Package 'AccelStab'

October 21, 2024

Title Accelerated Stability Kinetic Modelling

Version 2.0.2

Description
Estimate the Šesták–Berggren kinetic model (degradation model) from experimental data. A A closed-form (analytic) solution to the degradation model is implemented as a non-linear fit, allowing for the extrapolation of the degradation of a drug product - both in time and temperature. Parametric bootstrap, with kinetic parameters drawn from the multivariate t-distribution, and analytical formulae (the delta method) are available options to calculate the confidence and prediction intervals.
The results (modelling, extrapolations and statistical intervals) can be visualised with multiple plots. The examples illustrate the accelerated stability modelling in drugs and vaccines development.

License AGPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.1

Imports dplyr, ggplot2, minpack.lm, mvtnorm, stats, scales

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

Depends R (>= 4.1.0)

Config/testthat/edition 3

LazyData true

Language en-GB

URL https://github.com/AccelStab/AccelStab

BugReports https://github.com/AccelStab/AccelStab/issues

NeedsCompilation no

Author Bernard G Franq [aut, cph],
Ben Wells [cre],
Alex Ball [ctb]

Maintainer Ben Wells <benwells@bayezian.com>

Repository CRAN

Date/Publication 2024-10-21 12:50:07 UTC

Contents

`antigenicity` *Antigenicity Accelerated Stability Data*

Description

An example dataset containing antigenicity concentration data at different temperatures over a period of up to 147 days.

Usage

```
data(antigenicity)
```

Format

An object of class "data.frame" with 50 rows and 5 variables

time Number of days in years for which the datapoints are gathered.

conc The concentration at a time.

K The temperature in Kelvin.

Celsius The temperature in celsius.

days Number of days for which the datapoints are gathered.

excursion *Temperature Excursion*

Description

Predict a temperature excursion for a product.

Usage

```
excursion(
  step1_down_object,
  temp_changes,
  time_changes,
  CI = TRUE,
  PI = TRUE,
  draw = 10000,
  confidence_interval = 0.95,
  intercept = NULL,
  ribbon = TRUE,
  xname = NULL,
  yname = NULL,
  plot_simulations = FALSE
)
```

Arguments

`step1_down_object`
The fit object from the step1.down function (required).

`temp_changes` A list that represents the order of the temperatures that the product is subjected to. Must be the same length as time_changes.

`time_changes` List that represents the times at which the temperature changes, Starts from time zero and must be the same length as temp_changes.

`CI` Show confidence intervals.

`PI` Show prediction intervals.

`draw` Number of simulations used to estimate confidence intervals.

`confidence_interval`
Confidence level for the confidence and prediction intervals around the predictions (default 0.95).

`intercept` Use a forced y-intercept. If null, the fitted value will be used.

`ribbon` Add shade to confidence and prediction intervals (optional).

`xname` Label for the x-axis (optional).

`yname` Label for the y-axis (optional).

`plot_simulations`
If TRUE, randomly selects 100 of the simulations to display on the plot.

Details

Use the output from step1.down to run a temperature excursion prediction.

Value

An SB class object, a list including the following elements:

- *prediction* - A data frame containing the predictions with the confidence and prediction intervals.
- *simulations* - Matrix of the simulations.
- *excursion plot* - A plot with predictions and statistical intervals.
- *user_parameters* - List of users input parameters which is utilised by other functions in the package.

Examples

```
#load antigenicity
data(antigenicity)

#run step1.down fit
fit1 <- step1_down(data = antigenicity, y = "conc", .time = "time",
 C = "Celsius", max_time_pred = 3)

#run excursion function with fixed intercept.
excursion <- excursion(step1_down_object = fit1,
                       temp_changes = c(5,15,10),
                       time_changes = c(0.5,1.5,3),
                       CI = TRUE, PI = TRUE, draw = 4000,
                       confidence_interval = 0.95,
                       intercept = 80,
                       xname = "Time in years", yname = "Concentration",
                       ribbon = TRUE, plot_simulations = TRUE)

excursion$excursion_plot
```

---

potency *Potency Accelerated Stability Data*

Description

An example dataset containing potency data at different temperatures..

Usage

```
data(potency)
```

Format

An object of class `"data.frame"` with 78 rows and 3 variables

Time Time for which the datapoints are gathered.

Potency Measured potency at a time.

Celsius The temperature in celsius.

`step1_down` *Step1 Down Model*

Description

Fit the one-step Šesták–Berggren kinetic model.

Usage

```
step1_down(
  data,
  y,
  .time,
  K = NULL,
  C = NULL,
  validation = NULL,
  draw = 10000,
  parms = NULL,
  temp_pred_C = NULL,
  max_time_pred = NULL,
  confidence_interval = 0.95,
  by = 101,
  reparameterisation = FALSE,
  zero_order = FALSE
)
```

Arguments

`data` Dataframe containing accelerated stability data (required).

`y` Name of decreasing variable (e.g. concentration) contained within data (required).

`.time` Time variable contained within data (required).

`K` Kelvin variable (numeric or column name) (optional).

`C` Celsius variable (numeric or column name) (optional).

`validation` Validation dummy variable, the column must contain only 1s and 0s, 1 for validation data and 0 for fit data. (column name) (optional).

`draw` Number of simulations used to estimate confidence intervals. When set to NULL the calculus method is used, however this is not recommended.

`parms` Starting values for the parameters as a list - k1, k2, k3, and c0.

`temp_pred_C` Integer or numeric value to predict the response for a given temperature (in Celsius).

`max_time_pred` Maximum time to predict the response variable.

`confidence_interval`
Confidence level for the confidence and prediction intervals around the predictions (default 0.95).

`by` Number of points (on the time scale) to smooth the statistical intervals around the predictions.

`reparameterisation`
Use alternative parameterisation of the one-step model which aims to reduce correlation between k1 and k2.

`zero_order` Set kinetic order, k3, to zero (straight lines).

Details

Fit the one-step Šesták–Berggren kinetic (non-linear) model using accelerated stability data from an R dataframe format. Parameters are kept in even when not significant.

Value

An SB class object, a list including the following elements:

- *fit* - The non-linear fit.
- *data* - The data set.
- *prediction* - A data frame containing the predictions with the confidence and prediction intervals.
- *user_parameters* - List of users input parameters which is utilised by other functions in the package.

Examples

```
#load antigenicity and potency data.
data(antigenicity)
data(potency)

#Basic use of the step1.down function with C column defined.
fit1 <- step1_down(data = antigenicity, y = "conc", .time = "time", C = "Celsius", draw = 5000)

#Basic use of the step1.down function with K column defined.
fit2 <- step1_down(data = antigenicity, y = "conc", .time = "time", K = "K", draw = 5000)

#When zero_order = FALSE, the output suggests using zero_order = TRUE for Potency dataset.
fit3 <- step1_down(data = potency, y = "Potency", .time = "Time",C = "Celsius",
  reparameterisation = FALSE, zero_order = TRUE, draw = 5000)

#reparameterisation is TRUE.
fit4 <- step1_down(data = antigenicity, y = "conc", .time = "time",C = "Celsius",
  reparameterisation = TRUE, draw = 5000)
```

step1_down_rmse *Step1 Down Model Root Mean Square Error Calculation*

Description

Calculate Root Mean Square Error (RMSE) for the one-step Šesták–Berggren kinetic model.

Usage

```
step1_down_rmse(
  data,
  y,
  .time,
  K = NULL,
  C = NULL,
  parms,
  reparameterisation = FALSE
)
```

Arguments

`data`	Dataframe containing accelerated stability data (required).
`y`	Name of decreasing variable (e.g. concentration) contained within data (required).
`.time`	Time variable contained within data (required).
`K`	Kelvin variable (numeric or column name) (optional).
`C`	Celsius variable (numeric or column name) (optional).
`parms`	Values for the parameters as a list - k1, k2, k3, and c0. If multiple are provided all combinations will be used (required).
`reparameterisation`	Use alternative parameterisation of the one-step model which aims to reduce correlation between k1 and k2.

Details

Calculate RMSE for the one-step Šesták–Berggren kinetic (non-linear) model using user provided parameters.

Value

A data frame containing one row for each RMSE calculation

Examples

```
#load antigenicity and potency data.
data(antigenicity)
data(potency)

#Basic use of the step1_down_rmse function with C column defined.
rmse1 <- step1_down_rmse(data = antigenicity, y = "conc", .time = "time",
 C = "Celsius", parms = list(c0 = c(96,98,100), k1 = c(42,45),
  k2 = c(12000,12500), k3 = c(8,9,10)))

#Basic use of the step1_down_rmse function with K column defined.
rmse2 <- step1_down_rmse(data = antigenicity, y = "conc", .time = "time",
 K = "K", parms = list(c0 = c(98), k1 = c(42,45), k2 = c(12500), k3 = c(8,9)))

#reparameterisation is TRUE.
rmse3 <- step1_down_rmse(data = antigenicity, y = "conc", .time = "time",
 C = "Celsius", parms = list(c0 = c(100,95), k1 = c(2,2.5), k2 = c(12000,13000),
  k3 = c(9,10)), reparameterisation = TRUE)
```

`step1_plot_CI` *Plot Confidence Intervals*

Description

Plot the stability data and visualise the predictions with confidence intervals.

Usage

```
step1_plot_CI(
  step1_down_object,
  xname = NULL,
  yname = NULL,
  xlim = NULL,
  ylim = NULL,
  ribbon = FALSE
)
```

Arguments

`step1_down_object`	The fit object from the step1.down function (required).
`xname`	Label for the x-axis (optional).
`yname`	Label for the y-axis (optional).
`xlim`	x-axis limits (optional).
`ylim`	y-axis limits (optional).
`ribbon`	Add shade to confidence intervals (optional).

Details

Use the fit object obtained from the step1.down function to plot the data and visualise the predictions with confidence intervals applied. There is an option to view the confidence intervals as a ribbon. The confidence interval value is chosen in the step1.down function.

Value

Plot of stability data with prediction curves and confidence intervals.

Examples

```
#load antigenciity data
data(antigenicity)

#run step1.down fit
fit1 <- step1_down(data = antigenicity, y = "conc", .time = "time",
 C = "Celsius", max_time_pred = 3, confidence_interval = 0.9)

#plot raw data with prediction curves and confidence intervals.
step1_plot_CI(step1_down_object = fit1, xlim = NULL, ylim = NULL,
 xname = "Time (Years)", yname = "Concentration", ribbon = TRUE)
```

---

step1_plot_desc *Plot Stability Data*

Description

Plot raw accelerated stability data.

Usage

```
step1_plot_desc(
  data,
  y,
  .time,
  K = NULL,
  C = NULL,
  validation = NULL,
  xname = NULL,
  yname = NULL,
  xlim = NULL,
  ylim = NULL
)
```

Arguments

`data`	Dataframe containing accelerated stability data.
`y`	Name of decreasing variable (e.g. concentration) contained within data
`.time`	Time variable contained within data.
`K`	Kelvin variable (numeric or column name) (optional).
`C`	Celsius variable (numeric or column name) (optional).
`validation`	Validation dummy variable (column name) (optional).
`xname`	Label for the x-axis (optional).
`yname`	Label for the y-axis (optional).
`xlim`	x-axis limits (optional).
`ylim`	y-axis limits (optional).

Details

Plot the raw accelerated stability data by selecting the columns - response, time and temperature.

Value

Plot of raw accelerated stability data.

Examples

```
#load example datasets
data(antigenicity)
data(potency)

step1_plot_desc(data=antigenicity, y="conc", .time="time", C = "Celsius")

step1_plot_desc(data=potency, y="Potency", .time="Time", C = "Celsius")
```

`step1_plot_diagnostic` *Create Diagnostic Plots*

Description

Generate residual diagnostic plots from a step1_down fit.

Usage

```
step1_plot_diagnostic(step1_down_object, bins = 7)
```

Arguments

`step1_down_object`
The fit object from the step1_down function (required).

`bins` The number of bins in the Histogram plot (default 7).

Details

Use the fit object obtained from the step1_down function to plot the residual diagnostic plots, assess the quality of fit and search for anomalies. Plots created are: Residuals Histogram, Observed Vs Predicted results, Residuals Vs Predicted results and QQplot of Residuals.

Value

A list containing the four ggplot2 plots.

Examples

```
#load antigenicity data
data(antigenicity)

#run step1_down fit
fit1 <- step1_down(data = antigenicity, y = "conc", .time = "time",
 C = "Celsius", max_time_pred = 3)

#plot diagnostic plots to asses the fit
step1_plot_diagnostic(fit1)
```

`step1_plot_PI` *Plot Prediction Intervals*

Description

Plot the stability data and visualise the predictions with prediction intervals.

Usage

```
step1_plot_PI(
  step1_down_object,
  xname = NULL,
  yname = NULL,
  xlim = NULL,
  ylim = NULL,
  ribbon = FALSE
)
```

Arguments

step1_down_object
: The fit object from the step1.down function (required).

xname	Label for the x-axis (optional).
yname	Label for the y-axis (optional).
xlim	x-axis limits (optional).
ylim	y-axis limits (optional).
ribbon	Add shade to prediction intervals (optional).

Details

Use the fit object obtained from the step1.down function to plot the stability data and visualise the predictions with prediction intervals applied. There is an option to view the prediction intervals as a ribbon. The prediction interval value is chosen in the step1.down function.

Value

Plot of stability data with prediction curves and prediction intervals.

Examples

```
#load antigenicity data
data(antigenicity)

#run step1.down fit
fit1 <- step1_down(data = antigenicity, y = "conc", .time = "time",
 C = "Celsius", max_time_pred = 3)

#plot raw data with prediction curves and prediction intervals.
step1_plot_PI(step1_down_object = fit1, xlim = NULL, ylim = NULL,
 xname = "Time (Years)", yname = "Concentration", ribbon = TRUE)
```

step1_plot_pred *Plot Model Predictions*

Description

Plot the stability data and visualise the predictions.

Usage

```
step1_plot_pred(
  step1_down_object,
  xname = NULL,
  yname = NULL,
  xlim = NULL,
  ylim = NULL
)
```

Arguments

`step1_down_object`
: The fit object from the step1.down function (required).

xname	Label for the x-axis (optional).
yname	Label for the y-axis (optional).
xlim	x-axis limits (optional).
ylim	y-axis limits (optional).

Details

Use the fit object from the step1.down function to plot the accelerated stability data and visualise the predictions.

Value

Plot of accelerated stability data with prediction curves.

Examples

```
#load antigenicity data
data(antigenicity)

fit1 <- step1_down(data = antigenicity, y = "conc", .time = "time",
 C = "Celsius", max_time_pred = 3)

step1_plot_pred(step1_down_object = fit1, xlim = NULL, ylim = NULL,
 xname = "Time (Years)", yname = "Concentration")
```

---

step1_plot_T *Focus on Temperature*

Description

Plot the stability data and visualise the predictions with focus on one temperature.

Usage

```
step1_plot_T(
  step1_down_object,
  focus_T = NULL,
  xname = NULL,
  yname = NULL,
  xlim = NULL,
  ylim = NULL,
  ribbon = FALSE
)
```

Arguments

`step1_down_object` The fit object from the step1.down function (required).

`focus_T` Selected temperature to highlight on the plot.

`xname` Label for the x-axis (optional).

`yname` Label for the y-axis (optional).

`xlim` the x-axis limits (optional).

`ylim` the y-axis limits (optional).

`ribbon` adds shade to confidence and prediction intervals (optional).

Details

Plot the stability data and visualise the predictions focusing on one chosen temperature with confidence and prediction intervals.

Value

ggplot2 object with focus on chosen temperature.

Examples

```
#load potency data
data(potency)

#run step1_down fit
fit1 <- step1_down(data = potency, y = "Potency", .time = "Time",
 C = "Celsius", zero_order = TRUE)

#plot raw data with prediction curves with focus on temperature in dataset.
step1_plot_T(fit1, focus_T = 5,ribbon = TRUE, xlim = NULL, ylim = c(0,12),
 xname = "Time (Month)", yname = "Potency")

#plot raw data with prediction curves with focus on temperature not in dataset.
step1_plot_T(fit1, focus_T = -10,ribbon = TRUE, xlim = NULL, ylim = c(0,12),
 xname = "Time (Months)", yname = "Potency")
```

`step1_sample_mvt` *Sample the Multivariate t Distribution*

Description

Take a selected number of samples from the multivariate t distribution (mvt).

Usage

```
step1_sample_mvt(
  data,
  y,
  .time,
  K = NULL,
  C = NULL,
  validation = NULL,
  draw,
  parms = NULL,
  reparameterisation = FALSE,
  zero_order = FALSE
)
```

Arguments

`data` Dataframe containing accelerated stability data (required).

`y` Name of decreasing variable (e.g. concentration) contained within data (required).

`.time` Time variable contained within data (required).

`K` Kelvin variable (numeric or column name) (optional).

`C` Celsius variable (numeric or column name) (optional).

`validation` Validation dummy variable (column name) (optional).

`draw` Number of samples to draw from mvt (required).

`parms` Starting values for the parameters as a list - k1, k2, k3, and c0 (optional).

`reparameterisation` Use alternative parameterisation of the one-step model which aims to reduce correlation between k1 and k2.

`zero_order` Set kinetic order, k3, to zero (straight lines).

Details

Using the provided data the function creates a fit of the Šesták–Berggren kinetic model and then draws a selected number of samples from the mvt of the model parameters.

Value

A matrix containing parameter draws from the mvt distribution.

Examples

```
#load antigenicity data.
data(antigenicity)

#Basic use of the step1_sample_mvt function with C column defined and 1000 draws.
sample1 <- step1_sample_mvt(data = antigenicity, y = "conc", .time = "time",
 C = "Celsius", draw = 1000)
```

```
#Basic use of the step1_sample_mvt function with K column defined and 50000 draws
sample2 <- step1_sample_mvt(data = antigenicity, y = "conc", .time = "time",
 K = "K", draw = 50000)

#reparameterisation is TRUE and 10000 draws.
sample3 <- step1_sample_mvt(data = antigenicity, y = "conc", .time = "time",
C = "Celsius", reparameterisation = TRUE, draw = 10000)
```

Index