# Package: RBMRB (via r-universe)

October 14, 2026

**Title** Interface to the Biological Magnetic Resonance Data Bank ('BMRB')

**Version** 3.0.0

**Date** 2026-10-01

**Description** A comprehensive interface to the Biological Magnetic Resonance Data Bank ('BMRB') <https://bmrb.io>, mirroring all functionality of the 'PyBMRB' 'Python' package (v3.0). Provides four modules: (1) 'ChemicalShift' for fetching per-entry chemical shift data from the 'BMRB' 'REST' 'API' or local 'NMR-STAR' files; (2) 'ChemicalShiftStatistics' for database-wide statistical queries; (3) Spectra for simulating '1H-15N' 'HSQC', '1H-13C' 'HSQC', '1H-1H' 'TOCSY', and generic 2D 'NMR' spectra as interactive plots; and (4) Histogram for chemical shift distribution visualizations with optional filtering by pH, temperature, and ambiguity code. All interactive plots are rendered via 'plotly'. Uses the 'BMRB' REST API v2 <https://api.bmrb.io/v2/>.

**License** GPL-3

**URL** https://github.com/bmrb-io/RBMRB

**BugReports** https://github.com/bmrb-io/RBMRB/issues

**Encoding** UTF-8

**Depends** R (>= 4.1.0)

**Imports** httr2 (>= 1.0.0), jsonlite (>= 1.8.0), plotly (>= 4.10.0), cli (>= 3.4.0)

**Suggests** testthat (>= 3.0.0), knitr, rmarkdown, htmlwidgets, withr

**Config/testthat/edition** 3

**VignetteBuilder** knitr

**Config/roxygen2/version** 8.0.0

**NeedsCompilation** no

**Author** Kumaran Baskaran [aut, cre] (ORCID: <https://orcid.org/0000-0002-0418-0286>)

**Maintainer** Kumaran Baskaran <baskaran@uchc.edu>

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

**Date/Publication** 2026-10-14 17:21:33 UTC

**RemoteUrl** https://github.com/cran/RBMRB

**RemoteRef** HEAD

**RemoteSha** ab40c34969b405e8d44225aa205f879377a549b2

# Contents

bmrb_citation . . . 3
bmrb_debug_entry . . . 3
bmrb_fetch_entry . . . 4
bmrb_from_pdb . . . 4
bmrb_list_entries . . . 5
bmrb_pdb_ids . . . 6
bmrb_search . . . 6
bmrb_status . . . 7
bmrb_test . . . 7
bmrb_upload_entry . . . 8
c13hsqc . . . 8
conditional_cs_hist . . . 10
create_2d_peaklist . . . 11
create_c13hsqc_peaklist . . . 12
create_n15hsqc_peaklist . . . 13
create_tocsy_peaklist . . . 13
cs_from_bmrb . . . 14
cs_from_file . . . 15
cs_hist . . . 16
cs_hist2d . . . 17
cs_peek . . . 18
cs_stats . . . 19
cs_summary . . . 20
cs_table . . . 20
cs_to_df . . . 21
export_peak_list . . . 21
filter_cs_outliers . . . 22
generic_2d . . . 23
get_2d_cs_data . . . 24
get_cs_data . . . 25
get_cs_from_bmrb_db . . . 26
n15hsqc . . . 27
RBMRB_version . . . 28
standard_amino_acids . . . 29
standard_nucleic_acids . . . 29
tocsy . . . 30

**Index** **32**

## bmrb_citation — *Get the citation for a BMRB entry*

### Description

Get the citation for a BMRB entry

### Usage

```
bmrb_citation(bmrb_id, format = "bibtex")
```

### Arguments

| | |
|---|---|
| bmrb_id | A single BMRB entry ID. |
| format | Output format: "bibtex" (default), "json-ld", or "text". |

### Value

A character string containing the citation.

### Examples

```
## Not run:
cat(bmrb_citation(15060, format = "text"))

## End(Not run)
```

## bmrb_debug_entry — *Diagnose BMRB entry fetch problems*

### Description

Fetches a BMRB entry using both the zlib-compressed and plain JSON endpoints and reports what is found at each step. Run this when cs_from_bmrb returns no data.

### Usage

```
bmrb_debug_entry(bmrb_id)
```

### Arguments

| | |
|---|---|
| bmrb_id | A single BMRB entry ID (integer or character). |

### Value

Invisibly, the parsed entry JSON list, or NULL on failure.

### Examples

```
## Not run:
bmrb_debug_entry(15060)

## End(Not run)
```

---

## bmrb_fetch_entry *Fetch raw metadata for a BMRB entry*

---

### Description

Fetch raw metadata for a BMRB entry

### Usage

```
bmrb_fetch_entry(bmrb_ids)
```

### Arguments

| | |
|---|---|
| bmrb_ids | A single BMRB entry ID or a character/integer vector. |

### Value

A named list of raw entry metadata (API JSON as R list).

### Examples

```
## Not run:
meta <- bmrb_fetch_entry(15060)
names(meta[["15060"]])

## End(Not run)
```

---

## bmrb_from_pdb *Get BMRB IDs associated with a PDB entry*

---

### Description

Get BMRB IDs associated with a PDB entry

### Usage

```
bmrb_from_pdb(pdb_id)
```

### Arguments

| | |
|---|---|
| pdb_id | A PDB ID string (e.g. "2JM0"). |

### Value

A data.frame with columns `bmrb_id`, `match_type`, and `comment`.

### Examples

```
## Not run:
bmrb_from_pdb("2JM0")

## End(Not run)
```

---

## bmrb_list_entries *List all BMRB entry IDs*

---

### Description

Returns all public entry IDs in the specified BMRB database.

### Usage

```
bmrb_list_entries(database = "macromolecules")
```

### Arguments

| | |
|---|---|
| `database` | One of `"macromolecules"` (default), `"metabolomics"`, or `"chemcomps"`. |

### Value

A character vector of BMRB entry IDs.

### Examples

```
## Not run:
ids <- bmrb_list_entries()
length(ids)

## End(Not run)
```

## bmrb_pdb_ids *Get PDB IDs associated with a BMRB entry*

### Description

Get PDB IDs associated with a BMRB entry

### Usage

```
bmrb_pdb_ids(bmrb_id)
```

### Arguments

| | |
|---|---|
| bmrb_id | A single BMRB entry ID. |

### Value

A data.frame with columns pdb_id, match_type, and comment.

### Examples

```
## Not run:
bmrb_pdb_ids(15000)

## End(Not run)
```

## bmrb_search *Search BMRB using the instant-search endpoint*

### Description

Searches BMRB by title, author name, organism, amino-acid sequence, InChI string, or other terms.

### Usage

```
bmrb_search(term, database = "combined")
```

### Arguments

| | |
|---|---|
| term | A search string. |
| database | One of "macromolecules", "metabolomics", or "combined" (default). |

### Value

A data.frame with columns value, label, sub_date, authors, and link.

### Examples

```
## Not run:
bmrb_search("ubiquitin human")
bmrb_search("GB3", database = "macromolecules")

## End(Not run)
```

---

## bmrb_status *Get BMRB API server status*

---

### Description

Returns current server information including entry counts and database last-update timestamps from the BMRB REST API.

### Usage

```
bmrb_status()
```

### Value

A list with server status information.

### Examples

```
## Not run:
bmrb_status()

## End(Not run)
```

---

## bmrb_test *Run a connectivity self-test*

---

### Description

Performs three checks: BMRB API status, entry fetch and parse for entry 15060, and the chemical shift search endpoint. Useful for verifying that the BMRB API is reachable from your network. Progress is reported with `message()`, so it can be silenced with `suppressMessages()`.

### Usage

```
bmrb_test()
```

### Value

A named logical vector with elements `api_status`, `entry_fetch`, and `search_query` (invisibly).

## Examples

```
## Not run:
bmrb_test()

## End(Not run)
```

---

`bmrb_upload_entry` *Upload an NMR-STAR file to BMRB for temporary access*

---

## Description

Uploads a local NMR-STAR file to the BMRB server. The entry is stored for 7 days. The returned token can be used as a BMRB ID in all other package functions.

## Usage

```
bmrb_upload_entry(file_path)
```

## Arguments

| | |
|---|---|
| `file_path` | Path to a local NMR-STAR `.str` file. |

## Value

A character string token usable as a BMRB ID (invisibly).

## Examples

```
## Not run:
token <- bmrb_upload_entry("my_protein.str")
n15hsqc(bmrb_ids = token)

## End(Not run)
```

---

`c13hsqc` *Simulate a ^1H−^13C HSQC spectrum*

---

## Description

Equivalent to ‘pybmrb.Spectra.c13hsqc()‘. See [n15hsqc()] for parameter descriptions.

## Usage

```
c13hsqc(
  bmrb_ids = NULL,
  input_file_names = NULL,
  auth_tag = FALSE,
  legend = NULL,
  draw_trace = FALSE,
  peak_list = NULL,
  output_format = "html",
  output_file = NULL,
  output_image_width = 800,
  output_image_height = 600,
  show_visualization = TRUE
)
```

## Arguments

`bmrb_ids` BMRB entry ID(s) – integer, character, or vector.

`input_file_names` Path(s) to local NMR-STAR ‘.str‘ files.

`auth_tag` Use ‘Auth_seq_ID‘ for sequence numbering.

`legend` ‘NULL‘ (default), ‘"residue"‘, or ‘"dataset"‘.

`draw_trace` Draw lines connecting same-residue peaks across datasets.

`peak_list` Path to a 2-column CSV file of unassigned peaks (‘H_ppm‘, ‘N_ppm‘) to overlay.

`output_format` ‘"html"‘ (default), ‘"png"‘, ‘"jpg"‘, ‘"pdf"‘, ‘"svg"‘, or ‘"webp"‘.

`output_file` File path to save the visualization (optional).

`output_image_width`, `output_image_height` Image dimensions in pixels.

`show_visualization` Open in browser / display inline. Default ‘TRUE‘.

## Value

Invisibly, a list with the simulated peak list.

## Examples

```
## Not run:
c13hsqc(bmrb_ids = 15060, legend = "residue")

## End(Not run)
```

## conditional_cs_hist *Plot a conditional chemical shift histogram*

### Description

Shows the CS distribution of ‘atom‘ in ‘residue‘ before and after filtering by known chemical shifts of other atoms in the same residue. Mirrors ‘pybmrb.Histogram.conditional_hist()‘.

### Usage

```
conditional_cs_hist(
  residue,
  atom,
  filtering_rules,
  ph_min = NULL,
  ph_max = NULL,
  t_min = NULL,
  t_max = NULL,
  h_tolerance = 0.1,
  c_tolerance = 2,
  n_tolerance = 2,
  standard_amino_acids = TRUE,
  histnorm = NULL,
  plot_type = "histogram",
  output_format = "html",
  output_file = NULL,
  output_image_width = 800,
  output_image_height = 600,
  show_visualization = TRUE
)
```

### Arguments

`residue` Residue name.

`atom` Atom whose distribution to plot.

`filtering_rules` Named list of atom = value pairs used to filter, e.g. ‘list(CA = 64.5, H = 7.8)‘.

`ph_min`, `ph_max`, `t_min`, `t_max` Condition filters.

`h_tolerance` Tolerance (ppm) for H filters. Default 0.1.

`c_tolerance` Tolerance (ppm) for C filters. Default 2.0.

`n_tolerance` Tolerance (ppm) for N filters. Default 2.0.

`standard_amino_acids` Restrict to standard amino acids. Default TRUE.

`histnorm` Normalisation (see [cs_hist()]).

`plot_type` "histogram" (default), "box", or "violin".

`output_format`, `output_file`, `output_image_width`, `output_image_height`, `show_visualization`
See [cs_hist()].

## Value

Invisibly, a list with ‘all_values‘, ‘filtered_values‘, ‘labels‘.

## Examples

```
## Not run:
  conditional_cs_hist("SER", "N",
    filtering_rules = list(CA = 55.5, HA = 4.3))

## End(Not run)
```

---

`create_2d_peaklist` *Build a generic 2D peak list for any two atoms*

---

## Description

Equivalent to ‘pybmrb.Spectra.create_2d_peaklist()‘.

## Usage

```
create_2d_peaklist(
  atom_x,
  atom_y,
  bmrb_ids = NULL,
  input_file_names = NULL,
  auth_tag = FALSE,
  draw_trace = FALSE,
  include_preceding = FALSE,
  include_next = FALSE
)
```

## Arguments

`atom_x` Atom name for the X axis.

`atom_y` Atom name for the Y axis.

`bmrb_ids` BMRB entry ID(s) – integer, character, or vector.

`input_file_names`
Path(s) to local NMR-STAR ‘.str‘ files.

`auth_tag` Use ‘Auth_seq_ID‘ for sequence numbering.

`draw_trace` Draw lines connecting same-residue peaks across datasets.

`include_preceding`
: Include preceding (i-1) residue Y-axis peaks.

`include_next`
: Include following (i+1) residue Y-axis peaks.

## Value

Same structure as [create_n15hsqc_peaklist()].

---

`create_c13hsqc_peaklist` *Build a 1H-13C HSQC peak list*

---

## Description

Equivalent to ‘pybmrb.Spectra.create_c13hsqc_peaklist()‘.

## Usage

```
create_c13hsqc_peaklist(
  bmrb_ids = NULL,
  input_file_names = NULL,
  auth_tag = FALSE,
  draw_trace = FALSE
)
```

## Arguments

`bmrb_ids`
: BMRB entry ID(s) – integer, character, or vector.

`input_file_names`
: Path(s) to local NMR-STAR ‘.str‘ files.

`auth_tag`
: Use ‘Auth_seq_ID‘ for sequence numbering.

`draw_trace`
: Draw lines connecting same-residue peaks across datasets.

## Value

Same structure as [create_n15hsqc_peaklist()].

## create_n15hsqc_peaklist *Build a 1H-15N HSQC peak list*

### Description

Extracts backbone amide (and optionally sidechain NH) chemical shifts from BMRB entries / NMR-STAR files and returns the 2D peak coordinates. Equivalent to ‘pybmrb.Spectra.create_n15hsqc_peaklist()‘.

### Usage

```
create_n15hsqc_peaklist(
  bmrb_ids = NULL,
  input_file_names = NULL,
  auth_tag = FALSE,
  draw_trace = FALSE,
  include_sidechain = TRUE
)
```

### Arguments

| | |
|---|---|
| `bmrb_ids` | BMRB entry ID(s) – integer, character, or vector. |
| `input_file_names` | Path(s) to local NMR-STAR ‘.str‘ files. |
| `auth_tag` | Use ‘Auth_seq_ID‘ for sequence numbering. |
| `draw_trace` | Draw lines connecting same-residue peaks across datasets. |
| `include_sidechain` | Include sidechain NH2 peaks (ASN/GLN/ARG/TRP/HIS). |

### Value

A list with elements: ‘x‘ (H ppm), ‘y‘ (N ppm), ‘dataset‘, ‘residue‘, ‘info‘, ‘seq_id‘, ‘cs_track‘ (if ‘draw_trace = TRUE‘).

## create_tocsy_peaklist *Build a 1H-1H TOCSY peak list*

### Description

Pairs all protons within each residue (full spin-system). Both diagonal and off-diagonal peaks are included. Equivalent to ‘pybmrb.Spectra.create_tocsy_peaklist()‘.

### Usage

```
create_tocsy_peaklist(
  bmrb_ids = NULL,
  input_file_names = NULL,
  auth_tag = FALSE,
  draw_trace = FALSE
)
```

### Arguments

| | |
|---|---|
| `bmrb_ids` | BMRB entry ID(s) – integer, character, or vector. |
| `input_file_names` | Path(s) to local NMR-STAR ‘.str‘ files. |
| `auth_tag` | Use ‘Auth_seq_ID‘ for sequence numbering. |
| `draw_trace` | Draw lines connecting same-residue peaks across datasets. |

### Value

Same structure as [create_n15hsqc_peaklist()].

---

## `cs_from_bmrb` *Extract chemical shifts from BMRB database entries*

---

### Description

Fetches assigned chemical shift data for one or more BMRB entries. Uses the same zlib-compressed JSON endpoint as PyBMRB / pynmrstar. Mirrors ‘pybmrb.ChemicalShift.from_bmrb()‘.

### Usage

```
cs_from_bmrb(bmrb_ids, auth_tag = FALSE)
```

### Arguments

| | |
|---|---|
| `bmrb_ids` | A single BMRB entry ID or a vector of IDs. |
| `auth_tag` | Logical. Use ‘Auth_seq_ID‘ for sequence numbering. Default FALSE. |

### Value

A nested named list: ‘list(dataset_id = list(chain = list(seq_id = list(atom = cs_value), seq_ids = integer_vector)))‘.

## Examples

```
## Not run:
  cs <- cs_from_bmrb(15060)
  cs_peek(cs)
  cs_from_bmrb(c(17076, 17077))

## End(Not run)
```

---

`cs_from_file` *Extract chemical shifts from local NMR-STAR files*

---

## Description

Mirrors ‘pybmrb.ChemicalShift.from_file()‘.

## Usage

```
cs_from_file(file_paths, auth_tag = FALSE, data_set_ids = NULL)
```

## Arguments

| | |
|---|---|
| `file_paths` | Character vector of NMR-STAR file paths. |
| `auth_tag` | Logical. Use ‘Auth_seq_ID‘. Default FALSE. |
| `data_set_ids` | Optional character vector of dataset labels. |

## Value

Same nested-list structure as [cs_from_bmrb()].

## Examples

```
## Not run:
  cs <- cs_from_file("my_protein.str")

## End(Not run)
```

## cs_hist — *Plot a chemical shift distribution histogram*

### Description

Fetches database-wide chemical shift data and renders an interactive distribution plot. Mirrors ‘pybmrb.Histogram.hist()‘.

### Usage

```
cs_hist(
  residue = NULL,
  atom = NULL,
  list_of_atoms = NULL,
  filtered = TRUE,
  sd_limit = 10,
  ambiguity = "*",
  ph_min = NULL,
  ph_max = NULL,
  t_min = NULL,
  t_max = NULL,
  standard_amino_acids = TRUE,
  histnorm = NULL,
  plot_type = "histogram",
  output_format = "html",
  output_file = NULL,
  output_image_width = 800,
  output_image_height = 600,
  show_visualization = TRUE
)
```

### Arguments

| | |
|---|---|
| residue | Residue name (IUPAC 3-letter). Use "*" for any. |
| atom | Atom name (e.g. "CA", "HB*", "*"). |
| list_of_atoms | Optional character vector of "RES-ATOM" pairs (e.g. ‘c("ALA-CB","CYS-N")‘). Overrides residue/atom when supplied. |
| filtered | Remove outliers. Default TRUE. |
| sd_limit | SD factor for outlier removal. Default 10. |
| ambiguity | Ambiguity code filter ("*" = no filter). |
| ph_min, ph_max | pH range filter. |
| t_min, t_max | Temperature range filter (Kelvin). |
| standard_amino_acids | Restrict to standard amino acids. Default TRUE. |

`histnorm` NULL (count), "percent", "probability", or "probability density".

`plot_type` "histogram" (default), "box", or "violin".

`output_format` "html" (default), "png", "jpg", "pdf", "svg", "webp".

`output_file` File path to save the plot (optional).

`output_image_width`, `output_image_height` Image size in pixels.

`show_visualization` Display the figure. Default TRUE.

## Value

Invisibly, a list with elements ‘values‘ and ‘labels‘.

## Examples

```
## Not run:
  cs_hist("ALA", "CA")
  cs_hist(list_of_atoms = c("ALA-CB","GLY-CA","SER-CB"))
  cs_hist("ALA", "CA", histnorm = "probability density",
          plot_type = "violin")

## End(Not run)
```

---

`cs_hist2d` *Plot a 2D chemical shift correlation map*

---

## Description

Mirrors ‘pybmrb.Histogram.hist2d()‘.

## Usage

```
cs_hist2d(
  residue,
  atom1,
  atom2,
  filtered = TRUE,
  sd_limit = 10,
  ambiguity1 = "*",
  ambiguity2 = "*",
  ph_min = NULL,
  ph_max = NULL,
  t_min = NULL,
  t_max = NULL,
  histnorm = NULL,
  plot_type = "heatmap",
  output_format = "html",
```

```
  output_file = NULL,
  output_image_width = 800,
  output_image_height = 600,
  show_visualization = TRUE
)
```

## Arguments

- `residue` Residue name.
- `atom1` First atom (X axis).
- `atom2` Second atom (Y axis).
- `filtered`, `sd_limit` See [cs_hist()].
- `ambiguity1`, `ambiguity2` Per-atom ambiguity filters.
- `ph_min`, `ph_max`, `t_min`, `t_max` Condition filters.
- `histnorm` Normalisation (see [cs_hist()]).
- `plot_type` "heatmap" (default) or "contour".
- `output_format`, `output_file`, `output_image_width`, `output_image_height`, `show_visualization` See [cs_hist()].

## Value

Invisibly, a list with elements ‘atom1‘ and ‘atom2‘.

## Examples

```
## Not run:
  cs_hist2d("CYS", "CB", "N")
  cs_hist2d("ALA", "CA", "CB", plot_type = "contour")

## End(Not run)
```

---

`cs_peek` *Peek at a nested CS structure*

---

## Description

Reports the first few residues of each chain with `message()`, so the output can be silenced with `suppressMessages()`.

## Usage

```
cs_peek(cs_data, n_res = 5L)
```

### Arguments

| | |
|---|---|
| `cs_data` | Output of [cs_from_bmrb()] or [cs_from_file()]. |
| `n_res` | Number of residues to show per chain. |

### Value

`cs_data`, invisibly.

### Examples

```
## Not run:  cs <- cs_from_bmrb(15060); cs_peek(cs)
```

---

## cs_stats — *Compute summary statistics for a set of chemical shifts*

---

### Description

Compute summary statistics for a set of chemical shifts

### Usage

```
cs_stats(x)
```

### Arguments

| | |
|---|---|
| `x` | Numeric vector of chemical shift values. |

### Value

A named numeric vector: n, mean, sd, median, min, max.

### Examples

```
cs_stats(c(55.1, 55.3, 56.0, 54.8))
```

## cs_summary — *Summarise a nested chemical shift list*

### Description

Builds a tidy summary of datasets, chains, and atom coverage. The summary is returned rather than printed, so it prints automatically at the console and can also be assigned and reused.

### Usage

```
cs_summary(cs_data)
```

### Arguments

| | |
|---|---|
| cs_data | Nested list from [cs_from_bmrb()] or [cs_from_file()]. |

### Value

A data.frame with one row per dataset and chain.

### Examples

```
## Not run:  cs <- cs_from_bmrb(15060); cs_summary(cs)
```

## cs_table — *Get a flat data.frame of chemical shifts for one or more BMRB entries*

### Description

Get a flat data.frame of chemical shifts for one or more BMRB entries

### Usage

```
cs_table(bmrb_ids, auth_tag = FALSE)
```

### Arguments

| | |
|---|---|
| bmrb_ids | A single BMRB entry ID or a vector of IDs. |
| auth_tag | Logical. Use ‘Auth_seq_ID‘ for sequence numbering. Default FALSE. |

### Value

A data.frame suitable for use with ggplot2 or dplyr.

### Examples

```
## Not run:  df <- cs_table(c(15060, 17076))
```

## cs_to_df — *Convert a nested CS list to a tidy data.frame*

### Description

Convert a nested CS list to a tidy data.frame

### Usage

```
cs_to_df(cs_data)
```

### Arguments

| | |
|---|---|
| cs_data | Nested list from [cs_from_bmrb()] or [cs_from_file()]. |

### Value

A data.frame with columns Dataset, Chain, Seq_ID, Atom_ID, Val.

### Examples

```
## Not run:  cs <- cs_from_bmrb(15060); df <- cs_to_df(cs)
```

## export_peak_list — *Export a peak list to CSV or Sparky format*

### Description

Converts the output of any ‘create_*_peaklist()‘ function to a file or a data.frame. Equivalent to ‘pybmrb.Spectra.export_peak_list()‘.

### Usage

```
export_peak_list(
  peak_list,
  output_file_name = NULL,
  output_format = "csv",
  include_side_chain = TRUE
)
```

### Arguments

`peak_list` List returned by any ‘create_*_peaklist()‘ function.

`output_file_name` Path to write the output file. ‘NULL‘ returns only the data.frame without writing to disk.

`output_format` ‘"csv"‘ (default) or ‘"sparky"‘.

`include_side_chain` Include sidechain peaks. Default ‘TRUE‘.

### Value

Invisibly, a data.frame with the exported peak list.

### Examples

```
## Not run:
pl <- create_n15hsqc_peaklist(bmrb_ids = 15060)
export_peak_list(pl, "peaks.csv", output_format = "csv")
export_peak_list(pl, "peaks.sparky", output_format = "sparky")

## End(Not run)
```

---

## `filter_cs_outliers` *Remove outliers from a numeric vector*

---

### Description

Remove outliers from a numeric vector

### Usage

```
filter_cs_outliers(x, sd_limit = 10)
```

### Arguments

`x` Numeric vector.

`sd_limit` Number of standard deviations beyond which to remove.

### Value

Numeric vector with outliers removed.

### Examples

```
filter_cs_outliers(c(52, 55, 56, 54, 300), sd_limit = 2)
```

generic_2d *Plot a generic 2D NMR correlation spectrum*

## Description

Generates a 2D spectrum for any two atom types. Supports sequence-walk tracing, preceding/next-residue peaks, and all the same overlay options as the other spectrum functions. Equivalent to ‘pybmrb.Spectra.generic_2d()‘.

## Usage

```
generic_2d(
  atom_x,
  atom_y,
  bmrb_ids = NULL,
  input_file_names = NULL,
  auth_tag = FALSE,
  legend = NULL,
  draw_trace = FALSE,
  peak_list = NULL,
  include_preceding = FALSE,
  include_next = FALSE,
  seq_walk = FALSE,
  full_walk = FALSE,
  output_format = "html",
  output_file = NULL,
  output_image_width = 800,
  output_image_height = 600,
  show_visualization = TRUE
)
```

## Arguments

| | |
|---|---|
| atom_x | Atom name for the X axis (IUPAC, e.g. ‘"N"‘, ‘"CA"‘). |
| atom_y | Atom name for the Y axis. |
| bmrb_ids | BMRB entry ID(s) – integer, character, or vector. |
| input_file_names | Path(s) to local NMR-STAR ‘.str‘ files. |
| auth_tag | Use ‘Auth_seq_ID‘ for sequence numbering. |
| legend | ‘NULL‘ (default), ‘"residue"‘, or ‘"dataset"‘. |
| draw_trace | Draw lines connecting same-residue peaks across datasets. |
| peak_list | Path to a 2-column CSV file of unassigned peaks (‘H_ppm‘, ‘N_ppm‘) to overlay. |

include_preceding
: Include chemical shifts from the preceding (i-1) residue on the Y axis. Default ‘FALSE‘.

include_next
: Include chemical shifts from the following (i+1) residue on the Y axis. Default ‘FALSE‘.

seq_walk
: Draw a trace connecting sequential i->i+/-1 pairs for continuous sequence segments. Default ‘FALSE‘.

full_walk
: Like ‘seq_walk‘ but ignores missing residues. Default ‘FALSE‘.

output_format
: ‘"html"‘ (default), ‘"png"‘, ‘"jpg"‘, ‘"pdf"‘, ‘"svg"‘, or ‘"webp"‘.

output_file
: File path to save the visualization (optional).

output_image_width, output_image_height
: Image dimensions in pixels.

show_visualization
: Open in browser / display inline. Default ‘TRUE‘.

## Value

Invisibly, the peak list.

## Examples

```
## Not run:
generic_2d("N", "CA", bmrb_ids = 15000, legend = "residue")
generic_2d("N", "CA", bmrb_ids = 15000, include_next = TRUE,
           seq_walk = TRUE)

## End(Not run)
```

---

# get_2d_cs_data *Fetch paired 2D chemical shift data for two atoms in the same residue*

---

## Description

Mirrors ‘pybmrb.ChemicalShiftStatistics.get_2d_chemical_shifts()‘.

## Usage

```
get_2d_cs_data(
  residue,
  atom1,
  atom2,
  filtered = TRUE,
  sd_limit = 10,
  ambiguity1 = "*",
  ambiguity2 = "*",
  ph_min = NULL,
```

```
  ph_max = NULL,
  t_min = NULL,
  t_max = NULL
)
```

### Arguments

`residue` Residue name (IUPAC 3-letter).

`atom1` First atom (X axis).

`atom2` Second atom (Y axis).

`filtered`, `sd_limit` See [get_cs_data()].

`ambiguity1`, `ambiguity2` Per-atom ambiguity filters.

`ph_min`, `ph_max`, `t_min`, `t_max` Condition filters.

### Value

Named list with ‘atom1‘ and ‘atom2‘ numeric vectors.

### Examples

```
## Not run:  paired <- get_2d_cs_data("ALA", "CA", "CB")
```

---

## get_cs_data *Fetch chemical shift data from the BMRB database*

---

### Description

Downloads chemical shifts for the given residue and atom combination from the BMRB database. Mirrors ‘pybmrb.ChemicalShiftStatistics.get_data()‘.

### Usage

```
get_cs_data(
  residue = "*",
  atom = "*",
  filtered = TRUE,
  sd_limit = 10,
  ambiguity = "*",
  ph_min = NULL,
  ph_max = NULL,
  t_min = NULL,
  t_max = NULL,
  standard_amino_acids = TRUE,
  database = "macromolecules"
)
```

### Arguments

| | |
|---|---|
| residue | Residue in IUPAC 3-letter format (e.g. "ALA"). "*" = any. |
| atom | Atom name (e.g. "CA", "HB*"). "*" = any. |
| filtered | Remove outliers beyond sd_limit * SD. Default TRUE. |
| sd_limit | SD factor for outlier removal. Default 10. |
| ambiguity | Ambiguity code filter ("*" = no filter; 1 = unambiguous). |
| ph_min, ph_max | pH range filter (optional). |
| t_min, t_max | Temperature range in Kelvin (optional). |
| standard_amino_acids | Restrict to standard amino acids. Default TRUE. |
| database | "macromolecules" (default) or "metabolomics". |

### Value

A data.frame with columns Val, Comp_ID, Atom_ID, Ambiguity_code, Entry_ID, Comp_index_ID, pH, Temperature_K and others.

### Examples

```
## Not run:
  df <- get_cs_data("ALA", "CA")
  df <- get_cs_data("*", "HB*")

## End(Not run)
```

---

## get_cs_from_bmrb_db *Fetch raw BMRB chemical shift data (no outlier filtering)*

---

### Description

Mirrors ‘pybmrb.ChemicalShiftStatistics.get_data_from_bmrb()‘.

### Usage

```
get_cs_from_bmrb_db(
  residue = "*",
  atom = "*",
  list_of_atoms = NULL,
  ambiguity = "*",
  ph_min = NULL,
  ph_max = NULL,
  t_min = NULL,
  t_max = NULL,
  standard_amino_acids = TRUE,
  database = "macromolecules"
)
```

### Arguments

| | |
|---|---|
| residue | Residue in IUPAC 3-letter format (e.g. "ALA"). "*" = any. |
| atom | Atom name (e.g. "CA", "HB*"). "*" = any. |
| list_of_atoms | Optional "RES-ATOM" vector, e.g. ‘c("ALA-CB","GLY-CA")‘. |
| ambiguity | Ambiguity code filter ("*" = no filter; 1 = unambiguous). |
| ph_min, ph_max | pH range filter (optional). |
| t_min, t_max | Temperature range in Kelvin (optional). |
| standard_amino_acids | Restrict to standard amino acids. Default TRUE. |
| database | "macromolecules" (default) or "metabolomics". |

### Value

A data.frame of raw chemical shift records.

### Examples

```
## Not run:
  df  <- get_cs_from_bmrb_db("ALA", "CB")
  df2 <- get_cs_from_bmrb_db(list_of_atoms = c("ALA-CB","GLY-CA"))

## End(Not run)
```

---

## n15hsqc — *Simulate a ^1H−^15N HSQC spectrum*

---

### Description

Plots an interactive 1H-15N HSQC spectrum for one or more BMRB entries and/or NMR-STAR files. Overlay multiple datasets as scatter plots; optionally draw trace lines connecting peaks from the same residue position across datasets. Equivalent to ‘pybmrb.Spectra.n15hsqc()‘.

### Usage

```
n15hsqc(
  bmrb_ids = NULL,
  input_file_names = NULL,
  auth_tag = FALSE,
  legend = NULL,
  draw_trace = FALSE,
  include_sidechain = TRUE,
  peak_list = NULL,
  output_format = "html",
  output_file = NULL,
  output_image_width = 800,
  output_image_height = 600,
  show_visualization = TRUE
)
```

### Arguments

- `bmrb_ids` BMRB entry ID(s) – integer, character, or vector.
- `input_file_names` Path(s) to local NMR-STAR ‘.str‘ files.
- `auth_tag` Use ‘Auth_seq_ID‘ for sequence numbering.
- `legend` ‘NULL‘ (default), ‘"residue"‘, or ‘"dataset"‘.
- `draw_trace` Draw lines connecting same-residue peaks across datasets.
- `include_sidechain` Include sidechain NH2 peaks (ASN/GLN/ARG/TRP/HIS).
- `peak_list` Path to a 2-column CSV file of unassigned peaks (‘H_ppm‘, ‘N_ppm‘) to overlay.
- `output_format` ‘"html"‘ (default), ‘"png"‘, ‘"jpg"‘, ‘"pdf"‘, ‘"svg"‘, or ‘"webp"‘.
- `output_file` File path to save the visualization (optional).
- `output_image_width`, `output_image_height` Image dimensions in pixels.
- `show_visualization` Open in browser / display inline. Default ‘TRUE‘.

### Value

Invisibly, a list with elements ‘x‘ (H ppm), ‘y‘ (N ppm), ‘dataset‘, ‘residue‘, ‘info‘, and (when ‘draw_trace = TRUE‘) ‘cs_track‘.

### Examples

```
## Not run:
# Single entry
n15hsqc(bmrb_ids = 15060, legend = "residue")

# Compare two entries with trace lines
n15hsqc(bmrb_ids = c(17076, 17077), legend = "dataset", draw_trace = TRUE)

# Overlay user's own NMR-STAR file with BMRB entry
n15hsqc(bmrb_ids = 15060, input_file_names = "MyData.str",
        legend = "dataset")

## End(Not run)
```

---

## RBMRB_version *Get the installed RBMRB version*

---

### Description

Returns the version of the installed RBMRB package as a string.

### Usage

```
RBMRB_version()
```

### Value

A character string with the package version.

### Examples

```
RBMRB_version()
```

---

## standard_amino_acids *List the 20 standard amino acid codes*

---

### Description

List the 20 standard amino acid codes

### Usage

```
standard_amino_acids()
```

### Value

Character vector of 3-letter amino acid codes.

### Examples

```
standard_amino_acids()
```

---

## standard_nucleic_acids *List standard nucleic acid residue codes*

---

### Description

List standard nucleic acid residue codes

### Usage

```
standard_nucleic_acids()
```

### Value

Character vector of RNA and DNA residue codes.

### Examples

```
standard_nucleic_acids()
```

`tocsy` *Simulate a ^1H−^1H TOCSY spectrum*

## Description

Equivalent to ‘pybmrb.Spectra.tocsy()‘. See [n15hsqc()] for shared parameter descriptions.

## Usage

```
tocsy(
  bmrb_ids = NULL,
  input_file_names = NULL,
  auth_tag = FALSE,
  legend = NULL,
  draw_trace = FALSE,
  peak_list = NULL,
  output_format = "html",
  output_file = NULL,
  output_image_width = 800,
  output_image_height = 600,
  show_visualization = TRUE
)
```

## Arguments

| | |
|---|---|
| `bmrb_ids` | BMRB entry ID(s) – integer, character, or vector. |
| `input_file_names` | Path(s) to local NMR-STAR ‘.str‘ files. |
| `auth_tag` | Use ‘Auth_seq_ID‘ for sequence numbering. |
| `legend` | ‘NULL‘ (default), ‘"residue"‘, or ‘"dataset"‘. |
| `draw_trace` | Draw lines connecting same-residue peaks across datasets. |
| `peak_list` | Path to a 2-column CSV file of unassigned peaks (‘H_ppm‘, ‘N_ppm‘) to overlay. |
| `output_format` | ‘"html"‘ (default), ‘"png"‘, ‘"jpg"‘, ‘"pdf"‘, ‘"svg"‘, or ‘"webp"‘. |
| `output_file` | File path to save the visualization (optional). |
| `output_image_width`, `output_image_height` | Image dimensions in pixels. |
| `show_visualization` | Open in browser / display inline. Default ‘TRUE‘. |

## Value

Invisibly, a list with the simulated peak list.

## Examples

```
## Not run:
tocsy(bmrb_ids = 15060, legend = "residue")
tocsy(bmrb_ids = c(17074, 17076, 17077), legend = "dataset",
      draw_trace = TRUE)

## End(Not run)
```

# Index


bmrb_citation, 3
bmrb_debug_entry, 3
bmrb_fetch_entry, 4
bmrb_from_pdb, 4
bmrb_list_entries, 5
bmrb_pdb_ids, 6
bmrb_search, 6
bmrb_status, 7
bmrb_test, 7
bmrb_upload_entry, 8

c13hsqc, 8
conditional_cs_hist, 10
create_2d_peaklist, 11
create_c13hsqc_peaklist, 12
create_n15hsqc_peaklist, 13
create_tocsy_peaklist, 13
cs_from_bmrb, *3*, 14
cs_from_file, 15
cs_hist, 16
cs_hist2d, 17
cs_peek, 18
cs_stats, 19
cs_summary, 20
cs_table, 20
cs_to_df, 21

export_peak_list, 21

filter_cs_outliers, 22

generic_2d, 23
get_2d_cs_data, 24
get_cs_data, 25
get_cs_from_bmrb_db, 26

n15hsqc, 27

RBMRB_version, 28

standard_amino_acids, 29

standard_nucleic_acids, 29

tocsy, 30