Package ‘bioinactivation’

December 15, 2017

Type Package
Title Simulation of Dynamic Microbial Inactivation
Version 1.2.1
Date 2017-12-15
Description Prediction and adjustment to experimental data of microbial inactivation. Several models available in the literature are implemented.
License GPL-3
LazyData TRUE
Imports dplyr (>= 0.4.1), deSolve (>= 1.11), FME (>= 1.3.2), lazyeval (>= 0.1.10), ggplot2 (>= 2.0.0), MASS (>= 7.3-39), graphics (>= 3.1.3), stats (>= 3.1.3), rlang(>= 0.1.2)
Suggests knitr (>= 1.9), testthat (>= 0.9.1)
VignetteBuilder knitr
RoxygenNote 6.0.1
NeedsCompilation no
Author Alberto Garre [aut, cre], Pablo S. Fernandez [aut], Jose A. Egea [aut]
Maintainer Alberto Garre <garre.alberto@gmail.com>
Repository CRAN
Date/Publication 2017-12-15 12:09:39 UTC

R topics documented:

Arrhenius_iso ............................................. 2
Bigelow_iso ............................................ 3
build_temperature_interpolator .................. 4
check_model_params ................................. 4
dArrhenius_model ................................. 5
dBigelow_model .................................. 6
dGeeraerd_model .............................. 7
### Description

Returns the predicted logarithmic reduction in microbial count according to Arrhenius model for the time, temperature and model parameters given.

### Usage

```r
Arrhenius_iso(time, temp, k_ref, temp_ref, Ea)
```
Arguments

- **time**: numeric indicating the time at which the prediction is taken.
- **temp**: numeric indicating the temperature of the treatment.
- **k_ref**: numeric indicating the inactivation rate at the reference temperature.
- **temp_ref**: numeric indicating the reference temperature.
- **Ea**: numeric indicating the activation energy.

Value

A numeric with the predicted logarithmic reduction \( \log_{10}(N/N_0) \).

---

**Bigelow_iso**

**Isothermal Bigelow’s Model**

**Description**

Returns the predicted logarithmic reduction in microbial count according to Bigelow’s model for the time, temperature and model parameters given.

**Usage**

Bigelow_iso(time, temp, z, D_R, temp_ref)

**Arguments**

- **time**: numeric indicating the time at which the prediction is taken.
- **temp**: numeric indicating the temperature of the treatment.
- **z**: numeric defining the z-value.
- **D_R**: numeric defining the D-value at the reference temperature.
- **temp_ref**: numeric defining the reference temperature.

**Value**

A numeric with the predicted logarithmic reduction \( \log_{10}(N/N_0) \).
build_temperature_interpolator

Continuum Interpolation of Discrete Temperatures Values

Description

Builds an interpolator of the temperature at each time and its first derivative. First derivatives are approximated using forward finite differences (approxfun). It is assumed that temperature is 0 and constant outside the time interval provided.

Usage

build_temperature_interpolator(temperature_data)

Arguments

temperature_data
data frame with the values of the temperatures at each value of time. It needs to have 2 columns, named time and temperature.

Value

a list with with two elements:

• temp, the interpolator of the temperature and
• dtemp, the interpolator of its first derivative

See Also

approxfun

check_model_params

Correctness Check of Model Parameters

Description

Makes 3 checks of compatibility between the input parameters for the adjustment and the DOF of the inactivation model selected.

• Check of equal length of model DOF and input DOF. Raises a warning if failed.
• Check that every single one of the input DOF is a model DOF. Raises a warning if failed.
• Check that every single one of the model DOF are defined as an input DOF.

Usage

check_model_params(simulation_model, known_params, starting_points, adjust_vars)
Arguments

simulation_model
character with a valid model key.

known_params
named vector or list with the dof of the model known.

starting_points
named vector or list with the dof of the model to be adjusted.

adjust_vars
logical specifying whether the model variables need to be included in the check
(not used for isothermal fit)


dArrhenius_model  First derivative of the Arrhenius model

Description

Calculates the first derivative of the Arrhenius model with log-linear inactivation for dynamic problems at a given time for the model parameters provided and the environmental conditions given.

Usage

dArrhenius_model(t, x, parms, temp_profile)

Arguments

 t
numeric vector indicating the time of the experiment.

 x
list with the value of N at t.

 parms
parameters for the secondary model. No explicit check of their validity is performed.

 temp_profile
a function that provides the temperature at a given time.

Details

This function is compatible with the function predict_inactivation.

Value

The value of the first derivative of N at time t as a list.

Model Equation

\[
\frac{dN}{dt} = -k \times N
\]
Model parameters

- temp_ref: Reference temperature for the calculation.
- k_ref: Inactivation rate at the ref. temp.
- Ea: Activation energy.

See Also

predict_inactivation

---

**dBigelow_model**

*First Derivate of the Linear Bigelow Model*

**Description**

Calculates the first derivative of the linearized version of Bigelow’s model for dynamic problems at a given time for the model parameters provided and the environmental conditions given.

**Usage**

dBigelow_model(t, x, parms, temp_profile)

**Arguments**

- **t**: numeric vector indicating the time of the experiment.
- **x**: list with the value of N at t.
- **parms**: parameters for the secondary model. No explicit check of their validity is performed.
- **temp_profile**: a function that provides the temperature at a given time.

**Details**

The model is developed from the isothermal Bigelow’s model assuming during the derivation process that $D_T$ is time independent.

This function is compatible with the function `predict_inactivation`.

**Value**

The value of the first derivative of N at time t as a list.

**Model Equation**

\[
\frac{dN}{dt} = -N \frac{\ln(10)}{D_T(T)}
\]
Model parameters

- `temp_ref`: Reference temperature for the calculation,
- `D_R`: D-value at the reference temperature,
- `z`: z value.

See Also

`predict_inactivation`

---

**dGeeraerd_model**  
*First Derivate of Geeraerd’s Model*

**Description**

Calculates the first derivative of the Geeraerd’s model at a given time for the model parameters provided and the environmental conditions given.

**Usage**

`dGeeraerd_model(t, x, parms, temp_profile)`

**Arguments**

- `t` numeric vector indicating the time of the experiment.
- `x` list with the values of the variables at time `t`.
- `parms` parameters for the secondary model. No explicit check of their validity is performed (see section Model Parameters).
- `temp_profile` a function that provides the temperature at a given time.

**Details**

This function is compatible with the function `predict_inactivation`.

**Value**

A list with the value of the first derivatives of `N` and `C_c` at time `t`.

**Model Equation**

\[
\frac{dN}{dt} = -N \cdot k_{max} \cdot \alpha \cdot \gamma
\]

\[
\frac{dC_c}{dt} = -C_c \cdot k_{max}
\]
\[ \alpha = \frac{1}{1 + C_c} \]
\[ \gamma = 1 - \frac{N_{res}}{N} \]
\[ k_{max} = \frac{1}{D_T} \]
\[ \log_{10} D_T = \log_{10} D_R + \frac{T_R - T}{z} \]

Model Parameters
- `temp_ref`: Reference temperature for the calculation,
- `D_R`: D-value at the reference temperature,
- `z`: z value,
- `N_min`: Minimum value of N (defines the tail).

Notes
To define the Geeraerd model without tail, assign `N_min = 0`. For the model without shoulder, assign `C_0 = 0`.

See Also
- `predict_inactivation`

Description
Calculates the first derivative of Weibull-Mafart model at a given time for the model parameters provided and the environmental conditions given.

Usage
```
dMafart_model(t, x, parms, temp_profile)
```
dMafart_model

Arguments

t numeric vector indicating the time of the experiment.

x list with the value of N at t.

params parameters for the secondary model. No explicit check of their validity is performed (see section Model Parameters).

temp_profile a function that provides the temperature at a given time.

Details

The model is developed from the isothermal Weibull-Mafart model without taking into account in the derivation the time dependence of \( \delta_T \) for non-isothermal temperature profiles.

This function is compatible with the function predict_inactivation.

Value

The value of the first derivative of N at time t as a list.

Model Equation

\[
\frac{dN}{dt} = -N \cdot p \cdot (1/\delta)^p \cdot t^{p-1}
\]

\[
\delta(T) = \delta_{ref} \cdot 10^{-(T-T_{ref})/z}
\]

Model Parameters

- temp_ref: Reference temperature for the calculation.
- delta_ref: Value of the scale factor at the reference temperature.
- z: z-value.
- p: shape factor of the Weibull distribution.

Note

For t=0, \( dN = 0 \) unless \( n=1 \). Hence, a small shift needs to be introduced to t.

See Also

predict_inactivation
Description

Calculates the first derivative of Weibull-Peleg model at a given time for the model parameters provided and the environmental conditions given.

Usage

dPeleg_model(t, x, parms, temp_profile)

Arguments

t numeric vector indicating the time of the experiment.

x list with the value of logS at t.

parms parameters for the secondary model. No explicit check of their validity is performed (see section Model Parameters).

temp_profile a function that provides the temperature at a given time.

Details

The model is developed from the isothermal Weibull model without taking into account in the derivation the time dependence of $b$ for non-isothermal temperature profiles.

This function is compatible with the function predict_inactivation.

Value

The value of the first derivative of logS at time t as a list.

Model Equation

$$\frac{d(\log_{10}(S))}{dt} = -b(T) \cdot n \cdot \left(-\log_{10}(S)/b(T)\right)^{(n-1)/n}$$

$$b(T) = \ln(1 + e^{k_b \cdot (T - T_{crit})})$$

Model Parameters

- temp_crit: Temperature below which there is inactivation.
- k_b: slope of the b ~ temp line for temperatures above the critical one.
- n: shape factor of the Weibull distribution.
Note

For $\log S = 0$, $d\log S = 0$ unless $n=1$. Hence, a small shift needs to be introduced to $\log S$.

See Also

predict_inactivation

---

dynamic_inactivation  Example Dynamic Inactivation of a Microorganism

Description

Example of experimental data of the dynamic inactivation process of a microorganism.

Usage

data(dynamic_inactivation)

Format

A data frame with 19 rows and 2 variables.

Details

- time: Time in minutes of the measurement.
- N: Number of microorganism.
- temperature: Observed temperature.

---

fit_dynamic_inactivation  Fitting of Dynamic Inactivation Models

Description

Fits the parameters of an inactivation model to experimental data. The function modFit of the package FME is used for the adjustment.

Usage

fit_dynamic_inactivation(experiment_data, simulation_model, temp_profile, starting_points, upper_bounds, lower_bounds, known_params, ..., minimize_log = TRUE, tol0 = 1e-05)
Arguments

- **experiment_data**: data frame with the experimental data to be adjusted. It must have a column named “time” and another one named “N”.
- **simulation_model**: character identifying the model to be used.
- **temp_profile**: data frame with discrete values of the temperature for each time. It must have one column named `time` and another named `temperature` providing discrete values of the temperature at time points.
- **starting_points**: starting values of the parameters for the adjustment.
- **upper_bounds**: named numerical vector defining the upper bounds of the parameters for the adjustment.
- **lower_bounds**: named numerical vector defining the lower bounds of the parameters for the adjustment.
- **known_params**: named numerical vector with the fixed (i.e., not adjustable) model parameters.
- **...**: further arguments passed to `modFit`.
- **minimize_log**: logical. If `true`, the adjustment is based on the minimization of the error of the logarithmic count. `true` by default.
- **tol0**: numeric. Observations at time 0 make Weibull-based models singular. The time for observations taken at time 0 are changed for this value.

Value

A list of class `FitInactivation` with the following items:

- `fit_results`: a list of class `modFit` with the info of the adjustment.
- `best_prediction`: a list of class `SimulInactivation` with prediction made by the adjusted model.
- `data`: a data frame with the data used for the fitting.

See Also

- `modFit`

Examples

```r
## EXAMPLE 1 ------

data(dynamic_inactivation) # The example data set is used.

get_model_data() # Retrieve the valid model keys.

simulation_model <- "Peleg" # Peleg's model will be used

model_data <- get_model_data(simulation_model)
model_data$parameters # Set the model parameters
```
## fit_inactivation_MCMC

**Fitting of dynamic inactivation with MCMC**

### Description

Fits the parameters of an inactivation model to experimental using the Markov Chain Monte Carlo fitting algorithm implemented in the function `modMCMC` of the package [FME](https://example.com).

### Usage

```r
fit_inactivation_MCMC(experiment_data, simulation_model, temp_profile, 
starting_points, upper_bounds, lower_bounds, known_params, ..., 
minimize_log = TRUE, tol0 = 1e-05)
```

### Arguments

- **experiment_data**
  - data frame with the experimental data to be adjusted. It must have a column named “time” and another one named “N”.
- **simulation_model**
  - character identifying the model to be used.
- **temp_profile**
  - data frame with discrete values of the temperature for each time. It must have one column named `time` and another named `temperature` providing discrete values of the temperature at time points.
- **starting_points**
  - starting values of the parameters for the adjustment.
upper_bounds  named numerical vector defining the upper bounds of the parameters for the adjustment.
lower_bounds  named numerical vector defining the lower bounds of the parameters for the adjustment.
known_params  named numerical vector with the fixed (i.e., not adjustable) model parameters.
...  other arguments for modMCMC.
minimize_log  logical. If TRUE, the adjustment is based on the minimization of the error of the logarithmic count.
tol0  numeric. Observations at time 0 make Weibull-based models singular. The time for observations taken at time 0 are changed for this value.

Value
A list of class `FitInactivationMCMC` with the following items:

- `modMCMC`: a list of class `modMCMC` with the information of the adjustment process.
- `best_prediction`: a list of class `SimulInactivation` with the prediction generated by the best predictor.
- `data`: a data frame with the data used for the fitting.

Examples
```r
## EXAMPLE 1 ------
data(dynamic_inactivation)  # The example data set is used.
get_model_data()  # Retrieve the valid model keys.
simulation_model <- "Peleg"  # Peleg's model will be used
model_data <- get_model_data(simulation_model)
model_data$parameters  # Set the model parameters
dummy_temp <- data.frame(time = c(0, 1.25, 2.25, 4.6),
temperature = c(70, 105, 105, 70))  # Dummy temp. profile

## Set known parameters and initial points/bounds for unknown ones
known_params = c(temp_crit = 100)
starting_points <- c(n = 1, k_b = 0.25, N0 = 1e+05)
upper_bounds <- c(n = 2, k_b = 1, N0 = 1e6)
lower_bounds <- c(n = 0.5, k_b = 0.1, N0 = 1e4)
MCMC_fit <- fit_inactivation_MCMC(dynamic_inactivation, simulation_model,
dummy_temp, starting_points,
upper_bounds, lower_bounds,
known_params,
niter = 100)
# It is recommended to increase niter
```
fit_isothermal_inactivation

Fit of Isothermal Experiments

Description

Fits the parameters of the model chosen to a set of isothermal experiments using nonlinear regression through the function \texttt{nls}.

Usage

\begin{verbatim}
fit_isothermal_inactivation(model_name, death_data, starting_point, known_params, adjust_log = TRUE)
\end{verbatim}

Arguments

- \texttt{model_name}: character specifying the model to adjust.
- \texttt{death_data}: data frame with the experiment data where each row is one observation. It must have the following columns:
  - \texttt{log_diff}: Number of logarithmic reductions at each data point.
  - \texttt{temp}: Temperature of the data point.
  - \texttt{time}: Time of the data point.
- \texttt{starting_point}: List with the initial values of the parameters for the adjustment.
- \texttt{known_params}: List of the parameters of the model known.
- \texttt{adjust_log}: logical. If \texttt{TRUE}, the adjustment is based on the minimization of the error of the logarithmic microbial count. If \texttt{FALSE}, it is based on the minimization of the error of the microbial count. \texttt{TRUE} by default.

Value

An instance of class \texttt{IsoFitInactivation} with the results. This list has four entries:

- \texttt{nls}: The object of class \texttt{nls} with the results of the adjustment.
- \texttt{parameters}: a list with the values of the model parameters, both fixed and adjusted.
- \texttt{model}: a string with the key identifying the model used.
- \texttt{data}: the inactivation data used for the fit.

See Also

\texttt{nls}
### Examples

```r
## EXAMPLE 1

data("isothermal_inactivation") # data set used for the example.

get_isothermal_model_data() # retrieve valid model keys.
model_name <- "Bigelow" # Bigelow's model will be used for the adjustment.

model_data <- get_isothermal_model_data(model_name)
model_data$params # Get the parameters of the model

## Define the input arguments

known_params = list(temp_ref = 100)
starting_point <- c(z = 10, D_R = 1)

## Call the fitting function
iso_fit <- fit_isothermal_inactivation(model_name,
                                       isothermal_inactivation, starting_point,
                                       known_params)

## Output of the results
plot(iso_fit)

## END EXAMPLE 1
```

---

### Description

Provides information of the models implemented for fitting of isothermal data. This models are
valid only for isothermal adjustment with the function `fit_isothermal_inactivation`. To make
predictions with the function `predict_inactivation` or adjust dynamic experiments with `fit_dynamic_inactivation`,
use `get_model_data`.

### Usage

```r
get_isothermal_model_data(model_name = "valids")
```

### Arguments

- `model_name` Optional string with the key of the model to use.
The function `get_model_data` provides information about the function for dynamic predictions associated with a valid `simulation_model` key. If `simulation_model` is missing or `NULL`, a character vector of valid model keys is provided. This function is designed as an assistant for using the functions `predict_inactivation` and `fit_dynamic_inactivation`. For the adjustment of isothermal experiments with the function `fit_isothermal_inactivation`, use the function `get_isothermal_model_data`.

**Usage**

```r
get_model_data(simulation_model = NULL)
```

**Arguments**

- `simulation_model` (optional) character with a valid model key or `NULL`.

**Value**

If `simulation_model` is `NULL` or missing, a character vector of possible names. Otherwise, a list including information of the relevant function:

- `ode`: Pointer to the function defining the model ode.
- `cost`: Pointer to the function calculating the error of the approximation.
- `dtemp`: logical defining whether the function requires the definition of the first derivative of temperature.
- `variables`: a character vector defining which entry variables are needed by the model.
- `variables_priv`: for internal use only.
- `parameters`: character vector with the parameters needed by the model.

**See Also**

`predict_inactivation`, `fit_dynamic_inactivation`
**get_prediction_cost**  
*Error of the Prediction of Microbial Inactivation*

**Description**
Calculates the error of the prediction of microbial inactivation for the chosen inactivation model and the given parameters with respect to the experimental data provided. This function is compatible with the function `fit_dynamic_inactivation`.

**Usage**
```r
get_prediction_cost(data_for_fit, temp_profile, simulation_model, P, known_params)
```

**Arguments**
- `data_for_fit`  
  A data frame with the experimental data to fit. It must contain a column named “time” and another one named “N”.
- `temp_profile`  
  data.frame defining the temperature profile. It must have a column named “time” and another named “temperature”.
- `simulation_model`  
  character key defining the inactivation model.
- `P`  
  list with the unknown parameters of the model to be adjusted.
- `known_params`  
  list with the parameters of the model fixed (i.e., not adjusted)

**Value**
An instance of `modCost` with the error of the prediction.

**See Also**
- `modCost`, `fit_dynamic_inactivation`

---

**is.FitInactivation**  
*Test of FitInactivation object*

**Description**
Tests if an object is of class FitInactivation.

**Usage**
```r
is.FitInactivation(x)
```
### is.FitInactivationMCMC

Test of FitInactivationMCMC object

**Description**
Tests if an object is of class FitInactivationMCMC.

**Usage**

```r
is.FitInactivationMCMC(x)
```

**Arguments**

- `x`  
  object to be checked.

**Value**

A logic specifying whether `x` is of class `FitInactivationMCMC`.

---

### is.Isofitinactivation

Test of IsoFitInactivation object

**Description**
Tests if an object is of class IsoFitInactivation.

**Usage**

```r
is.Isofitinactivation(x)
```

**Arguments**

- `x`  
  object to be checked.

**Value**

A logic specifying whether `x` is of class `IsoFitInactivation`. 
is.PredInactivationMCMC

Test of PredInactivationMCMC object

Description
Tests if an object is of class PredInactivationMCMC.

Usage
is.PredInactivationMCMC(x)

Arguments
x object to be checked.

Value
A logic specifying whether x is of class PredInactivationMCMC

is.SimulInactivation
Test of SimulInactivation object

Description
Tests if an object is of class SimulInactivation.

Usage
is.SimulInactivation(x)

Arguments
x object to be checked.

Value
A logic specifying whether x is of class SimulInactivation
isothermal_inactivation

Example Isothermal Inactivation of a Microorganism

Description

Example of experimental data for an isothermal process of a microorganism.

Usage

data(isothermal_inactivation)

Format

A data frame with 36 rows and 3 variables.

Details

• time: Time in minutes of the measurement.
• temp: Temperature at which the experiment was made.
• log_diff: Logarithmic difference.

laterosporus_dyn

Example Dynamic Inactivation of a Laterosporus

Description

Example of experimental data of the dynamic inactivation process of Laterosporus.

Usage

data(laterosporus_dyn)

Format

A data frame with 20 rows and 3 variables.

Details

• time: Time in minutes of the measurement.
• temp: observed temperature.
• logN: recorded number of microorganism.
Example Isothermal Inactivation of a Laterosporus

Description

Example of experimental data for an isothermal process of Laterosporus.

Usage

data(laterosporus_iso)

Format

A data frame with 52 rows and 3 variables.

Details

- time: Time in minutes of the measurement.
- temp: Temperature at which the experiment was made.
- log_diff: Logarithmic difference.

Plot of FitInactivation Object

Description

Plots a comparison between the experimental data provided and the prediction produced by the model parameters adjusted for an instance of `fitinactivation`.

Usage

```r
## S3 method for class 'FitInactivation'
plot(x, y = NULL, ..., make_gg = TRUE,
     plot_temp = FALSE, label_y1 = "logN", label_y2 = "Temperature",
     ylims = NULL)
```

Arguments

- `x`: the object of class `FitInactivation` to plot.
- `y`: ignored
- `...`: additional arguments passed to `plot`.
- `make_gg`: logical. If TRUE, the plot is created using `ggplot2`. Otherwise, the plot is created with base R. TRUE by default.
**plot.FitInactivationMCMC**

`plot_temp` logical. Whether the temperature profile will be added to the plot. `FALSE` by default.

`label_y1` Label of the principal y-axis.

`label_y2` Label of the secondary y-axis.

`ylims` Numeric vector of length 2 with the limits of the y-axis. `NULL` by default (0, `max_count`).

**Value**

If `make_gg = FALSE`, the plot is created. Otherwise, an instance of `ggplot` is generated, printed and returned.

---

**plot.FitInactivationMCMC**

*Plot of FitInactivationMCMC Object*

**Description**

Plots a comparison between the experimental data provided and the prediction produced by the model parameters adjusted for an instance of `FitInactivationMCMC`.

**Usage**

```r
## S3 method for class 'FitInactivationMCMC'
plot(x, y = NULL, ..., make_gg = TRUE,
     plot_temp = FALSE, label_y1 = "logN", label_y2 = "Temperature",
     ylims = NULL)
```

**Arguments**

- `x` the object of class `FitInactivation` to plot.
- `y` ignored
- `...` additional arguments passed to `plot`.
- `make_gg` logical. If `TRUE`, the plot is created using `ggplot2`. Otherwise, the plot is created with base R. `TRUE` by default.
- `plot_temp` logical. Whether the temperature profile will be added to the plot. `FALSE` by default.
- `label_y1` Label of the principal y-axis.
- `label_y2` Label of the secondary y-axis.
- `ylims` Numeric vector of length 2 with the limits of the y-axis. `NULL` by default (0, `max_count`).

**Value**

If `make_gg = FALSE`, the plot is created. Otherwise, an instance of `ggplot` is generated, printed and returned.
plot.IsotInactivation

Plot of IsoFitInactivation Object

Description
For each one of the temperatures studied, plots a comparison between the predicted result and the experimental one for an instance of IsoFitInactivation.

Usage
```r
## S3 method for class 'IsoFitInactivation'
plot(x, y = NULL, ..., make_gg = FALSE)
```

Arguments
- `x`: the object of class IsoFitInactivation to plot.
- `y`: ignored
- `...`: additional arguments passed to `plot`.
- `make_gg`: logical. If TRUE, the plot is created using ggplot2. Otherwise, the plot is created with base R. FALSE by default.

plot.PredInactivationMCMC

Plot of PredInactivationMCMC Object

Description
Plots the prediction interval generated by `predict_inactivation_MCMC`.

Usage
```r
## S3 method for class 'PredInactivationMCMC'
plot(x, y = NULL, ..., make_gg = TRUE)
```

Arguments
- `x`: the object of class PredInactivationMCMC to plot.
- `y`: ignored
- `...`: additional arguments passed to `plot`.
- `make_gg`: logical. If TRUE, the plot is created using ggplot2. Otherwise, the plot is created with base R. TRUE by default.
plot.SimulInactivation

Details
The plot generated in ggplot (default) generates a dashed line with the mean of the MC simulations. Moreover, a ribbon with the 2 first quantiles (i.e. columns 3 and 4) is generated.

The plot generated with base R (make_gg = FALSE) generates a solid line with the mean of the MC simulations. Each one of the other quantiles included in the data frame are added with different

Value
If make_gg = FALSE, the plot is created. Otherwise, an instance of ggplot is generated, printed and returned.

plot.SimulInactivation

Plot of SimulInactivation Object

Description
Plots the predicted evolution of the logarithmic count with time for an instance of SimulInactivation.

Usage
## S3 method for class 'SimulInactivation'
plot(x, y = NULL, ..., make_gg = TRUE,
     plot_temp = FALSE, label_y1 = "logN", label_y2 = "Temperature",
     ylims = NULL)

Arguments
- x: The object of class SimulInactivation to plot.
- y: ignored
- ...: additional arguments passed to plot.
- make_gg: logical. If TRUE, the plot is created using ggplot2. Otherwise, the plot is created with base R. TRUE by default.
- plot_temp: logical. Whether the temperature profile will be added to the plot. FALSE by default.
- label_y1: Label of the principal y-axis.
- label_y2: Label of the secondary y-axis.
- ylims: Numeric vector of length 2 with the Limits of the y-axis. NULL by default (0, max_count).

Value
If make_gg = FALSE, the plot is created. Otherwise, an instance of ggplot is generated, printed and returned.
predict_inactivation  *Prediction of Dynamic Inactivation*

**Description**

Predicts the inactivation of a microorganism under isothermal or non-isothermal temperature conditions. The thermal resistance of the microorganism are defined with the input arguments.

**Usage**

```r
predict_inactivation(simulation_model, times, parms, temp_profile, ..., 
tol0 = 1e-05)
```

**Arguments**

- `simulation_model` character identifying the model to be used.
- `times` numeric vector of output times.
- `parms` list of parameters defining the parameters of the model.
- `temp_profile` data frame with discrete values of the temperature for each time. It must have one column named `time` and another named `temperature` providing discrete values of the temperature at time points.
- `...` Additional arguments passed to `ode`.
- `tol0` numeric. Observations at time 0 make Weibull-based models singular. The time for observations taken at time 0 are changed for this value. By default (`tol0 = 1e-5`)

**Details**

The value of the temperature is calculated at each value of time by linear interpolation of the values provided by the input argument `temp_profile`. The function `ode` of the package `deSolve` is used for the resolution of the differential equation.

**Value**

A list of class `SimulInactivation` with the results. It has the following entries:

- `model`: character defining the model use for the prediction.
- `model_parameters`: named numeric vector with the values of the model parameters used.
- `temp_approximations`: function used for the interpolation of the temperature. For a numeric value of time given, returns the value of the temperature and its first derivative.
- `simulation`: A data frame with the results calculated. Its first column contains the times at which the solution has been calculated. The following columns the values of the variables of the model. The three last columns provide the values of logN, S and logS.
predict_inactivation_MCMC

See Also
ode, get_model_data

Examples

```r
## EXAMPLE 1

## Retrieve the model keys available for dynamic models.
get_model_data()

## Set the input arguments
example_model <- "Geeraerd" # Geeraerd’s model will be used
times <- seq(0, 5, length=100) # values of time for output

model_data <- get_model_data(example_model) # Retrieve the data of the model used
print(model_data$parameters)
print(model_data$variables)
model_parms <- c(D_R = 1,
                 z = 10,
                 N_min = 100,
                 temp_ref = 100,
                 N0 = 100000,
                 C_c0 = 1000)

## Define the temperature profile for the prediction
temperature_profile <- data.frame(time = c(0, 1.25, 2.25, 4.6),
                                    temperature = c(70, 105, 105, 70))

## Call the prediction function
prediction_results <- predict_inactivation(example_model, times,
                                          model_parms, temperature_profile)

## Show the results
head(prediction_results$simulation)
plot(prediction_results)

## END EXAMPLE 1
```

predict_inactivation_MCMC

Dynamic Prediction Intervals from a Monte Carlo Adjustment

Description

Given a model adjustment of a dynamic microbial inactivation process performed using any of the functions in bioinactivation calculates probability intervals at each time point using a Monte Carlo method.
Usage

predict_inactivation_MCMC(fit_object, temp_profile, n_simulations = 100, times = NULL, quantiles = c(2.5, 97.5), additional_pars = NULL)

Arguments

fit_object An object of classes FitInactivationMCMC, IsoFitInactivation or FitInactivation.

temp_profile data frame with discrete values of the temperature for each time. It must have one column named time and another named temperature providing discrete values of the temperature at time points.

n_simulations a numeric indicating how many Monte Carlo simulations to perform. 100 by default.

times numeric vector specifying the time points when results are desired. If NULL, the times in MCMC_fit$best_prediction are used. NULL by default.

quantiles numeric vector indicating the quantiles to calculate in percentage. By default, it is set to c(2.5, 97.5) which generates a prediction interval with confidence 0.95. If NULL, the quantiles are not calculated and all the simulations are returned.

additional_pars Additional parameters not included in the adjustment (e.g. the initial number of microorganism in an isothermal fit).

Value

A data frame of class PredInactivationMCMC. On its first column, time at which the calculation has been made is indicated. If quantiles = NULL, the following columns contain the results of each simulation. Otherwise, the second and third columns provide the mean and median of the simulations at the given time point. Following columns contain the quantiles of the results.

sample_dynafit Random sample of the parameters of a FitInactivation object

Description

Function to be called by predict_inactivation_MCMC. Produces a random sample of the parameters adjusted from dynamic experiments with non linear regression.

Usage

sample_dynafit(dynamic_fit, times, n_simulations)

Arguments

dynamic_fit An object of class FitInactivationMCMC as generated by fit_inactivation_MCMC.

times numeric vector specifying the time points when results are desired. If NULL, the times in MCMC_fit$best_prediction are used.

n_simulations a numeric indicating how many Monte Carlo simulations to perform.
**sample_IsoFit**

**Details**

It is assumed that the parameters follow a Multivariate Normal distribution with the mean the parameters estimated by `modFit`. The unscaled covariance matrix returned by `modFit` is used.

The function produces a random sample using the function `mvrnorm`.

**Value**

Returns a list with 4 components.

- `par_sample`: data frame with the parameter sample.
- `simulation_model`: model key for the simulation
- `known_pars`: parameters of the model known
- `times`: points where the calculation is made.

---

**sample_IsoFit**

*Random sample of the parameters of a IsoFitInactivation object*

**Description**

Function to be called by `predict_inactivation_MCMC`. Produces a random sample of the parameters adjusted from isothermal experiments.

**Usage**

```r
sample_IsoFit(iso_fit, times, n_simulations)
```

**Arguments**

- `iso_fit` An object of class `FitInactivationMCMC` as generated by `fit_inactivation_MCMC`.
- `times` numeric vector specifying the time points when results are desired. If `NULL`, an equispaced interval between 0 and the maximum time of the observations with length 50 is used.
- `n_simulations` a numeric indicating how many Monte Carlo simulations to perform.

**Details**

It is assumed that the parameters follow a Multivariate Normal distribution with the mean and covariance matrix estimated by the adjustment. The function produces a random sample using the function `mvrnorm`.

**Value**

Returns a list with 4 components.

- `par_sample`: data frame with the parameter sample.
- `simulation_model`: model key for the simulation
- `known_pars`: parameters of the model known
- `times`: points where the calculation is made.
See Also
mvrnorm

sample_MCMCfit  Random sample of the parameters of a FitInactivationMCMC object

Description
Function to be called by predict_inactivation_MCMC. Produces a random sample of the parameters calculated on the iterations of the Monte Carlo simulation.

Usage
sample_MCMCfit(MCMC_fit, times, n_simulations)

Arguments
MCMC_fit  An object of class FitInactivationMCMC as generated by fit_inactivation_MCMC.
times  numeric vector specifying the time points when results are desired. If NULL, the times in MCMC_fit$best_prediction are used.
n_simulations  a numeric indicating how many Monte Carlo simulations to perform.

Value
Returns a list with 4 components.
  • par_sample: data frame with the parameter sample.
  • simulation_model: model key for the simulation
  • known_pars: parameters of the model known
  • times: points where the calculation is made.

summary.FitInactivation
Summary of a FitInactivation object

Description
Summary of a FitInactivation object

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

summary.FitInactivationMCMC

Arguments

object Instance of Fit Inactivation
... ignored

Description

Summary of a FitInactivationMCMC object

Usage

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

Arguments

object Instance of FitInactivationMCMC
... ignored

summary.IsoFitInactivation

Summary of a IsoFitInactivation object

Description

Summary of a IsoFitInactivation object

Usage

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

Arguments

object Instance of IsoFitInactivation
... ignored
WeibullMafart_iso  Isothermal Weibull-Mafart Model

Description
Returns the predicted logarithmic reduction in microbial count according to Weibull-Mafarts’s model for the time, temperature and model parameters given.

Usage
WeibullMafart_iso(time, temp, delta_ref, z, p, temp_ref)

Arguments
- **time**: numeric indicating the time at which the prediction is taken.
- **temp**: numeric indicating the temperature of the treatment.
- **delta_ref**: numeric defining the delta-value at the reference temperature.
- **z**: numeric defining the z-value.
- **p**: numeric defining shape factor of the Weibull distribution.
- **temp_ref**: numeric indicating the reference temperature.

Value
A numeric with the predicted logarithmic reduction \(\log_{10}(N/N_0)\).

WeibullPeleg_iso  Isothermal Weibull-Peleg Model

Description
Returns the predicted logarithmic reduction in microbial count according to Weibull-Peleg’s model for the time, temperature and model parameters given.

Usage
WeibullPeleg_iso(time, temp, n, k_b, temp_crit)

Arguments
- **time**: numeric indicating the time at which the prediction is taken.
- **temp**: numeric indicating the temperature of the treatment.
- **n**: numeric defining shape factor of the Weibull distribution.
- **k_b**: numeric indicating the slope of the b-temp line.
- **temp_crit**: numeric with the value of the critical temperature.
WeibullPeleg_iso

**Value**

A numeric with the predicted logarithmic reduction \(\log_{10}(N/N_0)\).
*Topic datasets
  dynamic_inactivation, 11
  isothermal_inactivation, 21
  laterosporus_dyna, 21
  laterosporus_iso, 22

approxfun, 4
Arrhenius_iso, 2
Bigelow_iso, 3
build_temperature_interpolator, 4
check_model_params, 4
dArrhenius_model, 5
dBigelow_model, 6
deSolve, 26
dGeeraerd_model, 7
dMafart_model, 8
dPeleg_model, 10
dynamic_inactivation, 11

fit_dynamic_inactivation, 11, 16–18
fit_inactivation_MCMC, 13, 28–30
fit_isothermal_inactivation, 15, 16, 17
FME, 11, 13

get_isothermal_model_data, 16, 17
get_model_data, 16, 17, 27
get_prediction_cost, 18

is.FitInactivation, 18
is.FitInactivationMCMC, 19
is.IsotInactivation, 19
is.PredInactivationMCMC, 20
is.SimulInactivation, 20
isothermal_inactivation, 21

laterosporus_dyna, 21
laterosporus_iso, 22

modCost, 18

modFit, 11, 12, 29
modMCMC, 13, 14
mvrnorm, 29, 30

nls, 15

ode, 26, 27

plot.FitInactivation, 22
plot.FitInactivationMCMC, 23
plot.IsotInactivation, 24
plot.PredInactivationMCMC, 24
plot.SimulInactivation, 25
predict_inactivation, 5–11, 16, 17, 26
predict_inactivation_MCMC, 24, 27, 28–30

sample_dynaFit, 28
sample_IsotFit, 29
sample_MCMCfit, 30
summary.FitInactivation, 30
summary.FitInactivationMCMC, 31
summary.IsotInactivation, 31

WeibullMafart_iso, 32
WeibullPeleg_iso, 32