Package 'kairos'

February 26, 2024

Title Analysis of Chronological Patterns from Archaeological Count Data

Version 2.1.0

Description A toolkit for absolute and relative dating and analysis of chronological patterns. This package includes functions for chronological modeling and dating of archaeological assemblages from count data. It provides methods for matrix seriation. It also allows to compute time point estimates and density estimates of the occupation and duration of an archaeological site.

License GPL (>= 3)

URL https://packages.tesselle.org/kairos/, https://github.com/tesselle/kairos

BugReports https://github.com/tesselle/kairos/issues

Depends R (>= 3.5), dimensio (>= 0.6.0)

Imports aion (>= 1.0.2), arkhe (>= 1.6.0), extraDistr, grDevices, methods, stats, utils

Suggests folio (>= 1.4.0), knitr, markdown, rsvg, svglite, tabula (>= 3.0.1), tinysnapshot, tinytest

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.0

Collate 'AllClasses.R' 'AllGenerics.R' 'aoristic.R' 'apportion.R' 'coerce.R' 'event_date.R' 'event_model.R' 'event_plot.R' 'event_resample.R' 'fit.R' 'kairos-deprecated.R' 'kairos-internal.R' 'kairos-package.R' 'mcd.R' 'mutators.R' 'permute.R' 'plot_time.R' 'reexport.R' 'seriate_average.R' 'seriate_rank.R' 'seriate_refine.R' 'show.R' 'subset.R' 'summary.R' 'validate.R' 'zzz.R'

NeedsCompilation no

Author Nicolas Frerebeau [aut, cre] (<https://orcid.org/0000-0001-5759-4944>, Université Bordeaux Montaigne),
Brice Lebrun [ctb] (<https://orcid.org/0000-0001-7503-8685>, Université Bordeaux Montaigne),
Ben Marwick [ctb] (<https://orcid.org/0000-0001-7879-4531>, University of Washington),
Anne Philippe [ctb] (<https://orcid.org/0000-0002-5331-5087>, Université de Nantes),
Université Bordeaux Montaigne [fnd],
CNRS [fnd]

Maintainer Nicolas Frerebeau <nicolas.frerebeau@u-bordeaux-montaigne.fr>

Repository CRAN

Date/Publication 2024-02-26 17:10:02 UTC

R topics documented:

aoristic *Aoristic Analysis*

Description

Computes the aoristic sum.

Usage

```
aoristic(x, y, ...)

## S4 method for signature 'numeric,numeric'
aoristic(
  x,
  y,
  step = 1,
  start = min(x),
  end = max(y),
  calendar = CE(),
  weight = TRUE,
  groups = NULL
)

## S4 method for signature 'ANY,missing'
aoristic(
  x,
  step = 1,
  start = NULL,
  end = NULL,
  calendar = CE(),
  weight = TRUE,
  groups = NULL
)
```

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.
step	A length-one integer vector giving the step size, i.e. the width of each time step in the time series (defaults to 1, i.e. annual level).
start	A length-one numeric vector giving the beginning of the time window.
end	A length-one numeric vector giving the end of the time window.
calendar	A TimeScale object specifying the calendar of x and y (see calendar()). Defaults to Gregorian Common Era.

`weight` A `logical` scalar: should the aoristic sum be weighted by the length of periods (default). If `FALSE` the aoristic sum is the number of elements within a time block.

`groups` A `factor` vector in the sense that `as.factor(groups)` defines the grouping. If `x` is a `list` (or a `data.frame`), `groups` can be a length-one vector giving the index of the grouping component (column) of `x`.

Details

Aoristic analysis is used to determine the probability of contemporaneity of archaeological sites or assemblages. The aoristic analysis distributes the probability of an event uniformly over each temporal fraction of the period considered. The aoristic sum is then the distribution of the total number of events to be assumed within this period.

Muller and Hinz (2018) pointed out that the overlapping of temporal intervals related to period categorization and dating accuracy is likely to bias the analysis. They proposed a weighting method to overcome this problem. This method is not implemented here (for the moment), see the **aoristAAR** package.

Value

An `AoristicSum` object.

Author(s)

N. Frerebeau

References

Crema, E. R. (2012). Modelling Temporal Uncertainty in Archaeological Analysis. *Journal of Archaeological Method and Theory*, 19(3): 440-61. doi:10.1007/s1081601191223.

Johnson, I. (2004). Aoristic Analysis: Seeds of a New Approach to Mapping Archaeological Distributions through Time. *In* Ausserer, K. F., Börner, W., Goriany, M. & Karlhuber-Vöckl, L. (ed.), *Enter the Past - The E-Way into the Four Dimensions of Cultural Heritage*, Oxford: Archaeopress, p. 448-52. BAR International Series 1227. doi:10.15496/publikation2085

Müller-Scheeßel, N. & Hinz, M. (2018). *Aoristic Research in R: Correcting Temporal Categorizations in Archaeology*. Presented at the Human History and Digital Future (CAA 2018), Tubingen, March 21. https://www.youtube.com/watch?v=bUBukex30QI.

Palmisano, A., Bevan, A. & Shennan, S. (2017). Comparing Archaeological Proxies for Long-Term Population Patterns: An Example from Central Italy. *Journal of Archaeological Science*, 87: 59-72. doi:10.1016/j.jas.2017.10.001.

Ratcliffe, J. H. (2000). Aoristic Analysis: The Spatial Interpretation of Unspecific Temporal Events. *International Journal of Geographical Information Science*, 14(7): 669-79. doi:10.1080/136588100424963.

Ratcliffe, J. H. (2002). Aoristic Signatures and the Spatio-Temporal Analysis of High Volume Crime Patterns. *Journal of Quantitative Criminology*, 18(1): 23-43. doi:10.1023/A:1013240828824.

See Also

roc(), plot()

Other chronological analysis: apportion(), fit(), roc()

Examples

```
## Data from Husi 2022
data("loire", package = "folio")

## Get time range
loire_range <- loire[, c("lower", "upper")]

## Calculate aoristic sum (normal)
aorist_raw <- aoristic(loire_range, step = 50, weight = FALSE)
plot(aorist_raw, col = "grey")

## Calculate aoristic sum (weights)
aorist_weighted <- aoristic(loire_range, step = 50, weight = TRUE)
plot(aorist_weighted, col = "grey")

## Calculate aoristic sum (weights) by group
aorist_groups <- aoristic(loire_range, step = 50, weight = TRUE,
                          groups = loire$area)
plot(aorist_groups, flip = TRUE, col = "grey")
image(aorist_groups)

## Rate of change
roc_weighted <- roc(aorist_weighted, n = 30)
plot(roc_weighted)

## Rate of change by group
roc_groups <- roc(aorist_groups, n = 30)
plot(roc_groups, flip = TRUE)
```

---

apportion *Chronological Apportioning*

Description

Chronological Apportioning

Usage

```
apportion(object, ...)

## S4 method for signature 'data.frame'
apportion(
  object,
```

```
  s0,
  s1,
  t0,
  t1,
  from = min(s0),
  to = max(s1),
  step = 25,
  method = c("uniform", "truncated"),
  z = 2,
  progress = getOption("kairos.progress")
)

## S4 method for signature 'matrix'
apportion(
  object,
  s0,
  s1,
  t0,
  t1,
  from = min(s0),
  to = max(s1),
  step = 25,
  method = c("uniform", "truncated"),
  z = 2,
  progress = getOption("kairos.progress")
)
```

Arguments

`object`	A $m \times p$ `numeric` `matrix` or `data.frame` of count data (absolute frequencies giving the number of individuals for each category, i.e. a contingency table). A `data.frame` will be coerced to a `numeric` `matrix` via `data.matrix()`.
`...`	Currently not used.
`s0`	A length-m `numeric` vector giving the site beginning dates expressed in CE years (BCE years must be given as negative numbers).
`s1`	A length-m `numeric` vector giving the site end dates expressed in CE years (BCE years must be given as negative numbers).
`t0`	A length-p `numeric` vector giving the type beginning dates expressed in CE years (BCE years must be given as negative numbers).
`t1`	A length-p `numeric` vector giving the type end dates expressed in CE years (BCE years must be given as negative numbers).
`from`	A length-one `numeric` vector giving the beginning of the period of interest (in years CE).
`to`	A length-one `numeric` vector giving the end of the period of interest (in years CE).
`step`	A length-one `integer` vector giving the step size, i.e. the width of each time step for apportioning (in years CE; defaults to 25).

method	A character string specifying the distribution to be used (type popularity curve). It must be one of "uniform" (uniform distribution) or "truncated" (truncated standard normal distribution). Any unambiguous substring can be given.
z	An integer value giving the lower and upper truncation points (defaults to 2). Only used if method is "truncated".
progress	A logical scalar: should a progress bar be displayed?

Value

A CountApportion object.

Author(s)

N. Frerebeau

References

Roberts, J. M., Mills, B. J., Clark, J. J., Haas, W. R., Huntley, D. L. & Trowbridge, M. A. (2012). A Method for Chronological Apportioning of Ceramic Assemblages. *Journal of Archaeological Science*, 39(5): 1513-20. doi:10.1016/j.jas.2011.12.022.

See Also

Other chronological analysis: aoristic(), fit(), roc()

---

data.frame *Coerce to a Data Frame*

Description

Coerce to a Data Frame

Usage

```
## S4 method for signature 'MeanDate'
as.data.frame(x, ..., calendar = getOption("kairos.calendar"))

## S4 method for signature 'AoristicSum'
as.data.frame(x, ..., calendar = getOption("kairos.calendar"))

## S4 method for signature 'IncrementTest'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

x	An object.
...	Further parameters to be passed to data.frame().
calendar	A TimeScale object specifying the target calendar (see calendar()). If NULL, *rata die* are returned.
row.names, optional	Currently not used.

Value

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

Author(s)

N. Frerebeau

See Also

Other mutators: model, mutators, series(), subset()

---

event *Event and Accumulation Dates*

Description

Fits a date event model.

Usage

```
event(object, dates, ...)

## S4 method for signature 'data.frame,numeric'
event(object, dates, rank = NULL, sup_row = NULL, calendar = CE(), ...)

## S4 method for signature 'matrix,numeric'
event(object, dates, rank = NULL, sup_row = NULL, calendar = CE(), ...)

## S4 method for signature 'EventDate'
summary(object, ...)
```

Arguments

object	A $m \times p$ numeric matrix or data.frame of count data (absolute frequencies giving the number of individuals for each category, i.e. a contingency table). A data.frame will be coerced to a numeric matrix via data.matrix().
dates	A numeric vector of dates. If named, the names must match the row names of object.
...	Further arguments to be passed to internal methods.
rank	An integer specifying the number of CA factorial components to be use for linear model fitting (see details). If NULL (the default), axes corresponding to at least 60% of the inertia will be used.
sup_row	A numeric or logical vector specifying the indices of the supplementary rows.
calendar	A TimeScale object specifying the calendar of dates (see calendar()). Defaults to Gregorian Common Era.

Details

This is an implementation of the chronological modeling method proposed by Bellanger and Husi (2012, 2013).

Event and accumulation dates are density estimates of the occupation and duration of an archaeological site (Bellanger and Husi 2012, 2013). The event date is an estimation of the *terminus post-quem* of an archaeological assemblage. The accumulation date represents the "chronological profile" of the assemblage. According to Bellanger and Husi (2012), accumulation date can be interpreted "at best [...] as a formation process reflecting the duration or succession of events on the scale of archaeological time, and at worst, as imprecise dating due to contamination of the context by residual or intrusive material." In other words, accumulation dates estimate occurrence of archaeological events and rhythms of the long term.

Dates are converted to *rata die* before any computation.

This method relies on strong archaeological and statistical assumptions (see vignette("event")).

Value

An EventDate object.

Author(s)

N. Frerebeau

References

Bellanger, L. & Husi, P. (2013). Mesurer et modéliser le temps inscrit dans la matière à partir d'une source matérielle : la céramique médiévale. In *Mesure et Histoire Médiévale*. Histoire ancienne et médiévale. Paris: Publication de la Sorbonne, p. 119-134.

Bellanger, L. & Husi, P. (2012). Statistical Tool for Dating and Interpreting Archaeological Contexts Using Pottery. *Journal of Archaeological Science*, 39(4), 777-790. doi:10.1016/j.jas.2011.06.031.

Bellanger, L., Tomassone, R. & Husi, P. (2008). A Statistical Approach for Dating Archaeological Contexts. *Journal of Data Science*, 6, 135-154.

Bellanger, L., Husi, P. & Tomassone, R. (2006). Une approche statistique pour la datation de contextes archéologiques. *Revue de Statistique Appliquée*, 54(2), 65-81.

Bellanger, L., Husi, P. & Tomassone, R. (2006). Statistical Aspects of Pottery Quantification for the Dating of Some Archaeological Contexts. *Archaeometry*, 48(1), 169-183. doi:10.1111/j.1475-4754.2006.00249.x.

Poblome, J. & Groenen, P. J. F. (2003). Constrained Correspondence Analysis for Seriation of Sagalassos Tablewares. In Doerr, M. & Apostolis, S. (eds.), *The Digital Heritage of Archaeology*. Athens: Hellenic Ministry of Culture.

See Also

`plot()`, `predict_event()`, `predict_accumulation()`, `jackknife()`, `bootstrap()`

Other dating methods: `mcd()`, `predict_event()`

Examples

```
## Data from Peeples and Schachner 2012
data("zuni", package = "folio")

## Assume that some assemblages are reliably dated (this is NOT a real example)
zuni_dates <- c(
  LZ0569 = 1097, LZ0279 = 1119, CS16 = 1328, LZ0066 = 1111,
  LZ0852 = 1216, LZ1209 = 1251, CS144 = 1262, LZ0563 = 1206,
  LZ0329 = 1076, LZ0005Q = 859, LZ0322 = 1109, LZ0067 = 863,
  LZ0578 = 1180, LZ0227 = 1104, LZ0610 = 1074
)

## Model the event and accumulation date for each assemblage
model <- event(zuni, zuni_dates, rank = 10)
plot(model, select = 1:10, event = TRUE, flip = TRUE)
```

`fit` *Frequency Increment Test*

Description

Frequency Increment Test

Usage

```
fit(object, dates, ...)

## S4 method for signature 'data.frame,numeric'
fit(object, dates, calendar = CE(), level = 0.95, roll = FALSE, window = 3)

## S4 method for signature 'matrix,numeric'
fit(object, dates, calendar = CE(), level = 0.95, roll = FALSE, window = 3)
```

Arguments

`object`	A $m \times p$ `numeric` `matrix` or `data.frame` of count data (absolute frequencies giving the number of individuals for each category, i.e. a contingency table). A `data.frame` will be coerced to a `numeric` `matrix` via `data.matrix()`.
`dates`	A length-m `numeric` vector of dates.
`...`	Currently not used.
`calendar`	A `TimeScale` object specifying the calendar of `dates` (see `calendar()`). Defaults to Gregorian Common Era.
`level`	A length-one `numeric` vector giving the confidence level.
`roll`	A `logical` scalar: should each time series be subsetted to look for episodes of selection?
`window`	An odd `integer` giving the size of the rolling window. Only used if `roll` is `TRUE`.

Details

The Frequency Increment Test (FIT) rejects neutrality if the distribution of normalized variant frequency increments exhibits a mean that deviates significantly from zero.

If `roll` is `TRUE`, each time series is subsetted according to `window` to see if episodes of selection can be identified among variables that might not show overall selection.

Value

An `IncrementTest` object.

Author(s)

N. Frerebeau

References

Feder, A. F., Kryazhimskiy, S. & Plotkin, J. B. (2014). Identifying Signatures of Selection in Genetic Time Series. *Genetics*, 196(2): 509-522. doi:10.1534/genetics.113.158220.

See Also

`plot()`

Other chronological analysis: `aoristic()`, `apportion()`, `roc()`

Examples

```
## Data from Crema et al. 2016
data("merzbach", package = "folio")

## Keep only decoration types that have a maximum frequency of at least 50
keep <- apply(X = merzbach, MARGIN = 2, FUN = function(x) max(x) >= 50)
counts <- merzbach[, keep]
```

```
## Group by phase
## We use the row names as time coordinates (roman numerals)
dates <- as.numeric(utils::as.roman(rownames(counts)))

## Frequency Increment Test
freq <- fit(counts, dates, calendar = NULL)

## Plot time vs abundance
plot(freq, calendar = NULL, ncol = 3, xlab = "Phases")

## Plot time vs abundance and highlight selection
freq <- fit(counts, dates, calendar = NULL, roll = TRUE, window = 5)
plot(freq, calendar = NULL, ncol = 3, xlab = "Phases")
```

---

mcd *Mean Ceramic Date*

Description

Estimates the Mean Ceramic Date of an assemblage.

Usage

```
mcd(object, dates, ...)

## S4 method for signature 'numeric,numeric'
mcd(object, dates, calendar = CE())

## S4 method for signature 'data.frame,numeric'
mcd(object, dates, calendar = CE())

## S4 method for signature 'matrix,numeric'
mcd(object, dates, calendar = CE())
```

Arguments

object	A $m \times p$ numeric matrix or data.frame of count data (absolute frequencies giving the number of individuals for each category, i.e. a contingency table). A data.frame will be coerced to a numeric matrix via data.matrix().
dates	A length-p numeric vector of dates expressed in years.
...	Currently not used.
calendar	A TimeScale object specifying the calendar of dates (see calendar()). Defaults to Gregorian Common Era.

Details

The Mean Ceramic Date (MCD) is a point estimate of the occupation of an archaeological site (South 1977). The MCD is estimated as the weighted mean of the date midpoints of the ceramic types (based on absolute dates or the known production interval) found in a given assemblage. The weights are the relative frequencies of the respective types in the assemblage.

A bootstrapping procedure is used to estimate the confidence interval of a given MCD. For each assemblage, a large number of new bootstrap replicates is created, with the same sample size, by resampling the original assemblage with replacement. MCDs are calculated for each replicates and upper and lower boundaries of the confidence interval associated with each MCD are then returned.

Value

A `MeanDate` object.

Author(s)

N. Frerebeau

References

South, S. A. (1977). *Method and Theory in Historical Archaeology*. New York: Academic Press.

See Also

plot(), bootstrap(), jackknife(), simulate()

Other dating methods: `event()`, `predict_event()`

Examples

```
## Data from Peeples and Schachner 2012
data("zuni", package = "folio")

## Set the start and end dates for each ceramic type
dates <- list(
  LINO = c(600, 875), KIAT = c(850, 950), RED = c(900, 1050),
  GALL = c(1025, 1125), ESC = c(1050, 1150), PUBW = c(1050, 1150),
  RES = c(1000, 1200), TULA = c(1175, 1300), PINE = c(1275, 1350),
  PUBR = c(1000, 1200), WING = c(1100, 1200), WIPO = c(1125, 1225),
  SJ = c(1200, 1300), LSJ = c(1250, 1300), SPR = c(1250, 1300),
  PINER = c(1275, 1325), HESH = c(1275, 1450), KWAK = c(1275, 1450)
)

## Calculate date midpoints
mid <- vapply(X = dates, FUN = mean, FUN.VALUE = numeric(1))

## Calculate MCD
(mc_dates <- mcd(zuni[100:125, ], dates = mid))

## Get MCD in years CE
time(mc_dates, calendar = CE())
```

```
## Plot
plot(mc_dates)

## Bootstrap resampling
boot <- bootstrap(mc_dates, n = 30)
head(boot)

## Jackknife resampling
jack <- jackknife(mc_dates)
head(jack)

## Simulation
sim <- simulate(mc_dates, nsim = 30)
plot(sim, interval = "range", pch = 16)
```

`model` *Extract Model Results*

Description

- `coef()` extracts model coefficients (see `stats::coef()`).
- `fitted()` extracts model fitted values (see `stats::fitted()`).
- `residuals()` extracts model residuals (see `stats::residuals()`).
- `sigma()` extracts the residual standard deviation (see `stats::sigma()`).
- `terms()` extracts model terms (see `stats::terms()`).

Usage

```
## S4 method for signature 'EventDate'
coef(object, calendar = NULL, ...)

## S4 method for signature 'EventDate'
fitted(object, calendar = NULL, ...)

## S4 method for signature 'EventDate'
residuals(object, calendar = NULL, ...)

## S4 method for signature 'EventDate'
sigma(object, calendar = NULL, ...)

## S4 method for signature 'EventDate'
terms(x, ...)
```

Arguments

calendar	A TimeScale object specifying the target calendar (see calendar()). If NULL (the default), *rata die* are returned.
...	Currently not used.
x, object	An EventDate object.

Author(s)

N. Frerebeau

See Also

Other mutators: data.frame, mutators, series(), subset()

Examples

```
## Data from Peeples and Schachner 2012
data("zuni", package = "folio")

## Assume that some assemblages are reliably dated (this is NOT a real example)
zuni_dates <- c(
  LZ0569 = 1097, LZ0279 = 1119, CS16 = 1328, LZ0066 = 1111,
  LZ0852 = 1216, LZ1209 = 1251, CS144 = 1262, LZ0563 = 1206,
  LZ0329 = 1076, LZ0005Q = 859, LZ0322 = 1109, LZ0067 = 863,
  LZ0578 = 1180, LZ0227 = 1104, LZ0610 = 1074
)

## Model the event and accumulation date for each assemblage
model <- event(zuni, zuni_dates, rank = 10)
plot(model, select = 1:10, event = TRUE, flip = TRUE)
```

---

mutators *Get or Set Parts of an Object*

Description

Getters and setters to retrieve or set parts of an object.

Usage

```
## S4 method for signature 'AoristicSum'
weights(object, ...)

## S4 method for signature 'CountApportion'
weights(object, ...)
```

Arguments

object	An object from which to get or set element(s).
...	Currently not used.

Author(s)

N. Frerebeau

See Also

Other mutators: data.frame, model, series(), subset()

---

permute *Rearranges a Data Matrix*

Description

- permute() rearranges a data matrix according to a permutation order.
- get_order() returns the seriation order for rows and/or columns.

Usage

```
permute(object, order, ...)

get_order(x, ...)

## S4 method for signature 'data.frame,PermutationOrder'
permute(object, order)

## S4 method for signature 'matrix,PermutationOrder'
permute(object, order)

## S4 method for signature 'PermutationOrder'
get_order(x, margin = c(1, 2))
```

Arguments

object	A $m \times p$ numeric matrix or data.frame of count data (absolute frequencies giving the number of individuals for each category, i.e. a contingency table). A data.frame will be coerced to a numeric matrix via data.matrix().
...	Currently not used.
x, order	A PermutationOrder object giving the permutation order for rows and columns.
margin	A numeric vector giving the subscripts which the rearrangement will be applied over: 1 indicates rows, 2 indicates columns, c(1, 2) indicates rows and columns.

Value

- `permute()` returns a permuted `matrix` or a permuted `data.frame` (the same as `object`).

Author(s)

N. Frerebeau

See Also

`dimensio::ca()`

Other seriation methods: `seriate_average()`, `seriate_rank()`, `seriate_refine()`

Examples

```
## Replicates Desachy 2004 results
data("compiegne", package = "folio")

## Get seriation order for columns on EPPM using the reciprocal averaging method
## Expected column order: N, A, C, K, P, L, B, E, I, M, D, G, O, J, F, H
(indices <- seriate_rank(compiegne, EPPM = TRUE, margin = 2))

## Get permutation order
get_order(indices, 1) # rows
get_order(indices, 2) # columns

## Permute columns
(new <- permute(compiegne, indices))

## See the vignette
## Not run:
utils::vignette("seriation")

## End(Not run)
```

---

plot_aoristic *Plot Aoristic Analysis*

Description

Plot Aoristic Analysis

Usage

```
## S4 method for signature 'AoristicSum,missing'
plot(
  x,
  calendar = getOption("kairos.calendar"),
  type = c("bar"),
```

```
  flip = FALSE,
  ncol = NULL,
  main = NULL,
  sub = NULL,
  ann = graphics::par("ann"),
  axes = TRUE,
  frame.plot = axes,
  panel.first = NULL,
  panel.last = NULL,
  ...
)

## S4 method for signature 'AoristicSum'
image(x, calendar = getOption("kairos.calendar"), ...)

## S4 method for signature 'RateOfChange,missing'
plot(
  x,
  calendar = getOption("kairos.calendar"),
  level = 0.95,
  flip = FALSE,
  ncol = NULL,
  main = NULL,
  sub = NULL,
  ann = graphics::par("ann"),
  axes = TRUE,
  frame.plot = axes,
  panel.first = NULL,
  panel.last = NULL,
  ...
)
```

Arguments

x	An AoristicSum object.
calendar	A TimeScale object specifying the target calendar (see calendar()).
type	A character string specifying whether bar or density should be plotted? It must be one of "bar" or "density". Any unambiguous substring can be given.
flip	A logical scalar: should the y-axis (ticks and numbering) be flipped from side 2 (left) to 4 (right) from series to series when facet is "multiple"?
ncol	An integer specifying the number of columns to use when facet is "multiple". Defaults to 1 for up to 4 series, otherwise to 2.
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.
...	Further parameters to be passed to panel (e.g. graphical parameters).
level	A length-one numeric vector giving the confidence level.

Value

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

Author(s)

N. Frerebeau

See Also

aoristic(), roc()

Other plotting methods: plot_event, plot_fit, plot_mcd, plot_time()

Examples

```
## Data from Husi 2022
data("loire", package = "folio")

## Get time range
loire_range <- loire[, c("lower", "upper")]

## Calculate aoristic sum (normal)
aorist_raw <- aoristic(loire_range, step = 50, weight = FALSE)
plot(aorist_raw, col = "grey")

## Calculate aoristic sum (weights)
aorist_weighted <- aoristic(loire_range, step = 50, weight = TRUE)
plot(aorist_weighted, col = "grey")

## Calculate aoristic sum (weights) by group
aorist_groups <- aoristic(loire_range, step = 50, weight = TRUE,
                          groups = loire$area)
plot(aorist_groups, flip = TRUE, col = "grey")
image(aorist_groups)

## Rate of change
roc_weighted <- roc(aorist_weighted, n = 30)
plot(roc_weighted)

## Rate of change by group
roc_groups <- roc(aorist_groups, n = 30)
```

```
plot(roc_groups, flip = TRUE)
```

---

plot_event *Event Plot*

Description

Produces an activity or a tempo plot.

Usage

```
## S4 method for signature 'EventDate,missing'
plot(
  x,
  type = c("activity", "tempo"),
  event = FALSE,
  calendar = getOption("kairos.calendar"),
  select = 1,
  n = 500,
  eps = 1e-09,
  col.accumulation = "black",
  col.event = "red",
  flip = FALSE,
  ncol = NULL,
  xlab = NULL,
  ylab = NULL,
  main = NULL,
  sub = NULL,
  ann = graphics::par("ann"),
  axes = TRUE,
  frame.plot = axes,
  ...
)
```

Arguments

x	An EventDate object.
type	A character string indicating the type of plot. It must be one of "activity" (default) or "tempo" (see details). Any unambiguous substring can be given.
event	A logical scalar: should the distribution of the event date be displayed? Only used if type is "activity".
calendar	A TimeScale object specifying the target calendar (see calendar()).
select	A numeric or character vector giving the selection of the assemblage that are drawn.
n	A length-one non-negative numeric vector giving the desired length of the vector of quantiles for density computation.

eps	A length-one numeric value giving the cutoff below which values will be removed.
col.accumulation	A color specification for the accumulation density curve.
col.event	A color specification for the event density curve.
flip	A logical scalar: should the y-axis (ticks and numbering) be flipped from side 2 (left) to 4 (right) from series to series when facet is "multiple"?
ncol	An integer specifying the number of columns to use when facet is "multiple". Defaults to 1 for up to 4 series, otherwise to 2.
xlab, ylab	A character vector giving the x and y axis labels.
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?
...	Further parameters to be passed to panel (e.g. graphical parameters).

Value

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

Event and Acccumulation Dates

plot() displays the probability estimate density curves of archaeological assemblage dates (*event* and *accumulation* dates; Bellanger and Husi 2012). The *event* date is plotted as a line, while the *accumulation* date is shown as a grey filled area.

The accumulation date can be displayed as a tempo plot (Dye 2016) or an activity plot (Philippe and Vibet 2020):

tempo A tempo plot estimates the cumulative occurrence of archaeological events, such as the slope of the plot directly reflects the pace of change.

activity An activity plot displays the first derivative of the tempo plot.

Author(s)

N. Frerebeau

References

Bellanger, L. & Husi, P. (2012). Statistical Tool for Dating and Interpreting Archaeological Contexts Using Pottery. *Journal of Archaeological Science*, 39(4), 777-790. doi:10.1016/j.jas.2011.06.031.

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.

Philippe, A. & Vibet, M.-A. (2020). Analysis of Archaeological Phases Using the R Package ArchaeoPhases. *Journal of Statistical Software, Code Snippets*, 93(1), 1-25. doi:10.18637/jss.v093.c01.

See Also

event()

event()

Other plotting methods: plot_aoristic, plot_fit, plot_mcd, plot_time()

Examples

```
## Data from Peeples and Schachner 2012
data("zuni", package = "folio")

## Assume that some assemblages are reliably dated (this is NOT a real example)
zuni_dates <- c(
  LZ0569 = 1097, LZ0279 = 1119, CS16 = 1328, LZ0066 = 1111,
  LZ0852 = 1216, LZ1209 = 1251, CS144 = 1262, LZ0563 = 1206,
  LZ0329 = 1076, LZ0005Q = 859, LZ0322 = 1109, LZ0067 = 863,
  LZ0578 = 1180, LZ0227 = 1104, LZ0610 = 1074
)

## Model the event and accumulation date for each assemblage
model <- event(zuni, zuni_dates, rank = 10)
plot(model, select = 1:10, event = TRUE, flip = TRUE)
```

---

plot_fit *Detection of Selective Processes*

Description

Produces an abundance *vs* time diagram.

Usage

```
## S4 method for signature 'IncrementTest,missing'
plot(
  x,
  calendar = getOption("kairos.calendar"),
  col.neutral = "#004488",
  col.selection = "#BB5566",
  col.roll = "grey",
  flip = FALSE,
  ncol = NULL,
  xlab = NULL,
  ylab = NULL,
  main = NULL,
  sub = NULL,
  ann = graphics::par("ann"),
  axes = TRUE,
  frame.plot = axes,
  ...
)
```

Arguments

x	An IncrementTest object to be plotted.
calendar	A TimeScale object specifying the target calendar (see calendar()).
col.neutral, col.selection, col.roll	A vector of colors.
flip	A logical scalar: should the y-axis (ticks and numbering) be flipped from side 2 (left) to 4 (right) from series to series when facet is "multiple"?
ncol	An integer specifying the number of columns to use when facet is "multiple". Defaults to 1 for up to 4 series, otherwise to 2.
xlab, ylab	A character vector giving the x and y axis labels.
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?
...	Further parameters to be passed to panel (e.g. graphical parameters).

Details

Results of the frequency increment test can be displayed on an abundance *vs* time diagram aid in the detection and quantification of selective processes in the archaeological record. If roll is TRUE, each time series is subsetted according to window to see if episodes of selection can be identified among decoration types that might not show overall selection. If so, shading highlights the data points where fit() identifies selection.

Value

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

Note

Displaying FIT results on an abundance *vs* time diagram is adapted from Ben Marwick's original idea.

Author(s)

N. Frerebeau

See Also

fit()

Other plotting methods: plot_aoristic, plot_event, plot_mcd, plot_time()

Examples

```
## Data from Crema et al. 2016
data("merzbach", package = "folio")

## Keep only decoration types that have a maximum frequency of at least 50
keep <- apply(X = merzbach, MARGIN = 2, FUN = function(x) max(x) >= 50)
counts <- merzbach[, keep]

## Group by phase
## We use the row names as time coordinates (roman numerals)
dates <- as.numeric(utils::as.roman(rownames(counts)))

## Frequency Increment Test
freq <- fit(counts, dates, calendar = NULL)

## Plot time vs abundance
plot(freq, calendar = NULL, ncol = 3, xlab = "Phases")

## Plot time vs abundance and highlight selection
freq <- fit(counts, dates, calendar = NULL, roll = TRUE, window = 5)
plot(freq, calendar = NULL, ncol = 3, xlab = "Phases")
```

---

plot_mcd *MCD Plot*

Description

MCD Plot

Usage

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

## S4 method for signature 'SimulationMeanDate,missing'
plot(
```

```
  x,
  calendar = getOption("kairos.calendar"),
  interval = "student",
  level = 0.8,
  decreasing = TRUE,
  main = NULL,
  sub = NULL,
  ann = graphics::par("ann"),
  axes = TRUE,
  frame.plot = axes,
  panel.first = NULL,
  panel.last = NULL,
  ...
)
```

Arguments

x	A MeanDate object.
calendar	A TimeScale object specifying the target calendar (see calendar()).
decreasing	A logical scalar: should the sort be increasing or decreasing?
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, y and z 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.
...	Further graphical parameters.
interval	A character string giving the type of confidence interval to be returned. It must be one "student" (the default), "normal", "percentiles" or "range" (min-max). Any unambiguous substring can be given.
level	A length-one numeric vector giving the confidence level. Only used if interval is not "range".

Value

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

Author(s)

N. Frerebeau

See Also

mcd()

Other plotting methods: plot_aoristic, plot_event, plot_fit, plot_time()

Examples

```
## Data from Peeples and Schachner 2012
data("zuni", package = "folio")

## Set the start and end dates for each ceramic type
dates <- list(
  LINO = c(600, 875), KIAT = c(850, 950), RED = c(900, 1050),
  GALL = c(1025, 1125), ESC = c(1050, 1150), PUBW = c(1050, 1150),
  RES = c(1000, 1200), TULA = c(1175, 1300), PINE = c(1275, 1350),
  PUBR = c(1000, 1200), WING = c(1100, 1200), WIPO = c(1125, 1225),
  SJ = c(1200, 1300), LSJ = c(1250, 1300), SPR = c(1250, 1300),
  PINER = c(1275, 1325), HESH = c(1275, 1450), KWAK = c(1275, 1450)
)

## Calculate date midpoints
mid <- vapply(X = dates, FUN = mean, FUN.VALUE = numeric(1))

## Calculate MCD
(mc_dates <- mcd(zuni[100:125, ], dates = mid))

## Get MCD in years CE
time(mc_dates, calendar = CE())

## Plot
plot(mc_dates)

## Bootstrap resampling
boot <- bootstrap(mc_dates, n = 30)
head(boot)

## Jackknife resampling
jack <- jackknife(mc_dates)
head(jack)

## Simulation
sim <- simulate(mc_dates, nsim = 30)
plot(sim, interval = "range", pch = 16)
```

---

plot_time *Abundance vs Time Plot*

Description

Produces an abundance *vs* time diagram.

Usage

```
plot_time(object, dates, ...)

## S4 method for signature 'data.frame,numeric'
plot_time(object, dates, calendar = getOption("kairos.calendar"), ...)

## S4 method for signature 'matrix,numeric'
plot_time(object, dates, calendar = getOption("kairos.calendar"), ...)
```

Arguments

object	A $m \times p$ numeric matrix or data.frame of count data (absolute frequencies giving the number of individuals for each category, i.e. a contingency table). A data.frame will be coerced to a numeric matrix via data.matrix().
dates	A numeric vector of dates.
...	Further parameters to be passed to aion::plot().
calendar	A TimeScale object specifying the target calendar (see calendar()).

Value

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

Author(s)

N. Frerebeau

See Also

Other plotting methods: plot_aoristic, plot_event, plot_fit, plot_mcd

Examples

```
## Data from Crema et al. 2016
data("merzbach", package = "folio")

## Coerce the merzbach dataset to a count matrix
## Keep only decoration types that have a maximum frequency of at least 50
keep <- apply(X = merzbach, MARGIN = 2, FUN = function(x) max(x) >= 50)
counts <- merzbach[, keep]

## Set dates
## We use the row names as time coordinates (roman numerals)
dates <- as.numeric(utils::as.roman(rownames(counts)))

## Plot abundance vs time
plot_time(counts, dates, calendar = NULL, ncol = 3, xlab = "Phases")
```

predict_event *Predict Event and Accumulation Dates*

Description

Estimates the event and accumulation dates of an assemblage.

Usage

```
predict_event(object, data, ...)

predict_accumulation(object, data, ...)

## S4 method for signature 'EventDate,missing'
predict_event(object, margin = 1, level = 0.95, calendar = NULL)

## S4 method for signature 'EventDate,matrix'
predict_event(object, data, margin = 1, level = 0.95, calendar = NULL)

## S4 method for signature 'EventDate,missing'
predict_accumulation(object, calendar = NULL)

## S4 method for signature 'EventDate,matrix'
predict_accumulation(object, data, level = 0.95, calendar = NULL)
```

Arguments

object	A $m \times p$ numeric matrix or data.frame of count data (absolute frequencies giving the number of individuals for each category, i.e. a contingency table). A data.frame will be coerced to a numeric matrix via data.matrix().
data	A numeric matrix or a data.frame of count data (absolute frequencies) for which to predict event and accumulation dates.
...	Further arguments to be passed to internal methods.
margin	A numeric vector giving the subscripts which the prediction will be applied over: 1 indicates rows, 2 indicates columns.
level	A length-one numeric vector giving the confidence level.
calendar	A TimeScale object specifying the target calendar (see calendar()). If NULL (the default), *rata die* are returned.

Value

- predict_event() returns a data.frame.
- predict_accumulation() returns a data.frame.

Author(s)

N. Frerebeau

References

Bellanger, L. & Husi, P. (2013). Mesurer et modéliser le temps inscrit dans la matière à partir d'une source matérielle : la céramique médiévale. In *Mesure et Histoire Médiévale*. Histoire ancienne et médiévale. Paris: Publication de la Sorbonne, p. 119-134.

Bellanger, L. & Husi, P. (2012). Statistical Tool for Dating and Interpreting Archaeological Contexts Using Pottery. *Journal of Archaeological Science*, 39(4), 777-790. doi:10.1016/j.jas.2011.06.031.

Bellanger, L., Tomassone, R. & Husi, P. (2008). A Statistical Approach for Dating Archaeological Contexts. *Journal of Data Science*, 6, 135-154.

Bellanger, L., Husi, P. & Tomassone, R. (2006). Une approche statistique pour la datation de contextes archéologiques. *Revue de Statistique Appliquée*, 54(2), 65-81.

Bellanger, L., Husi, P. & Tomassone, R. (2006). Statistical Aspects of Pottery Quantification for the Dating of Some Archaeological Contexts. *Archaeometry*, 48(1), 169-183. doi:10.1111/j.1475-4754.2006.00249.x.

See Also

`event()`

Other dating methods: `event()`, `mcd()`

Examples

```
## Data from Peeples and Schachner 2012
data("zuni", package = "folio")

## Assume that some assemblages are reliably dated (this is NOT a real example)
zuni_dates <- c(
  LZ0569 = 1097, LZ0279 = 1119, CS16 = 1328, LZ0066 = 1111,
  LZ0852 = 1216, LZ1209 = 1251, CS144 = 1262, LZ0563 = 1206,
  LZ0329 = 1076, LZ0005Q = 859, LZ0322 = 1109, LZ0067 = 863,
  LZ0578 = 1180, LZ0227 = 1104, LZ0610 = 1074
)

## Model the event and accumulation date for each assemblage
model <- event(zuni, zuni_dates, rank = 10)
plot(model, select = 1:10, event = TRUE, flip = TRUE)
```

`resample_event` *Resample Event Dates*

Description

- `bootstrap()` generate bootstrap estimations of an event.
- `jackknife()` generate jackknife estimations of an event.

Usage

```
## S4 method for signature 'EventDate'
jackknife(object, level = 0.95, progress = getOption("kairos.progress"), ...)

## S4 method for signature 'EventDate'
bootstrap(
  object,
  level = 0.95,
  probs = c(0.05, 0.95),
  n = 1000,
  progress = getOption("kairos.progress"),
  ...
)
```

Arguments

`object`	An `EventDate` object (typically returned by `event()`).
`level`	A length-one `numeric` vector giving the confidence level.
`progress`	A `logical` scalar: should a progress bar be displayed?
`...`	Further arguments to be passed to internal methods.
`probs`	A `numeric` vector of probabilities with values in $[0, 1]$.
`n`	A non-negative `integer` specifying the number of bootstrap replications.

Details

If `jackknife()` is used, one type/fabric is removed at a time and all statistics are recalculated. In this way, one can assess whether certain type/fabric has a substantial influence on the date estimate. A three columns `data.frame` is returned, giving the results of the resampling procedure (jackknifing fabrics) for each assemblage (in rows) with the following columns:

`mean` The jackknife mean (event date).

`bias` The jackknife estimate of bias.

`error` The standard error of predicted means.

If `bootstrap()` is used, a large number of new bootstrap assemblages is created, with the same sample size, by resampling each of the original assemblage with replacement. Then, examination of the bootstrap statistics makes it possible to pinpoint assemblages that require further investigation.

A five columns `data.frame` is returned, giving the bootstrap distribution statistics for each replicated assemblage (in rows) with the following columns:

`min` Minimum value.

`mean` Mean value (event date).

`max` Maximum value.

`Q5` Sample quantile to 0.05 probability.

`Q95` Sample quantile to 0.95 probability.

Value

A data.frame.

Author(s)

N. Frerebeau

See Also

event()

Other resampling methods: resample_mcd

---

resample_mcd *Resample Mean Ceramic Dates*

Description

- bootstrap() generate bootstrap estimations of an MCD.
- jackknife() generate jackknife estimations of an MCD.

Usage

```
## S4 method for signature 'MeanDate'
bootstrap(object, n = 1000, f = NULL, calendar = getOption("kairos.calendar"))

## S4 method for signature 'MeanDate'
jackknife(object, f = NULL, calendar = getOption("kairos.calendar"))

## S4 method for signature 'MeanDate'
simulate(object, nsim = 1000)
```

Arguments

object	A MeanDate object (typically returned by mcd()).
n	A non-negative integer specifying the number of bootstrap replications.
f	A function that takes a single numeric vector (the result of the resampling procedure) as argument.
calendar	A TimeScale object specifying the target calendar (see calendar()).
nsim	A non-negative integer specifying the number of simulations.

Value

If f is NULL, bootstrap() and jackknife() return a data.frame with the following elements (else, returns the result of f applied to the n resampled values) :

original The observed value.

mean The bootstrap/jackknife estimate of mean.

bias The bootstrap/jackknife estimate of bias.

error The boostrap/jackknife estimate of standard erro.

Author(s)

N. Frerebeau

See Also

Other resampling methods: resample_event

---

roc *Rate of Change*

Description

Computes the rate of change from an aoristic analysis.

Usage

```
roc(object, ...)

## S4 method for signature 'AoristicSum'
roc(object, n = 100)
```

Arguments

object	An AoristicSum object.
...	Currently not used.
n	A non-negative integer giving the number of replications (see details).

Value

A RateOfChange object.

Author(s)

N. Frerebeau

References

Baxter, M. J. & Cool, H. E. M. (2016). Reinventing the Wheel? Modelling Temporal Uncertainty with Applications to Brooch Distributions in Roman Britain. *Journal of Archaeological Science*, 66: 120-27. doi:10.1016/j.jas.2015.12.007.

Crema, E. R. (2012). Modelling Temporal Uncertainty in Archaeological Analysis. *Journal of Archaeological Method and Theory*, 19(3): 440-61. doi:10.1007/s1081601191223.

See Also

`aoristic()`, `plot()`

Other chronological analysis: `aoristic()`, `apportion()`, `fit()`

Examples

```
## Data from Husi 2022
data("loire", package = "folio")

## Get time range
loire_range <- loire[, c("lower", "upper")]

## Calculate aoristic sum (normal)
aorist_raw <- aoristic(loire_range, step = 50, weight = FALSE)
plot(aorist_raw, col = "grey")

## Calculate aoristic sum (weights)
aorist_weighted <- aoristic(loire_range, step = 50, weight = TRUE)
plot(aorist_weighted, col = "grey")

## Calculate aoristic sum (weights) by group
aorist_groups <- aoristic(loire_range, step = 50, weight = TRUE,
                          groups = loire$area)
plot(aorist_groups, flip = TRUE, col = "grey")
image(aorist_groups)

## Rate of change
roc_weighted <- roc(aorist_weighted, n = 30)
plot(roc_weighted)

## Rate of change by group
roc_groups <- roc(aorist_groups, n = 30)
plot(roc_groups, flip = TRUE)
```

`seriate_average` *Correspondence Analysis-Based Seriation*

Description

Correspondence Analysis-Based Seriation

Usage

```
seriate_average(object, ...)

## S4 method for signature 'data.frame'
seriate_average(object, margin = c(1, 2), axes = 1, ...)

## S4 method for signature 'matrix'
seriate_average(object, margin = c(1, 2), axes = 1, ...)
```

Arguments

object	A $m \times p$ numeric matrix or data.frame of count data (absolute frequencies giving the number of individuals for each category, i.e. a contingency table). A data.frame will be coerced to a numeric matrix via data.matrix().
...	Further arguments to be passed to internal methods.
margin	A numeric vector giving the subscripts which the rearrangement will be applied over: 1 indicates rows, 2 indicates columns, c(1, 2) indicates rows then columns, c(2, 1) indicates columns then rows.
axes	An integer vector giving the subscripts of the CA axes to be used.

Details

Correspondence analysis (CA) is an effective method for the seriation of archaeological assemblages. The order of the rows and columns is given by the coordinates along one dimension of the CA space, assumed to account for temporal variation. The direction of temporal change within the correspondence analysis space is arbitrary: additional information is needed to determine the actual order in time.

Value

An AveragePermutationOrder object.

Author(s)

N. Frerebeau

References

Ihm, P. (2005). A Contribution to the History of Seriation in Archaeology. In C. Weihs & W. Gaul (Eds.), *Classification: The Ubiquitous Challenge*. Berlin Heidelberg: Springer, p. 307-316. doi:10.1007/3540280847_34.

See Also

dimensio::ca()

Other seriation methods: permute(), seriate_rank(), seriate_refine()

Examples

```
## Replicates Desachy 2004 results
data("compiegne", package = "folio")

## Get seriation order for columns on EPPM using the reciprocal averaging method
## Expected column order: N, A, C, K, P, L, B, E, I, M, D, G, O, J, F, H
(indices <- seriate_rank(compiegne, EPPM = TRUE, margin = 2))

## Get permutation order
get_order(indices, 1) # rows
get_order(indices, 2) # columns

## Permute columns
(new <- permute(compiegne, indices))

## See the vignette
## Not run:
utils::vignette("seriation")

## End(Not run)
```

`seriate_rank` *Reciprocal Ranking Seriation*

Description

Reciprocal Ranking Seriation

Usage

```
seriate_rank(object, ...)

## S4 method for signature 'data.frame'
seriate_rank(object, EPPM = FALSE, margin = c(1, 2), stop = 100)

## S4 method for signature 'matrix'
seriate_rank(object, EPPM = FALSE, margin = c(1, 2), stop = 100)
```

Arguments

`object`	A $m \times p$ `numeric` `matrix` or `data.frame` of count data (absolute frequencies giving the number of individuals for each category, i.e. a contingency table). A `data.frame` will be coerced to a `numeric` `matrix` via `data.matrix()`.
`...`	Currently not used.
`EPPM`	A `logical` scalar: should the seriation be computed on EPPM instead of raw data?

margin	A numeric vector giving the subscripts which the rearrangement will be applied over: 1 indicates rows, 2 indicates columns, c(1, 2) indicates rows then columns, c(2, 1) indicates columns then rows.
stop	An integer giving the stopping rule (i.e. maximum number of iterations) to avoid infinite loop.

Details

This procedure iteratively rearrange rows and/or columns according to their weighted rank in the data matrix until convergence.

Note that this procedure could enter into an infinite loop. If no convergence is reached before the maximum number of iterations, it stops with a warning.

Value

A RankPermutationOrder object.

Author(s)

N. Frerebeau

References

Desachy, B. (2004). Le sériographe EPPM: un outil informatisé de sériation graphique pour tableaux de comptages. *Revue archéologique de Picardie*, 3(1), 39-56. doi:10.3406/pica.2004.2396.

Dunnell, R. C. (1970). Seriation Method and Its Evaluation. *American Antiquity*, 35(03), 305-319. doi:10.2307/278341.

Ihm, P. (2005). A Contribution to the History of Seriation in Archaeology. In C. Weihs & W. Gaul (Eds.), *Classification: The Ubiquitous Challenge*. Berlin Heidelberg: Springer, p. 307-316. doi:10.1007/3540280847_34.

See Also

Other seriation methods: permute(), seriate_average(), seriate_refine()

Examples

```
## Replicates Desachy 2004 results
data("compiegne", package = "folio")

## Get seriation order for columns on EPPM using the reciprocal averaging method
## Expected column order: N, A, C, K, P, L, B, E, I, M, D, G, O, J, F, H
(indices <- seriate_rank(compiegne, EPPM = TRUE, margin = 2))

## Get permutation order
get_order(indices, 1) # rows
get_order(indices, 2) # columns

## Permute columns
(new <- permute(compiegne, indices))
```

```
## See the vignette
## Not run:
utils::vignette("seriation")

## End(Not run)
```

`seriate_refine` *Refine CA-based Seriation*

Description

Refine CA-based Seriation

Usage

```
seriate_refine(object, ...)

## S4 method for signature 'AveragePermutationOrder'
seriate_refine(object, cutoff, margin = 1, axes = 1, n = 30, ...)

## S4 method for signature 'BootstrapCA'
seriate_refine(object, cutoff, margin = 1, axes = 1, ...)

## S4 method for signature 'RefinePermutationOrder'
hist(x, ...)
```

Arguments

`object`	A `PermutationOrder` object (typically returned by `seriate_average()`).
`...`	Further arguments to be passed to internal methods.
`cutoff`	A function that takes a numeric vector as argument and returns a single numeric value (see below).
`margin`	A length-one `numeric` vector giving the subscripts which the refinement will be applied over: `1` indicates rows, `2` indicates columns.
`axes`	An `integer` vector giving the subscripts of the CA axes to be used.
`n`	A non-negative `integer` giving the number of bootstrap replications.
`x`	A `RefinePermutationOrder` object

Details

`seriate_refine()` allows to identify samples that are subject to sampling error or samples that have underlying structural relationships and might be influencing the ordering along the CA space.

This relies on a partial bootstrap approach to CA-based seriation where each sample is replicated `n` times. The maximum dimension length of the convex hull around the sample point cloud allows to remove samples for a given `cutoff` value.

According to Peebles and Schachner (2012), "[this] point removal procedure [results in] a reduced dataset where the position of individuals within the CA are highly stable and which produces an ordering consistent with the assumptions of frequency seriation."

Value

- `seriate_refine()` returns a `RefinePermutationOrder` object.
- `hist()` is called it for its side-effects: it results in a histogram being displayed (invisibly returns `x`).

Methods (by class)

- `hist(RefinePermutationOrder)`: Compute and plot a histogram of convex hull maximum dimension length.

Author(s)

N. Frerebeau

References

Peeples, M. A., & Schachner, G. (2012). Refining correspondence analysis-based ceramic seriation of regional data sets. *Journal of Archaeological Science*, 39(8), 2818-2827. doi:10.1016/j.jas.2012.04.040.

See Also

Other seriation methods: `permute()`, `seriate_average()`, `seriate_rank()`

---

series *Sampling Times*

Description

Get the times at which a time series was sampled.

Usage

```
## S4 method for signature 'EventDate'
time(x, calendar = NULL)
```

Arguments

`x`	An R object.
`calendar`	A `TimeScale` object specifying the target calendar (see `calendar()`). If `NULL` (the default), *rata die* are returned.

Value

A numeric vector.

Author(s)

N. Frerebeau

See Also

Other mutators: data.frame, model, mutators, subset()

---

subset *Extract or Replace Parts of an Object*

Description

Operators acting on objects to extract or replace parts.

Usage

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

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

## S4 method for signature 'PermutationOrder,ANY,missing'
x[[i]]
```

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.
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 mutators: data.frame, model, mutators, series()

Index