Package ‘rspa’

May 24, 2022

Maintainer Mark van der Loo <mark.vanderloo@gmail.com>
License GPL-3
Title Adapt Numerical Records to Fit (in)Equality Restrictions
Type Package
LazyLoad yes
Author Mark van der Loo
Description Minimally adjust the values of numerical records in a data.frame, such that each record satisfies a predefined set of equality and/or inequality constraints. The constraints can be defined using the 'validate' package. The core algorithms have recently been moved to the 'lintools' package, refer to 'lintools' for a more basic interface and access to a version of the algorithm that works with sparse matrices.
Version 0.2.6
Depends R (>= 2.13.0)
Imports graphics, stats, validate, lintools
Suggests editrules, tinytest
URL https://github.com/markvanderloo/rspa
BugReports https://github.com/markvanderloo/rspa/issues
RoxygenNote 7.2.0
Encoding UTF-8
NeedsCompilation yes
Repository CRAN
Date/Publication 2022-05-24 10:00:02 UTC

R topics documented:

rspa-package .......................................................... 2
match_restrictions .................................................. 2
remove_tag ............................................................ 4
tagged_values ...................................................... 4
tag_missing ........................................................... 5
**Description**

Given a vector $x^0$, and a set linear restrictions of the form $a_i \cdot x_i = b_i$ and/or $a_i \cdot x_i \leq b_i$ with $i = 1, 2, \ldots, m$. This package finds the nearest vector to $x^0$ (in the (weighted) euclidean sense) that satisfies all restrictions.

**Details**

Much of this package’s functionality, including algorithms for working with large, sparse problems has been moved to the lintools package. This package will serve as a front-end for application of the succesive projection algorithm for data stored in data.frame like objects.

**match_restrictions**

Alter numeric data records to match linear (in)equality constraints.

**Description**

Apply the successive projection algorithm to adjust each record in dat to satisfy a set of linear (in)equality constraints.

**Usage**

```r
match_restrictions(
  dat,
  restrictions,
  adjust = rep(TRUE, ncol(dat)),
  weight = rep(1, ncol(dat)),
  remove_tag = TRUE,
  ...
)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dat</td>
<td>A data.frame</td>
</tr>
<tr>
<td>restrictions</td>
<td>An object of class validator</td>
</tr>
<tr>
<td>adjust</td>
<td>(optional) A logical matrix of dimensions dim(dat) where TRUE indicates that a value may be adjusted. When missing, the tagged_values are used. If no tagging was applied, adjust will default to an all TRUE matrix with dimensions equal to dim(dat).</td>
</tr>
<tr>
<td>weight</td>
<td>A weight vector of length ncol(dat) or a matrix of dimensions dim(dat).</td>
</tr>
<tr>
<td>remove_tag</td>
<td>if a value position indicator is present, remove it?</td>
</tr>
</tbody>
</table>

... arguments passed to project.
Value
dat, with values adapted.

Note on inequality restrictions
All inequality restrictions of the form \( a.x < b \) are treated as \( a.x \leq b \). The idea is to project the original record \( x \) onto the boundary defined by the (in)equations. Projection on a boundary defined by a strict inequation is illdefined since the value \( b \) in the restriction \( a.x < b \) is strictly outside the valid region.

See Also
tag_missing

Examples

```r
# a very simple adjustment example
v <- validate::validator(
x + y == 10,
x > 0,
y > 0
)

# x and y will be adjusted by the same amount
match_restrictions(data.frame(x=4,y=5), v)

# One of the inequalities violated
match_restrictions(data.frame(x=-1,y=5), v)

# Weighted distances: 'heavy' variables change less
match_restrictions(data.frame(x=4,y=5), v, weight=c(100,1))

# if w=1/x0, the ratio between coefficients of x0 stay the same (to first order)
x0 <- data.frame(x=4,y=5)
x1 <- match_restrictions(x0, v, weight=1/as.matrix(x0))
x0[,1]/x0[,2]
x1[,1] / x1[2]

# example of tag usage
v <- validate::validator(x + y == 1, x>0,y>0)
d <- data.frame(x=NA,y=0.5)
d <- tag_missing(d)
# impute
d[1,1] <- 1

# only the tagged values will be altered. The tag is
# removed afterwards.
match_restrictions(d,v)
```
remove_tag  Remove cell position tags

Description
Remove cell position tags

Usage
remove_tag(dat, ...)

Arguments
- dat [data.frame]
- ... Currently not used

Value
dat with tag removed

See Also
Other tagging: tag_missing(), tagged_values()

tagged_values  Retrieve tagged cell positions

Description
Retrieve tagged cell positions

Usage
tagged_values(dat, ...)

Arguments
- dat [data.frame]
- ... Currently not used

Value
A logical matrix, or NULL

See Also
Other tagging: remove_tag(), tag_missing()
**tag_missing**

Tag currently missing elements of a data.frame

---

**Description**

Attach an attribute that marks which cells are empty (NA).

**Usage**

```r
tag_missing(dat, ...)
```

**Arguments**

- `dat` [data.frame] to be tagged
- `...` Currently not used.

**Value**

`dat`, tagged for missing values.

**See Also**

Other tagging: `remove_tag()`, `tagged_values()`
Index

* tagging
  - remove_tag, 4
  - tag_missing, 5
  - tagged_values, 4

match_restrictions, 2

project, 2

remove_tag, 4, 4, 5
rspa-package, 2

tag_missing, 3, 4, 5
tagged_values, 2, 4, 4, 5

validator, 2