Package 'rTANDEM'

October 31, 2014

Type Package

Title Interfaces the tandem protein identification algorithm in R

Version 1.6.0

Date 2014-03-31

SystemRequirements rTANDEM uses expat and pthread libraries. See the README file for details.

Author@R c(person(``Frederic", ``Fournier",email=``frederic.fournier@crchuq.ulaval.ca"), person(``Charles",``Joly Beauparlant", email=``charles.joly-b@crchul.ulaval.ca"),person(``Rene Paradis", email=``rene.paradis@genome.ulaval.ca"),person(``Arnaud", ``Droit",email=``arnaud.droit@crchuq.ulaval.ca"))

Author Frederic Fournier <frederic.fournier@crchuq.ulaval.ca>, Charles Joly Beauparlant <charles.joly-beauparlant@crchul.ulaval.ca>, Rene Paradis <rene.paradis@genome.ulaval.ca>, Arnaud Droit <arnaud.droit@crchuq.ulaval.ca>

Maintainer Frederic Fournier <frederic.fournier@crchuq.ulaval.ca>

Description This package interfaces the tandem protein identification algorithm in R. Identification can be launched in the X!Tandem style, by using as sole parameter the path to a parameter file. But rTANDEM aslo provides extended syntax and functions to streamline launching analyses, as well as function to convert results, parameters and taxonomy to/from R. A related package, shinyTANDEM, provides visualization interface for result objects.

biocViews MassSpectrometry, Proteomics

License Artistic-1.0 | file LICENSE

Imports methods

Depends XML, Rcpp, data.table (>= 1.8.8)

Suggests biomaRt

LinkingTo Rcpp

R topics documented:

`rTANDEM-package` *An R encapsulation of X!TANDEM ('Jackhammer' release, 2013.06.15)*

Description

X!Tandem is an open source software for protein identification by tandem mass spectrometry, and rTANDEM encapsultates this software in R. rTANDEM provides a basic encapsulation of X!Tandem: it has a function that takes as an argument the path to an X!Tandem style parameter file and return the path to an X!tandem style output file. The package also presents functions to transform parameters or results files into R objects and vice versa. Some functions are also available to examine the results within R. An associated package, shinyTANDEM, provides a graphical interface to visualize results objects.

Details

Package:	rTANDEM
Type:	Package
Version:	1.2.3
Date:	2013-11-07
License:	Artistic License 1.0

Author(s)

Authors: Frederic Fournier, Charles Joly Beauparlant, Rene Paradis, Arnaud Droit Maintainer: Frederic Fournier <frederic.fournier@crchuq.ulaval.ca>, Charles Joly Beauparlant <charles.joly-beauparlant@crchul.ulaval.ca>

References

Robertson Craig and Ronald C. Beavis, TANDEM: matching proteins with mass spectra, Bioinformatics, 2004, 20 1466-7. http://www.thegpm.org/tandem/

Examples

```
# X!Tandem call style: we call tandem(input) on a single
# rTParam object.

# We create rTParam and from X!Tandem xml files
# located in the installation folder:
param <- GetParamFromXML(system.file("extdata/input.xml", package="rTANDEM"))

# We create a rTTaxo object and identify a database for yeast
taxonomy <- rTTaxo(
  taxon="yeast",
  format="peptide",
  URL= system.file("extdata/fasta/scd.fasta.pro", package="rTANDEM")
  )

# We will adjust those two objects to use one another and to use,
# the path of some data and configuration files located
# in the installation folder:
param <- setParamValue(param, list path, taxonomy information, taxonomy)
param <- setParamValue(param, list path, default parameters,
  value=system.file("extdata/default_input.xml", package="rTANDEM"))
param <- setParamValue(param, spectrum, path,
  value=system.file("extdata/test_spectra.mgf", package="rTANDEM"))
param <- setParamValue(param, output, xsl path,
  value=system.file("extdata/tandem-input-style.xsl", package="rTANDEM"))
param <- setParamValue(param, output, path,
  value=paste(getwd(), "output.xml", sep="/"))

# This is the main command to run rTANDEM. The output will be
# written to a file in the working directory and the function
# returns the path to this file.
output.file <- tandem(param)
output.file
```

`accessors` *Extract information from rTANDEM result object*

Description

The `GetProteins`, `GetPeptides` and `GetDegeneracy` functions are used to extract information from the rTANDEM result object.

Usage

```
GetProteins(results, log.expect=0, min.peptides=1L)
GetPeptides(protein.uid, results, expect=1, score=0)
GetDegeneracy(peptide.id, results)
```

Arguments

results	An object of the class rTResult that contains the result of an rTANDEM or X!Tandem analysis.
log.expect	X!Tandem provides a score of protein identification that is presented in terms of the log of the expect value of the identification. This score can be used as a threshold to discard low confidence identifications from the protein list.
expect	The expect value of peptide identification. This statistic can be used as a threshold to discard low confidence identifications from the peptide list.
min.peptides	The number of peptides involved in the identification of a given protein is computed. This number can be used as a threshold to discard identifications based on too few peptides from the protein list.
protein.uid	The tandem identifier of the protein (a numeric).
peptide.id	The tandem identifier of the peptide (a character).
score	The tandem score of the peptide identification. This score can be used as a threshold to discard low confidence identifications from the peptide list.

Value

`GetProteins` and `GetDegeneracy` return a data.table of proteins. `GetPeptides` returns a data.table of peptides with their ptm (post-translational modifications). Note that this table is generated through a merge of the peptide table and the ptm table: hence, if peptides has two ptm, it will occupy to rows in the resulting data.table.

Examples

```
# To show how to use the accessor functions, we need an rTANDEM result.
# We can produce one by running the example from the rTANDEM function,
# and reading it to R with GetResultsFromXML:
# output.file.path <- example(rTANDEM)

results <- GetResultsFromXML(output.file.path[[1]])

# To get a data.table of the proteins identified by at least 2 peptides
# and with an expect value of 0.05 or better:
proteins <- GetProteins(results, log.expect=-1.3, min.peptides=2)
proteins[, -c(4,5), with=FALSE] # Colums are removed for better display

# To get a list of the peptides used to identify the first protein
# (YCR012W, uid=576):
peptides <- GetPeptides(protein.uid="576", results)
peptides

# To get the list of proteins to which proteins a peptide belongs:
# (If a peptide belongs to more than one protein, it should not be
# used for quantification, as a biomarker or a MRM target.)
proteins.of.the.peptide <- GetDegeneracy(peptide.id="169.1.1",
results)
proteins.of.the.peptide[,label]
```

converters *Converts X!Tandem xml files to R objects and vice versa*

Description

Functions like `GetTaxoFromXML("pathToXML")`, `GetParamFromXML("pathToXML")`, `GetResultsFromXML(pathToXML)` creates R object from X!Tandem-style xml files. The functions `WriteParamToXML(paramObject)` and `WriteTaxoToXML(paramObject)` will create an X!Tandem-style xml file from an R object.

Usage

```
GetTaxoFromXML(xml.file)
GetParamFromXML(xml.file)
GetResultsFromXML(xml.file)
WriteParamToXML(param, file, embeddedParam=c("write", "skip", "merge"),
embeddedTaxo=c("write","skip") )
WriteTaxoToXML(taxo, file)
```

Arguments

`xml.file`	The path to the xml file that is to be read.
`file`	The name of the xml file that is to be created.
`param`	An object of class rTParam that will be used to create the corresponding xml file.
`taxo`	A rTTaxo object whose content will be written to an xml file.
`embeddedParam`	The behaviour to adopt if a rTParam object contains an embedded rTParam object in the "list path, default parameters" slot. The option "merge" will merge the two object together. The option "write" will call WriteParamToXML on the embedded rTParam object and write it to the the given file name plus suffixe "_default_param". It will then replace the embedded object by its path in the original object. The option "skip" will just ignore this slot.
`embeddedTaxo`	The behaviour to adopt if the rTParam object contains an embedded rTTaxo object in the "list path, taxonomy information" slot. The option "write" will call WriteTaxoToXML on the object and write it to the input file name plus suffixe "_taxonomy". It will then replace the rTTaxo object by its path in the container rTParam object. The option "skip" will just ignore this slot.

Value

'WriteParamToXML' and 'WriteTaxoToXML' have no return value: they are used for their side-effect of creating an xml file. 'GetTaxoFromXML' returns an object of the S3 class rTTaxo, 'GetParamFromXML' return an object of the S3 class rTParam, and 'GetResultsFromXML' returns and object of the S4 class rTResult.

Examples

```
## Not run:
# To write a parameter or taxonomy object to a single xml file:
WriteParamToXML(parameter_object, file="parameter_file")
# Produces the file parameter_file.

# To write a parameter object which has an embedded default
# parameter set to two xml files:
WriteParamToXML(parameter_object, file="param_file",
embeddedParam="write")
# Produces the files param_file and param_file_default_param.

# To write a parameter object that contains an embedded taxonomy to
# two different xml files:
WriteParamToXML(parameter_object, file="parameter_file",
embeddedTaxo="write")
# Produces the files parameter_file and parameter_file_taxonomy.

# To write a taxonomy object to a n xml file:
WriteTaxoToXML(taxonomy_object, file="taxonomy")
# Produces the file taxonomy.

# To read a taxonomy file in R:
taxonomy <- GetTaxoFromXML("taxonomy.xml")
# Read the xml file and create a taxonomy object of class rTTaxo.

# To read a parameter file in R:
param <- GetParamFromXML("parameters.xml")
# Read the xml file and create a parameter object of class rTParam.

# To read a result file in R:
results <- GetResultsFromXML("output.xml")
# Read the output from X!Tandem and creates a R object of class
# rTResult.


## End(Not run)
```

---

parameters *Creates and manipulates parameter objects for rTANDEM*

Description

Those functions instantiate parameter objects for rTANDEM and manipulate their values. Many of those functions give default values, either for general settings (e.g. setParamDefault), for instrument specific settings (e.g. setParamOrbitrap), or for function specific settings (e.g. setParamPTMTreeSearch).

Usage

```
rTParam()
setParamValue(param, category, parameter=NULL, value)
setParamDefault(param=NULL)
setParamPTMTreeSearch(param=NULL)
setParamOrbitrap(param=NULL)
setParamIonTrap(param=NULL)
setParamQuadTof05da(param=NULL)
setParamQuadTof100ppm(param=NULL)
rTTaxo(taxon=NA, format=NA, URL=NA)
addTaxon(taxonomy=NULL, taxon, format="peptide", URL)
## S3 method for class rTParam
print(x, ...)
```

Arguments

param	A parameter object of class rTParam.
x	A parameter object of class rTParam.
...	Further arguments passed to or from other methods.
category	A string representing a category of parameters (e.g. "output", "protein").
parameter	A string representing the name of a parameter.
value	The value to give to the parameter.
taxon	A string representing the name of a taxon (e.g. "homo sapiens").
taxonomy	A taxonomy object of class rTTaxo.
format	A string representing the type of the database. The four types are: "peptide", "saps", "mods" and "spectrum".
URL	A string representing the full path to the database file.

Value

The functions rTParam, and all functions setParam..., return an object of S3 class rTParam. The functions rTTaxo and addTaxon return an object of S3 class rTTaxo. print.rTParam is used for its side effect of displaying a parameter object.

References

thegpm.org/TANDEM/api/

See Also

GetParamFromXML, GetTaxoFromXML, WriteTaxoToXML, WriteParamToXML.

Examples

```
# Initialize an empty parameter object:
param <- rTParam()
print.rTParam(param)

# Set general values.
param <- setParamDefault(param)

# Add instrument specific values for a LTQ mass spectrometer.
param <- setParamIonTrap(param)
print.rTParam(param)
```

`rTResult-class` *Class "rTResult" and "rTResult_s"*

Description

A rTResult object is designed to contain the information of a X!Tandem analysis and allow data mining of this information. rTResult_s is a sub-class that adds a data.table slot for holding spectra.

Objects from the Class

Objects can be created by calls to `new("rTResult", ...)` or `new("rTResult_s", ...)`.

Slots

`result.file`: Object of class `"character"` ~~

`proteins`: Object of class `"data.table"` ~~

`peptides`: Object of class `"data.table"` ~~

`ptm`: Object of class `"data.table"` ~~

`spectra`: Object of class `"data.table"` ~~

`used.parameters`: Object of class `"data.frame"` ~~

`unused.parameters`: Object of class `"vector"` ~~

`sequence.source.paths`: Object of class `"vector"` ~~

`estimated.false.positive`: Object of class `"integer"` ~~

`total.peptides.used`: Object of class `"integer"` ~~

`total.proteins.used`: Object of class `"integer"` ~~

`total.spectra.assigned`: Object of class `"integer"` ~~

`total.spectra.used`: Object of class `"integer"` ~~

`total.unique.assigned`: Object of class `"integer"` ~~

`start.time`: Object of class `"character"` ~~

`xtandem.version`: Object of class `"character"` ~~

`quality.values`: Object of class `"vector"` ~~

`nb.input.models`: Object of class "integer" ~~

`nb.input.spectra`: Object of class "integer" ~~

`nb.partial.cleavages`: Object of class "integer" ~~

`nb.point.mutations`: Object of class "integer" ~~

`nb.potential.C.terminii`: Object of class "integer" ~~

`nb.potential.N.terminii`: Object of class "integer" ~~

`nb.unanticipated.cleavages`: Object of class "integer" ~~

`initial.modelling.time.total`: Object of class "numeric" ~~

`initial.modelling.time.per.spectrum`: Object of class "numeric" ~~

`load.sequence.models.time`: Object of class "numeric" ~~

`refinement.per.spectrum.time`: Object of class "numeric" ~~

Methods

No methods defined with class "rTResult" in the signature.

Author(s)

Authors: Frederic Fournier, Charles Joly Beauparlant, Rene Paradis, Arnaud Droit Maintainer: Frederic Fournier <frederic.fournier@crchuq.ulaval.ca>, Charles Joly Beauparlant <charles.joly-beauparlant@crchul.ulaval.ca>

Examples

```
showClass("rTResult")
```

---

tandem *Calls X!TANDEM ('Jackhammer' release, 2013.06.15) from R objects or xml files*

Description

The function `tandem(input)` takes a rTParam object or the path of a parameter file as argument and calls X!Tandem on it. The function `rtandem(data.file, taxon, taxonomy, default.parameters)` is a wrapper that can be used to circumvent the need for a rTParam input object (or of an xml input file).

Usage

```
tandem(input)
rtandem(data.file, taxon, taxonomy, default.parameters, output.path=NA)
```

Arguments

input	A path to a X!Tandem style parameter file or a rTParam object.
data.file	The path to the file containing the raw data to be analysed (in 'DTA', 'PKL' or 'MGF' format).
taxon	A string containing the taxon to be used for the analysis (e.g. "yeast" or "Homo sapiens").
taxonomy	Either a rTTaxo object or the path to a X!Tandem style taxonomy xml file.
default.parameters	Either a rTParam object containing the default parameters to be used, or the path to a X!Tandem style default-parameters xml file.
output.path	The path and name of the output file. If this name ends by ".xml" and the option 'path hashing' is enabled, a timestamp will be inserted just before the ".xml".

Value

Both `tandem(input)` and `rtandem(data.file, taxon, taxonomy, default.parameters)` returns the path of the xml output file generated.

Author(s)

Authors: Frederic Fournier, Charles Joly Beauparlant, Rene Paradis, Arnaud Droit

Maintainer: Frederic Fournier <frederic.fournier@crchuq.ulaval.ca>, Charles Joly Beauparlant <charles.joly-beauparlant@crchul.ulaval.ca>

References

Robertson Craig and Ronald C. Beavis, TANDEM: matching proteins with mass spectra, Bioinformatics, 2004, 20 1466-7. http://www.thegpm.org/tandem/

Examples

```
# X!Tandem call style: we call tandem(input) on a single
# rTParam object.

# We create rTParam from an X!Tandem xml file
# located in the installation folder:
param <- GetParamFromXML(system.file("extdata/input.xml", package="rTANDEM"))

# We create a rTTaxo object and identify a database for yeast
taxonomy <- rTTaxo(
  taxon="yeast",
  format="peptide",
  URL= system.file("extdata/fasta/scd.fasta.pro", package="rTANDEM")
  )

# We will adjust those two objects to use one another and to use
# the path of some data and configuration files located
# in the installation folder:
param <- setParamValue(param, list path, taxonomy information, taxonomy)
```

```
param <- setParamValue(param, list path, default parameters,
  value=system.file("extdata/default_input.xml", package="rTANDEM"))
param <- setParamValue(param, spectrum, path,
  value=system.file("extdata/test_spectra.mgf", package="rTANDEM"))
param <- setParamValue(param, output, xsl path,
  value=system.file("extdata/tandem-input-style.xsl", package="rTANDEM"))
param <- setParamValue(param, output, path,
  value=paste(getwd(), "output.xml", sep="/"))

# This is the main command to run rTANDEM. The output will be
# written to a file in the working directory and the function
# returns the path to this file.
output.file <- tandem(param)
output.file
```

`visualizers` *Plot spectrum from a rTResult object*

Description

Functions like `GetTaxoFromXML("pathToXML")`, `GetParamFromXML("pathToXML")`, `GetResultsFromXML(pathToXML)` creates R object from X!Tandem-style xml files. The functions `WriteParamToXML(paramObject)` and `WriteTaxoToXML(paramObject)` will create an X!Tandem-style xml file from an R object.

Usage

```
ms2.plot(spectrum.id, result)
```

Arguments

`spectrum.id`	The id of the spectrum to be plotted (from the field result@spectra$id).
`result`	A result object of class rTResult_s.

Value

'plot.ms2' returns a plot of the spectrum.

Examples

```
require(rTANDEM)
result <- GetResultsFromXML(
  system.file("extdata/result.xml", package="rTANDEM")
)

## Get the first spectra of the dataset and plot it:
spectrum.id <- result@spectra$id[[1]]
ms2.plot(spectrum.id, result)
```

Index