Package: opusreader2 (via r-universe)

October 28, 2026

Title Read Spectroscopic Data from Bruker OPUS Binary Files

Version 0.6.9.9999

Description Reads data from Bruker OPUS binary files of Fourier-Transform infrared spectrometers of the company Bruker Optics GmbH & Co. This package is released independently from Bruker, and Bruker and OPUS are registered trademarks of Bruker Optics GmbH & Co. KG. <https://www.bruker.com/en/products-and-solutions/infrared-and-raman/opus-spectroscopy-software/latest-release.html>. It lets you import both measurement data and parameters from OPUS files. The main method is `read_opus()`, which reads one or multiple OPUS files into a standardized list class. Behind the scenes, the reader parses the file header for assigning spectral blocks and reading binary data from the respective byte positions, using a reverse engineering approach. Infrared spectroscopy combined with chemometrics and machine learning is an established method to scale up chemical diagnostics in various industries and scientific fields.

License MIT + file LICENSE

URL https://opusreader2.spectral-cockpit.codefloe.page/, https://codefloe.com/spectral-cockpit/opusreader2/

BugReports https://codefloe.com/spectral-cockpit/opusreader2/issues/

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Suggests knitr, rmarkdown, testthat (>= 3.0.0), progressr, mirai (>= 2.5.0)

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Language en-US

Repository https://spectral-cockpit.r-universe.dev

Date/Publication 2026-09-28 01:12:16 UTC

RemoteUrl https://github.com/spectral-cockpit/opusreader2

RemoteRef HEAD

RemoteSha 408ecf8b66a2b5ffa6803a8e44f50bcbc030d240

Contents

opus_test_dsn 2
opus_test_file 3
print.list_opusreader2 3
read_opus 4
read_opus_single 7

Index **8**

`opus_test_dsn` *Get File Paths of Sample OPUS Files Included in the Package*

Description

Utility function that retrieves the location of the sample OPUS binary file on disk.

Usage

```
opus_test_dsn()
```

Value

a character vector with the paths OPUS files included in the package

Examples

```
(dsn <- opus_test_dsn())
```

opus_test_file *Get path of a Selected Sample OPUS File Included in the Package*

Description

Utility function that retrieves the location of the sample OPUS binary file on disk.

Usage

```
opus_test_file()
```

Value

a character vector with the path to a selected single sample OPUS file

Examples

```
(fn <- opus_test_file())
```

print.list_opusreader2 *Print method for collection of OPUS spectra with class* list_opusreader2

Description

Print method for collection of OPUS spectra with class `list_opusreader2`

Usage

```
## S3 method for class 'list_opusreader2'
print(x, ...)
```

Arguments

x	List with OPUS spectra collection of class `list_opusreader2`
...	Additional arguments passed to print method

Value

Returns `x` invisibly.

read_opus *Read OPUS binary files from Bruker spectrometers*

Description

Read and parse OPUS files with spectral data from individual measurements

Usage

```
read_opus(dsn, data_only = FALSE, parallel = FALSE, progress_bar = FALSE)
```

Arguments

dsn	data source name. Can be a path to a specific file or a path to a directory. The listing of the files in a directory is recursive.
data_only	read data and parameters with FALSE per default, or only read data NULL, which only returns the parsed data.
parallel	read files in parallel via "mirai" (non-blocking parallel map). Default is FALSE. See section "Details" for more information.
progress_bar	print a progress bar. Default is FALSE.

Value

List with OPUS spectra collection of class list_opusreader2. The individual elements are individual sample measurement data extracted from the corresponding OPUS binary files, as parsed from the encoded data blocks.

Each element in list_opusreader2 contains metadata and measurement data equivalent to their names displayed in the Bruker OPUS viewer software. However, we used snake_case and standardized the naming for better output handling.

Each parsed block element is a sublist that contains

1. the binary read instructions decoded/derived from the header ($block_type, $channel_type, $text_type and $additional_type, $offset (bytes), $next_offset (bytes), $chunk_size (bytes)).

2. if parameter block, nested list of specific parameters under $parameters, which has elements named according to capitalized Bruker-internal "three-letter-string" definitions (e.g., "DPF := Data Point Format").

Possible first-level block names and information provided include:

- refl_no_atm_comp_data_param : class parameter (viewer: "Data Parameters Refl". Parameter list with metadata for refl data block (refl).
- refl_no_atm_comp: class data (spectrum; viewer: "Refl"). Unprocessed (raw; i.e, not atmospherically compensated) reflectance spectra (:= sc_sample / sc_ref). Note that this element is the untreated spectra before an eventual "atmospheric compensation" routine is applied.

- refl_data_param : class parameter (viewer: "Data Parameters Refl"). Parameter list with metadata for refl data block (metadata of reflectance spectrum; see refl output). Note that this element only results if "atmospheric compensation" was activated in the OPUS measurement settings.
- refl: class data (spectrum; viewer: "Refl"). Atmospherically compensated reflectance spectra (:= sc_sample_corr / sc_ref_corr). This result spectrum only exists if either correction of CO2 and/or water vapour bands is set in OPUS setting (proprietary algorithm; could possibly be reverse engineered). If refl exists, it has always a corresponding untreated refl_no_atm_comp spectrum (the latter present in file but not shown in the OPUS viewer, where only (final) ab is displayed)
- quant_report_refl: class parameter (viewer: "Quant Report Refl"). Quantification report for tools of multivariate calibration on refl data (i.e., PLS regression) offered in the QUANT2 OPUS package. Nested list with Bruker-internal "three-letter-string" definitions. "TIT" is the title of a nested quantification table, "E<digit>[2]" stands probably for entry, "F<digit>[2]" for field, and "Z<digit>[2]" we do not yet know what it maps to. There seems more information needed, which we can get by expanding the header parsing algorithm.
- ab_no_atm_comp_data_param : class parameter (viewer: "Data Parameters AB"). Parameter list with metadata for ab data block (spectrum; see ab output).
- ab_no_atm_comp: class data (spectrum; viewer: "Refl"). Unprocessed (raw; i.e, not atmospherically compensated) reflectance spectra (:= sc_sample/ sc_ref).
- ab_data_param : class parameter (viewer: "Data Parameters Refl"). Parameter list with metadata for ab data block (spectrum; see ab). Note that this element only results if "atmospheric compensation" was activated in the OPUS measurement settings.
- ab: class data (spectrum; viewer: "AB"). Atmospherically compensated (apparent) absorbance spectra (:= log10(1 / (sc_sample_corr / sc_ref_corr)). Only exists if either correction of CO2 and/or water vapour bands is set in OPUS setting (proprietary algorithm; could possibly be reverse engineered). If AB exists, it has always a corresponding untreated ab_no_atm_comp spectrum (the latter present in file but not shown in the OPUS viewer, where only final ab is displayed).
- quant_report_ab: class parameter (viewer: "Quant Report AB"). Quantification report for tools of multivariate calibration on ab data (i.e., PLS regression) offered in the QUANT2 OPUS package. Nested list with Bruker-internal "three-letter-string" definitions. "TIT" is the title of a nested quantification table, "E<digit>[2]" stands probably for entry, "F<digit>[2]" for field, and "Z<digit>[2]" we do not yet know what it maps to. There seems more information needed, which we can get by expanding the header parsing algorithm.
- sc_sample_data_param: class parameter (metadata; viewer: "Data Parameters ScSm"). Describes the sc_sample data block (see sc_sample).
- sc_sample: class data (spectrum). Single channel (sc) spectrum of the sample (y-axis: intensity).
- ig_sample_data_param: class parameter (metadata; viewer: "Data Parameters IgSm").
- ig_sample: class data (signal, viewer: "IgSm"). Interferogram of the sample measurement. Oscillatory signal (x-axis: optical path difference (OPD); y-axis: amplitude of the signal).
- sc_ref_data_param: class parameter (metadata; viewer: "Data Parameters ScRf"). Describes the sc_sample data block (see sc_ref).

- sc_ref: class data (spectrum; viewer: "Data Parameters IgSm"). Single channel (sc) spectrum of the reference (background material: e.g., gold; y-axis: intensity).
- ig_ref_data_param: class parameter (metadata; viewer: "Data Parameters IgRf").
- ig_ref: class data (spectrum; viewer: "IgRf"). Interferogram of the reference measurement. (background material: e.g., gold). Oscillatory signal (x-axis: optical path difference (OPD); y-axis: amplitude of the signal)
- optics: class "parameter (metadata; viewer: "Optic Parameters"). Optic setup and settings such as "Accessory", "Detector Setting" or "Source Setting".
- optics_ref: class "parameter (metadata; viewer: "Optic Parameters Rf"). Optic setup and settings specific to reference measurement such as "Accessory", "Detector Setting" or "Source Setting".
- acquisition_ref: class parameter (metadata; viewer: "Acquisition parameters Rf". Settings such as ""Additional Data Treatment", (number) of "Background Scans" or "Result Spectrum" (e.g. value "Absorbance").
- fourier_transformation_ref: class parameter (metadata; viewer: "FT - Parameters Rf"). Fourier transform parameters of the reference sample like Apodization function ("APF"), end frequency limit ("HFQ"), start frequency limit ("LFQ"), nonlinearity correction ("NLI"), phase resolution ("PHR"), phase correction mode ("PHZ"), zero filling factor ("ZFF").
- fourier_transformation: class parameter (metadata; viewer: "FT - Parameters"). Fourier transform parameters of the sample measurement. See fourier_transformation_ref for possible elements contained.
- sample: class parameter (metadata; viewer: "Sample Parameters"). Extra information such as the "Operator Name", "Experiment", or "Location".
- acquisition: class parameter (metadata; viewer. "Acquisition Parameters"). Settings of sample measurement, such as "Additional Data Treatment", (number) of "Background Scans" or "Result Spectrum" (e.g. value "Absorbance").
- instrument_ref: class parameter (metadata; viewer: "Instrument Parameters Rf"). Detailed instrument properties for the reference measurement, such as "High Folding Limit", "Number of Sample Scans", or "Actual Signal Gain 2nd Channel".
- instrument: class parameter (metadata; viewer: "Instrument Parameters"). Detailed instrument properties for the sample measurement. See instrument_ref.
- lab_and_process_param_raw: class parameter (metadata). Laboratory sample annotations such as "Product Group", "Product", "Label Product Info 1" (e.g., for sample_id), "Value Product Info 1" (e.g., the sample_id value), or the "Measurement Position Microplate" (e.g., for HTS-XT extension)
- lab_and_process_param_processed: class parameter (metadata). Laboratory sample annotations for measurements done with additional OPUS data processing methods. See lab_and_process_param_raw
- info_block: class parameter (metadata). Infos such as "Operator", measurement "Group", "Sample ID". These can e.g. be set though the OPUS/LAB sofware interface.
- history: class parameter (metadata). Character vector with OPUS command history.
- unknown: if a block-type can not be matched, no parsing is done and an empty list entry is returned. This gives you a hint that there is a block that can not yet be parsed. You can take further steps by opening an issue.

Details

`read_opus()` is the high-level interface to read multiple OPUS files at once from a data source name (`dsn`).

It optionally supports parallel reads via the `{mirai}` backend. `{mirai}` provides a highly efficient asynchronous parallel evaluation framework via the Nanomsg Next Gen (NNG), high-performance, lightweight messaging library for distributed and concurrent applications.

When reading in parallel, a progress bar can be enabled.

`read_opus_single` *Read a single OPUS file*

Description

Read a single OPUS file

Usage

```
read_opus_single(dsn, data_only = FALSE)
```

Arguments

`dsn`	source path of an opus file
`data_only`	read data and parameters with `FALSE` per default, or only read data

Value

list with parsed OPUS measurement and data blocks of the spectrum measurement contained in the OPUS file. See `read_opus()` for details on the list elements returned (first-level block names and information provided).

Index

∗ **core**
read_opus, 4

opus_test_dsn, 2
opus_test_file, 3

print.list_opusreader2, 3

read_opus, 4
read_opus(), 7
read_opus_single, 7