Package: progme (via r-universe)

October 22, 2026

Type Package

Title Extended Progression Models for Repeated Measures

Version 0.1.0

Description Extended progression models for repeated measures in progressive disease clinical trials. Provides nonlinear mixed-effects progression models (linear, plateau, Emax, exponential decay), delayed treatment onset estimation by profile likelihood, joint longitudinal-survival models for informative dropout, competing-risks extensions, biomarker-informed adaptive dosing rules, and simulation-based power and sample-size calculation. The progression-model view of treatment effects, in which effects are characterised on the underlying disease trajectory rather than as additive shifts on the outcome scale, follows Raket (2022) <doi:10.1002/sim.9581>. The nonlinear mixed-effects machinery is built on the 'nlme' implementation described by Pinheiro and Bates (2000, ISBN 978-0-387-98957-0), and the joint modelling of longitudinal and time-to-event data follows Rizopoulos (2012, ISBN 978-1-4398-7286-4).

License GPL (>= 3)

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0)

Imports nlme, survival, ggplot2, stats, graphics, utils

Suggests rstan, testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

Config/testthat/edition 3

URL https://github.com/MukulBijalwan/progme

BugReports https://github.com/MukulBijalwan/progme/issues

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Mukul Bijalwan [aut, cre] (ORCID: <https://orcid.org/0009-0001-3040-6912>), Gunjan Aggrwal [aut], Mukul Jain [aut]

Maintainer Mukul Bijalwan <mukulbijalwan555@gmail.com>

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

Date/Publication 2026-10-22 10:30:09 UTC

RemoteUrl https://github.com/cran/progme

RemoteRef HEAD

RemoteSha 75ba93ebdccb5e03957c219c82d920ec4d1c14d1

Contents

adni_simulated . . . 3
biomarker_dosing_rule . . . 4
check_convergence . . . 4
default_priors . . . 5
delayed_effect . . . 6
emax_biomarker . . . 7
find_sample_size . . . 7
fit_pmrm . . . 8
goodness_of_fit . . . 9
initialize_subjects . . . 9
parametric_bootstrap . . . 10
parkinsons_simulated . . . 11
pkpd_concentration . . . 12
plot.pmrm_delayed . . . 12
pmrm_adaptive_trial . . . 13
pmrm_competing . . . 14
pmrm_delayed . . . 15
pmrm_joint . . . 17
pmrm_nlme . . . 18
population_trajectory . . . 19
power_pmrm . . . 20
predict.pmrm_nlme . . . 21
print.pmrm_delayed . . . 22
scenario_base . . . 22
simulate_biomarker . . . 23
simulate_joint_trial . . . 24
simulate_progression_trial . . . 25
subject_summaries . . . 26
summary.pmrm_competing . . . 27
summary.pmrm_delayed . . . 27
summary.pmrm_joint . . . 28
to_event_data . . . 29
vpc . . . 30

Index **31**

`adni_simulated` *Simulated ADNI-style Alzheimer's trial data*

Description

Longitudinal data from a simulated ADNI-style Alzheimer's disease trial with linear progression, a treatment effect with delayed onset, and subject-level dropout information.

Usage

```
adni_simulated
```

Format

A data frame with 2160 rows and 6 variables:

rid Subject identifier.

time Visit time in months.

trt Treatment indicator: 0 = control, 1 = active.

adas_cog ADAS-Cog score (higher values indicate worse cognition).

time_to_dropout Subject-level time to dropout.

dropout Dropout indicator: 1 = dropped out, 0 = censored.

Source

Simulated with `simulate_progression_trial()`; see `data-raw/create_data.R`.

Examples

```
head(adni_simulated)
table(adni_simulated$trt)
summary(adni_simulated$adas_cog)
```

biomarker_dosing_rule *Biomarker-informed adaptive dosing rule*

Description

Adjusts the dose one step up/down when the measured biomarker falls outside the target range, otherwise maintains the current dose.

Usage

```
biomarker_dosing_rule(
  biomarker_value,
  target_range = c(80, 120),
  dose_levels = c(0, 50, 100, 150, 200),
  current_dose_idx = 3
)
```

Arguments

`biomarker_value`
: Observed biomarker value.

`target_range`
: Two numeric values giving the acceptable range.

`dose_levels`
: Ordered vector of allowed doses.

`current_dose_idx`
: Index of the subject's current dose.

Value

List with `new_dose`, `new_dose_idx`, and `adjustment` ("up", "down", or "maintain").

Examples

```
biomarker_dosing_rule(130)  # above target range -> step the dose up
biomarker_dosing_rule(70)   # below target range -> step the dose down
biomarker_dosing_rule(100)  # inside the range -> keep the current dose
```

check_convergence *Convergence checks*

Description

For Stan/NIMBLE draws, computes R-hat and effective sample size when the `coda` inputs are supplied as a list of chains. For nlme fits, reports optimization convergence code and random-effects variance.

Usage

```
check_convergence(object, ...)

## Default S3 method:
check_convergence(object, ...)
```

Arguments

`object`	A fitted model (nlme-based or an mcmc.list / list of chains).
`...`	Unused.

Value

List of diagnostic values.

Examples

```
sim <- simulate_progression_trial(n_control = 20, n_treatment = 20, seed = 1)
fit <- pmrm_nlme(y ~ time * trt, data = sim)
check_convergence(fit)$converged
# R-hat for a list of scalar chains
check_convergence(list(rnorm(200), rnorm(200), rnorm(200)))$rhat
```

`default_priors` *Default priors for Bayesian progression models*

Description

Default priors for Bayesian progression models

Usage

```
default_priors(scale = 1)
```

Arguments

`scale`	Outcome scale factor (e.g., SD of the response) used to make weakly-informative priors scale-aware.

Value

Named list of prior family and parameter values for fixed effects, random-effect SDs, residual SD, association parameters, and survival shape.

Examples

```
default_priors(scale = 5)
```

delayed_effect *Delayed treatment effect function*

Description

Computes the fractional treatment effect at time `time` given treatment initiation time `trt_time` and a delay `delay`.

Usage

```
delayed_effect(
  time,
  trt_time = 0,
  delay = 0,
  shape = c("exponential", "step", "linear"),
  lambda = 0.1
)
```

Arguments

`time`	Numeric vector of observation times.
`trt_time`	Treatment initiation time (default 0).
`delay`	Delay before onset begins (tau).
`shape`	One of "exponential", "step", "linear".
`lambda`	Onset rate for the exponential shape.

Value

Numeric vector in [0, 1] giving the fraction of the full effect realized at each time.

Examples

```
t <- seq(0, 12, by = 2)
delayed_effect(t, delay = 4, shape = "step")
delayed_effect(t, delay = 0, shape = "exponential", lambda = 0.1)
```

emax_biomarker *Emax biomarker model*

Description

Emax biomarker model

Usage

```
emax_biomarker(conc, b_baseline = 100, emax = 30, ec50 = 5)
```

Arguments

conc	Drug concentration.
b_baseline	Baseline biomarker level.
emax	Maximum effect.
ec50	Concentration giving half-maximal effect.

Value

Numeric vector of the same length as `conc` giving the expected biomarker level, `b_baseline + emax * conc / (ec50 + conc)`. Values increase from `b_baseline` at zero concentration towards the asymptote `b_baseline + emax` at high concentrations.

Examples

```
conc <- seq(0, 20, by = 2)
emax_biomarker(conc, b_baseline = 100, emax = 30, ec50 = 5)
```

find_sample_size *Find sample size by simulation*

Description

Find sample size by simulation

Usage

```
find_sample_size(target_power = 0.8, n_grid = seq(50, 300, by = 25), ...)
```

Arguments

target_power	Desired power.
n_grid	Grid of per-arm sample sizes to evaluate.
...	Arguments passed to `power_pmrm()`.

Value

List with `optimal_n` and a `power_curve` data frame.

Examples

```
ss <- find_sample_size(target_power = 0.8, n_grid = c(25, 50),
                       effect_size = -1.5, delay = 0, n_sims = 5,
                       measurement_times = seq(0, 24, by = 12), seed = 11)
ss$optimal_n
ss$power_curve
```

`fit_pmrm` *Unified estimation wrapper*

Description

Unified estimation wrapper

Usage

```
fit_pmrm(model = c("nlme", "delayed", "joint"), ...)
```

Arguments

`model`	Model family: "nlme", "delayed", or "joint".
`...`	Arguments forwarded to the corresponding fitting function.

Value

The fitted model object.

Examples

```
sim <- simulate_progression_trial(n_control = 20, n_treatment = 20, seed = 1)
fit <- fit_pmrm("nlme", y ~ time * trt, data = sim)
class(fit)
```

goodness_of_fit *Goodness-of-fit summary*

Description

Goodness-of-fit summary

Usage

```
goodness_of_fit(fit, bins = 10)
```

Arguments

fit	A fitted pmrm_nlme object.
bins	Number of bins for residual quantile grouping.

Value

List with population residuals, conditional (EB) residuals, RMSE, and information criteria.

Examples

```
sim <- simulate_progression_trial(n_control = 20, n_treatment = 20, seed = 1)
fit <- pmrm_nlme(y ~ time * trt, data = sim)
g <- goodness_of_fit(fit)
g$rmse
g$AIC
```

initialize_subjects *Initialize subjects for an adaptive trial simulation*

Description

Initialize subjects for an adaptive trial simulation

Usage

```
initialize_subjects(
  n_subjects,
  n_visits = 12,
  dose_levels = c(0, 50, 100, 150, 200),
  start_dose_idx = 3,
  trt_prob = 0.5,
  seed = NULL
)
```

Arguments

`n_subjects`	Number of subjects.
`n_visits`	Number of planned visits.
`dose_levels`	Allowed dose levels.
`start_dose_idx`	Starting dose index.
`trt_prob`	Probability of assignment to active treatment.
`seed`	Random seed.

Value

Data frame of subject-level attributes.

Examples

```
initialize_subjects(6, n_visits = 4, seed = 42)
```

`parametric_bootstrap` *Parametric bootstrap for confidence intervals*

Description

Refits the supplied model on `n_boot` datasets simulated from the fitted model's estimated parameters (resampling residuals conditional on the fixed-effects mean structure).

Usage

```
parametric_bootstrap(
  fit,
  statistic = function(f) stats::coef(f)[1],
  n_boot = 200,
  seed = NULL,
  level = 0.95
)
```

Arguments

`fit`	A `pmrm_nlme` object.
`statistic`	Function of the refit returning the scalar of interest.
`n_boot`	Number of bootstrap replicates.
`seed`	Random seed.
`level`	Confidence level.

Value

List with `estimate`, `ci`, and the bootstrap draws.

Examples

```
sim <- simulate_progression_trial(n_control = 20, n_treatment = 20, seed = 1)
fit <- pmrm_nlme(y ~ time * trt, data = sim)
boot <- parametric_bootstrap(fit, n_boot = 5, seed = 1)
boot$estimate
boot$ci
```

---

parkinsons_simulated *Simulated Parkinson's trial data with informative dropout*

Description

Longitudinal UPDRS data from a simulated Parkinson's disease trial in which progression drives an informative dropout process, so that faster progressors leave the trial earlier. Intended for use with `pmrm_joint()` and `pmrm_competing()`.

Usage

```
parkinsons_simulated
```

Format

A data frame with 1800 rows and 6 variables:

id Subject identifier.

time Visit time in months.

trt Treatment indicator: 0 = control, 1 = active.

updrs UPDRS score (higher values indicate worse symptoms).

time_to_dropout Subject-level time to dropout.

dropout Dropout indicator: 1 = dropped out, 0 = censored.

Source

Simulated with `simulate_joint_trial()`; see `data-raw/create_data.R`.

Examples

```
head(parkinsons_simulated)
with(parkinsons_simulated, table(trt, dropout))
summary(parkinsons_simulated$updrs)
```

pkpd_concentration *PK/PD helpers for biomarker dosing*

Description

One-compartment oral PK concentration and an Emax biomarker response.

Usage

```
pkpd_concentration(time, dose, ka = 0.5, ke = 0.1, V = 10)
```

Arguments

`time`	Time(s) since dose.
`dose`	Administered dose.
`ka`	Absorption rate constant.
`ke`	Elimination rate constant.
`V`	Volume of distribution.

Value

Concentration at `time`.

Examples

```
t <- seq(0, 24, by = 4)
pkpd_concentration(t, dose = 100)
```

plot.pmrm_delayed *Plot a delayed-effect fit*

Description

Plot a delayed-effect fit

Usage

```
## S3 method for class 'pmrm_delayed'
plot(x, type = c("trajectory", "profile_likelihood"), ...)
```

Arguments

`x`	A `pmrm_delayed` object.
`type`	"trajectory" plots fitted population means by arm with the delay marked; "profile_likelihood" plots the log-likelihood vs delay with the chi-squared threshold line.
`...`	Unused.

Value

A ggplot object (invisibly).

Examples

```
sim <- simulate_progression_trial(n_control = 40, n_treatment = 40,
                                  effect_size = -0.8, delay = 6,
                                  times = seq(0, 24, by = 6), seed = 7)
fit <- pmrm_delayed(y ~ time * trt, data = sim,
                    delay_estimation = "profile",
                    delay_grid = c(0, 6), lambda = 0.2)
plot(fit, type = "profile_likelihood")
```

---

pmrm_adaptive_trial *Run a biomarker-informed adaptive trial simulation*

Description

Simulates visits sequentially: measure biomarkers with noise, apply the dosing rule, simulate progression and terminal events.

Usage

```
pmrm_adaptive_trial(
  n_subjects,
  progression_model = list(intercept = 20, slope = 1.5, trt_effect = -0.6, sigma = 2),
  pkpd_model = list(emax = 30, ec50 = 5, b_baseline = 100, noise_sd = 3),
  dosing_rule = biomarker_dosing_rule,
  n_visits = 12,
  visit_interval = 4,
  dropout_rate = 0.15,
  target_range = c(pkpd_model$b_baseline - 10, pkpd_model$b_baseline + 10),
  dose_levels = c(0, 50, 100, 150, 200),
  seed = NULL
)
```

Arguments

`n_subjects`	Number of subjects.
`progression_model`	List describing progression: intercept, slope, treatment effect on slope, residual sd.
`pkpd_model`	List with elements `emax`, `ec50`, `b_baseline`, `noise_sd`.
`dosing_rule`	Function like `biomarker_dosing_rule()`.
`n_visits`	Number of visits.
`visit_interval`	Weeks between visits.

dropout_rate	Annual dropout hazard.
target_range	Target biomarker range for dosing.
dose_levels	Allowed dose levels.
seed	Random seed.

Value

A list with long-format outcomes $long_data and final subject states $subjects.

Examples

```
out <- pmrm_adaptive_trial(n_subjects = 12, n_visits = 4, seed = 1)
head(out$long_data)
# subjects whose dose was adapted at least once
table(out$subjects$dose_idx)
```

---

pmrm_competing *Fit a competing-risks progression model*

Description

Fits cause-specific Cox proportional-hazards models for each terminal event type on top of the shared longitudinal mixed-effects submodel. Association with the longitudinal process enters through empirical-Bayes current value and slope at each subject's event/censoring time.

Usage

```
pmrm_competing(
  long_formula,
  surv_formulas = y ~ 1,
  data,
  time_var = "time",
  subject_var = "id",
  event_time_var = "time_to_dropout",
  event_var = "event",
  causes = c("dropout", "death"),
  association = c("current_value", "slope", "both"),
  ...
)
```

Arguments

long_formula	Longitudinal formula.
surv_formulas	Either a single survival formula or a named list of formulas, one per cause (right-hand side only; Surv response is added).
data	Long-format data frame.

time_var	Name of longitudinal time column.
subject_var	Name of subject id column.
event_time_var	Column holding time-to-event (defaults to the column name used by the shipped data sets).
event_var	Column holding the event code: 0 = censored, 1..K = cause.
causes	Character vector of cause labels in code order.
association	"current_value", "slope", or "both".
...	Additional arguments to pmrm_nlme().

Value

Object of class "pmrm_competing".

Examples

```
fit <- pmrm_competing(updrs ~ time * trt,
                      surv_formulas = ~ trt,
                      data = parkinsons_simulated,
                      event_var = "dropout",
                      causes = "dropout")
names(fit$cause_fits)
summary(fit)$hazard_ratios
```

---

pmrm_delayed *Fit a progression model with delayed treatment onset*

Description

The treatment effect enters the model exclusively through a delayed term, delta * Trt_i * t_ij * d(t_ij), where d() is a fractional onset function: treatment terms already present in formula (such as the trt and time:trt terms of y ~ time * trt) are replaced by this term. Keeping them alongside the delayed term would let an unrestricted treatment effect absorb the delay and make it unidentifiable, in which case the profile likelihood always peaks at zero delay. Because the delayed term is proportional to time, delta is the treatment effect on the progression slope at full onset.

Usage

```
pmrm_delayed(
  formula,
  data,
  delay_estimation = c("fixed", "profile", "bayesian"),
  delay_grid = seq(0, 24, by = 3),
  shape = c("exponential", "step", "linear"),
  lambda = 0.3,
  progression = "linear",
  conf_level = 0.95,
  ...
)
```

Arguments

`formula`	LME-style formula, e.g. `y ~ time * trt`. Terms involving the treatment indicator are replaced by the delayed treatment term; the remaining terms specify the common progression model.
`data`	Long-format data frame.
`delay_estimation`	One of "fixed", "profile", or "bayesian" ("bayesian" currently falls back to profile with a warning).
`delay_grid`	Grid of candidate delays (in the units of `time`).
`shape`	Onset shape passed to `delayed_effect()`.
`lambda`	Onset rate for exponential onset.
`progression`	Progression shape passed to `pmrm_nlme()`.
`conf_level`	Confidence level for the delay confidence interval.
`...`	Additional arguments to `pmrm_nlme()`.

Details

The data must contain a treatment indicator column named `trt` (any case), which is what identifies the treatment terms to replace. Pass other `pmrm_nlme()` options – most importantly `subject_var` and `time_var` – through `....`

When `delay_estimation = "profile"`, the delay is estimated by grid search over `delay_grid`, selecting the value maximizing log-likelihood; an approximate confidence set is derived from the chi-squared threshold (log-lik drop of qchisq(0.95, 1)/2). All candidates use the same fixed-effects structure, so the REML log-likelihoods are comparable across the grid.

Value

Object of class `"pmrm_delayed"` containing the best fit, the estimated delay, the profile likelihood, and the delay CI.

References

Raket LL (2022). Progression models for repeated measures: Estimating novel treatment effects in progressive diseases. Statistics in Medicine. doi:10.1002/sim.9581.

Examples

```
sim <- simulate_progression_trial(n_control = 40, n_treatment = 40,
                                  effect_size = -0.8, delay = 6,
                                  times = seq(0, 24, by = 6),
                                  residual_sd = 1.5, random_sd = 0.2, seed = 7)
fit <- pmrm_delayed(y ~ time * trt, data = sim,
                    delay_estimation = "profile",
                    delay_grid = c(0, 6, 12), lambda = 0.2)
fit$delay_estimate
summary(fit)$delay_ci
```

pmrm_joint *Fit a joint longitudinal-survival model*

Description

Fit a joint longitudinal-survival model

Usage

```
pmrm_joint(
  long_formula,
  surv_formula,
  data,
  event_data = NULL,
  time_var = "time",
  subject_var = "id",
  event_time_var = "time_to_dropout",
  event_var = "dropout",
  association = c("current_value", "slope", "both"),
  surv_model = c("weibull", "exponential", "stan_jm"),
  ...
)
```

Arguments

long_formula	Longitudinal mixed-model formula, e.g. `y ~ time * trt`.
surv_formula	Survival formula, e.g. `Surv(time_to_dropout,dropout) ~ trt`.
data	Long-format data frame with one row per observation; the event data are extracted from `event_data` if supplied, otherwise from per-subject columns in `data`.
event_data	Optional data frame with one row per subject containing the subject id and the variables on the right-hand side of `surv_formula`.
time_var	Name of the longitudinal time column.
subject_var	Name of the subject identifier column.
event_time_var	Name of the column holding time-to-event (defaults to the column name used by the shipped data sets).
event_var	Name of the event indicator column (1 = event).
association	One of "current_value", "slope", "both".
surv_model	Baseline hazard family: `"weibull"` or `"exponential"`. The legacy value `"stan_jm"` is accepted for backward compatibility but currently warns and falls back to `"weibull"`.
...	Additional arguments passed to the fitting engine.

Value

Object of class "pmrm_joint" with elements long_fit, surv_fit, association, and helper summaries.

References

Rizopoulos D (2012). Joint Models for Longitudinal and Time-to-Event Data: With Applications in R. Chapman and Hall/CRC Biostatistics Series. ISBN 978-1-4398-7286-4.

Examples

```
fit <- pmrm_joint(updrs ~ time * trt,
                  Surv(time_to_dropout, dropout) ~ trt,
                  data = parkinsons_simulated,
                  association = "current_value")
class(fit)
fit$assoc_terms
```

---

pmrm_nlme — *Fit a nonlinear progression mixed-effects model*

Description

Fits repeated-measures progression data using one of four common shapes (linear, plateau, emax, exponential). Linear models use nlme::lme; nonlinear shapes use nlme::nlme.

Usage

```
pmrm_nlme(
  formula,
  data,
  progression = c("linear", "plateau", "emax", "exponential"),
  random = NULL,
  treatment_var = "trt",
  time_var = "time",
  subject_var = "id",
  heteroscedastic = FALSE,
  start = NULL,
  ...
)
```

Arguments

formula	Formula such as y ~ time + trt (linear shape).
data	Data frame in long format.
progression	Progression shape: one of "linear", "plateau", "emax", "exponential".
random	Random effects specification; the grouping variable is taken from subject_var.

treatment_var	Name of the treatment indicator column.
time_var	Name of the time column.
subject_var	Name of the subject identifier column.
heteroscedastic	If TRUE allow residual variance to increase with time via a power variance function.
start	Optional named starting values for nonlinear shapes.
...	Additional arguments passed on to the fitting engine.

Value

An object of class "pmrm_nlme".

References

Pinheiro JC, Bates DM (2000). Mixed-Effects Models in S and S-PLUS. Springer Statistics and Computing. ISBN 978-0-387-98957-0.

Examples

```
sim <- simulate_progression_trial(n_control = 20, n_treatment = 20, seed = 1)
fit <- pmrm_nlme(y ~ time * trt, data = sim)
summary(fit)$tTable
```

---

population_trajectory *Population-level trajectory over a time grid*

Description

Population-level trajectory over a time grid

Usage

```
population_trajectory(object, times, trt = 0)
```

Arguments

object	A pmrm_nlme or pmrm_delayed object.
times	Numeric vector of times.
trt	Treatment arm value (default 0).

Value

Data frame with columns time, trt, predicted.

Examples

```
sim <- simulate_progression_trial(n_control = 20, n_treatment = 20, seed = 1)
fit <- pmrm_nlme(y ~ time * trt, data = sim)
population_trajectory(fit, times = seq(0, 24, by = 6), trt = c(0, 1))
```

`power_pmrm` *Simulation-based power for progression trials*

Description

Simulation-based power for progression trials

Usage

```
power_pmrm(
  n_per_arm,
  effect_size = -0.5,
  delay = 0,
  dropout_rate = 0.15,
  measurement_times = seq(0, 24, by = 6),
  n_sims = 200,
  alpha = 0.05,
  seed = NULL,
  ...
)
```

Arguments

`n_per_arm`	Subjects per arm.
`effect_size`	Treatment effect on slope (per year).
`delay`	Delayed onset in months.
`dropout_rate`	Annual dropout rate.
`measurement_times`	Visit schedule in months.
`n_sims`	Number of simulations.
`alpha`	One-sided... actually two-sided type-I error level.
`seed`	Random seed.
`...`	Passed to `simulate_progression_trial()`.

Value

List with power, mean estimate/SE, bias, MSE, and the non-convergence rate.

Examples

```
p <- power_pmrm(n_per_arm = 30, effect_size = -1.5, delay = 0,
                n_sims = 5, measurement_times = seq(0, 24, by = 12),
                seed = 11)
p$power
p$mean_estimate
```

`predict.pmrm_nlme` *Predict from a fitted progression model*

Description

Predict from a fitted progression model

Usage

```
## S3 method for class 'pmrm_nlme'
predict(object, newdata, level = c("population", "individual"), ...)
```

Arguments

`object`	A `pmrm_nlme` object.
`newdata`	New data frame containing the predictor columns.
`level`	Either "population" (fixed effects only) or "individual" (includes empirical-Bayes random effects).
`...`	Unused.

Value

Numeric vector of predictions.

Examples

```
sim <- simulate_progression_trial(n_control = 20, n_treatment = 20, seed = 1)
fit <- pmrm_nlme(y ~ time * trt, data = sim)
nd <- data.frame(time = c(0, 6, 12, 24), trt = c(0, 0, 1, 1))
predict(fit, newdata = nd)                # population level
head(predict(fit, level = "individual"))   # subject-specific
```

print.pmrm_delayed *Print method for delayed-effect fits*

Description

Print method for delayed-effect fits

Usage

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

Arguments

x	A pmrm_delayed object.
...	Passed to print.

Value

The input object x, invisibly (of class c("pmrm_delayed", "lme")). Called for its side effect of printing a concise summary (progression shape, estimated delay and, when available, delay confidence interval) to the console.

Examples

```
sim <- simulate_progression_trial(n_control = 40, n_treatment = 40,
                                  effect_size = -0.8, delay = 6,
                                  times = seq(0, 24, by = 6), seed = 7)
fit <- pmrm_delayed(y ~ time * trt, data = sim,
                    delay_estimation = "profile",
                    delay_grid = c(0, 6), lambda = 0.2)
print(fit)
```

scenario_base *Scenario builders for trial simulation studies*

Description

Named scenario lists consumed by power_pmrm() via do.call.

Usage

```
scenario_base()

scenario_optimistic()

scenario_pessimistic()
```

Value

A list of simulation arguments.

Examples

```
scenario_optimistic()
scenario_base()
scenario_pessimistic()
```

---

simulate_biomarker *Simulate biomarker measurements under the PK/PD model*

Description

Simulate biomarker measurements under the PK/PD model

Usage

```
simulate_biomarker(
  subjects,
  visit,
  visit_interval = 4,
  dose_levels = c(0, 50, 100, 150, 200),
  noise_sd = 3,
  emax = 30,
  ec50 = 5,
  b_baseline = 100
)
```

Arguments

`subjects`	Subject data frame from `initialize_subjects()`.
`visit`	Current visit index.
`visit_interval`	Interval between visits.
`dose_levels`	Dose levels vector.
`noise_sd`	Residual measurement SD.
`emax`, `ec50`, `b_baseline`	Emax-model parameters.

Value

Named numeric vector of observed biomarkers.

Examples

```
subj <- initialize_subjects(6, n_visits = 4, seed = 42)
simulate_biomarker(subj, visit = 2)
```

simulate_joint_trial *Simulate a trial with informative dropout via a joint model mechanism*

Description

Progression drives a Weibull hazard for dropout through the current latent value, so faster progressors leave earlier.

Usage

```
simulate_joint_trial(
  n_control = 50,
  n_treatment = 50,
  effect_size = -0.5,
  delay = 0,
  times = seq(0, 24, by = 3),
  intercept = 20,
  slope = 1,
  random_sd = 0.5,
  residual_sd = 2,
  alpha = 0.05,
  weibull_scale = 100,
  weibull_shape = 1.4,
  seed = NULL
)
```

Arguments

`n_control`, `n_treatment`
: Subjects per arm.

`effect_size`
: Treatment effect on the slope per unit of time (negative = slower progression).

`delay`
: Delayed onset (same units as `times`).

`times`
: Vector of measurement times.

`intercept`
: Population intercept at time 0.

`slope`
: Control-arm slope per unit time.

`random_sd`
: SD of subject random slope.

`residual_sd`
: Residual SD.

`alpha`
: Association of current value with the log-hazard.

`weibull_scale`, `weibull_shape`
: Weibull parameters for baseline hazard.

`seed`
: Random seed.

Value

A long-format data frame (one row per subject per visit time) with columns `id` (subject identifier), `time` (visit time), `trt` (treatment indicator, 0 = control, 1 = active), `y` (longitudinal outcome, `NA` after dropout), `time_to_dropout` (subject-level simulated event/censoring time) and `dropout` (subject-level event indicator, 1 = dropout observed, 0 = censored). Larger latent progression values produce larger hazards through `alpha`, so faster progressors tend to drop out earlier.

Examples

```
sim <- simulate_joint_trial(n_control = 20, n_treatment = 20,
                            times = seq(0, 24, by = 6), seed = 1)
head(sim)
# share of subjects with an observed dropout event
mean(tapply(sim$dropout, sim$id, function(z) z[1]))
```

`simulate_progression_trial` *Simulate a progression trial*

Description

Simulate a progression trial

Usage

```
simulate_progression_trial(
  n_control = 50,
  n_treatment = 50,
  effect_size = -0.5,
  delay = 0,
  dropout_rate = 0.15,
  times = seq(0, 24, by = 6),
  intercept = 20,
  slope = 1,
  random_sd = 0.3,
  residual_sd = 2,
  seed = NULL
)
```

Arguments

`n_control`, `n_treatment`	Subjects per arm.
`effect_size`	Treatment effect on the slope per unit of time (negative = slower progression).
`delay`	Delayed onset (same units as `times`).
`dropout_rate`	Annual dropout probability.

`times`	Vector of measurement times.
`intercept`	Population intercept at time 0.
`slope`	Control-arm slope per unit time.
`random_sd`	SD of subject random slope.
`residual_sd`	Residual SD.
`seed`	Random seed.

Value

Long-format data frame with columns `id`, `time`, `trt`, `y`.

Examples

```
sim <- simulate_progression_trial(n_control = 20, n_treatment = 20,
                                  effect_size = -0.5, delay = 6,
                                  times = seq(0, 24, by = 6), seed = 1)
head(sim)
aggregate(y ~ trt, data = sim, FUN = mean, na.rm = TRUE)
```

`subject_summaries` *Extract subject-level summary (EB intercept and slope)*

Description

Extract subject-level summary (EB intercept and slope)

Usage

```
subject_summaries(fit)
```

Arguments

`fit`	A `pmrm_nlme` object.

Value

Data frame with `id`, `b0` and `b1`.

Examples

```
sim <- simulate_progression_trial(n_control = 20, n_treatment = 20, seed = 1)
fit <- pmrm_nlme(y ~ time * trt, data = sim)
head(subject_summaries(fit))
```

summary.pmrm_competing *Summarize a competing-risks fit*

Description

Summarize a competing-risks fit

Usage

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

Arguments

object	A pmrm_competing object.
...	Unused.

Value

A list with the longitudinal fixed effects, the cause labels, the cause-specific hazard ratios, and per-cause concordance.

Examples

```
fit <- pmrm_competing(updrs ~ time * trt,
                      surv_formulas = ~ trt,
                      data = parkinsons_simulated,
                      event_var = "dropout",
                      causes = "dropout")
summary(fit)$fixed_effects
summary(fit)$concordance
```

summary.pmrm_delayed *Summarize a delayed-effect fit*

Description

Summarize a delayed-effect fit

Usage

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

Arguments

`object`	A `pmrm_delayed` object.
`...`	Unused.

Value

List with fixed effects, delay estimate/CI, and information criteria.

Examples

```
sim <- simulate_progression_trial(n_control = 40, n_treatment = 40,
                                  effect_size = -0.8, delay = 6,
                                  times = seq(0, 24, by = 6), seed = 7)
fit <- pmrm_delayed(y ~ time * trt, data = sim,
                    delay_estimation = "profile",
                    delay_grid = c(0, 6), lambda = 0.2)
s <- summary(fit)
s$delay_estimate
s$delay_ci
```

`summary.pmrm_joint` *Summarize a joint model fit*

Description

Summarize a joint model fit

Usage

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

Arguments

`object`	A `pmrm_joint` object.
`...`	Unused.

Value

A list with the progression fixed/random effects, the survival coefficients, the association structure and its coefficients, and the longitudinal/survival AIC values.

Examples

```
fit <- pmrm_joint(updrs ~ time * trt,
                  Surv(time_to_dropout, dropout) ~ trt,
                  data = parkinsons_simulated,
                  association = "both")
s <- summary(fit)
s$survival
s$association_coefficients
```

`to_event_data` *Reshape long trial data to per-subject event data*

Description

Reshape long trial data to per-subject event data

Usage

```
to_event_data(
  long_data,
  id_var = "id",
  time_to_event_var = "time_to_dropout",
  event_var = "dropout"
)
```

Arguments

`long_data` Long-format data frame.

`id_var`, `time_to_event_var`, `event_var` Column names.

Value

One-row-per-subject data frame.

Examples

```
sim <- simulate_joint_trial(n_control = 20, n_treatment = 20, seed = 1)
ev <- to_event_data(sim, id_var = "id",
                    time_to_event_var = "time_to_dropout",
                    event_var = "dropout")
head(ev)
```

vpc *Visual predictive check (VPC)*

Description

Visual predictive check (VPC)

Usage

```
vpc(fit, n_sims = 50, probs = c(0.05, 0.5, 0.95), seed = NULL)
```

Arguments

fit	Fitted pmrm_nlme object.
n_sims	Number of simulated datasets for the VPC.
probs	Quantiles to report.
seed	Random seed.

Value

Data frame of observed vs simulated quantiles by time bin.

Examples

```
sim <- simulate_progression_trial(n_control = 20, n_treatment = 20,
                                  times = seq(0, 24, by = 6), seed = 2)
fit <- pmrm_nlme(y ~ time * trt, data = sim)
vpc(fit, n_sims = 5, seed = 1)
```

Index

∗ **datasets**
    adni_simulated, 3
    parkinsons_simulated, 11

adni_simulated, 3

biomarker_dosing_rule, 4

check_convergence, 4

default_priors, 5
delayed_effect, 6

emax_biomarker, 7

find_sample_size, 7
fit_pmrm, 8

goodness_of_fit, 9

initialize_subjects, 9

parametric_bootstrap, 10
parkinsons_simulated, 11
pkpd_concentration, 12
plot.pmrm_delayed, 12
pmrm_adaptive_trial, 13
pmrm_competing, 14
pmrm_delayed, 15
pmrm_joint, 17
pmrm_nlme, 18
population_trajectory, 19
power_pmrm, 20
predict.pmrm_nlme, 21
print.pmrm_delayed, 22

scenario_base, 22
scenario_optimistic *(*scenario_base*)*, 22
scenario_pessimistic *(*scenario_base*)*, 22
simulate_biomarker, 23
simulate_joint_trial, 24
simulate_progression_trial, 25
subject_summaries, 26
summary.pmrm_competing, 27
summary.pmrm_delayed, 27
summary.pmrm_joint, 28

to_event_data, 29

vpc, 30