Package: msprog (via r-universe)

October 20, 2026

Type Package

Title Reproducible Assessment of Disability Course in Multiple Sclerosis

Version 1.0.1

Description Analyse disability course in multiple sclerosis (MS) from longitudinal data. The package provides a flexible framework for identifying disability events under user-specified criteria, allowing adaptation to different study designs and endpoints. Tools are included to facilitate transparent and reproducible reporting of the settings used in the analysis. For consensus-based recommendations on endpoint calculation and use of the package, see Montobbio et al. (2026) <doi:10.1177/13524585261478492>. For the original description of the computational framework and illustrative applications, see Montobbio et al. (2024) <doi:10.1177/13524585241243157>.

URL https://github.com/noemimontobbio/msprog

BugReports https://github.com/noemimontobbio/msprog/issues

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Imports stats, dplyr, rlang

Depends R (>= 4.1.0)

VignetteBuilder knitr

Suggests knitr, rmarkdown, xfun, DT

License MIT + file LICENSE

Repository https://noemimontobbio.r-universe.dev

Date/Publication 2026-09-04 14:50:18 UTC

RemoteUrl https://github.com/noemimontobbio/msprog

RemoteRef HEAD

RemoteSha 82e5de70a2fcba141b780a03a85f38224abc3a9f

Contents

compute_delta 2
is_event 3
MSprog 4
print.MSprogOutput 11
relapse_indep_from_bounds 12
toydata_relapses 14
toydata_visits 15
value_milestone 15

Index **19**

`compute_delta` *Default minimum clinically meaningful shift for different scales.*

Description

Compute the minimum clinically meaningful score change as a function of the reference value for some widely used scales (EDSS, NHPT, T25FW, or SDMT), according to the most established rule for each of these outcomes.

Usage

```
compute_delta(baseline, outcome = "edss")
```

Arguments

`baseline`	Outcome value at baseline.
`outcome`	One of:

- `"edss"` (Extended Disability Status Scale, default);
- `"nhpt"` (Nine-Hole Peg Test);
- `"t25fw"` (Timed 25-Foot Walk);
- `"sdmt"` (Symbol Digit Modalities Test).

Details

Default thresholds are meant to apply to all versions of each test (e.g., dominant or non-dominant hand for NHPT, best time or mean of two trials, etc.).

Value

Minimum clinically meaningful change from the provided baseline value. Specifically:

- EDSS: 1.5 if `baseline`=0, 1 if 0<`baseline`<=5.0, 0.5 if `baseline`>5.0
- NHPT and T25FW: 20% of `baseline`
- SDMT: either 4 points or 20% of `baseline`.

Examples

```
compute_delta(4.5) # default outcome is "edss"
compute_delta(55, outcome="sdmt")
```

`is_event` *Compare value to reference.*

Description

Check if an outcome value determines a valid worsening, or improvement, or change, from a given reference value.

Usage

```
is_event(
  x,
  baseline,
  type,
  outcome = "edss",
  worsening = NULL,
  delta_fun = NULL,
  sub_threshold = FALSE
)
```

Arguments

`x`	Outcome value to test.
`baseline`	Outcome value at baseline.
`type`	One of: • `"wors"` (worsening); • `"impr"` (improvement); • `"change"` (any valid change).
`outcome`	One of: • `"edss"` (Extended Disability Status Scale, default); • `"nhpt"` (Nine-Hole Peg Test); • `"t25fw"` (Timed 25-Foot Walk); • `"sdmt"` (Symbol Digit Modalities Test); • `"custom"` (only accepted when specifying non-`NULL` `worsening` – and `delta_fun` as well, if `sub_threshold=FALSE`). Outcome type determines the direction of worsening (see `worsening` argument) and selects the default definition of clinically meaningful change given the reference value (using the built-in function `compute_delta()`). The latter can be replaced by a custom function using the `delta_fun` argument.

worsening — The direction of worsening ("increase" if higher values correspond to worse disease course, "decrease" otherwise). This argument is only used when outcome is set to "custom". Otherwise, worsening is automatically set to "increase" for EDSS, NHPT, T25FW, and to "decrease" for SDMT.

delta_fun — Custom function specifying the minimum clinically meaningful change in the outcome measure from the provided reference value. The function provided must take a numeric value (reference score) as input, and return a numeric value corresponding to the minimum relevant shift from baseline, see example below. If outcome is set to "custom", a custom delta function must be specified by the user. For other values of outcome, if no delta_fun is specified, the built-in function compute_delta() is used internally. The argument is ignored if sub_threshold=TRUE.

sub_threshold — If TRUE, any confirmed worsening, or improvement, or change in the outcome measure is valid, regardless of delta_fun.

Value

A boolean value specifying if a valid event was found.

Examples

```
is_event(x=4.5, baseline=4, type="wors", outcome="edss")
is_event(x=50, baseline=57, type="wors", outcome="sdmt")
is_event(x=3, baseline=3.5, type="impr", outcome="edss", sub_threshold=TRUE)
```

MSprog *Assess multiple sclerosis disability course from longitudinal data.*

Description

Detect and characterise confirmed disability worsening (CDW) or improvement (CDI) events of an outcome measure (EDSS, NHPT, T25FW, or SDMT; or any custom outcome) based on repeated assessments through time (and on the dates of acute episodes, if any). The events are detected sequentially by scanning the outcome values in chronological order. Several qualitative and quantitative options are given as arguments that can be set by the user and reported as a complement to the results to ensure reproducibility.

Usage

```
MSprog(
  data,
  subj_col,
  value_col,
  date_col,
  outcome,
  relapse = NULL,
  rsubj_col = NULL,
```

```
  rdate_col = NULL,
  renddate_col = NULL,
  subjects = NULL,
  delta_fun = NULL,
  worsening = NULL,
 event = c("firstCDW", "firstCDI", "multiple", "firstPIRA", "firstRAW", "first"),
  RAW_PIRA = FALSE,
  baseline = c("fixed", "roving", "roving_impr", "roving_wors"),
  proceed_from = c("firstconf", "event"),
  sub_threshold_rebl = c("none", "change", "improvement", "worsening"),
  bl_geq = FALSE,
  relapse_rebl = FALSE,
  skip_local_extrema = c("none", "strict", "all"),
  validconf_col = NULL,
  conf_days = 12 * 7,
  conf_tol_days = c(7, 2 * 365.25),
  require_sust_days = 0,
  check_intermediate = TRUE,
  relapse_to_bl = 30,
  relapse_to_event = 0,
  relapse_to_conf = 30,
  relapse_assoc = 90,
  relapse_indep = NULL,
  impute_last_visit = 0,
  date_format = NULL,
  include_dates = FALSE,
  include_values = FALSE,
  include_stable = TRUE,
  verbose = 1
)
```

Arguments

`data`	Data frame containing longitudinal data, including: subject IDs, outcome values, visit dates.
`subj_col`	Name of data column with subject IDs.
`value_col`	Name of data column with outcome values.
`date_col`	Name of data column with visit dates.
`outcome`	Specifies the outcome type. Must be one of the following: • `"edss"` (Expanded Disability Status Scale) • `"nhpt"` (Nine-Hole Peg Test) • `"t25fw"` (Timed 25-Foot Walk) • `"sdmt"` (Symbol Digit Modalities Test) • `"custom"` (only accepted when specifying custom `delta_fun` and `worsening`). When it's not set to `"custom"`, outcome type triggers internal checks on value range, determines the direction of worsening (see `worsening` argument), and

selects the default definition of clinically meaningful change given the reference value (using the built-in function `compute_delta()`). The latter can be replaced by a custom function using the `delta_fun` argument.

`relapse` Optional data frame containing longitudinal data, including subject IDs and relapse onset dates.

`rsubj_col` Name of subject ID column in the `relapse` data frame, if different from the one in `data`.

`rdate_col` Name of relapse onset date column in the `relapse` data frame, if different from the date column in `data`.

`renddate_col` Name of relapse *end* date column in the `relapse` data frame (if applicable).

`subjects` Subset of subjects (a vector or list of IDs). If none is specified, all subjects listed in `data` are included.

`delta_fun` Custom function specifying the minimum clinically meaningful change in the outcome measure from the provided reference value. The function provided must take a numeric value (reference score) as input, and return a numeric value corresponding to the minimum shift from baseline, see example below. If `outcome` is set to `"custom"`, a custom delta function must be specified by the user. For other values of `outcome`, if no `delta_fun` is specified, the built-in function `compute_delta()` is used internally.

`worsening` The direction of worsening (`"increase"` if higher values correspond to worse disease course, `"decrease"` otherwise).

The given value is only used when `outcome` is set to `"custom"`. Otherwise, `worsening` is automatically set to `"increase"` if `outcome` is set to `"edss"`, `"nhpt"`, `"t25fw"`, and to `"decrease"` if `outcome` is set to `"sdmt"`.

`event` Character string specifying which events to detect. Must be one of the following.

- `"firstCDW"` (first CDW, default).
- `"firstCDI"` (first CDI).
- `"multiple"` (all confirmed events in chronological order).
- `"firstPIRA"` (first progression independent of relapse activity, PIRA).
- `"firstRAW"` (first relapse-associated worsening, RAW).
- `"first"` (only the very first confirmed event – CDI or CDW).

`RAW_PIRA` If `TRUE`, further classify CDW events based on their timing with respect to relapses (RAW, PIRA). The argument is ignored if `event` is set to `"firstCDI"` (not relevant) or to `"firstRAW"`/`"firstPIRA"` (always enabled).

`baseline` Specifies the baseline scheme. Must be one of the following.

- `"fixed"`: first eligible outcome value; recommended for randomised data (default).
- `"roving"`: updated after each CDI or CDW event to the visit determined by `proceed_from`; suitable for a multiple-event setting (i.e., when `event` is set to `"multiple"`) or when searching for a specific type of CDW (i.e., when `event` is set to `"firstPIRA"` or `"firstRAW"`).
- `"roving_impr"`: updated after every CDI (to the visit determined by `proceed_from`); suitable for a first-CDW setting to discard fluctuations around baseline; not recommended for multiple events.

- `"roving_wors"`: updated after every CDW (to the visit determined by `proceed_from`); suitable when searching for a specific type of CDW (i.e., when `event` is set to `"firstPIRA"` or `"firstRAW"`).

`proceed_from` After detecting a confirmed disability event, continue searching:

- from the next visit after the first qualifying confirmation visit if `proceed_from="firstconf"`
- from the next visit after the event onset if `proceed_from="event"`.

If `baseline` is set to `"roving"`, `"roving_impr"`, or `"roving_wors"`, when re-baselining after a confirmed disability event, the baseline is moved to:

- the first qualifying confirmation visit if `proceed_from="firstconf"`
- the event visit if `proceed_from="event"`.

`sub_threshold_rebl`

This argument is only used if `baseline` is not set to `"fixed"`. It controls whether and which "sub-threshold" events (i.e., *confirmed* changes in the outcome measure below the clinically meaningful threshold) can trigger a re-baseline. Must be one of the following.

- `"none"` (default): only use *clinically meaningful* confirmed changes for re-baseline.
- `"change"`: any confirmed sub-threshold change can potentially trigger a re-baseline.
- `"improvement"`: any confirmed sub-threshold improvement can potentially trigger a re-baseline.
- `"worsening"`: any confirmed sub-threshold worsening can potentially trigger a re-baseline.

See `delta_fun` argument and `compute_delta()` function for more details.

`bl_geq` This argument is only used if relapse-based re-baseline is enabled (`relapse_rebl=TRUE`). If `TRUE`, the new reference value must always be greater or equal than the previous one; when it is not, the old reference value is assigned to it [2].

`relapse_rebl` If `TRUE`, re-baseline after every relapse [2].

`skip_local_extrema`

This argument is only used if the baseline is moved. It controls re-baseline behaviour in the presence of local minima or maxima.

A visit `i` is a local minimum point for outcome `x` if

`x[i+1]>x[i]` and `x[i-1]>x[i]`.

Local maxima are defined similarly.

A visit `i` is a *strict* local minimum point for outcome `x` if

`x[i+1]-x[i]>=delta_fun(x[i])`;

`x[i-1]-x[i]>=delta_fun(x[i])`.

Strict local maxima are defined similarly.

When `x[i]=x[i-2]`, visit `i` is *not* considered a local extremum point even if the above conditions hold. This controls for cases where the outcome has an undulating course. The following argument values are accepted.

- `"none"` (default): local extrema are always accepted as valid baseline values.

- "strict": the baseline cannot be placed at a *strict* local minimum or maximum.
- "all": the baseline cannot be placed at a local minimum or maximum.

validconf_col
: Name of data column, if any, specifying which visits can (TRUE) or cannot (FALSE) be used as confirmation visits. If not specified (validconf_col=NULL), all visits are potentially used as confirmation visits.

conf_days
: Period before confirmation (days). Can be a single value, or vector of any length if considering multiple windows. If length(conf_days) > 1 (e.g., conf_days=c(12*7, 24*7)), the function detects events confirmed at *either* time point (e.g., "confirmed over 12 *or* 24 weeks") with their relative tolerance (as per conf_tol_days).

conf_tol_days
: Tolerance window for confirmation visit (days). Can be an integer (equal lower and upper tolerance) or vector of length 2 (different lower and upper tolerance). The right end of the interval (upper tolerance) may be set to Inf (confirmation window unbounded on the right – e.g., "confirmed over 12 *or more* weeks").

require_sust_days
: Minimum number of days over which a confirmed change must be sustained (i.e., confirmed at *all* visits occurring in the specified period) to be retained as an event. Events sustained for the remainder of the follow-up period are always retained regardless of follow-up duration. If require_sust_days=Inf, events are retained only when sustained for the remainder of the follow-up period.

  (Warning: if check_intermediate is set to FALSE, sustained change will be established based *only on the end* of the specified period.)

check_intermediate
: If TRUE (default), events are confirmed *over all intermediate visits* up to the confirmation visit.

  If set to FALSE (not recommended in most cases, as it may discard meaningful fluctuations), events will be confirmed *only at* the specified confirmation visit (and *only at the end* of the period defined by require_sust_days, if any).

relapse_to_bl
: Minimum distance (days) from the onset of a relapse for a visit to be used as baseline. Can be an integer (minimum distance from *last* relapse onset) or vector of length 2 (minimum distance from *last* relapse onset, minimum distance from *next* relapse onset). Note that setting the distance to zero means keeping the baseline where it is regardless of surrounding relapses.

  If relapse end dates are available (renddate_col), the minimum distance from last relapse onset is automatically set to the specific relapse duration, unless relapse_to_bl (or relapse_to_bl[1]) is zero (in which case relapse timing does not affect baseline placement).

  If the designated baseline does not satisfy this constraint, the baseline is moved to the next available visit.

relapse_to_event
: Minimum distance (days) from the onset of a relapse for an event to be considered as such. Can be an integer (minimum distance from *last* relapse onset) or vector of length 2 (minimum distance from *last* relapse onset, minimum distance from *next* relapse onset). Note that setting the distance to zero means retaining the event regardless of surrounding relapses.

  If relapse end dates are available (renddate_col), the minimum distance from last relapse onset is automatically set to the specific relapse duration, unless

relapse_to_event (or relapse_to_event[1]) is zero (in which case relapse timing does not affect event validation).

relapse_to_conf
: Minimum distance (days) from the onset of a relapse for a visit to be an eligible confirmation visit. Can be an integer (minimum distance from *last* relapse onset) or vector of length 2 (minimum distance from *last* relapse onset, minimum distance from *next* relapse onset). Note that setting the distance to zero means using any visit for confirmation regardless of surrounding relapses.

  If relapse end dates are available (renddate_col), the minimum distance from last relapse onset is automatically set to the specific relapse duration, unless relapse_to_conf (or relapse_to_conf[1]) is zero (in which case relapse timing does not affect selection of confirmation visits).

relapse_assoc
: Maximum distance (days) from the onset of a relapse for a CDW event to be classified as RAW. Can be an integer (maximum distance from *last* relapse onset) or vector of length 2 (maximum distance from *last* relapse onset, maximum distance from *next* relapse onset). If relapse end dates are available (renddate_col), the maximum distance from last relapse is automatically set to the specific relapse duration. The argument is ignored if RAW events are not detected (e.g., if event="firstCDW" and RAW_PIRA=FALSE).

relapse_indep
: Specifies relapse-free intervals for PIRA definition. Must be a named list list(prec=list(p0, p1), event=list(e0, e1), conf=list(c0, c1)) specifying the intervals around (any subset of) three checkpoints:

  1. a preceding visit, e.g., baseline or last visit before the worsening (p0 and p1)
  2. the event onset (e0 and e1)
  3. an eligible confirmation visit (c0 and c1).

  The auxiliary function relapse_indep_from_bounds() can be used to organise interval bounds into a named list correctly, by calling:
  relapse_indep_from_bounds(p0, p1, e0, e1, c0, c1)
  See relapse_indep_from_bounds() function docs for more details on how to define the intervals. If relapse end dates are available (renddate_col), it is possible to also define PIRA based on those by setting use_end_dates=TRUE in relapse_indep_from_bounds(). The argument is ignored if PIRA events are not detected (e.g., if event="firstCDW" and RAW_PIRA=FALSE).

impute_last_visit
: Imputation probability for worsening events occurring at the last available visit (i.e., with no confirmation). Unconfirmed worsening events occurring at the last visit are never imputed if impute_last_visit=0; they are always imputed if impute_last_visit=1; they are imputed with probability p, 0<p<1, if impute_last_visit=p. If a value N>1 is passed, unconfirmed worsening events are imputed only if occurring within N days of follow-up (e.g., in case of early discontinuation).

date_format
: Format of dates in the date_col and rdate_col columns of the input data. Can be specified as:

  - Standard format for dates (e.g., "%d-%m-%Y"; see strptime() docs for correct syntax).

- "day" if dates in are given as "days from start" (the starting point can be different for each subject – e.g., days from randomisation in a clinical trial); negative values are accepted.

If not specified, function as.Date() will try to infer it automatically.

include_dates If TRUE, output$results will include the dates of:

- event onset ("date" column)
- the initial baseline ("initial_bl_date" column)
- the current baseline ("bl_date" column)
- the last visit before event onset with a clinically meaningful score difference from it ("last_delta_date" column)
- the confirmation visit(s) ("conf<c>_date" column and, when relevant, "PIRA_conf<c>_date" column for each c in conf_days)

include_values If TRUE, output$results will include the outcome value at:

- event onset ("value" column)
- the initial baseline ("initial_bl_value" column)
- the current baseline ("bl_value" column)
- the last visit before event onset with a clinically meaningful score difference from it ("last_delta_value" column)
- the confirmation visit(s) ("conf<c>_value" column and, when relevant, "PIRA_conf<c>_value" column for each c in conf_days)

include_stable If TRUE, subjects with no confirmed events are included in output$results, with time2event = total follow up.

verbose One of:

- 0 (print no info);
- 1 (print concise info, default);
- 2 (print extended info).

Value

An object of class MSprogOutput with the following attributes:

- event_count: a data frame containing event counts for each subject (and the event sequence in case of multiple events).
- results: a data frame with extended info on each event for all subjects.
- settings: a list containing all the arguments used to compute the output.
- unconfirmed: a data frame with info on unconfirmed events (initial change from baseline, but no confirmation) for all subjects.

For a detailed description of output data frames, see ?MSprogOutput.

References

[1] Müller J, Cagol A, Lorscheider J, Tsagkas C, Benkert P, Yaldizli Ö, et al. Harmonizing definitions for progression independent of relapse activity in multiple sclerosis: A systematic review. JAMA Neurol. 2023;80:1232–45.

[2] Kappos L, Wolinsky JS, Giovannoni G, Arnold DL, Wang Q, Bernasconi C, et al. Contribution of relapse-independent progression vs relapse-associated worsening to overall confirmed disability accumulation in typical relapsing multiple sclerosis in a pooled analysis of 2 randomized clinical trials. JAMA Neurol. 2020;77:1132–40.

Examples

```
# 1. EDSS course
output <- MSprog(toydata_visits, subj_col="id", value_col="EDSS", date_col="date", outcome="edss",
    relapse=toydata_relapses, conf_days=12*7, conf_tol_days=30,
    event="multiple", baseline="roving", verbose=1)
print(output$results) # extended info on each event for all subjects
print(output$event_count) # event counts for each subject
# 2. SDMT course
output <- MSprog(toydata_visits, subj_col="id", value_col="SDMT", date_col="date", outcome="sdmt",
    relapse=toydata_relapses, conf_days=12*7, conf_tol_days=30,
    event="multiple", baseline="roving", verbose=1)
print(output$results) # extended info on each event for all subjects
print(output$event_count) # event counts for each subject
# 3. SDMT course, with a custom delta function
my_sdmt_delta <- function(reference_value) {min(c(reference_value/10, 3))}
output <- MSprog(toydata_visits, subj_col="id", value_col="SDMT", date_col="date", outcome="sdmt",
    delta_fun=my_sdmt_delta,
    relapse=toydata_relapses, conf_days=12*7, conf_tol_days=30,
    event="multiple", baseline="roving", verbose=1)
print(output$results) # extended info on each event for all subjects
print(output$event_count) # event counts for each subject
```

---

print.MSprogOutput *Textual description of criteria used to assess disability course.*

Description

`print` method for class `"MSprogOutput"`.

Usage

```
## S3 method for class 'MSprogOutput'
print(x, ...)
```

Arguments

`x`	An object of class `"MSprogOutput"` (result of a call to `MSprog()`).
`...`	Optional arguments for `print` methods. They are ignored in this function.

Details

The method prints out (1) the package version, (2) a full list of function arguments, and (3) a short paragraph describing the full set of criteria used to obtain the output.

Value

Invisibly returns `x`.

Examples

```
output <- MSprog(toydata_visits, "id", "EDSS", "date", "edss",
    relapse=toydata_relapses, conf_days=7*12, conf_tol_days=30,
    event="multiple", baseline="roving", verbose=2)
print(output) # textual description of parameters used to obtain output
```

`relapse_indep_from_bounds`

Define relapse-free intervals for PIRA definition.

Description

Organise the given interval bounds into a named list to be given as argument `relapse_indep` to function `MSprog()`. The relapse-free intervals may be anchored to (any subset of) the following three data-driven checkpoints.

1. `"prec"`: a visit preceding the event: can be (i) the current baseline, (ii) the last visit before event onset, or (iii) the last visit before event onset with a clinically meaningful score difference from it (i.e., for an outcome `x` with "higher score" = "worse", the last visit `i` such that `x[event] - x[i] >= delta_fun(x[i])`, and same for the confirmation visit).
2. `"event"`: the disability worsening event onset.
3. `"conf"`: an eligible confirmation visit.

Usage

```
relapse_indep_from_bounds(
  p0 = 0,
  p1 = 0,
  e0 = 0,
  e1 = 0,
  c0 = 0,
  c1 = 0,
  prec_type = "baseline",
  use_end_dates = FALSE
)
```

Arguments

p0	Days before preceding visit (>=0).
p1	Days after preceding visit (>=0), or NULL.
e0	Days before event onset (>=0), or NULL.
e1	Days after event onset (>=0), or NULL.
c0	Days before confirmation (>=0), or NULL.
c1	Days after confirmation (>=0).
prec_type	Which visit to use as "preceding visit". Must be one of: • "baseline": the current baseline. • "last": the last visit before event onset. • "last_delta: the last visit before event onset with a clinically meaningful score difference from it – i.e., for an outcome x with "higher score" = "worse", the last visit i where x[event onset] - x[i] >= delta_fun(x[i]) and x[confirmation] - x[i] >= delta_fun(x[i]).
use_end_dates	If TRUE, only the right bounds e1 and c1 are used, as the left bounds will be defined by the onset-to-end interval of each relapse. This option is only relevant when relapse *end* dates are provided (see renddate_col argument in MSprog()).

Details

If both ends of an interval are 0 (e.g., if both p0=0 and p1=0), the checkpoint is ignored. To merge two intervals together, set both the right end of the first interval and the left end of the second interval to NULL (e.g., "between baseline and event onset": p1=NULL and e0=NULL). Here are some examples:

- No relapses from 90dd before to 30dd after the event, and from 90dd before to 30dd after confirmation [1]:
  relapse_indep_from_bounds(e0=90,e1=30,c0=90,c1=30).
- No relapses between baseline and confirmation (high-specificity definition from [1]):
  relapse_indep_from_bounds(p0=0,p1=NULL,e0=NULL,e1=NULL,c0=NULL,c1=0).
- No relapses from baseline to 30dd after the event, and within confirmation+-30dd [2]:
  relapse_indep_from_bounds(p0=0,p1=NULL,e0=NULL,e1=30,c0=30,c1=30).

Value

A named list to be given as argument relapse_indep to function MSprog().

References

[1] Müller J, Cagol A, Lorscheider J, Tsagkas C, Benkert P, Yaldizli Ö, et al. Harmonizing definitions for progression independent of relapse activity in multiple sclerosis: A systematic review.

JAMA Neurol. 2023;80:1232–45.

[2] Kappos L, Wolinsky JS, Giovannoni G, Arnold DL, Wang Q, Bernasconi C, et al. Contribution of relapse-independent progression vs relapse-associated worsening to overall confirmed disability accumulation in typical relapsing multiple sclerosis in a pooled analysis of 2 randomized clinical trials. JAMA Neurol. 2020;77:1132–40.

`toydata_relapses` *Synthetic Relapse Data*

Description

Artificially generated relapse onset dates for some example patients in toydata_visits to illustrate the use of the package.

Usage

```
data(toydata_relapses)
```

Format

An object of class `data.frame`, with columns:

id Subject IDs.

date Relapse onset dates.

References

This data set was artificially created for the `msprog` package.

Examples

```
data(toydata_relapses)
head(toydata_relapses)
```

toydata_visits *Synthetic Longitudinal EDSS and SDMT Data*

Description

Artificially generated toy data set including Extended Disability Status Scale (EDSS) and Symbol Digit Modalities Test (SDMT) scores in a small cohort of example patients to illustrate the use of the package.

Usage

```
data(toydata_visits)
```

Format

An object of class `data.frame`, with columns:

id Subject IDs.

date Visit dates.

EDSS Synthetic EDSS scores (values between 0 and 10).

SDMT Synthetic SDMT scores (values between 0 and 110).

References

This data set was artificially created for the `msprog` package.

Examples

```
head(toydata_visits)
```

value_milestone *Time to disability milestone.*

Description

Scan the visits in chronological order to detect the first outcome value reaching or exceeding a specified disability milestone (e.g., EDSS>=6), *with confirmation*.

Usage

```
value_milestone(
  data,
  milestone,
  subj_col,
  value_col,
  date_col,
  outcome,
  worsening = NULL,
  relapse = NULL,
  rsubj_col = NULL,
  rdate_col = NULL,
  validconf_col = NULL,
  conf_days = 12 * 7,
  conf_tol_days = c(7, 2 * 365.25),
  require_sust_days = 0,
  relapse_to_event = 0,
  relapse_to_conf = 30,
  impute_last_visit = 0,
  date_format = NULL,
  verbose = 0
)
```

Arguments

`data`	Data frame containing longitudinal data, including: subject IDs, outcome values, visit dates.
`milestone`	Disability milestone (outcome value to check data against).
`subj_col`	Name of data column with subject IDs.
`value_col`	Name of data column with outcome values.
`date_col`	Name of data column with visit dates.
`outcome`	Specifies the outcome type. Must be one of the following: • `"edss"` (Expanded Disability Status Scale) • `"nhpt"` (Nine-Hole Peg Test) • `"t25fw"` (Timed 25-Foot Walk) • `"sdmt"` (Symbol Digit Modalities Test) • `"custom"` (only accepted when specifying argument `worsening`). When it's not set to `"custom"`, outcome type triggers internal checks on value range and determines the direction of worsening (see `worsening` argument).
`worsening`	The direction of worsening (`"increase"` if higher values correspond to worse disease course, `"decrease"` otherwise). The given value is only used when `outcome` is set to `"custom"`. Otherwise, `worsening` is automatically set to `"increase"` if `outcome` is set to `"edss"`, `"nhpt"`, `"t25fw"`, and to `"decrease"` if `outcome` is set to `"sdmt"`.

`relapse` Optional data frame containing longitudinal data, including subject IDs and relapse onset dates.

`rsubj_col` Name of subject ID column in the `relapse` data frame, if different from the one in `data`.

`rdate_col` Name of date column in the `relapse` data frame, if different from the date column in `data`.

`validconf_col` Name of data column, if any, specifying which visits can (`TRUE`) or cannot (`FALSE`) be used as confirmation visits. If not specified (`validconf_col=NULL`), all visits are potentially used as confirmation visits.

`conf_days` Period before confirmation (days). Can be a single value, or vector of any length if considering multiple windows. If `length(conf_days) > 1` (e.g., `conf_days=c(12*7, 24*7)`), the function retains milestones confirmed at *either* time point (e.g., "confirmed over 12 *or* 24 weeks") with their relative tolerance (as per `conf_tol_days`).

`conf_tol_days` Tolerance window for confirmation visit (days); can be an integer (equal lower and upper tolerance) or vector of length 2 (different lower and upper tolerance). The right end of the interval can be set to `Inf` (confirmation window unbounded on the right – e.g., "confirmed over 12 *or more* weeks").

`require_sust_days`
Minimum number of days over which the milestone must be sustained (i.e., confirmed at *all* visits occurring in the specified period). If the milestone is sustained for the remainder of the follow-up period, it is considered reached regardless of follow-up duration. If `require_sust_days=Inf`, values are retained only when sustained for the remainder of the follow-up period.

`relapse_to_event`
Minimum distance (days) from the onset of a relapse for the milestone to be considered reached. Can be an integer (minimum distance from *last* relapse onset) or vector of length 2 (minimum distance from *last* relapse onset, minimum distance from *next* relapse onset). Note that setting the distance to zero means retaining the event regardless of surrounding relapses.

`relapse_to_conf`
Minimum distance (days) from the onset of a relapse for a visit to be a valid confirmation visit. Can be an integer (minimum distance from *last* relapse onset) or vector of length 2 (minimum distance from *last* relapse onset, minimum distance from *next* relapse onset). Note that setting the distance to zero means using any visit for confirmation regardless of surrounding relapses.

`impute_last_visit`
Imputation probability when the milestone is reached at the last available visit (i.e., with no confirmation). Unconfirmed values exceeding the milestone at the last visit are never imputed if `impute_last_visit=0`; they are always imputed if `impute_last_visit=1`; they are imputed with probability `p`, `0<p<1`, if `impute_last_visit=p`. If a value `N>1` is passed, unconfirmed values exceeding the milestone are imputed only if occurring within `N` days of follow-up (e.g., in case of early discontinuation).

`date_format` Format of dates in the `date_col` and `rdate_col` columns of the input data. Can be specified as:

- Standard format for dates (e.g., `"%d-%m-%Y"`; see `strptime()` docs for correct syntax).
- `"day"` if dates in are given as "days from start" (the starting point can be different for each subject – e.g., days from randomisation in a clinical trial); negative values are accepted.

If not specified, function `as.Date()` will try to infer it automatically.

`verbose` One of:

- 0 (print no info)
- 1 (print concise info, default)
- 2 (print extended info).

Details

- "Reaching or exceeding" means either value>=milestone or value<=milestone, depending on the direction of worsening (see arguments `outcome` and `worsening`).
- An event is only considered "observed" if **confirmed**, i.e., if all values *up to* the confirmation visit reach or exceed the milestone.

Value

A data frame containing the following columns:

- `<date_col>`: the date of first reaching or exceeding the milestone with confirmation (or last date of follow-up if milestone is not reached or not confirmed).
- `<value_col`: the first value reaching or exceeding the milestone with confirmation, if present, otherwise no value.
- `"time2event"`: the time taken to reach or exceed the milestone (or total follow-up length if milestone is not reached or not confirmed).
- `"observed"`: whether the milestone was reached with confirmation (1) or not (0).

Index

∗ **datasets**
toydata_relapses, 14
toydata_visits, 15

as.Date(), *10*, *18*

compute_delta, 2
compute_delta(), *3, 4*, *6, 7*

is_event, 3

MSprog, 4
MSprog(), *11–13*

print.MSprogOutput, 11

relapse_indep_from_bounds, 12
relapse_indep_from_bounds(), *9*

strptime(), *9*, *18*

toydata_relapses, 14
toydata_visits, *14*, 15

value_milestone, 15