Package 'rstantools'

August 22, 2018

Type Package

Title Tools for Developing R Packages Interfacing with 'Stan'

Version 1.5.1

Date 2018-08-20

Maintainer Jonah Gabry <jsg2201@columbia.edu>

Description Provides various tools for developers of R packages interfacing with 'Stan' <http://mc-stan.org>, including functions to set up the required package structure, S3 generics and default methods to unify function naming across 'Stan'-based R packages, and vignettes with recommendations for developers.

License GPL (>= 3)

URL http://discourse.mc-stan.org/, http://mc-stan.org/rstantools/

BugReports https://github.com/stan-dev/rstantools/issues

Encoding UTF-8

LazyData true

SystemRequirements pandoc

Imports stats, utils

Suggests bayesplot (>= 1.5.0), rstan (>= 2.17.2), rstanarm (>= 2.17.4), shinystan (>= 2.4.0), loo (>= 2.0.0), testthat, covr, knitr, devtools, roxygen2 (>= 6.0.1), rmarkdown, rstudioapi, usethis (>= 1.3.0)

RoxygenNote 6.0.1

VignetteBuilder knitr

NeedsCompilation no

Author Jonah Gabry [aut, cre],
Ben Goodrich [aut],
Stefan Siegert [ctb],
Trustees of Columbia University [cph]

Repository CRAN

Date/Publication 2018-08-22 05:20:06 UTC

R topics documented:

`rstantools-package` *Tools for Developing R Packages Interfacing with Stan*

Description

The **rstantools** package provides various tools for developers of R packages interfacing with Stan (`http://mc-stan.org`), including functions to set up the required package structure, S3 generic methods to unify function naming across Stan-based R packages, and a vignette with guidelines for developers. To get started building a package see `rstan_package_skeleton`.

See Also

- Guidelines and recommendations for developers of R packages interfacing with Stan and a demonstration getting a simple package working can be found in the vignettes included with **rstantools** and at `http://mc-stan.org/rstantools/articles/`.

- After reading the guidelines for developers, if you have trouble setting up your package let us know on the the Stan Forums or at **rstantools** issue tracker.

- The useR2016 presentation How to Use (R)Stan to Estimate Models in External R Packages.

`bayes_R2` *Generic function and default method for Bayesian R-squared*

Description

Generic function and default method for Bayesian version of R-squared for regression models. See `bayes_R2.stanreg` in the **rstanarm** package for an example of defining a method.

Usage

```
bayes_R2(object, ...)

## Default S3 method:
bayes_R2(object, y, ...)
```

Arguments

`object`	The object to use.
`...`	Arguments passed to methods. See the methods in the **rstanarm** package for examples.
`y`	For the default method, a vector of y values the same length as the number of columns in the matrix used as `object`.

Value

`bayes_R2` methods should return a vector of length equal to the posterior sample size.

The default method just takes `object` to be a matrix of y-hat values (one column per observation, one row per posterior draw) and `y` to be a vector with length equal to `ncol(object)`.

See Also

- The **rstanarm** package for example methods (CRAN, GitHub).
- Guidelines and recommendations for developers of R packages interfacing with Stan and a demonstration getting a simple package working can be found in the vignettes included with **rstantools** and at `http://mc-stan.org/rstantools/articles/`.

init_cpp *Register functions implemented in C++*

Description

If you set up your package using rstan_package_skeleton before version 1.2.1 of **rstantools** it may be necessary for you to call this function yourself in order to pass R CMD check in R >= 3.4. If you used rstan_package_skeleton in **rstantools** version 1.2.1 or later then this has already been done automatically.

Usage

```
init_cpp(name, path)
```

Arguments

name	The name of your package as a string.
path	The path to the root directory for your package as a string. If not specified it is assumed that this is already the current working directory.

Value

This function is only called for its side effect of writing the necessary init.cpp file to the package's src/ directory.

log_lik *Generic function for pointwise log-likelihood*

Description

We define a new function log_lik rather than a logLik method because (in addition to the conceptual difference) the documentation for logLik states that the return value will be a single number, whereas log_lik returns a matrix. See log_lik.stanreg in the **rstanarm** package for an example.

Usage

```
log_lik(object, ...)
```

Arguments

object	The object to use.
...	Arguments passed to methods. See the methods in the **rstanarm** package for examples.

Value

log_lik methods should return a S by N matrix, where S is the size of the posterior sample (the number of draws from the posterior distribution) and N is the number of data points.

See Also

- The **rstanarm** package for example methods (CRAN, GitHub).
- Guidelines and recommendations for developers of R packages interfacing with Stan and a demonstration getting a simple package working can be found in the vignettes included with **rstantools** and at http://mc-stan.org/rstantools/articles/.

Examples

```
# See help("log_lik", package = "rstanarm")
```

---

loo-prediction *Generic functions for LOO predictions*

Description

See the methods in the **rstanarm** package for examples.

Usage

```
loo_linpred(object, ...)

loo_predict(object, ...)

loo_predictive_interval(object, ...)

loo_pit(object, ...)

## Default S3 method:
loo_pit(object, y, lw, ...)
```

Arguments

object	The object to use.
...	Arguments passed to methods. See the methods in the **rstanarm** package for examples.
y	For the default method of loo_pit, a vector of y values the same length as the number of columns in the matrix used as object.
lw	For the default method of loo_pit, a matrix of log-weights of the same length as the number of columns in the matrix used as object.

Value

`loo_predict`, `loo_linpred`, and `loo_pit` (probability integral transform) methods should return a vector with length equal to the number of observations in the data. `loo_predictive_interval` methods should return a two-column matrix formatted in the same way as for `predictive_interval`.

See Also

- The **rstanarm** package for example methods (CRAN, GitHub).
- Guidelines and recommendations for developers of R packages interfacing with Stan and a demonstration getting a simple package working can be found in the vignettes included with **rstantools** and at `http://mc-stan.org/rstantools/articles/`.

`nsamples` *Generic function for extracting the number of posterior samples*

Description

Extract the number of posterior samples stored in a fitted Bayesian model.

Usage

```
nsamples(object, ...)
```

Arguments

`object`	The object to use.
`...`	Arguments passed to methods. See the methods in the **rstanarm** package for examples.

`posterior_interval` *Generic function and default method for posterior uncertainty intervals*

Description

These intervals are often referred to as credible intervals, but we use the term uncertainty intervals to highlight the fact that wider intervals correspond to greater uncertainty. See `posterior_interval.stanreg` in the **rstanarm** package for an example.

Usage

```
posterior_interval(object, ...)

## Default S3 method:
posterior_interval(object, prob = 0.9, ...)
```

Arguments

`object`	The object to use.
`...`	Arguments passed to methods. See the methods in the **rstanarm** package for examples.
`prob`	A number $p \in (0, 1)$ indicating the desired probability mass to include in the intervals.

Value

`posterior_interval` methods should return a matrix with two columns and as many rows as model parameters (or a subset of parameters specified by the user). For a given value of `prob`, p, the columns correspond to the lower and upper $100p\%$ interval limits and have the names $100\alpha/2\%$ and $100(1-\alpha/2)\%$, where $\alpha = 1-p$. For example, if `prob=0.9` is specified (a 90% interval), then the column names would be `"5%"` and `"95%"`, respectively.

The default method just takes `object` to be a matrix (one column per parameter) and computes quantiles, with `prob` defaulting to 0.9.

See Also

- The **rstanarm** package for example methods (CRAN, GitHub).
- Guidelines and recommendations for developers of R packages interfacing with Stan and a demonstration getting a simple package working can be found in the vignettes included with **rstantools** and at http://mc-stan.org/rstantools/articles/.

Examples

```
# Default method takes a numeric matrix (of posterior draws)
draws <- matrix(rnorm(100 * 5), 100, 5) # fake draws
colnames(draws) <- paste0("theta_", 1:5)
posterior_interval(draws)

# Example using rstanarm package:
# posterior_interval method for 'stanreg' objects
## Not run:
if (require("rstanarm")) {
  fit <- stan_glmer(
    mpg ~ wt + am + (1|cyl),
    data = mtcars,
    QR = TRUE,
    prior = normal(0, 1),
    iter = 500 # to speed up example
  )
  rstanarm::posterior_interval(fit, prob = 0.5)
}

## End(Not run)

# Also see help("posterior_interval", package = "rstanarm")
```

posterior_linpred *Generic function for accessing the posterior distribution of the linear predictor*

Description

Extract the posterior draws of the linear predictor, possibly transformed by the inverse-link function. See `posterior_linpred.stanreg` in the **rstanarm** package for an example.

Usage

```
posterior_linpred(object, transform = FALSE, ...)
```

Arguments

`object`	The object to use.
`transform`	Should the linear predictor be transformed using the inverse-link function? The default is `FALSE`, in which case the untransformed linear predictor is returned.
`...`	Arguments passed to methods. See the methods in the **rstanarm** package for examples.

Value

`posterior_linpred` methods should return a D by N matrix, where D is the number of draws from the posterior distribution distribution and N is the number of data points.

See Also

- The **rstanarm** package for example methods (CRAN, GitHub).
- Guidelines and recommendations for developers of R packages interfacing with Stan and a demonstration getting a simple package working can be found in the vignettes included with **rstantools** and at `http://mc-stan.org/rstantools/articles/`.

Examples

```
# See help("posterior_linpred", package = "rstanarm")
```

posterior_predict *Generic function for drawing from the posterior predictive distribution*

Description

Draw from the posterior predictive distribution of the outcome. See `posterior_predict.stanreg` in the **rstanarm** package for an example.

Usage

```
posterior_predict(object, ...)
```

Arguments

`object`	The object to use.
`...`	Arguments passed to methods. See the methods in the **rstanarm** package for examples.

Value

`posterior_predict` methods should return a D by N matrix, where D is the number of draws from the posterior predictive distribution and N is the number of data points being predicted per draw.

See Also

- The **rstanarm** package for example methods (CRAN, GitHub).
- Guidelines and recommendations for developers of R packages interfacing with Stan and a demonstration getting a simple package working can be found in the vignettes included with **rstantools** and at `http://mc-stan.org/rstantools/articles/`.

Examples

```
# Example using rstanarm package:
# posterior_predict method for 'stanreg' objects

if (require("rstanarm")) {
  fit <- stan_glm(mpg ~ wt + am, data = mtcars)
  yrep <- posterior_predict(fit)
  all.equal(ncol(yrep), nobs(fit))

  nd <- data.frame(wt = mean(mtcars$wt), am = c(0, 1))
  ytilde <- posterior_predict(fit, newdata = nd)
  all.equal(ncol(ytilde), nrow(nd))
}


# Also see help("posterior_predict", package = "rstanarm")
```

predictive_error *Generic function and default method for predictive errors*

Description

Generic function and default method for computing predictive errors $y - y^{rep}$ (in-sample, for observed y) or $y - \tilde{y}$ (out-of-sample, for new or held-out y). See `predictive_error.stanreg` in the **rstanarm** package for an example.

Usage

```
predictive_error(object, ...)

## Default S3 method:
predictive_error(object, y, ...)
```

Arguments

`object`	The object to use.
`...`	Arguments passed to methods. See the methods in the **rstanarm** package for examples.
`y`	For the default method, a vector of y values the same length as the number of columns in the matrix used as `object`.

Value

`predictive_error` methods should return a D by N matrix, where D is the number of draws from the posterior predictive distribution and N is the number of data points being predicted per draw.

The default method just takes `object` to be a matrix and `y` to be a vector.

See Also

- The **rstanarm** package for example methods (CRAN, GitHub).
- Guidelines and recommendations for developers of R packages interfacing with Stan and a demonstration getting a simple package working can be found in the vignettes included with **rstantools** and at `http://mc-stan.org/rstantools/articles/`.

Examples

```
# default method
y <- rnorm(10)
ypred <- matrix(rnorm(500), 50, 10)
pred_errors <- predictive_error(ypred, y)
dim(pred_errors)
head(pred_errors)

# Also see help("predictive_error", package = "rstanarm")
```

predictive_interval *Generic function for predictive intervals*

Description

See predictive_interval.stanreg in the **rstanarm** package for an example.

Usage

```
predictive_interval(object, ...)

## Default S3 method:
predictive_interval(object, prob = 0.9, ...)
```

Arguments

object	The object to use.
...	Arguments passed to methods. See the methods in the **rstanarm** package for examples.
prob	A number $p \in (0, 1)$ indicating the desired probability mass to include in the intervals.

Value

predictive_interval methods should return a matrix with two columns and as many rows as data points being predicted. For a given value of prob, p, the columns correspond to the lower and upper $100p\%$ interval limits and have the names $100\alpha/2\%$ and $100(1 - \alpha/2)\%$, where $\alpha = 1 - p$. For example, if prob=0.9 is specified (a 90% interval), then the column names would be "5%" and "95%", respectively.

The default method just takes object to be a matrix and computes quantiles, with prob defaulting to 0.9.

See Also

- The **rstanarm** package for example methods (CRAN, GitHub).
- Guidelines and recommendations for developers of R packages interfacing with Stan and a demonstration getting a simple package working can be found in the vignettes included with **rstantools** and at http://mc-stan.org/rstantools/articles/.

Examples

```
# Default method takes a numeric matrix (of draws from posterior
# predictive distribution)
ytilde <- matrix(rnorm(100 * 5, sd = 2), 100, 5) # fake draws
predictive_interval(ytilde, prob = 0.8)
```

```
# Also see help("predictive_interval", package = "rstanarm")
```

`prior_summary` *Generic function for extracting information about prior distributions*

Description

See `prior_summary.stanreg` in the **rstanarm** package for an example.

Usage

```
prior_summary(object, ...)

## Default S3 method:
prior_summary(object, ...)
```

Arguments

`object`	The object to use.
`...`	Arguments passed to methods. See the methods in the **rstanarm** package for examples.

Value

`prior_summary` methods should return an object containing information about the prior distribution(s) used for the given model. The structure of this object will depend on the method.

The default method just returns `object$prior.info`, which is `NULL` if there is no `'prior.info'` element.

See Also

- The **rstanarm** package for example methods (CRAN, GitHub).
- Guidelines and recommendations for developers of R packages interfacing with Stan and a demonstration getting a simple package working can be found in the vignettes included with **rstantools** and at `http://mc-stan.org/rstantools/articles/`.

Examples

```
# See help("prior_summary", package = "rstanarm")
```

rstan_package_skeleton *Create the skeleton of a new* R *package with Stan programs*

Description

The `rstan_package_skeleton` function helps get you started developing R packages that interface with Stan via the **rstan** package. As of **rstantools** v1.5.0, `rstan_package_skeleton` calls `usethis::create_package` (instead of `utils::package.skeleton`) and then makes necessary adjustments so that the package can include Stan Programs that can be built into binary versions (i.e., pre-compiled like **rstanarm**).

See the **See Also** section below for links to recommendations for developers and a step by step walk-through of what to do after running `rstan_package_skeleton`.

Usage

```
rstan_package_skeleton(path, rstudio = TRUE, open = TRUE,
  stan_files = character(), travis = TRUE)
```

Arguments

`path`	A relative or absolute path to the new package to be created (terminating in the package name).
`rstudio, open`	See `usethis::create_package`.
`stan_files`	A character vector with paths to `.stan` files to include in the package (these files will be included in the `src/stan_files` directory). If not specified then the `.stan` files for the package can be manually placed into the appropriate directory later.
`travis`	Should a `.travis.yml` file be added to the package directory? Defaults to `TRUE`. The file has some settings already set to help with compilation issues, but we do not guarantee that it will work on travis-ci without manual adjustments.

Note

This function downloads several files from **rstanarm** package's GitHub repository to facilitate building the resulting package. Note that **rstanarm** is licensed under the GPL >= 3, so package builders who do not want to be governed by that license should not use the downloaded files that contain R code (that said, **Rcpp** is GPL, so not using the **rstanarm** files is not the only thing impeding use of other licenses). Otherwise, it may be worth considering whether it would be easier to include your `.stan` programs and supporting R code in the **rstanarm** package.

See Also

- The **rstanarm** repository on GitHub.

- Guidelines and recommendations for developers of R packages interfacing with Stan and a demonstration getting a simple package working can be found in the vignettes included with **rstantools** and at http://mc-stan.org/rstantools/articles/.

- After reading the guidelines for developers, if you have trouble setting up your package let us know on the the Stan Forums or at **rstantools** issue tracker.

- The useR2016 presentation How to Use (R)Stan to Estimate Models in External R Packages.

Index