Package: fioRa (via r-universe)

May 23, 2026

Title Mass-Spectra Prediction Using the FIORA Model

Version 0.3.7

Date 2026-01-22

Description Provides a wrapper for the python module 'FIORA' as well as a 'shiny'-App to facilitate data processing and visualization. 'FIORA' allows to predict Mass-Spectra based on the SMILES code of chemical compounds. It is described in the Nature Communications article by Nowatzky (2025) <doi:10.1038/s41467-025-57422-4>.

License MIT + file LICENSE

Imports bslib, config, golem, InterpretMSSpectrum, rcdk, shiny, shinyjs, waiter

Encoding UTF-8

RoxygenNote 7.3.3

URL https://github.com/janlisec/fioRa

BugReports https://github.com/janlisec/fioRa/issues

Suggests reticulate, S4Vectors, Spectra, spelling, testthat (>= 3.0.0), vdiffr, withr

Config/testthat/edition 3

Language en-US

Depends R (>= 3.5)

LazyData true

NeedsCompilation no

Author Jan Lisec [aut, cre]

Maintainer Jan Lisec <jan.lisec@bam.de>

Repository https://cran.r-universe.dev

Date/Publication 2026-01-23 13:00:02 UTC

RemoteUrl https://github.com/cran/fioRa

RemoteRef HEAD

RemoteSha 079ce18dab43e8d3f26add2d6fe33d87f2305fbe

Contents

`install_fiora` *Install the python module 'fiora' into a conda environment.*

Description

This function will check and perform the installation of three components in the following order: reticulate, miniconda and fiora. It will ensure that a working conda environment 'fiora' is available. This is a prerequisite for both, run_app and run_script.

Usage

```
install_fiora(conda_name = "fiora")
```

Arguments

`conda_name`	The name of the conda environment where fiora shall be installed to.

Value

A list providing the current OS and path information on the current python executable and the fiora script.

Examples

```
## Not run:
  # this will install packages and software on your machine
  install_fiora()

## End(Not run)
```

read_fiora *Read a fiora result file (mgf) into a R object.*

Description

Allows to import a 'fioRa' output file (msp like format) to a list like format or simplified to a data frame or Spectra object.

Usage

```
read_fiora(fl, fmt = c("list", "df", "Spectra"))
```

Arguments

fl	file.
fmt	A named list object is returned if not specified otherwise. Set 'fmt=df' to simplify the return value to a data frame. Use 'fmt=Spectra' to convert to a [Spectra::Spectra()] object.

Value

A result list of length = n_compounds containing metadata fields as sub-lists and the predicted MS^2 spectrum as sub-list 'spec'. You can set parameter 'fmt' to 'df' in which case each compound list will be coerced to a data frame row (spectra will be encoded as 'mz1:int1 mz2:int2 ...' and information regarding SMILES, adduct or formula per peak will be lost).

See Also

[Spectra::Spectra()]

Examples

```
fl <- system.file("extdata/annotated_output.mgf", package = "fioRa")

# read as data.frame (return a simplified version)
str(fioRa::read_fiora(fl = fl, fmt = "df"))


  # read as a list (standard case)
  fioRa::read_fiora(fl = fl)

  # read as Spectra object (requires Spectra-package)
  if (requireNamespace("Spectra", quietly = TRUE)) {
    fioRa::read_fiora(fl = fl, fmt = "Spectra")
   }
```

run_app *Run the Shiny Application.*

Description

Will open a Shiny App in the local browser.

Usage

```
run_app(
  onStart = NULL,
  options = list(),
  enableBookmarking = NULL,
  uiPattern = "/",
  ...
)
```

Arguments

onStart	A function that will be called before the app is actually run. This is only needed for shinyAppObj, since in the shinyAppDir case, a global.R file can be used for this purpose.
options	Named options that should be passed to the runApp call (these can be any of the following: "port", "launch.browser", "host", "quiet", "display.mode" and "test.mode"). You can also specify width and height parameters which provide a hint to the embedding environment about the ideal height/width for the app.
enableBookmarking	Can be one of "url", "server", or "disable". The default value, NULL, will respect the setting from any previous calls to enableBookmarking(). See enableBookmarking() for more information on bookmarking your app.
uiPattern	A regular expression that will be applied to each GET request to determine whether the ui should be used to handle the request. Note that the entire request path must match the regular expression in order for the match to be considered successful.
...	arguments to pass to golem_opts. See '?golem::get_golem_options' for more details.

Value

A shinyApp object. Will open a Shiny App in the local browser.

run_script *Predict MS^2 fragment spectra from SMILES code.*

Description

A wrapper around the python script ‘fiora-predict‘ using the fiora open source model to generate a MS^2 spectra for a compound with known SMILES code.

Usage

```
run_script(
 x = data.frame(Name = "Example_0", SMILES = "CC1=CC(=O)OC2=CC(OS(O)(=O)=O)=CC=C12",
    Precursor_type = "[M-H]-", CE = 17, Instrument_type = "HCD"),
 min_prob = 0.001,
 annotation = FALSE,
 fiora_script = NULL,
 fmt = c("list", "df", "Spectra", "file_only"),
 file_out = NULL,
 verbose = TRUE
)
```

Arguments

x	A data frame containing columns Name (compound names), SMILES (SMILES code of the compounds, Precursor_type (currently "[M-H]-" or "[M+H]+"), CE (Collision energy) and Instrument_type (i.e. HCD).
min_prob	Minimum peak probability to be recorded in the spectrum.
annotation	Return SMILES for fragments if TRUE.
fiora_script	Path to python script fiora-predict.
fmt	Set fmt to 'df' to simplify the return value to a data frame (named list otherwise). You may also use 'file_only' to omit output to the console if you specify a valid path in 'file_out'.
file_out	Specify a path to a file to store the FIORA result permanently.
verbose	Set to FALSE to omit messages.

Details

This wrapper will generate a fiora ready input file (csv-format) based on the user parameters which is stored as a temp file. It will ensure that the current version of the fiora package is installed in a respective python environment. It will use ‘system2()‘ to run the python script ‘fiora-predict‘ and import its result back into R using function ‘read_fiora()‘. You can try different installed version of ‘fiora‘ by providing the path the the script explicitly.

Value

A list with the fiora results for the specified compound(s).

Examples

```
## Not run:
  # !!! running this example will install the python module `fiora`
  td <- fioRa::test_data
  x <- setNames(data.frame(
    t(sapply(td[2:11], function(x) { strsplit(x, ",")[[1]] })),
    strsplit(td[1], ",")[[1]]
   )
  foo <- run_script(x = x)
  foo[[1]][["spec"]]
  # modify parameters
  run_script(x = x[1,,drop=FALSE], min_prob = 0.05)

  # you may also return a Spectra object
  run_script(x = x[1,,drop=FALSE], min_prob = 0.05, annotation = TRUE, fmt = "Spectra")

  # use a different fiora environment/model
 s_pth <- "c:/Users/jlisec/AppData/Local/r-miniconda/envs/fiora-0.1.0/Scripts/fiora-predict"

  # this setup will be used internally
 fioRa:::find_fiora_predict_paths(default_path = dirname(s_pth), script_name = basename(s_pth))

  # run the script
  foo2 <- run_script(x = x, fiora_script = s_pth)

  foo2[[1]][["spec"]]

  for (i in 1:length(foo)) {
    cat("\n")
    print(names(foo)[i])#'
    print(foo[[i]][["spec"]])
    print(foo2[[i]][["spec"]])
  }


## End(Not run)
```

`test_data` *The example set of test compounds provided with FIORA.*

Description

The example set of test compounds provided with FIORA.

Usage

```
data(test_data)
```

Format

A character vector of length = 12 containing the readLines equivalent of a fiora input file in csv format, with header, 10 example compounds, defined via their SMILES, and an empty final line (or vector element respectively).

Source

https://github.com/BAMeScience/fiora/blob/main/examples/example_input.csv

Index