Package 'Animal'

March 30, 2009

Type Package

Title Analyze time-coded animal behavior data

Version 1.01

Date 2009-03-26

Author Matti Pastell <matti.pastell@helsinki.fi>

Maintainer Matti Pastell <matti.pastell@helsinki.fi>

Description The Animal package is a collection of functions for analyzing animal (including humans) behavior data originating from a variety of sources. The package has functions to analyze time coded behaviors from CowLog (open source software for coding behaviors from digital video) data files and observation files with similar format. Other features include hourly, daily, weekly and monthly summaries of time coded data, analysis of RIC (roughage intake system, Insentec automation) data files and labeling measurement data according to behavioral observations for e.g model building purpose.

License GPL (>=3)

LazyLoad yes

Repository CRAN

URL http://www.mm.helsinki.fi/~mpastell/CowLog/Animal.html

R topics documented:

`RIC` *RIC roughage intake log*

Description

A roughage intake log of a one day from Insentec RIC feed measurement system from the University of Helsinki Viikki researc barn in Finland. The data has been imported to R with function read.RIC option clean=F

Usage

```
data(RIC)
```

Format

A data frame with 3242 observations on the following 16 variables.

`transponder` Transponder number

`cowID` Cow number

`trough` The number of feed trough

`begin` Start time of the visit

`end` End time of the visit

`duration` Visit duration in seconds

`begin.kg` Roughage before visit

`end.kg` Roughage after visit

`feed.type` Type of feed

`intake` Feed intake (kg)

`DM` Dry matter (kg)

`energy` Energy (VEM)

`protein` Protein (kg)

`crude.fibre` Crude fibre (kg)

`fat` Fat (kg)

`ash` Ash (kg)

References

B.V. Marknesse. Instructions for use. RIC - MANAGEMENT WINDOWS version RW:1.7. English. Insentec

`bouts.RIC` *Merge bouts from RIC roughage intake files*

Description

Merges single visits from roughage intake log file if the time difference between successive visits is less than specified time difference i.e merges multiple rows in the log file that are considered to be a single feeding bout to a single row.

Usage

```
bouts.RIC(data, bout.diff = 5)
```

Arguments

`data`	A data.frame read in with read.RIC, with clean=T option
`bout.diff`	The maximum time difference (in minutes) between visits in a single bout

Value

A data.frame with the values merged for individual bouts. All other objects are self.explanatory, but there are two that need further clarification

`bout.duration`
: This is the duration in minutes of the merged bout i.e. begin-end for the bout.

`intake.duration`
: This is the time in minutes that the cow has kept her head in the feeding trough during the bout. You may want to use this for calculating feeding speed (kg/min)

Note

The function is currently only implemented for merging feeding time and intake, thus protein, ash etc. are dropped from the resulting datafrane. The variables in the code are finnish and thus maybe difficult to follow. An english translation will (possibly) appear in the future.

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

See Also

`read.RIC`, `clean.RIC`

Examples

```
data(RIC)
cleaned.data <- clean.RIC(RIC)
bouts <- bouts.RIC(cleaned.data)
#With 8 minutes bout difference
bouts <- bouts.RIC(cleaned.data,bout.diff=8)
```

clean.RIC *Clean RIC roughage intake log file*

Description

Performs the following clean ups on RIC roughage intake files: Removes lines with Cow number 0 and lines with negative feed intake and visits with 0 duration. Equal using clean=T with read.RIC.

Usage

```
clean.RIC(data)
```

Arguments

data	A data.frame read in with read.RIC

Value

Cleaned data.frame

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

See Also

read.RIC, bouts.RIC

Examples

```
data(RIC)
cleaned.data <- clean.RIC(RIC)
```

cowAnalyze *Analyze time coded behavior data*

Description

This function provides descriptive statistics from time coded behavior datafiles recorded with CowLog.

Usage

```
cowAnalyze(file = NULL, states = NULL, events = NULL,
 state.names = NULL, event.names = NULL)
```

Arguments

file	CowLog data file, or a file in same format
states	A vector with the codes in the file that belong to states
events	A vector with codes in the file that belong to events
state.names	A character vector with the names for the states
event.names	A character vector with the names for the events

Value

state	Results for states
event	Results for events

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

References

Hanninen, L. & Pastell, M. CowLog: Open source software for coding behaviors from digital video. Behavior Research Methods. 41(2), 472-476.

http://www.mm.helsinki.fi/ mpastell/CowLog

Examples

```
##Analyze CowLog datafile named calf1.bh1,
## codes 1-3 are states and codes 4-5 are states.
## The names for the states are lying, standing, walking.
## Not run:
analyzed <-cowAnalyze(file='calf1.bh1',states=c(1,2,3),
events=c(4,5),state.names=c('lying','standing','walking'))
## End(Not run)
```

---

daily *Calculate daily values from time series*

Description

Calculate daily values (e.g mean or sum) from time series data. Allows to specify a subject to calculate daily values for several subjects.

Usage

```
daily(data, time, fun = sum, subject = NULL)
```

Arguments

data	A data vector that you want to calculate the daily values for
time	Time stamps for data in POSIXct format
fun	The function to apply, defaults to sum
subject	You can optionally specify to a subject. e.g. to get daily values for each cow in a herd.

Value

A data.frame with following elements

Day	Date
Subject	Appears only if you have specified a subject
Result	The result of the function

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

See Also

monthly, hourly, weekly

Examples

```
data(RIC)
RIC2 <- clean.RIC(RIC)
#Daily feed intake of a whole from data set RIC
herd <- daily(RIC2$intake,time=RIC2$begin,fun=sum)
#Daily feed intake of individual cows from data set RIC
herd <- daily(RIC2$intake,time=RIC2$begin,fun=sum,subject=RIC2$cowID)
```

---

day *Convert dates to day numbers*

Description

This function extracts the day of month from date objects

Arguments

x	A POSIXt object

Value

day	Day of month for the input object

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

See Also

week, day.string, hour, month

Examples

```
date <- Sys.time()
day.number <- day(date)
print(day.number)
```

day.string *Convert dates to string*

Description

This function converts POSIXt dates to a string reprentation of the date. The string format is convenient when calculating daily summaries etc.

Usage

```
day.string(x)
```

Arguments

x	A POSIXt object

Value

Day in string format.

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

See Also

day, hour, week, month

Examples

```
date <- Sys.time()
day.str <- day.string(date)
print(day.str)
```

delete.duplicates *Delete duplicated state events*

Description

Used internally by cowAnalyze

Usage

```
delete.duplicates(obs)
```

Arguments

obs	A dataframe containing state events

Author(s)

Matti Pastell

freq.count *Count frequency of behaviors*

Description

Used internally by cowAnalyze

Usage

```
freq.count(x)
```

Arguments

x	A numeric or factor vector of behaviors

Author(s)

Matti Pastell

hour *Convert times to hours*

Description

This function extracts the hour from date objects

Arguments

x	A POSIXt object

Value

hour	Hour (1-24) of the input object

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

See Also

day, day.string, week, month

Examples

```
date <- Sys.time()
hour.number <- hour(date)
print(hour.number)
```

`hourly` *Calculate hourly values from time series*

Description

Calculate hourly values (e.g mean or sum) from time series data. Allows to specify a subject to calculate hourly values for several subjects.

Usage

```
hourly(data, time, fun = sum, subject = NULL)
```

Arguments

`data`	A data vector that you want to calculate the hourly values for
`time`	Time stamps for data in POSIXct format
`fun`	The function to apply, defaults to sum
`subject`	You can optionally specify to a subject. e.g. to get hourly values for each cow in a herd.

Value

A data.frame with following elements

`Hour`	Hour 1-24
`Subject`	Appears only if you have specified a subject
`Result`	The result of the function

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

See Also

`daily`, `weekly`, `monthly`

Examples

```
data(RIC)
RIC2 <- clean.RIC(RIC)
#Hourly feed intake of a whole from data set RIC
herd <- hourly(RIC2$intake,time=RIC2$begin,fun=sum)
#Hourly feed intake of individual cows from data set RIC
herd <- hourly(RIC2$intake,time=RIC2$begin,fun=sum,subject=RIC2$cowID)
```

`label.data` *Label measurement data*

Description

This function labels measurement data according to behavioral observations. The format of behavioral observations follows CowLog convention (See details).

Usage

```
label.data(data, observation, db = 5, de = 5, min.length = 20)
```

Arguments

`data`	A data.frame holding the measurement data, one of the elements needs to be named time and hold the time stamp for each row.
`observation`	A data.frame with behavioral observations, one of the elements needs to be named time and hold the time stamps for the behaviors and another named behavior and hold the labels for respective behaviors. See details.
`db`	Specifies the delay in seconds from the observation to the start of labeling, this is sometimes useful if the data is used for model building and we want to eliminate the temporal inaccury due to behavioral observations. Defaults to 5
`de`	Similarly to db, specifies the (negative) delay in seconds from the end of observation. Defaults to 5
`min.length`	descibes the minimum length of the labeled data vector. Again sometimes we want to have long enough data vectors for model building and leave out too short bits. This is the length after substracting db and de. Defaults to 20

Details

The time stamp in behavior and data need to be in the same format i.e. POSIXt or seconds from the start in numeric format. The POSIXt is naturally more convinient since then the behavioral observation and the measurement need not to begin from the same time point. The data.frame observation need to have at least two elements: time and behavior (any additional elements are ignored). The time specifies the start of the corresponding behavior and the start time of the next behavior is used as the end of the previous one. The format is adopted from CowLog behavioral coding software (http://www.mm.helsinki.fi/ mpastell/CowLog).

Value

The original data with two additional elements: `label` which contains the labels for each row in the data and `freq` which tells the number of the label counting from the beginning.

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

month *Convert dates to month numbers*

Description

This function extracts the day of month from date objects

Arguments

x	A POSIXt object

Value

month	Month number of the input object

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

See Also

day, day.string, hour,week

Examples

```
date <- Sys.time()
month.number <- month(date)
print(month)
```

monthly *Calculate monthly values from time series*

Description

Calculate monthly values (e.g mean or sum) from time series data. Allows to specify a subject to calculate monthly values for several subjects.

Usage

```
monthly(data, time, fun = sum, subject = NULL)
```

Arguments

data	A data vector that you want to calculate the monthly values for
time	Time stamps for data in POSIXct format
fun	The function to apply, defaults to sum
subject	You can optionally specify to a subject. e.g. to get monthly values for each cow in a herd.

Value

A data.frame with following elements

Day	Date
Subject	Appears only if you have specified a subject
Result	The result of the function

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

See Also

monthly, hourly, weekly

Examples

```
data(RIC)
RIC2 <- clean.RIC(RIC)
#Monthly feed intake of a whole from data set RIC
herd <- monthly(RIC2$intake,time=RIC2$begin,fun=sum)
#Monthly feed intake of individual cows from data set RIC
herd <- monthly(RIC2$intake,time=RIC2$begin,fun=sum,subject=RIC2$cowID)
```

---

nunique *Count unique occurrences of variables*

Description

Returns the number of unique occurrences of each level in the input object.

Usage

```
nunique(x)
```

Arguments

x	Numeric, character of factor vector

Details

Provides a convenient way to calculate the unique occurrences of certain events in daily, hourly, weekly and monthly data e.g. calculate the number of unique animals that have used the feeding throughs each hour in dataset RIC (see examples).

Value

Number of unique levels in the input object.

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

Examples

```
#Lets count the number of unique cows that have started to eat each hour
#in the dataset RIC.
data(RIC)
data <- clean.RIC(RIC)
hourly(RIC$cowID,RIC$begin,nunique)
```

`partOfDay` *Code data into different parts of day*

Description

This function returns the part of day from time stamps. The day can be split into parts of different length with a chosen start time for the splits.

Usage

```
partOfDay(time, nsplit = 4, start = 1)
```

Arguments

`time`	Timestamp vector in POSIXct format
`nsplit`	Number of splits.
`start`	Start time of the split in hours (1-24)

Details

It is often useful to observe the amount of behaviors during different part of day e.g. if we want to find out how different behaviours are distributed over the entire day. This function returns the part of day from time stamps. The function returns only even hours, if nsplit provides intervals with decimal hours they will be rounded to nearest integer.

Value

A factor with the part of day for input timestamps with hour intervals as labels-

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

See Also

`hour`, `hourly`

Examples

```
#Look at the daily distribution of feed intake of cows
#in dataset RIC
data(RIC)
data <- clean.RIC(RIC)
#With default split
data$period <- partOfDay(data$begin)
#Plot the results
boxplot(intake~period,data=data,ylab='Feed intake (kg)',
xlab='Time of day',main='Default settings: start =1, nsplit=4')
#A different split with directly plotting the result
boxplot(intake~partOfDay(begin,nsplit=6,start=3),data=data,
ylab='Feed intake (kg)',xlab='Time of day',main='start=3,nsplit=6')
```

`read.RIC` *Read RIC feed measurement system log files*

Description

Reads in roughage intake log files produced by the Insentec RIC-Management Windows software. (VRyymmdd.DAT) The function converts the start and end times to POSIXct objects and adds the date to each time stamp from the file name.

Usage

```
read.RIC(file, clean = TRUE)
```

Arguments

`file`	The roughage intake log file, (VRyymmdd.DAT)
`clean`	If true the function removes lines with Cow number 0 and lines with negative feed intake and visits with 0 duration. Values TRUE of FALSE, Defaults to TRUE

Value

A data.frame with the formatted insentec data.

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

References

B.V. Marknesse. Instructions for use. RIC - MANAGEMENT WINDOWS version RW:1.7. English. Insentec

See Also

`clean.RIC`, `bouts.RIC`

Examples

```
## Not run: data <- read.RIC('VR080811.DAT')
```

state.durations *Calculate state durations*

Description

Used internally by cowAnalyze

Usage

```
state.durations(obs, state.names = NULL)
```

Arguments

obs	A dataframe containing state events
state.names	A character vector of state behavior names

Author(s)

Matti Pastell

week *Convert dates to week numbers*

Description

This function extracts the week number from date objects

Arguments

x	A POSIXt object

Value

week	Week number of the input object

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

See Also

day, day.string, hour, month

Examples

```
date <- Sys.time()
week.number <- week(date)
print(week.number)
```

`weekly` *Calculate weekly values from time series*

Description

Calculate weekly values (e.g mean or sum) from time series data. Allows to specify a subject to calculate weekly values for several subjects.

Usage

```
weekly(data, time, fun = sum, subject = NULL)
```

Arguments

`data`	A data vector that you want to calculate the weekly values for
`time`	Time stamps for data in POSIXct format
`fun`	The function to apply, defaults to sum
`subject`	You can optionally specify to a subject. e.g. to get weekly values for each cow in a herd.

Value

A data.frame with following elements

`Week`	Week number
`Subject`	Appears only if you have specified a subject
`Result`	The result of the function

Author(s)

Matti Pastell <matti.pastell@helsinki.fi>

See Also

`daily`, `hourly`, `monthly`

Examples

```
data(RIC)
RIC2 <- clean.RIC(RIC)
#Weekly feed intake of a whole from data set RIC
herd <- weekly(RIC2$intake,time=RIC2$begin,fun=sum)
#Weekly feed intake of individual cows from data set RIC
herd <- weekly(RIC2$intake,time=RIC2$begin,fun=sum,subject=RIC2$cowID)
```

Index