Package: ArchaeoPhases (via r-universe)

October 4, 2026

Type Package

Title Post-Processing of Markov Chain Monte Carlo Simulations for Chronological Modelling

Version 2.1.1

Maintainer Anne Philippe <anne.philippe@univ-nantes.fr>

Description Statistical analysis of archaeological dates and groups of dates. This package allows to post-process Markov Chain Monte Carlo (MCMC) simulations from 'ChronoModel' <https://chronomodel.com/>, 'Oxcal' <https://c14.arch.ox.ac.uk/oxcal.html> or 'BCal' <https://bcal.shef.ac.uk/>. It provides functions for the study of rhythms of the long term from the posterior distribution of a series of dates (tempo and activity plot). It also allows the estimation and visualization of time ranges from the posterior distribution of groups of dates (e.g. duration, transition and hiatus between successive phases) as described in Philippe and Vibet (2020) <doi:10.18637/jss.v093.c01>.

License GPL (>= 3)

URL https://ArchaeoStat.github.io/ArchaeoPhases/, https://github.com/ArchaeoStat/ArchaeoPhases

BugReports https://github.com/ArchaeoStat/ArchaeoPhases/issues

Depends R (>= 3.5), aion (>= 1.7.0)

Imports arkhe (>= 1.11.0), graphics, grDevices, methods, stats, tools, utils

Suggests ArchaeoData, coda, fontquiver, knitr, rmarkdown, rsvg, svglite, tinysnapshot, tinytest

VignetteBuilder knitr

Additional_repositories https://archaeostat.r-universe.dev

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

X-schema.org-applicationCategory Archaeological Science

X-schema.org-keywords markov-chain, bayesian-statistics, geochronology, radiocarbon-dates, archaeology, archaeological-science, r-package

Collate 'AllClasses.R' 'AllGenerics.R' 'ArchaeoPhases-defunct.R' 'ArchaeoPhases-deprecated.R' 'ArchaeoPhases-internal.R' 'ArchaeoPhases-package.R' 'activity.R' 'allen-mcmc.R' 'allen-relations.R' 'bind.R' 'boundaries.R' 'coerce.R' 'data.R' 'depth.R' 'duration.R' 'elapse.R' 'events.R' 'hiatus.R' 'interpolate.R' 'interval.R' 'mutators.R' 'occurrence.R' 'phases.R' 'plot.R' 'read.R' 'sensitivity.R' 'show.R' 'sort.R' 'subset.R' 'summary.R' 'tempo.R' 'test.R' 'transition.R' 'validate.R' 'zzz.R'

Config/roxygen2/version 8.0.0

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

Date/Publication 2026-06-30 10:05:36 UTC

RemoteUrl https://github.com/ArchaeoStat/ArchaeoPhases

RemoteRef v2.1.1

RemoteSha a14e10f9471c7904d3dbca93beb2677467470714

Contents

activity . . . 3
allen_analyze . . . 5
allen_complement . . . 6
allen_composition . . . 7
allen_converse . . . 9
allen_illustrate . . . 10
allen_intersect . . . 12
allen_joint_concurrency . . . 13
allen_observe . . . 14
allen_observe_frequency . . . 16
allen_relation . . . 17
allen_relation_code . . . 19
allen_union . . . 20
as_coda . . . 22
as_events . . . 23
as_phases . . . 24
bind . . . 26
boundaries . . . 27
bury . . . 28
check . . . 30
data.frame . . . 32
duration . . . 32

elapse . . . 33
hiatus . . . 34
interpolate . . . 36
interval_credible . . . 37
interval_hdr . . . 38
mcmc_events . . . 39
mcmc_phases . . . 40
names . . . 40
occurrence . . . 41
older . . . 42
phases . . . 43
plot_events . . . 45
plot_phases . . . 47
read_bcal . . . 49
read_chronomodel . . . 50
read_oxcal . . . 51
sensitivity . . . 52
sort . . . 53
sort.list . . . 54
subset . . . 55
summary . . . 56
tempo . . . 58
transition . . . 60

Index **62**

`activity` *Activity Plot*

Description

Plots the first derivative of the `tempo` plot Bayesian estimate.

Usage

```
activity(object, ...)

## S4 method for signature 'EventsMCMC'
activity(
  object,
  from = min(object),
  to = max(object),
  grid = getOption("ArchaeoPhases.grid")
)

## S4 method for signature 'CumulativeEvents'
activity(object)
```

```
## S4 method for signature 'ActivityEvents,missing'
plot(
  x,
  calendar = get_calendar(),
  main = NULL,
  sub = NULL,
  ann = graphics::par("ann"),
  axes = TRUE,
  frame.plot = axes,
  panel.first = NULL,
  panel.last = NULL,
  ...
)
```

Arguments

object	An EventsMCMC or a CumulativeEvents object.
...	Other graphical parameters may also be passed as arguments to this function, particularly, border, col, lwd or lty.
from	A length-one numeric vector giving the earliest date to estimate for (expressed in *rata die*).
to	A length-one numeric vector giving the latest date to estimate for (expressed in *rata die*.
grid	A length-one numeric vector specifying the number of equally spaced points of the temporal grid.
x	An ActivityEvents object.
calendar	A aion::TimeScale object specifying the target calendar (see calendar()).
main	A character string giving a main title for the plot.
sub	A character string giving a subtitle for the plot.
ann	A logical scalar: should the default annotation (title and x and y axis labels) appear on the plot?
axes	A logical scalar: should axes be drawn on the plot?
frame.plot	A logical scalar: should a box be drawn around the plot?
panel.first	An an expression to be evaluated after the plot axes are set up but before any plotting takes place. This can be useful for drawing background grids.
panel.last	An expression to be evaluated after plotting has taken place but before the axes, title and box are added.

Value

- activity() returns an ActivityEvents object.
- plot() is called it for its side-effects: it results in a graphic being displayed (invisibly returns x).

Author(s)

A. Philippe, M.-A. Vibet, T. S. Dye, N. Frerebeau

References

Dye, T. S. (2016). Long-term rhythms in the development of Hawaiian social stratification. *Journal of Archaeological Science*, 71: 1-9. doi:10.1016/j.jas.2016.05.006.

See Also

Other event tools: `elapse()`, `occurrence()`, `tempo()`

Examples

```
## Coerce to MCMC
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)
eve <- eve[1:10000, ]

## Tempo plot
tmp <- tempo(eve)
plot(tmp, interval = "credible", panel.first = grid())
plot(tmp, interval = "gauss", panel.first = grid())

## Activity plot
act <- activity(tmp)
plot(act, panel.first = grid())
```

`allen_analyze` *Analyze Composite Allen Relations*

Description

Visualize composite Allen relations with a Nokel lattice.

Usage

```
allen_analyze(x, y, ...)
```

Arguments

`x, y`	A `character` string denoting an Allen relation.
`...`	Further arguments to be passed to internal methods.

Value

`allen_analyze()` is called it for its side-effects: it results in a graphic being displayed.

Author(s)

T. S. Dye

See Also

Other Allen's intervals: allen_analyze_relations(), allen_complement(), allen_composition(), allen_converse(), allen_illustrate(), allen_illustrate_relations(), allen_intersect(), allen_joint_concurrency(), allen_observe(), allen_observe_frequency(), allen_plot(), allen_relation(), allen_relation_code(), allen_union()

Examples

```
allen_analyze("mDFo", "MdfO", main = "Composite reticulation relation")
```

---

allen_complement *Complement of an Allen Relation*

Description

Complement of an Allen Relation

Usage

```
allen_complement(x, ...)

## S4 method for signature 'character'
allen_complement(x)

## S4 method for signature 'matrix'
allen_complement(x)
```

Arguments

x	A character vector or matrix of Allen relations (typically returned by allen_relation()).
...	Currently not used.

Value

A character vector or matrix (same as x).

Author(s)

T. S. Dye, N. Frerebeau

References

Allen, J. F. (1983). Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11): 832-843. doi:10.1145/182.358434.

See Also

Other Allen's intervals: allen_analyze(), allen_analyze_relations(), allen_composition(), allen_converse(), allen_illustrate(), allen_illustrate_relations(), allen_intersect(), allen_joint_concurrency(), allen_observe(), allen_observe_frequency(), allen_plot(), allen_relation(), allen_relation_code(), allen_union()

Examples

```
## Data from Husi 2022
loire <- data.frame(
  lower = c(625, 700, 1200, 1225, 1250, 500, 1000, 1200,
            1325, 1375, 1200, 1300, 1375, 1275, 1325),
  upper = c(750, 825, 1250, 1275, 1325, 700, 1300, 1325,
            1400, 1500, 1300, 1375, 1500, 1325, 1425)
)

## Basic relations
allen_relation(loire$lower, loire$upper)

## Complement
(comp <- allen_complement("F")) # "pmoDseSdfOMP"

## Converse
(conv <- allen_converse(comp)) # "pmoFDseSdOMP"

## Composition
allen_composition("oFD", "oFDseS") # "pmoFD"

## Intersection
allen_intersect("pFsSf", "pmoFD") # "pF"

# Union
allen_union("pFsSf", "pmoFD") # "pmoFDsSf"
```

---

allen_composition *Composition of Allen Relations*

Description

Composition of Allen Relations

Usage

```
allen_composition(x, y, ...)

## S4 method for signature 'character,character'
allen_composition(x, y)
```

Arguments

x, y	A character vector of Allen relations.
...	Currently not used.

Value

A character vector.

Author(s)

T. S. Dye, N. Frerebeau

References

Allen, J. F. (1983). Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11): 832-843. doi:10.1145/182.358434.

See Also

Other Allen's intervals: allen_analyze(), allen_analyze_relations(), allen_complement(), allen_converse(), allen_illustrate(), allen_illustrate_relations(), allen_intersect(), allen_joint_concurrency(), allen_observe(), allen_observe_frequency(), allen_plot(), allen_relation(), allen_relation_code(), allen_union()

Examples

```
## Data from Husi 2022
loire <- data.frame(
  lower = c(625, 700, 1200, 1225, 1250, 500, 1000, 1200,
            1325, 1375, 1200, 1300, 1375, 1275, 1325),
  upper = c(750, 825, 1250, 1275, 1325, 700, 1300, 1325,
            1400, 1500, 1300, 1375, 1500, 1325, 1425)
)

## Basic relations
allen_relation(loire$lower, loire$upper)

## Complement
(comp <- allen_complement("F")) # "pmoDseSdfOMP"

## Converse
(conv <- allen_converse(comp)) # "pmoFDseSdOMP"

## Composition
allen_composition("oFD", "oFDseS") # "pmoFD"

## Intersection
allen_intersect("pFsSf", "pmoFD") # "pF"

# Union
allen_union("pFsSf", "pmoFD") # "pmoFDsSf"
```

allen_converse *Converse of an Allen Relation*

Description

Converse of an Allen Relation

Usage

```
allen_converse(x, ...)

## S4 method for signature 'character'
allen_converse(x)

## S4 method for signature 'matrix'
allen_converse(x)
```

Arguments

x	A character vector or matrix of Allen relations (typically returned by allen_relation()).
...	Currently not used.

Value

A character vector or matrix (same as x).

Author(s)

T. S. Dye, N. Frerebeau

References

Allen, J. F. (1983). Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11): 832-843. doi:10.1145/182.358434.

See Also

Other Allen's intervals: allen_analyze(), allen_analyze_relations(), allen_complement(), allen_composition(), allen_illustrate(), allen_illustrate_relations(), allen_intersect(), allen_joint_concurrency(), allen_observe(), allen_observe_frequency(), allen_plot(), allen_relation(), allen_relation_code(), allen_union()

Examples

```
## Data from Husi 2022
loire <- data.frame(
  lower = c(625, 700, 1200, 1225, 1250, 500, 1000, 1200,
            1325, 1375, 1200, 1300, 1375, 1275, 1325),
  upper = c(750, 825, 1250, 1275, 1325, 700, 1300, 1325,
            1400, 1500, 1300, 1375, 1500, 1325, 1425)
)

## Basic relations
allen_relation(loire$lower, loire$upper)

## Complement
(comp <- allen_complement("F")) # "pmoDseSdfOMP"

## Converse
(conv <- allen_converse(comp)) # "pmoFDseSdOMP"

## Composition
allen_composition("oFD", "oFDseS") # "pmoFD"

## Intersection
allen_intersect("pFsSf", "pmoFD") # "pF"

# Union
allen_union("pFsSf", "pmoFD") # "pmoFDsSf"
```

`allen_illustrate` *Illustrate Basic and Composite Allen Relations*

Description

Illustrate Basic and Composite Allen Relations

Usage

```
allen_illustrate(relations = "basic", ...)
```

Arguments

`relations`	A `character` string specifying the relation. It must be one of "`basic`", "`concurrent`", "`distinct`", "`stratigraphic`", "`branching`", "`transformation`", "`reticulation`", "`sequence`", "`branch`", "`transform`", or "`reticulate`" (see details).
`...`	Further arguments to be passed to internal methods.

Details

Illustrate basic and composite Allen relations for several chronological model domains with a Nokel lattice. Chronological model domains include stratigraphy and branching, transformative, and reticulate processes of artifact change.

The illustrative graphics include:

`basic` the 13 basic Allen relations (default);

`concurrent` concurrent relations;

`distinct` relations with distinct endpoints;

`stratigraphic` basic relations established by an observation of superposition;

`branching` basic branching relations;

`transformation` basic relations of transformation;

`reticulation` basic relations of reticulation;

`sequence` composite relations in a stratigraphic sequence;

`branch` composite relations of branching;

`transform` composite relations of transformation; or

`reticulate` composite relations of reticulation.

Value

`allen_illustrate()` is called it for its side-effects: it results in a graphic being displayed.

Author(s)

T. S. Dye

References

Harris, E. C. (1997). *Principles of Archaeological Stratigraphy*. Second edition. London: Academic Press.

Lyman, R. L. and O'Brien, M. J. (2017). "Sedation and Cladistics: The Difference between Anagenetic and Cladogenetic Evolution". In *Mapping Our Ancestors: Phylogenetic Approaches in Anthropology and Prehistory*, edited by Lipo, C. P., O'Brien, M. J., Couard, M., and Shennan, S. J. New York: Routledge. doi:10.4324/9780203786376.

Viola, T. (2020). *Peirce on the Uses of History*. De Gruyter. doi:10.1515/9783110651560. See chapter 3, "Historicity as Process", especially p. 83-88.

See Also

Other Allen's intervals: `allen_analyze()`, `allen_analyze_relations()`, `allen_complement()`, `allen_composition()`, `allen_converse()`, `allen_illustrate_relations()`, `allen_intersect()`, `allen_joint_concurrency()`, `allen_observe()`, `allen_observe_frequency()`, `allen_plot()`, `allen_relation()`, `allen_relation_code()`, `allen_union()`

Examples

```
## Plot the basic Allen relations
allen_illustrate()
```

`allen_intersect` *Intersection of Allen Relations*

Description

Intersection of Allen Relations

Usage

```
allen_intersect(x, y, ...)

## S4 method for signature 'character,character'
allen_intersect(x, y)
```

Arguments

x, y	A `character` vector of Allen relations.
...	Currently not used.

Value

A `character` vector.

Author(s)

T. S. Dye, N. Frerebeau

References

Allen, J. F. (1983). Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11): 832-843. doi:10.1145/182.358434.

See Also

Other Allen's intervals: `allen_analyze()`, `allen_analyze_relations()`, `allen_complement()`, `allen_composition()`, `allen_converse()`, `allen_illustrate()`, `allen_illustrate_relations()`, `allen_joint_concurrency()`, `allen_observe()`, `allen_observe_frequency()`, `allen_plot()`, `allen_relation()`, `allen_relation_code()`, `allen_union()`

Examples

```
## Data from Husi 2022
loire <- data.frame(
  lower = c(625, 700, 1200, 1225, 1250, 500, 1000, 1200,
            1325, 1375, 1200, 1300, 1375, 1275, 1325),
  upper = c(750, 825, 1250, 1275, 1325, 700, 1300, 1325,
            1400, 1500, 1300, 1375, 1500, 1325, 1425)
)

## Basic relations
allen_relation(loire$lower, loire$upper)

## Complement
(comp <- allen_complement("F")) # "pmoDseSdfOMP"

## Converse
(conv <- allen_converse(comp)) # "pmoFDseSdOMP"

## Composition
allen_composition("oFD", "oFDseS") # "pmoFD"

## Intersection
allen_intersect("pFsSf", "pmoFD") # "pF"

# Union
allen_union("pFsSf", "pmoFD") # "pmoFDsSf"
```

`allen_joint_concurrency` *Joint Concurrence of Two or More Observed Intervals*

Description

Estimates the age of an undated context based on the known depositional history of associated artifacts.

Usage

```
allen_joint_concurrency(x, groups, ...)

## S4 method for signature 'EventsMCMC,list'
allen_joint_concurrency(x, groups, ...)
```

Arguments

`x`	An `EventsMCMC` object containing the output of the MCMC algorithm.
`groups`	A `list` of (named) vector of names or indexes of columns in `x` (see `phases()`).
`...`	Currently not used.

Value

A PhasesMCMC object.

Author(s)

T. S. Dye

See Also

Other Allen's intervals: allen_analyze(), allen_analyze_relations(), allen_complement(), allen_composition(), allen_converse(), allen_illustrate(), allen_illustrate_relations(), allen_intersect(), allen_observe(), allen_observe_frequency(), allen_plot(), allen_relation(), allen_relation_code(), allen_union()

---

allen_observe *Observe the Relation Between two Phases*

Description

Plots an empirical Nökel lattice.

Usage

```
allen_observe(x, groups, ...)

## S4 method for signature 'PhasesMCMC,missing'
allen_observe(x, converse = TRUE, ...)

## S4 method for signature 'EventsMCMC,list'
allen_observe(x, groups, converse = TRUE, ...)
```

Arguments

x	An EventsMCMC or a PhasesMCMC object containing the output of the MCMC algorithm.
groups	A list of (named) vector of names or indexes of columns in x (see phases()).
...	Further arguments to be passed to internal methods.
converse	A logical scalar: should converse relations be observed?

Value

allen_observe() is called it for its side-effects: it results in a graphic being displayed (invisibly returns x).

Author(s)

T. S. Dye, N. Frerebeau

See Also

Other Allen's intervals: allen_analyze(), allen_analyze_relations(), allen_complement(), allen_composition(), allen_converse(), allen_illustrate(), allen_illustrate_relations(), allen_intersect(), allen_joint_concurrency(), allen_observe_frequency(), allen_plot(), allen_relation(), allen_relation_code(), allen_union()

Examples

```
if (requireNamespace("ArchaeoData", quietly = TRUE)) {
  ## Load the Anglo Saxon burials dataset
  path <- system.file("oxcal/burials.csv", package = "ArchaeoData")
  burials <- read.table(path, header = TRUE, sep = ",", dec = ".",
                        check.names = FALSE)

  ## Coerce to event
  burials <- as_events(burials, calendar = CE())

  ## Dates associated with bead BE3 Amber
  be3_amber <- c(
    "UB-4836 (WG27)", "UB-5208 (ApD107)", "UB-4965 (ApD117)",
    "UB-4735 (Ber022)", "UB-4739 (Ber134/1)", "UB-4728 (MH064)",
    "UB-4729 (MH068)", "UB-4732 (MH094)", "UB-4733 (MH095)",
    "UB-4734 (MH105c)", "UB-4984 (Lec018)", "UB-4709 (EH014)",
    "UB-4707 (EH079)", "UB-4708 (EH083)", "UB-6033 (WHes113)",
    "UB-4706 (WHes118)", "UB-4705 (WHes123)", "UB-6040 (CasD053)",
    "UB-6037 (CasD134)", "UB-6472 (BuD222)", "UB-6473 (BuD250)",
    "UB-6476 (BuD339)", "UB-4963 (SPTip208)", "UB-4890 (MelSG075)",
    "UB-4887 (MelSG082)", "UB-4888 (MelSG089)", "MaDE1 & E2",
    "UB-4552 (MaDE3)", "UB-4975 (AstCli12)", "UB-4835 (ApD134)",
    "SUERC-39108 ERLK G322", "SUERC-39109 ERL G362", "SUERC-39112 ERL G405",
    "SUERC-51560 ERL G038", "SUERC-39091 (ERL G003)", "SUERC-39092 (ERL G005)",
    "SUERC-39113 (ERL G417)", "SUERC-51549 (ERL G195)", "SUERC-51552 (ERL G107)",
    "SUERC-51550 (ERL G254)"
  )

  ## Dates associated with bead BE1 Dghnt
  be1_dghnt <- c(
    "UB-4503 (Lec148)", "UB-4506 (Lec172/2)", "UB-6038 (CasD183)",
    "UB-4512 (EH091)", "UB-4501 (Lec014)", "UB-4507 (Lec187)",
    "UB-4502 (Lec138)", "UB-4042 (But1674)", "SUERC-39100 (ERL G266)"
  )

  ## Construct a list of lists
  chains <- list(
    list("BE3-Amber" = be3_amber, "BE1-Dghnt" = be1_dghnt),
    list("BE1-Dghnt" = be1_dghnt, "BE3-Amber" = be3_amber)
  )

  ## Plot
  allen_observe(x = burials, groups = chains)

  ## Observe 2x2 frequency matrix of the relation of trunk to branch
```

```
  allen_observe_frequency(burials, groups = chains, set = "oFD")
}
```

`allen_observe_frequency` *Observed Frequency of an Allen Set*

Description

Creates a matrix of observed frequencies of a given Allen set among two or more groups of chains from the MCMC output of a Bayesian calibration.

Usage

```
allen_observe_frequency(x, groups, ...)

## S4 method for signature 'PhasesMCMC,missing'
allen_observe_frequency(x, set)

## S4 method for signature 'EventsMCMC,list'
allen_observe_frequency(x, groups, ...)
```

Arguments

`x`	An `EventsMCMC` or a `PhasesMCMC` object containing the output of the MCMC algorithm.
`groups`	A `list` of (named) vector of names or indexes of columns in `x` (see `phases()`).
`...`	Currently not used.
`set`	A `character` string representation of an Allen set.

Value

A square `matrix` of observed frequencies.

Author(s)

T. S. Dye, N. Frerebeau

See Also

Other Allen's intervals: `allen_analyze()`, `allen_analyze_relations()`, `allen_complement()`, `allen_composition()`, `allen_converse()`, `allen_illustrate()`, `allen_illustrate_relations()`, `allen_intersect()`, `allen_joint_concurrency()`, `allen_observe()`, `allen_plot()`, `allen_relation()`, `allen_relation_code()`, `allen_union()`

Examples

```
if (requireNamespace("ArchaeoData", quietly = TRUE)) {
  ## Load the Anglo Saxon burials dataset
  path <- system.file("oxcal/burials.csv", package = "ArchaeoData")
  burials <- read.table(path, header = TRUE, sep = ",", dec = ".",
                        check.names = FALSE)

  ## Coerce to event
  burials <- as_events(burials, calendar = CE())

  ## Dates associated with bead BE3 Amber
  be3_amber <- c(
    "UB-4836 (WG27)", "UB-5208 (ApD107)", "UB-4965 (ApD117)",
    "UB-4735 (Ber022)", "UB-4739 (Ber134/1)", "UB-4728 (MH064)",
    "UB-4729 (MH068)", "UB-4732 (MH094)", "UB-4733 (MH095)",
    "UB-4734 (MH105c)", "UB-4984 (Lec018)", "UB-4709 (EH014)",
    "UB-4707 (EH079)", "UB-4708 (EH083)", "UB-6033 (WHes113)",
    "UB-4706 (WHes118)", "UB-4705 (WHes123)", "UB-6040 (CasD053)",
    "UB-6037 (CasD134)", "UB-6472 (BuD222)", "UB-6473 (BuD250)",
    "UB-6476 (BuD339)", "UB-4963 (SPTip208)", "UB-4890 (MelSG075)",
    "UB-4887 (MelSG082)", "UB-4888 (MelSG089)", "MaDE1 & E2",
    "UB-4552 (MaDE3)", "UB-4975 (AstCli12)", "UB-4835 (ApD134)",
    "SUERC-39108 ERLK G322", "SUERC-39109 ERL G362", "SUERC-39112 ERL G405",
    "SUERC-51560 ERL G038", "SUERC-39091 (ERL G003)", "SUERC-39092 (ERL G005)",
    "SUERC-39113 (ERL G417)", "SUERC-51549 (ERL G195)", "SUERC-51552 (ERL G107)",
    "SUERC-51550 (ERL G254)"
  )

  ## Dates associated with bead BE1 Dghnt
  be1_dghnt <- c(
    "UB-4503 (Lec148)", "UB-4506 (Lec172/2)", "UB-6038 (CasD183)",
    "UB-4512 (EH091)", "UB-4501 (Lec014)", "UB-4507 (Lec187)",
    "UB-4502 (Lec138)", "UB-4042 (But1674)", "SUERC-39100 (ERL G266)"
  )

  ## Construct a list of lists
  chains <- list(
    list("BE3-Amber" = be3_amber, "BE1-Dghnt" = be1_dghnt),
    list("BE1-Dghnt" = be1_dghnt, "BE3-Amber" = be3_amber)
  )

  ## Plot
  allen_observe(x = burials, groups = chains)

  ## Observe 2x2 frequency matrix of the relation of trunk to branch
  allen_observe_frequency(burials, groups = chains, set = "oFD")
}
```

`allen_relation` *Allen Relation Between Definite Intervals*

Description

Allen Relation Between Definite Intervals

Usage

```
allen_relation(x, y, ...)

## S4 method for signature 'numeric,numeric'
allen_relation(x, y)

## S4 method for signature 'ANY,missing'
allen_relation(x)
```

Arguments

x, y	A numeric vector giving the lower and upper boundaries of the time intervals, respectively. If y is missing, an attempt is made to interpret x in a suitable way (see grDevices::xy.coords()).
...	Currently not used.

Details

Relation			Converse
precedes	(p)	(P)	preceded by
meets	(m)	(M)	met by
overlaps	(o)	(O)	overlapped by
finished by	(F)	(f)	finishes
contains	(D)	(d)	during
starts	(s)	(S)	started by
equals	(e)		

Value

A character matrix specifying the Allen relations.

Author(s)

T. S. Dye, N. Frerebeau

References

Allen, J. F. (1983). Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11): 832-843. doi:10.1145/182.358434.

Alspaugh, T. (2019). Allen's Interval Algebra. URL: https://thomasalspaugh.org/pub/fnd/allen.html.

See Also

Other Allen's intervals: allen_analyze(), allen_analyze_relations(), allen_complement(), allen_composition(), allen_converse(), allen_illustrate(), allen_illustrate_relations(), allen_intersect(), allen_joint_concurrency(), allen_observe(), allen_observe_frequency(), allen_plot(), allen_relation_code(), allen_union()

Examples

```
## Data from Husi 2022
loire <- data.frame(
  lower = c(625, 700, 1200, 1225, 1250, 500, 1000, 1200,
            1325, 1375, 1200, 1300, 1375, 1275, 1325),
  upper = c(750, 825, 1250, 1275, 1325, 700, 1300, 1325,
            1400, 1500, 1300, 1375, 1500, 1325, 1425)
)

## Basic relations
allen_relation(loire$lower, loire$upper)

## Complement
(comp <- allen_complement("F")) # "pmoDseSdfOMP"

## Converse
(conv <- allen_converse(comp)) # "pmoFDseSdOMP"

## Composition
allen_composition("oFD", "oFDseS") # "pmoFD"

## Intersection
allen_intersect("pFsSf", "pmoFD") # "pF"

# Union
allen_union("pFsSf", "pmoFD") # "pmoFDsSf"
```

---

allen_relation_code *The Basic Allen Relation Set*

Description

The Basic Allen Relation Set

Usage

```
allen_relation_code(...)

allen_relation_string(...)

allen_relation_concurrent(...)

allen_relation_distinct(...)
```

Arguments

...	Currently not used.

Value

- `allen_relation_code()` returns a `character` vector of one-letter codes for the thirteen basic Allen relations.
- `allen_relation_string()` returns a `character` vector of string descriptors of the Allen basic relations.
- `allen_relation_concurrent()` returns a `character` vector of nine one-letter codes for the Allen concurrent relations.
- `allen_relation_distinct()` returns the six value Allen relation set for intervals with distinct endpoints.

Note

The codes were proposed by Thomas Alspaugh.

Author(s)

T. S. Dye

References

Allen, J. F. (1983). Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11): 832-843. doi:10.1145/182.358434.

Alspaugh, T. (2019). Allen's Interval Algebra. URL: https://thomasalspaugh.org/pub/fnd/allen.html.

See Also

Other Allen's intervals: `allen_analyze()`, `allen_analyze_relations()`, `allen_complement()`, `allen_composition()`, `allen_converse()`, `allen_illustrate()`, `allen_illustrate_relations()`, `allen_intersect()`, `allen_joint_concurrency()`, `allen_observe()`, `allen_observe_frequency()`, `allen_plot()`, `allen_relation()`, `allen_union()`

`allen_union` *Union of Allen Relations*

Description

Union of Allen Relations

Usage

```
allen_union(x, y, ...)

## S4 method for signature 'character,character'
allen_union(x, y)
```

Arguments

x, y	A character vector of Allen relations.
...	Currently not used.

Value

A character vector.

Author(s)

T. S. Dye, N. Frerebeau

References

Allen, J. F. (1983). Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11): 832-843. doi:10.1145/182.358434.

See Also

Other Allen's intervals: allen_analyze(), allen_analyze_relations(), allen_complement(), allen_composition(), allen_converse(), allen_illustrate(), allen_illustrate_relations(), allen_intersect(), allen_joint_concurrency(), allen_observe(), allen_observe_frequency(), allen_plot(), allen_relation(), allen_relation_code()

Examples

```
## Data from Husi 2022
loire <- data.frame(
  lower = c(625, 700, 1200, 1225, 1250, 500, 1000, 1200,
            1325, 1375, 1200, 1300, 1375, 1275, 1325),
  upper = c(750, 825, 1250, 1275, 1325, 700, 1300, 1325,
            1400, 1500, 1300, 1375, 1500, 1325, 1425)
)

## Basic relations
allen_relation(loire$lower, loire$upper)

## Complement
(comp <- allen_complement("F")) # "pmoDseSdfOMP"

## Converse
(conv <- allen_converse(comp)) # "pmoFDseSdOMP"
```

```
## Composition
allen_composition("oFD", "oFDseS") # "pmoFD"

## Intersection
allen_intersect("pFsSf", "pmoFD") # "pF"

# Union
allen_union("pFsSf", "pmoFD") # "pmoFDsSf"
```

---

as_coda *Coerce to Coda*

Description

Extracts parallel chains from an MCMC object to create an mcmc.list object for use with **coda** diagnostic tools.

Usage

```
as_coda(from, ...)

## S4 method for signature 'MCMC'
as_coda(from, chains = 1)
```

Arguments

from	from An object to be coerced.
...	Currently not used.
chains	An integer specifying the number of parallel chains (defaults to 1).

Value

An coda::mcmc.list object.

Author(s)

A. Philippe, M.-A. Vibet

See Also

coda::mcmc(), coda::mcmc.list()

Other read methods: as_events(), as_phases(), check, read_bcal(), read_chronomodel, read_oxcal()

Examples

```
if (requireNamespace("coda", quietly = TRUE)) {
  ## Load coda
  library(coda)

  ## Coerce to MCMC
  eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)

  ## Coerce to coda
  mc <- as_coda(eve[, 1:2], chains = 3)
  plot(mc)

  ## Autocorrelation
  autocorr.plot(mc)

  ## Gelman-Rubin diagnostic
  ## The multivariate criterion can not be evaluated when a phase
  ## contains only one date. This induces colinearity problems.
  gelman.diag(mc)
  gelman.plot(mc)
}
```

---

as_events *Coerce to Events*

Description

Coerce to Events

Usage

```
as_events(from, ...)

## S4 method for signature 'matrix'
as_events(from, calendar, iteration = NULL)

## S4 method for signature 'data.frame'
as_events(from, calendar, iteration = NULL)
```

Arguments

`from`	from An object to be coerced.
`...`	Currently not used.
`calendar`	A `aion::TimeScale` object specifying the source calendar (see `calendar()`).
`iteration`	A length-one `numeric` vector specifying the index of the iteration column.

Value

An `EventsMCMC` object.

Author(s)

A. Philippe, M.-A. Vibet, N. Frerebeau

See Also

Other read methods: as_coda(), as_phases(), check, read_bcal(), read_chronomodel, read_oxcal()

Examples

```
## Coerce to events
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)

## Plot first event
plot(eve[, 1], interval = "hdr")

## Colorfull plot
plot(eve, col.density = c("#4477AA", "#EE6677", "#228833", "#CCBB44"))

## Plot events
plot(eve, calendar = CE(), interval = "credible", level = 0.68)
plot(eve, calendar = BP(), interval = "hdr", level = 0.68)

## Plot only 95% credible interval
plot(eve, density = FALSE, interval = "credible", lwd = 3, tcl = 0)
```

---

as_phases *Coerce to Phases*

Description

Coerce to Phases

Usage

```
as_phases(from, ...)

## S4 method for signature 'matrix'
as_phases(
  from,
  calendar = get_calendar(),
  start = seq(from = 1, to = ncol(from), by = 2),
  stop = start + 1,
  names = NULL,
  iteration = NULL
)

## S4 method for signature 'data.frame'
as_phases(
```

```
  from,
  calendar = get_calendar(),
  start = seq(from = 1, to = ncol(from), by = 2),
  stop = start + 1,
  names = NULL,
  iteration = NULL
)
```

Arguments

from	from An object to be coerced.
...	Currently not used.
calendar	A aion::TimeScale object specifying the source calendar (see calendar()).
start	An integer vector specifying the index of the columns corresponding to the beginning of the phases. If missing, every other column is used starting from the first column (after deleting the iteration column, if any).
stop	An integer vector specifying the index of the columns corresponding to the end of the phases. If missing, every other column is used starting from the second column (after deleting the iteration column, if any).
names	A character vector giving the names of the phases.
iteration	A length-one numeric vector specifying the index of the iteration column.

Value

A PhasesMCMC object.

Author(s)

A. Philippe, M.-A. Vibet, N. Frerebeau

See Also

Other read methods: as_coda(), as_events(), check, read_bcal(), read_chronomodel, read_oxcal()

Examples

```
## Coerce to phases
(pha <- as_phases(mcmc_phases, calendar = CE(), start = c(1, 3), iteration = 1))
summary(pha, calendar = CE())

## Plot phases
plot(pha)
plot(pha, succession = "hiatus")
plot(pha, succession = "transition")

## Compute phases from events
(eve <- as_events(mcmc_events, calendar = CE(), iteration = 1))

## Compute min-max range for all chains
```

```
pha1 <- phases(eve)
summary(pha1, calendar = CE())

## Compute min-max range by group
pha2 <- phases(eve, groups = list(phase_1 = c(1, 3), phase_2 = c(2, 4)))
summary(pha2, calendar = CE())
```

`bind` *Combine two MCMC Objects*

Description

Combine two MCMC Objects

Usage

```
## S4 method for signature 'MCMC,MCMC'
cbind2(x, y)
```

Arguments

x, y	An MCMC object.

Value

An MCMC object.

Author(s)

N. Frerebeau

See Also

Other subsetting methods: subset

Examples

```
## Events
(eve <- as_events(mcmc_events, calendar = CE(), iteration = 1))

eve[1:1000, ] # Select the first 1000 iterations
eve[, 1:2]    # Select the first 2 events

cbind2(eve[, 1:2], eve[, 3:4]) # Combine two MCMC objects
sort(eve, decreasing = TRUE)   # Sort events in descending order

## Phases
(pha <- as_phases(mcmc_phases, start = c(1, 3), calendar = CE(), iteration = 1))

pha[1:1000, , ]          # Select the first 1000 iterations
pha[, 1, , drop = FALSE] # Select the first phase
```

`boundaries` *Phase Time Range*

Description

Computes the shortest interval that satisfies $P(PhaseMin < IntervalInf < IntervalSup < PhaseMax|M) = level$ for each phase.

Usage

```
boundaries(x, y, ...)

## S4 method for signature 'numeric,numeric'
boundaries(x, y, level = 0.95)

## S4 method for signature 'PhasesMCMC,missing'
boundaries(x, level = 0.95)
```

Arguments

`x`, `y`	A `numeric` vector. If `y` is missing, `x` must be an `PhasesMCMC` object.
`...`	Currently not used.
`level`	A length-one `numeric` vector giving the confidence level.

Value

The endpoints of the shortest time range (at a given `level`).

Methods (by class)

- `boundaries(x = numeric, y = numeric)`: Returns a length-two `numeric` vector (terminal times).
- `boundaries(x = PhasesMCMC, y = missing)`: Returns a `TimeRange` object.

Author(s)

A. Philippe, M.-A. Vibet, N. Frerebeau

See Also

Other time ranges: `hiatus()`, `transition()`

Examples

```
## Coerce to events
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)
eve <- eve[1:10000, ]

## Compute min-max range by group
pha <- phases(eve, groups = list(A = c(1, 3), B = c(2, 4)))

## Compute phase ranges
bou <- boundaries(pha)
as.data.frame(bou)

## Compute phase transition
tra <- transition(pha)
as.data.frame(tra)

## Compute phase hiatus
hia <- hiatus(pha)
as.data.frame(hia)
```

---

bury *Age-Depth Modeling*

Description

Computes the age-depth curve from the output of the MCMC algorithm and the known depth of each dated samples.

Usage

```
bury(object, depth, ...)

## S4 method for signature 'EventsMCMC,numeric'
bury(object, depth, span = 0.75, degree = 2)

## S4 method for signature 'AgeDepthModel'
predict(object, newdata)

## S4 method for signature 'AgeDepthModel,missing'
plot(
  x,
  level = 0.95,
  calendar = get_calendar(),
  main = NULL,
  sub = NULL,
  ann = graphics::par("ann"),
  axes = TRUE,
  frame.plot = axes,
```

```
  panel.first = NULL,
  panel.last = NULL,
  ...
)
```

Arguments

object	An EventsMCMC object.
depth	A numeric vector giving of the depths of the dated samples.
...	Other graphical parameters may also be passed as arguments to this function, particularly, border, col, lwd, lty or pch.
span	A length-one numeric vector giving the regression parameter (see stats::loess()).
degree	A length-one numeric vector giving the degree of the polynomials to be used (see stats::loess()).
newdata	A numeric vector giving the depths at which ages will be predicted. If missing, the original data points are used.
x	An AgeDepthModel object.
level	A length-one numeric vector giving the confidence level.
calendar	A aion::TimeScale object specifying the target calendar (see calendar()).
main	A character string giving a main title for the plot.
sub	A character string giving a subtitle for the plot.
ann	A logical scalar: should the default annotation (title and x and y axis labels) appear on the plot?
axes	A logical scalar: should axes be drawn on the plot?
frame.plot	A logical scalar: should a box be drawn around the plot?
panel.first	An an expression to be evaluated after the plot axes are set up but before any plotting takes place. This can be useful for drawing background grids.
panel.last	An expression to be evaluated after plotting has taken place but before the axes, title and box are added.

Details

We assume it exists a function f relating the age and the depth $age = f(depth)$. We estimate the function using local regression (also called local polynomial regression): $f = loess(age\ depth)$. This estimated function f depends on the unknown dates. However, from the posterior distribution of the age/date sequence, we can evaluate the posterior distribution of the age function for each desired depth.

Value

- bury() returns an AgeDepthModel object.
- predict() returns an EventsMCMC object.
- plot() is called it for its side-effects: it results in a graphic being displayed (invisibly returns x).

Author(s)

A. Philippe

References

Jha, D. K., Sanyal, P. & Philippe, A. (2020). Multi-Proxy Evidence of Late Quaternary Climate and Vegetational History of North-Central India: Implication for the Paleolithic to Neolithic Phases. *Quaternary Science Reviews*, 229: 106121. doi:10.1016/j.quascirev.2019.106121.

Ghosh, S., Sanyal, P., Roy, S., Bhushan, R., Sati, S. P., Philippe, A. & Juyal, N. (2020). Early Holocene Indian Summer Monsoon and Its Impact on Vegetation in the Central Himalaya: Insight from dD and d13C Values of Leaf Wax Lipid. *The Holocene*, 30(7): 1063-1074. doi:10.1177/0959683620908639.

See Also

Other age-depth modeling tools: `interpolate()`

Examples

```
## Coerce to MCMC
eve <- matrix(rnorm(6000, (1:6)^2), ncol = 6, byrow = TRUE)
eve <- as_events(eve, calendar = CE())

## Compute an age-depth curve
age <- bury(eve, depth = 1:6)
plot(age)

## Predict new values
new <- predict(age, newdata = 1.5:5.5)
summary(new)

plot(eve)
plot(new)
```

---

check *Check for an Original MCMC File*

Description

Checks whether or not a file is identical to the one used to create an object.

Usage

```
is_original(object, ...)

## S4 method for signature 'MCMC'
is_original(object, file, download = FALSE)
```

```
## S4 method for signature 'PhasesMCMC'
is_original(object, file, download = FALSE)

## S4 method for signature 'CumulativeEvents'
is_original(object, file, download = FALSE)

## S4 method for signature 'ActivityEvents'
is_original(object, file, download = FALSE)

## S4 method for signature 'OccurrenceEvents'
is_original(object, file, download = FALSE)
```

Arguments

object	An object (typically an MCMC object).
...	Currently not used.
file	Either a path to a CSV file or a connection.
download	A logical scalar: should the remote file be downloaded and hashed locally?

Value

A logical: TRUE if the files match, FALSE otherwise.

Author(s)

T. S. Dye, N. Frerebeau

See Also

digest::digest()

Other read methods: as_coda(), as_events(), as_phases(), read_bcal(), read_chronomodel, read_oxcal()

Examples

```
## Not run:
## Import OxCal Output
path_output <- system.file("oxcal/ksarakil/MCMC_Sample.csv", package = "ArchaeoData")
url_output <- paste0("https://raw.githubusercontent.com/ArchaeoStat/ArchaeoData/master/",
                     "inst/oxcal/ksarakil/MCMC_Sample.csv")

oxcal <- read_oxcal(path_output)

## Check md5 sum
is_original(oxcal, path_output) # Same as local file? TRUE
is_original(oxcal, url_output, download = FALSE) # Same as remote file? FALSE
is_original(oxcal, url_output, download = TRUE) # Same as remote file? TRUE

## End(Not run)
```

data.frame *Coerce to a Data Frame*

Description

Coerce to a Data Frame

Usage

```
## S4 method for signature 'CumulativeEvents'
as.data.frame(x, ..., calendar = get_calendar())

## S4 method for signature 'ActivityEvents'
as.data.frame(x, ..., calendar = get_calendar())

## S4 method for signature 'OccurrenceEvents'
as.data.frame(x, ..., calendar = get_calendar())

## S4 method for signature 'TimeRange'
as.data.frame(x, ..., calendar = get_calendar())
```

Arguments

x	An R object.
...	Further parameters to be passed to data.frame().
calendar	A aion::TimeScale object specifying the target calendar (see calendar()).

Value

A data.frame with an extra time column giving the (decimal) years at which the time series was sampled.

Author(s)

N. Frerebeau

duration *Phase Duration*

Description

Phase Duration

Usage

```
duration(x, y, ...)

## S4 method for signature 'numeric,numeric'
duration(x, y)

## S4 method for signature 'PhasesMCMC,missing'
duration(x)
```

Arguments

x, y	A numeric vector. If y is missing, x must be an PhasesMCMC object.
...	Currently not used.

Author(s)

A. Philippe, M.-A. Vibet, N. Frerebeau

See Also

Other phase tools: phases()

Examples

```
## Coerce to phases
pha <- as_phases(mcmc_phases, start = c(1, 3), calendar = CE(), iteration = 1)

## Compute phase duration
dur <- duration(pha)
summary(dur)
```

---

elapse *Elapsed Time Scale*

Description

Elapsed Time Scale

Usage

```
elapse(object, ...)

## S4 method for signature 'MCMC'
elapse(object, origin = 1)
```

Arguments

object	An object (typically an MCMC object).
...	Currently not used.
origin	An integer giving the position of the column corresponding to the event from which elapsed time is calculated.

Value

Returns an object of the same class as object with an elapsed

An object of the same sort as object with a new time scale.

Note

There is no year 0 in BCE/CE scale.

Author(s)

N. Frerebeau

See Also

Other event tools: activity(), occurrence(), tempo()

Examples

```
## Coerce to events
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)

## Elapsed origin
eve_elapse <- elapse(eve, origin = 4)
plot(eve_elapse)
```

---

hiatus *Hiatus Between Two Dates*

Description

Tests for the existence of a hiatus between two parameters.

Usage

```
hiatus(x, y, ...)

## S4 method for signature 'numeric,numeric'
hiatus(x, y, level = 0.95)

## S4 method for signature 'EventsMCMC,missing'
```

```
hiatus(x, level = 0.95)

## S4 method for signature 'PhasesMCMC,missing'
hiatus(x, level = 0.95)
```

Arguments

x, y	A numeric vector. If y is missing, x must be an PhasesMCMC or an EventsMCMC object.
...	Currently not used.
level	A length-one numeric vector giving the confidence level.

Details

Finds if a gap exists between two dates and returns the longest interval that satisfies $P(x < HiatusInf < HiatusSup < y|M) = level$

The hiatus between two successive phases is the longest interval that satisfies $P(Phase1Max < IntervalInf < IntervalSup < Phase2Min|M) = level$ (this assumes that the phases are in temporal order constraint).

Value

The endpoints of the hiatus between successive events/phases (at a given level).

Methods (by class)

- hiatus(x = numeric, y = numeric): Returns a length-three numeric vector (terminal times and hiatus duration, if any).
- hiatus(x = EventsMCMC, y = missing): Returns a TimeRange object.
- hiatus(x = PhasesMCMC, y = missing): Returns a TimeRange object.

Author(s)

A. Philippe, M.-A. Vibet, N. Frerebeau

See Also

Other time ranges: boundaries(), transition()

Examples

```
## Coerce to MCMC
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)
eve <- eve[1:10000, ]

## Test for anteriority
older(eve)

## Test for hiatus
```

```
hia <- hiatus(eve)
as.data.frame(hia)
```

`interpolate` *Interpolate Between Two Dates*

Description

Interpolate Between Two Dates

Usage

```
interpolate(x, y, ...)

## S4 method for signature 'numeric,numeric'
interpolate(x, y)

## S4 method for signature 'EventsMCMC,missing'
interpolate(x, e1 = 1, e2 = 2)
```

Arguments

`x`	A `numeric` vector giving the output of the MCMC algorithm for the first parameter.
`y`	A `numeric` vector giving the output of the MCMC algorithm for the second parameter.
`...`	Currently not used.
`e1, e2`	An `integer` specifying.

Details

For a given output of MCMC algorithm, this function interpolates between to events x and y (assuming $x < y$).

Author(s)

N. Frerebeau

See Also

Other age-depth modeling tools: `bury()`

Examples

```
## Coerce to events
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)
eve <- eve[1:10000, ]

## Interpolate between two events
inter <- interpolate(eve, e1 = 2, e2 = 3)
plot(inter, level = 0.95, interval = "credible")
```

---

interval_credible *Bayesian Credible Interval*

Description

Computes the shortest credible interval of the output of the MCMC algorithm for a single parameter.

Usage

```
interval_credible(x, ...)

## S4 method for signature 'MCMC'
interval_credible(x, level = 0.95, calendar = get_calendar())
```

Arguments

x	An MCMC object containing the output of the MCMC algorithm.
...	Currently not used.
level	A length-one numeric vector giving the confidence level.
calendar	A aion::TimeScale object specifying the target calendar (see calendar()).

Details

A $(100 \times level)$ % credible interval is an interval that keeps $N \times (1 - level)$ elements of the sample outside the interval.

The $(100 \times level)$ % credible interval is the shortest of all those intervals.

For instance, the 95% credible interval is the central portion of the posterior distribution that contains 95% of the values.

Value

Returns a list of numeric matrix.

Author(s)

A. Philippe, M.-A. Vibet, T. S. Dye, N. Frerebeau

See Also

arkhe::interval_credible()

Other statistics: interval_hdr(), sensitivity(), summary()

Examples

```
## Coerce to events
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)
eve <- eve[1:10000, ]

## Rata die
interval_credible(eve, level = 0.95) # Credible interval
interval_hdr(eve, level = 0.68) # HPD interval

## BP
interval_credible(eve, level = 0.95, calendar = BP()) # Credible interval
interval_hdr(eve, level = 0.95, calendar = BP()) # HPD interval
```

---

interval_hdr *Bayesian HPD Regions*

Description

Bayesian HPD Regions

Usage

```
interval_hdr(x, y, ...)

## S4 method for signature 'MCMC,missing'
interval_hdr(x, level = 0.95, calendar = get_calendar(), ...)
```

Arguments

x	An MCMC object containing the output of the MCMC algorithm.
y	Currently not used.
...	Extra arguments to be passed to stats::density().
level	A length-one numeric vector giving the confidence level.
calendar	A aion::TimeScale object specifying the target calendar (see calendar()).

Value

Returns a list of numeric matrix.

Author(s)

A. Philippe, M.-A. Vibet, T. S. Dye, N. Frerebeau

References

Hyndman, R. J. (1996). Computing and graphing highest density regions. *American Statistician*, 50: 120-126. doi:10.2307/2684423.

See Also

`stats::density()`, `arkhe::interval_hdr()`

Other statistics: `interval_credible()`, `sensitivity()`, `summary()`

Examples

```
## Coerce to events
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)
eve <- eve[1:10000, ]

## Rata die
interval_credible(eve, level = 0.95) # Credible interval
interval_hdr(eve, level = 0.68) # HPD interval

## BP
interval_credible(eve, level = 0.95, calendar = BP()) # Credible interval
interval_hdr(eve, level = 0.95, calendar = BP()) # HPD interval
```

---

mcmc_events *Events*

Description

A data set containing information on the ages of four dated events.

Usage

```
mcmc_events
```

Format

A `data.frame` with 30,000 rows and 5 variables:

`iter` Iteration of the MCMC algorithm.

`E1` Information on event 1.

`E2` Information on event 2.

`E3` Information on event 3.

`E4` Information on event 4.

See Also

Other datasets: `mcmc_phases`

mcmc_phases *Phases*

Description

A data set containing information on the start and end dates of two phases.

Usage

```
mcmc_phases
```

Format

A data.frame with 30,000 rows and 5 variables:

`iter` Iteration of the MCMC algorithm.

`P2_alpha` Start date of Phase 2.

`P2_beta` End date of Phase 2.

`P1_alpha` Start date of Phase 1.

`P1_beta` End date of Phase 1.

See Also

Other datasets: mcmc_events

names *The Names of an Object*

Description

Get or set the names of an object.

Usage

```
## S4 method for signature 'MCMC'
names(x)

## S4 replacement method for signature 'MCMC'
names(x) <- value

## S4 method for signature 'PhasesMCMC'
names(x)

## S4 replacement method for signature 'PhasesMCMC'
names(x) <- value
```

Arguments

x	An object from which to get or set names.
value	A possible value for the names of x.

Value

An object of the same sort as x with the new names assigned.

Author(s)

N. Frerebeau

See Also

Other mutators: sort(), sort.list()

---

occurrence *Occurrence Plot*

Description

A statistical graphic designed for the archaeological study of when events of a specified kind occurred.

Usage

```
occurrence(object, ...)

## S4 method for signature 'EventsMCMC'
occurrence(object, level = 0.95)
```

Arguments

object	An EventsMCMC object.
...	Currently not used.
level	A length-one numeric vector giving the confidence level.

Details

If we have k events, then we can estimate the calendar date t corresponding to the smallest date such that the number of events observed before t is equal to k.

The occurrence() estimates these occurrences and gives the credible interval or the highest posterior density (HPD) region for a given level of confidence.

Value

An OccurrenceEvents object.

An OccurrenceEvents object.

Author(s)

A. Philippe, M.-A. Vibet, T. S. Dye, N. Frerebeau

See Also

Other event tools: activity(), elapse(), tempo()

Examples

```
## Coerce to MCMC
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)
eve <- eve[1:10000, ]

## Occurrence plot
occ <- occurrence(eve)
plot(occ, panel.first = graphics::grid())
```

---

older *Bayesian Test for Anteriority/Posteriority*

Description

A Bayesian test for checking the following assumption: "event x is older than event y".

Usage

```
older(x, y, ...)

## S4 method for signature 'numeric,numeric'
older(x, y)

## S4 method for signature 'EventsMCMC,missing'
older(x, y)
```

Arguments

x	A numeric vector giving the output of the MCMC algorithm for the first parameter, or an EventsMCMC object.
y	A numeric vector giving the output of the MCMC algorithm for the second parameter.
...	Currently not used.

Details

For a given output of MCMC algorithm, this function estimates the posterior probability of the event $x < y$ by the relative frequency of the event "the value of event x is less than the value of event y" in the simulated Markov chain.

Methods (by class)

- `older(x = numeric, y = numeric)`: Returns a length-one `numeric` vector (the posterior probability of the assumption: "event x is older than event y").
- `older(x = EventsMCMC, y = missing)`: Returns a `numeric` matrix of posterior probabilities.

Author(s)

A. Philippe, M.-A. Vibet, N. Frerebeau

Examples

```
## Coerce to MCMC
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)
eve <- eve[1:10000, ]

## Test for anteriority
older(eve)

## Test for hiatus
hia <- hiatus(eve)
as.data.frame(hia)
```

---

phases *Compute Phases*

Description

Constructs the minimum and maximum for a group of events (phase).

Usage

```
phases(x, groups, ...)

## S4 method for signature 'EventsMCMC,missing'
phases(x)

## S4 method for signature 'EventsMCMC,list'
phases(x, groups)
```

Arguments

x	An EventsMCMC.
groups	A list of (named) vector of names or indexes of columns in x (see phases()).
...	Currently not used.

Value

A PhasesMCMC object.

Note

The default value of start or end corresponds to a CSV file exported from ChronoModel.

Author(s)

A. Philippe, M.-A. Vibet, N. Frerebeau

See Also

Other phase tools: duration()

Examples

```
## Coerce to phases
(pha <- as_phases(mcmc_phases, calendar = CE(), start = c(1, 3), iteration = 1))
summary(pha, calendar = CE())

## Plot phases
plot(pha)
plot(pha, succession = "hiatus")
plot(pha, succession = "transition")

## Compute phases from events
(eve <- as_events(mcmc_events, calendar = CE(), iteration = 1))

## Compute min-max range for all chains
pha1 <- phases(eve)
summary(pha1, calendar = CE())

## Compute min-max range by group
pha2 <- phases(eve, groups = list(phase_1 = c(1, 3), phase_2 = c(2, 4)))
summary(pha2, calendar = CE())
```

plot_events *Plot Events*

Description

Plots credible intervals or HPD regions of a series of events.

Usage

```
## S4 method for signature 'MCMC,missing'
plot(
  x,
  calendar = get_calendar(),
  density = TRUE,
  interval = NULL,
  level = 0.95,
  sort = TRUE,
  decreasing = TRUE,
  main = NULL,
  sub = NULL,
  ann = graphics::par("ann"),
  axes = TRUE,
  frame.plot = FALSE,
  panel.first = NULL,
  panel.last = NULL,
  col.density = "grey",
  col.interval = "#77AADD",
  ...
)
```

Arguments

x	An MCMC object.
calendar	A aion::TimeScale object specifying the target calendar (see calendar()).
density	A logical scalar: should estimated density be plotted?
interval	A character string specifying the confidence interval to be drawn. It must be one of "credible" (credible interval) or "hdr" (highest posterior density interval). Any unambiguous substring can be given. If NULL (the default) no interval is computed.
level	A length-one numeric vector giving the confidence level.
sort	A logical scalar: should the data be sorted?
decreasing	A logical scalar: should the sort order be decreasing? Only used if sort is TRUE.
main	A character string giving a main title for the plot.
sub	A character string giving a subtitle for the plot.

ann	A logical scalar: should the default annotation (title and x and y axis labels) appear on the plot?
axes	A logical scalar: should axes be drawn on the plot?
frame.plot	A logical scalar: should a box be drawn around the plot?
panel.first	An an expression to be evaluated after the plot axes are set up but before any plotting takes place. This can be useful for drawing background grids.
panel.last	An expression to be evaluated after plotting has taken place but before the axes, title and box are added.
col.density, col.interval	A specification for the plotting colors.
...	Extra parameters to be passed to stats::density().

Value

plot() is called it for its side-effects: it results in a graphic being displayed (invisibly returns x).

Author(s)

A. Philippe, M.-A. Vibet, T. S. Dye, N. Frerebeau

See Also

stats::density()

Other plot methods: plot_phases

Examples

```
## Coerce to MCMC
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)

## Summary
summary(eve, calendar = CE())
summary(eve, calendar = BP())

## Plot events
plot(eve, calendar = CE(), interval = "credible", level = 0.68)
plot(eve, calendar = BP(), interval = "hdr", level = 0.68)
plot(eve[, 1], interval = "hdr")

## Compute phases
pha <- phases(eve, groups = list(B = c(2, 4), A = c(1, 3)))

## Summary
summary(pha, calendar = CE())
summary(pha, calendar = BP())

## Plot phases
plot(pha, calendar = BP())
plot(pha, succession = "hiatus")
plot(pha, succession = "transition")
```

plot_phases *Plot Phases*

Description

Plots the characteristics of a group of events (phase).

Usage

```
## S4 method for signature 'PhasesMCMC,missing'
plot(
  x,
  calendar = get_calendar(),
  density = TRUE,
  range = TRUE,
  succession = NULL,
  level = 0.95,
  sort = TRUE,
  decreasing = TRUE,
  legend = TRUE,
  main = NULL,
  sub = NULL,
  ann = graphics::par("ann"),
  axes = TRUE,
  frame.plot = FALSE,
  panel.first = NULL,
  panel.last = NULL,
  col.density = "grey",
  col.range = "black",
  col.succession = c("#77AADD", "#EE8866"),
  ...
)
```

Arguments

x	A PhasesMCMC object.
calendar	A aion::TimeScale object specifying the target calendar (see calendar()).
density	A logical scalar: should estimated density be plotted?
range	A logical scalar: should phase time range be plotted (see boundaries())?
succession	A character string specifying the additional time range to be displayed. It must be one of "hiatus" or "transition". If NULL (the default), no additional time ranges are displayed.
level	A length-one numeric vector giving the confidence level.
sort	A logical scalar: should the data be sorted?

decreasing	A logical scalar: should the sort order be decreasing? Only used if sort is TRUE.
legend	A logical scalar: should a legend be displayed?
main	A character string giving a main title for the plot.
sub	A character string giving a subtitle for the plot.
ann	A logical scalar: should the default annotation (title and x and y axis labels) appear on the plot?
axes	A logical scalar: should axes be drawn on the plot?
frame.plot	A logical scalar: should a box be drawn around the plot?
panel.first	An an expression to be evaluated after the plot axes are set up but before any plotting takes place. This can be useful for drawing background grids.
panel.last	An expression to be evaluated after plotting has taken place but before the axes, title and box are added.
col.density, col.range, col.succession	A specification for the plotting colors.
...	Extra parameters to be passed to stats::density().

Value

plot() is called it for its side-effects: it results in a graphic being displayed (invisibly returns x).

Author(s)

A. Philippe, M.-A. Vibet, T. S. Dye, N. Frerebeau

See Also

stats::density()

Other plot methods: plot_events

Examples

```
## Coerce to MCMC
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)

## Summary
summary(eve, calendar = CE())
summary(eve, calendar = BP())

## Plot events
plot(eve, calendar = CE(), interval = "credible", level = 0.68)
plot(eve, calendar = BP(), interval = "hdr", level = 0.68)
plot(eve[, 1], interval = "hdr")

## Compute phases
pha <- phases(eve, groups = list(B = c(2, 4), A = c(1, 3)))
```

```
## Summary
summary(pha, calendar = CE())
summary(pha, calendar = BP())

## Plot phases
plot(pha, calendar = BP())
plot(pha, succession = "hiatus")
plot(pha, succession = "transition")
```

`read_bcal` *Read BCal Output*

Description

Reads MCMC output.

Usage

```
read_bcal(file, ...)

## S4 method for signature 'character'
read_bcal(file, bin_width = 1, calendar = BP())
```

Arguments

`file`	A `character` string giving the name of the CSV file which the data are to be read from.
`...`	Currently not used.
`bin_width`	The bin width specified for the BCal calibration. Defaults to the BCal default of 1.
`calendar`	A `aion::TimeScale` object specifying the calendar (see `calendar()`). It should be `BP()` unless you change the default settings in 'BCal'.

Value

An `EventsMCMC` object.

Author(s)

T. S. Dye, N. Frerebeau

References

Buck C. E., Christen J. A. & James G. N. (1999). BCal: an on-line Bayesian radiocarbon calibration tool. *Internet Archaeology*, 7. doi:10.11141/ia.7.1.

See Also

utils::read.table()

Other read methods: as_coda(), as_events(), as_phases(), check, read_chronomodel, read_oxcal()

Examples

```
if (requireNamespace("ArchaeoData", quietly = TRUE)) {
  ## Construct the path to the data
  path_output <- system.file("bcal", "fishpond.csv", package = "ArchaeoData")

  ## Import BCal Output
  (bcal <- read_bcal(path_output))
}
```

---

read_chronomodel *Read ChronoModel Output*

Description

Reads MCMC output.

Usage

```
read_chronomodel_events(file, ...)

read_chronomodel_phases(file, ...)

## S4 method for signature 'character'
read_chronomodel_events(file, calendar = CE(), sep = ",", dec = ".")

## S4 method for signature 'character'
read_chronomodel_phases(file, calendar = CE(), sep = ",", dec = ".")
```

Arguments

file	A character string giving the name of the CSV file which the data are to be read from.
...	Currently not used.
calendar	A aion::TimeScale object specifying the calendar (see calendar()). It should be CE() unless you change the default settings in 'ChronoModel'.
sep	A character string specifying the field separator character (see utils::read.table()).
dec	A character string specifying the character used in the file for decimal points (see utils::read.table()).

Value

An EventsMCMC or a PhasesMCMC object.

Author(s)

T. S. Dye, N. Frerebeau

References

Lanos, Ph., Philippe, A. & Dufresne, Ph. (2015). Chronomodel: Chronological Modeling of Archaeological Data using Bayesian Statistics. URL: https://chronomodel.com/.

See Also

utils::read.table()

Other read methods: as_coda(), as_events(), as_phases(), check, read_bcal(), read_oxcal()

Examples

```
if (requireNamespace("ArchaeoData", quietly = TRUE)) {
  ## Construct the paths to the data
  path <- file.path("chronomodel", "ksarakil")
  path_events <- system.file(path, "Chain_all_Events.csv", package = "ArchaeoData")
  path_phases <- system.file(path, "Chain_all_Phases.csv", package = "ArchaeoData")

  ## Import ChronoModel events
  (chrono_events <- read_chronomodel_events(path_events))

  ## Import ChronoModel phases
  (chrono_phases <- read_chronomodel_phases(path_phases))
}
```

---

read_oxcal *Read OxCal Output*

Description

Reads MCMC output.

Usage

```
read_oxcal(file, ...)

## S4 method for signature 'character'
read_oxcal(file, calendar = CE())
```

Arguments

file	A character string giving the name of the CSV file which the data are to be read from.
...	Currently not used.
calendar	A aion::TimeScale object specifying the calendar (see calendar()). It should be CE() unless you change the default settings in 'OxCal'.

Value

An EventsMCMC object.

Author(s)

T. S. Dye, N. Frerebeau

References

Bronk Ramsey, C. (2009). Bayesian Analysis of Radiocarbon Dates. *Radiocarbon*, 51(1), 337-360. doi:10.1017/S0033822200033865.

See Also

utils::read.table()

Other read methods: as_coda(), as_events(), as_phases(), check, read_bcal(), read_chronomodel

Examples

```
if (requireNamespace("ArchaeoData", quietly = TRUE)) {
  ## Construct the path to the data
  path <- file.path("oxcal", "ksarakil")
  path_output <- system.file(path, "MCMC_Sample.csv", package = "ArchaeoData")

  ## Import OxCal Output
  (oxcal <- read_oxcal(path_output))
}
```

---

sensitivity *Sensitivity*

Description

Calculates the ranges of summary statistics from the output of two or more runs of the MCMC algorithm.

Usage

```
sensitivity(...)

## S4 method for signature 'EventsMCMC'
sensitivity(..., positions = NULL, level = 0.95)
```

Arguments

...	Any EventsMCMC object.
positions	A numeric vector specifying the positions of the columns corresponding to the MCMC chains of interest, or a character vector of column names.
level	A length-one numeric vector giving the confidence level.

Details

This function is useful for estimating the sensitivity of calibration results to different model parameters.

Value

A data.frame.

Author(s)

T. S. Dye, N. Frerebeau

See Also

summary()

Other statistics: interval_credible(), interval_hdr(), summary()

Examples

```
## Coerce to MCMC
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)

## Returns 0's
sensitivity(eve, eve)
```

---

sort *Sort an MCMC Object*

Description

Sort (or order) an object into ascending or descending temporal order.

Usage

```
## S4 method for signature 'MCMC'
sort(x, decreasing = FALSE)

## S4 method for signature 'PhasesMCMC'
sort(x, decreasing = FALSE)
```

Arguments

x	An MCMC object.
decreasing	A logical scalar: should the sort order be decreasing?

Value

An object of the same sort as x.

Author(s)

N. Frerebeau

See Also

Other mutators: names, sort.list()

Examples

```
## Events
(eve <- as_events(mcmc_events, calendar = CE(), iteration = 1))

eve[1:1000, ] # Select the first 1000 iterations
eve[, 1:2]    # Select the first 2 events

cbind2(eve[, 1:2], eve[, 3:4]) # Combine two MCMC objects
sort(eve, decreasing = TRUE)   # Sort events in descending order

## Phases
(pha <- as_phases(mcmc_phases, start = c(1, 3), calendar = CE(), iteration = 1))

pha[1:1000, , ]          # Select the first 1000 iterations
pha[, 1, , drop = FALSE] # Select the first phase
```

---

sort.list *Ordering Permutation of an MCMC Object*

Description

Returns a permutation which rearranges an object into ascending or descending temporal order.

Usage

```
## S4 method for signature 'MCMC'
sort.list(x, decreasing = FALSE)

## S4 method for signature 'PhasesMCMC'
sort.list(x, decreasing = FALSE)
```

Arguments

x	An MCMC object.
decreasing	A logical scalar: should the sort order be decreasing?

Value

An integer vector.

Author(s)

N. Frerebeau

See Also

Other mutators: names, sort()

Examples

```
## Events
(eve <- as_events(mcmc_events, calendar = CE(), iteration = 1))

eve[1:1000, ] # Select the first 1000 iterations
eve[, 1:2]    # Select the first 2 events

cbind2(eve[, 1:2], eve[, 3:4]) # Combine two MCMC objects
sort(eve, decreasing = TRUE)   # Sort events in descending order

## Phases
(pha <- as_phases(mcmc_phases, start = c(1, 3), calendar = CE(), iteration = 1))

pha[1:1000, , ]          # Select the first 1000 iterations
pha[, 1, , drop = FALSE] # Select the first phase
```

---

subset *Extract or Replace Parts of an Object*

Description

Operators acting on objects to extract or replace parts.

Usage

```
## S4 method for signature 'MCMC'
x[i, j, ..., drop = FALSE]

## S4 method for signature 'PhasesMCMC'
x[i, j, k, drop = FALSE]
```

Arguments

x	An object from which to extract element(s) or in which to replace element(s).
i, j, k	Indices specifying elements to extract or replace.
...	Currently not used.
drop	A logical scalar: should the result be coerced to the lowest possible dimension? This only works for extracting elements, not for the replacement.

Value

A subsetted object.

Author(s)

N. Frerebeau

See Also

Other subsetting methods: `bind`

Examples

```
## Events
(eve <- as_events(mcmc_events, calendar = CE(), iteration = 1))

eve[1:1000, ] # Select the first 1000 iterations
eve[, 1:2]    # Select the first 2 events

cbind2(eve[, 1:2], eve[, 3:4]) # Combine two MCMC objects
sort(eve, decreasing = TRUE)   # Sort events in descending order

## Phases
(pha <- as_phases(mcmc_phases, start = c(1, 3), calendar = CE(), iteration = 1))

pha[1:1000, , ]          # Select the first 1000 iterations
pha[, 1, , drop = FALSE] # Select the first phase
```

---

summary *Marginal Summary Statistics for Multiple MCMC Chains*

Description

Calculates summary statistics of the output of the MCMC algorithm for multiple parameters. Results are given in calendar years (BC/AD).

Usage

```
## S4 method for signature 'MCMC'
summary(object, level = 0.95, calendar = get_calendar())

## S4 method for signature 'PhasesMCMC'
summary(object, level = 0.95, calendar = get_calendar())
```

Arguments

`object`	An `MCMC` or a `PhasesMCMC` object.
`level`	A length-one `numeric` vector giving the confidence level.
`calendar`	A `aion::TimeScale` object specifying the target calendar (see `calendar()`).

Value

A data.frame where the rows correspond to the chains of interest and columns to the following statistics:

mean The mean of the MCMC chain.

sd The standard deviation of the MCMC chain.

min Minimum value of the MCMC chain.

q1 First quantile of the MCMC chain.

median Median of the MCMC chain.

q3 Third quantile of the MCMC chain.

max Maximum value of the MCMC chain.

lower Lower boundary of the credible interval of the MCMC chain at `level`.

upper Upper boundary of the credible interval of the MCMC chain at `level`.

Author(s)

A. Philippe, M.-A. Vibet, T. S. Dye, N. Frerebeau

See Also

Other statistics: `interval_credible()`, `interval_hdr()`, `sensitivity()`

Examples

```
## Coerce to MCMC
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)

## Summary
summary(eve, calendar = CE())
summary(eve, calendar = BP())

## Plot events
plot(eve, calendar = CE(), interval = "credible", level = 0.68)
plot(eve, calendar = BP(), interval = "hdr", level = 0.68)
plot(eve[, 1], interval = "hdr")

## Compute phases
pha <- phases(eve, groups = list(B = c(2, 4), A = c(1, 3)))

## Summary
summary(pha, calendar = CE())
summary(pha, calendar = BP())

## Plot phases
plot(pha, calendar = BP())
plot(pha, succession = "hiatus")
plot(pha, succession = "transition")
```

tempo *Tempo Plot*

Description

A statistical graphic designed for the archaeological study of rhythms of the long term that embodies a theory of archaeological evidence for the occurrence of events.

Usage

```
tempo(object, ...)

## S4 method for signature 'CumulativeEvents,missing'
plot(
  x,
  calendar = get_calendar(),
  interval = c("credible", "gauss"),
  col.tempo = "#004488",
  col.interval = "grey",
  main = NULL,
  sub = NULL,
  ann = graphics::par("ann"),
  axes = TRUE,
  frame.plot = axes,
  panel.first = NULL,
  panel.last = NULL,
  ...
)

## S4 method for signature 'EventsMCMC'
tempo(
  object,
  level = 0.95,
  count = FALSE,
  credible = TRUE,
  gauss = TRUE,
  from = min(object),
  to = max(object),
  grid = getOption("ArchaeoPhases.grid")
)
```

Arguments

object	An EventsMCMC object.
...	Other graphical parameters may also be passed as arguments to this function.
x	A CumulativeEvents object or an EventsMCMC object.

calendar	A aion::TimeScale object specifying the target calendar (see calendar()).
interval	A character string specifying the confidence interval to be drawn. It must be one of "credible" (credible interval) or "gauss" (Gaussian approximation of the credible interval). Any unambiguous substring can be given.
col.tempo, col.interval	A specification for the plotting colors.
main	A character string giving a main title for the plot.
sub	A character string giving a subtitle for the plot.
ann	A logical scalar: should the default annotation (title and x and y axis labels) appear on the plot?
axes	A logical scalar: should axes be drawn on the plot?
frame.plot	A logical scalar: should a box be drawn around the plot?
panel.first	An an expression to be evaluated after the plot axes are set up but before any plotting takes place. This can be useful for drawing background grids.
panel.last	An expression to be evaluated after plotting has taken place but before the axes, title and box are added.
level	A length-one numeric vector giving the confidence level.
count	A logical scalar: should the counting process be a number or a probability (the default)?
credible	A logical scalar: should the credible interval be computed/displayed?
gauss	A logical scalar: should the Gaussian approximation of the credible interval be computed/displayed?
from	A length-one numeric vector giving the earliest date to estimate for (expressed in *rata die*).
to	A length-one numeric vector giving the latest date to estimate for (expressed in *rata die*).
grid	A length-one numeric vector specifying the number of equally spaced points of the temporal grid.

Details

The tempo plot is one way to measure change over time: it estimates the cumulative occurrence of archaeological events in a Bayesian calibration. The tempo plot yields a graphic where the slope of the plot directly reflects the pace of change: a period of rapid change yields a steep slope and a period of slow change yields a gentle slope. When there is no change, the plot is horizontal. When change is instantaneous, the plot is vertical.

Value

- tempo() returns an CumulativeEvents object.
- plot() is called it for its side-effects: it results in a graphic being displayed (invisibly returns x).

Author(s)

A. Philippe, M.-A. Vibet, T. S. Dye, N. Frerebeau

References

Dye, T. S. (2016). Long-term rhythms in the development of Hawaiian social stratification. *Journal of Archaeological Science*, 71: 1-9. doi:10.1016/j.jas.2016.05.006.

See Also

Other event tools: `activity()`, `elapse()`, `occurrence()`

Examples

```
## Coerce to MCMC
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)
eve <- eve[1:10000, ]

## Tempo plot
tmp <- tempo(eve)
plot(tmp, interval = "credible", panel.first = grid())
plot(tmp, interval = "gauss", panel.first = grid())

## Activity plot
act <- activity(tmp)
plot(act, panel.first = grid())
```

---

transition *Transition Range Between Successive Phases*

Description

Estimates the transition endpoints between two phases.

Usage

```
transition(x, y, ...)

## S4 method for signature 'numeric,numeric'
transition(x, y, level = 0.95)

## S4 method for signature 'PhasesMCMC,missing'
transition(x, level = 0.95)
```

Arguments

x, y	A `numeric` vector. If y is missing, x must be an `PhasesMCMC` object.
...	Currently not used.
level	A length-one `numeric` vector giving the confidence level.

Details

The transition is the shortest interval that satisfies $P(IntervalInf < Phase1Max < Phase2Min < IntervalSup|M) = level$.

This assumes that the phases are in temporal order constraint.

Value

The endpoints of the transition interval for each pair of successive phases (at a given `level`).

Methods (by class)

- `transition(x = numeric, y = numeric)`: Returns a length-two `numeric` vector (terminal times of the transition interval).
- `transition(x = PhasesMCMC, y = missing)`: Returns a `TimeRange` object.

Author(s)

A. Philippe, M.-A. Vibet, N. Frerebeau

See Also

Other time ranges: `boundaries()`, `hiatus()`

Examples

```
## Coerce to events
eve <- as_events(mcmc_events, calendar = CE(), iteration = 1)
eve <- eve[1:10000, ]

## Compute min-max range by group
pha <- phases(eve, groups = list(A = c(1, 3), B = c(2, 4)))

## Compute phase ranges
bou <- boundaries(pha)
as.data.frame(bou)

## Compute phase transition
tra <- transition(pha)
as.data.frame(tra)

## Compute phase hiatus
hia <- hiatus(pha)
as.data.frame(hia)
```

Index

∗ **Allen's intervals**
allen_analyze, 5
allen_complement, 6
allen_composition, 7
allen_converse, 9
allen_illustrate, 10
allen_intersect, 12
allen_joint_concurrency, 13
allen_observe, 14
allen_observe_frequency, 16
allen_relation, 17
allen_relation_code, 19
allen_union, 20
∗ **age-depth modeling tools**
bury, 28
interpolate, 36
∗ **coercion methods**
data.frame, 32
∗ **datasets**
mcmc_events, 39
mcmc_phases, 40
∗ **event tools**
activity, 3
elapse, 33
occurrence, 41
tempo, 58
∗ **mutators**
names, 40
sort, 53
sort.list, 54
∗ **phase tools**
duration, 32
phases, 43
∗ **plot methods**
plot_events, 45
plot_phases, 47
∗ **read methods**
as_coda, 22
as_events, 23
as_phases, 24
check, 30
read_bcal, 49
read_chronomodel, 50
read_oxcal, 51
∗ **statistics**
interval_credible, 37
interval_hdr, 38
sensitivity, 52
summary, 56
∗ **subsetting methods**
bind, 26
subset, 55
∗ **tests**
older, 42
∗ **time ranges**
boundaries, 27
hiatus, 34
transition, 60
[,MCMC-method *(*subset*)*, 55
[,PhasesMCMC-method *(*subset*)*, 55

activity, 3
activity(), *34*, *42*, *60*
activity,CumulativeEvents-method *(*activity*)*, 3
activity,EventsMCMC-method *(*activity*)*, 3
activity-method *(*activity*)*, 3
ActivityEvents, *4*
AgeDepthModel, *29*
aion::TimeScale, *4*, *23*, *25*, *29*, *32*, *37*, *38*, *45*, *47*, *49–51*, *56*, *59*
allen_analyze, 5
allen_analyze(), *7–9*, *11*, *12*, *14–16*, *19–21*
allen_analyze_relations(), *6–9*, *11*, *12*, *14–16*, *19–21*
allen_complement, 6
allen_complement(), *6*, *8*, *9*, *11*, *12*, *14–16*, *19–21*

allen_complement,character-method (allen_complement), 6
allen_complement,matrix-method (allen_complement), 6
allen_complement-method (allen_complement), 6
allen_composition, 7
allen_composition(), *6, 7*, *9*, *11, 12*, *14–16*, *19–21*
allen_composition,character,character-method (allen_composition), 7
allen_composition-method (allen_composition), 7
allen_converse, 9
allen_converse(), *6–8*, *11, 12*, *14–16*, *19–21*
allen_converse,character-method (allen_converse), 9
allen_converse,matrix-method (allen_converse), 9
allen_converse-method (allen_converse), 9
allen_illustrate, 10
allen_illustrate(), *6–9*, *12*, *14–16*, *19–21*
allen_illustrate_relations(), *6–9*, *11, 12*, *14–16*, *19–21*
allen_intersect, 12
allen_intersect(), *6–9*, *11*, *14–16*, *19–21*
allen_intersect,character,character-method (allen_intersect), 12
allen_intersect-method (allen_intersect), 12
allen_joint_concurrency, 13
allen_joint_concurrency(), *6–9*, *11, 12*, *15, 16*, *19–21*
allen_joint_concurrency,EventsMCMC,list-method (allen_joint_concurrency), 13
allen_joint_concurrency-method (allen_joint_concurrency), 13
allen_observe, 14
allen_observe(), *6–9*, *11, 12*, *14*, *16*, *19–21*
allen_observe,EventsMCMC,list-method (allen_observe), 14
allen_observe,PhasesMCMC,missing-method (allen_observe), 14
allen_observe-method (allen_observe), 14
allen_observe_frequency, 16
allen_observe_frequency(), *6–9*, *11, 12*, *14, 15*, *19–21*
allen_observe_frequency,EventsMCMC,list-method (allen_observe_frequency), 16
allen_observe_frequency,PhasesMCMC,missing-method (allen_observe_frequency), 16
allen_observe_frequency-method (allen_observe_frequency), 16
allen_plot(), *6–9*, *11, 12*, *14–16*, *19–21*
allen_relation, 17
allen_relation(), *6–9*, *11, 12*, *14–16*, *20, 21*
allen_relation,ANY,missing-method (allen_relation), 17
allen_relation,numeric,numeric-method (allen_relation), 17
allen_relation-method (allen_relation), 17
allen_relation_code, 19
allen_relation_code(), *6–9*, *11, 12*, *14–16*, *19*, *21*
allen_relation_concurrent (allen_relation_code), 19
allen_relation_distinct (allen_relation_code), 19
allen_relation_string (allen_relation_code), 19
allen_union, 20
allen_union(), *6–9*, *11, 12*, *14–16*, *19, 20*
allen_union,character,character-method (allen_union), 20
allen_union-method (allen_union), 20
arkhe::interval_credible(), *38*
arkhe::interval_hdr(), *39*
as.data.frame,ActivityEvents-method (data.frame), 32
as.data.frame,CumulativeEvents-method (data.frame), 32
as.data.frame,OccurrenceEvents-method (data.frame), 32
as.data.frame,TimeRange-method (data.frame), 32
as_coda, 22
as_coda(), *24, 25*, *31*, *50–52*
as_coda,MCMC-method (as_coda), 22
as_coda-method (as_coda), 22
as_events, 23
as_events(), *22*, *25*, *31*, *50–52*
as_events,data.frame-method (as_events), 23

as_events,matrix-method *(as_events)*, 23
as_events-method *(as_events)*, 23
as_phases, 24
as_phases(), *22*, *24*, *31*, *50–52*
as_phases,data.frame-method *(as_phases)*, 24
as_phases,matrix-method *(as_phases)*, 24
as_phases-method *(as_phases)*, 24

bind, 26, *56*
boundaries, 27
boundaries(), *35*, *47*, *61*
boundaries,numeric,numeric-method *(boundaries)*, 27
boundaries,PhasesMCMC,missing-method *(boundaries)*, 27
boundaries-method *(boundaries)*, 27
BP(), *49*
bury, 28
bury(), *36*
bury,AgeDepthModel-method *(bury)*, 28
bury,EventsMCMC,numeric-method *(bury)*, 28
bury,EventsMCMC-method *(bury)*, 28
bury-method *(bury)*, 28

calendar(), *4*, *23*, *25*, *29*, *32*, *37*, *38*, *45*, *47*, *49–51*, *56*, *59*
cbind2,MCMC,MCMC-method *(bind)*, 26
CE(), *50*, *51*
character, *4–6*, *8–10*, *12*, *16*, *18*, *20*, *21*, *25*, *29*, *45*, *47–52*, *59*
check, *22*, *24*, *25*, 30, *50–52*
coda::mcmc(), *22*
coda::mcmc.list, *22*
coda::mcmc.list(), *22*
credible interval, *57*
CumulativeEvents, *4*, *58*, *59*

data.frame, 32, *32*, *39*, *40*, *53*, *57*
data.frame(), *32*
digest::digest(), *31*
duration, 32
duration(), *44*
duration,numeric,numeric-method *(duration)*, 32
duration,PhasesMCMC,missing-method *(duration)*, 32
duration-method *(duration)*, 32

elapse, 33
elapse(), *5*, *42*, *60*
elapse,MCMC-method *(elapse)*, 33
elapse-method *(elapse)*, 33
EventsMCMC, *4*, *13*, *14*, *16*, *23*, *29*, *35*, *41*, *42*, *44*, *49*, *50*, *52*, *58*

graphical parameters, *4*, *29*, *58*
grDevices::xy.coords(), *18*

hiatus, 34
hiatus(), *27*, *61*
hiatus,EventsMCMC,missing-method *(hiatus)*, 34
hiatus,EventsMCMC-method *(hiatus)*, 34
hiatus,numeric,numeric-method *(hiatus)*, 34
hiatus,PhasesMCMC,missing-method *(hiatus)*, 34
hiatus-method *(hiatus)*, 34

integer, *22*, *25*, *34*, *36*, *54*
interpolate, 36
interpolate(), *30*
interpolate,EventsMCMC,missing-method *(interpolate)*, 36
interpolate,numeric,numeric-method *(interpolate)*, 36
interpolate-method *(interpolate)*, 36
interval_credible, 37
interval_credible(), *39*, *53*, *57*
interval_credible,MCMC-method *(interval_credible)*, 37
interval_credible-method *(interval_credible)*, 37
interval_hdr, 38
interval_hdr(), *38*, *53*, *57*
interval_hdr,MCMC,missing-method *(interval_hdr)*, 38
interval_hdr-method *(interval_hdr)*, 38
is_original *(check)*, 30
is_original,ActivityEvents-method *(check)*, 30
is_original,CumulativeEvents-method *(check)*, 30
is_original,MCMC-method *(check)*, 30
is_original,OccurrenceEvents-method *(check)*, 30

is_original,PhasesMCMC-method (check), 30
is_original-method (check), 30

list, *13, 14*, *16*, *37, 38*, *44*
logical, *4*, *14*, *29*, *31*, *45–48*, *53–55*, *59*

matrix, *16*, *37, 38*
MCMC, *22*, *26*, *31*, *34*, *37, 38*, *45*, *53, 54*, *56*
mcmc_events, 39, *40*
mcmc_phases, *39*, 40

names, 40, *54, 55*
names,MCMC-method (names), 40
names,PhasesMCMC-method (names), 40
names<-,MCMC-method (names), 40
names<-,PhasesMCMC-method (names), 40
numeric, *4*, *18*, *23*, *25*, *27*, *29*, *33*, *35–38*, *41–43*, *45*, *47*, *52*, *56*, *59–61*

occurrence, 41
occurrence(), *5*, *34*, *60*
occurrence,EventsMCMC-method (occurrence), 41
occurrence-method (occurrence), 41
OccurrenceEvents, *42*
older, 42
older,EventsMCMC,missing-method (older), 42
older,EventsMCMC-method (older), 42
older,numeric,numeric-method (older), 42
older-method (older), 42

phases, 43
phases(), *13, 14*, *16*, *33*, *44*
phases,EventsMCMC,list-method (phases), 43
phases,EventsMCMC,missing-method (phases), 43
phases-method (phases), 43
PhasesMCMC, *14*, *16*, *25*, *27*, *33*, *35*, *44*, *47*, *50*, *56*, *60*
plot,ActivityEvents,missing-method (activity), 3
plot,AgeDepthModel,missing-method (bury), 28
plot,CumulativeEvents,missing-method (tempo), 58
plot,MCMC,missing-method (plot_events), 45
plot,PhasesMCMC,missing-method (plot_phases), 47
plot_events, 45, *48*
plot_phases, *46*, 47
predict,AgeDepthModel-method (bury), 28

rata die, *4*, *59*
read_bcal, 49
read_bcal(), *22*, *24, 25*, *31*, *51, 52*
read_bcal,character-method (read_bcal), 49
read_bcal-method (read_bcal), 49
read_chronomodel, *22*, *24, 25*, *31*, 50, *50*, *52*
read_chronomodel-method (read_chronomodel), 50
read_chronomodel_events (read_chronomodel), 50
read_chronomodel_events,character-method (read_chronomodel), 50
read_chronomodel_events-method (read_chronomodel), 50
read_chronomodel_phases (read_chronomodel), 50
read_chronomodel_phases,character-method (read_chronomodel), 50
read_chronomodel_phases-method (read_chronomodel), 50
read_oxcal, 51
read_oxcal(), *22*, *24, 25*, *31*, *50, 51*
read_oxcal,character-method (read_oxcal), 51
read_oxcal-method (read_oxcal), 51

sensitivity, 52
sensitivity(), *38, 39*, *57*
sensitivity,EventsMCMC-method (sensitivity), 52
sensitivity-method (sensitivity), 52
sort, 53
sort(), *41*, *55*
sort,MCMC-method (sort), 53
sort,PhasesMCMC-method (sort), 53
sort.list, 54
sort.list(), *41*, *54*
sort.list,MCMC-method (sort.list), 54
sort.list,PhasesMCMC-method (sort.list), 54
stats::density(), *38, 39*, *46*, *48*
stats::loess(), *29*

subset, *26*, 55
summary, 56
summary(), *38, 39*, *53*
summary,MCMC-method *(*summary*)*, 56
summary,PhasesMCMC-method *(*summary*)*, 56

tempo, *3*, 58
tempo(), *5*, *34*, *42*
tempo,EventsMCMC-method *(*tempo*)*, 58
tempo-method *(*tempo*)*, 58
TimeRange, *27*, *35*, *61*
transition, 60
transition(), *27*, *35*
transition,numeric,numeric-method
*(*transition*)*, 60
transition,PhasesMCMC,missing-method
*(*transition*)*, 60
transition-method *(*transition*)*, 60

utils::read.table(), *50–52*