Package 'stratigraphr'

September 12, 2026

Type Package

Title Archaeological Stratigraphy and Chronological Sequences

Version 0.5.0

Description A tidy framework for working with archaeological stratigraphy and chronology. Includes tools for reading, analysing, and visualising stratigraphic sequences (Harris matrices) as directed graphs following the definition of Dye and Buck (2015) <doi:10.1016/j.jas.2015.08.008> and an R interface to the Chronological Query Language (CQL) used in 'OxCal' by Bronk Ramsey (2009) <doi:10.1017/S0033822200033865>.

License MIT + file LICENSE

URL https://stratigraphr.joeroe.io, https://github.com/joeroe/stratigraphr

BugReports https://github.com/joeroe/stratigraphr/issues

Encoding UTF-8

LazyData true

Imports cli, Rdpack, checkmate, glue, igraph (>= 0.3.0), pillar, purrr, relations, rlang, sets, stringr, tidygraph, utils, vctrs, vroom,

Suggests covr, dplyr, ggraph, knitr, oxcAAR, readr, readxl, rmarkdown, spelling, testthat, tibble

VignetteBuilder knitr

RdMacros Rdpack

Depends R (>= 4.1)

Config/roxygen2/version 8.0.0

Language en-GB

NeedsCompilation no

Author Joe Roe [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-1011-1244>)

Maintainer Joe Roe <joe@joeroe.io>

Repository CRAN

Date/Publication 2026-09-12 13:50:02 UTC

Contents

`cql` *Chronological Query Language (CQL)*

Description

Provides an R interface for the Chronological Query Language (CQL), primarily used to input commands and describe models to OxCal.

Usage

```
cql(...)

as_cql(x)

## S3 method for class 'list'
as_cql(x)

## Default S3 method:
as_cql(x)
```

Arguments

`...`	`cql` objects to be assembled into a script.
`x`	Object to be coerced to a `cql` object.

Details

CQL scripts are represented by the S3 class `cql`. `cql()` takes the output of individual `cql_` functions and assembles them into a single script. See `vignette("cql")` for a tutorial.

List arguments to `cql()` are collapsed to produce a single script. If you want to coerce an object to `cql` and preserve its structure, use `as_cql()` instead.

Value

A CQL script.

References

https://c14.arch.ox.ac.uk/oxcalhelp/hlp_commands.html

See Also

Other CQL functions: `cql_boundary()`, `cql_n()`, `cql_options()`, `cql_other`, `cql_phase()`, `cql_r_date()`, `cql_sequence()`, `write_oxcal()`

Examples

```
# Assemble individual CQL commands into a script
cql(
  cql_r_date("ABC-001", 10100, 50),
  cql_r_date("ABC-002", 10200, 50)
)

# Coerce a character string to a cql object
as_cql("R_Date(\"ABC-001\", 10100, 50);")
```

---

cql_boundary *Describe a boundary constraint in CQL*

Description

The CQL command `Boundary` describes constraints within an ordered sequence (see `cql_sequence()`). Groups of events between two boundaries are assumed to be sampled from the same prior distribution. `Boundary` alone models a uniform prior. Other types of boundary, i.e. `Sigma_Boundary`, `Tau_Boundary`, and `Zero_Boundary` are not yet implemented in stratigraphr.

Boundaries can contain a `Transition` command, in which case they describe a non-instantaneous transition or 'trapezium' prior.

Usage

```
cql_boundary(name, ..., prior = NULL)

cql_sigma_boundary()

cql_tau_boundary()

cql_zero_boundary()

cql_transition(name, prior = NULL)
```

Arguments

name	Character. Label for a boundary or transition.
...	cql objects contained within a boundary.
prior	cql object. Expression describing the prior likelihood of a boundary or transition.

Value

A cql object.

References

https://c14.arch.ox.ac.uk/oxcalhelp/hlp_analysis_oper.html#group

https://c14.arch.ox.ac.uk/oxcalhelp/hlp_commands.html

See Also

Other CQL functions: cql(), cql_n(), cql_options(), cql_other, cql_phase(), cql_r_date(), cql_sequence(), write_oxcal()

Examples

```
# Uniform model with unknown boundaries
cql(
  cql_boundary("P1 start"),
  cql_phase("P1", list(
    cql_r_date("A", 5050, 30),
    cql_r_date("B", 5000, 30),
    cql_r_date("C", 4950, 30)
  )),
  cql_boundary("P1 end")
)

# Uniform model with boundaries with a prior likelihood
cql(
  cql_boundary("P1 start",
               prior = cql_date("P1S-Prior", cql_u("", 5200, 5100))),
```

```
  cql_phase("P1", list(
    cql_r_date("A", 5050, 30),
    cql_r_date("B", 5000, 30),
    cql_r_date("C", 4950, 30)
  )),
  cql_boundary("P1 end",
               prior = cql_date("P1E-Prior", cql_u("", 4800, 4900)))
)

# 'Trapezium' model
cql(
  cql_boundary("P1 start",
    cql_transition("")
  ),
  cql_phase("P1", list(
    cql_r_date("A", 5050, 30),
    cql_r_date("B", 5000, 30),
    cql_r_date("C", 4950, 30)
  )),
  cql_boundary("P1 end",
    cql_transition("")
  )
)
```

`cql_n` *Describe distributions in CQL*

Description

The CQL commands `N`, `LnN`, `T`, `U` and `Top_Hat` describe various types of probability distribution functions. `cql_n()` defines a normal distribution by its mean and standard deviation; `cql_lnn()` a log–normal distribution by its mean and standard deviation; `cql_t()` a Student's t distribution by degrees of freedom; `cql_u()` a uniform distribution by its start and end; `cql_top_hat()` a uniform distribution by its central point and width.

Usage

```
cql_n(name, mu, sigma, resolution = NULL)

cql_lnn(name, mu, sigma, resolution = NULL)

cql_t(name, freedom, scale = 1, resolution = NULL)

cql_top_hat(name, mid, half_width)

cql_u(name, from, to, resolution = NULL)
```

Arguments

name	Character. Label for the distribution.
mu	Integer. Mean of a normal or log–normal distribution.
sigma	Integer. Standard deviation of a normal or log–normal distribution.
resolution	Integer. Resolution of the PDF. Leave NULL (the default) to use the OxCal default.
freedom	Integer. Degrees of freedom of a Student's t distribution.
scale	Numeric. Optional scaling parameter for a Student's t distribution.
mid	Integer. Centre point of a uniform distribution.
half_width	Integer. Half-width of a uniform distribution.
from	Integer. Start point of a uniform distribution.
to	Integer. End point of a uniform distribution.

Value

A cql object.

References

https://c14.arch.ox.ac.uk/oxcalhelp/hlp_analysis_inform.html#param

https://c14.arch.ox.ac.uk/oxcalhelp/hlp_commands.html

See Also

Other CQL functions: cql(), cql_boundary(), cql_options(), cql_other, cql_phase(), cql_r_date(), cql_sequence(), write_oxcal()

Examples

```
# Describe a date as a uniform distribution
cql_date("U-Date", cql_u("U", -5000, -4000))
```

---

cql_options *Set OxCal options in CQL*

Description

This function generates a CQL command that sets OxCal's global options.

Usage

```
cql_options(
  bcad = TRUE,
  convergence_data = FALSE,
  curve = "intcal20.14c",
  cubic = TRUE,
  ensembles = 30,
  floruit = FALSE,
  intercept = FALSE,
  k_iterations = 30,
  plus_minus = FALSE,
  raw_data = FALSE,
  resolution = 5,
  round = FALSE,
  round_by = 0,
  sd1 = TRUE,
  sd2 = TRUE,
  sd3 = FALSE,
  uniform_span_prior = TRUE,
  use_f14c = TRUE,
  year = 1950.5,
  ...
)
```

Arguments

bcad	Logical. Whether BC/AD are used in the log file output.
convergence_data	Logical. Whether sample convergence data is included in the output data file.
curve	Character. The default calibration curve.
cubic	Logical. Whether cubic (as opposed to linear) interpolation is used for calibration curves.
ensembles	Integer. The number of age-depth ensembles stored during the analysis.
floruit	Logical. Whether quantile ranges are calculated instead of the default highest posterior density (hpd).
intercept	Logical. Whether the intercept method is used for radiocarbon calibration ranges.
k_iterations	Integer. The default number of MCMC passes.
plus_minus	Logical. Whether + and - are used in place of BC and AD in log files.
raw_data	Logical. Whether raw calibration curve data is included in the output data file.
resolution	Integer. The default bin size for probability distributions of Date and Interval type.
round	Logical. Whether ranges are rounded off.
round_by	Integer. Resolution of rounding (0 for automatic).
sd1	Logical. Whether 68.2% (1 σ) ranges are given in the log and tab delimited files.

sd2	Logical. Whether 95.4% (2 σ) ranges are given in the log and tab delimited files.
sd3	Logical. Whether 99.7% (3 σ) ranges are given in the log and tab delimited files.
uniform_span_prior	Logical. Whether the two extra prior factors suggested by Nicholls and Jones 2001 are used.
use_f14c	Logical. Whether all calibrations take place in F14C space (rather than BP space).
year	Numeric. The datum point for ages - the default is mid AD 1950.
...	Additional named arguments converted to OxCal options. See details.

Details

Parameter descriptions and defaults are taken from the OxCal v4.4 documentation (see references).

Options not explicitly included here can be specified as additional named arguments to this function. The name must exactly match the name of the option expected by OxCal; unknown or misspelled options are silently ignored.

Value

A cql object.

References

https://c14.arch.ox.ac.uk/oxcalhelp/hlp_analysis_detail.html#opt

See Also

Other CQL functions: cql(), cql_boundary(), cql_n(), cql_other, cql_phase(), cql_r_date(), cql_sequence(), write_oxcal()

Examples

```
# Use default options
cql_options()

# Set specific options
cql_options(curve = "intcal13.14c", k_iterations = 50)
```

---

cql_other *Other CQL functions (unimplemented)*

Description

These CQL commands are not yet implemented in stratigraphr and only included here for reference. Unimplemented commands can be included manually using cql() (see examples), but if you need one for an analysis please file a bug report!

Usage

```
cql_age(...)

cql_axis(...)

cql_c_combine(...)

cql_c_simulate(...)

cql_correl_matrix(...)

cql_correlation(...)

cql_covar_matrix(...)

cql_curve(...)

cql_delta_r(...)

cql_difference(...)

cql_end(...)

cql_exp(...)

cql_gap(...)

cql_interval(...)

cql_kde_model(...)

cql_kde_plot(...)

cql_label(...)

cql_line(...)

cql_mcmc_sample(...)

cql_mix_curves(...)

cql_number(...)

cql_offset(...)

cql_outlier(...)

cql_outlier_model(...)
```

```
cql_p(...)

cql_pois(...)

cql_prior(...)

cql_probability(...)

cql_r_combine(...)

cql_r_simulate(...)

cql_reservoir(...)

cql_sample(...)

cql_sapwood(...)

cql_sapwood_model(...)

cql_shift(...)

cql_start(...)

cql_after()

cql_before()

cql_combine()

cql_first()

cql_last()

cql_order()

cql_span()

cql_sum()
```

Arguments

...	Ignored.

Value

A warning not to use these functions!

See Also

Other CQL functions: cql(), cql_boundary(), cql_n(), cql_options(), cql_phase(), cql_r_date(), cql_sequence(), write_oxcal()

Examples

```
# Use cql() to manually write commands that are not yet implemented, e.g.:
cql("After(\"A\");")
```

---

cql_phase *Describe an unordered group in CQL*

Description

The CQL command Phase is used to describe an unordered group of events in a model.

Usage

```
cql_phase(name, cql)
```

Arguments

name	Character. Label for the phase.
cql	Vector of cql objects contained in the phase.

Value

A cql object.

References

https://c14.arch.ox.ac.uk/oxcalhelp/hlp_commands.html

See Also

Other CQL functions: cql(), cql_boundary(), cql_n(), cql_options(), cql_other, cql_r_date(), cql_sequence(), write_oxcal()

Examples

```
cql(
  cql_phase("Phase A", c(
    cql_r_date("ABC-001", 10100, 50),
    cql_r_date("ABC-002", 10200, 50),
    cql_r_date("ABC-003", 10300, 50)
  ))
)
```

`cql_r_date` *Describe dates in CQL*

Description

The CQL commands `R_Date`, `F14C_Date`, `C_Date` and `Date` are used to describe individual dated events in a model. `R_Date` and `F14C_Date` represent radiocarbon dates expressed in radiocarbon years (the conventional radiocarbon age, CRA) or *fraction modern* (F14C) respectively. `C_Date` represents a calendar date. `Date` directly specifies a date in OxCal's internal format.

Usage

```
cql_r_date(name, date, error)

cql_c_date(name, date, error)

cql_r_f14c(name, date, error)

cql_date(name, date)
```

Arguments

`name`	Character. Date label(s), usually a lab code.
`date`	Numeric. Date or dates expressed in radiocarbon years (`cql_r_date()`), F14C (`cql_f14c_date()`), calendar years (`cql_c_date()`), or OxCal's internal format (`cql_date()`) See details.
`error`	Integer. Uncertainty associated with the date(s).

Details

The era expected for calendar dates (BP or BC/AD) depends on a global option in OxCal, which defaults to BC/AD. BC dates are specified as negative values.

F14C measurements are recommended for modern, "post-bomb" radiocarbon dates (Reimer et al. 2004).

OxCal's internal date format, used with `Date()`, is a decimal Gregorian year, for details see: https://c14.arch.ox.ac.uk/oxcalhelp/hlp_analysis_calend.html

Value

A `cql` object, or a list of `cql` objects if the arguments are vectors.

References

https://c14.arch.ox.ac.uk/oxcalhelp/hlp_analysis_inform.html#date

https://c14.arch.ox.ac.uk/oxcalhelp/hlp_commands.html

Reimer PJ, Brown TA, Reimer RW (2004). "Discussion: Reporting and Calibration of Post-Bomb 14C Data." *Radiocarbon*, **46**(3), 1299–1304. ISSN 0033-8222, 1945-5755. doi:10.1017/S0033822200033154.

See Also

Other CQL functions: `cql()`, `cql_boundary()`, `cql_n()`, `cql_options()`, `cql_other`, `cql_phase()`, `cql_sequence()`, `write_oxcal()`

Examples

```
# Radiocarbon date
cql_r_date("ABC-001", 10100, 50)

# Calendar date
cql_c_date("ABC-002", -5000, 50)

# F14C date
cql_r_f14c("ABC-003", 5000, 10)

# Date in OxCal internal format
cql_date("ABC-004", 2020.5)

# Vectorized: multiple dates at once
cql_r_date(c("ABC-001", "ABC-002"), c(10100, 10200), c(50, 50))
```

---

cql_sequence *Describe an ordered group in CQL*

Description

The CQL command `Sequence` is used to describe an *ordered* group of events in a model. Specific types of sequence, i.e. `D_Sequence`, `P_Sequence`, `U_Sequence`, and `V_Sequence`, are not yet implemented in stratigraphr.

Usage

```
cql_sequence(name, cql, boundaries = FALSE)

cql_d_sequence()

cql_p_sequence()

cql_u_sequence()

cql_v_sequence()
```

Arguments

name	Character. Label for the sequence.
cql	Vector of cql objects contained in the sequence.
boundaries	Logical. If TRUE, adds Boundary constraints between each item in the sequence.

Details

OxCal expects events within sequences to be in chronological order (oldest to youngest) not stratigraphic order (youngest to oldest).

Value

A cql object.

References

https://c14.arch.ox.ac.uk/oxcalhelp/hlp_commands.html

See Also

Other CQL functions: cql(), cql_boundary(), cql_n(), cql_options(), cql_other, cql_phase(), cql_r_date(), write_oxcal()

Examples

```
cql(
  cql_sequence("Sequence A", c(
    cql_r_date("ABC-001", 10100, 50),
    cql_r_date("ABC-002", 10200, 50),
    cql_r_date("ABC-003", 10300, 50)
  ))
)
```

---

harris12 *Stratigraphy from Harris Figure 12*

Description

A model stratigraphic sequence used by Edward Harris (1979) to illustrate the construction of a Harris matrix.

Usage

```
harris12
```

Format

A data frame with 10 rows and 4 variables:

context Label of the stratigraphic unit.

above Unit stratigraphically above.

below Unit stratigraphically below.

equal Unit stratigraphically equal.

Source

Harris (1979), p. 39.

References

Harris EC (1979). *Principles of archaeological stratigraphy*. Academic Press, London. ISBN 9780123266514.

`read_lst` *Read an LST file*

Description

Reads stratigraphic data in LST format (used by BASP Harris, Stratify, and ArchEd) into a data frame.

Usage

```
read_lst(file, split = TRUE, sep = ",", locale = vroom::default_locale())
```

Arguments

`file`	Path to an LST file.
`split`	Controls how attributes that contain multiple values are handled. If `TRUE`, all attributes except `name` will be split if their values contain the character specified by `sep`. If `FALSE`, no splitting will be done. Alternatively, specify a vector of column names (corresponding to LST attribute names, see details) that should be split.
`sep`	Delimiter used to separate multiple values in attributes. multiple values. Ignored if `split = FALSE`. Default: `","`.
`locale`	`vroom::locale()` object specifying the character encoding and other region-specific settings used in the file. Defaults to `vroom::default_locale()`.

Details

This function supports both the original LST format used by BASP Harris and the "extended" format used by ArchEd and Stratify.

Value

A data frame. Each row represents one stratum. Columns contain the name of the stratum and the attributes associated with it (typically "above", "contemporary_with", "equal_to" and "below").

Attribute names are standardised when read into column names. Specifically, they are transformed to lower case and spaces are replaced with an underscore (_).

Splitting multiple values into vectors is useful if the output is to be used to construct a stratigraphic graph with `stratigraph()`, but will convert affected columns from atomic vectors to lists, which can make them awkward to work with in other contexts. Set `split = FALSE` to avoid this. You can manually split them later with `stringr::str_split()`.

References

http://archaeologic.al/wiki/Harris_Matrix#LST

Examples

```
# Example data from https://github.com/lparchaeology/harris2graph
# Simple LST (BASP Harris)
basp_lst <- system.file("extdata", "bonn.lst", package = "stratigraphr")
read_lst(basp_lst)

# Extended LST (Stratify, ArchEd)
stratify_lst <- system.file(
  "extdata", "stratify.lst", package = "stratigraphr"
)
read_lst(stratify_lst)
```

---

shub1 *Schematic stratigraphy of Shubayqa 1*

Description

A simplified version of the stratigraphy of Shubayqa 1, an Epipalaeolithic site in eastern Jordan, after Richter et al. (2017).

Usage

```
shub1
```

Format

A data frame with 30 rows, representing contexts, and 7 variables:

context integer; a unique identifier of the context.

type character; type of context, i.e. deposit, fill, cut, or structural.

above integer vector; context(s) stratigraphically above this one.

below integer vector; context(s) stratigraphically below this one.

equal integer vector; context(s) stratigraphically equal to this one.

phase character; Phase assigned to the context, for contexts that aren't structural.

structure character; for structural contexts, the name of the structure they belong to.

Details

The stratigraphy is a simplified version derived from the schematic section in Richter et al. (2017), figure 2. Context numbers were arbitrarily assigned and the stratigraphic relations are based on those evident in the diagram. Phase and structure names are also based on the diagram.

Source

Richter et al. (2017)

References

Richter T, Arranz-Otaegui A, Yeomans L, Boaretto E (2017). "High Resolution AMS Dates from Shubayqa 1, northeast Jordan Reveal Complex Origins of Late Epipalaeolithic Natufian in the Levant." *Scientific reports*, **7**(1), 17025. ISSN 2045-2322. doi:10.1038/s41598017170965.

`stratigraph` *Construct a stratigraphic graph*

Description

Takes a data frame describing stratigraphic units and relations and turns it into a graph representation.

Usage

```
stratigraph(data, label, relation, direction = c("above", "below"))
```

Arguments

`data`	Data frame of stratigraphic units, containing at least a unique label column and a column describing stratigraphic relations. The relation column can be either a list column (where each element is a vector of related units) or a regular column (where each row represents a single relation). If the relation column is a list, it will be automatically unnested to long format.
`label`	Name of the column containing labels of the stratigraphic units.
`relation`	Name of the column describing the stratigraphic relations between units.
`direction`	Direction described by `relation`, i.e. are the units in that column "above" or "below" the ones in units. Default: "above".

Value

A `stratigraph` object.

Examples

```
# Long format (one relation per row)
stratigraph(data.frame(
  label = c("A", "B", "C", "C"),
  below = c("B", "C", "A", "B")
), "label", "below")
```

`strat_connect` *Connect stratigraphic units*

Description

Constructs a table of directed edges between stratigraphic units based on a vector of relations (e.g. "above" or "below").

Usage

```
strat_connect(units, relations, direction = c("above", "below"))
```

Arguments

`units`	Vector of unit labels.
`relations`	Vector of relations.
`direction`	Are the units in `relations` "above" or "below" the ones in `units`?

Value

A data frame of directed edges represented by `to` and `from` columns, which can be used as the `nodes` argument to `tidygraph::tbl_graph()`.

Examples

```
strat_connect(
  units = c("A", "B", "C"),
  relations = list("B", "C", NA),
  direction = "above"
)
```

strat_is_mirror *Are two relation vectors mirrored?*

Description

Checks whether one vector of relations is the inverse of another. Typically used to confirm that "above" and "below" columns match and will result in the same stratigraphic graph.

Usage

```
strat_is_mirror(units, relation1, relation2)
```

Arguments

units	Vector of unit labels.
relation1	First vector of relations.
relation2	Second vector of relations.

Value

TRUE or FALSE

Examples

```
data("harris12")
strat_is_mirror(harris12$context, harris12$above, harris12$below)
```

strg_is_valid *Validation of stratigraphic graphs*

Description

Validation of stratigraphic graphs

Usage

```
strg_is_valid(stratigraph)

strg_validate(stratigraph, warn = FALSE)
```

Arguments

stratigraph	A stratigraph object (see stratigraph()).
warn	If TRUE, signal warnings instead of errors. Default: FALSE.

Details

Checks whether a stratigraph object is a valid stratigraphic graph. Currently looks for:

- Whether the graph contains cycles (violating the law of stratigraphic superposition)
- Whether the graph contains redundant relations (i.e. relations that are implied by transitivity)

`strg_is_valid()` returns `TRUE` or `FALSE` silently. `strg_validate()` signals an error (or warning if `warn = TRUE`) with details of any validity issues found.

Value

`strg_is_valid()` returns `TRUE` or `FALSE`. `strg_validate()` returns the object (or signals an error/warning if invalid).

Examples

```
data("harris12")
strg <- stratigraph(harris12, "context", "above")

strg_is_valid(strg)
strg_validate(strg)
```

---

strg_prune *Remove redundant relations from a stratigraph*

Description

strg_prune() removes redundant relations from a stratigraphic graph by computing its transitive reduction. The result is a 'pruned' graph that follows Harris' "Law of Stratigraphical Succession": that only the uppermost and undermost relations are significant when placing a unit in a stratigraphic sequence.

Usage

```
strg_prune(strg)
```

Arguments

`strg`	A `stratigraph()` object to prune.

Value

A stratigraph with redundant relations removed.

Examples

```
bushy_stratigraphy <- stratigraph(
  tibble::tibble(
    id = letters[1:5],
    above = list(NA, "a", "a", c("a", "b", "c"), c("a", "c", "d"))
  ),
  "id", "above"
)

strg_prune(bushy_stratigraphy)
```

`write_oxcal` *Write CQL to a file*

Description

Writes a CQL script (from `cql()`) to an .oxcal file, for input to OxCal.

Usage

```
write_oxcal(cql, file)
```

Arguments

`cql`	A `cql` object. See `cql()`.
`file`	Path to a file.

Value

Returns `cql` invisibly.

See Also

Other CQL functions: `cql()`, `cql_boundary()`, `cql_n()`, `cql_options()`, `cql_other`, `cql_phase()`, `cql_r_date()`, `cql_sequence()`

Examples

```
script <- cql(
  cql_r_date("ABC-001", 10100, 50),
  cql_r_date("ABC-002", 10200, 50)
)
write_oxcal(script, "script.oxcal")
```

Index