

# Package ‘longCatEDA’

October 13, 2022

**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 <stuellet@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
example2cat . . . . .	8
example2cont . . . . .	8

example3 . . . . .	9
levelCheck . . . . .	9
longCat . . . . .	10
longCat-class . . . . .	14
longCatPlot . . . . .	15
longContPlot . . . . .	19
lunique . . . . .	20
makePatterns . . . . .	21
norpt . . . . .	22
sort1 . . . . .	23
sorter . . . . .	24
summary-methods . . . . .	26

<b>Index</b>	<b>28</b>
--------------	-----------

---

longCatEDA-package	<i>Plot Categorical Longitudinal and Time-Series Data</i>
--------------------	---

---

## 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>

## References

Tueller, S. J., Van Dorn, R. A., & Bobashev, G. V. (2016). Visualization of categorical longitudinal and times series data (Report No. MR-0033-1602). Research Triangle Park, NC: RTI Press. <http://www.rti.org/publication/visualization-categorical-longitudinal-and-times-series-data>

## Examples

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

---

alignTime	<i>Align time data at a given state or event</i>
-----------	--

---

## 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

y	see y for <a href="#">longCat</a> .
times	see times for <a href="#">longCat</a> .
which.state	the state in y on which to align times (and if given, event.times). If which.state is given, which.event must be NULL.
nth.state	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.
not.state	instead of aligning at the nth state, align at the nth non-instance of which.state.
events	see events for <a href="#">longCat</a> .
event.times	see events.times for <a href="#">longCat</a> .
which.event	the event in events on which to align times and event.times. If which.event is given, which.state must be NULL.
nth.event	the nth occurrence of which.event on which to align times and event.times. If nth.event is given, nth.state must be NULL.
not.event	instead of aligning at the nth event, align at the nth non-instance of which.event.

**Value**

alignTime returns a list with two objects:

aligned.times The times matrix aligned at the nth.state of which.state.

aligned.event.times

The event.times matrix aligned at the nth.event of which.event.

**Author(s)**

Stephen Tueller

**References**

Tueller, S. J., Van Dorn, R. A., & Bobashev, G. V. (2016). Visualization of categorical longitudinal and times series data (Report No. MR-0033-1602). Research Triangle Park, NC: RTI Press. <http://www.rti.org/publication/visualization-categorical-longitudinal-and-times-series-data>

**See Also**

[longCatPlot](#) to plot longCat objects created by the [longCat](#) function.

**Examples**

```
# 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',
            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)
```

```
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

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

## Arguments

`events`            see [longCatEDA](#)  
`event.times`      see [longCatEDA](#)

## Author(s)

Stephen Tueller

---

`colChoose`*Internal Function for Selecting Color Schemes Used by longCatPlot*

---

**Description**

Internal function used by [longCatPlot](#).

**Usage**

```
colChoose(colScheme, nfactors, reverse = FALSE)
```

**Arguments**

<code>colScheme</code>	can be one of <ul style="list-style-type: none"><li>• 'gray' = a grayscale spectrum</li><li>• 'rainbow' see <a href="#">rainbow</a></li><li>• 'heat' see <a href="#">heat.colors</a></li><li>• 'terrain' see <a href="#">terrain.colors</a></li><li>• 'topo' see <a href="#">topo.colors</a></li><li>• 'cm' see <a href="#">cm.colors</a></li></ul> No default is given in the function definition, but the default in <a href="#">longCatPlot</a> passed to <code>colChoose</code> is 'heat'.
<code>nfactors</code>	see <code>nfactors</code> in values returned by <a href="#">longCat</a> .
<code>reverse</code>	logical - should color scheme be applied in reverse order to the levels of the categorical variable? Default is FALSE. See reverse input to <a href="#">longCatPlot</a> .

**Author(s)**

Stephen Tueller

**References**

Tueller, S. J., Van Dorn, R. A., & Bobashev, G. V. (2016). Visualization of categorical longitudinal and times series data (Report No. MR-0033-1602). Research Triangle Park, NC: RTI Press. <http://www.rti.org/publication/visualization-categorical-longitudinal-and-times-series-data>

**See Also**

[longCatPlot](#).

**Examples**

```

# color examples
par(mfrow=c(2,3), bg='wheat')
times <- c(1,100,200,300,400,500,600)
f3lc <- longCat( example3, times, Labels=rep('',5) )
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$nfactors, "PuBuGn"),
  lwd=.1, ylab='', legendBuffer = .25)
longCatPlot(f3lc, main='RColorBrewer: YlOrRd', cols=brewer.pal(f3lc$nfactors, "YlOrRd"),
  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)

```

```
par(mfrow=c(1,1), bg='transparent')  
## End(Not run)
```

---

example2cat

*Example data for [longCatPlot](#).*

---

### **Description**

Simulated data for illustrating [longCatPlot](#).

### **Usage**

```
example2cat
```

### **Format**

Data frame with 20 subjects and 6 time points

---

example2cont

*Simulated Data for Illustrating [longContPlot](#)*

---

### **Description**

Simulated data for illustrating [longContPlot](#).

### **Usage**

```
example2cont
```

### **Format**

Data frame with 20 subjects and 6 time points



---

`example3`*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**

Tueller, S. J., Van Dorn, R. A., & Bobashev, G. V. (2016). Visualization of categorical longitudinal and times series data (Report No. MR-0033-1602). Research Triangle Park, NC: RTI Press.

---

`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**

Tueller, S. J., Van Dorn, R. A., & Bobashev, G. V. (2016). Visualization of categorical longitudinal and times series data (Report No. MR-0033-1602). Research Triangle Park, NC: RTI Press. <http://www.rti.org/publication/visualization-categorical-longitudinal-and-times-series-data>

**See Also**

[longCat](#).

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 where cases were observed at 0, 6, and 12 time units. When applying longCatPlot, the first observation for each case will 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 allowed, 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

enough to show what state cases end in. If available timing data is a matrix of the same size as `y` and represents end points, the user must add a column at the beginning of the matrix designating start points. If available timing data represents duration instead of time points, the user must recode the data into cumulative time. For example, if a participant was in their first three states for 5, 7, and 11 time units, their row of times should be recoded to be `c(0, 5, 12, 23)`. The `times` and `event.times` matrices can be realigned using the `alignTime` function, or this can be done on the fly by accessing the `alignTime` options in the `longCatPlot` function.

Labels	a vector of numeric or character labels for the response options in <code>y</code> . Must be the same length as the number of unique non-missing values in <code>y</code> . Default is NULL and is assigned the values <code>1:max(unique(y))</code> .
tLabels	numeric or character labels for the time points in <code>times</code> . Default is NULL and is assigned the values <code>1:ncol(y)</code> .
id	An optional variable identifying or naming the rows of <code>y</code> . Returned as the first column of the matrix <code>order.y</code> (see <code>order.y</code> in the value section below).
events	An event matrix or <code>y.frame</code> which may be numeric or character (see <code>eventLabels</code> ). Whereas the data in <code>y</code> are states in which each case resides for some period of time, <code>events</code> are instantaneous events (or very short lived states) that can be attached to a single point in time at <code>event.times</code> . The number of rows in <code>events</code> and <code>event.times</code> must equal the number of rows in <code>y</code> , but can have as many columns as needed to capture all events of interest. Large numbers of events may cloud the resulting figures created by <code>longCatPlot</code> .
event.times	A matrix or <code>data.frame</code> of event times corresponding to each event in <code>events</code> . As opposed to the <code>times</code> matrix, which contains durations (except possibly the first column as described above), the <code>event.times</code> matrix is the time the event takes place (i.e., cumulative time).  The <code>times</code> and <code>event.times</code> matrices can be realigned using the <code>alignTime</code> function, or this can be done on the fly by accessing the <code>alignTime</code> options in the <code>longCatPlot</code> function.
eventLabels	If <code>events</code> is a character matrix, <code>eventLabels</code> should be left NULL and labels will be pulled from the data in <code>events</code> . If <code>events</code> is numeric, corresponding <code>eventLabels</code> can be supplied by the user as a character vector, for example, <code>c('event1', 'event2', 'etc.')</code> . The number of unique events (and corresponding event labels) should be kept small if possible, otherwise the event legend on Figures produced by <code>longCatPlot</code> may be truncated.

## Value

`longCat` returns an object of class `longCat` which is a list containing at least the following components:

<code>y</code>	<code>y</code>
<code>y.sorted</code>	<code>y</code> sorted (default is NULL unless <code>sorter</code> has been applied to the <code>longCat</code> object).
<code>dim</code>	the dimension of <code>y</code> .
<code>times</code>	the <code>times</code> object as described above.

endt	the endt object as described above.
times.sorted	if times is a matrix of the same dimension as y, times.sorted contains a matrix of individually varying times of observation with the same sorting as y.sorted.
endt.sorted	endt sorted after an lc object is passed to <a href="#">sorter</a>
labels	the labels vector as described above
tLabels	the tLabels vector as described above
factors	a vector containing the unique values in y. Not that if the unique values in y were not sequential integers starting at 1, both factors and y are recoded such that they contain sequential integers starting at 1.
IndTime	a logical indicator of whether times is a matrix of the same dimension as y. If TRUE, <a href="#">longCatPlot</a> treats these times as individually varying times of observation.
nfactors	the number of unique values in y, and is the same as the length of the factors vector. nfactors is determined by longCat. If users have data with more than 9 categories, continuous plotting methods are recommended via warning (e.g., try <a href="#">longContPlot</a> .)
sorted	a logical indicator of whether y has been sorted by the <a href="#">sorter</a> function. If TRUE, y.sorted (and times.sorted if IndTime is TRUE) will not be NULL.
ascending	logical indicator. If sorted is TRUE, this will indicate whether sorting was done ascending. (default is NULL unless <a href="#">sorter</a> has been applied to the longCat object).
group	a vector of the same length as the number of rows in y (default is NULL unless <a href="#">sorter</a> has been applied to the longCat object along with a grouping variable).
groupLabels	a optional vector of character or numeric labels for the group variable (see <a href="#">sorter</a> ).
order.y	A matrix with identification (see input id above) and sorting information. Rows of the matrix correspond to rows of y and the columns are id, and order variable, and a variable representing the unique data patterns in y. The former is returned only if id is provided to longCat. The latter two are only returned by <a href="#">sorter</a> . If only unique data patterns are desired, use <a href="#">makePatterns</a> ; see <a href="#">example(makePatterns)</a> .
events	A matrix of events.
event.times	A matrix of event times.
eventLabels	A vector of event labels.

### Author(s)

Stephen Tueller

### References

Tueller, S. J., Van Dorn, R. A., & Bobashev, G. V. (2016). Visualization of categorical longitudinal and times series data (Report No. MR-0033-1602). Research Triangle Park, NC: RTI Press. <http://www.rti.org/publication/visualization-categorical-longitudinal-and-times-series-data>

**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(963854)
times <- matrix(runif(600, 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)
event.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=event.times,
  eventLabels=eventLabels)
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)
```

```

y <- matrix(sample(c('1', '2', '3', '4', '5'), 500, replace=TRUE), 100, 5)
lc <- longCat(y)
par(mfrow=c(1,1), bg='cornsilk3', mar=c(5.1, 4.1, 4.1, 8.1), xpd=TRUE)
cols <- longCatPlot(lc, legendBuffer=0)
legend(6, 100, legend=lc$factors, lty=1, col=cols, lwd=2)
par(bg='transparent', mar = c(5, 4, 4, 2) + 0.1, xpd=FALSE)

# illustrate plotting with more than 9 categories
# (a warning is issued)
y <- matrix(sample(1:18, 500, replace=TRUE), 100, 5)
lc <- longCat(y)
par(mfrow=c(1,1), bg='cornsilk3', mar=c(5.1, 4.1, 4.1, 8.1), xpd=TRUE)
cols <- longCatPlot(lc, legendBuffer=0)
legend(6, 100, legend=lc$factors, lty=1, col=cols, lwd=2)
par(bg='transparent', mar = c(5, 4, 4, 2) + 0.1, xpd=FALSE)

## End(Not run)

```

---

longCat-class

Class "longCat"

---

## 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" ~~
eventLables: Object of class "character" ~~

```

### Methods

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

### References

Tueller, S. J., Van Dorn, R. A., & Bobashev, G. V. (2016). Visualization of categorical longitudinal and times series data (Report No. MR-0033-1602). Research Triangle Park, NC: RTI Press. <http://www.rti.org/publication/visualization-categorical-longitudinal-and-times-series-data>

### 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, llwd = 3,
  legendBuffer = 0.12, groupBuffer = 0, groupRotation = 90, gcex = 1,
  seg.len = 1, xlas = 0, xcex = 1, ecex = .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 <code>longCat</code> created by <code>longCat</code> . See <code>par</code> .
xlab	a label for the x-axis. Default is "Days". See <code>par</code> .
ylab	a label for the y-axis. Default is NULL which is changed to "Each Line Represents a Participant" with the sample size appended ( <code>lc\$dim[1]</code> , see <code>longCat</code> ). See <code>par</code> .
cols	a numeric or character list of colors. See <code>par</code> . Default is NULL. To use internal color schemes, use <code>colScheme</code> .
colScheme	select a color scheme. See <code>colChoose</code> for available options.
reverse	color schemes are applied from the lowest to highest level of categorical data in <code>lc\$y</code> or <code>lc\$y.sorted</code> . Set <code>reverse=TRUE</code> to reverse this. Default is FALSE.
lwd	set the width of horizontal lines. Default is .5. <code>lwd</code> should be reduced proportionally to the number of rows in <code>lc\$y</code> to avoid overlap in plotting.
lcex	character expansion factor for the legend text. Default is 1. See <code>par</code> .
llwd	set the width of lines in the legend. default is 3. See <code>lwd</code> .
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 <code>dev.new()</code> and resize prior to running <code>longCatPlot</code> . 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 <code>legendBuffer</code> , 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 <code>lc\$groupLabels</code> are long (see <code>longCat</code> ), rotation of the labels can be used reduce the needed size of <code>groupBuffer</code> . The value is in degrees ranging from -360 to +360.
gcex	character expansion factor for group labels. Default is 1. See <code>par</code> .
seg.len	Length of lines in the upper legend. Default is 1. See <code>legend</code> .
xlas	see <code>las</code> in <code>par</code> . Applied to the x-axis when <code>tLabels</code> are provided to <code>longCat</code> .
xcex	see <code>axis.cex</code> in <code>par</code> and <code>axis</code> . Applied to the x-axis when <code>tLabels</code> are provided to <code>longCat</code> .
ecex	see <code>cex</code> in <code>points</code> . 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 <code>c(1,2,3,4,5)</code> , you can specify that only <code>which.events=c(2,5)</code> be plotted.
n.events	how many events should be plotted, e.g., <code>n.events=3</code> plots the first three events for each participant.
event.pch	what plotting characters should be used. See <code>points</code> .



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 <a href="#">alignTime</a> .
nth.state	see <a href="#">alignTime</a> .
not.state	see <a href="#">alignTime</a> .
which.event	see <a href="#">alignTime</a> .
nth.event	see <a href="#">alignTime</a> .
not.event	see <a href="#">alignTime</a> .
...	Arguments to be passed to <a href="#">plot</a> (see <a href="#">par</a> ).

**Author(s)**

Stephen Tueller

**References**

Tueller, S. J., Van Dorn, R. A., & Bobashev, G. V. (2016). Visualization of categorical longitudinal and times series data (Report No. MR-0033-1602). Research Triangle Park, NC: RTI Press. <http://www.rti.org/publication/visualization-categorical-longitudinal-and-times-series-data>

**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)

item1 <- longCat(y=y_mv[,items==1], tLabels=tLabels)
cols <- longCatPlot(item1, ylab='', main='Item 1', legendBuffer=0, xlab="", xlas=2)
legend(length(unique(times))+.1, nrow(y_mv), legend=0:1, col=cols, lty=1, lwd=2, title='Response')

item2 <- longCat(y=y_mv[,items==2], tLabels=tLabels)
longCatPlot(item2, ylab='', main='Item 2', legendBuffer=0, xlab="", xlas=2)
legend(length(unique(times))+.1, nrow(y_mv), legend=0:1, col=cols, lty=1, lwd=2, title='Response')

item3 <- longCat(y=y_mv[,items==3], tLabels=tLabels)
longCatPlot(item3, ylab='', main='Item 3', legendBuffer=0, xlab="", xlas=2)
legend(length(unique(times))+.1, nrow(y_mv), 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=y_mv[,times==1], tLabels=Labels)
cols <- longCatPlot(time1, ylab='', main='Time 1', legendBuffer=0, xlab="")
legend(length(unique(times))+.1, nrow(y_mv), legend=0:1, col=cols, lty=1, lwd=2, title='Response')

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

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

time4 <- longCat(y=y_mv[,times==4], tLabels=Labels)
cols <- longCatPlot(time4, ylab='', main='Time 4', legendBuffer=0, xlab="")
legend(length(unique(times))+.1, nrow(y_mv), 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
pdf('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**

<code>y</code>	a data matrix or data frame in wide (as opposed to long) format with cases in rows and repeated observations in columns.
<code>times</code>	time points used for the x-axis in plotting. Either a vector of the same length as the number of columns in <code>y</code> (i.e., all cases have the same times of observation), or a matrix of the same dimension as <code>y</code> (i.e., individually varying times of observation). Default is NULL and is assigned the value <code>1:ncol(y)</code> .
<code>jog</code>	When <code>y</code> is integer data, it can be useful to jog all values by a small amount. When <code>jog=TRUE</code> , a random uniform variate in <code>[-.25, .25]</code> is added to each row in <code>y</code> .
<code>ylim</code>	see <a href="#">par</a> . Default is NULL and calculated from <code>y</code> .
<code>xlim</code>	see <a href="#">par</a> . Default is NULL and calculated from <code>y</code> .
<code>...</code>	Arguments to be passed to <a href="#">plot</a> . See <a href="#">par</a> .

**Author(s)**

Stephen Tueller

**References**

Tueller, S. J., Van Dorn, R. A., & Bobashev, G. V. (2016). Visualization of categorical longitudinal and times series data (Report No. MR-0033-1602). Research Triangle Park, NC: RTI Press. <http://www.rti.org/publication/visualization-categorical-longitudinal-and-times-series-data>

**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**

Tueller, S. J., Van Dorn, R. A., & Bobashev, G. V. (2016). Visualization of categorical longitudinal and times series data (Report No. MR-0033-1602). Research Triangle Park, NC: RTI Press. <http://www.rti.org/publication/visualization-categorical-longitudinal-and-times-series-data>

**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**

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

**Arguments**

<code>dat</code>	a matrix or data frame such as <code>lc\$y</code> from an <a href="#">longCat</a> object created by <a href="#">longCat</a> .
<code>times</code>	see <code>times</code> in <a href="#">longCat</a> .
<code>num</code>	logical indicator, should a numeric version of the concatenate rows be return. Default is TRUE. When <code>num=TRUE</code> , 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 <code>num=TRUE</code> and <code>num=FALSE</code> when experimenting to find the sorting that leads to the clearest plot. When <code>lc\$sorted=FALSE</code> and there is no missing data in <code>lc\$y</code> and <code>lc\$IntTime=FALSE</code> , <a href="#">longCatPlot</a> will change <code>num</code> to FALSE.
<code>mindur</code>	minimum duration. If <code>times</code> is a matrix or data frame of individually varying times of observation of the same dimension as <code>dat</code> , selecting <code>mindur &gt; 0</code> results in all cells in <code>y</code> corresponding to cells in <code>times - times[,1] &lt; mindur</code> being changed to NA (where <code>times - times[,1]</code> changes the <code>times</code> from a matrix of observed times to a matrix of durations for each state in <code>dat</code> ). This minimizes the effect of short durations on the sorting algorithm in <a href="#">sorter</a> . Default is NULL.
<code>igrpt</code>	Option to ignore repeated values when sorting, allowing the sorting algorithm in <a href="#">sorter</a> to smooth over regions of no change for each row in <code>lc\$y</code> . Default is FALSE. See <a href="#">norpt</a> .

**Value**

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

.

**Author(s)**

Stephen Tueller

## References

Tueller, S. J., Van Dorn, R. A., & Bobashev, G. V. (2016). Visualization of categorical longitudinal and times series data (Report No. MR-0033-1602). Research Triangle Park, NC: RTI Press. <http://www.rti.org/publication/visualization-categorical-longitudinal-and-times-series-data>

## See Also

[sorter](#)

## Examples

```
# create an arbitrary matrix and demonstrate
temp <- matrix( sample(1:9, 40, replace=TRUE), 10, 4)
print(temp)
makePatterns(temp, num=FALSE)

# examine the unique patterns of data
bindat <- matrix( sample(0:1, 500, replace=TRUE), 100, 5)
uniquePatterns <- makePatterns( bindat, num=FALSE)
as.matrix( table( uniquePatterns ) )
```

---

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, 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

Tueller, S. J., Van Dorn, R. A., & Bobashev, G. V. (2016). Visualization of categorical longitudinal and times series data (Report No. MR-0033-1602). Research Triangle Park, NC: RTI Press. <http://www.rti.org/publication/visualization-categorical-longitudinal-and-times-series-data>

## See Also

igrpt input to [makePatterns](#)

## Examples

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

---

sort1

*sort1*

---

## 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

<code>id1</code>	The identification variable for one group. See also <code>id</code> for <a href="#">longCat</a> .
<code>y1</code>	The longitudinal data for one group. See also <code>y</code> for <a href="#">longCat</a> .
<code>times1</code>	The time data for one group. See also <code>times</code> for <a href="#">longCat</a> .
<code>events1</code>	The events data for one group. See also <code>events</code> for <a href="#">longCat</a> .
<code>event.times1</code>	The event times for one group. See also <code>event.times</code> for <a href="#">longCat</a> .
<code>group1</code>	The group identification variable for one group. See also <code>group</code> for <a href="#">sorter</a> .
<code>ascending</code>	See <code>ascending</code> for <a href="#">sorter</a> .
<code>whichColumns</code>	See <code>whichColumns</code> for <a href="#">sorter</a> .
<code>initFirst</code>	See <code>initFirst</code> for <a href="#">sorter</a> .

## 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$sorted=FALSE`. If an object has already been sorted (`lc$sort=TRUE`), `sorter` will not resort it, but will print a code example of how to use multiple sortings.

### Usage

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

### Arguments

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

<code>groupLabels</code>	a vector of numeric or character labels of the same length as the number of unique values in <code>group</code> . Default is <code>NULL</code> . If <code>group</code> is not <code>NULL</code> and <code>groupLabels</code> is not provided, then the numeric values in <code>group</code> are used as the labels.
<code>ggap</code>	a number zero to 1. The proportion of blank rows to be plotted between groups when <code>group</code> is specified. The default of <code>NULL</code> 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 longitudinal and times series data (Report No. MR-0033-1602). Research Triangle Park, NC: RTI Press. <http://www.rti.org/publication/visualization-categorical-longitudinal-and-times-series-data>

**See Also**

[longCat](#) and [longCatPlot](#).

**Examples**

```
### 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) + 0.1, xpd=FALSE)

### using the sorted data from the previous plot, repeate 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/
groupplc.df <- data.frame(id=1:nrow(groupplc$group.sorted),
                          group=groupplc$group.sorted[,1], groupplc$y.sorted)
groupplc.long <- reshape(groupplc.df,
                          varying = names(groupplc$y.sorted),
                          v.names = "score",
                          timevar = "time",
                          times = times[1:ncol(groupplc$y.sorted)],
                          direction = "long")
groupplc.long$score <- factor(groupplc.long$score)
groupplc.long$group <- factor(groupplc.long$group, level=3:1)
# remove NA's introduced using group option in sorter
groupplc.long <- na.omit(groupplc.long)
library(ggplot2)
ggplot(groupplc.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)
labels <- c('Street', 'Drug Tx', 'Jail', 'Prison', 'Unknown')
eventLabels=c('Arrest', 'Drug Test', 'Hearing')
eventlcl <- longCat(example3, times=times, Labels=labels,
                   events=events, event.times=event.times,
                   eventLabels=eventLabels)
set.seed(4290)
groupevent <- sample(1:3, nrow(example3), replace=TRUE)
groupeventlcl <- sorter(eventlcl, group=groupevent)
par(mfrow=c(1,1), bg='cornsilk3', mar=c(5.1, 4.1, 4.1, 12.1), xpd=TRUE)
cols <- longCatPlot(groupeventlcl, legendBuffer=0, groupBuffer=0.15,
                   main='Grouping and Events')
legend(610, 130, legend=groupeventlcl$Labels, lty=1, col=cols, lwd=2)
legend(610, 60, legend=groupeventlcl$eventLabels,
       pch=1:length(groupeventlcl$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

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