Package 'JointFPM'

July 21, 2025

Type Package

Title A Parametric Model for Estimating the Mean Number of Events

Version 1.2.2

Date 2025-02-20

Description Implementation of a parametric joint model for modelling recurrent and competing event processes using generalised survival models as described in Entrop et al., (2005) <doi:10.1002/bimj.70038>. The joint model can subsequently be used to predict the mean number of events in the presence of competing risks at different time points. Comparisons of the mean number of event functions, e.g. the differences in mean number of events between two exposure groups, are also available.

URL https://github.com/entjos/JointFPM, https://entjos.github.io/JointFPM/

BugReports https://github.com/entjos/JointFPM/issues

License CC BY 4.0

Encoding UTF-8

RoxygenNote 7.3.2

Imports rstpm2 (>= 1.5.2), survival (>= 3.2-13), data.table (>= 1.14.2), rlang (>= 1.1.0), lifecycle, rmutil, cli, matrixStats, statmod

Depends R (>= 4.1.0)

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Config/testthat/parallel true

Config/testthat/start-first watcher, Parallel*

LazyData true

NeedsCompilation no

Author Joshua P. Entrop [aut, cre, cph] (ORCID: <https://orcid.org/0000-0003-1614-8096>), Alessandro Gasparini [ctb], Mark Clements [ctb]

Maintainer Joshua P. Entrop <joshuaentrop@posteo.de>

Repository CRAN

Date/Publication 2025-02-20 12:20:14 UTC

Contents

`bladder1_stacked` *Stacked version of the bladder1 dataset included in the survival package*

Description

This dataset includes the bladder1 dataset included the survival package, which has been transformed into stacked format for use with `JointFPM`. The stacked datset includes one row per individual for the competing event and one rows per individual for each reoccurrence of bladder cancer.

Usage

```
bladder1_stacked
data(bladder1_stacked)
```

Format

A data frame with 412 rows and 11 columns

Details

For more information please take a look at `?survival::bladder`.

`JointFPM` *Joint FPMs for recurrent and competing events.*

Description

Fits a joint flexible parametric survival model (FPM) for a recurrent and terminal event. The joint model can be used to predict the mean number of events at different time points. This function is a wrapper around `rstpm2::stpm2()`.

Usage

```
JointFPM(
  surv,
  re_model,
  ce_model,
  re_indicator,
  ce_indicator,
  df_ce = 3,
  df_re = 3,
  tvc_re_terms = NULL,
  tvc_ce_terms = NULL,
  cluster,
  data,
  control = list(),
  ...
)
```

Arguments

`surv`	A formula of the following form `Surv(...) ~ 1`. The `Surv` objects needs to be of `type == 'counting'` with the following arguments: `time`: Start of follow-up time for each event episode, i.e., usually 0 for the competing event and the first occurrence of the recurrent event. For every subsequent event the follow-up can either be 0 if gap time is the underlying time scale or the time of the previous event if total time is the underlying time scale. `time2`: End of follow-up, i.e., either occurrence of a terminal or recurrent event, or time of censoring. `status`: Event indicator for both terminal and recurrent event. `type`: Has to be `counting`.
`re_model`	A formula object specifying the model for the recurrent event with an empty right hand side of the formula, e.g. `~ sex`.
`ce_model`	A formula object specifying the model for the competing event with an empty right hand side of the formula, e.g. `~ sex`.

`re_indicator` Indicator that defines which rows in the dataset belong to the recurrent event process. These are usually more than one row per observations. The variable name needs to be passed as a character vector.

`ce_indicator` Indicator that defines which row in the dataset belong to the competing event process. The variable name needs to be passed as a character vector.

`df_ce` Defines the number of knots used to model the baseline hazard function for the competing event process.

`df_re` Defines the number of knots used to model the baseline hazard function for the recurrent event process.

`tvc_re_terms` A named list defining the number of knots used to model potential time-varying effects of variables included in the recurrent event model. This list should be of form `list(<var_name> = <no. of knots>)`.

`tvc_ce_terms` A named list defining the number of knots used to model potential time-varying effects of variables included in the competing event model. This list should be of form `list(<var_name> = <no. of knots>)`.

`cluster` A character vector specifying the name of the variable that defines unique observations in the dataset passed to the function.

`data` A stacked dataset that includes both data on the recurrent and competing event process. The dataset should have one row for each observation including the follow-up time and event indicator for the competing event and possibly multiple rows for each observation including the follow-up times and event indicator for the recurrent event, e.g.:

```
id st_start  st_end re status
 1       0      6.88  0      1
 1       0      6.88  1      0
 2       0      8.70  0      1
 2       0      8.70  1      0
 3       0     10     0      0
 3       0      1.78  1      1
 3       1.78   6.08  1      1
 3       6.08  10     1      0
 4       0      6.07  0      1
 4       0      6.07  1      0
```

`control` List of arguments passed to rstpm2::gsm.control.

`...` Additional arguments to be passed to rstpm2::stpm2.

Value

An object of class `JointFPM` with the following elements:

`model:` The fitted FPM object,

`re_terms:` The terms used to model the recurrent event model,

`ce_terms:` The terms used to model the competing event model,

`re_indicator:` The name of the indicator variable of the recurrent event

Examples

```
JointFPM(Surv(time  = start,
              time2 = stop,
              event = event,
              type  = 'counting') ~ 1,
         re_model = ~ pyridoxine + thiotepa,
         ce_model = ~ pyridoxine + thiotepa,
         re_indicator = "re",
         ce_indicator = "ce",
         df_ce = 3,
         df_re = 3,
         tvc_ce_terms = list(pyridoxine = 2,
                             thiotepa   = 2),
         tvc_re_terms = list(pyridoxine = 2,
                             thiotepa   = 2),
         cluster  = "id",
         data     = bladder1_stacked)
```

`mean_no` *Non-parametric estimation of mean number of events*

Description

[Experimental]

Usage

```
mean_no(
  formula,
  re_indicator,
  ce_indicator,
  data,
  re_control = list(),
  ce_control = list()
)
```

Arguments

`formula`	A formula passed to `survfit`.
`re_indicator`	The name of a variable indicating that these rows in the dataset belong to the risksets of the recurrent event process.
`ce_indicator`	The name of a variable indicating that these rows in the datasets belong to the riskset of the competing event process.
`data`	A `data.frame` in stacked format. The dataset needs to include one row for the competing event and one row for each risk episode of the recurrent event.

re_control	An optional list with arguments passed to survfit when computing risksets for the recurrent event.
ce_control	An optional list with arguments passed to survfit when computing risksets for the competing event.

Value

A data.frame including the estimated mean number of events expn at times t within strata strata.

---

predict.JointFPM *Post-estimation function for JointFPMs*

Description

Predicts different estimates from a joint flexible parametric model. Currently only the estimation of the mean number of events at different time points is supported.

Usage

```
## S3 method for class 'JointFPM'
predict(
  object,
  type = "mean_no",
  newdata,
  t,
  exposed = NULL,
  ci_fit = TRUE,
  method = "romberg",
  ngq = 30,
  ...
)
```

Arguments

object	A joint flexible parametric model of class JointFPM.
type	A character vector defining the estimate of interest. Currently available options are: mean_no: Estimates the mean number of events at time(s) t. diff: Estimates the difference in mean number of events between exposed and unexposed at time(s) t. marg_mean_no: Estimates the marginal mean number of events. marg_diff: Estimates the marginal difference in the mean number of events.
newdata	A data.frame with one row including the variable values used for t he prediction. One value for each variable used in either the recurrent or competing event model is required when predicting mean_no or diff. For marg_mean_no or marg_diff, this includes the variable that you would like your marginal estimate to be conditioned on.

t	A vector defining the time points used for the prediction.
exposed	A function that takes newdata as an argument and creates a new dataset for the exposed group. This argument is required if type = 'diff'. Please see details for more information.
ci_fit	Logical indicator for whether confidence intervals should be estimated for the fitted estimates using the delta method.
method	The method used for the underlying numerical integration procedure. Defaults to "romberg", which uses the rmutil::int() function, but it is possible to use Gaussian quadrature by setting method = "gq" instead.
ngq	Number of quadrature nodes used when method = "gq". Defaults to 30, which lead to accurate results (compared to method = "romberg") in our experience.
...	Added for compatibility with other predict functions.

Details

The function required for the exposed argument must take the newdata dataset as argument and transform it to a new dataset that defines the exposed group. Assume we assume that we have a model with one variable trt which is a 0/1 coded treatment indicator. If we would like to obtain the difference in mean number of events comparing the untreated to treated group we could use the following function assuming that newdata = data.frame(trt = 0):

```
function(x){transform(x, trt = 1)}
```

Value

A data.frame with the following columns:

- t: The time for the prediction,
- fit: The point estimate of the prediction,
- lci: The lower confidence interval limit,
- uci: The upper confidence interval limit.

Examples

```
bldr_model <- JointFPM(Surv(time  = start,
                            time2 = stop,
                            event = event,
                            type  = 'counting') ~ 1,
                       re_model = ~ pyridoxine + thiotepa,
                       ce_model = ~ pyridoxine + thiotepa,
                       re_indicator = "re",
                       ce_indicator = "ce",
                       df_ce = 3,
                       df_re = 3,
                       cluster  = "id",
                       data     = bladder1_stacked)

predict(bldr_model,
```

```
          newdata = data.frame(pyridoxine = 1,
                               thiotepa   = 0),
          t       =  c(10, 20),
          ci_fit  = FALSE)
```

`summary.JointFPM` *Summarises a* `JointFPM` *objects*

Description

This is a summary function for `JointFPM` objects, created with `JointFPM()`. The function improves the readability of the output.

Usage

```
## S3 method for class 'JointFPM'
summary(object, ...)
```

Arguments

`object`	An `JointFPM` object.
`...`	Other arguments that should be passed to the function.

Value

No return value, called for side effects.

`test_dfs_JointFPM` *Tests DFs for JointFPMs.*

Description

Test of different degrees of freedoms (DFs) for joint flexible parametric survival models.

[Experimental]

Usage

```
test_dfs_JointFPM(
  surv,
  re_model,
  ce_model,
  re_indicator,
  ce_indicator,
  dfs_ce,
  dfs_re,
```

```
  tvc_re_terms = NULL,
  tvc_ce_terms = NULL,
  cluster,
  data
)
```

Arguments

surv	A formula of the following form `Surv(...) ~ 1`. The `Surv` objects needs to be of `type == 'counting'` with the following arguments: **`time`:** Start of follow-up time for each event episode, i.e., usually 0 for the competing event and the first occurrence of the recurrent event. For every subsequent event the follow-up can either be 0 if gap time is the underlying time scale or the time of the previous event if total time is the underlying time scale. **`time2`:** End of follow-up, i.e., either occurrence of a terminal or recurrent event, or time of censoring. **`status`:** Event indicator for both terminal and recurrent event. **`type`:** Has to be `counting`.
re_model	A formula object specifying the model for the recurrent event with an empty right hand side of the formula, e.g. `~ sex`.
ce_model	A formula object specifying the model for the competing event with an empty right hand side of the formula, e.g. `~ sex`.
re_indicator	Indicator that defined which rows in the dataset belong to the recurrent event process. These are usually more than one row per observations. The variable name needs to be passed as a character vector.
ce_indicator	Indicator that defined which row in the dataset belong to the competing event process. The variable name needs to be passed as a character vector.
dfs_ce	Defines the number of knots used to model the baseline hazard function for the competing event process.
dfs_re	Defines the number of knots used to model the baseline hazard function for the recurrent event process.
tvc_re_terms	A named list defining the numbers of knots used to model potential time-varying effects of variables included in the recurrent event model. This list should be of form `list(<var_name> = <no. of knots>)`.
tvc_ce_terms	A named list defining the numbers of knots used to model potential time-varying effects of variables included in the competing event model. This list should be of form `list(<var_name> = <no. of knots>)`.
cluster	A chara vector specifying the name of the variable that defines unique observation in the dataset passed to the function.
data	A stacked dataset that including both data on the recurrent and competing event process. The dataset should have one row for each observation including the follow-up time and event indicator for the competing event and possibly multiple rows for each observation including the follow-up times and event indicator for the recurrent event, e.g.:

```
id st_start  st_end re status
 1      0      6.88  0      1
 1      0      6.88  1      0
 2      0      8.70  0      1
 2      0      8.70  1      0
 3      0     10     0      0
 3      0      1.78  1      1
 3      1.78   6.08  1      1
 3      6.08  10     1      0
 4      0      6.07  0      1
 4      0      6.07  1      0
```

Value

A `data.frame` with one row per combination of baseline hazards DFs, and the DFs of the time varying covariates, and the corresponding AIC and BIC.

Examples

```
# Test different dfs
test_dfs_JointFPM(Surv(time  = start,
                       time2 = stop,
                       event = event,
                       type  = 'counting') ~ 1,
                  re_model = ~ pyridoxine + thiotepa,
                  ce_model = ~ pyridoxine + thiotepa,
                  re_indicator = "re",
                  ce_indicator = "ce",
                  dfs_ce = 1:3,
                  dfs_re = 2,
                  tvc_ce_terms = list(thiotepa   = 1:2),
                  tvc_re_terms = list(pyridoxine = 2),
                  cluster  = "id",
                  data     = bladder1_stacked)
```

Index