

Package: `adjustr` (via `r-universe`)

August 20, 2024

Encoding UTF-8

Type Package

Title Stan Model Adjustments and Sensitivity Analyses using Importance Sampling

Version 0.1.2

Description Functions to help assess the sensitivity of a Bayesian model (fitted using the `rstan` package) to the specification of its likelihood and priors. Users provide a series of alternate sampling specifications, and the package uses Pareto-smoothed importance sampling to estimate posterior quantities of interest under each specification.

License BSD_3_clause + file LICENSE

Depends R (>= 3.6.0), `dplyr` (>= 1.0.0)

Imports `tidyselect`, `purrr`, `stringr`, `methods`, `utils`, `stats`, `rstan`, `rlang`, `ggplot2`, `loo`

Suggests `extraDistr`, `tidyr`, `testthat`, `covr`, `knitr`, `rmarkdown`

URL <https://corymccartan.github.io/adjustr/>

BugReports <https://github.com/CoryMcCartan/adjustr/issues>

LazyData true

RoxygenNote 7.1.1

VignetteBuilder `knitr`

Repository <https://corymccartan.r-universe.dev>

RemoteUrl <https://github.com/CoryMcCartan/adjustr>

RemoteRef v0.1.2

RemoteSha 2347cb57a9a050f4ed292734e3d89c96ff76cc66

Contents

adjustr	2
adjust_weights	3
as.data.frame.adjustr_spec	4
dplyr.adjustr_spec	5
extract_samp_stmts	6
get_resampling_idxes	6
make_spec	7
pull.adjustr_weighted	8
spec_plot	9
summarise.adjustr_weighted	10
Index	12

adjustr	<i>adjustr: Stan Model Adjustments and Sensitivity Analyses using Importance Sampling</i>
---------	---

Description

Functions to help assess the sensitivity of a Bayesian model to the specification of its likelihood and priors, estimated using the rstan package. Users provide a series of alternate sampling specifications, and the package uses Pareto-smoothed importance sampling to estimate posterior quantities of interest under each specification.

Details

See the list of key functions and the example below. Full package documentation available at <https://corymccartan.github.io/adjustr/>.

Key Functions

- [make_spec](#)
- [adjust_weights](#)
- [summarize](#)
- [plot](#)

adjust_weights	<i>Compute Pareto-smoothed Importance Weights for Alternative Model Specifications</i>
----------------	--

Description

Given a set of new sampling statements, which can be parametrized by a data frame or list, compute Pareto-smoothed importance weights and attach them to the specification object, for further calculation and plotting.

Usage

```
adjust_weights(spec, object, data = NULL, keep_bad = FALSE, incl_orig = TRUE)
```

Arguments

spec	An object of class <code>adjustr_spec</code> , probably produced by <code>make_spec</code> , containing the new sampling statements to replace their counterparts in the original Stan model, and the data, if any, by which these sampling statements are parametrized.
object	A model object, either of type <code>stanfit</code> , <code>stanreg</code> , <code>brmsfit</code> , or a list with two elements: <code>model</code> containing a <code>CmdStanModel</code> , and <code>fit</code> containing a <code>CmdStanMCMC</code> object.
data	The data that was used to fit the model in <code>object</code> . Required only if one of the new sampling specifications involves Stan data variables.
keep_bad	When <code>FALSE</code> (the strongly recommended default), alternate specifications which deviate too much from the original posterior, and which as a result cannot be reliably estimated using importance sampling (i.e., if the Pareto shape parameter is larger than 0.7), have their weights discarded—weights are set to <code>NA_real_</code> .
incl_orig	When <code>TRUE</code> , include a row for the original model specification, with all weights equal. Can facilitate comparison and plotting later.

Details

This function does the bulk of the sensitivity analysis work. It operates by parsing the model code from the provided Stan object, extracting the parameters and their sampling statements. It then uses R metaprogramming/tidy evaluation tools to flexibly evaluate the log density for each draw and each sampling statement, under the original and alternative specifications. From these, the function computes the overall importance weight for each draw and performs Pareto-smoothed importance sampling. All of the work is performed in R, without recompiling or refitting the Stan model.

Value

A tibble, produced by converting the provided specs to a tibble (see `as.data.frame.adjustr_spec`), and adding columns `.weights`, containing vectors of weights for each draw, and `.pareto_k`, containing the diagnostic Pareto shape parameters. Values greater than 0.7 indicate that importance

sampling is not reliable. If `incl_orig` is TRUE, a row is added for the original model specification. Weights can be extracted with the `pull.adjustr_weighted` method. The returned object also includes the model sample draws, in the `draws` attribute.

References

Vehtari, A., Simpson, D., Gelman, A., Yao, Y., & Gabry, J. (2015). Pareto smoothed importance sampling. [arXiv preprint arXiv:1507.02646](https://arxiv.org/abs/1507.02646).

See Also

[make_spec](#), [summarize.adjustr_weighted](#), [spec_plot](#)

Examples

```
## Not run:
model_data = list(
  J = 8,
  y = c(28, 8, -3, 7, -1, 1, 18, 12),
  sigma = c(15, 10, 16, 11, 9, 11, 10, 18)
)

spec = make_spec(eta ~ student_t(df, 0, 1), df=1:10)
adjust_weights(spec, eightschools_m)
adjust_weights(spec, eightschools_m, keep_bad=TRUE)

spec = make_spec(y ~ student_t(df, theta, sigma), df=1:10)
adjust_weights(spec, eightschools_m, data=model_data)
# will throw an error because `y` and `sigma` aren't provided
adjust_weights(spec, eightschools_m)

## End(Not run)
```

as.data.frame.adjustr_spec

Convert an adjustr_spec Object Into a Data Frame

Description

Returns the data frame of specification parameters, with added columns of the form `.samp_1`, `.samp_2`, ... for each sampling statement (or just `.samp` if there is only one sampling statement).

Usage

```
## S3 method for class 'adjustr_spec'
as.data.frame(x, ...)
```

Arguments

x the adjustr_spec object
 ... additional arguments to underlying method

dplyr.adjustr_spec dplyr *Methods for adjustr_spec Objects*

Description

Core `dplyr` verbs which don't involve grouping (`filter`, `arrange`, `mutate`, `select`, `rename`, and `slice`) are implemented and operate on the underlying table of specification parameters.

Usage

```
## S3 method for class 'adjustr_spec'
filter(.data, ..., .preserve = FALSE)

## S3 method for class 'adjustr_spec'
arrange(.data, ...)

## S3 method for class 'adjustr_spec'
rename(.data, ...)

## S3 method for class 'adjustr_spec'
select(.data, ...)

## S3 method for class 'adjustr_spec'
slice(.data, ..., .preserve = FALSE)
```

Arguments

.data the adjustr_spec object
 ... additional arguments to underlying method
 .preserve as in `filter` and `slice`

Examples

```
## Not run:
spec = make_spec(eta ~ student_t(df, 0, 1), df=1:10)

arrange(spec, desc(df))
slice(spec, 4:7)
filter(spec, df == 2)

## End(Not run)
```

extract_samp_stmts *Extract Model Sampling Statements From a Stan Model.*

Description

Prints a list of sampling statements extracted from the model block of a Stan program, with each labelled "parameter" or "data" depending on the type of variable being sampled.

Usage

```
extract_samp_stmts(object)
```

Arguments

object A `stanfit` model object.

Value

Invisibly returns a list of sampling formulas.

Examples

```
## Not run:
extract_samp_stmts(eightschools_m)
#> Sampling statements for model 2c8d1d8a30137533422c438f23b83428:
#> parameter eta ~ std_normal()
#> data      y ~ normal(theta, sigma)

## End(Not run)
```

get_resampling_idx *Get Importance Resampling Indices From Weights*

Description

Takes a vector of weights, or data frame or list containing sets of weights, and resamples indices for use in later computation.

Usage

```
get_resampling_idx(x, frac = 1, replace = T)
```

Arguments

x	A vector of weights, a list of weight vectors, or a data frame of type <code>adjustr_weighted</code> containing a <code>.weights</code> list-column of weights.
frac	A real number giving the fraction of draws to resample; the default, 1, resamples all draws. Smaller values should be used when <code>replace=FALSE</code> .
replace	Whether sampling should be with replacement. When weights are extreme it may make sense to use <code>replace=FALSE</code> , but accuracy is not guaranteed in these cases.

Value

A vector, list, or data frame, depending of the type of `x`, containing the sampled indices. If any weights are NA, the indices will also be NA.

Examples

```
## Not run:
spec = make_spec(eta ~ student_t(df, 0, 1), df=1:10)
adjusted = adjust_weights(spec, eightschools_m)

get_resampling_idx(adjusted)
get_resampling_idx(adjusted, frac=0.1, replace=FALSE)

## End(Not run)
```

make_spec

Set Up Model Adjustment Specifications

Description

Takes a set of new sampling statements, which can be parametrized by other arguments, data frames, or lists, and creates an `adjustr_spec` object suitable for use in `adjust_weights`.

Usage

```
make_spec(...)
```

Arguments

... Model specification. Each argument can either be a formula, a named vector, data frames, or lists.

Formula arguments provide new sampling statements to replace their counterparts in the original Stan model. All such formulas must be of the form `variable ~ distribution(parameters)`, where `variable` and `parameters` are Stan data variables or parameters, or are provided by other arguments to this function (see below), and where `distribution` matches one of the univariate

Stan distributions. Arithmetic expressions of parameters are also allowed, but care must be taken with multivariate parameter arguments. Since specifications are passed as formulas, R's arithmetic operators are used, not Stan's. As a result, matrix and elementwise multiplication in Stan sampling statements may not be interpreted correctly. Moving these computations out of sampling statements and into a local variables will ensure correct results.

For named vector arguments, each entry of the vector will be substituted into the corresponding parameter in the sampling statements. For data frames, each entry in each column will be substituted into the corresponding parameter in the sampling statements.

List arguments are coerced to data frames. They can either be lists of named vectors, or lists of lists of single-element named vectors.

The lengths of all parameter arguments must be consistent. Named vectors can have length 1 or must have length equal to the number of rows in all data frame arguments and the length of list arguments.

Value

An object of class `adjustr_spec`, which is essentially a list with two elements: `samp`, which is a list of sampling formulas, and `params`, which is a list of lists of parameters. Core [dplyr verbs](#) which don't involve grouping ([filter](#), [arrange](#), [mutate](#), [select](#), [rename](#), and [slice](#)) are supported and operate on the underlying table of specification parameters.

See Also

[adjust_weights](#), [summarize.adjustr_weighted](#), [spec_plot](#)

Examples

```
make_spec(eta ~ cauchy(0, 1))

make_spec(eta ~ student_t(df, 0, 1), df=1:10)

params = tidyr::crossing(df=1:10, infl=c(1, 1.5, 2))
make_spec(eta ~ student_t(df, 0, 1),
          y ~ normal(theta, infl*sigma),
          params)
```

pull.adjustr_weighted *Extract Weights From an adjustr_weighted Object*

Description

This function modifies the default behavior of `dplyr::pull` to extract the `.weights` column.

Usage

```
## S3 method for class 'adjustr_weighted'
pull(.data, var = ".weights", name = NULL, ...)
```

Arguments

.data	A table of data
var	A variable, as in pull . The default returns the .weights column, and if there is only one row, it returns the first element of that column
name	Ignored
...	Ignored

spec_plot	<i>Plot Posterior Quantities of Interest Under Alternative Model Specifications</i>
-----------	---

Description

Uses weights computed in [adjust_weights](#) to plot posterior quantities of interest versus specification parameters

Usage

```
spec_plot(
  x,
  by,
  post,
  only_mean = FALSE,
  ci_level = 0.8,
  outer_level = 0.95,
  ...
)
```

Arguments

x	An <code>adjustr_weighted</code> object.
by	The x-axis variable, which is usually one of the specification parameters. Can be set to 1 if there is only one specification. Automatically quoted and evaluated in the context of x.
post	The posterior quantity of interest, to be computed for each resampled draw of each specification. Should evaluate to a single number for each draw. Automatically quoted and evaluated in the context of x.
only_mean	Whether to only plot the posterior mean. May be more stable.
ci_level	The inner credible interval to plot. Central $100 \times \text{ci_level}$ posterior draws.
outer_level	The outer credible interval to plot.
...	Ignored.

Value

A `ggplot` object which can be further customized with the **ggplot2** package.

See Also

[adjust_weights](#), [summarize.adjustr_weighted](#)

Examples

```
## Not run:
spec = make_spec(eta ~ student_t(df, 0, scale),
                 df=1:10, scale=seq(2, 1, -1/9))
adjusted = adjust_weights(spec, eightschools_m)

spec_plot(adjusted, df, theta[1])
spec_plot(adjusted, df, mu, only_mean=TRUE)
spec_plot(adjusted, scale, tau)

## End(Not run)
```

summarise.adjustr_weighted

Summarize Posterior Distributions Under Alternative Model Specifications

Description

Uses weights computed in [adjust_weights](#) to compute posterior summary statistics. These statistics can be compared against their reference values to quantify the sensitivity of the model to aspects of its specification.

Usage

```
## S3 method for class 'adjustr_weighted'
summarise(.data, ..., .resampling = FALSE, .model_data = NULL)

## S3 method for class 'adjustr_weighted'
summarize(.data, ..., .resampling = FALSE, .model_data = NULL)
```

Arguments

`.data` An `adjustr_weighted` object.

`...` Name-value pairs of expressions. The name of each argument will be the name of a new variable, and the value will be computed for the posterior distribution of eight alternative specification. For example, a value of `mean(theta)` will compute the posterior mean of `theta` for each alternative specification.

Also supported is the custom function `wasserstein`, which computes the Wasserstein- p distance between the posterior distribution of the provided expression under the new model and under the original model, with $p=1$ the default. Lower the spacing parameter from the default of 0.005 to compute a finer (but slower) approximation.

The arguments in `...` are automatically quoted and evaluated in the context of `.data`. They support unquoting and splicing.

- `.resampling` Whether to compute summary statistics by first resampling the data according to the weights. Defaults to `FALSE`, but will be used for any summary statistic that is not `mean`, `var` or `sd`.
- `.model_data` Stan model data, if not provided in the earlier call to `adjust_weights`.

Value

An `adjustr_weighted` object, with the new columns specified in `...` added.

See Also

[adjust_weights](#), [spec_plot](#)

Examples

```
## Not run:
model_data = list(
  J = 8,
  y = c(28, 8, -3, 7, -1, 1, 18, 12),
  sigma = c(15, 10, 16, 11, 9, 11, 10, 18)
)

spec = make_spec(eta ~ student_t(df, 0, 1), df=1:10)
adjusted = adjust_weights(spec, eightschools_m)

summarize(adjusted, mean(mu), var(mu))
summarize(adjusted, wasserstein(mu, p=2))
summarize(adjusted, diff_1 = mean(y[1] - theta[1]), .model_data=model_data)
summarize(adjusted, quantile(tau, probs=c(0.05, 0.5, 0.95)))

## End(Not run)
```

Index

`adjust_weights`, 2, 3, 7–11
`adjustr`, 2
`arrange`, 5, 8
`arrange.adjustr_spec`
 (`dplyr.adjustr_spec`), 5
`as.data.frame.adjustr_spec`, 3, 4

`brmsfit`, 3

`CmdStanMCMC`, 3
`CmdStanModel`, 3

`dplyr`, 5
`dplyr verbs`, 8
`dplyr.adjustr_spec`, 5

`extract_samp_stmts`, 6

`filter`, 5, 8
`filter.adjustr_spec`
 (`dplyr.adjustr_spec`), 5

`get_resampling_idx`s, 6
`ggplot`, 10

`make_spec`, 2–4, 7
`mutate`, 5, 8

`plot`, 2
`pull`, 9
`pull.adjustr_weighted`, 4, 8

`rename`, 5, 8
`rename.adjustr_spec`
 (`dplyr.adjustr_spec`), 5

`select`, 5, 8
`select.adjustr_spec`
 (`dplyr.adjustr_spec`), 5
`slice`, 5, 8
`slice.adjustr_spec`
 (`dplyr.adjustr_spec`), 5

`spec_plot`, 4, 8, 9, 11
`stanfit`, 3, 6
`stanreg`, 3
`summarise.adjustr_weighted`, 10
`summarize`, 2
`summarize.adjustr_weighted`, 4, 8, 10
`summarize.adjustr_weighted`
 (`summarise.adjustr_weighted`),
 10