Package: TTM (via r-universe)

October 28, 2026

Type Package

Title Terminal Trend Models on the Retrospective Time Scale

Version 0.1.0

Description Fits retrospective-time longitudinal models for terminal trend analysis using weighted generalized estimating equations and related joint modeling tools. The package provides functions to estimate spline-based baseline and treatment effect trajectories on the retrospective time scale, obtain pointwise confidence intervals, and visualize fitted trajectories.

License GPL-2

URL https://github.com/TChang-W/TTM

BugReports https://github.com/TChang-W/TTM/issues

Depends R (>= 4.4.0)

Imports dplyr, geepack, geer, ggplot2 (>= 3.4.0), lme4, magrittr, splines, stats, survival, utils

Suggests knitr, rmarkdown

VignetteBuilder knitr

LazyData true

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/pak/sysreqs cmake make libicu-dev

Repository https://tchang-w.r-universe.dev

Date/Publication 2026-10-16 20:16:48 UTC

RemoteUrl https://github.com/tchang-w/ttm

RemoteRef HEAD

RemoteSha c3b070d3f328f2b5ac94e9158147f37d3189c770

Contents

get_beta_A 2
get_beta_mu 4
plot_beta_A 5
plot_beta_mu 7
Retro_data 8
RetroJM 9
test_beta_slope 12
TTM 13
TTM_data 16
TTM_linear 17

Index **20**

`get_beta_A` *Evaluate the estimated treatment effect trajectory on the retrospective time scale*

Description

Computes the estimated treatment effect trajectory, $\beta_A(t^*)$, at user-supplied retrospective time points from a fitted `TTM()` or `RetroJM()` object.

The trajectory is formed from the treatment main effect together with the treatment-by-spline interaction terms in the fitted retrospective-time longitudinal model. Pointwise confidence intervals are constructed using a normal approximation based on the estimated covariance matrix of the fixed effects.

Usage

```
get_beta_A(new_times, results, conf.level = 0.95)
```

Arguments

`new_times`	Numeric vector of retrospective time points t^* at which to evaluate the treatment effect trajectory.
`results`	A fitted object returned by `TTM()` or `RetroJM()`. The object must contain spline knot information and the estimated fixed effects with their covariance matrix.
`conf.level`	Numeric scalar giving the confidence level for pointwise confidence intervals. Must be strictly between `0` and `1`.

Details

For a fitted retrospective-time model of the form

$$E[Y(t^*)] = \beta_X^\top X + \beta_\mu(t^*) + A \times \beta_A(t^*),$$

this function evaluates $\beta_A(t^*)$ at specified t^*

The spline basis is reconstructed using the boundary knots and internal knots stored in `results`, so the supplied `new_times` should be on the same retrospective-time scale as used when fitting the model.

Confidence intervals are pointwise, not simultaneous.

Value

A named list with components:

time_grid The input retrospective time points in `new_times`.

estimate Estimated treatment effect values $\beta_A(t^*)$.

standard_error Pointwise standard errors of the estimated treatment effect trajectory.

CI.lower Lower pointwise confidence limits.

CI.upper Upper pointwise confidence limits.

See Also

`TTM()`, `get_beta_mu()`, `plot_beta_A()`

Examples

```
fit <- TTM(
  data_long = TTM_data,
  id = "id",
  treatment = "A",
  outcome = "Y",
  covariates = c("X1", "X2"),
  os_time = "OS_time",
  event = "event",
  t_pros = "t_pros"
)

trt_eff <- get_beta_A(
  new_times = seq(0, 10, by = 1),
  results = fit,
  conf.level = 0.95
)

trt_eff$estimate
trt_eff$CI.lower
trt_eff$CI.upper
```

`get_beta_mu` *Get the estimated baseline retrospective trajectory with pointwise confidence intervals*

Description

Evaluates the estimated baseline longitudinal trajectory on the retrospective timescale, denoted by $\beta_\mu(t^*)$, at user-supplied time points. Pointwise normal approximation confidence intervals are returned based on the weighted GEE fixed effect estimates from a fitted `TTM()` or `RetroJM()` object.

Usage

```
get_beta_mu(new_times, results, conf.level = 0.95)
```

Arguments

`new_times`	A numeric vector of retrospective time points at which the baseline trajectory is to be evaluated.
`results`	A fitted object returned by `TTM()` or `RetroJM()`.
`conf.level`	A numeric confidence level for the pointwise confidence interval. Must be strictly between `0` and `1`. The default is `0.95`.

Details

The baseline trajectory corresponds to the mean spline-based effect of retrospective time for the reference treatment group and baseline covariate pattern encoded by the model intercept-like term `X0` and spline terms.

For a fitted retrospective-time model of the form

$$E[Y(t^*)] = \beta_X^\top X + \beta_\mu(t^*) + A \times \beta_A(t^*),$$

this function evaluates $\beta_\mu(t^*)$ at specified t^*

The spline basis is reconstructed using the boundary knots and internal knots stored in `results`, so the supplied `new_times` should be on the same retrospective-time scale as used when fitting the model.

Confidence intervals are pointwise, not simultaneous.

Value

A list containing:

time_grid The user-supplied retrospective time points.

estimate The estimated baseline trajectory $\beta_\mu(t^*)$.

standard_error The pointwise standard error of the estimated trajectory.

CI.lower The lower bound of the pointwise confidence interval.

CI.upper The upper bound of the pointwise confidence interval.

See Also

TTM(), get_beta_A(), plot_beta_mu()

Examples

```
fit <- TTM(
  data_long = TTM_data,
  id = "id",
  treatment = "A",
  outcome = "Y",
  covariates = c("X1", "X2"),
  os_time = "OS_time",
  event = "event",
  t_pros = "t_pros"
)

baseline <- get_beta_mu(
  new_times = seq(0, 10, by = 1),
  results = fit,
  conf.level = 0.95
)

baseline$estimate
baseline$CI.lower
baseline$CI.upper
```

---

plot_beta_A — *Plot the estimated treatment effect trajectory on the retrospective time scale*

Description

Plots the estimated treatment effect trajectory, $\beta_A(t^*)$, together with pointwise confidence intervals from a fitted TTM() or RetroJM() object.

The plotted trajectory is formed from the treatment main effect together with the treatment-by-spline interaction terms in the fitted retrospective-time longitudinal model. Pointwise confidence intervals are constructed using a normal approximation based on the estimated covariance matrix of the fixed effects.

Usage

```
plot_beta_A(results, xlim = NULL, conf.level = 0.95)
```

Arguments

`results`	A fitted object returned by `TTM()` or `RetroJM()`. The object must contain spline knot information and the estimated fixed effects with their covariance matrix.
`xlim`	Optional numeric vector of length 2 giving the range of retrospective time to display. If `NULL`, the spline boundary range stored in `results` is used.
`conf.level`	Numeric scalar giving the confidence level for pointwise confidence intervals. Must be strictly between `0` and `1`.

Details

This function evaluates the treatment effect trajectory over a fine grid of retrospective time values and plots the estimated curve together with its lower and upper pointwise confidence limits.

The spline basis is reconstructed using the boundary knots and internal knots stored in `results`, so the plotted time axis should be interpreted on the same retrospective-time scale used when fitting the model.

Confidence intervals are pointwise, not simultaneous.

Value

Invisibly returns a named list with components:

plot The `ggplot2` plot object.

time_grid The retrospective time grid used for plotting.

estimate Estimated treatment effect values $\beta_A(t^*)$.

standard_error Pointwise standard errors of the estimated treatment effect trajectory.

CI.lower Lower pointwise confidence limits.

CI.upper Upper pointwise confidence limits.

See Also

`TTM()`, `get_beta_A()`, `plot_beta_mu()`

Examples

```
fit <- TTM(
  data_long = TTM_data,
  id = "id",
  treatment = "A",
  outcome = "Y",
  covariates = c("X1", "X2"),
  os_time = "OS_time",
  event = "event",
  t_pros = "t_pros"
)

plot_beta_A(
  results = fit,
```

```
  xlim = c(0, 10),
  conf.level = 0.95
)
```

`plot_beta_mu` *Plot the estimated baseline trajectory on the retrospective time scale*

Description

Plots the estimated baseline retrospective trajectory, $\beta_\mu(t^*)$, together with pointwise confidence intervals from a fitted `TTM()` or `RetroJM()` object.

The plotted trajectory is formed from the intercept-like baseline term together with the spline terms in the fitted retrospective-time longitudinal model. Pointwise confidence intervals are constructed using a normal approximation based on the estimated covariance matrix of the fixed effects.

Usage

```
plot_beta_mu(results, xlim = NULL, conf.level = 0.95)
```

Arguments

`results`	A fitted object returned by `TTM()` or `RetroJM()`. The object must contain spline knot information and the estimated fixed effects with their covariance matrix.
`xlim`	Optional numeric vector of length 2 giving the range of retrospective time to display. If `NULL`, the spline boundary range stored in `results` is used.
`conf.level`	Numeric scalar giving the confidence level for pointwise confidence intervals. Must be strictly between `0` and `1`.

Details

This function evaluates the baseline trajectory over a fine grid of retrospective time values and plots the estimated curve together with its lower and upper pointwise confidence limits.

The spline basis is reconstructed using the boundary knots and internal knots stored in `results`, so the plotted time axis should be interpreted on the same retrospective-time scale used when fitting the model.

Confidence intervals are pointwise, not simultaneous.

Value

Invisibly returns a named list with components:

plot The `ggplot2` plot object.

time_grid The retrospective time grid used for plotting.

estimate Estimated baseline trajectory values $\beta_\mu(t^*)$.

standard_error Pointwise standard errors of the estimated baseline trajectory.

CI.lower Lower pointwise confidence limits.

CI.upper Upper pointwise confidence limits.

See Also

`TTM()`, `get_beta_mu()`, `plot_beta_A()`

Examples

```
fit <- TTM(
  data_long = TTM_data,
  id = "id",
  treatment = "A",
  outcome = "Y",
  covariates = c("X1", "X2"),
  os_time = "OS_time",
  event = "event",
  t_pros = "t_pros"
)

plot_beta_mu(
  results = fit,
  xlim = c(0, 10),
  conf.level = 0.95
)
```

`Retro_data` *Simulated retrospective joint modeling data for* `RetroJM()`

Description

A simulated long-format data set with repeated measurements of a longitudinal outcome before death or censoring. It is used in the vignette of `RetroJM()`. It has the same structure as TTM_data.

Usage

```
Retro_data
```

Format

A data frame with 3586 rows (1000 subjects) and 17 variables:

id Subject identifier.

A Binary treatment indicator (0 = control, 1 = treatment).

X1 Binary baseline covariate.

X2 Non-negative continuous baseline covariate.

u Subject-level random effect generated in the simulation.

b_longi Random intercept for the longitudinal outcome.

randomslope Random slope generated for the longitudinal outcome.

OS_time Observed time, the minimum of the death, dropout, and censoring times (months).

event Event indicator: 0 = censored, 1 = death, 2 = dropout.

death Death indicator (1 = death observed).

dropout Dropout indicator (1 = dropout observed).

death_time Simulated (latent) death time.

dropout_time Simulated (latent) dropout time.

censoring_time Simulated (latent) administrative censoring time.

t_pros Prospective measurement time since baseline.

t_retro True retrospective time, `death_time - t_pros`.

Y Longitudinal outcome.

Source

Simulated data.

Examples

```
data("Retro_data", package = "TTM")
head(Retro_data)
```

`RetroJM` *Fit the two-stage model for survival imputation and longitudinal analysis*

Description

Fits a two-stage retrospective joint modeling procedure. In stage 1, a Cox proportional hazards model is fit to overall survival and used to impute death times for censored individuals. In stage 2, the imputed datasets are used to fit a linear mixed effects model for the longitudinal outcome measured on a retrospective timescale from death. Fixed effects estimates and associated uncertainty are then pooled across imputations using standard multiple imputation combining rules.

Usage

```
RetroJM(
  data_long,
  id = "id",
  treatment = "A",
  outcome = "Y",
  covariates = c("X1", "X2"),
  os_time = "OS_time",
```

```
  death = "death",
  t_pros = "t_pros",
  random_effects = c("1", "A"),
  n_impute = 10,
  n_inner_knot = 3
)
```

Arguments

`data_long`	A long-format `data.frame` containing one row per subject per visit. The dataset must include the subject identifier, treatment variable, longitudinal outcome, survival time-to-event outcome, prospective follow-up time, and any baseline covariates used in the models.
`id`	A character string giving the name of the subject identifier column. Each unique value should correspond to one individual.
`treatment`	A character string giving the name of the treatment or exposure variable. This variable is included in both the survival and longitudinal models. It is typically binary, but other codings may be used if supported by the model specification.
`outcome`	A character string giving the name of the longitudinal outcome variable to be modeled, such as a quality-of-life score.
`covariates`	A character vector of baseline covariate names to include as adjustment variables in both the survival and longitudinal submodels.
`os_time`	A character string giving the name of the overall survival time variable. This should represent observed event or censoring time on the prospective timescale.
`death`	A character string giving the name of the event indicator variable. This should typically be coded as `1` for death and `0` for censoring.
`t_pros`	A character string giving the name of the prospective follow-up time variable for each longitudinal observation. This is used together with survival time to define retrospective time from death or imputed death.
`random_effects`	A character vector specifying the random effects terms to include in the linear mixed effects model. For example, `c("1", "A")` specifies a random intercept and a random slope for treatment.
`n_impute`	A positive integer giving the number of imputed death times, and therefore the number of imputed datasets, to generate for censored subjects. Larger values generally improve stability of pooled estimates at the cost of increased computation time.
`n_inner_knot`	A positive integer giving the number of internal knots used for the natural spline basis that models the retrospective time trend in the longitudinal submodel. Increased number of knots may improve the model capacity to capture complicated trajectories at the risk of capturing noises.

Details

This function implements a two-stage approach for analyzing longitudinal outcomes that are conceptually aligned by time before death.

Cox proportional hazards model: $\lambda_i(t) = \exp\Big(\alpha_0(t) + A_i\alpha_A(t)\Big)$

Longitudinal linear mixed effect Model: $Y_i(t^*) = \beta_\mu(t^*) + A_i\beta_A(t^*) + X_i^T\psi_X + W_ib_i + \epsilon_i(t^*)$

In the first stage, a Cox proportional hazards model is fit using treatment and baseline covariates. For subjects who are censored, conditional survival distributions beyond the censoring time are derived from the fitted Cox model, and death times are multiply imputed.

In the second stage, each imputed dataset is reconstructed by replacing censored survival times with imputed death times and fit the longitudinal process on the retrospective timescale. Then a natural spline basis is constructed on the retrospective time scale to capture the baseline trajectory and treatment caused trajectory for outcome variable, and a linear mixed effects model is fit to the longitudinal outcome.

The longitudinal model includes treatment, baseline covariates, spline terms for retrospective time, and treatment-by-spline interactions. Random effects are included according to `random_effects` by user specification.

Fixed effects estimates and covariance matrices from the mixed models are pooled across imputations using Rubin-style combining rules. Variance components are summarized by simple averaging across imputations.

This function assumes that:

- `data_long` is in long format, with repeated outcome measurements per subject,
- survival and censoring information are available at the subject level,
- the longitudinal outcome is observed prior to death or censoring,
- covariates included in the models are measured consistently across rows.

Value

A named list with components:

survival_formula The survival model formula used in the Cox model.

survival_model The fitted `survival::coxph` object from stage 1.

longitudinal_formula The linear mixed effects model formula used in stage 2.

imputed_death_times A matrix of imputed death times for censored subjects, with one column per imputation and one row per censored subject.

pooled_longitudinal_model A list containing pooled results from the longitudinal mixed effects models:

- **fixed_effects** Pooled fixed effects estimates.
- **fixed_effects_vcov** Pooled covariance matrix of the fixed effects estimates.
- **residual_variance** Average residual variance across imputations.
- **random_effects_covariance** Average random effects covariance parameters across imputations.

spline_boundary_knots Boundary knots used for the natural spline basis.

spline_inner_knots Internal knots used for the natural spline basis.

n_inner_knots The number of internal knots used.

variable_name_map A lookup table showing original variable names and any internally transformed names used during preprocessing.

Examples

```
library(TTM)
data("Retro_data", package = "TTM")

fit <- RetroJM(
  data_long = Retro_data,
  id = "id",
  treatment = "A",
  outcome = "Y",
  covariates = c("X1", "X2"),
  os_time = "OS_time",
  death = "death",
  t_pros = "t_pros",
  random_effects = c("1", "A"),
  n_impute = 10,
  n_inner_knot = 3
)

fit$pooled_longitudinal_model$fixed_effects
fit$pooled_longitudinal_model$fixed_effects_vcov

plot_beta_mu(fit, xlim = c(0, quantile(Retro_data$OS_time, 0.7)), conf.level = 0.95)
plot_beta_A(fit, xlim = c(0, quantile(Retro_data$OS_time, 0.7)), conf.level = 0.95)

get_beta_mu(new_times = 1:10, results = fit, conf.level = 0.95)
get_beta_A(new_times = 1:10, results = fit, conf.level = 0.95)
```

---

test_beta_slope *Test for a change in slope at a knot of a linear-spline trajectory*

Description

Computes the change in slope of either the baseline trajectory or the treatment-effect trajectory around a knot and tests its significance. This function is for `TTM_linear()` fits with a B-spline basis (`spline_type = "bs"`) only.

Usage

```
test_beta_slope(new_times, fit, var_name)
```

Arguments

`new_times`	Numeric vector of knot positions. Needs to be of length 3 where the middle value is the knot and the other two values are slightly before and after the knot.
`fit`	A fitted `TTM_linear()` object
`var_name`	Either `"X0"` (baseline trajectory beta_mu) or `"A"` (treatment effect trajectory beta_A)

Value

A data.frame with columns `slope1` (slope before the knot), `slope2` (slope after the knot), `diff` (`slope2 - slope1`), `se_diff` (standard error of `diff`), and `p_value` (two-sided Wald test p-value).

Examples

```
fit_linear <- TTM_linear(
  data_long = TTM_data,
  id = "id",
  treatment = "A",
  outcome = "Y",
  covariates = c("X1", "X2"),
  os_time = "OS_time",
  event = "event",
  t_pros = "t_pros",
  knots = 6,
  spline_type = "bs"
)
test_beta_slope(new_times = c(5, 6, 7), fit = fit_linear, var_name = "X0")
test_beta_slope(new_times = c(5, 6, 7), fit = fit_linear, var_name = "A")
```

TTM — *Fit a weighted GEE terminal trend model on the retrospective time scale*

Description

Fits a retrospective-time longitudinal model using weighted generalized estimating equations (WGEE).

Retrospective time is the time from prospective follow-up time to the death time. A natural spline basis is constructed for retrospective time to allow flexible terminal-time trajectories. To account for informative dropout and administrative censoring, subject-specific weights are estimated. The weighted longitudinal mean model is then fit with a Gaussian GEE among subjects with observed death events.

Uncertainty for the regression coefficients is estimated using a delete-group jackknife.

Usage

```
TTM(
  data_long,
  id = "id",
  treatment = "A",
  outcome = "Y",
  covariates = c("X1", "X2"),
  os_time = "OS_time",
  event = "event",
  t_pros = "t_pros",
```

```
  jk_block_size = NULL,
  n_inner_knot_list = range(10),
  corstr = "ar1",
  criteria = "RJC"
)
```

Arguments

`data_long`	A long-format `data.frame` with one row per subject-visit. The data must include the columns specified by `id`, `treatment`, `outcome`, `covariates`, `os_time`, `event`, and `t_pros`, as well as `censoring_time` and `death_time`.
`id`	Character scalar giving the subject identifier column name.
`treatment`	Character scalar giving the treatment column name.
`outcome`	Character scalar giving the longitudinal outcome column name.
`covariates`	Character vector of baseline covariate column names.
`os_time`	Character scalar giving the observed survival time column name.
`event`	Character scalar giving the event indicator column name. The function assumes `1 = death`, `2 = dropout`, and all other values are treated as neither death nor dropout.
`t_pros`	Character scalar giving the prospective follow-up time column name.
`jk_block_size`	Positive integer giving the number of delete-group jackknife groups used to estimate coefficient variability. Must be greater than 1.
`n_inner_knot_list`	Positive integer vector giving the candidates of number of internal knots for the natural spline basis in retrospective time. The final value will be selected using criteria.
`corstr`	Character scalar specifying the working correlation structure passed to `geepack::geeglm()`, such as `"independence"`, `"exchangeable"`, or `"ar1"`.
`criteria`	Character scalar naming the model selection criterion, computed by `geer::geecriteria()`, used to choose the number of internal knots from `n_inner_knot_list`. One of `"QIC"`, `"CIC"`, `"RJC"`, `"QICu"`, or `"GESSC"`; the candidate with the smallest value is selected.

Details

The fitted longitudinal model includes:

$$E[Y(t^*)] = \beta_\mu(t^*) + A \times \beta_A(t^*) + \beta_X^T X$$

where $\beta_\mu(t^*)$ and $\beta_A(t^*)$ are represented by a natural spline basis, and $t^* = D_i - t_{pros}$ where D_i is the death time.

Dropout weights are derived from a Cox model for the hazard of dropout and empirical estimates of censoring. The GEE fit is obtained with weights. Only observations from subjects with `event == 1` are used in the final longitudinal model fit.

Value

A named list with components:

dropout_formula The formula used to fit the dropout Cox model.

dropout_model A fitted `survival::coxph` object for dropout.

longitudinal_formula The formula used in the weighted GEE fit.

weighted_gee A list containing:

- **fixed_effects** Jackknife mean of the GEE coefficient estimates.
- **fixed_effects_vcov** Delete-group jackknife covariance matrix for `fixed_effects`.
- **fixed_effects_sd** Standard errors from `fixed_effects_vcov`.
- **jackknife_coefficients** Matrix of leave-group-out coefficient estimates, one row per jackknife replicate.

spline_boundary_knots Boundary knots used for the retrospective-time natural spline basis.

spline_inner_knots Internal knots used for the retrospective-time natural spline basis.

n_inner_knots The requested number of internal spline knots.

variable_name_map Mapping between original variable names and the standardized names created during preprocessing.

weights A `data.frame` containing subject-level analysis weights.

method Character string identifying the fitted method, equal to `"TTM"`.

Examples

```
library(TTM)
data("TTM_data", package = "TTM")

fit <- TTM(
  data_long = TTM_data,
  id = "id",
  treatment = "A",
  outcome = "Y",
  covariates = c("X1", "X2"),
  os_time = "OS_time",
  event = "event",
  t_pros = "t_pros",
  jk_block_size = NULL,
  n_inner_knot_list = range(10),
  corstr = "ar1"
)

fit$weighted_gee$fixed_effects
fit$weighted_gee$fixed_effects_vcov

plot_beta_mu(
  fit,
  xlim = c(0, stats::quantile(TTM_data$OS_time, 0.7)),
  conf.level = 0.95
)
```

```
plot_beta_A(
  fit,
  xlim = c(0, stats::quantile(TTM_data$OS_time, 0.7)),
  conf.level = 0.95
)

get_beta_mu(new_times = 1:10, results = fit, conf.level = 0.95)
get_beta_A(new_times = 1:10, results = fit, conf.level = 0.95)
```

`TTM_data` *Simulated terminal trend data for* `TTM()`

Description

A simulated long-format data set with repeated measurements of a longitudinal outcome before death, dropout, or censoring. It is used in the examples and the vignette of `TTM()` and `TTM_linear()`. The script used to generate the data is in the `data-raw` folder of the source repository.

Usage

```
TTM_data
```

Format

A data frame with 3681 rows (1000 subjects) and 17 variables:

id Subject identifier.

A Binary treatment indicator (0 = control, 1 = treatment).

X1 Binary baseline covariate.

X2 Non-negative continuous baseline covariate.

u Subject-level random effect generated in the simulation.

b_longi Random intercept for the longitudinal outcome.

randomslope Random slope generated for the longitudinal outcome.

OS_time Observed time, the minimum of the death, dropout, and censoring times (months).

event Event indicator: 0 = censored, 1 = death, 2 = dropout.

death Death indicator (1 = death observed).

dropout Dropout indicator (1 = dropout observed).

death_time Simulated (latent) death time.

dropout_time Simulated (latent) dropout time.

censoring_time Simulated (latent) administrative censoring time.

t_pros Prospective measurement time since baseline.

t_retro True retrospective time, `death_time - t_pros`.

Y Longitudinal outcome.

Source

Simulated data.

Examples

```
data("TTM_data", package = "TTM")
head(TTM_data)
table(TTM_data$event[!duplicated(TTM_data$id)])
```

TTM_linear — *Fit a weighted GEE terminal trend model with a linear spline basis*

Description

Fits a retrospective-time longitudinal model using weighted generalized estimating equations (WGEE).

Retrospective time is the time from prospective follow-up time to the death time. A natural spline basis is constructed for retrospective time to allow flexible terminal-time trajectories. To account for informative dropout and administrative censoring, subject-specific weights are estimated. The weighted longitudinal mean model is then fit with a Gaussian GEE among subjects with observed death events.

Uncertainty for the regression coefficients is estimated using a delete-group jackknife.

Usage

```
TTM_linear(
  data_long,
  id = "id",
  treatment = NULL,
  outcome = "Y",
  covariates = c("X1", "X2"),
  os_time = "OS_time",
  event = "event",
  t_pros = "t_pros",
  knots = NULL,
  jk_block_size = NULL,
  corstr = "ar1",
  spline_type = "bs",
  criteria = "RJC"
)
```

Arguments

`data_long`	A long-format `data.frame` with one row per subject-visit. The data must include the columns specified by `id`, `treatment`, `outcome`, `covariates`, `os_time`, `event`, and `t_pros`, as well as `censoring_time` and `death_time`.
`id`	Character scalar giving the subject identifier column name.

`treatment`	Character scalar giving the treatment column name. Can only accept NULL or binary group indicator.
`outcome`	Character scalar giving the longitudinal outcome column name.
`covariates`	Character vector of baseline covariate column names.
`os_time`	Character scalar giving the observed survival time column name.
`event`	Character scalar giving the event indicator column name. The function assumes `1 = death`, `2 = dropout`, and all other values are treated as neither death nor dropout.
`t_pros`	Character scalar giving the prospective follow-up time column name.
`knots`	Knots specified in advance. If null, the knot is selected automatically
`jk_block_size`	Positive integer giving the number of delete-group jackknife groups used to estimate coefficient variability. Must be greater than 1.
`corstr`	Character scalar specifying the working correlation structure passed to `geepack::geeglm()`, such as `"independence"`, `"exchangeable"`, or `"ar1"`.
`spline_type`	Character scalar specifying the type of spline basis to use. Options are `"ns"` for natural splines or `"bs"` for B-splines. The default is `"bs"`.
`criteria`	Currently unused; accepted for consistency with `TTM()`.

Details

The fitted longitudinal model includes:

$$E[Y(t^*)] = \beta_\mu(t^*) + A \times \beta_A(t^*) + \beta_X^T X$$

where $\beta_\mu(t^*)$ and $\beta_A(t^*)$ are represented by a natural spline basis, and $t^* = D_i - t_{pros}$ where D_i is the death time.

Dropout weights are derived from a Cox model for the hazard of dropout and empirical estimates of censoring. The GEE fit is obtained with weights. Only observations from subjects with `event == 1` are used in the final longitudinal model fit.

Value

A named list with components:

dropout_formula The formula used to fit the dropout Cox model.

dropout_model A fitted `survival::coxph` object for dropout.

longitudinal_formula The formula used in the weighted GEE fit.

weighted_gee A list containing:

- **fixed_effects** Jackknife mean of the GEE coefficient estimates.
- **fixed_effects_vcov** Delete-group jackknife covariance matrix for `fixed_effects`.
- **fixed_effects_sd** Standard errors from `fixed_effects_vcov`.
- **jackknife_coefficients** Matrix of leave-group-out coefficient estimates, one row per jackknife replicate.

spline_boundary_knots Boundary knots used for the retrospective-time natural spline basis.

spline_inner_knots Internal knots used for the retrospective-time natural spline basis.

n_inner_knots The requested number of internal spline knots.

variable_name_map Mapping between original variable names and the standardized names created during preprocessing.

weights A `data.frame` containing subject-level analysis weights.

method Character string identifying the fitted method, equal to `"TTM"`.

Examples

```
library(TTM)
data("TTM_data", package = "TTM")

fit <- TTM_linear(
  data_long = TTM_data,
  id = "id",
  treatment = "A",
  outcome = "Y",
  covariates = c("X1", "X2"),
  os_time = "OS_time",
  event = "event",
  t_pros = "t_pros",
  knots = 6,
  jk_block_size = NULL,
  spline_type = "bs",
  corstr = "ar1"
)

fit$weighted_gee$fixed_effects
fit$weighted_gee$fixed_effects_vcov

plot_beta_mu(
  fit,
  xlim = c(0, stats::quantile(TTM_data$OS_time, 0.7)),
  conf.level = 0.95
)

plot_beta_A(
  fit,
  xlim = c(0, stats::quantile(TTM_data$OS_time, 0.7)),
  conf.level = 0.95
)

get_beta_mu(new_times = 1:10, results = fit, conf.level = 0.95)
get_beta_A(new_times = 1:10, results = fit, conf.level = 0.95)
```

Index

∗ **datasets**
    Retro_data, 8
    TTM_data, 16

get_beta_A, 2
get_beta_mu, 4

plot_beta_A, 5
plot_beta_mu, 7

Retro_data, 8
RetroJM, 9
RetroJM(), *8*

test_beta_slope, 12
TTM, 13
TTM(), *16*
TTM_data, *8*, 16
TTM_linear, 17
TTM_linear(), *16*