Package 'groupedHyperframe'

March 25, 2025

Type Package

Title Grouped Hyper Data Frame: An Extension of Hyper Data Frame Object

Version 0.1.0

Date 2025-03-24

Maintainer Tingting Zhan <tingtingzhan@gmail.com>

Description An S3 class 'groupedHyperframe' that inherits from hyper data frame. Batch processes on point-pattern hyper column. Aggregation of function-value-table hyper column(s) and numeric hyper column(s) over a nested grouping structure.

RoxygenNote 7.3.2

LazyData true

LazyDataCompression xz

Encoding UTF-8

License GPL-2

Depends R (>= 4.4)

Language en-US

Imports cli, parallel, stats, nlme, matrixStats, pracma, spatstat.explore, spatstat.geom

Suggests knitr, survival, spatstat.data, spatstat.univar, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Tingting Zhan [aut, cre] (<https://orcid.org/0000-0001-9971-4844>), Inna Chervoneva [aut] (<https://orcid.org/0000-0002-9104-4505>)

Repository CRAN

Date/Publication 2025-03-25 15:50:09 UTC

Contents

`groupedHyperframe-package` *groupedHyperframe: Grouped Hyper Data Frame: An Extension of Hyper Data Frame Object*

Description

An S3 class 'groupedHyperframe' that inherits from hyper data frame. Batch processes on point-pattern hyper column. Aggregation of function-value-table hyper column(s) and numeric hyper column(s) over a nested grouping structure.

Author(s)

Maintainer: Tingting Zhan <tingtingzhan@gmail.com> (ORCID)

Authors:

- Inna Chervoneva <Inna.Chervoneva@jefferson.edu> (ORCID)

References

To be added.

`.nncross` *Alternative Interface of nncross.ppp*

Description

An alternative interface of function nncross.ppp.

Usage

```
.nncross(X, i, j, ...)
```

Arguments

X	see **Details**
i, j	character or integer scalars. See functions Gcross, etc. for more details
...	additional parameters of nncross.ppp

Details

Function `.nncross()` creates an interface similar to functions Gcross, etc., which takes an is.multitype ppp.object and two mark values `i` and `j`, then calls the workhorse function nncross.ppp with parameter `what = 'dist'`. If mark values `i` and `j` does not exist in the ppp.object, a `NULL` value will be returned.

Value

Function `.nncross()` returns a numeric vector if `i` and `j` are valid mark values of ppp.object `X`; otherwise returns a `NULL` value.

Examples

```
library(spatstat.data)
library(spatstat.geom)

(xs = split.ppp(amacrine))
(a1 = nncross(X = xs$off, Y = xs$on, what = 'dist'))
a2 = .nncross(amacrine, i = 'off', j = 'on')
a3 = .nncross(amacrine, i = 1L, j = 2L)
stopifnot(identical(a1, a2), identical(a1, a3))

.nncross(amacrine, i = 'a', j = 'b') # exception handling
```

---

aggregate_fv *Aggregate fv.objects by Cluster*

Description

Aggregate information in fv.objects by sample clustering.

Usage

```
aggregate_fv(
  X,
  by = stop("must specify `by`"),
  f_aggr_ = c("mean", "median", "max", "min"),
  ...
)
```

Arguments

X	a groupedHyperframe, containing one or more fv.object column(s)
by	one-sided formula, sample clustering. Use only one-level hierarchy (e.g., ~patient or ~image). Do not use multi-level hierarchy (e.g., ~patient/image)
f_aggr_	character scalar, method to aggregate within cluster, currently supports 'mean', 'median', 'max', and 'min'.
...	additional parameters, currently not in use

Value

Function aggregate_fv() returns a data.frame, with aggregated information stored in matrix-columns.

Note that hyperframe does not support matrix-column (for good reasons!). Therefore, function aggregate_fv() must return a data.frame, instead of a hyperframe.

Examples

```
library(spatstat.data)
library(spatstat.geom)
flu$pattern[] = flu$pattern |>
 lapply(FUN = `mark_name<-`, value = 'stain') # read ?flu carefully
r = seq.int(from = 0, to = 100, by = 5)
m = flu |>
 subset(stain == 'M2-M1') |>
 Gcross_(i = 'M1', j = 'M2', r = r, correction = 'best', mc.cores = 1L) |>
 as.groupedHyperframe(group = ~ virustype/frameid) |>
 aggregate_fv(by = ~ virustype, mc.cores = 1L)
names(m)
dim(m$pattern.G.value)
dim(m$pattern.G.cumtrapz)
```

---

aggregate_num *Aggregate numeric hypercolumns and/or marks, by Cluster*

Description

Aggregate numeric hypercolumns and/or marks by sample clustering.

Usage

```
aggregate_num(
  X,
  by = stop("must specify `by`"),
  FUN,
  FUN.name = deparse1(substitute(FUN)),
  f_aggr_ = c("mean", "median", "max", "min"),
  mc.cores = switch(.Platform$OS.type, windows = 1L, detectCores()),
```

```
  ...
)

aggregate_quantile(X, ...)

aggregate_kerndens(X, ...)
```

Arguments

X	a groupedHyperframe, containing either or all of • one or more numeric hypercolumns • one-and-only-one ppp-hypercolumns with one or more numeric marks
by	one-sided formula, one-level hierarchy clustering, e.g., ~patient or ~image. Do **not** use multi-level hierarchy, e.g., ~patient/image
FUN	function to extract information, currently supports functions quantile and kerndens
FUN.name	(optional) character scalar, user-friendly name of FUN
f_aggr_	character scalar, method to aggregate within cluster, currently supports 'mean', 'median', 'max', and 'min'.
mc.cores	integer scalar, see function mclapply. Default is 1L on Windows, or detectCores on Mac. CRAN requires mc.cores <= 2L in examples.
...	additional parameters of function FUN

Details

Function aggregate_quantile() is a wrapper of workhorse function aggregate_num() with FUN = quantile.

Function aggregate_kerndens() is a wrapper of workhorse function aggregate_num() with FUN = kerndens.

Value

Function aggregate_num() returns a data.frame, with aggregated information stored in matrix-columns.

---

groupedHyperframe *Grouped Hyper Data Frame*

Description

A class groupedHyperframe to represent hyperframe with a (multilevel) hierarchical structure.

Details

The class groupedHyperframe inherits from class hyperframe. This class has additional attributes

attr(,'group') formula

Value

A groupedHyperframe

---

grouped_ppp *groupedHyperframe with One-and-Only-One ppp-hypercolumn*

Description

..

Usage

```
grouped_ppp(
  formula,
  data,
  coords = ~x + y,
  window = owin(xrange = range(.x), yrange = range(.y)),
  ...
)
```

Arguments

formula	formula in the format of `m1+m2 ~ y+x1+x2 | g1/g2`, where m_i's are one or more marks, y and x_j's are the endpoint and predictor(s) for downstream analysis, and g_k are one or more hierarchical grouping structure (in the fashion of parameter `random` of function lme)
data	data.frame
coords	formula, variable names of x- and y-coordinates in `data`. Default `~x+y`. End-user may use `coords = FALSE` to indicate the absence of coordinates information in `data`.
window	an observation window owin, default is the x- and y-span of `coords` in `data`.
...	additional parameters, currently not in use

Details

...

Value

Function `grouped_ppp()` returns a groupedHyperframe with **one-and-only-one** ppp-hypercolumn.

If `coords = FALSE`, then a groupedHyperframe with **one-and-only-one** `'pseudo.ppp'`-hypercolumn is returned.

Examples

```
library(survival) # to help ?spatstat.geom::hyperframe understand ?survival::Surv
grouped_ppp(hladr + phenotype ~ OS + gender + age | patient_id/image_id,
  data = wrobel_lung, mc.cores = 1L)
```

user_hyperframe *User Interface of Operations on hyperframe with One-and-Only-One ppp-hypercolumn*

Description

See workhorse functions `fv_hyperframe()` and `dist_hyperframe()`.

Usage

```
Emark_(X, correction = "none", ...)

Vmark_(X, correction = "none", ...)

markcorr_(X, correction = "none", ...)

markvario_(X, correction = "none", ...)

Gcross_(X, correction = "none", ...)

Jcross_(X, correction = "none", ...)

Kcross_(X, correction = "none", ...)

Lcross_(X, correction = "none", ...)

nncross_(X, ...)
```

Arguments

`X`	a hyperframe
`correction`	character scalar, see functions markcorr, Gcross, etc. Default `'none'` to save computing time.
`...`	additional parameters of user operation

Details

See explanations in workhorse functions `fv_hyperframe()` and `dist_hyperframe()`.

Value

See explanations in workhorse functions `fv_hyperframe()` and `dist_hyperframe()`.

Examples

```
library(spatstat.data)
library(spatstat.geom)
# no good example for [Emark_.hyperframe]
# no hyperframe with ppp-hypercolumn with numeric marks

flu$pattern[] = flu$pattern |>
 lapply(FUN = `mark_name<-`, value = 'stain') # read ?flu carefully

r = seq.int(from = 0, to = 100, by = 5)
flu |>
 subset(stain == 'M2-M1') |>
 Gcross_(i = 'M1', j = 'M2', r = r, correction = 'best', mc.cores = 1L)

flu |>
 subset(stain == 'M2-M1') |>
 nncross_(i = 'M1', j = 'M2', mc.cores = 1L)
```

Index