Package: inferMM (via r-universe)

June 5, 2026

Type Package

Title Variance-Aware Michaelis-Menten Estimation and Inference

Version 0.0.3

Description Variance-aware Michaelis-Menten estimation, model screening, grouped enzyme-kinetic analyses, and clustered repeated-measurement workflows. The package implements profile-score estimators under working variance functions, together with a lightweight cluster-aware working-covariance extension, Wald and bootstrap confidence intervals, prediction utilities, and simulation helpers. Related methodology is discussed by Kim and Ma (2012) <doi:10.1007/s10463-011-0332-y>, Kim (2023) <doi:10.1002/sta4.606>, and Ma and Genton (2010) <doi:10.1111/j.1467-9868.2010.00741.x>.

License GPL-3

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

Depends R (>= 4.1.0)

Imports graphics, grDevices, stats

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Mijeong Kim [aut, cre], Minkyoung Cha [aut], Ah Young Jeong [aut]

Maintainer Mijeong Kim <m.kim@ewha.ac.kr>

Repository https://mijeong-kim.r-universe.dev

Date/Publication 2026-06-04 23:24:33 UTC

RemoteUrl https://github.com/mijeong-kim/infermm

RemoteRef HEAD

RemoteSha c5b9905f05490b96cbcf019d1b4f2ef64b749571

Contents

`inferMM-package` *inferMM: Variance-Aware Michaelis-Menten Estimation and Inference*

Description

The **inferMM** package provides variance-aware tools for Michaelis-Menten estimation, clustered repeated-measurement analysis, model screening, grouped curve analysis, simulation, and prediction.

Details

The package focuses on enzyme-kinetic settings where measurement variability may increase with substrate concentration, making standard homoscedastic nonlinear least squares inadequate for interval estimation and uncertainty quantification.

See Also

`fit_mm`, `cluster_mm`, `report_mm`, `screen_mm`, `group_mm`, `simulate_mm_data`

`alves_demo` *Subset of the Alves Soil Enzyme Kinetics Data*

Description

A filtered subset of the Alves et al. soil exoenzyme kinetics data, restricted to depth `00-10` and temperature `10` for fast reproducible package examples.

Format

A data frame with 247 rows and 6 variables:

`core` Soil core identifier.

`depth` Depth interval.

`enzyme` Enzyme name.

`temperature` Incubation temperature.

`substrate_conc` Substrate concentration.

`activity` Observed enzyme activity.

Source

Prepared from the Alves study files in the project workspace.

Examples

```
head(alves_demo)
table(alves_demo$enzyme)
```

---

cluster_mm *Fit a Cluster-Aware Michaelis-Menten Model*

Description

Estimate shared Michaelis-Menten mean parameters from repeated or clustered assay data using a Ma-Genton-style working covariance with a random effect on $V_{\max}$ and a low-dimensional residual working variance function.

Usage

```
cluster_mm(data, s, v, cluster,
  variance = c("log", "sqrt", "cuberoot", "constant", "power", "auto"),
  power = NULL, power_selection = c("quasi_aic", "quasi_bic"),
  power_grid = default_power_grid(), interval_level = 0.95,
  invalid_response_values = c(-9999),
  invalid_substrate_values = numeric(0), allow_zero_substrate = TRUE,
  sort_x = TRUE)

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

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

## S3 method for class 'cluster_mm'
confint(object, parm = c("Vmax", "Km"),
```

```
  level = object$interval_level, method = c("wald", "bootstrap"),
  B = 399L, bootstrap_type = c("pairs"),
  bootstrap_ci = c("studentized", "basic", "percentile"),
  seed = NULL, ...)

## S3 method for class 'cluster_mm'
print(x,
  digits = max(3L, getOption("digits") - 2L), ...)

## S3 method for class 'cluster_mm'
summary(object, level = object$interval_level,
  method = c("wald", "bootstrap"), B = 399L,
  bootstrap_type = c("pairs"),
  bootstrap_ci = c("studentized", "basic", "percentile"),
  seed = NULL, ...)

## S3 method for class 'summary.cluster_mm'
print(x,
  digits = max(3L, getOption("digits") - 2L), ...)

## S3 method for class 'cluster_mm'
predict(object, newdata = NULL, se.fit = FALSE,
  interval = c("none", "confidence", "prediction"),
  level = object$interval_level, ...)

## S3 method for class 'cluster_mm'
plot(x, interval = TRUE,
  interval_type = c("prediction", "confidence"),
  level = x$interval_level, n_points = 200,
  xlab = "Substrate concentration", ylab = "Reaction velocity",
  main = NULL, pch = 19, col.points = "#1f78b4",
  col.line = "#222222",
  col.band = grDevices::adjustcolor("#9ecae1", alpha.f = 0.45), ...)
```

Arguments

`data`	A data frame containing clustered Michaelis-Menten observations.
`s`	Name of the substrate concentration column.
`v`	Name of the response column.
`cluster`	Character vector naming one or more clustering columns.
`variance`	Character string naming a built-in residual working variance model. Use `"power"` with a fixed exponent or `"auto"` to compare `log(S + 1)` against a grid of power functions.
`power`	Optional exponent used when `variance = "power"`.
`power_selection`	Criterion used when `variance = "auto"`: `"quasi_aic"` or `"quasi_bic"`.
`power_grid`	Candidate exponents considered when `variance = "auto"`.

interval_level	Confidence level used for Wald intervals.
invalid_response_values	Values to remove before fitting.
invalid_substrate_values	Values to remove before fitting.
allow_zero_substrate	Logical; if FALSE, substrate concentrations must be strictly positive.
sort_x	Logical; sort observations within each cluster by substrate concentration.
object	A fitted "cluster_mm" object.
parm	Parameter names for confint().
level	Confidence level for intervals.
method	Interval construction method. The current implementation supports "wald" or a cluster-pairs "bootstrap" sensitivity analysis.
B	Number of bootstrap resamples when method = "bootstrap".
bootstrap_type	Bootstrap scheme for clustered fits. The current implementation supports "pairs" only.
bootstrap_ci	Bootstrap confidence-interval calibration: "studentized", "basic", or "percentile".
seed	Deprecated and ignored. Call stats::set.seed() before bootstrap-based summaries or intervals if reproducibility is needed.
x	A fitted "cluster_mm" object for the print and plot methods.
digits	Number of printed digits.
newdata	Optional numeric vector of new substrate concentrations.
se.fit	Logical; if TRUE, return standard errors for mean predictions.
interval	Prediction type: none, confidence, or prediction.
interval_type	For the plot method, draw either a prediction or confidence band when interval = TRUE.
n_points	Number of grid points used for plotting the fitted curve.
xlab, ylab, main, pch, col.points, col.line, col.band	Graphical parameters for the plot method.
...	Additional arguments passed to S3 methods.

Details

The current implementation treats repeated measurements through a working covariance of the form $Z_i B Z_i^\top + \gamma A_i$, where A_i is induced by the chosen residual variance function and the random-effect component is specialized to a first-order perturbation of $V_{\max}$. This provides a lightweight cluster-aware extension of the single-curve profile-score estimator without requiring a fully parametric nonlinear mixed effects specification.

Numerically, the routine initializes the clustered fit from the corresponding pooled single-curve estimator and then alternates a one-dimensional update for K_m with plug-in updates for $V_{\max}$ and moment-based updates for the variance components. The pooled start is obtained from the same single-curve profile-score machinery used by fit_mm(), including the constant-variance special

case. Fits that reach the maximum number of outer iterations without meeting the convergence tolerance remain available but carry an explicit convergence warning in the object summary.

Because small numbers of clusters can make first-order interval calibration fragile, printed summaries suppress interval output by default when fewer than four clusters are available or when there are fewer than six distinct substrate concentrations. In that regime, `confint()` remains available as an explicit sensitivity analysis and issues a warning.

Value

The main fitting function returns an object of class `"cluster_mm"`. `predict()` returns either a numeric vector or a data frame containing pointwise confidence or prediction intervals together with working and marginal variance estimates.

Examples

```
cluster_fit <- cluster_mm(
  data = subset(alves_demo, enzyme == "BG"),
  s = "substrate_conc",
  v = "activity",
  cluster = "core",
  variance = "sqrt"
)

coef(cluster_fit)
summary(cluster_fit)
confint(cluster_fit)
head(
  predict(
    cluster_fit,
    newdata = seq(0, 700, length.out = 6),
    interval = "confidence"
  )
)


plot(cluster_fit, interval_type = "confidence")
report_mm(cluster_fit, interval_type = "confidence")
```

`fit_mm` *Fit a Variance-Aware Michaelis-Menten Model*

Description

Estimate Michaelis-Menten parameters under a constant or concentration-dependent working variance specification.

Usage

```
fit_mm(x, y,
  variance = c("log", "sqrt", "cuberoot", "constant", "power", "auto"),
  power = NULL, power_selection = c("quasi_aic", "quasi_bic"),
  power_grid = default_power_grid(), interval_level = 0.95,
  km_bounds = NULL, allow_zero_substrate = TRUE)

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

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

## S3 method for class 'fit_mm'
confint(object, parm = c("Vmax", "Km"),
  level = object$interval_level, method = c("wald", "bootstrap"),
  B = 399L, bootstrap_type = c("wild", "pairs"),
  bootstrap_ci = c("studentized", "basic", "percentile"),
  wild_weights = c("mammen", "rademacher"), seed = NULL, ...)

## S3 method for class 'fit_mm'
print(x, digits = max(3L, getOption("digits") - 2L), ...)

## S3 method for class 'fit_mm'
summary(object, level = object$interval_level,
  method = c("wald", "bootstrap"), B = 399L,
  bootstrap_type = c("wild", "pairs"),
  bootstrap_ci = c("studentized", "basic", "percentile"),
  wild_weights = c("mammen", "rademacher"), seed = NULL, ...)

## S3 method for class 'summary.fit_mm'
print(x,
  digits = max(3L, getOption("digits") - 2L), ...)

## S3 method for class 'fit_mm'
predict(object, newdata = NULL, se.fit = FALSE,
  interval = c("none", "confidence", "prediction"),
  level = object$interval_level, ...)

## S3 method for class 'fit_mm'
plot(x, interval = TRUE,
  interval_type = c("prediction", "confidence"),
  level = x$interval_level, n_points = 200,
  xlab = "Substrate concentration", ylab = "Reaction velocity",
  main = NULL, pch = 19, col.points = "#1f78b4",
  col.line = "#222222",
  col.band = grDevices::adjustcolor("#9ecae1", alpha.f = 0.45), ...)
```

Arguments

x	Numeric vector of substrate concentrations. For the plot method, a fitted object of class "fit_mm".
y	Numeric vector of observed reaction velocities.
variance	Character string naming a built-in working variance model. Use "power" with a fixed exponent or "auto" to compare log(S + 1) against a grid of power functions.
power	Optional exponent used when variance = "power".
power_selection	Criterion used when variance = "auto": "quasi_aic" or "quasi_bic".
power_grid	Candidate exponents considered when variance = "auto".
interval_level	Confidence level used for interval estimation.
km_bounds	Numeric vector of length two giving the search bounds for K_m.
allow_zero_substrate	Logical; if FALSE, substrate concentrations must be strictly positive.
object	A fitted "fit_mm" object.
parm	Parameter names for confint().
level	Confidence level for intervals.
method	Interval construction method: "wald" or "bootstrap".
B	Number of bootstrap resamples when method = "bootstrap".
bootstrap_type	Bootstrap scheme: "wild" or "pairs".
bootstrap_ci	Bootstrap interval calibration used when method = "bootstrap": "studentized", "basic", or "percentile".
wild_weights	Wild-bootstrap multipliers used when bootstrap_type = "wild": "mammen" or "rademacher".
seed	Deprecated and ignored. Call stats::set.seed() before bootstrap-based summaries or intervals if reproducibility is needed.
digits	Number of printed digits.
newdata	Optional numeric vector of new substrate concentrations.
se.fit	Logical; if TRUE, return standard errors for mean predictions.
interval	Prediction type: none, confidence, or prediction.
interval_type	For the plot method, draw either a prediction or confidence band when interval = TRUE.
n_points	Number of grid points used for plotting the fitted curve.
xlab, ylab, main, pch, col.points, col.line, col.band	Graphical parameters for the plot method.
...	Additional arguments passed to S3 methods.

Details

Built-in working variance models are `"constant"`, `"log"`, `"sqrt"`, `"cuberoot"`, `"power"`, and `"auto"`. All built-in single-curve fits use the Michaelis-Menten profile-score estimator from the companion methodology work, with the constant-variance case treated as the special case $h(S) \equiv 1$. Confidence intervals can be computed either from the default asymptotic Wald approximation or from a bootstrap refitting scheme. The bootstrap option supports percentile, basic, and studentized calibration, together with wild or pairs resampling. The default printed summaries suppress interval output when the fitted curve has fewer than 20 observations or fewer than 6 distinct substrate concentrations; in that setting, `confint()` remains available as an explicit sensitivity analysis tool and issues a warning. The automatic option compares `log(S + 1)` with a user-supplied grid of S^p working variance functions and keeps the best fit according to quasi-AIC or quasi-BIC.

Value

The main fitting function returns an object of class `"fit_mm"`. `predict()` returns either a numeric vector or a data frame containing prediction intervals.

Examples

```
one_curve <- subset(sdl_demo, enzyme == "1111")

fit <- fit_mm(
  x = one_curve$s_uM,
  y = one_curve$v_uM_per_min,
  variance = "sqrt"
)

fit_power <- fit_mm(
  x = one_curve$s_uM,
  y = one_curve$v_uM_per_min,
  variance = "power",
  power = 0.4
)

fit_auto <- fit_mm(
  x = one_curve$s_uM,
  y = one_curve$v_uM_per_min,
  variance = "auto",
  power_selection = "quasi_aic"
)

coef(fit)
confint(fit)
set.seed(1)
confint(
  fit,
  method = "bootstrap",
  B = 99,
  bootstrap_ci = "studentized",
  wild_weights = "mammen"
)
```

```
head(predict(fit, newdata = seq(0, 80, length.out = 6), interval = "prediction"))


plot(fit)
report_mm(fit, interval_type = "confidence")
```

`group_mm` *Fit Multiple Michaelis-Menten Curves by Group*

Description

Clean a data frame, fit one or more working variance models within each group, and compare the competing fits using quasi-AIC as the primary criterion, with quasi-BIC and RMSE reported alongside it.

Usage

```
group_mm(data, s, v, groups = NULL,
  variance_models = default_variance_models(), power_values = NULL,
  include_auto = FALSE, power_selection = c("quasi_aic", "quasi_bic"),
  power_grid = default_power_grid(), interval_level = 0.95,
  invalid_response_values = c(-9999),
  invalid_substrate_values = numeric(0), allow_zero_substrate = TRUE,
  sort_x = TRUE, quiet = FALSE)

## S3 method for class 'group_mm'
print(x,
  digits = max(3L, getOption("digits") - 2L), ...)

## S3 method for class 'group_mm'
plot(x, which_groups = NULL, model = NULL,
  interval = TRUE, interval_type = c("prediction", "confidence"),
  level = NULL, ncol = NULL, main = NULL,
  xlab = "Substrate concentration", ylab = "Reaction velocity", ...)
```

Arguments

`data`	A data frame containing Michaelis-Menten observations.
`s`	Name of the substrate concentration column.
`v`	Name of the response column.
`groups`	Optional character vector of grouping columns.
`variance_models`	Character vector of built-in variance models. The special value `"auto"` adds a candidate that compares `log(S + 1)` with a grid of power functions and keeps the better fit.

power_values	Optional numeric vector of fixed exponents for additional S^power candidates.
include_auto	Logical; if TRUE, add one automatic candidate that selects between log(S + 1) and the supplied power_grid.
power_selection	Criterion used by the automatic candidate: "quasi_aic" or "quasi_bic".
power_grid	Candidate exponents used by the automatic candidate.
interval_level	Confidence level forwarded to fit_mm.
invalid_response_values	Values to remove before fitting.
invalid_substrate_values	Values to remove before fitting.
allow_zero_substrate	Logical; if FALSE, substrate concentrations must be strictly positive.
sort_x	Logical; sort observations within each group by concentration.
quiet	Logical; if TRUE, suppress progress messages.
x	A grouped fit object for the print and plot methods.
which_groups	Optional subset of groups to plot, matched against group_id or group_label.
model	Optional model name to use for every group. When NULL, the plotting method uses the best model recorded for each group.
interval	Logical; if TRUE, add interval bands to each panel.
interval_type	Either "prediction" or "confidence".
level	Optional interval level. Defaults to the level stored in each fit.
ncol	Optional number of panel columns in the grouped display.
main	Optional panel title or titles.
xlab	Label for the x-axis.
ylab	Label for the y-axis.
digits	Number of printed digits.
...	Additional arguments passed to S3 methods.

Value

An object of class "group_mm" containing cleaned data, fit summaries, augmented residual output, grouped rankings, and fitted objects.

Examples

```
grouped <- group_mm(
  data = sdl_demo,
  s = "s_uM",
  v = "v_uM_per_min",
  groups = "enzyme",
  variance_models = c("constant", "sqrt"),
  power_values = 0.4,
  include_auto = TRUE,
```

```
  quiet = TRUE
)

grouped
head(
  grouped$comparison$best_by_group[
    ,
    c("group_label", "model", "selected_model", "quasi_aic", "quasi_bic", "rmse")
  ]
)

plot(grouped, interval_type = "confidence")
```

`report_mm` *Print and Plot a Michaelis-Menten Fit*

Description

Convenience wrapper that prints a fitted-model summary and draws the fitted curve with a confidence or prediction band.

Usage

```
report_mm(object, level = object$interval_level,
  method = c("wald", "bootstrap"), B = 399L,
  bootstrap_type = c("wild", "pairs"),
  bootstrap_ci = c("studentized", "basic", "percentile"),
  wild_weights = c("mammen", "rademacher"), seed = NULL,
  interval_type = c("confidence", "prediction", "none"),
  digits = max(3L, getOption("digits") - 2L), ...)
```

Arguments

`object`	A fitted object returned by `fit_mm` or `cluster_mm`.
`level`	Confidence level used in the printed summary and plot.
`method`	Interval method used in the printed summary: `"wald"` or `"bootstrap"`.
`B`	Number of bootstrap resamples when `method = "bootstrap"`.
`bootstrap_type`	Bootstrap scheme used when `method = "bootstrap"`: `"wild"` or `"pairs"`.
`bootstrap_ci`	Bootstrap interval calibration used when `method = "bootstrap"`: `"studentized"`, `"basic"`, or `"percentile"`.
`wild_weights`	Wild-bootstrap multipliers used when `bootstrap_type = "wild"`: `"mammen"` or `"rademacher"`.
`seed`	Deprecated and ignored. Call `stats::set.seed()` before bootstrap-based summaries if reproducibility is needed.
`interval_type`	Character string indicating whether the plotted band should be `"confidence"`, `"prediction"`, or `"none"`.
`digits`	Number of printed digits.
`...`	Additional graphical arguments passed to `plot`.

Value

The original fitted object, returned invisibly. When `method = "bootstrap"` and the object inherits from `"fit_mm"`, the printed summary uses bootstrap standard errors and confidence intervals; the plotted band remains the usual pointwise confidence or prediction band from `plot.fit_mm()`. For very small curves (fewer than 20 observations or fewer than 6 distinct substrate concentrations), the printed summary reports point estimates only by default and leaves interval construction to an explicit `confint()` call. For sparse clustered fits, the printed summary similarly suppresses interval output by default and treats `confint()` as an explicit sensitivity analysis; bootstrap summaries are not currently available for `"cluster_mm"` objects.

Examples

```
one_curve <- subset(sdl_demo, enzyme == "1111")
fit <- fit_mm(
  x = one_curve$s_uM,
  y = one_curve$v_uM_per_min,
  variance = "sqrt"
)


report_mm(fit, interval_type = "confidence")
set.seed(1)
report_mm(
  fit,
  method = "bootstrap",
  B = 99,
  bootstrap_ci = "studentized",
  wild_weights = "mammen",
  interval_type = "confidence"
)

cluster_fit <- cluster_mm(
  data = subset(alves_demo, enzyme == "BG"),
  s = "substrate_conc",
  v = "activity",
  cluster = "core",
  variance = "sqrt"
)
report_mm(cluster_fit, interval_type = "confidence")
```

`screen_mm` *Screen Working Variance Models*

Description

Fit several working variance models to the same Michaelis-Menten curve and order them primarily by quasi-AIC, while also reporting quasi-BIC, RMSE, and weighted residual sums of squares for reference.

Usage

```
screen_mm(x, y, variance_models = default_variance_models(),
  power_values = NULL, include_auto = FALSE,
  power_selection = c("quasi_aic", "quasi_bic"),
  power_grid = default_power_grid(), interval_level = 0.95,
  allow_zero_substrate = TRUE, km_bounds = NULL, quiet = FALSE)

## S3 method for class 'screen_mm'
print(x, digits = max(3L, getOption("digits") - 2L), ...)
```

Arguments

x	Numeric vector of substrate concentrations, or a screening object for the print method.
y	Numeric vector of observed reaction velocities.
variance_models	Character vector of built-in variance models to compare. The special value "auto" adds a candidate that compares log(S + 1) with a grid of power functions and keeps the better fit.
power_values	Optional numeric vector of fixed exponents for additional S^power candidates.
include_auto	Logical; if TRUE, add one automatic candidate that selects between log(S + 1) and the supplied power_grid.
power_selection	Criterion used by the automatic candidate: "quasi_aic" or "quasi_bic".
power_grid	Candidate exponents used by the automatic candidate.
interval_level	Confidence level forwarded to fit_mm.
allow_zero_substrate	Logical; if FALSE, substrate concentrations must be strictly positive.
km_bounds	Optional search bounds for K_m.
quiet	Logical; if TRUE, suppress progress messages.
digits	Number of printed digits.
...	Additional arguments passed to S3 methods.

Value

An object of class "screen_mm" with a fitted-model table and the successful fits.

Examples

```
one_curve <- subset(sdl_demo, enzyme == "1111")
screen <- screen_mm(
  x = one_curve$s_uM,
  y = one_curve$v_uM_per_min,
  power_values = c(0.4, 0.6),
  include_auto = TRUE,
  quiet = TRUE
```

```
)

screen$table[, c("model", "selected_model", "quasi_aic", "quasi_bic", "rmse")]
```

`sdl_demo` *Self-Driving Laboratory Michaelis-Menten Demo Data*

Description

A lightweight self-driving-laboratory enzyme-kinetic dataset used in the package examples and vignette.

Format

A data frame with 168 rows and 5 variables:

`enzyme` Enzyme identifier.

`s_uM` Substrate concentration in micromolar units.

`replicate` Replicate index.

`v_uM_per_s` Velocity in micromolar units per second.

`v_uM_per_min` Velocity in micromolar units per minute.

Source

Prepared from the SDL study files in the project workspace.

Examples

```
head(sdl_demo)
table(sdl_demo$enzyme)
```

`simulate_mm_data` *Simulate Michaelis-Menten Data*

Description

Generate a synthetic Michaelis-Menten dataset with bounded heteroscedastic variance and either Gaussian or moderately skewed errors.

Usage

```
simulate_mm_data(n = 100, x = seq(1, 100, length.out = n),
  vmax = 100, km = 20, variance_shape = c("mm", "exp", "hill"),
  error = c("normal", "skewed"), s0 = 1, t2 = 9, c = 20,
  a = 0.05, alpha = 2, skew_shape = 4, seed = NULL)
```

Arguments

n	Number of observations when x is not supplied.
x	Optional substrate concentration grid.
vmax	True $V_{\max}$ value.
km	True K_m value.
variance_shape	Character string naming the true variance pattern.
error	Character string naming the error distribution.
s0, t2, c, a, alpha	Variance-shape parameters.
skew_shape	Shape parameter for the skewed gamma-based error.
seed	Deprecated and ignored. Call stats::set.seed() before simulate_mm_data() if reproducibility is needed.

Value

A data frame containing the simulated observations.

Examples

```
set.seed(1)
sim_dat <- simulate_mm_data(variance_shape = "hill", error = "skewed")
head(sim_dat)
```

---

variance_function *Create Working Variance Functions*

Description

Construct a working variance function for variance-aware Michaelis-Menten fitting.

Usage

```
variance_function(model = c("constant", "log", "sqrt", "cuberoot", "power"),
  power = NULL)
```

Arguments

model	Character string naming the working variance specification.
power	Optional exponent used when model = "power".

Value

A function of substrate concentration x.

Examples

```
h_log <- variance_function("log")
h_sqrt <- variance_function("sqrt")
h_log(c(0, 5, 10))
h_sqrt(c(0, 5, 10))
```

Index