Package: rtemis.bio (via r-universe)

May 13, 2026

Title rtemis Bio-informatics

Version 0.4.2

Date 2026-02-14

Description Bio-informatics utilities extending rtemis package.

URL `https://rtemis.bio.rtemis.org`

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports cli, data.table, rtemis, rtemis.utils, stats, S7, utils

Suggests biomaRt, colorspace, dplyr, httr, jsonlite, plotly, seqinr, testthat (>= 3.0.0)

Additional_repositories `https://rtemis-org.r-universe.dev`

Config/testthat/edition 3

Repository https://ictml-project.r-universe.dev

Date/Publication 2026-02-15 23:56:07 UTC

RemoteUrl https://github.com/rtemis-org/rtemis.bio

RemoteRef HEAD

RemoteSha 5d26225d2a463a1b804167816454b2b53b6accaa

Contents

`rtemis.bio-package` *rtemis.bio: Bioinformatics ops*

Description

Bioinformatics utilities

Author(s)

Maintainer: E.D. Gennatas `<gennatas@gmail.com>` (ORCID)

See Also

Useful links:

- `https://rtemis.bio.rtemis.org`

`aa_sub` *Perform amino acid substitutions*

Description

Perform amino acid substitutions

Usage

```
aa_sub(x, substitutions, verbosity = 1L)
```

Arguments

`x`	Character vector: Amino acid sequence. e.g. `"ARND"` or `c("A", "R", "N", "D")`.
`substitutions`	Character vector: Substitutions to perform in the format "OriginalPositionNew", e.g. `c("C291A", "C322A")`.
`verbosity`	Integer: Verbosity level.

Value

Character vector with substitutions performed.

Author(s)

EDG

`aggregate.Xt` *Aggregate method for* `Xt` *object*

Description

Aggregate method for `Xt` object

Usage

```
## S3 method for class 'Xt'
aggregate(
  x,
  groupname,
  fn = mean,
  backend = getOption("rtemis_backend", "base"),
  ...
)
```

Arguments

`x`	`Xt` object.
`groupname`	Character: Grouping variable.
`fn`	Function: Function to apply to each group.
`backend`	Character: "base", "data.table", or "dplyr"; backend to use for aggregation.
`...`	Additional arguments passed to `fn`.

Author(s)

EDG

`as_A3` *as_A3*

Description

as_A3

Usage

```
as_A3(x)
```

Arguments

`x`	List: Named list with elements `Sequence`, `Annotations`, `UniprotID`. `Annotations` is a named list with possible elements `Site`, `Region`, `PTM`, `Cleavage_site`, `Variant`, `Description`, `Reference`.

Value

`A3` object.

Author(s)

EDG

`as_Xt` *as_Xt*

Description

as_Xt

Usage

```
as_Xt(x)
```

Arguments

`x`	Object to convert to `Xt`.

Value

`Xt` object.

Author(s)

EDG

`create_A3` *Create an* `A3` *object*

Description

Creates an `A3` object given amino acid sequence and annotations.

Usage

```
create_A3(
  sequence,
  site = NULL,
  region = NULL,
  ptm = NULL,
  cleavage_site = NULL,
  variant = NULL,
  uniprotid = NULL,
  description = NULL,
  reference = NULL
)
```

Arguments

`sequence`	Character: Amino acid sequence.
`site`	Named list of vectors of integer indices of sites, e.g. `list("N-terminal repeat" = c(46, 47, 52), "Microtubule binding domain" = c(244, 245, 246))`
`region`	Named list of integer indices, e.g. `list("Phosphodegron" = c(46, 47, 48, 49, 50, 51), "KXGS" = c(259, 260, 261, 262))` or character vectors with index range of regions in format `start:end`, e.g. `list(Phosphodegron = c("46:51", "149:154"), KXGS = c("259:262", "290:293"))`
`ptm`	Named list of vectors with indices of post-translational modifications, e.g. `list("Phosphorylation" = c(17, 18, 29, 30), "Acetylation" = c(148, 150, 163))`
`cleavage_site`	Named list of cleavage sites, e.g. `list(CTSL = c(54, 244, 319), CTSD = c(340, 391, 426))`
`variant`	List of lists with variant information. Each list must contain a `position` element.
`uniprotid`	Character: Uniprot ID.
`description`	Character: Description of the data / experiment.
`reference`	Character: Link to reference (journal publication, preprint, etc.)

Details

We choose to keep NULL elements as empty lists in JSON, since we want users to be able to easily add annotations, whether programmaticaly, using a web app, or manually.

Value

`A3` object

Author(s)

EDG

`create_Xt` *Create an `Xt` object*

Description

Creates an `Xt` object from time series data.

Usage

```
create_Xt(
  x,
  y,
  x2 = NULL,
  y2 = NULL,
  zt = NULL,
  shade = NULL,
  group = NULL,
  xunits = NULL,
  yunits = NULL,
  y2units = NULL,
  description = NULL,
  reference = NULL
)
```

Arguments

`x`	Named list of datetime vectors.
`y`	Named list of numeric vectors: When plotted, these will correspond to the left y-axis.
`x2`	Named list of datetime vectors: When plotted, these will correspond to the right x-axis. If not provided, `x` will be used for both `y` and `y2`.
`y2`	Named list of numeric vectors: When plotted, these will correspond to the right y-axis.
`zt`	Numeric vector: Zeitgeber time. If provided, this will be used to label x-axis ticks. Assumes a single datetime vector in `x`. Elements in `zt` must correspond to elements in `x`.
`shade`	Binary vector: `0` indicates no shading, `1` indicates shading. If provided, this will be used to shade the plot.

`group`	Named list of factors: Grouping variable(s).
`xunits`	Character: Units for `x`.
`yunits`	Character: Units for `y`.
`y2units`	Character: Units for `y2`.
`description`	Character: Description of the data / experiment.
`reference`	Character: Link to reference (journal publication, preprint, etc.)

Value

`Xt` object

Author(s)

EDG

`gene2sequence` *Get the sequence of a gene*

Description

Get the sequence of a gene

Usage

```
gene2sequence(
  gene,
  organism = "hsapiens",
  biomart = "ensembl",
  host = "https://www.ensembl.org",
  seq_type = "coding",
  verbosity = 1L
)
```

Arguments

`gene`	Character: Gene name.
`organism`	Character: Organism name.
`biomart`	Character: Biomart name.
`host`	Character: Host address.
`seq_type`	Character: Sequence type to retrieve. See biomaRt::getSequence.
`verbosity`	Integer: Verbosity level.

Value

data.frame with columns "gene", "ensembl_transcript_id" and "sequence".

Author(s)

EDG

`get_alphafold` *Get AlphaFold info for a given UniProt ID*

Description

Get AlphaFold info for a given UniProt ID

Usage

```
get_alphafold(uniprotid)
```

Arguments

`uniprotid`	Character: UniProt ID.

Value

data frame with AlphaFold info.

Author(s)

EDG

`light_dark_ratio` *Calculate light/dark ratio for `Xt` object*

Description

Calculates light/dark ratio for each `y` and `y2` timeseries in an `Xt` object.

Usage

```
light_dark_ratio(
  x,
  groupname = "Lights",
  fn = mean,
  backend = getOption("rtemis_backend", "data.table"),
  ...
)
```

Arguments

`x`	`Xt` object.
`groupname`	Character: Grouping variable.
`fn`	Function: Function to apply to each group.
`backend`	Character: "base", "data.table", or "dplyr"; backend to use for aggregation.
`...`	Additional arguments passed to `fn`.

Value

data.frame with columns for group and summary statistic.

Author(s)

EDG

`plot.A3` *Plot method for `A3` object*

Description

Plot method for `A3` object

Usage

```
## S3 method for class 'A3'
plot(x, ...)
```

Arguments

`x`	`A3` object.
`...`	Additional arguments passed to rtemis::draw_protein.

Value

`plotly` object.

Author(s)

EDG

`plot.Xt` *Plot method for `Xt` object*

Description

Plot method for `Xt` object

Usage

```
## S3 method for class 'Xt'
plot(x, ...)
```

Arguments

`x`	`Xt` object.
`...`	Additional arguments passed to rtemis::draw_xt.

Value

`plotly` object.

Author(s)

EDG

`read_A3json` *Read `A3` object from JSON file*

Description

Read `A3` object from JSON file

Usage

```
read_A3json(filepath, verbosity = 0L)
```

Arguments

`filepath`	Character: Path to JSON file.
`verbosity`	Integer: if greater than 0, print messages.

Value

`A3` object.

Author(s)

EDG

`read_Xtjson` *Read `Xt` object from JSON file*

Description

Read `Xt` object from JSON file

Usage

```
read_Xtjson(filepath, verbosity = 0L)
```

Arguments

`filepath`	Character: Path to JSON file.
`verbosity`	Integer: if greater than 0, print messages.

Details

Note that factors saved under `group` are written as character by write_Xtjson and when they are read back in, they are converted back to factors using factor. This means that the levels will be set alphabetically. If needed, reorder them after reading in the JSON file using factor.

Value

`Xt` object.

Author(s)

EDG

`summary.A3` *Summary method for `A3` object*

Description

Summary method for `A3` object

Usage

```
## S3 method for class 'A3'
summary(object, ...)
```

Arguments

`object`	`A3` object.
`...`	Not used

Value

Called for side effects, prints summary to console.

Author(s)

EDG

`uniprot_get` *Get protein sequence from UniProt*

Description

Get protein sequence from UniProt

Usage

```
uniprot_get(
  accession,
  baseURL = "https://rest.uniprot.org/uniprotkb",
  verbosity = 1L
)
```

Arguments

`accession`	Character: UniProt Accession number - e.g. "Q9UMX9".
`baseURL`	Character: UniProt rest API base URL.
`verbosity`	Integer: Verbosity level.

Value

List with three elements: Identifier, Annotation, and Sequence.

Author(s)

E.D. Gennatas

Examples

```
## Not run:
mapt <- uniprot_get("Q9UMX9")

## End(Not run)
```

`write_A3json` *Write `A3` object to JSON file*

Description

Write `A3` object to JSON file

Usage

```
write_A3json(x, filepath, overwrite = FALSE)
```

Arguments

`x`	`A3` object, as created by `as_A3()`.
`filepath`	Character: Path to save JSON file.
`overwrite`	Logical: If TRUE, overwrite existing file.

Value

Nothing. Writes JSON file.

Author(s)

EDG

`write_Xtjson` *Write `Xt` object to JSON file*

Description

Write `Xt` object to JSON file

Usage

```
write_Xtjson(x, filepath, overwrite = FALSE)
```

Arguments

`x`	`Xt` object, as created by as_Xt.
`filepath`	Character: Path to save JSON file.
`overwrite`	Logical: If TRUE, overwrite existing file.

Value

Nothing. Writes JSON file.

Author(s)

EDG

Index