Package ‘breathteststan’

July 15, 2020

Type Package

Title Stan-Based Fit to Gastric Emptying Curves

Version 0.8.0

Description Stan-based curve-fitting function
for use with package 'breathtestcore' by the same author.
Stan functions are refactored here for easier testing.

License GPL (>= 3)

Encoding UTF-8

LazyData true

ByteCompile true

Depends R (>= 4.0), methods, Rcpp (>= 1.0.5)

Imports dplyr, purrr, rstan (>= 2.21.1), rstantools (>= 2.1.1), stringr, tibble, tidyr

Suggests ggplot2, shinystan, bayesplot, testthat, covr, knitr, rmarkdown, breathtestcore(>= 0.7.1.0)

LinkingTo StanHeaders, rstan, BH (>= 1.72), Rcpp, RcppEigen

URL `https://github.com/dmenne/breathteststan`

BugReports `https://github.com/dmenne/breathteststan/issues`

NeedsCompilation yes

SystemRequirements GNU make

RoxygenNote 7.1.1

Biarch true

Author Dieter Menne [aut, cre],
Menne Biomed Consulting Tuebingen [cph],
Benjamin Misselwitz [fnd],
Mark Fox [fnd],
University Hospital of Zurich, Dep. Gastroenterology [fnd, dtc]

Maintainer Dieter Menne <`dieter.menne@menne-biomed.de`>

Repository CRAN

Date/Publication 2020-07-14 23:00:02 UTC

R topics documented:

`sigma.breathteststanfit` *S3 method to exctract the residual standard deviation*

Description

Functions for S3 method defined in breathtestcore for `stan_fit` and `stan_group fit`.

Usage

```
## S3 method for class 'breathteststanfit'
sigma(object, ...)
```

Arguments

`object`	A Stan-based fit
`...`	Not used

Value

A numeric value giving the sigma (= average residual standard deviation) term from the Stan fit.

`stan_fit` *Bayesian Stan fit to 13C Breath Data*

Description

Fits exponential beta curves to 13C breath test series data using Bayesian Stan methods. See `https://menne-biomed.de/blog/breath-test-stan` for a comparision between single curve, mixed-model population and Bayesian methods.

Usage

```
stan_fit(
  data,
  dose = 100,
  sample_minutes = 15,
  student_t_df = 10,
  chains = 2,
  iter = 1000,
  model = "breath_test_1",
  seed = 4711
)
```

Arguments

`data` Data frame or tibble as created by `cleanup_data`, with mandatory columns `patient_id,group,minute` and `pdr`. It is recommended to run all data through `cleanup_data` which will insert dummy columns for `patient_id` and `minute` if the data are distinct, and report an error if not. Since the Bayesian method is stabilized by priors, it is possible to fit single curves.

`dose` Dose of acetate or octanoate. Currently, only one common dose for all records is supported.

`sample_minutes` If mean sampling interval is < sampleMinutes, data are subsampled using a spline algorithm

`student_t_df` When student_t_df < 10, the student distribution is used to model the residuals. Recommended values to model typical outliers are from 3 to 6. When student_t_df >= 10, the normal distribution is used.

`chains` Number of chains for Stan

`iter` Number of iterations for each Stan chain

`model` Name of model; use `names(stanmodels)` for other models.

`seed` Optional seed for rstan

Value

A list of classes "breathteststanfit" and "breathtestfit" with elements

- `coef` Estimated parameters as data frame in a key-value format with columns `patient_id,group,parameter,method` and `value`. Has an attribute AIC.
- `data` The effectively analyzed data. If density of points is too high, e.g. with BreathId devices, data are subsampled before fitting.
- `stan_fit` The Stan fit for use with `shinystan::launch_shiny` or extraction of chains.

See Also

Base methods `coef,plot,print`; methods from package `broom: tidy,augment`.

Examples

```
library(breathtestcore)
suppressPackageStartupMessages(library(dplyr))
d = breathtestcore::simulate_breathtest_data(n_records = 3) # default 3 records
data = breathtestcore::cleanup_data(d$data)
# Use more than 80 iterations and 4 chains for serious fits
fit = stan_fit(data, chains = 1, iter = 80)
plot(fit) # calls plot.breathtestfit
# Extract coefficients and compare these with those
# used to generate the data
options(digits = 2)
cf = coef(fit)
cf %>%
  filter(grepl("m|k|beta", parameter )) %>%
  select(-method, -group) %>%
  tidyr::spread(parameter, value) %>%
  inner_join(d$record, by = "patient_id") %>%
  select(patient_id, m_in = m.y, m_out = m.x,
         beta_in = beta.y, beta_out = beta.x,
         k_in = k.y, k_out = k.x)
# For a detailed analysis of the fit, use the shinystan library

library(shinystan)
# launch_shinystan(fit$stan_fit)

# The following plots are somewhat degenerate because
# of the few iterations in stan_fit
suppressPackageStartupMessages(library(rstan))
stan_plot(fit$stan_fit, pars = c("beta[1]","beta[2]","beta[3]"))
stan_plot(fit$stan_fit, pars = c("k[1]","k[2]","k[3]"))
stan_plot(fit$stan_fit, pars = c("m[1]","m[2]","m[3]"))
```

Index