Package ‘pmrm’

January 30, 2026

Title Progression Models for Repeated Measures

Description A progression model for repeated measures (PMRM)
is a continuous-time nonlinear mixed-effects model for
longitudinal clinical trials in progressive diseases.
Unlike mixed models for repeated measures (MMRMs),
which estimate treatment effects as linear combinations
of additive effects on the outcome scale,
PMRMs characterize treatment effects
in terms of the underlying disease trajectory.
This framing yields clinically interpretable quantities
such as average time saved
and percent reduction in decline due to treatment.
This package implements frequentist PMRMs by
Raket (2022) <doi:10.1002/sim.9581> using
'RTMB' by Kristensen (2016) <doi:10.18637/jss.v070.i05>.

Version 0.0.2

License MIT + file LICENSE

URL `https://github.com/openpharma/pmrm`,
`https://openpharma.github.io/pmrm/`

BugReports `https://github.com/openpharma/pmrm/issues`

Depends R (>= 3.5.0)

Imports dplyr, generics, ggplot2, Matrix, nlme, RTMB (>= 1.8), rlang,
stats, tibble, tidyselect, utils, vctrs

Suggests knitr, markdown, rmarkdown, readr, scales, testthat (>=
3.0.0)

Encoding UTF-8

Language en-US

VignetteBuilder knitr

Config/testthat/edition 3

RoxygenNote 7.3.3

NeedsCompilation no

Author William Michael Landau [aut, cre] (ORCID: <https://orcid.org/0000-0003-1878-3253>),
Lars Lau Raket [aut] (ORCID: <https://orcid.org/0000-0001-7099-2314>),
Kasper Kristensen [aut] (ORCID: <https://orcid.org/0000-0003-3425-3762>),
Eli Lilly and Company [cph, fnd]

Maintainer William Michael Landau <will.landau.oss@gmail.com>

Repository CRAN

Date/Publication 2026-01-30 10:50:12 UTC

Contents

`AIC.pmrm_fit` *Akaike information criterion (AIC)*

Description

Extract the Akaike information criterion (AIC) of a progression model for repeated measures (PMRM).

Usage

```
## S3 method for class 'pmrm_fit'
AIC(object, ..., k = NULL)
```

Arguments

`object`	A fitted model object of class `"pmrm_fit"`.
`...`	Not used.
`k`	Not used. Must be `NULL`.

Value

Numeric scalar, the Akaike information criterion (AIC) of the fitted model.

See Also

Other model comparison: `BIC.pmrm_fit()`, `confint.pmrm_fit()`, `deviance.pmrm_fit()`, `glance.pmrm_fit()`, `logLik.pmrm_fit()`, `summary.pmrm_fit()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
AIC(fit)
```

BIC.pmrm_fit *Bayesian information criterion (BIC)*

Description

Extract the Bayesian information criterion (BIC) of a progression model for repeated measures (PMRM).

Usage

```
## S3 method for class 'pmrm_fit'
BIC(object, ..., k = NULL)
```

Arguments

`object`	A fitted model object of class "`pmrm_fit`".
`...`	Not used.
`k`	Not used. Must be `NULL`

Value

Numeric scalar, the Bayesian information criterion (BIC) of the fitted model.

See Also

Other model comparison: `AIC.pmrm_fit()`, `confint.pmrm_fit()`, `deviance.pmrm_fit()`, `glance.pmrm_fit()`, `logLik.pmrm_fit()`, `summary.pmrm_fit()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
BIC(fit)
```

coef.pmrm_fit *Treatment effect parameters*

Description

Extract the `theta` parameter from a progression model for repeated measures.

Usage

```
## S3 method for class 'pmrm_fit'
coef(object, ...)
```

Arguments

`object`	A fitted model object of class `"pmrm_fit"`.
`...`	Not used.

Details

See `vignette("models", package = "pmrm")` for details.

Value

For proportional models, a named vector of `theta` estimates with one element for each active study arm. For non-proportional models, a named matrix of `theta` with one row for each active study arm and one column for each post-baseline scheduled visit. Elements, rows, and columns are named with arm/visit names as appropriate.

See Also

Other estimates: `VarCorr.pmrm_fit()`, `pmrm_marginals()`, `tidy.pmrm_fit()`, `vcov.pmrm_fit()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
coef(fit)
```

`confint.pmrm_fit` *Confidence intervals of parameters*

Description

Compute confidence intervals of the family of model parameters specified in `parm`.

Usage

```
## S3 method for class 'pmrm_fit'
confint(object, parm = NULL, level = 0.95, ...)
```

Arguments

`object`	a fitted model object.
`parm`	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
`level`	the confidence level required.
`...`	additional argument(s) for methods.

Details

See `vignette("models", package = "pmrm")` for details.

Value

A numeric matrix with one row for each treatment effect parameter (`theta`) and named columns with the lower and upper bounds of 2-sided confidence intervals on the parameters.

See Also

Other model comparison: `AIC.pmrm_fit()`, `BIC.pmrm_fit()`, `deviance.pmrm_fit()`, `glance.pmrm_fit()`, `logLik.pmrm_fit()`, `summary.pmrm_fit()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
confint(fit)
```

deviance.pmrm_fit *Deviance*

Description

Extract the deviance (defined here as `-2 * log_likelihood`) of a fitted progression model for repeated measures.

Usage

```
## S3 method for class 'pmrm_fit'
deviance(object, ...)
```

Arguments

`object`	A fitted model object of class `"pmrm_fit"`.
`...`	Not used.

Value

Numeric scalar, the deviance.

See Also

Other model comparison: `AIC.pmrm_fit()`, `BIC.pmrm_fit()`, `confint.pmrm_fit()`, `glance.pmrm_fit()`, `logLik.pmrm_fit()`, `summary.pmrm_fit()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
deviance(fit)
```

`fitted.pmrm_fit` *Fitted values*

Description

Compute the fitted values of a fitted progression model for repeated measures.

Usage

```
## S3 method for class 'pmrm_fit'
fitted(object, data = object$data, adjust = TRUE, ...)
```

Arguments

`object`	A fitted model object of class `"pmrm_fit"`.
`data`	A `tibble` or data frame with one row per patient visit. This is the new data for making predictions. It must have all the same columns as the original you fit with the model, except that the outcome column can be entirely absent. `object$data` is an example dataset that will work. It is just like the original data, except that rows with missing responses are removed, and the remaining rows are sorted by patient ID and categorical scheduled visit.
`adjust`	`TRUE` or `FALSE`. `adjust = TRUE` returns estimates and inference for covariate-adjusted `mu_ij` values (defined in `vignette("models", package = "pmrm")`) for new data. `adjust = FALSE` instead returns inference on `mu_ij - W %*% gamma`, the non-covariate-adjusted predictions useful in plotting a continuous disease progression trajectory in `plot.pmrm_fit()`.
`...`	Not used.

Details

For `pmrm`, `fitted()` is much faster than `predict()` for large datasets, but the output only includes the estimates (no measures of uncertainty).

Value

A numeric vector of fitted values corresponding to the rows of the data supplied in the `data` argument.

See Also

Other predictions: `predict.pmrm_fit()`, `residuals.pmrm_fit()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
str(fitted(fit))
```

`glance.pmrm_fit` *Glance at a PMRM.*

Description

Return a one-row `tibble` of model comparison metrics for a fitted PMRM.

Usage

```
## S3 method for class 'pmrm_fit'
glance(x, ...)
```

Arguments

`x`	A fitted model x of class `"pmrm_fit"`.
`...`	Not used.

Value

A `tibble` with one row and columns with the following columns:

- `model`: `"decline"` or `"slowing"`.
- `parameterization`: `"proportional"` or `"nonproportional"`.
- `n_observations`: number of non-missing observations in the data.
- `n_parameters`: number of true model parameters.
- `log_likelihood`: maximized log likelihood of the model fit.
- `deviance`: deviance of the fitted model, defined here as `-2` * `log_likelihood`.
- `aic`: Akaike information criterion.
- `bic`: Bayesian information criterion.

This format is designed for easy comparison of multiple fitted models.

See Also

Other model comparison: `AIC.pmrm_fit()`, `BIC.pmrm_fit()`, `confint.pmrm_fit()`, `deviance.pmrm_fit()`, `logLik.pmrm_fit()`, `summary.pmrm_fit()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
glance(fit)
```

`logLik.pmrm_fit` *Extract the log likelihood.*

Description

Extract the maximized log likelihood of a progression model for repeated measures (PMRM).

Usage

```
## S3 method for class 'pmrm_fit'
logLik(object, ...)
```

Arguments

`object`	A fitted model object of class `"pmrm_fit"`.
`...`	Not used.

Value

Numeric scalar, the maximized log likelihood of the fitted model.

See Also

Other model comparison: `AIC.pmrm_fit()`, `BIC.pmrm_fit()`, `confint.pmrm_fit()`, `deviance.pmrm_fit()`, `glance.pmrm_fit()`, `summary.pmrm_fit()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
logLik(fit)
```

---

plot.pmrm_fit *Plot a fitted PMRM.*

Description

Plot a fitted progression model for repeated measures (PMRM) against the data.

Usage

```
## S3 method for class 'pmrm_fit'
plot(
  x,
  y = NULL,
  ...,
  confidence = 0.95,
  show_data = TRUE,
  show_marginals = TRUE,
  show_predictions = FALSE,
  facet = TRUE,
  alpha = 0.25
)
```

Arguments

`x`	A fitted model object of class `"pmrm_fit"` returned by a `pmrm` model-fitting function.
`y`	Not used.
`...`	Not used.
`confidence`	Numeric between 0 and 1, the confidence level to use in the 2-sided confidence intervals.

show_data	TRUE to plot data-based visit-specific data means and confidence intervals as boxes. FALSE to omit.
show_marginals	TRUE to plot model-based confidence intervals and estimates of marginal means as boxes and horizontal lines within those boxes, respectively. Uses pmrm_marginals() with the given level of confidence. FALSE to omit.
show_predictions	TRUE to plot expected outcomes and confidence bands with lines and shaded regions, respectively. Uses predict.pmrm_fit() with adjust = FALSE and the given level of confidence on the original dataset used to fit the model. Predictions on a full dataset are generally slow, so the default is FALSE.
facet	TRUE to facet the plot by study arm, FALSE to overlay everything in a single panel.
alpha	Numeric between 0 and 1, opacity level of the model-based confidence bands.

Details

The plot shows the following elements:

- Raw estimates and confidence intervals on the data, as boxes (if show_data is TRUE).
- Model-based estimates and confidence intervals as points and error bars, respectively (if show_marginals is TRUE).
- Continuous model-based estimates and confidence bands as lines and shaded regions, respectively (if show_predictions is TRUE).

Value

A ggplot object with the plot.

See Also

Other visualization: print.pmrm_fit()

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
plot(fit)
```

pmrm_estimates *Parameter estimates and confidence intervals*

Description

Report parameter estimates and confidence intervals for a progression model for repeated measures (PMRM).

Usage

```
pmrm_estimates(
  fit,
 parameter = c("theta", "beta", "alpha", "gamma", "sigma", "phi", "rho", "Sigma",
    "Lambda"),
  confidence = 0.95
)
```

Arguments

`fit`	A fitted model object of class `"pmrm_fit"` returned by a `pmrm` model-fitting function.
`parameter`	Character string, name of the type of model parameter to summarize. Must be one of `"beta"`, `"theta"`, `"alpha"`, `"gamma"`, `"sigma"`, `"rho"`, `"Sigma"`, or `"Lambda"`.
`confidence`	Numeric between 0 and 1, the confidence level to use in 2-sided normal confidence intervals.

Value

A `tibble` with one row for each scalar element of the selected model parameter and columns with estimates, standard errors, lower and upper bounds of two-sided normal confidence intervals, and indexing variables. If applicable, the indexing variables are `arm` and/or `visit` to indicate the study arm and study visit. If there is no obvious indexing factor in the data, then a generic integer `index` column is used. For covariance matrices, elements are identified with the `visit_row` and `visit_column` columns.

`beta` is not a true parameter. Instead, it is a function of `theta` and fixed at zero for the control arm and at baseline. At these marginals, the standard errors and confidence intervals for `beta` are `NA_real_`.

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_nonproportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_nonproportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
pmrm_estimates(fit, parameter = "beta")
pmrm_estimates(fit, parameter = "alpha")
```

`pmrm_marginals` *Marginal means*

Description

Report the estimates and standard errors of marginal means at each study arm and visit. The assumed visit times should have been given in the `marginals` argument of the model-fitting function. Use the `type` argument to choose marginal means of the outcomes, marginal estimates of change from baseline, and marginal estimates of treatment effects.

Usage

```
pmrm_marginals(fit, type = c("outcome", "change", "effect"), confidence = 0.95)
```

Arguments

`fit`	A `pmrm` fitted model object returned by a model-fitting function.
`type`	Character string. `"outcome"` reports marginal means on the outcome scale, `"change"` reports estimates of change from baseline, and `"effect"` reports estimates of treatment effects (change from baseline of each active arm minus that of the control arm.)
`confidence`	A numeric from 0 to 1 with the confidence level for confidence intervals.

Value

A `tibble` with one row per marginal mean and columns with the estimate, standard error, 2-sided confidence bounds, and indicator columns. Some estimates, standard errors, and confidence bounds may be `NA_real_` if they correspond to the reference level subtracted out in change-from-baseline or treatment effect calculations.

See Also

Other estimates: `VarCorr.pmrm_fit()`, `coef.pmrm_fit()`, `tidy.pmrm_fit()`, `vcov.pmrm_fit()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
pmrm_marginals(fit)
```

`pmrm_model_decline_nonproportional`

Fit the non-proportional decline model.

Description

Fit the non-proportional decline model to a clinical dataset on a progressive disease.

Usage

```
pmrm_model_decline_nonproportional(
  data,
  outcome,
  time,
  patient,
  visit,
  arm,
  covariates = ~0,
  visit_times = NULL,
  spline_knots = visit_times,
  spline_method = c("natural", "fmm"),
  reml = FALSE,
  hessian = c("divergence", "never", "always"),
  saddle = FALSE,
  control = list(eval.max = 4000L, iter.max = 4000L),
  initial_method = c("regression", "regression_control", "zero"),
  initial = NULL,
  silent = TRUE
)
```

Arguments

`data`	A data frame or `tibble` of clinical data.
`outcome`	Character string, name of the column in the data with the numeric outcome variable on the continuous scale. Could be a clinical measure of healthy or of disease severity. Baseline is part of the model, so the `outcome` should not already be a change from baseline. The vector of outcomes may have missing values, either with explicit `NA`s or with rows in the data missing for one or more visits.
`time`	Character string, name of the column in the data with the numeric time variable on the continuous scale. This time is the time since enrollment/randomization of each patient. A time value of 0 should indicate baseline.
`patient`	Character string, name of the column in the data with the patient ID. This vector could be a numeric, integer, factor, or character vector. `pmrm` automatically converts it into an unordered factor.
`visit`	Character string, name of the column in the data which indicates the study visit of each row. This column could be a numeric, integer, factor, or character vector. An ordered factor is highly recommended because `pmrm` with levels assumed to be in chronological order. The minimum visit must be baseline.
`arm`	Character string, name of the column in the data which indicates the study arm of each row. This column could be a numeric, integer, factor, or character vector. An ordered factor is highly recommended because `pmrm` automatically converts `data[[arm]]` into an ordered factor anyway. The minimum level is assumed to be the control arm.
`covariates`	Partial right-sided formula of concomitant terms in the model for covariate adjustment (e.g. by age, gender, biomarker status, etc.). Should not include main variables such as the values of `outcome`, `time`, `patient`, `visit`, or `arm`. The columns in the data referenced in the formula must not have any missing values. Set `covariates` to `~ 0` (default) to opt out of covariate adjustment. The intercept term is removed from the model matrix `W` whether or not the formula begins with '~ 0.
`visit_times`	Numeric vector, the continuous scheduled time of each study visit (since randomization). If `NULL`, each visit time is automatically set set to the median of the observed times at categorical visit in the data.
`spline_knots`	Numeric vector of spline knots on the continuous scale, including boundary knots.
`spline_method`	Character string, spline method to use for the base model. Must be `"natural"` or `"fmm"`. See `stats::splinefun()` for details.
`reml`	`TRUE` to fit the model with restricted maximum likelihood (REML), which involves integrating out fixed effects. `FALSE` to use unrestricted maximum likelihood. If `reml` is `TRUE`, then `hessian` is automatically set to `"never"`.
`hessian`	Character string controlling when to supply the Hessian matrix of the objective function to the optimizer `stats::nlminb()`. Supplying the Hessian usually slows down optimization but may improve convergence in some cases, particularly saddle points in the objective function. The `hessian` argument is automatically set to `"never"` whenever `reml` is `TRUE`. The `hessian` argument must be one of the following values:

- `"divergence"`: first try the model without supplying the Hessian. Then if the model does not converge, retry while supplying the Hessian.
- `"never"`: fit the model only once and do not supply the Hessian to `stats::nlminb()`.
- `"always"`: fit the model once and supply the Hessian to `stats::nlminb()`.

`saddle` `TRUE` to check if the optimization hit a saddle point, and if it did, treat the model fit as if it diverged. `FALSE` to skip this check for the sake of speed.

`control` A named list of control parameters passed directly to the `control` argument of `stats::nlminb()`.

`initial_method` Character string, name of the method for computing initial values. Ignored unless `initial` is `NULL`. Must have one of the following values:

- `"regression"`: sets the spline vertical distances `alpha` to the fitted values at the knots of a simple linear regression of the responses versus continuous time. Sets all the other true model parameters to 0.
- `"regression_control"`: like `"regression"` except we only use the data from the control group. Sets all the other true model parameters to 0.
- `"zero"`: sets all true model parameters to 0, including `alpha`.

`initial` If `initial` is a named list, then `pmrm` uses this list as the initial parameter values for the optimization. Otherwise, `pmrm` automatically computes the starting values using the method given in the `initial_method` argument (see below).

If `initial` is a list, then it must have the following named finite numeric elements conforming to all the true parameters defined in `vignette("models", package = "pmrm")`:

- `alpha`: a vector with the same length as `spline_knots`.
- `theta`: a matrix with `K - 1` rows and `J - 1` columns, where `K` is the number of study arms and `J` is the number of study visits.
- `gamma`: a vector with `V` elements, where `V` is the number of columns in the covariate adjustment model matrix `W`. If you are unsure of `V`, simply fit a test model (e.g. `fit <- pmrm_model_decline_nonproportional(...)`) and then check `ncol(fit$constants$W)`.
- `phi`: a vector with the same length as `visit_times` (which may be different from the length of `spline_knots`).
- `rho`: a vector with `J * (J - 1) / 2` elements, where `J` is the length of `visit_times`.

You can generate an example of the format of this list by fitting a test model (e.g. `fit <- pmrm_model_decline_nonproportional(...)`) and then extracting `fit$initial` or `fit$final`.

`silent` As MakeADFun.

Details

See `vignette("models", package = "pmrm")` for details.

Value

A `pmrm` fit object of class `c("pmrm_fit_decline", "pmrm_fit")`. For details, see the "pmrm fit objects" section of this help file.

pmrm fit objects

A "pmrm_fit" object is a classed list returned by modeling functions. It has the following named elements:

- data: a tibble, the input data with the missing outcomes removed and the remaining rows sorted by patient and visit within patient. The data has a special "pmrm_data" class and should not be modified by the user.
- constants: a list of fixed quantities from the data that the objective function uses in the optimization. Most of these quantities are defined in the modeling and simulation vignettes in the pmrm package. n_visits is a positive integer vector with the number of non-missing outcomes for each patient.
- options: a list of low-level model-fitting options for RTMB.
- objective: the objective function for the optimization. Returns the minus log likelihood of the model. The arguments are (1) a list of constants, and (2) a list of model parameters. Both arguments have strict formatting requirements. For (1), see the constants element of the fitted model object. For (2), see initial or final. model$fn (from the model element of the fitted model object) contains a copy of the objective function that only takes a parameter list. (The constants are in the closure of model$fn.)
- model: model object returned by RTMB::MakeADFun() with the compiled objective function and gradient. The elements can be supplied to an optimization routine in R such as stats::nlminb().
- optimization: the object returned by stats::nlminb() to perform the optimization that estimates the parameters. optimization$convergence equals 0 if an only if the model converges.
- report: object returned by RTMB::sdreport() which has information on the standard deviations of model parameters.
- initial: a list of model parameters initial values. Includes true parameters like theta and alpha but does not include derived parameters like beta or sigma. You can supply your own list of similarly formatted initial values to the initial argument of the modeling function you choose.
- final: a list of model parameter estimates after optimization, but not including derived parameters like beta or sigma. The format is exactly the same as initial (see above) to help deal with divergent model fits. If your model fit diverged and you want to try resume the optimization with slightly better values, you can modify values in final and supply the result to the initial argument of the modeling function.
- estimates: a full list of parameter estimates, including derived parameters
- standard_errors: a list of parameter standard errors.
- metrics: a list of high-level model metrics, including:
  - n_observations: positive integer scalar, number of non-missing observations in the data.
  - n_parameters: positive integer scalar, number of model parameters in the data. Includes true parameters like theta but excludes downstream functions of parameters such as beta.
  - log_likelihood: numeric scalar, the maximized log likelihood of the fitted model.

  - `deviance`: deviance of the fitted model, defined here as `-2 * log_likelihood`.
  - `aic`: numeric scalar, the Akaike information criterion of the fitted model.
  - `bic`: numeric scalar, the Bayesian information criterion of the fitted model.

- `spline`: a vectorized function that accepts continuous time `x` and returns the value of the fitted spline `f(x | spline_knots, alpha)` at time `x` given the user-specified knots `spline_knots` and the maximum likelihood estimates of `alpha`. Useful for diagnosing strange behavior in the fitted spline. If the spline behaves oddly, especially extrapolating beyond the range of the time points, please consider adjusting the knots `spline_knots` or the initial values of `alpha` when refitting the model.

See Also

Other models: `pmrm_model_decline_proportional()`, `pmrm_model_slowing_nonproportional()`, `pmrm_model_slowing_proportional()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_nonproportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_nonproportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
str(fit$estimates)
names(fit)
```

`pmrm_model_decline_proportional`

Fit the proportional decline model.

Description

Fit the proportional decline model to a clinical dataset on a progressive disease.

Usage

```
pmrm_model_decline_proportional(
  data,
  outcome,
  time,
  patient,
  visit,
  arm,
  covariates = ~0,
  visit_times = NULL,
  spline_knots = visit_times,
  spline_method = c("natural", "fmm"),
  reml = FALSE,
  hessian = c("divergence", "never", "always"),
  saddle = FALSE,
  control = list(eval.max = 4000L, iter.max = 4000L),
  initial_method = c("regression", "regression_control", "zero"),
  initial = NULL,
  silent = TRUE
)
```

Arguments

`data`	A data frame or `tibble` of clinical data.
`outcome`	Character string, name of the column in the data with the numeric outcome variable on the continuous scale. Could be a clinical measure of healthy or of disease severity. Baseline is part of the model, so the `outcome` should not already be a change from baseline. The vector of outcomes may have missing values, either with explicit `NA`s or with rows in the data missing for one or more visits.
`time`	Character string, name of the column in the data with the numeric time variable on the continuous scale. This time is the time since enrollment/randomization of each patient. A time value of 0 should indicate baseline.
`patient`	Character string, name of the column in the data with the patient ID. This vector could be a numeric, integer, factor, or character vector. `pmrm` automatically converts it into an unordered factor.
`visit`	Character string, name of the column in the data which indicates the study visit of each row. This column could be a numeric, integer, factor, or character vector. An ordered factor is highly recommended because `pmrm` with levels assumed to be in chronological order. The minimum visit must be baseline.
`arm`	Character string, name of the column in the data which indicates the study arm of each row. This column could be a numeric, integer, factor, or character vector. An ordered factor is highly recommended because `pmrm` automatically converts `data[[arm]]` into an ordered factor anyway. The minimum level is assumed to be the control arm.
`covariates`	Partial right-sided formula of concomitant terms in the model for covariate adjustment (e.g. by age, gender, biomarker status, etc.). Should not include main variables such as the values of `outcome`, `time`, `patient`, `visit`, or `arm`. The columns in the data referenced in the formula must not have any missing values. Set `covariates` to `~ 0` (default) to opt out of covariate adjustment. The intercept term is removed from the model matrix `W` whether or not the formula begins with '~ 0.

`visit_times` Numeric vector, the continuous scheduled time of each study visit (since randomization). If `NULL`, each visit time is automatically set set to the median of the observed times at categorical visit in the data.

`spline_knots` Numeric vector of spline knots on the continuous scale, including boundary knots.

`spline_method` Character string, spline method to use for the base model. Must be `"natural"` or `"fmm"`. See `stats::splinefun()` for details.

`reml` `TRUE` to fit the model with restricted maximum likelihood (REML), which involves integrating out fixed effects. `FALSE` to use unrestricted maximum likelihood. If `reml` is `TRUE`, then `hessian` is automatically set to `"never"`.

`hessian` Character string controlling when to supply the Hessian matrix of the objective function to the optimizer `stats::nlminb()`. Supplying the Hessian usually slows down optimization but may improve convergence in some cases, particularly saddle points in the objective function.

The `hessian` argument is automatically set to `"never"` whenever `reml` is `TRUE`.

The `hessian` argument must be one of the following values:

- `"divergence"`: first try the model without supplying the Hessian. Then if the model does not converge, retry while supplying the Hessian.
- `"never"`: fit the model only once and do not supply the Hessian to `stats::nlminb()`.
- `"always"`: fit the model once and supply the Hessian to `stats::nlminb()`.

`saddle` `TRUE` to check if the optimization hit a saddle point, and if it did, treat the model fit as if it diverged. `FALSE` to skip this check for the sake of speed.

`control` A named list of control parameters passed directly to the `control` argument of `stats::nlminb()`.

`initial_method` Character string, name of the method for computing initial values. Ignored unless `initial` is `NULL`. Must have one of the following values:

- `"regression"`: sets the spline vertical distances `alpha` to the fitted values at the knots of a simple linear regression of the responses versus continuous time. Sets all the other true model parameters to 0.
- `"regression_control"`: like `"regression"` except we only use the data from the control group. Sets all the other true model parameters to 0.
- `"zero"`: sets all true model parameters to 0, including `alpha`.

`initial` If `initial` is a named list, then `pmrm` uses this list as the initial parameter values for the optimization. Otherwise, `pmrm` automatically computes the starting values using the method given in the `initial_method` argument (see below).

If `initial` is a list, then it must have the following named finite numeric elements conforming to all the true parameters defined in `vignette("models", package = "pmrm")`:

- `alpha`: a vector with the same length as `spline_knots`.
- `theta`: a vector with `K - 1` elements, where `K` is the number of study arms.
- `gamma`: a vector with `V` elements, where `V` is the number of columns in the covariate adjustment model matrix `W`. If you are unsure of `V`, simply fit a test model (e.g. `fit <- pmrm_modepmrm_model_decline_proportionall_decline(...)`) and then check `ncol(fit$constants$W)`.

- phi: a vector with the same length as `visit_times` (which may be different from the length of `spline_knots`).
- rho: a vector with `J * (J - 1) / 2` elements, where `J` is the length of `visit_times`.

You can generate an example of the format of this list by fitting a test model (e.g. `fit <- pmrm_model_decline_proportional(...)`) and then extracting `fit$initial` or `fit$final`.

`silent`	As MakeADFun.

Details

See `vignette("models", package = "pmrm")` for details.

Value

A `pmrm` fit object of class `c("pmrm_fit_decline", "pmrm_fit")`. For details, see the "pmrm fit objects" section of this help file.

pmrm fit objects

A `"pmrm_fit"` object is a classed list returned by modeling functions. It has the following named elements:

- `data`: a `tibble`, the input data with the missing outcomes removed and the remaining rows sorted by patient and visit within patient. The data has a special `"pmrm_data"` class and should not be modified by the user.
- `constants`: a list of fixed quantities from the data that the objective function uses in the optimization. Most of these quantities are defined in the modeling and simulation vignettes in the `pmrm` package. `n_visits` is a positive integer vector with the number of non-missing outcomes for each patient.
- `options`: a list of low-level model-fitting options for `RTMB`.
- `objective`: the objective function for the optimization. Returns the minus log likelihood of the model. The arguments are (1) a list of constants, and (2) a list of model parameters. Both arguments have strict formatting requirements. For (1), see the `constants` element of the fitted model object. For (2), see `initial` or `final`. `model$fn` (from the `model` element of the fitted model object) contains a copy of the objective function that only takes a parameter list. (The constants are in the closure of `model$fn`.)
- `model`: model object returned by `RTMB::MakeADFun()` with the compiled objective function and gradient. The elements can be supplied to an optimization routine in R such as `stats::nlminb()`.
- `optimization`: the object returned by `stats::nlminb()` to perform the optimization that estimates the parameters. `optimization$convergence` equals 0 if an only if the model converges.
- `report`: object returned by `RTMB::sdreport()` which has information on the standard deviations of model parameters.
- `initial`: a list of model parameters initial values. Includes true parameters like `theta` and `alpha` but does not include derived parameters like `beta` or `sigma`. You can supply your own list of similarly formatted initial values to the `initial` argument of the modeling function you choose.

- `final`: a list of model parameter estimates after optimization, but not including derived parameters like `beta` or `sigma`. The format is exactly the same as `initial` (see above) to help deal with divergent model fits. If your model fit diverged and you want to try resume the optimization with slightly better values, you can modify values in `final` and supply the result to the `initial` argument of the modeling function.
- `estimates`: a full list of parameter estimates, including derived parameters
- `standard_errors`: a list of parameter standard errors.
- `metrics`: a list of high-level model metrics, including:
  - `n_observations`: positive integer scalar, number of non-missing observations in the data.
  - `n_parameters`: positive integer scalar, number of model parameters in the data. Includes true parameters like `theta` but excludes downstream functions of parameters such as `beta`.
  - `log_likelihood`: numeric scalar, the maximized log likelihood of the fitted model.
  - `deviance`: deviance of the fitted model, defined here as `-2 * log_likelihood`.
  - `aic`: numeric scalar, the Akaike information criterion of the fitted model.
  - `bic`: numeric scalar, the Bayesian information criterion of the fitted model.
- `spline`: a vectorized function that accepts continuous time `x` and returns the value of the fitted spline `f(x | spline_knots, alpha)` at time `x` given the user-specified knots `spline_knots` and the maximum likelihood estimates of `alpha`. Useful for diagnosing strange behavior in the fitted spline. If the spline behaves oddly, especially extrapolating beyond the range of the time points, please consider adjusting the knots `spline_knots` or the initial values of `alpha` when refitting the model.

See Also

Other models: `pmrm_model_decline_nonproportional()`, `pmrm_model_slowing_nonproportional()`, `pmrm_model_slowing_proportional()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
str(fit$estimates)
names(fit)
```

pmrm_model_slowing_nonproportional

Fit the non-proportional slowing model.

Description

Fit the non-proportional slowing model to a clinical dataset on a progressive disease.

Usage

```
pmrm_model_slowing_nonproportional(
  data,
  outcome,
  time,
  patient,
  visit,
  arm,
  covariates = ~0,
  visit_times = NULL,
  spline_knots = visit_times,
  spline_method = c("natural", "fmm"),
  reml = FALSE,
  hessian = c("divergence", "never", "always"),
  saddle = FALSE,
  control = list(eval.max = 4000L, iter.max = 4000L),
  initial_method = c("regression", "regression_control", "zero"),
  initial = NULL,
  silent = TRUE
)
```

Arguments

`data`	A data frame or `tibble` of clinical data.
`outcome`	Character string, name of the column in the data with the numeric outcome variable on the continuous scale. Could be a clinical measure of healthy or of disease severity. Baseline is part of the model, so the `outcome` should not already be a change from baseline. The vector of outcomes may have missing values, either with explicit `NA`s or with rows in the data missing for one or more visits.
`time`	Character string, name of the column in the data with the numeric time variable on the continuous scale. This time is the time since enrollment/randomization of each patient. A time value of 0 should indicate baseline.
`patient`	Character string, name of the column in the data with the patient ID. This vector could be a numeric, integer, factor, or character vector. `pmrm` automatically converts it into an unordered factor.

`visit`	Character string, name of the column in the data which indicates the study visit of each row. This column could be a numeric, integer, factor, or character vector. An ordered factor is highly recommended because `pmrm` with levels assumed to be in chronological order. The minimum visit must be baseline.
`arm`	Character string, name of the column in the data which indicates the study arm of each row. This column could be a numeric, integer, factor, or character vector. An ordered factor is highly recommended because `pmrm` automatically converts `data[[arm]]` into an ordered factor anyway. The minimum level is assumed to be the control arm.
`covariates`	Partial right-sided formula of concomitant terms in the model for covariate adjustment (e.g. by age, gender, biomarker status, etc.). Should not include main variables such as the values of `outcome`, `time`, `patient`, `visit`, or `arm`. The columns in the data referenced in the formula must not have any missing values. Set `covariates` to `~ 0` (default) to opt out of covariate adjustment. The intercept term is removed from the model matrix `W` whether or not the formula begins with '~ 0.
`visit_times`	Numeric vector, the continuous scheduled time of each study visit (since randomization). If `NULL`, each visit time is automatically set set to the median of the observed times at categorical visit in the data.
`spline_knots`	Numeric vector of spline knots on the continuous scale, including boundary knots.
`spline_method`	Character string, spline method to use for the base model. Must be `"natural"` or `"fmm"`. See `stats::splinefun()` for details.
`reml`	`TRUE` to fit the model with restricted maximum likelihood (REML), which involves integrating out fixed effects. `FALSE` to use unrestricted maximum likelihood. If `reml` is `TRUE`, then `hessian` is automatically set to `"never"`.
`hessian`	Character string controlling when to supply the Hessian matrix of the objective function to the optimizer `stats::nlminb()`. Supplying the Hessian usually slows down optimization but may improve convergence in some cases, particularly saddle points in the objective function. The `hessian` argument is automatically set to `"never"` whenever `reml` is `TRUE`. The `hessian` argument must be one of the following values: • `"divergence"`: first try the model without supplying the Hessian. Then if the model does not converge, retry while supplying the Hessian. • `"never"`: fit the model only once and do not supply the Hessian to `stats::nlminb()`. • `"always"`: fit the model once and supply the Hessian to `stats::nlminb()`.
`saddle`	`TRUE` to check if the optimization hit a saddle point, and if it did, treat the model fit as if it diverged. `FALSE` to skip this check for the sake of speed.
`control`	A named list of control parameters passed directly to the `control` argument of `stats::nlminb()`.
`initial_method`	Character string, name of the method for computing initial values. Ignored unless `initial` is `NULL`. Must have one of the following values: • `"regression"`: sets the spline vertical distances `alpha` to the fitted values at the knots of a simple linear regression of the responses versus continuous time. Sets all the other true model parameters to 0.

- `"regression_control"`: like `"regression"` except we only use the data from the control group. Sets all the other true model parameters to 0.
- `"zero"`: sets all true model parameters to 0, including `alpha`.

`initial` If `initial` is a named list, then `pmrm` uses this list as the initial parameter values for the optimization. Otherwise, `pmrm` automatically computes the starting values using the method given in the `initial_method` argument (see below).

If `initial` is a list, then it must have the following named finite numeric elements conforming to all the true parameters defined in `vignette("models", package = "pmrm")`:

- `alpha`: a vector with the same length as `spline_knots`.
- `theta`: a matrix with `K - 1` rows and `J - 1` columns, where `K` is the number of study arms and `J` is the number of study visits.
- `gamma`: a vector with `V` elements, where `V` is the number of columns in the covariate adjustment model matrix `W`. If you are unsure of `V`, simply fit a test model (e.g. `fit <- pmrm_model_slowing_nonproportional(...)`) and then check `ncol(fit$constants$W)`.
- `phi`: a vector with the same length as `visit_times` (which may be different from the length of `spline_knots`).
- `rho`: a vector with `J * (J - 1) / 2` elements, where `J` is the length of `visit_times`.

You can generate an example of the format of this list by fitting a test model (e.g. `fit <- pmrm_model_slowing_nonproportional(...)`) and then extracting `fit$initial` or `fit$final`.

`silent` As MakeADFun.

Details

See `vignette("models", package = "pmrm")` for details.

Value

A `pmrm` fit object of class `c("pmrm_fit_slowing", "pmrm_fit")`. For details, see the "pmrm fit objects" section of this help file.

pmrm fit objects

A `"pmrm_fit"` object is a classed list returned by modeling functions. It has the following named elements:

- `data`: a `tibble`, the input data with the missing outcomes removed and the remaining rows sorted by patient and visit within patient. The data has a special `"pmrm_data"` class and should not be modified by the user.
- `constants`: a list of fixed quantities from the data that the objective function uses in the optimization. Most of these quantities are defined in the modeling and simulation vignettes in the `pmrm` package. `n_visits` is a positive integer vector with the number of non-missing outcomes for each patient.
- `options`: a list of low-level model-fitting options for `RTMB`.

- `objective`: the objective function for the optimization. Returns the minus log likelihood of the model. The arguments are (1) a list of constants, and (2) a list of model parameters. Both arguments have strict formatting requirements. For (1), see the `constants` element of the fitted model object. For (2), see `initial` or `final`. `model$fn` (from the `model` element of the fitted model object) contains a copy of the objective function that only takes a parameter list. (The constants are in the closure of `model$fn`.)
- `model`: model object returned by `RTMB::MakeADFun()` with the compiled objective function and gradient. The elements can be supplied to an optimization routine in R such as `stats::nlminb()`.
- `optimization`: the object returned by `stats::nlminb()` to perform the optimization that estimates the parameters. `optimization$convergence` equals 0 if an only if the model converges.
- `report`: object returned by `RTMB::sdreport()` which has information on the standard deviations of model parameters.
- `initial`: a list of model parameters initial values. Includes true parameters like `theta` and `alpha` but does not include derived parameters like `beta` or `sigma`. You can supply your own list of similarly formatted initial values to the `initial` argument of the modeling function you choose.
- `final`: a list of model parameter estimates after optimization, but not including derived parameters like `beta` or `sigma`. The format is exactly the same as `initial` (see above) to help deal with divergent model fits. If your model fit diverged and you want to try resume the optimization with slightly better values, you can modify values in `final` and supply the result to the `initial` argument of the modeling function.
- `estimates`: a full list of parameter estimates, including derived parameters
- `standard_errors`: a list of parameter standard errors.
- `metrics`: a list of high-level model metrics, including:
  - `n_observations`: positive integer scalar, number of non-missing observations in the data.
  - `n_parameters`: positive integer scalar, number of model parameters in the data. Includes true parameters like `theta` but excludes downstream functions of parameters such as `beta`.
  - `log_likelihood`: numeric scalar, the maximized log likelihood of the fitted model.
  - `deviance`: deviance of the fitted model, defined here as `-2 * log_likelihood`.
  - `aic`: numeric scalar, the Akaike information criterion of the fitted model.
  - `bic`: numeric scalar, the Bayesian information criterion of the fitted model.
- `spline`: a vectorized function that accepts continuous time `x` and returns the value of the fitted spline `f(x | spline_knots, alpha)` at time `x` given the user-specified knots `spline_knots` and the maximum likelihood estimates of `alpha`. Useful for diagnosing strange behavior in the fitted spline. If the spline behaves oddly, especially extrapolating beyond the range of the time points, please consider adjusting the knots `spline_knots` or the initial values of `alpha` when refitting the model.

See Also

Other models: `pmrm_model_decline_nonproportional()`, `pmrm_model_decline_proportional()`, `pmrm_model_slowing_proportional()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_slowing_nonproportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_slowing_nonproportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
str(fit$estimates)
names(fit)
```

`pmrm_model_slowing_proportional`

Fit the proportional slowing model.

Description

Fit the proportional slowing model to a clinical dataset on a progressive disease.

Usage

```
pmrm_model_slowing_proportional(
  data,
  outcome,
  time,
  patient,
  visit,
  arm,
  covariates = ~0,
  visit_times = NULL,
  spline_knots = visit_times,
  spline_method = c("natural", "fmm"),
  reml = FALSE,
  hessian = c("divergence", "never", "always"),
  saddle = FALSE,
  control = list(eval.max = 4000L, iter.max = 4000L),
  initial_method = c("regression", "regression_control", "zero"),
  initial = NULL,
  silent = TRUE
)
```

Arguments

`data`	A data frame or `tibble` of clinical data.
`outcome`	Character string, name of the column in the data with the numeric outcome variable on the continuous scale. Could be a clinical measure of healthy or of disease severity. Baseline is part of the model, so the `outcome` should not already be a change from baseline. The vector of outcomes may have missing values, either with explicit `NA`s or with rows in the data missing for one or more visits.
`time`	Character string, name of the column in the data with the numeric time variable on the continuous scale. This time is the time since enrollment/randomization of each patient. A time value of 0 should indicate baseline.
`patient`	Character string, name of the column in the data with the patient ID. This vector could be a numeric, integer, factor, or character vector. `pmrm` automatically converts it into an unordered factor.
`visit`	Character string, name of the column in the data which indicates the study visit of each row. This column could be a numeric, integer, factor, or character vector. An ordered factor is highly recommended because `pmrm` with levels assumed to be in chronological order. The minimum visit must be baseline.
`arm`	Character string, name of the column in the data which indicates the study arm of each row. This column could be a numeric, integer, factor, or character vector. An ordered factor is highly recommended because `pmrm` automatically converts `data[[arm]]` into an ordered factor anyway. The minimum level is assumed to be the control arm.
`covariates`	Partial right-sided formula of concomitant terms in the model for covariate adjustment (e.g. by age, gender, biomarker status, etc.). Should not include main variables such as the values of `outcome`, `time`, `patient`, `visit`, or `arm`. The columns in the data referenced in the formula must not have any missing values. Set `covariates` to `~ 0` (default) to opt out of covariate adjustment. The intercept term is removed from the model matrix `W` whether or not the formula begins with '~ 0.
`visit_times`	Numeric vector, the continuous scheduled time of each study visit (since randomization). If `NULL`, each visit time is automatically set set to the median of the observed times at categorical visit in the data.
`spline_knots`	Numeric vector of spline knots on the continuous scale, including boundary knots.
`spline_method`	Character string, spline method to use for the base model. Must be `"natural"` or `"fmm"`. See `stats::splinefun()` for details.
`reml`	`TRUE` to fit the model with restricted maximum likelihood (REML), which involves integrating out fixed effects. `FALSE` to use unrestricted maximum likelihood. If `reml` is `TRUE`, then `hessian` is automatically set to `"never"`.
`hessian`	Character string controlling when to supply the Hessian matrix of the objective function to the optimizer `stats::nlminb()`. Supplying the Hessian usually slows down optimization but may improve convergence in some cases, particularly saddle points in the objective function. The `hessian` argument is automatically set to `"never"` whenever `reml` is `TRUE`. The `hessian` argument must be one of the following values:

- `"divergence"`: first try the model without supplying the Hessian. Then if the model does not converge, retry while supplying the Hessian.
- `"never"`: fit the model only once and do not supply the Hessian to `stats::nlminb()`.
- `"always"`: fit the model once and supply the Hessian to `stats::nlminb()`.

`saddle` `TRUE` to check if the optimization hit a saddle point, and if it did, treat the model fit as if it diverged. `FALSE` to skip this check for the sake of speed.

`control` A named list of control parameters passed directly to the `control` argument of `stats::nlminb()`.

`initial_method` Character string, name of the method for computing initial values. Ignored unless `initial` is `NULL`. Must have one of the following values:

- `"regression"`: sets the spline vertical distances `alpha` to the fitted values at the knots of a simple linear regression of the responses versus continuous time. Sets all the other true model parameters to 0.
- `"regression_control"`: like `"regression"` except we only use the data from the control group. Sets all the other true model parameters to 0.
- `"zero"`: sets all true model parameters to 0, including `alpha`.

`initial` If `initial` is a named list, then `pmrm` uses this list as the initial parameter values for the optimization. Otherwise, `pmrm` automatically computes the starting values using the method given in the `initial_method` argument (see below).

If `initial` is a list, then it must have the following named finite numeric elements conforming to all the true parameters defined in `vignette("models", package = "pmrm")`:

- `alpha`: a vector with the same length as `spline_knots`.
- `theta`: a vector with `K - 1` elements, where `K` is the number of study arms.
- `gamma`: a vector with `V` elements, where `V` is the number of columns in the covariate adjustment model matrix `W`. If you are unsure of `V`, simply fit a test model (e.g. `fit <- pmrm_model_slowing_proportional(...)`) and then check `ncol(fit$constants$W)`.
- `phi`: a vector with the same length as `visit_times` (which may be different from the length of `spline_knots`).
- `rho`: a vector with `J * (J - 1) / 2` elements, where `J` is the length of `visit_times`.

You can generate an example of the format of this list by fitting a test model (e.g. `fit <- pmrm_model_slowing_proportional(...)`) and then extracting `fit$initial` or `fit$final`.

`silent` As MakeADFun.

Details

See `vignette("models", package = "pmrm")` for details.

Value

A `pmrm` fit object of class `c("pmrm_fit_slowing", "pmrm_fit")`. For details, see the "pmrm fit objects" section of this help file.

pmrm fit objects

A `"pmrm_fit"` object is a classed list returned by modeling functions. It has the following named elements:

- `data`: a `tibble`, the input data with the missing outcomes removed and the remaining rows sorted by patient and visit within patient. The data has a special `"pmrm_data"` class and should not be modified by the user.
- `constants`: a list of fixed quantities from the data that the objective function uses in the optimization. Most of these quantities are defined in the modeling and simulation vignettes in the `pmrm` package. `n_visits` is a positive integer vector with the number of non-missing outcomes for each patient.
- `options`: a list of low-level model-fitting options for `RTMB`.
- `objective`: the objective function for the optimization. Returns the minus log likelihood of the model. The arguments are (1) a list of constants, and (2) a list of model parameters. Both arguments have strict formatting requirements. For (1), see the `constants` element of the fitted model object. For (2), see `initial` or `final`. `model$fn` (from the `model` element of the fitted model object) contains a copy of the objective function that only takes a parameter list. (The constants are in the closure of `model$fn`.)
- `model`: model object returned by `RTMB::MakeADFun()` with the compiled objective function and gradient. The elements can be supplied to an optimization routine in R such as `stats::nlminb()`.
- `optimization`: the object returned by `stats::nlminb()` to perform the optimization that estimates the parameters. `optimization$convergence` equals 0 if an only if the model converges.
- `report`: object returned by `RTMB::sdreport()` which has information on the standard deviations of model parameters.
- `initial`: a list of model parameters initial values. Includes true parameters like `theta` and `alpha` but does not include derived parameters like `beta` or `sigma`. You can supply your own list of similarly formatted initial values to the `initial` argument of the modeling function you choose.
- `final`: a list of model parameter estimates after optimization, but not including derived parameters like `beta` or `sigma`. The format is exactly the same as `initial` (see above) to help deal with divergent model fits. If your model fit diverged and you want to try resume the optimization with slightly better values, you can modify values in `final` and supply the result to the `initial` argument of the modeling function.
- `estimates`: a full list of parameter estimates, including derived parameters
- `standard_errors`: a list of parameter standard errors.
- `metrics`: a list of high-level model metrics, including:
  - `n_observations`: positive integer scalar, number of non-missing observations in the data.
  - `n_parameters`: positive integer scalar, number of model parameters in the data. Includes true parameters like `theta` but excludes downstream functions of parameters such as `beta`.
  - `log_likelihood`: numeric scalar, the maximized log likelihood of the fitted model.

  - `deviance`: deviance of the fitted model, defined here as `-2 * log_likelihood`.
  - `aic`: numeric scalar, the Akaike information criterion of the fitted model.
  - `bic`: numeric scalar, the Bayesian information criterion of the fitted model.
- `spline`: a vectorized function that accepts continuous time `x` and returns the value of the fitted spline `f(x | spline_knots, alpha)` at time `x` given the user-specified knots `spline_knots` and the maximum likelihood estimates of `alpha`. Useful for diagnosing strange behavior in the fitted spline. If the spline behaves oddly, especially extrapolating beyond the range of the time points, please consider adjusting the knots `spline_knots` or the initial values of `alpha` when refitting the model.

See Also

Other models: `pmrm_model_decline_nonproportional()`, `pmrm_model_decline_proportional()`, `pmrm_model_slowing_nonproportional()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_slowing_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_slowing_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
str(fit$estimates)
names(fit)
```

`pmrm_simulate_decline_nonproportional` *Simulate non-proportional decline model.*

Description

Simulate a dataset from the non-proportional decline model.

Usage

```
pmrm_simulate_decline_nonproportional(
  patients = 300,
  visit_times = seq(from = 0, to = 4, by = 1),
  spline_knots = visit_times,
  spline_method = c("natural", "fmm"),
  tau = 0,
  alpha = log(spline_knots + 1),
  beta = cbind(0, rbind(0, rep(0.2, length(visit_times) - 1L), rep(0.3,
    length(visit_times) - 1L))),
  gamma = numeric(0L),
  sigma = rep(1, length(visit_times)),
  rho = rep(0, length(visit_times) * (length(visit_times) - 1L)/2L)
)
```

Arguments

`patients`	Positive integer scalar, total number of patients in the output dataset. Patients are allocated (roughly) uniformly across the study arms.
`visit_times`	Numeric vector, the continuous scheduled time after randomization of each study visit.
`spline_knots`	Numeric vector of spline knots on the continuous scale, including boundary knots.
`spline_method`	Character string, spline method to use for the base model. Must be `"natural"` or `"fmm"`. See `stats::splinefun()` for details.
`tau`	Positive numeric scalar, standard deviation for jittering the simulated time points. Defaults to 0 so that the observed continuous times are just the scheduled visit times.
`alpha`	Numeric vector of spline coefficients for simulating the mean function `f(t_{ij}	spline_knots, alph` Must have `length(spline_knots)` elements.
`beta`	Numeric matrix with one row for each study arm (including the control arm) and one column for each study visit (including baseline). See `vignette("models", package = "pmrm")` for details on this parameter.
`gamma`	Numeric vector of model coefficients for covariate adjustment. The simulation functions in `pmrm` simulate `length(gamma)` columns for the covariate adjustment model matrix `W`. Set to `numeric(0)` to omit covariates.
`sigma`	A positive numeric vector of visit-level standard deviation parameters.
`rho`	A finite numeric vector of correlation parameters. Must have length `J * (J - 1) / 2`, where `J` is `length(visit_times)`. The full covariance matrix `Sigma` is given by `diag(sigma) %*% RTMB::unstructured(length(sigma))$corr(rho) %*% diag(sigma)`.

Details

See `vignette("models", package = "pmrm")` for details.

Value

A `tibble` of clinical data simulated from the model. See the "Simulated data" section of this help file for details.

Simulated data

The datasets returned from the simulation functions have one row per patient visit and the following columns which conform to the notation from `vignette("models", package = "pmrm")`:

- `patient`: Character vector of patient ID labels.
- `visit`: Ordered factor of clinical visits with labels included. `min(visit)` indicates the baseline visit.
- `arm`: Ordered factor of study arms with visits included. `min(arm)` indicates the control arm.
- `i`: integer ID of each patient.
- `j`: integer ID of each clinical visit. `j == 1` at baseline.
- `k`: integer ID of the study arm of patient `i`. `k == 1` for the control arm.
- `y`: clinical outcomes.
- `t`: observed continuous time since baseline.
- `beta`: the scalar component of the treatment effect parameter `beta` defined for patient `i`.
- `mu`: expected clinical outcome at the given patient visit.
- `w_*`: columns of the covariate adjustment model matrix `W`.
- `e`: residuals.

See Also

Other simulations: `pmrm_simulate_decline_proportional()`, `pmrm_simulate_slowing_nonproportional()`, `pmrm_simulate_slowing_proportional()`

Examples

```
pmrm_simulate_decline_nonproportional()
```

---

pmrm_simulate_decline_proportional

Simulate proportional decline model.

Description

Simulate a dataset from the proportional decline model.

Usage

```
pmrm_simulate_decline_proportional(
  patients = 300,
  visit_times = seq(from = 0, to = 4, by = 1),
  spline_knots = visit_times,
  spline_method = c("natural", "fmm"),
  tau = 0,
  alpha = log(spline_knots + 1),
  beta = c(0, 0.1, 0.2),
  gamma = numeric(0L),
  sigma = rep(1, length(visit_times)),
  rho = rep(0, length(visit_times) * (length(visit_times) - 1L)/2L)
)
```

Arguments

`patients`	Positive integer scalar, total number of patients in the output dataset. Patients are allocated (roughly) uniformly across the study arms.
`visit_times`	Numeric vector, the continuous scheduled time after randomization of each study visit.
`spline_knots`	Numeric vector of spline knots on the continuous scale, including boundary knots.
`spline_method`	Character string, spline method to use for the base model. Must be `"natural"` or `"fmm"`. See `stats::splinefun()` for details.
`tau`	Positive numeric scalar, standard deviation for jittering the simulated time points. Defaults to 0 so that the observed continuous times are just the scheduled visit times.
`alpha`	Numeric vector of spline coefficients for simulating the mean function `f(t_{ij}	spline_knots, alph` Must have `length(spline_knots)` elements.
`beta`	Numeric vector with one element per study arm (including the control arm). See `vignette("models", package = "pmrm")` for details on this parameter.
`gamma`	Numeric vector of model coefficients for covariate adjustment. The simulation functions in `pmrm` simulate `length(gamma)` columns for the covariate adjustment model matrix `W`. Set to `numeric(0)` to omit covariates.
`sigma`	A positive numeric vector of visit-level standard deviation parameters.
`rho`	A finite numeric vector of correlation parameters. Must have length `J * (J - 1) / 2`, where `J` is `length(visit_times)`. The full covariance matrix `Sigma` is given by `diag(sigma) %*% RTMB::unstructured(length(sigma))$corr(rho) %*% diag(sigma)`.

Details

See `vignette("models", package = "pmrm")` for details.

Value

A `tibble` of clinical data simulated from the model. See the "Simulated data" section of this help file for details.

Simulated data

The datasets returned from the simulation functions have one row per patient visit and the following columns which conform to the notation from `vignette("models", package = "pmrm")`:

- `patient`: Character vector of patient ID labels.

- `visit`: Ordered factor of clinical visits with labels included. `min(visit)` indicates the baseline visit.
- `arm`: Ordered factor of study arms with visits included. `min(arm)` indicates the control arm.
- `i`: integer ID of each patient.
- `j`: integer ID of each clinical visit. `j == 1` at baseline.
- `k`: integer ID of the study arm of patient `i`. `k == 1` for the control arm.
- `y`: clinical outcomes.
- `t`: observed continuous time since baseline.
- `beta`: the scalar component of the treatment effect parameter `beta` defined for patient `i`.
- `mu`: expected clinical outcome at the given patient visit.
- `w_*`: columns of the covariate adjustment model matrix `W`.
- `e`: residuals.

See Also

Other simulations: `pmrm_simulate_decline_nonproportional()`, `pmrm_simulate_slowing_nonproportional()`, `pmrm_simulate_slowing_proportional()`

Examples

```
pmrm_simulate_decline_proportional()
```

---

pmrm_simulate_slowing_nonproportional

Simulate non-proportional slowing model.

Description

Simulate a dataset from the non-proportional slowing model.

Usage

```
pmrm_simulate_slowing_nonproportional(
  patients = 300,
  visit_times = seq(from = 0, to = 4, by = 1),
  spline_knots = visit_times,
  spline_method = c("natural", "fmm"),
  tau = 0,
  alpha = log(spline_knots + 1),
  beta = cbind(0, rbind(0, rep(0.2, length(visit_times) - 1L), rep(0.3,
    length(visit_times) - 1L))),
  gamma = numeric(0L),
  sigma = rep(1, length(visit_times)),
  rho = rep(0, length(visit_times) * (length(visit_times) - 1L)/2L)
)
```

Arguments

`patients`	Positive integer scalar, total number of patients in the output dataset. Patients are allocated (roughly) uniformly across the study arms.
`visit_times`	Numeric vector, the continuous scheduled time after randomization of each study visit.
`spline_knots`	Numeric vector of spline knots on the continuous scale, including boundary knots.
`spline_method`	Character string, spline method to use for the base model. Must be `"natural"` or `"fmm"`. See `stats::splinefun()` for details.
`tau`	Positive numeric scalar, standard deviation for jittering the simulated time points. Defaults to 0 so that the observed continuous times are just the scheduled visit times.
`alpha`	Numeric vector of spline coefficients for simulating the mean function `f(t_{ij} | spline_knots, alph` Must have `length(spline_knots)` elements.
`beta`	Numeric matrix with one row for each study arm (including the control arm) and one column for each study visit (including baseline). See `vignette("models", package = "pmrm")` for details on this parameter.
`gamma`	Numeric vector of model coefficients for covariate adjustment. The simulation functions in `pmrm` simulate `length(gamma)` columns for the covariate adjustment model matrix `W`. Set to `numeric(0)` to omit covariates.
`sigma`	A positive numeric vector of visit-level standard deviation parameters.
`rho`	A finite numeric vector of correlation parameters. Must have length `J * (J - 1) / 2`, where `J` is `length(visit_times)`. The full covariance matrix `Sigma` is given by `diag(sigma) %*% RTMB::unstructured(length(sigma))$corr(rho) %*% diag(sigma)`.

Details

See `vignette("models", package = "pmrm")` for details.

Value

A `tibble` of clinical data simulated from the slowing model. See the "Simulated data" section of this help file for details.

Simulated data

The datasets returned from the simulation functions have one row per patient visit and the following columns which conform to the notation from `vignette("models", package = "pmrm")`:

- `patient`: Character vector of patient ID labels.
- `visit`: Ordered factor of clinical visits with labels included. `min(visit)` indicates the baseline visit.
- `arm`: Ordered factor of study arms with visits included. `min(arm)` indicates the control arm.
- `i`: integer ID of each patient.

- `j`: integer ID of each clinical visit. `j == 1` at baseline.
- `k`: integer ID of the study arm of patient `i`. `k == 1` for the control arm.
- `y`: clinical outcomes.
- `t`: observed continuous time since baseline.
- `beta`: the scalar component of the treatment effect parameter `beta` defined for patient `i`.
- `mu`: expected clinical outcome at the given patient visit.
- `w_*`: columns of the covariate adjustment model matrix `W`.
- `e`: residuals.

See Also

Other simulations: `pmrm_simulate_decline_nonproportional()`, `pmrm_simulate_decline_proportional()`, `pmrm_simulate_slowing_proportional()`

Examples

```
pmrm_simulate_slowing_nonproportional()
```

---

pmrm_simulate_slowing_proportional
Simulate proportional slowing model.

Description

Simulate a dataset from the proportional slowing model.

Usage

```
pmrm_simulate_slowing_proportional(
  patients = 300,
  visit_times = seq(from = 0, to = 4, by = 1),
  spline_knots = visit_times,
  spline_method = c("natural", "fmm"),
  tau = 0,
  alpha = log(spline_knots + 1),
  beta = c(0, 0.1, 0.2),
  gamma = numeric(0L),
  sigma = rep(1, length(visit_times)),
  rho = rep(0, length(visit_times) * (length(visit_times) - 1L)/2L)
)
```

Arguments

`patients`	Positive integer scalar, total number of patients in the output dataset. Patients are allocated (roughly) uniformly across the study arms.
`visit_times`	Numeric vector, the continuous scheduled time after randomization of each study visit.
`spline_knots`	Numeric vector of spline knots on the continuous scale, including boundary knots.
`spline_method`	Character string, spline method to use for the base model. Must be `"natural"` or `"fmm"`. See `stats::splinefun()` for details.
`tau`	Positive numeric scalar, standard deviation for jittering the simulated time points. Defaults to 0 so that the observed continuous times are just the scheduled visit times.
`alpha`	Numeric vector of spline coefficients for simulating the mean function `f(t_{ij} | spline_knots, alph` Must have `length(spline_knots)` elements.
`beta`	Numeric vector with one element per study arm (including the control arm). See `vignette("models", package = "pmrm")` for details on this parameter.
`gamma`	Numeric vector of model coefficients for covariate adjustment. The simulation functions in `pmrm` simulate `length(gamma)` columns for the covariate adjustment model matrix `W`. Set to `numeric(0)` to omit covariates.
`sigma`	A positive numeric vector of visit-level standard deviation parameters.
`rho`	A finite numeric vector of correlation parameters. Must have length `J * (J - 1) / 2`, where `J` is `length(visit_times)`. The full covariance matrix `Sigma` is given by `diag(sigma) %*% RTMB::unstructured(length(sigma))$corr(rho) %*% diag(sigma)`.

Details

See `vignette("models", package = "pmrm")` for details.

Value

A `tibble` of clinical data simulated from the slowing model. See the "Simulated data" section of this help file for details.

Simulated data

The datasets returned from the simulation functions have one row per patient visit and the following columns which conform to the notation from `vignette("models", package = "pmrm")`:

- `patient`: Character vector of patient ID labels.
- `visit`: Ordered factor of clinical visits with labels included. `min(visit)` indicates the baseline visit.
- `arm`: Ordered factor of study arms with visits included. `min(arm)` indicates the control arm.
- `i`: integer ID of each patient.
- `j`: integer ID of each clinical visit. `j == 1` at baseline.

- k: integer ID of the study arm of patient i. k == 1 for the control arm.
- y: clinical outcomes.
- t: observed continuous time since baseline.
- beta: the scalar component of the treatment effect parameter beta defined for patient i.
- mu: expected clinical outcome at the given patient visit.
- w_*: columns of the covariate adjustment model matrix W.
- e: residuals.

See Also

Other simulations: pmrm_simulate_decline_nonproportional(), pmrm_simulate_decline_proportional(), pmrm_simulate_slowing_nonproportional()

Examples

```
pmrm_simulate_slowing_proportional()
```

---

predict.pmrm_fit *Predict new outcomes*

Description

Return the expected values, standard errors, and confidence intervals of new outcomes.

Usage

```
## S3 method for class 'pmrm_fit'
predict(object, data = object$data, adjust = TRUE, confidence = 0.95, ...)
```

Arguments

object	A fitted model object of class "pmrm_fit".
data	A tibble or data frame with one row per patient visit. This is the new data for making predictions. It must have all the same columns as the original you fit with the model, except that the outcome column can be entirely absent. object$data is an example dataset that will work. It is just like the original data, except that rows with missing responses are removed, and the remaining rows are sorted by patient ID and categorical scheduled visit.
adjust	TRUE or FALSE. adjust = TRUE returns estimates and inference for covariate-adjusted mu_ij values (defined in vignette("models", package = "pmrm")) for new data. adjust = FALSE instead returns inference on mu_ij - W %*% gamma, the non-covariate-adjusted predictions useful in plotting a continuous disease progression trajectory in plot.pmrm_fit().
confidence	Numeric between 0 and 1, the confidence level to use in the 2-sided confidence intervals.
...	Not used.

Value

A `tibble` with one row for each row in the `data` argument and columns `"estimate"`, `"standard_error"`, `"lower"`, and `"upper"`. Columns `"lower"` and `"upper"` are lower and upper bounds of 2-sided confidence intervals on the means. (The confidence intervals are not actually truly prediction intervals because they do not include variability from residuals.)

See Also

Other predictions: `fitted.pmrm_fit()`, `residuals.pmrm_fit()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
new_data <- pmrm_simulate_decline_proportional(
  patients = 1,
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
new_data$y <- NULL # Permitted but not strictly necessary.
predict(fit, new_data)
```

`print.pmrm_fit` *Print a fitted PMRM.*

Description

Print a fitted progression model for repeated measures (PMRM).

Usage

```
## S3 method for class 'pmrm_fit'
print(x, digits = 3L, ...)
```

Arguments

`x`	A fitted progression model for repeated measures (PMRM).
`digits`	Non-negative integer, number of digits for rounding.
`...`	Not used.

Value

`NULL` (invisibly). Called for side effects (printing to the R console).

See Also

Other visualization: `plot.pmrm_fit()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
print(fit)
```

`residuals.pmrm_fit` `pmrm` *residuals.*

Description

Compute the residuals (responses minus fitted values) of a fitted progression model for repeated measures.

Usage

```
## S3 method for class 'pmrm_fit'
residuals(object, ..., data = object$data, adjust = TRUE)
```

Arguments

`object`	A fitted model object of class `"pmrm_fit"`.
`...`	Not used.
`data`	A `tibble` or data frame with one row per patient visit. This is the new data for making predictions. It must have all the same columns as the original you fit with the model, except that the outcome column can be entirely absent. `object$data` is an example dataset that will work. It is just like the original data, except that rows with missing responses are removed, and the remaining rows are sorted by patient ID and categorical scheduled visit.
`adjust`	`TRUE` or `FALSE`. `adjust = TRUE` returns estimates and inference for covariate-adjusted `mu_ij` values (defined in `vignette("models", package = "pmrm")`) for new data. `adjust = FALSE` instead returns inference on `mu_ij - W %*% gamma`, the non-covariate-adjusted predictions useful in plotting a continuous disease progression trajectory in `plot.pmrm_fit()`.

Value

A numeric vector of residuals corresponding to the rows of the data supplied in the `data` argument.

See Also

Other predictions: `fitted.pmrm_fit()`, `predict.pmrm_fit()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
str(residuals(fit))
```

`summary.pmrm_fit` *Summarize a PMRM.*

Description

Summarize a progression model for repeated measures (PMRM).

Usage

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

Arguments

`object`	A fitted model object of class `"pmrm_fit"`.
`...`	Not used.

Value

A `tibble` with one row and columns with the following columns:

- `model`: `"decline"` or `"slowing"`.
- `parameterization`: `"proportional"` or `"nonproportional"`.
- `n_observations`: number of non-missing observations in the data.
- `n_parameters`: number of true model parameters.
- `log_likelihood`: maximized log likelihood of the model fit.
- `deviance`: deviance of the fitted model, defined here as `-2 * log_likelihood`.
- `aic`: Akaike information criterion.
- `bic`: Bayesian information criterion.

This format is designed for easy comparison of multiple fitted models.

See Also

Other model comparison: `AIC.pmrm_fit()`, `BIC.pmrm_fit()`, `confint.pmrm_fit()`, `deviance.pmrm_fit()`, `glance.pmrm_fit()`, `logLik.pmrm_fit()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
summary(fit)
```

tidy.pmrm_fit — *Tidy a fitted PMRM.*

Description

Return tidy parameter summaries of a progression model for repeated measures (PMRM).

Usage

```
## S3 method for class 'pmrm_fit'
tidy(x, ...)
```

Arguments

`x`	A fitted progression model for repeated measures (PMRM).
`...`	Not used.

Value

A tidy `tibble` with one row for each treatment effect model parameter (`theta`) and columns with the parameter name (study arm and/or visit it corresponds to), estimate, and standard error. This format aligns with the `tidy()` method of similar fitted models in R.

See Also

Other estimates: `VarCorr.pmrm_fit()`, `coef.pmrm_fit()`, `pmrm_marginals()`, `vcov.pmrm_fit()`

Examples

```
  set.seed(0L)
  simulation <- pmrm_simulate_decline_proportional(
    visit_times = seq_len(5L) - 1,
    gamma = c(1, 2)
  )
  fit <- pmrm_model_decline_proportional(
    data = simulation,
    outcome = "y",
    time = "t",
    patient = "patient",
    visit = "visit",
    arm = "arm",
    covariates = ~ w_1 + w_2
  )
  tidy(fit)
```

VarCorr.pmrm_fit *Estimated covariance matrix*

Description

Extract estimated covariance matrix among visits within patients.

Usage

```
## S3 method for class 'pmrm_fit'
VarCorr(x, sigma = NA, ...)
```

Arguments

x	A fitted model object of class "pmrm_fit".
sigma	Not used for pmrm.
...	Not used.

Value

A matrix J rows and J columns, where J is the number of scheduled visits in the clinical trial.

See Also

Other estimates: coef.pmrm_fit(), pmrm_marginals(), tidy.pmrm_fit(), vcov.pmrm_fit()

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
VarCorr(fit)
```

`vcov.pmrm_fit` *Treatment effect parameter covariance matrix*

Description

Extract the covariance matrix of the treatment effect parameters of a progression model for repeated measures.

Usage

```
## S3 method for class 'pmrm_fit'
vcov(object, ...)
```

Arguments

`object`	A fitted model object of class `"pmrm_fit"`.
`...`	Not used.

Value

A matrix with covariance of each pair of `theta` parameters. Rows and columns are labeled (by just study arm for proportional models, arm and visit for non-proportional models.)

See Also

Other estimates: `VarCorr.pmrm_fit()`, `coef.pmrm_fit()`, `pmrm_marginals()`, `tidy.pmrm_fit()`

Examples

```
set.seed(0L)
simulation <- pmrm_simulate_decline_proportional(
  visit_times = seq_len(5L) - 1,
  gamma = c(1, 2)
)
fit <- pmrm_model_decline_proportional(
  data = simulation,
  outcome = "y",
  time = "t",
  patient = "patient",
  visit = "visit",
  arm = "arm",
  covariates = ~ w_1 + w_2
)
vcov(fit)
```

Index