Package ‘longCatEDA’

April 18, 2017

Type Package
Title Package for Plotting Categorical Longitudinal and Time-Series Data
Version 0.31
Date 2017-03-28
Author Stephen Tueller. Funded by the National Institute on Drug Abuse (NIDA) Award number 1R03DA030850, the National Institute on Alcohol Abuse and Alcoholism (NIAAA) Award Number R03 AA019775, and the National Institute of Justice Award Number 2011-RY-BX-0003.
Maintainer Stephen Tueller <stueller@rti.org>
Description Methods for plotting categorical longitudinal and time-series data by mapping individuals to the vertical space (each horizontal line represents a participant), time (or repeated measures) to the horizontal space, categorical (or discrete) states as facets using color or shade, and events to points using plotting characters. Sorting individuals in the vertical space and (or) stratifying them by groups can reveal patterns in the changes over time.
License GPL (>= 3)
Depends methods, stats
Suggests RColorBrewer, colorspace, MASS, ggplot2
LazyData TRUE
NeedsCompilation no
Repository CRAN
Date/Publication 2017-04-18 12:49:57 UTC

R topics documented:

longCatEDA-package ........................................... 2
alignTime ..................................................... 3
clean.events .................................................. 5
colChoose .................................................... 6
eexample2cat .................................................. 8
eexample2cont .............................................. 8
Plot Categorical Longitudinal and Time-Series Data

Description

Package to implement horizontal line plots for categorical (ordinal or nominal) longitudinal or time-series data. This is done by mapping individuals to the vertical space (each horizontal line represents a participant), time (or repeated measures) to the horizontal space, categorical (or discrete) states as facets using color or shade, and events to points. Sorting individuals in the vertical space can reveal patterns in changes over time.

Details

Package: longCatEDA
Type: Package
Version: 0.31
Date: 29Oct2016
License: GNU GPL
Depends: methods

Example usage is longCatPlot(longCat(y)) where y is a matrix or data frame in wide format with participants in rows and repeated observations in columns. The function longCat returns an object of class longCat which can be plotted using longCatPlot. Options for sorting and/or stratifying by groups are implemented using sorter which returns a sorted and/or stratified object of class longCat.

Author(s)

Stephen Tueller
Maintainer: Stephen Tueller <stueller@rti.org>
alignTime

References


Examples

par(bg='cornsilk3')
longCatPlot( longCat( example3 ) )
par(bg='transparent')

alignTime

Align time data at a given state or event

Description

Align time and event.times at a given state in y or event. All options can also be called via longCatPlot.

Usage

alignTime(y, times,
   which.state=NULL, nth.state=NULL, not.state=FALSE,
   events=NULL, event.times=NULL,
   which.event=NULL, nth.event=NULL, not.event=FALSE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>see y for longCat.</td>
</tr>
<tr>
<td>times</td>
<td>see times for longCat.</td>
</tr>
<tr>
<td>which.state</td>
<td>the state in y on which to align times (and if given, event.times). If which.state is given, which.event must be NULL.</td>
</tr>
<tr>
<td>nth.state</td>
<td>the nth occurrence of which.state on which to align times (and if given, event.times). If nth.state is given, nth.event must be NULL.</td>
</tr>
<tr>
<td>not.state</td>
<td>instead of aligning at the nth state, align at the nth non-instance of which.state.</td>
</tr>
<tr>
<td>events</td>
<td>see events for longCat.</td>
</tr>
<tr>
<td>event.times</td>
<td>see events.times for longCat.</td>
</tr>
<tr>
<td>which.event</td>
<td>the event in events on which to align times and event.times. If which.event is given, which.state must be NULL.</td>
</tr>
<tr>
<td>nth.event</td>
<td>the nth occurrence of which.event on which to align times and event.times. If nth.event is given, nth.state must be NULL.</td>
</tr>
<tr>
<td>not.event</td>
<td>instead of aligning at the nth event, align at the nth non-instance of which.event.</td>
</tr>
</tbody>
</table>
**Value**

alignTime returns a list with two objects:

- `aligned.times`: The times matrix aligned at the \( n \)th state of which state.
- `aligned.event.times`: The event times matrix aligned at the \( n \)th event of which event.

**Author(s)**

Stephen Tueller

**References**


**See Also**

- `longCatPlot` to plot longCat objects created by the `longCat` function.

**Examples**

```r
# illustrate individually varying times of observation
set.seed(642531)
y <- matrix(sample(1:5, 500, replace=TRUE), 100, 5)
set.seed(963854)
times <- matrix(runif(600, 1, 3), 100, 6)
# times must be cumulative
times <- t(apply(times, 1, cumsum))
# align at the 2nd instance of state 3
times23 <- alignTime(y, times, which.state=3, nth.state=2)$aligned.times
# plotting
labels <- c('Street', 'Drug Tx', 'Jail', 'Prison', 'Unknown')
lc <- longCat(y, times=times, Labels=labels)
lc23 <- longCat(y, times=times23, Labels=labels)
par(mfrow=c(3,1), bg='cornsilk3')
longCatPlot(lc, main='Raw Times', legendBuffer=.5, ylab='')
longCatPlot(lc23, main='Aligned: 2nd Instance of Jail (via alignTime)',
           xlab='Days Since 2nd Instance of Jail (via alignTime)',
           legendBuffer=.5, ylab='')
# repeat calling alignment from longCatPlot, not quite identical due to sorting
# on unaligned data
longCatPlot(lc, main='Aligned: 2nd Instance of Jail (via longCatPlot)', legendBuffer=.5,
           which.state=3, nth.state=2, ylab='', xlab='Days Since 2nd Instance of Jail')
par(mfrow=c(1,1), bg='transparent')

# illustrate the adding event indicators
set.seed(45962)
events <- matrix(sample(1:3, 200, replace=TRUE), 100, 2)
set.seed(23498)
```
clean.events

```r
event.times <- matrix(sample(c(times), 200, replace=FALSE), 100, 2)
# align at the 1st instance of event 2
alignedTimes <- alignTime(y, times, events=events, event.times=event.times,
which.event=2, nth.event=1)
times12 <- alignedTimes$aligned.times
event.times12 <- alignedTimes$aligned.event.times
# plotting
eventLabels=c('Arrest', 'Drug Test', 'Hearing')
lc <- longCat(y, times=times, Labels=labels,
   events=events, event.times=event.times,
   eventLabels=eventLabels)
lc12 <- longCat(y, times=times12, Labels=labels,
   events=events, event.times=event.times12,
   eventLabels=eventLabels)
par(mfrow=c(2,1), bg='cornsilk3', mar=c(5.1, 4.1, 4.1, 12.1), xpd=TRUE)
cols <- longCatPlot(lc, legendBuffer=.5,
   main='Superimpose Events', ylab='')
cols <- longCatPlot(lc12, , legendBuffer=.5,
   main='Aligned: 1st Drug Test',
   xlab='Days Since 1st Drug Test', ylab='')
legend(15.5, 50, legend=lc$eventLabels, pch=1:length(lc$eventLabels))
par(mfrow=c(1,1), bg='transparent', mar = c(5, 4, 4, 2) + 0.1, xpd=FALSE)
```

---

clean.events
clean.events

**Description**

A helper function for `longCatEDA` which cleans out missing data from events and `event.times`.

**Usage**

```r
clean.events(events, event.times)
```

**Arguments**

- `events` see `longCatEDA`
- `event.times` see `longCatEDA`

**Author(s)**

Stephen Tueller
Internal Function for Selecting Color Schemes Used by longCatPlot

Description

Internal function used by longCatPlot.

Usage

colChoose(colScheme, nFactors, reverse = FALSE)

Arguments

colScheme can be one of
- 'gray' = a grayscale spectrum
- 'rainbow' see rainbow
- 'heat' see heat.colors
- 'terrain' see terrain.colors
- 'topo' see topo.colors
- 'cm' see cm.colors

No default is given in the function definition, but the default in longCatPlot passed to colChoose is 'heat'.

nFactors see nFactors in values returned by longCat.

reverse logical - should color scheme be applied in reverse order to the levels of the categorical variable? Default is FALSE. See reverse input to longCatPlot.

Author(s)

Stephen Tueller

References


See Also

longCatPlot.
Examples

# color examples
par(mfrow=c(2,3), bg='wheat')
times <- c(1,100,200,300,400,500,600)
longCatPlot(f3lc, main='colScheme=gray', colScheme='gray',
  lwd=1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='colScheme=rainbow', colScheme='rainbow',
  lwd=1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='colScheme=heat', colScheme='heat',
  lwd=1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='colScheme=terrain', colScheme='terrain',
  lwd=1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='colScheme=topo', colScheme='topo',
  lwd=1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='colScheme=cm', colScheme='cm',
  lwd=1, ylab='', legendBuffer = .25)
par(mfrow=c(1,1), bg='transparent')

# Not run:
# illustrate the use of colors from the package RColorBrewer
library(RColorBrewer)
par(mfrow=c(2,3), bg='cornsilk3')
longCatPlot(f3lc, main='RColorBrewer: Blues', cols=brewer.pal(f3lc$nFactors, "Blues"),
  lwd=1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='RColorBrewer: Greens', cols=brewer.pal(f3lc$nFactors, "Greens"),
  lwd=1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='RColorBrewer: PuBuGn', cols=brewer.pal(f3lc$nFactors, "PuBuGn"),
  lwd=1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='RColorBrewer: Y1OrRd', cols=brewer.pal(f3lc$nFactors, "Y1OrRd"),
  lwd=1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='RColorBrewer: Spectral', cols=brewer.pal(f3lc$nFactors, "Spectral"),
  lwd=1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='RColorBrewer: Accent', cols=brewer.pal(f3lc$nFactors, "Accent"),
  lwd=1, ylab='', legendBuffer = .25)
par(mfrow=c(1,1), bg='transparent')

# illustrate the use of colors from the package colorspace
library(colorspace)
par(mfrow=c(2,3), bg='cornsilk3')
longCatPlot(f3lc, main='colorspace: rainbow_hcl', cols=rainbow_hcl(f3lc$nFactors),
  lwd=1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='colorspace: sequential_hcl', cols=sequential_hcl(f3lc$nFactors),
  lwd=1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='colorspace: heat_hcl', cols=heat_hcl(f3lc$nFactors),
  lwd=1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='colorspace: terrain_hcl', cols=terrain_hcl(f3lc$nFactors),
  lwd=1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='colorspace: diverge_hcl', cols=diverge_hcl(f3lc$nFactors),
  lwd=1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='colorspace: diverge_hsv', cols=diverge_hsv(f3lc$nFactors),
  lwd=1, ylab='', legendBuffer = .25)
Example data for `longCatPlot`.

**Description**
Simulated data for illustrating `longCatPlot`.

**Usage**
```
example2cat
```

**Format**
Data frame with 20 subjects and 6 time points

---

Simulated Data for Illustrating `longContPlot`

**Description**
Simulated data for illustrating `longContPlot`.

**Usage**
```
example2cont
```

**Format**
Data frame with 20 subjects and 6 time points
Simulated Data for Illustrating longCat and longCatPlot

Description
Simulated data for illustrating longCat and longCatPlot.

Usage
example3

Format
Data frame with 100 subjects and 6 time points

References

levelCheck
Function to Check Factor Levels

Description
Internal function of checking the levels of y called by longCat.

Usage
levelCheck(y)

Arguments
y a data matrix, see documentation for y in longCat.

Author(s)
Stephen Tueller

References

See Also
longCat.
Creation of Objects of Class longCat

Description

Function to create objects of class longCat.

Usage

longCat(y, times = NULL, Labels = NULL, tLabels = NULL, id = NULL,
       events = NULL, event.times = NULL, eventLabels = NULL)

Arguments

y

a data matrix or data frame of numeric states in wide (as opposed to long) format with cases in rows and repeated observations in columns. It is recommended that y have 9 or fewer unique non-missing levels. Labels for the numeric states are given in Labels.

times

The times object designates start and stop points for each plotted interval. It is either a vector with length being the number of columns in y plus one, NULL, or a matrix with the same number of rows as y and one more column than in y. Negative values are allowed such as would be the case if time is centered at an intervention point, negative values represent times prior to the intervention, and positive times represent times after the intervention.

If times is a vector, it is assumed that cases in each row in y is observed at the same time points for the same durations. For example, if times=c(0, 6, 12, 13), this indicates a design were cases were observed at 0, 6, and 12 time units. When applying longCatPlot, the first observation for each case with extend from 0 to 6, the second observation for each case will extend from 6 to 12, and the third observation will extend from 12 to 13. The value selected at the end may be arbitrary as cases may not have been followed for any additional time. A value may be selected to maintain consistent interval sizes. Continuing the example, one could use times=c(0, 6, 12, 18) instead of times=c(0, 6, 12, 13) even though cases weren’t actually followed past month 12. Unequal spacing is allow, for example times=c(0, 3, 12, 18). In this case, participants are observed for at baseline, 3 time units, and 12 time units, where the final status is either intentionally or arbitrarily extended 18-12=6 time units at the right end of the plot. Missing values are not allowed.

If times=NULL (the default), times=0:ncol(y) will be assigned (i.e., starting at 0 times units and increase to 1 time unit, 2 time units, etc.).

When times is a matrix, each case has a unique set of observation times. In this case, missing values are allowed, but each case should have at least one observation and a start and stop point for that observation. If available timing data is a matrix of the same size as y and represents start points, the user must add a column at the end designating how far to the right time points should be extended. As noted above, this value may be arbitrary and should be large.
longCat returns an object of class longCat which is a list containing at least the following components:

- `y` is the response matrix.
- `y.sorted` is `y` sorted (default is NULL unless `sorter` has been applied to the longCat object).
- `dim` is the dimension of `y`.
- `times` is the times object as described above.
endt

times.sorted

endt.sorted

labels

tLabels

factors

IndTime

nFactors

sorted

ascending

group

groupLabels

order.y

events

event.times

eventLabels

Author(s)

Stephen Tueller

References

See Also

longCatPlot to plot longCat objects created by the longCat function.

Examples

# create the longcat object similar to Figure 2 in Tueller (2016)
times <- c(1,100,200,300,400,500,600)
f2lc <- longCat(example2cat, times)

# object summary
summary(f2lc)

# compare growth curves to longcat
par(mfrow=c(1,2), bg='cornsilk3')
longContPlot(example2cat, times, ylim=c(1,5),
    main='Growth Curves', ylab='', xlab='Days')
longCatPlot(f2lc, lwd=4, main='Horizontal Line Plot', colScheme='heat', legendBuffer=.2)
par(mfrow=c(1,1), bg='transparent')

# illustrate individually varying times of observation
set.seed(642531)
y <- matrix(sample(1:5, 500, replace=TRUE), 100, 5)
set.seed(953854)
times <- matrix(runif(500, 1, 3), 100, 6)
# times must be cumulative
times <- t(apply(times, 1, cumsum))
lc <- longCat(y, times=times)
par(mfrow=c(1,1), bg='cornsilk3', mar=c(5.1, 4.1, 4.1, 10.1), xpd=TRUE)
cols <- longCatPlot(lc, legendBuffer=0, groupBuffer=0,
    main='Individually Varying Times of Observation')
legend(15.5, 100, legend=lc$Labels, lty=1, col=cols, lwd=2)
par(bg='transparent', mar = c(5, 4, 4, 2) + 0.1, xpd=FALSE)

# illustrate the adding event indicators
set.seed(45962)
events <- matrix(sample(1:3, 200, replace=TRUE), 100, 2)
set.seed(23498)
etevent.times <- matrix(sample(c(times), 200, replace=FALSE), 100, 2)
labels <- c('Street', 'Drug Tx', 'Jail', 'Prison', 'Unknown')
eventLabels=c('Arrest', 'Drug Test', 'Hearing')
lc <- longCat(y, times=times, Labels=labels,
    events=events, event.times=evento.times,
    eventLabels=eventoLabels)
par(mfrow=c(1,1), bg='cornsilk3', mar=c(5.1, 4.1, 4.1, 12.1), xpd=TRUE)
cols <- longCatPlot(lc, legendBuffer=0, groupBuffer=0,
    main='Superimpose Events Over States')
legend(15.5, 100, legend=lc$Labels, lty=1, col=cols, lwd=2)
legend(15.5, 40, legend=lc$eventLabels, pch=1:length(lc$eventLabels))
par(bg='transparent', mar = c(5, 4, 4, 2) + 0.1, xpd=FALSE)

## Not run:
# illustrate handling non time-ordered input (e.g., factor analysis data)
longCat-class

Description

An object of class longCat.

Objects from the Class

Objects can be created by calls of the form new("longCat", ...).

Slots

y: Object of class "matrix" --
y.sorted: Object of class "matrix" --
dim: Object of class "integer" --
times: Object of class "matrix" --
times.sorted: Object of class "matrix" --
Labels: Object of class "character" --
factors: Object of class "numeric" --
IndTime: Object of class "logical" --
nfactors: Object of class "integer" --
sorted: Object of class "logical" --
ascending: Object of class "logical" --
group: Object of class "matrix" --
group.sorted: Object of class "matrix" --
groupLabels: Object of class "character" --
order.y: Object of class "matrix" --
order.y.sorted: Object of class "matrix" --
events: Object of class "matrix" --
event.times: Object of class "matrix" --
events.sorted: Object of class "matrix" --
event.times.sorted: Object of class "matrix" --
eventLabels: Object of class "character" --

Methods

summary signature(object = "longCat"): ...

References


Examples

showClass("longCat")

longCatPlot

Plotting of lc objects

Description

Function to plot longCat objects created by longCat.

Usage

longCatPlot(lc, xlab = "Days",
    ylab = NULL, cols = NULL,
    colScheme = "heat", reverse = FALSE, lwd = 0.5, lcex = 1, l1wd = 3,
    legendBuffer = 0.12, groupBuffer = 0, groupRotation = 90, gcex = 1,
    seg.len = 1, xlas = 0, xecx = 1, ececx = .5, event.col=1,
    plot.events=TRUE, which.events=NULL,
    n.events=NULL, event.pch=NULL,
    texclude=NULL, sort=TRUE,
    which.state=NULL, nth.state=NULL, not.state=FALSE,
    which.event=NULL, nth.event=NULL, not.event=FALSE, ...)
Arguments

lc an object of class longCat created by longCat. See par.
xlab a label for the x-axis. Default is "Days". See par.
ylab a label for the y-axis. Default is NULL which is changed to "Each Line Represents a Participant" with the sample size appended (lc$dim[1], see longCat). See par.
cols a numeric or character list of colors. See par. Default is NULL. To use internal color schemes, use colScheme.
colScheme select a color scheme. See colChoose for available options.
reverse color schemes are applied from the lowest to highest level of categorical data in lc$y or lc$y.sorted. Set reverse=TRUE to reverse this. Default is FALSE.
lwd set the width of horizontal lines. Default is .5. lwd should be reduced proportionally to the number of rows in lc$y to avoid overlap in plotting.
lcex character expansion factor for the legend text. Default is 1. See par.
l1wd set the width of lines in the legend. default is 3. See lwd.
legendBuffer set proportion of the plot to retain for legends, must be in [0,1]. Note that the legend is very sensitive to the scaling of the graphics device. Users are advised to maximize their device and rerun, or call dev.new() and resize prior to running longCatPlot. Default is .12 (i.e., 12% of the vertical plot area is retained for the legend). Set to 0 if no legend is desired. See the examples for moving the legend outside of the plotting margins.
groupBuffer similar to legendBuffer, but for group labels on the left side of the plot. Default is 0 (i.e., 0% of the horizontal plot area is retained for group labels). Can take on any value in [0,1].
groupRotation if lc$groupLabels are long (see longCat), rotation of the labels can be used reduce the needed size of groupBuffer. The value is in degrees ranging from -360 to +360.
gcex character expansion factor for group labels. Default is 1. See par.
seg.len Length of lines in the upper legend. Default is 1. See legend.
xlas see las in par. Applied to the x-axis when tLabels are provided to longCat.
xcex see axis.cex in par and axis. Applied to the x-axis when tLabels are pro-

dvided to longCat.
ecex see cex in points. This is used to size the points used to plot event points if events is not NULL.
event.col color for plotting event indicators.
plot.events logical - should events be plotted?
which.events numeric vector - which events should be plotted? For example, if the events have values c(1,2,3,4,5), you can specify that only which.events=c(2,5) be plotted.
n.events how many events should be plotted, e.g., n.events=3 plots the first three events for each participant.
event.pch what plotting characters should be used. See points.
longCatPlot

texclude

A vector of length 2 indicating the range of times and event times to be plotted, e.g., texclude=c(10,20) will plot data for time points 10 and larger up to and including 20.

sort

logical - should longCatPlot sort the data on the fly using intelligent defaults? If data are already sorted via sorter, this will be ignored.

which.state

see alignTime.

nth.state

see alignTime.

not.state

see alignTime.

which.event

see alignTime.

nth.event

see alignTime.

not.event

see alignTime.

... Arguments to be passed to plot (see par).

Author(s)

Stephen Tueller

References


See Also

longCat.

Examples

# Illustrate longCatPlot with the legend outside the plot
par(mfrow=c(1,1), bg='cornsilk3', mar=c(5.1, 4.1, 4.1, 8.1), xpd=TRUE)
cols <- longCatPlot(  
  longCat(example3),  
  legendBuffer=0,  
  main='Horizontal Line Plot')
legend(7.1, 100, legend=1:5, col=cols, lty=1, lwd=2)
par(bg='transparent', mar = c(5, 4, 4, 2) + 0.1, xpd=FALSE)

### visualizing multivariate data: 3 items at 4 time points
library(MASS)
Sigma <- matrix(.25, 12, 12)
diag(Sigma) <- 1
set.seed(9845)
mu <- rep(c(-.5, 0,.5), 4) + rnorm(12, 0,.25)
set.seed(539)
ymv <- apply(mvrnorm(n=100, mu=mu, Sigma = Sigma), 2, cut, breaks=c(-Inf, 0, Inf), labels=c(0,1))
apply(ymv, 2, table)
(items <- rep(1:3, 4))
times <- sort(rep(1:4, 3)))
tLabels <- paste('Time', 1:4)
Labels <- paste('Item', 1:3)

# plot time points within items
par(mfrow=c(2,2), bg='cornsilk3', mar=c(5.1, 4.1, 4.1, 8.1), xpd=TRUE)

time1 <- longCat(y=ymv[,items==1], tLabels=tLabels)
cols <- longCatPlot(time1, ylab='', main='Item 1', legendBuffer=0, xlab='', xlas=2)
legend(length(unique(times))+.1, nrow(ymv), legend=0:1, col=cols, lty=1, lwd=2, title='Response')

time2 <- longCat(y=ymv[,items==2], tLabels=tLabels)
longCatPlot(time2, ylab='', main='Item 2', legendBuffer=0, xlab='', xlas=2)
legend(length(unique(times))+.1, nrow(ymv), legend=0:1, col=cols, lty=1, lwd=2, title='Response')

time3 <- longCat(y=ymv[,items==3], tLabels=tLabels)
longCatPlot(time3, ylab='', main='Item 3', legendBuffer=0, xlab='', xlas=2)
legend(length(unique(times))+.1, nrow(ymv), legend=0:1, col=cols, lty=1, lwd=2, title='Response')

time4 <- longCat(y=ymv[,items==4], tLabels=tLabels)
cols <- longCatPlot(time4, ylab='', main='Item 4', legendBuffer=0, xlab='', xlas=2)
legend(length(unique(times))+.1, nrow(ymv), legend=0:1, col=cols, lty=1, lwd=2, title='Response')

par(bg='transparent', mar = c(5, 4, 4, 2) + 0.1, xpd=FALSE)

# plot items within time points
par(mfrow=c(2,2), bg='cornsilk3', mar=c(5.1, 4.1, 4.1, 8.1), xpd=TRUE)

time1 <- longCat(y=ymv[,times==1], tLabels=Labels)
cols <- longCatPlot(time1, ylab='', main='Time 1', legendBuffer=0, xlab='')
legend(length(unique(times))+.1, nrow(ymv), legend=0:1, col=cols, lty=1, lwd=2, title='Response')

time2 <- longCat(y=ymv[,times==2], tLabels=Labels)
cols <- longCatPlot(time2, ylab='', main='Time 2', legendBuffer=0, xlab='')
legend(length(unique(times))+.1, nrow(ymv), legend=0:1, col=cols, lty=1, lwd=2, title='Response')

time3 <- longCat(y=ymv[,times==3], tLabels=Labels)
cols <- longCatPlot(time3, ylab='', main='Time 3', legendBuffer=0, xlab='')
legend(length(unique(times))+.1, nrow(ymv), legend=0:1, col=cols, lty=1, lwd=2, title='Response')

time4 <- longCat(y=ymv[,times==4], tLabels=Labels)
cols <- longCatPlot(time4, ylab='', main='Time 4', legendBuffer=0, xlab='')
legend(length(unique(times))+.1, nrow(ymv), legend=0:1, col=cols, lty=1, lwd=2, title='Response')

par(mfrow=c(1,1), bg='transparent', mar = c(5, 4, 4, 2) + 0.1, xpd=FALSE)

## Not run:
## for data sets with many rows, writing directly to a file
## is much faster and unaffected by device resizing, see ?pdf
df('C:/mydir/mysubdir/myfile.pdf')
par(bg='cornsilk3')
longCatPlot(f3lc, main='Sorted', colScheme='heat', lwd=2)
par(mfrow=c(1,1), bg='transparent')
dev.off()
## see ?jpeg for picture file options

## End(Not run)
longContPlot  

Plot Continuous Longitudinal Data

Description

Function to plot continuous longitudinal or time-series data.

Usage

longContPlot(y, times = NULL, jog=FALSE, ylim = NULL, xlim = NULL, ...)  

Arguments

y  
a data matrix or data frame in wide (as opposed to long) format with cases in rows and repeated observations in columns.

times  
time points used for the x-axis in plotting. Either a vector of the same length as the number of columns in y (i.e., all cases have the same times of observation), or a matrix of the same dimension as y (i.e., individually varying times of observation). Default is NULL and is assigned the value 1:ncol(y).

jog  
When y is integer data, it can be useful to jog all values by a small amount. When jog=TRUE, a random uniform variate in [-.25, .25] is added to each row in y.

ylim  
see par. Default is NULL and calculated from y.

xlim  
see par. Default is NULL and calculated from y.

...  
Arguments to be passed to plot. See par.

Author(s)

Stephen Tueller

References


See Also

longCatPlot.
Examples

# longitudinal plot
times <- c(1,100,200,300,400,500)
par(mfrow=c(1,1), bg='cornsilk3')
longContPlot(example2cont, times, ylim=c(-2,6), main='', ylab='', xlab='Day')
par(mfrow=c(1,1), bg='transparent')

# jogging example
times <- c(1,100,200,300,400,500)
par(mfrow=c(1,2), bg='cornsilk3')
longContPlot(example2cat, times, ylim=c(0,6),
             main='Growth Curves', ylab='', xlab='Days')
longContPlot(example2cat, times, jog=TRUE, ylim=c(0,6),
             main='Growth Curves + Jogging',
             ylab='', xlab='Days')
par(mfrow=c(1,1), bg='transparent')# compare growth curves to longCat

---

**lunique**

*Find the Unique Number of Factors*

**Description**

Function to find the unique number of factors in a matrix or data frame of categorical data. Used internally by `longCat`.

**Usage**

lunique(y)

**Arguments**

- **y**
  
  See `y` for `longCat`.

**Author(s)**

Stephen Tueller

**References**

**makePatterns**

*Concatenate Multivariate Data into Numeric or Character Patterns*

**Description**

Function to concatenate the columns of a matrix or data frame for each row into a single character variable, which can optionally be reconverted to numeric. Called internally by `sorter`. For example, a row of a matrix containing c(1, 2, 3, 5) will be concatenated to "1235".

**Usage**

```r
makePatterns(dat, times, num = TRUE, mindur = NULL, igrpt = FALSE)
```

**Arguments**

- `dat`:
  a matrix or data frame such as `lc$y` from an `longCat` object created by `longCat`.

- `times`:
  see times in `longCat`.

- `num`:
  logical indicator, should a numeric version of the concatenate rows be return. Default is TRUE. When num=TRUE, the return is rescaled by moving a decimal point between the first and second digits. This ensures that, under different numbers of observations or missing data, ordering is not unduly impacted by patterns of missing data. Users are encouraged to try sorting with num=TRUE and num=FALSE when experimenting to find the sorting that leads to the clearest plot. When lc$sorted=FALSE and there is no missing data in lc$y and lc$IntTime=FALSE, longCatPlot will change num to FALSE.

- `mindur`:
  minimum duration. If times is a matrix or data frame of individually varying times of observation of the same dimension as dat, selecting mindur > 0 results in all cells in y corresponding to cells in times - times[1,] < mindur being changed to NA (where times - times[1] changes the times from a matrix of observed times to a matrix of durations for each state in dat). This minimizes the effect of short durations on the sorting algorithm in `sorter`. Default is NULL.

- `igrpt`:
  Option to ignore repeated values when sorting, allowing the sorting algorithm in `sorter` to smooth over regions of no change for each row in lc$y. Default is FALSE. See `norpt`.

**Value**

- `out`:
  A vector of patterns of length nrow(dat)

**Author(s)**

Stephen Tueller
norpt

A function to take out repeated observations in a vector of data for sorting purposes.

Description

See the example.

Usage

norpt(alist = c(1, 2, 2, 3, 3, 4, 4, 4, 4, 5))

Arguments

alist

Value

A vector excluding repeated values with trailing NA’s to fill the vector to original length.

Author(s)

Stephen Tueller
References

See Also
igrpt input to makePatterns

Examples
(alist = c(1,2,2,3,3,4,4,4,4,5))
norpt(alist)

Description
Helper function for sorter, doing within group sorting with sorter looping through groups (even if there is only one group) using sort1.

Usage

sort1(id1, y1, times1, events1, event.times1, group1, ascending = TRUE, whichColumns = NULL, initFirst = FALSE)

Arguments

id1 The identification variable for one group. See also id for longCat.
y1 The longitudinal data for one group. See also y for longCat.
times1 The time data for one group. See also times for longCat.
events1 The events data for one group. See also events for longCat.
event.times1 The event times for one group. See also event.times for longCat.
group1 The group identification variable for one group. See also group for sorter.
ascending See ascending for sorter.
whichColumns See whichColumns for sorter.
initFirst See initFirst for sorter.

Author(s)
Stephen Tueller
**sorter**

*General Sorting Function*

**Description**

A function to sort an `longCat` object created by `longCat`. `sorter` must be used directly when stratified plots of subgroups is desired, or when sorting other than the default sorting is desired. Otherwise, `sorter` is used internally with the defaults by `longCatPlot` if `lc$sort=FALSE`. If an object has already been sorted (`lc$sorted=TRUE`), `sorter` will not resort it, but will print a code example of how to use multiple sortings.

**Usage**

```r
sorter(lc, ascending = TRUE, whichColumns = NULL, num = TRUE,
       mindur = NULL, igrpt = FALSE, customSort = NULL,
       initFirst = FALSE, group = NULL, groupLabels = NULL,
       ggap = NULL)
```

**Arguments**

- **lc**: an object of class `longCat` created by `longCat`.
- **ascending**: logical - should sorting be done ascending. Default is `TRUE`.
- **whichColumns**: a numeric list indicating which columns in `lc$y` should be used for sorting (e.g., `c(1, 5, 7)`). Useful if, for example, an intervention occurs after data collection has started, and the user is not interested in sorting on pre-intervention observations.
- **num**: see `makePatterns` for details.
- **mindur**: see `makePatterns`.
- **igrpt**: should `sorter` (i)gnore (r)e(p)ea(t)ed values for each row in `lc$y` for sorting purposes? See `norpt`.
- **customSort**: a vector of the same length as the number of rows in `lc$y` providing a user defined variable on which to sort the data prior to secondarily applying the default sort. If group is not `NULL`, group will be sorted on prior to the `customSort` variable. Alternatively, `lc$y` can be sorted without calling `sorter` using `lc$y.sorted <- lc$y[o, ]` where `o` is the `order` (e.g., use `o <- order(customSort)`). The user must also set `lc$sorted <- TRUE` to prevent on-the-fly default sorting from being carried out by `longCatPlot`. Users unfamiliar with sorting in R should take care not to confuse `order` with `sort`. Default is `NULL`. If any values on `customSort` are missing, the function will return an error message.
- **initFirst**: if `customSort` is not `NULL`, setting `initFirst=TRUE` will sort on initial values prior to the custom sorting variable.
- **group**: a vector of the same length as the number of rows in `lc$y` indicating group membership. Default is `NULL`. If group is `NA`, corresponding rows in `lc$y` will be deleted prior to completing the sorting, and a warning indicating this has been
done is printed to the console. If a large number of cases have missing data on
the grouping variable, consider recoding the missings into their own group, e.g.,
group[is.na(group)] <- -999 and add a missing label to groupLabels, e.g.
groupLabels=c("Missing", "Group1", "Group2", 'Etc.').

**groupLabels**

A vector of numeric or character labels of the same length as the number of
unique values in group. Default is NULL. If group is not NULL and groupLabels
is not provided, then the numeric values in group are used as the labels.

**ggap**

A number zero to 1. The proportion of blank rows to be plotted between groups
when group is specified. The default of NULL is set to 0.05 when groups are
present, 0.0 when there are no groups.

**Value**

Returns an object of class longCat where lc$sorted=TRUE. See longCat for values.

**Author(s)**

Stephen Tueller

**References**

Tueller, S. J., Van Dorn, R. A., & Bobashev, G. V. (2016). Visualization of categorical longitudi-

**See Also**

longCat and longCatPlot.

**Examples**

```r
### create a plot like that in Figure 3 from Tueller, Van Dorn, & Bobashev (2016)
par(mfrow=c(1,2), bg='cornsilk3')
times <- c(1,100,200,300,400,500,600)
f3lc <- longCat(example3, times); f3lc$sorted <- TRUE; f3lc$y.sorted <- f3lc$y
longCatPlot(f3lc, main='Unsorted', colScheme='heat', lwd=2, legendBuffer=.2)
f3lc <- longCat(example3, times)
longCatPlot(f3lc, main='Sorted', colScheme='heat', lwd=2, legendBuffer=.2)

### sort with a grouping variable and plot
par(mfrow=c(1,1), bg='cornsilk3', mar=c(5.1, 4.1, 4.1, 9.1), xpd=TRUE)
times <- c(1,100,200,300,400,500,600)
lc <- longCat(example3, times)
group <- sample(1:3, nrow(example3), replace=TRUE)
grouplc <- sorter(lc, group=group, groupLabels=1:3)
cols <- longCatPlot(grouplc, groupBuffer=.15, main='Grouped Data',
                   colScheme='heat',
                   lwd=2, legendBuffer=0)
legend(610, 130, legend=1:5, col=cols, lty=1, lwd=2)
par(bg='transparent', mar = c(5, 4, 4, 2) + .1, xpd=FALSE)

### using the sorted data from the previous plot, repeat using ggplot2
```
Following the example of Figure 4 of bdemarest’s answer on [https://stackoverflow.com/questions/11513149/good-ways-to-visualize-longitudinal-categorical-data-in-r/](https://stackoverflow.com/questions/11513149/good-ways-to-visualize-longitudinal-categorical-data-in-r/)

grouplc['df' <- data.frame(id=1:nrow(grouplc$group.sorted),
    group=grouplc$group.sorted[,1], grouplc$y.sorted)
grouplc['long' <- reshape(grouplc['df',
    varying = names(grouplc$y.sorted),
    v.names = "score",
    timevar = "time",
    times = times[, ncol(grouplc$y.sorted)],
    direction = "long")
grouplc['long'['score' <- factor(grouplc['long'['score'])
grouplc['long'['group' <- factor(grouplc['long'['group', level=3:1)

# remove NA's introduced using group option in sorter
grouplc['long' <- na.omit(grouplc['long])
library(ggplot2)
ggplot(grouplc['long', aes(x=time, y=id, fill=score)) +
    geom_tile(colour="transparent") +
    scale_fill_manual(values=cols) +
    facet_grid(group ~ ., space="free_y", scales="free_y")

### sort with a grouping variable and events and plot

times <- c(1,100,200,300,400,500,600)
set.seed(45962)
events <- matrix(sample(1:3, nrow(example3)*2, replace=TRUE), nrow(example3), 2)
set.seed(23498)
event.times <- matrix(sample(min(times):max(times), nrow(example3)*2, replace=TRUE),
    nrow(example3), 2)
lables <- c('Street', 'Drug Tx', 'Jail', 'Prison', 'Unknown')
eventLabels=c('Arrest', 'Drug Test', 'Hearing')
eventlc <- longCat(example3, times=times, Labels=lables,
    events=events, event.times=event.times,
    eventLabels=eventLabels)
set.seed(4290)
groupevent <- sample(1:3, nrow(example3), replace=TRUE)
groupeventlc <- sorter(eventlc, group=groupevent)
par(mfrow=c(1,1), bg='cornsilk3', mar=c(5.1, 4.1, 4.1, 12.1), xpd=TRUE)
cols <- longCatPlot(groupeventlc, legendBuffer=0, groupBuffer=0.15,
    main='Grouping and Events')
legend(610, 130, legend=groupeventlc$Labels, lty=1, col=cols, lwd=2)
legend(610, 60, legend=groupeventlc$eventLabels,
    pch=1:length(groupeventlc$eventLabels))
par(bg='transparent', mar = c(5, 4, 4, 2) + 0.1, xpd=FALSE)

summary-methods ~~ Methods for Function `summary` ~~

**Description**

~~ Methods for function `summary` ~~~
Methods

signature(object = "ANY")
signature(object = "longCat")
## Index

<table>
<thead>
<tr>
<th>Topic</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textasciitilde kwd1</td>
<td></td>
</tr>
<tr>
<td>norpt, 22</td>
<td></td>
</tr>
<tr>
<td>\textasciitilde kwd2</td>
<td></td>
</tr>
<tr>
<td>norpt, 22</td>
<td></td>
</tr>
<tr>
<td>classes</td>
<td></td>
</tr>
<tr>
<td>longCat-class, 14</td>
<td></td>
</tr>
<tr>
<td>datasets</td>
<td></td>
</tr>
<tr>
<td>example2cat, 8</td>
<td></td>
</tr>
<tr>
<td>example2cont, 8</td>
<td></td>
</tr>
<tr>
<td>example3, 9</td>
<td></td>
</tr>
<tr>
<td>methods</td>
<td></td>
</tr>
<tr>
<td>summary-methods, 26</td>
<td></td>
</tr>
<tr>
<td>package</td>
<td></td>
</tr>
<tr>
<td>longCatEDA-package, 2</td>
<td></td>
</tr>
<tr>
<td>alignTime, 3, 11, 17</td>
<td></td>
</tr>
<tr>
<td>axis, 16</td>
<td></td>
</tr>
<tr>
<td>clean.events, 5</td>
<td></td>
</tr>
<tr>
<td>cm.colors, 6</td>
<td></td>
</tr>
<tr>
<td>colChoose, 6, 16</td>
<td></td>
</tr>
<tr>
<td>example2cat, 8</td>
<td></td>
</tr>
<tr>
<td>example2cont, 8</td>
<td></td>
</tr>
<tr>
<td>example3, 9</td>
<td></td>
</tr>
<tr>
<td>heat.colors, 6</td>
<td></td>
</tr>
<tr>
<td>legend, 16</td>
<td></td>
</tr>
<tr>
<td>levelCheck, 9</td>
<td></td>
</tr>
<tr>
<td>longCat, 2–4, 6, 9, 10, 13–17, 20, 21, 23–25</td>
<td></td>
</tr>
<tr>
<td>longCat-class, 14</td>
<td></td>
</tr>
<tr>
<td>longCatEDA, 5</td>
<td></td>
</tr>
<tr>
<td>longCatEDA (longCatEDA-package), 2</td>
<td></td>
</tr>
<tr>
<td>longCatEDA-package, 2</td>
<td></td>
</tr>
<tr>
<td>longCatPlot, 2–4, 6, 8, 9, 11–13, 15, 19, 21, 24, 25</td>
<td></td>
</tr>
<tr>
<td>longContPlot, 8, 12, 19</td>
<td></td>
</tr>
<tr>
<td>lunique, 20</td>
<td></td>
</tr>
<tr>
<td>makePatterns, 12, 21, 23, 24</td>
<td></td>
</tr>
<tr>
<td>norpt, 21, 22, 24</td>
<td></td>
</tr>
<tr>
<td>order, 24</td>
<td></td>
</tr>
<tr>
<td>par, 16, 17, 19</td>
<td></td>
</tr>
<tr>
<td>plot, 17, 19</td>
<td></td>
</tr>
<tr>
<td>points, 16</td>
<td></td>
</tr>
<tr>
<td>rainbow, 6</td>
<td></td>
</tr>
<tr>
<td>sort, 24</td>
<td></td>
</tr>
<tr>
<td>sort1, 23</td>
<td></td>
</tr>
<tr>
<td>sorter, 2, 11, 12, 21–23, 24</td>
<td></td>
</tr>
<tr>
<td>summary, ANY-method (summary-methods), 26</td>
<td></td>
</tr>
<tr>
<td>summary, longCat-method (summary-methods), 26</td>
<td></td>
</tr>
<tr>
<td>summary-methods, 26</td>
<td></td>
</tr>
<tr>
<td>terrain.colors, 6</td>
<td></td>
</tr>
<tr>
<td>topo.colors, 6</td>
<td></td>
</tr>
</tbody>
</table>