Package: predmicror (via r-universe)

June 22, 2026

Title Fitting Predictive Microbiology Models

Version 1.3.2

Date 2026-06-05

Description Provides predictive microbiology model functions and convenience wrappers for fitting primary growth, microbial inactivation, dynamic, omnibus, and cardinal parameter models to experimental data using nonlinear least squares and related mixed-effects or time-varying workflows. Includes helper functions for extracting fitted values, calculating model diagnostics, and comparing fitted models. Implemented model families include those described by: Zwietering et al. (1990) <doi:10.1128/AEM.56.6.1875-1881.1990>, Baranyi and Roberts (1994) <doi:10.1016/0168-1605(94)90157-0>, Baranyi and Roberts (1995) <doi:10.1016/0168-1605(94)00121-L>, Buchanan et al. (1997) <doi:10.1006/fmic.1997.0125>, Richards (1959) <doi:10.1093/jxb/10.2.290>, Fang et al. (2012) <doi:10.1111/j.1750-3841.2012.02873.x>, Fang et al. (2013) <doi:10.1016/j.fm.2012.12.005>, Huang (2008) <doi:10.1111/j.1750-3841.2008.00785.x>, Huang (2009) <doi:10.1016/j.jfoodeng.2008.07.011>, Huang (2013) <doi:10.1016/j.foodcont.2012.11.019>, Geeraerd et al. (2005) <doi:10.1016/j.ijfoodmicro.2004.11.038>, van Boekel (2002) <doi:10.1016/S0168-1605(01)00742-5>, Peleg (1999) <doi:10.1016/S0963-9969(99)00081-2>, Mafart et al. (2002) <doi:10.1016/S0168-1605(01)00624-9>, Albert and Mafart (2005) <doi:10.1016/j.ijfoodmicro.2004.10.016>, Rosso et al. (1993) <doi:10.1006/jtbi.1993.1099>, Rosso et al. (1995) <doi:10.1128/AEM.61.2.610-616.1995>, and Rosso et al. (1996) <doi:10.4315/0362-028X-59.9.944>.

Depends R (>= 3.4.0)

Imports graphics, gslnls, nlme, Rdpack, stats, utils

RdMacros Rdpack

LazyLoad yes

LazyData true

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

NeedsCompilation no

Suggests knitr, rmarkdown, htmltools, commonmark, shinyAce, testthat (>= 3.0.0), ggplot2, deSolve, FME, minpack.lm, reshape2, readxl, shiny

VignetteBuilder knitr

URL `https://fsqanalytics.github.io/predmicror/`, `https://github.com/fsqanalytics/predmicror`

BugReports `https://github.com/fsqanalytics/predmicror/issues`

Config/testthat/edition 3

Config/pak/sysreqs libgsl0-dev

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

Date/Publication 2026-06-14 00:46:54 UTC

RemoteUrl https://github.com/fsqanalytics/predmicror

RemoteRef HEAD

RemoteSha 8b118229158d2f68c602cda81cfb5e7794275e70

Contents

`aw` *Data of aw*

Description

A dataset containing water activity and growth rate data

Usage

```
data(aw)
```

Format

A data frame with 9 rows and 3 variables:

aw Water activity

sqrtGR Square root of the growth rate

GR Growth rate

`BaranyiFM` *Baranyi and Roberts full growth model*

Description

`BaranyiFM` function to fit the Baranyi & Roberts full growth model to a complete microbial growth curve. Returns the model parameters estimated according to data collected in microbial growth experiments.

Usage

```
BaranyiFM(t, Y0, Ymax, MUmax, lag)
```

Arguments

`t`	is a numeric vector indicating the time of the experiment
`Y0`	is the natural logarithm of the initial microbial concentration (`ln(N0)`) at time=0
`Ymax`	is the natural logarithm of the maximum concentration (`ln(Nmax)`) reached by the microorganism
`MUmax`	is the maximum specific growth rate given in time units
`lag`	is the duration of the lag phase in time units

Details

The model's inputs are:

`t`: time, assuming time zero as the beginning of the experiment.

`Y(t)`: the natural logarithm of the microbial concentration (`ln(N(t))`) measured at time t.

Users should make sure that the microbial concentration input is entered in natural logarithm, `Y(t) = ln(N(t))`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez `<vcadavez@ipb.pt>` and Ursula Gonzales-Barron `<ubarron@ipb.pt>`

References

Baranyi J, Roberts TA (1994). "A dynamic approach to predicting bacterial growth in food." *International Journal of Food Microbiology*, **23**, 277-294.

Examples

```
library(gslnls)
data(growthfull)
initial_values <- list(Y0 = -0.1, Ymax = 22, MUmax = 1.7, lag = 5)
fit <- gsl_nls(lnN ~ BaranyiFM(Time, Y0, Ymax, MUmax, lag),
  data = growthfull,
  start = initial_values
)
summary(fit)
```

`BaranyiRM` *Baranyi and Roberts reduced growth model*

Description

`BaranyiRM` function to fit the Baranyi and Roberts growth model to a reduced microbial growth curve. Returns the model parameters estimated according to data collected in microbial growth experiments.

Usage

```
BaranyiRM(t, Y0, MUmax, lag)
```

Arguments

`t`	is a numeric vector indicating the time of the experiment
`Y0`	is the natural logarithm of the initial microbial concentration (`ln(N0)`) at time=0
`MUmax`	is the maximum specific growth rate given in time units
`lag`	is the duration of the lag phase in time units

Details

Model's inputs are:

`t`: time, assuming time zero as the beginning of the experiment.

`Y(t)`: the natural logarithm of the microbial concentration (`ln(N(t))`) measured at time t.

Users should make sure that the microbial concentration input is entered in natural logarithm, `Y(t) = ln(N(t))`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez, <`vcadavez@ipb.pt`> and Ursula Gonzales-Barron, <`ubarron@ipb.pt`>

References

Baranyi J, Roberts TA (1995). "Mathematics of predictive microbiology." *International Journal of Food Microbiology*, **26**, 199-218.

Examples

```
## Example: Baranyi reduced model
library(gslnls)
data(growthred) # simulated data set.
initial_values <- list(Y0 = 0.1, MUmax = 1.7, lag = 5) # define the initial values
fit <- gsl_nls(lnN ~ BaranyiRM(Time, Y0, MUmax, lag),
  data = growthred,
  start = initial_values
)
summary(fit)
```

`bias_factor` *Bias and accuracy factors*

Description

`bias_factor()` and `accuracy_factor()` calculate the multiplicative model bias and accuracy factors commonly used in predictive microbiology model validation.

Usage

```
bias_factor(observed, predicted)

accuracy_factor(observed, predicted)
```

Arguments

`observed, predicted`	Numeric vectors. Values must be positive.

Value

A numeric scalar.

Examples

```
observed <- c(7.0, 6.0, 5.0)
predicted <- c(7.1, 5.9, 5.2)
bias_factor(observed, predicted)
accuracy_factor(observed, predicted)
```

`bixina` *Data concerning* Staphylococcus aureus *microbial inactivation in beef*

Description

A dataset containing time (minutes), repetition and number of microrganisms (ln N).

Usage

```
data(bixina)
```

Format

A data frame with 18 rows and 3 variables:

Time Time in minutes

Rep Repetition

lnN Number of microrganism in ln scale

`BuchananRM` *Buchanan reduced growth model*

Description

`BuchananRM` function to fit the Buchanan reduced growth model to a reduced microbial growth curve. Returns the model parameters estimated according to data collected in microbial growth experiments.

Usage

```
BuchananRM(t, Y0, MUmax, lag)
```

Arguments

`t`	is a numeric vector indicating the time of the experiment
`Y0`	is the natural logarithm of the initial microbial concentration (`ln(N0)`) at time=0
`MUmax`	is the maximum specific growth rate given in time units
`lag`	is the duration of the lag phase in time units

Details

Model's inputs are:

`t`: time, assuming time zero as the beginning of the experiment.

`Y(t)`: the natural logarithm of the microbial concentration (`ln(N(t))`) measured at time t.

Users should make sure that the microbial concentration input is entered in natural logarithm, `Y(t) = ln(N(t))`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez, `<vcadavez@ipb.pt>` and Ursula Gonzales-Barron, `<ubarron@ipb.pt>`

References

Buchanan RL, Whiting RC, Damert WC (1997). "When is simple good enough: a comparison of the Gompertz, Baranyi, and three-phase linear models for fitting bacterial growth curves." *Food Microbiology*, **14**(4), 313-326. ISSN 0740-0020. doi:10.1006/fmic.1997.0125.

Examples

```
## Example: Buchanan reduced model
library(gslnls)
data(growthred) # simulated data set.
initial_values <- list(Y0 = 0, MUmax = 1.7, lag = 5) # define the initial values
fit <- gsl_nls(lnN ~ BuchananRM(Time, Y0, MUmax, lag),
  data = growthred,
  start = initial_values
)
summary(fit)
```

`CMAW` *Cardinal model for water activity*

Description

`CMAW` function to fit the water activity cardinal model (Rosso et al., 1993). Returns the model parameters estimated according to data collected in microbial growth experiments.

Usage

```
CMAW(x, AWmin, MUopt, AWopt)
```

Arguments

`x`	is a numeric vector indicating the water activity of the experiment
`AWmin`	is minimum water activity for growth
`MUopt`	is the optimum growth rate
`AWopt`	is optimum water activity for growth

Details

The model's inputs are:

`x`: Water activity

`sqrtGR`: the square root of the growth rate (h^{-1})

Users should make sure that the growth rate input is entered after a square root transformation, `sqrGR = sqrt(GR)`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez `<vcadavez@ipb.pt>` and Ursula Gonzales-Barron `<ubarron@ipb.pt>`

References

Rosso L, Lobry J, Charles (Bajard) S, Flandrois J (1995). "Convenient Model To Describe the Combined Effects of Temperature and pH on Microbial Growth." *Applied and environmental microbiology*, **61**, 610–6. doi:10.1128/AEM.61.2.610616.1995.

Examples

```
library(gslnls)
data(aw)
initial_values <- list(AWmin = 0.89, MUopt = 1.0, AWopt = 0.98)
fit <- gsl_nls(sqrtGR ~ CMAW(aw, AWmin, MUopt, AWopt),
  data = aw,
  start = initial_values
)
summary(fit)
```

`CMInh` *Cardinal model for growth inhibitors*

Description

`CMInh` function to fit the growth inhibitors cardinal model (Rosso et al, 1993). Returns the model parameters estimated according to data collected in microbial growth experiments.

Usage

```
CMInh(x, MIC, MUopt, alpha)
```

Arguments

`x`	is a numeric vector indicating the inhibitor concentration of the experiment
`MIC`	is the minimum inhibitory concentration (mM or %, accordingly)
`MUopt`	is the optimum growth rate
`alpha`	is the shape parameter of the curve ($\alpha = 1$ the shape is linear; $\alpha > 1$ the shape is downward concave; and $\alpha < 1$ the shape is upward concave)

Details

The model's inputs are:

`x`: growth inhibitor concentration

`sqrtGR`: the square root of the growth rate ($time^{-1}$)

Users should make sure that the growth rate input is entered after a square root transformation, $sqrGR = sqrt(GR)$.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez `<vcadavez@ipb.pt>` and Ursula Gonzales-Barron `<ubarron@ipb.pt>`

References

Rosso L, Lobry J, Charles (Bajard) S, Flandrois J (1995). “Convenient Model To Describe the Combined Effects of Temperature and pH on Microbial Growth.” *Applied and environmental microbiology*, **61**, 610–6. doi:10.1128/AEM.61.2.610616.1995.

Examples

```
library(gslnls)
data(inh)
initial_values <- list(MIC = 0.89, MUopt = 1.0, alpha = 1)
fit <- gsl_nls(sqrtGR ~ CMInh(Conce, MIC, MUopt, alpha),
  data = inh,
  start = initial_values
)
summary(fit)
```

`CMPH` *Cardinal model for pH*

Description

`CMPH` function to fit the pH cardinal model (Rosso et al, 1993). Returns the model parameters estimated according to data collected in microbial growth experiments.

Usage

```
CMPH(x, pHmax, pHmin, MUopt, pHopt)
```

Arguments

`x`	is a numeric vector indicating the pH of the experiment
`pHmax`	is the maximum pH for growth
`pHmin`	is the minimum pH for growth
`MUopt`	is the optimum growth rate
`pHopt`	is the optimum pH for growth

Details

The model's inputs are:

`x`: pH

`sqrtGR`: the square root of the growth rate ($time^{-1}$)

Users should make sure that the growth rate input is entered after a square root transformation, `sqrGR = sqrt(GR)`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez `<vcadavez@ipb.pt>` and Ursula Gonzales-Barron `<ubarron@ipb.pt>`

References

Rosso L, Lobry J, Charles (Bajard) S, Flandrois J (1995). “Convenient Model To Describe the Combined Effects of Temperature and pH on Microbial Growth.” *Applied and environmental microbiology*, **61**, 610–6. doi:10.1128/AEM.61.2.610616.1995.

Examples

```
library(gslnls)
data(ph)
initial_values <- list(pHmax = 9, pHmin = 3, MUopt = 1.0, pHopt = 7)
fit <- gsl_nls(sqrtGR ~ CMPH(pH, pHmax, pHmin, MUopt, pHopt),
  data = ph,
  start = initial_values
)
summary(fit)
```

`CMTI` *Cardinal model for temperature*

Description

`CMTI` function to fit the temperature cardinal model (Rosso et al, 1993). Returns the model parameters estimated according to data collected in microbial growth experiments.

Usage

```
CMTI(x, Tmax, Tmin, MUopt, Topt)
```

Arguments

`x`	is a numeric vector indicating the temperature of the experiment
`Tmax`	maximum temperature for growth
`Tmin`	is minimum temperature for growth
`MUopt`	is the optimum growth rate
`Topt`	is optimum temperature for growth

Details

The model’s inputs are:

`x`: Temperature

`sqrtGR`: the square root of the growth rate (h^{-1})

Users should make sure that the growth rate input is entered after a square root transformation, `sqrGR = sqrt(GR)`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez `<vcadavez@ipb.pt>` and Ursula Gonzales-Barron `<ubarron@ipb.pt>`

References

Rosso L, Lobry J, Charles (Bajard) S, Flandrois J (1995). "Convenient Model To Describe the Combined Effects of Temperature and pH on Microbial Growth." *Applied and environmental microbiology*, **61**, 610–6. doi:10.1128/AEM.61.2.610616.1995.

Examples

```
library(gslnls)
data(salmonella)
initial_values <- list(Tmax = 42, Tmin = 1, MUopt = 1.0, Topt = 37)
fit <- gsl_nls(sqrtGR ~ CMTI(Temp, Tmax, Tmin, MUopt, Topt),
  data = salmonella,
  start = initial_values
)
summary(fit)

plot(salmonella$Temp, salmonella$sqrtGR^2)
lines(salmonella$Temp, fitted(fit)^2, col = "green")
plot(salmonella$Temp, salmonella$sqrtGR)
lines(salmonella$Temp, fitted(fit), col = "red")
```

`compare_models` *Compare fitted predmicror models*

Description

`compare_models()` combines the output of `fit_metrics()` for two or more fitted models. It is useful when choosing between alternative primary growth, inactivation, or cardinal models fitted to the same response scale.

Usage

```
compare_models(..., sort_by = c("AIC", "BIC", "RMSE", "MAE", "none"))
```

Arguments

`...`	`predmicror_fit` objects, or a single list of `predmicror_fit` objects.
`sort_by`	Character string. One of `"AIC"`, `"BIC"`, `"RMSE"`, `"MAE"`, or `"none"`.

Value

A data frame with one row per fitted model.

Examples

```
data(growthfull)
huang <- fit_growth(
  growthfull,
  model = "HuangFM",
  time = "Time",
  response = "lnN",
  start = list(Y0 = 0, Ymax = 22, MUmax = 1.7, lag = 5)
)
baranyi <- fit_growth(
  growthfull,
  model = "BaranyiFM",
  time = "Time",
  response = "lnN",
  start = list(Y0 = 0, Ymax = 22, MUmax = 1.7, lag = 5)
)
compare_models(huang = huang, baranyi = baranyi)
```

`dynamic_profile` *Create a dynamic environmental profile*

Description

`dynamic_profile()` stores time-varying environmental conditions such as temperature, pH, or water activity for dynamic predictive microbiology simulations. The profile is interpolated internally when dynamic models are solved between observed profile points.

Usage

```
dynamic_profile(time, temperature = NULL, ph = NULL, aw = NULL, ...)
```

Arguments

`time`	Numeric vector with profile times.
`temperature`	Optional numeric vector with temperatures at `time`.
`ph`	Optional numeric vector with pH values at `time`.
`aw`	Optional numeric vector with water activity values at `time`.
`...`	Additional named numeric vectors with the same length as `time`.

Value

A `predmicror_dynamic_profile` data frame sorted by time.

See Also

`predict_dynamic_growth()`, `predict_dynamic_inactivation()`

Examples

```
profile <- dynamic_profile(
  time = c(0, 5, 10, 15),
  temperature = c(10, 4, 12, 20)
)
profile
```

`dynamic_sensitivity` *Finite-difference sensitivity for dynamic predictions*

Description

`dynamic_sensitivity()` perturbs parameters in a dynamic prediction and returns unscaled and scaled sensitivity coefficients. It is designed as a lightweight diagnostic for sampling design and parameter identifiability.

Usage

```
dynamic_sensitivity(
  type = c("growth", "inactivation"),
  profile,
  start,
  parameters = NULL,
  relative_delta = 1e-06,
  times = NULL,
  ...
)
```

Arguments

`type`	Character string. One of `"growth"` or `"inactivation"`.
`profile`	A `dynamic_profile()` object or compatible data frame.
`start`	Named list of parameter values passed to the dynamic prediction function.
`parameters`	Character vector of parameter names to perturb. Defaults to all numeric scalar entries in `start`.
`relative_delta`	Relative perturbation size.
`times`	Optional output times.
`...`	Additional arguments passed to `predict_dynamic_growth()` or `predict_dynamic_inactivation()`.

Value

A data frame with time, parameter, prediction, sensitivity, and scaled sensitivity.

Examples

```
profile <- dynamic_profile(time = c(0, 5, 10), temperature = c(10, 15, 20))
sens <- dynamic_sensitivity(
  "growth",
  profile = profile,
  start = list(logN0 = 2, logNmax = 8, a = 0.08, Tmin = 7, lag = 1),
  times = seq(0, 10, by = 2),
  dt = 0.25
)
head(sens)
```

`FangNLM` *Fang no lag growth model*

Description

`FangNLM` function to fit the Fang no lag growth model to an incomplete microbial growth curve. Returns the model parameters estimated according to data collected in microbial growth experiments.

Usage

```
FangNLM(t, Y0, Ymax, MUmax)
```

Arguments

`t`	is a numeric vector indicating the time of the experiment
`Y0`	is the natural logarithm of the initial microbial concentration (`ln(N0)`) at time=0
`Ymax`	is the natural logarithm of the maximum concentration (`ln(Nmax)`) reached by the microorganism
`MUmax`	is the maximum specific growth rate given in time units

Details

Model's inputs are:

`t`: time, assuming time zero as the beginning of the experiment.

`Y(t)`: the natural logarithm of the microbial concentration (`ln(N(t))`) measured at time t.

Users should make sure that the microbial concentration input is entered in natural logarithm, `Y(t) = ln(N(t))`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez, `<vcadavez@ipb.pt>` and Ursula Gonzales-Barron, `<ubarron@ipb.pt>`

References

Fang T, Gurtler JB, Huang L (2012). "Growth Kinetics and Model Comparison of Cronobacter sakazakii in Reconstituted Powdered Infant Formula." *Journal of Food Science*, **77**(9), E247-E255. doi:10.1111/j.17503841.2012.02873.x. Fang T, Liu Y, Huang L (2013). "Growth kinetics of Listeria monocytogenes and spoilage microorganisms in fresh-cut cantaloupe." *Food Microbiology*, **34**(1), 174-181. ISSN 0740-0020. doi:10.1016/j.fm.2012.12.005.

Examples

```
## Example: Fang no lag model
library(gslnls)
data(growthnolag) # simulated data set.
initial_values <- list(Y0 = 0, Ymax = 22, MUmax = 1.7) # define the initial values
fit <- gsl_nls(lnN ~ FangNLM(Time, Y0, Ymax, MUmax),
  data = growthnolag,
  start = initial_values
)
summary(fit)
```

`fit_cardinal` *Fit a cardinal parameter model*

Description

`fit_cardinal()` validates the input data, builds the nonlinear model formula, fits it with `gslnls::gsl_nls()`, and returns a `predmicror_fit` object.

Usage

```
fit_cardinal(data, model, x, response = "sqrtGR", start, ...)
```

Arguments

`data`	A data frame containing the environmental factor and response variables.
`model`	A model name, either quoted or unquoted. See `predmicror_models()`.
`x`	Column containing the environmental factor values, either quoted or unquoted.
`response`	Column containing the response values, either quoted or unquoted. Defaults to `"sqrtGR"`.
`start`	Named list of initial parameter values for the selected model.
`...`	Additional arguments passed to `gslnls::gsl_nls()`.

Details

Cardinal models expect the response to be the square root of the growth rate, usually `sqrtGR`.

Value

A `predmicror_fit` object with the fitted model and metadata.

Examples

```
data(salmonella)
fit <- fit_cardinal(
  salmonella,
  model = "CMTI",
  x = "Temp",
  response = "sqrtGR",
  start = list(Tmax = 42, Tmin = 1, MUopt = 1, Topt = 37)
)
coef(fit)
```

`fit_dynamic_growth` *Fit dynamic microbial growth models*

Description

`fit_dynamic_growth()` estimates selected parameters of a dynamic Huang-type growth model by repeatedly solving the dynamic model and minimizing the residual sum of squares against observed data.

Usage

```
fit_dynamic_growth(
  data,
  profile,
  time,
  response,
  start,
  estimate = NULL,
  fixed = NULL,
  lower = NULL,
  upper = NULL,
  model = "huang",
  secondary = "huang_sqrt",
  scale = c("log10", "ln"),
  dt = 0.01,
  method = "rk4",
  temperature = "temperature",
  optimizer = "L-BFGS-B",
  control = list(),
  ...
)
```

Arguments

`data`	Data frame with observed microbial counts.
`profile`	A `dynamic_profile()` object or compatible data frame.
`time`, `response`	Column names in `data` containing observation time and microbial response.
`start`	Named list of starting parameter values passed to `predict_dynamic_growth()`. Numeric scalar entries are candidates for estimation.
`estimate`	Character vector of parameter names to estimate. If `NULL`, all numeric scalar entries in `start` except those listed in `fixed` are estimated.
`fixed`	Optional named list of parameters to keep fixed during fitting.
`lower`, `upper`	Optional named numeric vectors or lists with lower and upper bounds for estimated parameters.
`model`, `secondary`, `scale`, `dt`, `method`, `temperature`	Arguments passed to `predict_dynamic_growth()`.
`optimizer`	Optimization method passed to `stats::optim()`.
`control`	Optional control list passed to `stats::optim()`.
`...`	Additional arguments passed to `predict_dynamic_growth()`.

Value

A `predmicror_dynamic_fit` object.

Examples

```
profile <- dynamic_profile(time = c(0, 10), temperature = c(20, 20))
obs <- data.frame(time = c(0, 5, 10), logN = c(2, 3.3, 5.1))
fit <- fit_dynamic_growth(
  obs,
  profile = profile,
  time = "time",
  response = "logN",
  start = list(logN0 = 2, logNmax = 8, MUmax = 0.4, lag = 0),
  estimate = "MUmax",
  secondary = "constant",
  dt = 0.25
)
coef(fit)
```

`fit_dynamic_inactivation`

Fit dynamic microbial inactivation models

Description

`fit_dynamic_inactivation()` estimates selected parameters of a dynamic Weibull-Peleg inactivation model by minimizing the residual sum of squares against observed data.

Usage

```
fit_dynamic_inactivation(
  data,
  profile,
  time,
  response,
  start,
  estimate = NULL,
  fixed = NULL,
  lower = NULL,
  upper = NULL,
  model = "weibull_peleg",
  secondary = "constant",
  dt = 0.01,
  method = "rk4",
  temperature = "temperature",
  optimizer = "L-BFGS-B",
  control = list(),
  ...
)
```

Arguments

`data`	Data frame with observed microbial counts.
`profile`	A `dynamic_profile()` object or compatible data frame.
`time, response`	Column names in `data` containing observation time and microbial response.
`start`	Named list of starting parameter values passed to `predict_dynamic_growth()`. Numeric scalar entries are candidates for estimation.
`estimate`	Character vector of parameter names to estimate. If `NULL`, all numeric scalar entries in `start` except those listed in `fixed` are estimated.
`fixed`	Optional named list of parameters to keep fixed during fitting.
`lower, upper`	Optional named numeric vectors or lists with lower and upper bounds for estimated parameters.
`model, secondary, dt, method, temperature`	Arguments passed to `predict_dynamic_inactivation()`.
`optimizer`	Optimization method passed to `stats::optim()`.
`control`	Optional control list passed to `stats::optim()`.
`...`	Additional arguments passed to `predict_dynamic_growth()`.

Value

A `predmicror_dynamic_fit` object.

Examples

```
profile <- dynamic_profile(time = c(0, 10), temperature = c(60, 60))
obs <- data.frame(time = c(0, 5, 10), logN = c(7, 6, 5))
fit <- fit_dynamic_inactivation(
  obs,
  profile = profile,
  time = "time",
  response = "logN",
  start = list(logN0 = 7, b = 0.15, n = 1),
  estimate = "b",
  dt = 0.25
)
coef(fit)
```

`fit_growth` *Fit a primary growth model*

Description

`fit_growth()` validates the input data, builds the nonlinear model formula, fits it with `gslnls::gsl_nls()`, and returns a `predmicror_fit` object.

Usage

```
fit_growth(data, model, time, response = "lnN", start, ...)
```

Arguments

`data`	A data frame containing the time and response variables.
`model`	A model name, either quoted or unquoted. See `predmicror_models()`.
`time`	Column containing time values, either quoted or unquoted.
`response`	Column containing the response values, either quoted or unquoted. Defaults to `"lnN"`.
`start`	Named list of initial parameter values for the selected model.
`...`	Additional arguments passed to `gslnls::gsl_nls()`.

Details

Growth models expect the response to be the natural logarithm of the microbial concentration, usually `lnN`.

Value

A `predmicror_fit` object with the fitted model and metadata.

Examples

```
data(growthfull)
fit <- fit_growth(
  growthfull,
  model = "HuangFM",
  time = "Time",
  response = "lnN",
  start = list(Y0 = 0, Ymax = 22, MUmax = 1.7, lag = 5)
)
coef(fit)
```

`fit_inactivation` *Fit a microbial inactivation model*

Description

`fit_inactivation()` validates the input data, builds the nonlinear model formula, fits it with `gslnls::gsl_nls()`, and returns a `predmicror_fit` object.

Usage

```
fit_inactivation(data, model, time, response = "logN", start, ...)
```

Arguments

`data`	A data frame containing the time and response variables.
`model`	A model name, either quoted or unquoted. See `predmicror_models()`.
`time`	Column containing time values, either quoted or unquoted.
`response`	Column containing the response values, either quoted or unquoted. Defaults to `"logN"`.
`start`	Named list of initial parameter values for the selected model.
`...`	Additional arguments passed to `gslnls::gsl_nls()`.

Details

Inactivation models expect the response to be the base 10 logarithm of the microbial concentration, usually `logN`.

Value

A `predmicror_fit` object with the fitted model and metadata.

Examples

```
data(mafart2005Li11)
fit <- fit_inactivation(
  mafart2005Li11,
  model = "WeibullM",
  time = "Time",
  response = "logN",
  start = list(Y0 = 10, sigma = 3, alpha = 1)
)
coef(fit)
```

`fit_metrics` *Calculate model diagnostics for a fitted predmicror model*

Description

`fit_metrics()` summarizes goodness-of-fit and information criteria for a `predmicror_fit` object. The metrics are calculated on the response scale used in the fitted model.

Usage

```
fit_metrics(object, ...)

## Default S3 method:
fit_metrics(object, ...)

## S3 method for class 'predmicror_fit'
fit_metrics(object, ...)
```

Arguments

`object`	A `predmicror_fit` object.
`...`	Currently unused. Included for future extension and S3 compatibility.

Value

A one-row data frame with columns: `fit`, `model`, `type`, `response`, `response_scale`, `n` (observations), `p` (parameters), `SSE`, `RMSE`, `MAE`, `bias`, `RSE`, `R2`, `adj_R2`, `logLik`, `AIC`, `BIC`, and `converged`.

Examples

```
data(growthfull)
fit <- fit_growth(
  data = growthfull,
  model = "HuangFM",
  time = "Time",
  response = "lnN",
  start = list(Y0 = 0, Ymax = 22, MUmax = 1.7, lag = 5)
```

```
)
fit_metrics(fit)
```

`fit_omnibus` *Fit omnibus predictive microbiology models*

Description

`fit_omnibus()` fits a nonlinear mixed-effects model in which the primary model is one of the parameterised `predmicror` primary models and one or more primary-model parameters are described by secondary covariate formulas.

Usage

```
fit_omnibus(
  data,
  type = c("growth", "inactivation"),
  primary,
  time,
  response,
  group,
  secondary = NULL,
  random,
  correlation = NULL,
  start,
  method = "ML",
  control = NULL,
  ...
)

fit_omnibus_growth(
  data,
  primary,
  time,
  response,
  group,
  secondary = NULL,
  random,
  correlation = NULL,
  start,
  method = "ML",
  control = NULL,
  ...
)

fit_omnibus_inactivation(
  data,
```

```
  primary,
  time,
  response,
  group,
  secondary = NULL,
  random,
  correlation = NULL,
  start,
  method = "ML",
  control = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`type`	Character string. One of `"growth"` or `"inactivation"`.
`primary`	Character string naming a primary model registered in `predmicror`, for example `"HuangNLM"` or `"WeibullM"`.
`time`, `response`, `group`	Column names for time, response, and grouping variable.
`secondary`	Optional named list of formulas, one per primary-model parameter. Parameters not listed are modelled as intercept-only effects.
`random`	Random-effects formula passed to `nlme::nlme()`. If the formula does not include a grouping term, `| group` is added automatically.
`correlation`	Optional correlation structure. Use `"AR1"` for `nlme::corAR1()` within groups, `NULL` for none, or pass an `nlme` correlation object.
`start`	Numeric vector of starting values for fixed effects, in the order expected by `nlme::nlme()`.
`method`	Estimation method passed to `nlme::nlme()`.
`control`	Optional `nlme::nlmeControl()` object.
`...`	Additional arguments passed to `nlme::nlme()`.

Value

A `predmicror_omnibus_fit` object.

Examples

```
set.seed(1)
dat <- do.call(rbind, lapply(1:4, function(g) {
  Time <- c(1, 2, 4, 6, 8, 10)
  sigma <- 5 + 0.4 * g
  data.frame(
    Condition = g,
    Time = Time,
    Temp = 55 + g,
```

```
    logN = WeibullM(Time, Y0 = 7, sigma = sigma, alpha = 1.1) +
      rnorm(length(Time), 0, 0.03)
  )
}))
fit <- fit_omnibus_inactivation(
  dat,
  primary = "WeibullM",
  time = "Time",
  response = "logN",
  group = "Condition",
  secondary = list(sigma = ~ Temp),
  random = Y0 ~ 1,
  start = c(Y0 = 7, sigma = 1, sigma.Temp = 0.08, alpha = 1)
)
fit_metrics(fit)
```

`GeeraerdST` *Geeraerd inactivation model*

Description

`GeeraerdST` inactivation model for microbial inactivation curve. Returns the model parameters estimated according to data collected in microbial inactivation experiments.

Usage

```
GeeraerdST(x, Y0, Yres, kmax, Sl)
```

Arguments

`x`	is a numeric vector indicating the heating time under a constant temperature of the experiment
`Y0`	is the initial (time=0) bacterial concentration (ln(N0))
`Yres`	is a low asymptote reflecting the presence of a resistant sub-population (ln(Nres))
`kmax`	is the maximum inactivation rate
`Sl`	represents shoulder phase preceding the sharp inactivation slope of the curve

Details

The model's inputs are:

`t`: time, assuming time zero as the beginning of the experiment.

`N(t)`: the bacterial concentration measured at time t.

Users should make sure that the bacterial concentration input is entered in natural logarithm, `Y(t) = ln(N(t))`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez `<vcadavez@ipb.pt>` and Ursula Gonzales-Barron `<ubarron@ipb.pt>`

References

Geeraerd AH, Valdramidis VP, Van Impe JF (2005). "GInaFiT, a freeware tool to assess non-log-linear microbial survivor curves." *International Journal of Food Microbiology*, **102**(1), 95-105. ISSN 0168-1605. doi:10.1016/j.ijfoodmicro.2004.11.038.

Examples

```
library(gslnls)
data(mafart2005Li11)
mafart2005Li11$lnN <- log(10) * mafart2005Li11$logN
initial_values <- list(Y0 = 18, Yres = 2, kmax = 0.7, Sl = 4)
fit <- gsl_nls(lnN ~ GeeraerdST(Time, Y0, Yres, kmax, Sl),
  data = mafart2005Li11,
  start = initial_values
)
summary(fit)

plot(lnN ~ Time, data = mafart2005Li11)
lines(mafart2005Li11$Time, predict(fit), col = "blue")
```

`growthfull` *Data of a complete curve of microbial growth*

Description

A dataset containing simulated data for a full growth model

Usage

```
data(growthfull)
```

Format

A data frame with 13 rows and 3 variables.

Time Time in minutes

logN Number of microrganism in log10 scale

lnN Number of microrganism in ln scale

`growthnolag` *Data of a no lag curve of microbial growth*

Description

A dataset containing simulated data for a no lag growth model

Usage

```
data(growthnolag)
```

Format

A data frame with 10 rows and 3 variables.

Time Time in minutes

logN Number of microrganism in log10 scale

lnN Number of microrganism in ln scale

`growthred` *Data of a reduced curve of microbial growth*

Description

A dataset containing simulated data for a reduced growth model

Usage

```
data(growthred)
```

Format

A data frame with 9 rows and 3 variables.

Time Time in minutes

logN Number of microrganism in log10 scale

lnN Number of microrganism in ln scale

`HuangFM` *Huang full growth model*

Description

`HuangFM` function to fit the Huang full growth model to complete microbial growth curve. Returns the model parameters estimated according to data collected in microbial growth experiments.

Usage

```
HuangFM(t, Y0, Ymax, MUmax, lag)
```

Arguments

`t`	is a numeric vector indicating the time of the experiment
`Y0`	is the natural logarithm of the initial microbial concentration (`ln(N0)`) at time=0
`Ymax`	is the natural logarithm of the maximum concentration (`ln(Nmax)`) reached by the microorganism
`MUmax`	is the maximum specific growth rate given in time units
`lag`	is the duration of the lag phase in time units

Details

Model's inputs are:

`t`: time, assuming time zero as the beginning of the experiment.

`Y(t)`: the natural logarithm of the microbial concentration (`ln(N(t))`) measured at time t.

Users should make sure that the microbial concentration input is entered in natural logarithm, `Y(t) = ln(X(t))`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez, `<vcadavez@ipb.pt>` and Ursula Gonzales-Barron, `<ubarron@ipb.pt>`

References

Huang L (2008). "Growth Kinetics of Listeria monocytogenes in Broth and Beef Frankfurters-Determination of Lag Phase Duration and Exponential Growth Rate under Isothermal Conditions." *Journal of Food Science*, **73**(5), E235-E242. doi:10.1111/j.17503841.2008.00785.x.

Examples

```
## Example: Huang full model
library(gslnls)
data(growthfull) # simulated data set.
initial_values <- list(Y0 = 0, Ymax = 22, MUmax = 1.7, lag = 5) # define the initial values
## Call the fitting function
fit <- gsl_nls(lnN ~ HuangFM(Time, Y0, Ymax, MUmax, lag),
  data = growthfull,
  start = initial_values
)
summary(fit)

confint(fit)

preds <- data.frame(predict(fit, interval = "prediction", level = 0.95))
plot(lnN ~ Time, data = growthfull, ylim = c(-1, 22))
lines(growthfull$Time, preds$fit, col = "blue")
lines(growthfull$Time, preds$upr, col = "red")
lines(growthfull$Time, preds$lwr, col = "red")
```

`HuangNLM` *Huang no lag growth model*

Description

`HuangNLM` function to fit the Huang no lag growth model to an incomplete microbial growth curve. Returns the model parameters estimated according to data collected in microbial growth experiments.

Usage

```
HuangNLM(t, Y0, Ymax, MUmax)
```

Arguments

`t`	is a numeric vector indicating the time of the experiment
`Y0`	is the natural logarithm of the initial microbial concentration (`ln(N0)`) at time=0
`Ymax`	is the natural logarithm of the maximum concentration (`ln(Nmax)`) reached by the microorganism
`MUmax`	is the maximum specific growth rate given in time units

Details

Model's inputs are:

`t`: time, assuming time zero as the beginning of the experiment.

`Y(t)`: the natural logarithm of the microbial concentration (`ln(N(t))` measured at time t.

Users should make sure that the microbial concentration input is entered in natural logarithm, `Y(t) = ln(N(t))`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez, <`vcadavez@ipb.pt`> and Ursula Gonzales-Barron, <`ubarron@ipb.pt`>

References

Huang (2013).

Examples

```
## Example: Huang no lag model
library(gslnls)
data(growthnolag) # simulated data set.
initial_values <- list(Y0 = 0, Ymax = 22, MUmax = 1.7) # define the initial values
## Call the fitting function
fit <- gsl_nls(lnN ~ HuangNLM(Time, Y0, Ymax, MUmax),
  data = growthnolag,
  start = initial_values
)
summary(fit)
```

`HuangRGS` *Huang reparameterized Gompertz survival model*

Description

`HuangRGS` reparametrized Gompertz survival model for microbial inactivation. Returns the model parameters estimated according to data collected in microbial inactivation experiments.

Usage

```
HuangRGS(x, Y0, k, M)
```

Arguments

`x`	is a numeric vector indicating the heating time under a constant temperature of the experiment
`Y0`	is the initial microbial concentration (ln(cfu 1/g))
`k`	is the inactivation rate (1/s)
`M`	is a time constant (s)

Details

The model's inputs are:

`t`: time, assuming time zero as the beginning of the experiment.

`Y(t)`: the natural logarithm of the bacterial concentration (`ln(X(t))`) measured at time t.

Users should make sure that the bacterial concentration input is entered in natural logarithm, `Y(t) = ln(X(t))`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez `<vcadavez@ipb.pt>` and Ursula Gonzales-Barron `<ubarron@ipb.pt>`

References

Huang L (2009). "Thermal inactivation of Listeria monocytogenes in ground beef under isothermal and dynamic temperature conditions." *Journal of Food Engineering*, **90**(3), 380-387. ISSN 0260-8774. doi:10.1016/j.jfoodeng.2008.07.011.

Examples

```
library(gslnls)
data(bixina)
initial_values <- list(Y0 = 5.6, k = 0.37, M = 6.8)
fit <- gsl_nls(lnN ~ HuangRGS(Time, Y0, k, M),
  data = bixina,
  start = initial_values
)
summary(fit)

plot(lnN ~ Time, data = bixina)
lines(bixina$Time, predict(fit), col = "blue")
```

`HuangRM` *Huang reduced growth model*

Description

`HuangRM` function to fit the Huang reduced growth model to a reduced microbial growth curve. Returns the model parameters estimated according to data collected in microbial growth experiments.

Usage

```
HuangRM(t, Y0, MUmax, lag)
```

Arguments

`t`	is a numeric vector indicating the time of the experiment
`Y0`	is the natural logarithm of the initial microbial concentration (`ln(N0)`) at time=0
`MUmax`	is the maximum specific growth rate given in time units
`lag`	is the duration of the lag phase in time units

Details

Model's inputs are:

`t`: time, assuming time zero as the beginning of the experiment.

`Y(t)`: the natural logarithm of the microbial concentration (`ln(N(t))`) measured at time t.

Users should make sure that the microbial concentration input is entered in natural logarithm, `Y(t) = ln(N(t))`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez, `<vcadavez@ipb.pt>` & Ursula Gonzales-Barron, `<ubarron@ipb.pt>`

References

Huang L (2008). "Growth Kinetics of Listeria monocytogenes in Broth and Beef Frankfurters-Determination of Lag Phase Duration and Exponential Growth Rate under Isothermal Conditions." *Journal of Food Science*, **73**(5), E235-E242. doi:10.1111/j.17503841.2008.00785.x.

Examples

```
## Example: Huang reduced model
library(gslnls)
data(growthred) # simulated data set.
initial_values <- list(Y0 = 0, MUmax = 1.7, lag = 5) # define the initial values
fit <- gsl_nls(lnN ~ HuangRM(Time, Y0, MUmax, lag),
  data = growthred,
  start = initial_values
)
summary(fit)
```

`inh` *Data of INH antimicrobials*

Description

A dataset containing antimicrobial concentration and growth rate data

Usage

```
data(inh)
```

Format

A data frame with 8 rows and 3 variables.

Conce Antimicrobial concentration

sqrtGR Square root of the growth rate

GR Growth rate

`mafart2005Li11` *Data of microbial inactivation Albert and Mafart (2005)*

Description

A dataset containing time and log10 counts for microbial inactivation

Usage

```
data(mafart2005Li11)
```

Format

A data frame with 10 rows and 2 variables.

Time Time

logN Number of microrganism in log10 scale

`omnibus_secondary` *Define an omnibus secondary model*

Description

`omnibus_secondary()` defines a secondary model specification that can be used inside `fit_omnibus()`, `fit_omnibus_growth()`, or `fit_omnibus_inactivation()`. It allows a primary-model parameter to be described by another parameterised model already available in `predmicror`.

Usage

```
omnibus_secondary(model, x, transform = c("identity", "square"))
```

Arguments

`model`	Character string naming a registered `predmicror` model, for example `"CMTI"`, `"CMAW"`, `"CMPH"`, or `"CMInh"`.
`x`	Character string naming the covariate column used by the secondary model.
`transform`	Character string. One of `"identity"` or `"square"`.

Value

An `omnibus_secondary` specification.

Examples

```
omnibus_secondary("CMTI", x = "Temp")
omnibus_secondary("CMTI", x = "Temp", transform = "square")
```

`ph` *Data pH*

Description

A dataset containing pH and growth rate data

Usage

```
data(ph)
```

Format

A data frame with 14 rows and 3 variables.

pH pH of food

sqrtGR Square root of the growth rate

GR Growth rate

`predict_dynamic_growth` *Predict microbial growth under dynamic environmental conditions*

Description

`predict_dynamic_growth()` solves a differential Huang-type growth model over a time-varying environmental profile. It is intended for forward prediction under dynamic temperature conditions and uses an internal fourth-order Runge-Kutta solver.

Usage

```
predict_dynamic_growth(
  profile,
  model = "huang",
  secondary = "huang_sqrt",
  start,
  times = NULL,
  scale = c("log10", "ln"),
  dt = 0.01,
  method = "rk4",
  temperature = "temperature"
)
```

Arguments

`profile`	A `dynamic_profile()` object or data frame with a `time` column and an environmental variable, usually `temperature`.
`model`	Character string. Currently only `"huang"` is supported.
`secondary`	Character string defining the secondary model for the growth rate. One of `"huang_sqrt"`, `"huang_full_sqrt"`, or `"constant"`.
`start`	Named list of parameters. Use `logN0` and `logNmax` for base-10 input, or `Y0` and `Ymax` for natural-log input. For `secondary = "huang_sqrt"`, supply `a` and `Tmin`. For `secondary = "huang_full_sqrt"`, also supply `b` and `Tmax`. For `secondary = "constant"`, supply `MUmax`.
`times`	Optional numeric vector of output times. If `NULL`, a regular sequence covering the profile is created from `dt`.
`scale`	Scale of the returned `response` column. One of `"log10"` or `"ln"`.
`dt`	Maximum integration step used by the internal RK4 solver.
`method`	Solver method. Currently only `"rk4"` is implemented.
`temperature`	Name of the temperature column in `profile`.

Value

A `predmicror_dynamic_prediction` data frame with time, prediction, `lnN`, `logN`, temperature, and metadata attributes.

Examples

```
profile <- dynamic_profile(
  time = c(0, 5, 10, 15, 20),
  temperature = c(10, 4, 15, 15, 10)
)
pred <- predict_dynamic_growth(
  profile = profile,
  start = list(logN0 = 2, logNmax = 8.8, a = 0.0886, Tmin = 8.91, lag = 2),
  dt = 0.25
)
head(pred)
```

`predict_dynamic_inactivation` *Predict microbial inactivation under dynamic environmental conditions*

Description

`predict_dynamic_inactivation()` solves a dynamic Weibull-Peleg type inactivation model over a time-varying environmental profile. The output is returned on the base-10 logarithmic scale.

Usage

```
predict_dynamic_inactivation(
  profile,
  model = "weibull_peleg",
  secondary = "constant",
  start,
  times = NULL,
  dt = 0.01,
  method = "rk4",
  temperature = "temperature"
)
```

Arguments

`profile`	A `dynamic_profile()` object or data frame with a `time` column.
`model`	Character string. Currently only `"weibull_peleg"` is supported.
`secondary`	Character string defining the secondary model for the Peleg rate parameter `b`. Use `"constant"` for constant conditions or `"z_value"` for a log-linear temperature effect.
`start`	Named list. Supply `logN0`, shape parameter `n`, and either `b` for `secondary = "constant"` or `b_ref`, `T_ref`, and `z` for `secondary = "z_value"`.
`times`	Optional numeric vector of output times.
`dt`	Maximum integration step used by the internal RK4 solver.
`method`	Solver method. Currently only `"rk4"` is implemented.
`temperature`	Name of the temperature column in `profile`.

Value

A `predmicror_dynamic_prediction` data frame with time, `logN`, `log_survival`, temperature, and metadata attributes.

Examples

```
profile <- dynamic_profile(time = c(0, 10, 20), temperature = c(55, 58, 60))
pred <- predict_dynamic_inactivation(
  profile = profile,
  start = list(logN0 = 7, b_ref = 0.15, T_ref = 55, z = 8, n = 1.2),
  secondary = "z_value",
  dt = 0.25
)
head(pred)
```

`predmicror_assistant` *Assistant for predmicror*

Description

Answer questions about predmicror models using a deterministic local registry and, when available, an optional local Ollama model for prose. Model-fitting code is generated from package metadata and statically checked before being returned.

Usage

```
predmicror_assistant(
  query,
  model = "llama3-groq-tool-use:8b",
  root = NULL,
  data = NULL,
  file = NULL,
  sheet = NULL,
  sep = NULL,
  dec = ".",
  na.strings = c("", "NA"),
  task = NULL,
  time = NULL,
  response = NULL,
  temperature = NULL,
  ph = NULL,
  aw = NULL,
  inhibitor = NULL,
  return_context = FALSE,
  conversation = NULL,
  backend = c("auto", "ollama", "deterministic"),
  prefer_wrappers = TRUE,
```

```
  verify_code = TRUE,
  return_trace = FALSE
)
```

Arguments

`query`	Character question to ask.
`model`	Ollama model name used when `backend` is `"auto"` or `"ollama"`.
`root`	Path to the package root for context collection. Defaults to the installed package path when available, otherwise the current working directory.
`data`	Optional data frame to profile and use for data-aware code generation.
`file`	Optional path to a .csv, .txt, .tsv, .xls, or .xlsx file to read, profile, and use for data-aware code generation.
`sheet`	Optional Excel sheet name or index when `file` is .xls or .xlsx.
`sep`	Optional field separator for delimited text files. If NULL, a simple separator detector is used.
`dec`	Decimal mark for delimited text files.
`na.strings`	Character vector of strings to treat as missing values when reading delimited text files.
`task`	Optional data task override. Use one of `"growth"`, `"inactivation"`, or `"cardinal"`; `NULL` keeps automatic detection.
`time`, `response`, `temperature`, `ph`, `aw`, `inhibitor`	Optional column-name overrides used when `data` or `file` is supplied.
`return_context`	Logical; if TRUE, returns a list with answer and context.
`conversation`	Optional list or character vector with prior questions and answers to include as conversational context.
`backend`	Character string. One of `"auto"`, `"ollama"`, or `"deterministic"`. `"auto"` uses Ollama when the CLI is available and otherwise falls back to deterministic registry-based output.
`prefer_wrappers`	Logical; if TRUE, generated fitting examples prefer `fit_growth()`, `fit_inactivation()`, and `fit_cardinal()` over direct `gslnls::gsl_nls()` calls.
`verify_code`	Logical; if TRUE, statically checks generated R code with `parse()` and simple model-scale/signature rules.
`return_trace`	Logical; if TRUE, returns prompt, backend, candidates, generated code, and validation metadata for debugging.

Value

Character response by default; list with answer and context when `return_context = TRUE` or list with trace when `return_trace = TRUE`.

Examples

```
predmicror_assistant("How do I fit a Huang model?", backend = "deterministic")
```

`predmicror_assistant_app`

Launch the predmicror assistant Shiny app

Description

Starts the local assistant app bundled with the package. The app can read delimited text and Excel files, preview uploaded data, profile columns, apply optional manual column overrides, and pass the data profile to `predmicror_assistant()`. If the bundled app is not available, a fallback app is created from the installed R functions.

Usage

```
predmicror_assistant_app(
  model = "llama3-groq-tool-use:8b",
  root = NULL,
  host = "127.0.0.1",
  port = NULL,
  launch.browser = interactive()
)
```

Arguments

`model`	Default Ollama model name used by `predmicror_assistant()`. The app also exposes a model selector populated from `ollama list` when Ollama is available.
`root`	Optional package root used for assistant context collection.
`host`	Host passed to `shiny::runApp()`. Defaults to `"127.0.0.1"`.
`port`	Optional port passed to `shiny::runApp()`.
`launch.browser`	Logical; whether to launch a browser.

Value

Runs the Shiny app.

Examples

```
predmicror_assistant_app()
```

`predmicror_augment` *Extract fitted values and residuals from a predmicror fit*

Description

`predmicror_augment()` returns the original data, or optional new data, with columns containing fitted values and residuals. It is intentionally lightweight and does not require the `broom` package.

Usage

```
predmicror_augment(object, ...)

## Default S3 method:
predmicror_augment(object, ...)

## S3 method for class 'predmicror_fit'
predmicror_augment(object, newdata = NULL, ...)

## S3 method for class 'predmicror_fit'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

`object`, `x`	A `predmicror_fit` object.
`...`	Additional arguments passed to `predict()`.
`newdata`	Optional data frame used for prediction. If `NULL`, the data stored in the fitted object are used.
`row.names`, `optional`	Arguments required by the base `as.data.frame()` generic. They are accepted for method compatibility.

Value

A data frame containing the original columns plus `.fitted`, `.resid` when the response column is available, `.model`, and `.type`.

Examples

```
data(growthfull)
fit <- fit_growth(
  data = growthfull,
  model = "HuangFM",
  time = "Time",
  response = "lnN",
  start = list(Y0 = 0, Ymax = 22, MUmax = 1.7, lag = 5)
)
head(predmicror_augment(fit))
```

`predmicror_fit_methods`

Methods for `predmicror_fit` *objects*

Description

These methods delegate to the fitted nonlinear least-squares object returned by `gslnls::gsl_nls()`, while preserving the model metadata added by the `fit_*()` wrappers.

Usage

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

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

## S3 method for class 'predmicror_fit'
predict(object, newdata = NULL, ...)

## S3 method for class 'predmicror_fit'
plot(
  x,
  xlab = x$x,
  ylab = paste0(x$response, " (", x$response_scale, ")"),
  ...
)

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

## S3 method for class 'predmicror_fit'
fitted(object, ...)

## S3 method for class 'predmicror_fit'
residuals(object, ...)

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

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

## S3 method for class 'predmicror_fit'
AIC(object, ..., k = 2)

## S3 method for class 'predmicror_fit'
```

```
BIC(object, ...)
```

Arguments

`x`, `object`	A `predmicror_fit` object.
`...`	Additional arguments passed to the underlying method.
`newdata`	Optional data frame for prediction. If omitted, predictions are computed for the original data.
`xlab`, `ylab`	Axis labels used by `plot()`.
`k`	Penalty used by `AIC()`.

Value

The value returned by the corresponding method for the underlying nonlinear model object. `plot()` invisibly returns `x`.

`predmicror_models` *List models available through the fitting wrappers*

Description

List models available through the fitting wrappers

Usage

```
predmicror_models(type = "all")
```

Arguments

`type`	Character string. One of `"all"`, `"growth"`, `"inactivation"`, or `"cardinal"`.

Value

A data frame with the model type, model name, expected response scale, and parameters that must be supplied in `start`.

Examples

```
predmicror_models()
predmicror_models("growth")
```

`predmicror_omnibus_methods`

Methods for omnibus fits

Description

Methods for omnibus fits

Usage

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

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

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

## S3 method for class 'predmicror_omnibus_fit'
fitted(object, ...)

## S3 method for class 'predmicror_omnibus_fit'
residuals(object, ...)

## S3 method for class 'predmicror_omnibus_fit'
predict(object, newdata = NULL, level = 0, ...)

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

## S3 method for class 'predmicror_omnibus_fit'
AIC(object, ..., k = 2)

## S3 method for class 'predmicror_omnibus_fit'
BIC(object, ...)
```

Arguments

`x, object`	A `predmicror_omnibus_fit` object.
`...`	Additional arguments passed to the underlying `nlme` method.
`newdata`	Optional data frame for prediction.
`level`	Prediction level passed to `stats::predict()`.
`k`	Penalty used by `AIC()`.

Value

The value returned by the corresponding method.

`RichardsNLM` *Richards no lag growth model*

Description

`RichardsNLM` function to fit the Richards no lag growth model to an incomplete microbial growth curve. Returns the model parameters estimated according to data collected in microbial growth experiments.

Usage

```
RichardsNLM(t, Y0, Ymax, MUmax, m = 1)
```

Arguments

`t`	is a numeric vector indicating the time of the experiment
`Y0`	is the natural logarithm of the initial microbial concentration (`ln(N0)`) at time=0
`Ymax`	is the natural logarithm of the maximum concentration (`ln(Nmax)`) reached by the microorganism
`MUmax`	is the maximum specific growth rate given in time units
`m`	is the shape parameter of the Richards model (default = 1)

Details

Model's inputs are:

`t`: time, assuming time zero as the beginning of the experiment.

`Y(t)`: the natural logarithm of the microbial concentration (`ln(N(t))`) measured at time t.

Users should make sure that the microbial concentration input is entered in natural logarithm, `Y(t) = ln(N(t))`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez, <`vcadavez@ipb.pt`> and Ursula Gonzales-Barron, <`ubarron@ipb.pt`>

References

Richards JF (1959). "A flexible growth function for empirical use." *J Exp Bot*, **1**(10), 290-310.

Examples

```
## Example: Richards no lag model
library(gslnls)
data(growthnolag) # simulated data set.
initial_values <- list(Y0 = 0, Ymax = 22, MUmax = 1.7)
## Fitting the function to the experimental data
fit <- gsl_nls(lnN ~ RichardsNLM(Time, Y0, Ymax, MUmax),
  data = growthnolag,
  start = initial_values
)
summary(fit)
```

`RossoFM` *Rosso full growth model*

Description

`RossoFM` function to fit the Rosso full growth model to complete microbial growth curve. Returns the model parameters estimated according to data collected in microbial growth experiments.

Usage

```
RossoFM(t, Y0, Ymax, MUmax, lag)
```

Arguments

`t`	is a numeric vector indicating the time of the experiment
`Y0`	is the natural logarithm of the initial microbial concentration (`ln(N0)`) at time=0
`Ymax`	is the natural logarithm of the maximum concentration (`ln(Nmax)`) reached by the microorganism
`MUmax`	is the maximum specific growth rate given in time units
`lag`	is the duration of the lag phase in time units

Details

Model's inputs are:

`t`: time, assuming time zero as the beginning of the experiment.

`Y(t)`: the natural logarithm of the microbial concentration (`ln(N(t))`) measured at time t.

Users should make sure that the microbial concentration input is entered in natural logarithm, `Y(t) = ln(X(t))`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez (<`vcadavez@ipb.pt`>) and Ursula Gonzales-Barron (<`ubarron@ipb.pt`>)

References

Rosso L, Bajard S, Flandrois JP, Lahellec C, Fournaud J, Veit P (1996). "Differential growth of Listeria monocytogenes at 4 and 8 ºC: Consequences for the Shelf Life of Chilled Products." *Journal of Food Protection*, **59**(9), 944-949. ISSN 0362-028X. doi:10.4315/0362028X59.9.944.

Examples

```
## Example: Rosso full model
library(gslnls)
data(growthfull) # simulated data set.
initial_values <- list(Y0 = 0.04, Ymax = 21, MUmax = 1.9, lag = 5.0) # define the initial values
fit <- gsl_nls(lnN ~ RossoFM(Time, Y0, Ymax, MUmax, lag),
  data = growthfull,
  start = initial_values
)
summary(fit)
```

`salmonella` *Potential growth of* Salmonella typhimurium *on cooked chicken*

Description

A dataset containing the specific growth rate of *Salmonella typhimurium* on cooked chicken

Usage

```
data(salmonella)
```

Format

A data frame with 21 rows and 3 variables:

Temp	Temperature (ºC)
GR	Specific growth rate (log10/h)
sqrtGR	Square root of the specific growth rate (GR)

References

Oscar TP (2002). "Development and validation of a tertiary simulation model for predicting the potential growth of Salmonella typhimurium on cooked chicken." *International Journal of Food Microbiology*, **76**(3), 177-190. ISSN 0168-1605. doi:10.1016/S01681605(02)000259.

`validate_omnibus_leave_one_out` *Validate an omnibus fit by leaving out one group*

Description

Validate an omnibus fit by leaving out one group

Usage

```
validate_omnibus_leave_one_out(object, group_value, level = 0, ...)
```

Arguments

`object`	A `predmicror_omnibus_fit` object.
`group_value`	Group value to leave out.
`level`	Prediction level. Defaults to `0` for population-level prediction of the left-out group.
`...`	Additional arguments passed to `fit_omnibus()` during refitting.

Value

A list with the refitted model, validation data, predictions, residuals, bias factor, and accuracy factor.

`WeibullM` *Weibull inactivation model Mafart*

Description

`WeibullM` inactivation model for microbial inactivation curve. Returns the model parameters estimated according to data collected in microbial inactivation experiments.

Usage

```
WeibullM(x, Y0, sigma, alpha)
```

Arguments

`x`	is a numeric vector indicating the heating time under a constant temperature of the experiment
`Y0`	is the natural logarithm of the initial (time=0) bacterial concentration (N0)
`sigma`	is the time of first decimal reduction
`alpha`	which is a shape parameter

Details

The model's inputs are:

`t`: time, assuming time zero as the beginning of the experiment.

`Y(t)`: the natural logarithm of the bacterial concentration (`ln(N(t))`) measured at time t.

Users should make sure that the bacterial concentration input is entered in natural logarithm, `Y(t) = ln(N(t))`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez `<vcadavez@ipb.pt>` and Ursula Gonzales-Barron `<ubarron@ipb.pt>`

References

Mafart P, Couvert O, Gaillard S, Leguerinel (2002). "On calculating sterility in thermal preservation methods: application of the Weibull frequency distribution model." *International Journal of Food Microbiology*, **72**, 107-113.

Examples

```
library(gslnls)
data(bixina)
initial_values <- list(Y0 = 5.75, sigma = 12.8, alpha = 2.4)
fit <- gsl_nls(lnN ~ WeibullM(Time, Y0, sigma, alpha),
  data = bixina,
  start = initial_values
)
summary(fit)

plot(lnN ~ Time, data = bixina)
lines(bixina$Time, predict(fit), col = "blue")
```

`WeibullMM` *Weibull inactivation modified model Mafart*

Description

`WeibullMM` inactivation model for microbial inactivation curve. Returns the model parameters estimated according to data collected in microbial inactivation experiments.

Usage

```
WeibullMM(x, Y0, Yres, sigma, alpha)
```

Arguments

`x`	is a numeric vector indicating the heating time under a constant temperature of the experiment
`Y0`	is the log10 of the initial bacterial concentration (at time t=0)
`Yres`	is the log10 of the residual bacterial concentration (at the end of the experiment)
`sigma`	represents the time of the first decimal reduction concentration for the part of the population not belonging to Yres
`alpha`	is the shape parameter and allows to catch the curve concavity or convexity

Details

The model's inputs are:

`t`: time, assuming time zero as the beginning of the experiment.

`Y(t)`: the bacterial concentration ($Y(t)$) measured at time t.

Users should make sure to use the base 10 logarithm bacterial concentration (Y(t)) as input.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez `<vcadavez@ipb.pt>` and Ursula Gonzales-Barron `<ubarron@ipb.pt>`

References

Mafart et al. (2005).

Examples

```
library(gslnls)
data(bixina)
initial_values <- list(Y0 = 5.6, Yres = 1, sigma = 2, alpha = 1)
bixina$logN <- log10(exp(bixina$lnN))
fit <- gsl_nls(logN ~ WeibullMM(Time, Y0, Yres, sigma, alpha),
  data = bixina,
  start = initial_values
)
summary(fit)

plot(logN ~ Time, data = bixina)
lines(bixina$Time, predict(fit), col = "blue")
```

`WeibullPH` *Weibull inactivation model Peleg and Huang*

Description

`WeibullPH` inactivation model for microbial inactivation curve. Returns the model parameters estimated according to data collected in microbial inactivation experiments.

Usage

```
WeibullPH(t, Y0, k, alpha)
```

Arguments

`t`	is a numeric vector indicating the time of the experiment
`Y0`	is the natural logarithm of the initial (time=0) bacterial concentration
`k`	is the inactivation rate (ln units/time)
`alpha`	is the shape parameter

Details

The model's inputs are:

t: time, assuming time zero as the beginning of the experiment.

Y(t): the natural logarithm of the bacterial concentration X(t) measured at time t.

Users should make sure that the bacterial concentration input is entered in natural logarithm, Y(t) = ln(X(t)).

The following parameters can be estimated using Weibull function:

`t`: is heating time under a constant temperature

`Y0`: is the initial (time=0) bacterial counts in natural logarithm of the initial bacterial counts;

`k`: is the inactivation rate (ln units/time)

`alpha`: is the shape parameter of the survival curve

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez `<vcadavez@ipb.pt>` and Ursula Gonzales-Barron `<ubarron@ipb.pt>`

References

Huang L (2009). "Thermal inactivation of Listeria monocytogenes in ground beef under isothermal and dynamic temperature conditions." *Journal of Food Engineering*, **90**(3), 380-387. ISSN 0260-8774. doi:10.1016/j.jfoodeng.2008.07.011.

Examples

```
library(gslnls)
data(bixina)
initial_values <- list(Y0 = 6.0, k = 1.0, alpha = 0.2)
fit <- gsl_nls(lnN ~ WeibullPH(Time, Y0, k, alpha),
  data = bixina,
  start = initial_values
)
summary(fit)

plot(lnN ~ Time, data = bixina)
lines(bixina$Time, predict(fit), col = "blue")
```

`ZwieteringFM` *Zwietering full growth model*

Description

`ZwieteringFM` function to fit the Zwietering full growth model to a complete microbial growth curve. Returns the model parameters estimated according to data collected in microbial growth experiments.

Usage

```
ZwieteringFM(t, Y0, Ymax, MUmax, lag)
```

Arguments

`t`	is a numeric vector indicating the time of the experiment
`Y0`	is the natural logarithm of the initial microbial concentration (`ln(N0)`) at time=0
`Ymax`	is the natural logarithm of the maximum concentration (`ln(Nmax)`) reached by the microorganism
`MUmax`	is the maximum specific growth rate given in time units
`lag`	is the duration of the lag phase in time units

Details

Model's inputs are:

`t`: time, assuming time zero as the beginning of the experiment.

`Y(t)`: the natural logarithm of the microbial concentration (`ln(N(t))`) measured at time t.

Users should make sure that the microbial concentration input is entered in natural logarithm, `Y(t) = ln(X(t))`.

Value

A numeric vector with the fitted values

Author(s)

Vasco Cadavez <`vcadavez@ipb.pt`> and Ursula Gonzales-Barron <`ubarron@ipb.pt`>

References

Zwietering MH, Jongenburger I, Rombouts FM, van't Riet K (1990). "Modeling of the Bacterial Growth Curve." *Applied and Environmental Microbiology*, **56**(6), 1875-1881. ISSN 0099-2240.

Examples

```
## Example: Zwietering full model
library(gslnls)
data(growthfull) # simulated data set.
initial_values <- list(Y0 = 0, Ymax = 22, MUmax = 1.7, lag = 5) # define the initial values
fit <- gsl_nls(lnN ~ ZwieteringFM(Time, Y0, Ymax, MUmax, lag),
  data = growthfull,
  start = initial_values
)
summary(fit)
```

Index