Package 'breathteststan'

May 13, 2017

Type Package

Title Stan-Based Fit to Gastric Emptying Curves

Version 0.3.0

Date 2017-05-04

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

Depends R (>= 3.4.0), Rcpp (>= 0.12.10), methods

Imports rstan (>= 2.15.1), tibble, purrr, dplyr, stringr, tidyr,
breathtestcore

Suggests testthat, covr, knitr

LinkingTo StanHeaders (>= 2.15.0.1), rstan (>= 2.15.1), BH (>=
1.62.0.1), Rcpp (>= 0.12.10), RcppEigen (>= 0.3.3.3.0)

URL https://github.com/dmenne/breathteststshiny

NeedsCompilation yes

RoxygenNote 6.0.1

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 2017-05-13 13:36:49 UTC

R topics documented:

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")
```

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.

Value

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

- coef Estimated parameters as data frame in a key-value format with columns patient_id, group, parameter, metho 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 = simulate_breathtest_data(n_records = 3) # default 3 records
data = cleanup_data(d$data)
# Use more than 100 iterations and 4 chains for serious fits
# For execution on a local, multicore CPU with excess RAM we recommend
# calling \code{rstan_options(auto_write = TRUE)}.
fit = stan_fit(data, chains = 1, iter = 100)
plot(fit) # calls plot.breathtestfit
# Extract coefficients and compare these with those
# used to generate the data
options(digits = 2)
cf = coef(fit)
cf %>%
  filter(grepl("m|k|beta", parameter )) %>%
  select(-method, -group) %>%
  tidyr::spread(parameter, value) %>%
  inner_join(d$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
## Not run:
library(shinystan)
launch_shinystan(fit$stan_fit)

## End(Not run)
# 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