

Package ‘biogrowth’

August 23, 2021

Type Package

Title Modelling of Population Growth

Version 0.2.3

Description Modelling of population growth under static and dynamic environmental conditions. Includes functions for model fitting and making prediction under isothermal and dynamic conditions. The methods (algorithms & models) are based on predictive microbiology (See Perez-Rodriguez and Valero (2012, ISBN:978-1-4614-5519-6)).

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Imports deSolve (>= 1.28), tibble (>= 3.0.3), dplyr (>= 0.8.5), FME (>= 1.3.6), MASS (>= 7.3), rlang (>= 0.4.7), purrr (>= 0.3.4), ggplot2 (>= 3.3.2), cowplot (>= 1.0.0), lamW (>= 1.3.0), tidyr (>= 1.0.2), formula.tools (>= 1.7.1)

Suggests knitr, rmarkdown, tidyverse (>= 1.3.0)

VignetteBuilder knitr

Depends R (>= 2.10)

NeedsCompilation no

Author Alberto Garre [aut, cre] (<<https://orcid.org/0000-0002-4404-3550>>),
Jeroen Koomen [aut],
Heidy den Besten [aut],
Marcel Zwietering [aut]

Maintainer Alberto Garre <garre.alberto@gmail.com>

Repository CRAN

Date/Publication 2021-08-23 11:20:02 UTC

R topics documented:

approx_env 3

arabian_tractors	4
calculate_gammas	4
calculate_gammas_secondary	5
check_primary_pars	5
check_secondary_pars	6
check_stochastic_pars	6
conditions_pH_temperature	7
CPM_model	7
dBaranyi	8
distribution_to_logcount	9
DynamicGrowth	10
example_cardinal	11
example_dynamic_growth	12
example_env_conditions	12
extract_primary_pars	13
extract_secondary_pars	13
FitDynamicGrowth	14
FitDynamicGrowthMCMC	16
FitIsoGrowth	18
FitMultipleDynamicGrowth	20
FitMultipleGrowthMCMC	23
FitSecondaryGrowth	25
fit_dynamic_growth	27
fit_isothermal_growth	29
fit_MCMC_growth	30
fit_multiple_growth	32
fit_multiple_growth_MCMC	34
fit_secondary_growth	36
full_Ratkowski	38
get_dyna_residuals	38
get_iso_residuals	39
get_multi_dyna_residuals	39
get_secondary_residuals	40
growth_pH_temperature	41
growth_salmonella	41
is.DynamicGrowth	42
is.FitDynamicGrowth	42
is.FitDynamicGrowthMCMC	43
is.FitIsoGrowth	43
is.FitMultipleDynamicGrowth	44
is.FitMultipleDynamicGrowthMCMC	44
is.FitSecondaryGrowth	45
is.IsothermalGrowth	45
is.MCMCgrowth	46
is.StochasticGrowth	46
IsothermalGrowth	47
iso_Baranyi	48
iso_repGompertz	48

<code>approx_env</code>	3
<code>lambda_to_Q0</code>	49
<code>logistic_model</code>	49
<code>MCMCgrowth</code>	50
<code>multiple_experiments</code>	51
<code>predict_dynamic_growth</code>	52
<code>predict_isothermal_growth</code>	53
<code>predict_MCMC_growth</code>	54
<code>predict_stochastic_growth</code>	56
<code>primary_model_data</code>	58
<code>Q0_to_lambda</code>	58
<code>richards_model</code>	59
<code>secondary_model_data</code>	59
<code>StochasticGrowth</code>	60
<code>TimeDistribution</code>	61
<code>time_to_logcount</code>	62
<code>trilinear_model</code>	63
<code>zwietering_gamma</code>	63
Index	65

<code>approx_env</code>	<i>Generates functions for linear interpolation of environmental conditions</i>
-------------------------	---

Description

Generates functions for linear interpolation of environmental conditions

Usage

```
approx_env(env_conditions)
```

Arguments

`env_conditions` A tibble describing the variation of the environmental conditions through the storage time. Must contain a column named `time` and as many additional columns as environmental factors.

Value

A list of functions that return the value of each environmental condition for some storage time

arabian_tractors	<i>Number of tractors in the Arab World according to the World Bank</i>
------------------	---

Description

A dataset showing the increase in tractors in the Arab World. It was retrieved from <https://data.worldbank.org/indicator/AG.AA>

Usage

```
arabian_tractors
```

Format

A tibble with 40 rows (each corresponding to one year) and 7 columns:

year Year for the recording

tractors Number of tractors

calculate_gammas	<i>Calculates every gamma factor</i>
------------------	--------------------------------------

Description

A helper function for [predict_dynamic_growth](#) that calculates the value of every gamma factor corresponding to some storage time.

Usage

```
calculate_gammas(this_t, env_func, sec_models)
```

Arguments

`this_t` Storage time

`env_func` A list of functions (generated using `approxfun`) that give the value of each environmental function for some storage time.

`sec_models` A nested list describing the secondary models.

Value

A vector of gamma factors (one per environmental factor).

`calculate_gammas_secondary`*Gamma factors for fitting secondary models*

Description

A helper for fitting the secondary gamma models. Calculates the gamma factors corresponding to the models defined and the experimental conditions. In order for it to work, the environmental factors must be named identically in the 3 arguments.

Usage

```
calculate_gammas_secondary(sec_model_names, my_data, secondary_models)
```

Arguments`sec_model_names`

named character vector defining the type of secondary model. Its names correspond to the environmental condition and the values define the corresponding type of secondary model.

`my_data`

Tibble of experimental conditions.

`secondary_models`

A list defining the parameters of the secondary models.

Value

a numeric vector of length `nrow(my_data)` with the gamma factor for each experimental condition.

`check_primary_pars`*Basic check of parameters for primary models*

Description

Checks that: the model name is correct, the right number of model parameters have been defined and that the parameters have the right names

Usage

```
check_primary_pars(model_name, pars)
```

Arguments`model_name`

Model identifier

`pars`

A named list of model parameters

Value

If there is no error, the model function.

check_secondary_pars *Basic checks of secondary parameters*

Description

Checks that the model names are correct, that no parameter is defined twice, that every parameter is defined and that no unknown parameter has been defined. Raises an error if any of these conditions is not met.

Usage

```
check_secondary_pars(
  starting_point,
  known_pars,
  sec_model_names,
  primary_pars = "mu_opt"
)
```

Arguments

`starting_point` Named vector with initial values for the model parameters to estimate from the data. The growth rate under optimum conditions must be named `mu_opt`. The rest must be called `'env_factor'+ '_' + 'parameter'`. For instance, the minimum pH for growth is `'pH_xmin'`.

`known_pars` Named vector of fixed model parameters. Must be named using the same convention as `starting_point`.

`sec_model_names` Named character vector defining the secondary model for each environmental factor.

`primary_pars` Character vector with the parameter names of the primary model.

check_stochastic_pars *Model definition checks for predict_stochastic_growth*

Description

Does several checks of the model parameters. Besides those by `check_primary_pars`, it checks that `corr_matrix` is square, that `pars` and `corr_matrix` have compatible dimensions, and that `pars` has the correct names.

Usage

```
check_stochastic_pars(model_name, pars, corr_matrix)
```

Arguments

model_name	Character describing the primary growth model.
pars	A tibble describing the parameter uncertainty (see details).
corr_matrix	Correlation matrix of the model parameters. Defined in the same order as in pars. An identity matrix by default (uncorrelated parameters).

conditions_pH_temperature

Conditions during a dynamic growth experiment

Description

A dataset to demonstrate the use of fit_dynamic_growth. The observations environmental conditions are described in conditions_pH_temperature.

Usage

```
conditions_pH_temperature
```

Format

A tibble with 4 rows and 3 columns:

time elapsed time

temperature temperature

pH pH

CPM_model

Secondary Cardinal Parameter (CPM) model

Description

Secondary cardinal parameter model as defined by Rosso et al. (1995).

Usage

```
CPM_model(x, xmin, xopt, xmax, n)
```

Arguments

x	Value of the environmental factor.
xmin	Minimum value for growth.
xopt	Optimum value for growth.
xmax	Maximum value for growth.
n	Order of the CPM model.

Value

The corresponding gamma factor.

dBaranyi

Baranyi growth model

Description

Microbial growth model as defined in Baranyi and Roberts (1994). It has been implemented according to the requirements of [ode](#). For consistency the function for isothermal growth, calculations are done considering mu is in log10 scale. In other words, it is multiplied by ln(10).

Usage

```
dBaranyi(time, state, pars, env_func, sec_models)
```

Arguments

time	numeric vector (length 1) of storage time
state	named numeric vector with two components: Q and N
pars	named numeric vector of model parameters (Nmax and mu_opt)
env_func	named list of functions returning the values of the environmental conditions for time (t)
sec_models	named list of parameters of the secondary model

Value

A numeric vector of two components according to the requirements of [ode](#).

`distribution_to_logcount`*Distribution of times to reach a certain microbial count*

Description

Returns the probability distribution of the storage time required for the microbial count to reach `log_count` according to the predictions of a stochastic model. Calculations are done using linear interpolation of the individual model predictions.

Usage

```
distribution_to_logcount(model, log_count)
```

Arguments

<code>model</code>	An instance of <code>StochasticGrowth</code> or <code>MCMCgrowth</code> .
<code>log_count</code>	The target microbial count.

Value

An instance of [TimeDistribution](#).

Examples

```
## We need an instance of StochasticGrowth

my_model <- "modGompertz"
my_times <- seq(0, 30, length = 100)
n_sims <- 3000

library(tibble)

pars <- tribble(
  ~par, ~mean, ~sd, ~scale,
  "logN0", 0, .2, "original",
  "mu", 2, .3, "sqrt",
  "lambda", 4, .4, "sqrt",
  "C", 6, .5, "original"
)

stoc_growth <- predict_stochastic_growth(my_model, my_times, n_sims, pars)

## We can now call the function

time_distrib <- distribution_to_logcount(stoc_growth, 4)

## And plot the results
```

```
plot(time_distrib)
```

DynamicGrowth

DynamicGrowth class

Description

The DynamicGrowth class contains the results of a growth prediction under dynamic conditions. Its constructor is [predict_dynamic_growth](#).

A subclass of list with items:

- simulation: A tibble with the model prediction
- gammas: A tibble with the value of each gamma factor for each value of times.
- env_conditions: A list of functions interpolating the environmental conditions.
- primary_pars: A list with the model parameters of the primary model.
- sec_models: A nested list defining the secondary models.

Usage

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

```
## S3 method for class 'DynamicGrowth'  
plot(  
  x,  
  y = NULL,  
  ...,  
  add_factor = NULL,  
  ylims = NULL,  
  label_y1 = "logN",  
  label_y2 = add_factor,  
  line_col = "black",  
  line_size = 1,  
  line_type = "solid",  
  line_col2 = "black",  
  line_size2 = 1,  
  line_type2 = "dashed"  
)
```

Arguments

x	The object of class DynamicGrowth to plot.
...	additional arguments passed to plot.
y	ignored
add_factor	whether to plot also one environmental factor. If NULL (default), no environmental factor is plotted. If set to one character string that matches one entry of x\$env_conditions, that condition is plotted in the secondary axis
ylims	A two dimensional vector with the limits of the primary y-axis.
label_y1	Label of the primary y-axis.
label_y2	Label of the secondary y-axis.
line_col	Aesthetic parameter to change the colour of the line geom in the plot, see: geom_line
line_size	Aesthetic parameter to change the thickness of the line geom in the plot, see: geom_line
line_type	Aesthetic parameter to change the type of the line geom in the plot, takes numbers (1-6) or strings ("solid") see: geom_line
line_col2	Same as lin_col, but for the environmental factor.
line_size2	Same as line_size, but for the environmental factor.
line_type2	Same as lin_type, but for the environmental factor.

Functions

- `print.DynamicGrowth`: print of the model
- `plot.DynamicGrowth`: predicted growth curve under dynamic conditions.

example_cardinal	<i>Growth rates obtained for several growth experiments</i>
------------------	---

Description

An example dataset illustrating the requirements of the [fit_secondary_growth](#) function.

Usage

```
example_cardinal
```

Format

A data frame with 64 rows and 3 variables:

temperature storage temperature (°C)
pH pH of the media
mu specific growth rate (log10 CFU/h)

example_dynamic_growth

Microbial growth under dynamic conditions

Description

An example dataset illustrating the requirements of the [fit_dynamic_growth](#) function.

Usage

example_dynamic_growth

Format

A data frame with 30 rows and 2 variables:

time elapsed time (h)

logN log population size (log10 CFU)

example_env_conditions

Environmental conditions during a dynamic experiment

Description

An example dataset illustrating the requirements of the [fit_dynamic_growth](#) function.

Usage

example_env_conditions

Format

A data frame with 3 rows and 3 variables:

time elapsed time (h)

temperature storage temperature (°C)

aw water activity

extract_primary_pars *A helper to build the primary models*

Description

Most of the functions for fitting mix in the vectors parameters for the primary and secondary models, but the functions for making predictions need that they are separated. This one extracts the parameters of the primary model.

Usage

```
extract_primary_pars(this_p, known_pars)
```

Arguments

`this_p` A named vector of model parameters (usually, the ones fitted).
`known_pars` Another named vector of model parameters (usually the known ones).

Value

A list with the parameters of the primary model

extract_secondary_pars
A helper to build the secondary models

Description

Most of the functions for fitting mix in the vectors parameters for the primary and secondary models, but the functions for making predictions need that they are separated. This one extracts the parameters of the secondary model.

Usage

```
extract_secondary_pars(this_p, known_pars, sec_model_names)
```

Arguments

`this_p` A named vector of model parameters (usually, the ones fitted).
`known_pars` Another named vector of model parameters (usually the known ones).
`sec_model_names` A named character vector defining for each environmental factor (vector names) the type of secondary model (vector values).

Value

A nested list defining the secondary models.

FitDynamicGrowth *FitDynamicGrowth class*

Description

The FitDynamicGrowth class contains a model fitted based on growth data under dynamic conditions. Its constructor is [fit_dynamic_growth](#).

It is a subclass of list with the items:

- fit_results: the object returned by modFit.
- best_prediction: the model prediction for the fitted parameters.
- env_conditions: environmental conditions for the fit.
- data: data used for the fit.
- starting: starting values for model fitting
- known: parameter values set as known.
- sec_models: a named vector with the secondary model for each environmental factor

Usage

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

## S3 method for class 'FitDynamicGrowth'
plot(
  x,
  y = NULL,
  ...,
  add_factor = NULL,
  ylims = NULL,
  label_y1 = "logN",
  label_y2 = add_factor,
  line_col = "black",
  line_size = 1,
  line_type = 1,
  point_col = "black",
  point_size = 3,
  point_shape = 16,
  line_col2 = "black",
  line_size2 = 1,
  line_type2 = "dashed"
)

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

```

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

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

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

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

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

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

```

Arguments

x	The object of class <code>FitDynamicGrowth</code> to plot.
...	ignored
y	ignored
add_factor	whether to plot also one environmental factor. If <code>NULL</code> (default), no environmental factor is plotted. If set to one character string that matches one entry of <code>x\$env_conditions</code> , that condition is plotted in the secondary axis
ylims	A two dimensional vector with the limits of the primary y-axis.
label_y1	Label of the primary y-axis.
label_y2	Label of the secondary y-axis.
line_col	Aesthetic parameter to change the colour of the line geom in the plot, see: geom_line
line_size	Aesthetic parameter to change the thickness of the line geom in the plot, see: geom_line
line_type	Aesthetic parameter to change the type of the line geom in the plot, takes numbers (1-6) or strings ("solid") see: geom_line
point_col	Aesthetic parameter to change the colour of the point geom, see: geom_point
point_size	Aesthetic parameter to change the size of the point geom, see: geom_point
point_shape	Aesthetic parameter to change the shape of the point geom, see: geom_point
line_col2	Same as <code>line_col</code> , but for the environmental factor.
line_size2	Same as <code>line_size</code> , but for the environmental factor.
line_type2	Same as <code>line_type</code> , but for the environmental factor.
object	an instance of <code>FitDynamicGrowth</code> .
times	A numeric vector with the time points for the simulations. <code>NULL</code> by default (using the same time points as those for the simulation).
newdata	a tibble describing the environmental conditions (as <code>env_conditions</code>) in predict_dynamic_growth . If <code>NULL</code> (default), uses the same conditions as those for fitting.

Functions

- `print.FitDynamicGrowth`: comparison between the fitted model and the data.
- `plot.FitDynamicGrowth`: comparison between the fitted model and the data.
- `summary.FitDynamicGrowth`: statistical summary of the fit.
- `residuals.FitDynamicGrowth`: residuals of the model.
- `coef.FitDynamicGrowth`: vector of fitted parameters.
- `vcov.FitDynamicGrowth`: (unscaled) variance-covariance matrix of the model, calculated as $1/(0.5 * \text{Hessian})$
- `deviance.FitDynamicGrowth`: deviance of the model.
- `fitted.FitDynamicGrowth`: fitted values.
- `predict.FitDynamicGrowth`: model predictions.

`FitDynamicGrowthMCMC` *FitDynamicGrowthMCMC class*

Description

The `FitDynamicGrowthMCMC` a model fitted based on a dynamic growth experiment using an MCMC algorithm. Its constructor is `fit_MCMC_growth`.

It is a subclass of list with the items:

- `fit_results`: the object returned by `modMCMC`.
- `best_prediction`: the model prediction for the fitted parameters.
- `env_conditions`: environmental conditions for the fit.
- `data`: data used for the fit.
- `starting`: starting values for model fitting
- `known`: parameter values set as known.
- `sec_models`: a named vector with the secondary model for each environmental factor

Usage

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

## S3 method for class 'FitDynamicGrowthMCMC'
plot(
  x,
  y = NULL,
  ...,
  add_factor = NULL,
  ylims = NULL,
  label_y1 = "logN",
```



```

    label_y2 = add_factor,
    line_col = "black",
    line_size = 1,
    line_type = 1,
    point_col = "black",
    point_size = 3,
    point_shape = 16,
    line_col2 = "black",
    line_size2 = 1,
    line_type2 = "dashed"
  )

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

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

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

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

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

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

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

```

Arguments

x	The object of class FitDynamicGrowthMCMC to plot.
...	ignored
y	ignored
add_factor	whether to plot also one environmental factor. If NULL (default), no environmental factor is plotted. If set to one character string that matches one entry of x\$env_conditions, that condition is plotted in the secondary axis
ylims	A two dimensional vector with the limits of the primary y-axis.
label_y1	Label of the primary y-axis.
label_y2	Label of the secondary y-axis.
line_col	Aesthetic parameter to change the colour of the line geom in the plot, see: geom_line

<code>line_size</code>	Aesthetic parameter to change the thickness of the line geom in the plot, see: geom_line
<code>line_type</code>	Aesthetic parameter to change the type of the line geom in the plot, takes numbers (1-6) or strings ("solid") see: geom_line
<code>point_col</code>	Aesthetic parameter to change the colour of the point geom, see: geom_point
<code>point_size</code>	Aesthetic parameter to change the size of the point geom, see: geom_point
<code>point_shape</code>	Aesthetic parameter to change the shape of the point geom, see: geom_point
<code>line_col2</code>	Same as <code>lin_col</code> , but for the environmental factor.
<code>line_size2</code>	Same as <code>line_size</code> , but for the environmental factor.
<code>line_type2</code>	Same as <code>lin_type</code> , but for the environmental factor.
<code>object</code>	an instance of <code>FitDynamicGrowthMCMC</code> .
<code>times</code>	A numeric vector with the time points for the simulations. NULL by default (using the same time points as those for the simulation).
<code>newdata</code>	a tibble describing the environmental conditions (as <code>env_conditions</code>) in predict_dynamic_growth . If NULL (default), uses the same conditions as those for fitting.

Functions

- `print.FitDynamicGrowthMCMC`: print of the model
- `plot.FitDynamicGrowthMCMC`: compares the model fitted against the data.
- `summary.FitDynamicGrowthMCMC`: statistical summary of the fit.
- `residuals.FitDynamicGrowthMCMC`: model residuals.
- `coef.FitDynamicGrowthMCMC`: vector of fitted model parameters.
- `vcov.FitDynamicGrowthMCMC`: variance-covariance matrix of the model, estimated as the variance of the samples from the Markov chain.
- `deviance.FitDynamicGrowthMCMC`: deviance of the model, calculated as the sum of squared residuals for the parameter values resulting in the best fit.
- `fitted.FitDynamicGrowthMCMC`: vector of fitted values.
- `predict.FitDynamicGrowthMCMC`: vector of model predictions.

FitIsoGrowth

FitIsoGrowth class

Description

The `FitIsoGrowth` class contains a growth model fitted to data under static conditions. Its constructor is [fit_isothermal_growth](#).

It is a subclass of `list` with the items:

- `data`: data used for model fitting
- `model`: name of the primary inactivation model
- `starting_point`: initial value of the model parameters
- `known`: fixed model parameters
- `fit`: object returned by [modFit](#)
- `best_prediction`: model prediction for the model fitted.

Usage

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

## S3 method for class 'FitIsoGrowth'
plot(
  x,
  y = NULL,
  ...,
  line_col = "black",
  line_size = 1,
  line_type = 1,
  point_col = "black",
  point_size = 3,
  point_shape = 16
)

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

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

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

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

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

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

## S3 method for class 'FitIsoGrowth'
predict(object, times = NULL, ...)
```

Arguments

x	The object of class FitIsoGrowth to plot.
...	ignored
y	ignored
line_col	Aesthetic parameter to change the colour of the line geom in the plot, see: geom_line
line_size	Aesthetic parameter to change the thickness of the line geom in the plot, see: geom_line

line_type	Aesthetic parameter to change the type of the line geom in the plot, takes numbers (1-6) or strings ("solid") see: geom_line
point_col	Aesthetic parameter to change the colour of the point geom, see: geom_point
point_size	Aesthetic parameter to change the size of the point geom, see: geom_point
point_shape	Aesthetic parameter to change the shape of the point geom, see: geom_point
object	an instance of FitIsoGrowth
times	numeric vector describing the time points for the prediction. If NULL (default), uses the same points as those used for fitting.

Functions

- `print.FitIsoGrowth`: print of the model
- `plot.FitIsoGrowth`: compares the fitted model against the data.
- `summary.FitIsoGrowth`: statistical summary of the fit.
- `residuals.FitIsoGrowth`: vector of model residuals.
- `coef.FitIsoGrowth`: vector of fitted model parameters.
- `vcov.FitIsoGrowth`: variance-covariance matrix of the model, estimated as $1/(0.5*\text{Hessian})$
- `deviance.FitIsoGrowth`: deviance of the model.
- `fitted.FitIsoGrowth`: vector of fitted values.
- `predict.FitIsoGrowth`: vector of model predictions.

FitMultipleDynamicGrowth

FitMultipleDynamicGrowth class

Description

The `FitMultipleDynamicGrowth` class contains a model fitted to a set of experiments gathered under dynamic conditions. Its constructor is [fit_multiple_growth](#).

It is a subclass of list with the items:

- `fit_results`: the object returned by `modFit`.
- `best_prediction`: a list with the models predictions for each condition.
- `data`: a list with the data used for the fit.
- `starting`: starting values for model fitting
- `known`: parameter values set as known.
- `sec_models`: a named vector with the secondary model for each environmental factor.

Usage

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

## S3 method for class 'FitMultipleDynamicGrowth'
plot(
  x,
  y = NULL,
  ...,
  add_factor = NULL,
  ylims = NULL,
  label_x = "time",
  label_y1 = "logN",
  label_y2 = add_factor,
  line_col = "black",
  line_size = 1,
  line_type = "solid",
  line_col2 = "black",
  line_size2 = 1,
  line_type2 = "dashed",
  point_size = 3,
  point_shape = 16,
  subplot_labels = "AUTO"
)

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

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

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

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

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

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

## S3 method for class 'FitMultipleDynamicGrowth'
predict(object, times = NULL, newdata = NULL, ...)
```

Arguments

<code>x</code>	an instance of <code>FitMultipleDynamicGrowth</code> .
<code>...</code>	ignored
<code>y</code>	ignored
<code>add_factor</code>	whether to plot also one environmental factor. If NULL (default), no environmental factor is plotted. If set to one character string that matches one entry of <code>x\$env_conditions</code> , that condition is plotted in the secondary axis
<code>ylimits</code>	A two dimensional vector with the limits of the primary y-axis.
<code>label_x</code>	label of the x-axis
<code>label_y1</code>	Label of the primary y-axis.
<code>label_y2</code>	Label of the secondary y-axis.
<code>line_col</code>	Aesthetic parameter to change the colour of the line geom in the plot, see: geom_line
<code>line_size</code>	Aesthetic parameter to change the thickness of the line geom in the plot, see: geom_line
<code>line_type</code>	Aesthetic parameter to change the type of the line geom in the plot, takes numbers (1-6) or strings ("solid") see: geom_line
<code>line_col2</code>	Same as <code>lin_col</code> , but for the environmental factor.
<code>line_size2</code>	Same as <code>line_size</code> , but for the environmental factor.
<code>line_type2</code>	Same as <code>lin_type</code> , but for the environmental factor.
<code>point_size</code>	Size of the data points
<code>point_shape</code>	shape of the data points
<code>subplot_labels</code>	labels of the subplots according to <code>plot_grid</code> .
<code>object</code>	Instance of <code>FitMultipleDynamicGrowth</code> .
<code>times</code>	A numeric vector with the time points for the simulations. NULL by default (using the same time points as those for the simulation).
<code>newdata</code>	a tibble describing the environmental conditions (as <code>env_conditions</code>) in fit_multiple_growth . If NULL (default), uses the same conditions as those for fitting.

Functions

- `print.FitMultipleDynamicGrowth`: print of the model
- `plot.FitMultipleDynamicGrowth`: comparison between the fitted model and the experimental data.
- `summary.FitMultipleDynamicGrowth`: statistical summary of the fit.
- `residuals.FitMultipleDynamicGrowth`: calculates the model residuals. Returns a tibble with 4 columns: time (storage time), logN (observed count), exp (name of the experiment) and res (residual).
- `coef.FitMultipleDynamicGrowth`: vector of fitted parameters.
- `vcov.FitMultipleDynamicGrowth`: (unscaled) variance-covariance matrix, estimated as $1/(0.5 * \text{Hessian})$.

- `deviance.FitMultipleDynamicGrowth`: deviance of the model.
- `fitted.FitMultipleDynamicGrowth`: fitted values. They are returned as a tibble with 3 columns: time (storage time), exp (experiment identifier) and fitted (fitted value).
- `predict.FitMultipleDynamicGrowth`: model predictions. They are returned as a tibble with 3 columns: time (storage time), logN (observed count), and exp (name of the experiment).

FitMultipleGrowthMCMC *FitMultipleGrowthMCMC class*

Description

The `FitMultipleGrowthMCMC` class contains a model fitted to a set of dynamic experiments using an MCMC algorithm. Its constructor is `fit_multiple_growth_MCMC`.

It is a subclass of list with the items:

- `fit_results`: the object returned by `modFit`.
- `best_prediction`: a list with the models predictions for each condition.
- `data`: a list with the data used for the fit.
- `starting`: starting values for model fitting
- `known`: parameter values set as known.
- `sec_models`: a named vector with the secondary model for each environmental factor.

Usage

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

```
## S3 method for class 'FitMultipleGrowthMCMC'
plot(
  x,
  y = NULL,
  ...,
  add_factor = NULL,
  ylims = NULL,
  label_x = "time",
  label_y1 = "logN",
  label_y2 = add_factor,
  line_col = "black",
  line_size = 1,
  line_type = "solid",
  line_col2 = "black",
  line_size2 = 1,
  line_type2 = "dashed",
  point_size = 3,
  point_shape = 16,
```

```

    subplot_labels = "AUTO"
)

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

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

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

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

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

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

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

```

Arguments

x	an instance of FitMultipleGrowthMCMC.
...	ignored
y	ignored
add_factor	whether to plot also one environmental factor. If NULL (default), no environmental factor is plotted. If set to one character string that matches one entry of x\$env_conditions, that condition is plotted in the secondary axis
ylims	A two dimensional vector with the limits of the primary y-axis.
label_x	label of the x-axis
label_y1	Label of the primary y-axis.
label_y2	Label of the secondary y-axis.
line_col	Aesthetic parameter to change the colour of the line geom in the plot, see: geom_line
line_size	Aesthetic parameter to change the thickness of the line geom in the plot, see: geom_line
line_type	Aesthetic parameter to change the type of the line geom in the plot, takes numbers (1-6) or strings ("solid") see: geom_line
line_col2	Same as lin_col, but for the environmental factor.
line_size2	Same as line_size, but for the environmental factor.
line_type2	Same as lin_type, but for the environmental factor.

point_size	Size of the data points
point_shape	shape of the data points
subplot_labels	labels of the subplots according to plot_grid.
object	Instance of FitMultipleGrowthMCMC.
times	A numeric vector with the time points for the simulations. NULL by default (using the same time points as those for the simulation).
newdata	a tibble describing the environmental conditions (as env_conditions) in fit_multiple_growth . If NULL (default), uses the same conditions as those for fitting.

Functions

- `print.FitMultipleGrowthMCMC`: print of the model
- `plot.FitMultipleGrowthMCMC`: comparison between the model fitted and the data.
- `summary.FitMultipleGrowthMCMC`: statistical summary of the fit.
- `residuals.FitMultipleGrowthMCMC`: model residuals. They are returned as a tibble with 4 columns: time (storage time), logN (observed count), exp (name of the experiment) and res (residual).
- `coef.FitMultipleGrowthMCMC`: vector of fitted model parameters.
- `vcov.FitMultipleGrowthMCMC`: variance-covariance matrix of the model, estimated as the variance of the samples from the Markov chain.
- `deviance.FitMultipleGrowthMCMC`: deviance of the model, calculated as the sum of squared residuals of the prediction with the lowest standard error.
- `fitted.FitMultipleGrowthMCMC`: fitted values of the model. They are returned as a tibble with 3 columns: time (storage time), exp (experiment identifier) and fitted (fitted value).
- `predict.FitMultipleGrowthMCMC`: model predictions. They are returned as a tibble with 3 columns: time (storage time), logN (observed count), and exp (name of the experiment).

FitSecondaryGrowth *FitSecondaryGrowth class*

Description

The `FitSecondaryGrowth` class contains a model fitted to a set of growth rates gathered under a variety of static conditions. Its constructor is [fit_secondary_growth](#).

It is a subclass of list with the items:

- `fit_results`: object returned by [modFit](#).
- `secondary_model`: secondary model fitted to the data.
- `mu_opt_fit`: estimated growth rate under optimum conditions.
- `data`: data used for the fit.
- `transformation`: type of transformation of mu for the fit.

Usage

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

## S3 method for class 'FitSecondaryGrowth'
plot(x, y = NULL, ..., which = 1, add_trend = FALSE)

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

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

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

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

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

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

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

Arguments

x	An instance of FitSecondaryGrowth.
...	ignored
y	ignored.
which	A numeric with the type of plot. 1 for obs versus predicted (default), 2 for gamma curve
add_trend	Whether to add a trend line (only for which=2)
object	an instance of FitSecondaryGrowth.
newdata	A tibble describing the environmental conditions as in fit_secondary_growth . If NULL, it uses the same conditions as for model fitting (default).

Functions

- `print.FitSecondaryGrowth`: print of the model
- `plot.FitSecondaryGrowth`: plots to evaluate the goodness of the fit.
- `summary.FitSecondaryGrowth`: statistical summary of the fit.
- `residuals.FitSecondaryGrowth`: vector of model residuals.

- `coef.FitSecondaryGrowth`: vector of fitted model parameters.
- `vcov.FitSecondaryGrowth`: variance-covariance matrix of the model, estimated as $1/(0.5 * \text{Hessian})$
- `deviance.FitSecondaryGrowth`: deviance of the model.
- `fitted.FitSecondaryGrowth`: vector of fitted values.
The fitted values are returned in the same scale as the one used for the fitting (sqrt, log or none).
- `predict.FitSecondaryGrowth`: vector of model predictions.

fit_dynamic_growth *Fit dynamic growth models*

Description

Fits a growth model to a data obtained under dynamic conditions using the one-step approach (non-linear regression).

Usage

```
fit_dynamic_growth(
  fit_data,
  env_conditions,
  starting_point,
  known_pars,
  sec_model_names,
  ...,
  check = TRUE,
  formula = logN ~ time
)
```

Arguments

- | | |
|-----------------------------|--|
| <code>fit_data</code> | Tibble with the data to use for model fit. It must contain a column with the elapsed time (named "time" by default) and another one with the decimal logarithm of the observed population size (named "logN" by default). Different column names can be specified using the "formula" argument. |
| <code>env_conditions</code> | Tibble with the (dynamic) environmental conditions during the experiment. It must have one column with the elapsed time (named "time" by default) and as many columns as required with the environmental conditions. A different column name can be specified using the "formula" argument, although it must be the same one as in "fit_data". Note that only those defined in "sec_model_names" will be considered for the model fit. |
| <code>starting_point</code> | A named vector of starting values for the model parameters. Parameters for the primary model must be named in the usual way. Parameters for the secondary model are named as <code>env_factor+'_'+parameter</code> . For instance, the maximum growth temperature shall be named <code>'temperature_xmax'</code> . |

known_pars	A named vectors of known model parameters (i.e. not fitted). They must be named using the same convention as for starting_point.
sec_model_names	A named character vector defining the secondary model for each environmental factor. The names define the factor and the value the type of model. Names must match columns in fit_data and env_conditions.
...	Additional arguments passed to modFit.
check	Whether to check model parameters (TRUE by default).
formula	an object of class "formula" describing the x and y variables. logN ~ time as a default.

Value

An instance of `FitDynamicGrowth`.

Examples

```
## We use the datasets included in the package

data("example_dynamic_growth")
data("example_env_conditions")

## Define the secondary models

sec_model_names <- c(temperature = "CPM", aw= "CPM")

## Any model parameter can be fixed

known_pars <- list(Nmax = 1e4, # Primary model
  N0 = 1e0, Q0 = 1e-3, # Initial values of the primary model
  mu_opt = 4, # mu_opt of the gamma model
  temperature_n = 1, # Secondary model for temperature
  aw_xmax = 1, aw_xmin = .9, aw_n = 1 # Secondary model for water activity
)

## The remaining parameters need initial values

my_start <- list(temperature_xmin = 25, temperature_xopt = 35,
  temperature_xmax = 40, aw_xopt = .95)

## We can now call the fitting function

my_dyna_fit <- fit_dynamic_growth(example_dynamic_growth, example_env_conditions,
  my_start, known_pars, sec_model_names)

summary(my_dyna_fit)

## We can compare the data and the fitted curve

plot(my_dyna_fit)
```

```
## We can plot any environmental condition using add_factor

plot(my_dyna_fit, add_factor = "aw",
     label_y1 = "Log count (log CFU/ml)",
     label_y2 = "Water activity")
```

fit_isothermal_growth *Fit primary growth models*

Description

Fits a primary growth model to data obtained under static environmental conditions.

Usage

```
fit_isothermal_growth(
  fit_data,
  model_name,
  starting_point,
  known_pars,
  ...,
  check = TRUE,
  formula = logN ~ time
)
```

Arguments

fit_data	Tibble of data for the fit. It must have two columns, one with the elapsed time (time by default) and another one with the decimal logarithm of the population size (logN by default). Different column names can be defined using the formula argument.
model_name	Character defining the primary growth model
starting_point	Named vector of initial values for the model parameters.
known_pars	Named vector of known model parameters (not fitted).
...	Additional arguments passed to <code>modFit</code> .
check	Whether to do some basic checks (TRUE by default).
formula	an object of class "formula" describing the x and y variables. logN ~ time as a default.

Value

An instance of `FitIsoGrowth`.

Examples

```
## Some dummy data

library(tibble)

my_data <- tibble(time = c(0, 25, 50, 75, 100),
                  logN = c(2, 2.5, 7, 8, 8))

## Choose the model

my_model <- "Baranyi"

## Initial values for the model parameters

start = c(logNmax = 8, lambda = 25, logN0 = 2)

## Any model parameter can be fixed

known <- c(mu = .2)

## Now, we can call the function

static_fit <- fit_isothermal_growth(my_data, my_model, start, known)

summary(static_fit)

## We can plot the fitted model against the observations

plot(static_fit)
```

fit_MCMC_growth

Fit growth models using MCMC

Description

Fits a growth model to a data obtained under dynamic conditions using the one-step approach (MCMC algorithm).

Usage

```
fit_MCMC_growth(
  fit_data,
  env_conditions,
  starting_point,
  known_pars,
  sec_model_names,
  niter,
```

```

    ...,
    check = TRUE,
    formula = logN ~ time
  )

```

Arguments

fit_data	Tibble with the data to use for model fit. It must contain a column with the elapsed time (named "time" by default) and another one with the decimal logarithm of the observed population size (named "logN" by default). Different column names can be specified using the "formula" argument.
env_conditions	Tibble with the (dynamic) environmental conditions during the experiment. It must have one column with the elapsed time (named "time" by default) and as many columns as required with the environmental conditions. A different column name can be specified using the "formula" argument, although it must be the same one as in "fit_data". Note that only those defined in "sec_model_names" will be considered for the model fit.
starting_point	A named vector of starting values for the model parameters. Parameters for the primary model must be named in the usual way. Parameters for the secondary model are named as env_factor+'_'+parameter. For instance, the maximum growth temperature shall be named 'temperature_xmax'.
known_pars	A named vectors of known model parameters (i.e. not fitted). They must be named using the same convention as for starting_point.
sec_model_names	A named character vector defining the secondary model for each environmental factor. The names define the factor and the value the type of model. Names must match columns in fit_data and env_conditions.
niter	number of iterations of the MCMC algorithm.
...	Additional arguments passed to modFit.
check	Whether to check model parameters (TRUE by default).
formula	an object of class "formula" describing the x and y variables. logN ~ time as a default.

Value

An instance of `FitDynamicGrowthMCMC`.

Examples

```

## We use the example data included in the package

data("example_dynamic_growth")
data("example_env_conditions")

## Definition of the secondary models
sec_model_names <- c(temperature = "CPM", aw= "CPM")

```

```

## Any model parameter can be fixed
known_pars <- list(Nmax = 1e4, # Primary model
  N0 = 1e0, Q0 = 1e-3, # Initial values of the primary model
  mu_opt = 4, # mu_opt of the gamma model
  temperature_n = 1, # Secondary model for temperature
  aw_xmax = 1, aw_xmin = .9, aw_n = 1 # Secondary model for water activity
)

## We need starting values for the remaining parameters

my_start <- list(temperature_xmin = 25, temperature_xopt = 35,
  temperature_xmax = 40,
  aw_xopt = .95)

## We can now call the fitting function

set.seed(12124) # Setting seed for repeatability

my_MCMC_fit <- fit_MCMC_growth(example_dynamic_growth, example_env_conditions,
  my_start, known_pars, sec_model_names, niter = 3000)

## Always check the MCMC chain!!

plot(my_MCMC_fit$fit_results)

## We can compare data against fitted curve

plot(my_MCMC_fit)

## Any environmental factor can be included using add_factor

plot(my_MCMC_fit, add_factor = "temperature",
  label_y1 = "Count (log CFU/ml)", label_y2 = "Temperature (C)")

```

fit_multiple_growth *Fitting growth models to multiple dynamic experiments*

Description

This functions enables to fit a growth model using a dataset comprised of several experiments with potentially different dynamic experimental conditions. Note that the definition of secondary models must comply with the ‘secondary_model_data’ function.

Usage

```

fit_multiple_growth(
  starting_point,

```



```

    experiment_data,
    known_pars,
    sec_model_names,
    ...,
    check = TRUE,
    formula = logN ~ time
  )

```

Arguments

starting_point a named vector of starting values for the model parameters.

experiment_data a nested list with the experimental data. Each entry describes one experiment as a list with two elements: data and conditions. data is a tibble with a column giving the elapsed time (named "time" by default) and another one with the decimal logarithm of the population size (named "logN" by default). conditions is a tibble with one column giving the elapsed time (using the same name as data) and as many additional columns as environmental factors. The default column names can be changed with the formula argument.

known_pars named vector of known model parameters

sec_model_names named character vector with names the environmental conditions and values the secondary model (see secondary_model_data).

... additional arguments for `modFit`.

check Whether to check the validity of the models. TRUE by default.

formula an object of class "formula" describing the x and y variables. `logN ~ time` as a default.

Value

An instance of `FitMultipleDynamicGrowth`.

Examples

```

## We will use the multiple_experiments data set
data("multiple_experiments")

## For each environmental factor, we need to defined a model
sec_names <- c(temperature = "CPM", pH = "CPM")

## Any model parameter can be fixed
known <- list(Nmax = 1e8, N0 = 1e0, Q0 = 1e-3,
             temperature_n = 2, temperature_xmin = 20, temperature_xmax = 35,
             pH_n = 2, pH_xmin = 5.5, pH_xmax = 7.5, pH_xopt = 6.5)

```

```
## The rest require starting values for model fitting
start <- list(mu_opt = .8, temperature_xopt = 30)

## We can now call the fitting function

global_fit <- fit_multiple_growth(start, multiple_experiments, known, sec_names)

## Parameter estimates can be retrieved with summary

summary(global_fit)

## We can compare fitted model against observations

plot(global_fit)

## Any single environmental factor can be added to the plot using add_factor

plot(global_fit, add_factor = "temperature")
```

fit_multiple_growth_MCMC

Fitting growth models to multiple dynamic experiments using MCMC

Description

This functions enables to fit a growth model using a dataset comprised of several experiments with potentially different dynamic experimental conditions.

Usage

```
fit_multiple_growth_MCMC(
  starting_point,
  experiment_data,
  known_pars,
  sec_model_names,
  niter,
  ...,
  check = TRUE,
  formula = logN ~ time
)
```

Arguments

`starting_point` a named vector of starting values for the model parameters.

experiment_data	a nested list with the experimental data. Each entry describes one experiment as a list with two elements: data and conditions. data is a tibble with a column giving the elapsed time (named "time" by default) and another one with the decimal logarithm of the population size (named "logN" by default). conditions is a tibble with one column giving the elapsed time (using the same name as data) and as many additional columns as environmental factors. The default column names can be changed with the formula argument.
known_pars	named vector of known model parameters
sec_model_names	named character vector with names the environmental conditions and values the secondary model (see secondary_model_data).
niter	number of samples of the MCMC algorithm.
...	additional arguments for modMCMC (e.g. upper and lower bounds).
check	Whether to check the validity of the models. TRUE by default.
formula	an object of class "formula" describing the x and y variables. $\log N \sim \text{time}$ as a default.

Value

An instance of `FitMultipleGrowthMCMC`.

Examples

```
## We will use the multiple_experiments data set
data("multiple_experiments")

## For each environmental factor, we need to defined a model
sec_names <- c(temperature = "CPM", pH = "CPM")

## Any model parameter can be fixed
known <- list(Nmax = 1e8, N0 = 1e0, Q0 = 1e-3,
  temperature_n = 2, temperature_xmin = 20, temperature_xmax = 35,
  pH_n = 2, pH_xmin = 5.5, pH_xmax = 7.5, pH_xopt = 6.5)

## The rest require starting values for model fitting
start <- list(mu_opt = .8, temperature_xopt = 30)

## We can now call the fitting function

set.seed(12412)
global_MCMC <- fit_multiple_growth_MCMC(start, multiple_experiments, known, sec_names, niter = 1000,
  lower = c(.2, 29), # lower limits of the model parameters
  upper = c(.8, 34)) # upper limits of the model parameters
```

```
## Parameter estimates can be retrieved with summary
summary(global_MCMC)

## We can compare fitted model against observations
plot(global_MCMC)

## Any single environmental factor can be added to the plot using add_factor
plot(global_MCMC, add_factor = "temperature")
```

fit_secondary_growth *Fit secondary growth models*

Description

Fits a secondary growth model to a set of growth rates obtained experimentally. Modelling is done according to the gamma concept proposed by Zwietering (1992) and cardinal parameter models.

Usage

```
fit_secondary_growth(
  fit_data,
  starting_point,
  known_pars,
  sec_model_names,
  transformation = "sq",
  ...,
  check = TRUE,
  formula = mu ~ .
)
```

Arguments

fit_data	Tibble with the data used for the fit. It must have one column with the observed growth rate (named mu by default; can be changed using the "formula" argument) and as many columns as needed with the environmental factors.
starting_point	Named vector with initial values for the model parameters to estimate from the data. The growth rate under optimum conditions must be named mu_opt. The rest must be called 'env_factor'+ '_'+'parameter'. For instance, the minimum pH for growth is 'pH_xmin'.
known_pars	Named vector of fixed model parameters. Must be named using the same convention as starting_point.

sec_model_names	Named character vector defining the secondary model for each environmental factor.
transformation	Character defining the transformation of mu for model fitting. One of sq (square root; default), log (log-transform) or none (no transformation).
...	Additional arguments passed to <code>modFit</code> .
check	Whether to do some basic checks (TRUE by default).
formula	an object of class "formula" describing the y variable. The right hand side must be ".". By default $\mu \sim \dots$

Value

An instance of `FitSecondaryGrowth`.

Examples

```
## We use the data included in the package
data("example_cardinal")

## Define the models to fit
sec_model_names <- c(temperature = "Zwietering", pH = "CPM")

## Any model parameter can be fixed
known_pars <- list(mu_opt = 1.2, temperature_n = 1,
  pH_n = 2, pH_xmax = 6.8, pH_xmin = 5.2)

## Initial values must be given for every other parameter
my_start <- list(temperature_xmin = 5, temperature_xopt = 35,
  pH_xopt = 6.5)

## We can now call the fitting function
fit_cardinal <- fit_secondary_growth(example_cardinal, my_start, known_pars, sec_model_names)

## With summary, we can look at the parameter estimates
summary(fit_cardinal)

## The plot function compares predictions against observations
plot(fit_cardinal)

## Passing which = 2, generates a different kind of plot
plot(fit_cardinal, which = 2)
plot(fit_cardinal, which = 2, add_trend = TRUE)
```

full_Ratkowski	<i>Full Ratkowsky model</i>
----------------	-----------------------------

Description

Gamma model adapted from the one by Ratkowsky et al. (1983).

Usage

```
full_Ratkowski(x, xmin, xmax, c)
```

Arguments

x	Value of the environmental factor.
xmin	Minimum value for growth
xmax	Maximum value for growth
c	Parameter defining the speed of the decline

get_dyna_residuals	<i>Residuals of dynamic prediction</i>
--------------------	--

Description

Function for calculating residuals of a dynamic prediction according to the requirements of [modFit](#).

Usage

```
get_dyna_residuals(  
  this_p,  
  fit_data,  
  env_conditions,  
  known_pars,  
  sec_model_names,  
  cost = NULL  
)
```

Arguments

this_p	named vector of model parameters
fit_data	tibble with the data for the fit
env_conditions	tibble with the environmental conditions
known_pars	named vector of known model parameters
sec_model_names	named character vector with names the environmental conditions and values the secondary model (e.g. 'CPM').
cost	an instance of modCost to be combined (to fit multiple models).

Value

An instance of `modCost`.

get_iso_residuals	<i>Residuals of isothermal prediction</i>
-------------------	---

Description

Residuals of isothermal prediction

Usage

```
get_iso_residuals(this_p, fit_data, model_name, known_pars)
```

Arguments

this_p	named vector of model parameters to fit
fit_data	tibble with the data for the fit
model_name	character defining the primary growth model
known_pars	named vector of fixed model parameters

Value

An instance of `modCost`.

get_multi_dyna_residuals	<i>Residuals of multiple dynamic predictions</i>
--------------------------	--

Description

Function for calculating residuals of dynamic predictions under different conditions for the same model parameters according to the requirements of `modFit`.

Usage

```
get_multi_dyna_residuals(this_p, experiment_data, known_pars, sec_model_names)
```

Arguments

this_p	named vector of model parameters
experiment_data	a nested list with the experimental data. Each entry describes one experiment as a list with two elements: data and conditions. data is a tibble with two columns: time and logN. conditions is a tibble with one column named time and as many additional columns as environmental factors.
known_pars	named vector of known model parameters
sec_model_names	named character vector with names the environmental conditions and values the secondary model (see secondary_model_data).

Value

an instance of modCost.

get_secondary_residuals

Residuals of secondary models

Description

Residual function for [fit_secondary_growth](#).

Usage

```
get_secondary_residuals(
  this_p,
  my_data,
  known_pars,
  sec_model_names,
  transformation
)
```

Arguments

this_p	Named vector of model parameter values.
my_data	Tibble with the data used for the fit.
known_pars	Named vector of fixed model parameters.
sec_model_names	Named character vector defining the secondary model for each environmental factor.
transformation	Character defining the transformation of mu for model fitting. One of sq (square root), log (log-transform) or none (no transformation).

Value

A numeric vector of residuals.

growth_pH_temperature *Example of dynamic growth*

Description

A dataset to demonstrate the use of `fit_dynamic_growth`. The values of the environmental conditions are described in `conditions_pH_temperature`.

Usage

```
growth_pH_temperature
```

Format

A tibble with 20 rows and 2 columns:

time elapsed time

logN decimal logarithm of the population size

growth_salmonella *Growth of Salmonella spp in broth*

Description

An example dataset to illustrate `fit_isothermal_growth`. It describes the growth of *Salmonella* spp. in broth. It was retrieved from ComBase (ID: B092_10).

Usage

```
growth_salmonella
```

Format

A tibble with 21 rows and 2 columns:

time elapsed time in hours.

logN observed population size (log CFU/g).

`is.DynamicGrowth` *Test of DynamicGrowth object*

Description

Tests if an object is of class `DynamicGrowth`.

Usage

`is.DynamicGrowth(x)`

Arguments

`x` object to be checked.

Value

A boolean specifying whether `x` is of class `DynamicGrowth`

`is.FitDynamicGrowth` *Test of FitDynamicGrowth object*

Description

Tests if an object is of class `FitDynamicGrowth`.

Usage

`is.FitDynamicGrowth(x)`

Arguments

`x` object to be checked.

Value

A boolean specifying whether `x` is of class `FitDynamicGrowth`

is.FitDynamicGrowthMCMC

Test of FitDynamicGrowthMCMC object

Description

Tests if an object is of class FitDynamicGrowthMCMC.

Usage

is.FitDynamicGrowthMCMC(x)

Arguments

x object to be checked.

Value

A boolean specifying whether x is of class FitDynamicGrowthMCMC

is.FitIsoGrowth

Test of FitIsoGrowth object

Description

Tests if an object is of class FitIsoGrowth.

Usage

is.FitIsoGrowth(x)

Arguments

x object to be checked.

Value

A boolean specifying whether x is of class FitIsoGrowth

`is.FitMultipleDynamicGrowth`

Test of FitMultipleDynamicGrowth object

Description

Tests if an object is of class `FitMultipleDynamicGrowth`.

Usage

`is.FitMultipleDynamicGrowth(x)`

Arguments

`x` object to be checked.

Value

A boolean specifying whether `x` is of class `FitMultipleDynamicGrowth`

`is.FitMultipleDynamicGrowthMCMC`

Test of FitMultipleDynamicGrowthMCMC object

Description

Tests if an object is of class `FitMultipleDynamicGrowthMCMC`.

Usage

`is.FitMultipleDynamicGrowthMCMC(x)`

Arguments

`x` object to be checked.

Value

A boolean specifying whether `x` is of class `FitMultipleDynamicGrowthMCMC`

is.FitSecondaryGrowth *Test of FitSecondaryGrowth object*

Description

Tests if an object is of class `FitSecondaryGrowth`.

Usage

`is.FitSecondaryGrowth(x)`

Arguments

`x` object to be checked.

Value

A boolean specifying whether `x` is of class `FitSecondaryGrowth`

is.IsothermalGrowth *Test of IsothermalGrowth object*

Description

Tests if an object is of class `IsothermalGrowth`.

Usage

`is.IsothermalGrowth(x)`

Arguments

`x` object to be checked.

Value

A boolean specifying whether `x` is of class `IsothermalGrowth`

is.MCMCgrowth *Test of MCMCgrowth object*

Description

Tests if an object is of class MCMCgrowth.

Usage

is.MCMCgrowth(x)

Arguments

x object to be checked.

Value

A boolean specifying whether x is of class MCMCgrowth

is.StochasticGrowth *Test of StochasticGrowth object*

Description

Tests if an object is of class StochasticGrowth.

Usage

is.StochasticGrowth(x)

Arguments

x object to be checked.

Value

A boolean specifying whether x is of class StochasticGrowth

IsothermalGrowth *IsothermalGrowth class*

Description

The IsothermalGrowth class contains the results of a growth prediction under isothermal conditions. Its constructor is [predict_isothermal_growth](#).

It is a subclass of list with the items:

- simulation: A tibble with the model simulation.
- model: The name of the model used for the predictions.
- pars: A list with the values of the model parameters.

Usage

```
## S3 method for class 'IsothermalGrowth'  
print(x, ...)  
  
## S3 method for class 'IsothermalGrowth'  
plot(x, y = NULL, ..., line_col = "black", line_size = 1, line_type = "solid")
```

Arguments

x	The object of class IsothermalGrowth to plot.
...	ignored.
y	ignored
line_col	Aesthetic parameter to change the colour of the line, see: geom_line
line_size	Aesthetic parameter to change the thickness of the line, see: geom_line
line_type	Aesthetic parameter to change the type of the line, takes numbers (1-6) or strings ("solid") see: geom_line

Functions

- `print.IsothermalGrowth`: print of the model
- `plot.IsothermalGrowth`: plot of the predicted growth curve.

iso_Baranyi	<i>Isothermal Baranyi model</i>
-------------	---------------------------------

Description

Baranyi growth model as defined by Baranyi and Roberts (1994). We use the solution calculated by Poschet et al. (2005, doi: <https://doi.org/10.1016/j.ijfoodmicro.2004.10.008>) after log-transformation according to MONTE CARLO ANALYSIS FOR MICROBIAL GROWTH CURVES, by Oksuz and Buzrul.

Usage

```
iso_Baranyi(times, logN0, mu, lambda, logNmax)
```

Arguments

times	Numeric vector of storage times
logN0	Initial log microbial count
mu	Maximum specific growth rate (in ln CFU/[t])
lambda	Lag phase duration
logNmax	Maximum log microbial count

Value

Numeric vector with the predicted microbial count.

iso_repGompertz	<i>Reparameterized Gompertz model</i>
-----------------	---------------------------------------

Description

Reparameterized Gompertz growth model defined by Zwietering et al. (1990).

Usage

```
iso_repGompertz(times, logN0, C, mu, lambda)
```

Arguments

times	Numeric vector of storage times
logN0	Initial log microbial count
C	Difference between logN0 and the maximum log-count.
mu	Maximum specific growth rate (in ln CFU/[t])
lambda	Lag phase duration

Value

Numeric vector with the predicted microbial count.

lambda_to_Q0	<i>Q0 from lag phase duration</i>
--------------	-----------------------------------

Description

Convenience function to calculate the value of Q0 for the Baranyi model from the duration of the lag phase

Usage

```
lambda_to_Q0(lambda, mu)
```

Arguments

lambda	Duration of the lag phase.
mu	Specific growth rate in the exponential phase ($\log_{10}(\text{units})/[\text{time}]$).

logistic_model	<i>Logistic growth model</i>
----------------	------------------------------

Description

Logistic growth model

Usage

```
logistic_model(times, logN0, mu, lambda, C)
```

Arguments

times	Numeric vector of storage times
logN0	Initial log microbial count
mu	Maximum specific growth rate (in $\ln \text{CFU}/[t]$)
lambda	Lag phase duration
C	Difference between $\log N_0$ and the maximum log-count.

Value

Numeric vector with the predicted microbial count

MCMCgrowth

*MCMCgrowth class***Description**

The MCMCgrowth class contains the results of a growth prediction consider parameter variability based on a model fitted using [fit_MCMC_growth](#) or [fit_multiple_growth_MCMC](#).

It is a subclass of list with items:

- `sample`: Parameter sample used for the calculations.
- `simulations`: Individual growth curves calculated based on the parameter sample.
- `quantiles`: Tibble with the limits of the credible intervals (5
- `model`: Instance of `FitDynamicGrowthMCMC` used for predictions.
- `env_conditions`: A tibble with the environmental conditions of the simulation.

Usage

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

## S3 method for class 'MCMCgrowth'
plot(
  x,
  y = NULL,
  ...,
  add_factor = NULL,
  alpha_80 = 0.5,
  fill_80 = "grey",
  alpha_90 = 0.5,
  fill_90 = "grey",
  label_y1 = "logN",
  label_y2 = add_factor,
  line_col = "black",
  line_type = 1,
  line_size = 1,
  line_type2 = 2,
  line_col2 = "black",
  line_size2 = 1,
  ylims = NULL
)
```

Arguments

`x` The object of class MCMCgrowth to plot.
`...` ignored.

y	ignored
add_factor	Includes the variation of one environmental factor in the plot. It must be one of the column names in x\$env_conditions.
alpha_80	transparency of the ribbon for the 80th posterior. .5 by default.
fill_80	fill colour of the ribbon for the 80th posterior. "grey" by default.
alpha_90	transparency of the ribbon for the 90th posterior. .5 by default.
fill_90	fill colour of the ribbon for the 90th posterior. "grey" by default.
label_y1	label of the primary y axis. "logN" by default.
label_y2	label of the secondary y axis. The name of the environmental factor by default.
line_col	colour of the line representing the median. "black" by default.
line_type	linetype for the line representing the median. solid by default.
line_size	size of the line representing the median. 1 by default.
line_type2	linetype for the line representing the environmental condition. Dashed by default.
line_col2	colour of the line representing the environmental condition. "black" by default.
line_size2	size of the line representing the environmental condition. 1 by default.
ylims	limits of the primary y-axis. NULL by default (let ggplot choose).

Functions

- `print.MCMCgrowth`: print of the model
- `plot.MCMCgrowth`: plot of predicted growth (prediction band).

`multiple_experiments` *A set of growth experiments under dynamic conditions*

Description

An example dataset illustrating the requirements of `fit_multiple_growth` and `fit_multiple_growth_MCMC`.

Usage

```
multiple_experiments
```

Format

A nested list with two elements. Each element corresponds to one experiment and is described by a list with two data frames:

data a tibble describing the microbial counts. It has 2 columns: time (elapsed time) and logN (logarithm of the microbial count).

conditions a tibble describing the environmental conditions. It has 3 columns: time (elapsed time), temperature (storage temperature) and pH (pH of the media).

predict_dynamic_growth

Growth under dynamic conditions

Description

Predicts population growth under dynamic conditions based on the Baranyi model (Baranyi and Roberts, 1994) and secondary models based on the gamma concept (Zwietering et al. 1992).

Usage

```
predict_dynamic_growth(
  times,
  env_conditions,
  primary_pars,
  secondary_models,
  ...,
  check = TRUE,
  formula = . ~ time
)
```

Arguments

times	Numeric vector of storage times to make the predictions
env_conditions	Tibble describing the variation of the environmental conditions during storage. It must have with the elapsed time (named <code>time</code> by default; can be changed with the "formula" argument), and as many additional columns as environmental factors.
primary_pars	A named list defining the parameters of the primary model and the initial values of the model variables. That is, with names <code>mu_opt</code> , <code>Nmax</code> , <code>N0</code> , <code>Q0</code> .
secondary_models	A nested list describing the secondary models.
...	Additional arguments for ode .
check	Whether to check the validity of the models. TRUE by default.
formula	An object of class "formula" describing the x variable. <code>. ~ time</code> as a default.

Details

Model predictions are done by linear interpolation of the environmental conditions defined in `env_conditions`.

Value

An instance of [DynamicGrowth](#).

Examples

```
## Definition of the environmental conditions

library(tibble)

my_conditions <- tibble(time = c(0, 5, 40),
  temperature = c(20, 30, 35),
  pH = c(7, 6.5, 5)
)

## Definition of the model parameters

my_primary <- list(mu_opt = 2,
  Nmax = 1e8, N0 = 1e0,
  Q0 = 1e-3)

sec_temperature <- list(model = "Zwietering",
  xmin = 25, xopt = 35, n = 1)

sec_pH = list(model = "CPM",
  xmin = 5.5, xopt = 6.5,
  xmax = 7.5, n = 2)

my_secondary <- list(
  temperature = sec_temperature,
  pH = sec_pH
)

my_times <- seq(0, 50, length = 1000)

## Do the simulation

dynamic_prediction <- predict_dynamic_growth(my_times,
  my_conditions, my_primary,
  my_secondary)

## Plot the results

plot(dynamic_prediction)

## We can plot some environmental factor with add_factor

plot(dynamic_prediction, add_factor = "temperature", ylims= c(0, 8),
  label_y1 = "Microbial count (log CFU/ml)",
  label_y2 = "Storage temperature (C)")
```

predict_isothermal_growth

Isothermal microbial growth

Description

Predicts population growth under static conditions using primary models.

Usage

```
predict_isothermal_growth(model_name, times, model_pars, check = TRUE)
```

Arguments

model_name	Character defining the growth model.
times	Numeric vector of storage times for the predictions.
model_pars	Named vector or list defining the values of the model parameters.
check	Whether to do basic checks (TRUE by default).

Value

An instance of [IsothermalGrowth](#).

Examples

```
## Define the simulations parameters

my_model <- "modGompertz"
my_pars <- list(logN0 = 2, C = 6, mu = .2, lambda = 25)
my_time <- seq(0, 100, length = 1000)

## Do the simulation

static_prediction <- predict_isothermal_growth(my_model, my_time, my_pars)

## Plot the results

plot(static_prediction)
```

predict_MCMC_growth *Stochastic growth of MCMC fit*

Description

Makes a stochastic prediction of microbial growth based on a growth model fitted using [fit_MCMC_growth](#) or [fit_multiple_growth_MCMC](#). This function predicts growth curves for niter samples (with replacement) of the samples of the MCMC algorithm. Then, credible intervals are calculated based on the quantiles of the model predictions at each time point.

Usage

```
predict_MCMC_growth(MCMCfit, times, env_conditions, niter, newpars = NULL)
```

Arguments

MCMCfit	An instance of FitDynamicGrowthMCMC or FitMultipleGrowthMCMC.
times	Numeric vector of storage times for the predictions.
env_conditions	Tibble with the (dynamic) environmental conditions during the experiment. It must have one column named 'time' with the storage time and as many columns as required with the environmental conditions.
niter	Number of iterations.
newpars	A named list defining new values for the some model parameters. The name must be the identifier of a model already included in the model. These parameters do not include variation, so defining a new value for a fitted parameters "fixes" it. NULL by default (no new parameters).

Value

An instance of [MCMCgrowth](#).

Examples

```
## We need a FitDynamicGrowthMCMC object

data("example_dynamic_growth")
data("example_env_conditions")

sec_model_names <- c(temperature = "CPM", aw= "CPM")

known_pars <- list(Nmax = 1e4, # Primary model
  N0 = 1e0, Q0 = 1e-3, # Initial values of the primary model
  mu_opt = 4, # mu_opt of the gamma model
  temperature_n = 1, # Secondary model for temperature
  aw_xmax = 1, aw_xmin = .9, aw_n = 1 # Secondary model for water activity
)

my_start <- list(temperature_xmin = 25, temperature_xopt = 35,
  temperature_xmax = 40,
  aw_xopt = .95)

set.seed(12124) # Setting seed for repeatability

my_MCMC_fit <- fit_MCMC_growth(example_dynamic_growth, example_env_conditions,
  my_start, known_pars, sec_model_names, niter = 3000)

## Define the conditions for the simulation

my_times <- seq(0, 15, length = 50)
niter <- 2000
```

```

newpars <- list(N0 = 1e-1, # A parameter that was fixed
               temperature_xmax = 120 # A parameter that was fitted
               )

## Make the simulations

my_MCMC_prediction <- predict_MCMC_growth(my_MCMC_fit,
    my_times,
    example_env_conditions, # It could be different from the one used for fitting
    niter,
    newpars)

## We can plot the prediction interval

plot(my_MCMC_prediction)

## We can also get the quantiles at each time point

print(my_MCMC_prediction$quantiles)

```

predict_stochastic_growth

Isothermal growth with variability

Description

Stochastic simulation of microbial growth based on probability distributions of the parameters of the primary model. It is included by Monte Carlo simulation considering the parameters follow a multivariate normal distribution.

Usage

```

predict_stochastic_growth(
  model_name,
  times,
  n_sims,
  pars,
  corr_matrix = diag(nrow(pars)),
  check = TRUE
)

```

Arguments

model_name	Character describing the primary growth model.
times	Numeric vector of storage times for the simulations.

n_sims	Number of simulations.
pars	A tibble describing the parameter uncertainty (see details).
corr_matrix	Correlation matrix of the model parameters. Defined in the same order as in pars. An identity matrix by default (uncorrelated parameters).
check	Whether to do some tests. FALSE by default.

Details

They are defined in the pars argument using a tibble with 4 columns:

- par: identifier of the model parameter (according to [primary_model_data](#)),
- mean: mean value of the model parameter.,
- sd: standard deviation of the model parameter.,
- scale: scale at which the model parameter is defined. Valid values are 'original' (no transformation), 'sqrt' square root or 'log' log-scale. The parameter sample is generated considering the parameter follows a marginal normal distribution at this scale, and is later converted to the original scale for calculations.

Value

An instance of [StochasticGrowth](#).

Examples

```
## Definition of the simulation settings

my_model <- "Baranyi"
my_times <- seq(0, 30, length = 100)
n_sims <- 3000

library(tibble)

pars <- tribble(
  ~par, ~mean, ~sd, ~scale,
  "logN0", 0, .2, "original",
  "mu", 2, .3, "sqrt",
  "lambda", 4, .4, "sqrt",
  "logNmax", 6, .5, "original"
)

## Calling the function

stoc_growth <- predict_stochastic_growth(my_model, my_times, n_sims, pars)

## We can plot the results

plot(stoc_growth)

## Adding parameter correlation
```

```

my_cor <- matrix(c(1, 0, 0, 0,
  0, 1, 0.7, 0,
  0, 0.7, 1, 0,
  0, 0, 0, 1),
  nrow = 4)

stoc_growth2 <- predict_stochastic_growth(my_model, my_times, n_sims, pars, my_cor)

plot(stoc_growth2)

```

primary_model_data *Metainformation of primary growth models*

Description

Metainformation of primary growth models

Usage

```
primary_model_data(model_name = NULL)
```

Arguments

model_name The name of the model or NULL (default).

Value

If model_name is NULL, returns a character string with the available models. If is a valid identifier, it returns a list with metainformation about the model. If model_name name is not a valid identifier, raises an error.

Q0_to_lambda *Lag phase duration from Q0*

Description

Convenience function to calculate the lag phase duration (lambda) of the Baranyi model from the maximum specific growth rate and the initial value of the variable Q.

Usage

```
Q0_to_lambda(q0, mu)
```

Arguments

q0	Initial value of the variable Q.
mu	Specific growth rate in the exponential phase (log10(units)/[time]).

Details

Note that this function uses the unit system of biogrowth (i.e. log10). Care must be taken when using parameters obtained from other sources.

richards_model	<i>Richards growth model</i>
----------------	------------------------------

Description

Richards growth model

Usage

```
richards_model(times, logN0, mu, lambda, C, nu)
```

Arguments

times	Numeric vector of storage times
logN0	Initial log microbial count
mu	Maximum specific growth rate (in ln CFU/[t])
lambda	Lag phase duration
C	Difference between logN0 and the maximum log-count.
nu	Parameter describing the transition between growth phases

secondary_model_data	<i>Metainformation of secondary growth models</i>
----------------------	---

Description

Metainformation of secondary growth models

Usage

```
secondary_model_data(model_name = NULL)
```

Arguments

model_name	The name of the model or NULL (default).
------------	--

Value

If `model_name` is `NULL`, returns a character string with the available models. If is a valid identifier, it returns a list with meta-information about the model. If `model_name` name is not a valid identifier, raises an error.

StochasticGrowth *StochasticGrowth class*

Description

The `StochasticGrowth` class contains the results of a growth prediction under isothermal conditions considering parameter uncertainty. Its constructor is [predict_stochastic_growth](#).

It is a subclass of list with the items:

- `sample`: parameter sample used for the calculations.
- `simulations`: growth curves predicted for each parameter.
- `quantiles`: limits of the credible intervals (5 each time point).
- `model`: Model used for the calculations.
- `mus`: Mean parameter values used for the simulations.
- `sigma`: Variance-covariance matrix used for the simulations.

Usage

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

## S3 method for class 'StochasticGrowth'
plot(
  x,
  y = NULL,
  ...,
  line_col = "black",
  line_size = 0.5,
  line_type = "solid",
  ribbon80_fill = "grey",
  ribbon90_fill = "grey",
  alpha80 = 0.5,
  alpha90 = 0.4
)
```

Arguments

<code>x</code>	The object of class <code>StochasticGrowth</code> to plot.
<code>...</code>	ignored.
<code>y</code>	ignored

line_col	Aesthetic parameter to change the colour of the line geom in the plot, see: geom_line
line_size	Aesthetic parameter to change the thickness of the line geom in the plot, see: geom_line
line_type	Aesthetic parameter to change the type of the line geom in the plot, takes numbers (1-6) or strings ("solid") see: geom_line
ribbon80_fill	fill colour for the space between the 10th and 90th quantile, see: geom_ribbon
ribbon90_fill	fill colour for the space between the 5th and 95th quantile, see: geom_ribbon
alpha80	transparency of the ribbon aesthetic for the space between the 10th and 90th quantile. Takes a value between 0 (fully transparent) and 1 (fully opaque)
alpha90	transparency of the ribbon aesthetic for the space between the 5th and 95th quantile. Takes a value between 0 (fully transparent) and 1 (fully opaque).

Functions

- `print.StochasticGrowth`: print of the model
- `plot.StochasticGrowth`: Growth prediction (prediction band) considering parameter uncertainty.

TimeDistribution	<i>TimeDistribution class</i>
------------------	-------------------------------

Description

The `TimeDistribution` class contains an estimate of the probability distribution of the time to reach a given microbial count. Its constructor is [distribution_to_logcount](#).

It is a subclass of list with the items:

- `distribution` Sample of the distribution of times to reach `log_count`.
- `summary` Summary statistics of distribution (mean, sd, median, q10 and q90).

Usage

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

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

## S3 method for class 'TimeDistribution'
plot(x, y = NULL, ..., bin_width = NULL)
```

Arguments

x	The object of class TimeDistribution to plot.
...	ignored.
object	An instance of TimeDistribution.
y	ignored.
bin_width	A number that specifies the width of a bin in the histogram, see: geom_histogram . NULL by default.

Functions

- `print.TimeDistribution`: print of the model
- `summary.TimeDistribution`: summary of the model
- `plot.TimeDistribution`: plot of the distribution of the time to reach a microbial count.

<code>time_to_logcount</code>	<i>Time to reach a given microbial count</i>
-------------------------------	--

Description

Returns the storage time required for the microbial count to reach `log_count` according to the predictions of `model`. Calculations are done using linear interpolation of the model predictions.

Usage

```
time_to_logcount(model, log_count)
```

Arguments

model	An instance of IsothermalGrowth or DynamicGrowth.
log_count	The target log microbial count.

Value

The predicted time to reach `log_count`.

Examples

```
## First of all, we will get an IsothermalGrowth object

my_model <- "modGompertz"
my_pars <- list(logN0 = 2, C = 6, mu = .2, lambda = 25)
my_time <- seq(0, 100, length = 1000)

static_prediction <- predict_isothermal_growth(my_model, my_time, my_pars)
plot(static_prediction)
```

```
## And now we calculate the time to reach a microbial count
time_to_logcount(static_prediction, 2.5)

## If log_count is outside the range of the predicted values, NA is returned
time_to_logcount(static_prediction, 20)
```

trilinear_model	<i>Trilinear growth model</i>
-----------------	-------------------------------

Description

Trilinear growth model defined by Buchanan et al. (1997).

Usage

```
trilinear_model(times, logN0, mu, lambda, logNmax)
```

Arguments

times	Numeric vector of storage times
logN0	Initial log microbial count
mu	Maximum specific growth rate (in ln CFU/[t])
lambda	Lag phase duration
logNmax	Maximum log microbial count

Value

Numeric vector with the predicted microbial count.

zwietering_gamma	<i>Zwietering gamma model</i>
------------------	-------------------------------

Description

Gamma model as defined by Zwietering et al. (1992). To avoid unreasonable predictions, it has been modified setting gamma=0 for values of x outside [xmin, xopt]

Usage

```
zwietering_gamma(x, xmin, xopt, n)
```

Arguments

x	Value of the environmental factor.
xmin	Minimum value of the environmental factor for growth.
xopt	Maximum value for growth
n	Exponent of the secondary model

Value

The corresponding gamma factor.

Index

* datasets

- arabian_tractors, 4
 - conditions_pH_temperature, 7
 - example_cardinal, 11
 - example_dynamic_growth, 12
 - example_env_conditions, 12
 - growth_pH_temperature, 41
 - growth_salmonella, 41
 - multiple_experiments, 51
- approx_env, 3
- arabian_tractors, 4
- calculate_gammas, 4
- calculate_gammas_secondary, 5
- check_primary_pars, 5
- check_secondary_pars, 6
- check_stochastic_pars, 6
- coef.FitDynamicGrowth
(FitDynamicGrowth), 14
- coef.FitDynamicGrowthMCMC
(FitDynamicGrowthMCMC), 16
- coef.FitIsoGrowth (FitIsoGrowth), 18
- coef.FitMultipleDynamicGrowth
(FitMultipleDynamicGrowth), 20
- coef.FitMultipleGrowthMCMC
(FitMultipleGrowthMCMC), 23
- coef.FitSecondaryGrowth
(FitSecondaryGrowth), 25
- conditions_pH_temperature, 7
- CPM_model, 7
- dBaranyi, 8
- deviance.FitDynamicGrowth
(FitDynamicGrowth), 14
- deviance.FitDynamicGrowthMCMC
(FitDynamicGrowthMCMC), 16
- deviance.FitIsoGrowth (FitIsoGrowth), 18
- deviance.FitMultipleDynamicGrowth
(FitMultipleDynamicGrowth), 20
- deviance.FitMultipleGrowthMCMC
(FitMultipleGrowthMCMC), 23
- deviance.FitSecondaryGrowth
(FitSecondaryGrowth), 25
- distribution_to_logcount, 9, 61
- DynamicGrowth, 10, 52
- example_cardinal, 11
- example_dynamic_growth, 12
- example_env_conditions, 12
- extract_primary_pars, 13
- extract_secondary_pars, 13
- fit_dynamic_growth, 12, 14, 27
- fit_isothermal_growth, 18, 29, 41
- fit_MCMC_growth, 16, 30, 50, 54
- fit_multiple_growth, 20, 22, 25, 32, 51
- fit_multiple_growth_MCMC, 23, 34, 50, 51, 54
- fit_secondary_growth, 11, 25, 26, 36, 40
- FitDynamicGrowth, 14, 28
- FitDynamicGrowthMCMC, 16, 31
- FitIsoGrowth, 18, 29
- FitMultipleDynamicGrowth, 20, 33
- FitMultipleGrowthMCMC, 23, 35
- FitSecondaryGrowth, 25, 37
- fitted.FitDynamicGrowth
(FitDynamicGrowth), 14
- fitted.FitDynamicGrowthMCMC
(FitDynamicGrowthMCMC), 16
- fitted.FitIsoGrowth (FitIsoGrowth), 18
- fitted.FitMultipleDynamicGrowth
(FitMultipleDynamicGrowth), 20
- fitted.FitMultipleGrowthMCMC
(FitMultipleGrowthMCMC), 23
- fitted.FitSecondaryGrowth
(FitSecondaryGrowth), 25
- full_Ratkowski, 38
- geom_histogram, 62

- geom_line, [11](#), [15](#), [17–20](#), [22](#), [24](#), [47](#), [61](#)
- geom_point, [15](#), [18](#), [20](#)
- geom_ribbon, [61](#)
- get_dyna_residuals, [38](#)
- get_iso_residuals, [39](#)
- get_multi_dyna_residuals, [39](#)
- get_secondary_residuals, [40](#)
- gompertz (iso_repGompertz), [48](#)
- growth_pH_temperature, [41](#)
- growth_salmonella, [41](#)

- is.DynamicGrowth, [42](#)
- is.FitDynamicGrowth, [42](#)
- is.FitDynamicGrowthMCMC, [43](#)
- is.FitIsoGrowth, [43](#)
- is.FitMultipleDynamicGrowth, [44](#)
- is.FitMultipleDynamicGrowthMCMC, [44](#)
- is.FitSecondaryGrowth, [45](#)
- is.IsothermalGrowth, [45](#)
- is.MCMCgrowth, [46](#)
- is.StochasticGrowth, [46](#)
- iso_Baranyi, [48](#)
- iso_repGompertz, [48](#)
- IsothermalGrowth, [47](#), [54](#)

- lambda_to_Q0, [49](#)
- logistic_model, [49](#)

- MCMCgrowth, [50](#), [55](#)
- modCost, [39](#)
- modFit, [18](#), [25](#), [29](#), [33](#), [37–39](#)
- modGompertz (iso_repGompertz), [48](#)
- multiple_experiments, [51](#)

- ode, [8](#), [52](#)

- plot.DynamicGrowth (DynamicGrowth), [10](#)
- plot.FitDynamicGrowth (FitDynamicGrowth), [14](#)
- plot.FitDynamicGrowthMCMC (FitDynamicGrowthMCMC), [16](#)
- plot.FitIsoGrowth (FitIsoGrowth), [18](#)
- plot.FitMultipleDynamicGrowth (FitMultipleDynamicGrowth), [20](#)
- plot.FitMultipleGrowthMCMC (FitMultipleGrowthMCMC), [23](#)
- plot.FitSecondaryGrowth (FitSecondaryGrowth), [25](#)
- plot.IsothermalGrowth (IsothermalGrowth), [47](#)

- plot.MCMCgrowth (MCMCgrowth), [50](#)
- plot.StochasticGrowth (StochasticGrowth), [60](#)
- plot.TimeDistribution (TimeDistribution), [61](#)

- predict.FitDynamicGrowth (FitDynamicGrowth), [14](#)
- predict.FitDynamicGrowthMCMC (FitDynamicGrowthMCMC), [16](#)
- predict.FitIsoGrowth (FitIsoGrowth), [18](#)
- predict.FitMultipleDynamicGrowth (FitMultipleDynamicGrowth), [20](#)
- predict.FitMultipleGrowthMCMC (FitMultipleGrowthMCMC), [23](#)
- predict.FitSecondaryGrowth (FitSecondaryGrowth), [25](#)

- predict_dynamic_growth, [4](#), [10](#), [15](#), [18](#), [52](#)
- predict_isothermal_growth, [47](#), [53](#)
- predict_MCMC_growth, [54](#)
- predict_stochastic_growth, [56](#), [60](#)

- primary_model_data, [57](#), [58](#)

- print.DynamicGrowth (DynamicGrowth), [10](#)
- print.FitDynamicGrowth (FitDynamicGrowth), [14](#)
- print.FitDynamicGrowthMCMC (FitDynamicGrowthMCMC), [16](#)
- print.FitIsoGrowth (FitIsoGrowth), [18](#)
- print.FitMultipleDynamicGrowth (FitMultipleDynamicGrowth), [20](#)
- print.FitMultipleGrowthMCMC (FitMultipleGrowthMCMC), [23](#)
- print.FitSecondaryGrowth (FitSecondaryGrowth), [25](#)
- print.IsothermalGrowth (IsothermalGrowth), [47](#)
- print.MCMCgrowth (MCMCgrowth), [50](#)
- print.StochasticGrowth (StochasticGrowth), [60](#)
- print.TimeDistribution (TimeDistribution), [61](#)

- Q0_to_lambda, [58](#)

- residuals.FitDynamicGrowth (FitDynamicGrowth), [14](#)
- residuals.FitDynamicGrowthMCMC (FitDynamicGrowthMCMC), [16](#)
- residuals.FitIsoGrowth (FitIsoGrowth), [18](#)

residuals.FitMultipleDynamicGrowth
 (FitMultipleDynamicGrowth), 20

residuals.FitMultipleGrowthMCMC
 (FitMultipleGrowthMCMC), 23

residuals.FitSecondaryGrowth
 (FitSecondaryGrowth), 25

richards_model, 59

secondary_model_data, 59

StochasticGrowth, 57, 60

summary.FitDynamicGrowth
 (FitDynamicGrowth), 14

summary.FitDynamicGrowthMCMC
 (FitDynamicGrowthMCMC), 16

summary.FitIsoGrowth (FitIsoGrowth), 18

summary.FitMultipleDynamicGrowth
 (FitMultipleDynamicGrowth), 20

summary.FitMultipleGrowthMCMC
 (FitMultipleGrowthMCMC), 23

summary.FitSecondaryGrowth
 (FitSecondaryGrowth), 25

summary.TimeDistribution
 (TimeDistribution), 61

time_to_logcount, 62

TimeDistribution, 9, 61

trilinear_model, 63

vcov.FitDynamicGrowth
 (FitDynamicGrowth), 14

vcov.FitDynamicGrowthMCMC
 (FitDynamicGrowthMCMC), 16

vcov.FitIsoGrowth (FitIsoGrowth), 18

vcov.FitMultipleDynamicGrowth
 (FitMultipleDynamicGrowth), 20

vcov.FitMultipleGrowthMCMC
 (FitMultipleGrowthMCMC), 23

vcov.FitSecondaryGrowth
 (FitSecondaryGrowth), 25

zwietering_gamma, 63