus:

signals A data.frame of active signals: id, wavelength_nm, bandwidth_nm.

comp — a list with scalar element post_time_min, plus:

temp_controls Two-row data.frame (Left/Right): side, temperature_C, not_ready_limit_C, equilibration_time_min.

sampler — a list with scalar elements: thermostat_installed, draw_speed_uL_min, eject_speed_uL_min, wait_after_draw_min, injection_volume_uL, wash_time_s.

Examples

```
## Not run:
read_agilent_amx(path)

## End(Not run)
```

read_agilent_d *Read files from 'Agilent ChemStation' .D directories*

Description

Reads files from 'Agilent' .D directories.

Usage

```
read_agilent_d(
  path,
  what = c("dad", "chroms", "peak_table"),
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  collapse = TRUE
)
```

Arguments

path	Path to 'Agilent' .D directory.
what	Whether to extract chromatograms (chroms), DAD data (dad) and/or peak tables (peak_table). Accepts multiple arguments.
format_out	Class of output. Either matrix, data.frame, or data.table.
data_format	Whether to return data in wide (default) or long format.
read_metadata	Logical. Whether to attach metadata. Defaults to TRUE.
metadata_format	Format to output metadata. Either chromconverter or raw.
collapse	Logical. Whether to collapse lists that only contain a single element. Defaults to TRUE.

Details

Currently this function is limited to reading .uv, .ch and peak_table elements.

Value

A list of chromatograms in the format specified by data_format and format_out. If data_format is wide, the chromatograms will be returned with retention times as rows and columns containing signal intensity for each signal. If long format is requested, retention times will be in the first column. The format_out argument determines whether the chromatogram is returned as a matrix, data.frame or data.table. Metadata can be attached to the chromatogram as attributes if read_metadata is TRUE.

Author(s)

Ethan Bass

See Also

Other 'Agilent' parsers: `read_agilent_dx()`, `read_chemstation_ch()`, `read_chemstation_csv()`, `read_chemstation_ms()`, `read_chemstation_reports()`, `read_chemstation_uv()`

Examples

```
read_agilent_d("tests/testthat/testdata/RUTIN2.D")
```

---

read_agilent_dx *Read 'Agilent' DX files*

Description

Reads 'Agilent' `.dx` files.

Usage

```
read_agilent_dx(
  path,
  what = c("chroms", "dad"),
  path_out = NULL,
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  collapse = TRUE
)
```

Arguments

`path`	Path to Agilent `.dx` file.
`what`	Whether to extract chromatograms (`chroms`), DAD data (`dad`) and/or auxiliary instrumental data (`instrument`) (e.g., temperature, pressure, solvent composition, etc.). Accepts multiple arguments.
`path_out`	A directory to export unzipped files. If a path is not specified, the files will be written to a temp directory on the disk. The function will overwrite existing folders in the specified directory that share the basename of the file specified by `path`.
`format_out`	Class of output. Either `matrix`, `data.frame`, or `data.table`.
`data_format`	Whether to return data in `wide` (default) or `long` format.
`read_metadata`	Logical. Whether to attach metadata. Defaults to `TRUE`.

metadata_format
: Format to output metadata. Either chromconverter or raw.

collapse
: Logical. Whether to collapse lists that only contain a single element. Defaults to TRUE.

Details

This function unzips 'Agilent' .dx into a temporary directory using unzip and calls the appropriate parser on the unzipped file.

Value

A chromatogram in the format specified by format_out (retention time x wavelength).

Author(s)

Ethan Bass

See Also

Other 'Agilent' parsers: read_agilent_d(), read_chemstation_ch(), read_chemstation_csv(), read_chemstation_ms(), read_chemstation_reports(), read_chemstation_uv()

Examples

```
## Not run:
read_agilent_dx(path)

## End(Not run)
```

---

read_asm *Read 'Allotrope Simple Model' (ASM) 2D chromatograms*

Description

Reads 'Allotrope Simple Model' files into R.

Usage

```
read_asm(
  path,
  data_format = c("wide", "long"),
  format_out = c("matrix", "data.frame", "data.table"),
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  collapse = TRUE
)
```

Arguments

path	Path to ASM .json file.
data_format	Whether to return data in wide (default) or long format.
format_out	Class of output. Either matrix, data.frame, or data.table.
read_metadata	Logical. Whether to attach metadata. Defaults to TRUE.
metadata_format	Format to output metadata. Either chromconverter or raw.
collapse	Logical. Whether to collapse lists that only contain a single element. Defaults to TRUE.

Value

A 2D chromatogram in the format specified by data_format and format_out. If data_format is wide, the chromatogram will be returned with retention times as rows and a single column for the intensity. If long format is requested, two columns will be returned: one for the retention time and one for the intensity. The format_out argument determines whether the chromatogram is returned as a matrix, data.frame, or data.table. Metadata can be attached to the chromatogram as attributes if read_metadata is TRUE.

Author(s)

Ethan Bass

---

read_cdf *Read CDF*

Description

Reads 'Analytical Data Interchange' (ANDI) netCDF (.cdf) files.

Usage

```
read_cdf(
  path,
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  what = NULL,
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  collapse = TRUE,
  ...
)
```

Arguments

path	Path to ANDI netCDF file.
format_out	Class of output. Either matrix, data.frame, or data.table.
data_format	Whether to return data in wide or long format. For 2D files, "long" format returns the retention time as the first column of the data.frame or matrix while "wide" format returns the retention time as the rownames of the object. This argument applies only to 2D chromatograms, since MS data will always be returned in long format.
what	For ANDI chrom files, whether to extract chroms and/or peak_table. For ANDI ms files, whether to extract MS1 scans (MS1) or the total ion chromatogram (TIC).
read_metadata	Logical. Whether to attach metadata. Defaults to TRUE.
metadata_format	Format to output metadata. Either chromconverter or raw.
collapse	Logical. Whether to collapse lists that only contain a single element. Defaults to TRUE.
...	Additional arguments to parser. The ms_format argument can be used here to specify whether to return mass spectra in list format or as a data.frame.

Value

A chromatogram in the format specified by the format_out and data_format arguments.

Author(s)

Ethan Bass

---

read_chemstation_ch *Read 'Agilent ChemStation' CH files*

Description

Reads 'Agilent ChemStation' .ch files.

Usage

```
read_chemstation_ch(
  path,
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  scale = TRUE,
  source_file = NULL
)
```

Arguments

path	Path to 'Agilent' .ch file.
format_out	Class of output. Either matrix, data.frame, or data.table.
data_format	Whether to return data in wide (default) or long format.
read_metadata	Logical. Whether to attach metadata. Defaults to TRUE.
metadata_format	Format to output metadata. Either chromconverter or raw.
scale	Whether to scale the data by the scaling factor present in the file. Defaults to TRUE. 'MassHunter' seems to ignore the scaling factor in at least some types of 'ChemStation' files.
source_file	Source file from which chromatogram data was originally derived.

Details

'Agilent' .ch files come in several different formats. This parser can automatically detect and read several versions of these files from 'Agilent ChemStation' and 'Agilent OpenLab', including versions 30 and 130, which are generally produced by ultraviolet detectors, as well as 81, 179, and 181 which are generally produced by flame ionization (FID) detectors.

Value

A 2D chromatogram in the format specified by data_format and format_out. If data_format is wide, the chromatogram will be returned with retention times as rows and a single column for the intensity. If long format is requested, two columns will be returned: one for the retention time and one for the intensity. The format_out argument determines whether the chromatogram is returned as a matrix, data.frame, or data.table. Metadata can be attached to the chromatogram as attributes if read_metadata is TRUE.

Note

This function was adapted from the Chromatography Toolbox (© James Dillon 2014).

Author(s)

Ethan Bass

See Also

Other 'Agilent' parsers: read_agilent_d(), read_agilent_dx(), read_chemstation_csv(), read_chemstation_ms(), read_chemstation_reports(), read_chemstation_uv()

Examples

```
read_chemstation_ch("tests/testthat/testdata/chemstation_130.ch")
```

`read_chemstation_csv` *Read 'Agilent ChemStation' CSV files*

Description

Reads 'Agilent Chemstation' `.csv` files.

Usage

```
read_chemstation_csv(
  path,
  format_out = "matrix",
  data_format = "wide",
  read_metadata = TRUE
)
```

Arguments

`path`	Path to 'Agilent' `.csv` file.
`format_out`	Class of output. Either `matrix`, `data.frame`, or `data.table`.
`data_format`	Whether to return data in `wide` (default) or `long` format.
`read_metadata`	Logical. Whether to attach metadata. Defaults to `TRUE`. There is no instrumental metadata saved in the CSV files so this will only attach metadata about the settings used by chromConverter to parse the file.

Details

'Agilent Chemstation' CSV files are encoded in UTF-16.

Value

A chromatogram in the format specified by `format_out` and `data_format`.

Author(s)

Ethan Bass

See Also

Other 'Agilent' parsers: `read_agilent_d()`, `read_agilent_dx()`, `read_chemstation_ch()`, `read_chemstation_ms()`, `read_chemstation_reports()`, `read_chemstation_uv()`

Examples

```
read_chemstation_csv("tests/testthat/testdata/dad1.csv")
```

`read_chemstation_ms` *Read 'Agilent ChemStation' MS file.*

Description

Reads 'Agilent ChemStation MSD Spectral Files' beginning with `x01/x32/x00/x00`.

Usage

```
read_chemstation_ms(
  path,
  what = c("MS1", "BPC", "TIC"),
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = "long",
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  collapse = TRUE
)
```

Arguments

`path`	Path to 'Agilent' `.ms` file.
`what`	What stream to get: current options are `MS1`, `BPC` and/or `TIC`. If a stream is not specified, the function will return all streams.
`format_out`	Class of output. Either `matrix`, `data.frame`, or `data.table`.
`data_format`	Whether to return data in `wide` (default) or `long` format.
`read_metadata`	Logical. Whether to attach metadata. Defaults to `TRUE`.
`metadata_format`	Format to output metadata. Either `chromconverter` or `raw`.
`collapse`	Logical. Whether to collapse lists that only contain a single element. Defaults to `TRUE`.

Value

A list of chromatograms in the format specified by `data_format` and `format_out`. If `data_format` is `wide`, 2D chromatograms will be returned with retention times as rows and a single column for the intensity. Otherwise, two columns will be returned: one for the retention time and one for the intensity. MS data will always be returned in long format. The `format_out` argument determines whether the chromatogram is returned as a `matrix`, `data.frame`, or `data.table`. Metadata can be attached to the chromatogram as attributes if `read_metadata` is `TRUE`.

Note

Many thanks to Evan Shi and Eugene Kwan for providing helpful information on the structure of these files in the rainbow documentation.

Author(s)

Ethan Bass

See Also

Other 'Agilent' parsers: `read_agilent_d()`, `read_agilent_dx()`, `read_chemstation_ch()`, `read_chemstation_csv()`, `read_chemstation_reports()`, `read_chemstation_uv()`

Examples

```
## Not run:
read_chemstation_ms(path)

## End(Not run)
```

`read_chemstation_reports` *Read 'Agilent ChemStation' report files.*

Description

Reads 'Agilent ChemStation' reports into R.

Usage

```
read_chemstation_reports(
  paths,
  data_format = c("chromatographr", "original"),
  metadata_format = c("chromconverter", "raw")
)
```

Arguments

`paths`	Paths to 'ChemStation' report files.
`data_format`	Format to output data. Either `chromatographr` or `chemstation`.
`metadata_format`	Format to output metadata. Either `chromconverter` or `raw`.

Value

A `data.frame` containing the information from the specified 'ChemStation' report.

Author(s)

Ethan Bass

See Also

Other 'Agilent' parsers: read_agilent_d(), read_agilent_dx(), read_chemstation_ch(), read_chemstation_csv(), read_chemstation_ms(), read_chemstation_uv()

`read_chemstation_uv` *Read 'Agilent ChemStation' DAD files*

Description

Agilent `.uv` files come in several different formats. This parser can automatically detect and read several versions of these files from 'Agilent ChemStation' and 'Agilent OpenLab', including versions `31` and `131`.

Usage

```
read_chemstation_uv(
  path,
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  scale = TRUE,
  source_file = NULL
)
```

Arguments

`path`	Path to 'Agilent' `.uv` file.
`format_out`	Class of output. Either `matrix`, `data.frame`, or `data.table`.
`data_format`	Whether to return data in `wide` (default) or `long` format.
`read_metadata`	Logical. Whether to attach metadata. Defaults to `TRUE`.
`metadata_format`	Format to output metadata. Either `chromconverter` or `raw`.
`scale`	Whether to scale the data by the scaling factor present in the file. Defaults to `TRUE`.
`source_file`	Source file from which UV data was originally derived.

Value

A 3D chromatogram in the format specified by `data_format` and `format_out`. If `data_format` is `wide`, the chromatogram will be returned with retention times as rows and wavelengths as columns. If `long` format is requested, three columns will be returned: one for the retention time, one for the wavelength and one for the intensity. The `format_out` argument determines whether the chromatogram is returned as a `matrix`, `data.frame`, or `data.table`. Metadata will be attached to the chromatogram as attributes if `read_metadata` is `TRUE`.

Note

This function was adapted from the parser in the rainbow project licensed under GPL 3 by Evan Shi https://rainbow-api.readthedocs.io/en/latest/agilent/uv.html.

Author(s)

Ethan Bass

See Also

Other 'Agilent' parsers: read_agilent_d(), read_agilent_dx(), read_chemstation_ch(), read_chemstation_csv(), read_chemstation_ms(), read_chemstation_reports()

Examples

```
read_chemstation_uv("tests/testthat/testdata/dad1.uv")
```

---

read_chromatotec *Read 'Chromatotec' file*

Description

Reads 'Chromatotec' .Chrom files.

Usage

```
read_chromatotec(
  path,
  what = c("chrom", "peak_table"),
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  collapse = TRUE
)
```

Arguments

path	The path to 'Chromatotec' .Chrom file.
what	Whether to extract chromatograms (chrom) and/or peak_table data. Accepts multiple arguments.
format_out	Class of output. Either matrix, data.frame, or data.table.
data_format	Whether to return data in wide (default) or long format.
read_metadata	Logical. Whether to attach metadata. Defaults to TRUE.

metadata_format
: Format to output metadata. Either chromconverter or raw.

collapse
: Logical. Whether to collapse lists that only contain a single element. Defaults to TRUE.

Value

A chromatogram and/or peak table from the specified path, according to the value of what. Chromatograms are returned in the format specified by format_out.

Author(s)

Ethan Bass

Examples

```
## Not run:
read_chromatotec(path)

## End(Not run)
```

---

read_chromeleon *Read 'Chromeleon' ASCII files*

Description

Reads 'Thermo Fisher Chromeleon™ CDS' ASCII (.txt) files.

Usage

```
read_chromeleon(
  path,
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  decimal_mark = NULL
)
```

Arguments

path
: Path to 'Chromeleon' ASCII file.

format_out
: Class of output. Either matrix, data.frame, or data.table.

data_format
: Whether to return data in wide (default) or long format.

read_metadata
: Logical. Whether to attach metadata. Defaults to TRUE.

metadata_format
: Format to output metadata. Either chromconverter or raw.

decimal_mark — Which character is used as the decimal separator in the file. By default, decimal mark will be detected automatically, but it can also be manually set as "." or ",".

Value

A chromatogram in the format specified by format_out (retention time x wavelength).

Author(s)

Ethan Bass

---

read_chroms *Read Chromatograms*

Description

Reads chromatograms from specified folders or vector of paths using either an internal parser or bindings to an external library, such as Aston, Entab, ThermoRawFileParser, OpenChrom, rainbow.

Usage

```
read_chroms(
  paths,
  format_in = c("agilent_d", "agilent_dx", "asm", "chemstation", "chemstation_fid",
    "chemstation_ch", "chemstation_csv", "chemstation_ms", "chemstation_uv",
    "masshunter_dad", "chromeleon_uv", "chromatotec", "mzml", "mzxml", "mdf",
   "shimadzu_ascii", "shimadzu_dad", "shimadzu_fid", "shimadzu_gcd", "shimadzu_qgd",
   "shimadzu_lcd", "thermoraw", "varian_sms", "waters_arw", "waters_raw", "msd", "csd",
    "wsd", "csv", "other"),
  find_files,
  pattern = NULL,
  parser = c("", "chromconverter", "aston", "entab", "thermoraw", "openchrom", "rainbow"),
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  path_out = NULL,
  export_format = c("", "csv", "chemstation_csv", "cdf", "mzml", "animl", "arw"),
  force = FALSE,
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  progress_bar,
  cl = 1,
  verbose = getOption("verbose"),
  sample_names = c("basename", "sample_name"),
  dat = NULL,
  ...
)
```

Arguments

paths	Paths to data files or directories containing the files.
format_in	Format of files to be imported/converted. Current options include: agilent_d, agilent_dx, chemstation, chemstation_uv, chemstation_ch, chemstation_csv, chemstation_ms, masshunter, masshunter_dad, chromeleon_uv, shimadzu_ascii, shimadzu_fid, shimadzu_dad, thermoraw, waters_arw, waters_raw, mzml, mzxml, cdf, mdf, msd, csd, wsd, or other.
find_files	Logical. Set to TRUE (default) if you are providing the function with a folder or vector of folders containing the files. Otherwise, set to FALSE.
pattern	pattern (e.g. a file extension). Defaults to NULL, in which case file extension will be deduced from format_in.
parser	What parser to use (optional). Current option are chromconverter, aston,, entab, thermoraw, openchrom, rainbow.
format_out	Class of output. Either matrix, data.frame, or data.table.
data_format	Whether to output data in wide or long format. Either wide (default) or long.
path_out	Path for exporting files. If path is not specified, the user will be prompted to create a temp directory.
export_format	Export format. Currently the options include .csv, chemstation_csv (utf-16 encoding), cdf, mzml, animl and arw.
force	Logical. Whether to overwrite files when exporting. Defaults to FALSE.
read_metadata	Logical, whether to attach metadata (if it's available). Defaults to TRUE.
metadata_format	Format to output metadata. Either chromconverter or raw.
progress_bar	Logical. Whether to show progress bar. Defaults to TRUE if pbapply is installed.
cl	Argument to pbapply specifying the number of clusters to use or a cluster object created by makeCluster. Defaults to 1.
verbose	Logical. Whether to print output from external parsers to the R console.
sample_names	Which sample names to use. Options are basename to use the filename (default) or sample_name to use the sample name encoded in the file metadata.
dat	Existing list of chromatograms to append results. Defaults to NULL.
...	Additional arguments to parser.

Details

Provides a unified interface to all chromConverter parsers. Currently recognizes 'Agilent ChemStation' (.uv, .ch, .dx), 'Agilent MassHunter' (.dad), 'Thermo RAW' (.raw), 'Waters ARW' (.arw), 'Waters RAW' (.raw), 'Chromeleon ASCII' (.txt), 'Shimadzu ASCII' (.txt), 'Shimadzu GCD' (.gcd), 'Shimadzu LCD' (.lcd, DAD and chromatogram streams) and 'Shimadzu QGD' (.qgd) files. Also, wraps 'OpenChrom' parsers, which include many additional formats. To use 'Entab', 'ThermoRawFileParser', or 'OpenChrom' parsers, they must be separately installed. Please see the instructions in the README for further details.

If paths to individual files are provided, read_chroms will try to infer the file format and select an appropriate parser. However, when providing paths to directories, the file format must be specified using the format_in argument.

Value

A list of chromatograms in `matrix`, `data.frame`, or `data.table` format, according to the value of `format_out`. Chromatograms may be returned in either `wide` or `long` format according to the value of `data_format`.

Side effects

If `export_format` is provided, chromatograms will be exported in the specified format specified into the folder specified by `path_out`. Files can currently be converted to `csv`, `mzml`, `cdf`, `arw`. If an `openchrom` parser is selected, ANIML format is available as an additional option.

Author(s)

Ethan Bass

Examples

```
path <- "tests/testthat/testdata/dad1.uv"
chr <- read_chroms(path, find_files = FALSE, format_in = "chemstation_uv")
```

`read_mdf` *Read 'Lumex' MDF*

Description

Reads 'Lumex' `.mdf` files.

Usage

```
read_mdf(
  path,
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  read_metadata = TRUE
)
```

Arguments

`path`	The path to a 'Lumex' `.mdf` file.
`format_out`	Class of output. Either `matrix`, `data.frame`, or `data.table`.
`data_format`	Whether to return data in `wide` (default) or `long` format.
`read_metadata`	Logical. Whether to attach metadata. Defaults to `TRUE`.

Value

A chromatogram in the format specified by the `format_out` and `data_format` arguments.

Author(s)

Ethan Bass

`read_mzml` *Read mzML files*

Description

Extracts data from `mzML` files using parsers from either RaMS or mzR. The RaMS parser (default) will only return data in tidy (long) format. The mzR parser will return data in wide format. Currently the mzR-based parser is configured to return only DAD data.

Usage

```
read_mzml(
  path,
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  parser = c("RaMS", "mzR"),
 what = c("MS1", "MS2", "BPC", "TIC", "DAD", "chroms", "metadata", "everything"),
  verbose = FALSE,
  ...
)
```

Arguments

`path`	Path to `.mzml` file.
`format_out`	Class of output. Only applies if `mzR` is selected. Either `matrix`, `data.frame`, or `data.table`. `RaMS` will return a list of data.tables regardless of what is selected here.
`data_format`	Whether to return data in `wide` (default) or `long` format.
`parser`	What parser to use. Either `RaMS` or `mzR`.
`what`	What types of data to return (argument to RaMS::grabMSdata). Options include `MS1`, `MS2`, `BPC`, `TIC`, `DAD`, `chroms`, `metadata`, or `everything`).
`verbose`	Argument to `grabMSdata` controlling verbosity.
`...`	Additional arguments to `grabMSdata`.

Value

If `RaMS` is selected, the function will return a list of "tidy" `data.table` objects. If `mzR` is selected, the function will return a chromatogram in `matrix` or `data.frame` format according to the value of `format_out`.

Author(s)

Ethan Bass

read_peaklist *Read peak lists*

Description

Reads peak lists from specified folders or vector of paths.

Usage

```
read_peaklist(
  paths,
  find_files,
  format_in = c("chemstation", "shimadzu_fid", "shimadzu_dad", "shimadzu_lcd",
    "shimadzu_gcd", "chromatotec"),
  pattern = NULL,
  data_format = c("chromatographr", "original"),
  metadata_format = c("chromconverter", "raw"),
  read_metadata = TRUE,
  progress_bar,
  cl = 1
)
```

Arguments

paths	Paths to files or folders containing peak list files.
find_files	Logical. Set to TRUE (default) if you are providing the function with a folder or vector of folders containing the files. Otherwise, set to FALSE.
format_in	Format of files to be imported/converted. Current options include: chemstation, shimadzu_fid, shimadzu_dad, shimadzu_lcd, and shimadzu_gcd.
pattern	A pattern (e.g. a file extension). Defaults to NULL, in which case the file extension will be deduced from format_in.
data_format	Either chromatographr or original.
metadata_format	Format to output metadata. Either chromconverter or raw.
read_metadata	Logical. Whether to attach metadata. Defaults to TRUE.
progress_bar	Logical. Whether to show progress bar. Defaults to TRUE if pbapply is installed.
cl	Argument to pbapply specifying the number of clusters to use or a cluster object created by makeCluster. Defaults to 1.

Value

A list of data.frames containing information about peaks where each list element represents a sample and each row represents an individual peak in that sample.

Author(s)

Ethan Bass

Examples

```
path <- "tests/testthat/testdata/RUTIN2.D"
peak_list <- read_peaklist(path)
peak_list[["RUTIN2"]][["254"]]
```

---

read_shimadzu *Read 'Shimadzu' ASCII*

Description

Reads 'Shimadzu' ASCII `.txt`) files. These files can be exported from 'Shimadzu LabSolutions' by right clicking on samples in the sample list and selecting `File Conversion:Convert to ASCII`.

Usage

```
read_shimadzu(
  path,
  what = "chroms",
  format_in = NULL,
  include = c("fid", "lc", "dad", "uv", "tic"),
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  peaktable_format = c("chromatographr", "original"),
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  ms_format = c("data.frame", "list"),
  collapse = TRUE,
  scale = TRUE
)
```

Arguments

`path`	Path to Shimadzu `.txt` ASCII file.
`what`	Whether to extract chromatograms (`chroms`), `peak_table`, and/or `ms_spectra`. Accepts multiple arguments.
`format_in`	This argument is deprecated and is no longer required.
`include`	Which chromatograms to include. Options are `fid`, `dad`, `uv`, `tic`, and `status`.
`format_out`	Class of output. Either `matrix`, `data.frame`, or `data.table`.
`data_format`	Whether to return data in `wide` (default) or `long` format.
`peaktable_format`	Whether to return peak tables in `chromatographr` or `original` format.

read_metadata	Logical. Whether to attach metadata. Defaults to TRUE.
metadata_format	Format to output metadata. Either chromconverter or raw.
ms_format	Whether to return mass spectral data as a (long) data.frame or a list.
collapse	Logical. Whether to collapse lists that only contain a single element. Defaults to TRUE.
scale	Whether to scale the data by the scaling factor present in the file. Defaults to TRUE.

Value

A nested list of elements from the specified file, where the top levels are chromatograms, peak tables, and/or mass spectra according to the value of what. Chromatograms are returned in the format specified by format_out.

Author(s)

Ethan Bass

See Also

Other 'Shimadzu' parsers: read_shimadzu_gcd(), read_shimadzu_lcd(), read_shimadzu_qgd(), read_sz_lcd_2d(), read_sz_lcd_3d()

Examples

```
path <- "tests/testthat/testdata/ladder.txt"
read_shimadzu(path)
```

---

read_shimadzu_gcd *Read 'Shimadzu' GCD*

Description

Read chromatogram data streams from 'Shimadzu' .gcd files.

Usage

```
read_shimadzu_gcd(
  path,
  what = "chroms",
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  collapse = TRUE
)
```

Arguments

path	Path to 'Shimadzu' .gcd file.
what	What stream to get: current options are chromatograms (chroms) and/or peak lists (peak_table). If a stream is not specified, the function will default to chroms.
format_out	Class of output. Either matrix, data.frame, or data.table.
data_format	Whether to return data in wide (default) or long format.
read_metadata	Logical. Whether to attach metadata. Defaults to TRUE.
metadata_format	Format to output metadata. Either chromconverter or raw.
collapse	Logical. Whether to collapse lists that only contain a single element. Defaults to TRUE.

Details

A parser to read chromatogram data streams from 'Shimadzu' .gcd files. GCD files are encoded as 'Microsoft' OLE documents. The parser relies on the olefile package in Python to unpack the files. The PDA data is encoded in a stream called PDA 3D Raw Data:3D Raw Data. The GCD data stream contains a segment for each retention time, beginning with a 24-byte header.

The 24 byte header consists of the following fields:

- 4 bytes: segment label (17234).
- 4 bytes: Little-endian integer specifying the sampling interval in milliseconds.
- 4 bytes: Little-endian integer specifying the number of values in the file.
- 4 bytes: Little-endian integer specifying the total number of bytes in the file (However, this seems to be off by a few bytes?).
- 8 bytes of 00s

After the header, the data are simply encoded as 64-bit (little-endian) floating-point numbers. The retention times can be (approximately?) derived from the number of values and the sampling interval encoded in the header.

Value

A 2D chromatogram in the format specified by data_format and format_out. If data_format is wide, the chromatogram will be returned with retention times as rows and a single column for the intensity. If long format is requested, two columns will be returned: one for the retention time and one for the intensity. The format_out argument determines whether the chromatogram is returned as a matrix, data.frame, or data.table. Metadata can be attached to the chromatogram as attributes if read_metadata is TRUE.

Note

This parser is experimental and may still need some work. It is not yet able to interpret much metadata from the files.

Author(s)

Ethan Bass

See Also

Other 'Shimadzu' parsers: `read_shimadzu()`, `read_shimadzu_lcd()`, `read_shimadzu_qgd()`, `read_sz_lcd_2d()`, `read_sz_lcd_3d()`

`read_shimadzu_lcd` *Read 'Shimadzu' LCD*

Description

Read 3D PDA or 2D chromatogram streams from 'Shimadzu' `.lcd` files.

Usage

```
read_shimadzu_lcd(
  path,
  what,
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  scale = TRUE,
  collapse = TRUE
)
```

Arguments

`path`	Path to 'Shimadzu' `.lcd` file.
`what`	What stream to get: current options are `pda`, chromatograms (`chroms`), `tic`, and/or peak lists (`peak_table`). If a stream is not specified, the function will default to `pda` if the PDA stream is present.
`format_out`	Class of output. Either `matrix`, `data.frame`, or `data.table`.
`data_format`	Whether to return data in `wide` (default) or `long` format.
`read_metadata`	Logical. Whether to attach metadata. Defaults to `TRUE`.
`metadata_format`	Format to output metadata. Either `chromconverter` or `raw`.
`scale`	Whether to scale the data by the scaling factor present in the file. Defaults to `TRUE`.
`collapse`	Logical. Whether to collapse lists that only contain a single element. Defaults to `TRUE`.

Details

A parser to read data from 'Shimadzu' `.lcd` files. LCD files are encoded as 'Microsoft' OLE documents. The parser relies on the olefile package in Python to unpack the files. The PDA data is encoded in a stream called `PDA 3D Raw Data:3D Raw Data`. The PDA data stream contains a segment for each retention time, beginning with a 24-byte header.

The 24 byte header consists of the following fields:

- 4 bytes: segment label (`17234`).
- 4 bytes: Little-endian integer specifying the sampling rate along the time axis for 2D streams or along the spectral axis (?) for PDA streams.
- 4 bytes: Little-endian integer specifying the number of values in the file (for 2D data) or the number of wavelength values in each segment (for 3D data).
- 4 bytes: Little-endian integer specifying the total number of bytes in the segment.
- 8 bytes of `00`.

For 3D data, Each time point is divided into two sub-segments, which begin and end with an integer specifying the length of the sub-segment in bytes. 2D data are structured similarly but with more segments. All known values in this the LCD data streams are little-endian and the data are delta-encoded. The first hexadecimal digit of each value is a sign digit specifying the number of bytes in the delta and whether the value is positive or negative. The sign digit represents the number of hexadecimal digits used to encode each value. Even numbered sign digits correspond to positive deltas, whereas odd numbers indicate negative deltas. Positive values are encoded as little-endian integers, while negative values are encoded as two's complements. The value at each position is derived by subtracting the delta at each position from the previous value.

Value

A chromatogram or list of chromatograms in the format specified by `data_format` and `format_out`. If `data_format` is `wide`, the chromatogram(s) will be returned with retention times as rows and a single column for the intensity. If `long` format is requested, two columns will be returned: one for the retention time and one for the intensity. The `format_out` argument determines whether chromatograms are returned in `matrix`, `data.frame`, or `data.table` format. Metadata will be attached to the chromatogram as attributes when `read_metadata` is `TRUE`.

Note

My parsing of the date-time format seems to be a little off, since the acquisition times diverge slightly from the ASCII file.

Author(s)

Ethan Bass

See Also

Other 'Shimadzu' parsers: `read_shimadzu()`, `read_shimadzu_gcd()`, `read_shimadzu_qgd()`, `read_sz_lcd_2d()`, `read_sz_lcd_3d()`

Examples

```
## Not run:
read_shimadzu_lcd(path)

## End(Not run)
```

`read_shimadzu_qgd` *Read 'Shimadzu' QGD files*

Description

Reads 'Shimadzu GCMSsolution' `.qgd` GC-MS data files.

Usage

```
read_shimadzu_qgd(
  path,
  what = c("MS1", "TIC"),
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  collapse = TRUE
)
```

Arguments

`path`	Path to 'Shimadzu' `.qgd` file.
`what`	What stream to get: current options are `MS1` and/or `TIC`. If a stream is not specified, the function will return both streams.
`format_out`	Class of output. Either `matrix`, `data.frame`, or `data.table`.
`data_format`	Whether to return data in `wide` (default) or `long` format.
`read_metadata`	Logical. Whether to attach metadata. Defaults to `TRUE`.
`metadata_format`	Format to output metadata. Either `chromconverter` or `raw`.
`collapse`	Logical. Whether to collapse lists that only contain a single element. Defaults to `TRUE`.

Details

The MS data is stored in the "GCMS Raw Data" storage, which contains a `MS Raw Data` stream with MS scans, a `TIC Data` stream containing the total ion chromatogram, and a `Retention Time` stream containing the retention times. All known values are little-endian. The retention time stream is a simple array of 4-byte integers. The TIC stream is a simple array of 8-byte integers corresponding to retention times stored in the retention time stream. The MS Raw Data stream is blocked by retention time. Each block begins with a header consisting of the following elements:

- scan number (4-byte integer)
- retention time (4-byte integer)
- unknown (12-bytes)
- number of bytes in intensity values (2-byte integer)
- unknown (8-bytes)

After the header, the rest of the block consists of an array of mz values and intensities. The mz values are encoded as 2-byte integers where each mz value is scaled by a factor of 20. Intensities are encoded as (unsigned) integers with variable byte-length defined by the value in the header.

Value

A chromatogram or list of chromatograms in the format specified by `data_format` and `format_out`. If `data_format` is `wide`, the chromatogram(s) will be returned with retention times as rows and a single column for the intensity. If `long` format is requested, two columns will be returned: one for the retention time and one for the intensity. The `format_out` argument determines whether chromatograms are returned as a `matrix`, `data.frame`, or `data.table`. Metadata will be attached to the chromatogram as attributes if `read_metadata` is `TRUE`.

Note

This parser is experimental and may still need some work. It is not yet able to interpret much metadata from the files.

Author(s)

Ethan Bass

See Also

Other 'Shimadzu' parsers: `read_shimadzu()`, `read_shimadzu_gcd()`, `read_shimadzu_lcd()`, `read_sz_lcd_2d()`, `read_sz_lcd_3d()`

---

read_sz_lcd_2d *Read 'Shimadzu' LCD 2D data*

Description

Reads 2D PDA data stream from 'Shimadzu' `.lcd` files.

Usage

```
read_sz_lcd_2d(
  path,
  format_out = "data.frame",
  data_format = "wide",
  read_metadata = TRUE,
  metadata_format = "shimadzu_lcd",
  scale = TRUE
)
```

Arguments

path	Path to 'Shimadzu' .lcd 2D data file.
format_out	Matrix or data.frame.
data_format	Either wide (default) or long.
read_metadata	Logical. Whether to attach metadata.
metadata_format	Format to output metadata. Either chromconverter or raw.
scale	Whether to scale the data by the value factor.

Details

A parser to read chromatogram data streams from 'Shimadzu' .lcd files. LCD files are encoded as 'Microsoft' OLE documents. The parser relies on the olefile package in Python to unpack the files. The chromatogram data is encoded in streams titled LSS Raw Data:Chromatogram Ch<#>. The chromatogram data streams begin with a 24-byte header.

The 24 byte header consists of the following fields:

- 4 bytes: segment label (17234).
- 4 bytes: Little-endian integer specifying the sampling rate (in milliseconds).
- 4 bytes: Little-endian integer specifying the number of values in the file.
- 4 bytes: Little-endian integer specifying the total number of bytes in the file.
- 8 bytes of 00s

Each segment is divided into multiple sub-segments, which begin and end with an integer specifying the length of the sub-segment in bytes. All known values in this data stream are little-endian and the data are delta-encoded. The first hexadecimal digit of each value is a sign digit specifying the number of bytes in the delta and whether the value is positive or negative. The sign digit represents the number of hexadecimal digits used to encode each value. Even numbered sign digits correspond to positive deltas, whereas odd numbers indicate negative deltas. Positive values are encoded as little-endian integers, while negative values are encoded as two's complements. The value at each position is derived by subtracting the delta at each position from the previous value.

Value

One or more 2D chromatograms from the chromatogram streams in matrix or data.frame format, according to the value of format_out. If multiple chromatograms are found, they will be returned as a list of or long format according to the value of data_format'.

Author(s)

Ethan Bass

See Also

Other 'Shimadzu' parsers: `read_shimadzu()`, `read_shimadzu_gcd()`, `read_shimadzu_lcd()`, `read_shimadzu_qgd()`, `read_sz_lcd_3d()`

`read_sz_lcd_3d` *Read 'Shimadzu' LCD 3D data*

Description

Reads 3D PDA data stream from 'Shimadzu' `.lcd` files.

Usage

```
read_sz_lcd_3d(
  path,
  format_out = "matrix",
  data_format = "wide",
  read_metadata = TRUE,
  metadata_format = "shimadzu_lcd",
  scale = TRUE
)
```

Arguments

`path`	Path to 'Shimadzu' `.lcd` 3D data file.
`format_out`	Class of output. Either `matrix`, `data.frame`, or `data.table`.
`data_format`	Either `wide` (default) or `long`.
`read_metadata`	Logical. Whether to attach metadata.
`metadata_format`	Format to output metadata. Either `chromconverter` or `raw`.
`scale`	Whether to scale the data by the value factor.

Details

A parser to read PDA data from 'Shimadzu' `.lcd` files. LCD files are encoded as 'Microsoft' OLE documents. The parser relies on the olefile package in Python to unpack the files. The PDA data is encoded in a stream called `PDA 3D Raw Data:3D Raw Data`. The PDA data stream contains a segment for each retention time, beginning with a 24-byte header.

The 24 byte header consists of the following fields:

- 4 bytes: segment label (`17234`).

- 4 bytes: Little-endian integer specifying the wavelength bandwidth (?).
- 4 bytes: Little-endian integer specifying the number of wavelength values in the segment.
- 4 bytes: Little-endian integer specifying the total number of bytes in the segment.
- 8 bytes of `00`s

Each segment is divided into two sub-segments, which begin and end with an integer specifying the length of the sub-segment in bytes. All known values in this data stream are little-endian and the data are delta-encoded. The first hexadecimal digit of each value is a sign digit specifying the number of bytes in the delta and whether the value is positive or negative. The sign digit represents the number of hexadecimal digits used to encode each value. Even numbered sign digits correspond to positive deltas, whereas odd numbers indicate negative deltas. Positive values are encoded as little-endian integers, while negative values are encoded as two's complements. The value at each position is derived by subtracting the delta at each position from the previous value.

Value

A 3D chromatogram from the PDA stream in `matrix`, `data.frame`, or `data.table` format, according to the value of `format_out`. The chromatograms will be returned in `wide` or `long` format according to the value of `data_format`.

Author(s)

Ethan Bass

See Also

Other 'Shimadzu' parsers: `read_shimadzu()`, `read_shimadzu_gcd()`, `read_shimadzu_lcd()`, `read_shimadzu_qgd()`, `read_sz_lcd_2d()`

---

read_thermoraw *Read ThermoRaw*

Description

Converts ThermoRawFiles to `mzML` by calling the ThermoRawFileParser from the command-line.

Usage

```
read_thermoraw(
  path,
  path_out = NULL,
  format_out = c("matrix", "data.frame"),
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw"),
  verbose = getOption("verbose")
)
```

Arguments

path	Path to 'Thermo' .raw file.
path_out	Path to directory to export mzML files. If path_out isn't specified, a temp directory will be used.
format_out	Class of output. Either matrix, data.frame, or data.table.
read_metadata	Logical. Whether to attach metadata. Defaults to TRUE.
metadata_format	Format to output metadata. Either chromconverter or raw.
verbose	Logical. Whether to print output from ThermoRawFileParser to the console.

Details

To use this function, the ThermoRawFileParser must be manually installed.

Value

A chromatogram in the format specified by the format_out and data_format arguments.

Side effects

Exports chromatograms in mzML format to the folder specified by path_out.

Author(s)

Ethan Bass

References

Hulstaert Niels, Jim Shofstahl, Timo Sachsenberg, Mathias Walzer, Harald Barsnes, Lennart Martens, and Yasset Perez-Riverol. ThermoRawFileParser: Modular, Scalable, and Cross-Platform RAW File Conversion. *Journal of Proteome Research* **19**, no. 1 (January 3, 2020): 537–42. doi:10.1021/acs.jproteome.9b00328.

See Also

Other external parsers: call_entab(), call_openchrom(), call_rainbow(), sp_converter(), uv_converter()

Examples

```
## Not run:
read_thermoraw(path)

## End(Not run)
```

read_varian_peaklist *Read 'Varian' peak list.*

Description

Read peak list(s) from 'Varian MS Workstation'.

Usage

```
read_varian_peaklist(path)
```

Arguments

path	Path to 'Varian' peak list file.

Value

A data.frame containing the information from the specified report.

Author(s)

Ethan Bass

See Also

Other 'Varian' parsers: read_varian_sms()

Examples

```
## Not run:
read_varian_peaklist(path)

## End(Not run)
```

read_varian_sms *Read 'Varian' SMS*

Description

Reads 'Varian Workstation' SMS files.

Usage

```
read_varian_sms(
  path,
  what = c("MS1", "TIC", "BPC"),
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = "long",
  read_metadata = TRUE,
  collapse = TRUE
)
```

Arguments

path	Path to 'Varian' .SMS files.
what	Whether to extract chromatograms (chroms) and/or MS1 data. Accepts multiple arguments.
format_out	Class of output. Either matrix, data.frame, or data.table.
data_format	Whether to return data in wide (default) or long format.
read_metadata	Logical. Whether to attach metadata. Defaults to TRUE.
collapse	Logical. Whether to collapse lists that only contain a single element. Defaults to TRUE.

Details

Varian SMS files begin with a "DIRECTORY" with offsets for each section. The first section (in all the files I've been able to inspect) is "MSData" generally beginning at byte 3238. This MSdata section is in turn divided into two sections. The first section (after a short header) contains chromatogram data. Some of the information found in this section includes scan numbers, retention times, (as 64-bit floats), the total ion chromatogram (TIC), the base peak chromatogram (BPC), ion time (µsec), as well as some other unidentified information. The scan numbers and intensities for the TIC and BPC are stored at 4-byte little-endian integers. Following this section, there is a series of null bytes, followed by a series of segments containing the mass spectra.

The encoding scheme for the mass spectra is somewhat more complicated. Each scan is represented by a series of values of variable length separated from the next scan by two null bytes. Within these segments, values are paired. The first value in each pair represents the delta-encoded mass-to-charge ratio, while the second value represents the intensity of the signal. Values in this section are variable-length, big-endian integers that are encoded using a selective bit masking based on the leading digit (d) of each value. The length of each integer seems to be determined as 1 + (d %/% 4). Integers beginning with digits 0-3 are simple 2-byte integers. If d >= 4, values are determined by masking to preserve the lowest n bits according to the following scheme:

- d = 4-5 -> preserve lowest 13 bits
- d = 6-7 -> preserve lowest 14 bits
- d = 8-9 -> preserve lowest 21 bits
- d = 10-11 (A-B) -> preserve lowest 22 bits
- d = 12-13 (C-D) -> preserve lowest 27 bits
- d = 14-15 (E-F) -> preserve lowest 28 bits (?)

Value

A chromatogram or list of chromatograms from the specified file, according to the value of `what`. Chromatograms are returned in the format specified by `format_out`.

Note

There is still only limited support for the extraction of metadata from this file format. Also, the timestamp conversions aren't quite right.

Author(s)

Ethan Bass

See Also

Other 'Varian' parsers: `read_varian_peaklist()`

Examples

```
## Not run:
read_varian_sms(path)

## End(Not run)
```

`read_waters_arw` *Read 'Waters' ASCII (.arw)*

Description

Reads 'Waters' ASCII `.arw` files.

Usage

```
read_waters_arw(
  path,
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw")
)
```

Arguments

`path`	Path to Waters `.arw` file.
`format_out`	Class of output. Either `matrix`, `data.frame`, or `data.table`.
`data_format`	Whether to return data in `wide` (default) or `long` format.
`read_metadata`	Logical. Whether to attach metadata. Defaults to `TRUE`.
`metadata_format`	Format to output metadata. Either `chromconverter` or `raw`.

Details

For help exporting files from Empower, you can consult the official documentation: How_to_export_3D_raw_data_from_Em

Value

A chromatogram in the format specified by the `format_out` and `data_format` arguments.

Author(s)

Ethan Bass

See Also

Other 'Waters' parsers: `read_waters_raw()`

`read_waters_raw` *Read 'Waters' RAW*

Description

Reads 'Waters MassLynx' (`.raw`) files into R.

Usage

```
read_waters_raw(
  path,
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw")
)
```

Arguments

`path`	Path to Waters `.raw` file.
`format_out`	Class of output. Either `matrix`, `data.frame`, or `data.table`.
`data_format`	Whether to return data in `wide` (default) or `long` format.
`read_metadata`	Logical. Whether to attach metadata. Defaults to `TRUE`.
`metadata_format`	Format to output metadata. Either `chromconverter` or `raw`.

Value

A chromatogram in the format specified by the `format_out` and `data_format` arguments.

Note

For now this parser only reads 1D chromatograms (not mass spectra or DAD data) and does not support parsing of metadata from 'Waters' RAW files.

Author(s)

Ethan Bass

See Also

Other 'Waters' parsers: `read_waters_arw()`

`sp_converter` *Converter for 'Agilent MassHunter' UV files*

Description

Converts a single chromatogram from MassHunter `.sp` format to R `data.frame` using the Aston file parser.

Usage

```
sp_converter(
  path,
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw")
)
```

Arguments

`path`	Path to file.
`format_out`	Class of output. Either `matrix`, `data.frame`, or `data.table`.
`data_format`	Whether to return data in `wide` (default) or `long` format.
`read_metadata`	Logical. Whether to attach metadata. Defaults to `TRUE`.
`metadata_format`	Format to output metadata. Either `chromconverter` or `raw`.

Value

A chromatogram in the format specified by the `format_out` and `data_format` arguments.

See Also

Other external parsers: `call_entab()`, `call_openchrom()`, `call_rainbow()`, `read_thermoraw()`, `uv_converter()`

uv_converter *Converter for 'Agilent ChemStation' UV files*

Description

Converts a single chromatogram from ChemStation `.uv` format to R `data.frame`.

Usage

```
uv_converter(
  path,
  format_out = c("matrix", "data.frame", "data.table"),
  data_format = c("wide", "long"),
  correction = TRUE,
  read_metadata = TRUE,
  metadata_format = c("chromconverter", "raw")
)
```

Arguments

`path`	Path to file
`format_out`	Class of output. Either `matrix`, `data.frame`, or `data.table`.
`data_format`	Whether to return data in `wide` (default) or `long` format.
`correction`	Logical. Whether to apply empirical correction. Defaults is TRUE.
`read_metadata`	Logical. Whether to attach metadata. Defaults to `TRUE`.
`metadata_format`	Format to output metadata. Either `chromconverter` or `raw`.

Details

Uses the Aston file parser.

Value

A chromatogram in the format specified by the `format_out` and `data_format` arguments.

See Also

Other external parsers: `call_entab()`, `call_openchrom()`, `call_rainbow()`, `read_thermoraw()`, `sp_converter()`

`write_andi_chrom` *Write ANDI chrom CDF file from chromatogram*

Description

Exports a chromatogram in ANDI (Analytical Data Interchange) chromatography format (ASTM E1947-98). This format can only accommodate unidimensional data. For two-dimensional chromatograms, the column to export can be specified using the `lambda` argument. Otherwise, a warning will be generated and the first column of the chromatogram will be exported.

Usage

```
write_andi_chrom(x, path_out, sample_name = NULL, lambda = NULL, force = FALSE)
```

Arguments

`x`	A chromatogram in (wide) format.
`path_out`	The path to write the file.
`sample_name`	The name of the file. If a name is not provided, the name will be derived from the `sample_name` attribute.
`lambda`	The wavelength to export (for 2-dimensional chromatograms). Must be a string matching one the columns in `x` or the index of the column to export.
`force`	Whether to overwrite existing files at the specified path. Defaults to `FALSE`.

Value

Invisibly returns the path to the written CDF file.

Side effects

Exports a chromatogram in ANDI chromatography format (netCDF) in the directory specified by `path_out`. The file will be named according to the value of `sample_name`. If no `sample_name` is provided, the `sample_name` attribute will be used if it exists.

Author(s)

Ethan Bass

See Also

Other write functions: `write_chroms()`, `write_mzml()`

`write_chroms` *Write chromatograms*

Description

Writes chromatograms to disk in the format specified by `export_format`: either `mzml`, `cdf`, `csv`, or `arw`.

Usage

```
write_chroms(
  chrom_list,
  path_out,
  export_format = c("mzml", "cdf", "csv", "arw"),
  what = "",
  force = FALSE,
  show_progress = TRUE,
  verbose = getOption("verbose"),
  ...
)
```

Arguments

`chrom_list`	A list of chromatograms.
`path_out`	Path to directory for writing files.
`export_format`	Format to export files: either `"mzml"`, `"cdf"`, `"csv"`, `"arw"`.
`what`	What to write. Argument to `write_cdf` and `write_mzml`. Either `"MS1"` or `"chrom"`.
`force`	Logical. Whether to overwrite existing files. Defaults to `TRUE`.
`show_progress`	Logical. Whether to show progress bar. Defaults to `TRUE`.
`verbose`	Logical. Whether to print verbose output.
`...`	Additional arguments to write function.

Value

No return value. The function is called for its side effects.

Side effects

Exports a chromatogram in the file format specified by `export_format` in the directory specified by `path_out`.

Author(s)

Ethan Bass

See Also

Other write functions: `write_andi_chrom()`, `write_mzml()`

`write_mzml` *Write mzML*

Description

This function constructs mzML files by writing XML strings directly to a file connection. While this approach is fast, it may be less flexible than methods based on an explicit Document Object Model (DOM).

Usage

```
write_mzml(
  data,
  path_out,
  sample_name = NULL,
  what = NULL,
  instrument_info = NULL,
  compress = TRUE,
  indexed = TRUE,
  force = FALSE,
  show_progress = TRUE,
  verbose = getOption("verbose")
)
```

Arguments

`data`	List of `data.frames` or `data.tables` containing spectral data.
`path_out`	The path to write the file.
`sample_name`	The name of the file. If a name is not provided, the name will be derived from the `sample_name` attribute.
`what`	Which streams to write to mzML: `"MS1"`, `"MS2"`, `"TIC"`, `"BPC"`, and/or `"DAD"`.
`instrument_info`	Instrument info to write to mzML file.
`compress`	Logical. Whether to use zlib compression. Defaults to `TRUE`.
`indexed`	Logical. Whether to write indexed mzML. Defaults to `TRUE`.
`force`	Logical. Whether to overwrite existing files at `path_out`. Defaults to `FALSE`.
`show_progress`	Logical. Whether to show progress bar. Defaults to `TRUE`.
`verbose`	Logical. Whether or not to print status messages.

Details

The function supports writing various types of spectral data including MS1, TIC (Total Ion Current), BPC (Base Peak Chromatogram), and DAD (Diode Array Detector) data. DAD spectra are written as electromagnetic radiation spectra (`MS:1000804`) using Thermo's naming convention with `controllerType=4` in the spectrum ID for compatibility with existing tools. Support for MS2 may be added in a future release.

If `indexed = TRUE`, the function will generate an indexed mzML file, which allows faster random access to spectra.

Value

Invisibly returns the path to the written mzML file.

Author(s)

Ethan Bass

See Also

Other write functions: `write_andi_chrom()`, `write_chroms()`

Index