Package 'Transition'

July 8, 2025

Type Package

Title Characterise Transitions in Test Result Status in Longitudinal Studies

Version 1.0.0

Date 2025-07-04

Description
Analyse data from longitudinal studies to characterise changes in values of semi-quantitative outcome variables within individual subjects, using high performance C++ code to enable rapid processing of large datasets. A flexible methodology is available for codifying these state transitions.

License MIT + file LICENSE

Imports Rcpp (>= 1.0.14)

LinkingTo Rcpp

RoxygenNote 7.3.2

Encoding UTF-8

Depends R (>= 4.1.0)

LazyData true

URL https://mark-eis.github.io/Transition/

NeedsCompilation yes

Author Mark Eisler [aut, cre, cph] (ORCID: <https://orcid.org/0000-0001-6843-3345>),
Ana Rabaza [aut] (ORCID: <https://orcid.org/0000-0002-9713-0797>)

Maintainer Mark Eisler <mark.eisler@bristol.ac.uk>

Repository CRAN

Date/Publication 2025-07-08 10:00:02 UTC

Contents

`Transition-package` *Characterise Transitions in Test Result Status in Longitudinal Studies*

Description

Analyse data from longitudinal studies to characterise changes in values of semi-quantitative outcome variables within individual subjects, using high performance C++ code to enable rapid processing of large datasets. A flexible methodology is available for codifying these state transitions.

Package Content

Index of help topics:

```
Blackmore               Exercise Histories of Eating-Disordered and
                        Control Subjects
PreviousDate            Find Previous Test Date for Subject
PreviousResult          Find Previous Test Result for Subject
Transition-package      Characterise Transitions in Test Result Status
                        in Longitudinal Studies
Transitions             Identify Temporal Transitions in Longitudinal
                        Study Data
uniques                 Unique Values for Subject, Timepoint and Result
```

Maintainer

Mark Eisler <mark.eisler@bristol.ac.uk>

Author(s)

Mark Eisler [aut, cre, cph] (ORCID: <https://orcid.org/0000-0001-6843-3345>), Ana Rabaza [aut] (ORCID: <https://orcid.org/0000-0002-9713-0797>)

`Blackmore` *Exercise Histories of Eating-Disordered and Control Subjects*

Description

The `Blackmore` data frame has 945 rows and 4 columns. Blackmore and Davis's data on exercise histories of 138 teenaged girls hospitalized for eating disorders and 98 control subjects.

Usage

```
Blackmore
```

Format

This data frame contains the following columns:

subject a factor with subject id codes. There are several observations for each subject, but because the girls were hospitalized at different ages, the number of cases and the age at the last case vary.

age subject's age in years at the time of observation; all but the last observation for each subject were collected retrospectively at intervals of two years, starting at age 8.

exercise the amount of exercise in which the subject engaged, expressed as estimated hours per week.

group a factor with levels: `control`, Control subjects; `patient`, Eating-disordered patients.

Note

The original version in package **carData** states there are 98 `control` subjects, but the actual data only have 93 unique values, and that assuming `subject` id codes with suffixes `a` and `b` represent different individuals; otherwise, there are just 87.

Source

Personal communication from Elizabeth Blackmore and Caroline Davis, York University.

References

Davis C, Blackmore E, Katzman DK, Fox J. (2005). Female adolescents with anorexia nervosa and their parents: a case-control study of exercise attitudes and behaviours. *Psychological Medicine* **35**(3):377-386. doi:10.1017/S0033291704003447

`PreviousDate` *Find Previous Test Date for Subject*

Description

`get_prev_dates()` identifies the previous test date for individual subjects and timepoints in a longitudinal study.

`add_prev_date()` interpolates these previous test dates into a data frame for further analysis.

Usage

```
add_prev_date(
  object,
  subject = "subject",
  timepoint = "timepoint",
  result = "result"
)

get_prev_date(
  object,
  subject = "subject",
  timepoint = "timepoint",
  result = "result"
)
```

Arguments

`object`	a `data.frame` (or object coercible by `as.data.frame()` to a data frame) containing the data to be analysed.
`subject`	`character`, name of the column (of type `integer` or `factor`) identifying individual study subjects; default `"subject"`.
`timepoint`	`character`, name of the column recording time points (as `Dates`) of testing of subjects; default `"timepoint"`.
`result`	`character`, name of the column (of type `ordered factor`) recording test results; default `"result"`.

Details

See `Transitions` *details*.

Value

`add_prev_date()`

A `data.frame` based on `object`, with an added column of class `Date` containing the values of the previous test dates.

`get_prev_date()`
: An vector of length `nrow(object)`, class `Date`, containing the values of the previous test dates ordered in the exact sequence of the `subject` and `timepoint` in `object`.

See Also

`data.frame`, `Dates`, `ordered factor`.

Other transitions: `PreviousResult`, `Transitions`, `uniques()`

Examples

```
 ## Continuing example from `add_transitions()`
  # subject, timepoint and result arguments all defaults and required types
Blackmore |> str()

  # Integer vector of the previous test dates
get_prev_date(Blackmore)

  # Add column of  previous test dates to data frame
add_prev_date(Blackmore) |> head(32)

rm(Blackmore)

###
## Example on formatting numeric values as R dates

#  Data frame containing year as numeric: 2018 to 2025
(df <- data.frame(
    subject = rep(1001:1003),
    timepoint = rep(2018:2025, each = 3),
    result = gl(3, 4, lab = c("jolly", "good", "show"), ordered = TRUE)
    ))

#  Convert to R dates
df <- transform(df,
           timepoint = as.Date(paste(timepoint, "01", "01", sep = "-"))
      )

  # Add column of test result transitions (defaults: cap = 0, modulate = 0)
(df <- add_transitions(df))

# Format R dates just to show the year
transform(df, timepoint = format(timepoint, "%Y"))

#  Data frame containing year and month as numeric: July 2024 to June 2025
(df <- data.frame(
           subject = 1001:1002,
           year = rep(2024:2025, each = 12),
           month = rep(c(7:12, 1:6), each = 2),
            result = gl(2, 3, lab = c("low", "high"), ordered = TRUE)
        ))

#  Convert to R dates
df <- transform(df, timepoint = as.Date(paste(year, month, "01", sep = "-")))


  # Add column of test result transitions (defaults: cap = 0, modulate = 0)
(df <- add_transitions(df))

# Format R dates just to show the month and year
transform(df, timepoint = format(timepoint, "%b-%Y"))

rm(df)
```

`PreviousResult` *Find Previous Test Result for Subject*

Description

`get_prev_results()` identifies the previous test result for individual subjects and timepoints in a longitudinal study.

`add_prev_result()` interpolates these previous test results into a data frame for further analysis.

Usage

```
add_prev_result(
  object,
  subject = "subject",
  timepoint = "timepoint",
  result = "result"
)

get_prev_result(
  object,
  subject = "subject",
  timepoint = "timepoint",
  result = "result"
)
```

Arguments

`object`	a `data.frame` (or object coercible by `as.data.frame()` to a data frame) containing the data to be analysed.
`subject`	`character`, name of the column (of type `integer` or `factor`) identifying individual study subjects; default `"subject"`.

`timepoint`	`character`, name of the column recording time points (as `Dates`) of testing of subjects; default `"timepoint"`.
`result`	`character`, name of the column (of type `ordered factor`) recording test results; default `"result"`.

Details

See `Transitions` *details*.

Value

`add_prev_result()`
: A `data.frame` based on `object`, with an added column of type `ordered factor` containing the values of the previous test results.

`get_prev_result()`
: An `ordered factor` of length `nrow(object)`, containing the values of the previous test results ordered in the exact sequence of the `subject` and `timepoint` in `object`.

See Also

`data.frame`, `Dates`, `ordered factor`.

Other transitions: `PreviousDate`, `Transitions`, `uniques()`

Examples

```
 ## Continuing example from `add_transitions()`
  # subject, timepoint and result arguments all defaults and required types
Blackmore |> str()

  # Previous test results as ordered factor
get_prev_result(Blackmore)

  # Previous test result as column of data frame
(Blackmore <- add_prev_result(Blackmore)) |> head(32)

rm(Blackmore)
```

`Transitions` *Identify Temporal Transitions in Longitudinal Study Data*

Description

`get_transitions()` identifies temporal transitions in test results for individual subjects in a longitudinal study.

`add_transitions()` interpolates these transitions into a data frame for further analysis.

Usage

```
add_transitions(
  object,
  subject = "subject",
  timepoint = "timepoint",
  result = "result",
  transition = "transition",
  cap = 0L,
  modulate = 0L
)

get_transitions(
  object,
  subject = "subject",
  timepoint = "timepoint",
  result = "result",
  cap = 0L,
  modulate = 0L
)
```

Arguments

`object`	a `data.frame` (or object coercible by `as.data.frame()` to a data frame) containing the data to be analysed.
`subject`	`character`, name of the column (of type `integer` or `factor`) identifying individual study subjects; default `"subject"`.
`timepoint`	`character`, name of the column recording time points (as `Dates`) of testing of subjects; default `"timepoint"`.
`result`	`character`, name of the column (of type `ordered factor`) recording test results; default `"result"`.
`transition`	`character`, name to be used for a new column (of type `integer`) to record transitions; default `"transition"`.
`cap`	`integer`, required for calculating transitions; default `0L`.
`modulate`	`integer`, required for calculating transitions; default `0L`.

Details

The data can be presented in any order e.g., ordered by `subject`, by `timepoint`, forwards or backwards in time, or entirely at random, and may have unbalanced designs with different time points or numbers of test results per subject. However, the *user* is responsible for ensuring the data contain unique combinations of `subject`, `timepoint` and `result`; if not, outputs will be undefined.

Time points should be formatted as `Dates` and included in data frame `object` in the column named as specified by argument `timepoint` (see *Note*).

Test results should be semi-quantitiative, formatted as `ordered factor` and included in data frame `object` in the column named as specified by argument `result` (see *Note*).

Temporal transitions in the test `results` for each `subject` within the `object` `data.frame` are characterised using methods governed by options `cap` and `modulate`. If these two parameters are both zero (their defaults), a simple arithmetic difference between the levels of the present and previous result is calculated. Otherwise, if the value of `modulate` is a positive, non-zero integer, the arithmetic difference is subjected to integer division by that value. Finally, if `cap` is a positive, non-zero integer, the (possibly modulated) absolute arithmetic difference is capped at that value.

Value

`add_transitions()`
: A `data.frame` based on `object`, with an added column of type `integer` containing the values of the test result transitions.

`get_transitions()`
: An `integer vector` of length `nrow(object)`, containing the values of the test result transitions ordered in the exact sequence of the `subject` and `timepoint` in `object`.

Note

Time points represented by `integer` or `numeric` values can be converted to R `Dates` conveniently using `as.Date()`. If only *year* information is available, arbitrary values could be used consistently for month and day e.g., 1st of January of each year; likewise, the first day of each month could be used arbitrary, if only the *year* and *month* were known.

Quantitive results available as `numeric` data can be converted to a semi-quantitative `ordered factor` conveniently using `cut()` (see *examples*).

See Also

`data.frame`, `Dates`, and `ordered factor`.

Other transitions: `PreviousDate`, `PreviousResult`, `uniques()`

Examples

```
  # Inspect Blackmore data frame using {base} str()
Blackmore |> str()

  # {base} hist() gives insights into the "exercise" column,
  #   useful for choosing `breaks` and `labels` in cut() below
hist(Blackmore$exercise, include.lowest = TRUE, plot = FALSE)[1:2]

  # Tweak Blackmore data frame by converting "age" to dates for the argument
  #   timepoint (using an arbitrary "origin" of 1-Jan-2000), and converting
  #   "exercise" to an ordered factor "result" with {base} cut()
Blackmore <- transform(Blackmore,
    timepoint = as.Date("2000-01-01") + round(age * 365.25),
    result = cut(
        exercise,
        breaks = seq(0, 30, 2),
        labels = paste0("<=", seq(0, 30, 2)[-1]),
        include.lowest = TRUE,
        ordered_result = TRUE
    )
)

  # subject, timepoint and result arguments now defaults and required types
Blackmore |> str()

  # Integer vector of test result transitions (defaults: cap = modulate = 0)
get_transitions(Blackmore)

  # Tabulate values of transitions
get_transitions(Blackmore) |> table()

  # Effect of cap argument
get_transitions(Blackmore, cap = 6) |> table()

  # Effect of modulate argument
get_transitions(Blackmore, modulate = 2) |> table()

  # Add column of test result transitions to data frame
add_transitions(Blackmore) |> head(22)

  # Showing transitions as either positive (1) or negative (-1)
  #   (defaults to modulate = 0)
add_transitions(Blackmore, cap = 1) |> head(14)

rm(Blackmore)
```

---

uniques *Unique Values for Subject, Timepoint and Result*

Description

uniques() identifies unique values for subjects, timepoints and test results in longitudinal study data.

Usage

```
uniques(
  object,
  subject = "subject",
  timepoint = "timepoint",
  result = "result"
)
```

Arguments

object	a data.frame (or object coercible by as.data.frame() to a data frame) containing the data to be analysed.

`subject`	`character`, name of the column (of type `integer` or `factor`) identifying individual study subjects; default `"subject"`.
`timepoint`	`character`, name of the column recording time points (as `Dates`) of testing of subjects; default `"timepoint"`.
`result`	`character`, name of the column (of type `ordered factor`) recording test results; default `"result"`.

Details

See `Transitions` *details*.

Works for `subject` as either an `integer vector` or a `factor`.

Value

A `list` of three elements

`1.`	An `integer vector` or `factor` of unique subject identifications.
`2.`	A `vector` of class `Date` of unique timepoints in the study.
`3.`	An `ordered factor` of unique values for results of the study.

See Also

`data.frame`, `Dates`, `ordered factor`.

Other transitions: `PreviousDate`, `PreviousResult`, `Transitions`

Examples

```
 ## Continuing example from `add_transitions()`
  # subject, timepoint and result arguments all defaults and required types
  # (native subject is factor)
uniques(Blackmore)
  #
Blackmore <- transform(Blackmore, subject = as.integer(subject))
  # subject now as integer
Blackmore |> str()
uniques(Blackmore)

rm(Blackmore)
```

Index